How the Internet Works & Networking Security I

CMSC 23200, Spring 2026, Lecture 8

Grant Ho and David Cash

University of Chicago, 04/16/2026
Some slides adapted from Blase Ur, Peyrin Kao, and Zakir Durumeric

Logistics

- Assignment 3 (Crypto): **Due Tonight, 11:59pm**
- Assignment 4 (Public Key & TLS): **Released Friday**
- Discussion Section Next Week on TLS & Public Key (04/20)

- Midterm on May 5th: In-Class
- Final Exam time set: Wed, May 27 at 6pm
  One COMBINED for both sections

Outline

- Networking Background / How the Internet Works
  - The Internet & Networking Protocols
  - Protocol Layers and Addressing
  - Protocol Headers & Encapsulation
- Networking Threat Models
- ARP Security

The Internet

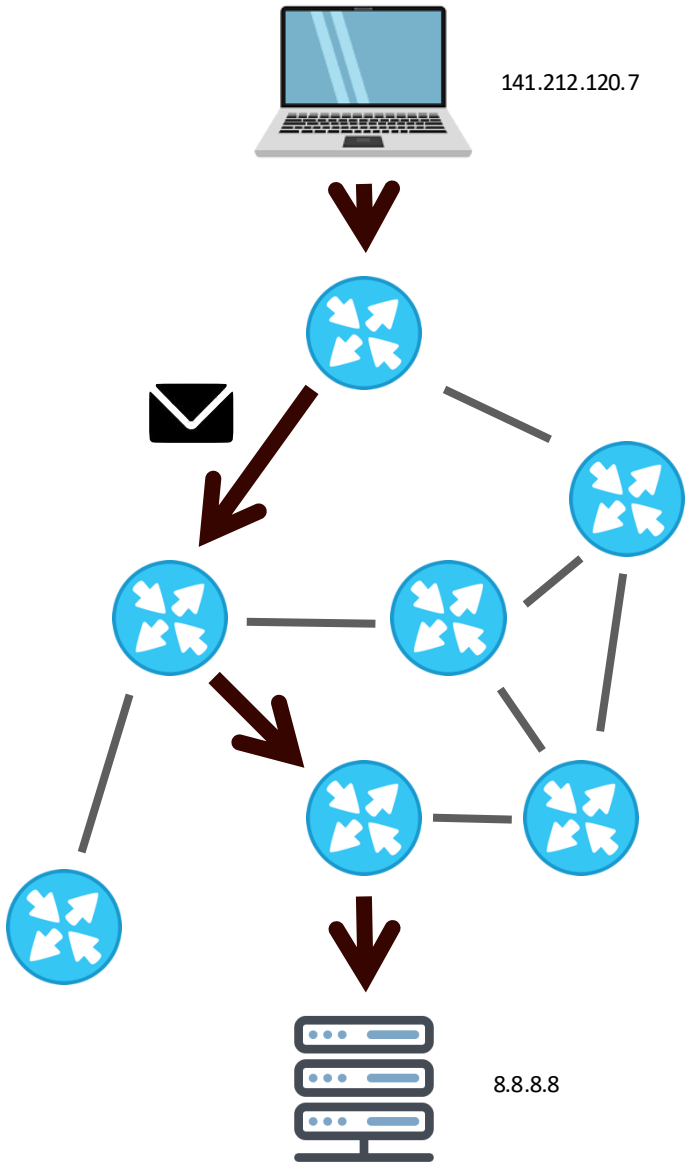
Global network that provides **best-effort** delivery of **packets** between connected hosts

Packet: a structured sequence of bytes

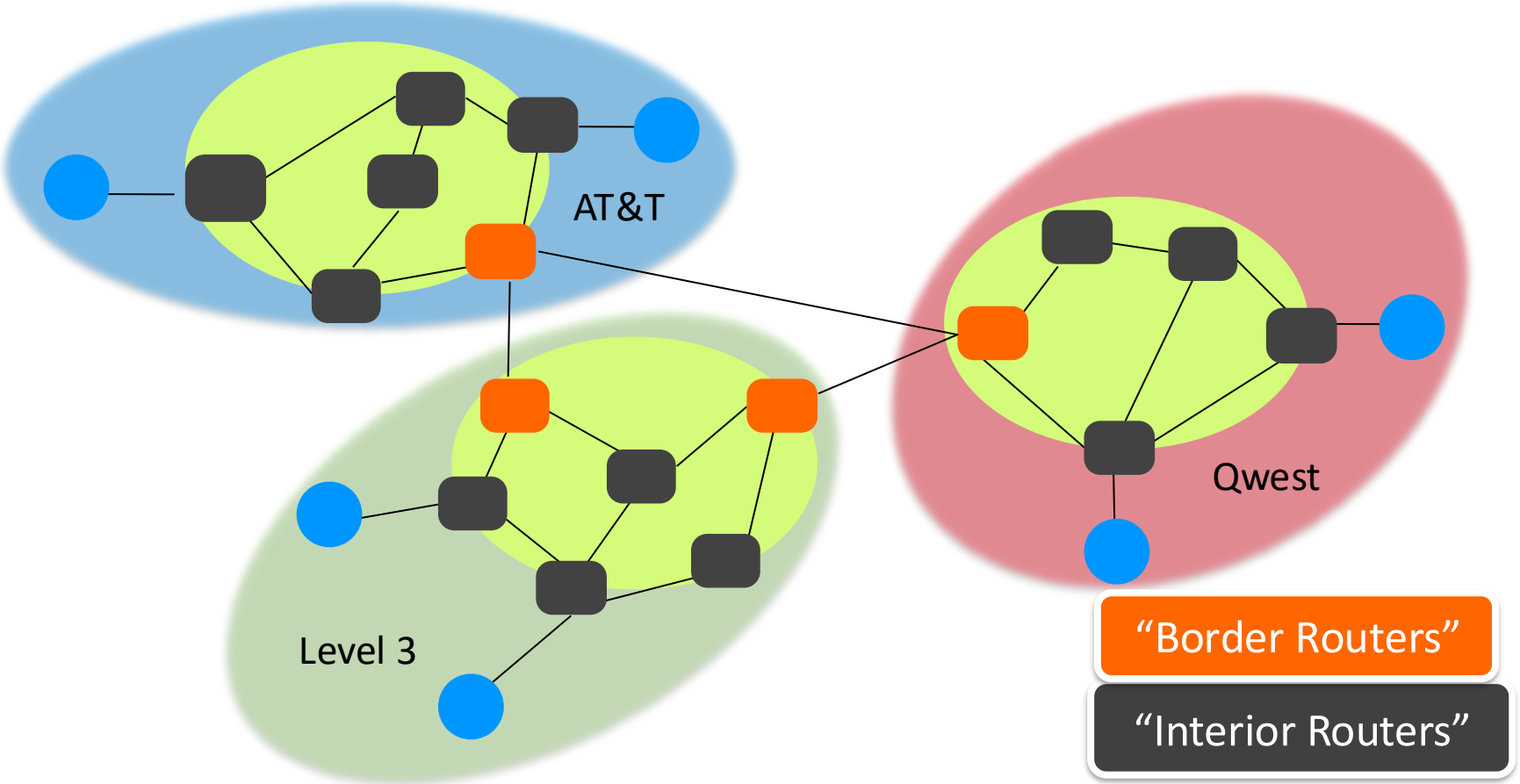
- Header: metadata used by network
- Payload: user data to be transported

Every host has a unique identifier — IP address

Series of routers receive packets,
look at destination address on the header,
and send it one hop towards the destination



The Internet From 10,000 Feet

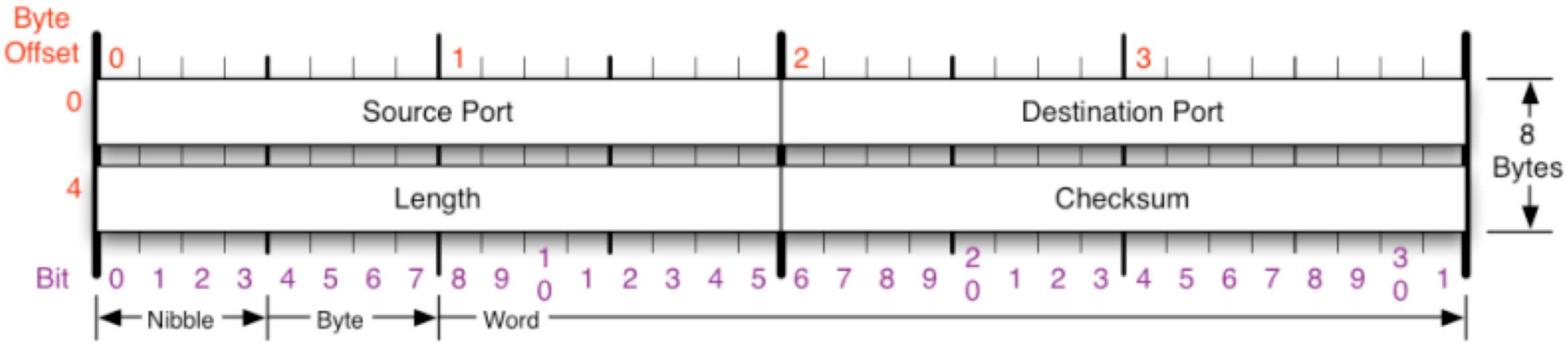


Network Protocols

Protocols: universal agreements that define how hosts communicate

Syntax: communication structured (e.g., msg format and order)

Semantics: meaning of communication (e.g., what actions taken on transmit or receipt of message, what assumptions can be made.)



Example: What bytes contain each field in a packet header

Layers (OSI Model)

Networks use a stack of protocol layers

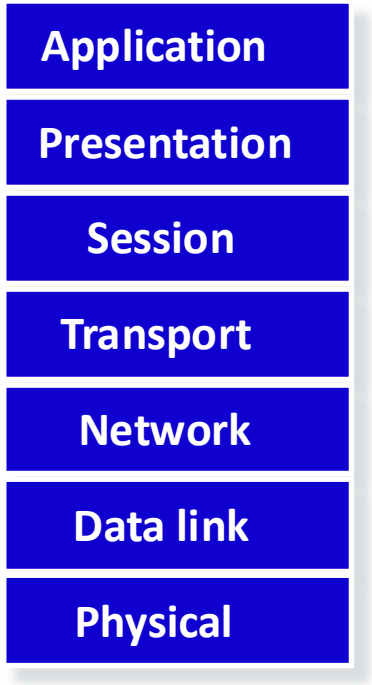
- Each layer has different responsibilities.

