CSCI 466: Networks

Review

Reese Pearsall

Fall 2022

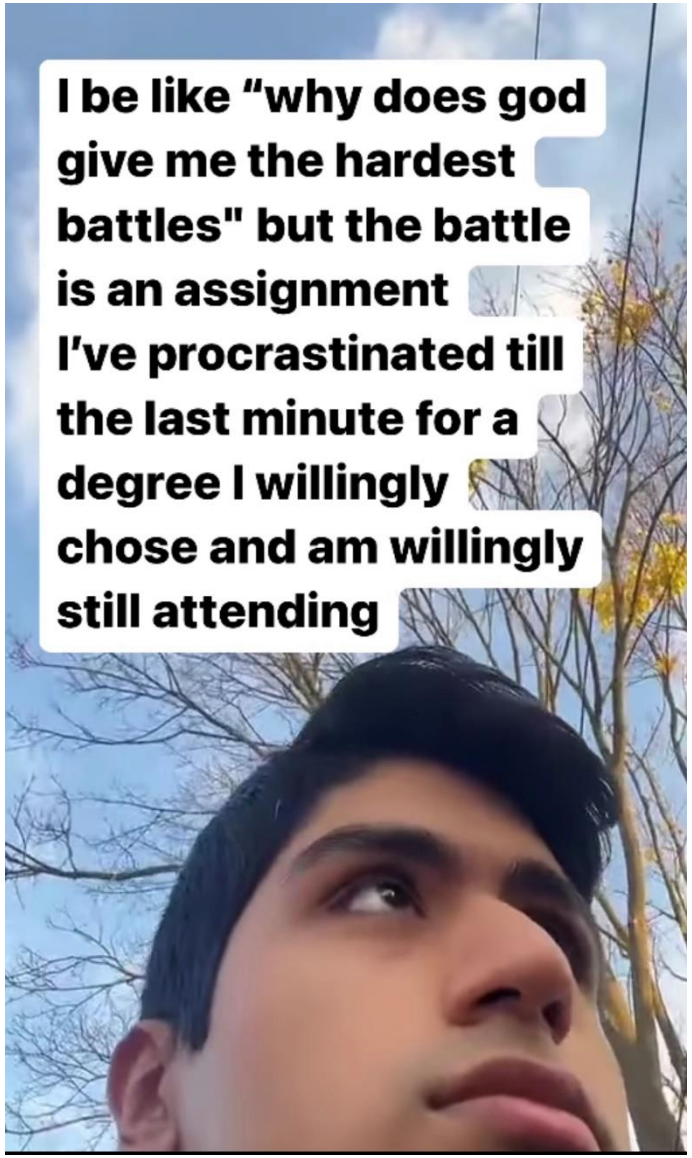
Announcements

No lecture on Wednesday. Workday for PA4. I'll be in Reid 202 during class time if you need help

Homework 4 due **Friday** @ 11:59 PM

PA4 due **Sunday** @ 11:59 PM (We are currently at 49% response rate)

Final Exam: Wednesday December 14th 12:00 – 1:50 PM (same room)



Final Exam Structure

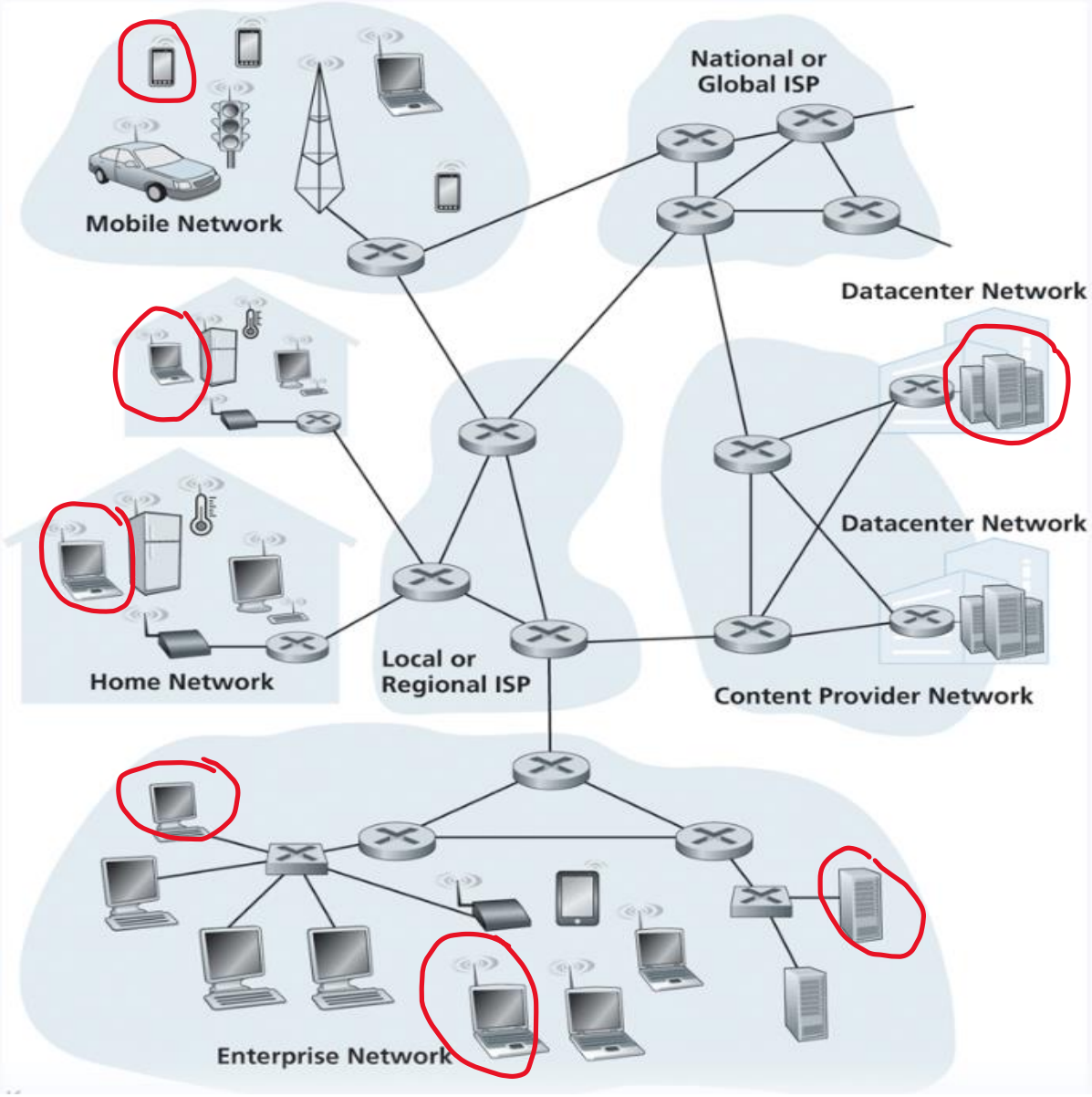
No notes allowed
A curve won't be applied (unless needed)
15% of your grade
Please show up

Part I. OSI Model
For each layer
- Name the layer (Ex. Network Layer)
- Provide a primary responsibility/functionality (Ex. Forwarding and Routing)
- Provide the unit of data that is being transmitted (Ex. Datagram)

Part II. Short answer questions (~18)
- I will pull some of these questions straight from the homeworks (HW1, HW2, HW3)

You can expect 2-4 questions from **each** the following sections
- Internet Structure
- Application Layer
- Transport Layer
- Network Layer (Data Plane)
- Network Layer (Control Plane)
- Link Layer
- Network Security

Internet Structure



Devices that are connected to network are called **hosts** or **end systems**

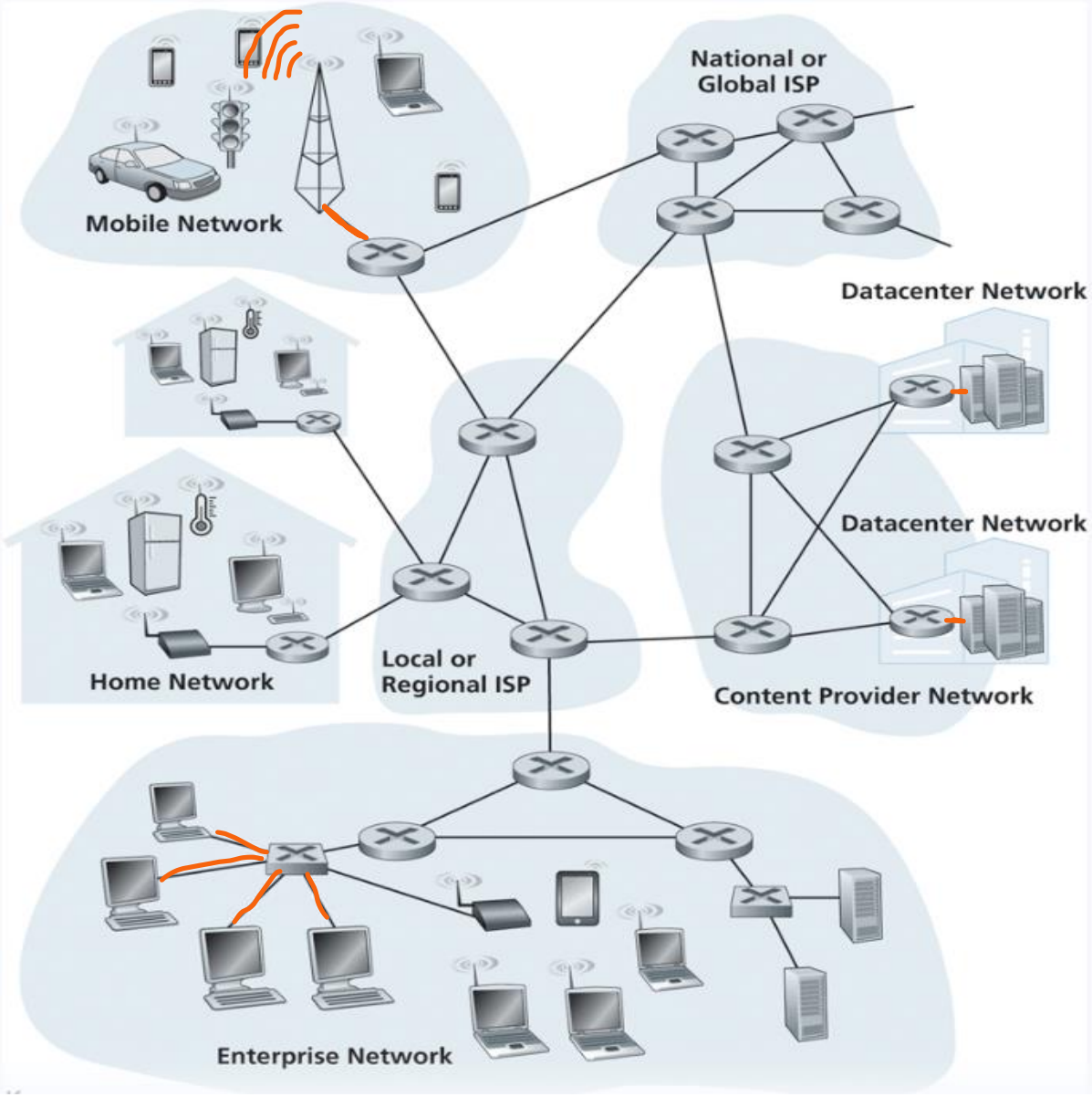
- **Two types: Clients and Servers**

End systems are connected together by a network of **communication links** and **packet switches/routers**

The **network edge** consists of end systems

The **network core** consists routers and other packet switches (usually owned by an ISP)

Internet Structure



Devices that are connected to network are called **hosts** or **end systems**

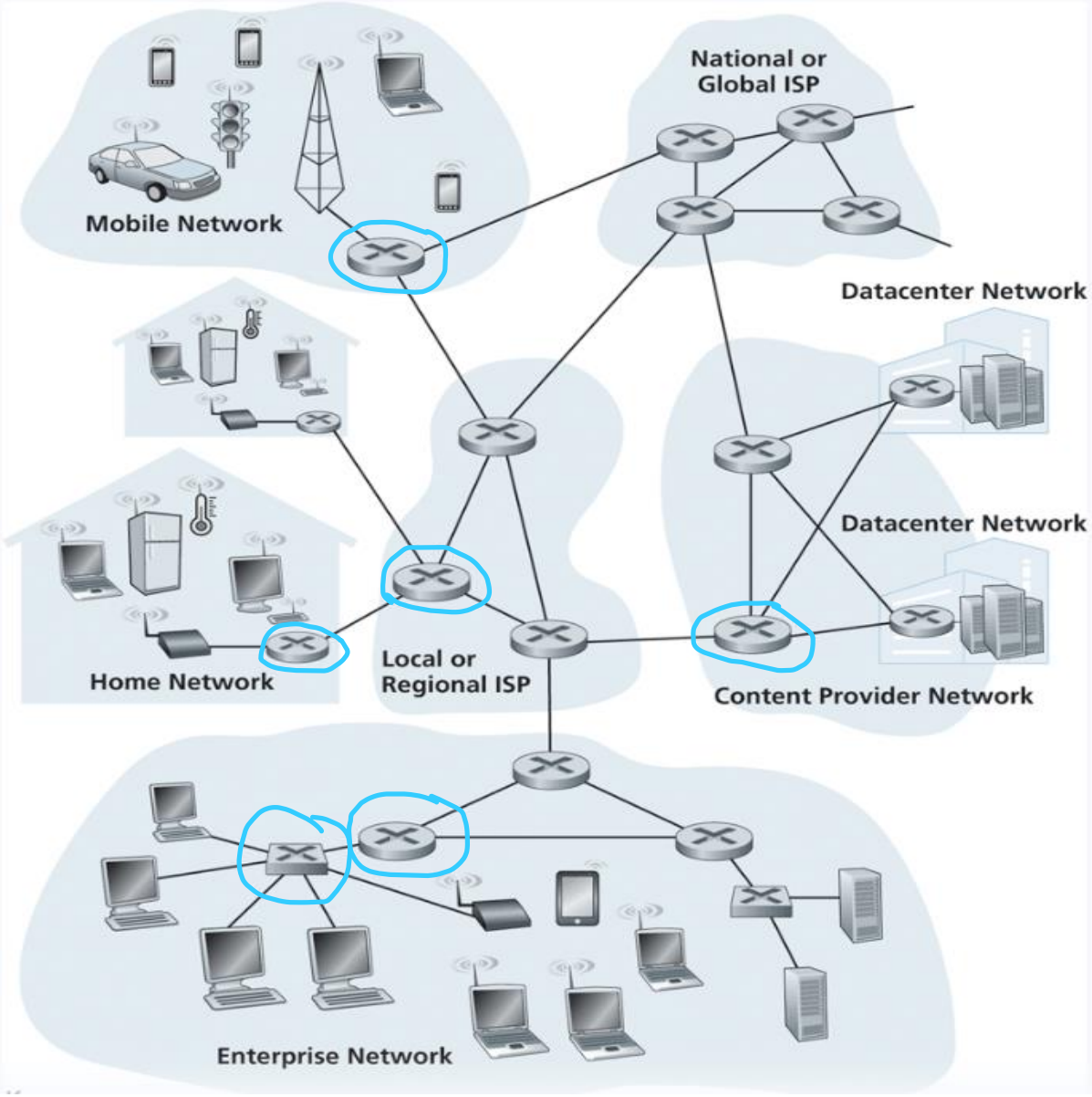
- **Two types: Clients and Servers**

End systems are connected together by a network of **communication links** and **packet switches/routers**

The **network edge** consists of end systems

The **network core** consists routers and other packet switches (usually owned by an ISP)

Internet Structure



Devices that are connected to network are called **hosts** or **end systems**

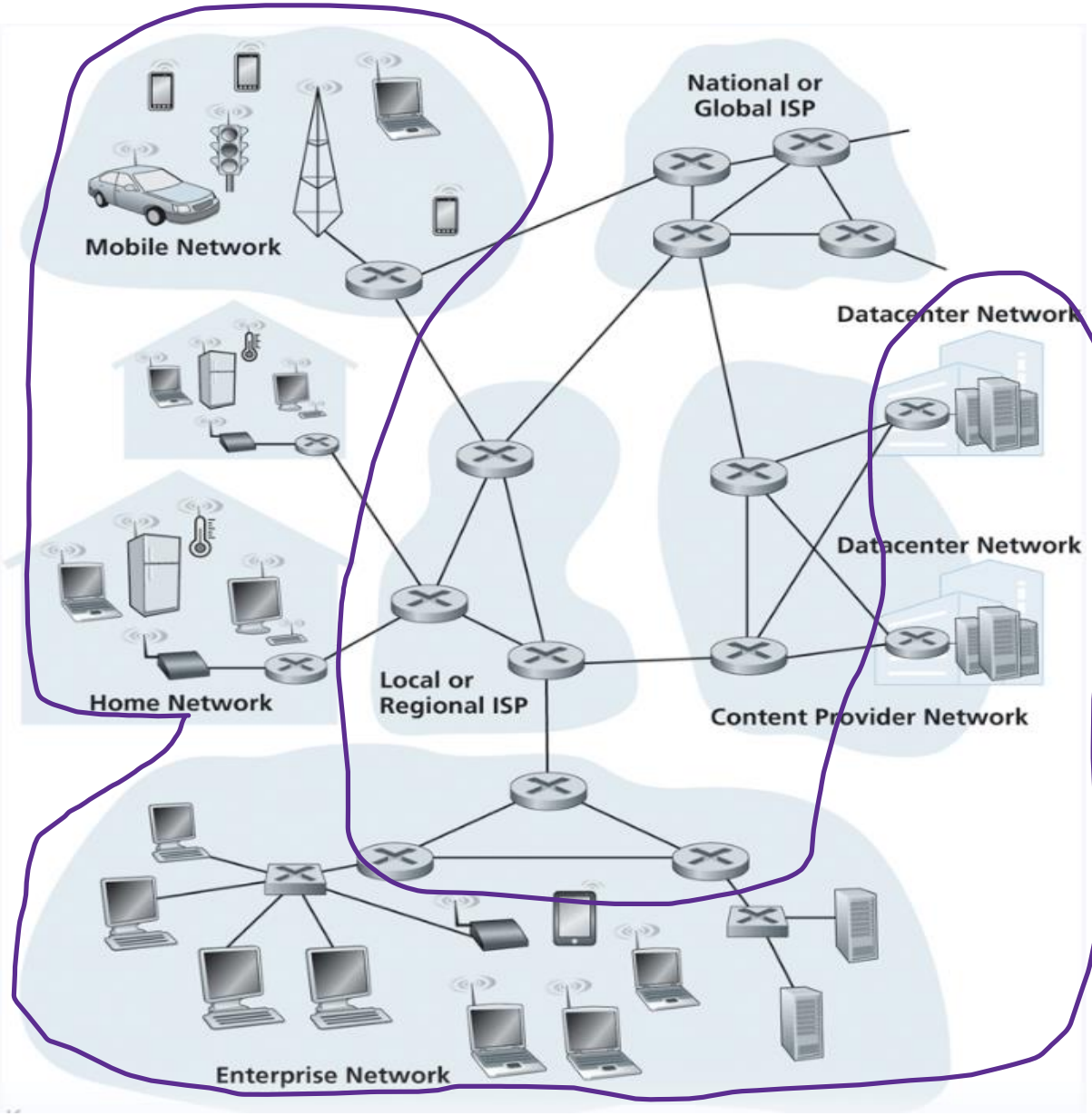
- **Two types: Clients and Servers**

End systems are connected together by a network of **communication links** and **packet switches/routers**

The **network edge** consists of end systems

The **network core** consists routers and other packet switches (usually owned by an ISP)

Internet Structure



Devices that are connected to network are called **hosts** or **end systems**

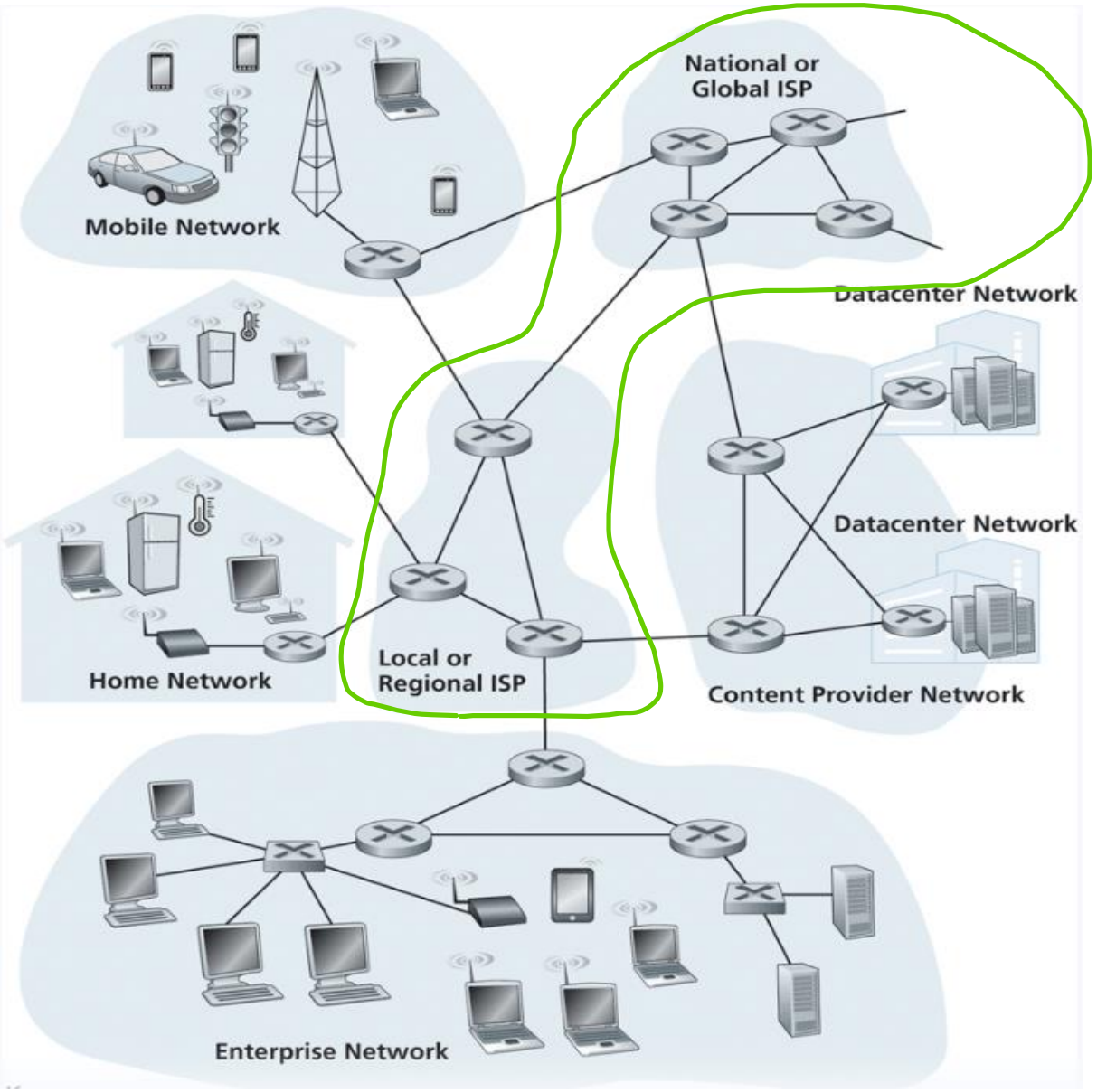
- **Two types: Clients and Servers**

End systems are connected together by a network of **communication links** and **packet switches/routers**

