Pure Aloha efficiency

P(success by given node) = P(node transmits) ·

P(no other node transmits in $[t_0-1,t_0]$ ·

P(no other node transmits in $[t_0,t_0+1]$

$= p \cdot (1-p)^{N-1} \cdot (1-p)^{N-1}$

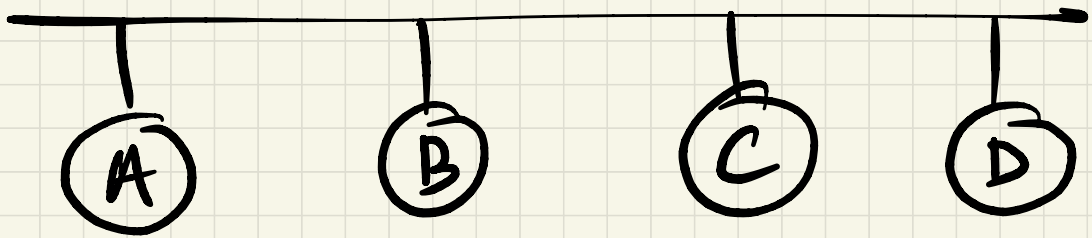
$= p \cdot (1-p)^{2(N-1)}$

… choosing optimum p and then letting n -> infty …

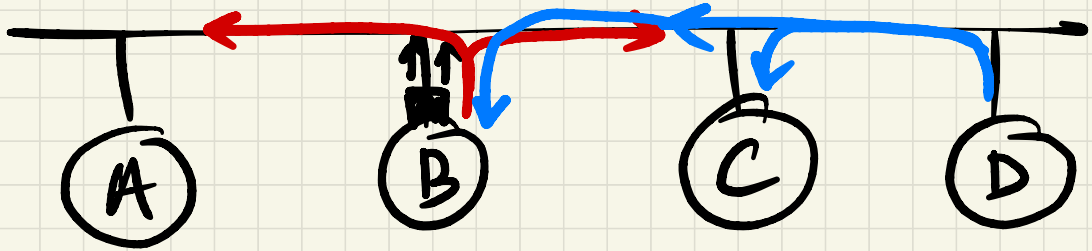
$= 1/(2e) = .18$

Even worse !

Carrier Sense Multiple Access (CSMA)



Carrier Sense Multiple Access (CSMA)



