Data Link Layer: CSMA/CD, ARP

A/PROF. DUY NGO

CSMA (Carrier Sense Multiple Access)

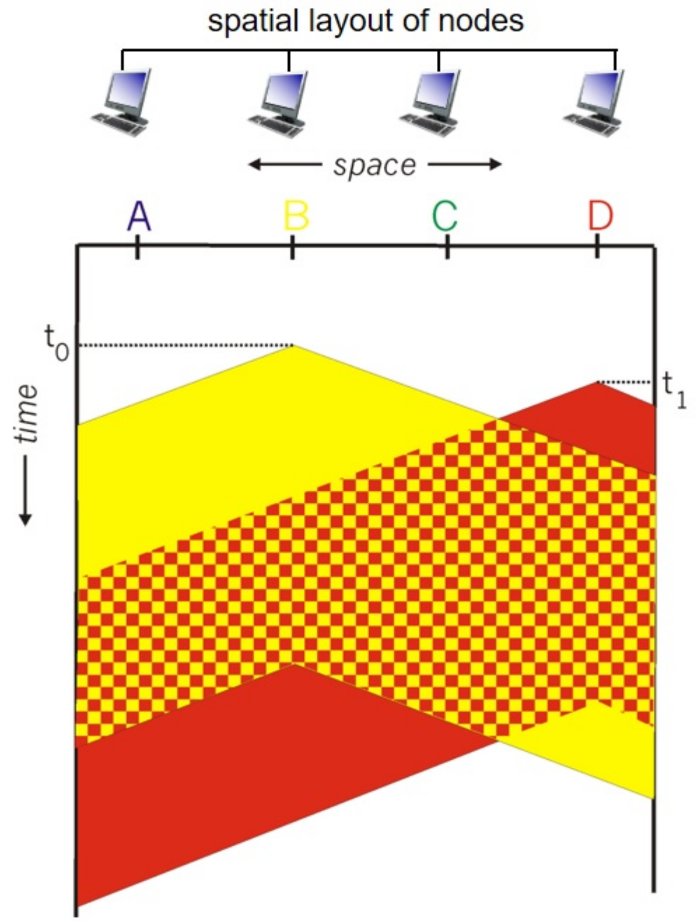
CSMA: listen before transmit:

- **if channel sensed idle:** transmit entire frame
- **if channel sensed busy,** defer transmission

human analogy: “don’t interrupt others!”

CSMA Collisions

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

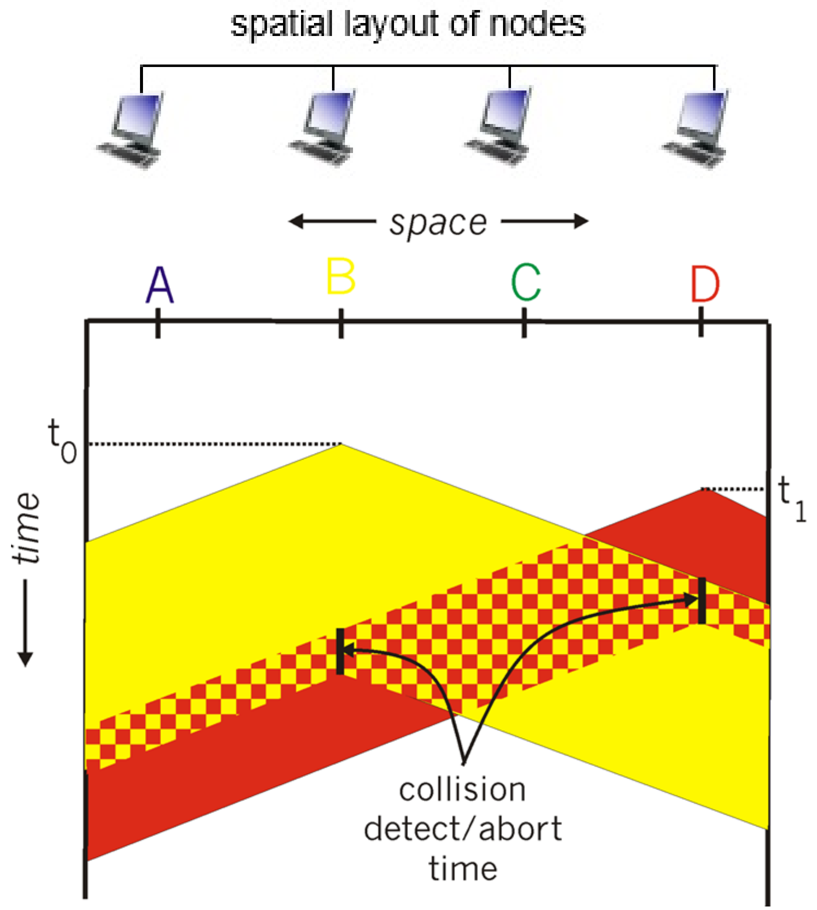


CSMA/CD (Collision Detection) (1 of 2)

