Key Exchange & How the Internet Works

CMSC 23200, Spring 2025, Lecture 6

Grant Ho

University of Chicago, 04/10/2025
Some slides adapted from Blasé Ur, Peyrin Kao, and Zakir Durumeric

Logistics

Assignment 2 (Buffer Overflow): **Due Tonight, 11:59pm**

Assignment 3 (Crypto): **Released Friday evening**

No Discussion Section Next Week (04/16)

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

Why not use asymmetric crypto for everything?

Answer

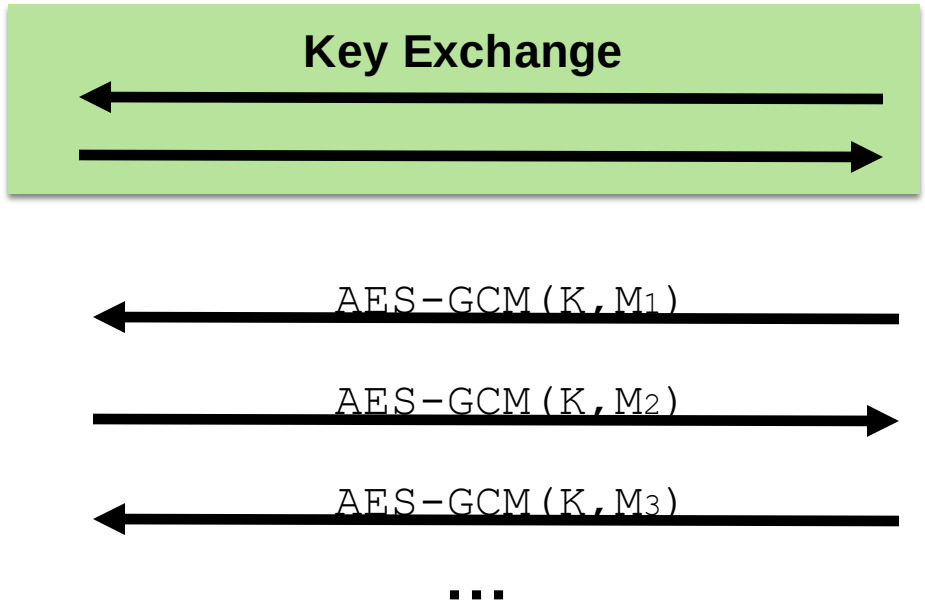
Symmetric key crypto algorithms are **MUCH** faster

Security Goal / Pre-shared key?	**Confidentiality**	**Authenticity/Integrity**
Yes ("Symmetric")	Symmetric Encryption	Message Authentication Code (MAC)
No ("Asymmetric")	Public-Key Encryption	Digital Signatures

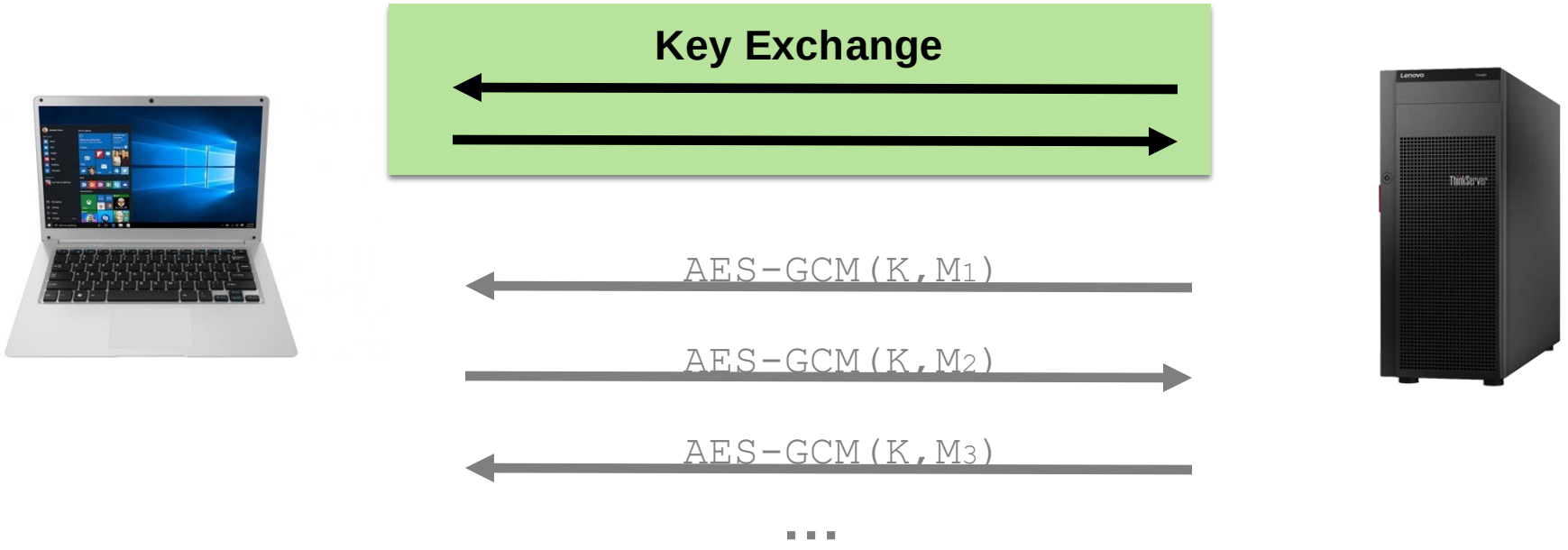
Hybrid Encryption: Real-world Secure Channels

Strategy:

1. Alice & Bob use a key exchange protocol to share their secret key(s)
2. Alice & Bob then use symmetric authenticated encryption (fast) for all their msg's



Key Exchange Protocols



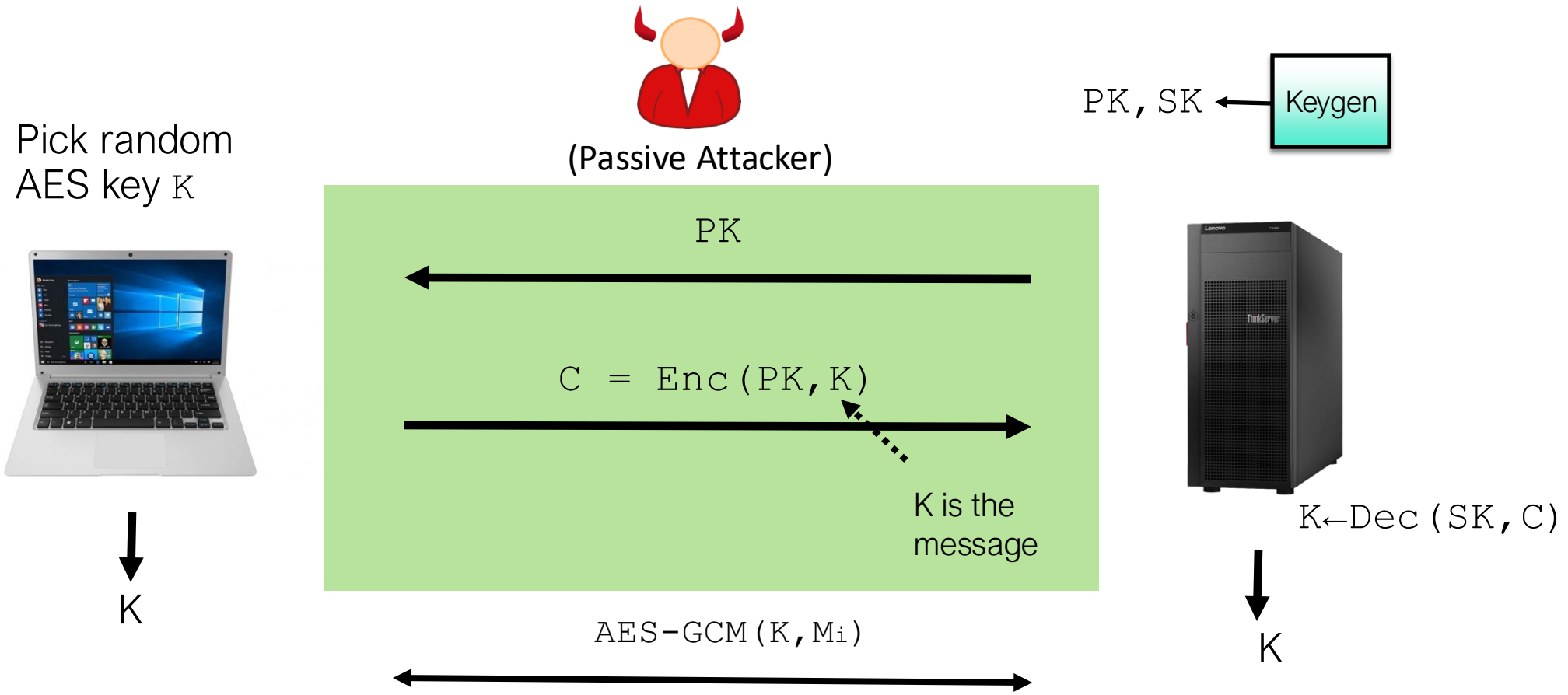
How can two parties securely share a symmetric key with each other even if:

1. They've never met before
2. An attacker is present & eavesdropping when they first meet

Key Exchange using Public Key Crypto

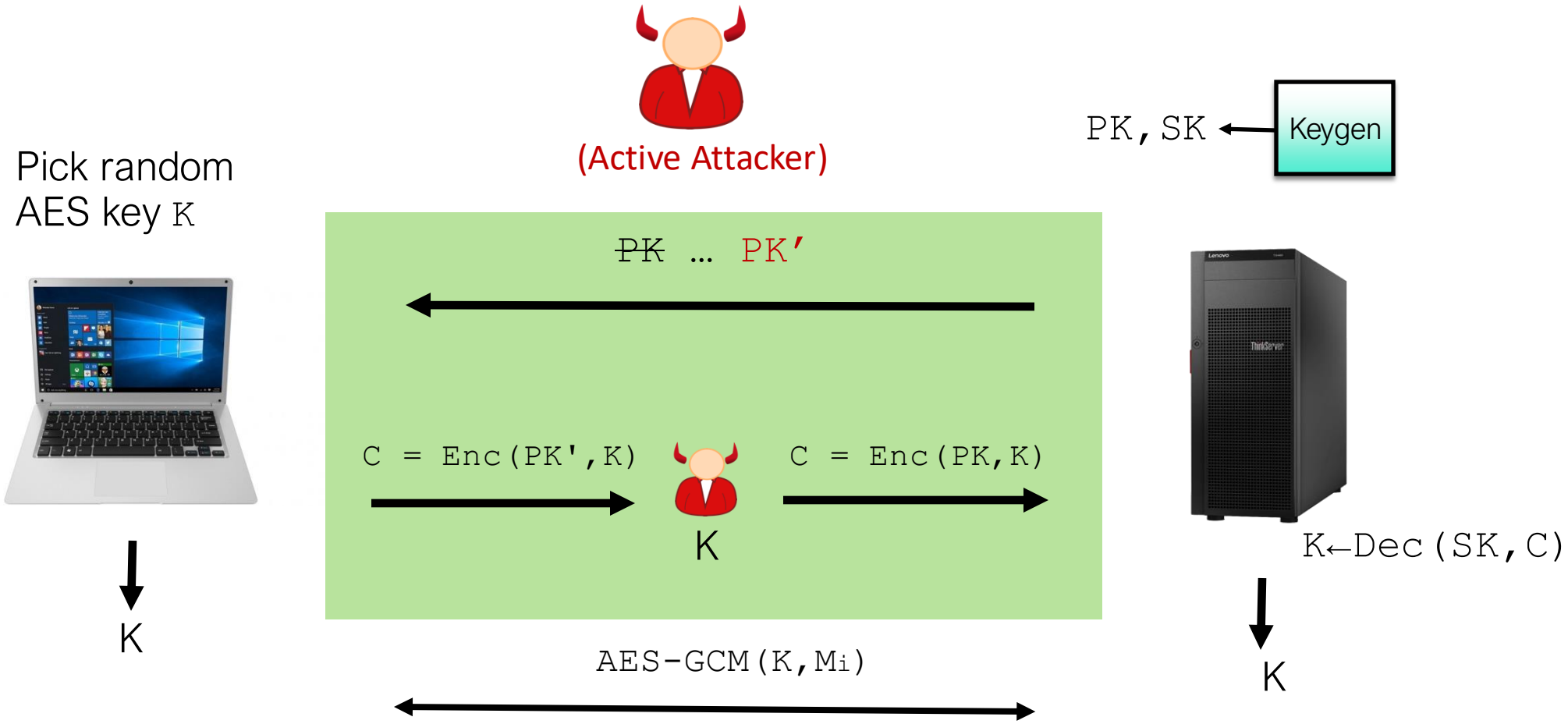
Goal: Establish secret key `K` to use for Authenticated Encryption.

`(KeyGen, Enc, Dec)` is a public-key encryption scheme.



Key Exchange using Public Key Crypto

Q: How do we make this secure against an active attacker?
A: Certificates w/ Signatures (Next Week)



Diffie-Hellman:
An alternative approach to key exchange

The modulus N for RSA is relatively large

- Problem: slows down encryption/decryption

Diffie-Hellman: A totally different, faster approach based on different math

- Invented in 1970s, but new ideas have recently made it the standard choice
- Strictly speaking, ***not*** public-key encryption:
    - DH focused on exchanging a symmetric key securely, and ***not*** general-purpose algorithm for encrypting data

Starting Point: Discrete Logarithm Problem

Discrete Logarithm Problem:

Given: Prime p, integers g, X.

Find: integer r such that $g^r = X \bmod p$

Taking this *Discrete Log* (computing r): believed to be computationally infeasible for large primes (p)

- Different from factoring: Only one prime.
- Contrast with logarithms with real numbers, which are easy to compute: *Discrete* logarithms appear to be hard to compute
- Largest solved instances: 795-bit prime p (Dec 2019)

Diffie-Hellman Key Exchange

Parameters: (fixed in standards, used by everyone):

Prime p (1024 bit usually)

Number g (usually 2) (p, g)

Alice (Client)

Bob (Server)

Diffie-Hellman Key Exchange

Parameters: (fixed in standards, used by everyone):

Prime p (1024 bit usually)

Number g (usually 2) (p, g)

Network Working Group
Request for Comments: 5114
Category: Informational

M. Lepinski
S. Kent
BBN Technologies
January 2008

Additional Diffie-Hellman Groups for Use

Status of This Memo

This memo provides information for the Inte
not specify an Internet standard of any kin
memo is unlimited.

Abstract

This document describes eight Diffie-Hellma
in conjunction with IETF protocols to provi
communications. The groups allow implement
with a variety of security protocols, e.g.,
(SSH), Transport Layer Security (TLS), and
(IKE).

3. 2048-bit MODP Group

This group is assigned id 14.

This prime is: 2^2048 - 2^1984 - 1 + 2^64 * { [2^1918 pi] + 124476 }

Its hexadecimal value is:

FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1
29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD
EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245
E485B576 625E7EC6 F44C42E9 A637ED6B 0BFF5CB6 F406B7ED
EE386BFB 5A899FA5 AE9F2411 7C4B1FE6 49286651 ECE45B3D
C2007CB8 A163BF05 98DA4836 1C55D39A 69163FA8 FD24CF5F
83655D23 DCA3AD96 1C62F356 208552BB 9ED52907 7096966D
670C354E 4ABC9804 F1746C08 CA18217C 32905E46 2E36CE3B
E39E772C 180E8603 9B2783A2 EC07A28F B5C55DF0 6F4C52C9
DE2BCBF6 95581718 3995497C EA956AE5 15D22618 98FA0510
15728E5A 8AACAA68 FFFFFFFF FFFFFFFF

The generator is: 2.