Lower layers provide services to layers above

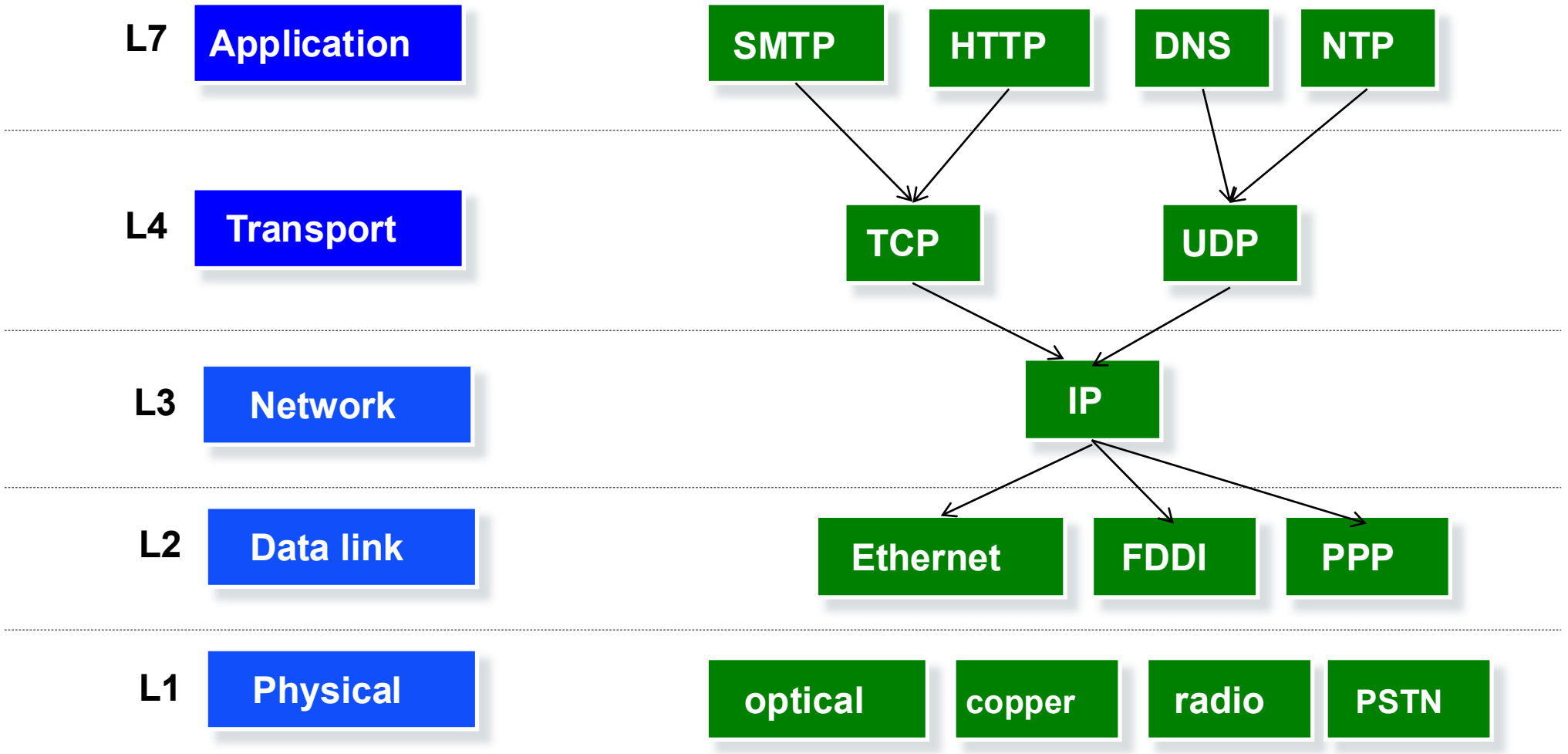
- Don’t care what higher layers do

Higher layers use services of layers below

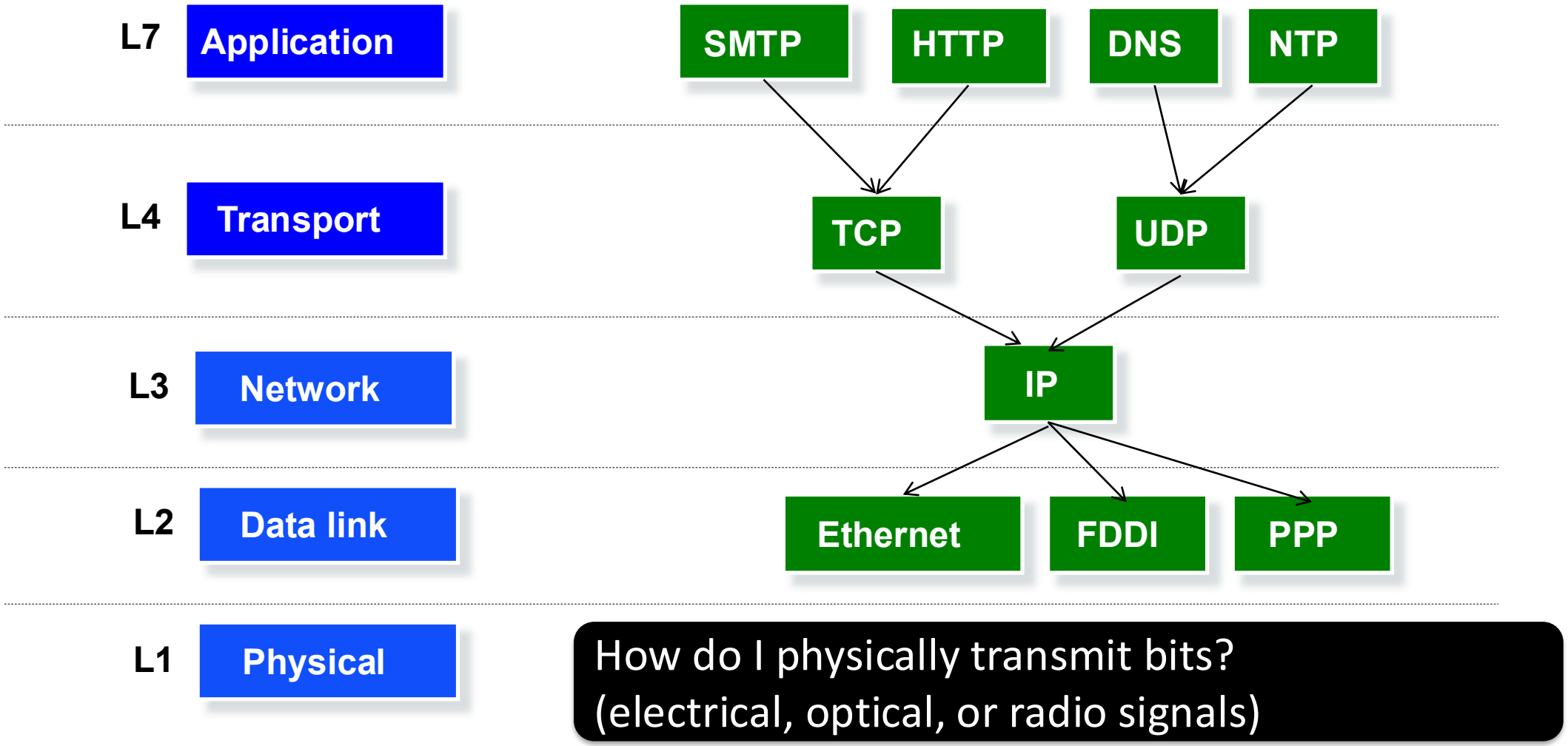
- Don’t worry about how it works



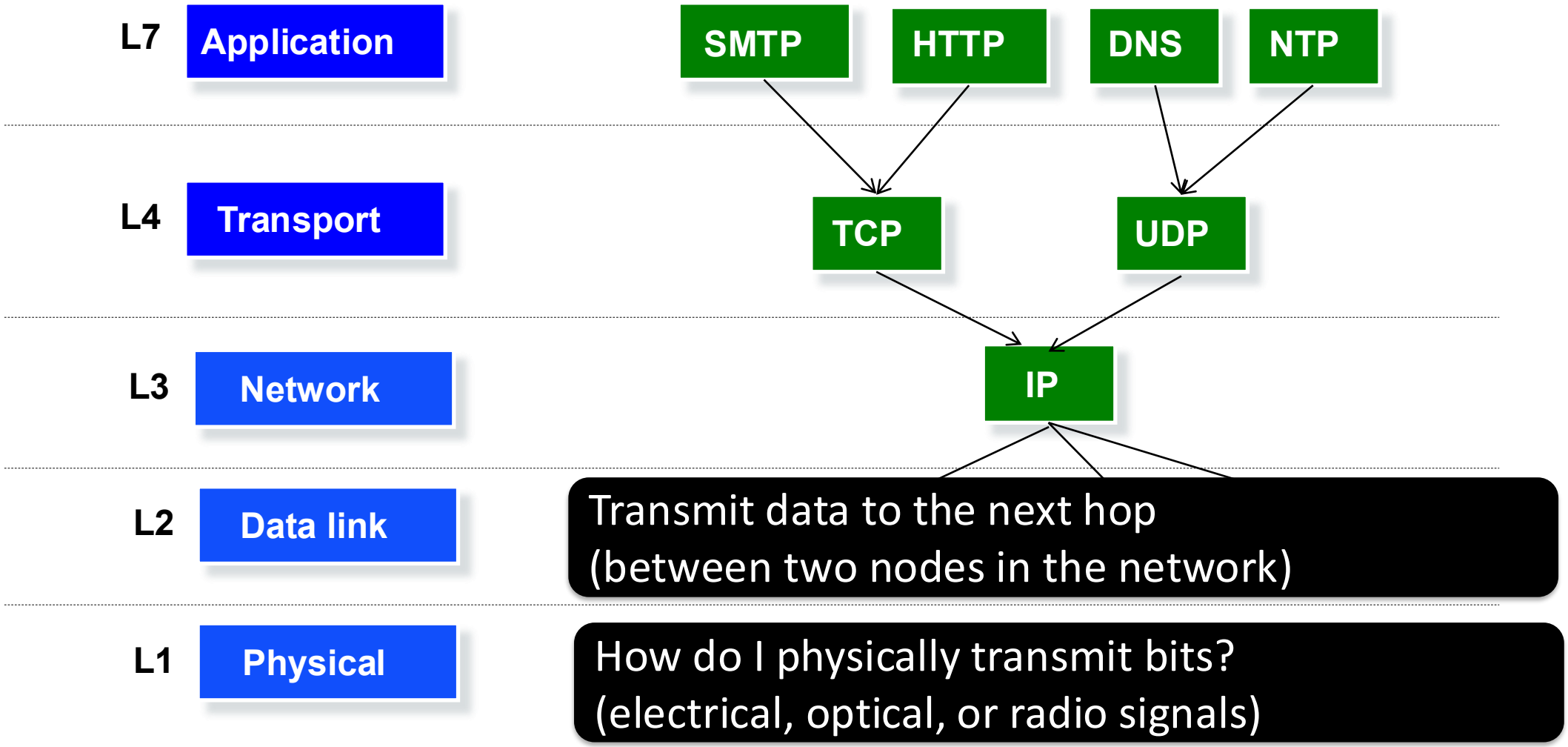
Protocols at Each Layer



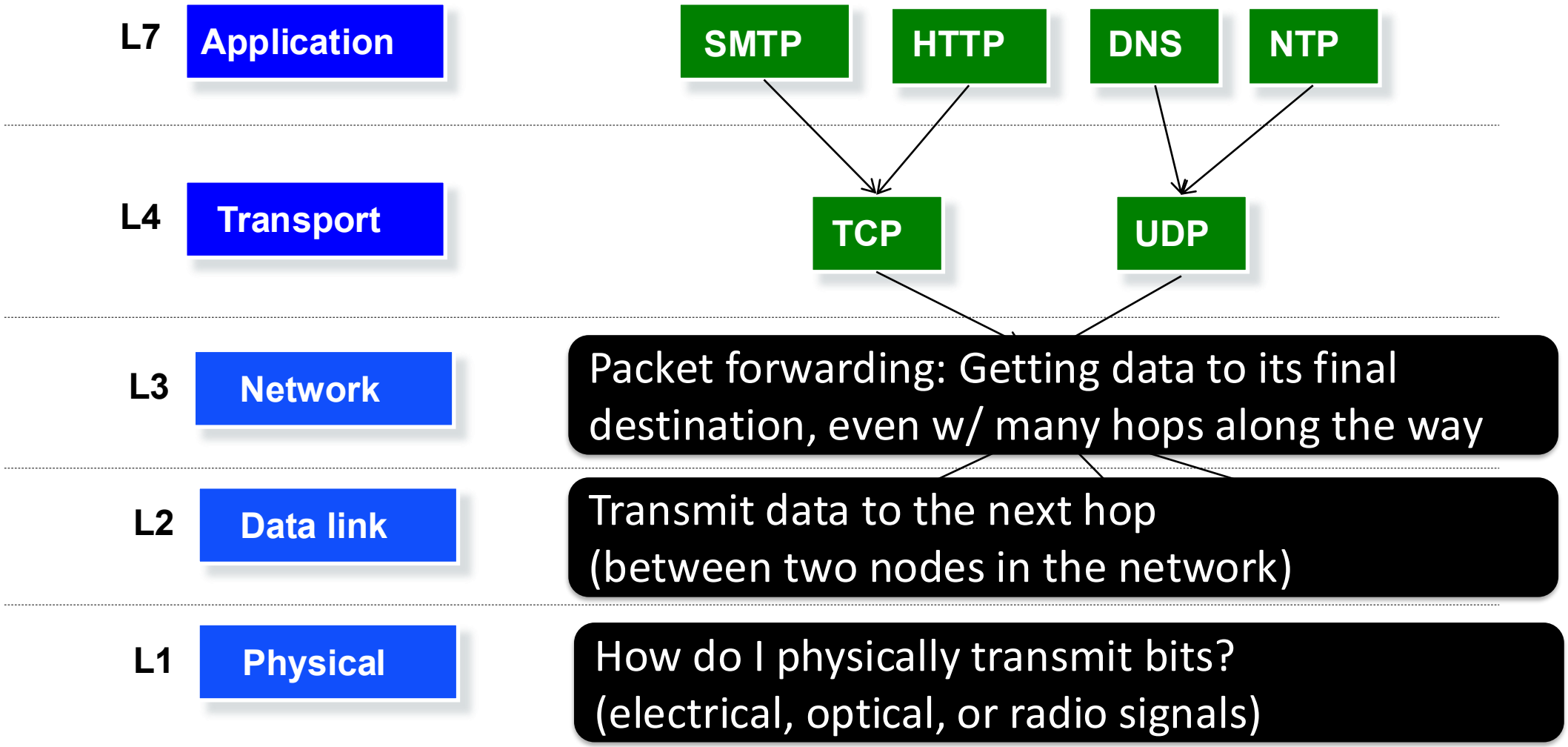
Goals at Each Layer



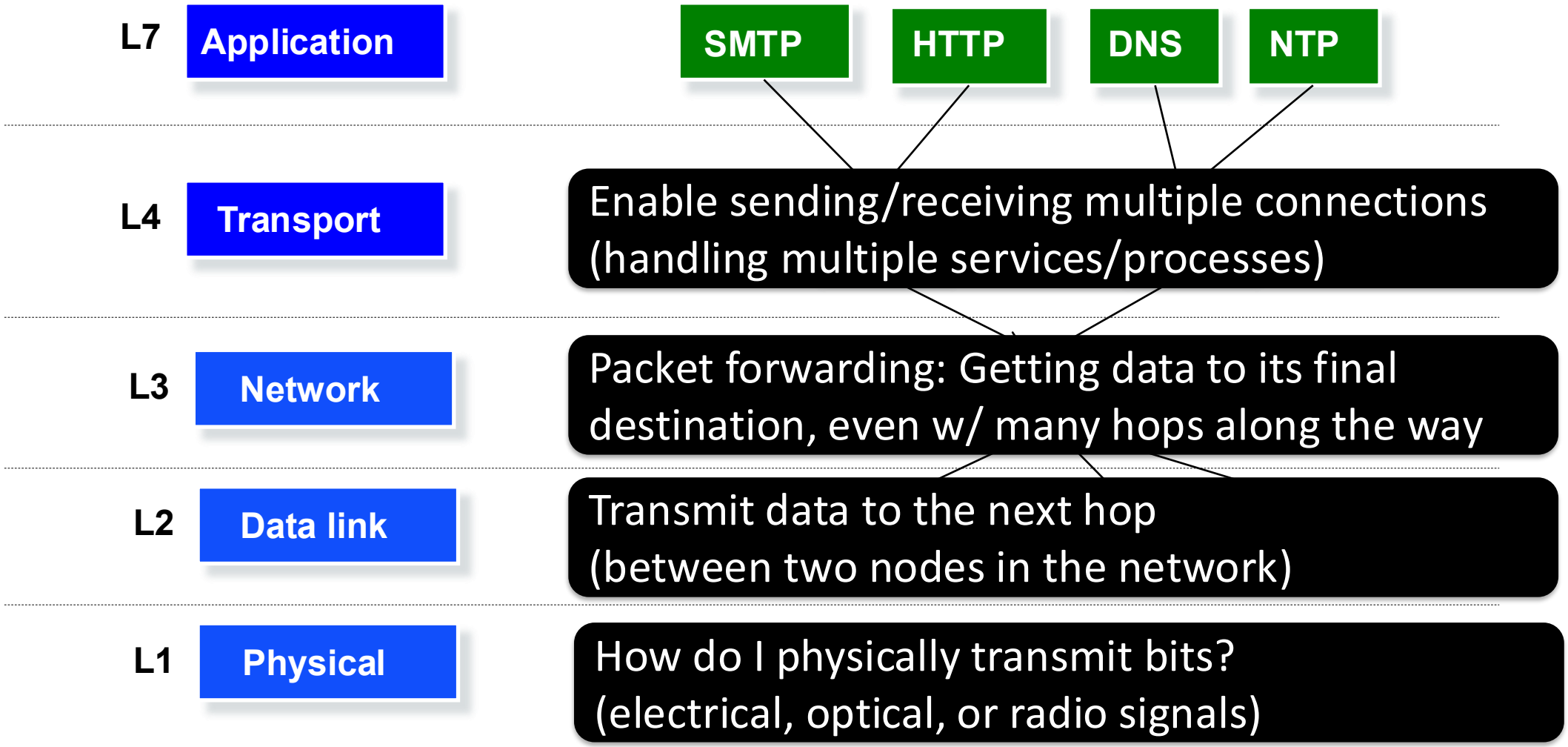
Goals at Each Layer



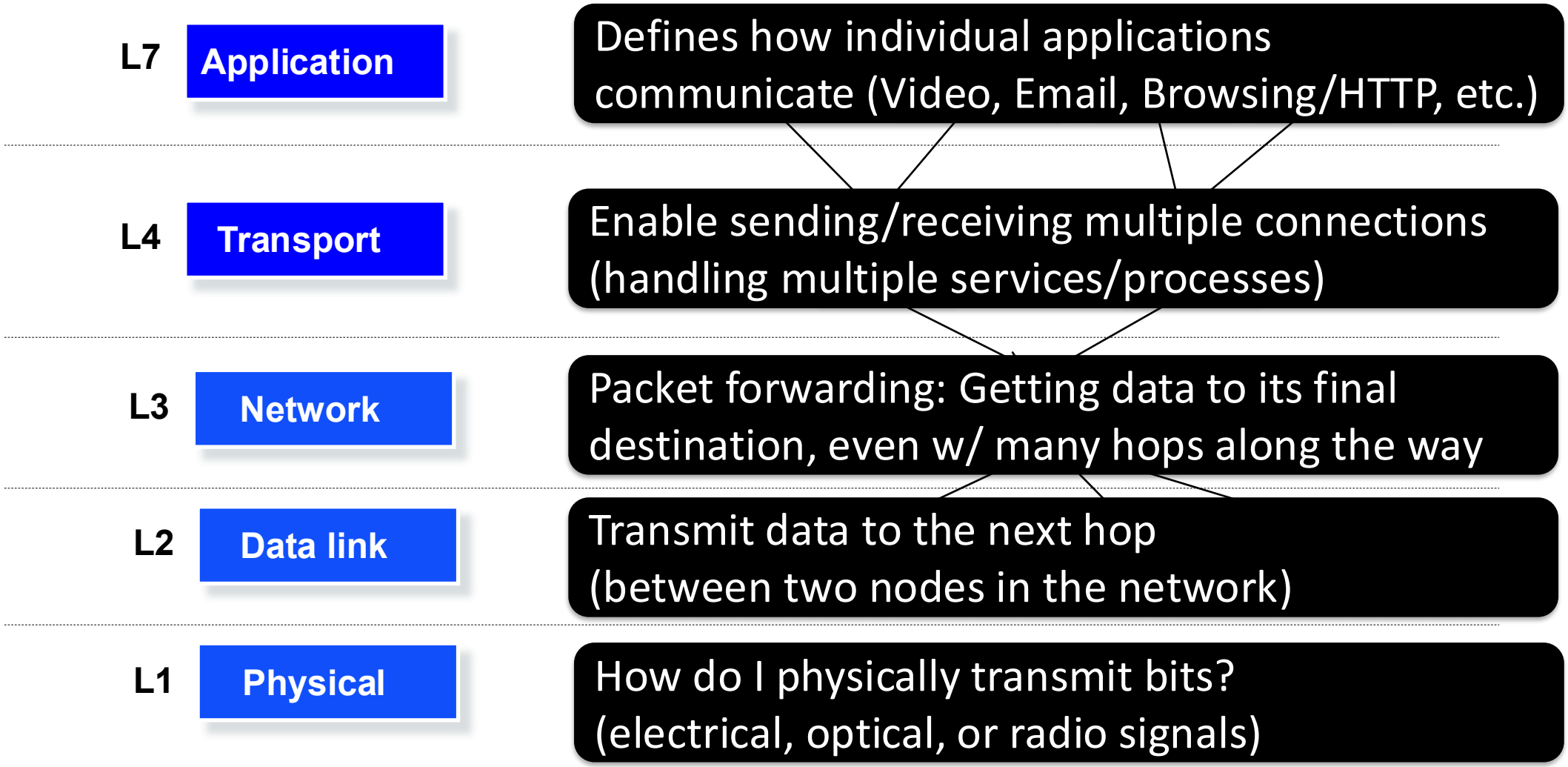
Goals at Each Layer



Goals at Each Layer



Goals at Each Layer



Goal: Be addressable on a local network
(to transmit between two local machines)

Solution: MAC Addresses (Link Layer)

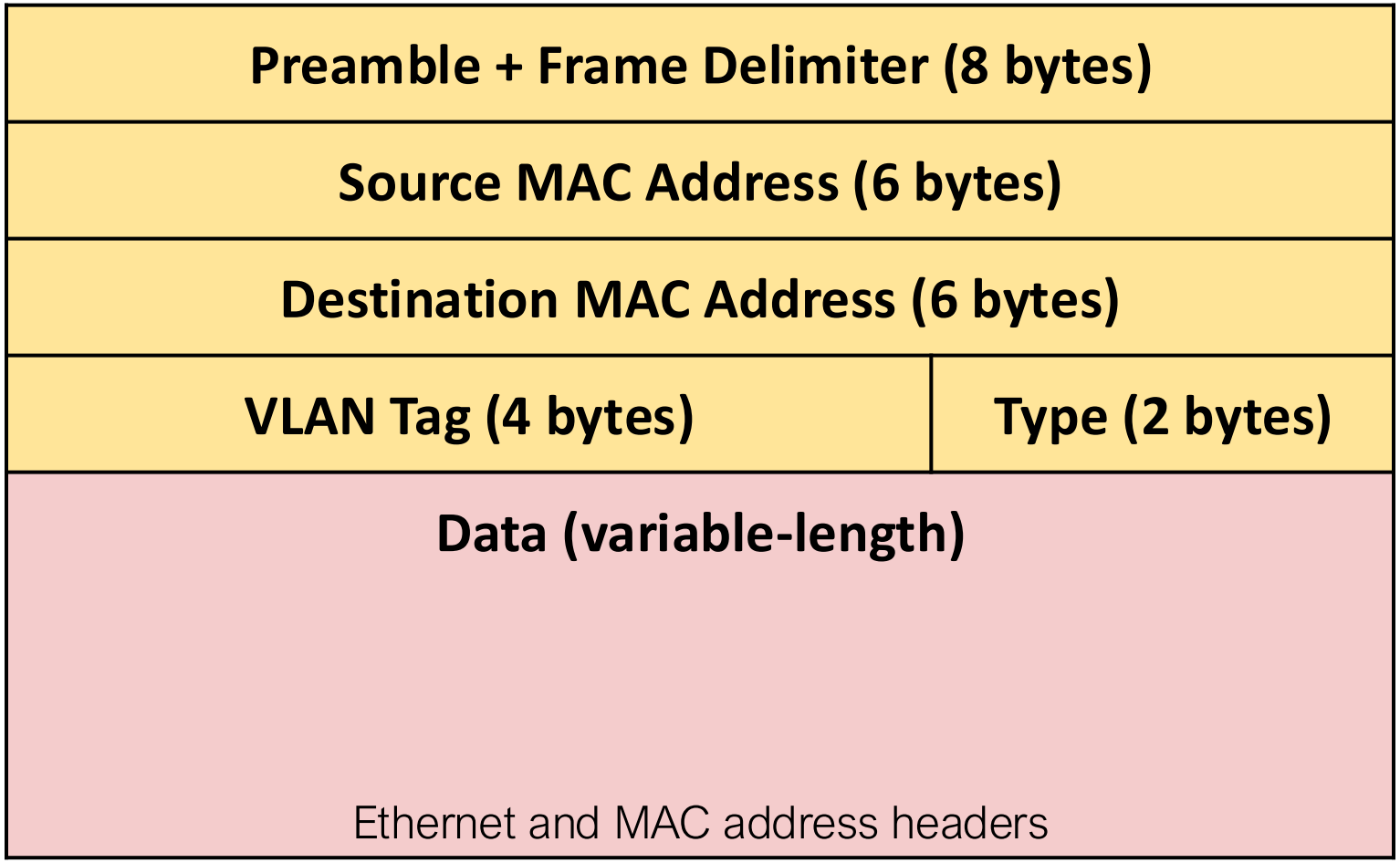
MAC (Media Access Control) Address

- Unique-*ish* 48-bit number associated with network interface controller (NIC): hardware to interact w/ network

12:34:56:78:9A:BC

- Usually assigned by manufacturers
  - In theory, doesn't ever change for a piece of hardware
  - In practice, MAC addresses can be spoofed
- See *ifconfig* and similar commands

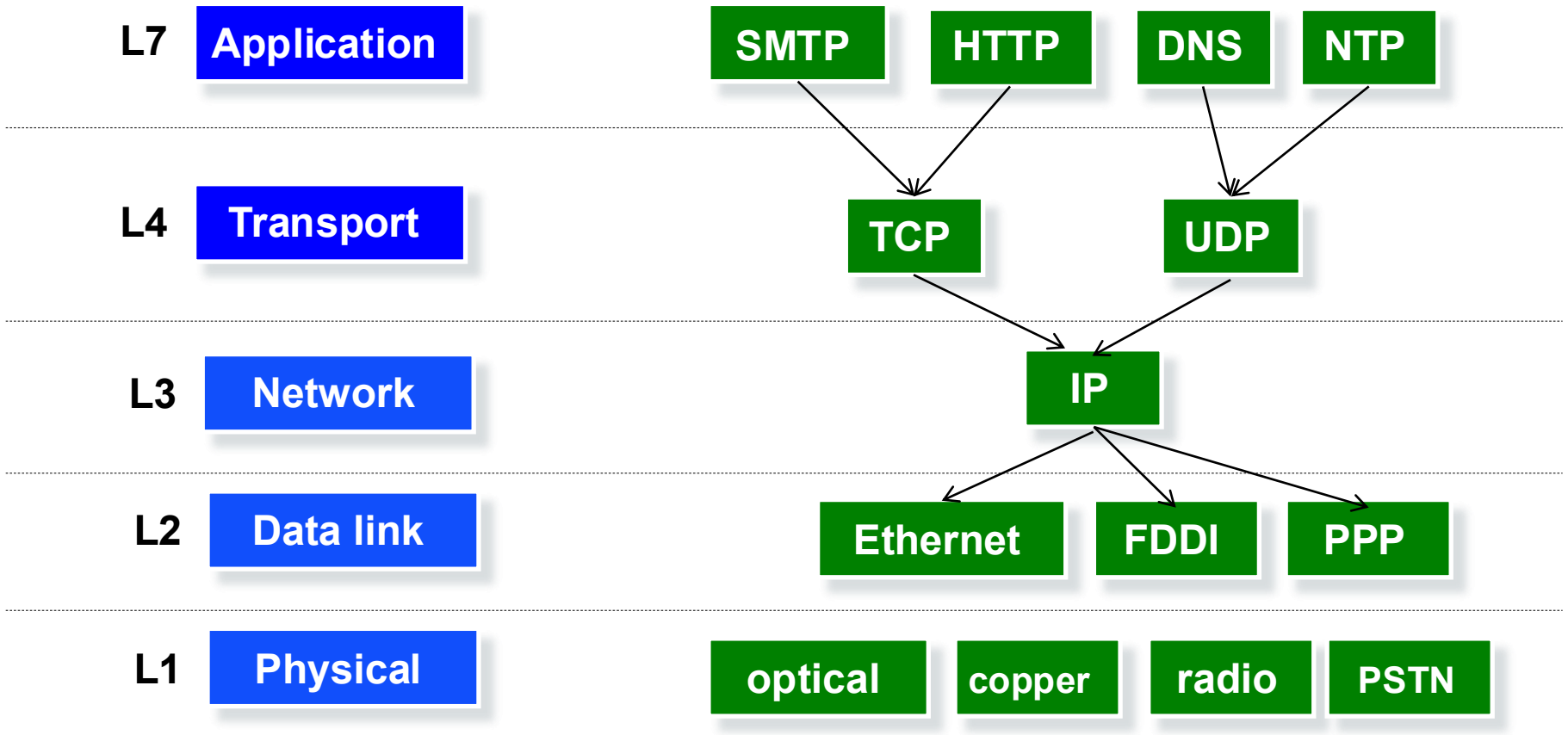
Layer 2 Header: Add Metadata to Every Message Frame