CSMA/CD: carrier sensing, deferral as in CSMA
- collisions **detected** within short time
- colliding transmissions aborted, reducing channel wastage

- collision detection:
  - easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - difficult in wireless LANs: received signal strength overwhelmed by local transmission strength
- human analogy: the polite conversationalist

CSMA/CD (Collision Detection) (2 of 2)



Ethernet CSMA/CD Algorithm

1. NIC receives datagram from network layer, creates frame
2. If NIC senses channel idle, starts frame transmission. If NIC senses channel busy, waits until channel idle, then transmits.
3. If NIC transmits entire frame without detecting another transmission, NIC is done with frame!
4. If NIC detects another transmission while transmitting, aborts transmission
5. After aborting, NIC enters **binary (exponential) backoff:**

   – after *m*th collision, NIC chooses *K* at random from $\{0,1,2, \ldots, 2^m - 1\}$.

     NIC waits K·512 bit times, returns to Step 2

   – longer backoff interval with more collisions

CSMA/CD Efficiency

- t_{prop} = max prop delay between 2 nodes in LAN
- t_{trans} = time to transmit max-size frame

$$efficiency = \frac{1}{1 + 5t_{prop} / t_{trans}}$$

- efficiency goes to 1
  - as t_{prop} goes to 0
  - as t_{trans} goes to infinity
- better performance than ALOHA, and simple, cheap, decentralised!

"Taking Turns" MAC Protocols (1 of 3)

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

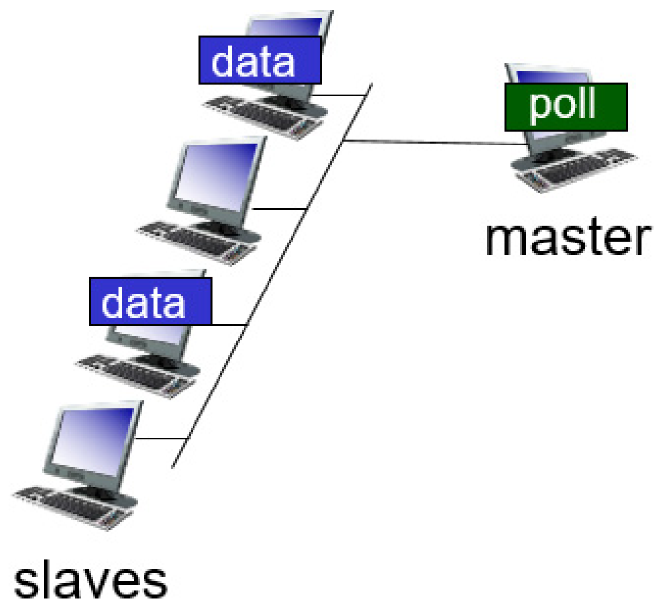
"taking turns" protocols

- look for best of both worlds!

“Taking Turns” MAC Protocols (2 of 3)

polling:

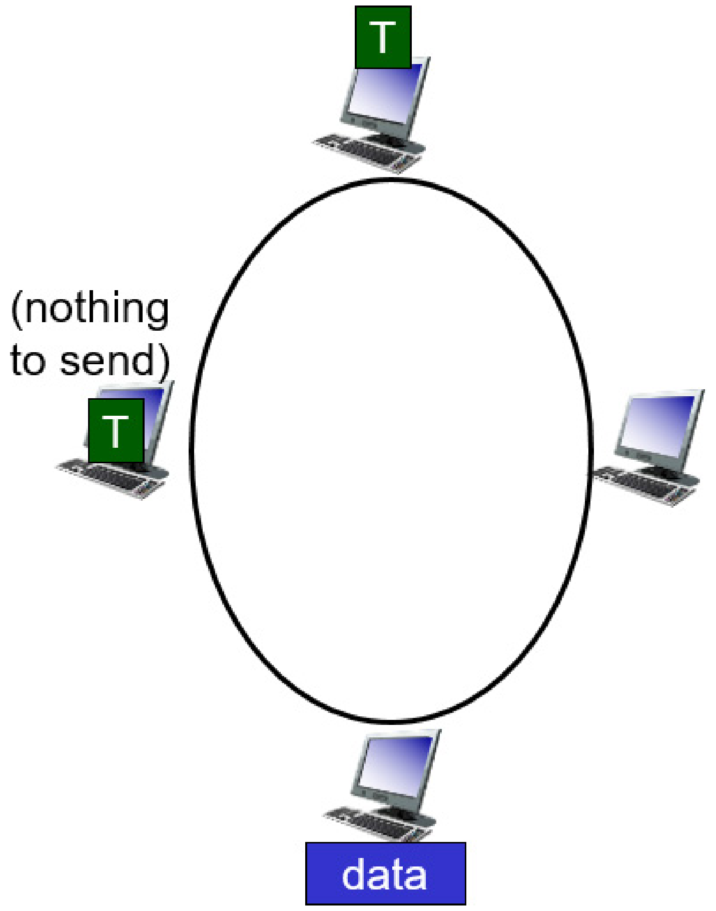
- master node “invites” slave nodes to transmit in turn
- typically used with “dumb” slave devices
- concerns:
  - polling overhead
  - latency
  - single point of failure (master)



