Lecture 21: Link Layer

COMP 332, Spring 2024

Victoria Manfredi

Acknowledgements: materials adapted from Computer Networking: A Top Down Approach 7th edition: ©1996-2016, J.F Kurose and K.W. Ross, All Rights Reserved as well as from slides by Abraham Matta at Boston University, and some material from Computer Networks by Tannenbaum and Wetherall.

Today

1. Announcements
   - Homework 7 due Wed. at 11:59p
   - Homework 8 posted (no coding)

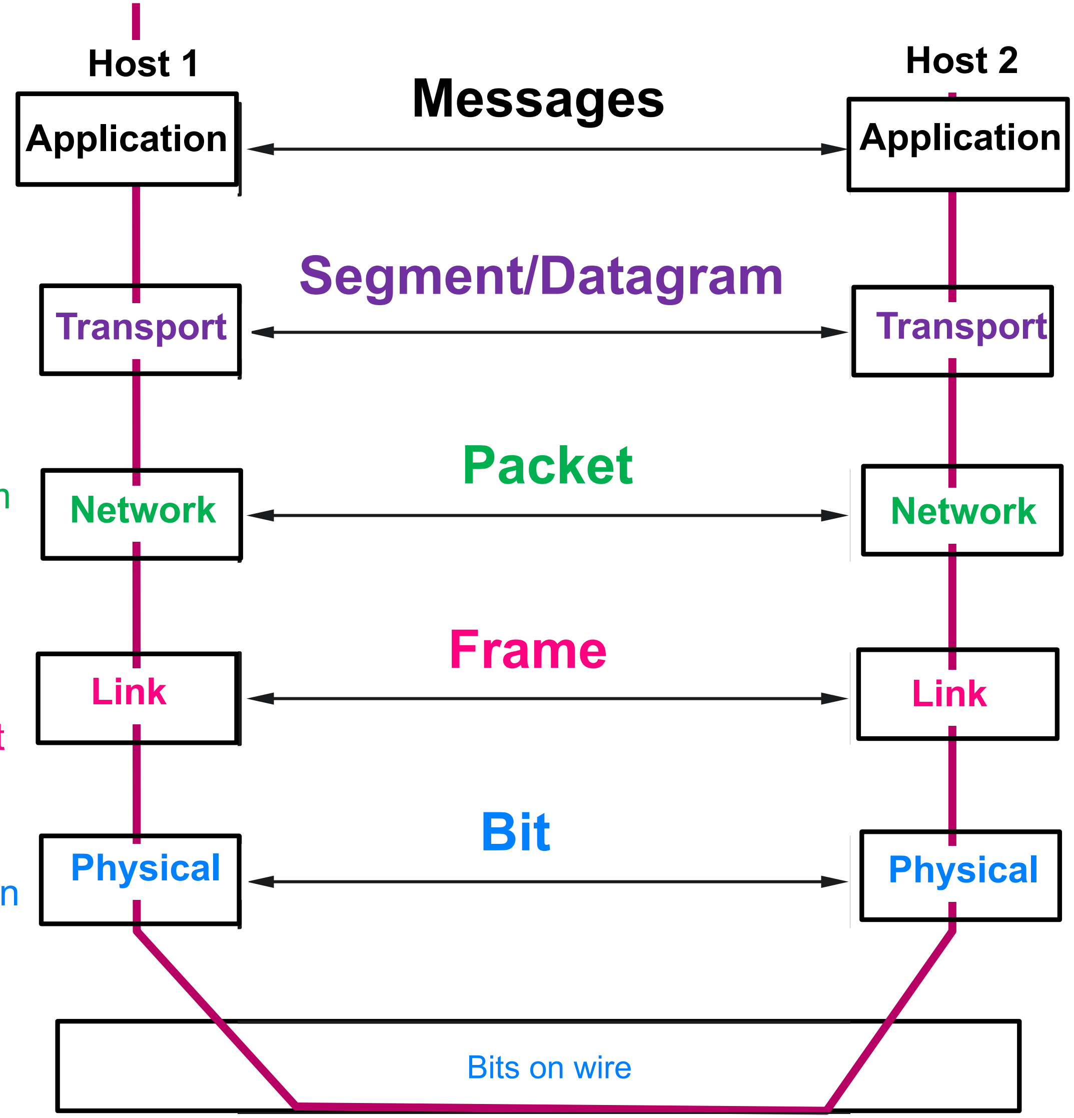
2. Link layer
   - overview
   - MAC addresses
   - Address Resolution Protocol (ARP)
   - Switches

3. A day in the life of a web request

Link Layer
OVERVIEW

Internet protocol stack

Protocols	Service provided
• HTTP, DNS, DHCP, TLS, …	• Support network applications
• TCP, UDP, QUIC, …	• Deliver msgs to app endpoints, flow control, reliability
• IP, routing protocols (BGP, OSPF, RIP, …)	• Route segments from source to destination host (indirect)
• Ethernet, 802.11, …	• Move pkt over link from one host to next host (direct)
• Ethernet phy, 802.11 phy, …	• Move individual bits in frame from one host to next



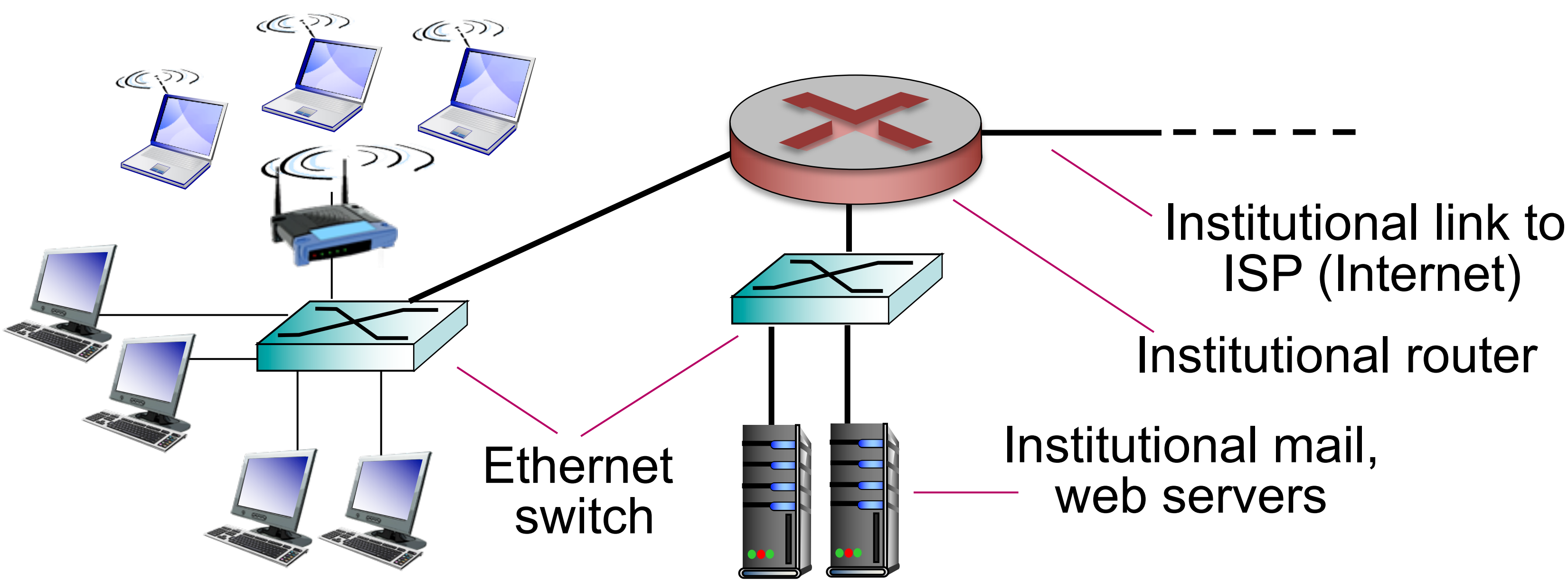
Link layer

Goal

– divide packet up into frames, transfer frame across link

Different link media have different characteristics

– will use different protocols with different services to transfer frame
– e.g., in local area network (LAN), frames broadcast to every device



Link layer services

Framing

- encapsulate packet into frame
- add header, trailer to detect start/end of frame

Link access

- if shared media like wireless link, how to share?
  - time/freq division, random access, …
- MAC addresses in frame headers identify src, dst
  - different from IP address, only used within network

Reliable delivery between 2 end hosts of link

- low-bit error links rarely use: fiber, some twisted pair
- high bit error: wireless link

Q: why both link-level and end-end reliability?

Other link layer services

Flow control

- pacing between adjacent sending and receiving nodes

Error detection

- errors caused by signal attenuation, noise, interference, …
- receiver detects presence of errors, e.g., via checksum
  - signals sender for retransmission or drops frame

Error correction

- receiver identifies and corrects bit error(s)
- don't necessarily need to retransmit: instead use coding

Half-duplex vs. full-duplex

- half duplex, e.g., wireless link
  - nodes at both ends of link can transmit, but not at same time

Where is the link layer implemented?

