

# Computer Networks

## COL 334/672

Network layer

Tarun Mangla

*Slides adapted from KR*

Sem 1, 2024-25

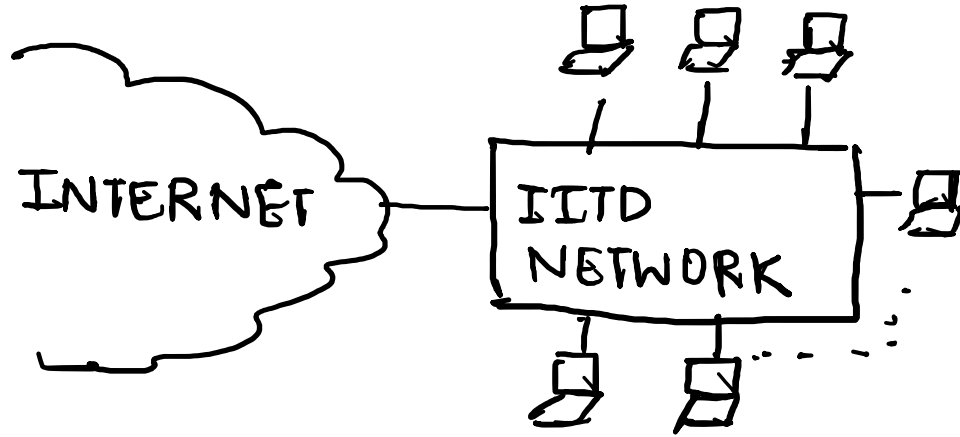
# Today's Class

Continue our discussion of the network layer

- How do you work with a limited set of IPv4 addresses?
- How does a host get IP address from the network?
- How does a host find the MAC address for an IP address in the subnet?

# IITD Network

→ ①. Having a private n/w - & a global IP is shared amongst multiple people

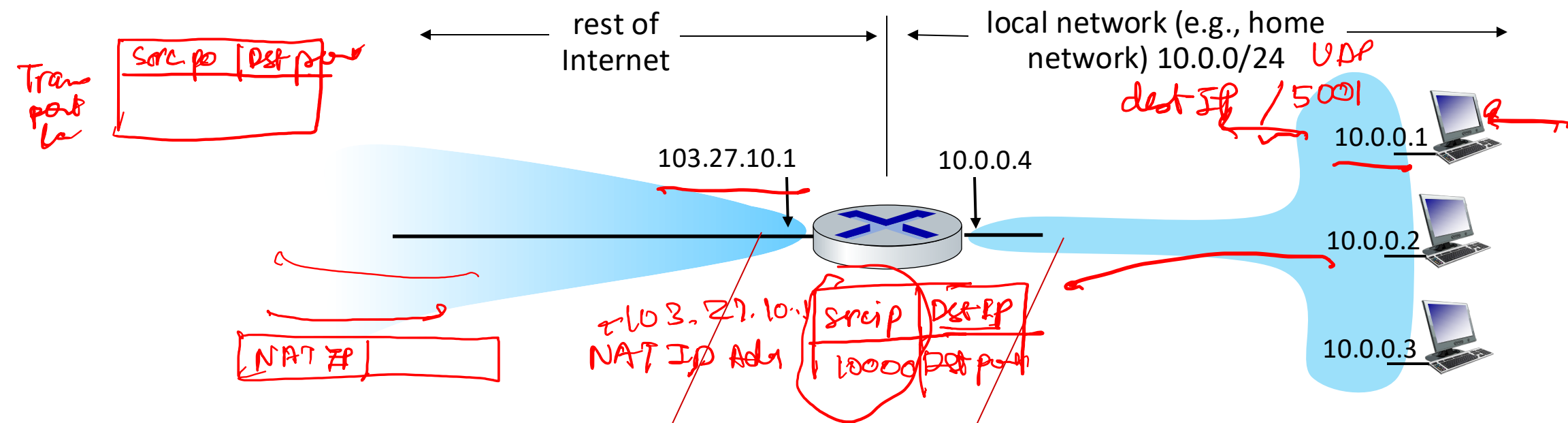


Only 1024 IPv4 addresses but  $\gg$  1024 hosts

How do you connect  $\gg$  1024 hosts, with 1024 addresses? to Internet

# NAT: Network Address Translation

**NAT:** all devices in local network share just **one** IPv4 address as far as outside world is concerned



*all* datagrams *leaving* local network have *same* source NAT IP address: 103.27.10.1,

datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

# NAT: network address translation

**implementation:** NAT router must (transparently):

- outgoing datagrams: replace (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
  - remote clients/servers will respond using (NAT IP address, new port #) as destination address
- remember (in NAT translation table) every (source IP address, port #) to (NAT IP address, new port #) translation pair
- incoming datagrams: replace (NAT IP address, new port #) in destination fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

① layering

new protocol ≠ TCP/UDP

STUN :