Listen before you talk

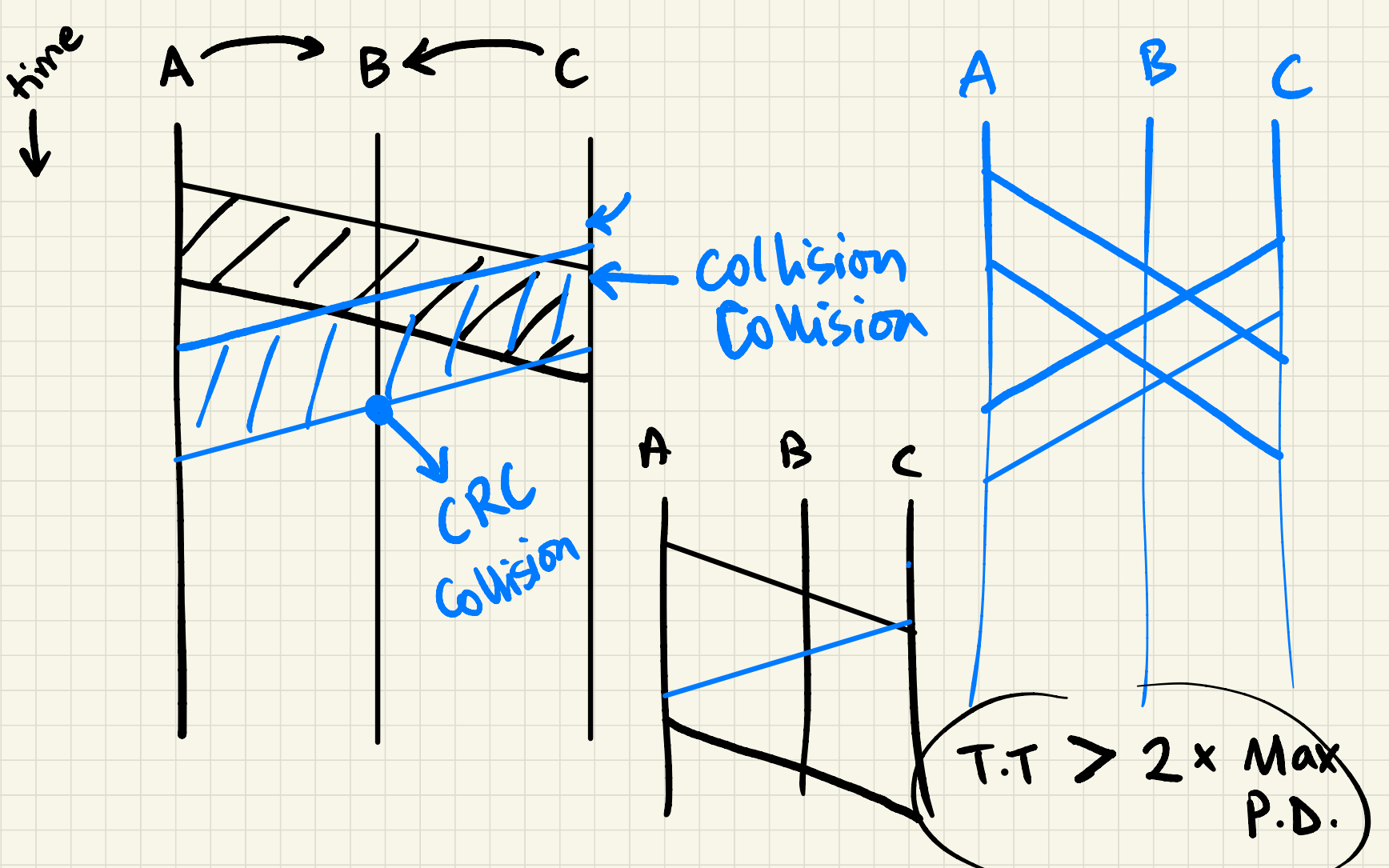
When B is transmitting, B can check if:

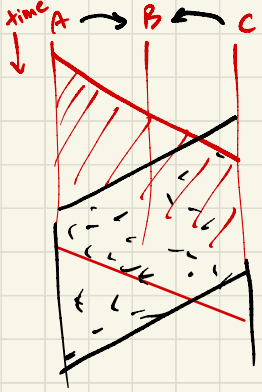
transmitted sig == received sig

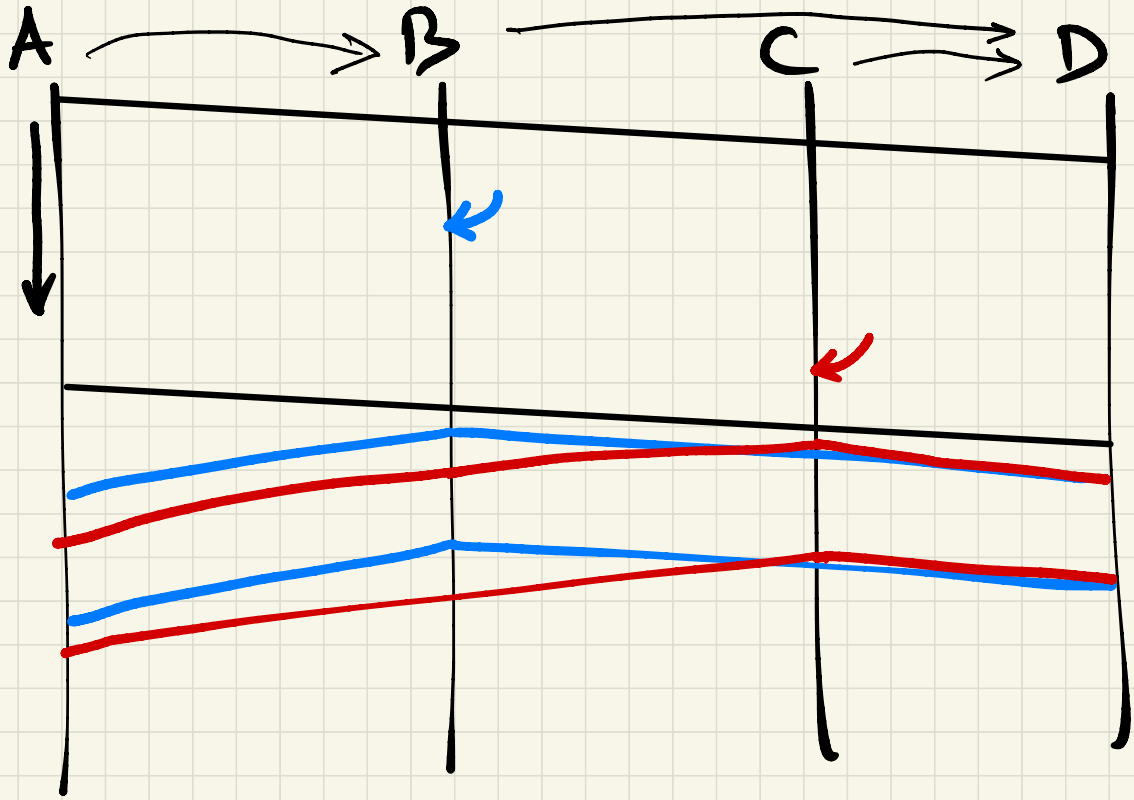
(red) (red + blue)

(Collision Detection)