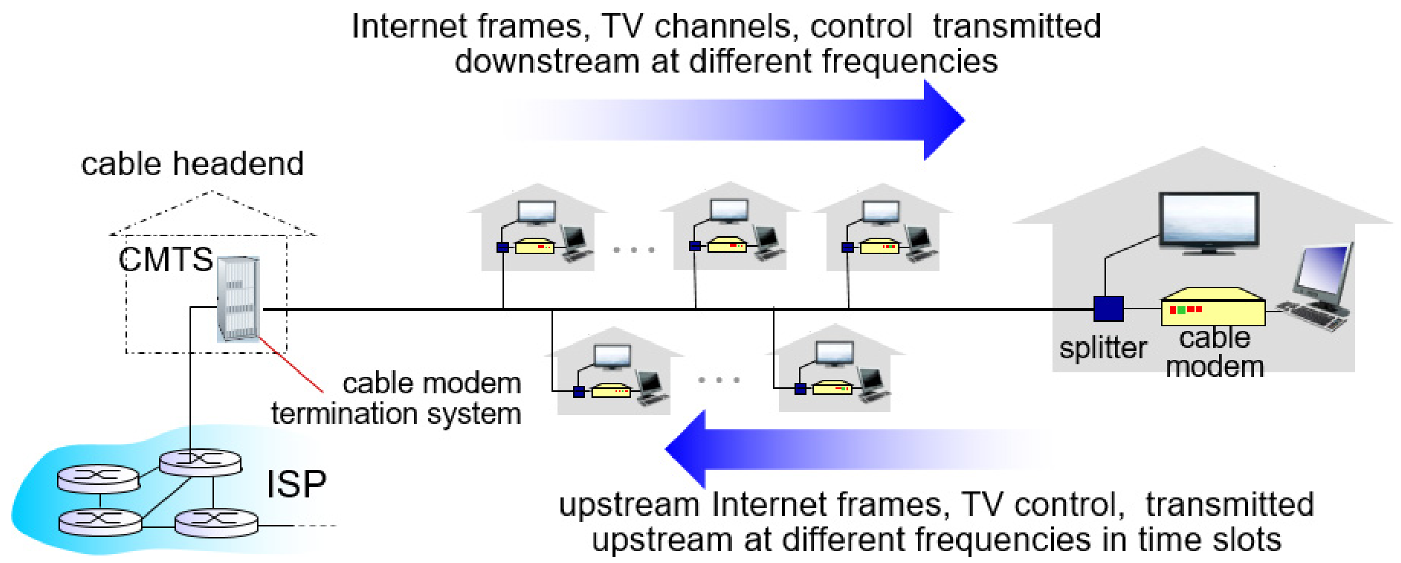
"Taking Turns" MAC Protocols (3 of 3)

token passing:

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

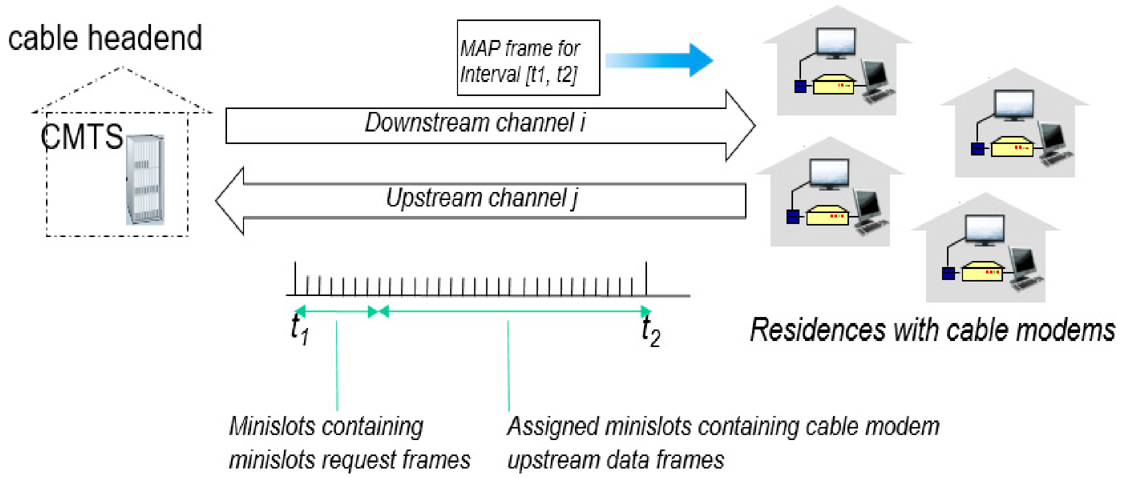


Cable Access Network (1 of 2)