Diffie-Hellman Key Exchange

Parameters: (fixed in standards, used by everyone):

Prime p (1024 bit usually)

Number g (usually 2) (p, g)

Pick $r_A \in \{1, \dots, p-1\}$
$A \leftarrow g^{r_A} mod p$

Pick $r_B \in \{1, \dots, p-1\}$
$B \leftarrow g^{r_B} mod p$

Alice (Client)

A →

← B

Bob (Server)

$K \leftarrow A^{r_B} mod p =$

Diffie-Hellman Key Exchange

Parameters: (fixed in standards, used by everyone):

Prime p (1024 bit usually)

Number g (usually 2) (p, g)

Pick $r_A \in \{1, \dots, p-1\}$
$A \leftarrow g^{r_A} mod p$

Pick $r_B \in \{1, \dots, p-1\}$
$B \leftarrow g^{r_B} mod p$

Alice (Client)

A →

← B

Bob (Server)

$K \leftarrow B^{r_A} mod p = g^{r_b r_A} mod p$

$K \leftarrow A^{r_B} mod p = g^{r_A r_B} mod p$

Correctness: $B^{r_A} = (g^{r_B})^{r_A} = g^{r_A r_B} = (g^{r_A})^{r_B} = A^{r_B} mod p$

Security of Diffie-Hellman (Passive Attacker)

$r_A \in \{1, \ldots, p-1\}$
$A \leftarrow g^{r_A} mod p$

Pick $r_B \in \{1, \ldots, p-1\}$
$B \leftarrow g^{r_B} mod p$

Alice (Client)

A →

← B

Bob (Server)

$K \leftarrow B^{r_A} mod p$

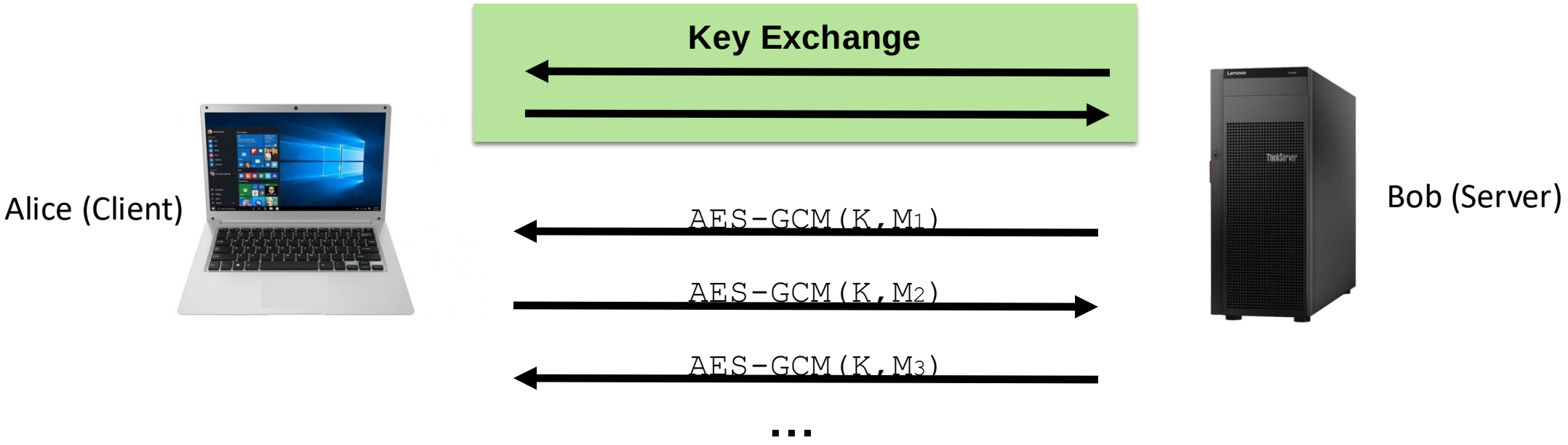
Knows: p, g, A, B
Compute K?

$K \leftarrow A^{r_B} mod p$

Best attack known: Compute discrete log of A or B

Hybrid Encryption: Assembling Crypto Tools

1. Alice & Bob use a key exchange protocol to share their secret key(s)
   - [Passive Attackers]: Public Key Crypto or Diffie-Hellman
2. Alice & Bob then use (symmetric) authenticated encryption for all msg's
   - Symmetric Encryption for Confidentiality + MAC's for Integrity



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

The Internet

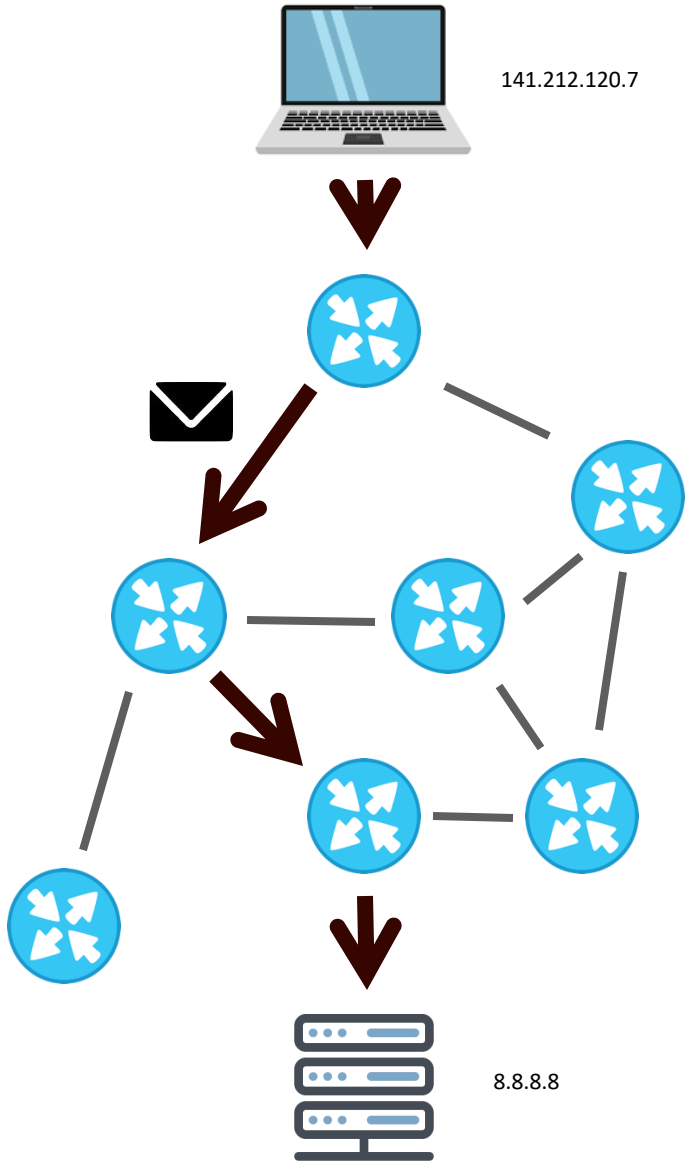
Global network that provides **best-effort** delivery of **packets** between connected hosts

Packet: a structured sequence of bytes

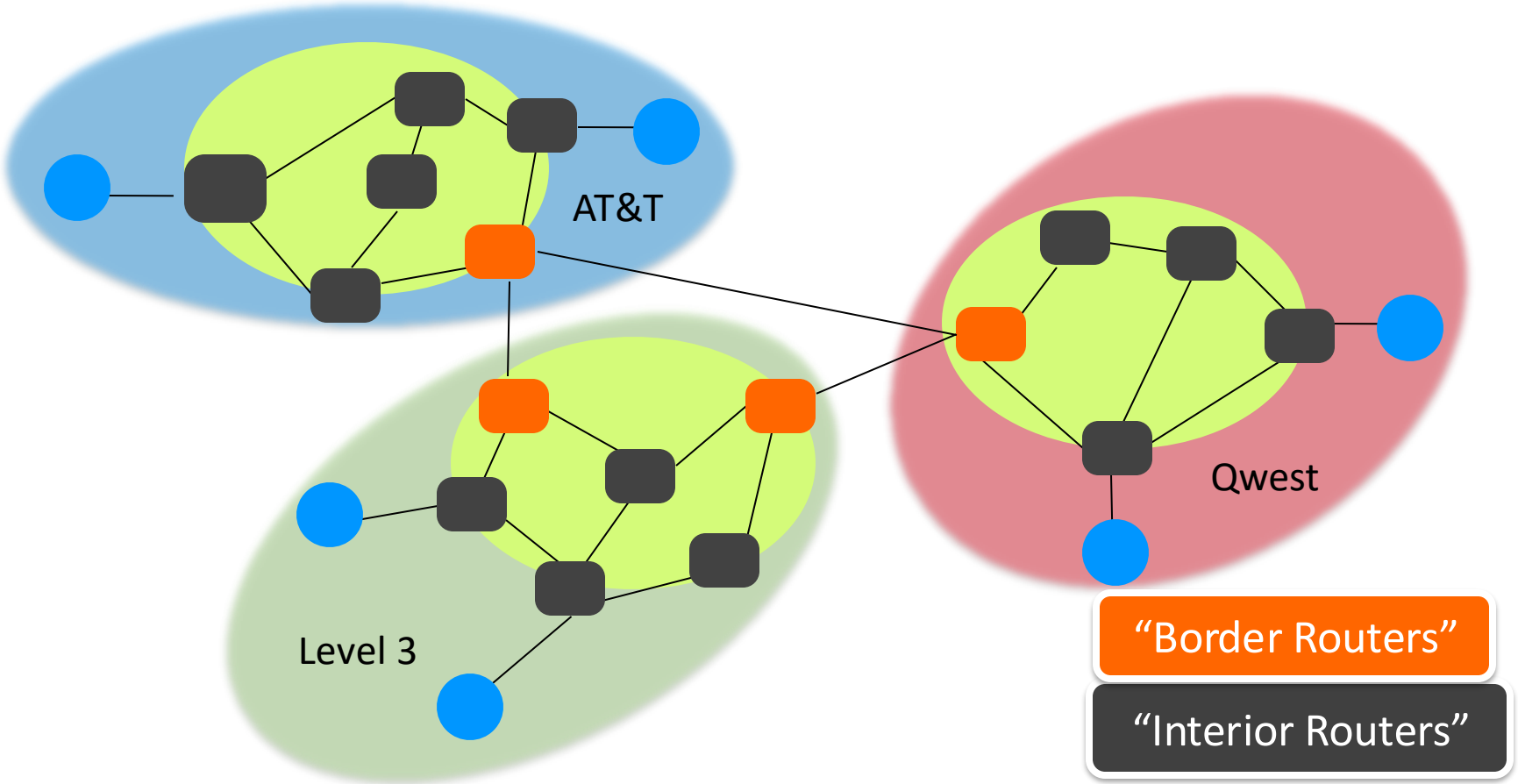
- Header: metadata used by network
- Payload: user data to be transported

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

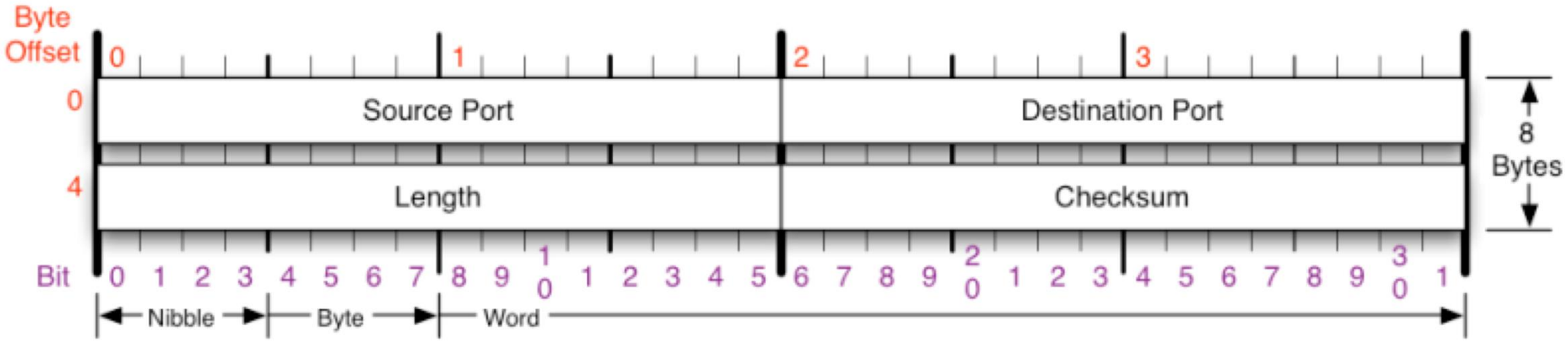


Network Protocols

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

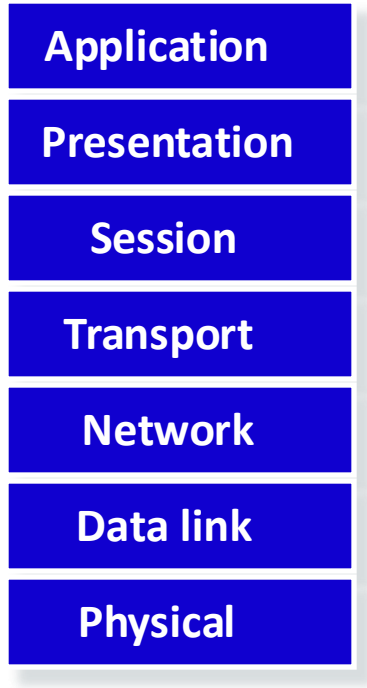
- Each layer has different responsibilities.
- A layer interacts only with layer above and layer below