The **network edge** consists of end systems

The **network core** consists routers and other packet switches (usually owned by an ISP)

Internet Structure



Devices that are connected to network are called **hosts** or **end systems**

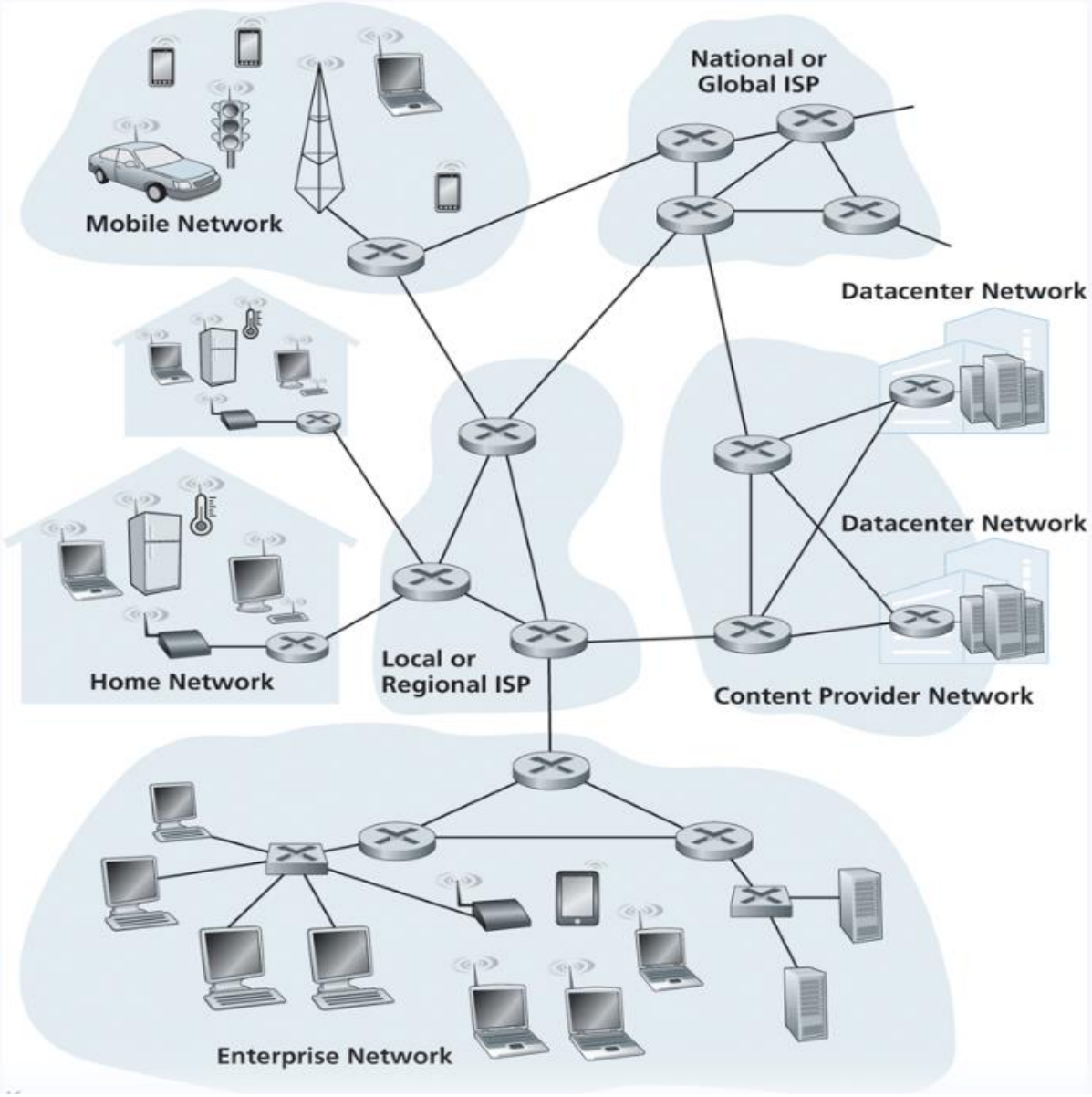
- **Two types: Clients and Servers**

End systems are connected together by a network of **communication links** and **packet switches/routers**

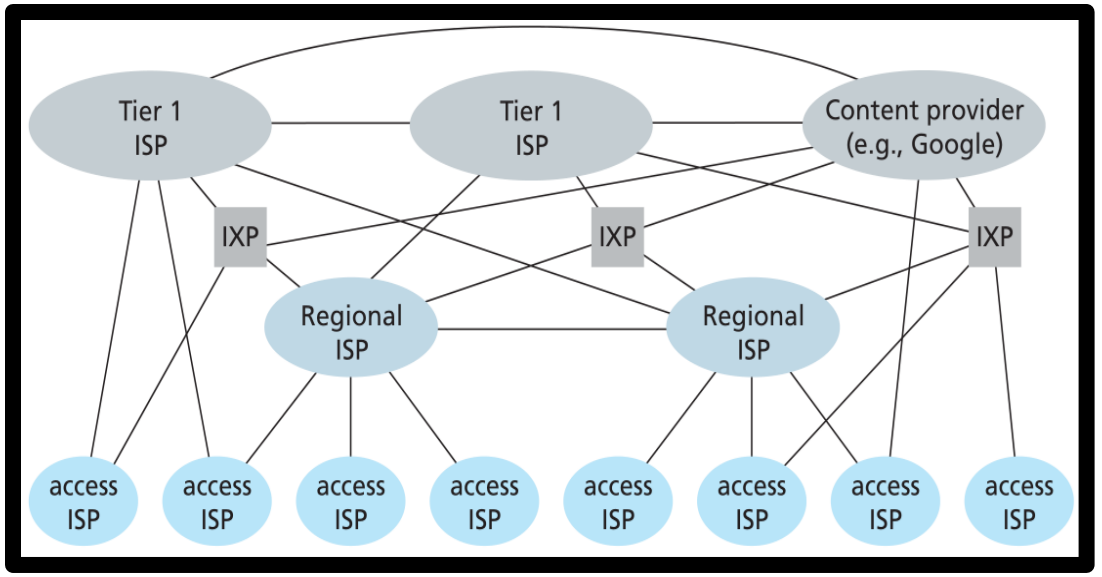
The **network edge** consists of end systems

The **network core** consists routers and other packet switches (usually owned by an ISP)

Internet Structure



The internet is **hierarchical.** We access the internet through an ISP, which typically belong to a larger network

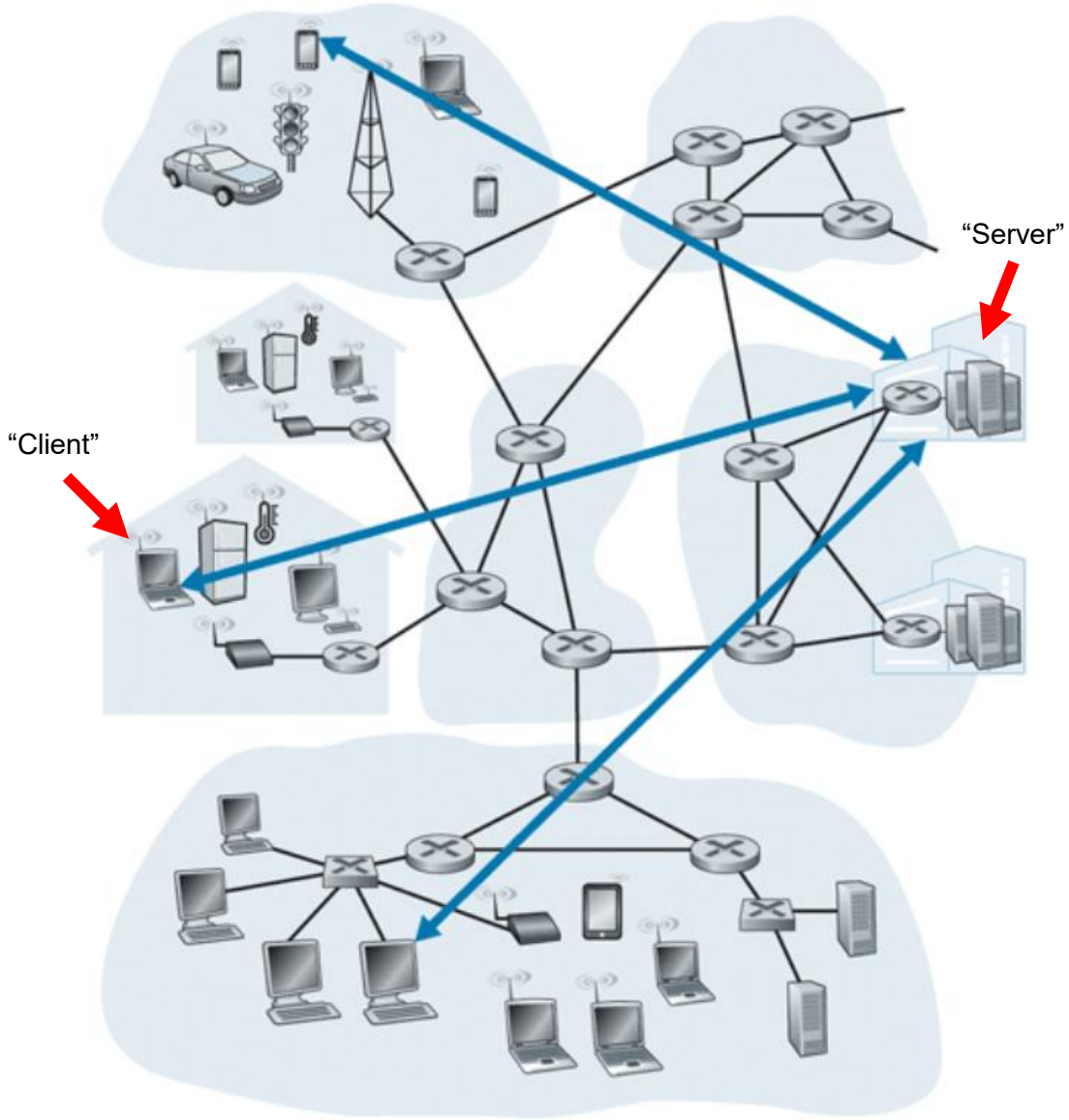


Big ISP : 192.xxx.xxx.xxx

Sub ISP: 192.42.xxx.xxx

Sub ISP: 192.42.221.xxx

Internet Structure

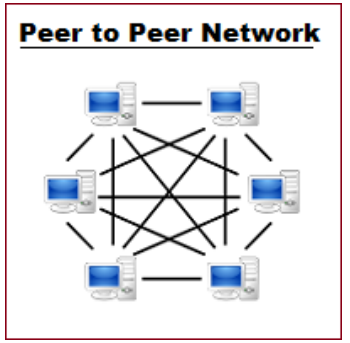


Client-server architecture

Clients do not directly interact with each other.

Hosts communicate with a dedicated, always online server that acts as a middleman between two hosts. Servers are usually hosted in a **data center**

+ Easier to administer
+ More secure
+ Ability to back up data
- Potential single point of failure
- Expensive
- Distance



P2P Architecture

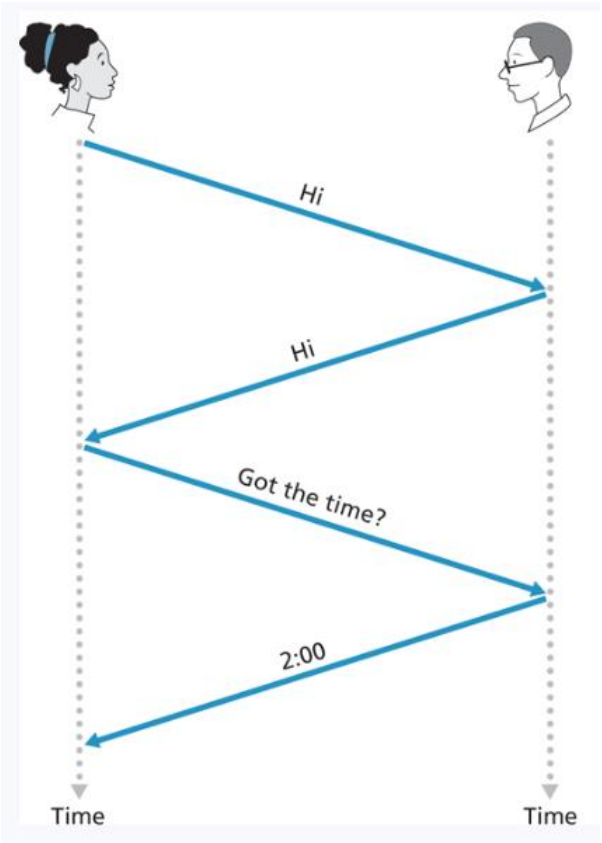
No reliance on a dedicated server

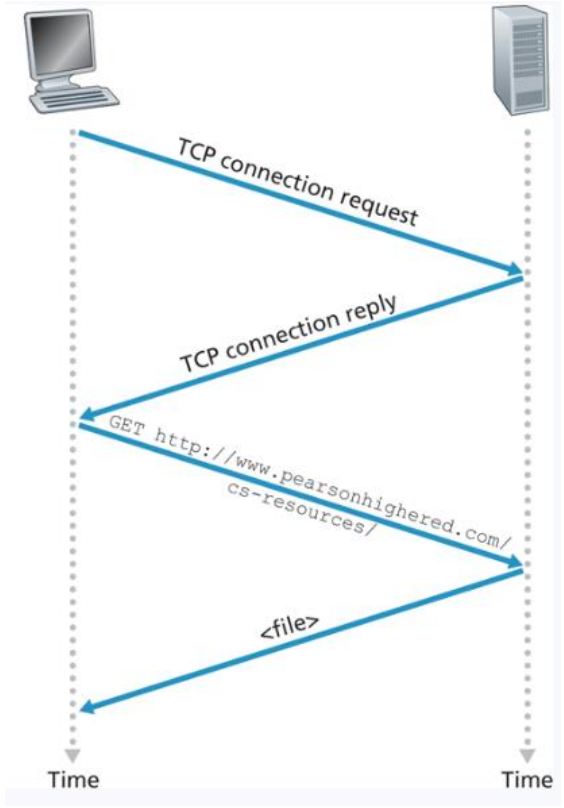
Each endpoint has same power and responsibilities.

Endpoints can be both a server and an endpoint

+ No central server
+ Availability
+ Self Scaling
- Less Secure
- ISP Friendliness
- More difficult to administer

Internet Structure





A **protocol** defines the format and the order of messages exchanges between two or more communicating entities, as well as the actions taken on the transmission and/or receipt of the message or event

Internet Engineering Task Force (IETF)
STD: 7
Request for Comments: 9293
Obsoletes: 793, 879, 2873, 6093, 6429, 6528, 6691
Updates: 1011, 1122, 5961
Category: Standards Track
ISSN: 2070-1721

W. Eddy, Ed.
MTI Systems
August 2022

Transmission Control Protocol (TCP)

2.2. Key TCP Concepts

TCP provides a reliable, in-order, byte-stream service to applications.

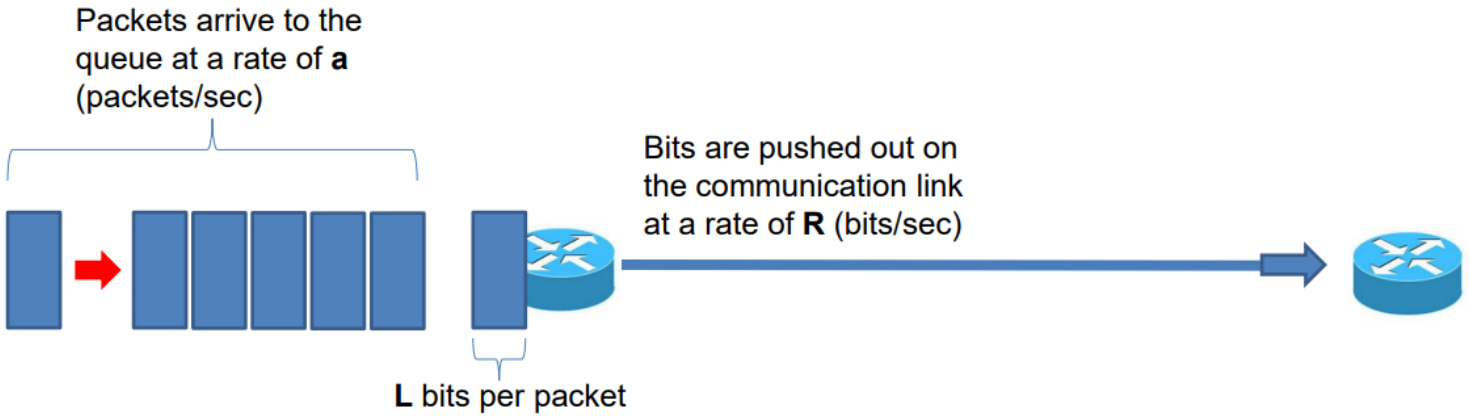
The application byte-stream is conveyed over the network via TCP segments, with each TCP segment sent as an Internet Protocol (IP) datagram.

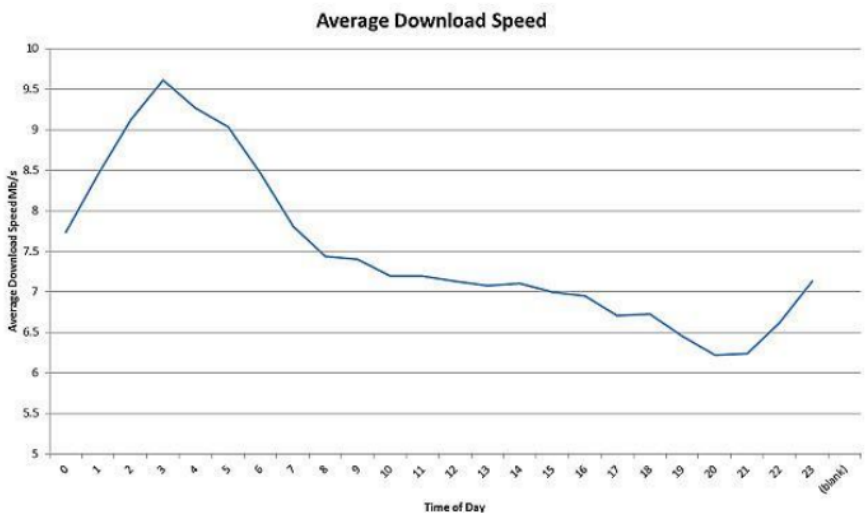
TCP reliability consists of detecting packet losses (via sequence numbers) and errors (via per-segment checksums), as well as correction via retransmission.

TCP supports unicast delivery of data. There are anycast applications that can successfully use TCP without modifications, though there is some risk of instability due to changes of lower-layer forwarding behavior [46].