Internet frames, TV channels, control transmitted downstream at different frequencies
cable headend
CMTS
cable modem termination system
ISP
splitter
cable modem
upstream Internet frames, TV control, transmitted upstream at different frequencies in time slots

- **multiple** 40 Mbps downstream (broadcast) channels
  - single CMTS transmits into channels
- **multiple** 30 Mbps upstream channels
  - **multiple access:** all users contend for certain upstream channel time slots (others assigned)

Cable Access Network (2 of 2)


cable headend
CMTS
MAP frame for Interval [t1, t2]
Downstream channel i
Upstream channel j
t_1
t_2
Residences with cable modems
Minislots containing minislots request frames
Assigned minislots containing cable modem upstream data frames

DOCSIS: data over cable service interface spec

- 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 (Carrier Sense Multiple Access/Collision Avoidance) used in 802.11
- **taking turns**
  - polling from central site, token passing
  - Bluetooth, FDDI, token ring

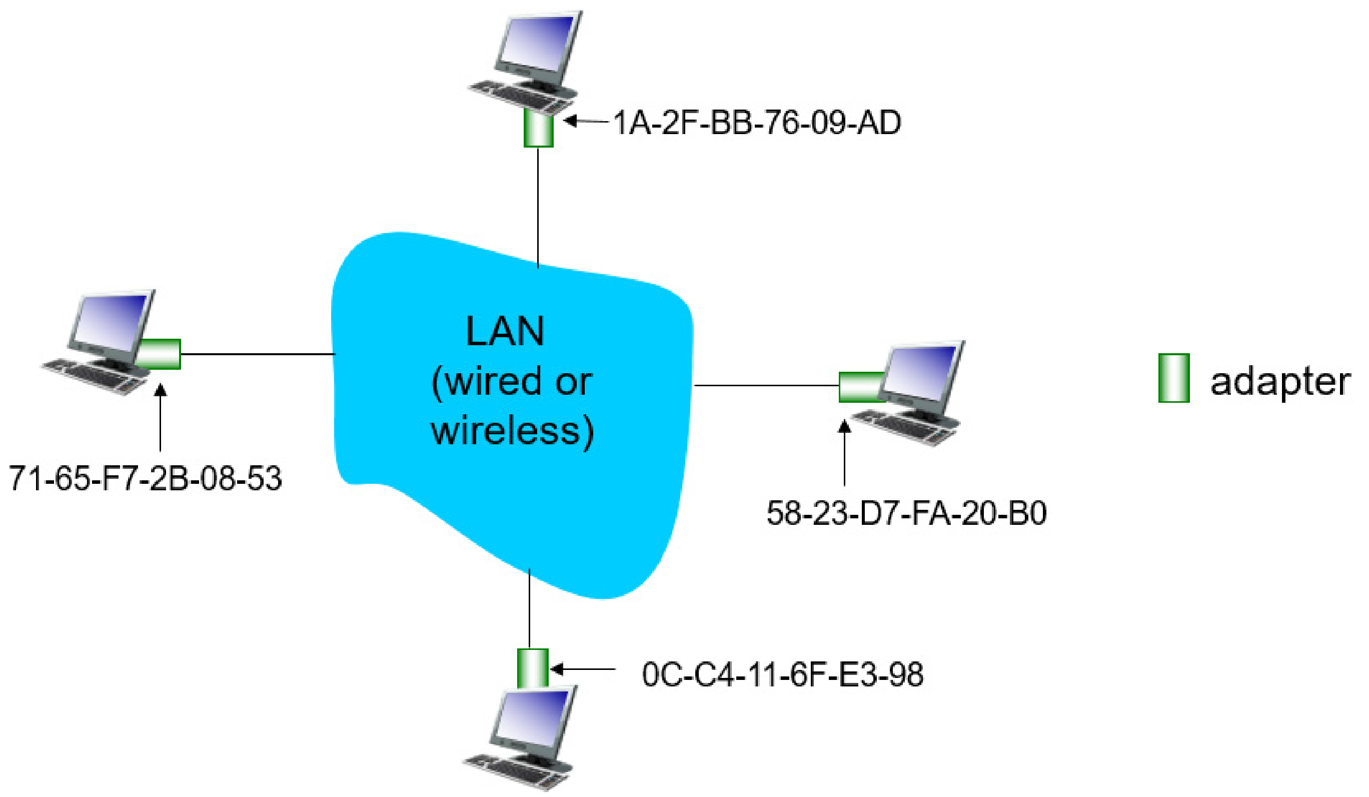
MAC Addresses and ARP

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

LAN Addresses and ARP

each adapter on LAN has unique **LAN** address

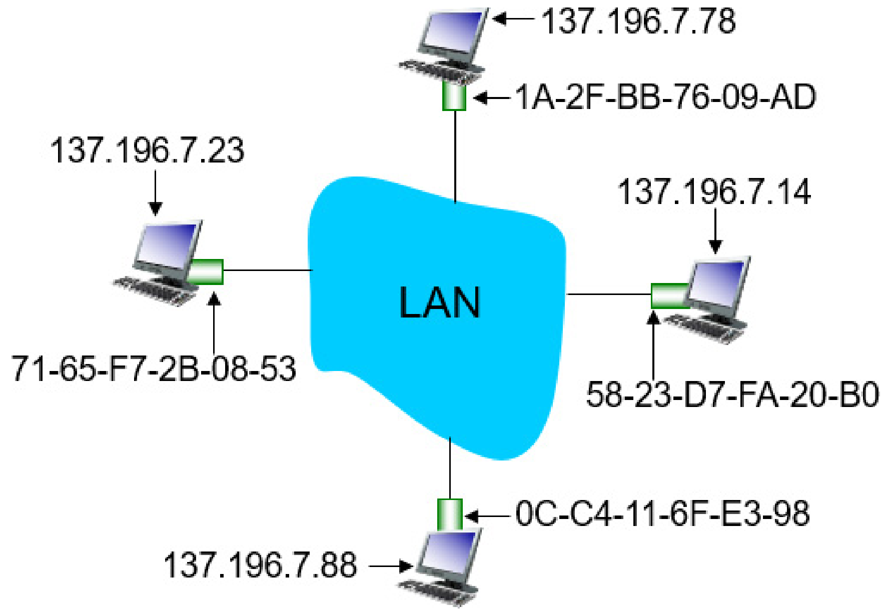


LAN Addresses

- 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 LAN card from one LAN to another
- IP hierarchical address **not** portable
  - address 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)

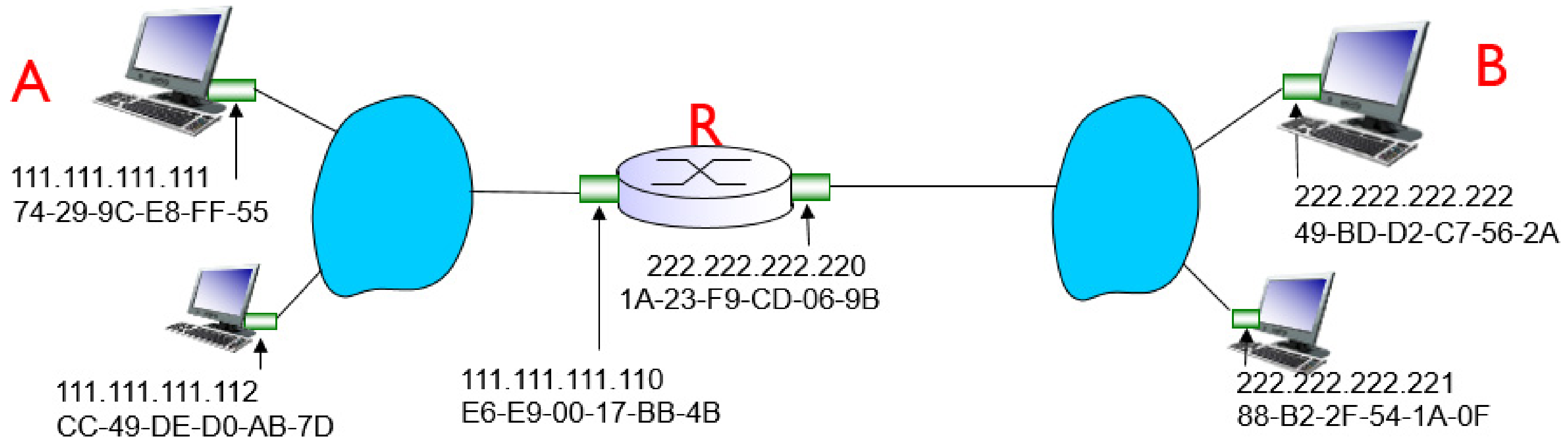
ARP Protocol: Same LAN

- A wants to send datagram to B
  - B's MAC address not in A's ARP table.
- A **broadcasts** ARP query packet, containing B's IP address
  - destination MAC address = FF-FF-FF-FF-FF-FF
  - all nodes 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**

Addressing: Routing to Another LAN (1 of 5)