Lower layers provide services to layers above

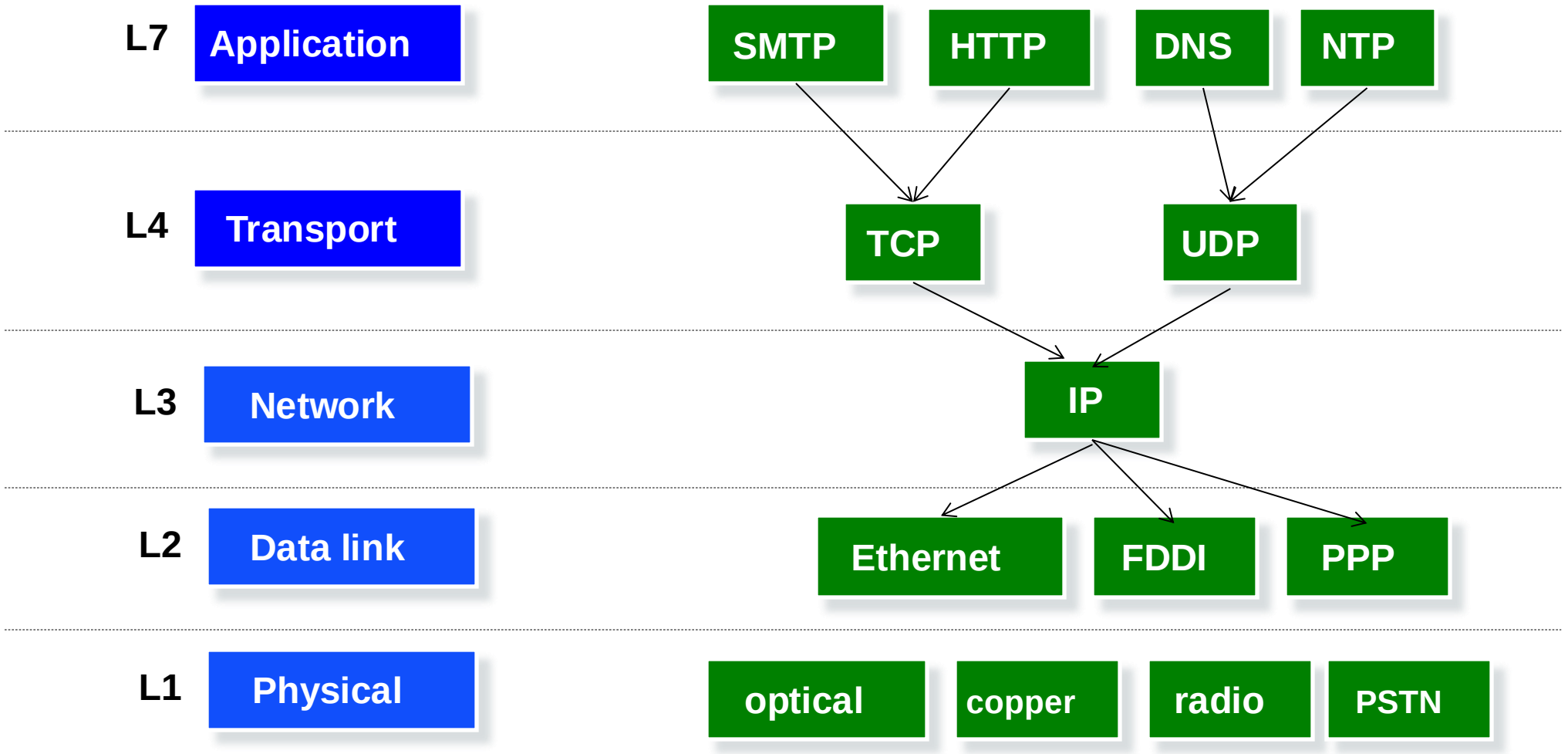
- Don’t care what higher layers do

Higher layers use services of layers below

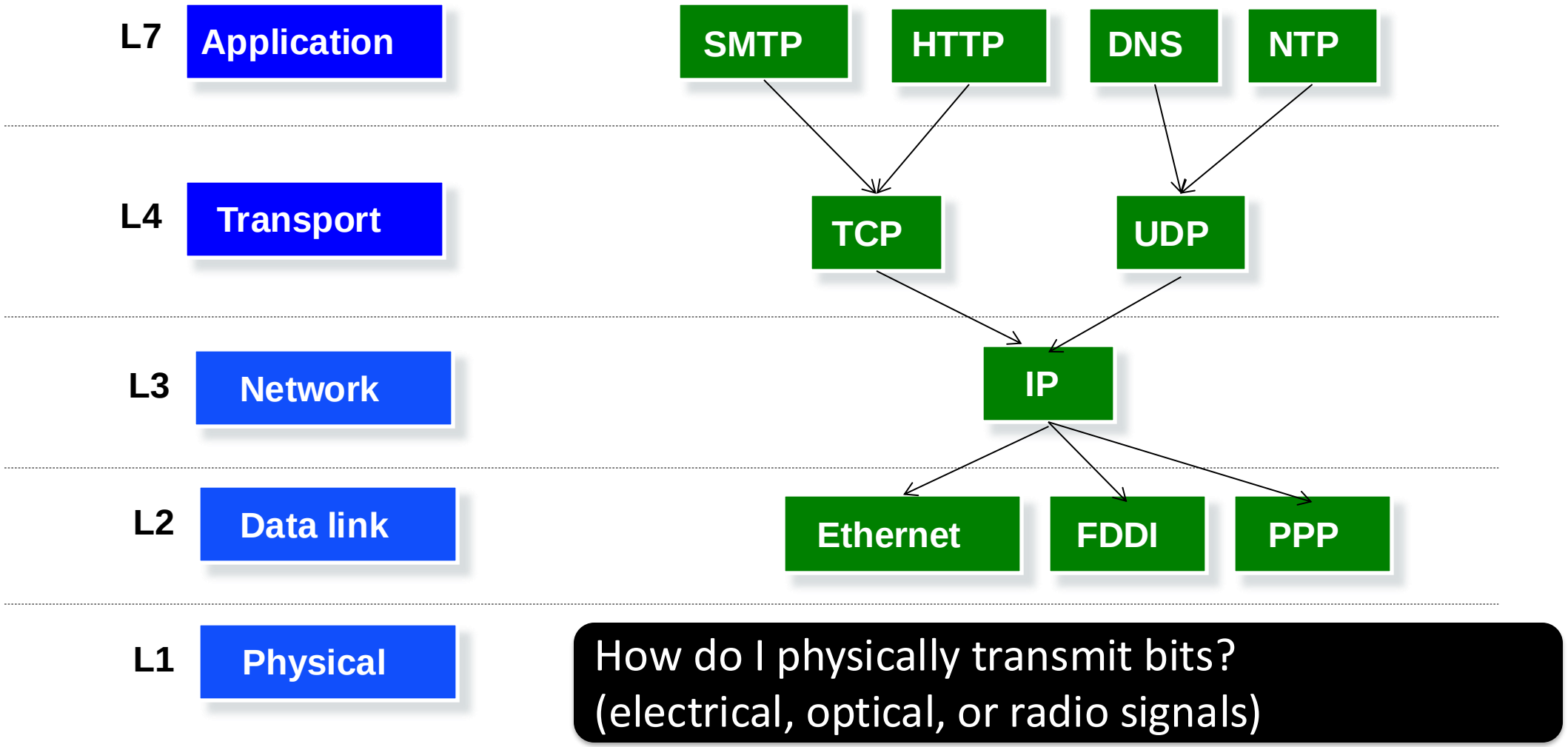
- Don’t worry about how it works



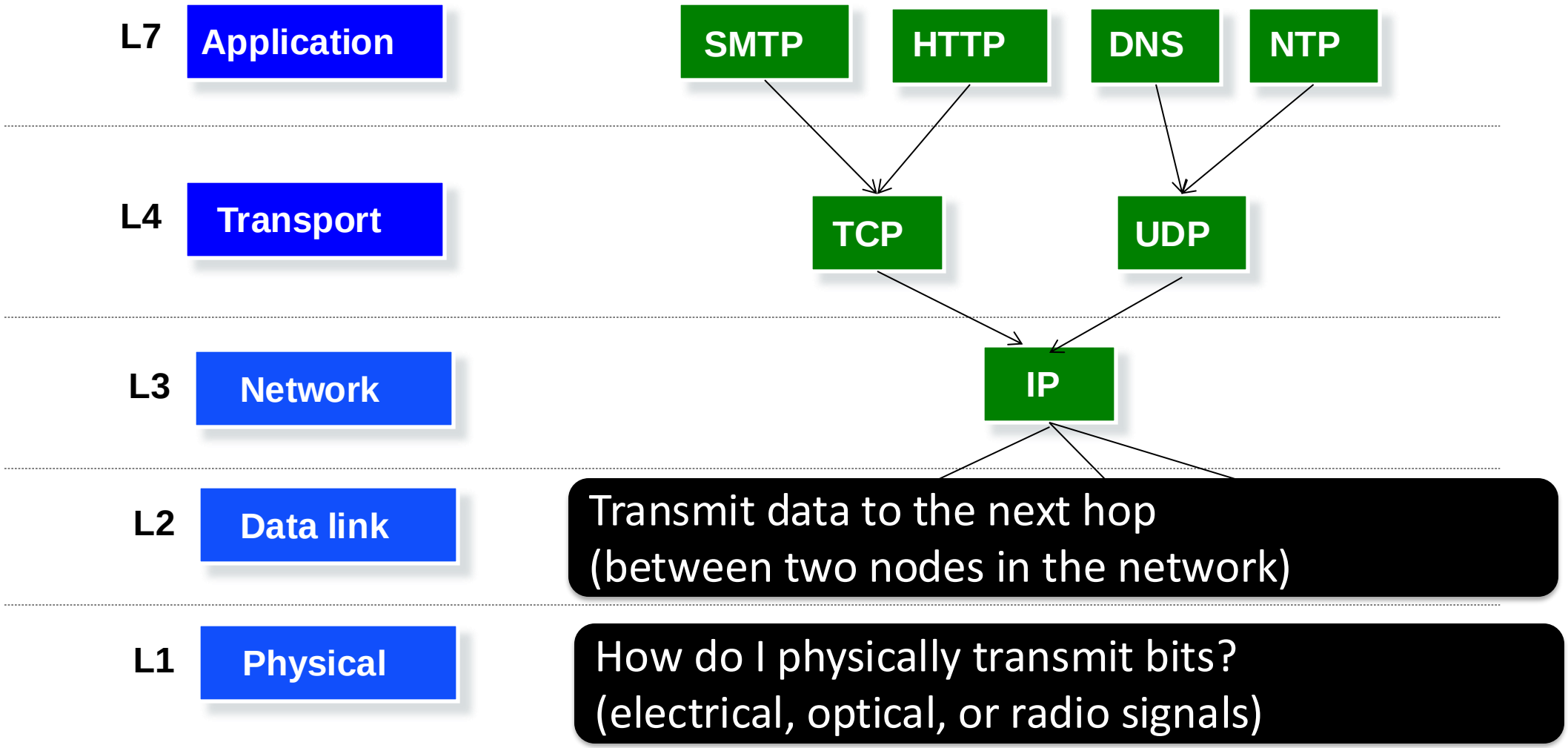
Protocols at Each Layer



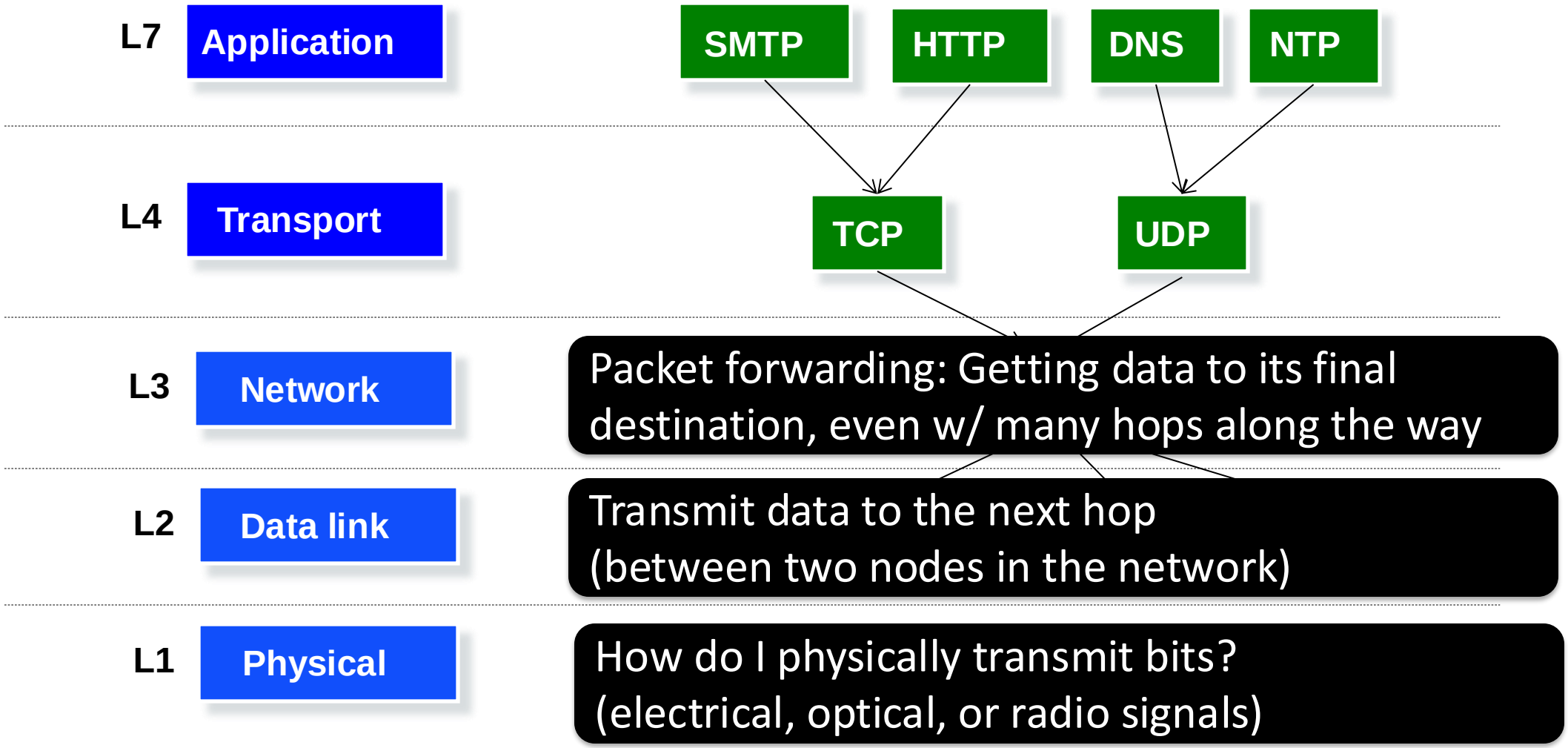
Goals at Each Layer



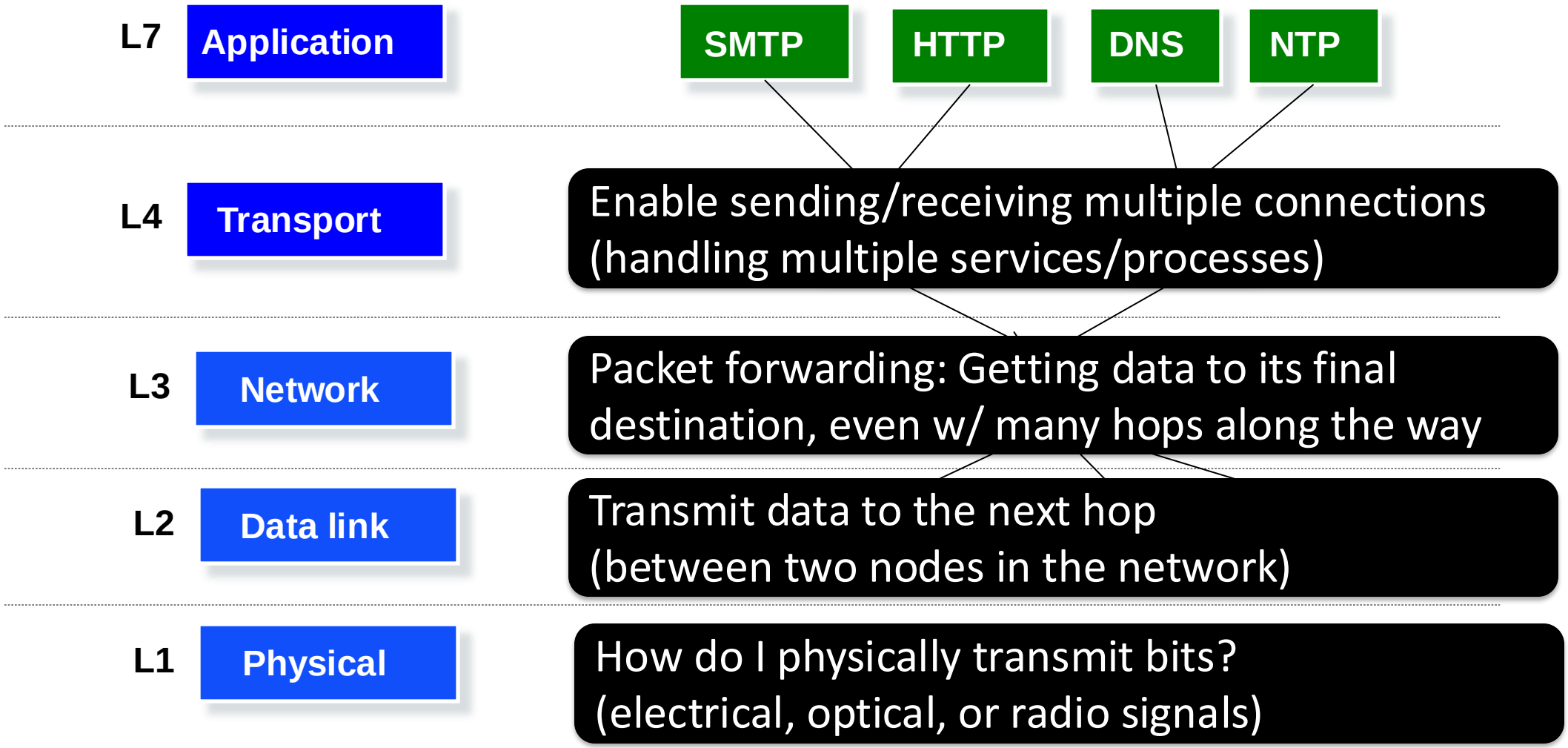
Goals at Each Layer



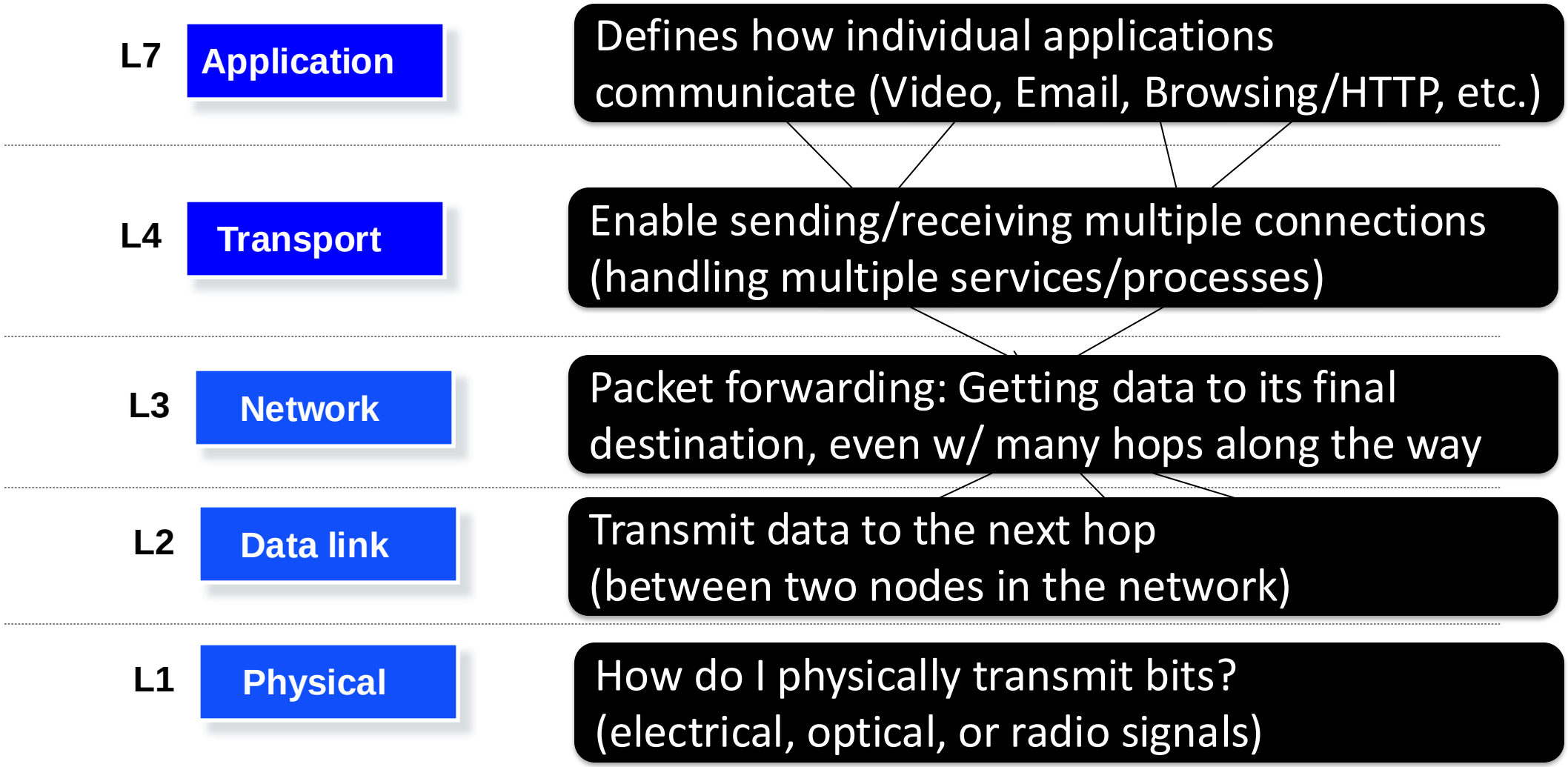
Goals at Each Layer



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)

