University of Colorado **Boulder**

CSCI 3403 INTRO TO CYBERSECURITY

Lecture: 13-1

Topic: Modern Protocols

Presenter: Matt Niemiec

Announcements

- Project 3 (final project) is posted
- Next homework will be posted Friday or Saturday
- Upcoming guest lecturers
  - From Proofpoint on 4/9
  - From Rule4 on 4/21
  - From Twitter on 4/23 (Andy Sayler)

Exam Extra Credit

- Research a topic that won't be discussed in class
- Get credit in one of the following ways
  - Record a 10-minute (updated) video explanation of your topic for up to 10%. To receive credit for this, see Moodle
  - Present your topic for 5 minutes in recitation for up to 15%. Depending on demand, this may be first-come-first-serve. If you got an 85% or better on the midterm, please leave this for others. Poll opens right after class
  - If you're selected as an outstanding project from recitation, give a 10-minute presentation for up to 25% (total)
- Percentages apply to your higher exam score

Exam Extra Credit Criteria

- Will be graded on at least the following:
  - Interesting topic/information relevant to cybersecurity
  - Quality, professional preparation and presentation
  - Inspires the listener to want to learn more and provides resources to do so
  - Shows insight and depth in research presented in an appropriate manner for the given timeframe
  - Responds knowledgeably and accurately to any questions asked

Some Extra Credit Potential Topics

- Network security
  - Wireless security, honeypots, cloud security, SIEM, Tor
- Applied security
  - OWASP Top 10, reverse engineering, penetration testing
- Crypto
  - Common crypto libraries, homomorphic encryption
- Windows
  - Windows/AD security, Windows CLI
- Miscellaneous
  - Ethics in security, auditing, data provenance
- Or anything else! Just run it by Matt or your TA

Technology Recap 3/17 (Old stuff)

- Piazza is used for content-related questions
- Feedback: https://forms.gle/WRUUbPkmFNsa6q3D6
- Instructor/TA email is used for individual circumstances
- cyber@Colorado.edu is used for accommodations/logistical questions
- Moodle is used for assignments, slides, and additional resources

Technology Recap 3/17 (New stuff)

- Calendar is used for holding all Zoom meetings, instructions, and meeting IDs
  - May contain due dates, but not guaranteed
- Lecture Zoom ID: https://cuboulder.zoom.us/j/633893668
  - This and others found in Google Calendar
- Lecture capture folder: https://drive.google.com/drive/folders/1VMrHEigP4AgDwRnRPTsgQS35EAozc19-?usp=sharing

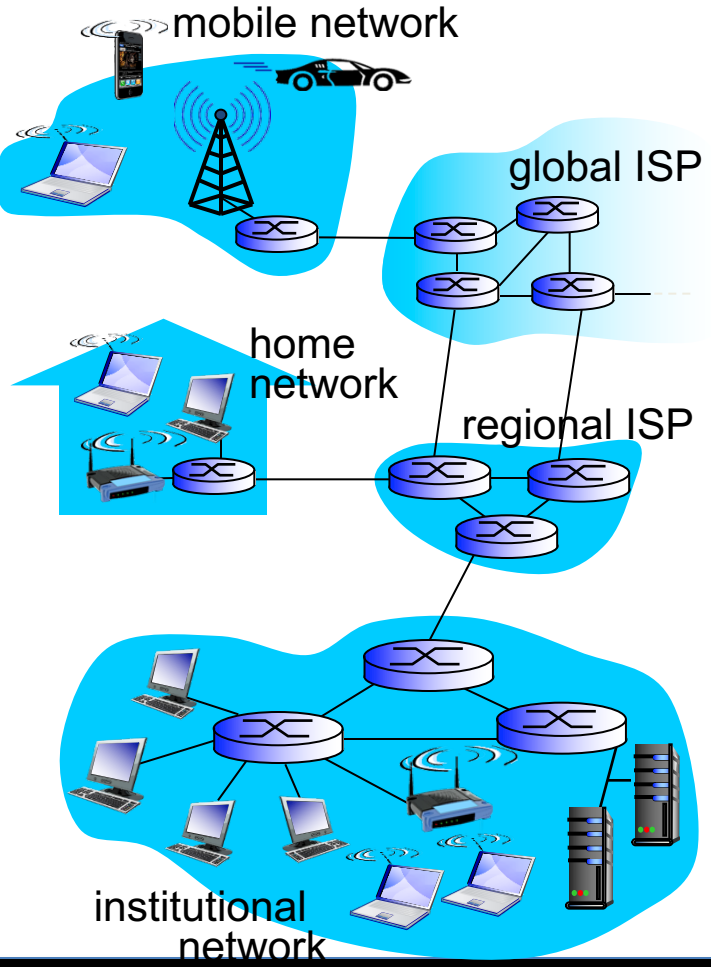
Networking Recap

Networking

- Computers talk to each other on the Internet
- We need to look at the following:
  - What's a computer network?
- We're looking at a computer network in terms of layers

A Look at Network Structure:

- Network edge:
  - Hosts: clients and servers
  - Servers often in data centers
- Access networks, physical media: wired, wireless communication links
- Network core:
  - Interconnected routers
  - Network of networks



What's a Protocol?

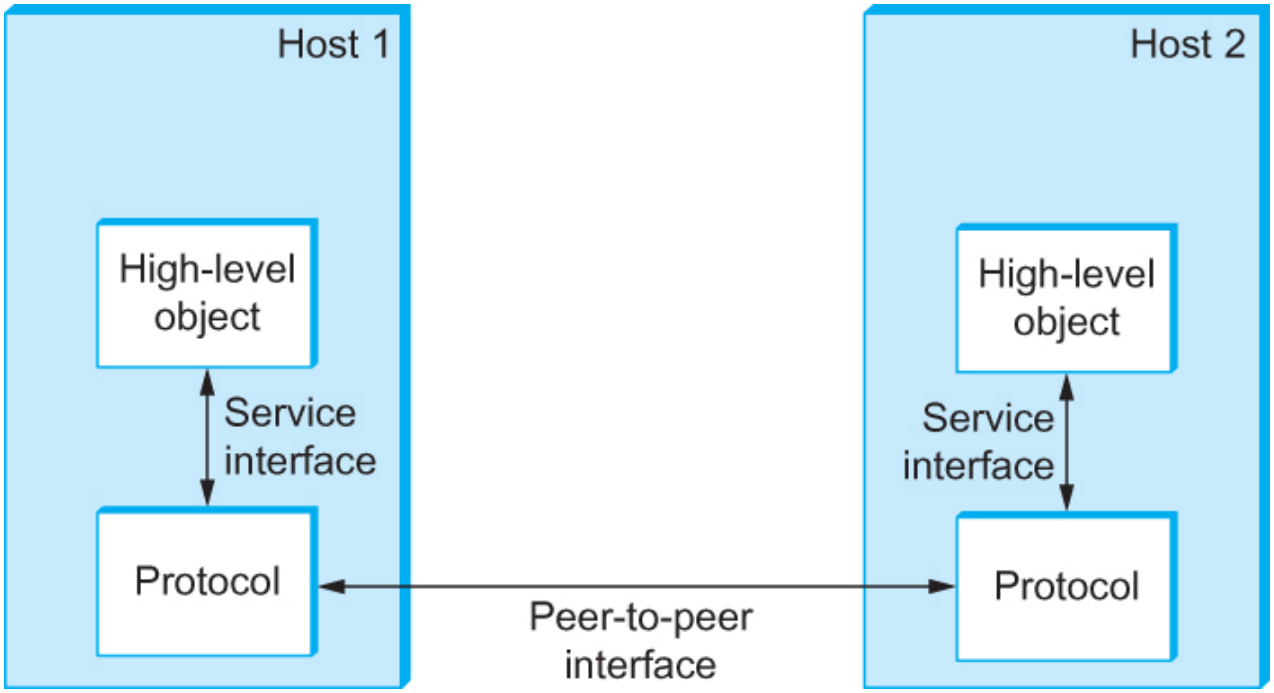
- Human protocols:
  - "What's the time?"
  - "I have a question"
  - Introductions

- … Specific messages sent
- … Specific actions taken when messages received, or other events

- Network protocols:
  - Machines rather than humans
  - All communication activity in Internet governed by protocols

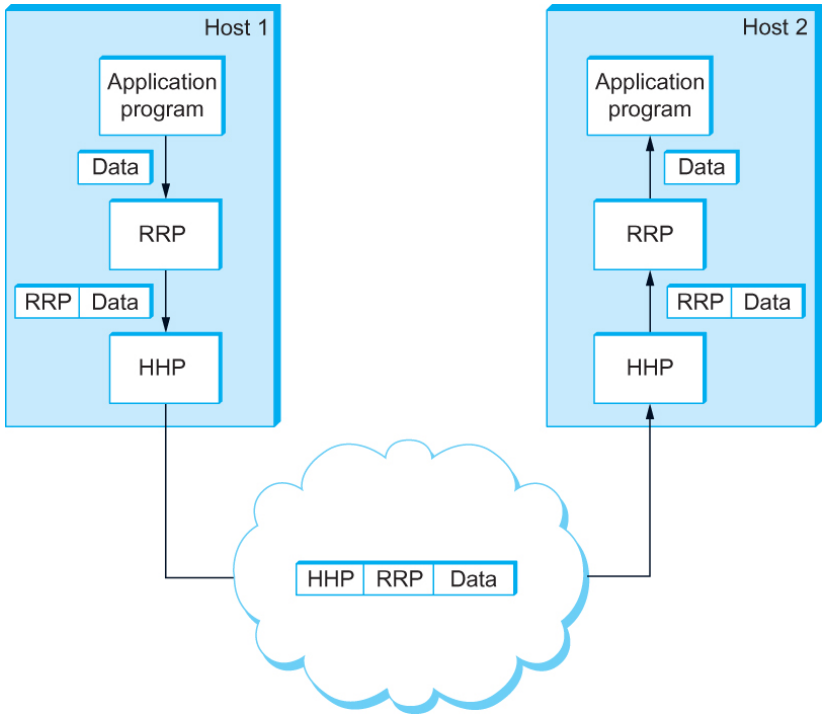
Protocols define format, order of messages sent and received among network entities, and actions taken on msg transmission, receipt

Interfaces

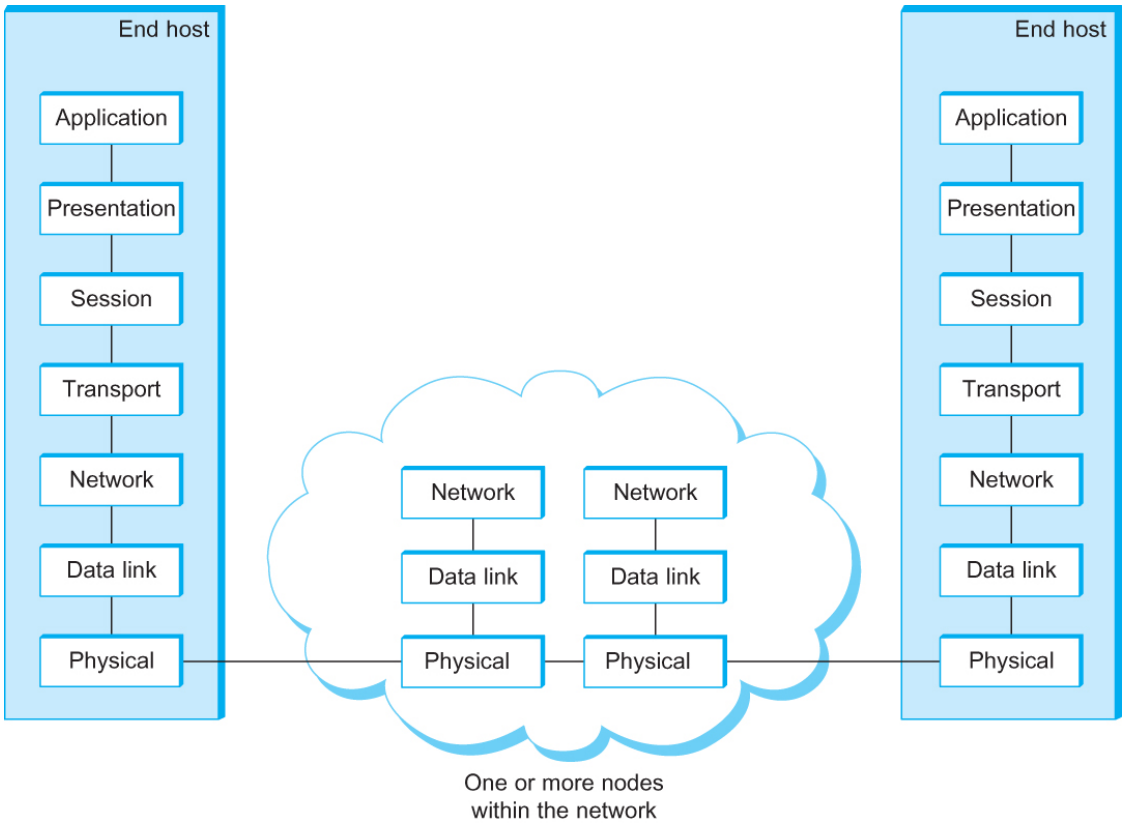


Service and Peer Interfaces

Encapsulation



High-level messages are encapsulated inside of low-level messages



The OSI 7-layer Model

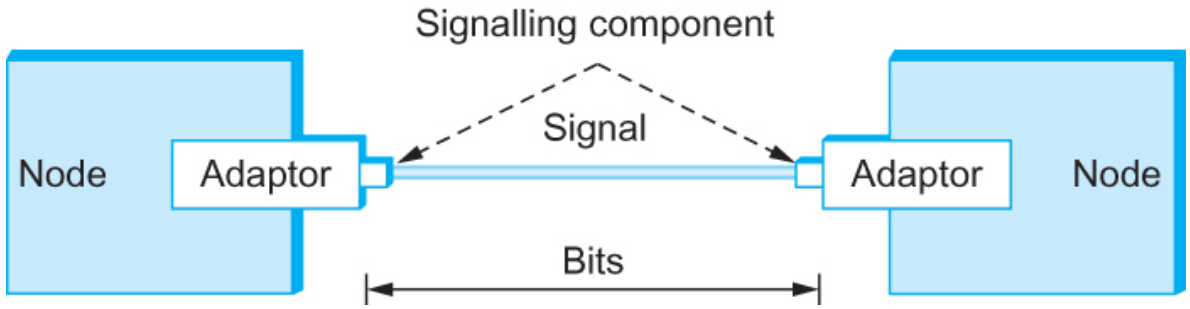
OSI – Open Systems Interconnection

Physical Layer

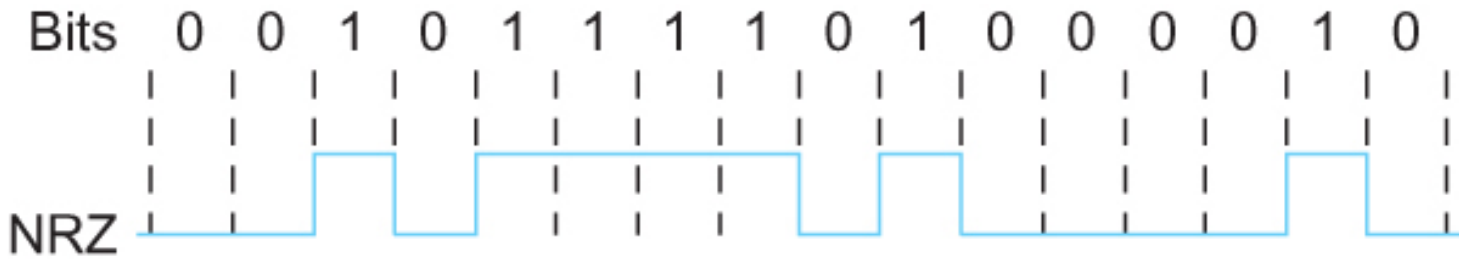
- Sends 1s and 0s across a physical/wireless link
- Ensure reliability
- Placing binary data on a link is called *encoding*
  - We will know that encoding exists, but not deal with it directly
- Provides framing
  - Gives metadata in order to better transmit (e.g. checksums, error detection/correction, number of bytes being transmitted in frame)

- Example: I want to send you 60 bytes of data. Here

Encoding: Most Basic



Signals travel between signaling components; bits flow between adaptors



NRZ encoding of a bit stream

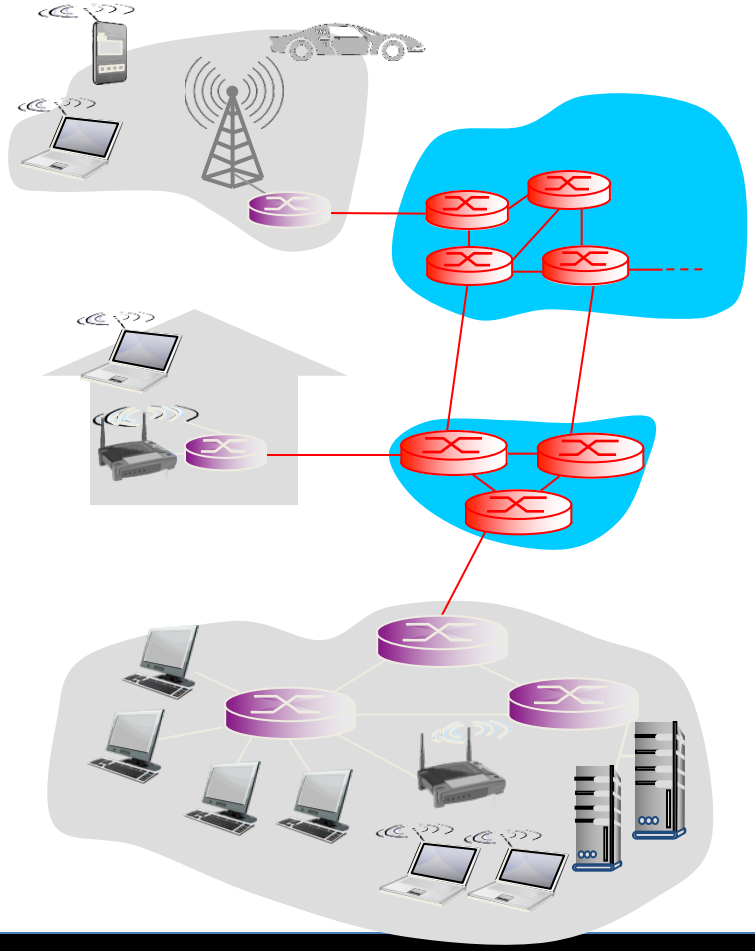
Data Link Layer

- Provides some basic frame distribution
- Introduces *switching* and *bridging*, but not yet routing
  - We won't get caught up with these
- Begins providing scalability
- Transfers are performed based on the physical address of the recipient
  - Media Access Control (MAC)

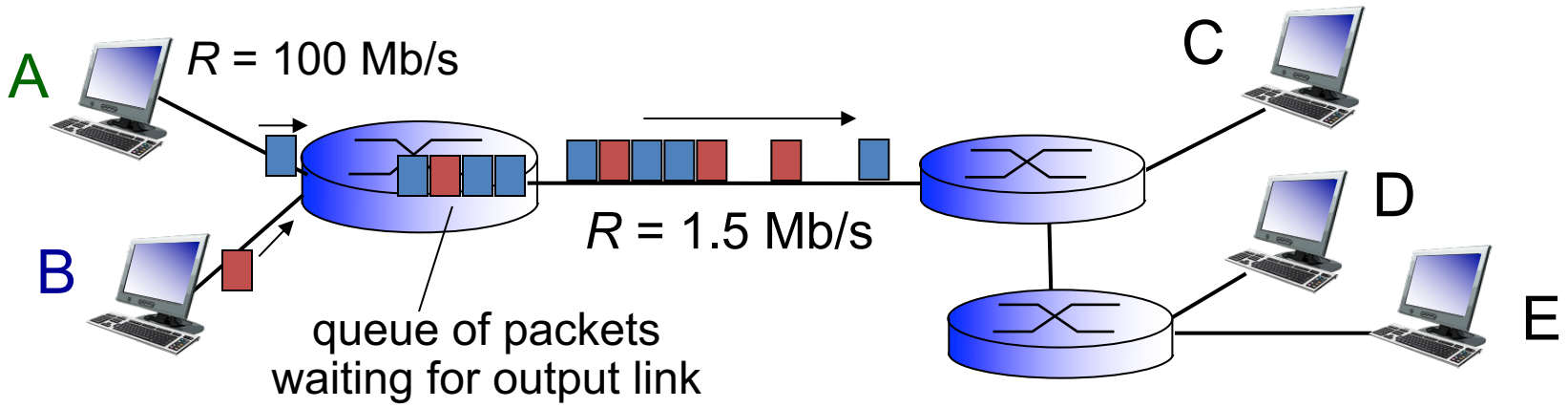
- Example: Hello! You are directly connected to Bob, or know exactly how to get to him. Please deliver these 60 bytes of data to him

The Network Core

- Mesh of interconnected routers
- Packet-switching: hosts break application-layer messages into packets
  - Forward packets from one router to the next, across links on path from source to destination
  - Each packet transmitted at full link capacity