TURN : Relay

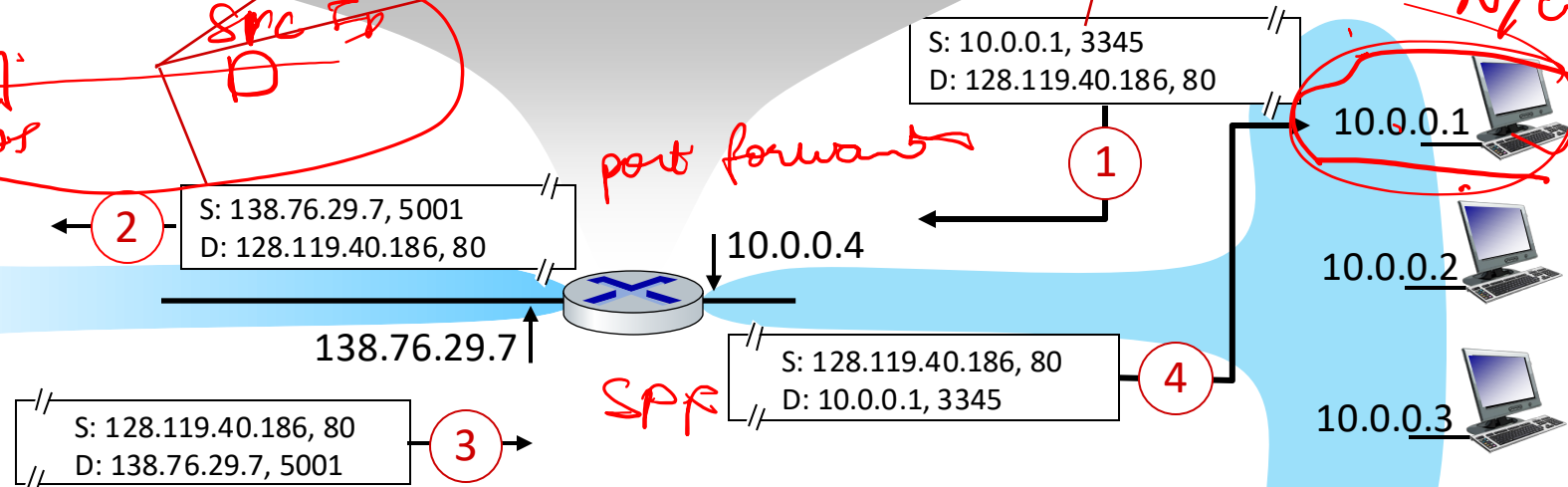
# NAT: network address translation

Public IP

2: NAT router changes datagram source address from 10.0.0.1, 3345 to 138.76.29.7, 5001, updates table

NAT translation table	
WAN side addr	LAN side addr
138.76.29.7, 5001	10.0.0.1, 3345
.....	.....

1: host 10.0.0.1 sends datagram to 128.119.40.186, 80  
*Security from the outside*



3: reply arrives, destination address: 138.76.29.7, 5001

# NAT: network address translation

- NAT has been controversial:
  - routers “should” only process up to layer 3
  - violates end-to-end argument (port # manipulation by network-layer device)
  - NAT traversal: what if client wants to connect to server behind NAT?
- but NAT is here to stay:
  - extensively used in home and institutional nets, 4G/5G cellular nets

End-to-end arg  
↓

“The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.)

Saltzer, Reed, Clark 1981

# Middleboxes

→ NAT  
→ Firewall / load balancer

Middlebox (RFC 3234)

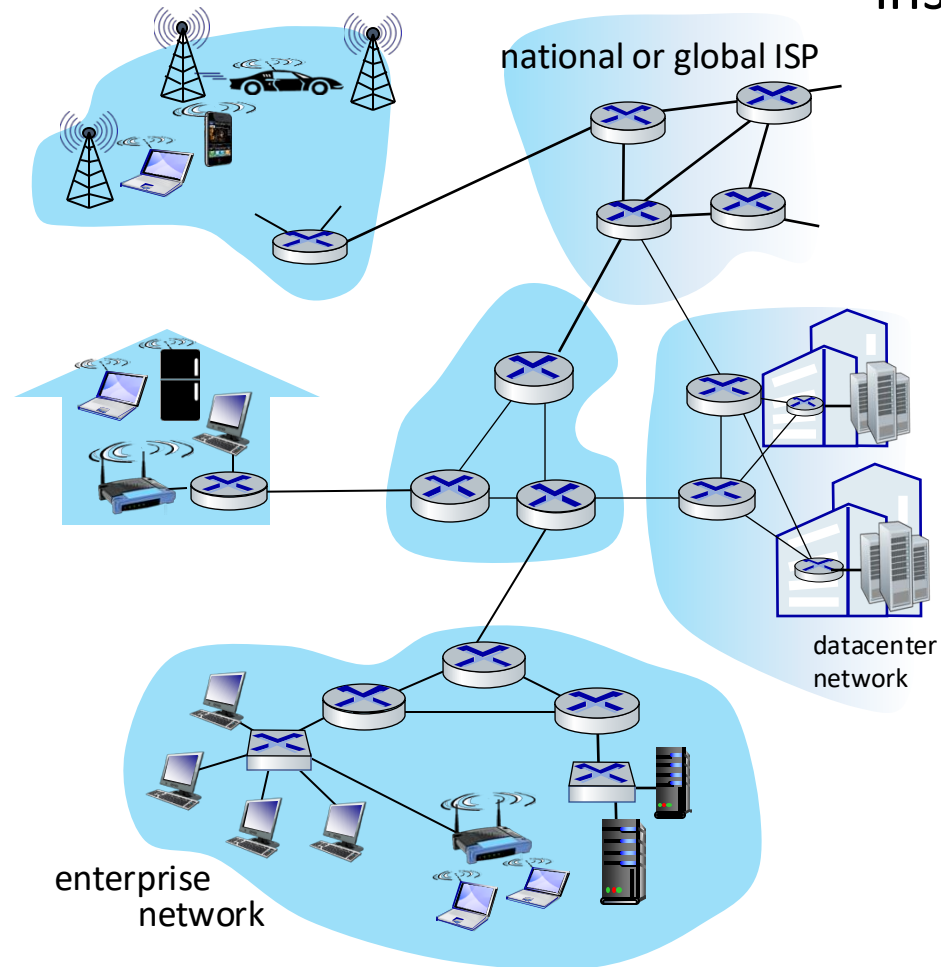
“any intermediary box performing functions apart from normal, standard functions of an IP router on the data path between a source host and destination host”