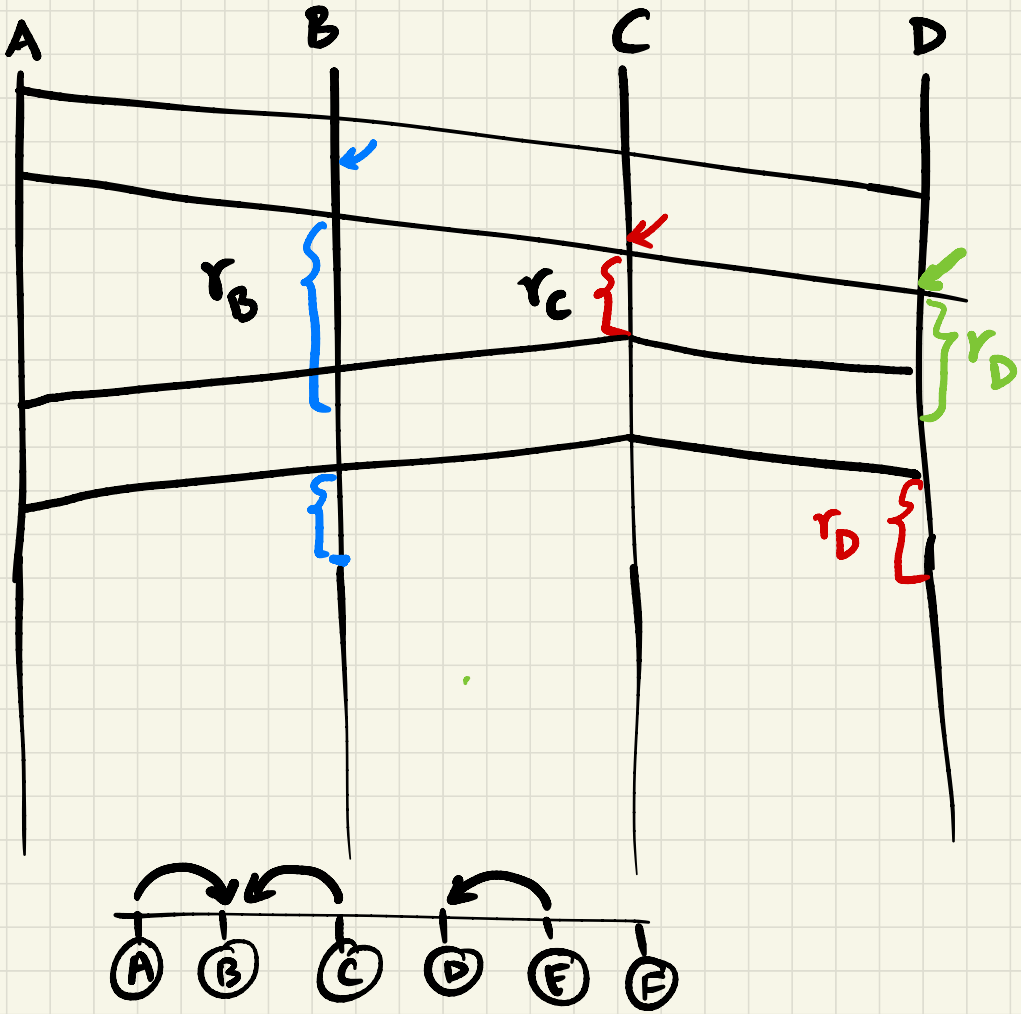
CSMA/CD


time
A
B
C
collision
collision
CRC
collision
A
B
C
A
B
C
T.T > 2 x Max
P.D.



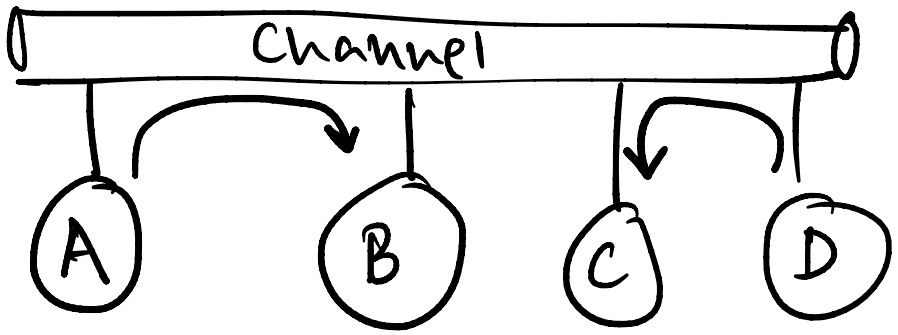


wait a rand amount of time once channel becomes "idle"


A
B
C
D
r_B
r_C
r_D
r_D
A
B
C
D
E
F

Ethernet uses CSMA/CD

- No slots
- adapter doesn't transmit if it senses that some other adapter is transmitting, that is, carrier sense
- transmitting adapter aborts when it senses that another adapter is transmitting, that is, collision detection
- Before attempting a retransmission, adapter waits a random time, that is, random access



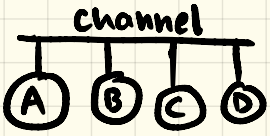
Ethernet CSMA/CD algorithm

1. Adaptor receives datagram from net layer & creates frame
2. If adapter senses channel idle, it starts to transmit frame. If it senses channel busy, waits until channel idle and then transmits
3. If adapter transmits entire frame without detecting another transmission, the adapter is done with frame !
4. If adapter detects another transmission while transmitting, aborts and sends jam signal
5. After aborting, adapter enters **exponential backoff**: after the mth collision, adapter chooses a K at random from $\{0,1,2,\ldots,2^m-1\}$. Adapter waits K·512 bit times and returns to Step 2

So current protocol:

Contention Window:

$CW_{min} = 15$ slots



I have a pkt to send

Listen to channel

Is channel idle

Y

N

Pick Random No. R

$[0, CW_{min}]$

Count down R to zero

CW

Contention Window

Keep listening while counting down

Is Channel Busy?

Is countdown = zero?