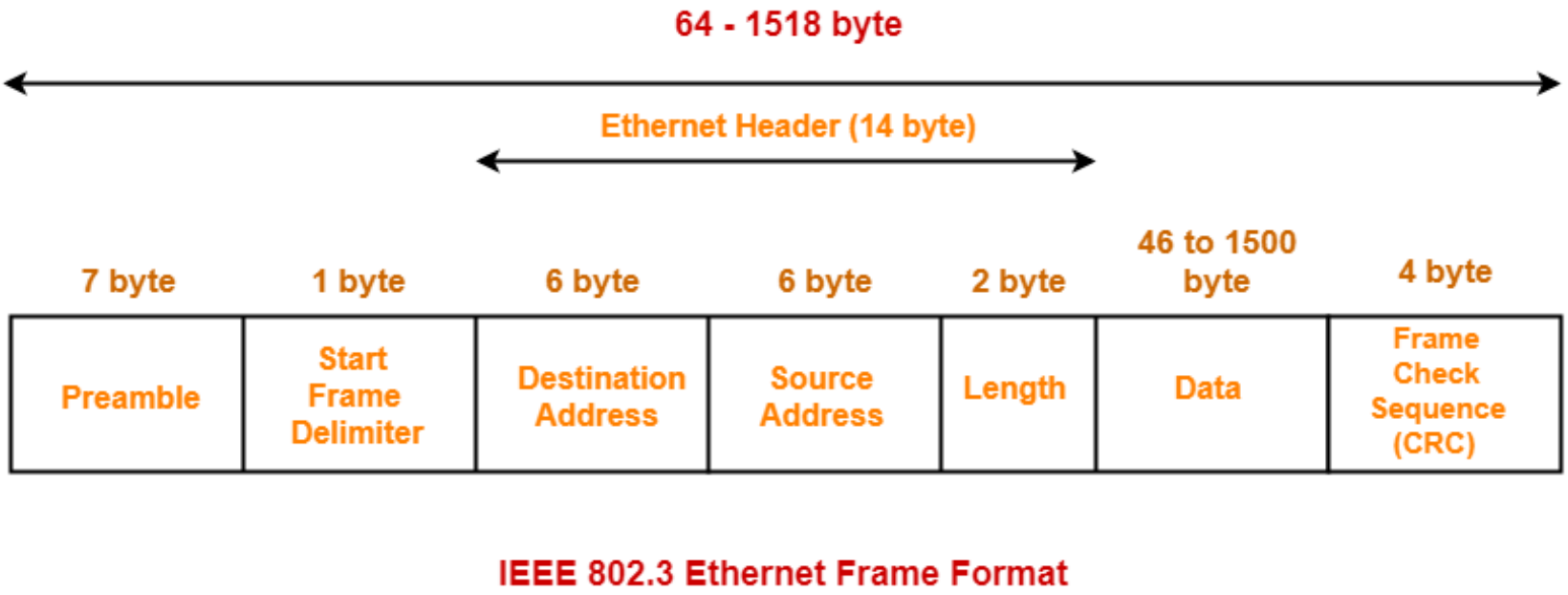
Packet Switching: Queueing Delay, Loss



Queuing and loss:

- If arrival rate (in bits) to link exceeds transmission rate of link for a period of time:
  - Packets will queue, wait to be transmitted on link
  - Packets can be dropped (lost) if memory (buffer) fills up

Current Transmission Example



IEEE 802.3 Ethernet Frame Format

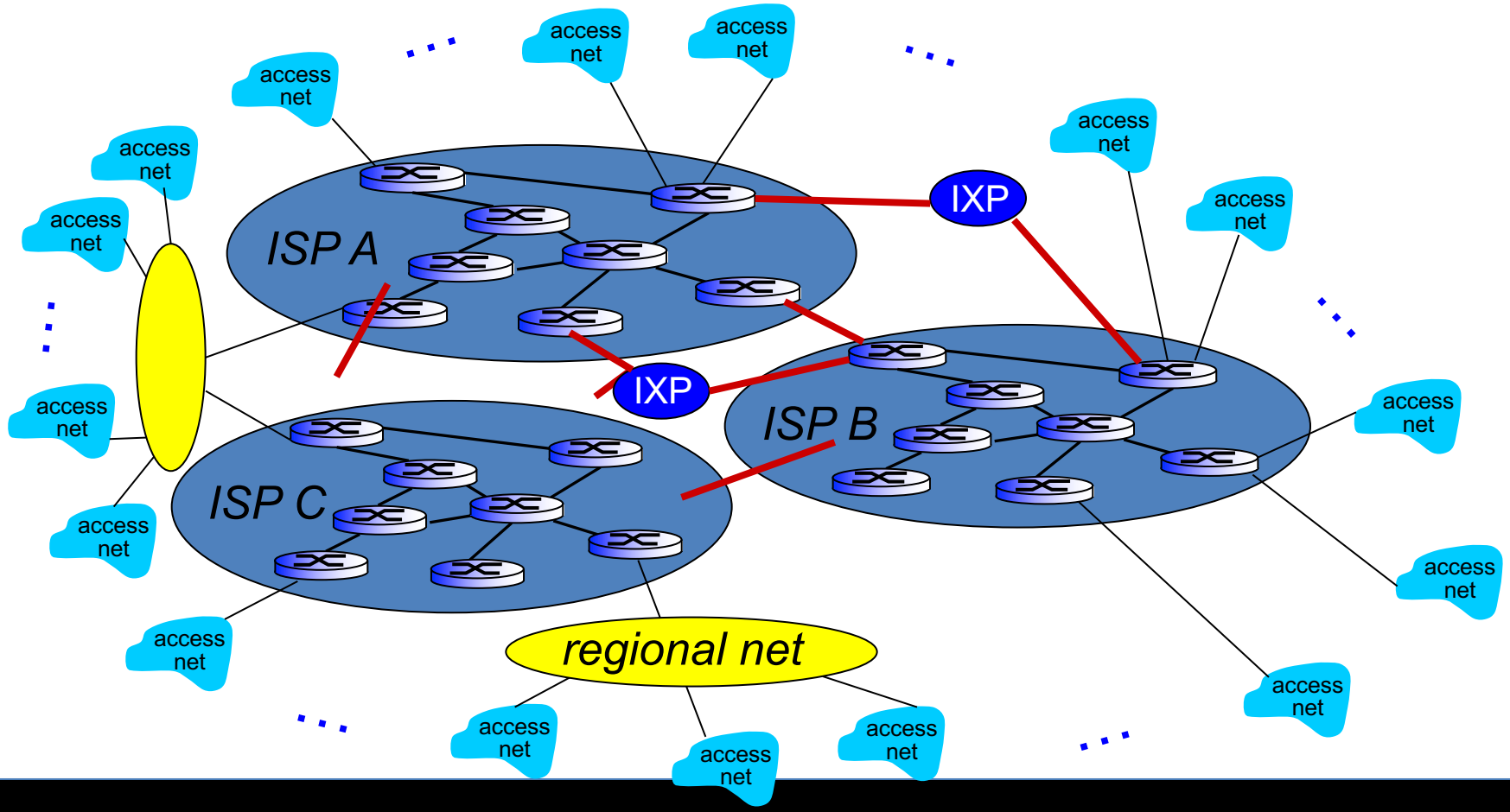
Network Layer

- This is the “Internet of Internets”
- Protocols in this layer: **IP**
- Programs such as *ping* and *traceroute* operate here
- Transfers can no longer be done with physical address
  - Must use IP address

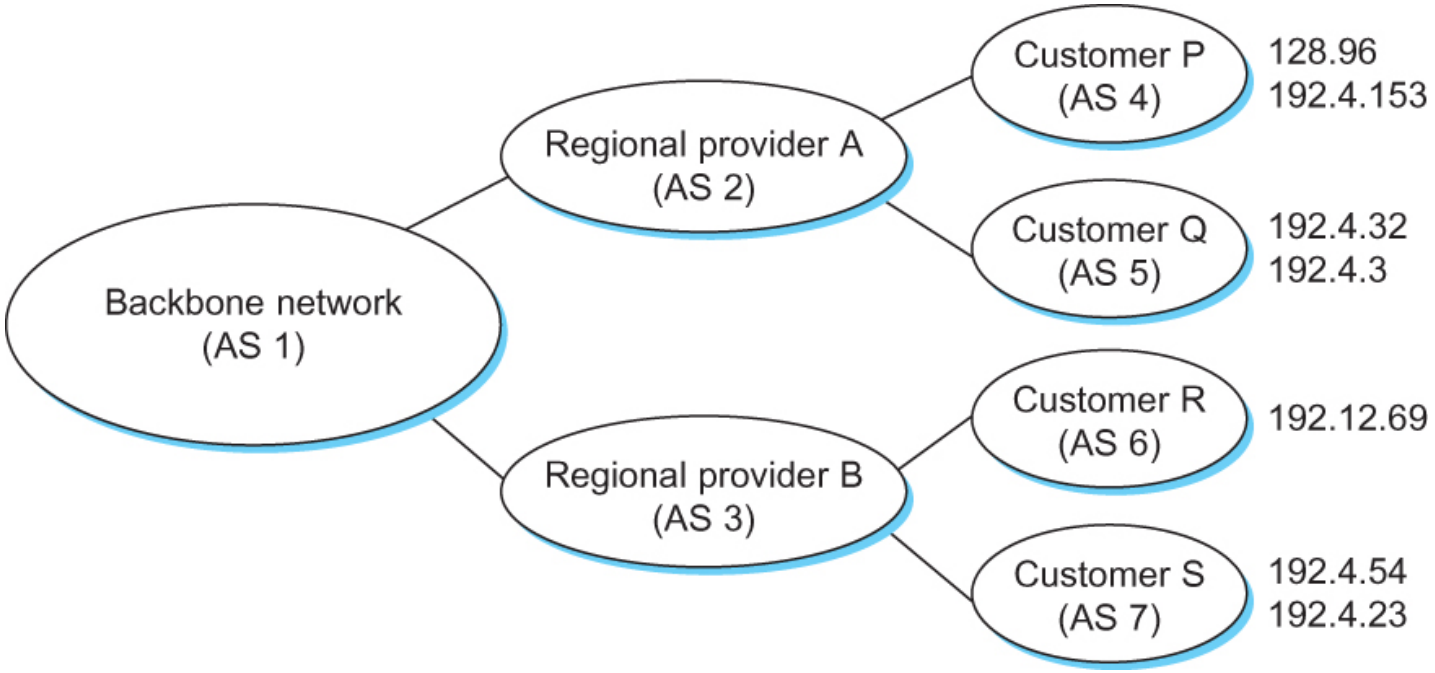
- Example: Here’s an address for Bob and 60B of data. Discuss with your neighbors to find a good route to Bob, then pass it along until it reaches him

Internet Structure: Network of Networks

...and regional networks may arise to connect access nets to ISPs



BGP Example

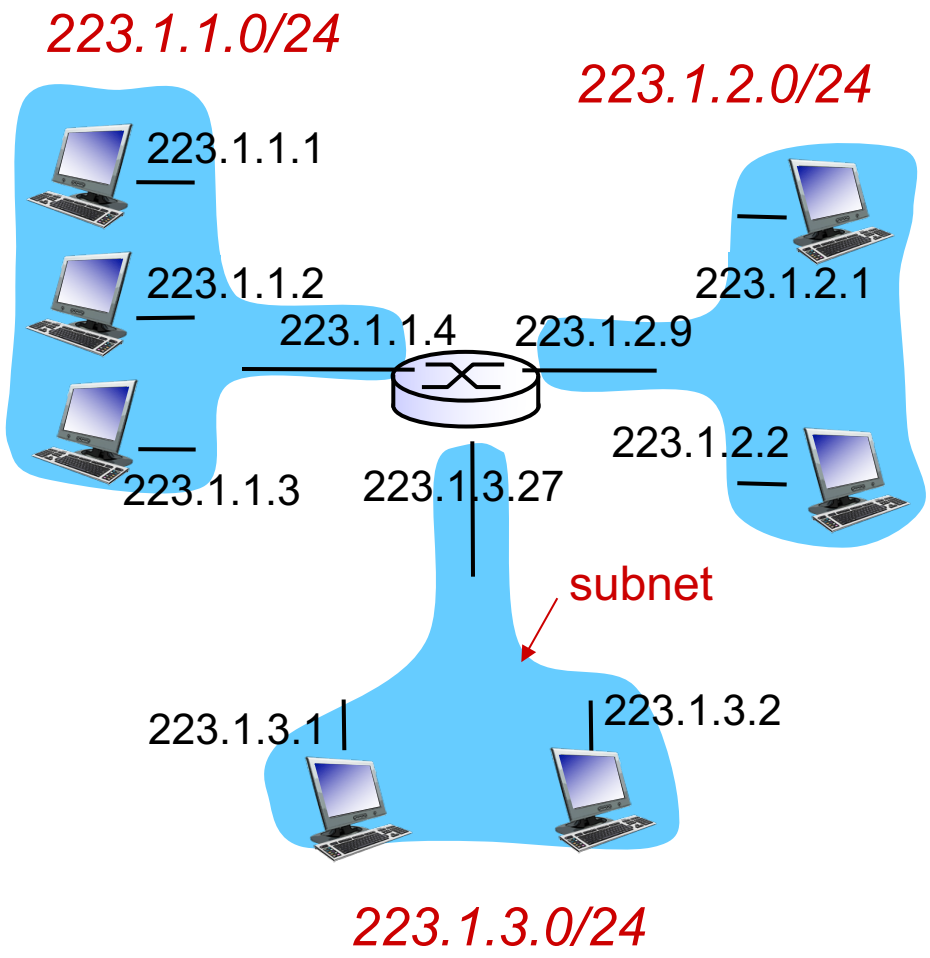


Example of a network running BGP

Subnets

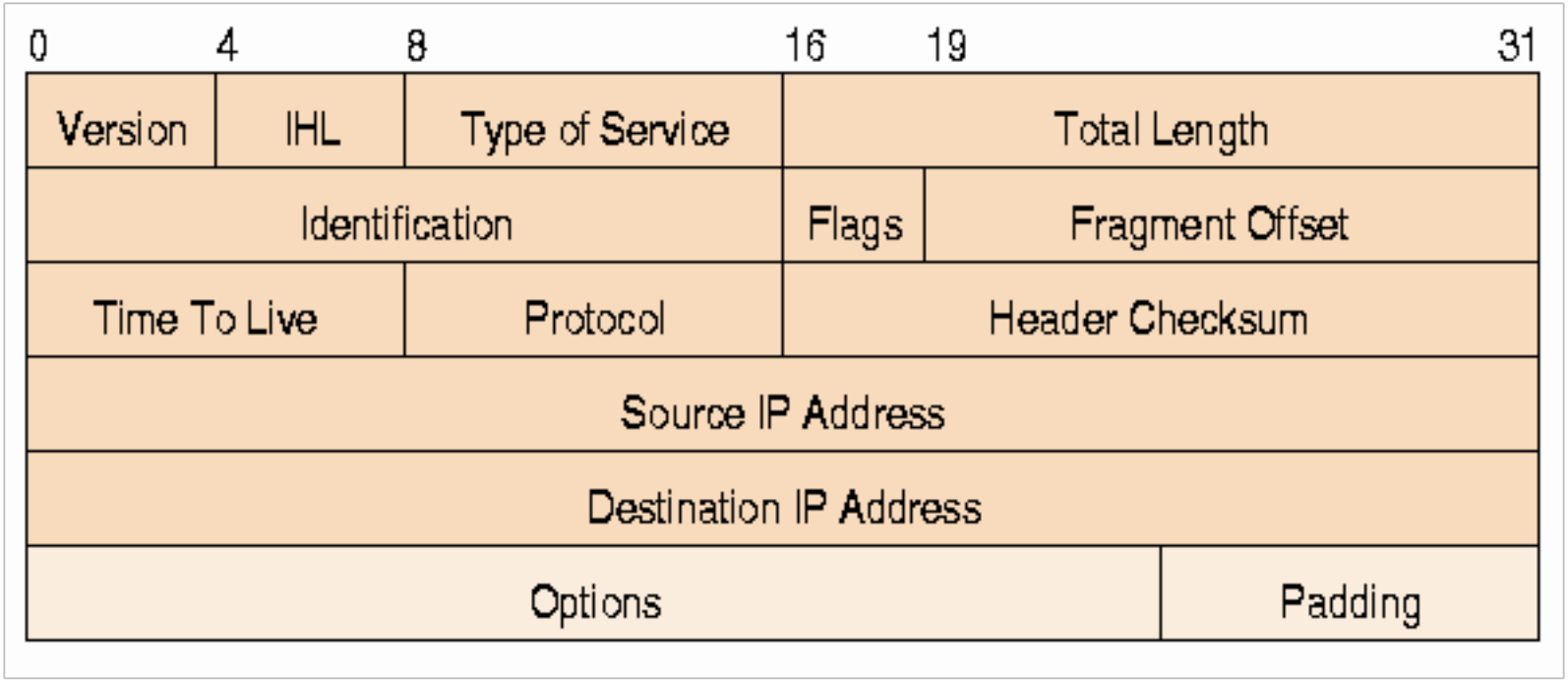
Recipe

- To determine the subnets, detach each interface from its host or router, creating islands of isolated networks
  - Each isolated network is called a subnet



subnet mask: /24

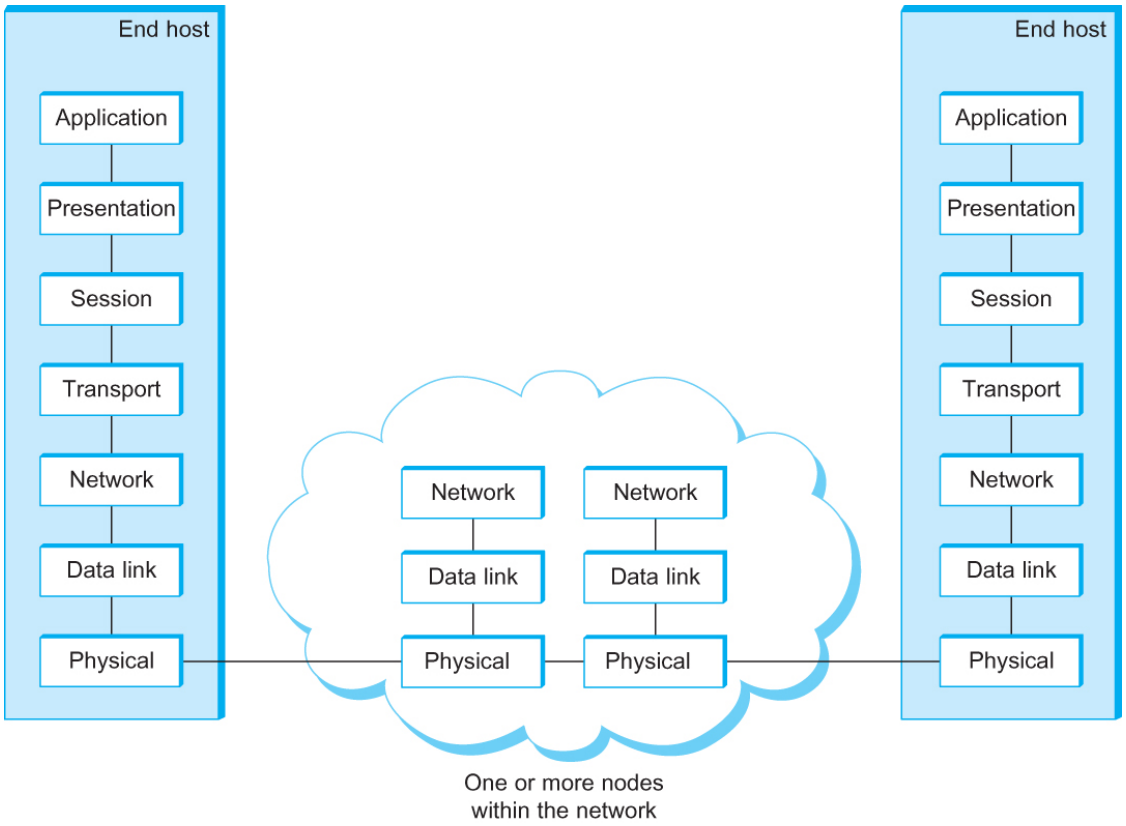
Current Packet Header



*Source: https://freesoft.org

Layer 4: Transport Layer

*Slides adapted from Computer Networking: A top-Down Approach



The OSI 7-layer Model

OSI – Open Systems Interconnection

So Far...

- We can send packets anywhere
- We can manage and scale the internet to hold billions of hosts
- We can guarantee a route from one person to another
- Sounds great!

So Far...

- We can send packets anywhere
- We can manage and scale the internet to hold billions of hosts
- We can guarantee a route from one person to another
- Sounds great!
- Problems?

So Far...

- We can send packets anywhere
- We can manage and scale the internet to hold billions of hosts
- We can guarantee a route from one person to another
- Sounds great!
- Problems?
  - Our connections are unreliable
  - We can only direct info to a machine, not to different processes on that machine!