# Middleboxes everywhere!

**NAT:** home,  
cellular,  
institutional

**Application-specific:** service  
providers,  
institutional,  
CDN



**Firewalls, IDS:** corporate,  
institutional, service providers,  
ISPs

**Load balancers:**  
corporate, service  
provider, data center,  
mobile nets

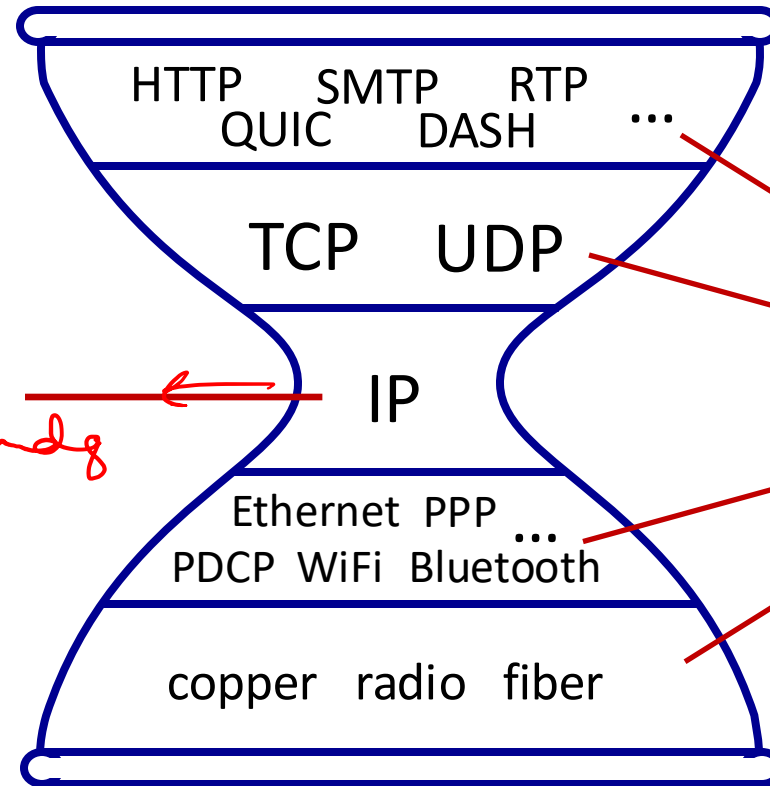
**Caches:** service  
provider, mobile, CDNs

# The IP hourglass

Internet's "thin waist":

- one network layer protocol: IP
- must be implemented by every (billions) of Internet-connected devices

*Route & Forward*

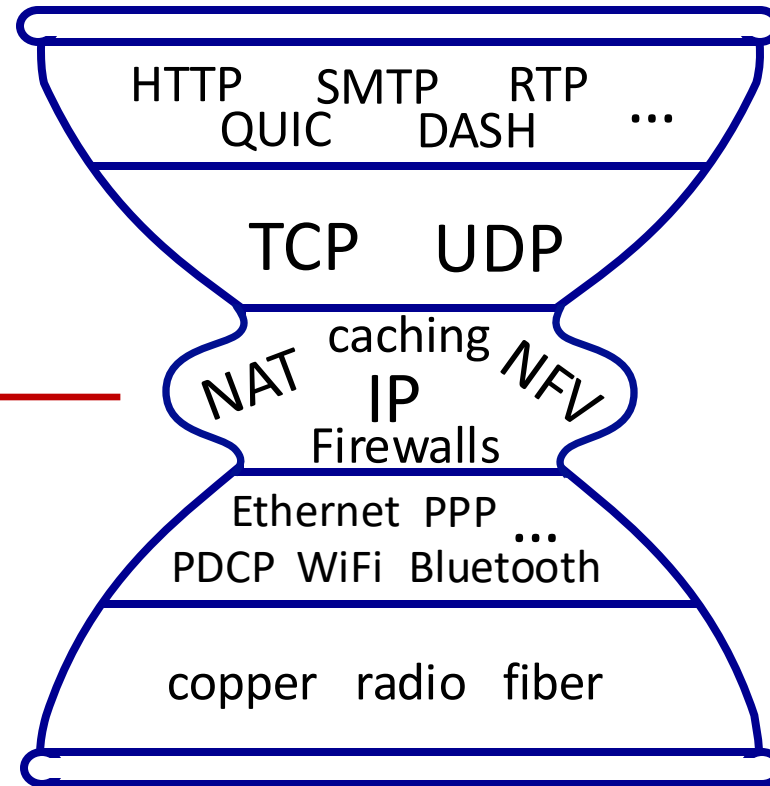


*many* protocols in physical, link, transport, and application layers

# The IP hourglass, at middle age

Internet's middle age  
"love handles"?

- middleboxes, — operating inside the network



# Today's Class

- How do you work with a limited set of IPv4?
- **How does a host get IP address from the network?**
- How does a host find the MAC address for an IP address in the subnet?
- Putting it all together

Client Server

DHCP

# IP addresses: how to get one?

x UDP  
server  
DHCP  
server

11111

N/w

End host

How does *host* get IP address?

- hard-coded by sysadmin in config file (e.g. `/etc/rc.config` in UNIX)
- **DHCP**: **D**ynamic **H**ost **C**onfiguration **P**rotocol: dynamically get address from as server
  - “plug-and-play”

Example: `10.0.0.1` to `10.0.0.255`  
Wait for IP address?