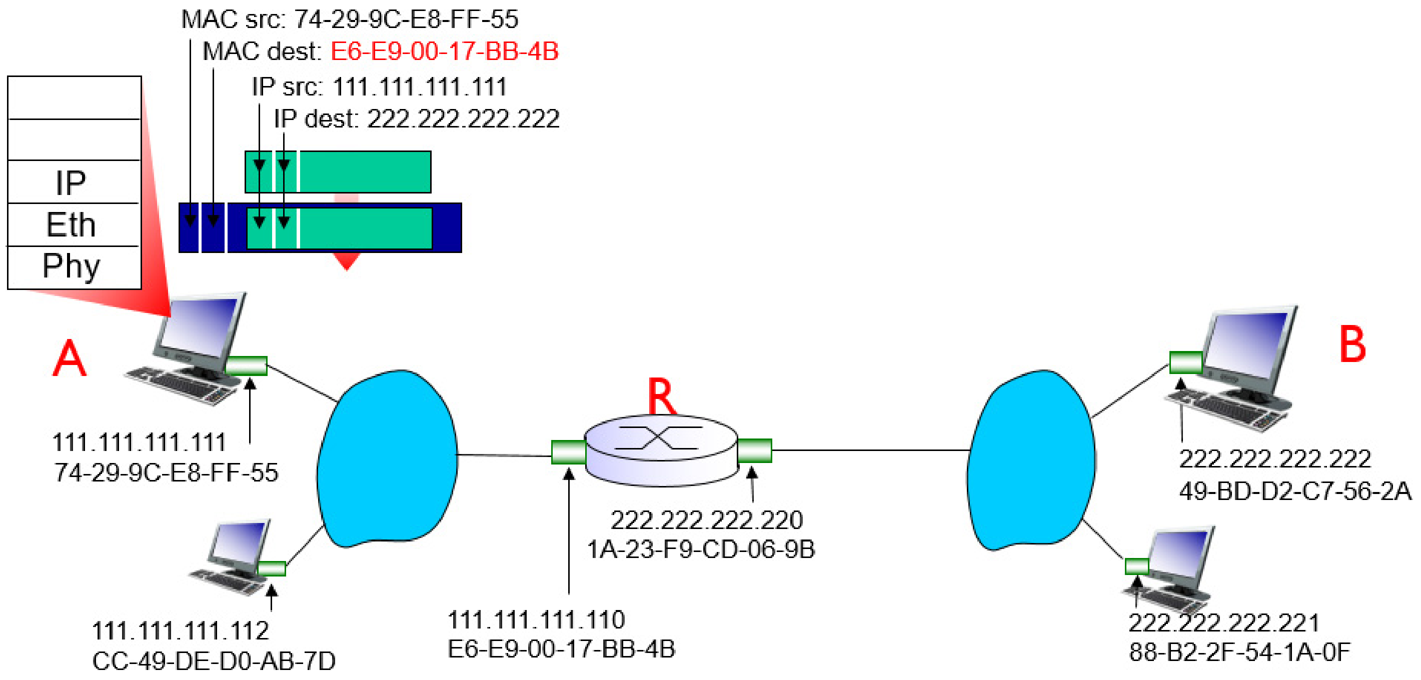
walkthrough: **send datagram from A to B via R**