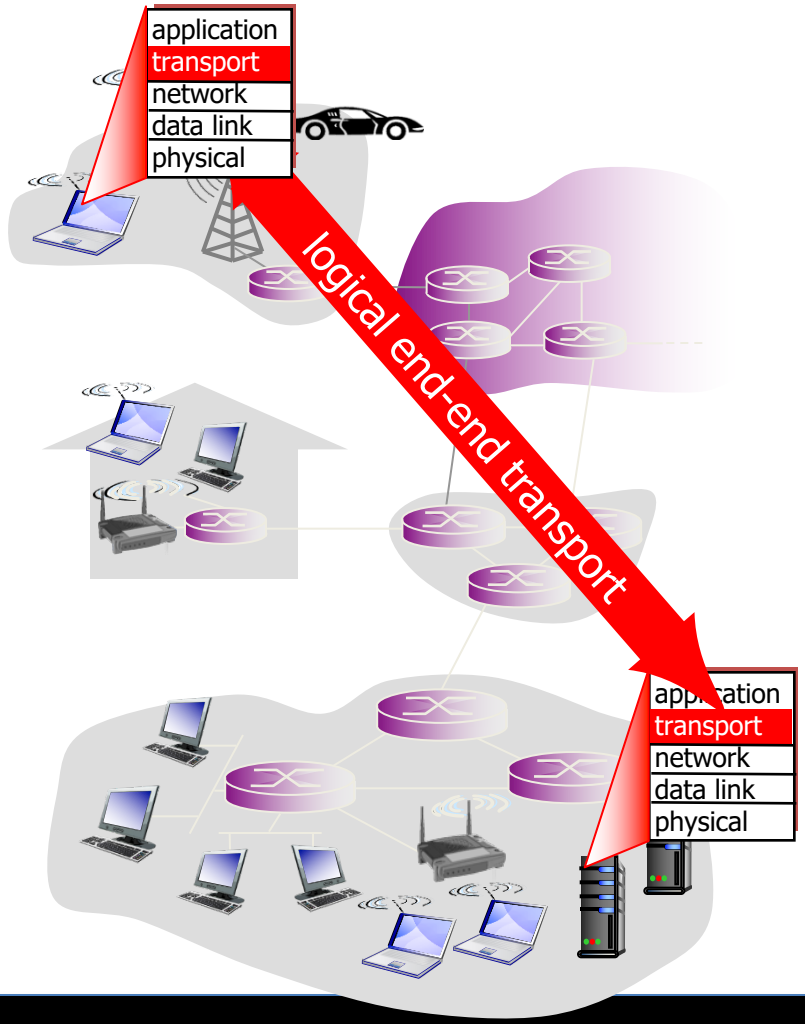
Transport Layer

- Uses IP addressing to create services
  - Rather than simply delivering to a machine, we can run multiple services per machine
- Introduces the concept of ports for “process-to-process” communication
- Protocols in this layer: **TCP, UDP**

- Example: Bob has a service on port 80. Use the internet’s knowledge to deliver my 60B to him and wait for his confirmation

Transport Services and Protocols

- Provide logical communication between app processes running on different hosts
- Transport protocols run in end systems
  - Send side: breaks app messages into segments, passes to network layer
  - Receiver side: reassembles segments into messages, passes to app layer
- More than one transport protocol available to apps
  - Internet: TCP and UDP



Transport vs. Network Layer

- Network layer: logical communication between hosts
- Transport layer: logical communication between processes
  - Relies on, enhances, network layer services

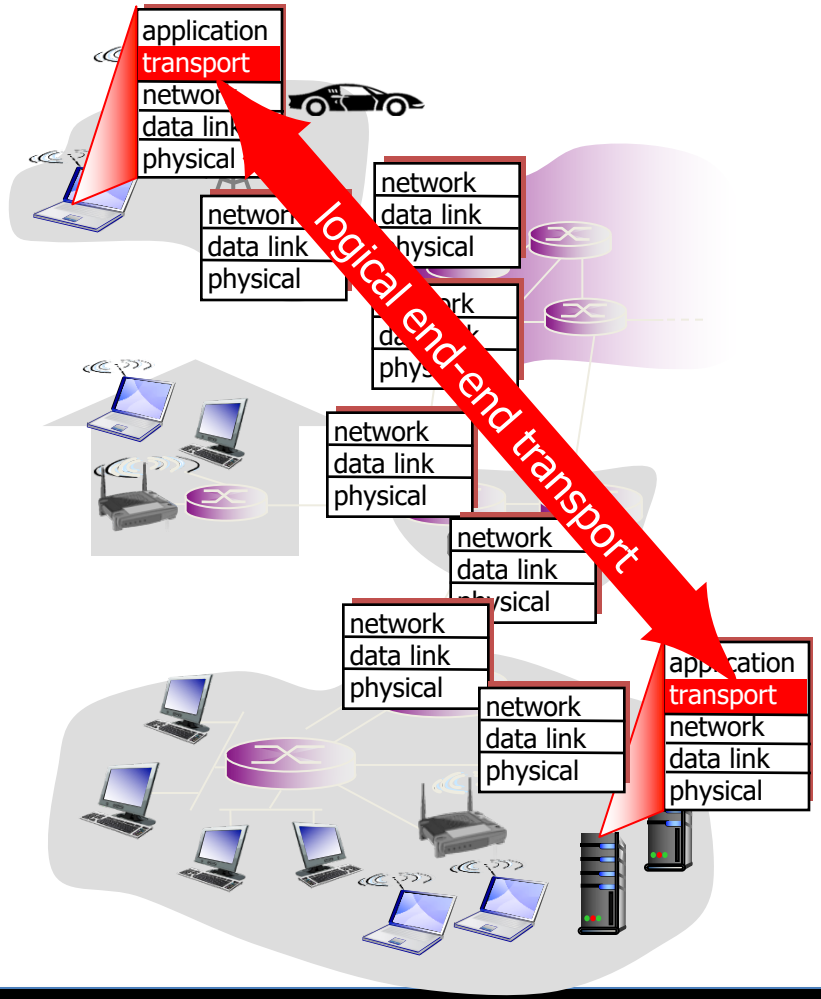
Household Analogy:

12 kids in Ann's house sending letters to 12 kids in Bill's house:

- hosts = houses
- processes = kids
- app messages = letters in envelopes
- transport protocol = Ann and Bill who demux to in-house siblings
- network-layer protocol = postal service

Internet Transport-Layer Protocols

- Reliable, in-order delivery (TCP)
  - Congestion control
  - Flow control
  - Connection setup
- Unreliable, unordered delivery: UDP
  - No-frills extension of “best-effort” IP
- Services not available:
  - Delay guarantees
  - Bandwidth guarantees



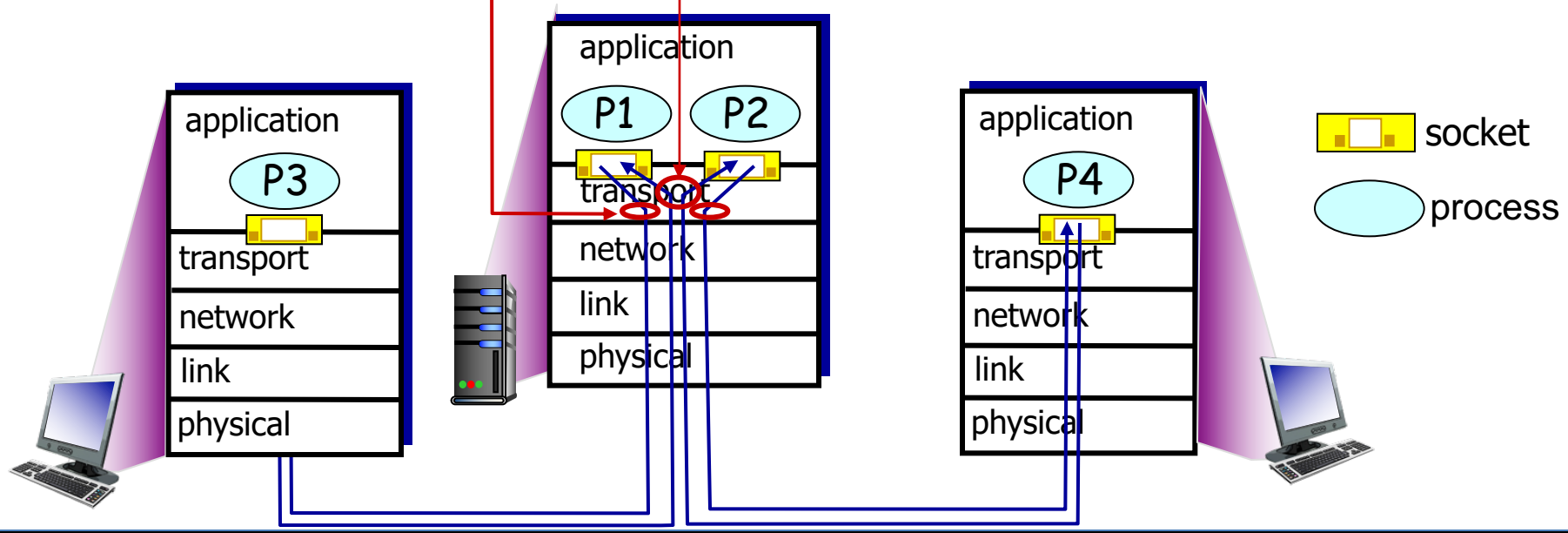
Multiplexing and Demultiplexing

Multiplexing at Sender:

Handle data from multiple sockets, add transport header (later used for demultiplexing)

Demultiplexing at Receiver:

Use header info to deliver received segments to correct socket



Socket

- What is a socket?
  - The point where a local application process attaches to the network
  - An interface between an application and the network
  - An application creates the socket
- The interface defines operations for
  - Creating a socket
  - Attaching a socket to the network
  - Sending and receiving messages through the socket
  - Closing the socket

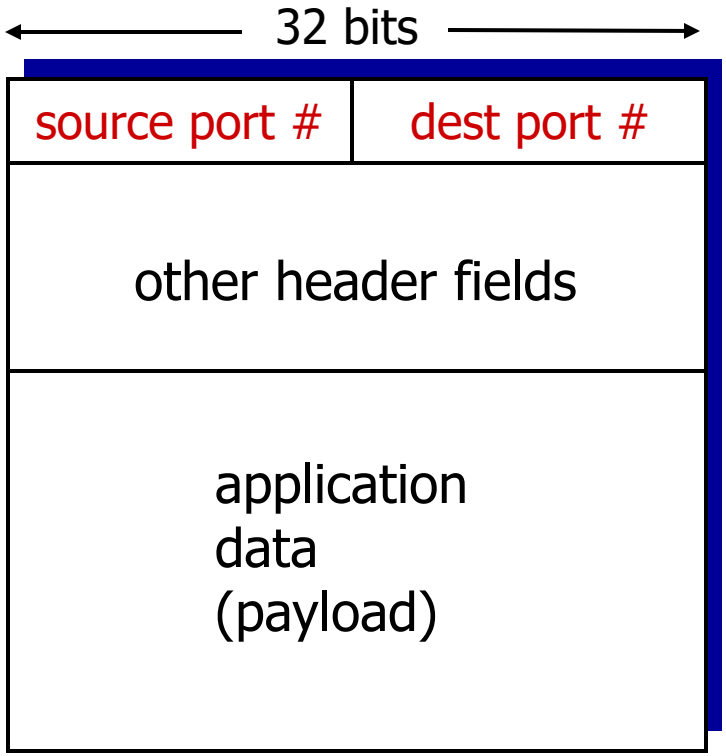
Socket

- Socket Family
  - PF_INET denotes the Internet family
  - PF_UNIX denotes the Unix pipe facility
  - PF_PACKET denotes direct access to the network interface (i.e., it bypasses the TCP/IP protocol stack)

- Socket Type
  - SOCK_STREAM is used to denote a byte stream
  - SOCK_DGRAM is an alternative that denotes a message-oriented service, such as that provided by UDP

Connectionless Demultiplexing

- Host receives IP datagrams
  - Each datagram has source IP address, destination IP address
  - Each datagram carries one transport-layer segment
  - Each segment has source, destination port number
- Host uses IP addresses & port numbers to direct segment to appropriate socket



TCP/UDP segment format

Connection-Oriented Demultiplexing

- TCP socket identified by 4-tuple:
  - Source IP address
  - Source port number
  - Dest IP address
  - Dest port number
- Demultiplexing: receiver uses all four values to direct segment to appropriate socket
- Server host may support many simultaneous TCP sockets:
  - Each socket identified by its own 4-tuple
- Web servers have different sockets for each connecting client
  - Non-persistent HTTP will have different socket for each request

UDP: User Datagram Protocol

- "No frills," "bare bones" Internet transport protocol
- "Best effort" service, UDP segments may be:
  - Lost
  - Delivered out-of-order to app
- Connectionless:
  - No handshaking between UDP sender, receiver
  - Each UDP segment handled independently of others

- UDP use:
  - Streaming multimedia apps (loss tolerant, rate sensitive)
  - DNS
  - SNMP
- Reliable transfer over UDP:
  - Add reliability at application layer
  - Application-specific error recovery!

UDP Checksum

Goal: detect “errors” (e.g., flipped bits) in transmitted segment

Sender:

- Treat segment contents, including header fields, as sequence of 16-bit integers
- Checksum: addition (one’s complement sum) of segment contents
- Sender puts checksum value into UDP checksum field

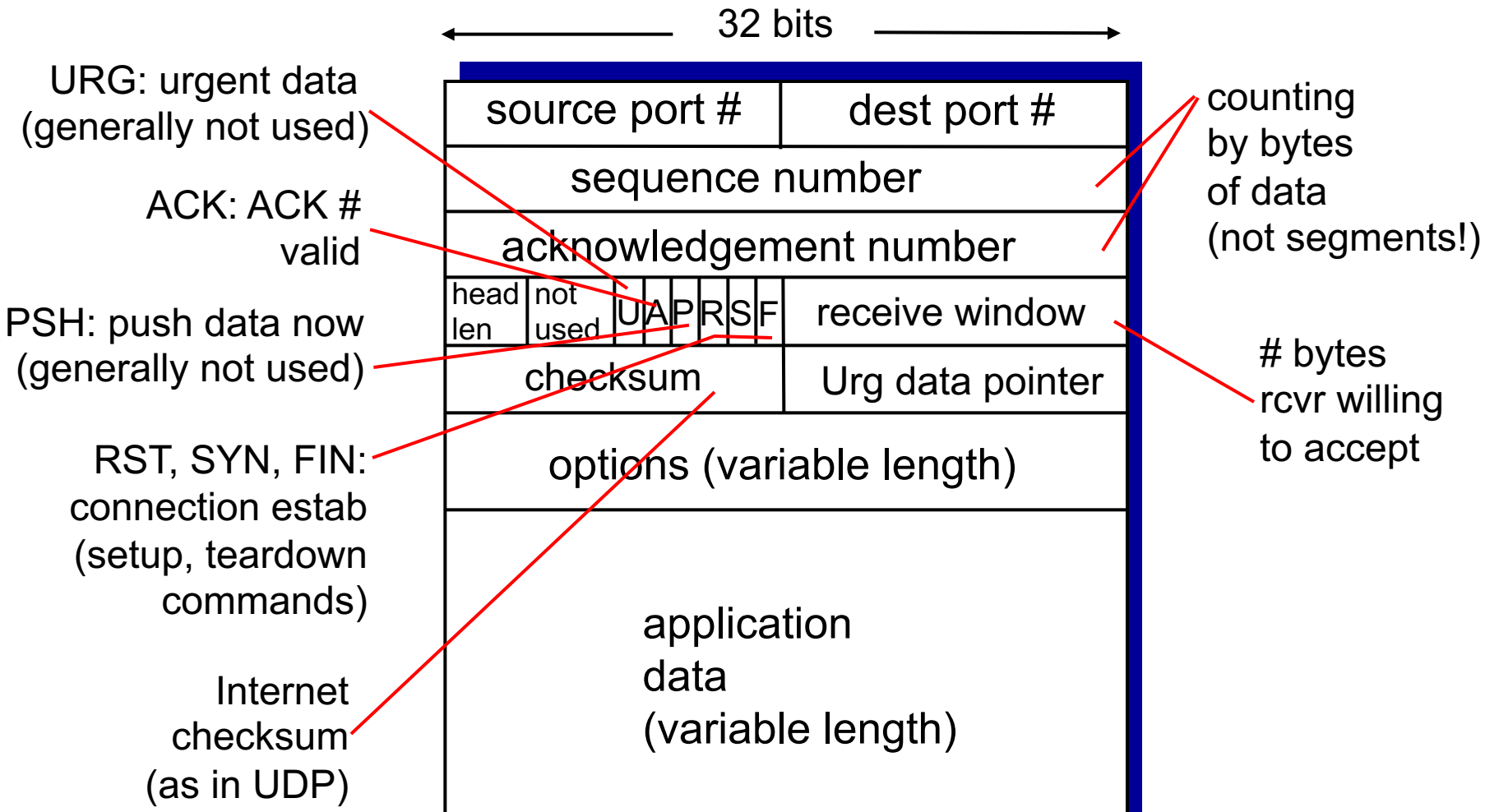
Receiver:

- Compute checksum of received segment
- Check if computed checksum equals checksum field value:
  - NO - error detected
  - YES - no error detected. But maybe errors nonetheless?

TCP: Overview

- Point-to-point:
  - One sender, one receiver
- Reliable, in-order byte steam:
  - No “message boundaries”
- Pipelined:
  - TCP congestion and flow control set window size
- Full duplex data:
  - Bi-directional data flow in same connection
  - MSS: maximum segment size
- Connection-oriented:
  - Handshaking (exchange of control msgs) inits sender, receiver state before data exchange
- Flow controlled:
  - Sender will not overwhelm receiver

TCP Segment Structure



TCP Sequence Numbers, ACKs

Sequence numbers:

- Byte stream “number” of first byte in segment’s data

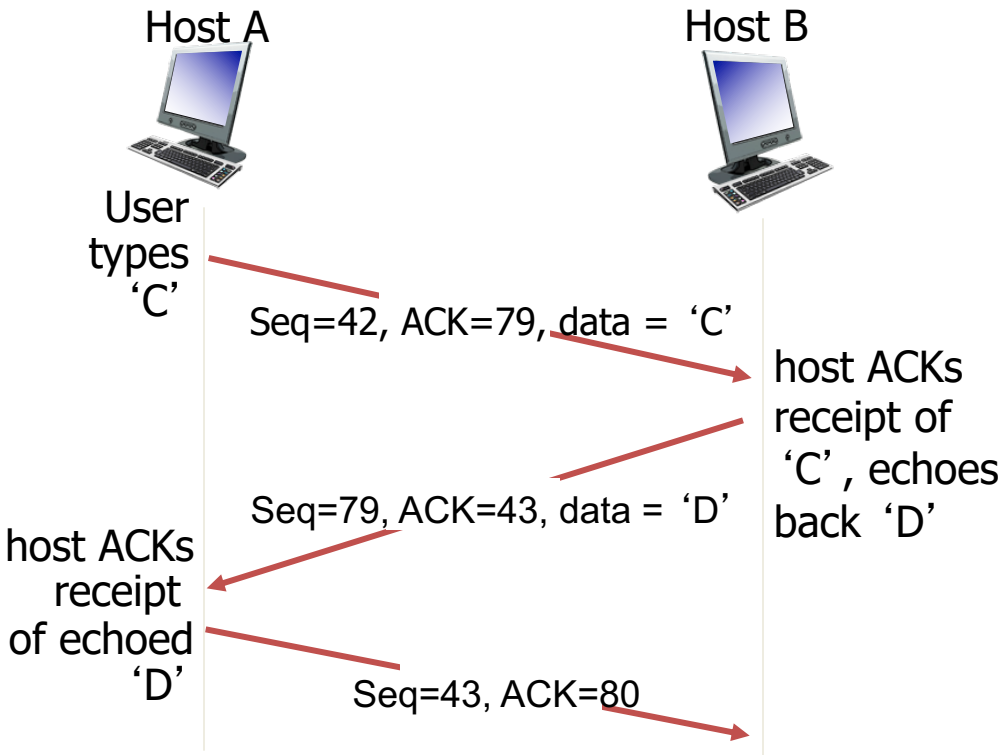
Acknowledgements:

- Seq # of next byte expected from other side
- Cumulative ACK

Q: How receiver handles out-of-order segments

A: TCP spec doesn’t say, - up to implementor

TCP Sequence Numbers, ACKs



simple telnet scenario

TCP Round-Trip Time (RTT), Timeout

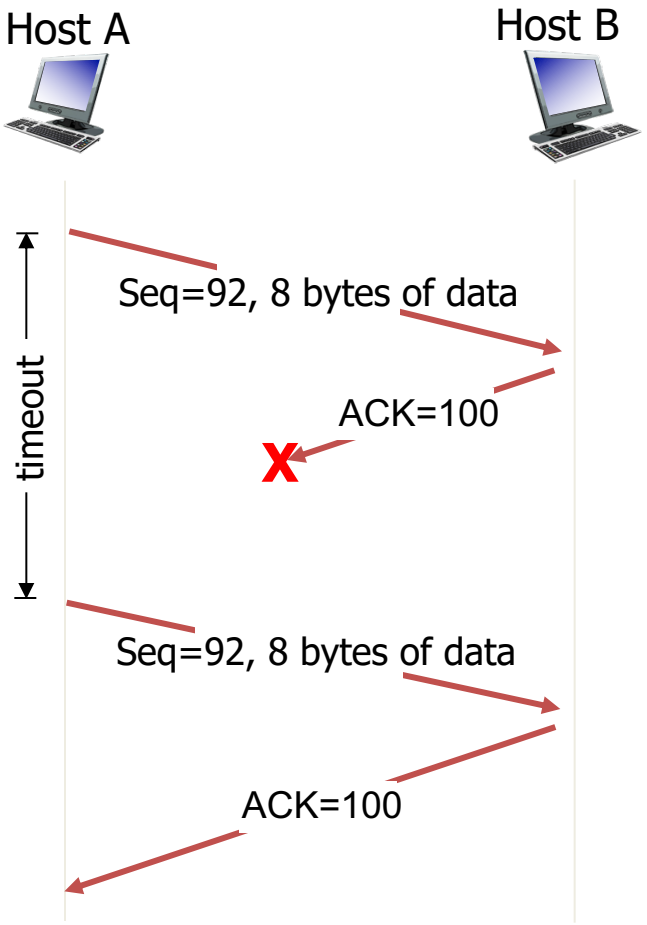
Q: How to set TCP timeout value?

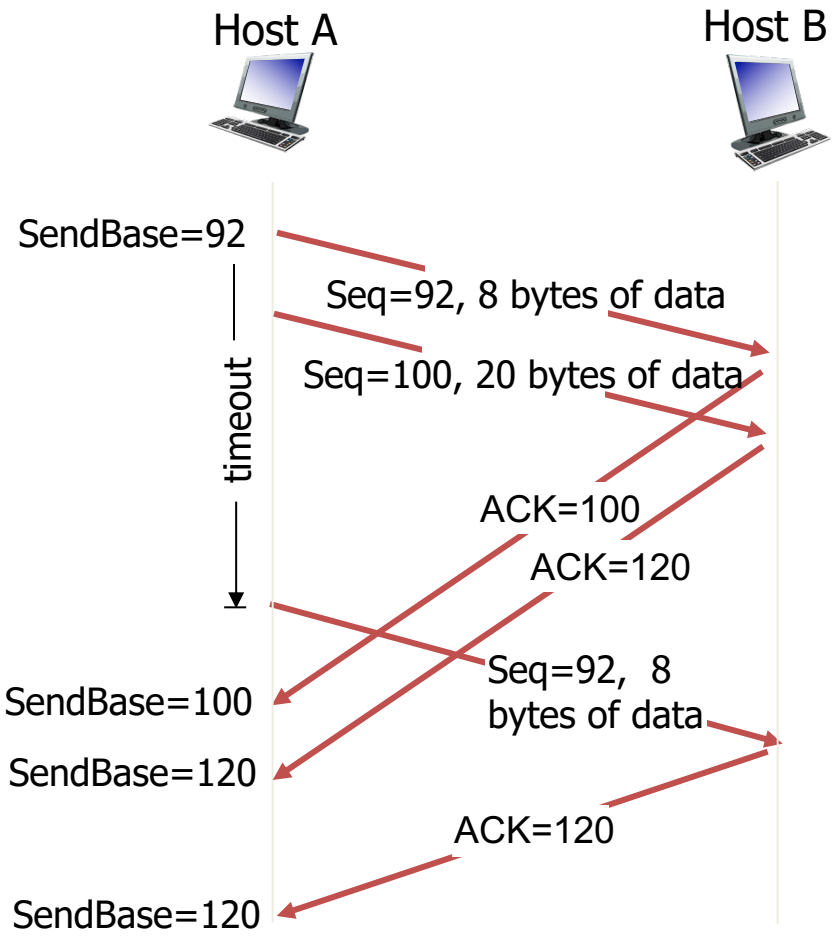
- Longer than RTT
  - But RTT varies
- Too short: premature timeout, unnecessary retransmissions
- Too long: slow reaction to segment loss

Q: How to estimate RTT?