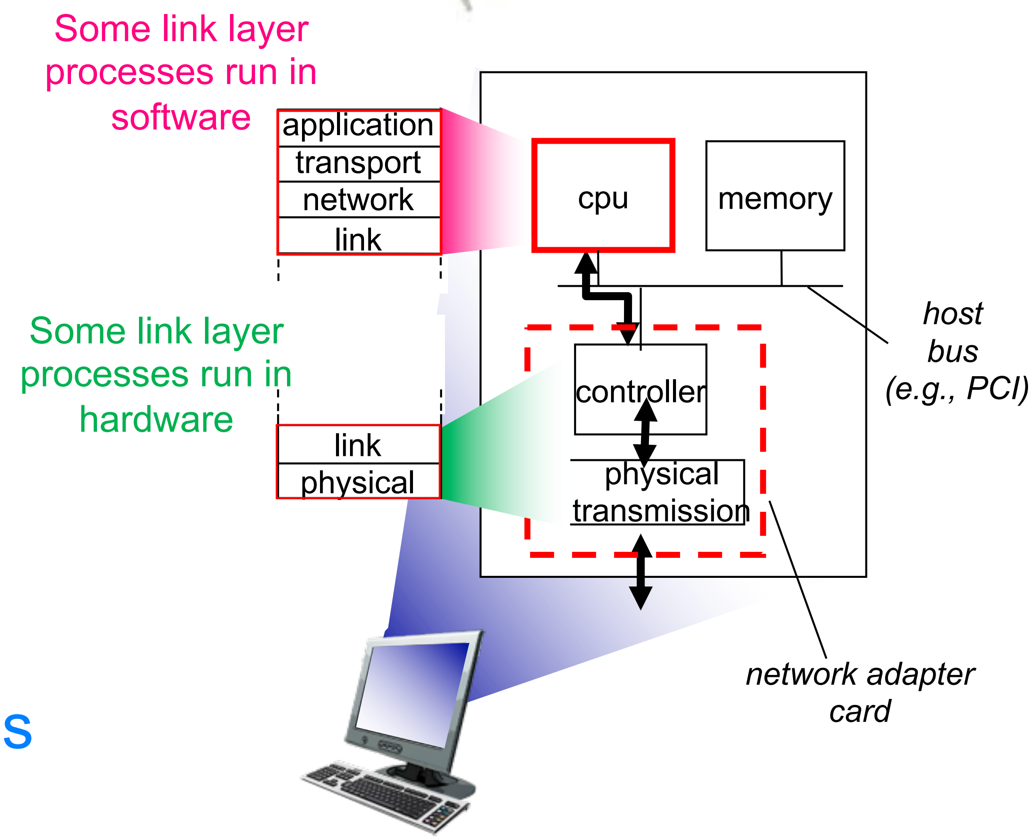
In network interface card (NIC) or on chip

- Ethernet card
- 802.11 card
- Ethernet chipset

Implements

- link layer
- physical layer: e.g., transmit radio wave

Attaches into system buses on host's system



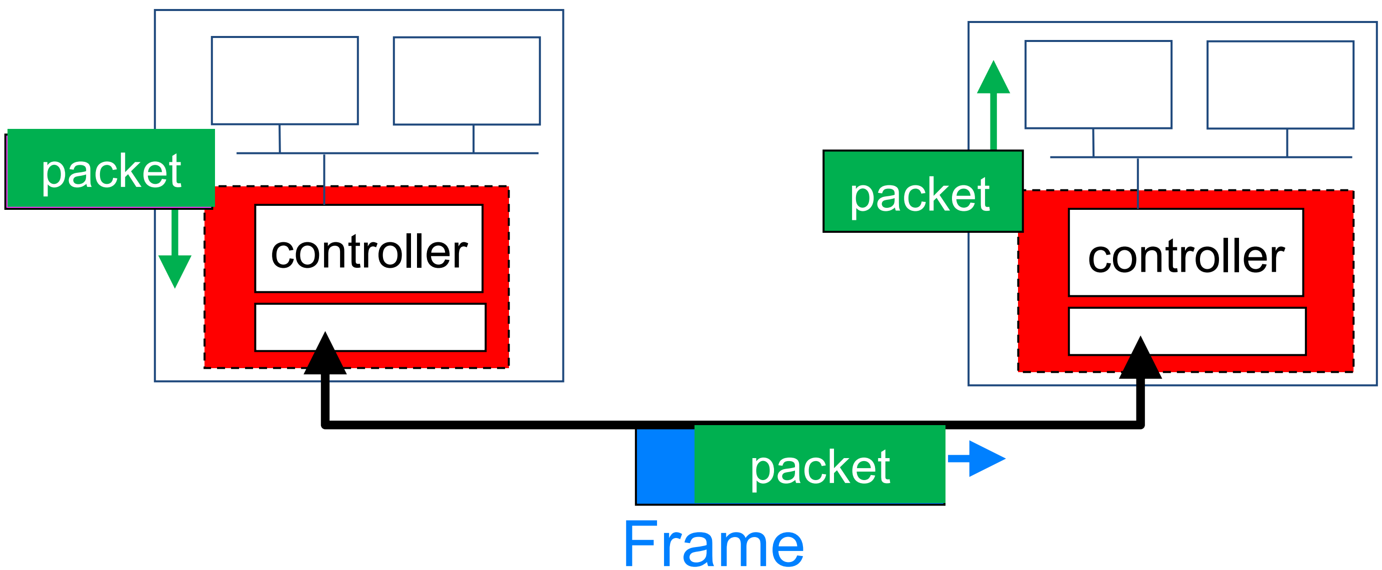
Communication between network interfaces

Sender

- encapsulates pkt in frame
- adds error checking bits, rdt, flow control, …

Receiver

- looks for errors, rdt, flow control, …
- extracts pkt, passes to upper layer at receiving side



Link Layer

MAC ADDRESSES

MAC addresses

32-bit IP address

- software address: network-layer address for interface
- used for layer 3 (network layer) forwarding

48-bit MAC address

- link-layer address for interface
  - aka hardware, Ethernet, LAN, or physical address
  - e.g., 1A-2F-BB-76-09-AD
  - burned in NIC read only memory, also sometimes software settable
- used for layer 2 (link layer) forwarding
  - get frame from one interface to another physically-connected interface

Why both MAC and IP addresses?

IP address: like postal address

- hierarchical address: not portable
- changes with location
  - address depends on IP subnet to which node is attached

MAC address: like SSN

- flat address: portable
- does not change with location
  - can move LAN card from one LAN to another

LANs designed for arbitrary network layer protocol, not just IP

- don’t want to pass frame up to network layer for every frame
  - faster, even if on same LAN to not go up to network layer

Your MAC address

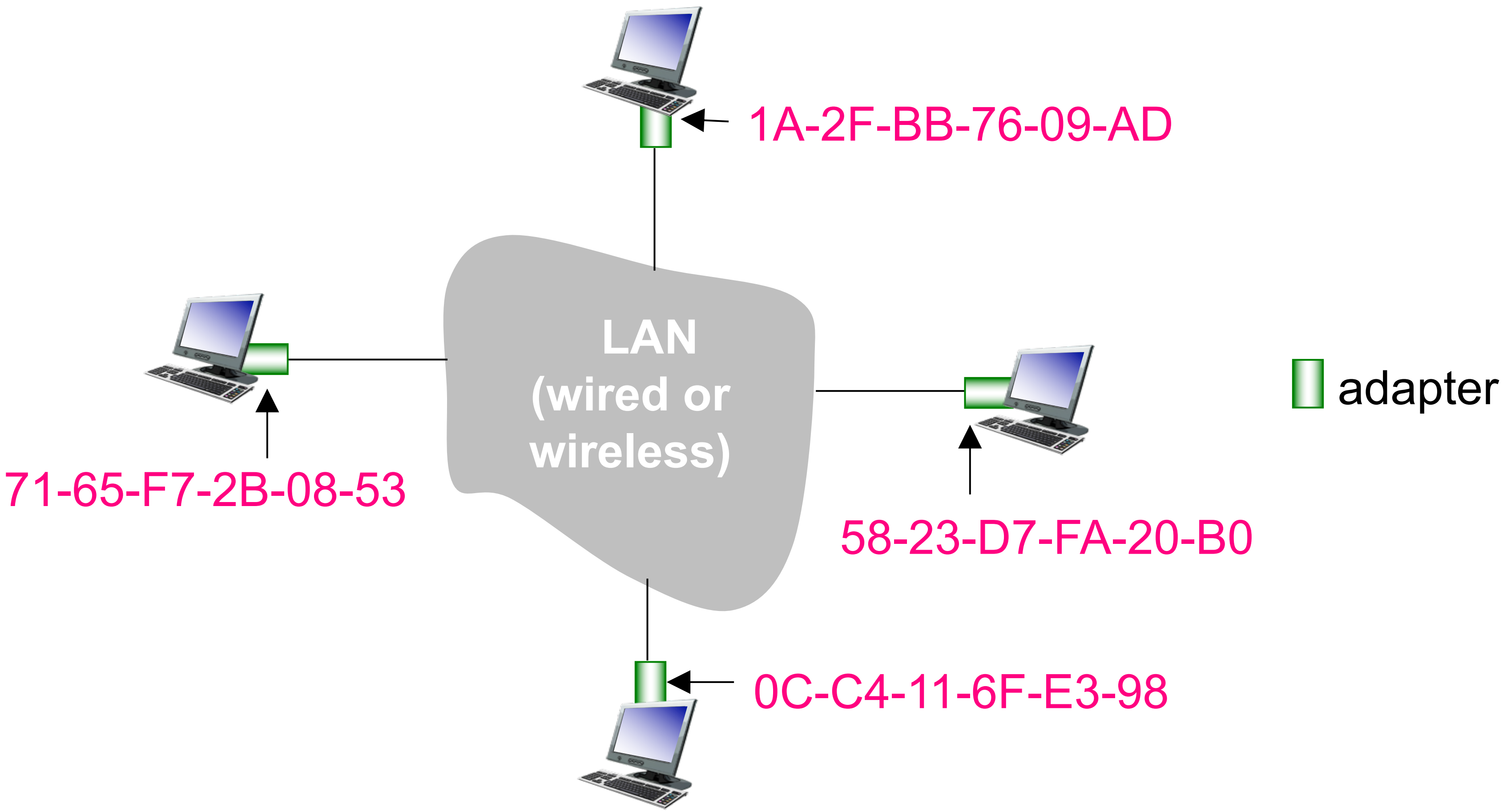
```
> ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
        inet 127.0.0.1 netmask 0xff000000
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        nd6 options=201<PERFORMNUD,DAD>
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        ether 78:4f:43:73:43:26
        inet6 fe80::1c8d:4bcb:b52d:9d1d%en0 prefixlen 64 secured scopeid 0x5
        inet 129.133.187.174 netmask 0xfffff000 broadcast 129.133.191.255
```

```
▶ Frame 264: 1440 bytes on wire (11520 bits), 1440 bytes captured (11520 bits) on interface 0
▼ Ethernet II, Src: JuniperN_1e:18:01 (3c:8a:b0:1e:18:01), Dst: Apple_73:43:26 (78:4f:43:73:43:26)
  ▶ Destination: Apple_73:43:26 (78:4f:43:73:43:26)
  ▶ Source: JuniperN_1e:18:01 (3c:8a:b0:1e:18:01)
    Type: IPv4 (0x0800)
▶ Internet Protocol Version 4, Src: a104-96-210-190.deploy.static.akamaitechnologies.com (104.96.2
▶ Transmission Control Protocol, Src Port: 443, Dst Port: 57106, Seq: 730864352, Ack: 3232279727, L
```