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



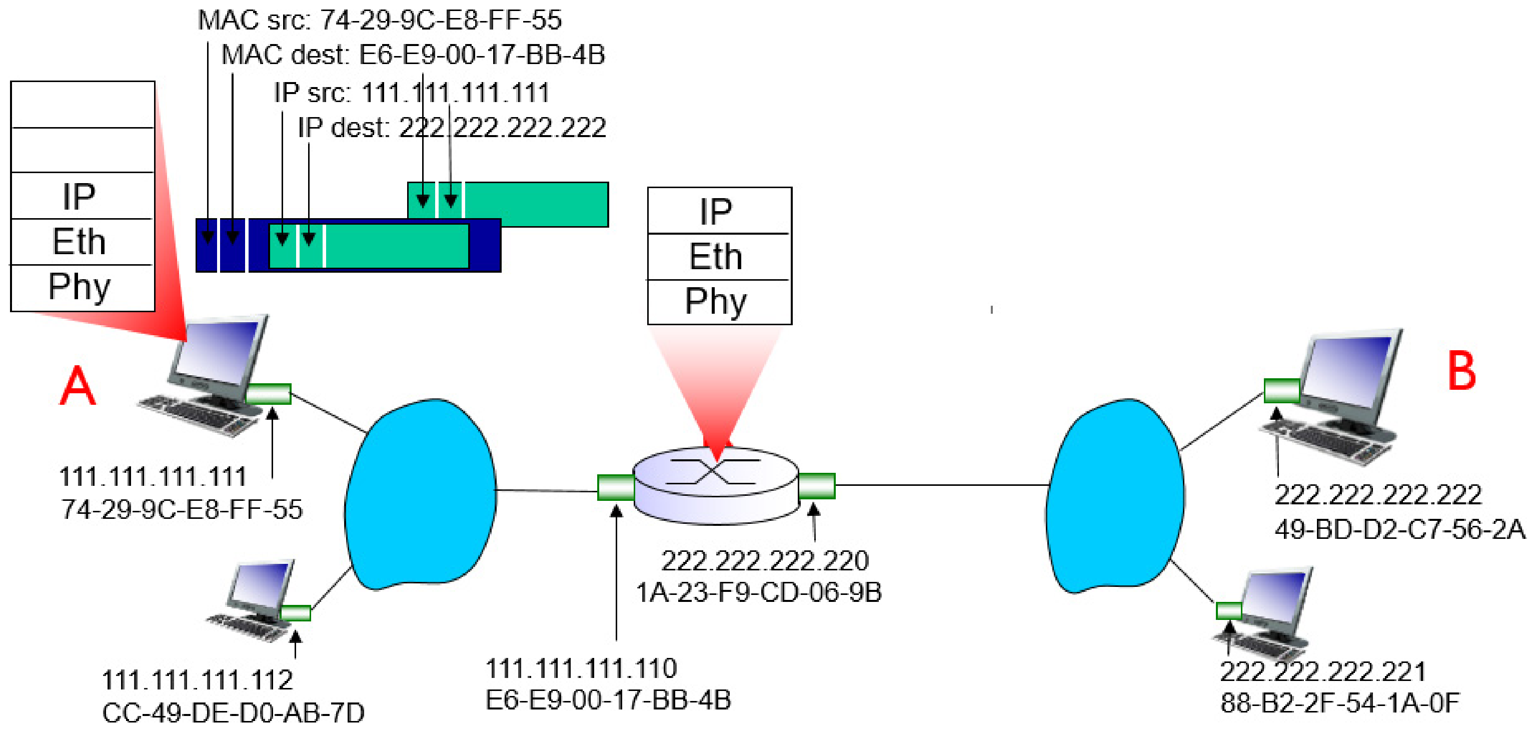
Addressing: Routing to Another LAN (2 of 5)

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



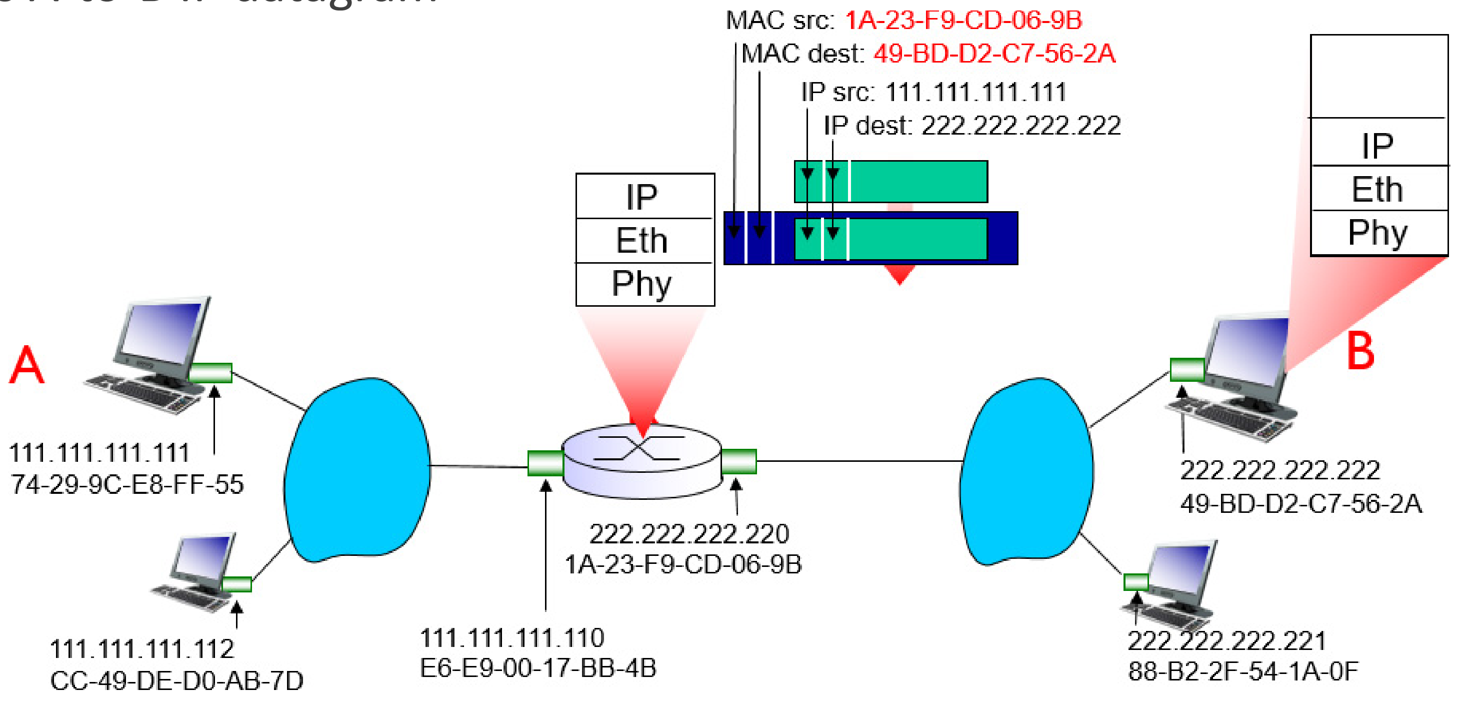
Addressing: Routing to Another LAN (3 of 5)