# DHCP: Dynamic Host Configuration Protocol

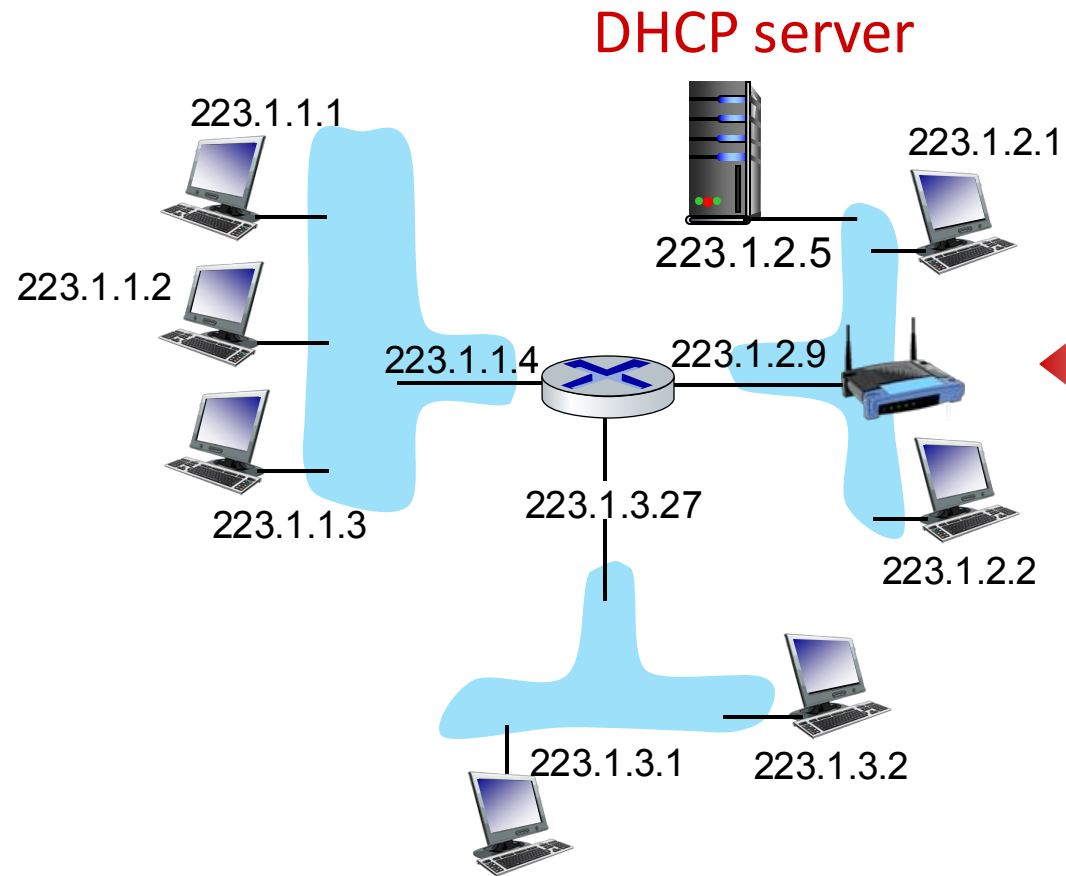
**goal:** host *dynamically* obtains IP address from network server when it “joins” network

- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/on)
- support for mobile users who join/leave network ?

## DHCP overview:

- host broadcasts **DHCP discover** msg [optional]
- DHCP server responds with **DHCP offer** msg [optional]
- host requests IP address: **DHCP request** msg
- DHCP server sends address: **DHCP ack** msg

# DHCP client-server scenario



Typically, DHCP server will be co-located in router, serving all subnets to which router is attached



arriving **DHCP client** needs address in this network

# → DHCP client-server scenario

Access Control  
list

DHCP server 2

DHCP server: 223.1.2.5



DHCP discover

Broadcast: is there a  
DHCP server out there?

DHCP offer

Broadcast: I'm a DHCP  
server! Here's an IP  
address you can use

DHCP request

Broadcast: OK. I would  
like to use this IP address!

DHCP ACK

Broadcast: OK. You've  
got that IP address!

Arriving client



The two steps above can  
be skipped "if a client  
remembers and wishes to  
reuse a previously  
allocated network address"  
[RFC 2131]



# DHCP: more than IP addresses

DHCP can return more than just allocated IP address on subnet:

- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

# Today's Class

Continue our discussion of the network layer

- How do you work with a limited set of IPv4 addresses?
- How does a host get IP address from the network?
- **How does a host find the MAC address for an IP address in the subnet?**