Take a minute and find your MAC address

LAN addresses and ARP

Each adapter on LAN has unique LAN address



LAN: Local Area Network
ARP: Address Resolution Protocol

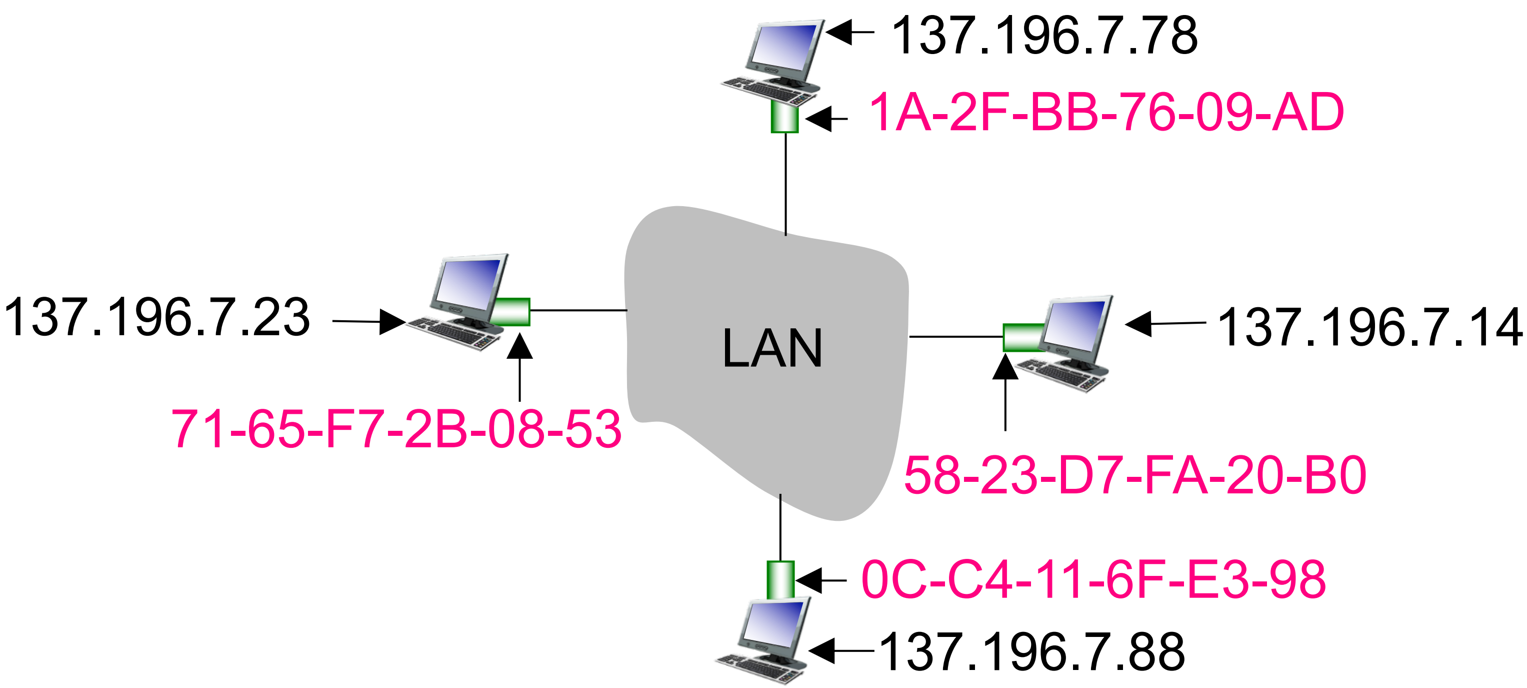
Link Layer

ADDRESS RESOLUTION PROTOCOL

Address Resolution Protocol (ARP)

ARP

- link layer protocol that translates between MAC and IP addr
- ARP table in every IP device on LAN
  - <IP addr; MAC addr; TTL>
  - TTL: time after which addr mapping forgotten (typically 20 min)



Q: what layer is ARP in?

ARP in wireshark

No.	Time	Source	Destination	Length	Info
5406	48.129810	Apple_73:43:26	6e:57:ca:90:05:64	42	Who has 172.20.10.1? Tell 172.20.10.11
5410	48.153207	Apple_73:43:26	6e:57:ca:90:05:64	42	Who has 172.20.10.1? Tell 172.20.10.11
5413	48.193996	Apple_73:43:26	6e:57:ca:90:05:64	42	Who has 172.20.10.1? Tell 172.20.10.11
5416	48.277611	Apple_73:43:26	6e:57:ca:90:05:64	42	Who has 172.20.10.1? Tell 172.20.10.11
5417	48.280822	6e:57:ca:90:05:64	Apple_73:43:26	42	172.20.10.1 is at 6e:57:ca:90:05:64
5418	48.281053	Apple_73:43:26	Broadcast	42	Gratuitous ARP for 172.20.10.11 (Request)
5423	48.376210	Apple_73:43:26	Broadcast	42	Who has 172.20.10.1? Tell 172.20.10.11
5424	48.377694	6e:57:ca:90:05:64	Apple_73:43:26	42	172.20.10.1 is at 6e:57:ca:90:05:64
5661	51.723958	vmanfredis-MacBook-Pro-2.l…	Broadcast	42	Who has 172.20.10.1? Tell 172.20.10.11
5662	52.043516	vmanfredis-MacBook-Pro-2.l…	Broadcast	42	Gratuitous ARP for 172.20.10.11 (Request)
5696	52.217609	6e:57:ca:90:05:64	vmanfredis-MacBoo…	42	172.20.10.1 is at 6e:57:ca:90:05:64
5721	52.367215	vmanfredis-MacBook-Pro-2.l…	Broadcast	42	Who has 172.20.10.1? Tell 172.20.10.11
5802	52.483589	6e:57:ca:90:05:64	vmanfredis-MacBoo…	42	172.20.10.1 is at 6e:57:ca:90:05:64