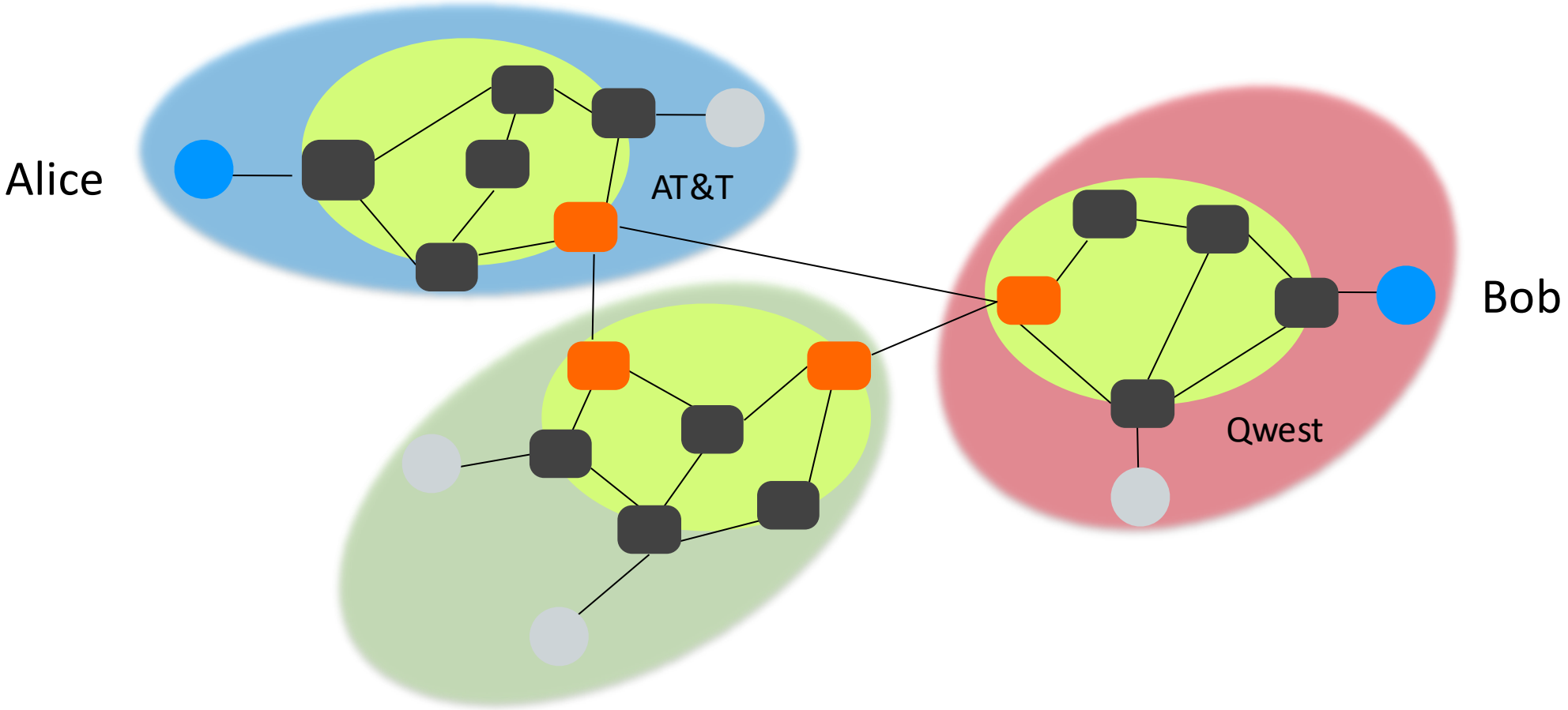
MAC Addresses Used on Link Layer

- Ethernet (plugged in)
  - Some hardware (e.g., hubs) repeats all traffic
  - Some hardware (e.g., switches) filters by MAC address
- Wi-Fi (802.11)
  - Your Wi-Fi card typically filters only unicast traffic for you and broadcast traffic (special MAC address: FF:FF:FF:FF:FF:FF)
  - Exception: promiscuous/monitor modes

Protocols at Different Layers



The Internet From 10,000 Feet



Goal: Be addressable on the Internet

Solution: IP Addresses (Network Layer)

Internet Protocol (IP)

Internet Protocol (IP) defines what packets that cross the Internet need to look like to be processed by routers to get from src -> final destination

Every host is assigned a unique identifier (“IP Address”)

Every packet has an IP header that indicates its sender and receiver

Routers forward packet along to *try* to get it to the destination host

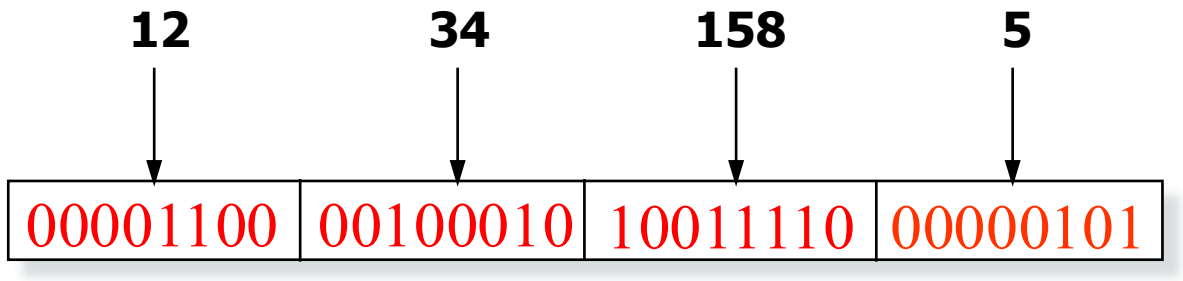
Rest of the packet should be ignored by the router

IP Addresses (IPv4)

- Unique-*ish* 32-bit number associated with host

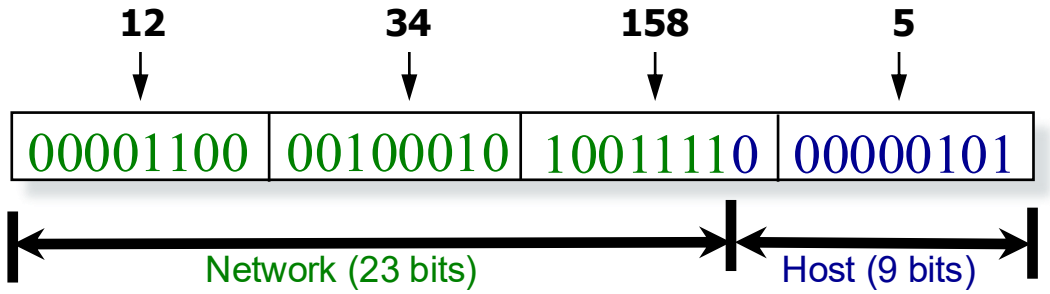
  00001100 00100010 10011110 00000101

- Represented with “dotted quad” notation
  - e.g., 12.34.158.5

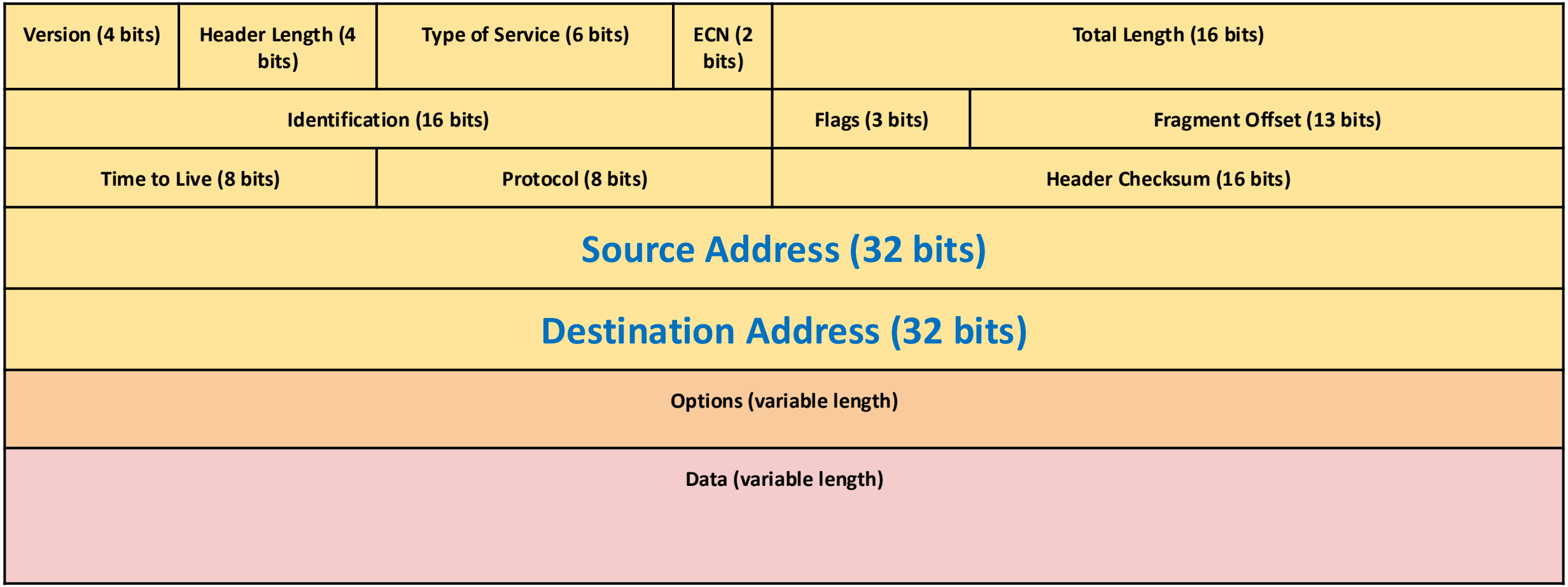


Hierarchy in IP Addressing

- 32 bits are partitioned into a prefix and suffix components
- Prefix is the network component; suffix is host component
- Interdomain routing operates on the network prefix

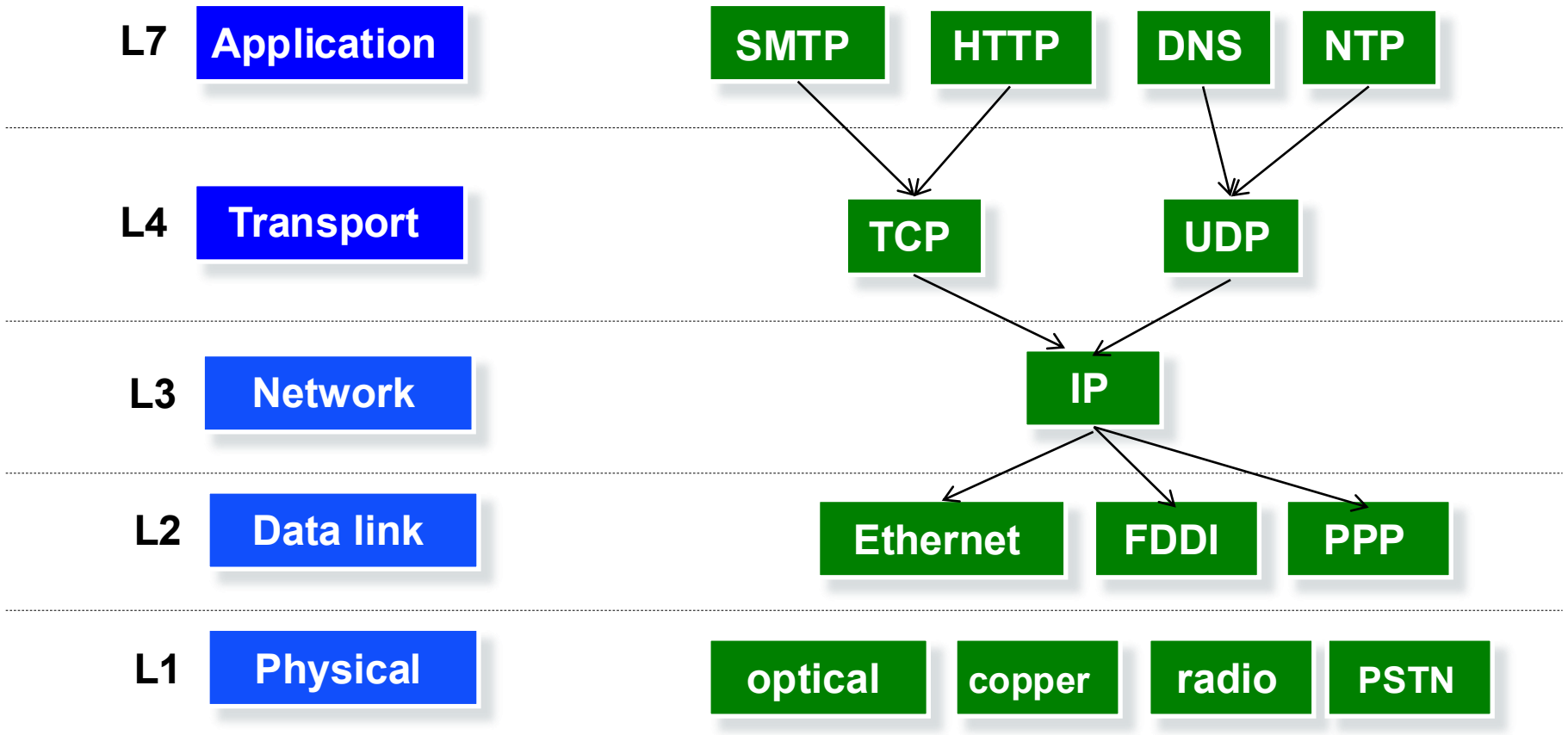


Layer 3 (IP) Header: Add Metadata to Every Packet



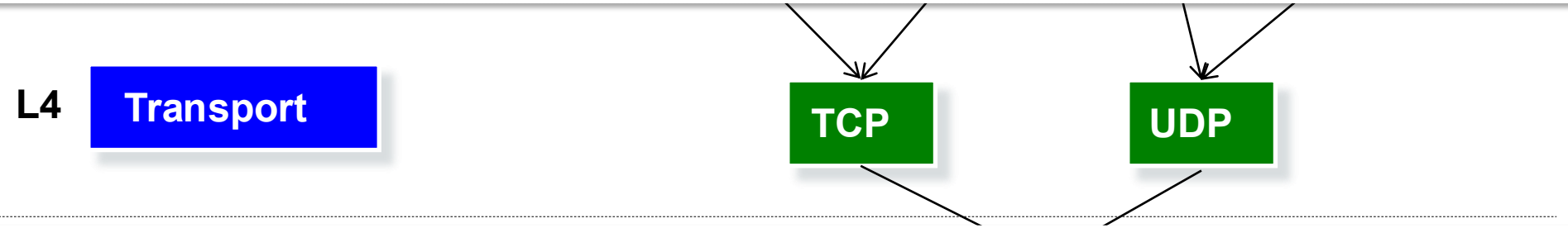
IPv4 header

Protocols at Different Layers



Goals:

1. Handle multiple connections streams &
2. Get ALL of the data to its destination



Solutions:

1. Transport layer protocols (ports)
2. TCP at the transport layer

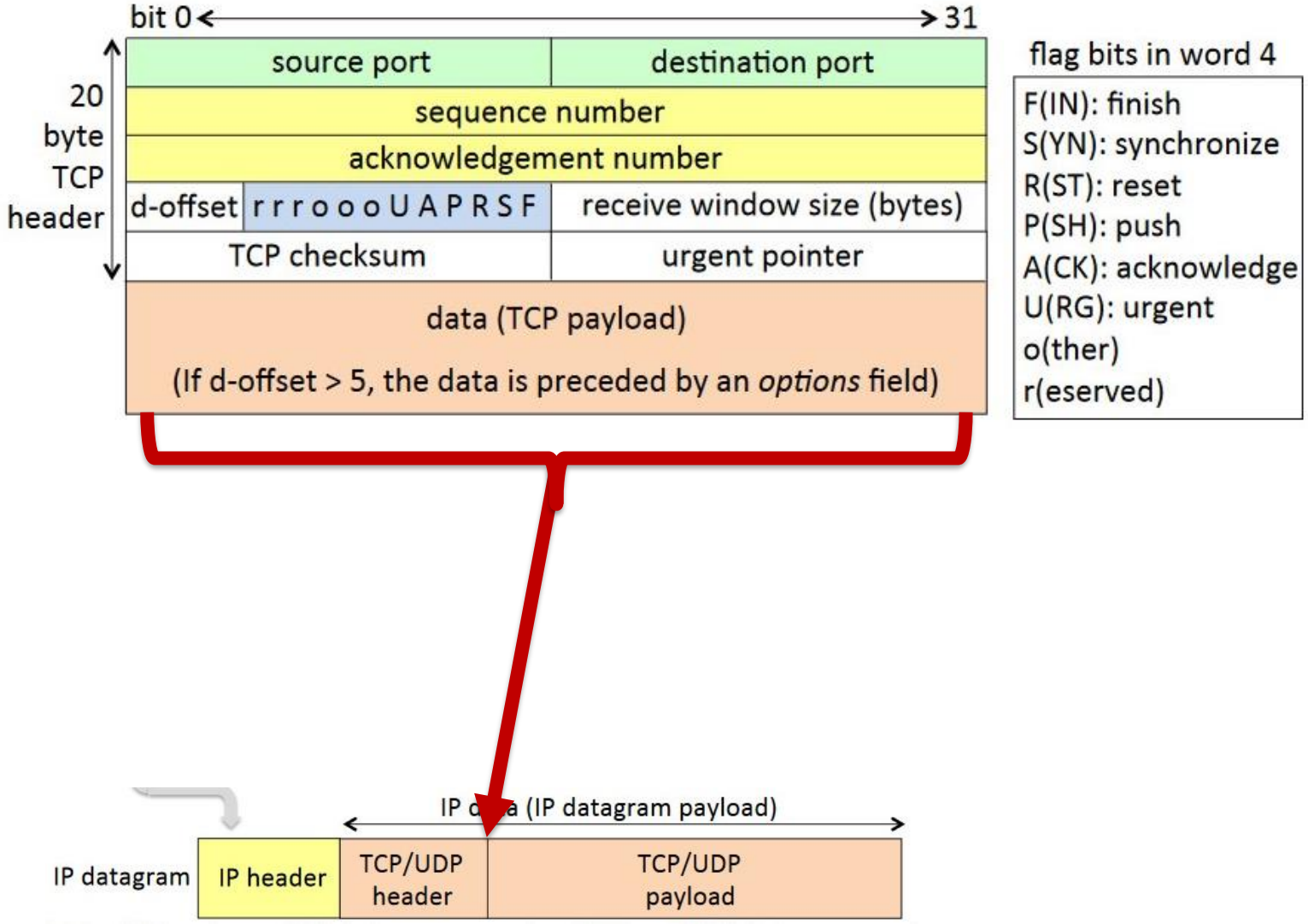
TCP (Transmission Control Protocol)

- Multiplexes between services
- Multi-packet connections
- Handles loss, duplication, & out-of-order delivery
  — all received data ACKnowledged
- Flow control
  — sender doesn't overwhelm recipient
- Congestion control
  — sender doesn't overwhelm network

Common TCP (Default) Ports

- 22: SSH
- 25: SMTP
- 53: DNS
- 67, 68: DHCP
- 80: HTTP
- 143: IMAP
- 443: HTTPS
- Ports 49152-65535 are used by client programs

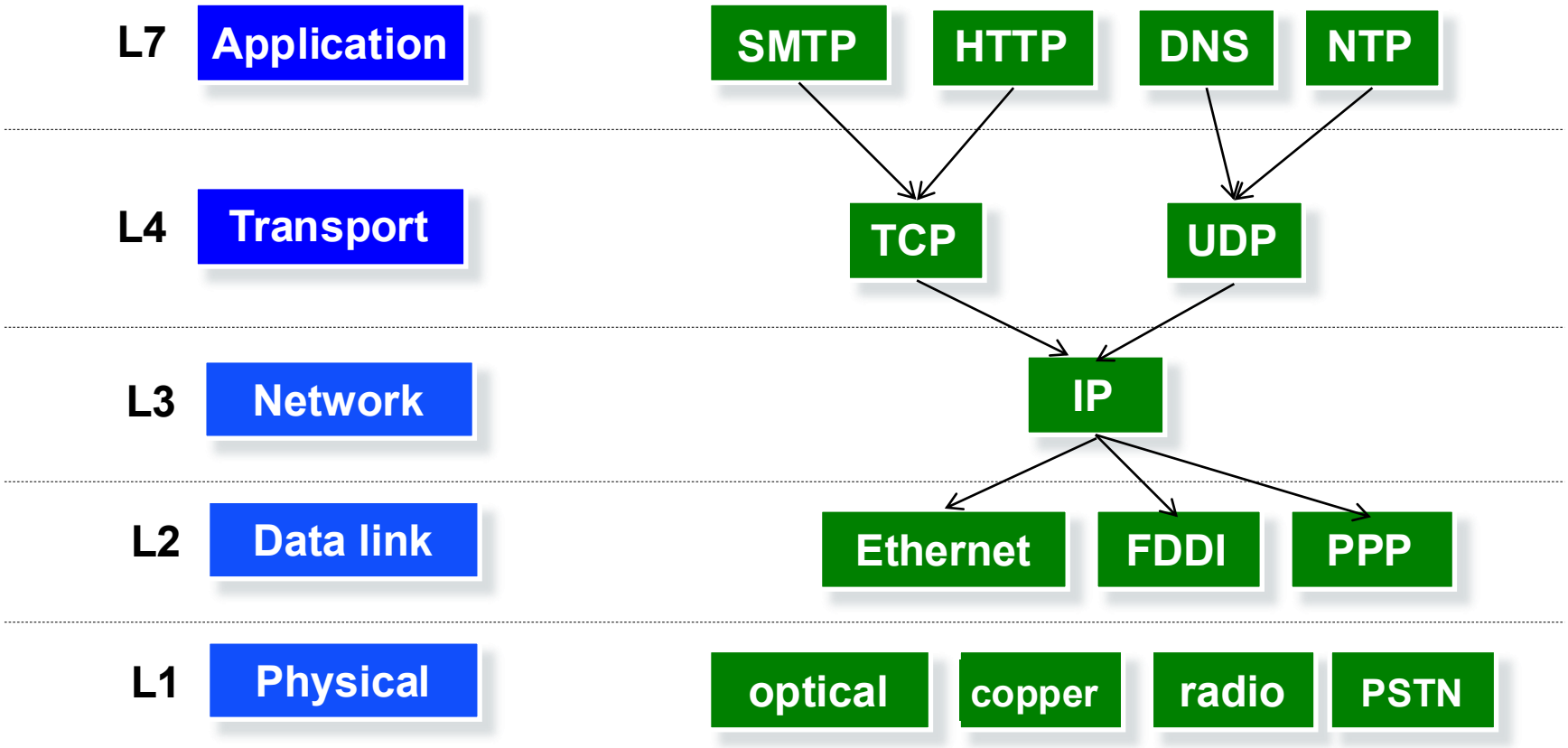
Layer 4 (TCP) Header: Add Metadata to Every Packet