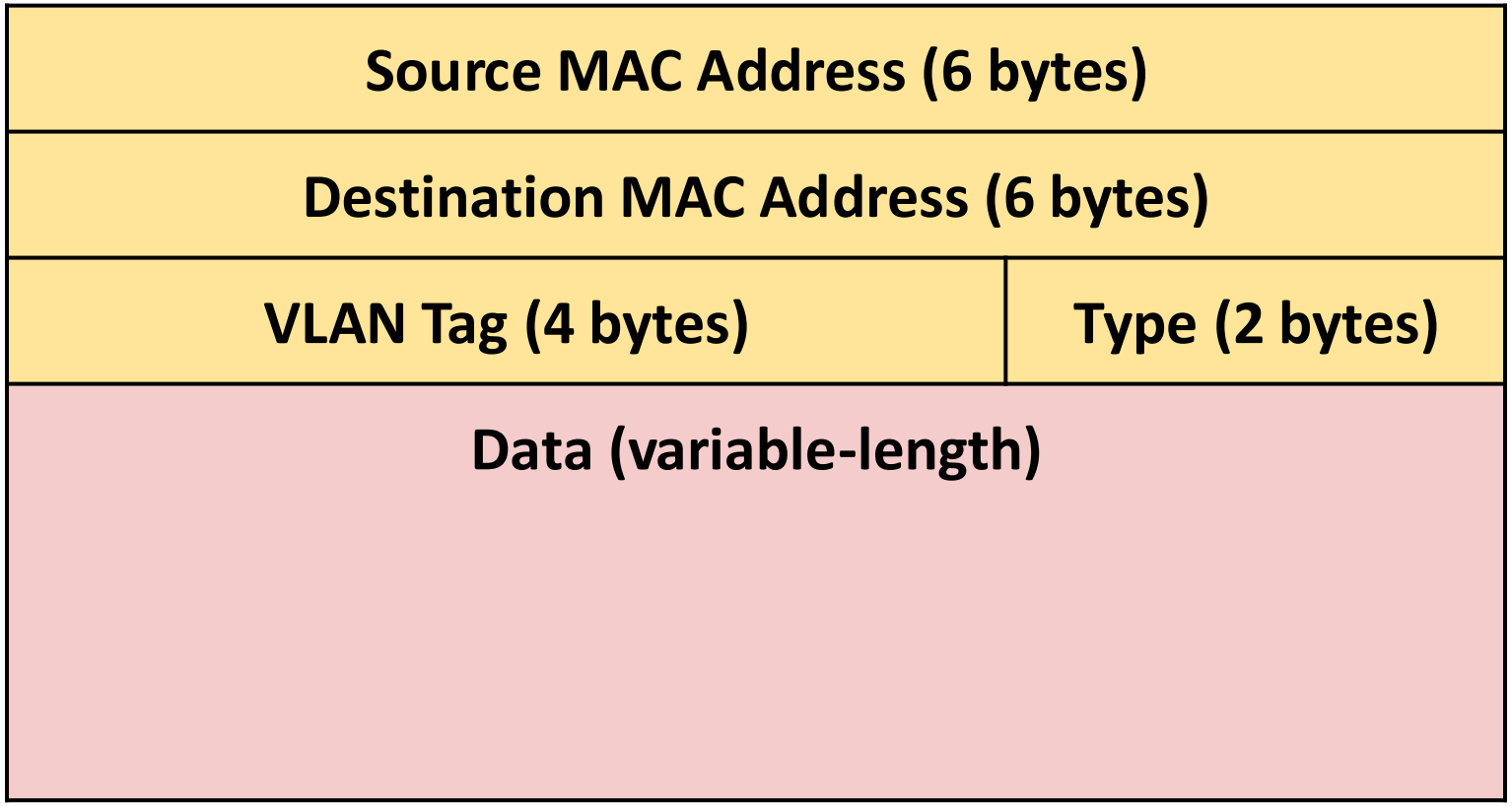
MAC (Media Access Control) Address

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

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

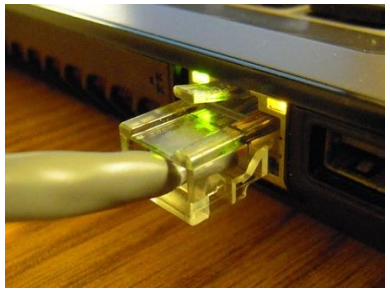
- Usually assigned by manufacturers
  - In theory, doesn't ever change for a piece of hardware
  - In practice, MAC addresses can be spoofed
- See *ifconfig* and similar commands

Layer 2 Header: Add Metadata to Every Message Frame



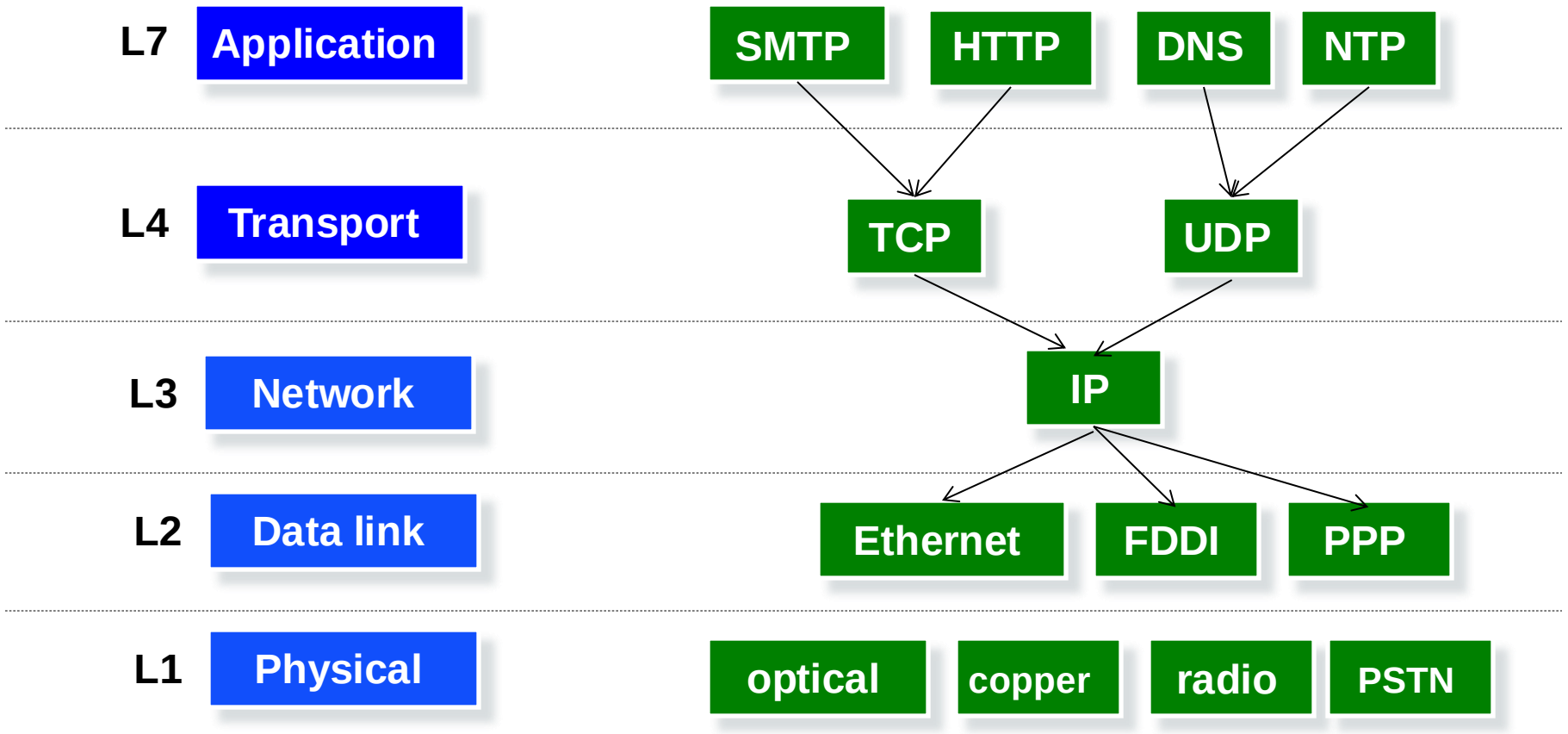
Ethernet and MAC address headers

MAC Addresses Used on Link Layer

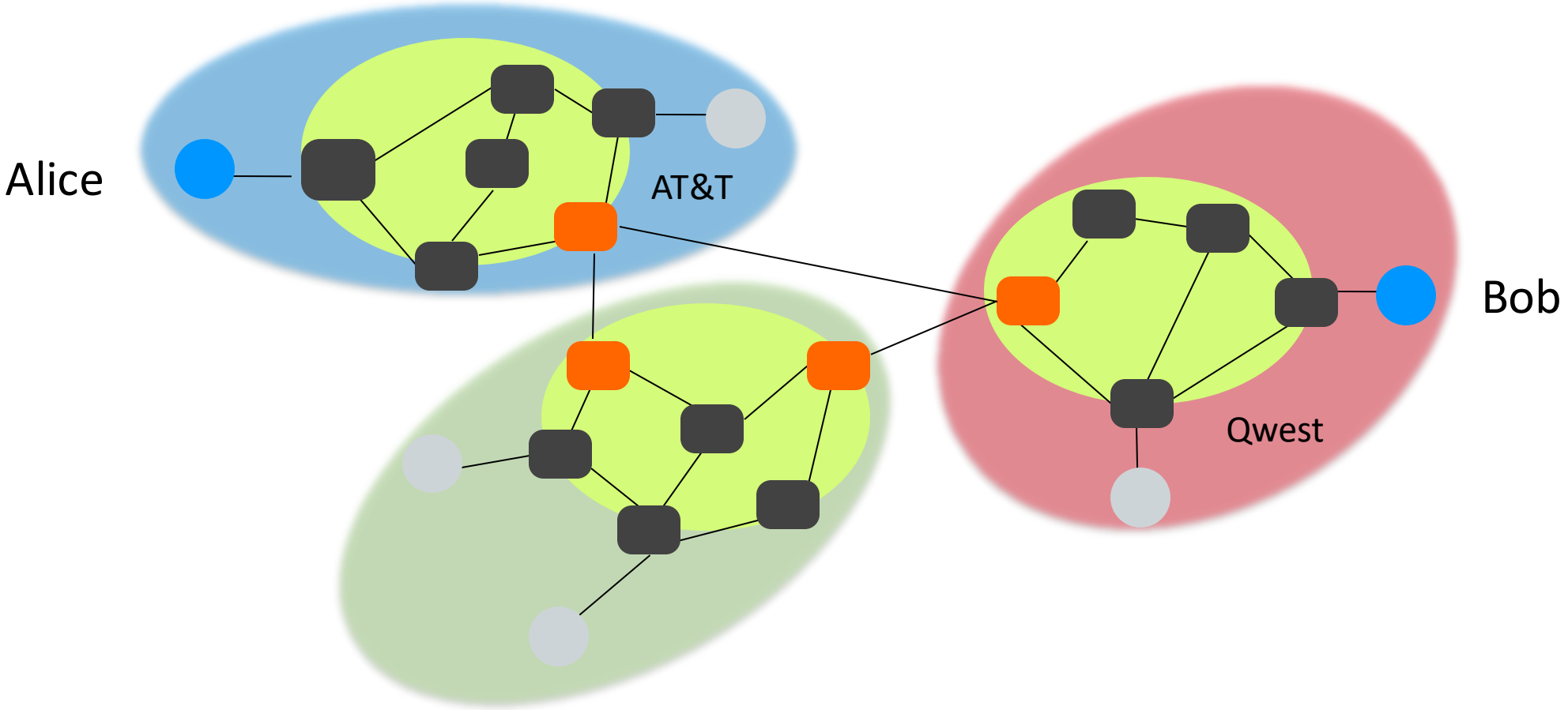


- Ethernet (plugged in)
  - Some hardware (e.g., hubs) repeats all traffic
  - Some hardware (e.g., switches) filters by MAC address
- Wi-Fi (802.11)
  - Your Wi-Fi card typically filters only unicast traffic for you and broadcast traffic (special MAC address: FF:FF:FF:FF:FF:FF)
  - Exception: promiscuous/monitor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

Protocols at Different Layers

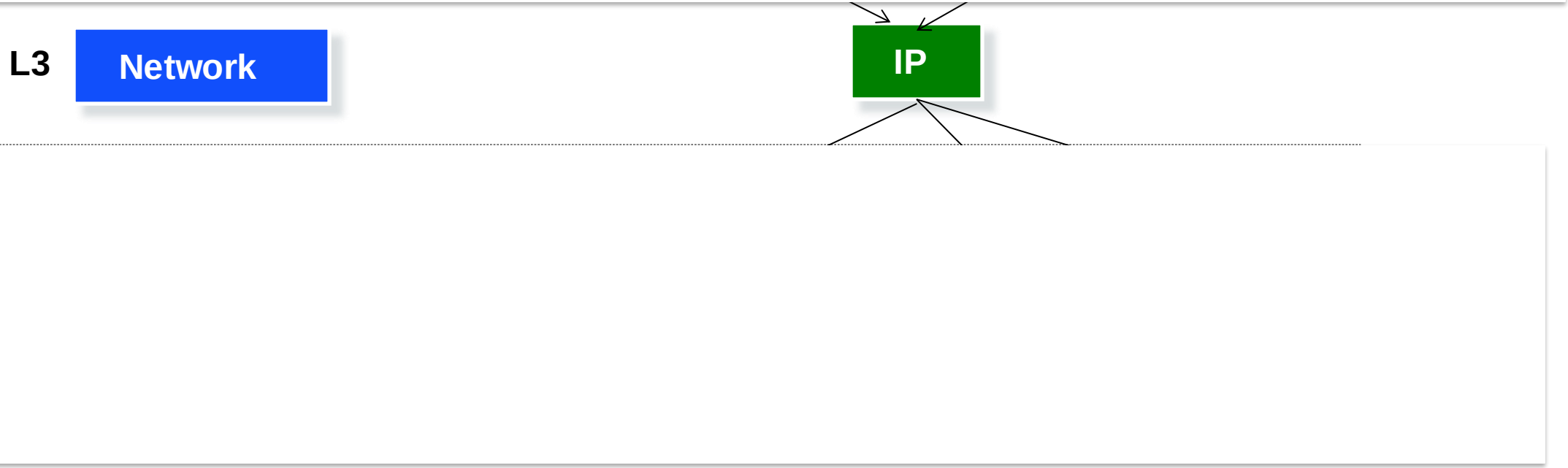


The Internet From 10,000 Feet



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

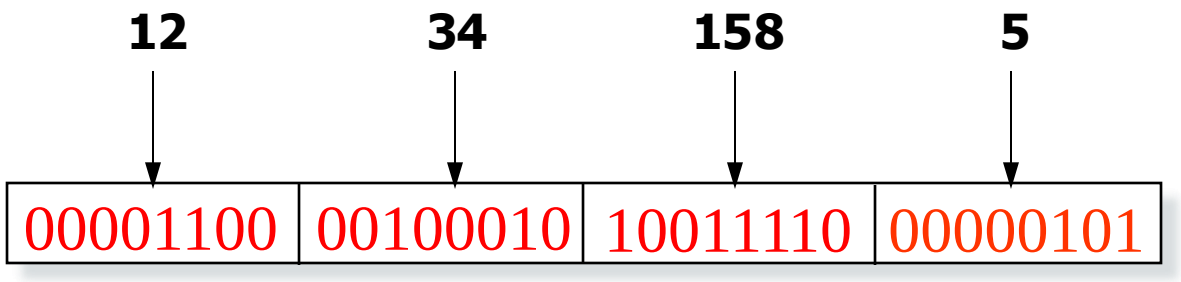
Rest of the packet should be ignored by the router