# How to get interface's MAC address knowing its IP address?

IP → MAC table

ARP spoofing

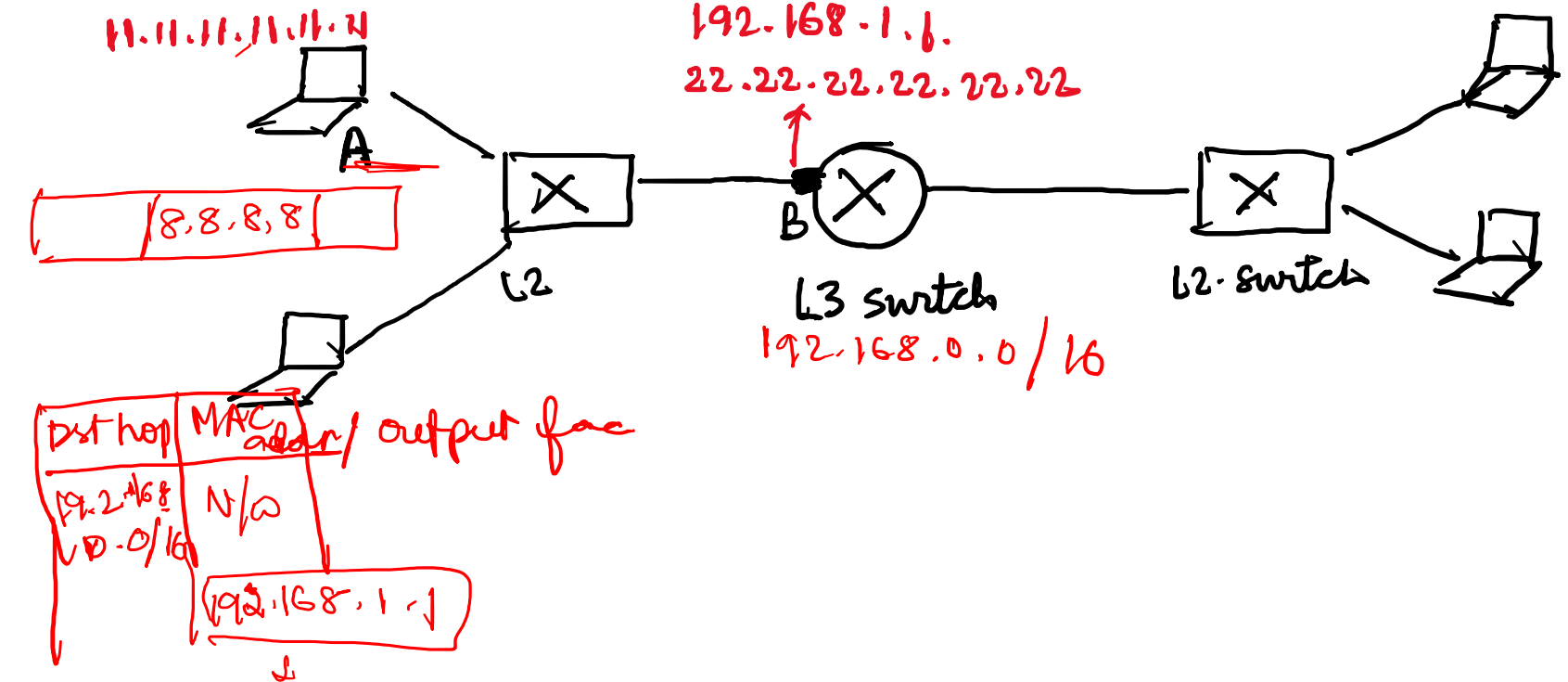
■ **Question:** how to determine interface's MAC address, knowing its IP address?

SRC MAC	DST MAC (BROADCAST)	What is MAC of 192.168.1.1
---------	---------------------	----------------------------