Freeze Countdown

Transmit and keep listening

Is Tx done?

Is Foreign Signal?

collision

Abort Tx

send jam

Prepare retransmission

Wait for ACK

In Ethernet, no ACK, so go back to "I have a pkt".

Reset Random # range to min. value

$= 15$

CW_{min}

Pick new Random # R+

$R \in [0, 31]$

Increase Random # range

$CW = 31$

(→) Some key points.

(1) Collision happens always at the Receiver. Transmitter may detect collision by observing a foreign signal, but that doesn't always mean collision is at T_x.

(2) Channel is wasted because of random count down ⇒ called BACKOFF. This is the price to be paid for distributed coordination.

(3) The above protocol assumes that a T_x can transmit and listen at the same time. Possible in wired networks like Ethernet. Harder in wireless networks.

(4) T_x detects foreign signal and can tell for sure that collision is happening at R_x. This assumes channel is identical at T_x and R_x. True for wired networks, not for wireless.

MAC protocols

- TDMA / FDMA
- Randomized protocols
- Taking turns.

"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

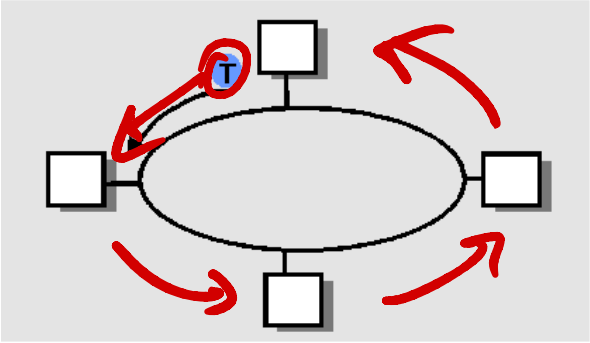
Bluetooth

Polling:

- master node "invites" slave nodes to transmit in turn
- concerns:
  - polling overhead
  - latency
  - single point of failure (master)

Token passing:

- control **token** passed from one node to next sequentially.
- token message
- concerns:
  - token overhead
  - latency
  - single point of failure (token)



Summary of MAC protocols

- What do you do with a shared media?
  - Channel Partitioning, by time, frequency or code
    - Time Division, Frequency Division
  - Random partitioning (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 a central site, token passing