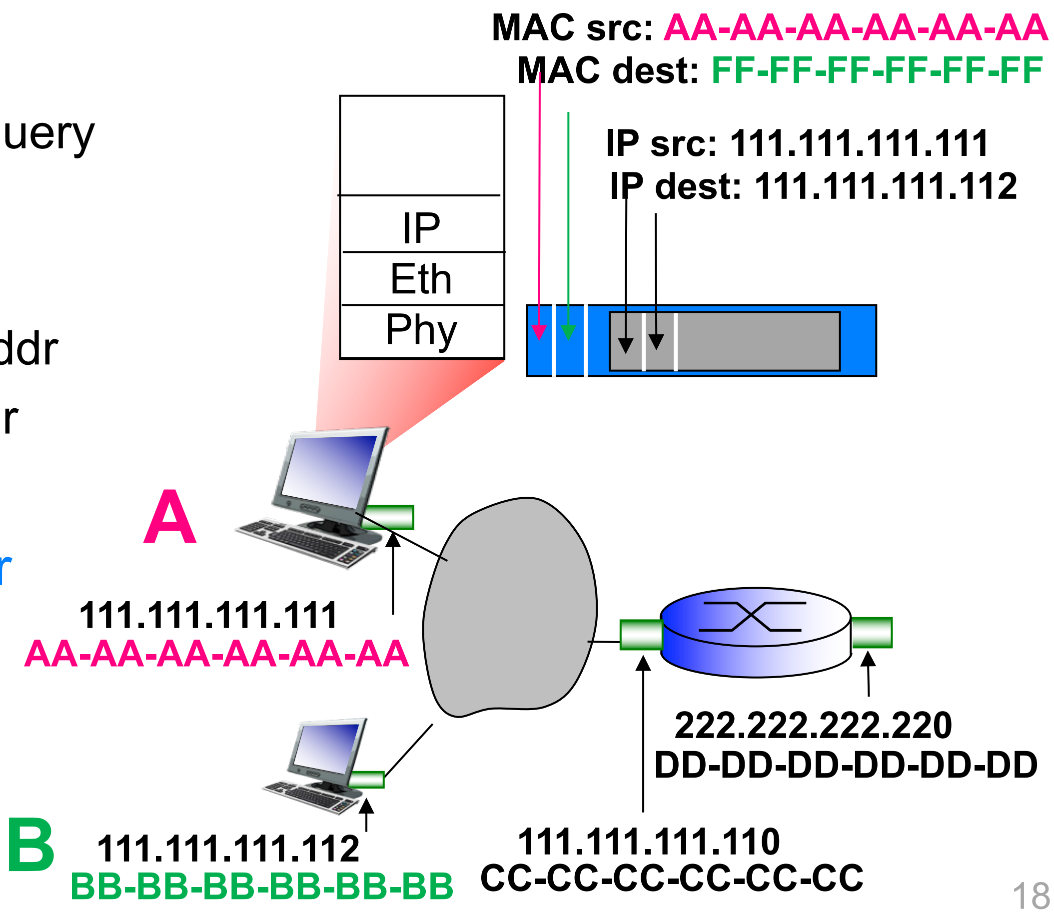
```
▶ Frame 5406: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▼ Ethernet II, Src: vmanfredis-MacBook-Pro-2.local (78:4f:43:73:43:26), Dst: 6e:57:ca:90:05:64 (6e:57:ca:90:05:64)
  ▶ Destination: 6e:57:ca:90:05:64 (6e:57:ca:90:05:64)
  ▶ Source: vmanfredis-MacBook-Pro-2.local (78:4f:43:73:43:26)
    Type: ARP (0x0806)
▼ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: vmanfredis-MacBook-Pro-2.local (78:4f:43:73:43:26)
    Sender IP address: vmanfredis-MacBook-Pro-2.local (172.20.10.11)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 172.20.10.1 (172.20.10.1)
```

Try running wireshark yourself and filter for arp

Forwarding within same LAN

A wants to send pkt to B but B's MAC addr not in A's ARP table

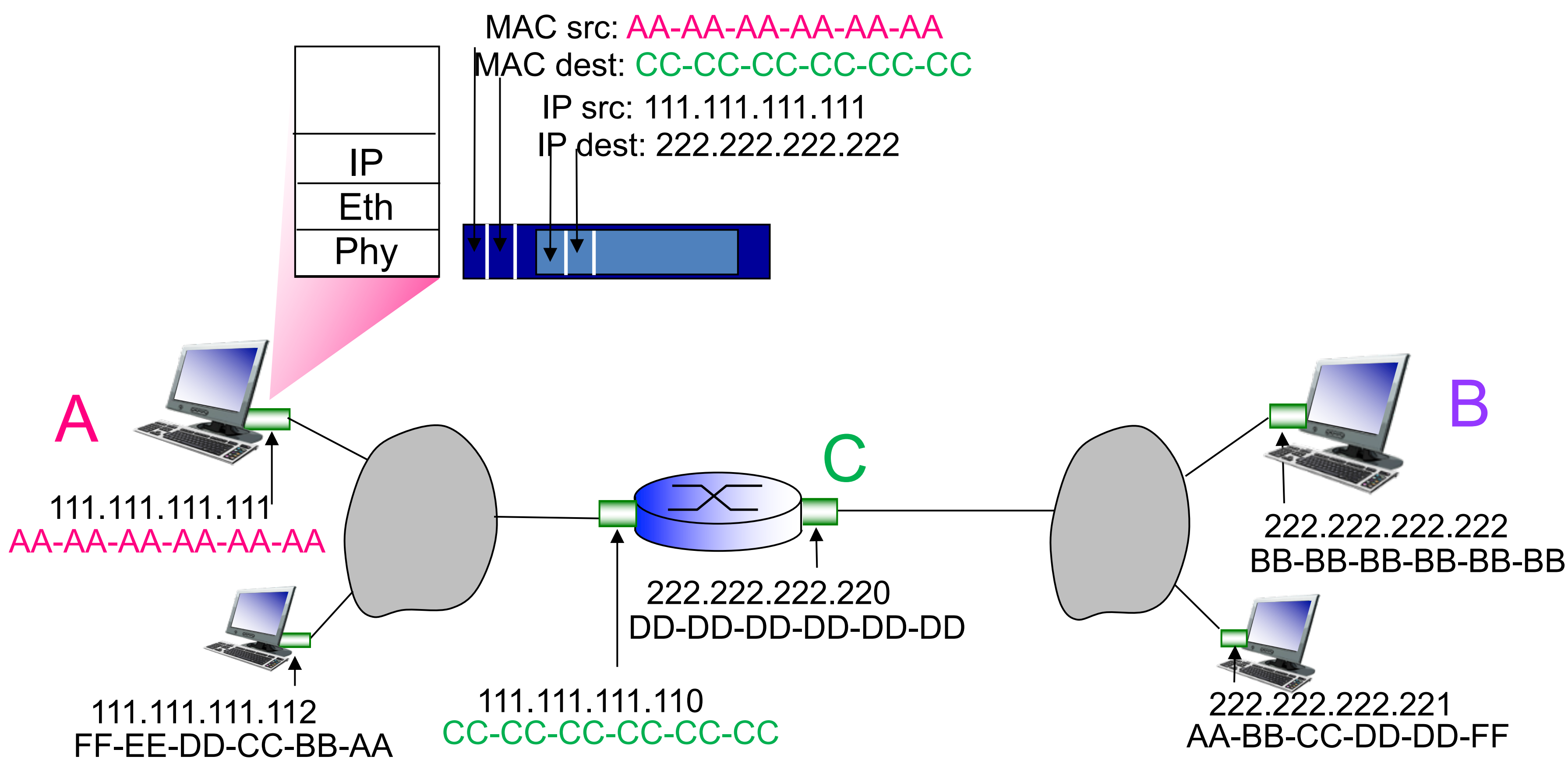
1. A broadcasts ARP query containing B's IP addr
   - destination MAC addr
     - FF-FF-FF-FF-FF-FF
   - all nodes on LAN receive query
2. B receives ARP query
   - replies to A with its MAC addr
   - frame sent to A's MAC addr
3. A caches IP,MAC addr pair
   - until TTL expires



Routing to another LAN

Send pkt from A to B via gateway router C

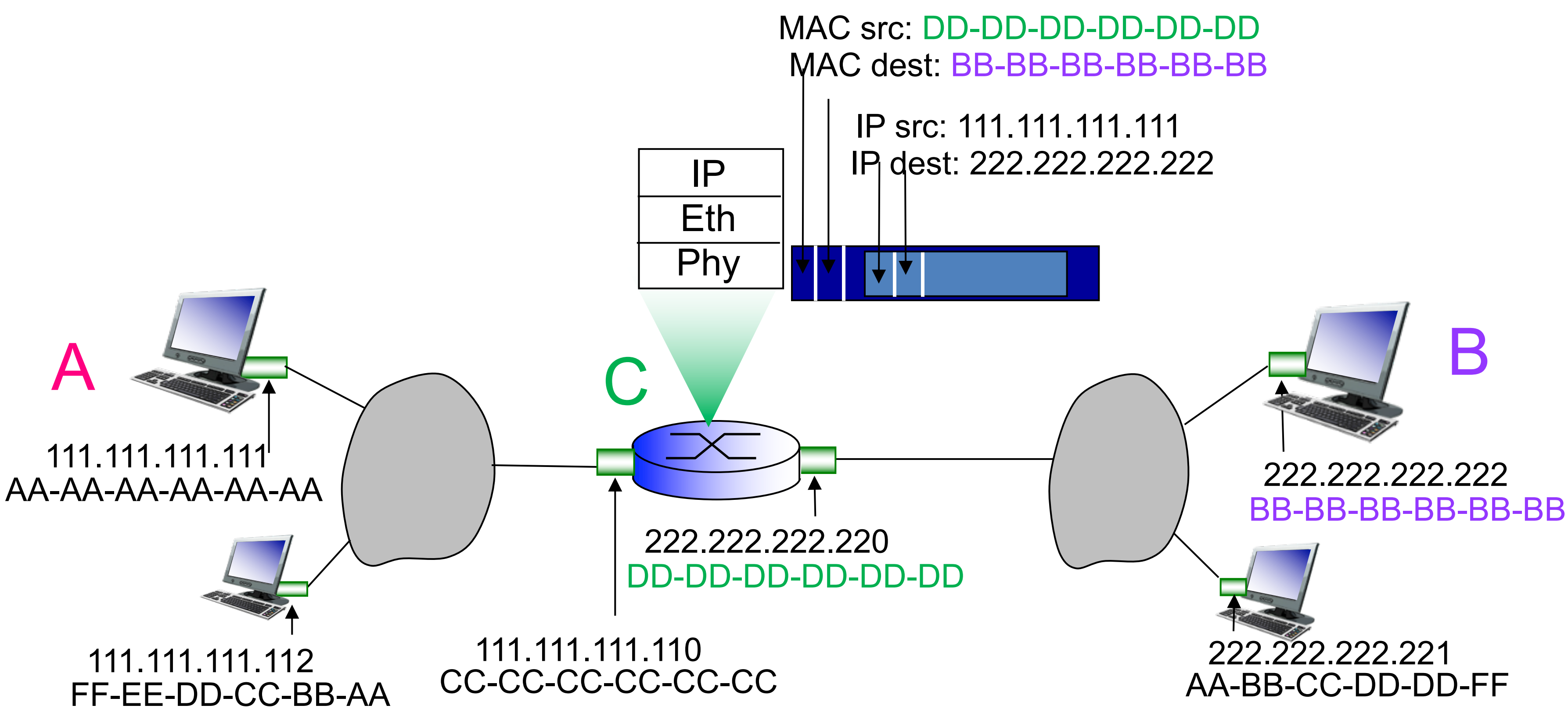
- assume A knows B's IP addr, C's IP addr, C's MAC addr