Response

22...	11...	He
-------	-------	----

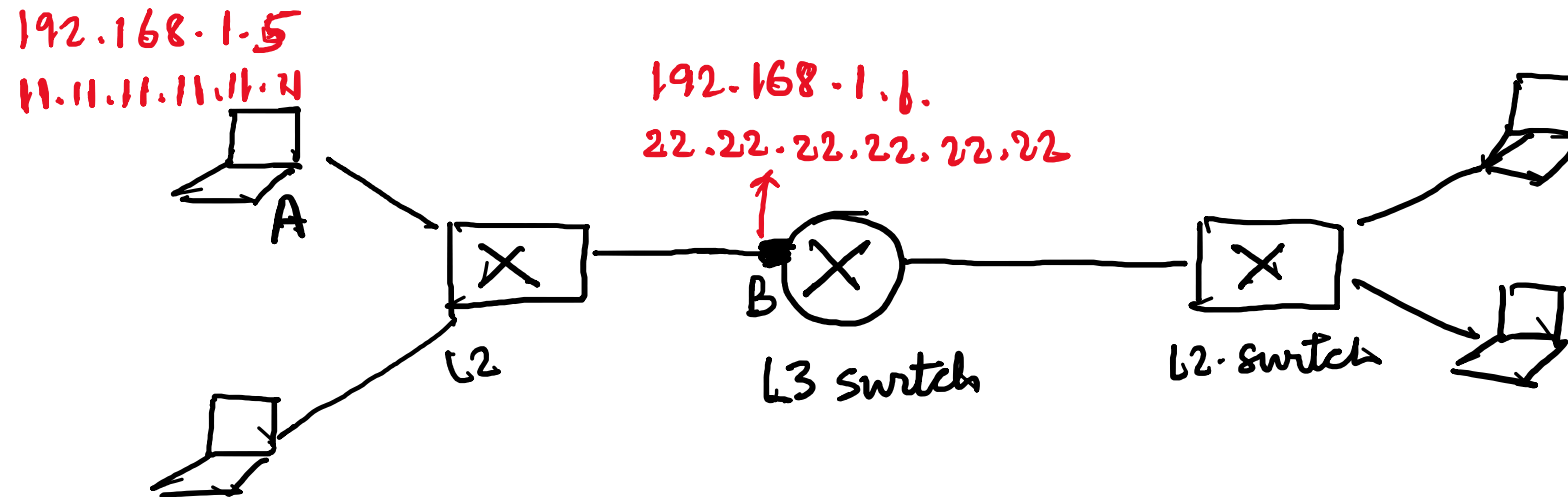
192.168.1.5  
11.11.11.11.11.11



B  
8.8.8.8

# How to get interface's MAC address knowing its IP address?

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



## Address Resolution Protocol:

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

- IP/MAC address mappings for some LAN nodes, and TTL

< IP address; MAC address; TTL >

A broadcasts ARP query, containing B's IP addr

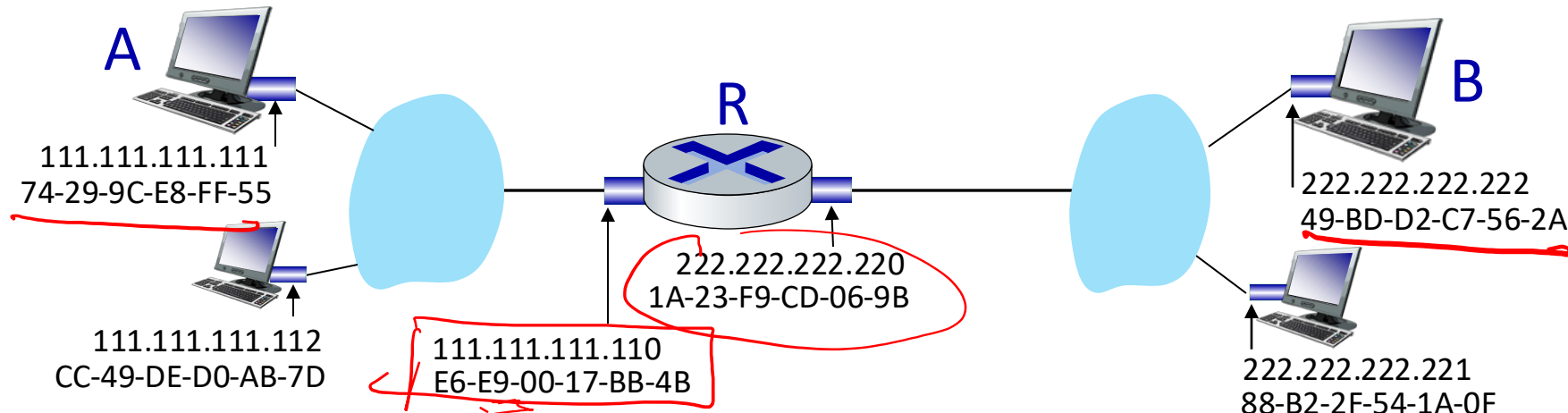
- destination MAC address = FF-FF-FF-FF-FF-FF
- all nodes on LAN receive ARP query

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

# Routing to another subnet: addressing

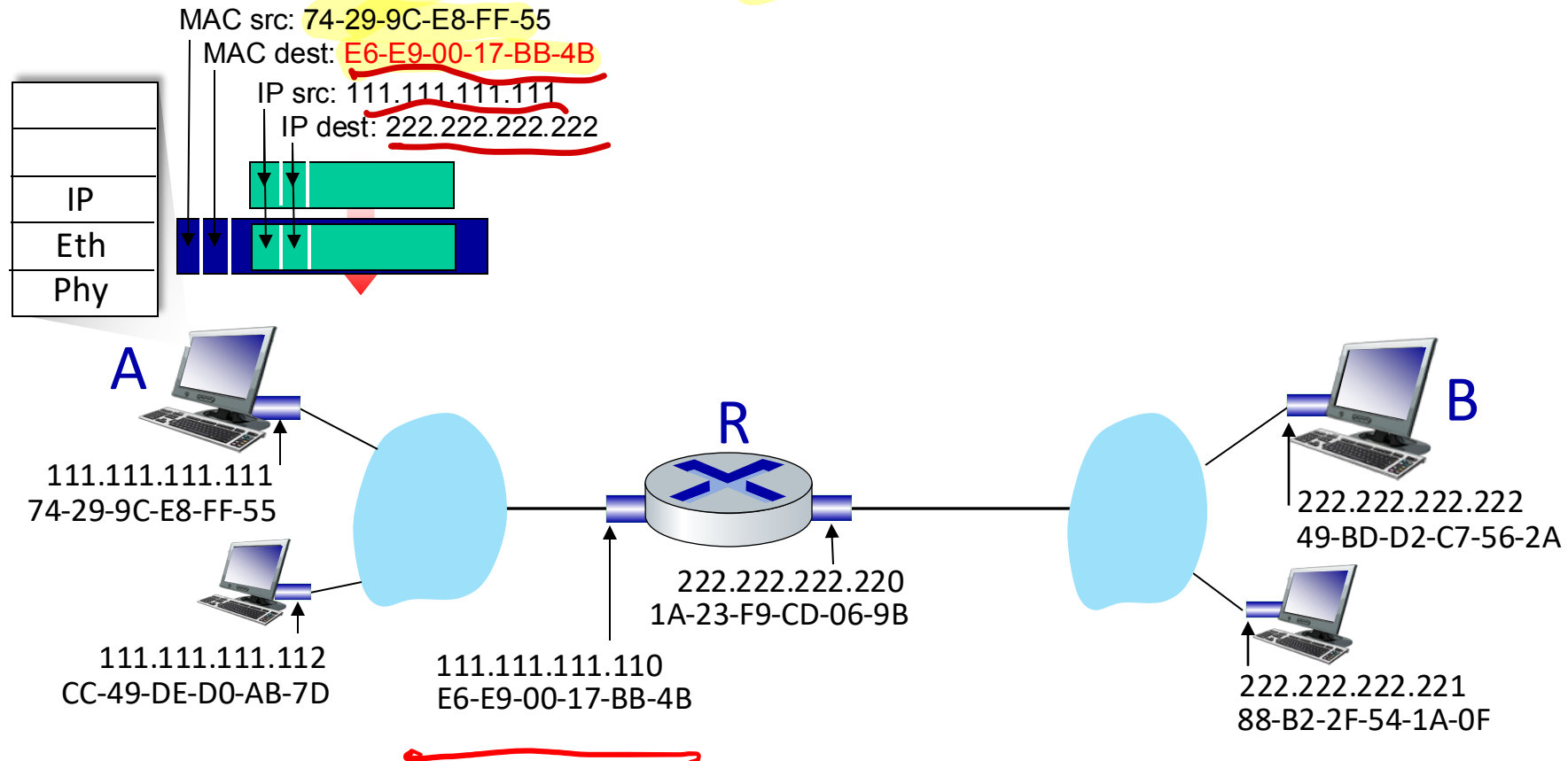
walkthrough: sending a datagram from *A* to *B* via *R*

- 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?)