Link Layer

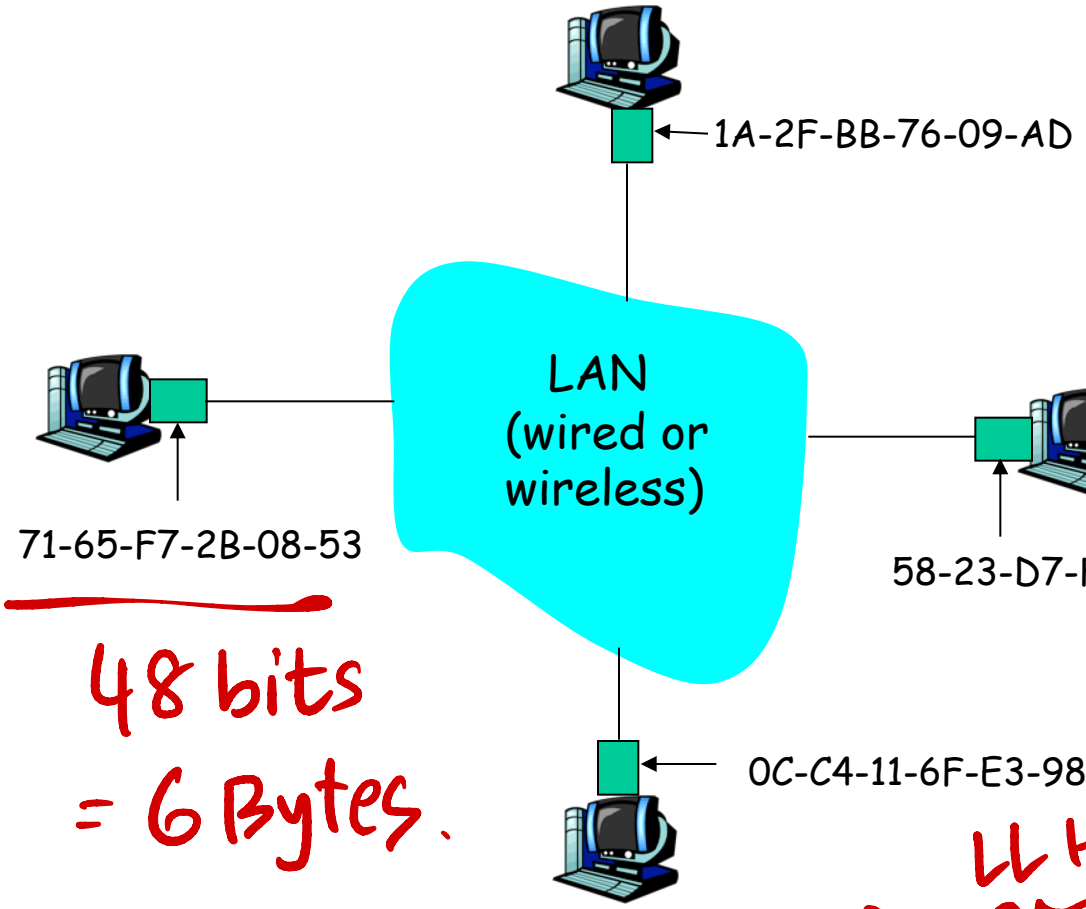
- 5.1 Introduction and services
- 5.2 Error detection and correction
- 5.3Multiple access protocols
- 5.4 Link-Layer Addressing
- 5.5 Ethernet
- 5.6 Hubs and switches

MAC Addresses and ARP

- 32-bit IP address:
  - *network-layer* address
  - used to get datagram to destination IP subnet
- MAC (or LAN or physical or Ethernet) address:
  - used to get frame from one interface to another physically-connected interface (same network)
  - 48 bit MAC address (for most LANs) burned in the adapter ROM

LAN Addresses and ARP

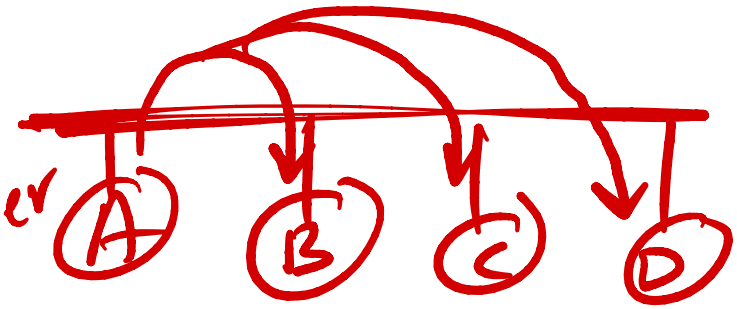
Each adapter on LAN has unique LAN address = MAC address



Broadcast address =
FF-FF-FF-FF-FF-FF

= adapter

48 bits
= 6 Bytes.