Routing to another LAN

Send pkt from A to B via gateway router C

– assume A knows B's IP addr, C's IP addr, C's MAC addr

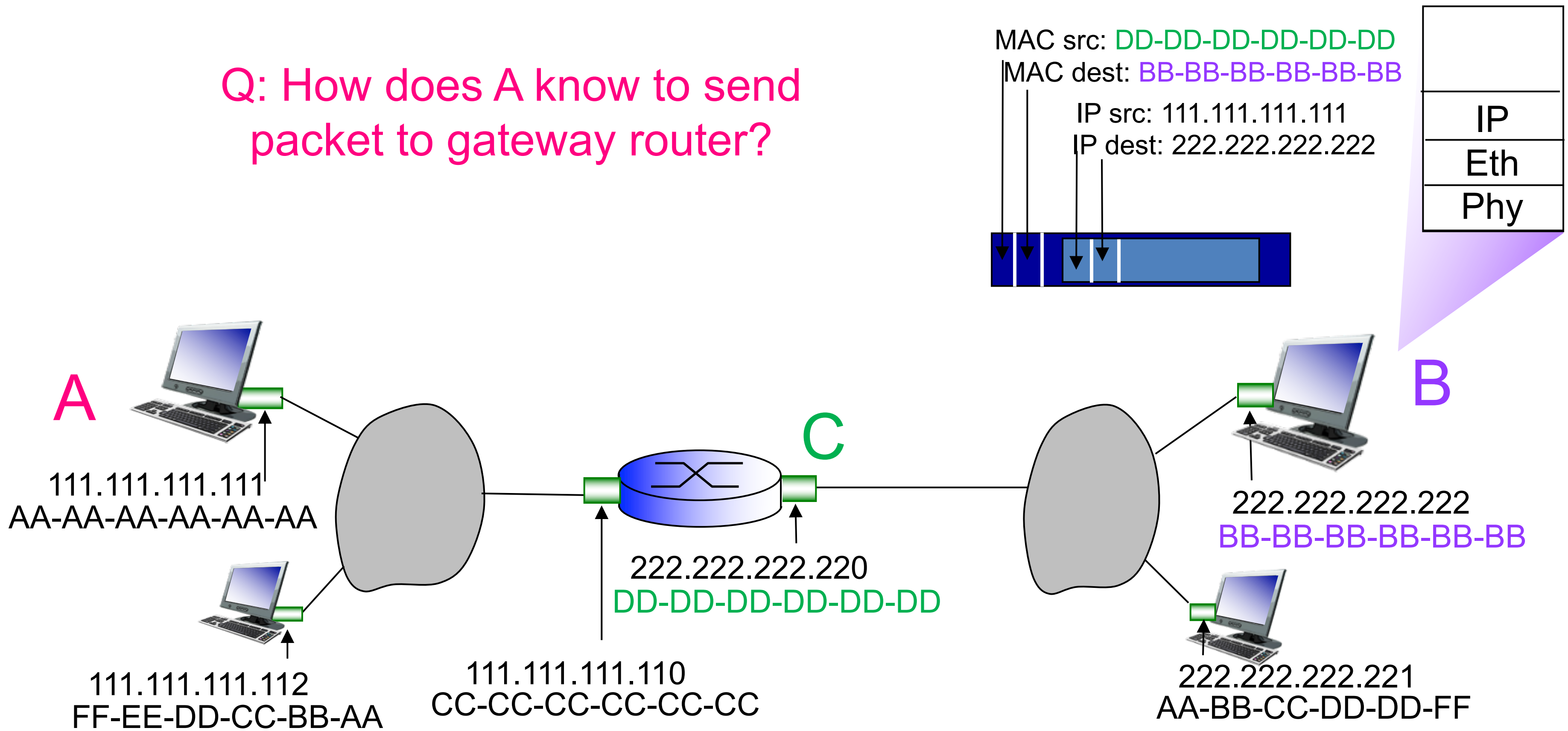


Routing to another LAN

Send pkt from A to B via gateway router C

– assume A knows B's IP addr, C's IP addr, C's MAC addr

Q: How does A know to send packet to gateway router?



Link Layer

SWITCHES

Ethernet switch

Ethernet

- dominant wired link layer protocol

Switch

- link-layer device to store and forward Ethernet frames
  - examine incoming frame’s MAC address
  - selectively forward frame to one-or-more outgoing links
- transparent
  - hosts are unaware of presence of switches
- self-learning
  - switches do not need to be configured

Aside

- you’ll see Ethernet listed as link layer protocol when you use wifi
  - quirk of wireshark, which we won’t get into

Switches vs. routers

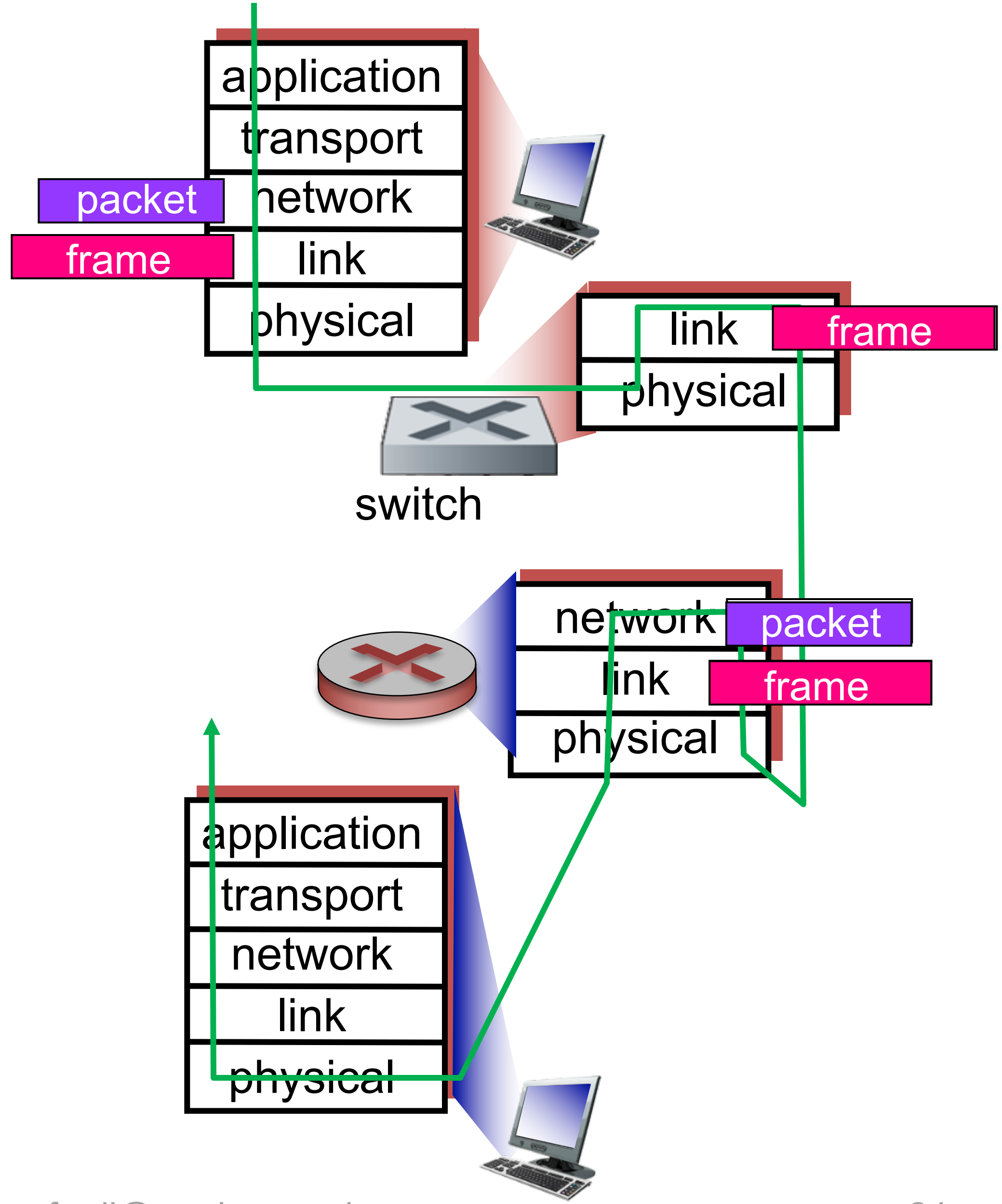
Both are store-and-forward

Routers

- data plane only examines network-layer headers
- control plane (BGP) may look at app layer