IP Addresses (IPv4)

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

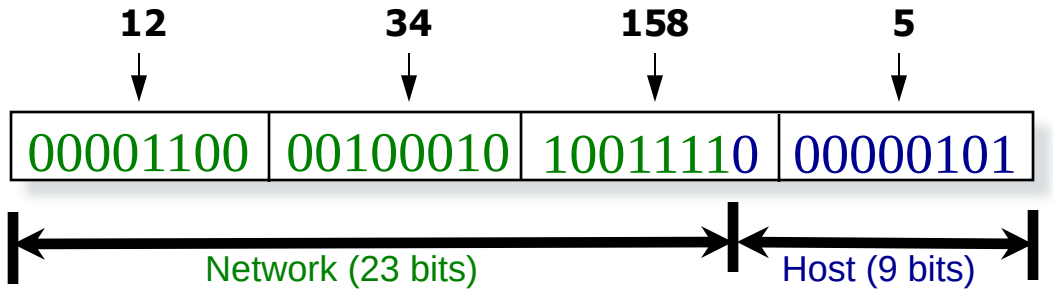
  00001100 00100010 10011110 00000101

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

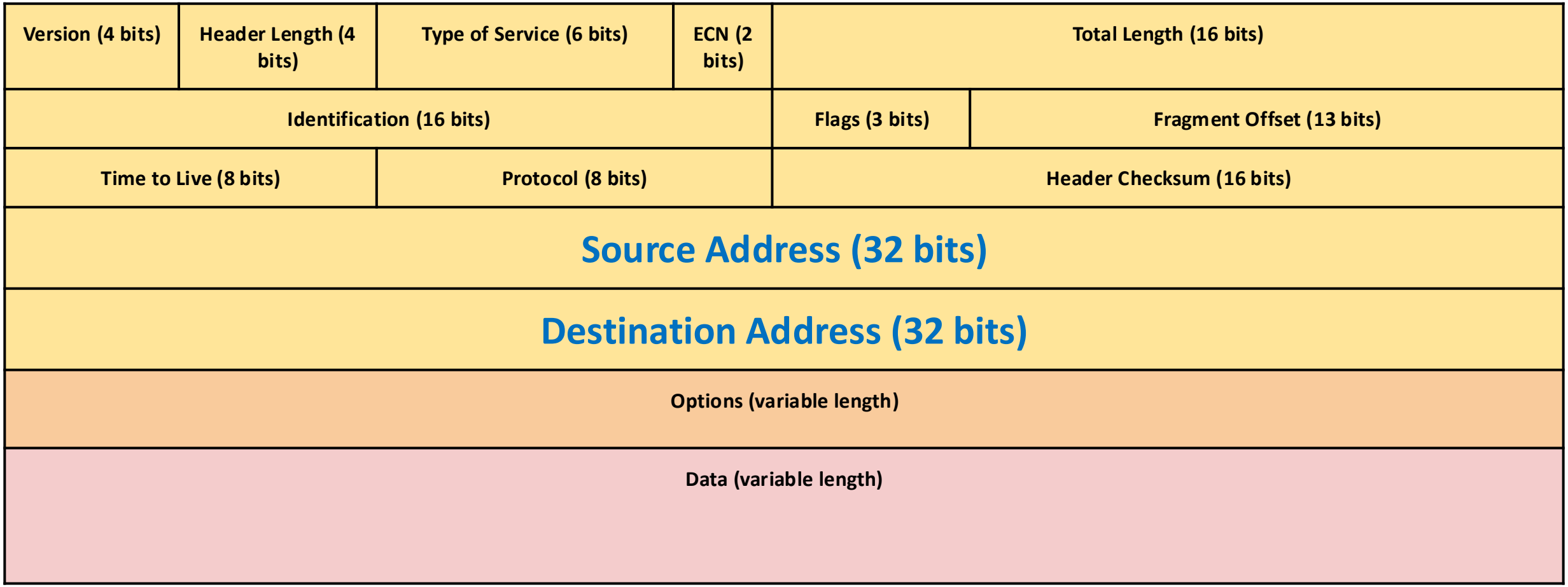


Hierarchy in IP Addressing

- 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

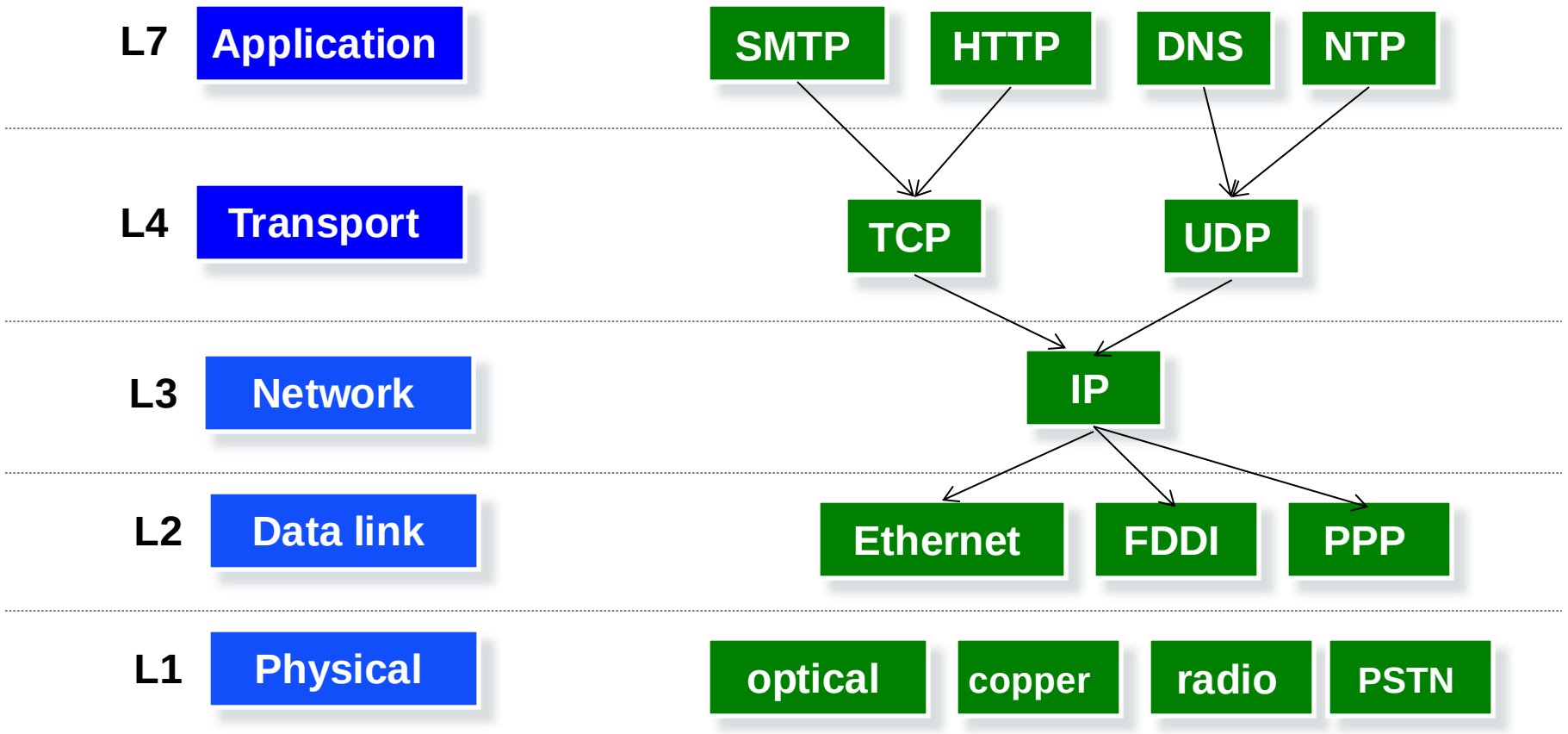


Layer 3 (IP) Header: Add Metadata to Every Packet



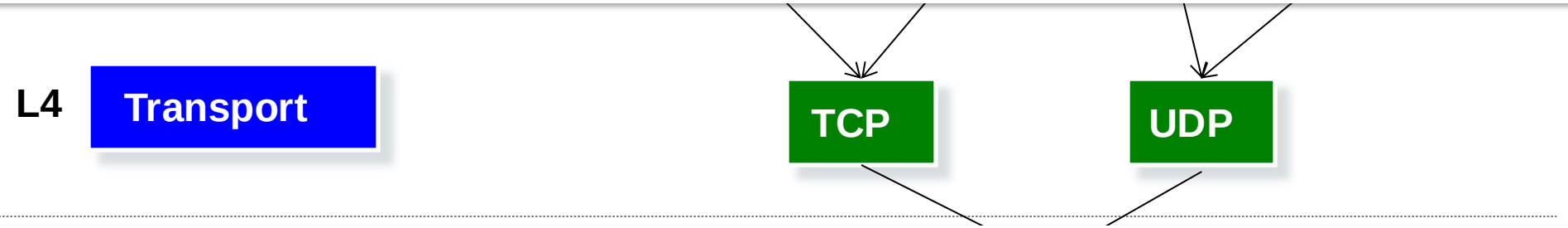
IPv4 header

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

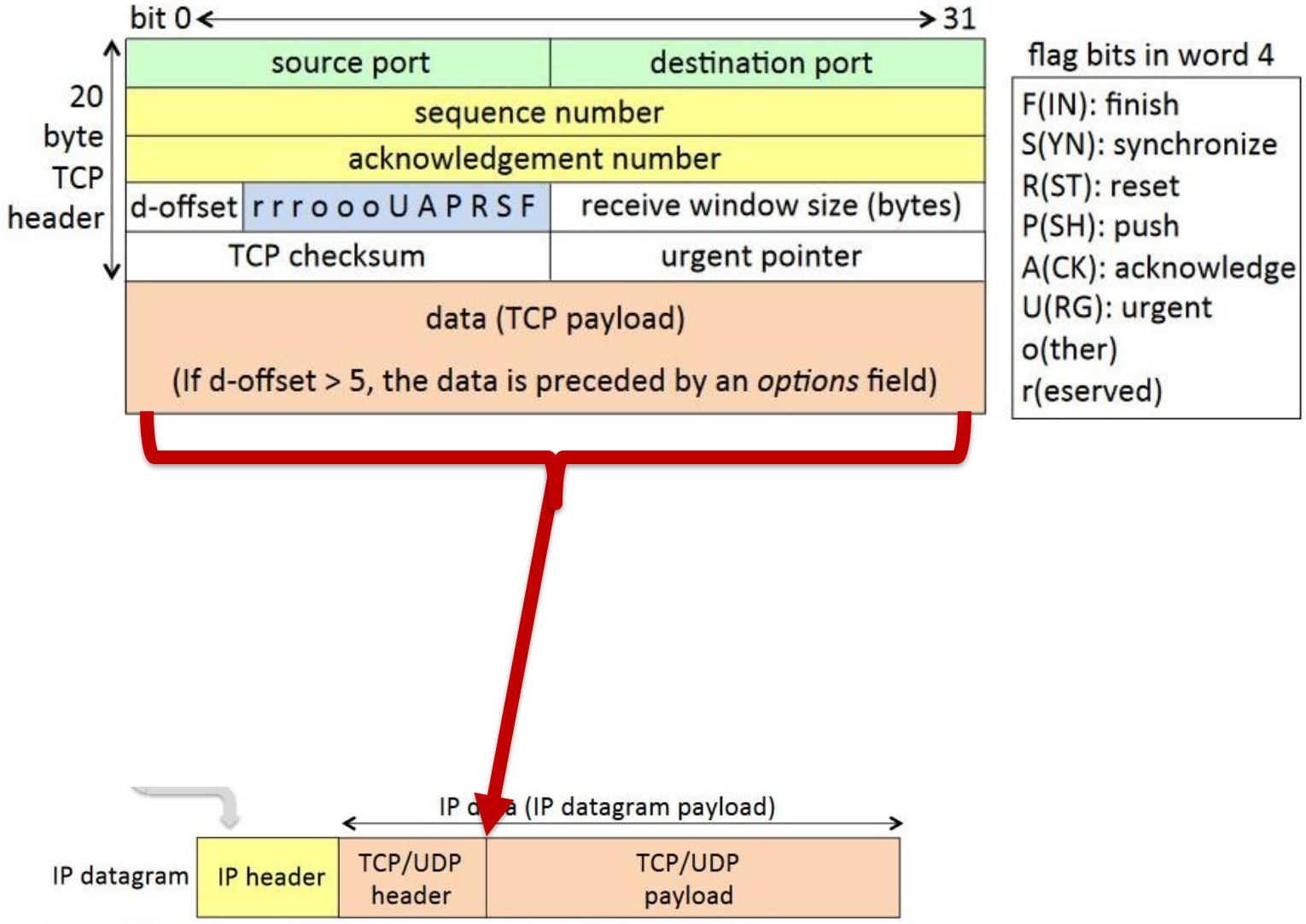
TCP (Transmission Control Protocol)

- 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

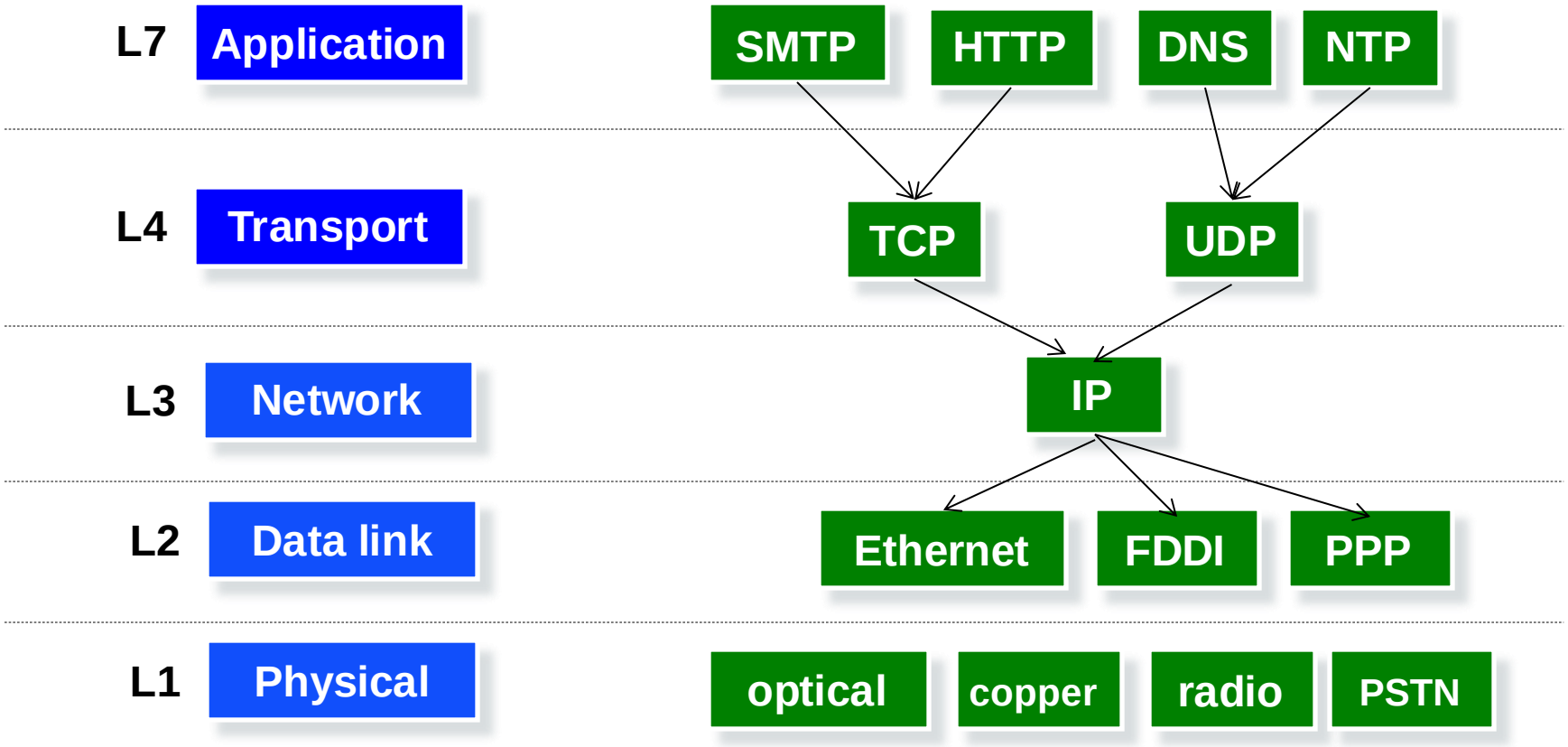
Layer 4 (TCP) Header: Add Metadata to Every Packet