Outline: Key Exchange & Networking Background

- Key Exchange Protocols (vs. Passive Attackers)
  - Hybrid Encryption Recap
  - Key Exchange w/ RSA & Diffie-Hellman
- Networking Background / How the Internet Works
  - The Internet & Networking Protocols
  - Protocol Layers and Addressing
  - Protocol Headers & Encapsulation

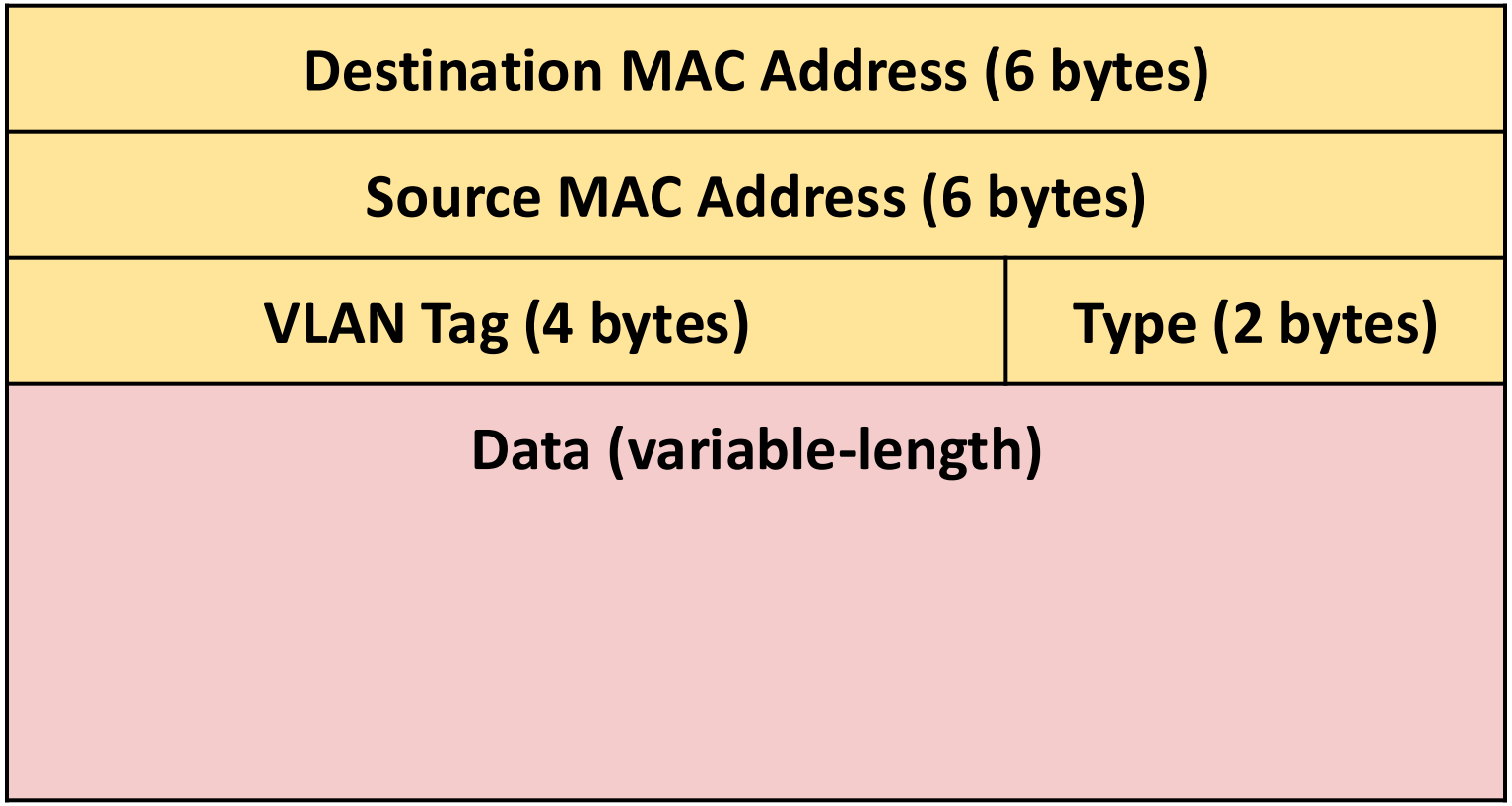
Layers of Abstraction and Headers



Lots of headers across all the layers

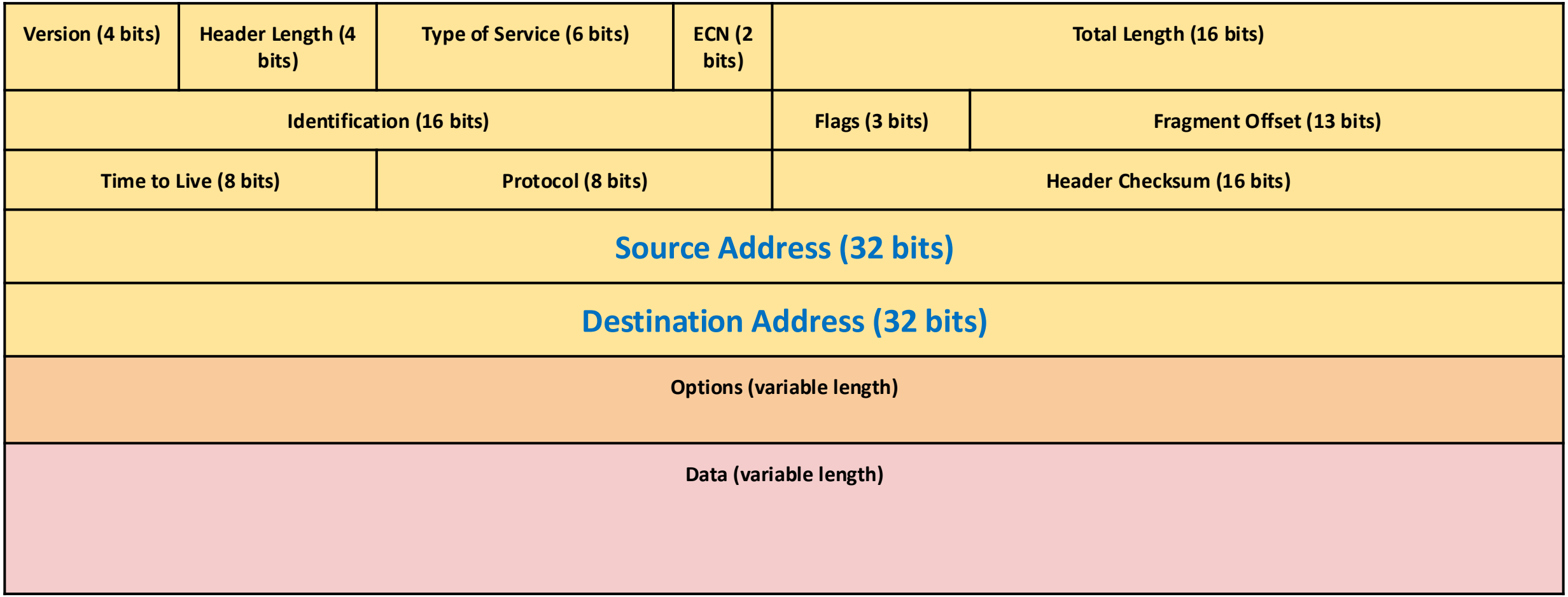
How do we combine & organize them when sending messages?

Layer 2 Header: Send Data b/t Two Hops



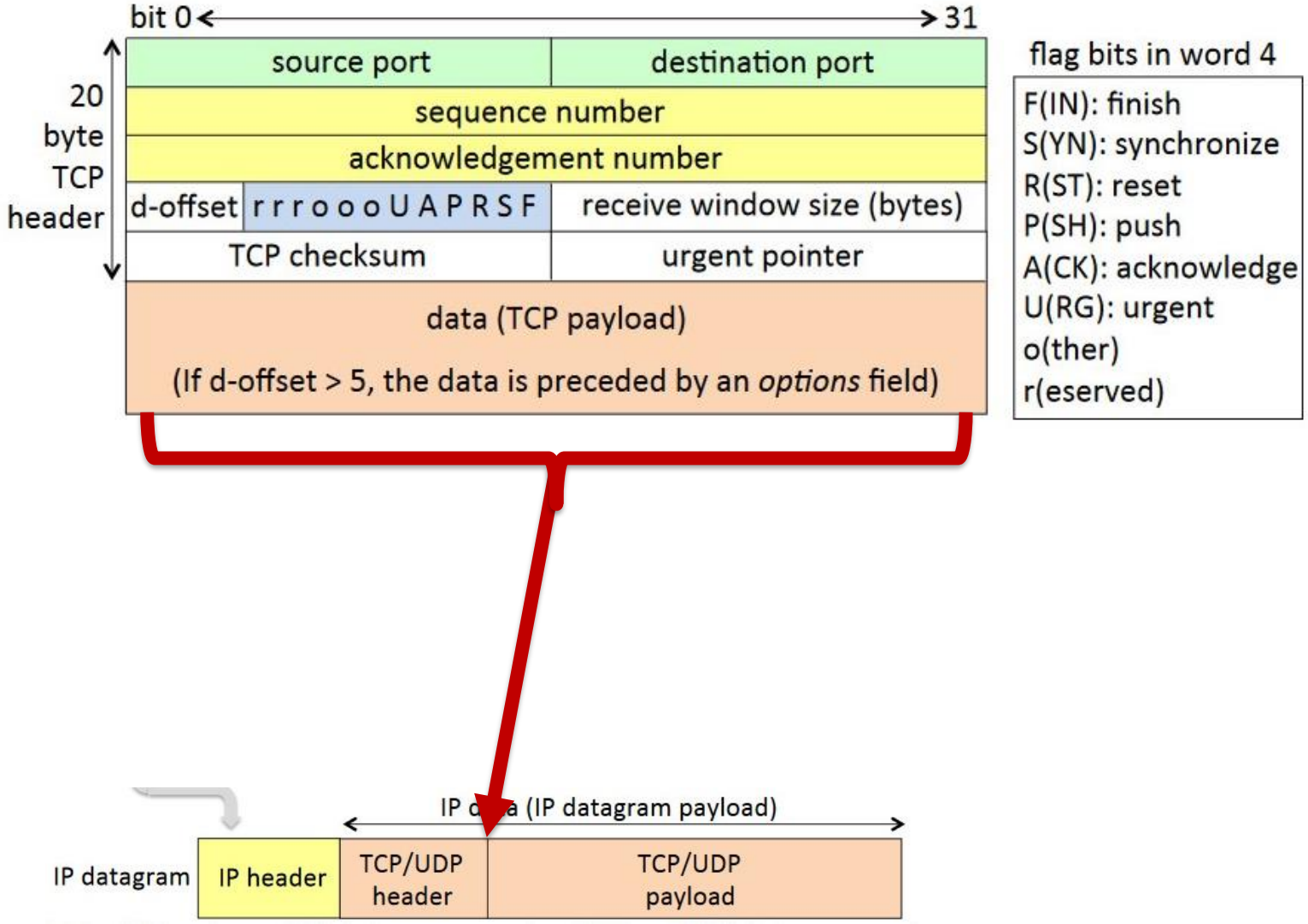
Ethernet and MAC address headers

Layer 3 Header (IP): Route Packet to Final Destination



IPv4 header

Layer 4 Header (TCP): Handle multiple connections



Outline

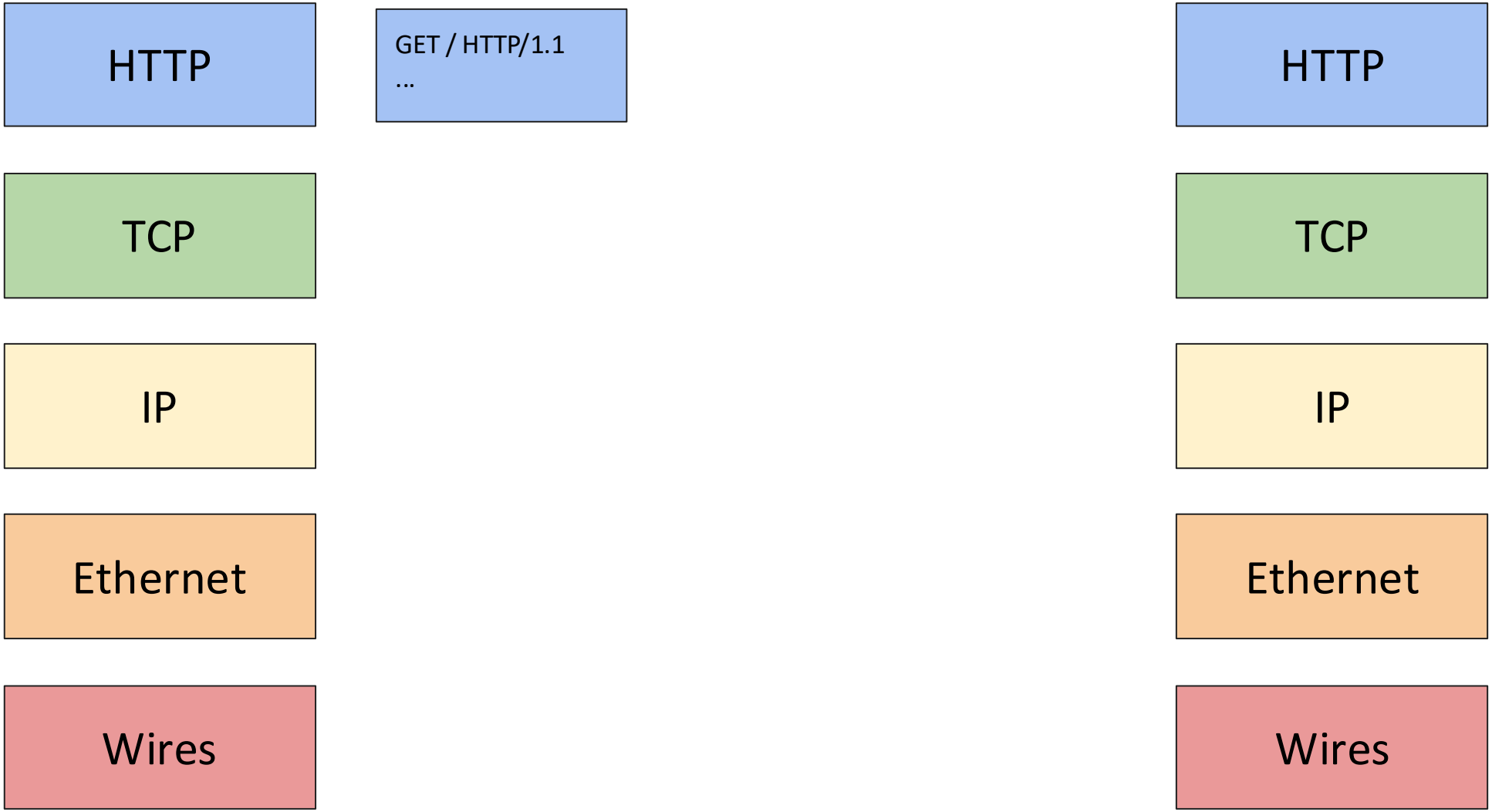
- Networking Background / How the Internet Works
  - The Internet & Networking Protocols
  - Protocol Layers and Addressing
  - Protocol Headers & Encapsulation
- Networking Threat Models
- ARP Security

Protocol Header Encapsulation

- The application starts with the content (payload) to send
- As the sender's machine constructs the packet to ***send*** to someone else (moving to lower layers), it wraps additional headers around the message

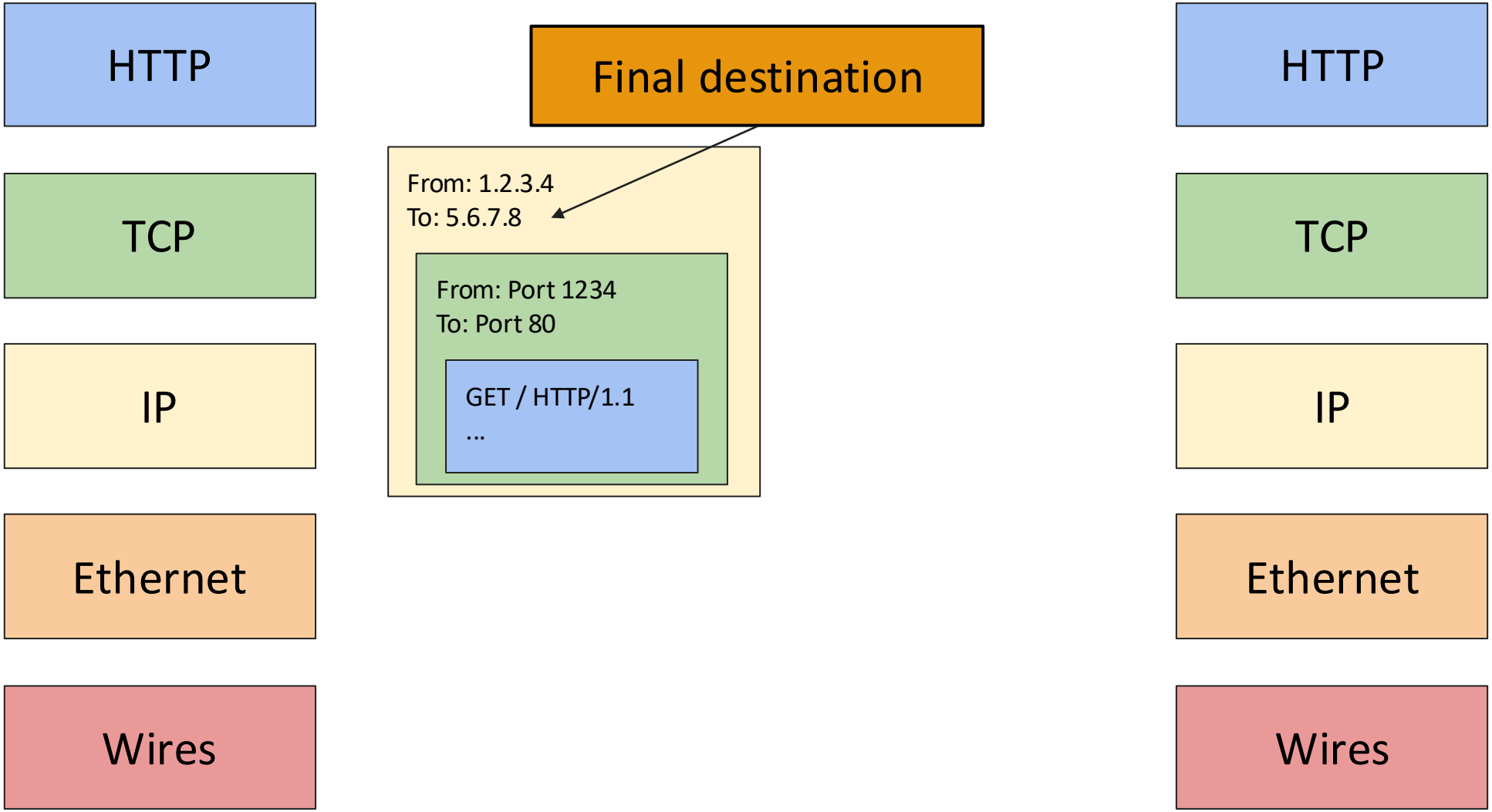
Example: HTTP Request

(from Peyrin Kao)