Switches

- examine link-layer headers



Switches vs. routers

Both have forwarding tables

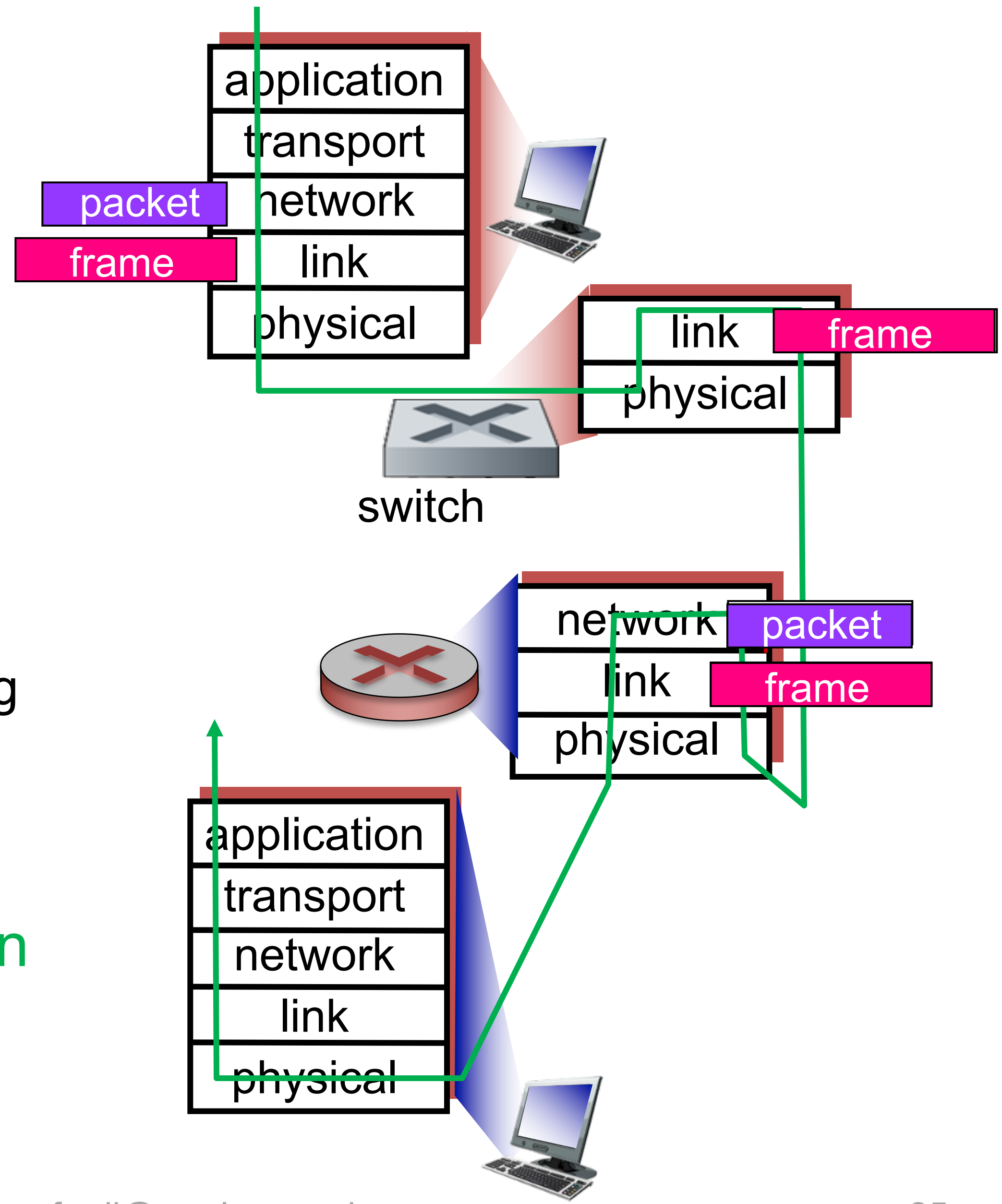
Routers

– compute tables using routing algorithms, IP addr

Switches

– learn forwarding table using flooding, MAC addr

Q: Key differences between switches and routers?



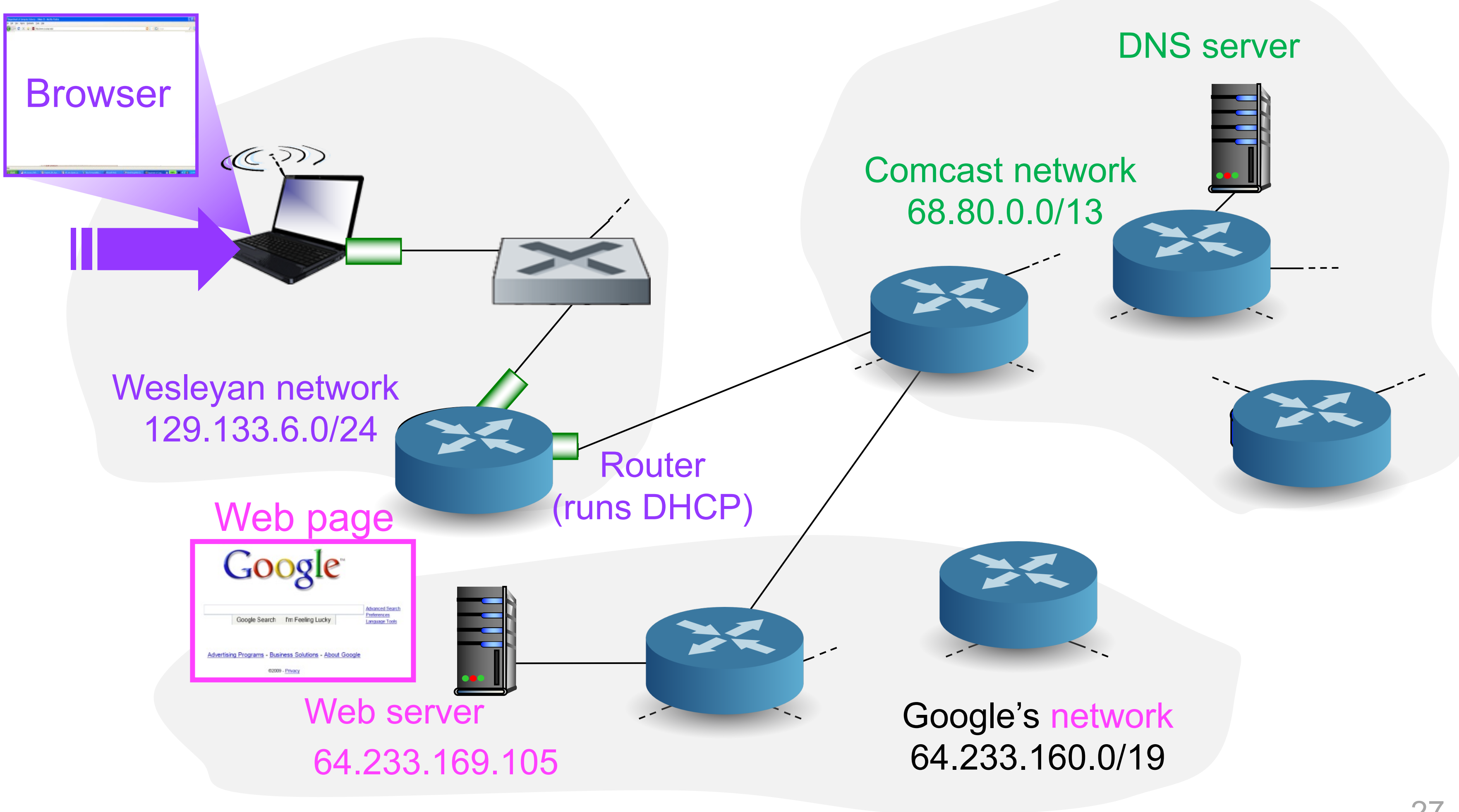
Summarizing Example

A DAY IN THE LIFE OF A WEB REQUEST

What really happens when you enter URL?

How does your laptop download www.google.com?