The rules and details are always specific in an **RFC**

Internet Structure



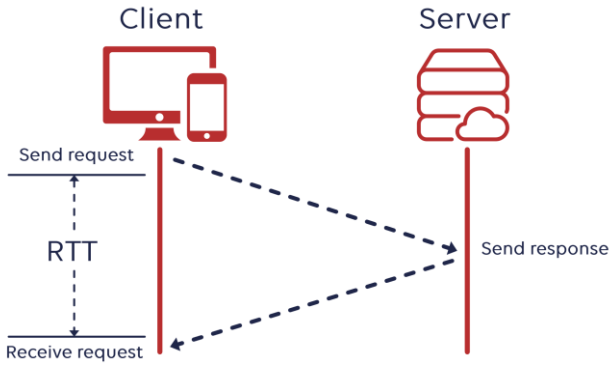


Throughput is the amount of data transferred from one place to another within a given time period

(A lot of factors affect throughput)

Traffic Intensity $= \frac{L * a}{R}$ ← Ratio of **average bits that arrive at queue** to **how quick we can process one bit**

If traffic intensity > 1 ? Bits arrive to the queue faster than we can process them **Bad!**



Round time trip

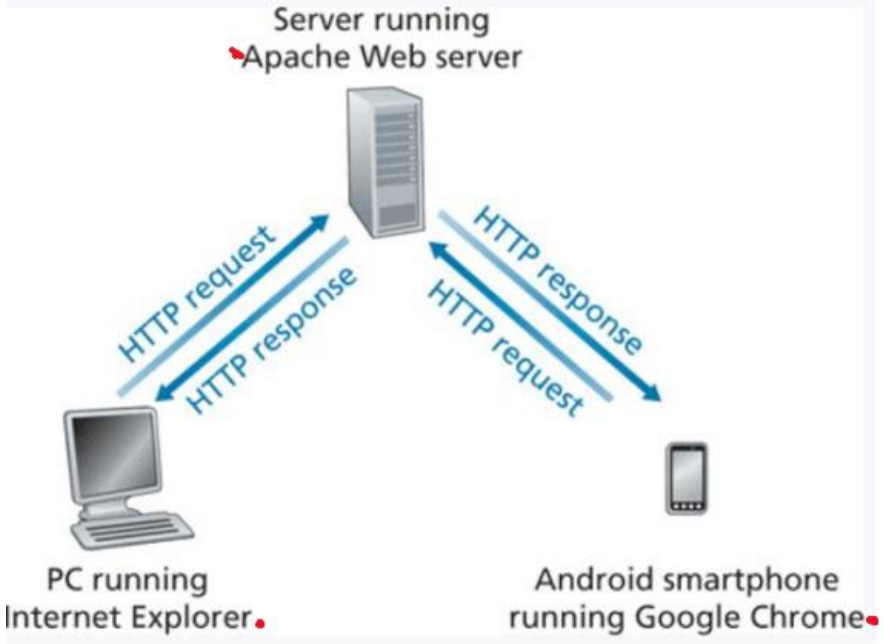
You won't need to calculate these values on the final exam, but you should have an understanding of <u>what</u> they measure and the <u>meaning</u> of that value

Application Layer

The layer which **interacts directly with applications** and provide necessary protocols and services for web applications. Specifies the shared commination protocol(s) that will be used by hosts in a communication protocol

Humans interact through the application layer. Application layer also provides engineers an interface and methods for web communication

HyperText Transfer Protocol (HTTP)- protocol that dictates the transmitting of hypermedia documents, such as HTML and other webpage objects



Application Layer

On the final exam, you should be able to describe (from a high level) how HTTP works)

HyperText Transfer Protocol (HTTP)- protocol that dictates the transmitting of hypermedia documents, such as HTML and other webpage objects

1. We provide the **URL** for the web resource that we want

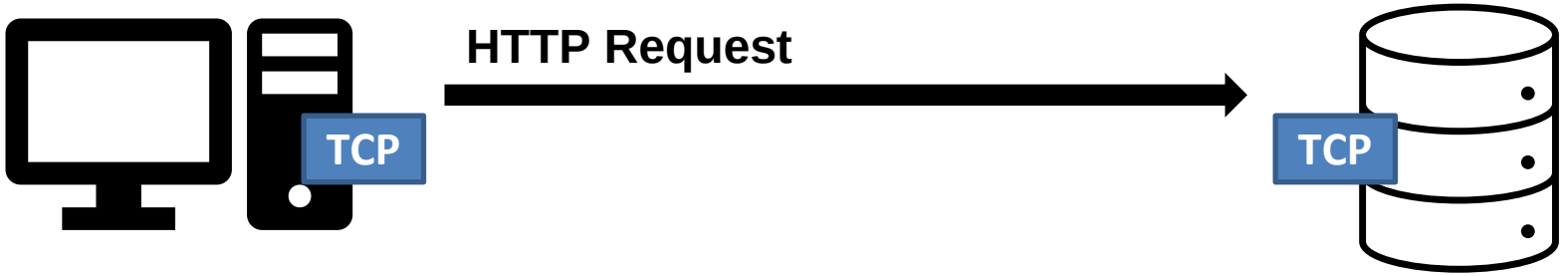
Uniform Resource Locator (**URL**)- Addressing scheme for web objects

```
scheme://domain:port/path_to_object?query_string
```

Object

```
http://cs.montana.edu/pearsall/classes/fall2022/466/main.html
```

2. HTTP is built on TCP, so an HTTP client will first initiate A TCP connection (port 80) and establish a connection socket and does the TCP handshake



3. The HTTP client sends an **HTTP Request** to the server via its socket

HTTP Request

```
Access-Control-Allow-Credentials: true
Access-Control-Allow-Methods: POST, GET, OPTIONS, DELETE, PUT
Access-Control-Allow-Origin: https://www.geeksforgeeks.org
Cache-Control: s-maxage=86400, max-age=3, must-revalidate
Connection: keep-alive, Keep-Alive
Content-Encoding: gzip
Content-Length: 555
Content-Type: text/html; charset=UTF-8
Date: Mon, 04 Nov 2019 11:59:33 GMT
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Keep-Alive: timeout=5, max=100
```

GET: Download resource
HEAD: Get resource metadata
POST: Upload form contents
PUT: Upload object to URL
DELETE: Delete object from URL

Application Layer

On the final exam, you should be able to describe (from a high level) how HTTP works)

HyperText Transfer Protocol (HTTP)- protocol that dictates the transmitting of hypermedia documents, such as HTML and other webpage objects

1. We provide the **URL** for the web resource that we want

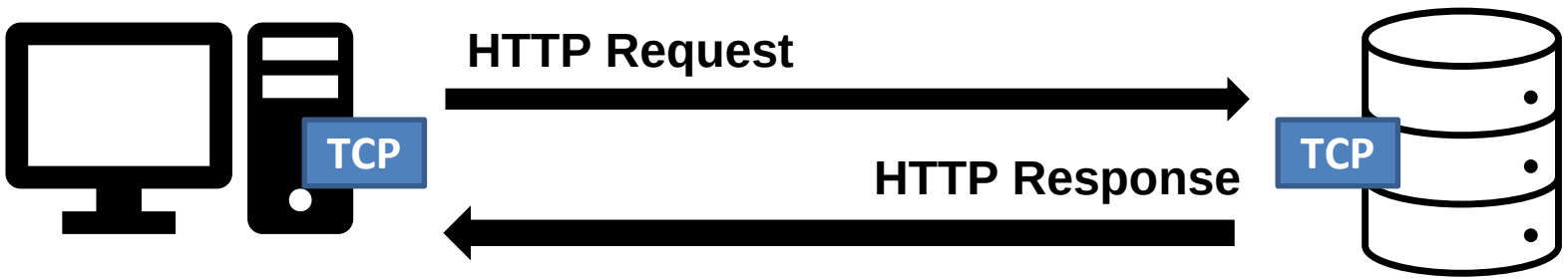
Uniform Resource Locator (**URL**)- Addressing scheme for web objects

```
scheme://domain:port/path_to_object?query_string
```

Object

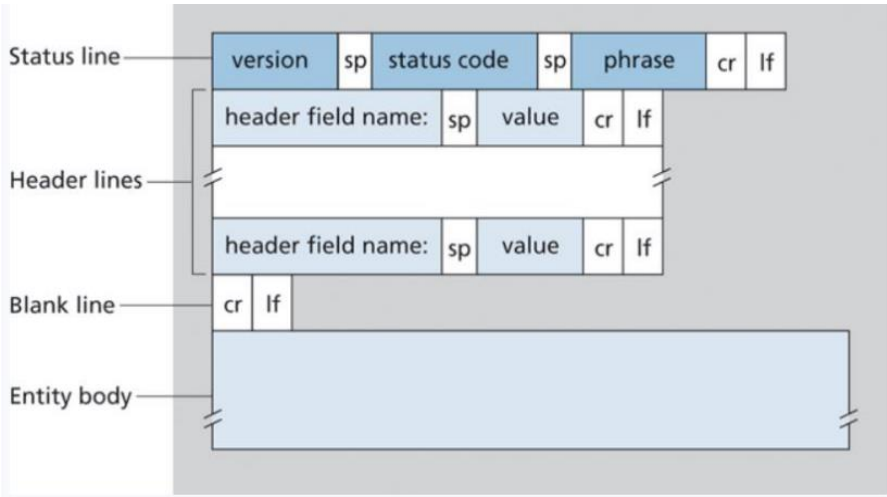
```
http://cs.montana.edu/pearsall/classes/fall2022/466/main.html
```

2. HTTP is built on TCP, so an HTTP client will first initiate A TCP connection (port 80) and establish a connection socket and does the TCP handshake



3. The HTTP client sends an **HTTP Request** to the server via its socket

4. The HTTP server process the request, retrieves the object, and sends it back to the client as an **HTTP response**

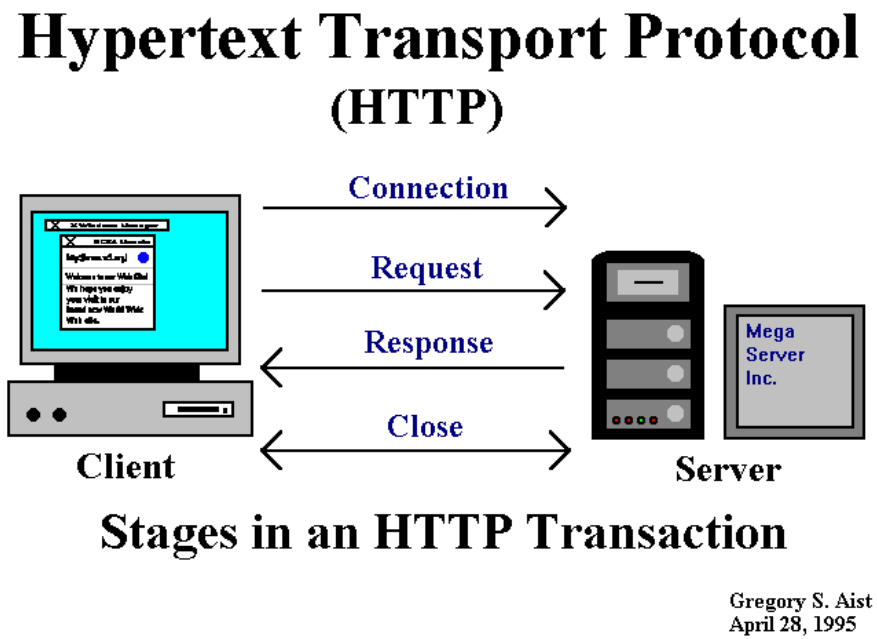


- ❑ Informational Responses (100s)
- ❑ Successful Responses (200s)
- ❑ Redirection messages (300s)
- ❑ Client error response (400s)
- ❑ Server error response (500s)

Application Layer

On the final exam, you should be able to describe (from a high level) how HTTP works)

HyperText Transfer Protocol (HTTP)- protocol that dictates the transmitting of hypermedia documents, such as HTML and other webpage objects



Probably someone in this class: *Who cares??*

When you graduate and get your job as a software engineer/web developer, the application you are writing, or the API you are working with will be using HTTP to communicate with other hosts.

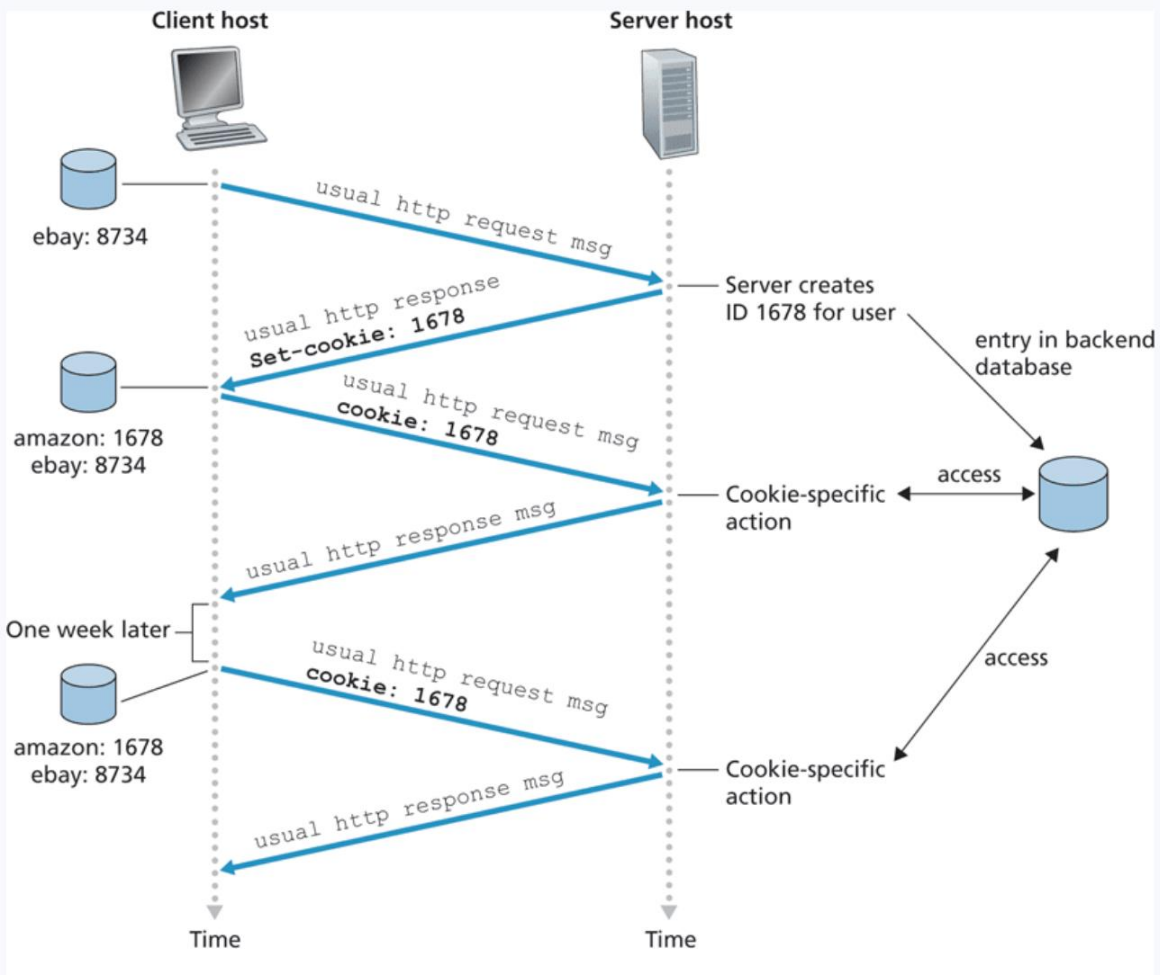
It's important to have an understanding and intuition about what's going on under the head, and to quickly remember the structure of an HTTP request *(GET, URL)* and the structure of an HTTP *response (200 OK)*

On the final exam, you should be able to describe (from a high level) how HTTP works)

HyperText Transfer Protocol (HTTP)- protocol that dictates the transmitting of hypermedia documents, such as HTML and other webpage objects

Cookies are pieces of information that are exchanged between browsers and web servers to identify users in active connections

- Authentication
- Tracking & Advertisement
- Session Management



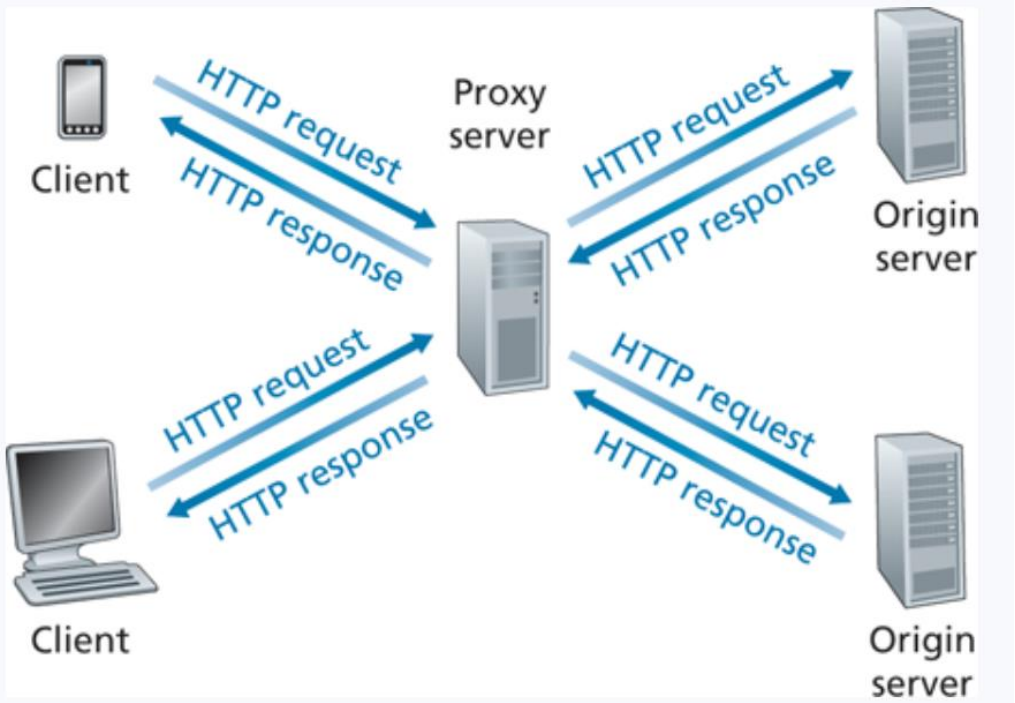
On the final exam, you should be able to describe (from a high level) how HTTP works)

HyperText Transfer Protocol (HTTP)- protocol that dictates the transmitting of hypermedia documents, such as HTML and other webpage objects

A **web cache**– also called a **proxy server**– is a network entity that satisfies HTTP requests on the behalf of an origin Web server

* Improves response time (especially if the the cache has the object that is requested)

* The connection from the client to the cache is typically much faster than the connection from client to host server



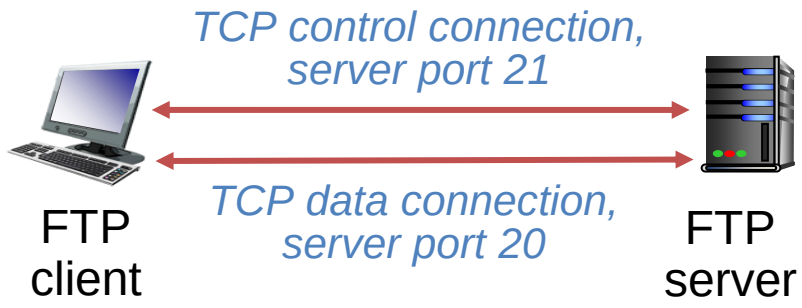
1. Browser/Client establishes a TCP connection to the Web cache and sends an HTTP request

2. Web cache checks its local storage for the requested object

3. If the web cache does not have the object, establish TCP with an origin server and issue an HTTP request

4. Web cache stores a local copy of the object, then issues an HTTP response with the object

File Transfer Protocol (FTP)- protocol used for transferring files from server to client



- FTP communicates over two connections
  - Port 21 for control information
  - Port 20 for data
- Differences from HTTP
  - Control communication “out-of-band”
  - Server maintains per client state: authentication, current directory

- FTP procedure:
  1. FTP client contacts FTP server at port 21, using TCP
  2. Client authorized over control connection
  3. Client browses remote directory, sends commands over control connection
  4. When server receives file transfer command, server opens 2nd TCP data connection (for file) to client
  5. After transferring one file, server closes data connection

Application Layer

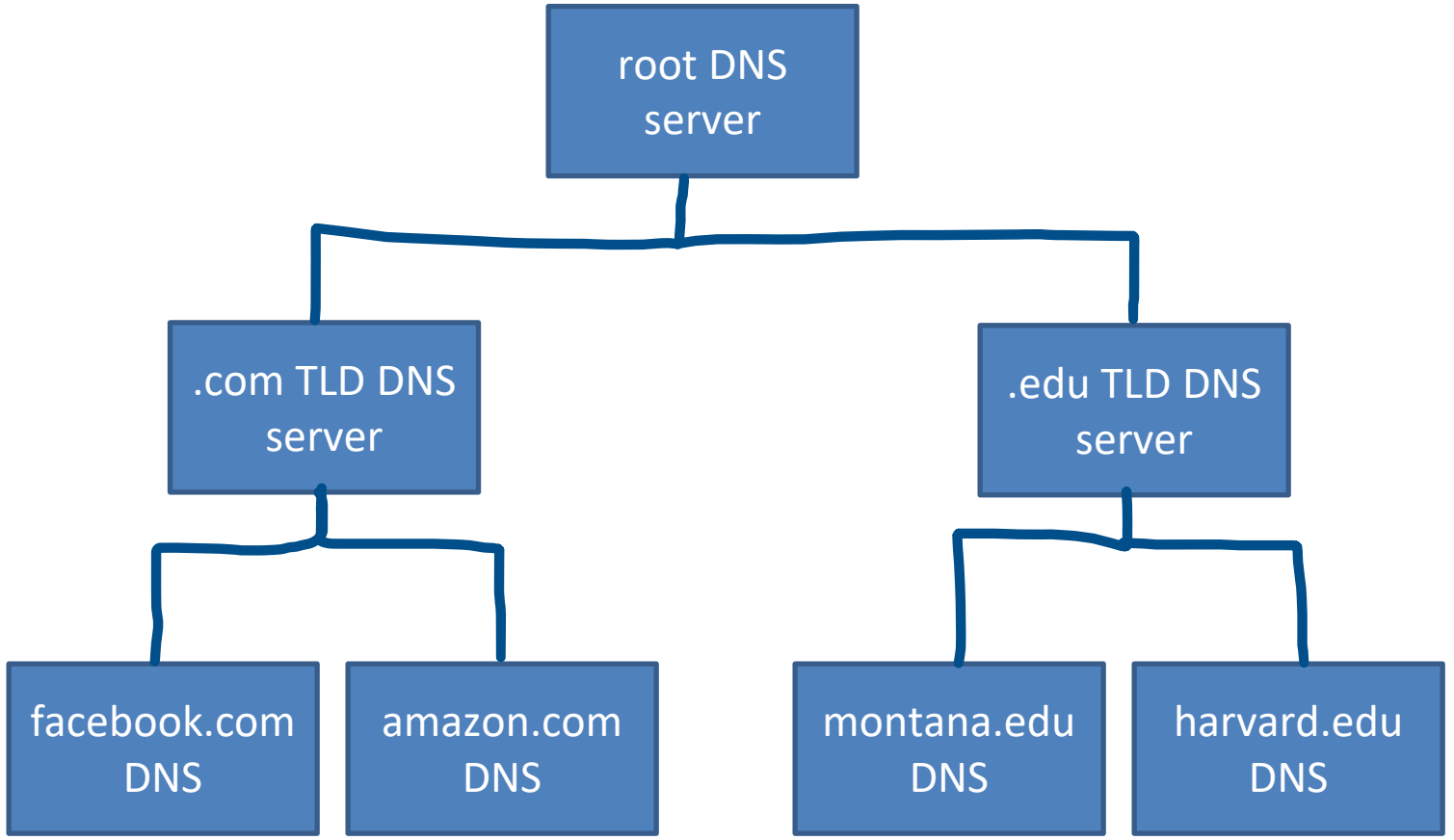
On the final exam, you should be able to describe (from a high level) how DNS works

- DNS is a **distributed**, **hierarchical** database used for mapping hostnames to IP address (no DNS server has all the records!)
  - Prior to creating a TCP connection and sending an HTTP request, we first need to issue a DNS request!
  - (Built on UDP, lookups happen on port 53)

Hierarchy consists of different types of DNS servers:

Authoritative DNS servers- Organization's own DNS with up-to-date records

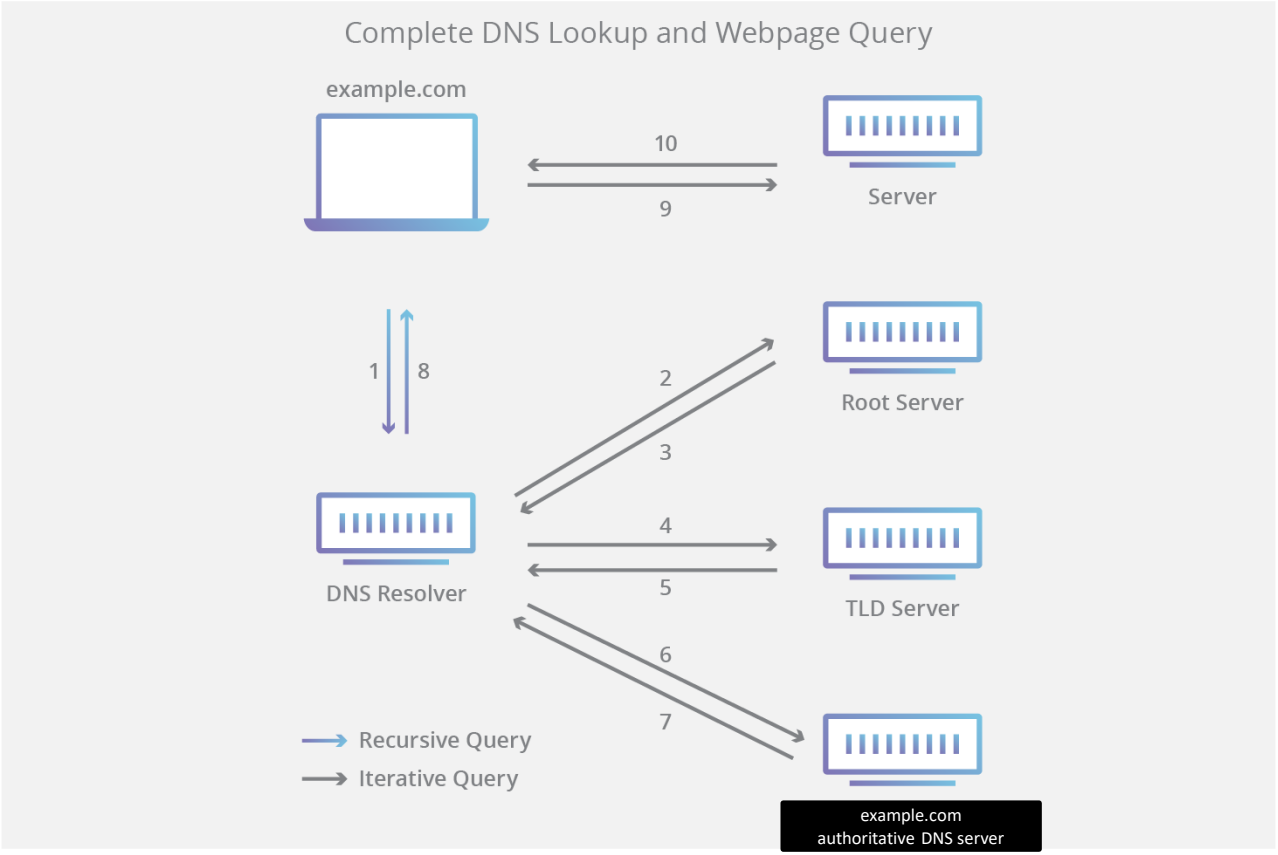
Top-level domain (TLD) servers- responsible for keeping IP addresses for authoritative DNS servers for each top-level domain (.com, .edu, .jp, etc)



Application Layer

On the final exam, you should be able to describe (from a high level) what DNS is and how DNS works

- DNS is a **distributed**, **hierarchical** database used for mapping hostnames to IP address (no DNS server has all the records!)
  - Prior to creating a TCP connection and sending an HTTP request, we first need to issue a DNS request!
  - (Built on UDP, lookups happen on port 53)



There are also local DNS servers, and local DNS entries on your machine

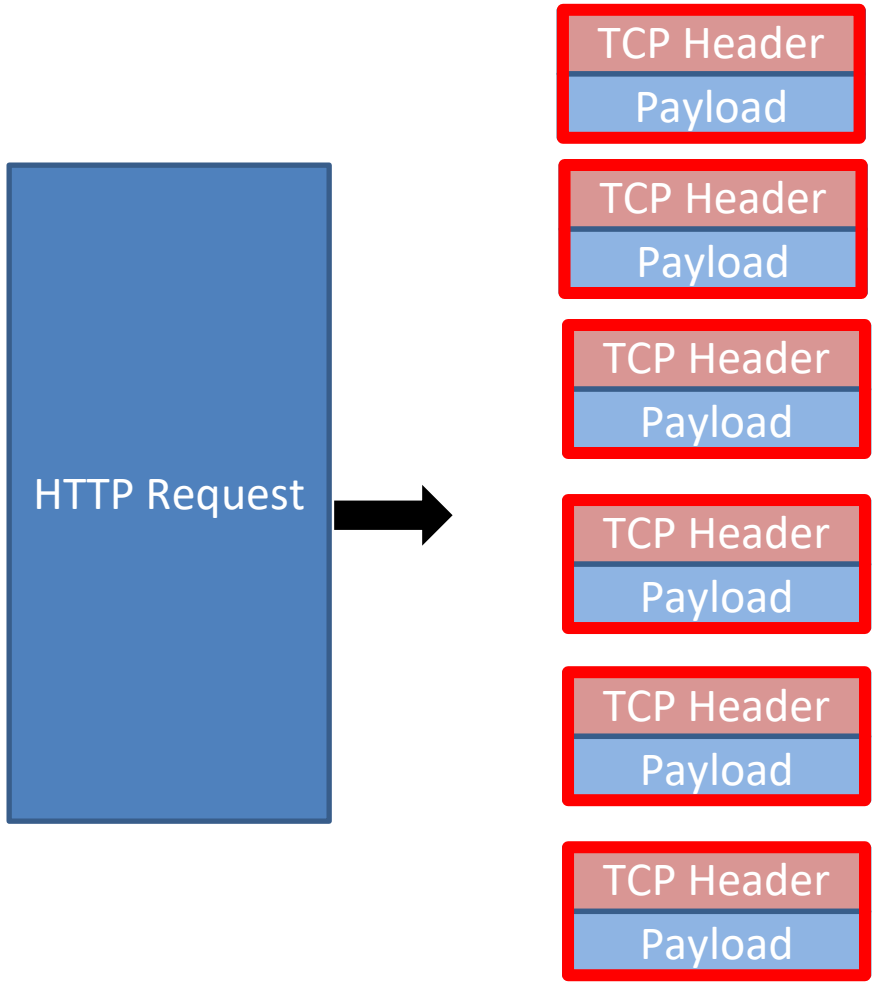
Root DNS servers- responsible for maintaining IP addresses for TLD servers

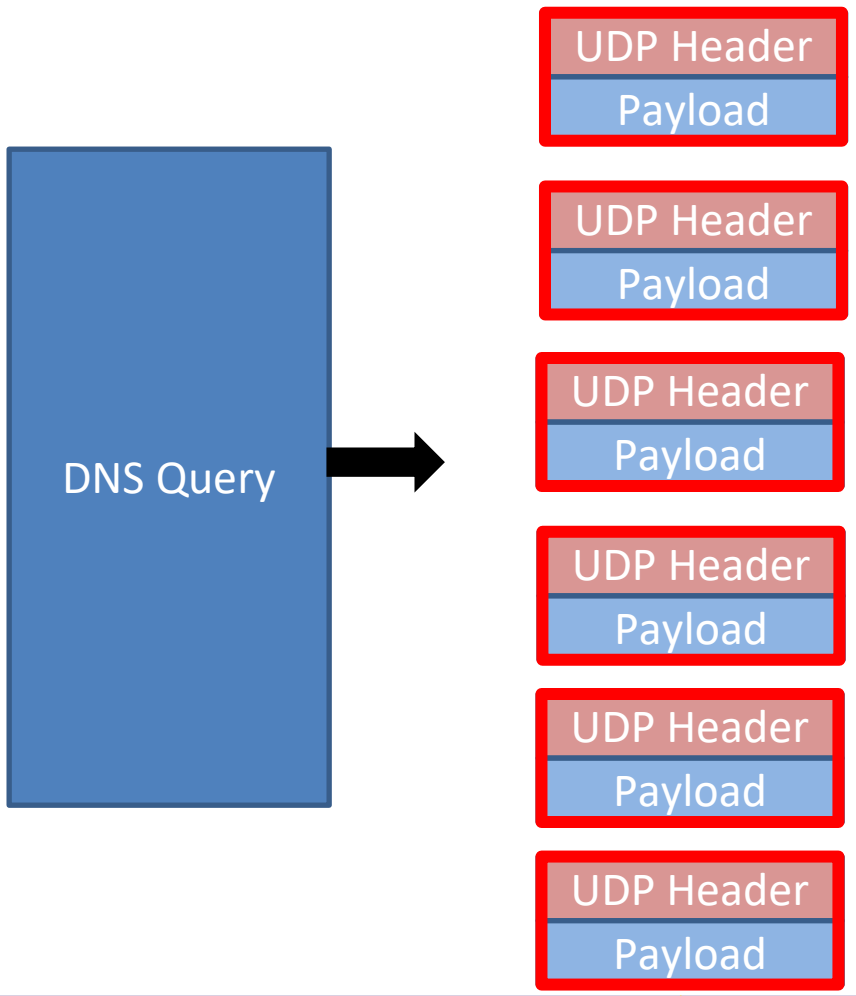
Top-level domain (TLD) servers- responsible for keeping IP addresses for authoritative DNS servers for each top-level domain (.com, .edu, .jp, etc)

Authoritative DNS servers- Organization's own DNS with up-to-date records

Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

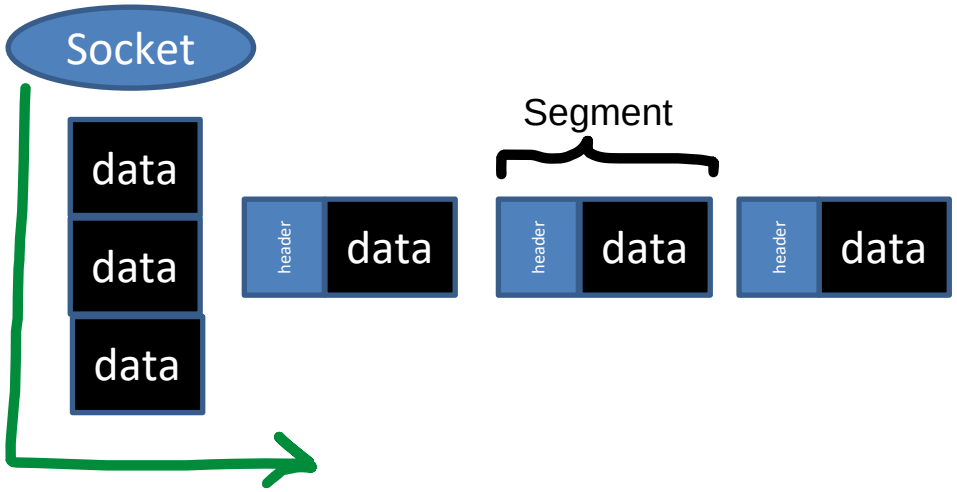
Our messages from the application are now divided up into transport layer segments



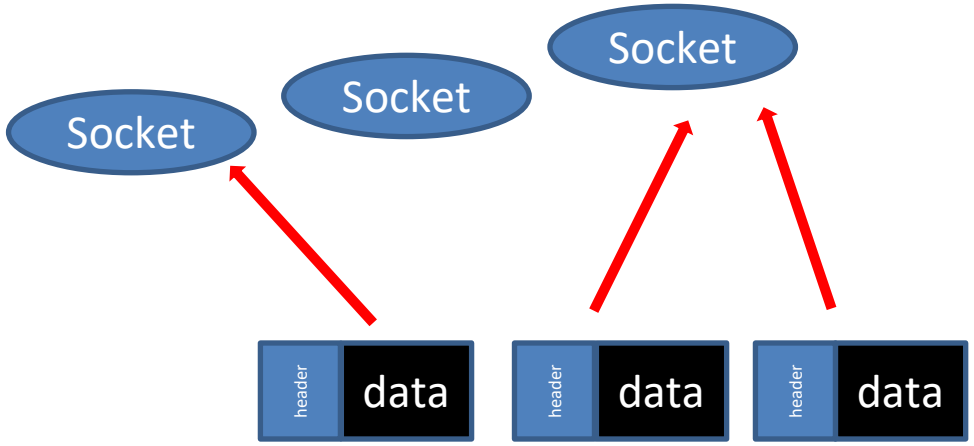


Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

Multiplexing is the process of gathering chunks from sockets, encapsulating chunks with header information, and passing the segment into the network layer



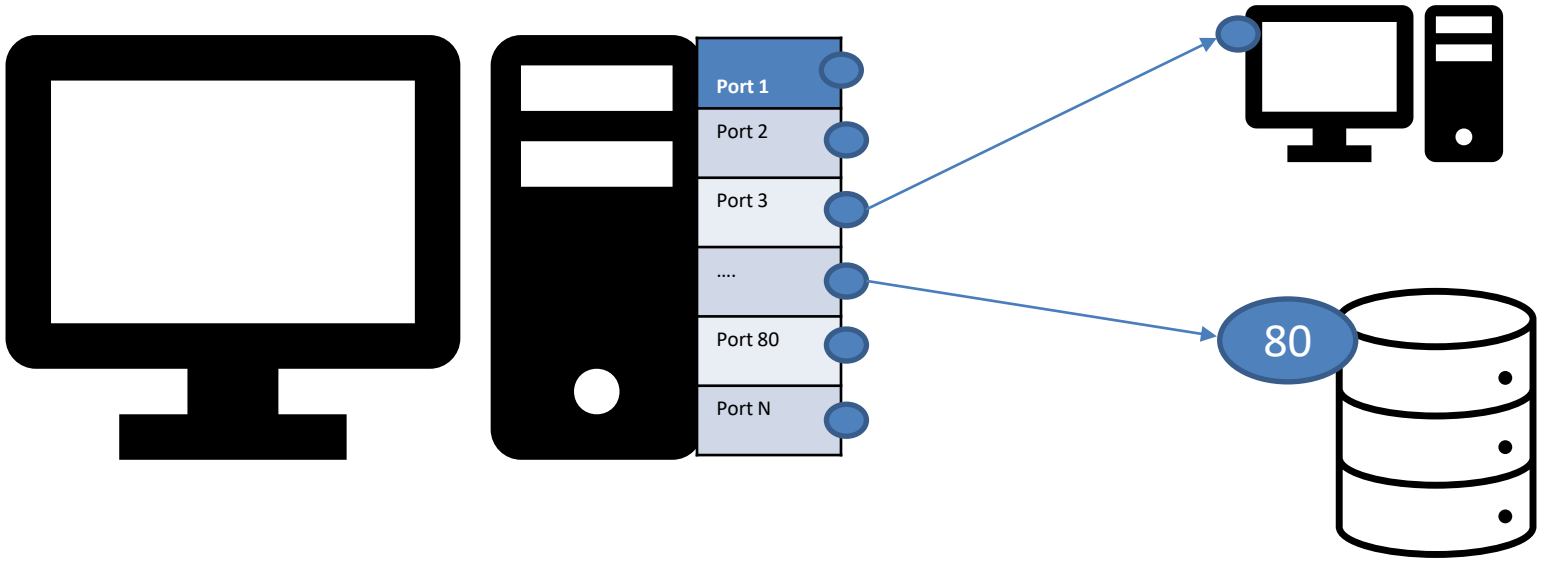
Demultiplexing is the receiving segments from the transport layer and delivering the segment to the correct socket.



Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

Segments are delivered to a **port** *(which a **socket** is linked to)*

Different services/protocols are linked to different ports



Service	Port
HTTP	80
HTTPS	443
DNS	53
FTP	20 and 21
SSH	22
BGP	179

These are some common ports you should really know for the final exam ☺

Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

User Datagram Prot. (UDP)

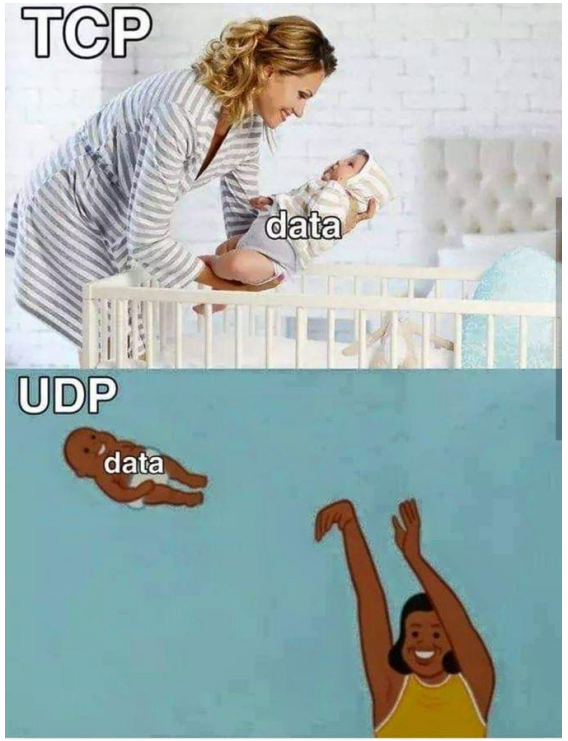
Unreliable data transfer

- Connection-less
  - Don't know if receiver is present
- No flow control
  - Overflow at receiver possible
- No congestion control
  - Sender can overload the network
- No guarantees on
  - End-to-end delay
  - Throughput
  - Security

Transmission Control Prot. (TCP)

Reliable stream transport

- Connection-oriented
  - Establishes receiver presence
- Flow control
  - Sender won't overwhelm receiver
- Congestion control
  - Senders won't overload network
- No guarantees on
  - End-to-end delay
  - Throughput
  - Security

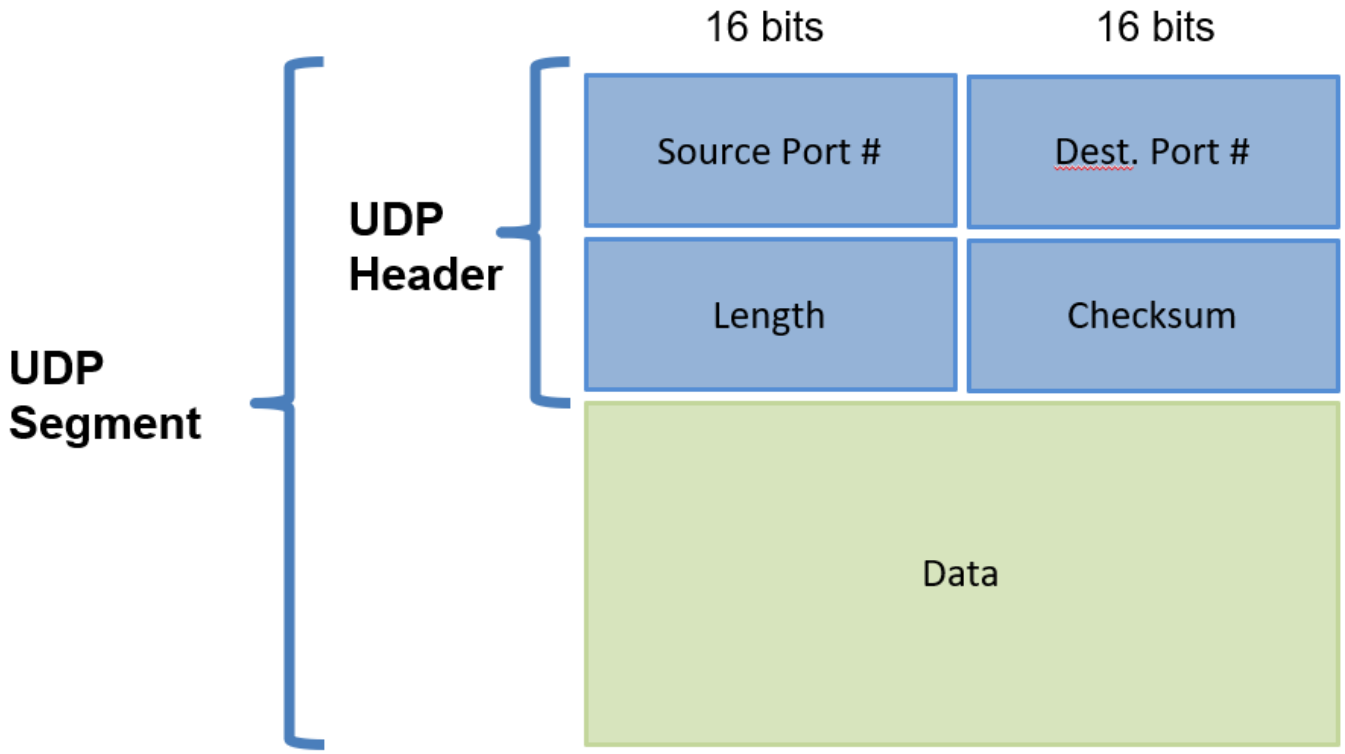


Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

User Datagram Prot. (UDP)

Unreliable data transfer

- Connection-less
  - Don't know if receiver is present
- No flow control
  - Overflow at receiver possible
- No congestion control
  - Sender can overload the network
- No guarantees on
  - End-to-end delay
  - Throughput
  - Security



Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

It would be very helpful for you to review these ☺

Checksum- Used to detect bit errors in transmitted pacets

Timer- Used to timeout/retransmit a packet, possibly because the packet (or its ACK) was lost within a channel

Sequence Number- Used for sequential numbering of packets of data flowing form sender to receiver. Gaps in sequence number of packets allow the receiver to detect a lost or duplicate packet

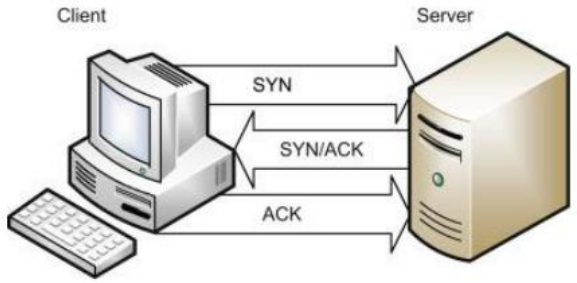
Acknowledgement- Used by the receiver to tell the sender that a packet or set of packets has been received correctly. ACKs will typically carry the sequence # of the packet being acknowledged

Negative Acknowledgement- Used by the receiver to tell the sender that a packet has not been received correctly. Negative acknowledgements will typically carry the sequence number of the packet that was not received correctly

Window, pipelining- The sender may be restricted to sending only packets with sequence numbers that fall within a given range. By allowing multiple packets to be transmitted but not yet acknowledged, sender utilization can be increased over a stop-and-wait mode of operation.