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

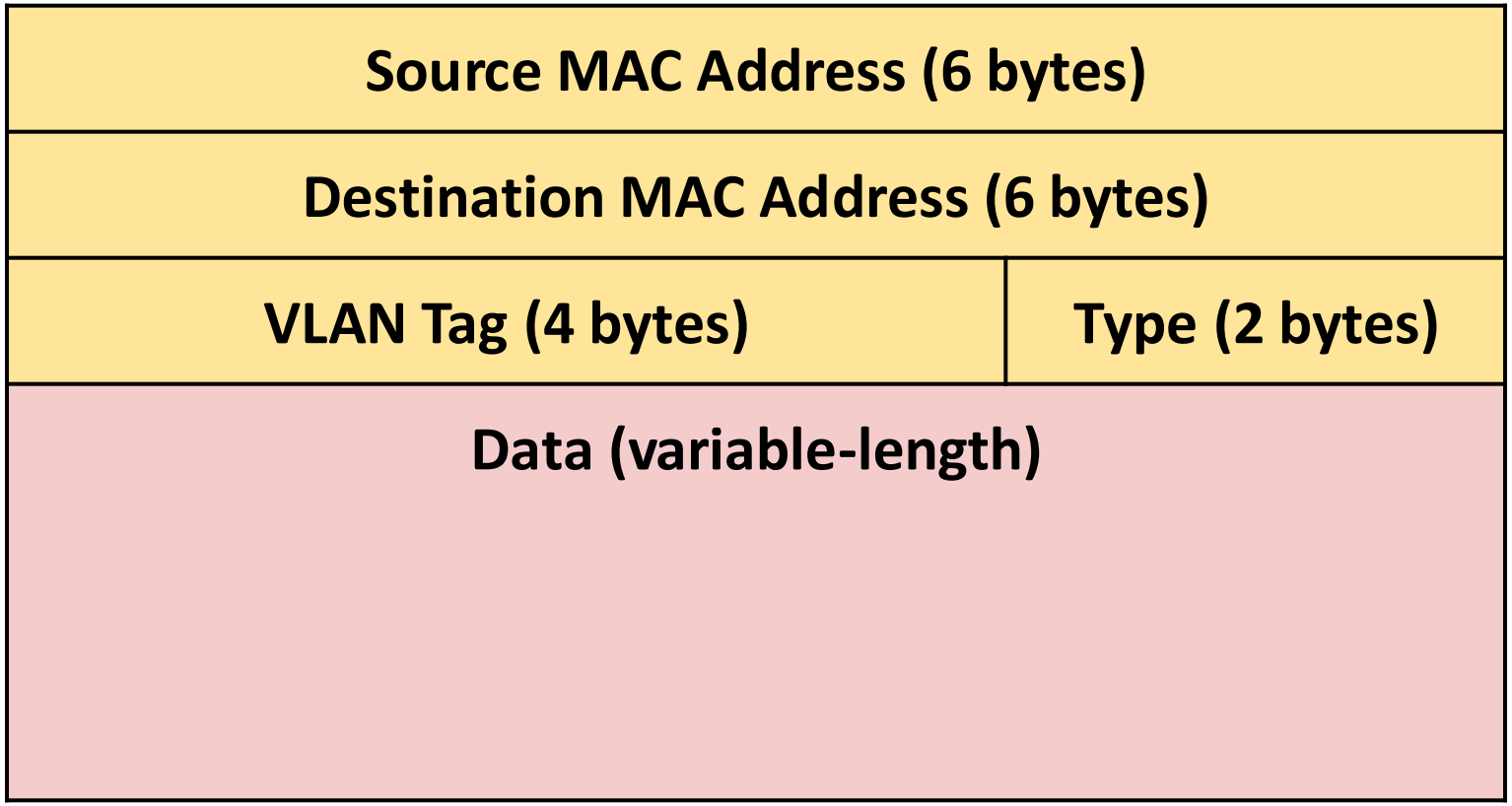
Layers of Abstraction and Headers



Lots of headers across all the layers

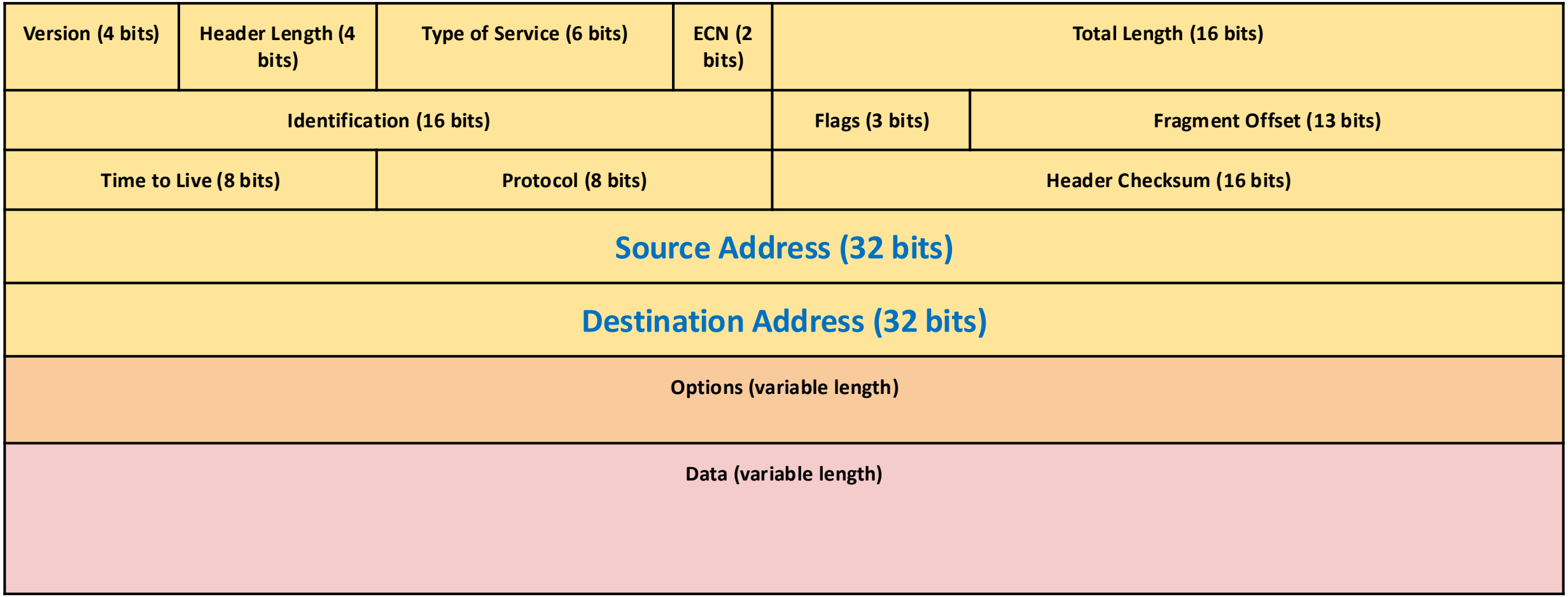
How do we combine & organize them when sending messages?