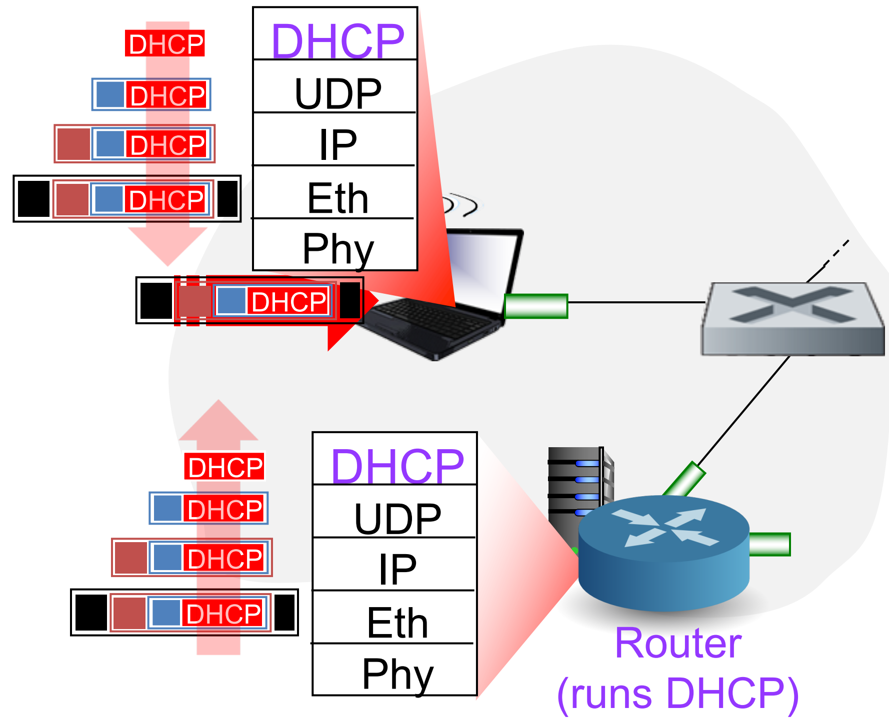
3 min: write down the protocols



Connecting to Internet

Connecting laptop needs

- its own IP addr
- IP addr of first-hop router
- IP addr of DNS server



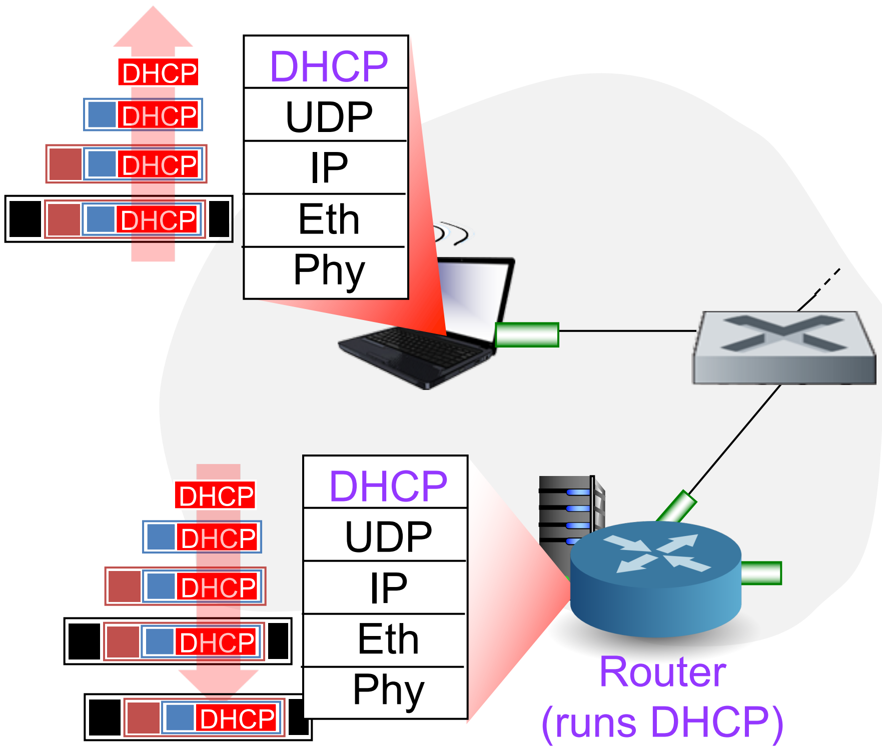
How? DHCP request

- encapsulated in UDP
- encapsulated in IP
- encapsulated in Ethernet
- broadcast on LAN
  - dst: FF-FF-FF-FF-FF-FF

Router running DHCP server receives DHCP request

- Ethernet demuxed to IP demuxed to UDP demuxed to DHCP

DHCP server sends response



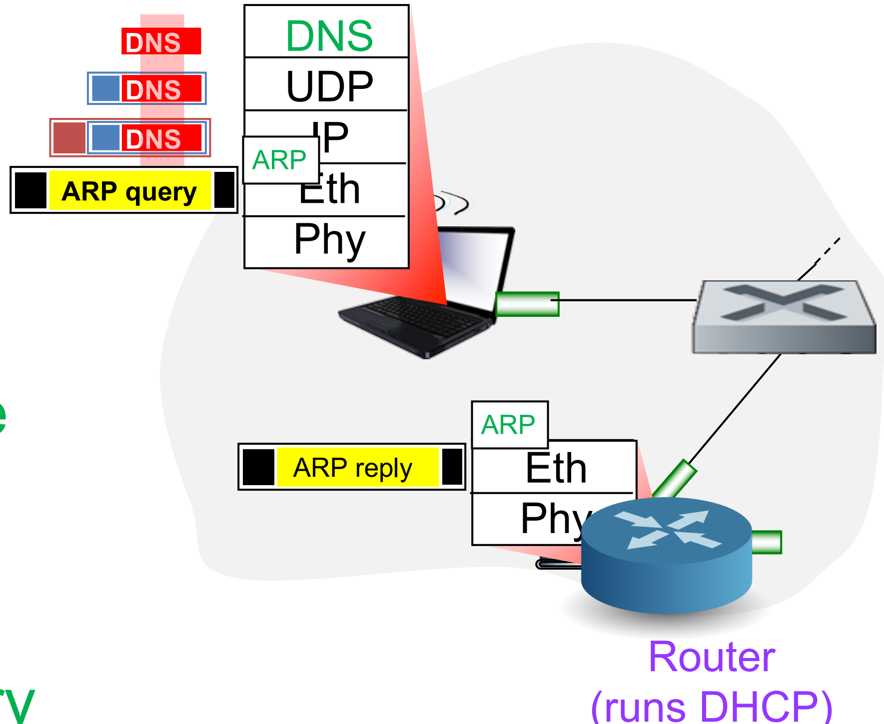
DHCP server sends DHCP ACK

- contains
  - IP addr assigned to client
  - subnet block (network mask)
  - IP addr of 1st-hop router
  - name & IP addr of DNS server
- encapsulate
  - in UDP, then IP, then Ethernet
- forward to client
  - through LAN via switch
  - (switch has learned where client is)

Client needs IP addr of www.google.com

How? DNS query created

- encapsulated in UDP
- encapsulated in IP
- encapsulated in Ethernet



But Client needs MAC address of router interface

- to send Ethernet frame
- broadcasts ARP query

Router receives ARP query

- sends ARP reply with MAC addr of router interface

Client gets MAC addr of 1st-hop router

- can now send frame containing DNS query

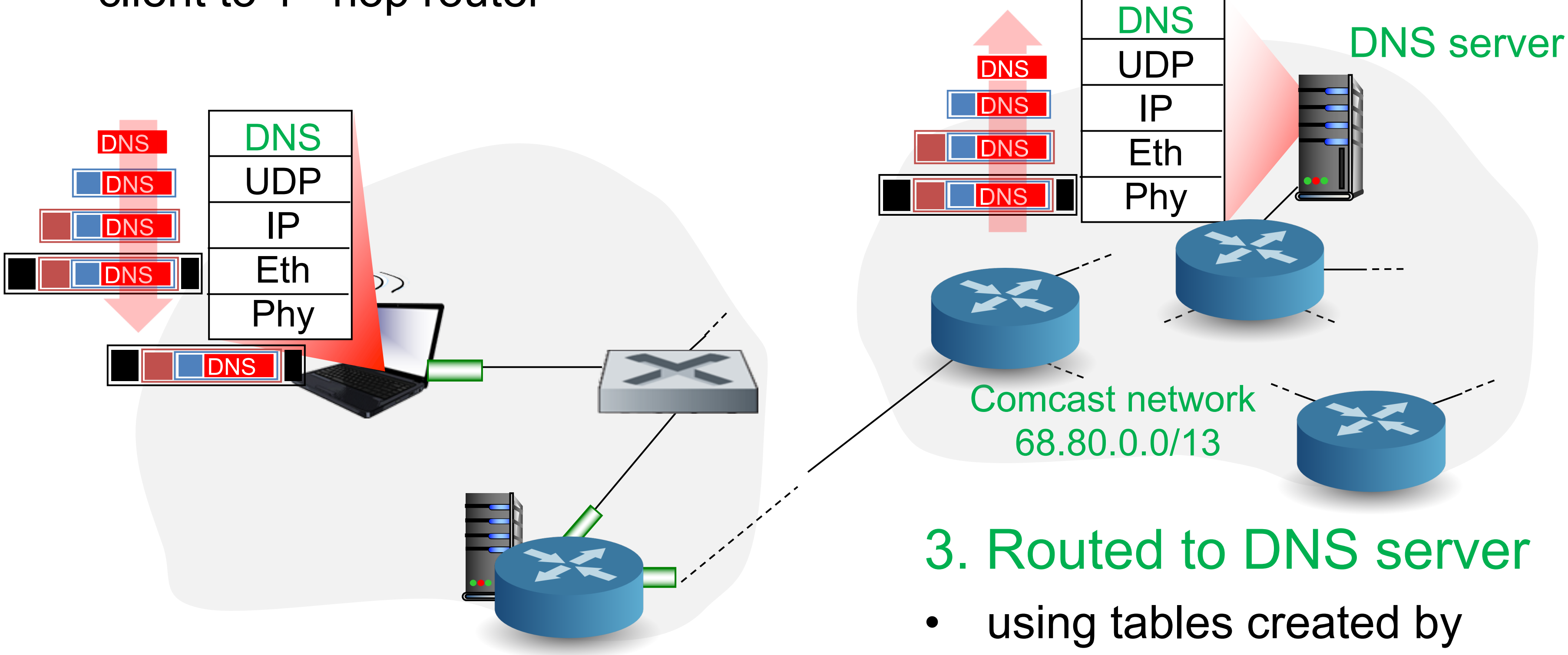
Client now needs IP addr of www.google.com