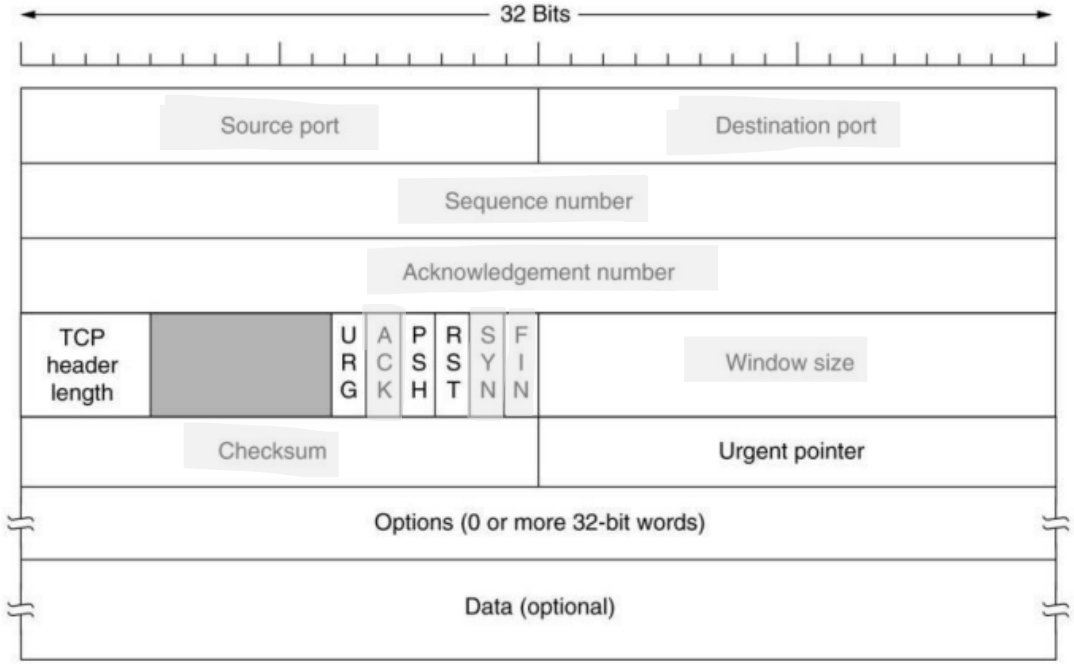
Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

- point-to-point:
  - one sender, one receiver
- reliable, in-order *byte steam:*
- pipelined:
  - TCP congestion and flow control set window size

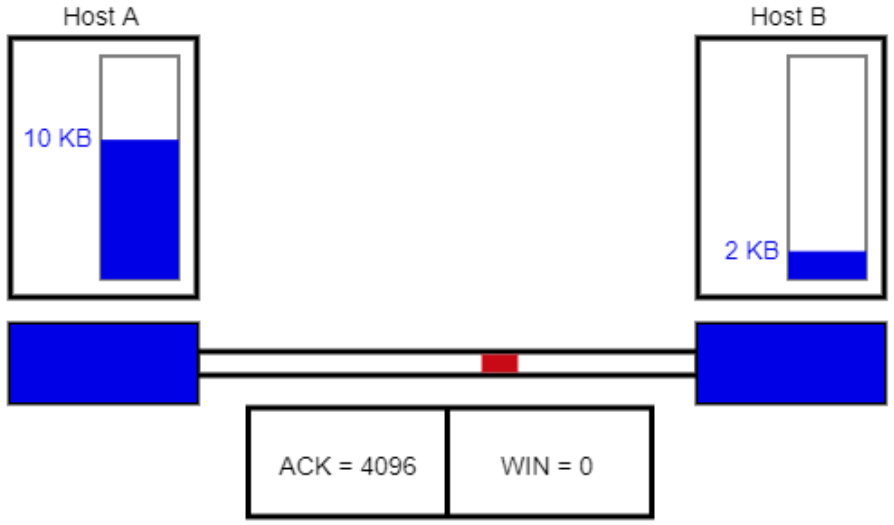


TCP Handshake

- full duplex data:
  - bi-directional data flow in same connection
- connection-oriented:
  - handshaking (exchange of control msgs) in its sender, receiver state before data exchange
- flow controlled:
  - sender will not overwhelm receiver



Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

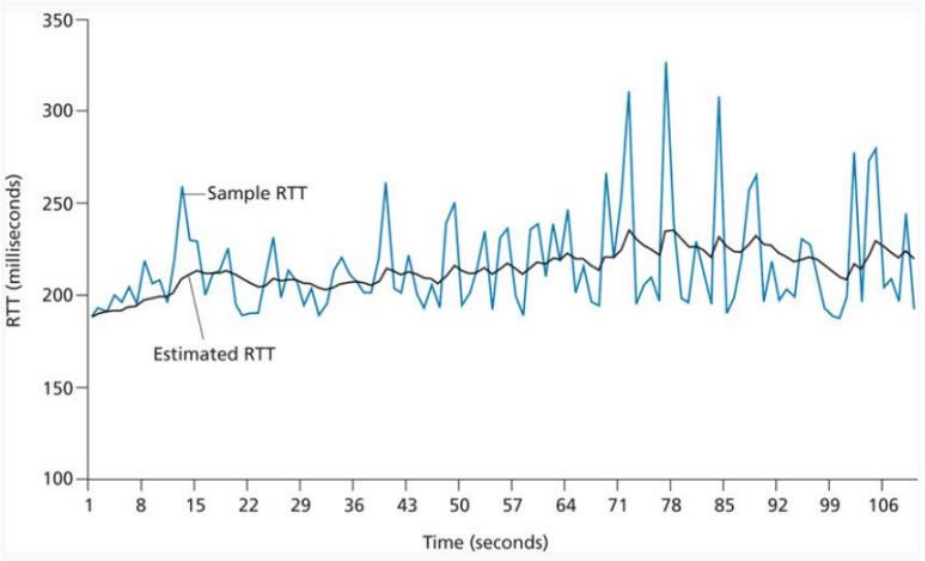


TCP sends back amount of available buffer space in the receiver

This helps make sure we don't overwhelm the receiver

TCP is **self-clocking**, which means it will control itself if it notices congestion in a network

-See how many dropped packets we are getting
-Amount of duplicate ACKs received
-Amount of UnAcked packets

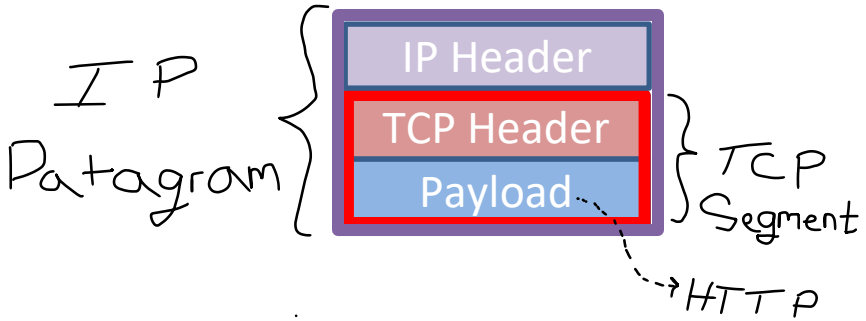


The timeout value for TCP is dynamically calculated by looking at RTT values for packets

Transport Layer - Provides host-to-host, **reliable data transfer**, and dictates the flow of data

Given a scenario and application requirements, you should be able to pick between UDP and TCP and give a reasonable justification for your choice

Our segments from the transport layer are now encapsulated into network layer datagrams



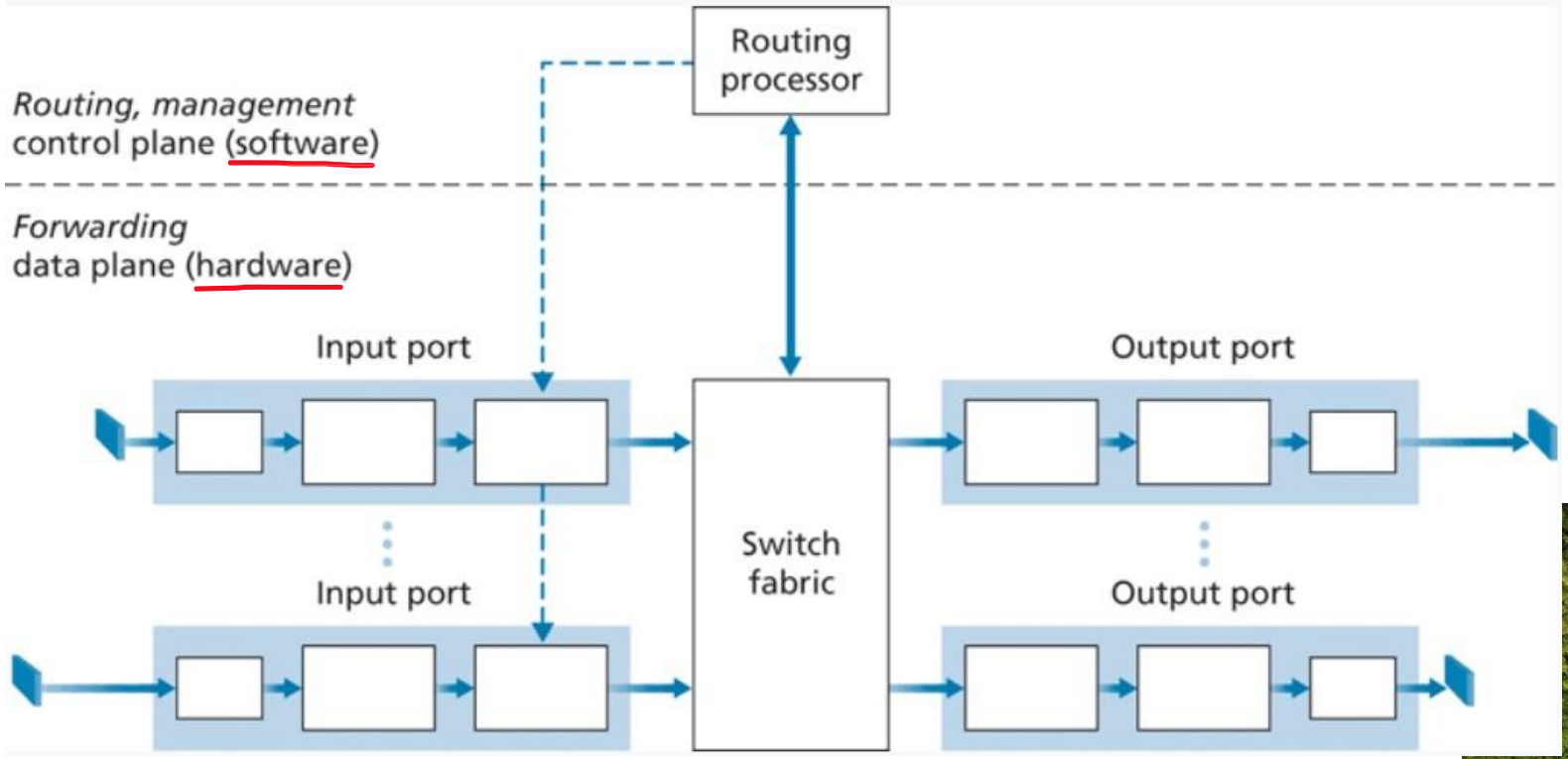
Control Plane

Routing: determine route taken by packets from source to destination

Data Plane

Forwarding: move packets from router's input to appropriate router output

Network Layer - Provides Routing and Forwarding Functionality



1. Destination-based forwarding

- Forwarding decisions are based on the **destination** of the packet

2. Generalized forwarding

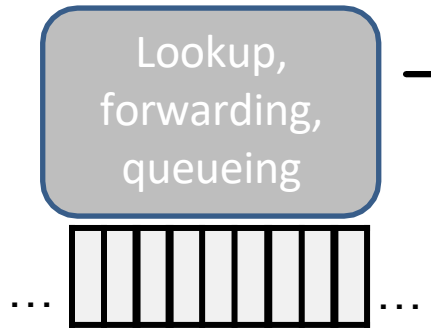
- Forwarding decisions based on any set of header field values

Longest prefix matching when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address

Address range	Interface (output link)
11001000 00010111 00010*** *********	1
11001000 00010111 00011000 *********	2
11001000 00010111 00011*** ********	3
otherwise	4

1. Destination-based forwarding

- Forwarding decisions are based on the **destination** of the packet



Routing Table

Address range	Interface (output link)
128.11.52.0 – 128.11.52.255	1
153.90.2.0 – 153.90.2.255	2
153.90.2.87 – 153.90.2.89	3

Routing Tables are maintained **internally**, and help make forwarding decisions

Network Layer - Provides Routing and Forwarding Functionality

Longest prefix matching when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address

Address range	Interface (output link)
128.28.XXX.XXX	1
128.27.XXX.XXX	2

Address range	Interface (output link)
128.28.113.XXX	1
128.28.114.XXX	2

Address range	Interface (output link)
128.28.114.23	1
128.27.114.16	2
128.27.114.44	3

Address range	Interface (output link)
128.28.113.1	1
128.27.113.2	2
128.27.113.3	3

Address range	Interface (output link)
128.27.1.XXX	1
128.27.2.XXX	2

IP Address: Globally unique* 32 bit (4 byte) **dotted decimal** number assigned to interfaces on hosts and routers

(Think of this as your ZIP code)

193.32.216.9

=

11000001 00100000 11011000 00001001

128	64	32	16	8	4	2	1
1	**1**	0	0	0	0	0	**1**

128 + 64 + 1 = **193**

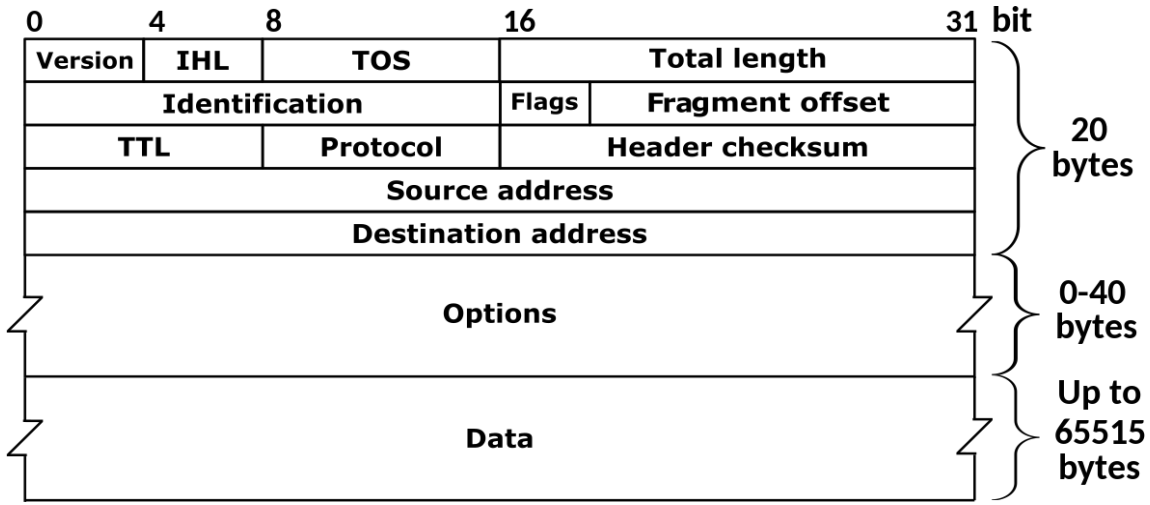
193 = `11000001` in binary

Network Layer - Provides Routing and Forwarding Functionality

On the final exam, you should know the difference between IPv4 and IPv6, and why we need IPv6

IP Protocol

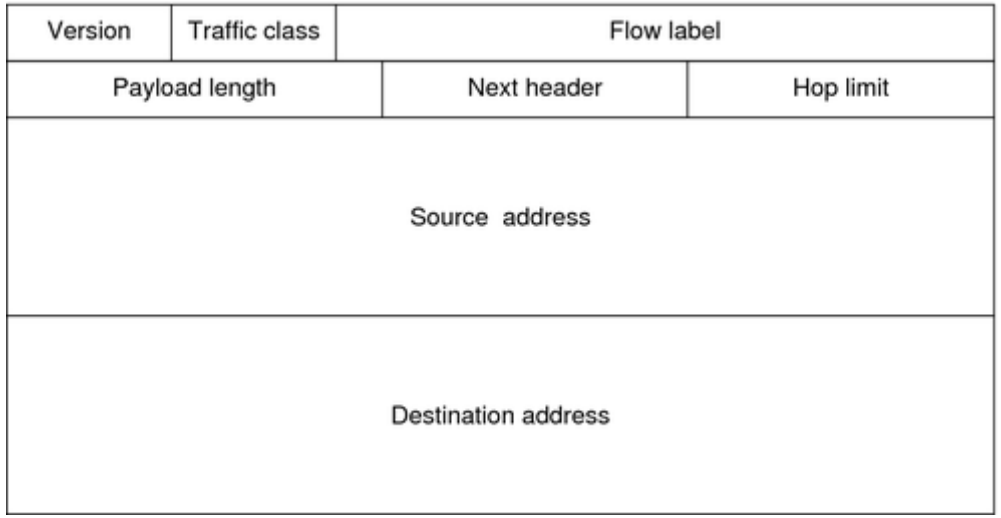
IPv4



IPv4: 32-bit addresses (decimal)

192.149.252.76

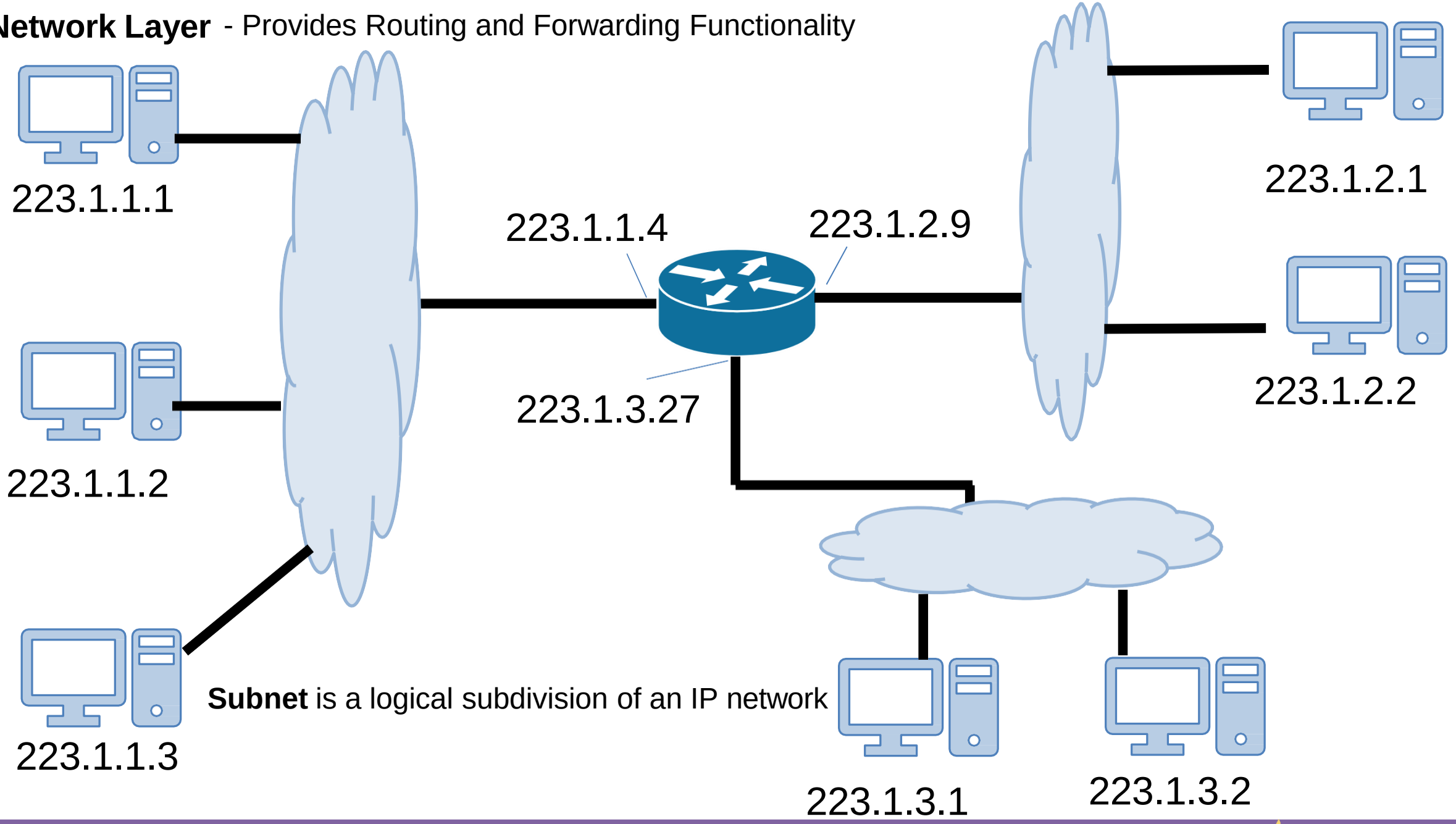
IPv6



IPv6: 128-bit addresses (hexademical)

3ffe:1900:fe21:4545::