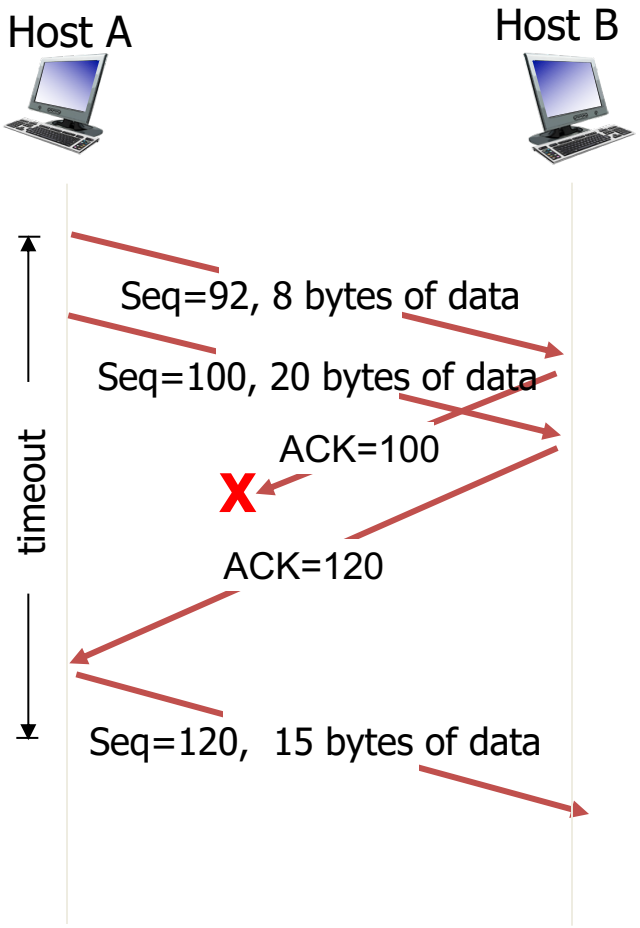
- SampleRTT: measured time from segment transmission until ACK receipt
  - Ignore retransmissions
- SampleRTT will vary, want estimated RTT “smoother”
  - Average several recent measurements, not just current SampleRTT

TCP: Retransmission Scenarios



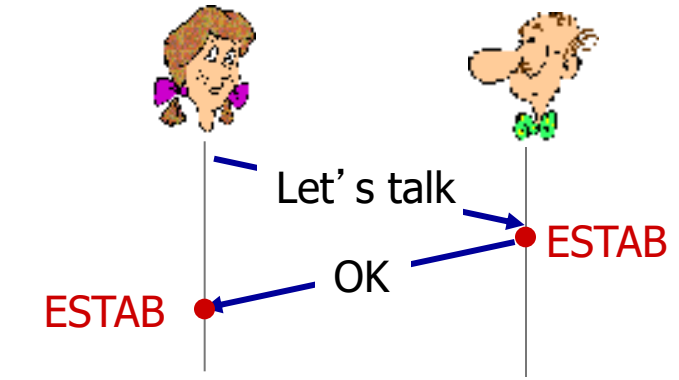


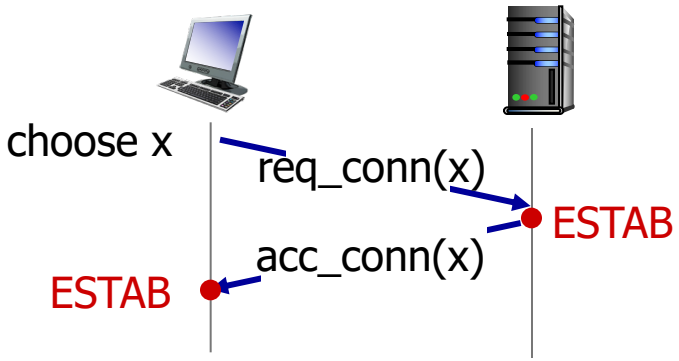
TCP: Retransmission Scenarios



Agreeing to Establish a Connection

2-way handshake:

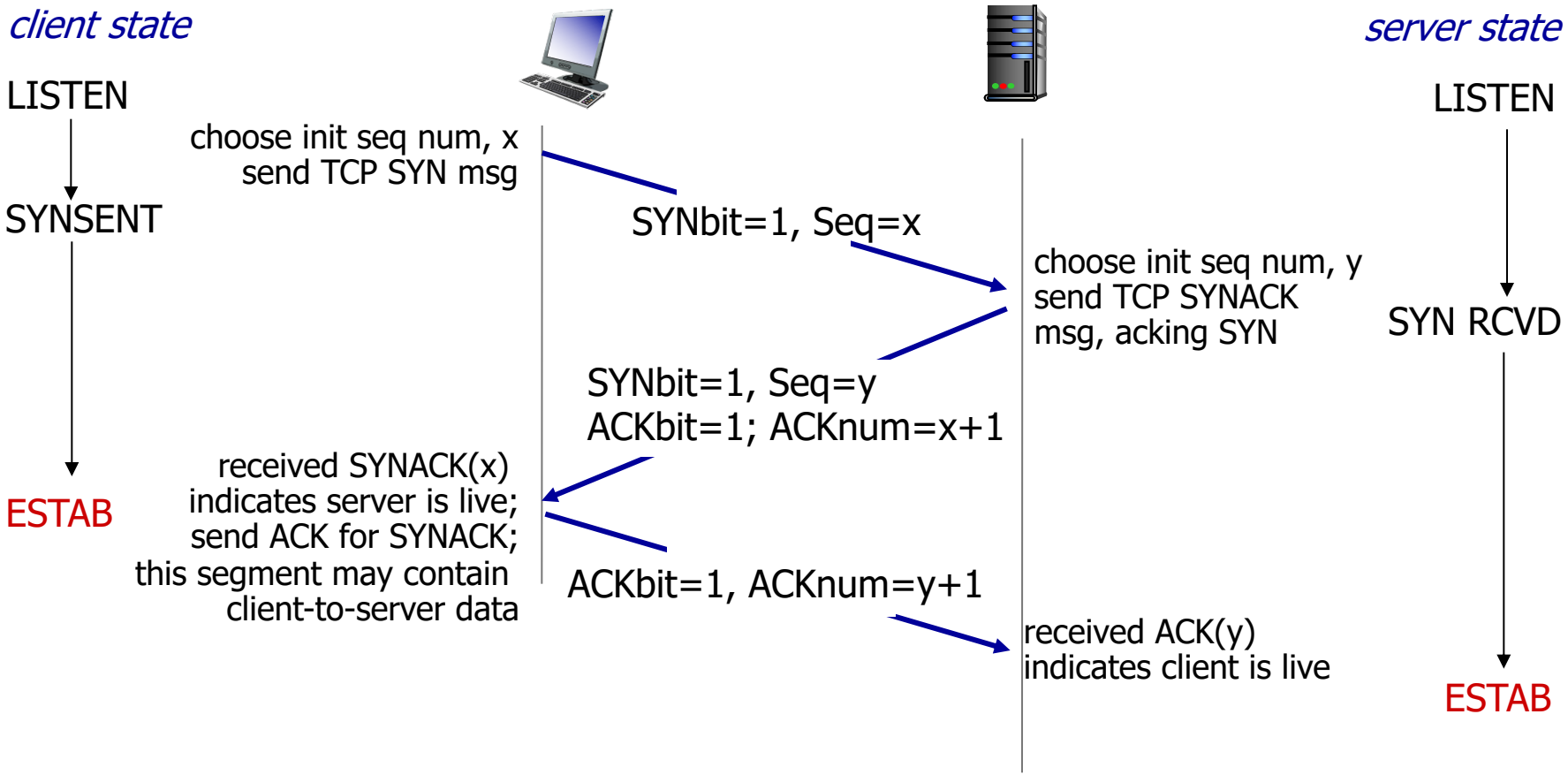




Q: Will 2-way handshake always work in network?

- Variable delays
- Retransmitted messages (e.g. req_conn(x)) due to message loss
- Message reordering
- Can't "see" other side

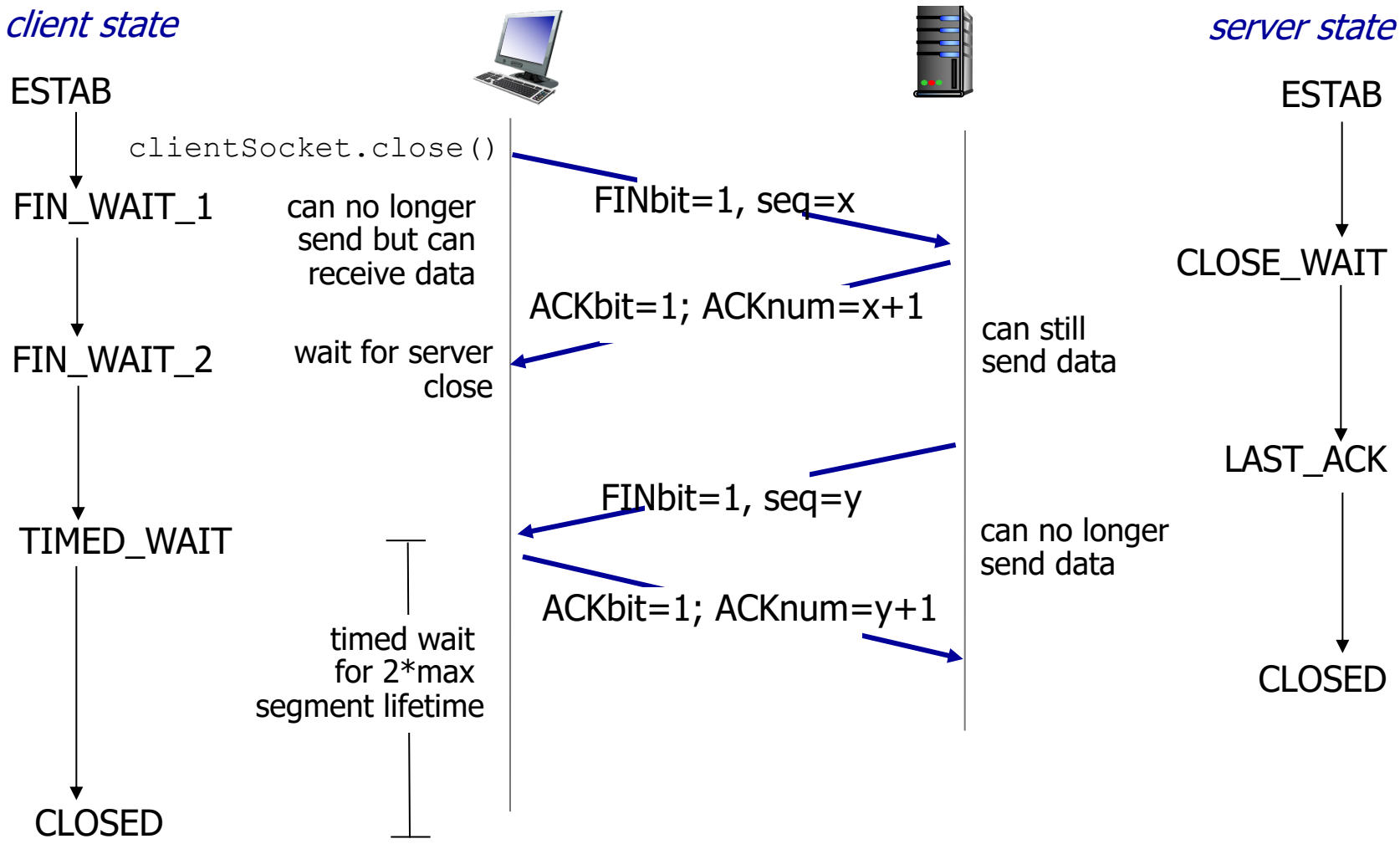
TCP 3-Way Handshake



TCP: Closing a Connection

- Client, server each close their side of connection
  - Send TCP segment with FIN bit = 1
- Respond to received FIN with ACK
  - On receiving FIN, ACK can be combined with own FIN
- Simultaneous FIN exchanges can be handled

TCP: Closing a Connection



Congestion Control

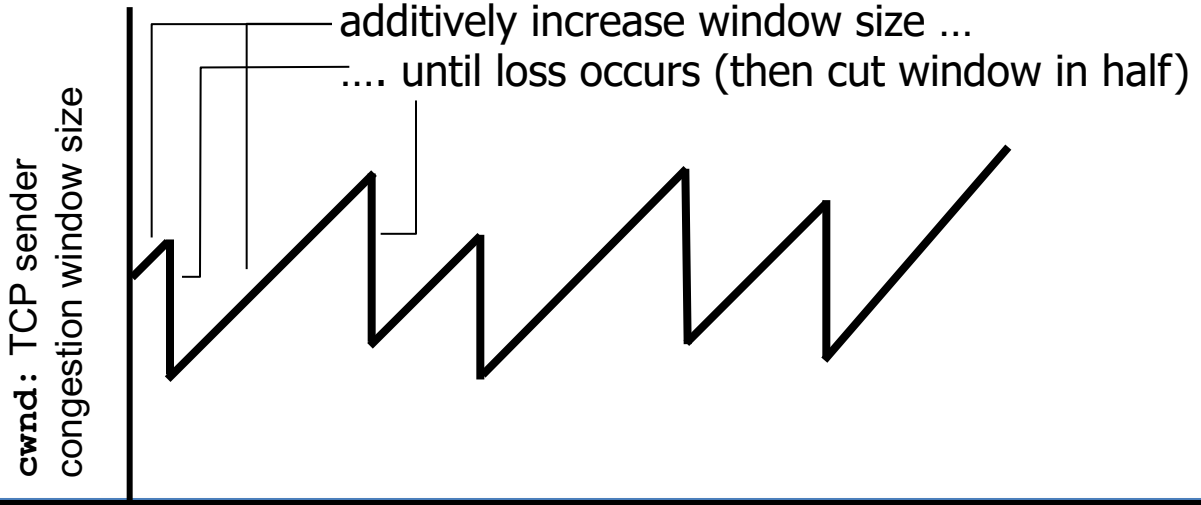
Congestion:

- Informally: “too many sources sending too much data too fast for network to handle”
- Different from flow control!
- Manifestations:
  - Lost packets (buffer overflow at routers)
  - Long delays (queueing in router buffers)

TCP Congestion Control: Additive Increase Multiplicative Decrease

- Approach: sender increases transmission rate (window size), probing for usable bandwidth, until loss occurs
- Additive increase: increase current window by 1 MSS every RTT until loss detected
- Multiplicative decrease: cut current window in half after loss

AIMD saw tooth behavior: probing for bandwidth



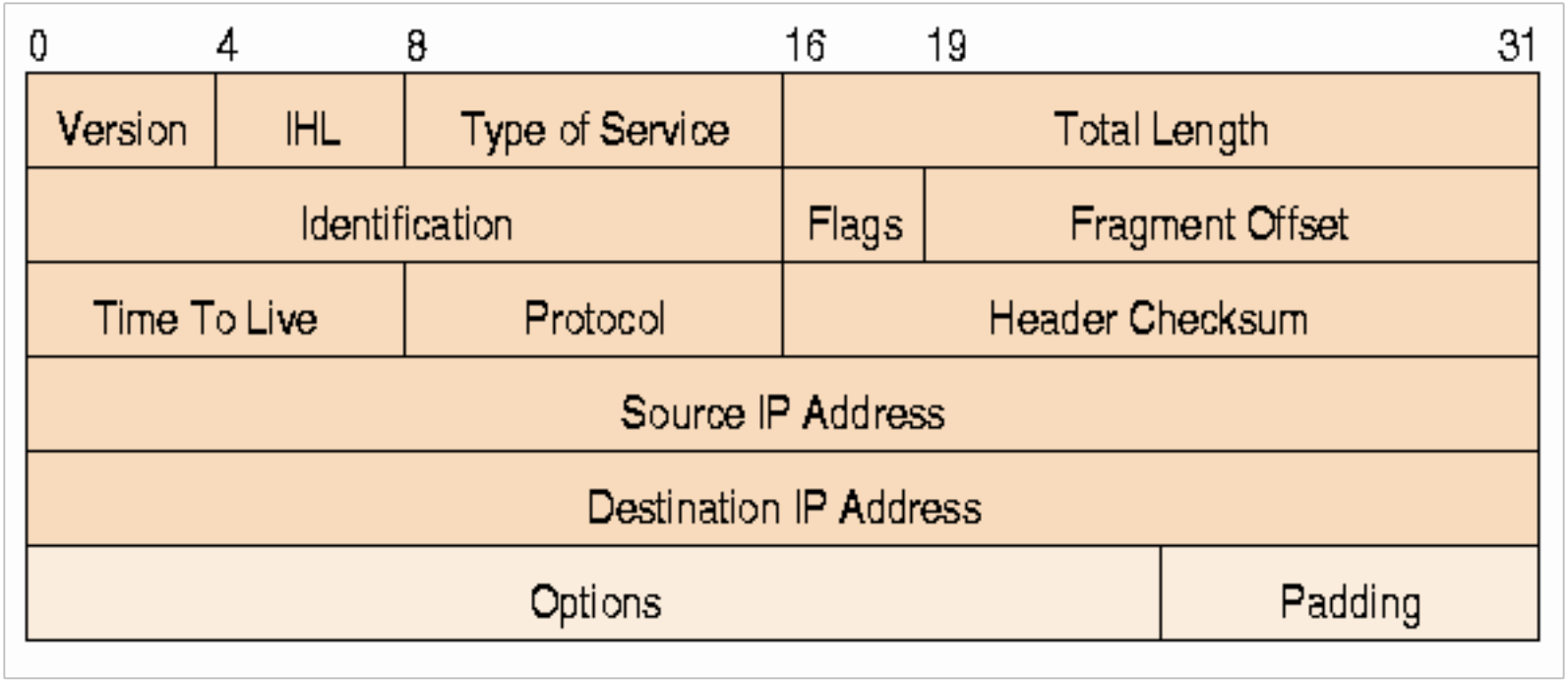
TCP Reliable Data Transfer

- TCP creates reliable data transfer service on top of IP's unreliable service
  - Pipelined segments
  - Cumulative acks
  - Single retransmission timer
- Retransmissions triggered by:
  - Timeout events
  - Duplicate acks
- Regulates data transfer rates based on feedback from network to avoid excessive congestion

Last Word: UDP Superpower: X-cast

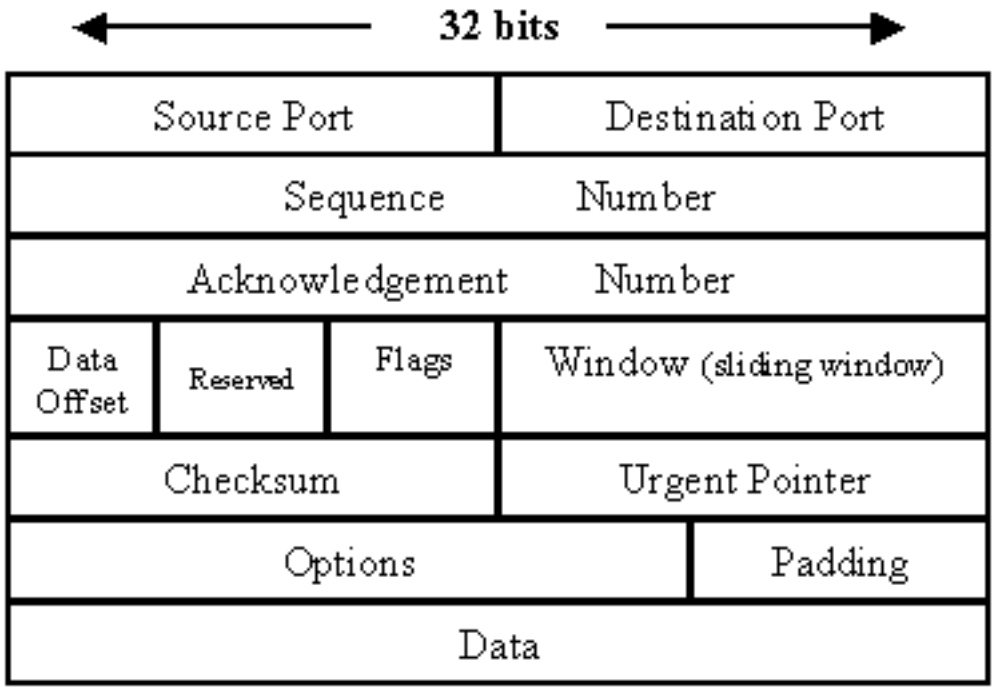
- Unicast: a one-to-one communication
  - The only form supported by TCP
- Broadcast: Send some message to everybody
  - Example: ARP asks everybody for a physical address
- Multicast: Send some message to everybody who has subscribed to your multicast address
  - Example: Deliver live video to multiple hosts
- Anycast: Talk to any one of *n* servers
  - Useful in context of a CDN
  - TCP okay for shorter connections