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



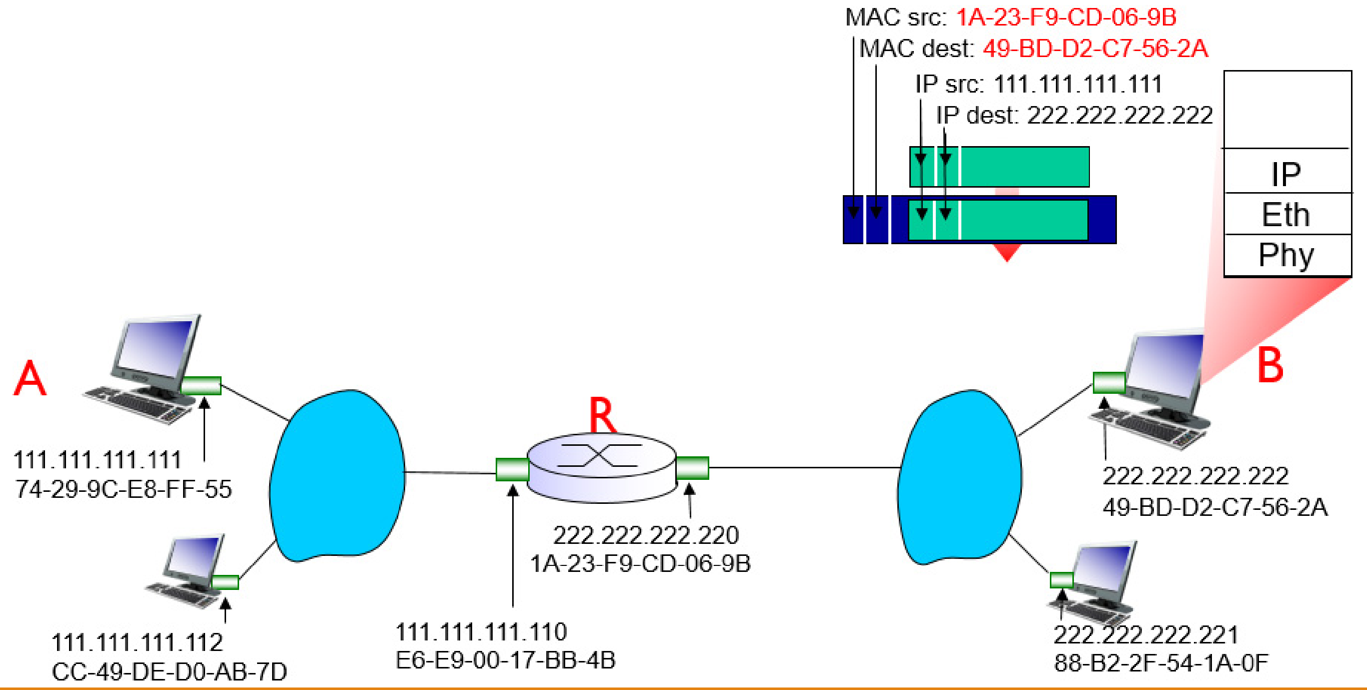
Addressing: Routing to Another LAN (4 of 5)

- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as destination address, frame contains A-to-B IP datagram



Addressing: Routing to Another LAN (5 of 5)

- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram



MAC src: 1A-23-F9-CD-06-9B
MAC dest: 49-BD-D2-C7-56-2A
IP src: 111.111.111.111
IP dest: 222.222.222.222
IP
Eth
Phy
A
111.111.111.111
74-29-9C-E8-FF-55
111.111.111.112
CC-49-DE-D0-AB-7D
R
111.111.111.110
E6-E9-00-17-BB-4B
222.222.222.220
1A-23-F9-CD-06-9B
B
222.222.222.222
49-BD-D2-C7-56-2A
222.222.222.221
88-B2-2F-54-1A-0F