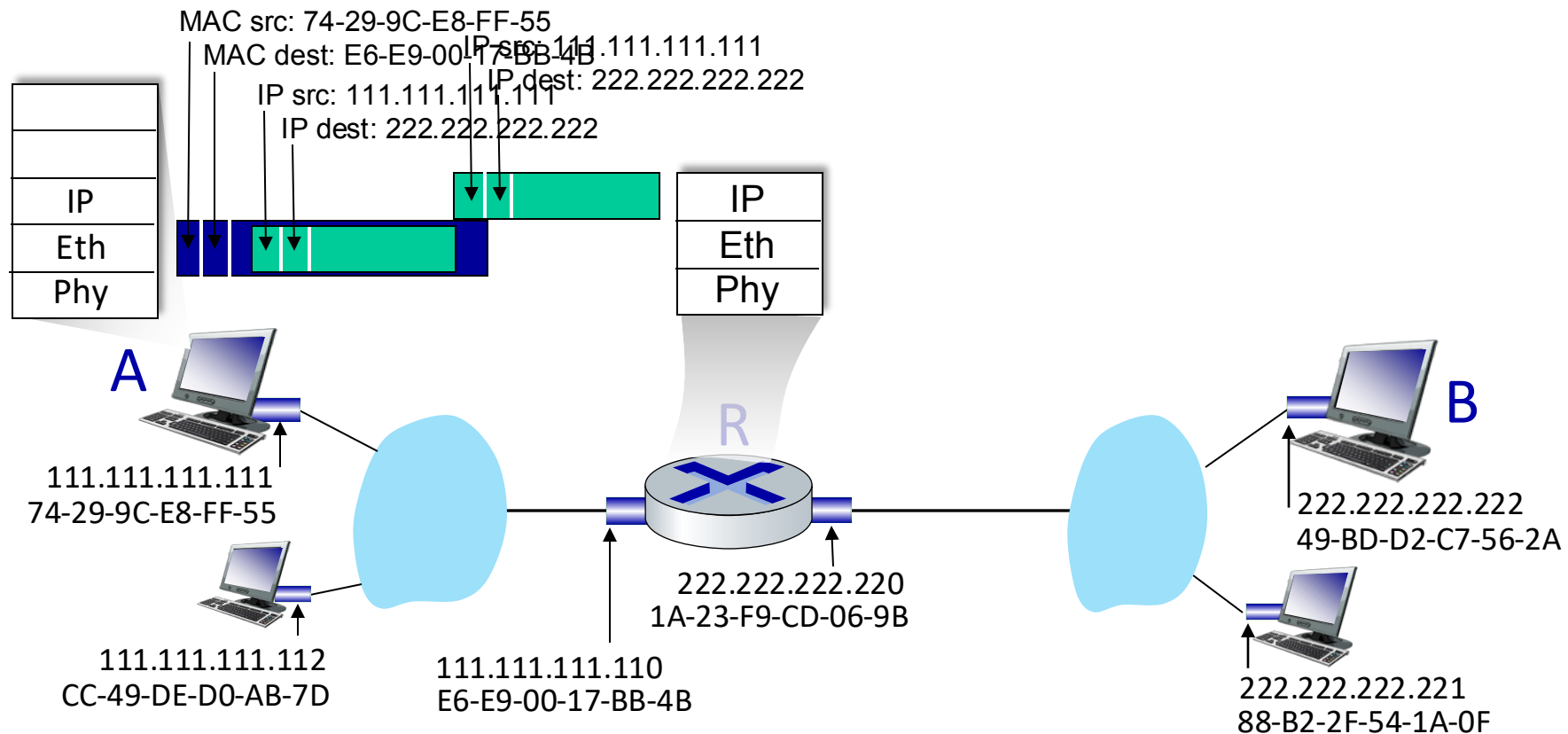
# Routing to another subnet: addressing

- 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



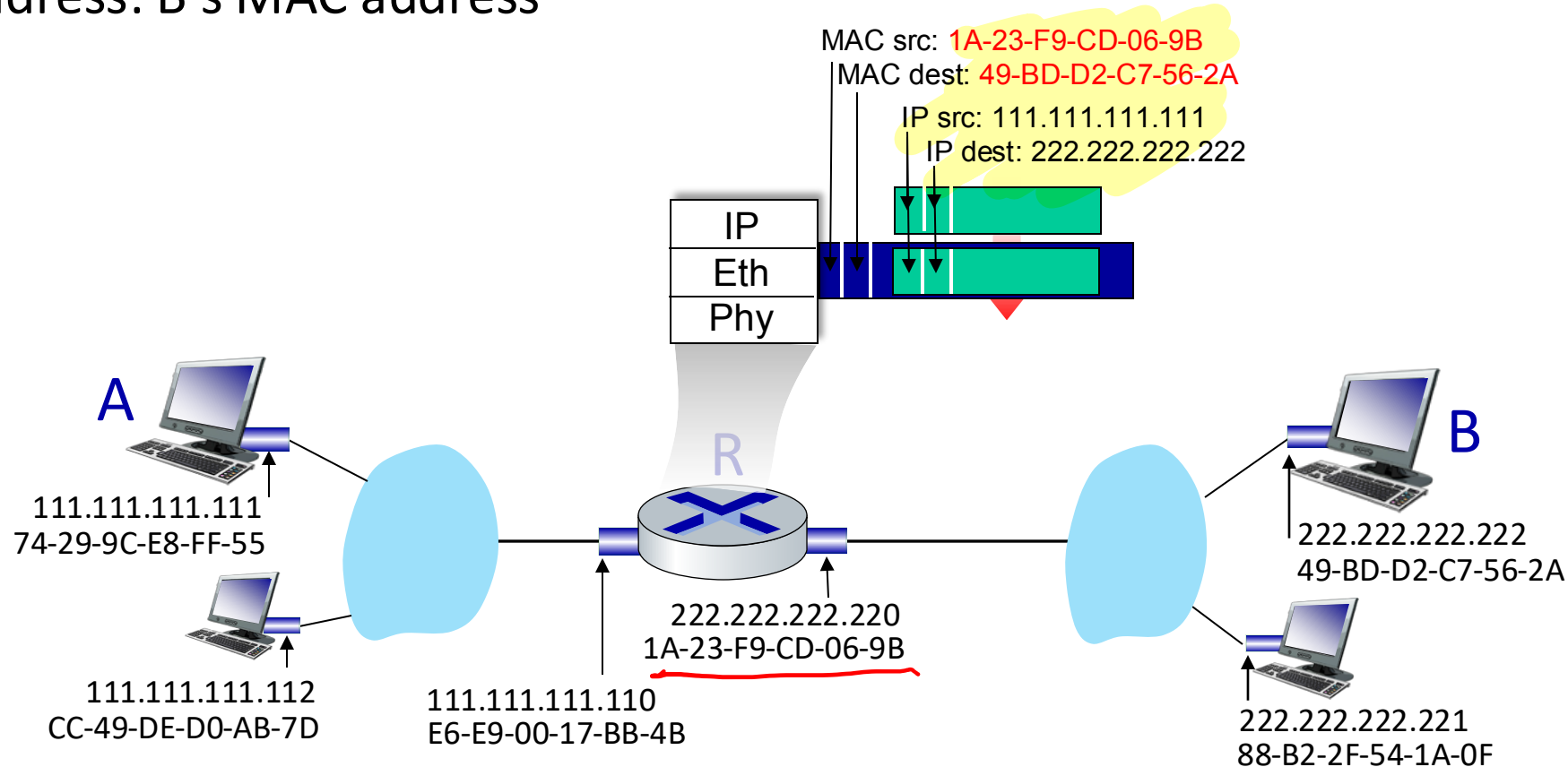
# Routing to another subnet: addressing

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