Previous Packet Structure



*Source: https://freesoft.org

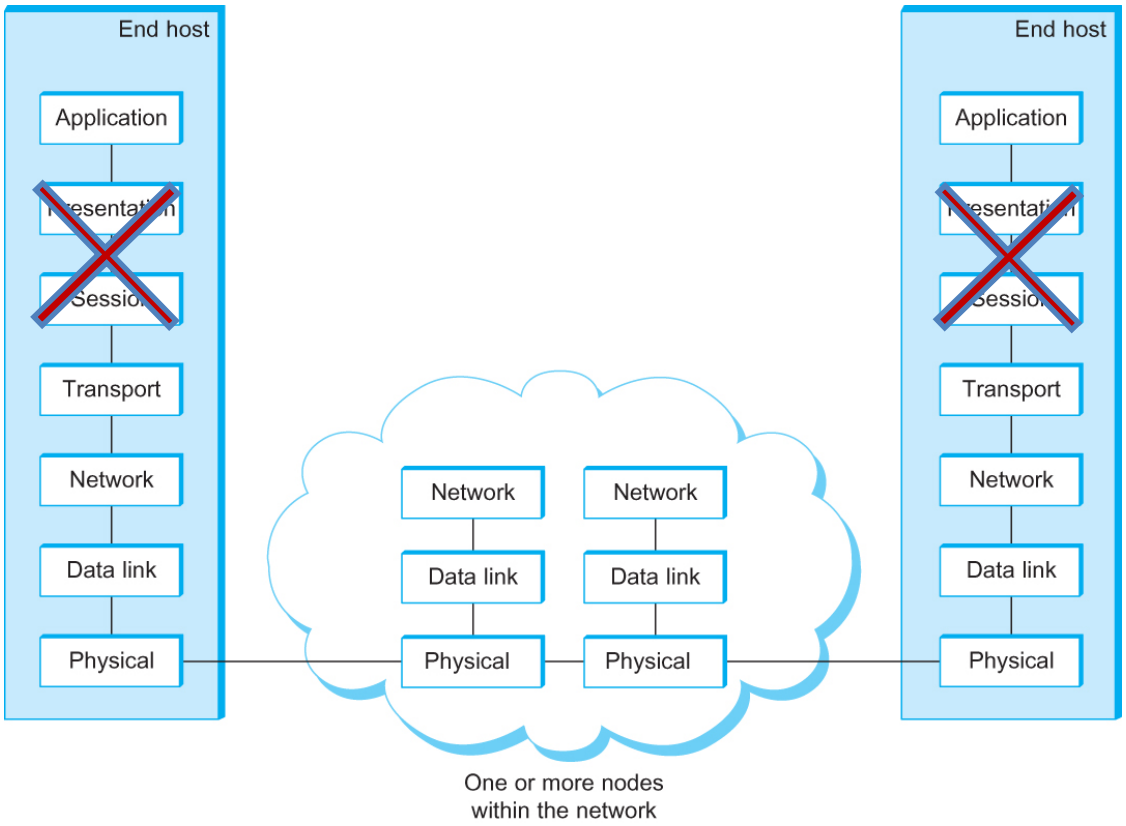
Current Packet Structure



*Source: https://techrepublic.com

Layer 5: Application Layer

*Slides adapted from Computer Networking: A top-Down Approach



The OSI 7-layer Model

OSI – Open Systems Interconnection

Application Layer

- Applications that use TCP/UDP to communicate
  - Anything that uses these is in the application layer
- Protocols in this layer: **DNS, HTTP, FTP, SMTP, POP, all their secure versions, and many, many more**

- Example: Bob is running an HTTP server on port 80. Here is my (hopefully encrypted) cookie and headers: I'd like to send a GET request to retrieve the homepage

Protocol Hourglass

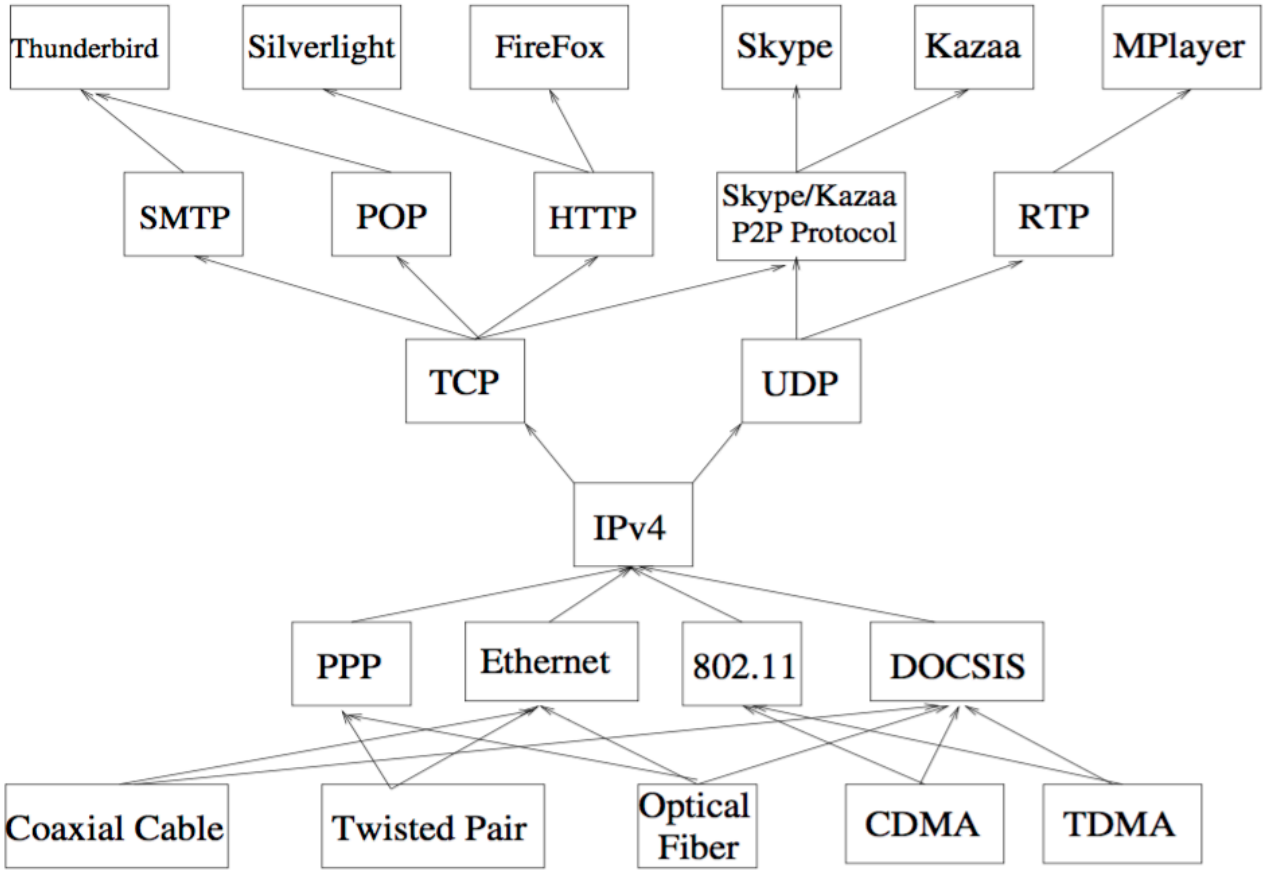


Figure 1: An (incomplete) illustration of the hourglass Internet architecture.

Modern Protocols

Modern Protocols

- “Modern” as in “modernly-used”, not “modernly created”
- Help us examine some of what’s used to drive the internet today
- Gives us an idea of where common exploits lie

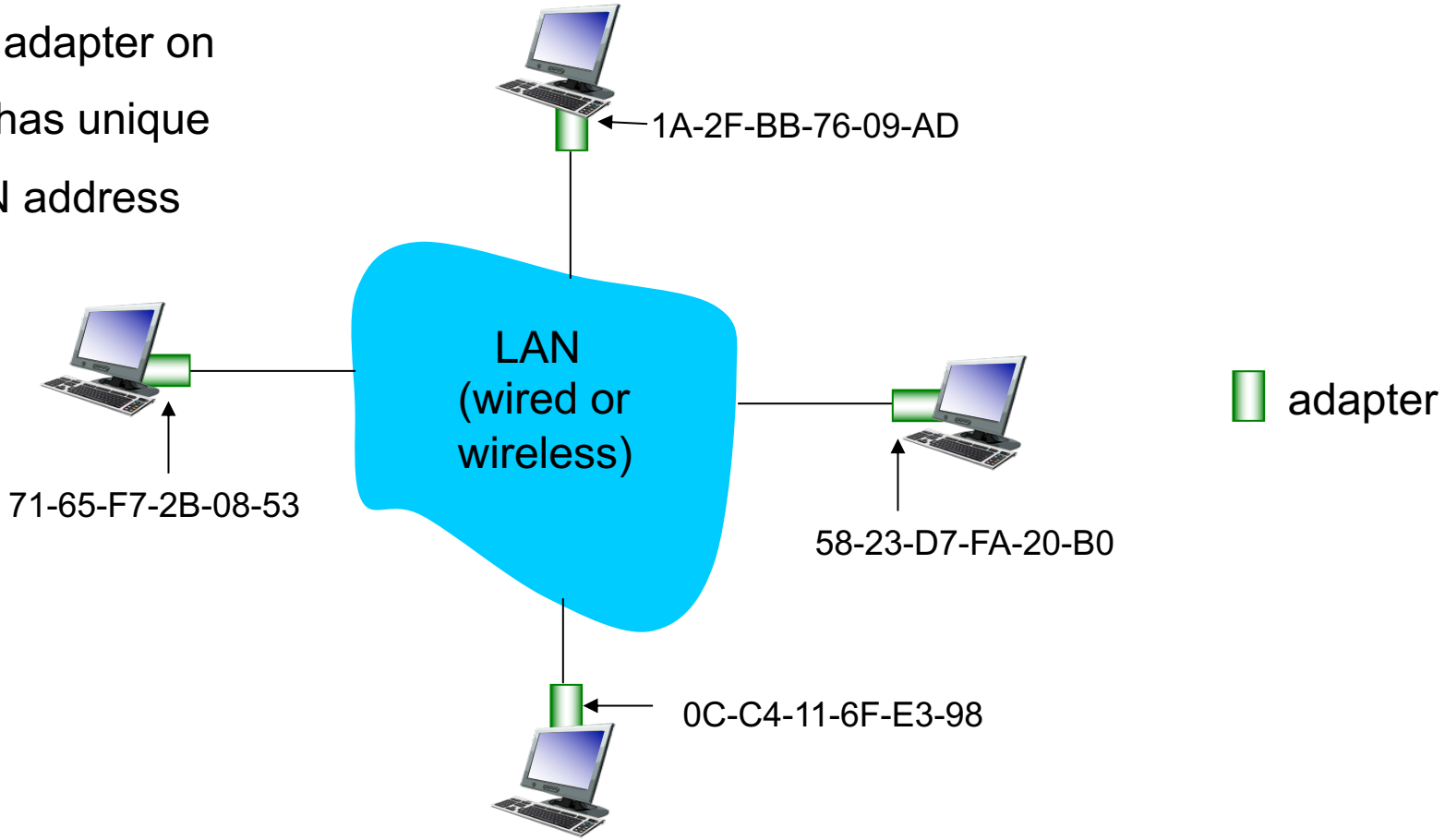
Address Resolution Protocol (ARP)

Modern Protocols

- 32-bit IP address:
  - Network-layer address for interface
  - Used for layer 3 (network layer) forwarding
- 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
  - E.g.: 1A-2F-BB-76-09-AD

LAN Addresses And ARP

Each adapter on LAN has unique LAN address

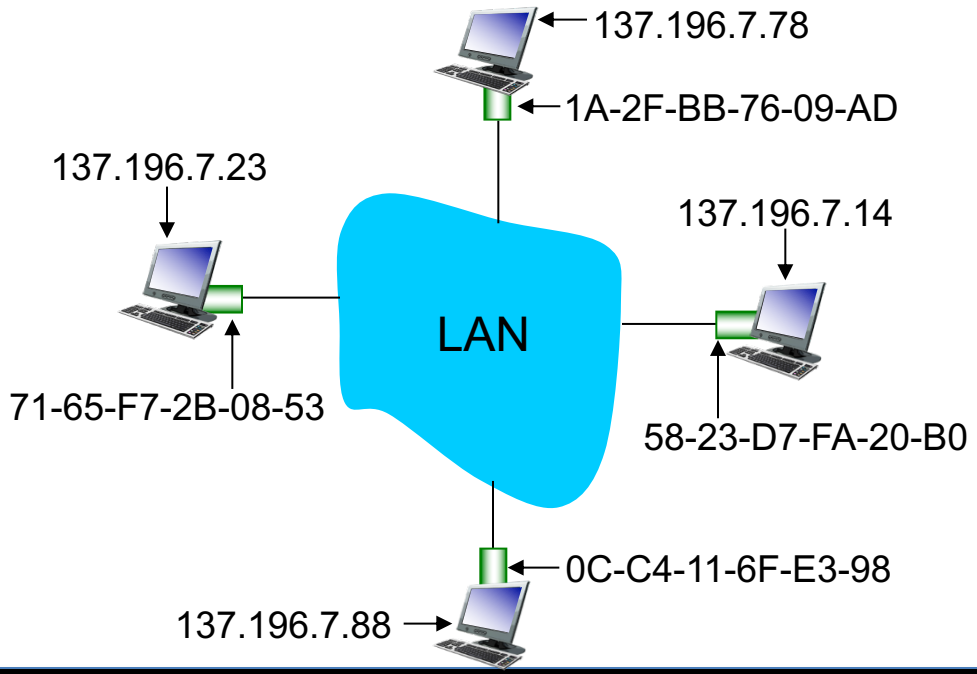


LAN Addresses, Continued

- MAC address allocation administered by IEEE
- Manufacturer buys portion of MAC address space (to assure uniqueness)
- Analogy:
  - MAC address: like Social Security Number
  - IP address: like postal address
- MAC home address provides portability
  - Can move network card from one LAN to another
- IP hierarchical address not portable
  - Address depends on IP subnet to which node is attached

ARP: Address Resolution Protocol

Question: How to determine interface's MAC address, knowing its IP address?



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

- IP/MAC address mappings for some LAN nodes:
  - < IP address; MAC address; TTL>
- TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

ARP Protocol: Same LAN

- A wants to send datagram to B
  - B's MAC address not in A's ARP table.
- A broadcasts ARP query packet, containing B's IP address
  - Destination MAC address = FF-FF-FF-FF-FF-FF (Broadcast)
  - All nodes on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) MAC address
  - Frame sent to A's MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - Soft state: information that times out (goes away) unless refreshed
- ARP is "plug-and-play":
  - Nodes create their ARP tables without intervention from net administrator

ARP Poisoning (ARP Spoofing)

- Wait for an ARP query, then lie, saying that the victim's IP address is at your MAC address
  - Hijack a legitimate session
  - Get all traffic to come to you
- Once updated, the default gateway will broadcast the updated address to all other devices on the network
- Very easy to carry out, even today
  - Watch out for your roommates!

Domain Name Service (DNS)

DNS: Domain Name System

- People: many identifiers:
  - SSN, name, passport #
- Internet hosts, routers:
  - IP address (32 bit) - used for addressing datagrams
  - “Name”, e.g., www.yahoo.com - used by humans

Q: How to map between IP address and name, and vice versa?

- /etc/hosts

Domain Name System:

- Distributed database implemented in hierarchy of many name servers
- Application-layer protocol: hosts, name servers communicate to resolve names (address/name translation)
  - Note: core Internet function, implemented as application-layer protocol
  - complexity at network’s “edge”

DNS: Services And Structure

DNS services

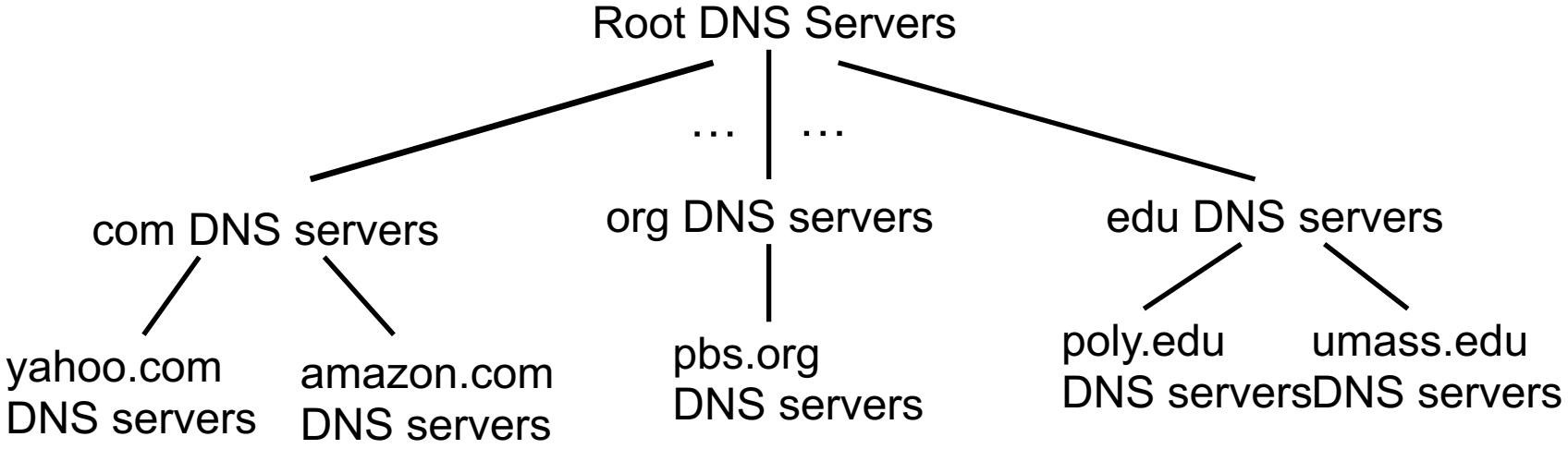
- Hostname to IP address translation
- Host aliasing
  - Canonical, alias names
- Mail server aliasing
- Load distribution
  - Replicated Web servers: many IP addresses correspond to one name

Q: Why not centralize DNS?

- Single point of failure
- Traffic volume
- Distant centralized database
- Maintenance

A: Doesn't scale!

DNS: A Distributed, Hierarchical Database

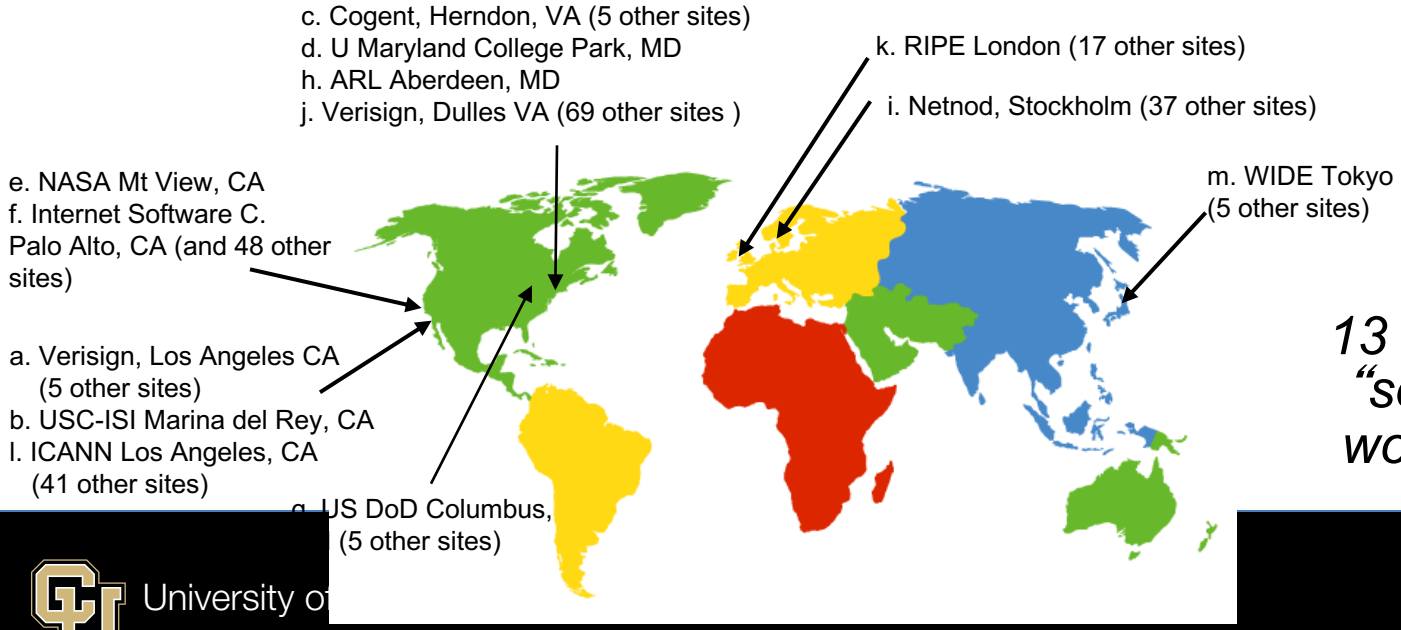


Client wants IP for www.amazon.com; 1st approx:

- Client queries root server to find ".com" DNS server
- Client queries ".com" DNS server to get amazon.com DNS server
- Client queries amazon.com DNS server to get IP address for www.amazon.com

DNS: Root Name Servers

- Contacted by local name server that can not resolve name
- Root name server:
  - Contacts authoritative name server if name mapping not known
  - Gets mapping
  - Returns mapping to local name server



13 root name "servers" worldwide

TLD Authoritative Servers

Top-level domain (TLD) servers:

- Responsible for com, org, net, edu, aero, jobs, museums, and all top-level country domains, e.g.: uk, fr, ca, jp
- Network Solutions maintains servers for .com TLD
- Educause for .edu TLD

Authoritative DNS servers:

- Organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- Can be maintained by organization or service provider