Network Layer - Provides Routing and Forwarding Functionality

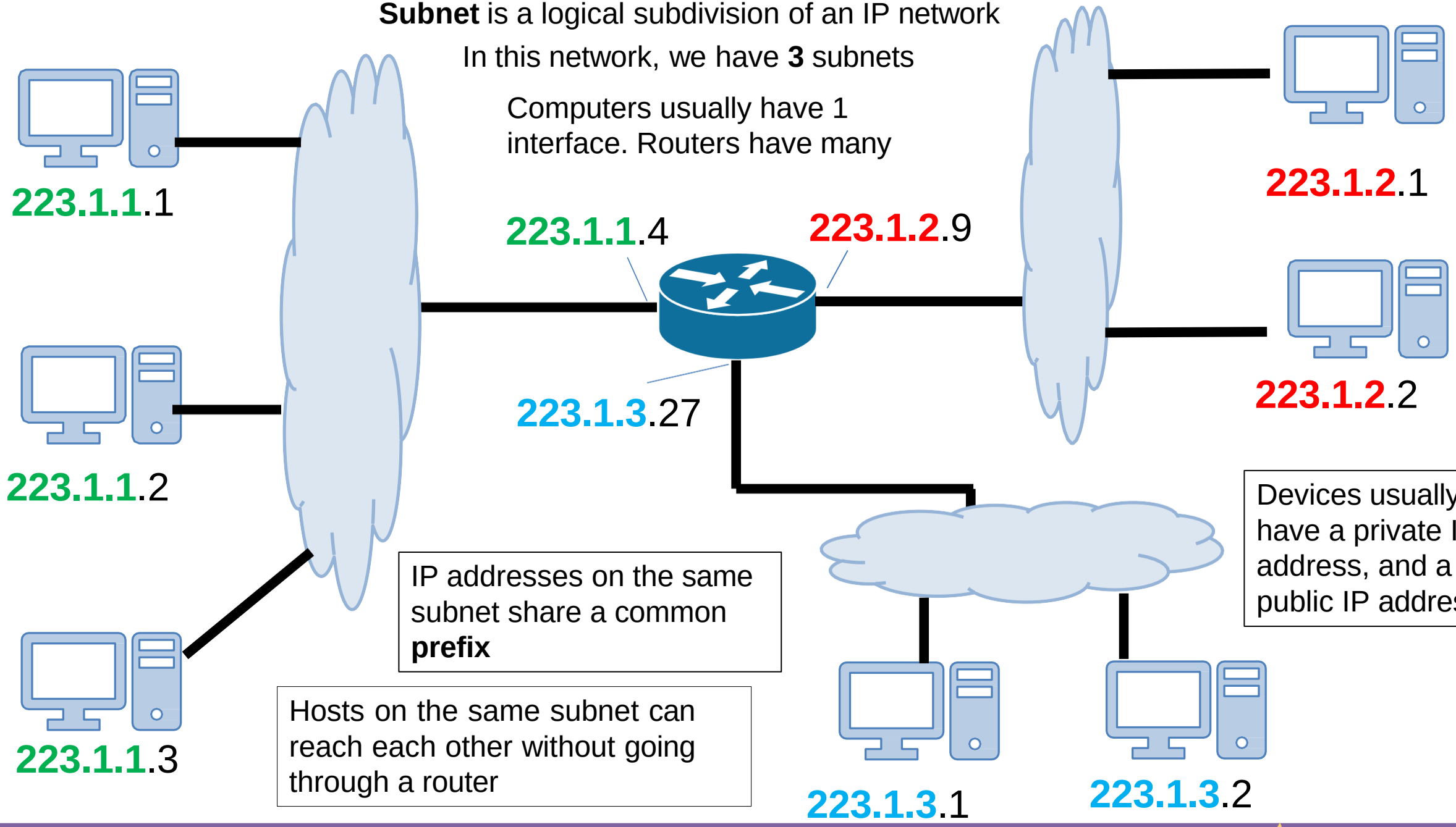


Subnet is a logical subdivision of an IP network

Subnet is a logical subdivision of an IP network

In this network, we have **3** subnets

Computers usually have 1 interface. Routers have many

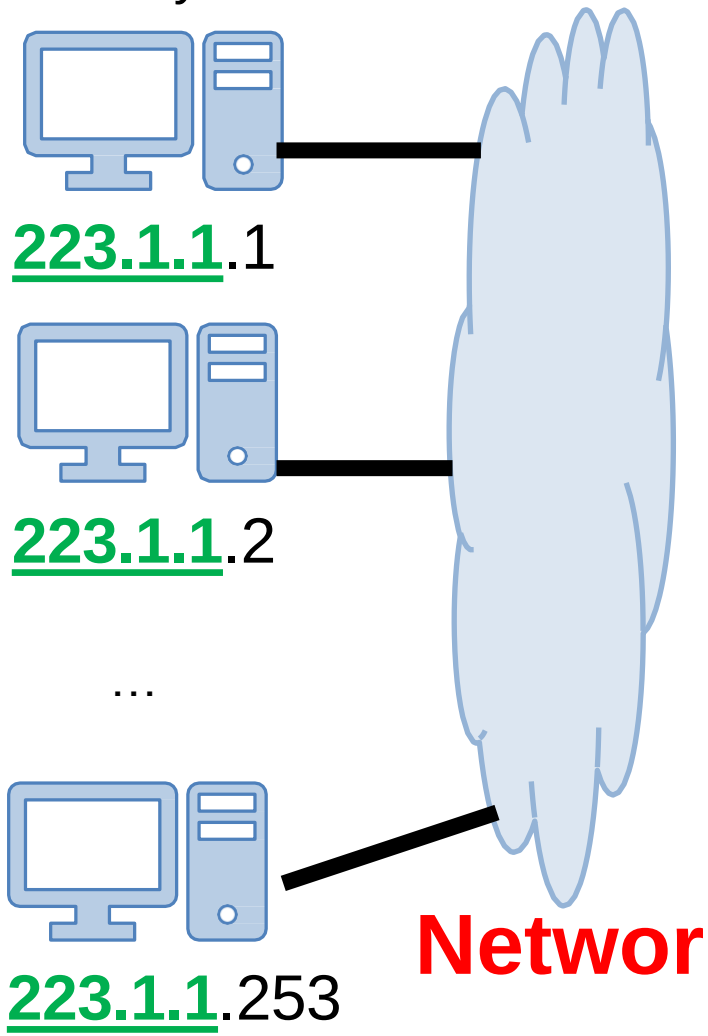


IP addresses on the same subnet share a common **prefix**

Hosts on the same subnet can reach each other without going through a router

Devices usually have a private IP address, and a public IP address

It is very common to have a **range** of IP addresses assigned to you (random assignment would be chaos)



Subnet mask

223.1.1.0/24 *(We can have /8, /16, /1, /2, /3, etc)*

The leftmost 24 bits represent the prefix of the subnet

11111111 11111111 11111111 XXXXXXXX = **255.255.255**.0

223.1.1.67 ✓

223.2.1.67 ✗

223.2.1.255 ✗

The number of host bits available controls the amount of IP addresses available to you (more host bits = more available IP addresses)

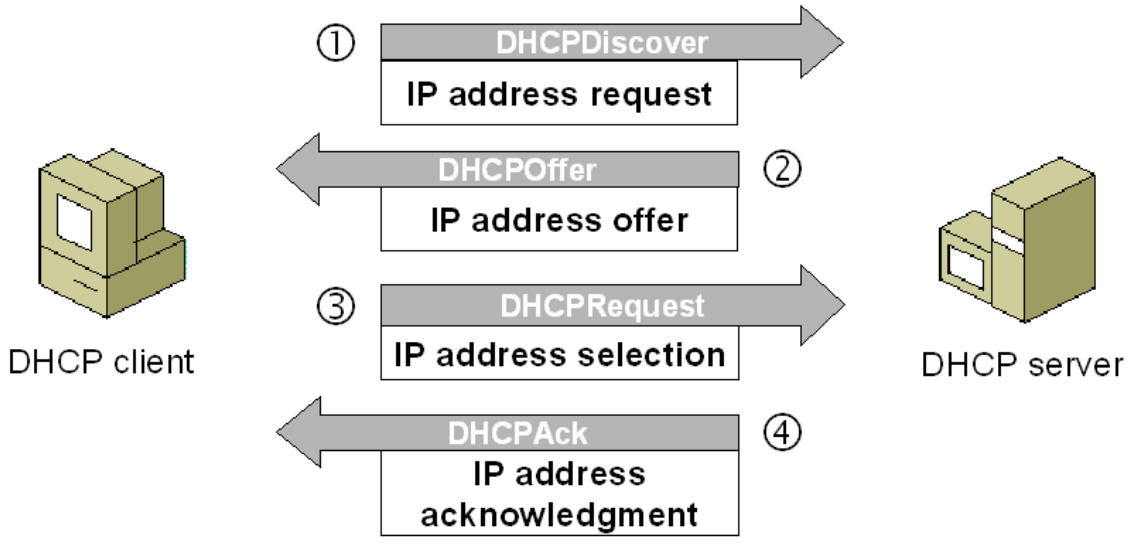
Network bits **193.32.216.9** **Host bits**

11000001 00100000 11011000 00001001

Dynamic Host Configuration Protocol (DHCP) is a **plug-and-play**, client-server protocol that allows a host to obtain an IP address automatically

When a host is automatically assigned an IP address, it might keep that one forever, or the IP addresses can be temporary

(more common)

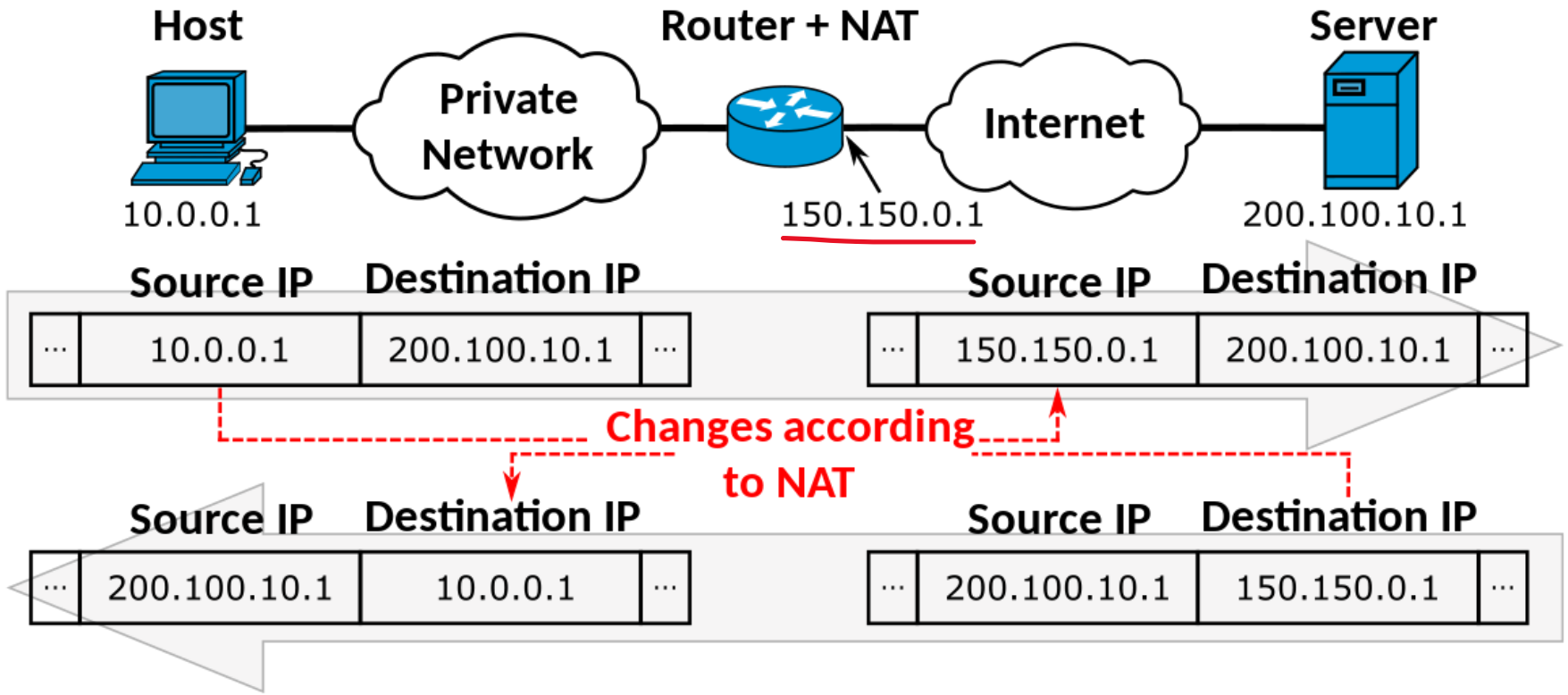


On the final exam, you should know what DHCP is, and when it is used

NAT is a translation of multiple private IP addresses to one single public IP address

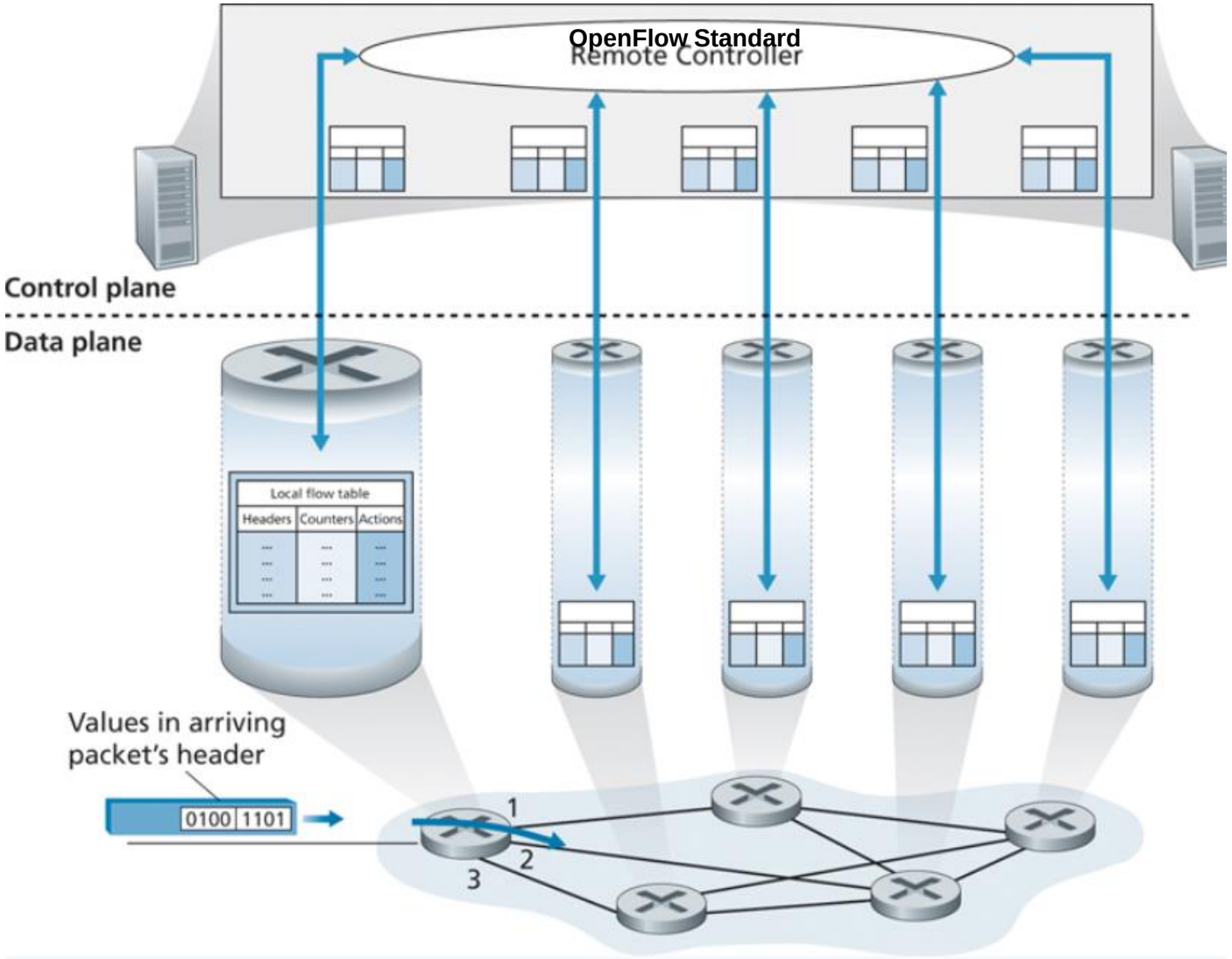
- Hides details of inner home network from outside world
- All incoming traffic will have same public IP, all outgoing will have same public IP

Routers get assigned their one public IP address through DHCP!



Network Layer

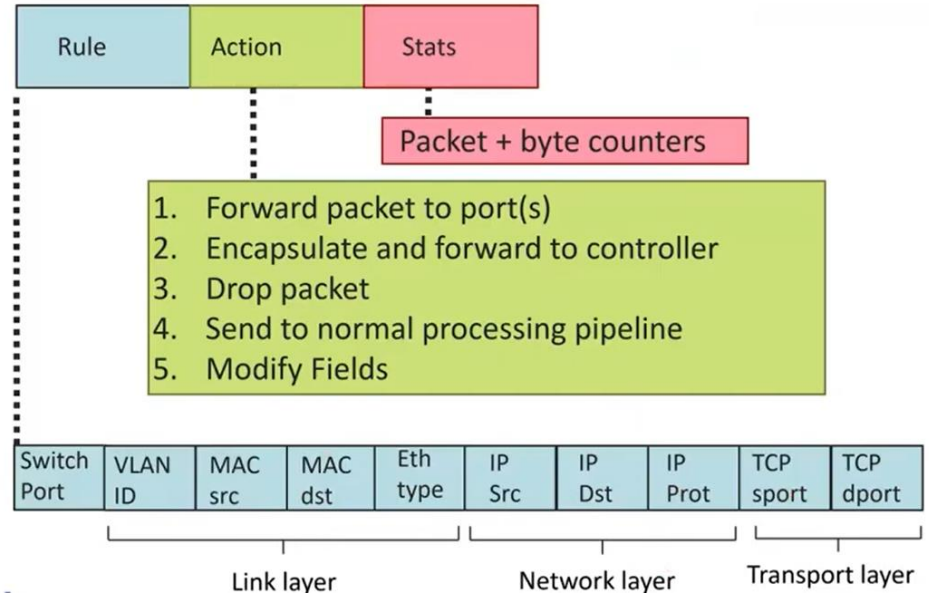
Generalized Forwarding and Software Defines Network (SDN)



We need **headers/rules**, which are going to the values the remote controller is going to evaluate

We need **actions** to do based on some pattern match

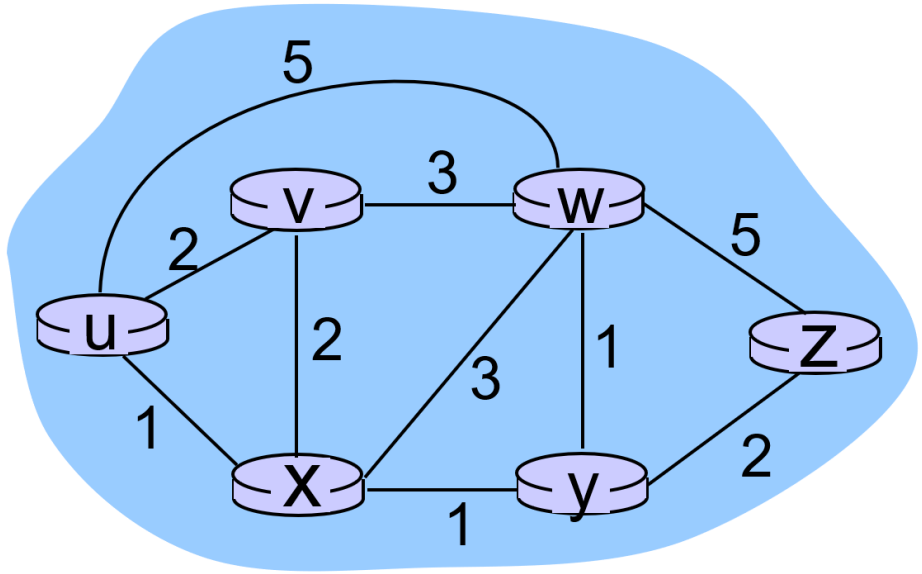
Match	Action
Ingress Port = 1 ; IP Src = 10.3.* .* ; IP Dst = 10.2.* .*	Forward(4)



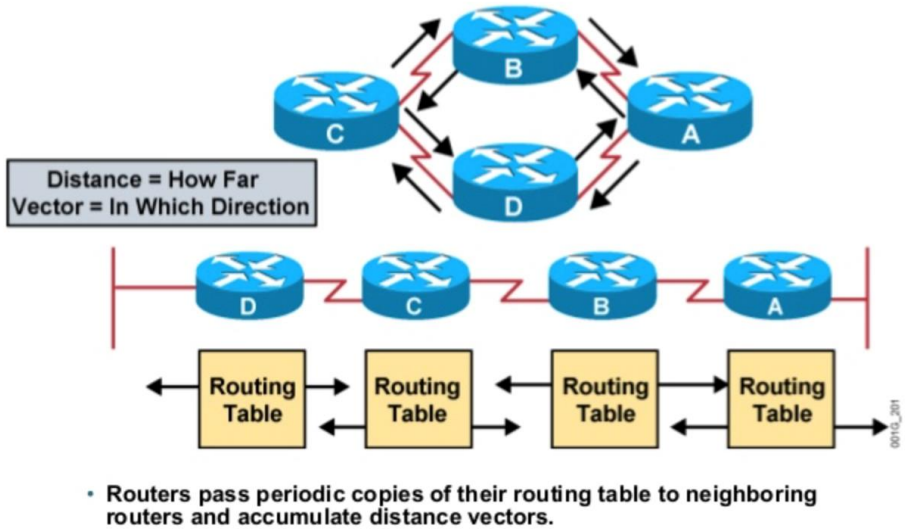
On the final exam, you should know the differences between these two algorithms, and from a high-level how they work

Routing Algorithms

Link State
- AKA **Dijkstra's algorithm**
- Centralized Algorithm (requires edge costs of entire network)
- Updates are event triggered

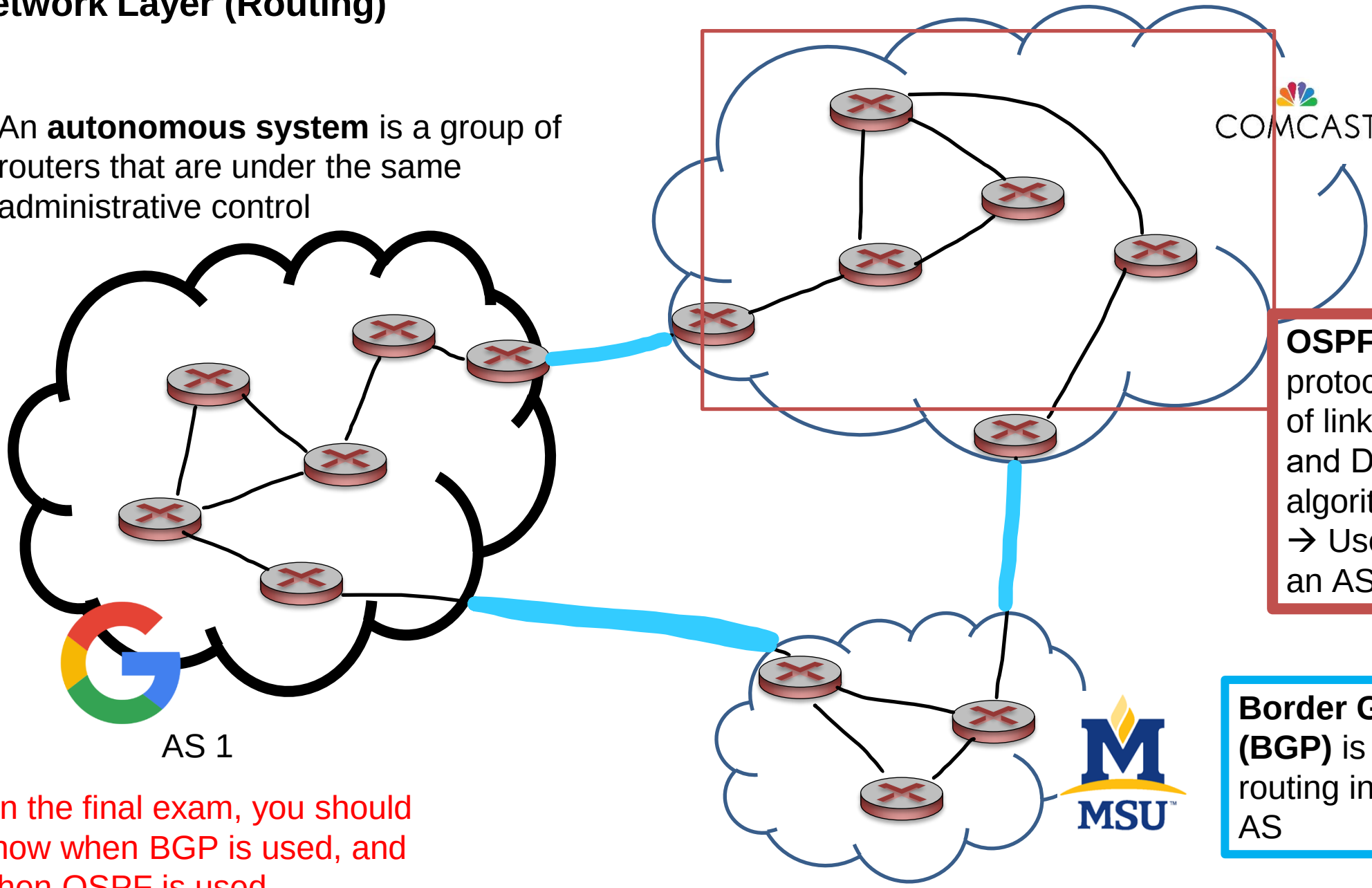


Distance Vector
- Frequently exchange routing table information between neighbors
- Not centralized (does not requires edge costs of entire network beforehand)
- Updates happen frequently