Layer 2 Header: Send Data b/t Two Hops



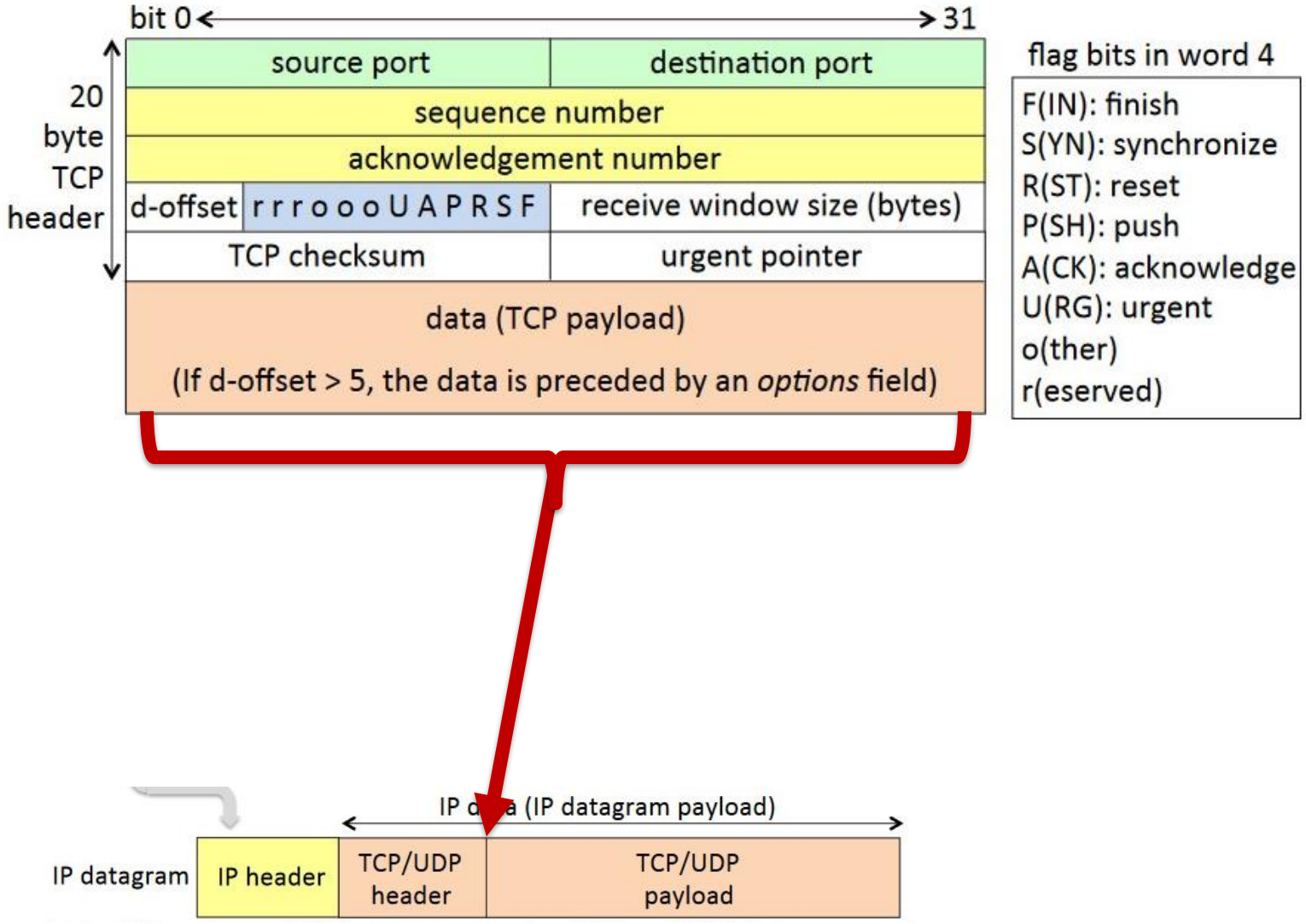
Ethernet and MAC address headers

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



IPv4 header

Layer 4 Header (TCP): Handle multiple connections

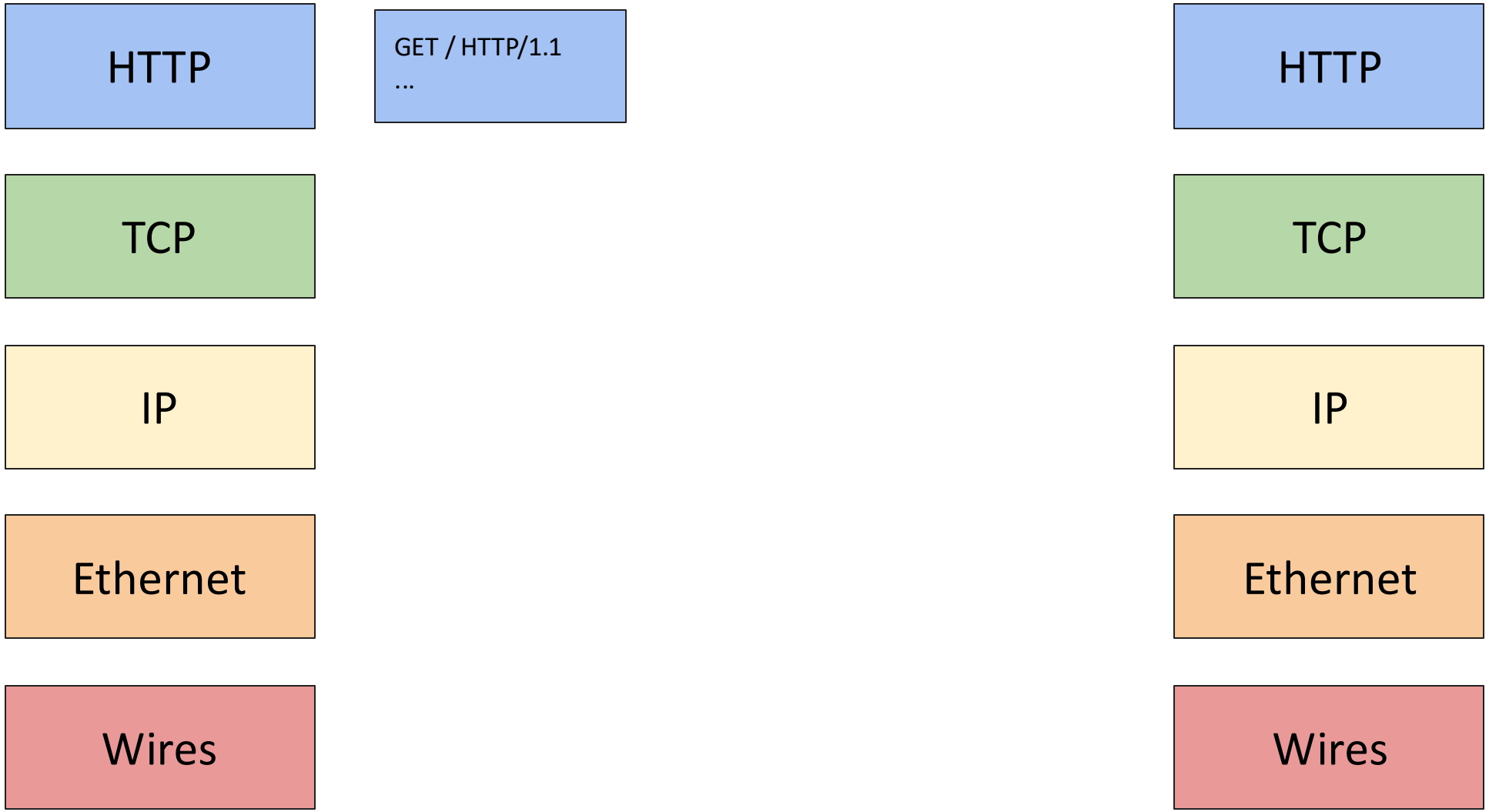


Protocol Header Encapsulation

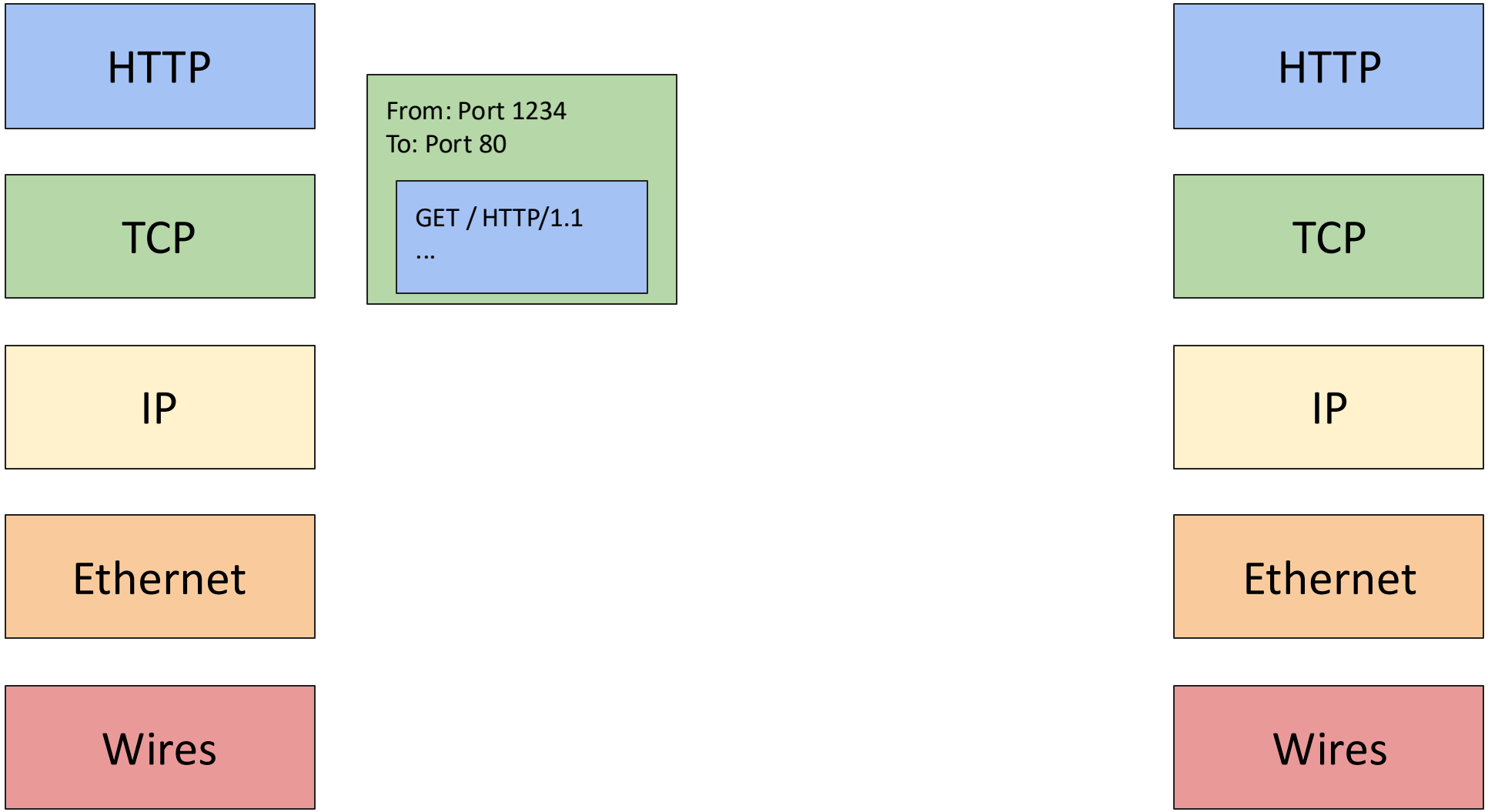
- The application starts with the content (payload) to send
- As the sender's machine constructs the packet to ***send*** to someone else (moving to lower layers), it wraps additional headers around the message

Example: HTTP Request

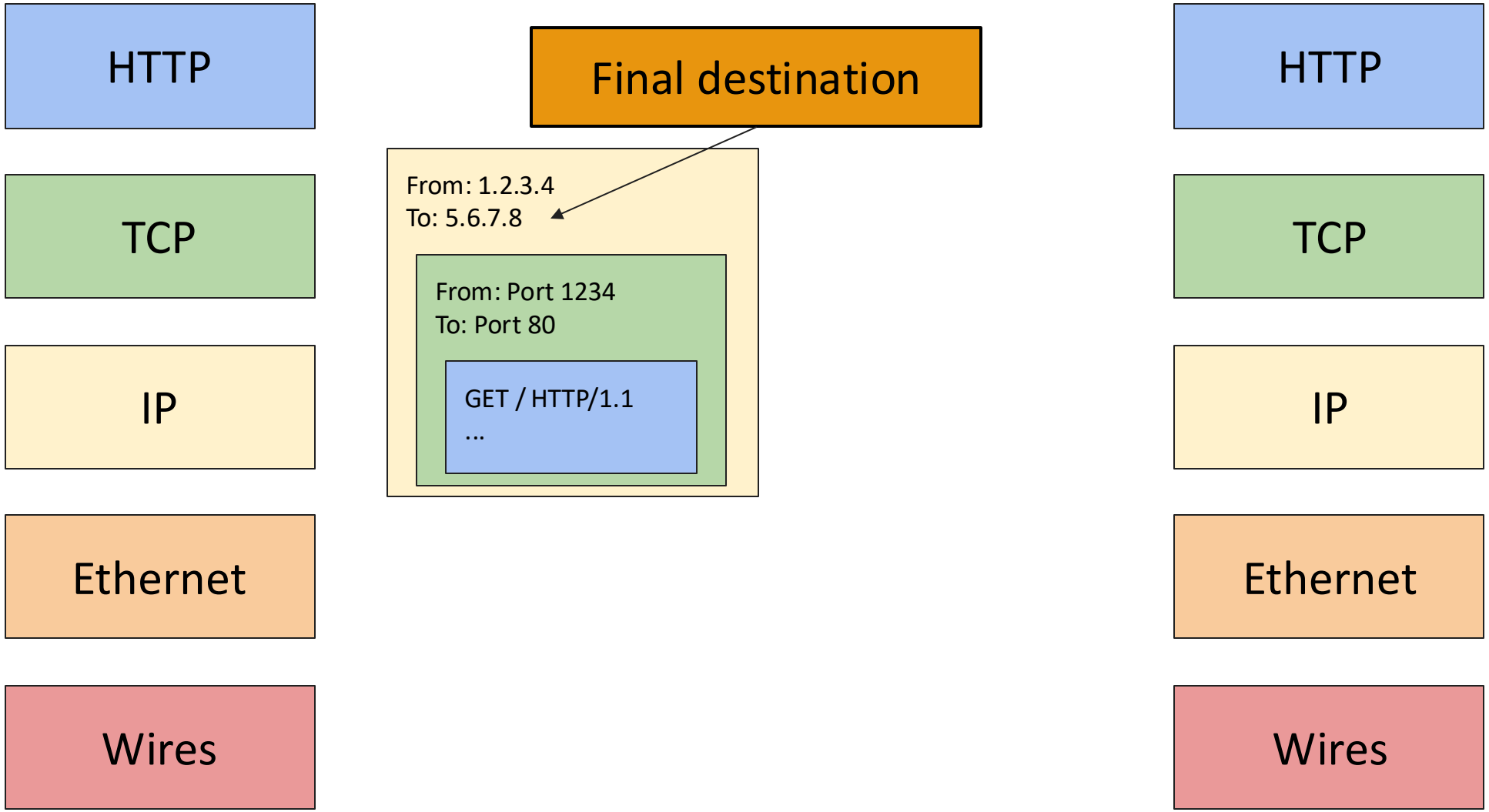
(from Peyrin Kao)