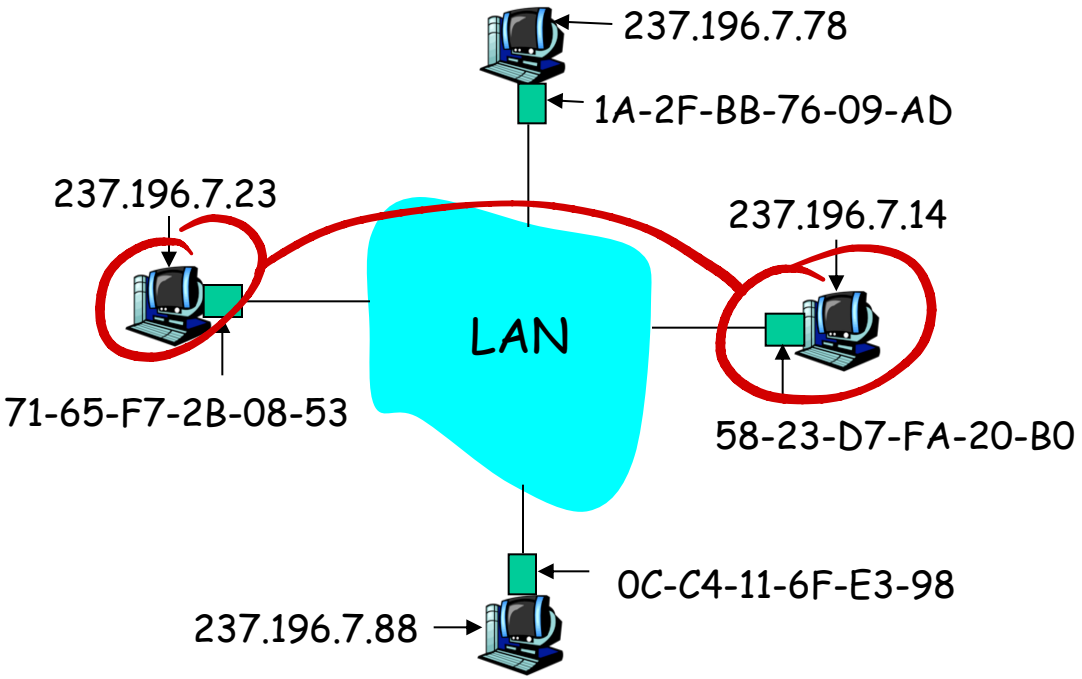


LAN Address (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)
- Analogy:
  - (a) MAC address: like Social Security Number
  - (b) IP address: like postal address
- MAC flat address ➜ portability
  - can move LAN card from one LAN to another
- IP hierarchical address NOT portable
  - depends on IP subnet to which node is attached

ARP: Address Resolution Protocol

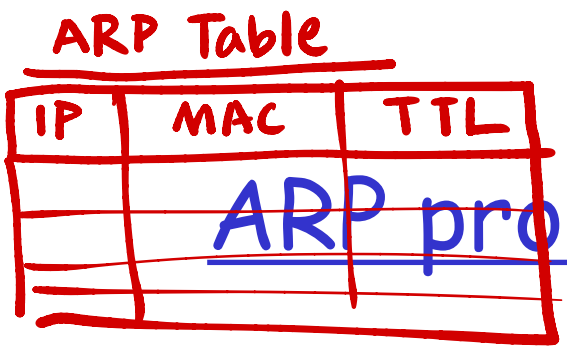
Question: how to determine MAC address of B knowing B's IP address?



- Each IP node (Host, Router) on LAN has ARP table
- ARP Table: IP/MAC address mappings for some LAN nodes

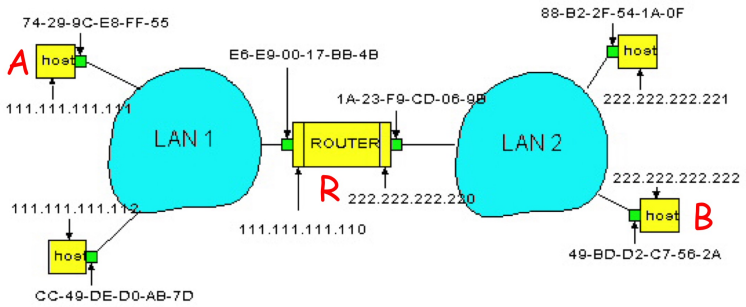
  < IP address; MAC address; TTL>

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



ARP protocol: Same LAN (network)