1. IP pkt containing DNS query

- forwarded via LAN switch from client to 1st hop router

4. DNS server replies to client

- with IP addr of www.google.com



3. Routed to DNS server

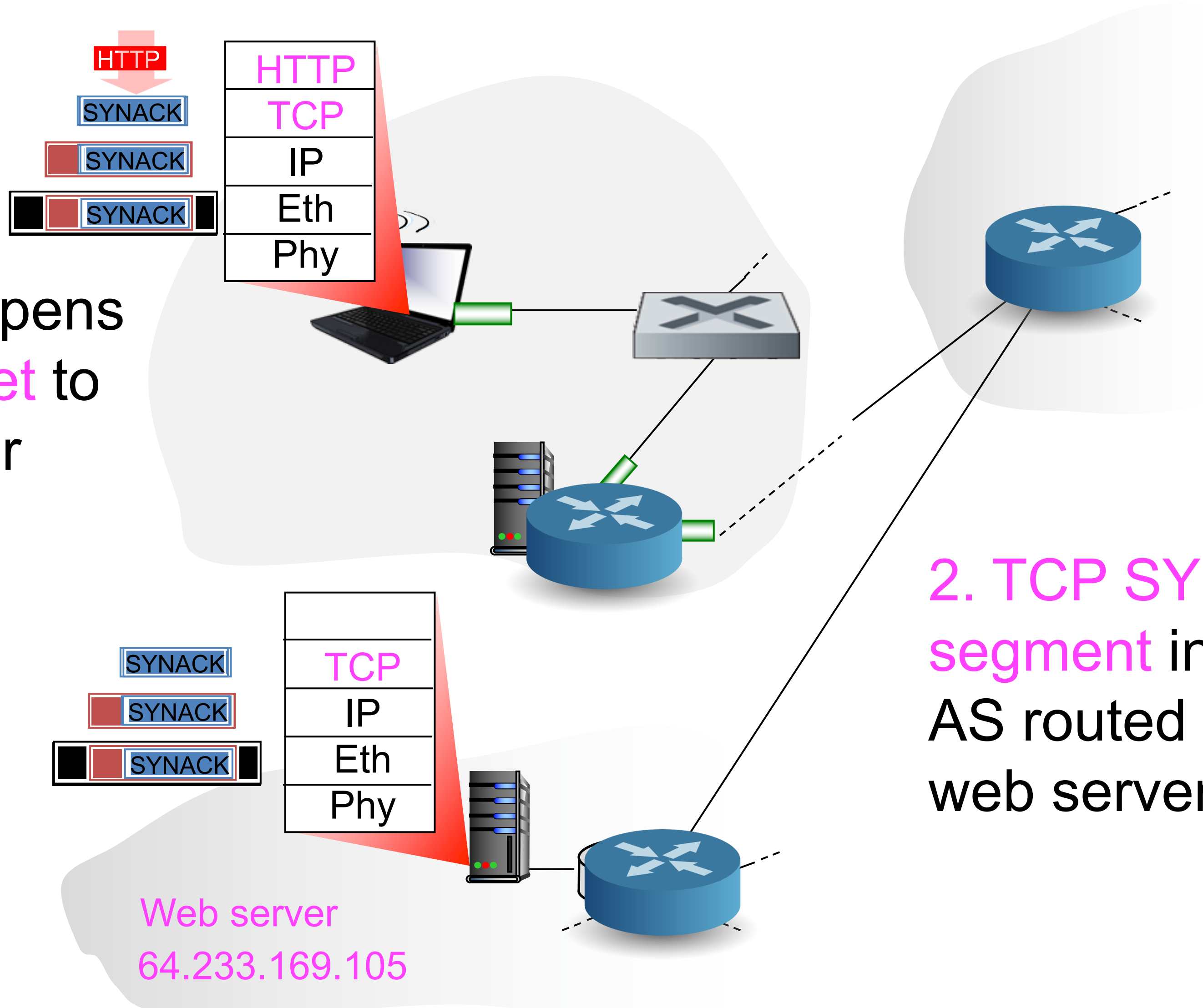
- using tables created by e.g., OSPF and BGP

2. IP pkt forwarded

- from campus network into Comcast network

Client opens TCP connection to carry HTTP

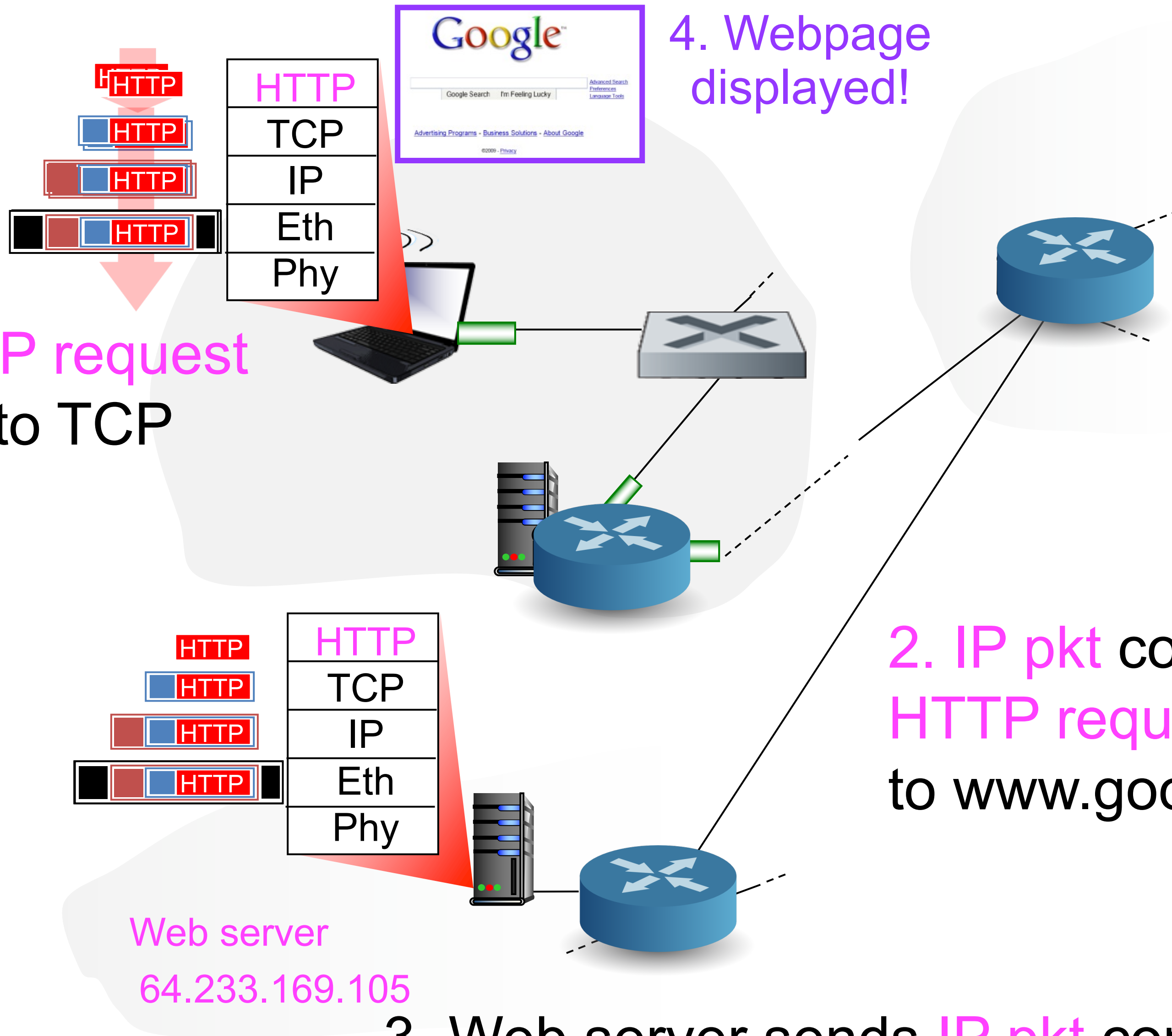
1. Client opens TCP socket to web server



2. TCP SYN segment inter-AS routed to web server

3. Web server responds with TCP SYNACK and client replies with TCP ACK. Connection established!

Client sends HTTP request and receives reply



1. HTTP request sent into TCP socket

2. IP pkt containing HTTP request routed to www.google.com

3. Web server sends IP pkt containing HTTP reply routed back to client

4. Webpage displayed!