# Routing to another subnet: addressing

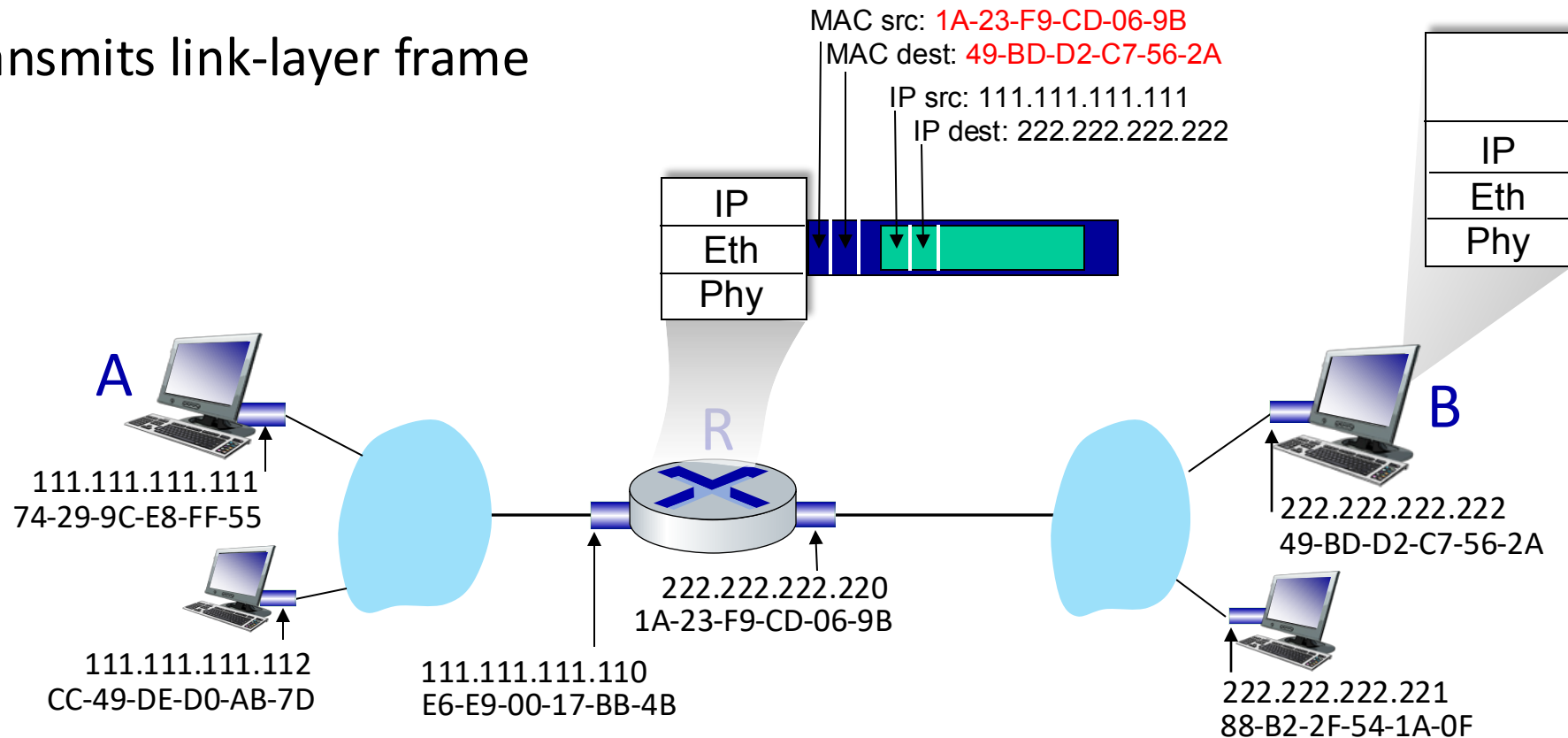
- R determines outgoing interface, passes datagram with IP source A, destination B to link layer
- R creates link-layer frame containing A-to-B IP datagram. Frame destination address: B's MAC address





# 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