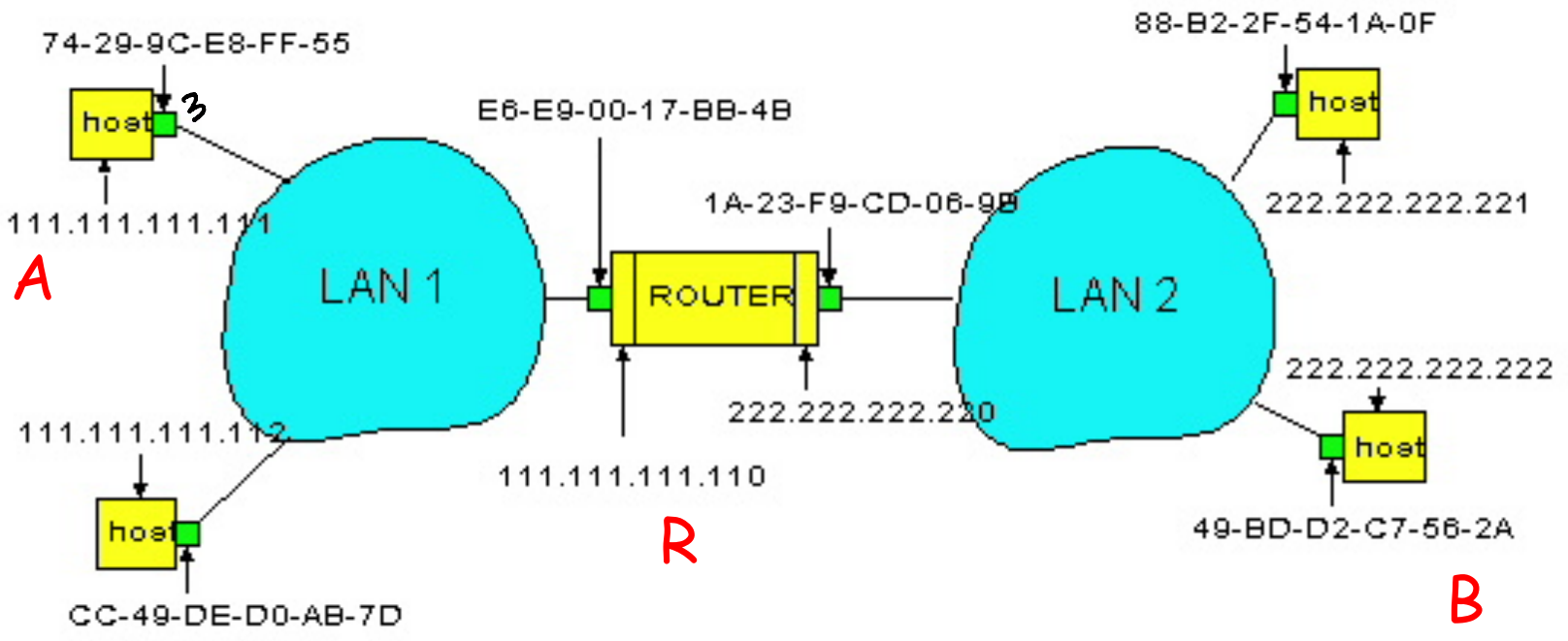
- A wants to send datagram to B, and B's MAC address not in A's ARP table.
- A broadcasts ARP query packet, containing B's IP address
  - Dest MAC address = FF-FF-FF-FF-FF-FF
  - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) MAC address
  - frame sent to A's MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - soft state: information that times out (goes away) unless refreshed
- ARP is "plug-and-play":
  - nodes create their ARP tables without intervention from net administrator



Routing to another LAN

walkthrough: send datagram from A to B via R

assume A know's B IP address



- Two ARP tables in router R, one for each IP network (LAN)

- A creates datagram with source A, destination B
- A uses ARP to get R's MAC address for 111.111.111.110
- A creates link-layer frame with R's MAC address as dest, frame contains A-to-B IP datagram
- A's adapter sends frame
- R's adapter receives frame
- R removes IP datagram from Ethernet frame, sees its destined to B
- R uses ARP to get B's MAC address
- R creates frame containing A-to-B IP datagram sends to B