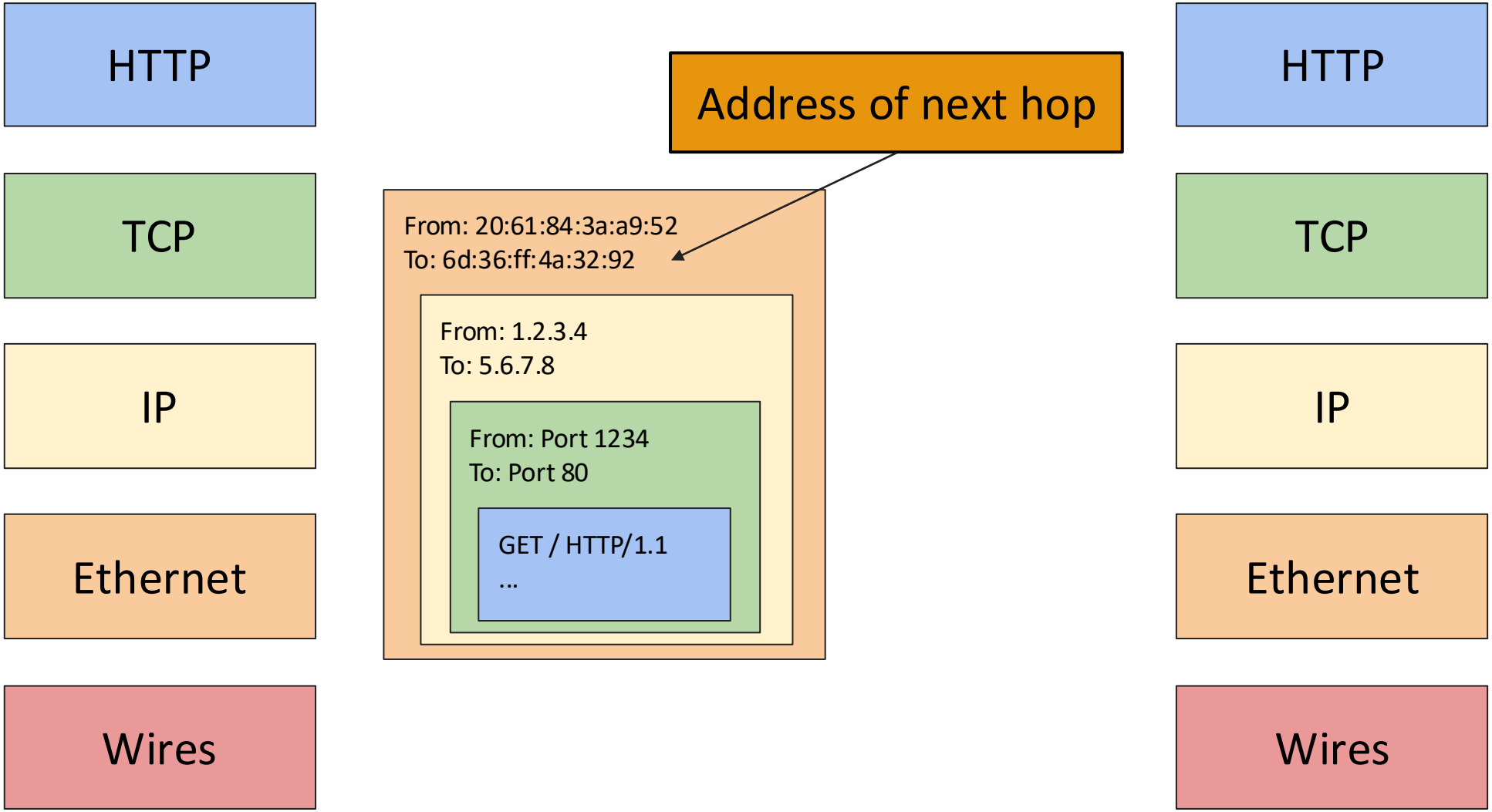


Example: HTTP Request

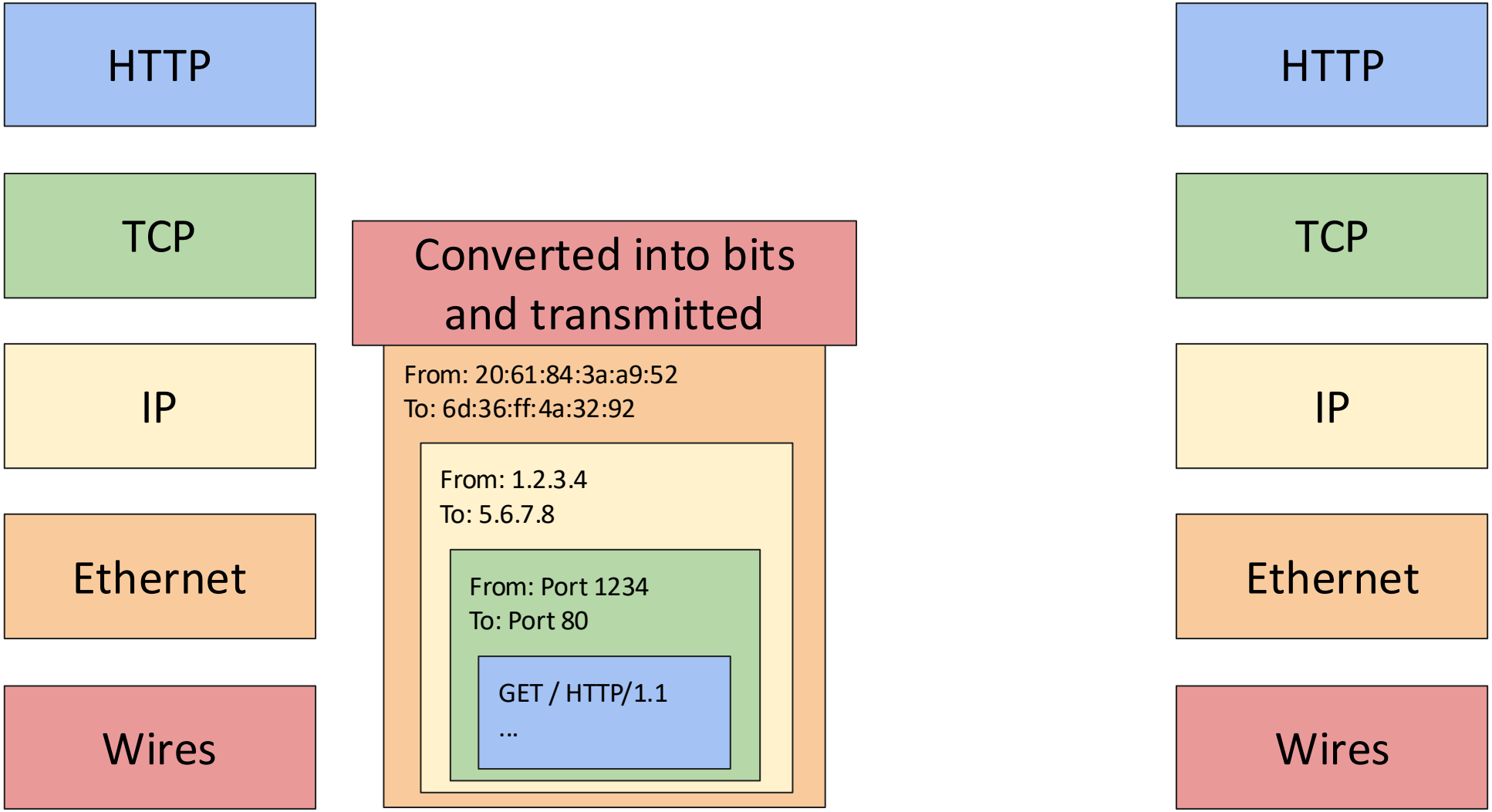
Example: HTTP Request



Example: HTTP Request



Example: HTTP Request

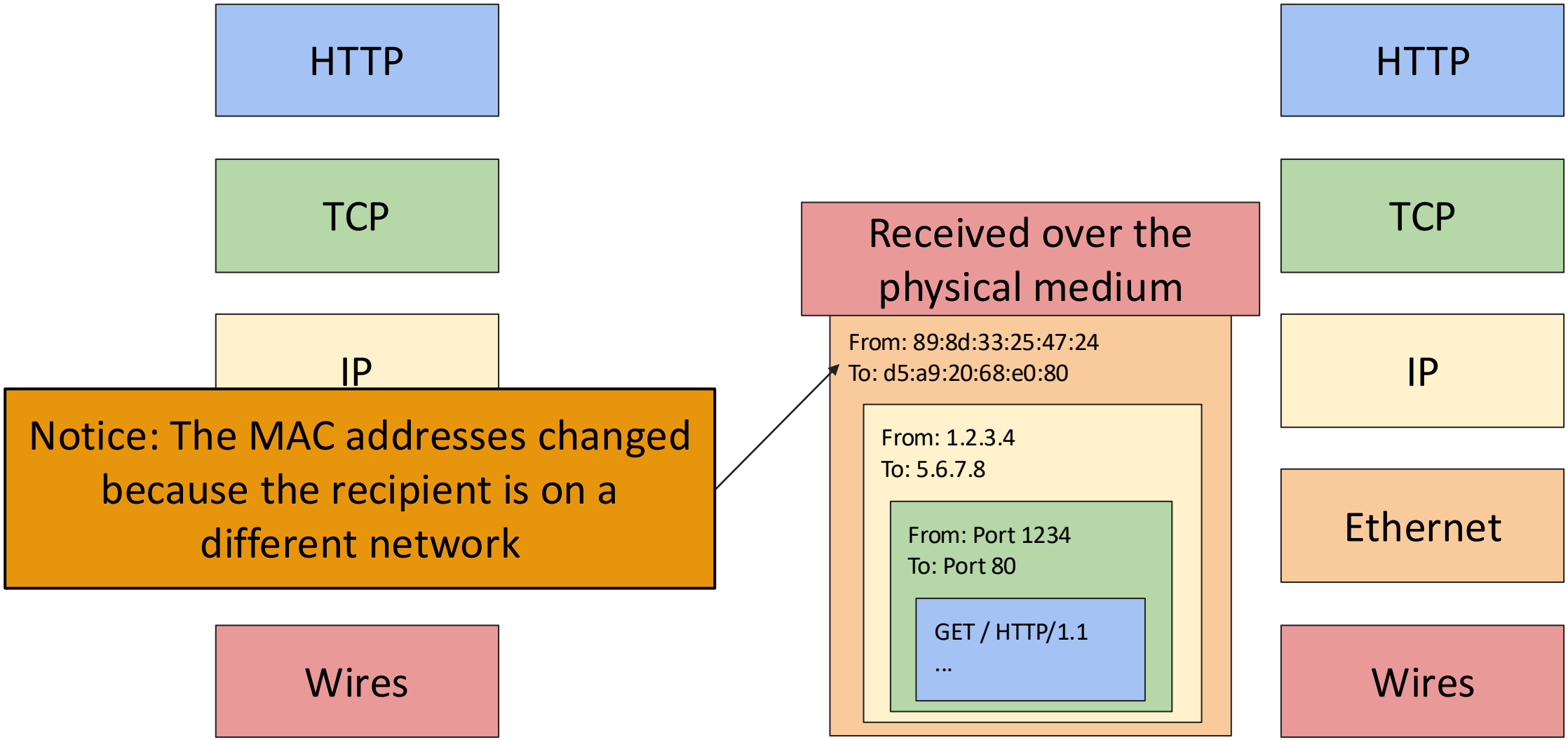


Layers of Abstraction and Headers

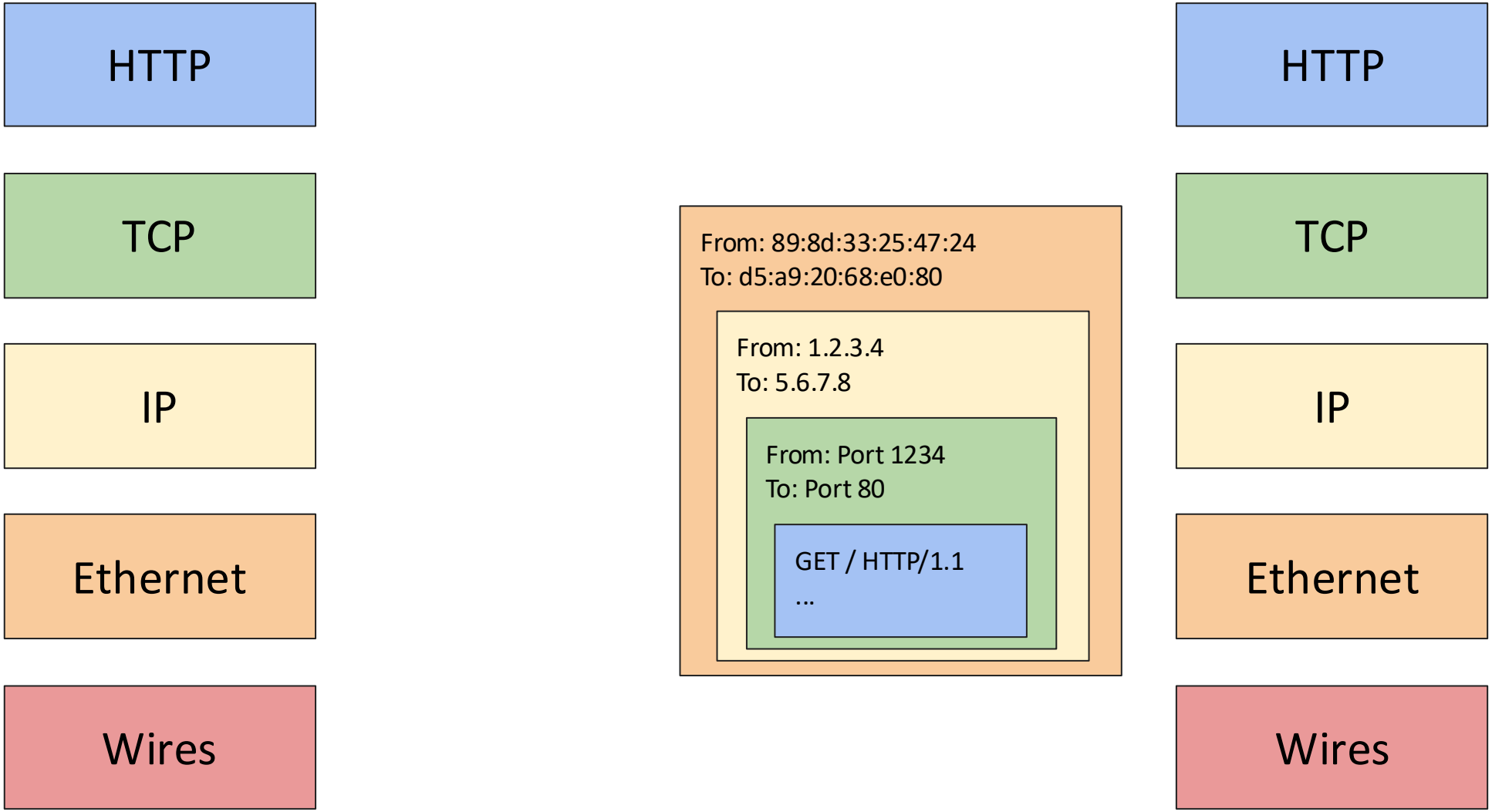
- The application starts with the content (payload) to send
- As the sender's machine constructs the packet to ***send*** to someone else (moving to lower layers), wrap additional headers around the message

- When machines receive a message, they peel off headers around the message to either:
  repackage & send to next hop (intermediate hop) or
  fully open & process (final destination)