Local DNS Name Server

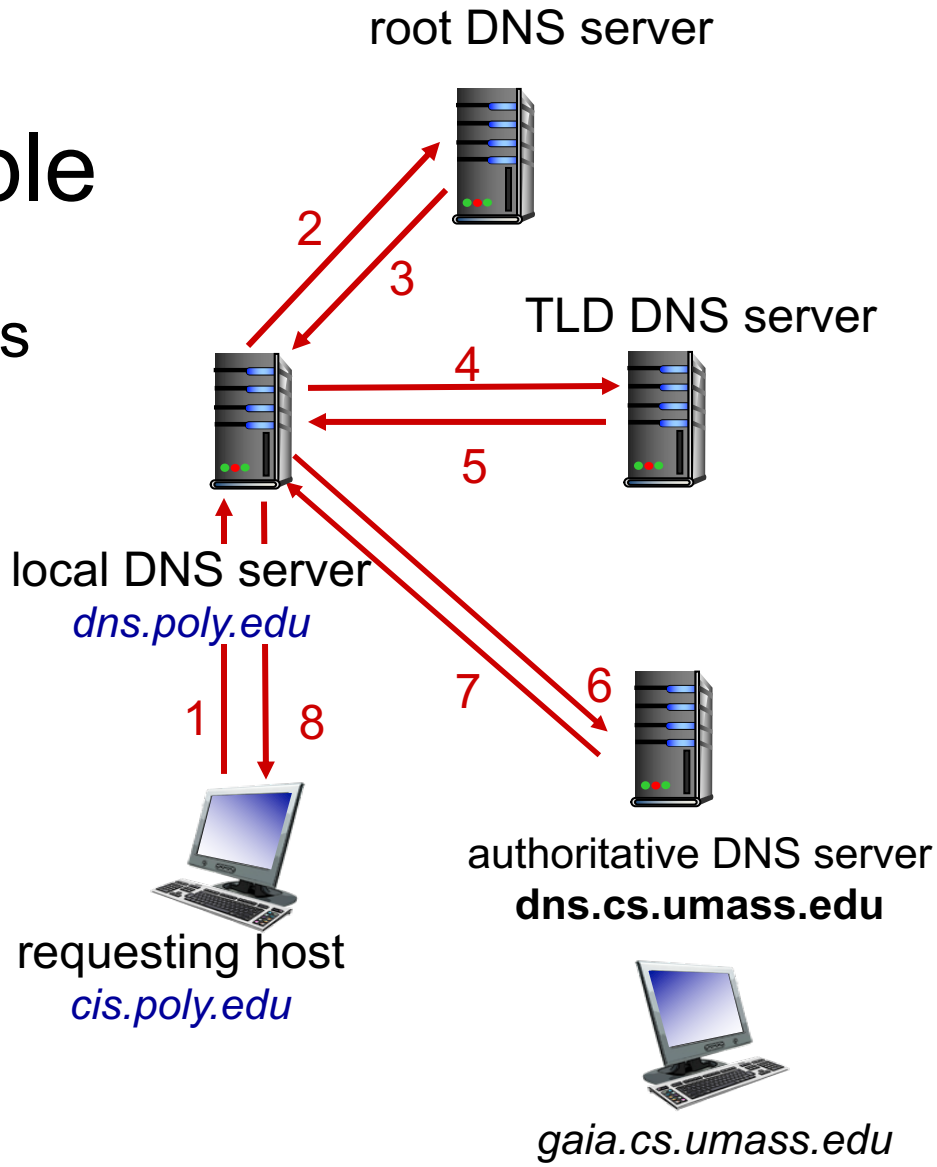
- Does not strictly belong to hierarchy
- Each ISP (residential ISP, company, university) has one
  - Also called “default name server”
- When host makes DNS query, query is sent to its local DNS server
  - Has local cache of recent name-to-address translation pairs (but may be out of date!)
  - Acts as proxy, forwards query into hierarchy

DNS Name Resolution Example

- Host at cis.poly.edu wants IP address for gaia.cs.umass.edu

Iterated query:

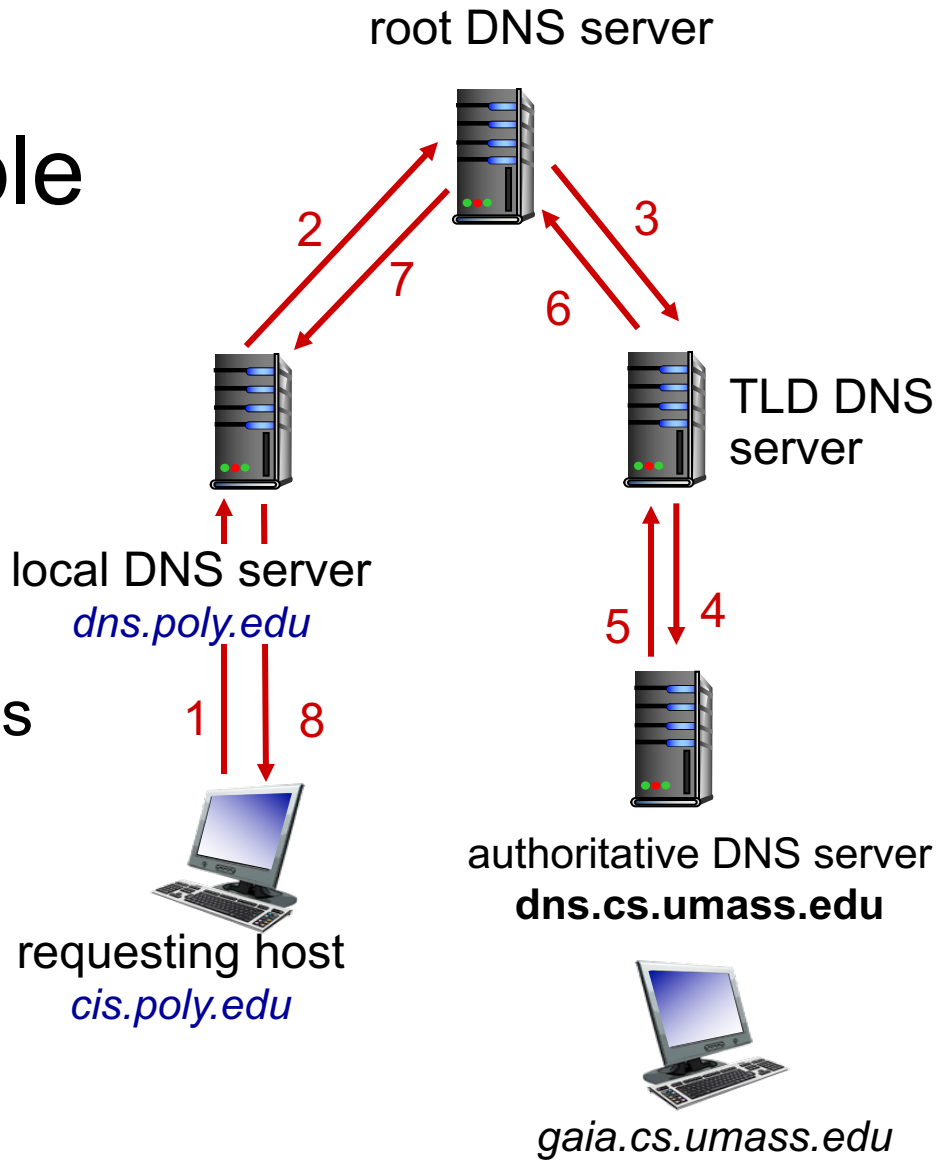
- Contacted server replies with name of server to contact
- "I don't know this name, but ask this server"



DNS Name Resolution Example

Recursive query:

- Puts burden of name resolution on contacted name server
- Heavy load at upper levels of hierarchy?



DNS: Caching, Updating Records

- Once (any) name server learns mapping, it caches mapping
  - Cache entries timeout (disappear) after some time (TTL)
  - TLD servers typically cached in local name servers
- Thus root name servers not often visited
  - Cached entries may be out-of-date (best effort name-to-address translation!)
- If name host changes IP address, may not be known Internet-wide until all TTLs expire
  - Update/notify mechanisms proposed IETF standard

DNS Records

DNS: Distributed db storing resource records (RR)

RR format: `(name, value, type, ttl)`

type=A

- **`name`** is hostname
- **`value`** is IP address

type=NS

- **`name`** is domain (e.g., foo.com)
- **`value`** is hostname of authoritative name server for this domain

type=CNAME

- **`name`** is alias name for some "canonical" (the real) name
- **`www.ibm.com`** is really **`servereast.backup2.ibm.com`**
- **`value`** is canonical name

type=MX

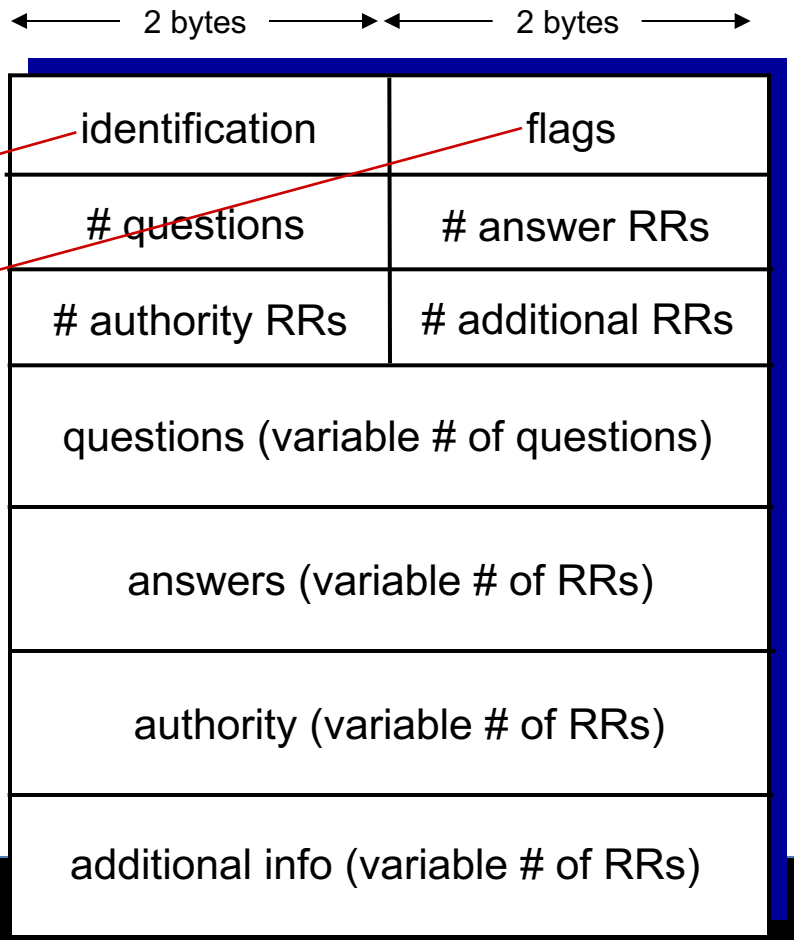
- **`value`** is name of mailserver associated with **`name`**

DNS Protocol And Messages

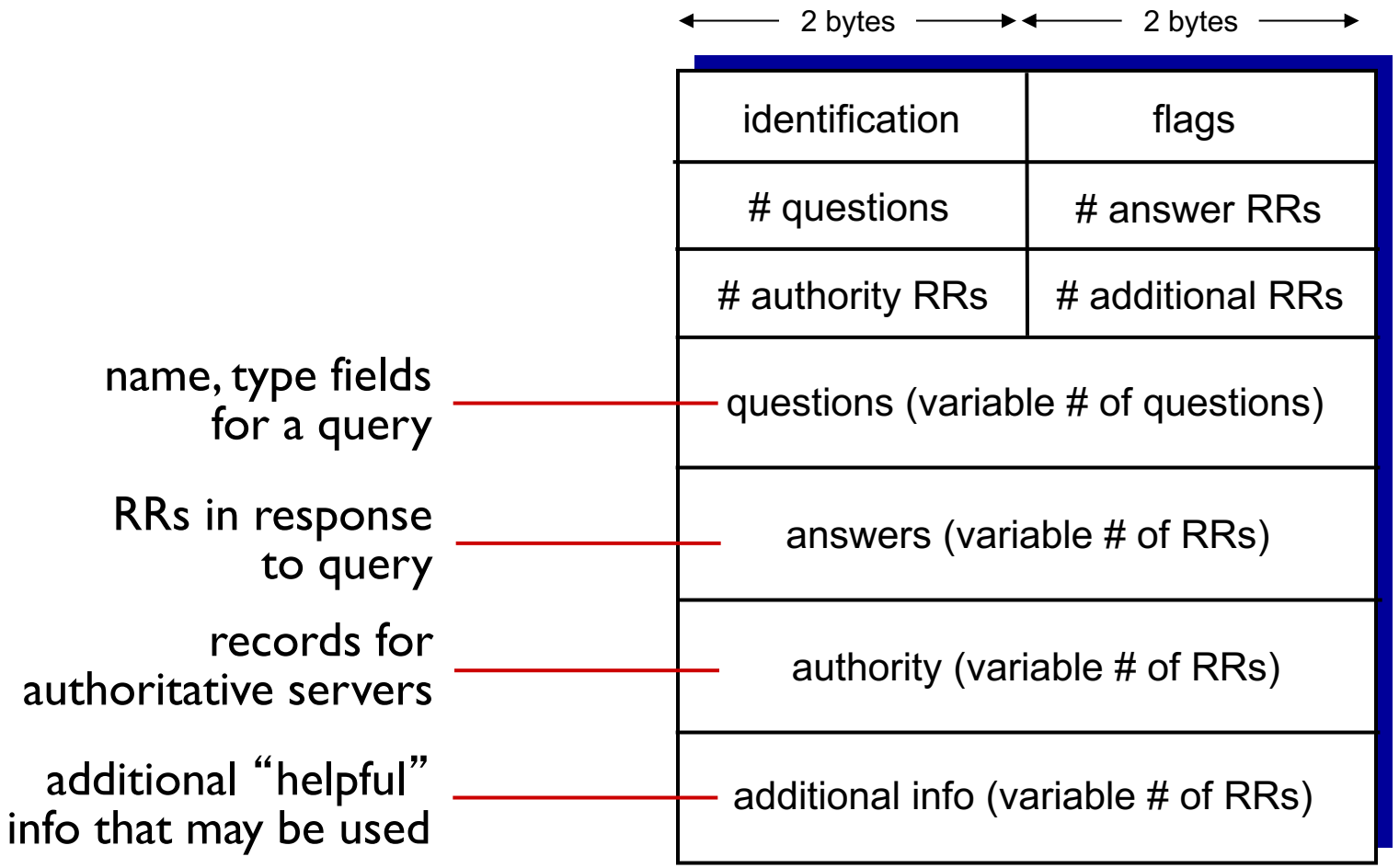
Query and reply messages, both with same message format

msg header

- Identification: 16 bit # for query, reply to query uses same #
- Flags:
  - Query or reply
  - Recursion desired
  - Recursion available
  - Reply is authoritative



DNS Protocol And Messages



Inserting Records Into DNS

- Example: new startup “Network Utopia”
- Register name networkuptopia.com at DNS registrar (e.g., Network Solutions)
  - Provide names, IP addresses of authoritative name server (primary and secondary)
  - Registrar inserts two RRs into .com TLD server: (networkutopia.com, dns1.networkutopia.com, NS)

    (dns1.networkutopia.com, 212.212.212.1, A)
- Create authoritative server type A record for www.networkuptopia.com; type MX record for networkutopia.com

Attacking DNS

DDoS attacks

- Bombard root servers with traffic
  - Not successful to date
  - Traffic Filtering
  - Local DNS servers cache IPs of TLD servers, allowing root server bypass
- Bombard TLD servers
  - Potentially more dangerous

Redirect attacks

- Man-in-middle
  - Intercept queries
- DNS poisoning
  - Send bogus relies to DNS server, which caches

Exploit DNS for DDoS

- Send queries with spoofed source address: target IP
- Requires amplification

Secure DNS Functions

- DNSSec was the best we had
  - Provided integrity, but not privacy!
  - Very complex to implement
- DNS over SSL was a promising candidate
  - Uses own port – obvious you're using encrypted SSL
- Google announced DOH (DNS over HTTPS)
  - Concerns about Google monitoring your traffic (they could before, too!)
  - Seems to be the direction we're headed
- Remember: In some countries you can be killed/face serious rejection for your internet traffic!

Mail Protocols

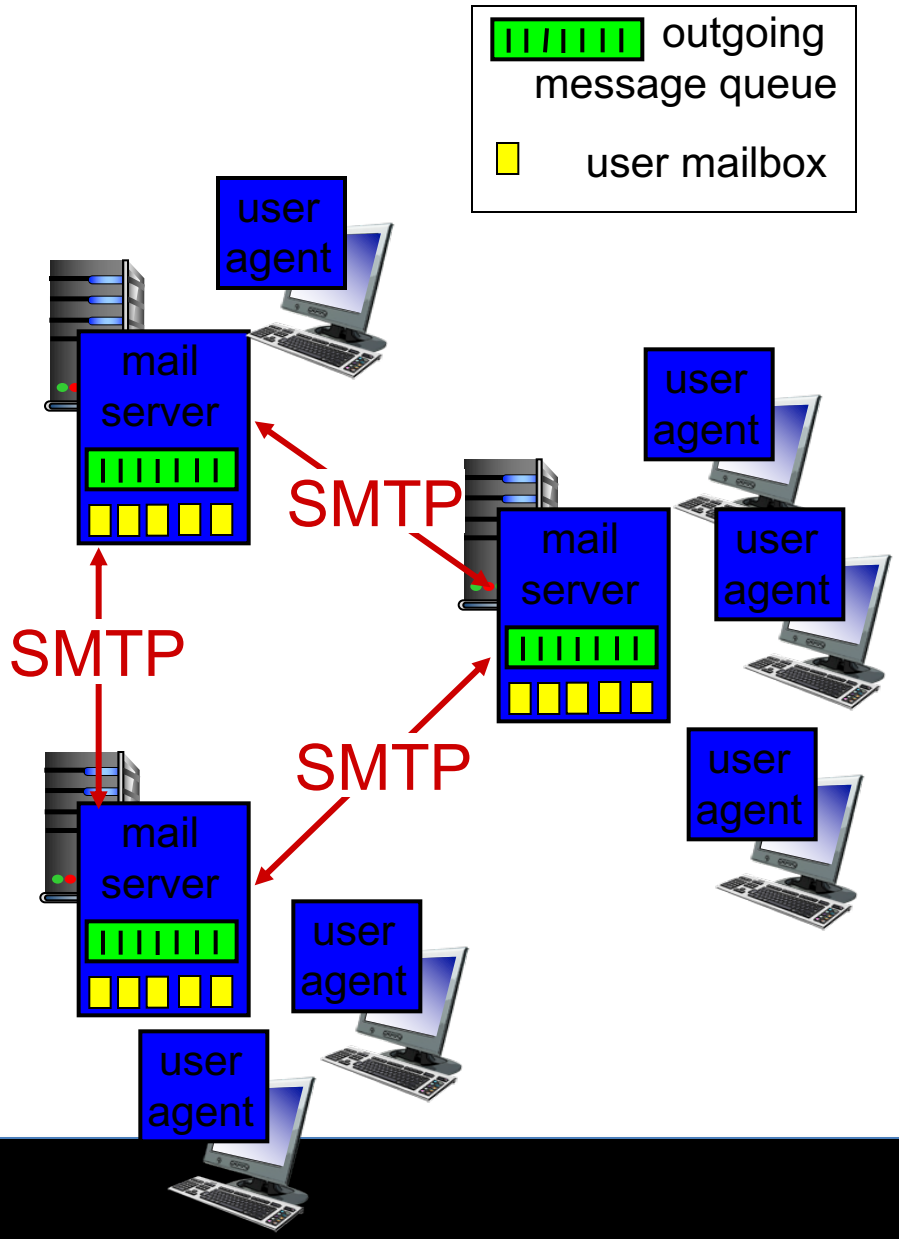
Electronic Mail

Three major components:

- User agents
- Mail servers
- Simple mail transfer protocol: SMTP

User Agent

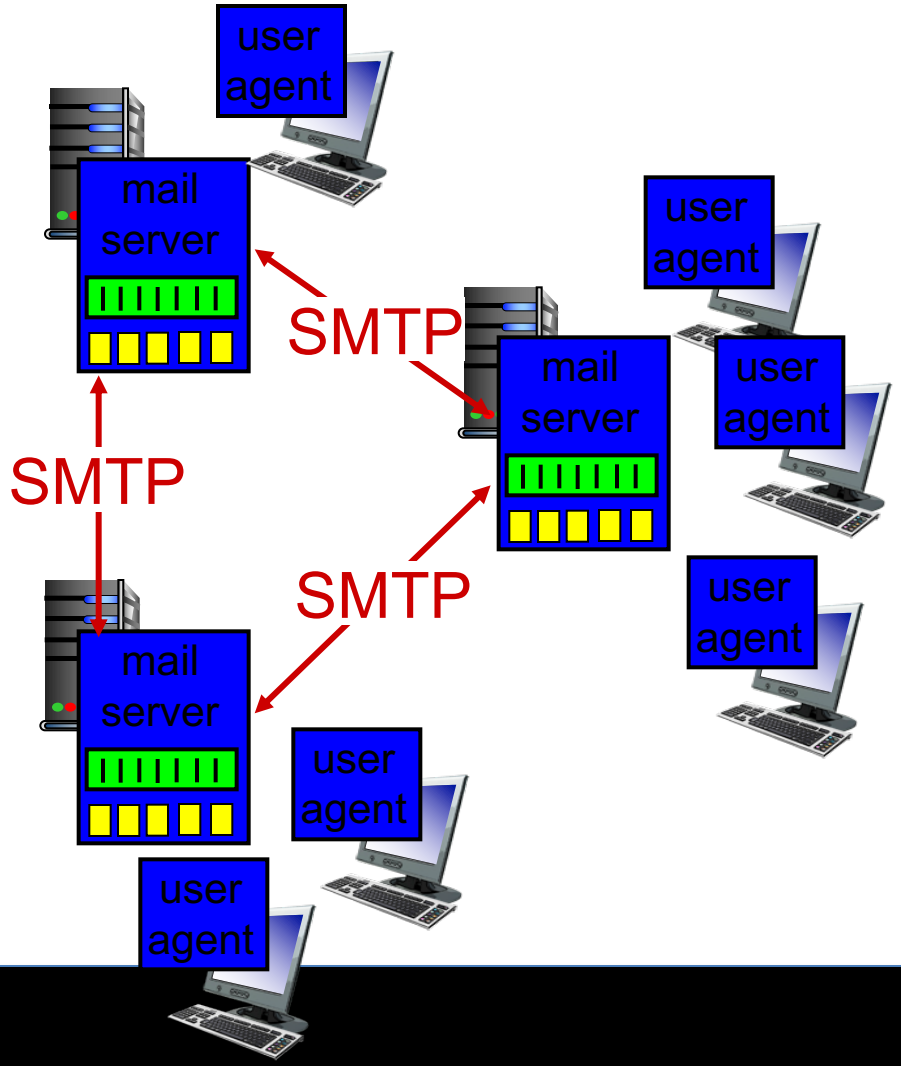
- A.K.A. “mail reader”
- Composing, editing, reading mail messages
- E.g., Outlook, Thunderbird, iPhone mail client
- Outgoing, incoming messages stored on server



Electronic Mail: Mail Servers

Mail servers:

- Mailbox contains incoming messages for user
- Message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages
  - Client: sending mail server
  - "Server": receiving mail server



Electronic Mail: SMTP

- Uses TCP to reliably transfer email message from client to server, port 25
- Direct transfer: sending server to receiving server
- Three phases of transfer
  - Handshaking (greeting)
  - Transfer of messages
  - Closure
- Command/response interaction (like HTTP, FTP)
  - Commands: ASCII text
  - Response: status code and phrase
- Messages must be in 7-bit ASCI

Alice Sends Message to Bob

1) Alice uses UA to compose message “to” bob@someschool.edu

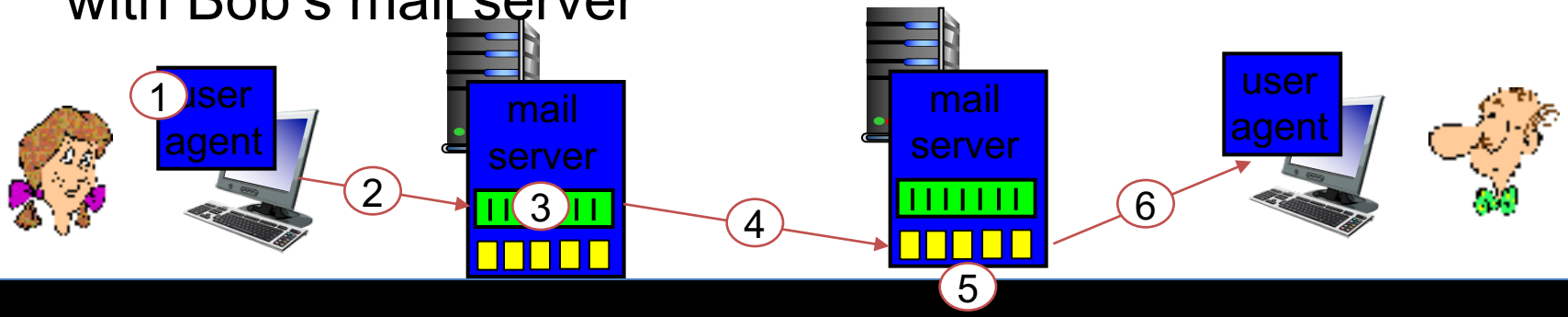
2) Alice’s UA sends message to her mail server; message placed in message queue

3) client side of SMTP opens TCP connection with Bob’s mail server

4) SMTP client sends Alice’s message over the TCP connection

5) Bob’s mail server places the message in Bob’s mailbox

6) Bob invokes his user agent to read message



Sample SMTP Interaction

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Try SMTP Interaction For Yourself

- **`telnet servername 25`**
- see 220 reply from server
- enter HELO, MAIL FROM, RCPT TO, DATA, QUIT commands

The above lets you send email without using email client (reader)

Note: You specify the "From" address!