Network Layer (Routing)

An **autonomous system** is a group of routers that are under the same administrative control



AS 1

OSPF is a link-state protocol that uses flooding of link-state information and Dijkstra's least-cost algorithm
→ Used for routing *within* an AS

Border Gateway Protocol (BGP) is used for exchanging routing information *between* AS

On the final exam, you should know when BGP is used, and when OSPF is used

ICMP (Internet Control Message Protocol)

- used by hosts & routers to communicate network-level information
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- network-layer "above" IP:
  - ICMP msgs carried in IP datagrams
- ICMP message: type, code plus first 8 bytes of IP datagram causing error

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Datagrams get encapsulated into **frames**

The link layer is responsible for the **actual node-to-node delivery** of data and ensure error-free transmission of information (handles a variety of mediums)

NIC (Network Interface Controller)- Integrated into the motherboard and allows the machine to use LL services such as ethernet (combination of hardware, software, and some firmware)

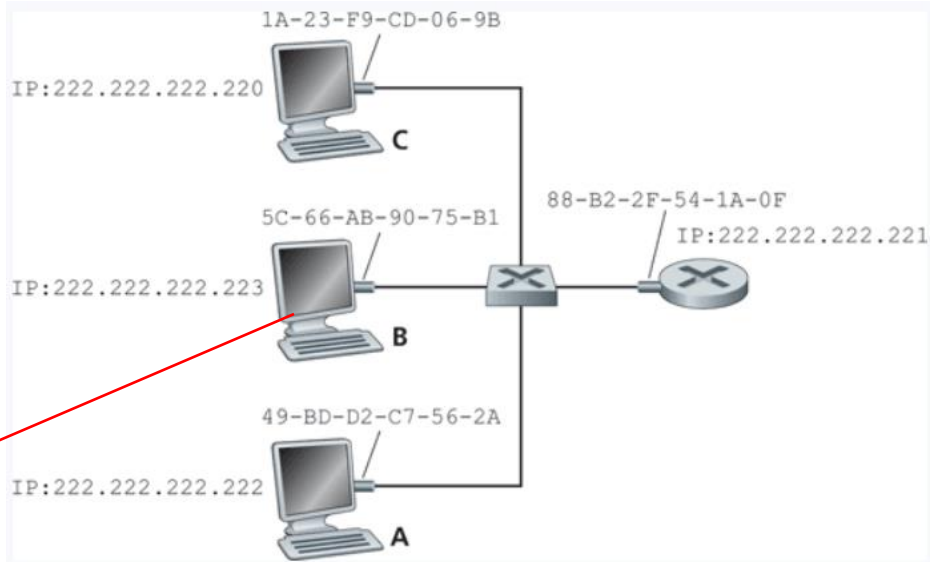
MAC (Media Access Control) Addresses

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

MAC Address is your SSN, IP address is your Postal code ☺

Link Layer



Protocol for mapping **IP Addresses** to **MAC addresses**

Used *only* for hosts and router interfaces **on the same subnet**

First the machine checks its **ARP table**

IP Address	MAC Address	TTL
222.222.222.221	88-B2-2F-54-1A-0F	13:45:00
222.222.222.223	5C-66-AB-90-75-B1	13:52:00

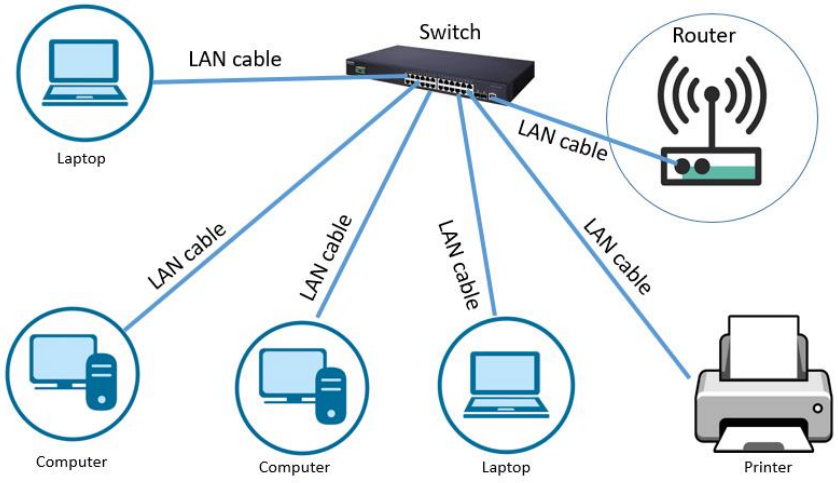
If the entry does not exist in the table, construct and send an **ARP** packet

Broadcasts the ARP packet to all interfaces on the LAN (255.255.255.255)

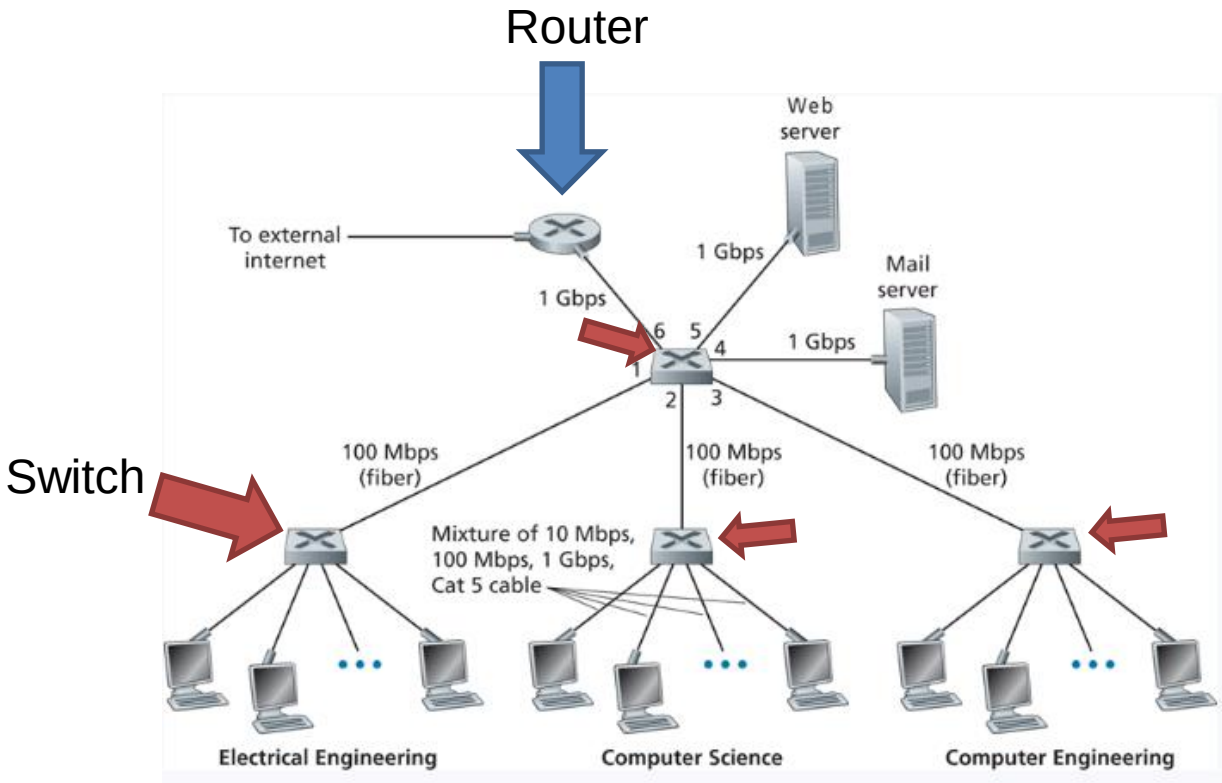
These tables are self-updated, and do not require manual entry*

On the final exam, you should know what ARP is, and when it is used

Local Area Network (LAN)- A collection of devices in one physical location, typically that share a centralized internet connection

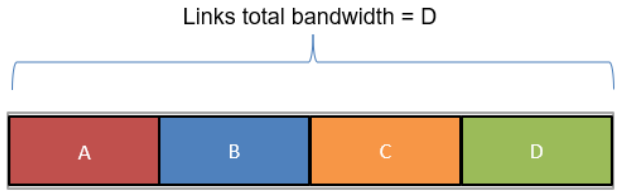


Local Area Network

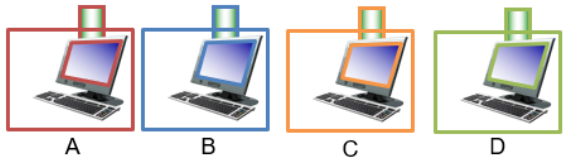


Challenge: Some devices in a LAN will be transmitted data at the same time, thus a **collision** can occur at a receiver

Dealing with Collision: Multiple Access Control Protocols



- Divide channel into **N** slots
- Each node gets (on average) D/N bandwidth
- Get to transmit data for a fixed amount of time, and then next node gets to transmit



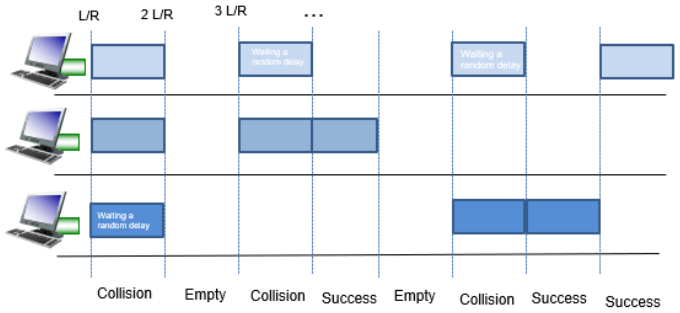
N nodes

Channel Partitioning

Collisions will occur, but we will try to *recover* from them

Slotted ALOHA: Divide up time into discrete L/R "slots"

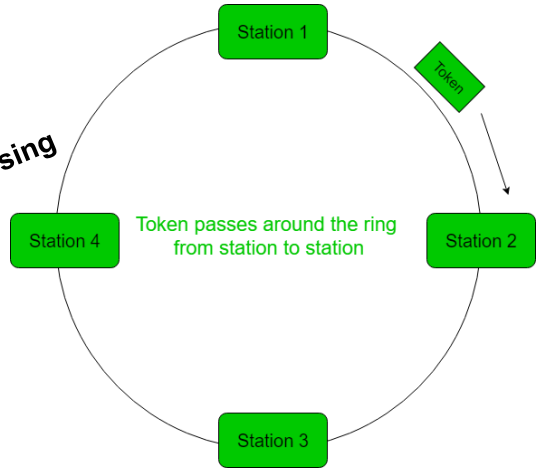
If collisions occur, the colliding nodes will flip a coin to see who should retransmit



L = size of frame
R = Bandwidth

L/R = Time needed to transmit one frame

Token Passing



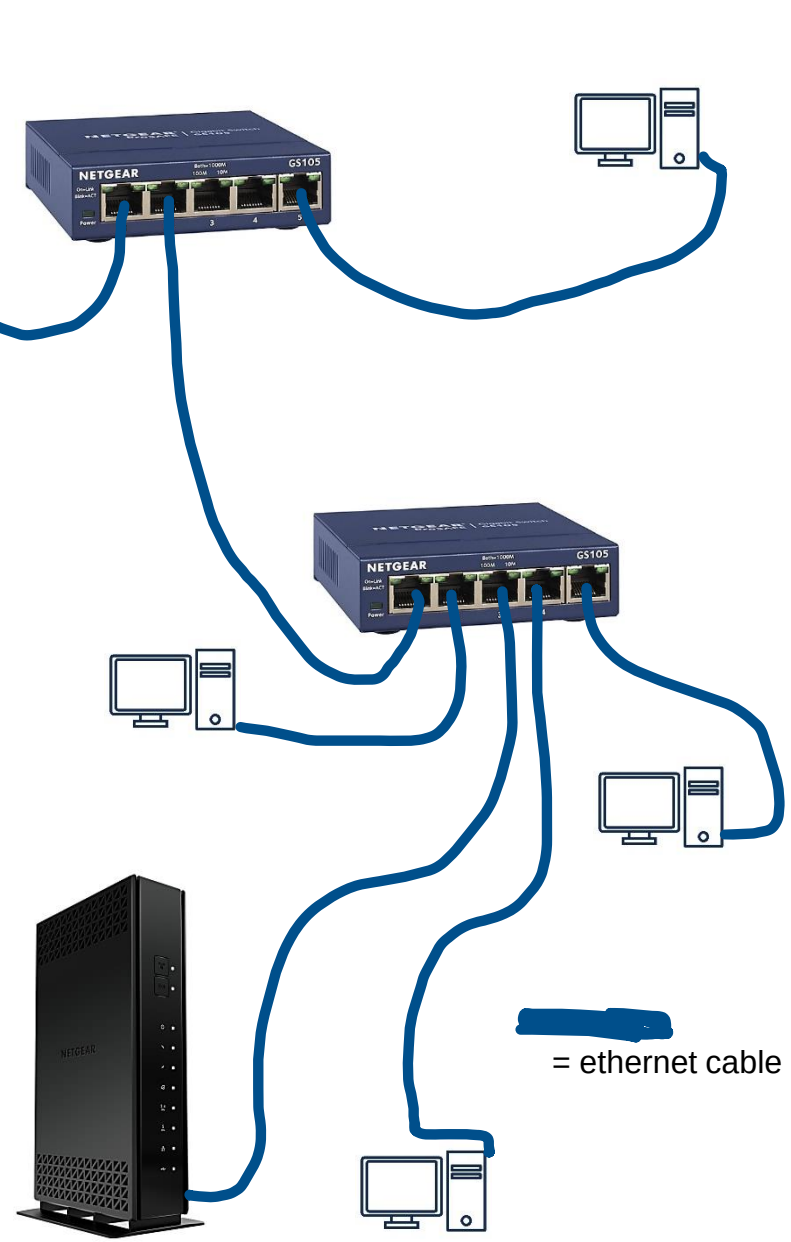
Link Layer

Ethernet: Dominant wired LAN technology

(synchronization) (mac addresses) (ip datagram) (error checking bits)

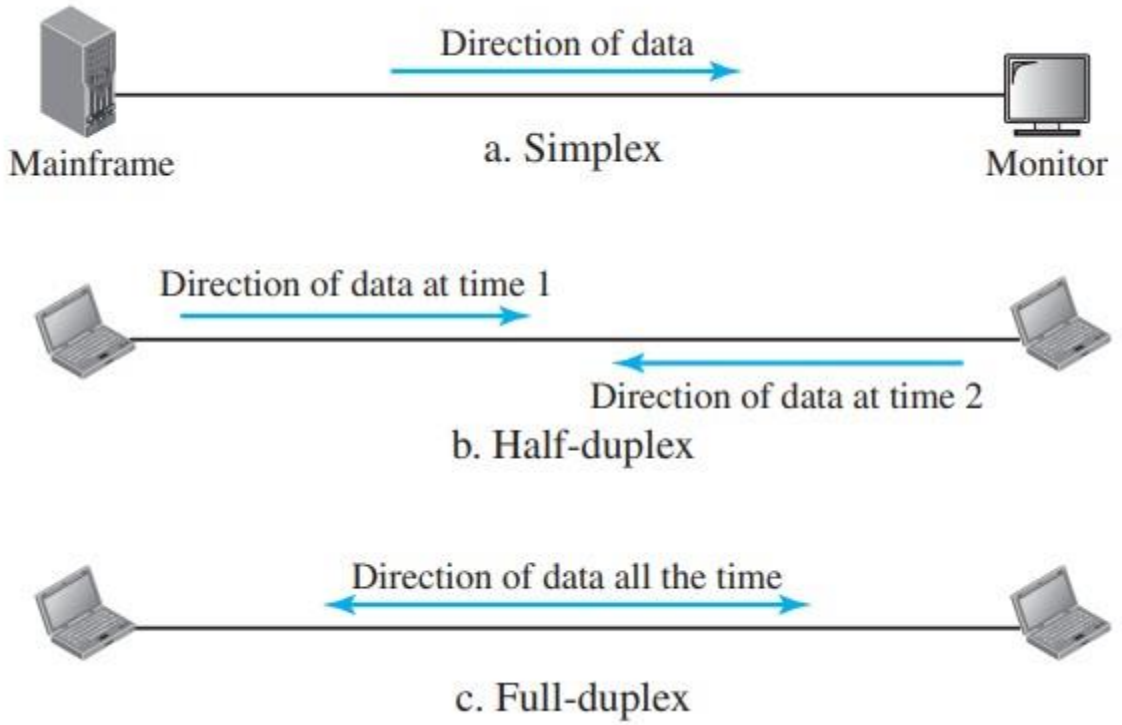
Preamble	SFD	Destination Address	Source Address	Length	Data	CRC
7 Bytes	1 Byte	6 Bytes	6 Bytes	2 Bytes	46 – 1500 Bytes (non extended)	4 Bytes

Ethernet Frame Format



Ethernet switches will store and forward ethernet frames

- Hosts have *dedicated*, direct connection to switch
- Ethernet protocol used on each incoming link, **but no collisions between links**
- Transparent: Hosts are not aware they are connected to a switch
- Plug and play; self-learning


Direction of data
Mainframe
Monitor
a. Simplex
Direction of data at time 1
Direction of data at time 2
b. Half-duplex
Direction of data all the time
c. Full-duplex

Confidentiality- Making sure that only the sender and receiver can read the message (encryption)

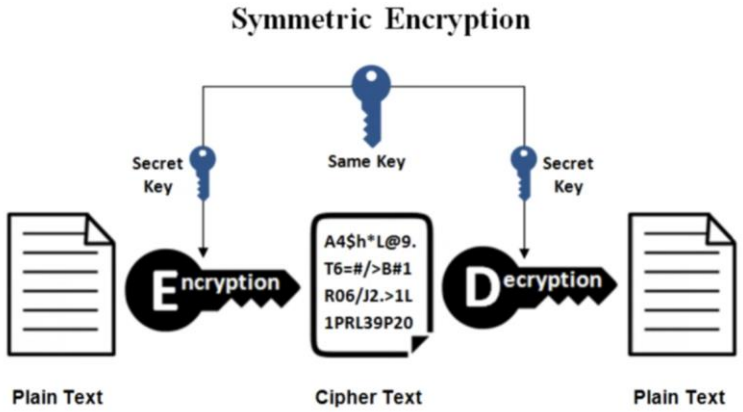
Authentication- Making sure that you are communicating with the person you think you are (encryption + hashing)

Integrity- Making sure the message does not get tampered with during transmission (hashing)

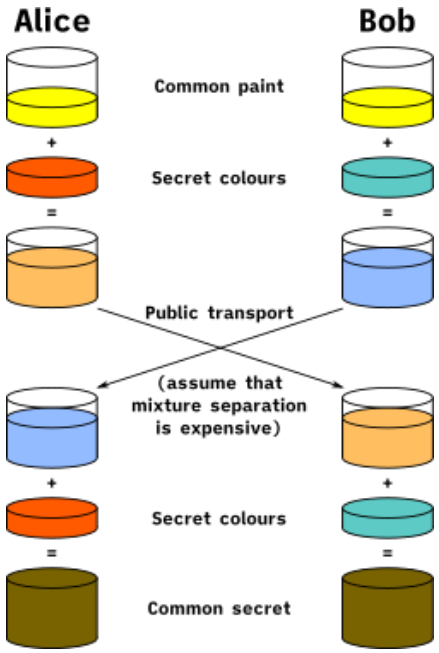
On the final exam, you should know the definition of these three terms

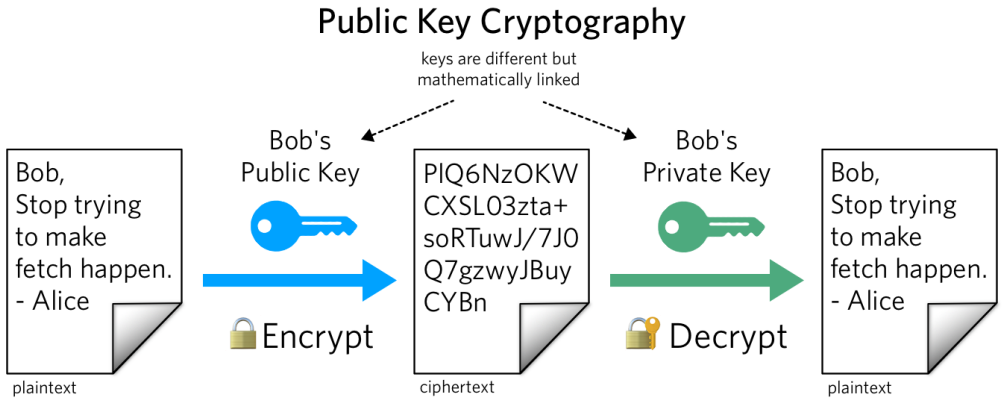
Network Security

On the final exam, you should know the differences and limitations between SCrypto, and ACrypto



- Same key used for encrypting and decrypting
- Using block ciphers (AES), we can encrypt an arbitrary size of data
- Issue: How to securely share secret keys with each other?