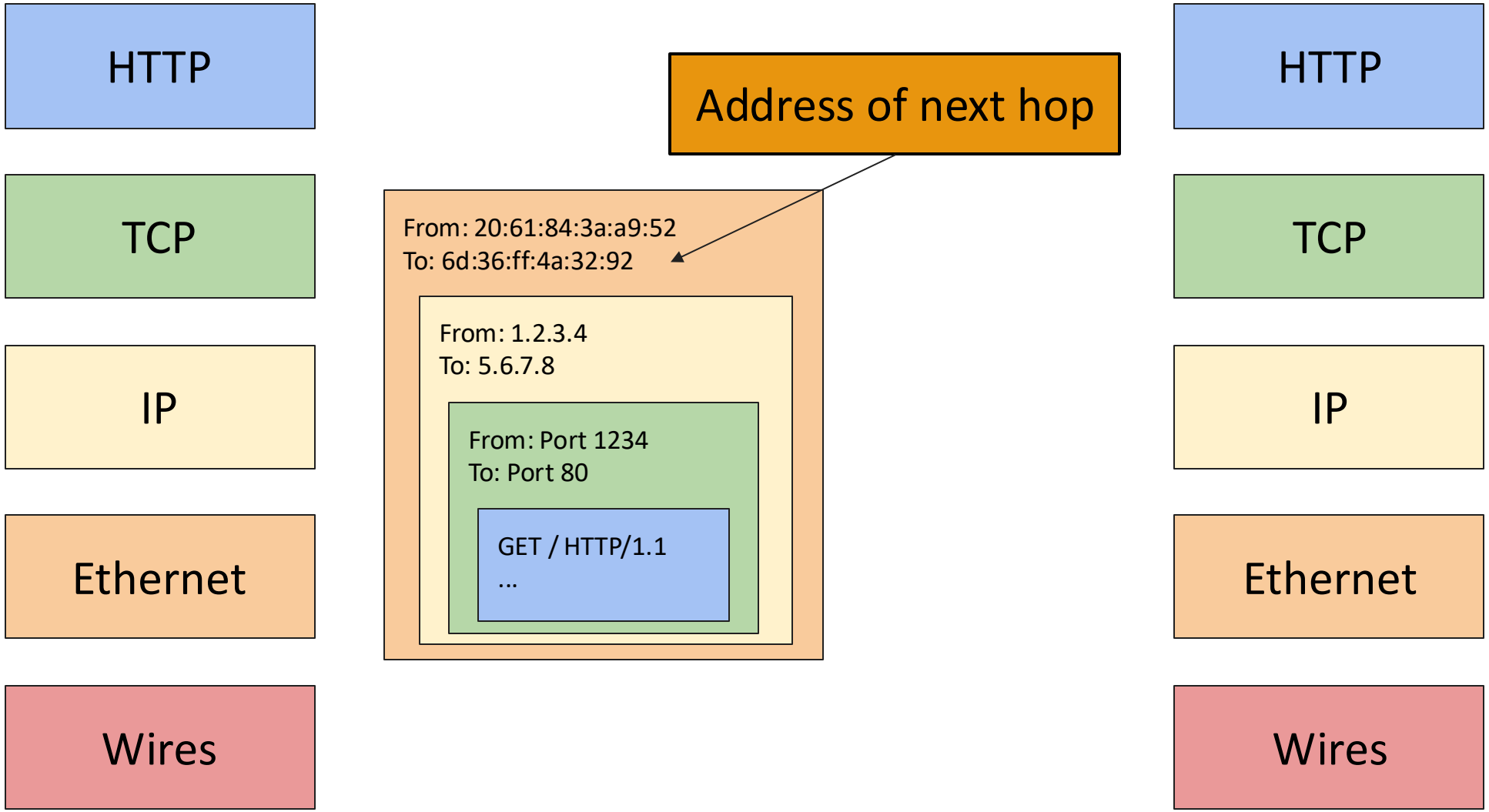
Example: HTTP Request



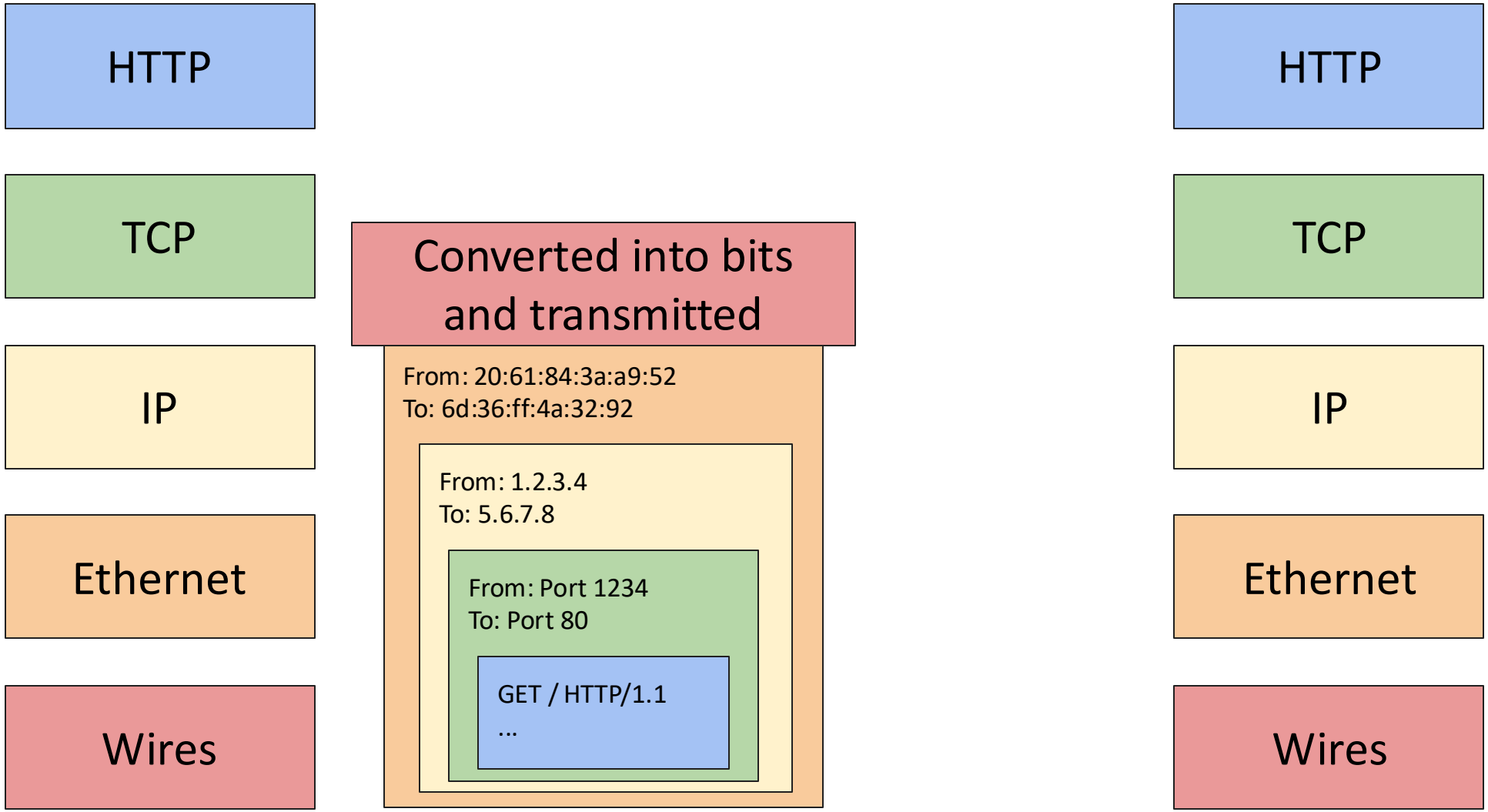
Example: HTTP Request



Example: HTTP Request



Example: HTTP Request

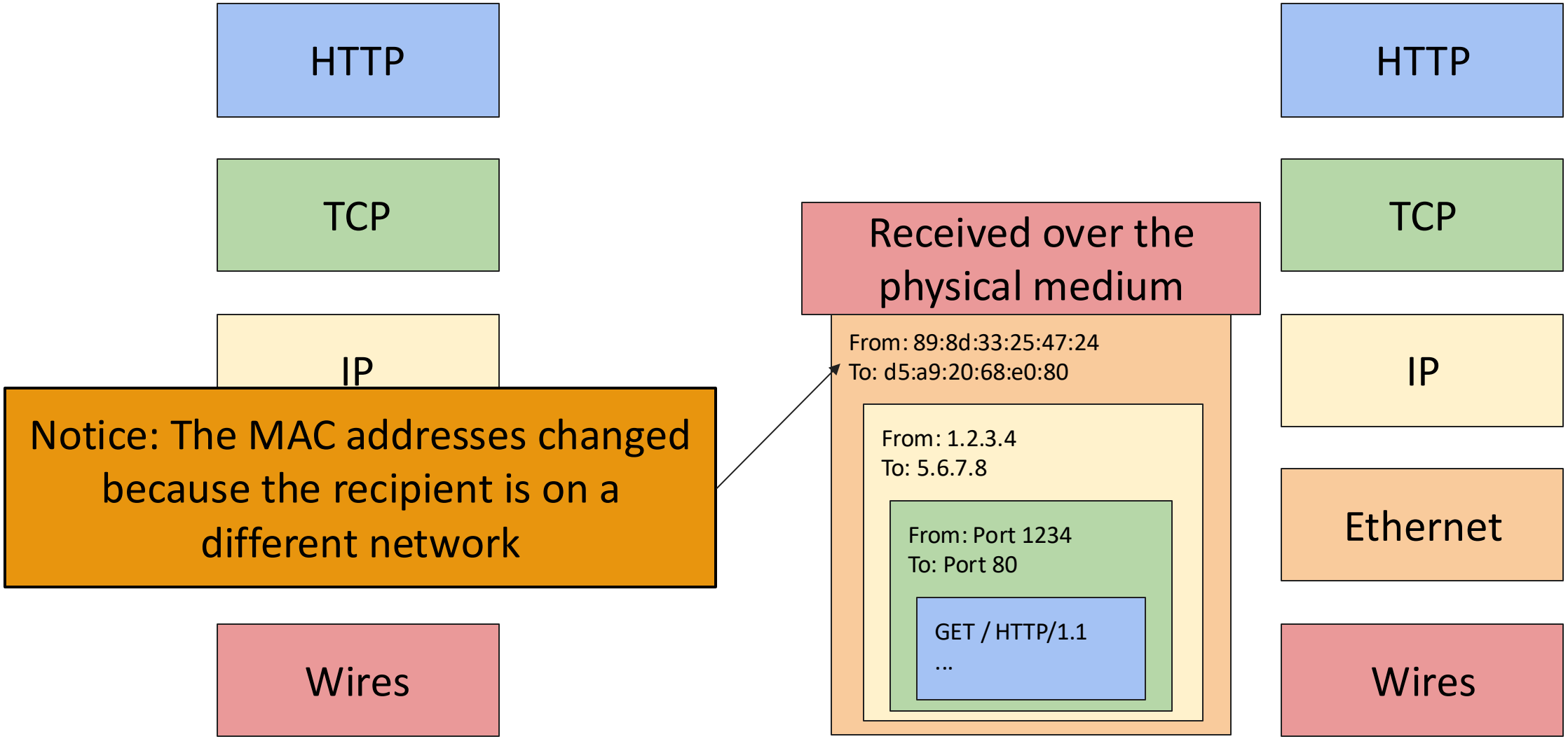


Layers of Abstraction and Headers

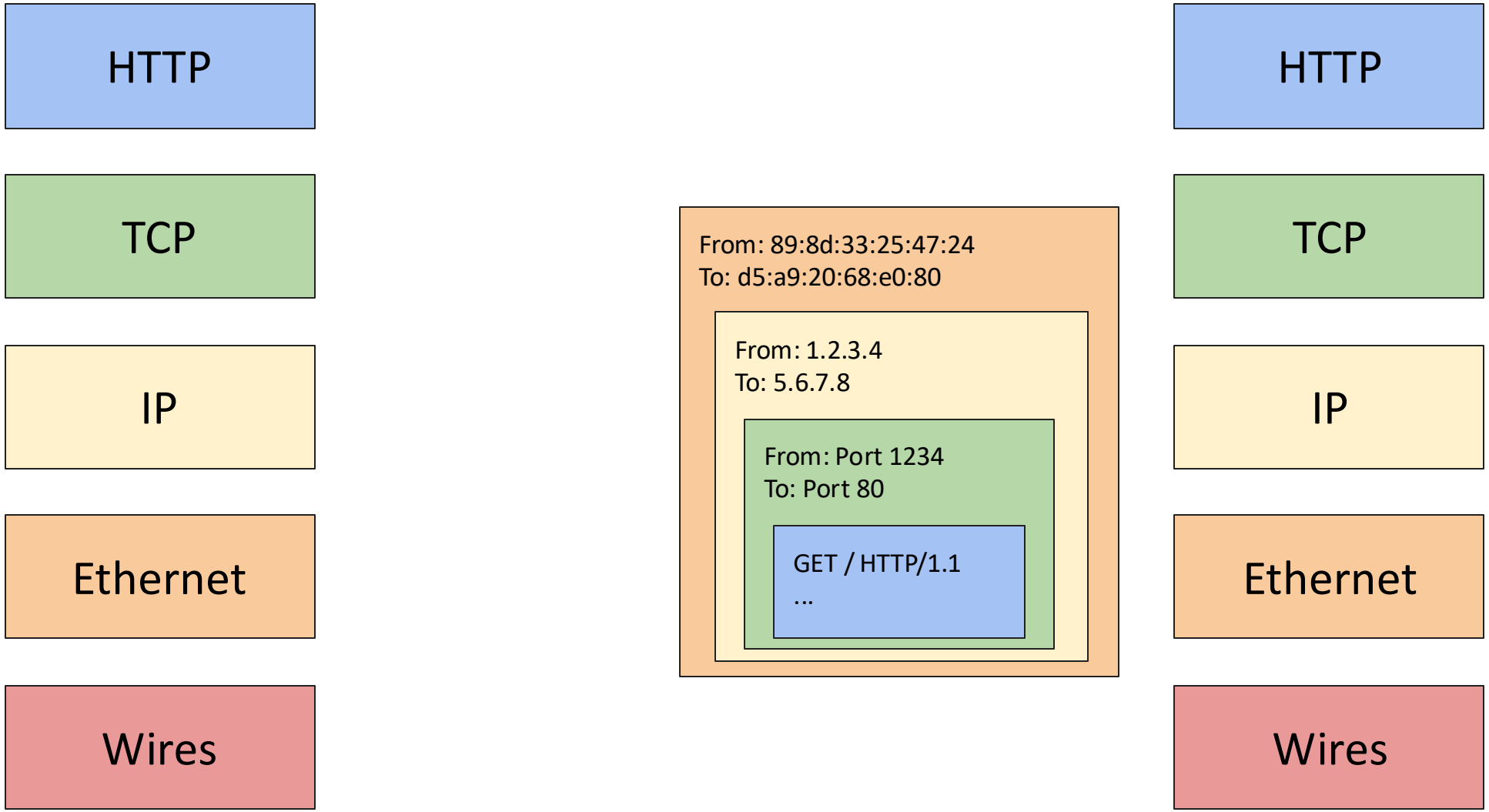
- The application starts with the content (payload) to send
- As the sender's machine constructs the packet to ***send*** to someone else (moving to lower layers), wrap additional headers around the message

- 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)

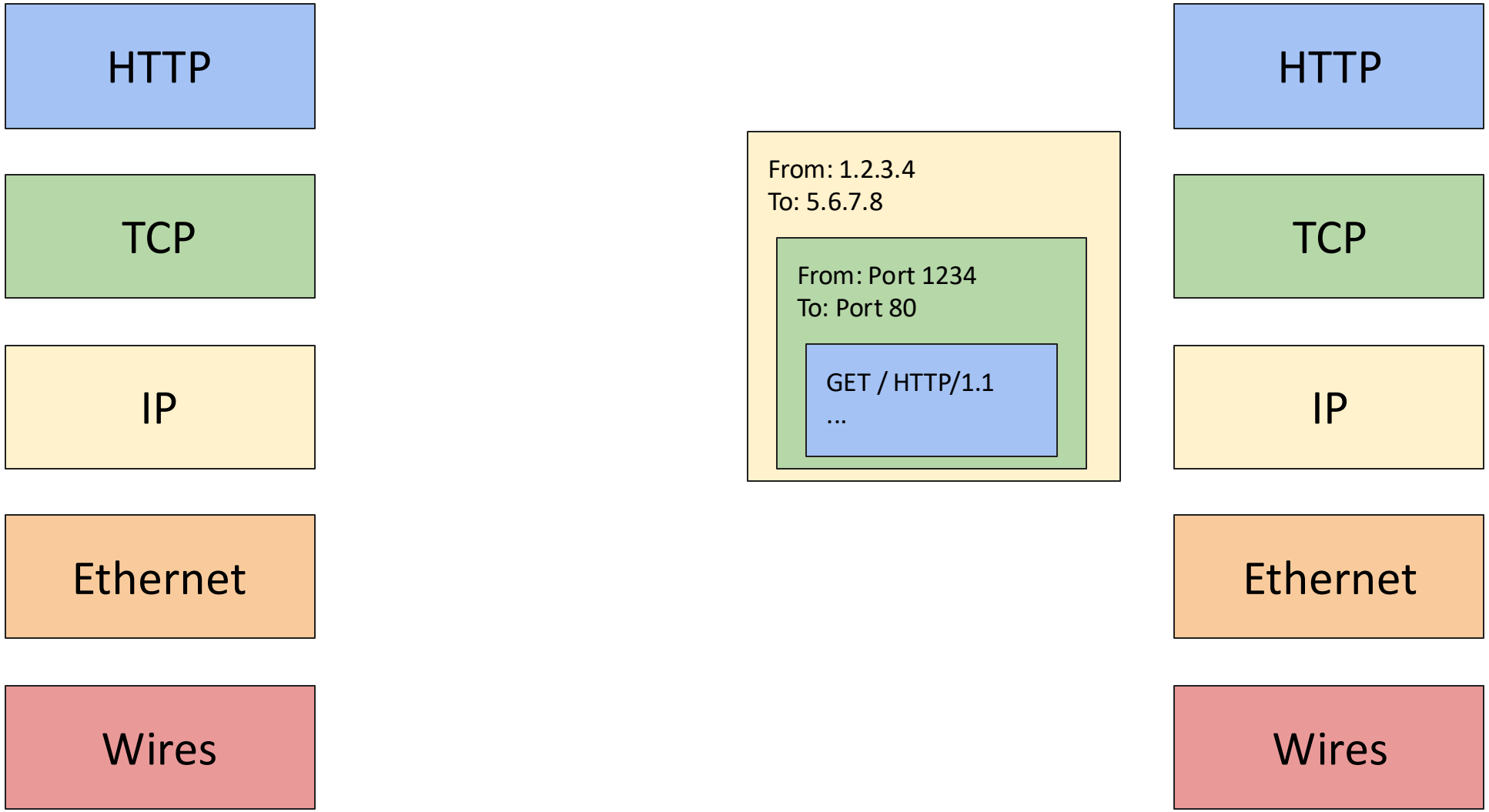
Example: HTTP Request



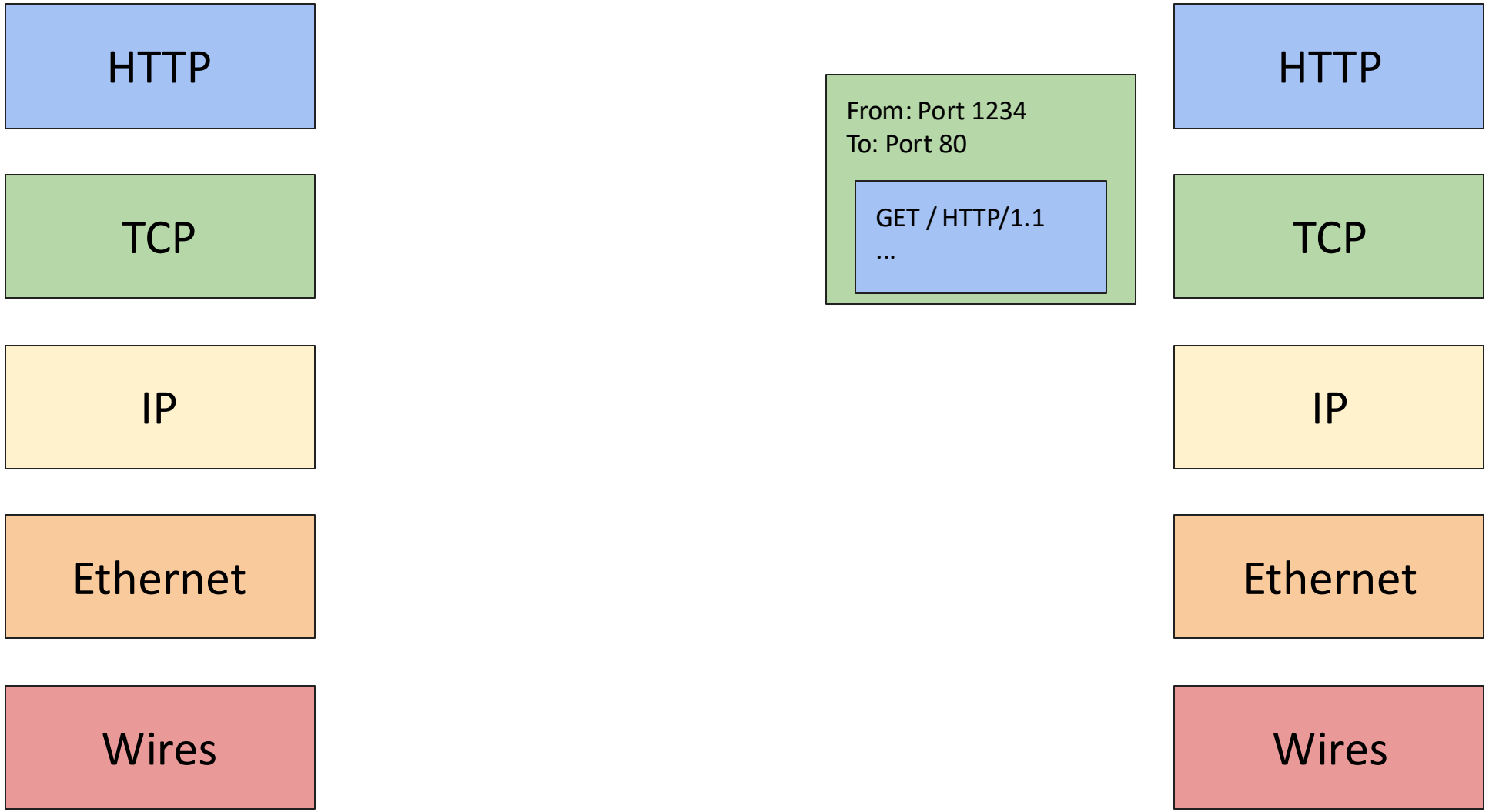
Example: HTTP Request



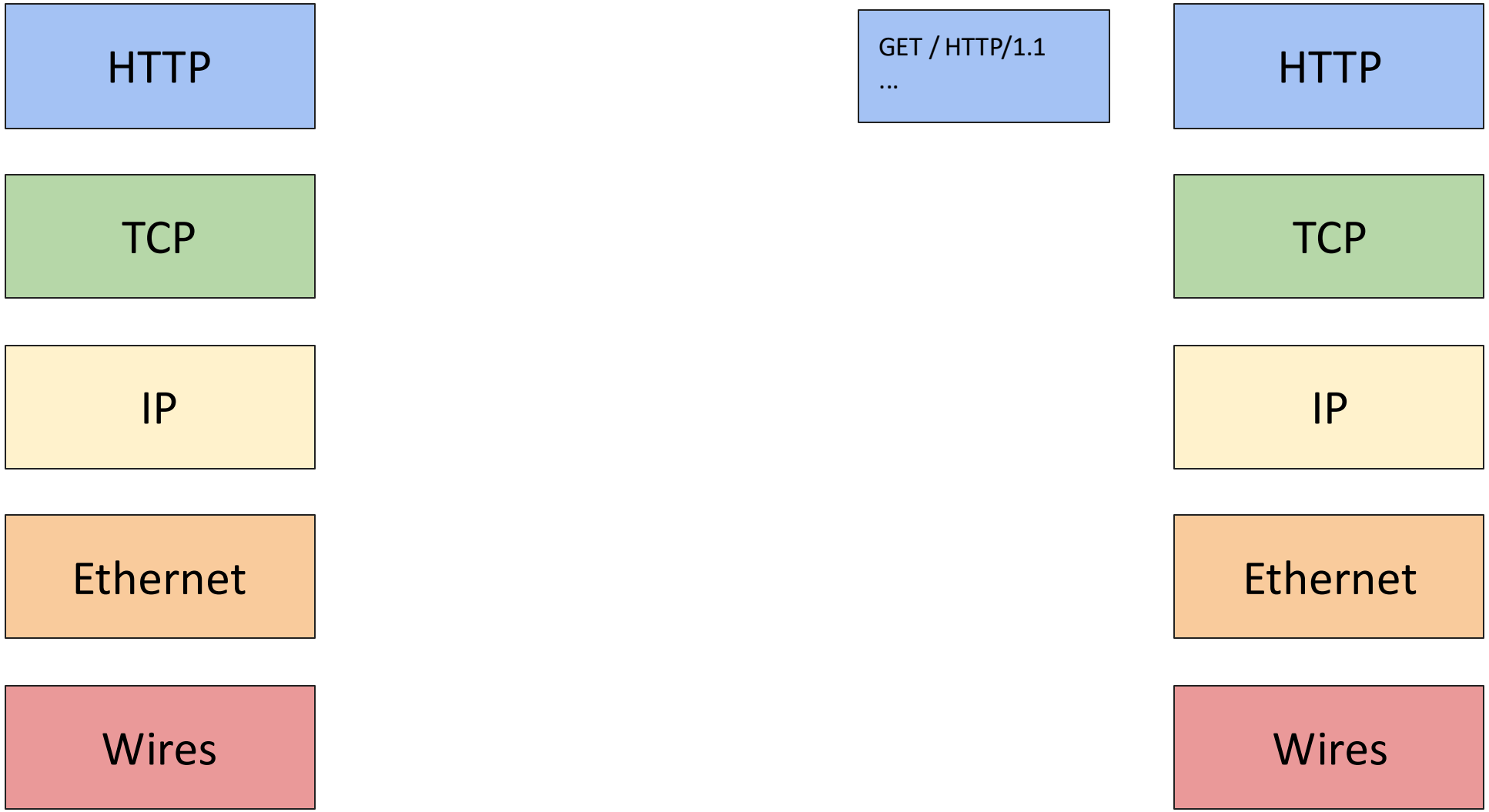
Example: HTTP Request



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