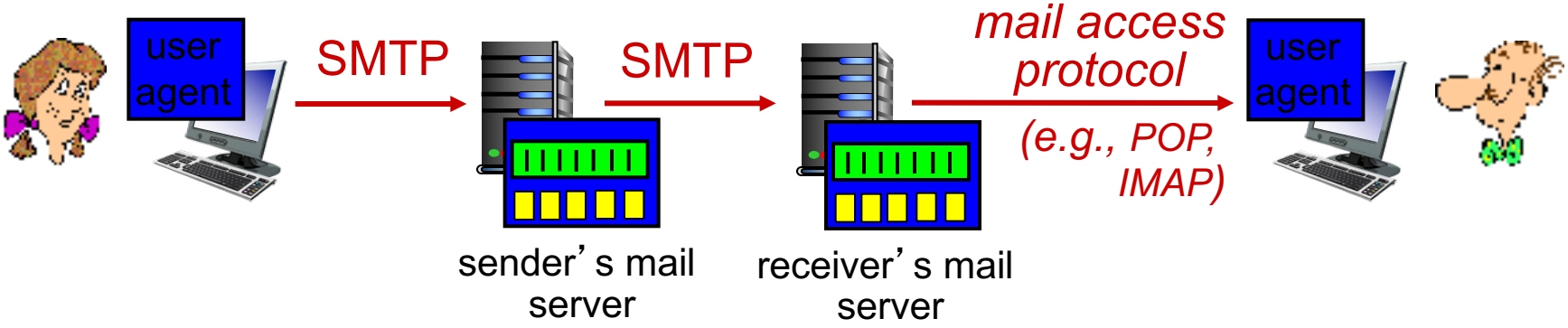
SMTP: Final Words

- SMTP uses persistent connections
- SMTP isn't encrypted!
  - You can read others' SMTP traffic
- SMTP doesn't validate the sender
  - You can falsely spoof an SMTP message
- Everything that makes SMTP secure is an add-on

Comparison with HTTP:

- HTTP: pull
- SMTP: push
- Both have ASCII command/response interaction, status codes
- HTTP: each object encapsulated in its own response msg
- SMTP: multiple objects sent in multipart msg

More Mail Access Protocols



- SMTP: delivery/storage to receiver's server
- Mail access protocol: retrieval from server
  - POP: Post Office Protocol: authorization, download
  - IMAP: Internet Mail Access Protocol: more features, including manipulation of stored messages on server
  - HTTP: Gmail, Yahoo! Mail, etc.
- None of these are encrypted!

Memes and Jokes

Networking Joke (1)

- “Knock knock”
- “Who’s there?”
- “SYN flood”
- “SYN flood who?”
- “Knock knock”

Networking Joke (2)

- “Did you hear about the chaos at the networking manufacturer?”

Networking Joke (2)

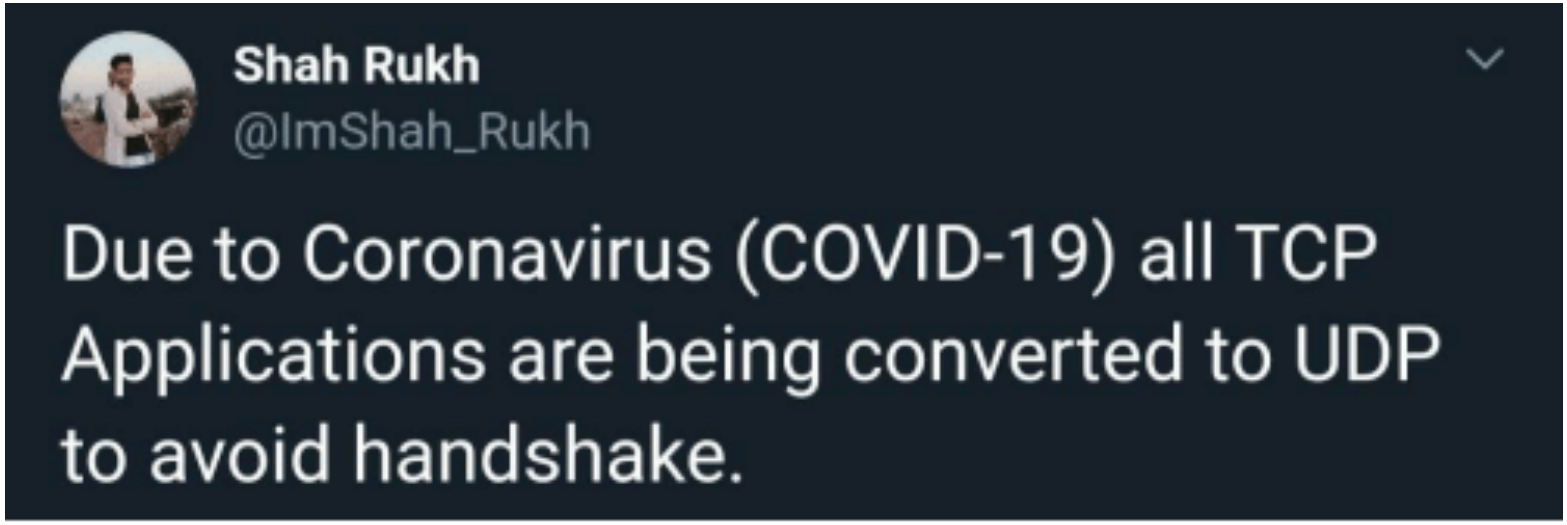
- “Did you hear about the chaos at the networking manufacturer?”
- It was Panic at the Cisco

"Hi, I'd like to hear a TCP joke."
"Hello, would you like to hear a TCP joke?"
"Yes, I'd like to hear a TCP joke."
"OK, I'll tell you a TCP joke."
"Ok, I will hear a TCP joke."
"Are you ready to hear a TCP joke?"
"Yes, I am ready to hear a TCP joke."
"Ok, I am about to send the TCP joke. It will last 10 seconds, it has two characters, it does not have a setting, it ends with a punchline."
"Ok, I am ready to get your TCP joke that will last 10 seconds, has two characters, does not have an explicit setting, and ends with a punchline."
"I'm sorry, your connection has timed out.
...Hello, would you like to hear a TCP joke?"


Kirk Bater
@KirkBater
Follow
This image is a TCP/IP Joke. This tweet is a UDP joke. I don't care if you get it.
Thread
iamkirkbater and jkjustjoshing
iamkirkbater Aug 23rd, 2017 at 9:37 AM
in #www
Do you want to hear a joke about TCP/IP?
7
7 replies
jkjustjoshing 5 months ago
Yes, I'd like to hear a joke about TCP/IP
iamkirkbater 5 months ago
Are you ready to hear the joke about TCP/IP?
jkjustjoshing 5 months ago
I am ready to hear the joke about TCP/IP
iamkirkbater 5 months ago
Here is a joke about TCP/IP.
iamkirkbater 5 months ago
Did you receive the joke about TCP/IP?
jkjustjoshing 5 months ago
I have received the joke about TCP/IP.
iamkirkbater 5 months ago
Excellent. You have received the joke about TCP/IP. Goodbye.

You vs the guy she tells you not to worry about.

UDP	TCP
Unreliable	Reliable
Connectionless	Connection-oriented
No windowing or retransmission	Segment retransmission and flow control through windowing
No sequencing	Segment sequencing
No acknowledgement	Acknowledge segments



UDP is the way to go

First Look at Security

*Slides adapted from Computer Networking: A top-Down Approach

Network Security

- Field of network security:
  - How bad guys can attack computer networks
  - How we can defend networks against attacks
  - How to design architectures that are immune to attacks
- Internet not originally designed with (much) security in mind
  - Original vision: “a group of mutually trusting users attached to a transparent network”
  - Internet protocol designers playing “catch-up”
  - security considerations in all layers!

Bad guys: Put Malware Into Hosts Via Internet

- Malware can get on host from:
  - Virus: self-replicating infection by receiving/executing object (e.g., e-mail attachment)
  - Worm: self-replicating infection by passively receiving object that gets itself executed
- Spyware malware can record keystrokes, web sites visited, upload info to collection site
- Infected host can be enrolled in botnet, used for spam. DDoS attacks

There Are Bad People Out There!

Q: What can a “bad guy (or girl)” do?

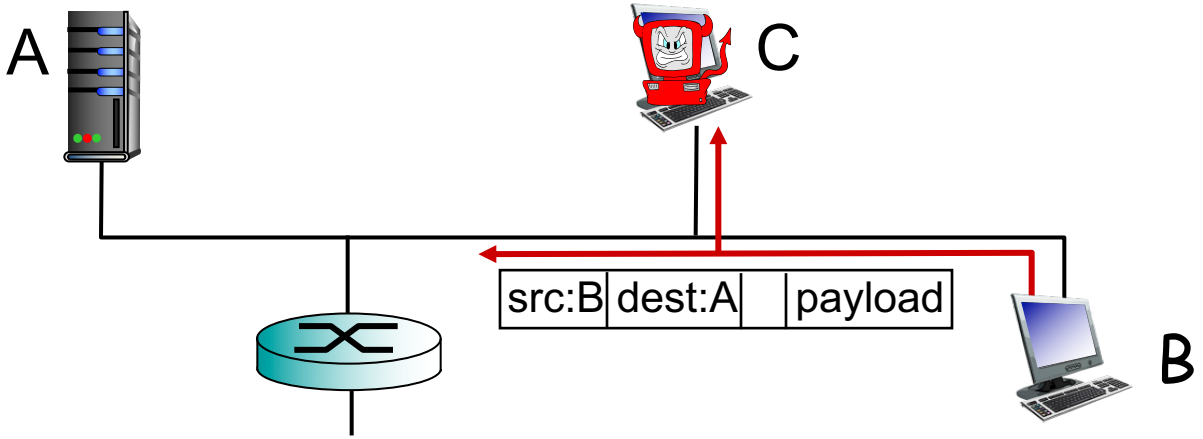
A lot!

- Eavesdrop: intercept messages
- Actively insert messages into connection
- Impersonation: can fake (spoof) source address in packet (or any field in packet)
- Hijacking: “take over” ongoing connection by removing sender or receiver, inserting himself in place
- Denial of service: prevent service from being used by others (e.g., by overloading resources)

Bad Guys Can Sniff Packets

Packet “sniffing”:

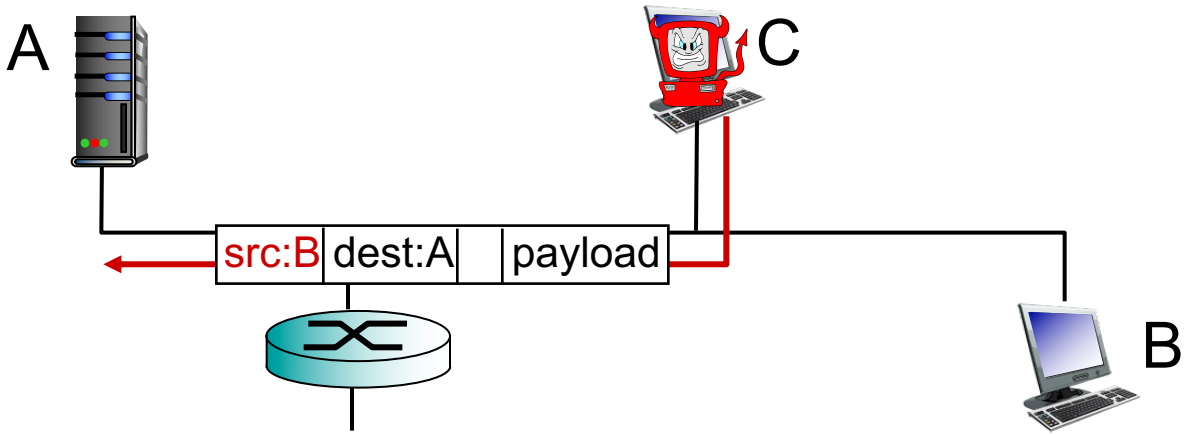
- Broadcast media (shared ethernet, wireless)
- Promiscuous network interface reads/records all packets (e.g., including passwords!) passing by



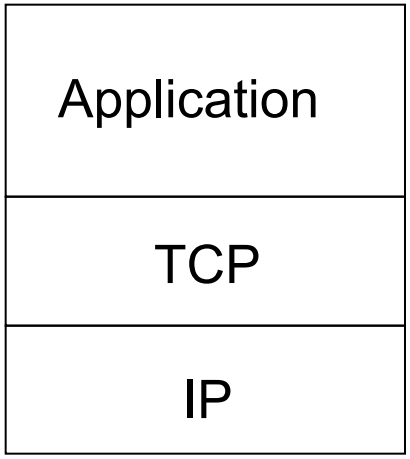
- Wireshark software is a (free) packet-sniffer

Adversaries Can Use Fake Addresses

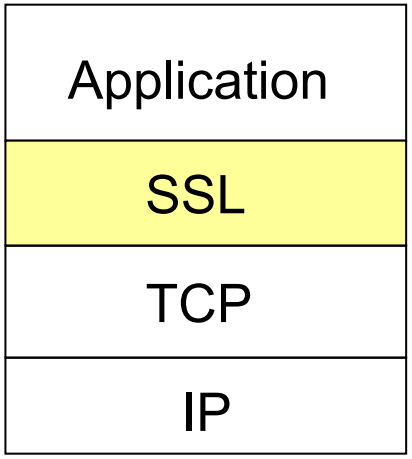
IP spoofing: send packet with false source address



SSL And TCP/IP



normal application



application with SSL

- SSL provides application programming interface (API) to applications
- C and Java SSL libraries/classes readily available

What Is Network-Layer Confidentiality?

Between two network entities:

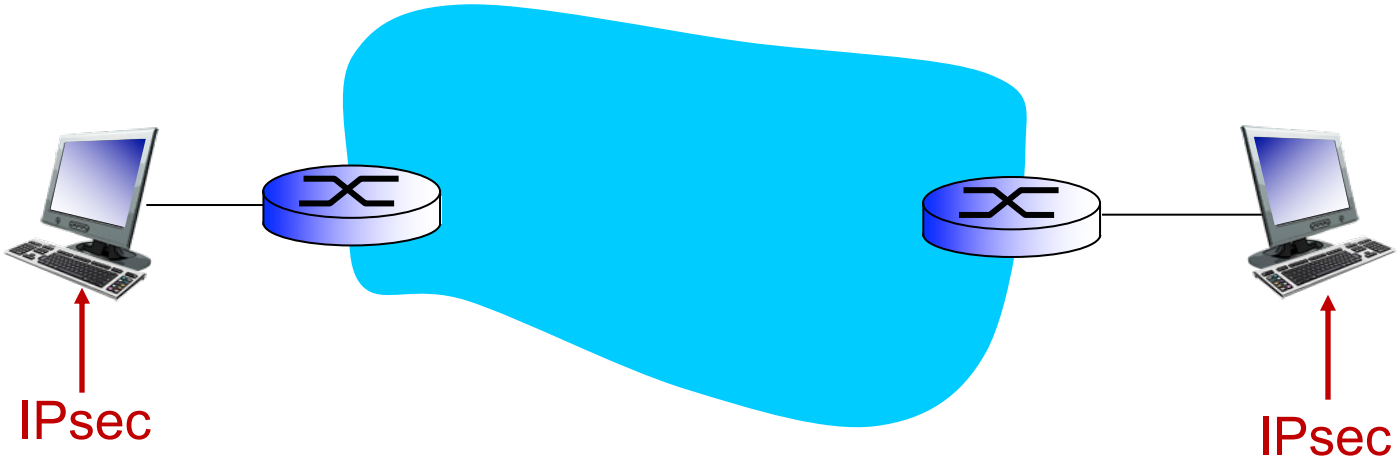
- Sending entity encrypts datagram payload, payload could be:
  - TCP or UDP segment, ICMP message, OSPF message ….
- All data sent from one entity to other would be hidden:
  - Web pages, e-mail, P2P file transfers, TCP SYN packets …
- “Blanket coverage”

Virtual Private Networks (VPNs)

Motivation:

- Institutions often want private networks for security.
  - Costly: separate routers, links, DNS infrastructure.
- VPN: institution's inter-office traffic is sent over public Internet instead
  - Encrypted before entering public Internet
  - Logically separate from other traffic

IPsec Transport Mode



- IPsec datagram emitted and received by end-system
- Protects upper level protocols

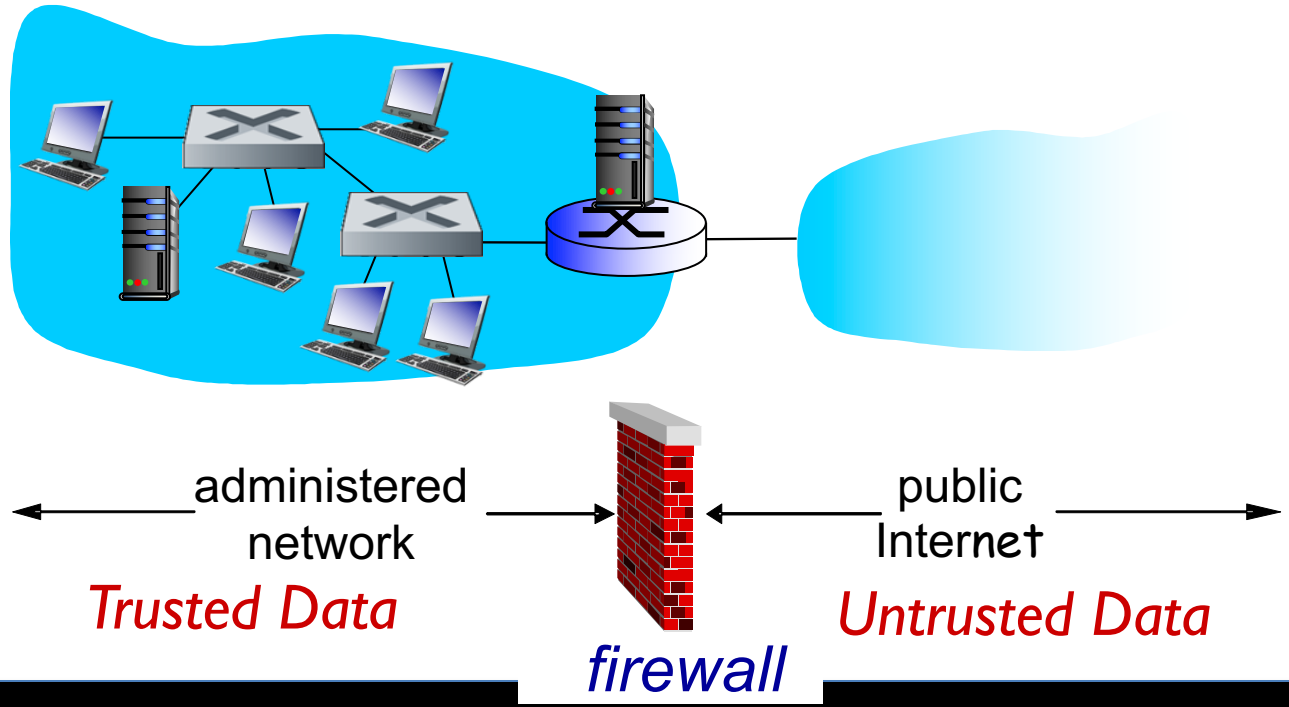
Wireless Security

- Establish encryption to prevent traffic sniffing
- We won’t have time to cover here
  - May make a good research project!
- The gist: WEP is bad; WPA is better; use WPA2

Firewalls

Firewall

isolates organization's internal net from larger Internet, allowing some packets to pass, blocking others