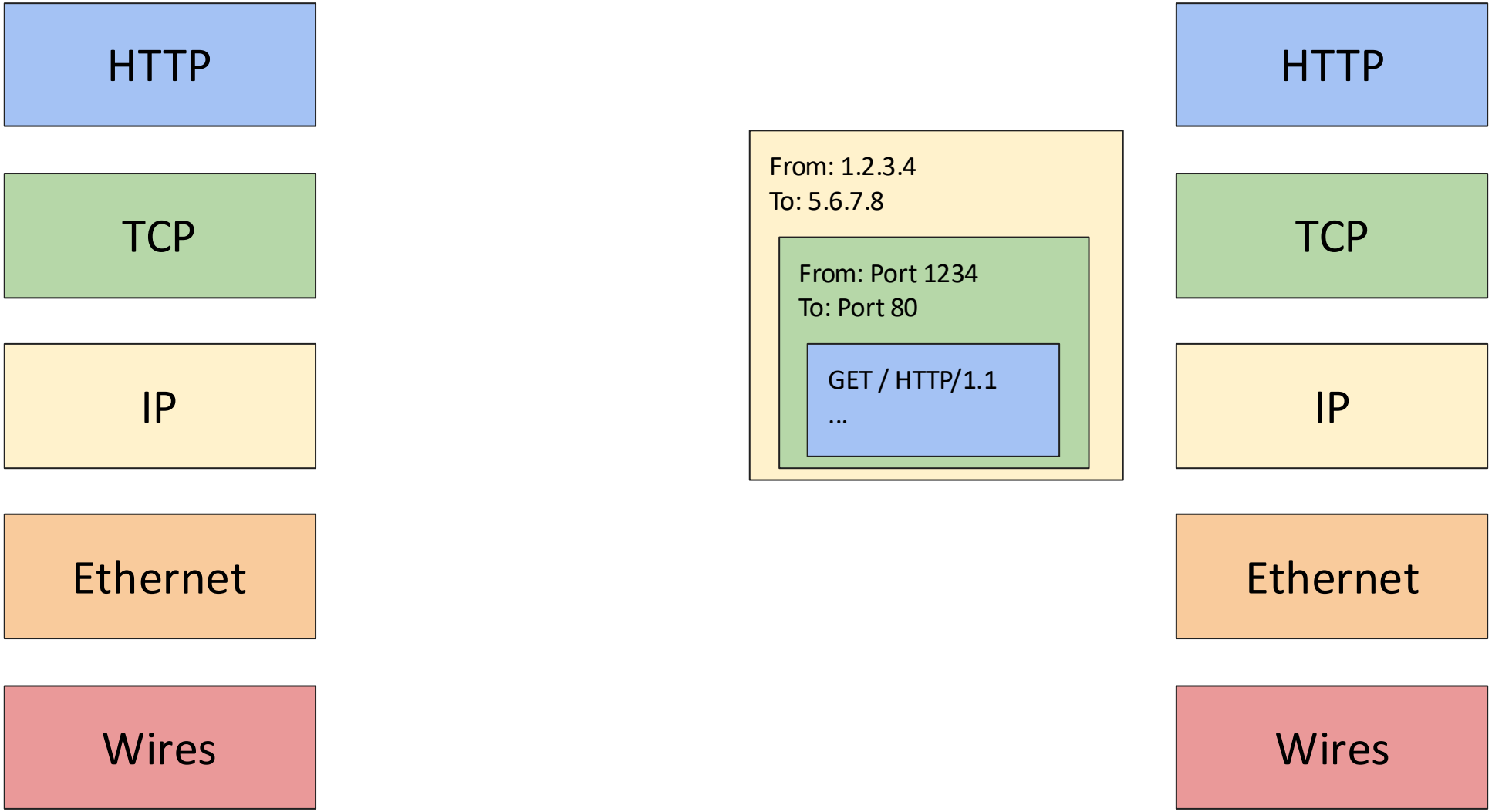
Example: HTTP Request



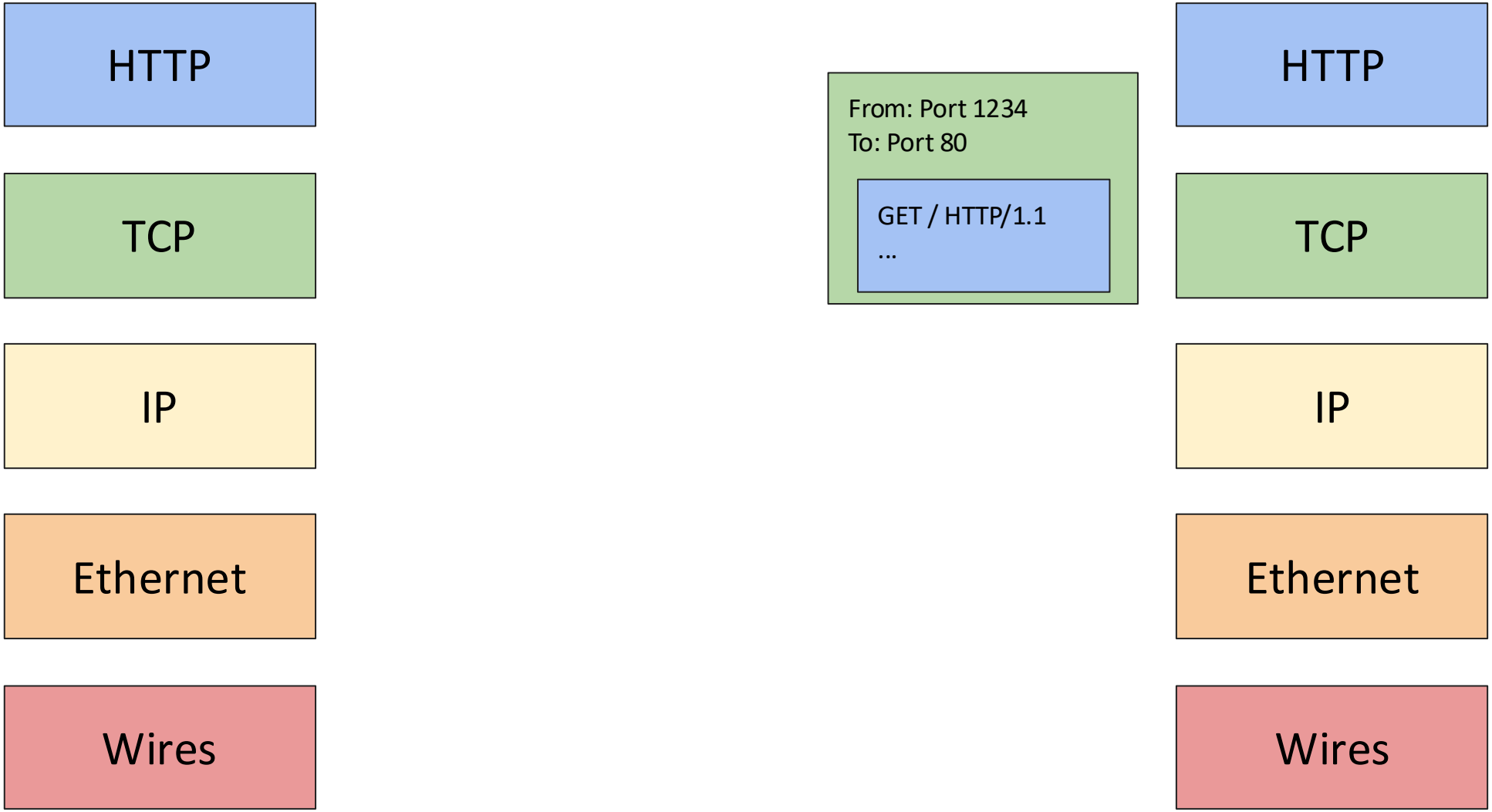
Example: HTTP Request



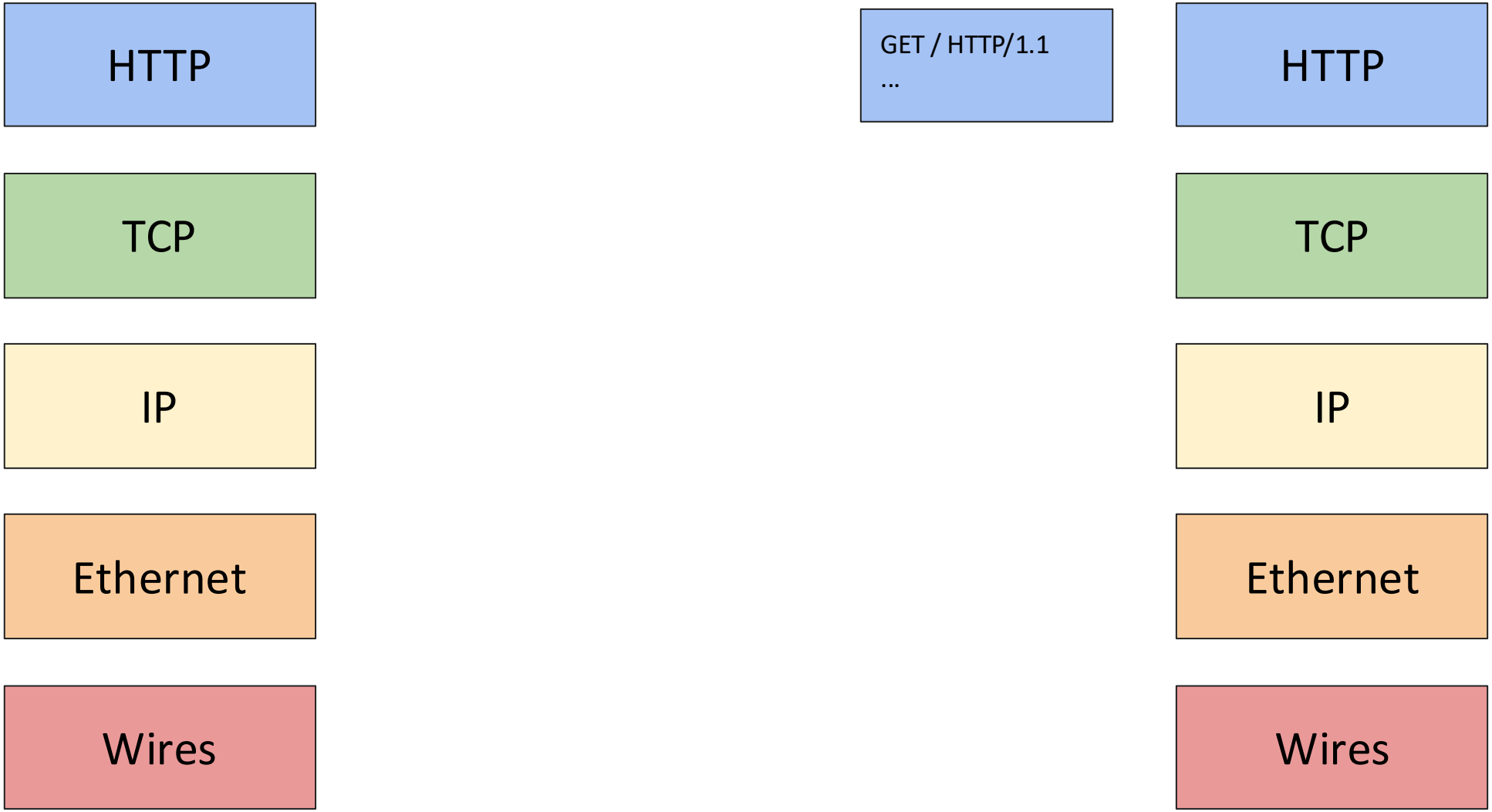
Example: HTTP Request



Example: HTTP Request

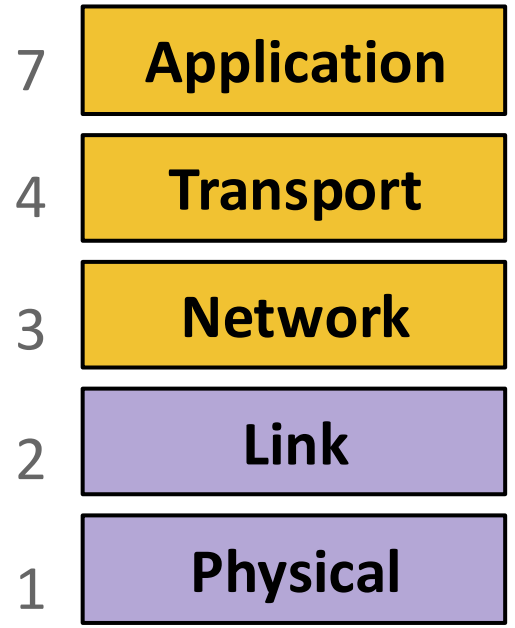


Example: HTTP Request



Networking Protocols & Headers

- Internet: A global network of computers
  - **Protocols:** Agreed-upon systems of communication
- Uses **layers** of protocols
  - Each layer handles a specific problem, and abstracts it away from other layers
- Add **protocol headers** for (most) layers: key information that provides clean abstractions
  - Wrap (Add) headers when Sending data
  - Unwrap (Remove) headers when Receiving data



Outline

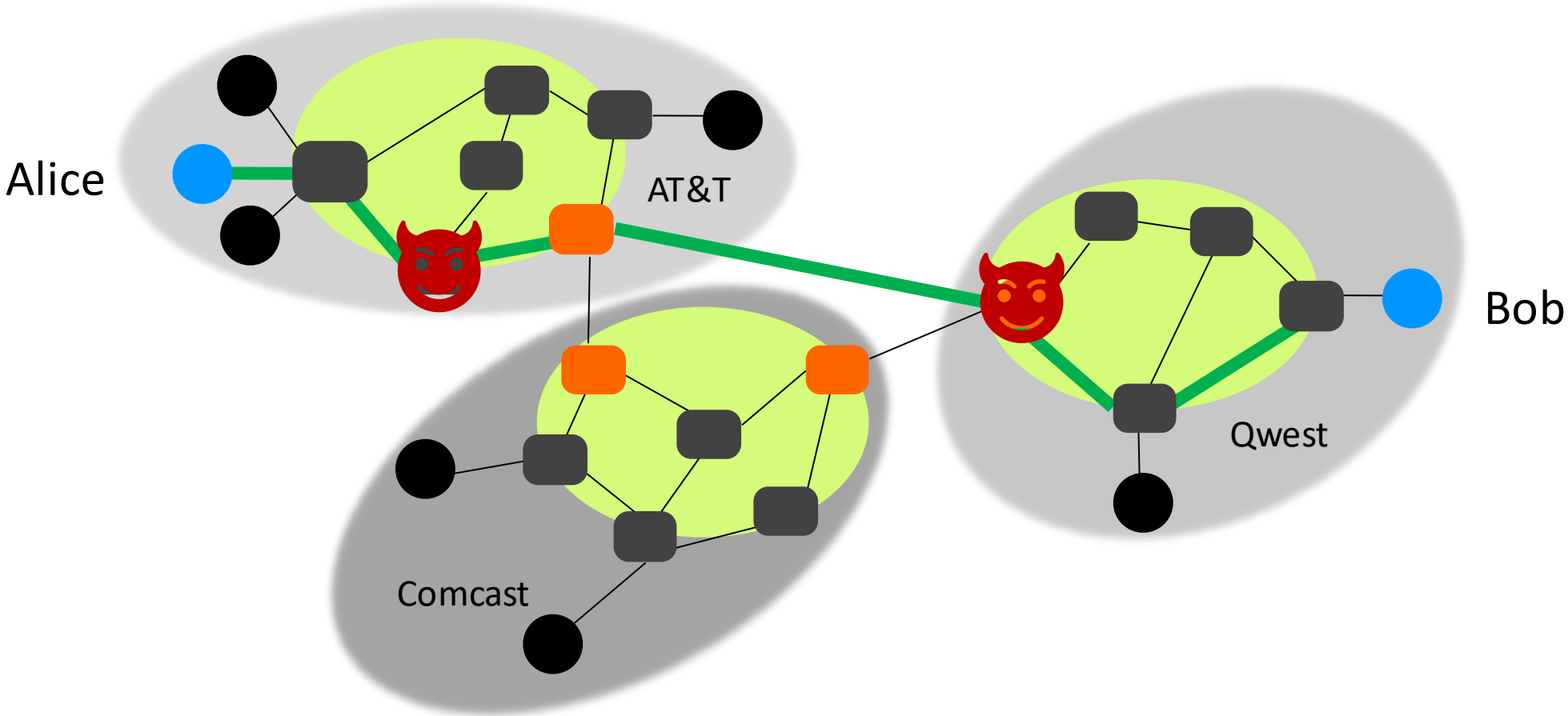
- Networking Background / How the Internet Works
  - The Internet & Networking Protocols
  - Protocol Layers and Addressing
  - Protocol Headers & Encapsulation
- Networking Threat Models
- ARP Security

Network Threat Model: 3 Types of Attackers

Alice & Bob want to communicate over the Internet:
What kinds of attackers do they need to worry about?

	Can modify or delete packets	Can read packets	Can inject new packets
1) In-Path attacker (Man-in-the-middle)	✓	✓	✓

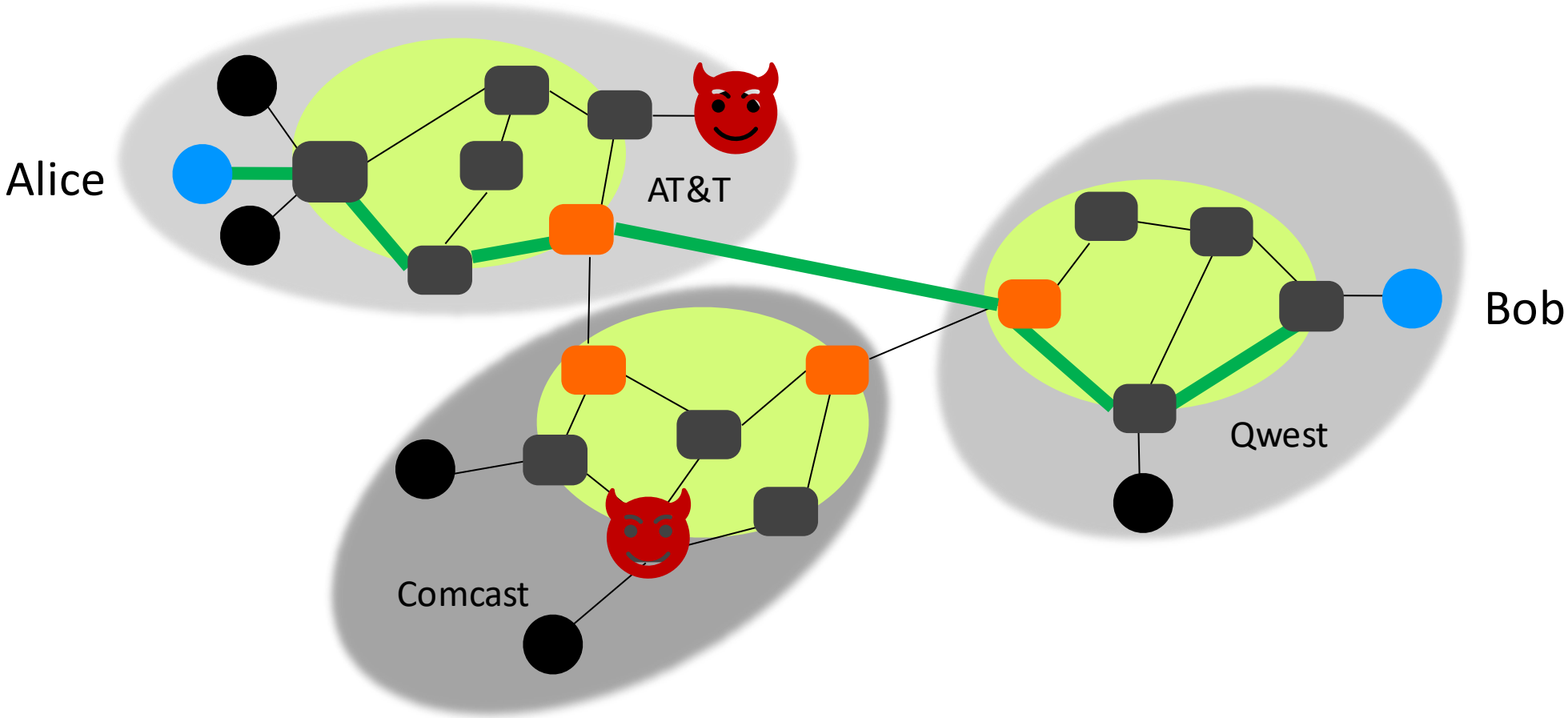
In-Path (MITM) Attacker Examples



Network Threat Model: 3 Types of Attackers

	Can modify or delete packets	Can read packets	Can inject new packets
1) In-Path attacker (Man-in-the-middle)	✓	✓	✓
3) Off-path attacker			✓

Off-Path Attacker Examples

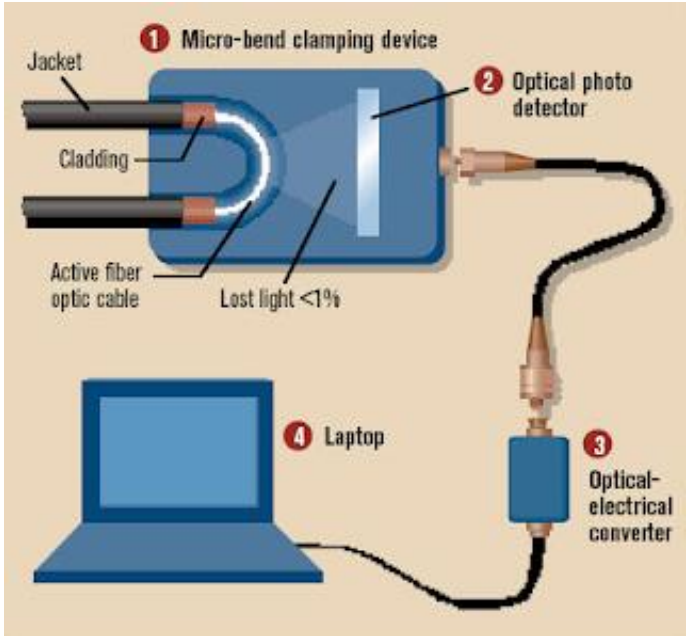


Network Threat Model: 3 Types of Attackers

	Can modify or delete packets	Can read packets	Can inject new packets
1) In-Path attacker (Man-in-the-middle)	✓	✓	✓
2) On-path attacker		✓	✓
3) Off-path attacker			✓

Real-World On-Path Attackers

- How might a real-life attacker read packets?
- Layer 1 attack: Use a special device to read bits being transmitted across space



Real-World On-Path Attackers

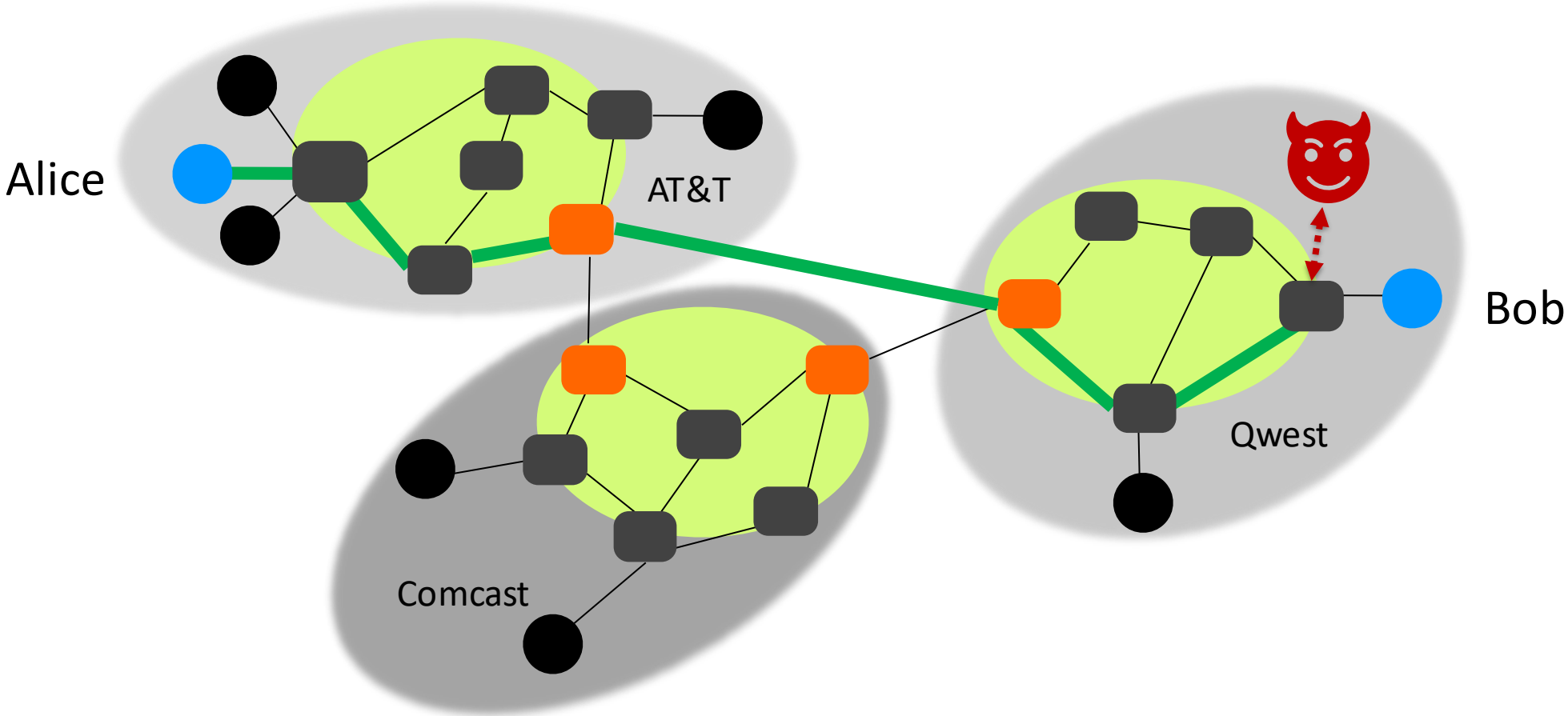
Military.com

Operation Ivy Bells

Matthew Carle *February 6, 2017*

In an effort to alter the balance of the Cold War, divers from the USS Halibut scoured the ocean floor for a five-inch diameter cable that carried secret Soviet communications between military bases. The divers found the cable and installed a listening device. Upon their return to the United States, the NSA analyzed the recordings and found that a surprising amount of sensitive Soviet information travelled through the lines without encryption. The original tap was later discovered by the Soviets and is now on exhibit at the KGB museum in Moscow.