- Two keys: Public Key (a lock), and a price key (the key)
- Public key is used to encrypt. Private key used to decrypt message
- Using math, we can securely send messages over an unsecure channel without sharing any sensitive information (Diffie Hellman)
- Issue: We can not encrypt stuff bigger than our key (2048 bits)

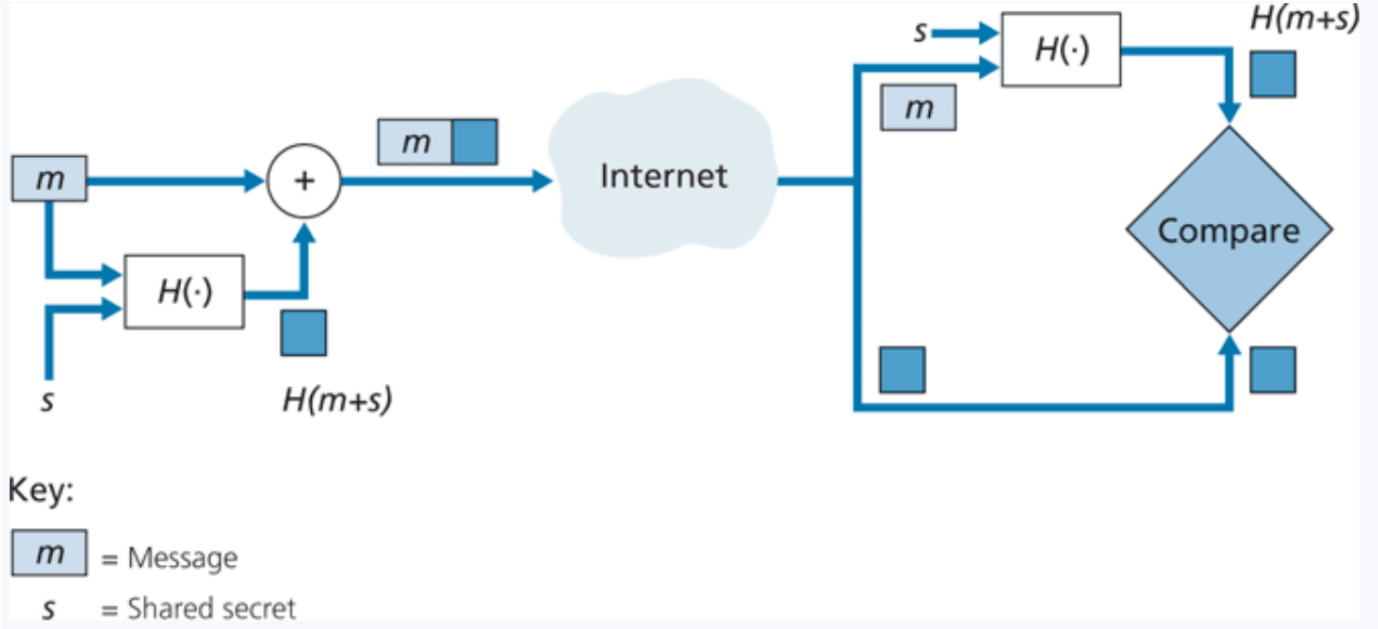
- Often times, symmetric and asymmetric cryptography are used **together**

(use RSA to send the key for symmetric crypto!)

1. Append a message with a shared secret (m + s)
2. Compute hash of (m+s) → H(m+s)
3. Send H(m+s) with message m
4. **Sender sends: (H(m+s), m)**

1. Receiver gets **(H(m+s), m)**
2. Append m with shared secret s (m + s)
3. Compute H(m+s)
4. The value receiver computed should match the H(m+s) he received

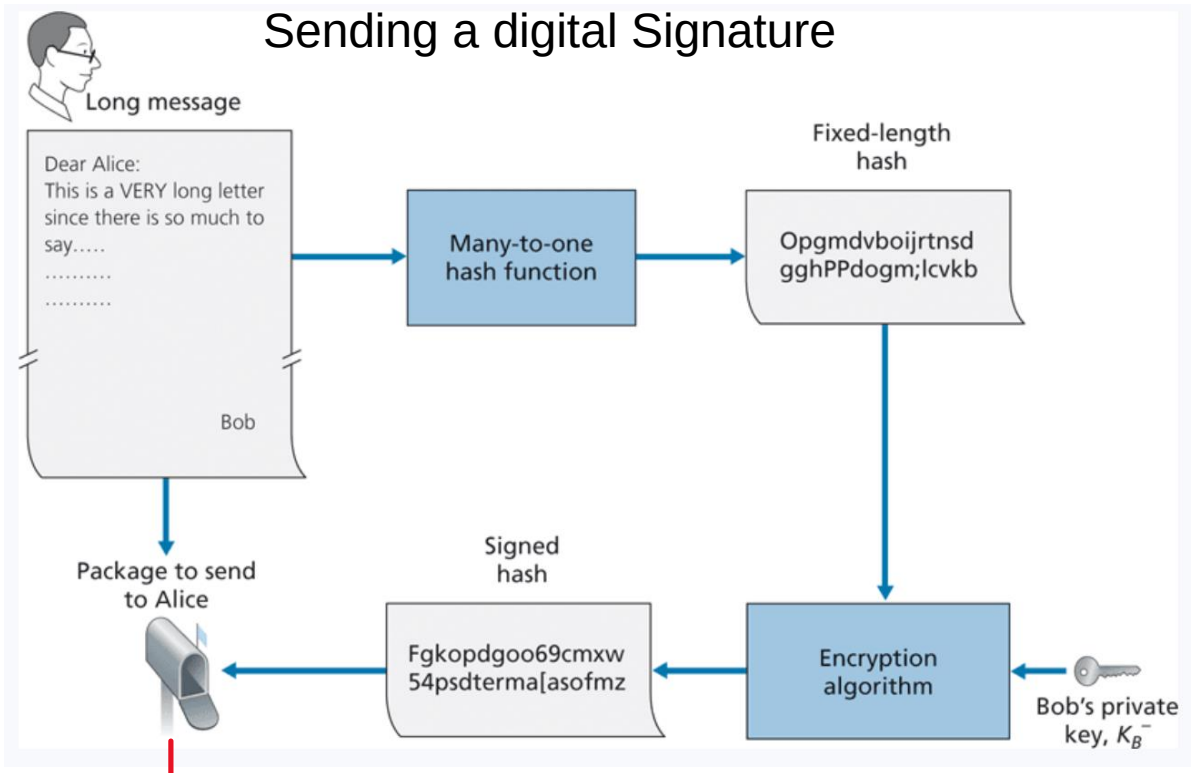
Used for **message integrity**!

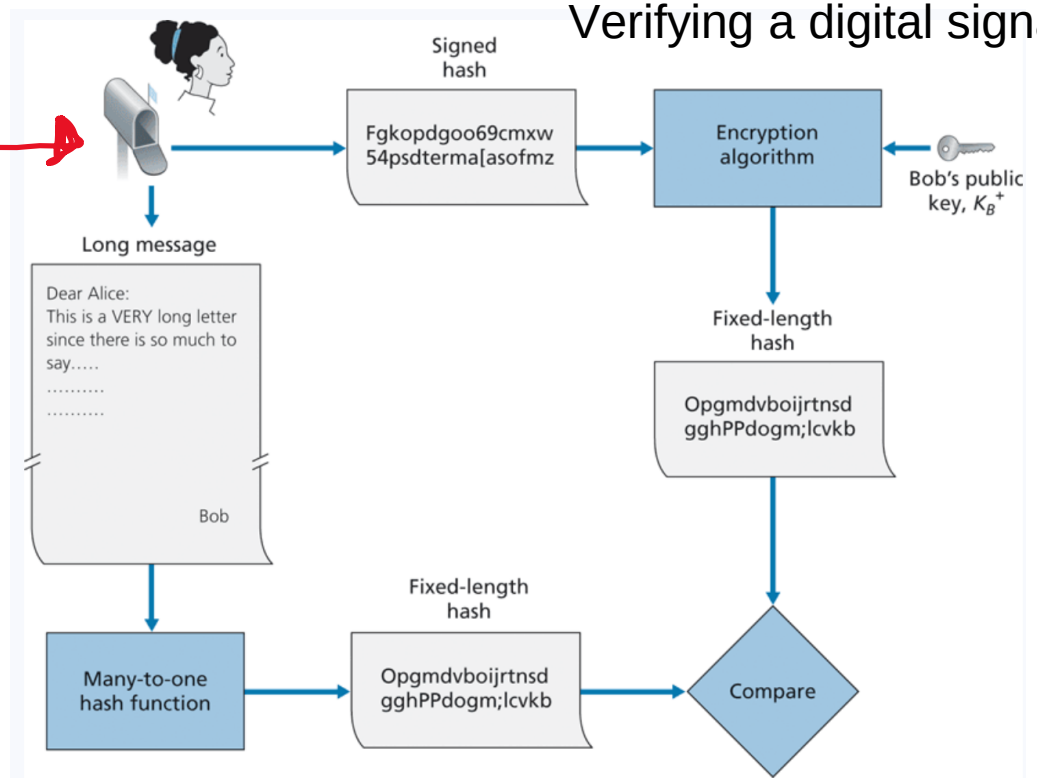


No encryption required!

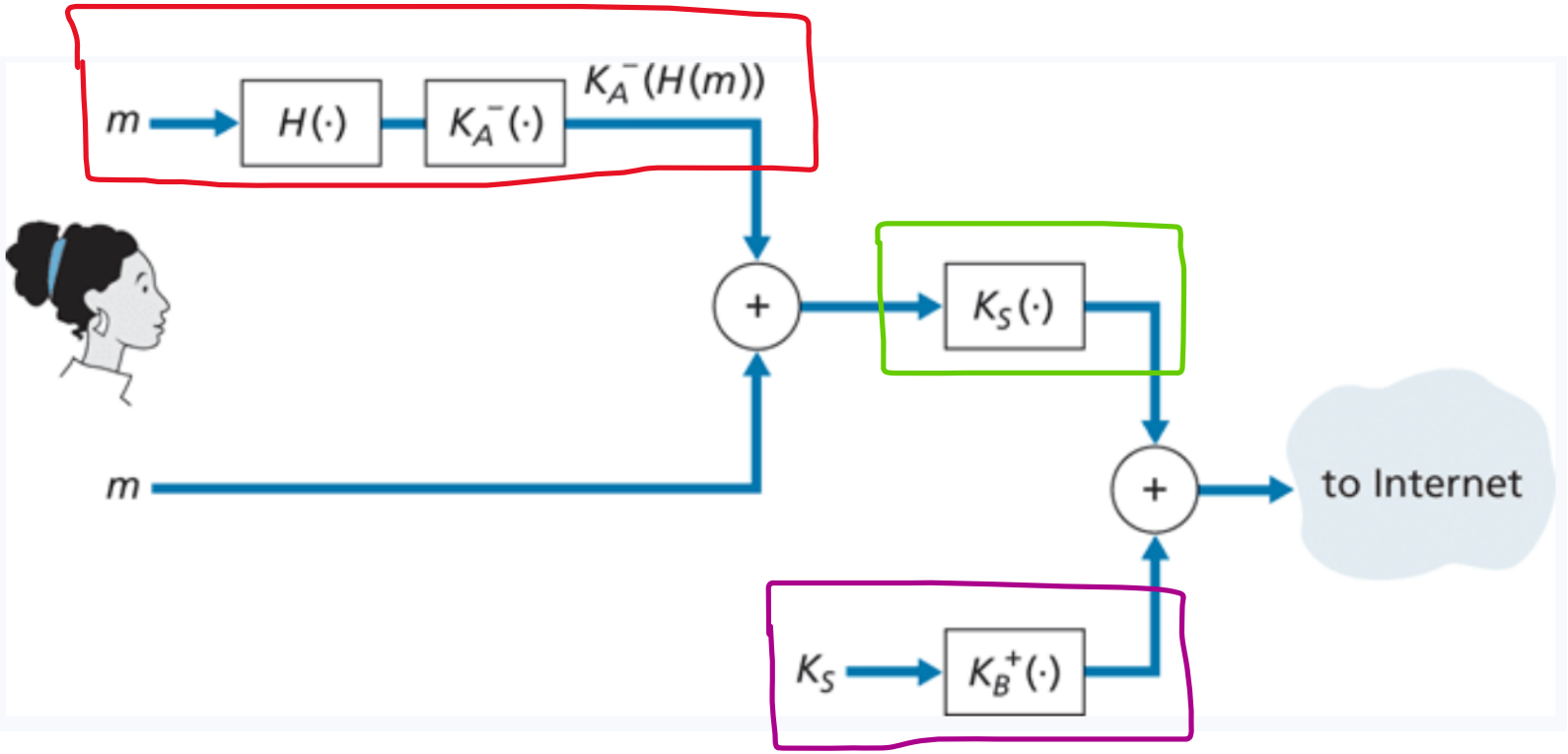
Digital Signatures used both hashing and encryption for authentication

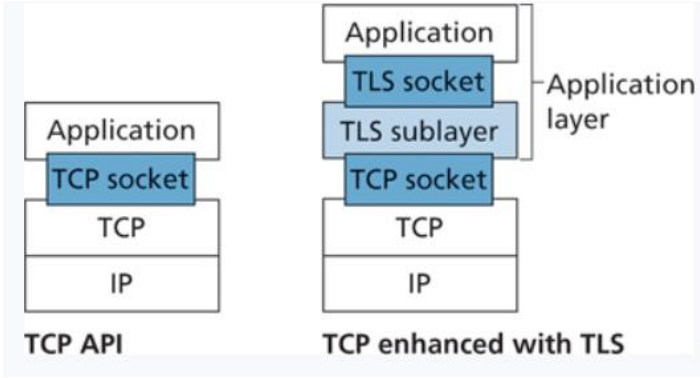
Bob encrypts his hashed message using his **private key**, and sends the signed hash, along with message to Alice. Alice decrypts using his **public key** and verifies that the hashes match





Symmetric Crypto, Asymmetric Crypto, and Hashing all work together to send secure, authentic messages





- **Transport Layer Security (TLS**) is a protocol used to provide communication security over a TCP connection
- This exists somewhere between the application layer and transport layer

TLS will always be running if you are doing web communication with http**s**

Port 443

https = Hypertext Transfer Protocol ***Secure***

HTTPS/TLS will handle all the encryption, key generation, certificate checking, authentication for you!

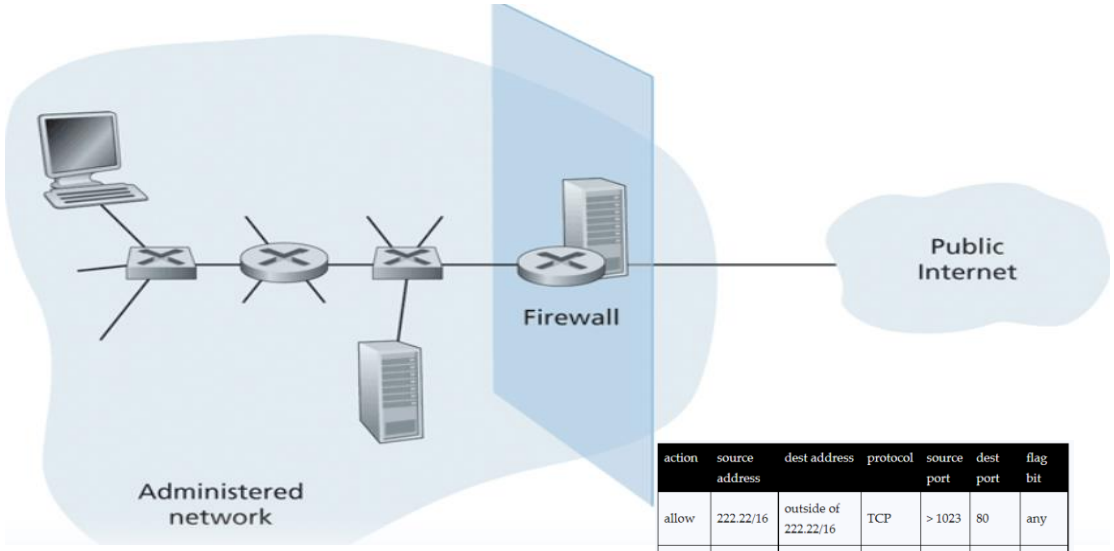
UDP does not use TLS

Endpoint Security

A **firewall** is a combination of hardware and software that isolates an organizations internal network form the internet at large, allowing some packets to pass, and blocking others (Looks at packet information)

An **intrusion prevention system (IPS)** filters out suspicious traffic, **prior** to traffic arriving to network

(Looks at packet information and other signatures)



action	source address	dest address	protocol	source port	dest port	flag bit
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	–
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	–
deny	all	all	all	all	all	all

An **intrusion detection system (IDS)** will generate an alert when potentially malicious traffic is observed

Types of Endpoint Intrusion Systems

- Signature Based
- Anomaly-Based

OSI MODEL

Application Layer

Provides protocols for sending and receiving data between services and web applications (HTTP)

Messages

OSI MODEL

Application Layer	Provides protocols for sending and receiving data between services and web applications (HTTP)	**Messages**
Presentation Layer	Encoding, Compressing, and Encrypting Data	**Messages**

OSI MODEL

Application Layer	Provides protocols for sending and receiving data between services and web applications (HTTP)	**Messages**
Presentation Layer	Encoding, Compressing, and Encrypting Data	**Messages**
Session Layer	Authentication. Manages, monitors, “sessions” between endpoints	**Messages**

OSI MODEL

Application Layer	Provides protocols for sending and receiving data between services and web applications (HTTP)	**Messages**
Presentation Layer	Encoding, Compressing, and Encrypting Data	**Messages**
Session Layer	Authentication. Manages, monitors, “sessions” between endpoints	**Messages**
Transport Layer	Provides host-to-host, **reliable data transfer**, and dictates the flow of data	**Segments**

OSI MODEL

Application Layer	Provides protocols for sending and receiving data between services and web applications (HTTP)	**Messages**
Presentation Layer	Encoding, Compressing, and Encrypting Data	**Messages**
Session Layer	Authentication. Manages, monitors, “sessions” between endpoints	**Messages**
Transport Layer	Provides host-to-host, **reliable data transfer**, and dictates the flow of data	**Segments**
Network Layer	Forwarding and Routing of Data. Logical Addressing	**Datagrams**

OSI MODEL

Application Layer	Provides protocols for sending and receiving data between services and web applications (HTTP)	**Messages**
Presentation Layer	Encoding, Compressing, and Encrypting Data	**Messages**
Session Layer	Authentication. Manages, monitors, “sessions” between endpoints	**Messages**
Transport Layer	Provides host-to-host, **reliable data transfer**, and dictates the flow of data	**Segments**
Network Layer	Forwarding and Routing of Data. Logical Addressing	**Datagrams**
Data Link Layer	Handles the formatting and *physical* addressing of the data before transmitting bits	**Frames**

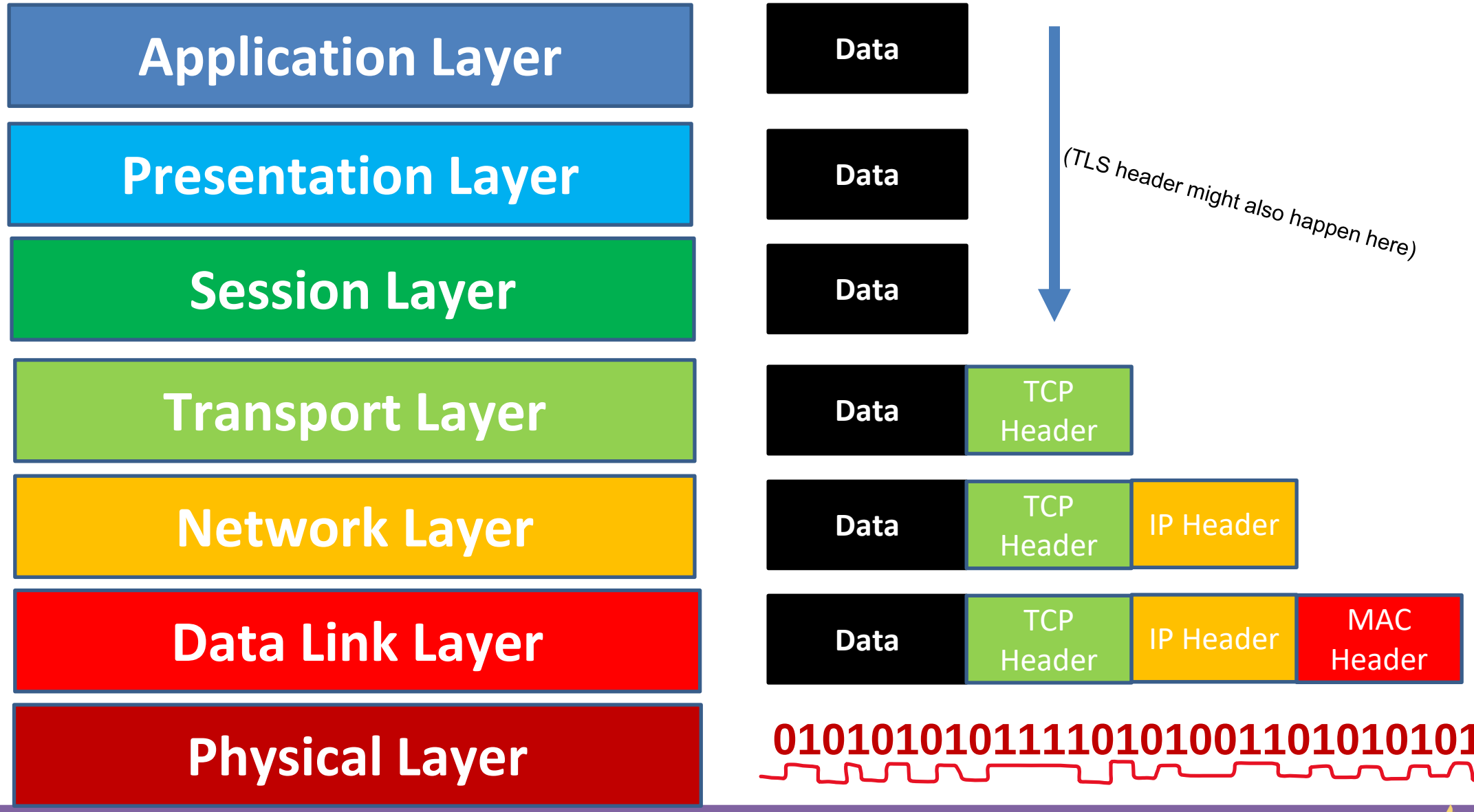
OSI MODEL

Application Layer	Provides protocols for sending and receiving data between services and web applications (HTTP)	**Messages**
Presentation Layer	Encoding, Compressing, and Encrypting Data	**Messages**
Session Layer	Authentication. Manages, monitors, “sessions” between endpoints	**Messages**
Transport Layer	Provides host-to-host, **reliable data transfer**, and dictates the flow of data	**Segments**
Network Layer	Forwarding and Routing of Data. Logical Addressing	**Datagrams**
Data Link Layer	Handles the formatting and *physical* addressing of the data before transmitting bits	**Frames**
Physical Layer	Transmits bits into physical signals over some medium	**Bits**

OSI MODEL

Application Layer	A	Away
Presentation Layer	Penguin	Pizza
Session Layer	Said	Sausage
Transport Layer	That	Throw
Network Layer	Nobody	Not
Data Link Layer	Drinks	Do
Physical Layer	Pepsi	Please

OSI MODEL


Application Layer
Presentation Layer
Session Layer
Transport Layer
Network Layer
Data Link Layer
Physical Layer
Data
Data
Data
(TLS header might also happen here)
Data
TCP Header
Data
TCP Header
IP Header
Data
TCP Header
IP Header
MAC Header
01010101011110101001101010101010

Any Questions?

Thank You!

Thank you for your patience, flexibility, and kindness ☺

This was a class that I never expected to teach ☺

There were a lot of long nights, and I know things were not perfect, but I am happy with how things went

I hope you enjoyed this class, and I hope the stuff you learned will be helpful in your career. Networks concepts are your veggies (boring but important)

I will be teaching 476 and 132 next semester 😎

Connect with me on LinkedIn!

If I can be of assistance to you for anything in the future, please let me know!

Congrats to those that are graduating next weekend! I hope you find a job that you love!