Firewalls: Why

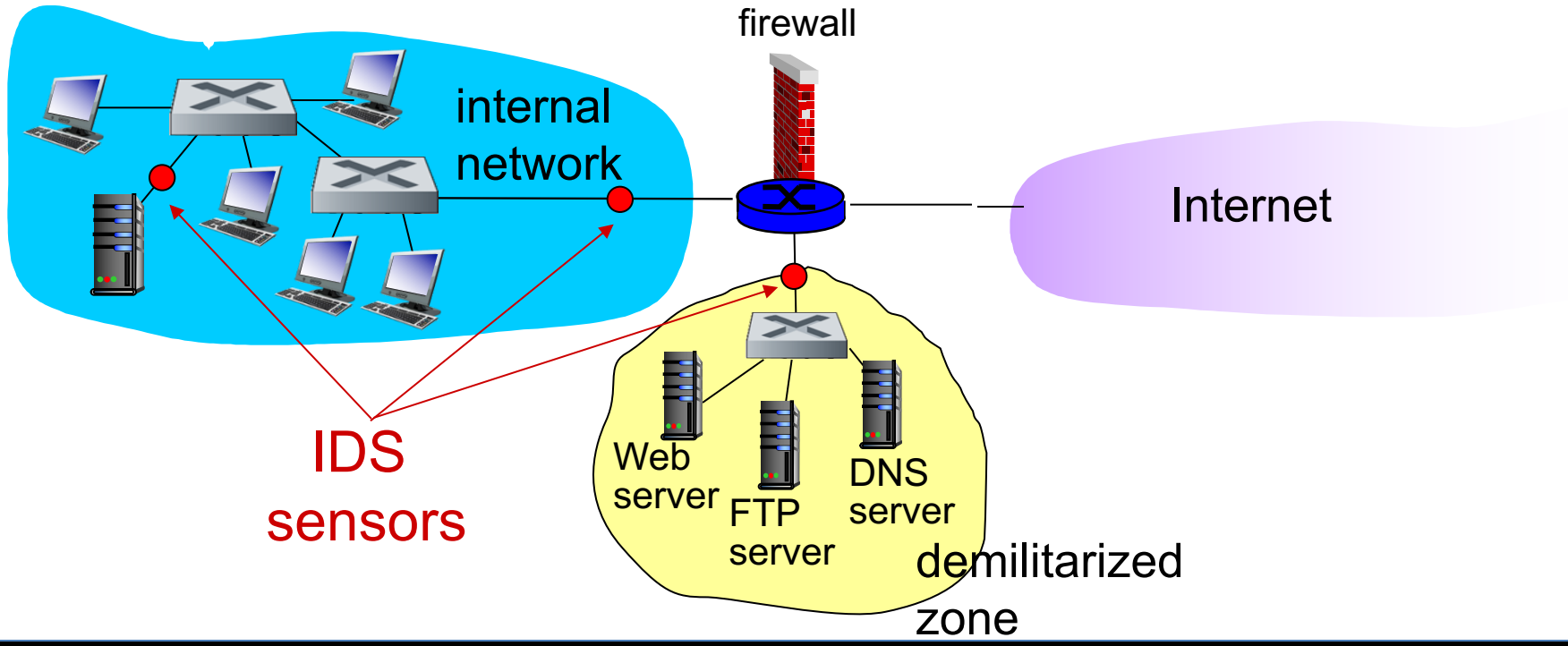
- Prevent denial of service attacks:
  - SYN flooding: attacker establishes many bogus TCP connections, no resources left for “real” connections
- Prevent illegal modification/access of internal data
  - E.g., attacker replaces CIA’s homepage with something else
- Allow only authorized access to inside network
  - Set of authenticated users/hosts
- Three types of firewalls:
  - Stateless packet filters
  - Stateful packet filters
  - Application gateways

Intrusion Detection Systems

- Packet filtering:
  - Operates on TCP/IP headers only
  - No correlation check among sessions
- IDS: Intrusion Detection System
  - Deep Packet Inspection (DPI): look at packet contents (e.g., check character strings in packet against database of known virus, attack strings)
  - Examine correlation among multiple packets
    - Port scanning
    - Network mapping
    - DoS attack

Intrusion Detection Systems

- Multiple IDSs: different types of checking at different locations



Dynamic Host Configuration Protocol (DHCP)

Local IP Addresses: How to Get One?

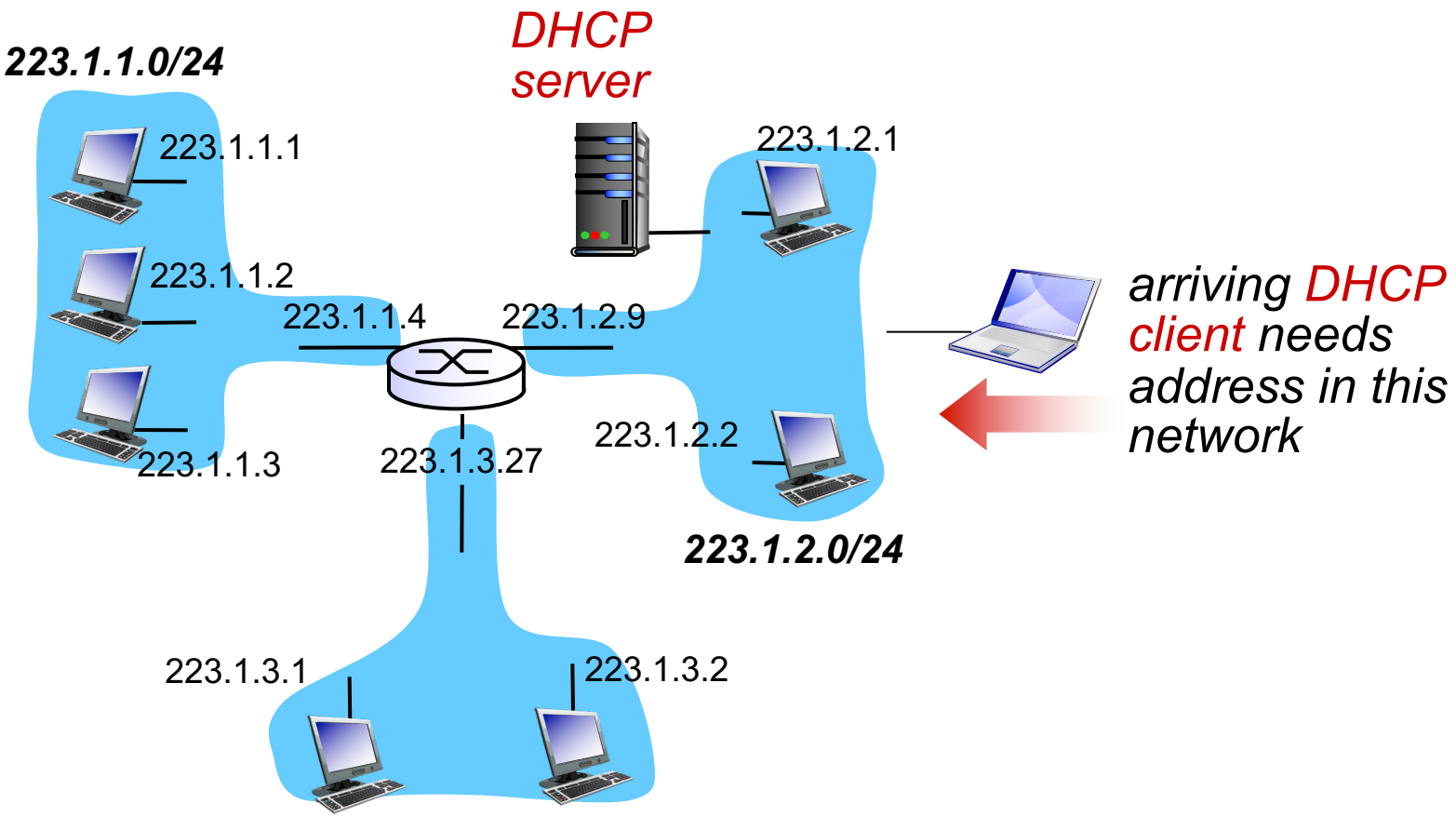
Q: How does a host get IP address?

- Hard-coded by system admin in a file
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - “Plug-and-play”

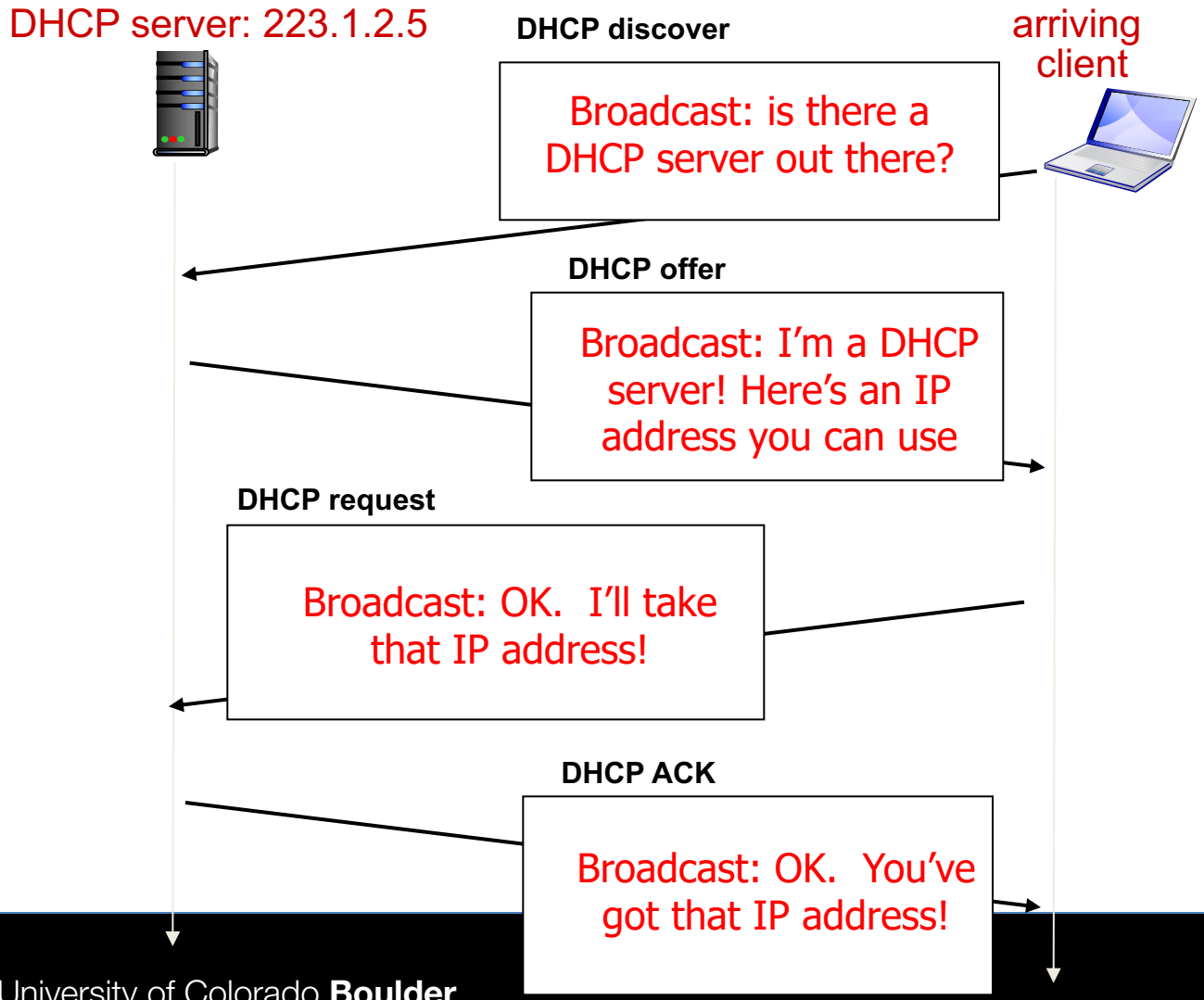
DHCP: Dynamic Host Configuration Protocol

- Goal: allow host to dynamically obtain its IP address from network server when it joins network
  - Can renew its lease on address in use
  - Allows reuse of addresses (only hold address while connected/“on”)
  - Support for mobile users who want to join network (more shortly)
- DHCP overview:
  - Host broadcasts “DHCP discover” msg [optional]
  - DHCP server responds with “DHCP offer” msg [optional]
  - Host requests IP address: “DHCP request” msg
  - DHCP server sends address: “DHCP ack” msg

DHCP client-server scenario



DHCP client-server scenario

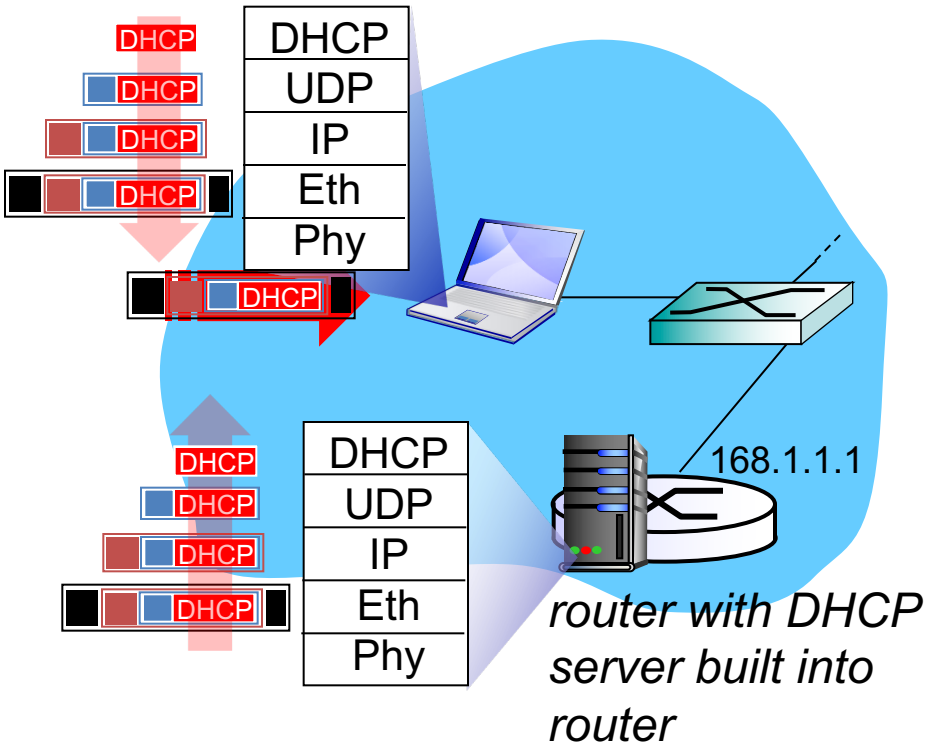


DHCP: more than IP addresses

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

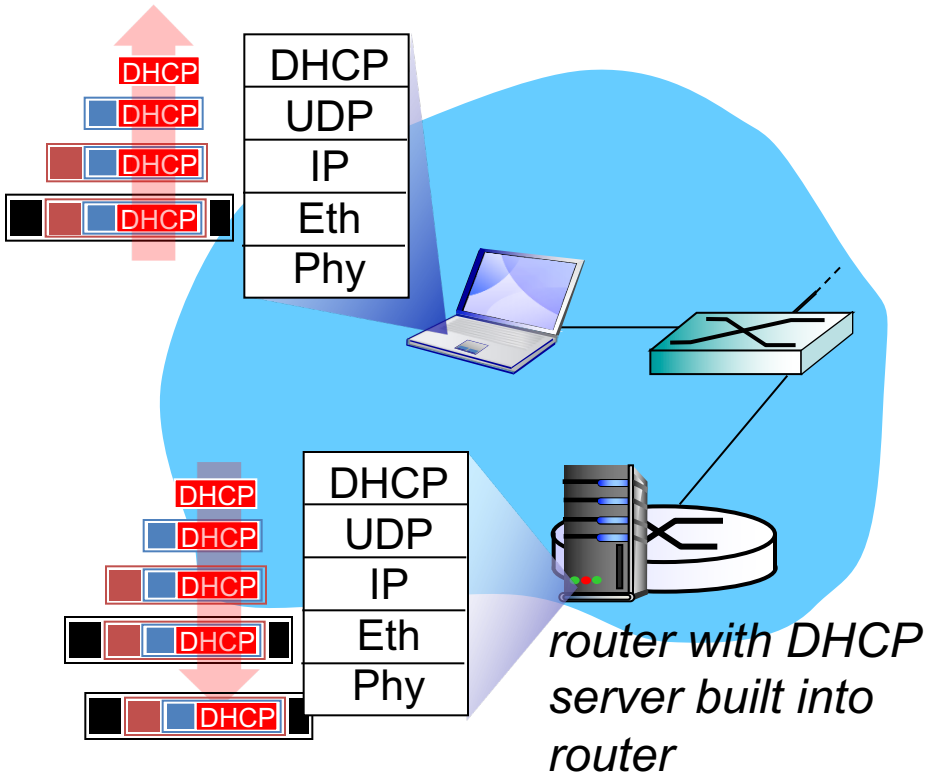
- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

DHCP: example



- **connecting laptop needs its IP address, addr of first-hop router, addr of DNS server: use DHCP**
- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.1 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

DHCP: example



- **DCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server**

- encapsulation of DHCP server, frame forwarded to client, demuxing up to DHCP at client
- client now knows its IP address, name and IP address of DSN server, IP address of its first-hop router

DHCP: Wireshark output (home LAN)

request

Message type: **Boot Request (1)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) **DHCP Message Type = DHCP Request**
Option: (61) Client identifier
    Length: 7; Value: 010016D323688A;
    Hardware type: Ethernet
    Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Option: (t=50,l=4) Requested IP Address = 192.168.1.101
Option: (t=12,l=5) Host Name = "nomad"
Option: (55) Parameter Request List
    Length: 11; Value: 010F03062C2E2F1F21F92B
    1 = Subnet Mask; 15 = Domain Name
    3 = Router; 6 = Domain Name Server
    44 = NetBIOS over TCP/IP Name Server

reply

Message type: **Boot Reply (2)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 192.168.1.101 (192.168.1.101)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 192.168.1.1 (192.168.1.1)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) DHCP Message Type = DHCP ACK
Option: (t=54,l=4) Server Identifier = 192.168.1.1
Option: (t=1,l=4) Subnet Mask = 255.255.255.0
Option: (t=3,l=4) Router = 192.168.1.1
Option: (6) Domain Name Server
    Length: 12; Value: 445747E2445749F244574092;
    IP Address: 68.87.71.226;
    IP Address: 68.87.73.242;
    IP Address: 68.87.64.146
Option: (t=15,l=20) Domain Name = "hsd1.ma.comcast.net."

IP addresses: how to get one?

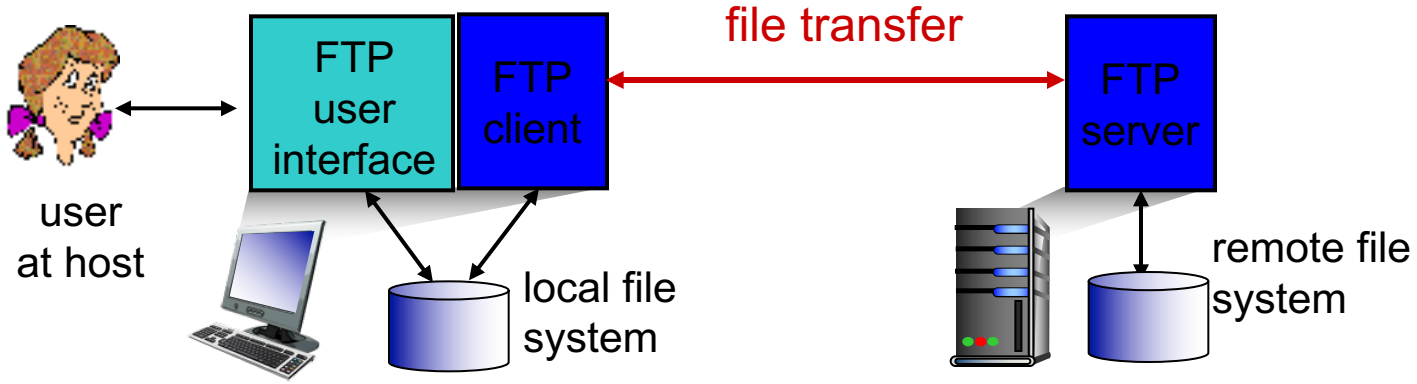
Q: how does *network* get subnet part of IP addr?

A: gets allocated portion of its provider ISP's address space

ISP's block	11001000 00010111 00010000	00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000	00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010	00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100	00000000	200.23.20.0/23
...	.....	....	....
Organization 7	11001000 00010111 00011110	00000000	200.23.30.0/23

File Transfer Protocol (FTP)

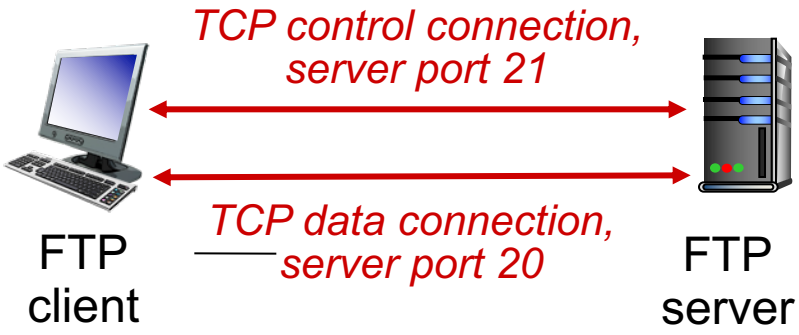
FTP: The File Transfer Protocol



- Transfer file to/from remote host
- Client/server model
  - Client: side that initiates transfer (either to/from remote)
  - Server: remote host
- FTP server: port 21
- **Secure alternatives**: SFTP or SCP (different-ish use)

FTP: Separate Control, Data Connections

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



- Server opens another TCP data connection to transfer another file
- FTP server maintains “state”: current directory, earlier authentication

FTP Commands And Responses

Sample commands:

- Sent as ASCII text over control channel
- USER username
- PASS password
- LIST return list of file in current directory
- RETR filename retrieves (gets) file
- STOR filename stores (puts) file onto remote host

Sample return codes

- Status code and phrase (as in HTTP)
- 331 Username OK, password required
- 125 data connection already open; transfer starting
- 425 Can’t open data connection
- 452 Error writing file