On-Path Attacker Examples



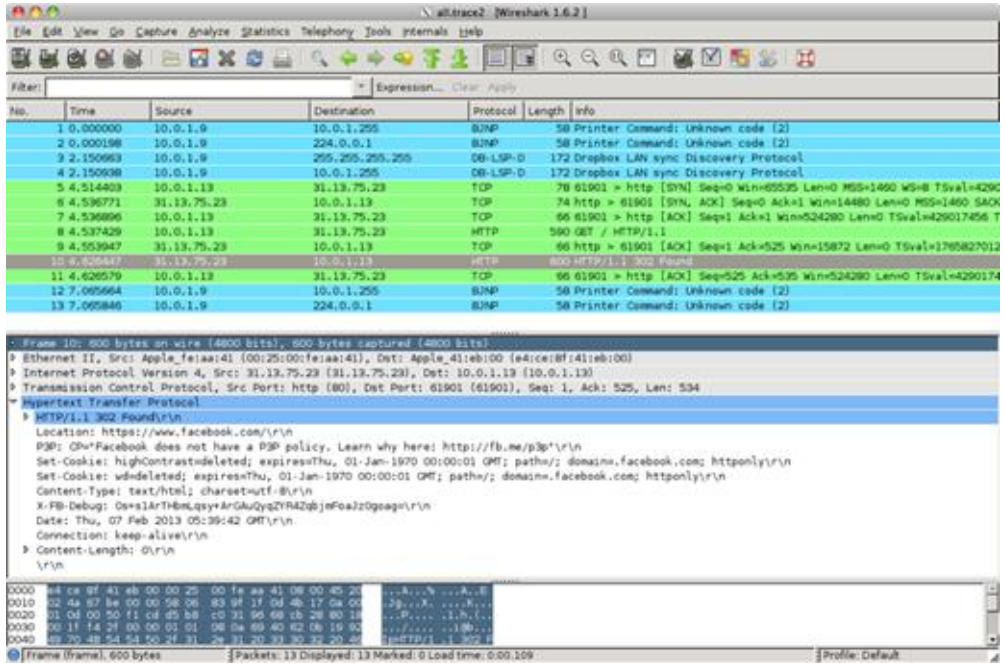
Real-World On-Path Attackers

- LAN (Local Area Network) is a network of connected machines
  - Any machine on LAN can send packets to other machines on the LAN
- Some LANs use **broadcast technologies** (e.g., Wifi)
  - Every packet gets sent to every machine on the LAN
  - Each machine agrees to ignore packets where the destination is a different machine, if they follow protocol
- A machine can break the agreement and read packets meant for other machines
  - This is called **promiscuous mode**

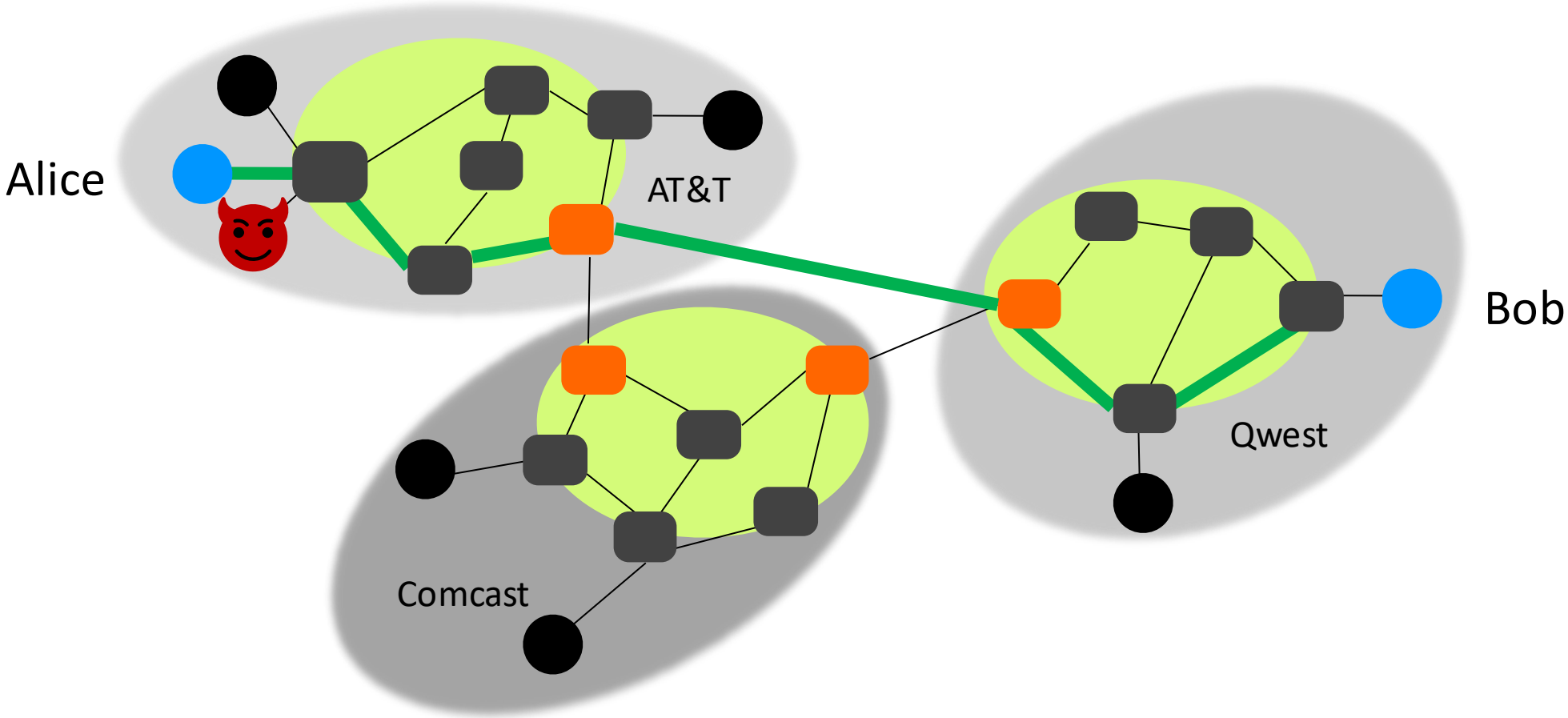
Real-World On-Path Attackers

- **`tcpdump`**: A program for reading packets on the local network
  - Uses promiscuous mode to read other machines' broadcast packets
- Wireshark: A graphical user interface (GUI) for analyzing **`tcpdump`** packets

```
demo 2 % tcpdump -r all.trace2
reading from file all.trace2, link-type EN10MB (Ethernet)
21:39:37.772367 IP 10.0.1.9.60627 > 10.0.1.255.canon-bjnp2: UDP, length 16
21:39:37.772565 IP 10.0.1.9.62137 > all-systems.mcast.net.canon-bjnp2: UDP, length 16
21:39:39.923030 IP 10.0.1.9.17500 > broadcasthost.17500: UDP, length 130
21:39:39.923305 IP 10.0.1.9.17500 > 10.0.1.255.17500: UDP, length 130
21:39:42.286770 IP 10.0.1.13.61901 > star-01-02-pao1.facebook.com.http: Flags [S], seq 2
523449627, win 65535, options [mss 1460,nop,wscale 3,nop,nop,TS val 429017455 ecr 0,sack
OK,eol], length 0
21:39:42.309138 IP star-01-02-pao1.facebook.com.http > 10.0.1.13.61901: Flags [S.], seq
3585654832, ack 2523449628, win 14480, options [mss 1460,sackOK,TS val 1765826995 ecr 42
9017455,nop,wscale 9], length 0
21:39:42.309263 IP 10.0.1.13.61901 > star-01-02-pao1.facebook.com.http: Flags [.], ack 1
, win 65535, options [nop,nop,TS val 429017456 ecr 1765826995], length 0
21:39:42.309796 IP 10.0.1.13.61901 > star-01-02-pao1.facebook.com.http: Flags [P.], seq
1:525, ack 1, win 65535, options [nop,nop,TS val 429017456 ecr 1765826995], length 524
21:39:42.326314 IP star-01-02-pao1.facebook.com.http > 10.0.1.13.61901: Flags [.], ack 5
25, win 31, options [nop,nop,TS val 1765827012 ecr 429017456], length 0
21:39:42.398814 IP star-01-02-pao1.facebook.com.http > 10.0.1.13.61901: Flags [P.], seq
1:535, ack 525, win 31, options [nop,nop,TS val 1765827083 ecr 429017456], length 534
21:39:42.398946 IP 10.0.1.13.61901 > star-01-02-pao1.facebook.com.http: Flags [.], ack 5
35, win 65535, options [nop,nop,TS val 429017457 ecr 1765827083], length 0
21:39:44.838031 IP 10.0.1.9.54277 > 10.0.1.255.canon-bjnp2: UDP, length 16
21:39:44.838213 IP 10.0.1.9.62896 > all-systems.mcast.net.canon-bjnp2: UDP, length 16
```



On-Path Attacker Examples



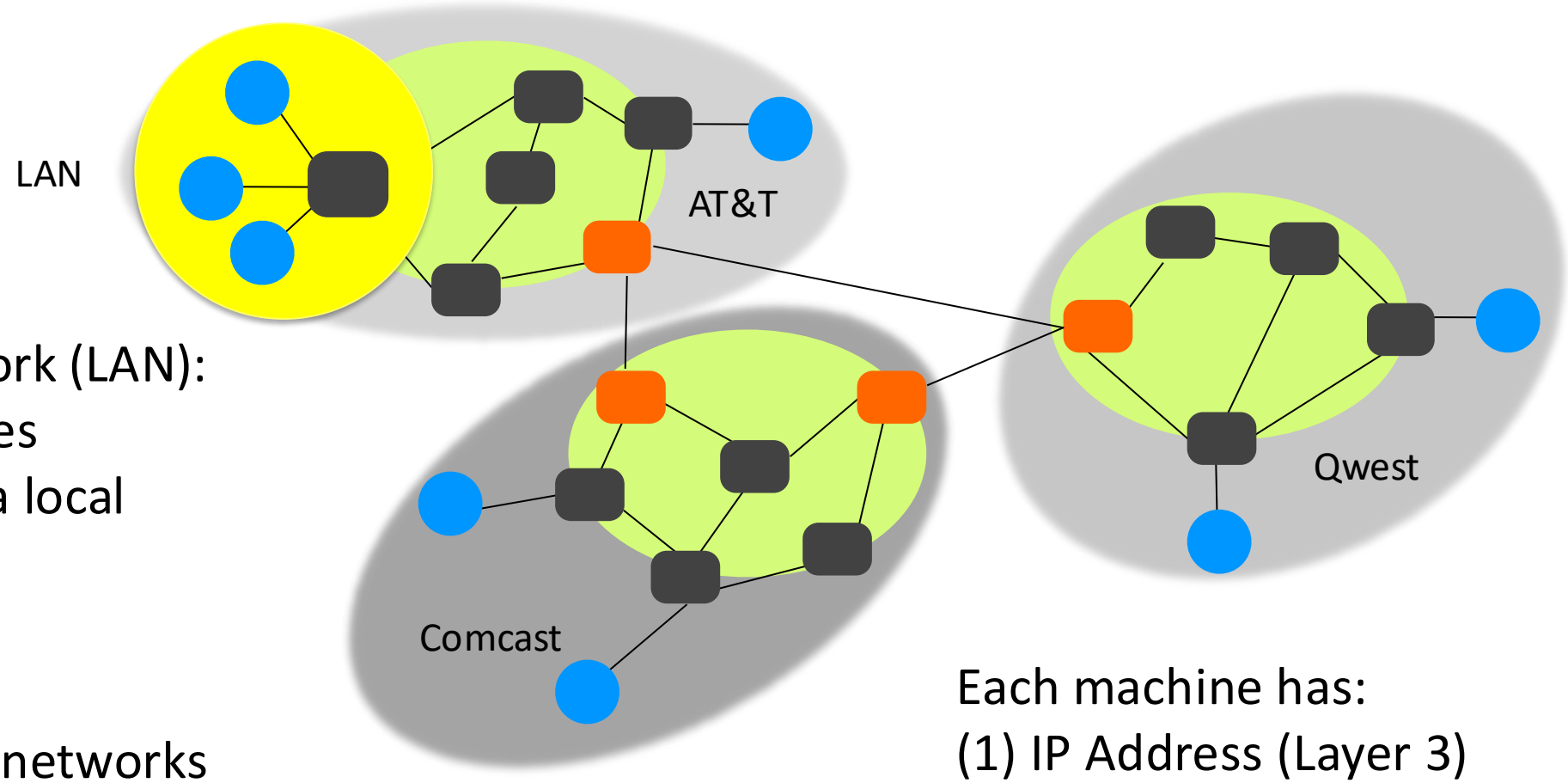
Network Attacks

- Attacks on confidentiality
  (e.g., eavesdropping, side channel information, scanning)
- Attacks on integrity
  (e.g., spoofing, packet injection)
- Attacks on availability
  (e.g., denial of service, or **DoS**)

Outline: Key Exchange & Networking Background

- Networking Background / How the Internet Works
  - The Internet & Networking Protocols
  - Protocol Layers and Addressing
  - Protocol Headers & Encapsulation
- Networking Threat Models
- ARP Security

Recall: The Internet From 10,000 Feet



Local Area Network (LAN):

- Set of machines connected in a local network

Internet (IP):

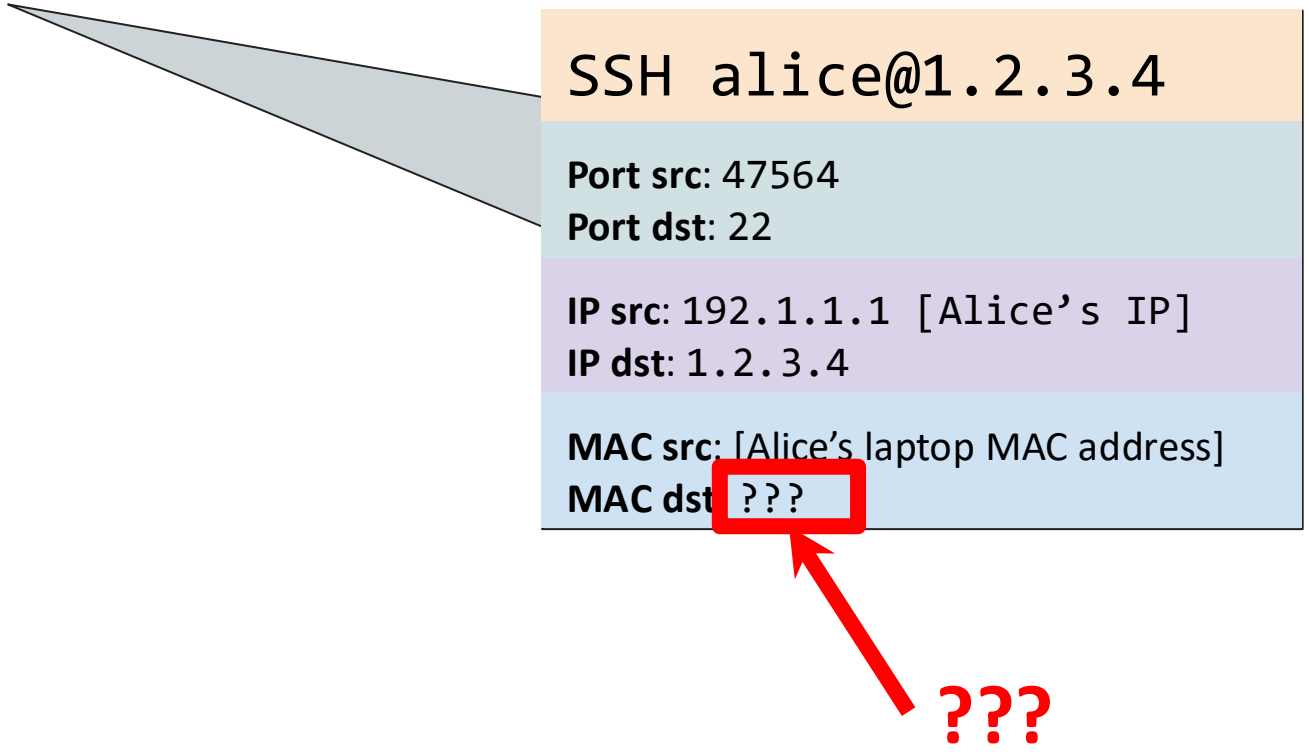
- Set of smaller networks connected via routers

Each machine has:
(1) IP Address (Layer 3)
(2) MAC Address (Layer 2)

Address Resolution Protocol (ARP)

The Problem: Alice knows Bob's IP address & wants to send him data (e.g., Alice performs ssh login to VM [Bob] @ IP address = `1.2.3.4`)

- What should she fill in for the Layer 2 header (MAC Address)?

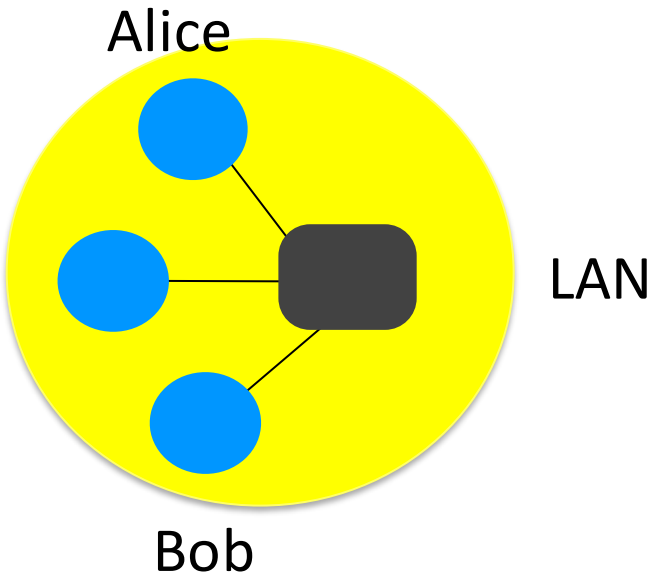


Address Resolution Protocol (ARP)

The Problem: Alice knows Bob's IP address & wants to send him data (e.g., Alice performs ssh login to VM [Bob] @ IP address = `1.2.3.4`)

- What should she fill in for the Layer 2 header (MAC Address)?

ARP: Translates IP addresses to MAC addresses



Address Resolution Protocol (ARP)

The Problem: Alice knows Bob's IP address & wants to send him data (e.g., Alice performs ssh login to VM [Bob] @ IP address = `1.2.3.4`)

- What should she fill in for the Layer 2 header (MAC Address)?

ARP: Translates IP addresses to MAC addresses

1. Alice checks her ARP cache to see if she already knows Bob's MAC address.
2. If Bob's MAC addr not in the cache, Alice **broadcasts** to everyone on the LAN: "What is the MAC address of **`1.2.3.4`**?"
3. Bob responds by sending a message only to Alice: "My IP is **`1.2.3.4`** and my MAC address is **`ca:fe:f0:0d:be:ef`**."
   Everyone else does nothing.
4. Alice caches Bob's MAC address & uses it.

Address Resolution Protocol (ARP)

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC

Alice

Bob

Charlie

Dave

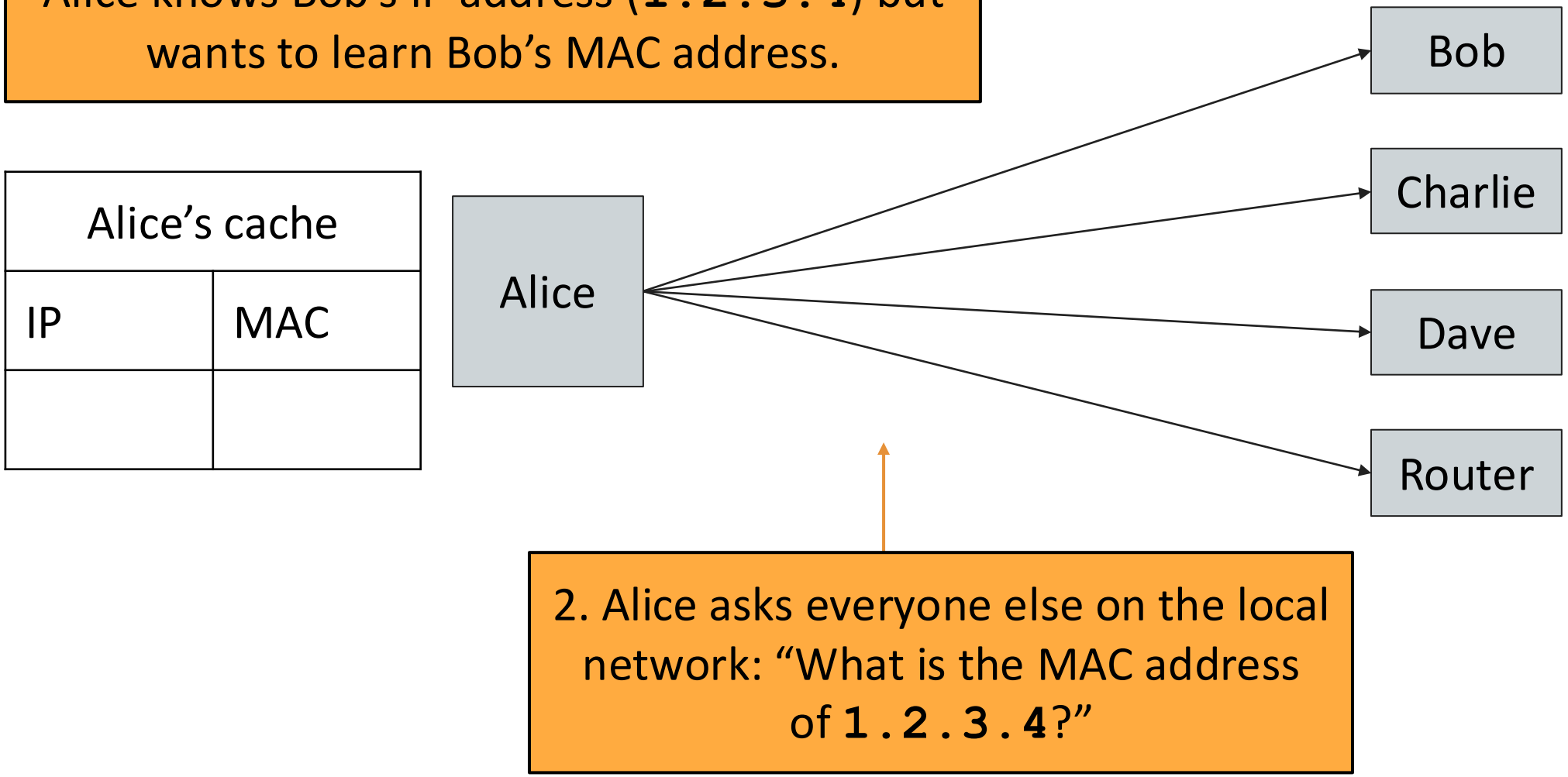
Router

1. Alice checks her cache to see if she already knows the MAC address corresponding to `1.2.3.4`.

Not in the cache: she must make a request to find out.

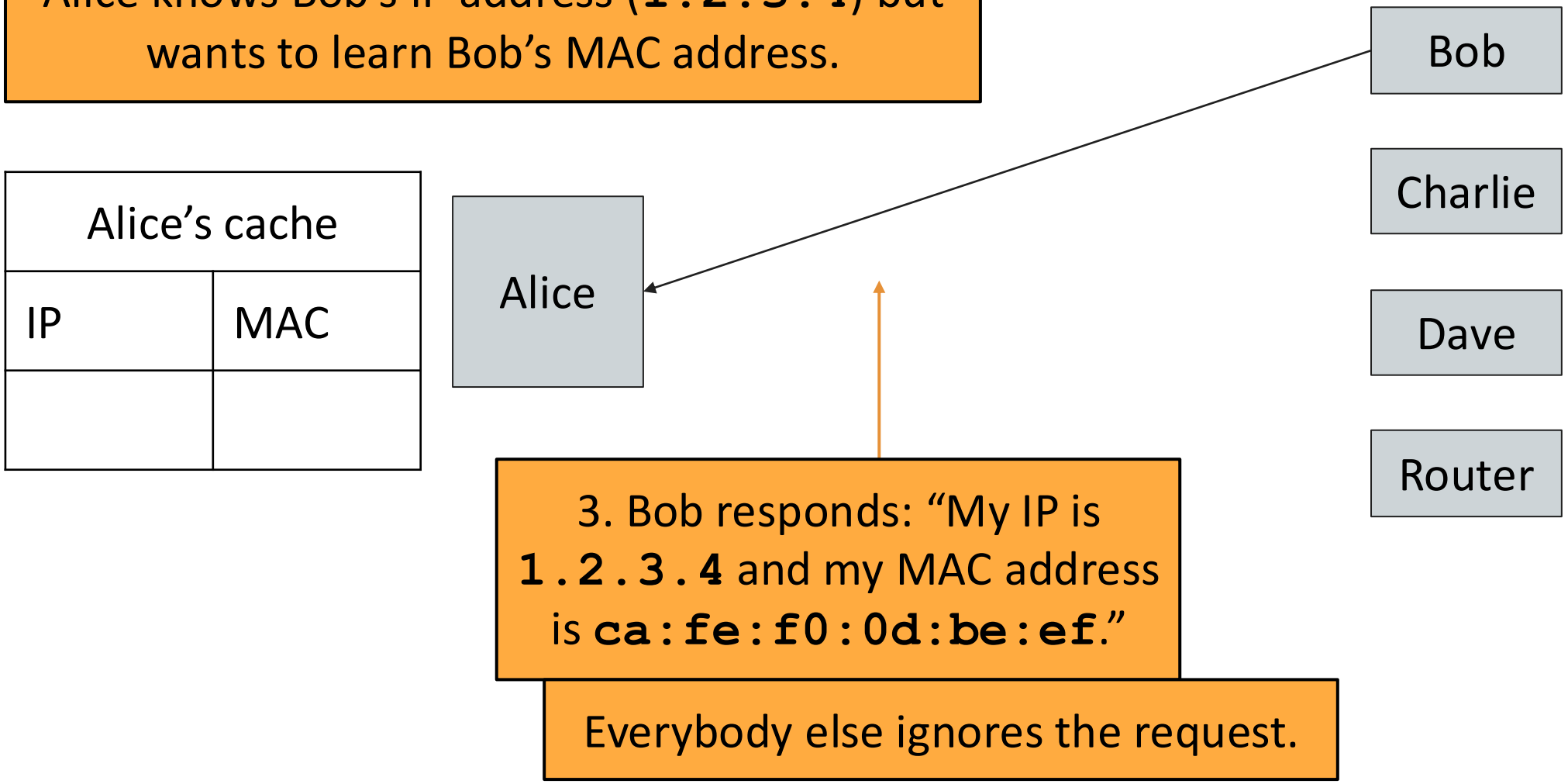
Address Resolution Protocol (ARP)

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.



Address Resolution Protocol (ARP)

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.



Address Resolution Protocol (ARP)

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC
`1.2.3.4`	`ca:fe:f0:0d:be:ef`

Alice

Bob

Charlie

Dave

Router

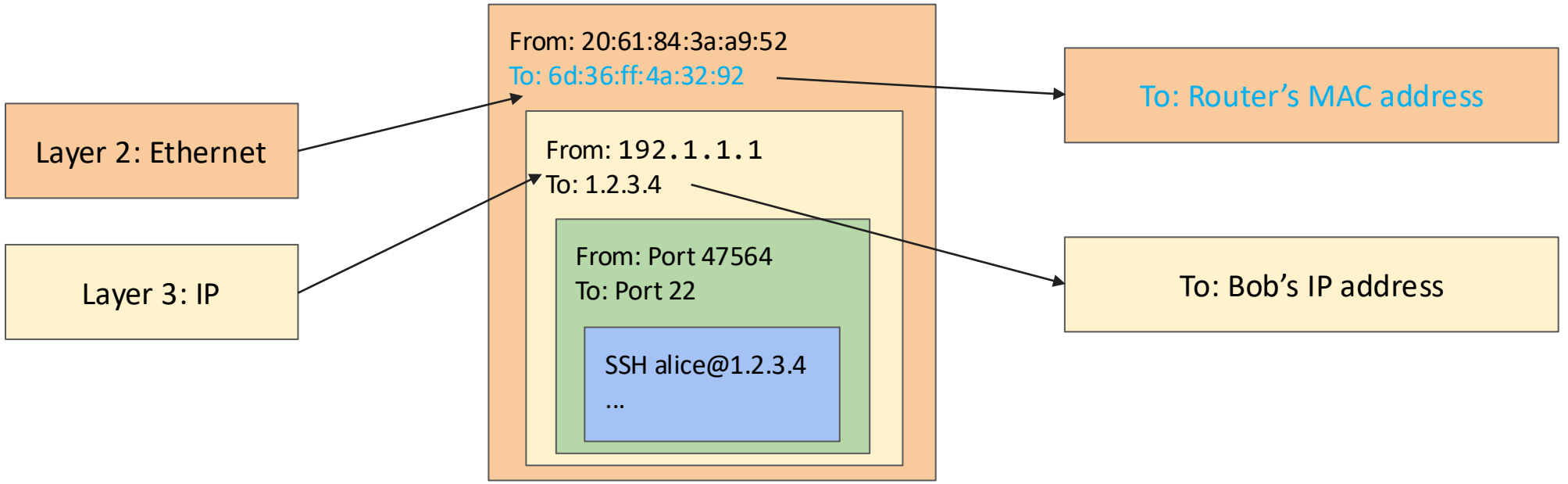
4. Alice adds Bob's MAC address to her cache.

NOTE: All received ARP replies are cached, even if no request was sent!!

Address Resolution Protocol (ARP)

If Bob is outside of the LAN, the router responds w/ its own MAC address

- If Alice wants to send a packet to Bob, she sends the packet to the router
- The router can forward the packet to other LANs to reach Bob



Spoofing Attacks

- Anybody can send their own packets through the network
- **Spoofing**: Lying about the identity of a packet's sender
  - The attacker can lie about source addresses in the packet header
  - Example: Mallory sends a message and says the message is from Bob
- All 3 types of attackers can spoof packets
  - However, some spoofing attacks may be harder if the attacker can't read or modify packets

Spoofing Attacks on ARP (v1)

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC

Alice

Bob

Charlie

Mallory

Router

1. Alice checks her cache to see if she already knows the MAC address corresponding to `1.2.3.4`.

Since her cache is empty, she must make a request to find out.

Attacks on ARP (v1)

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.

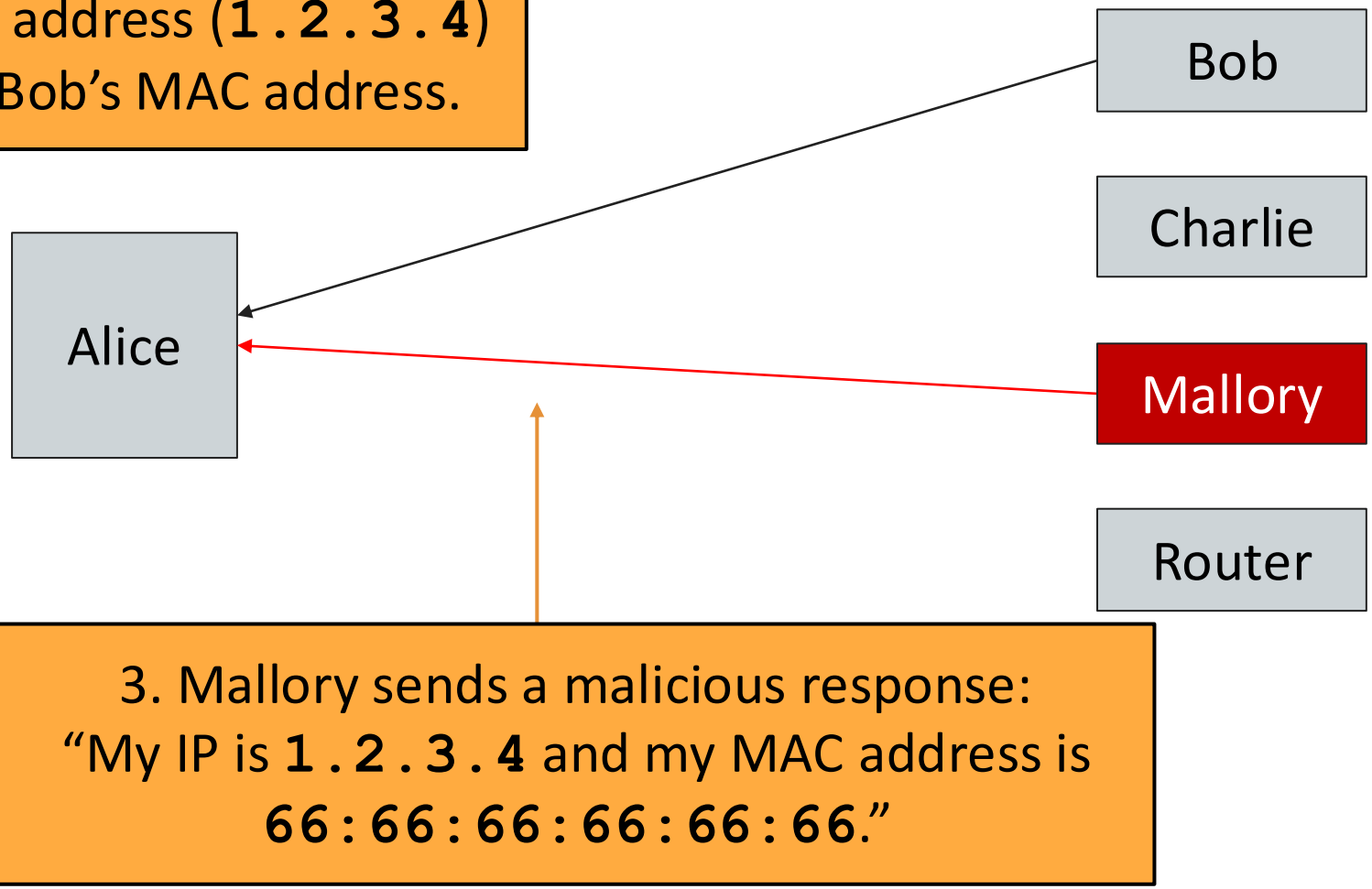
Alice's cache	
IP	MAC

2. Alice asks everyone else on the local network: "What is the MAC address of `1.2.3.4`?"

Attacks on ARP (v1)

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC



Attacks on ARP (v1)

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC
`1.2.3.4`	`66:66:66:66:66:66`

Alice

Bob

Charlie

Mallory

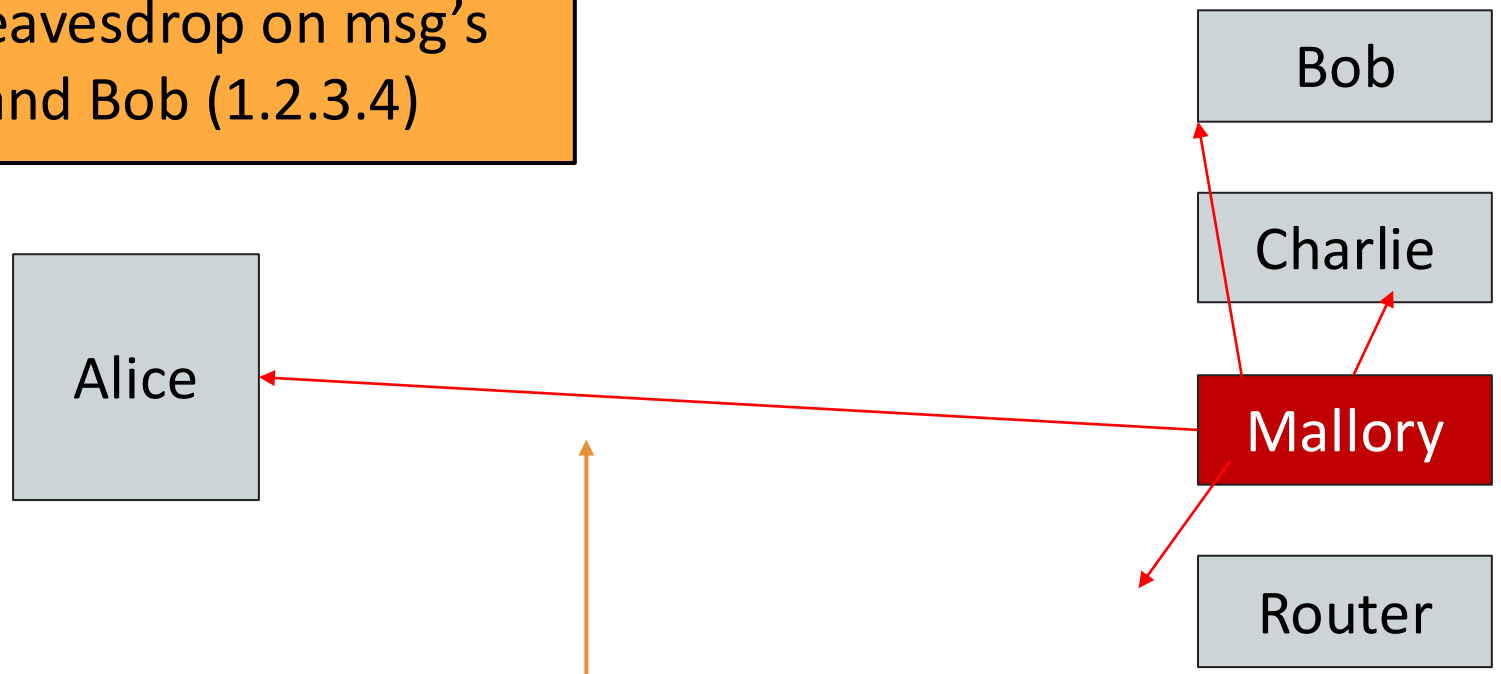
Router

4. Alice adds Mallory's malicious MAC address to her cache for Bob's IP addr!

Attacks on ARP (v2)

Mallory wants to eavesdrop on msg's between Alice and Bob (1.2.3.4)

Alice's cache	
IP	MAC
`1.2.3.4`	`1:1:1:1`

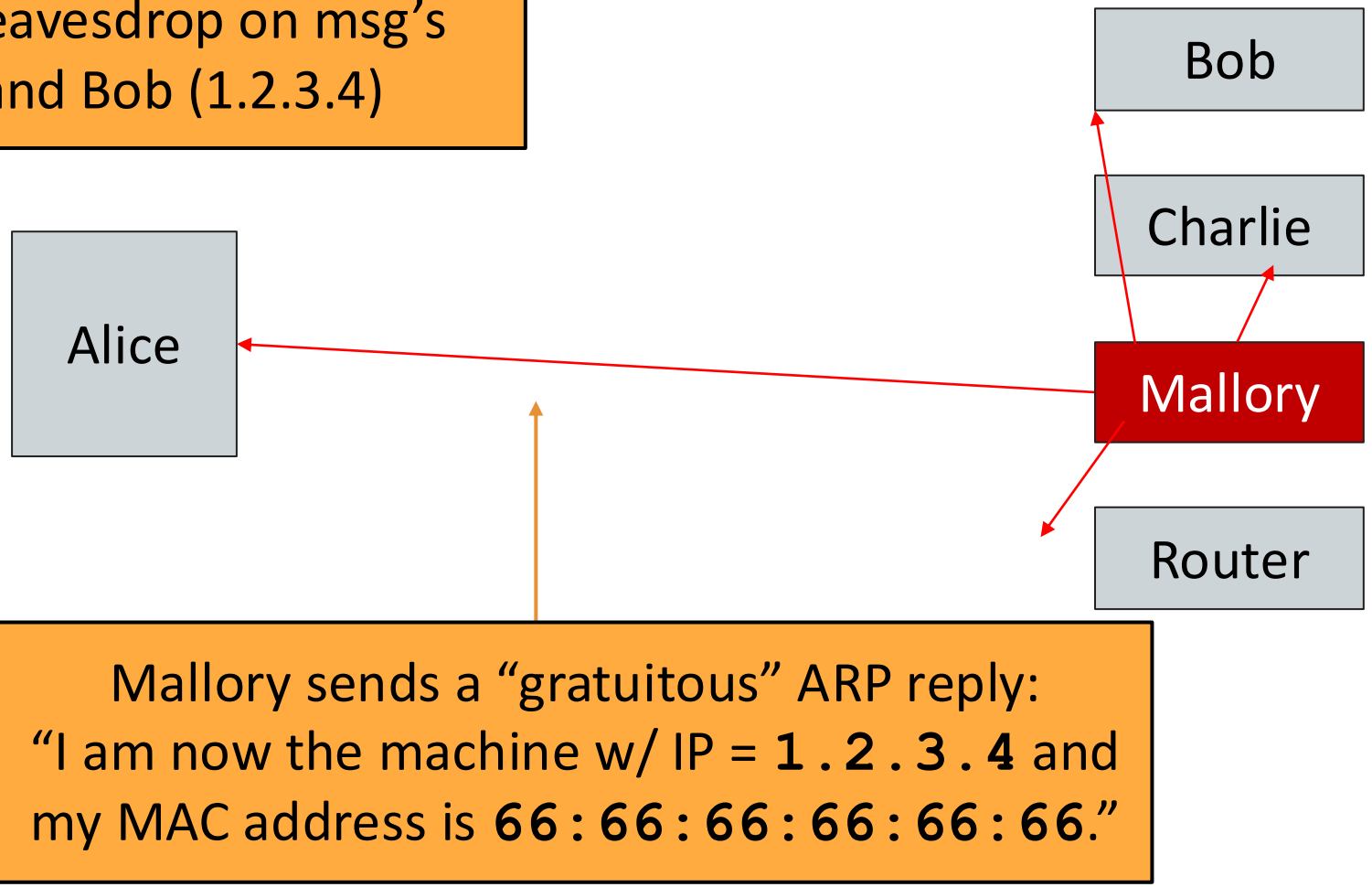


Mallory sends a "gratuitous" ARP reply:
"I am now the machine w/ IP = **`1.2.3.4`** and my MAC address is **`66:66:66:66:66:66`**."

Attacks on ARP (v2)

Mallory wants to eavesdrop on msg's between Alice and Bob (1.2.3.4)

Alice's cache	
IP	MAC
`1.2.3.4`	`66:66:66:66:66:66`



Mallory sends a "gratuitous" ARP reply:
"I am now the machine w/ IP = **`1.2.3.4`** and my MAC address is **`66:66:66:66:66:66`**."

Attack: ARP Spoofing

- Alice has no way of verifying the ARP response
- ARP protocol assumes that responses are accurate information and machines obey this assumption

- ARP spoofing requires Mallory to be in the same LAN as Alice
- ARP spoofing lets Mallory become a man-in-the-middle (MITM)
  - Alice thinks that Bob's MAC address is **`66:66:...:66`** (Mallory's MAC address)
  - When Alice sends a message to Bob, she is actually sending the msg to Mallory
  - Mallory can modify the message and then send the modified message to Bob

ARP Spoofing: Defenses

- Repeater Hubs vs. Switches
- Use **switches** to avoid broadcasts and ARP requests
  - When Alice wants to msg Bob, she sends the msg to a switch on the LAN
  - The switch maintains a cache of IP <-> MAC mappings
  - If Bob's MAC address in the cache, the switch sends the msg directly to Bob
  - Otherwise, the switch broadcasts the message
- Benefits of switches
  - Efficiency: Fewer broadcast requests
  - Security: Reduces the number of messages broadcast to the entire LAN & isolation: can create "virtual" VLANs in software (guest Wifi vs. main Wifi)