Introduction to Networks

CS439: Principles of Computer Systems

April 13, 2015

Where We Are In The Course

- We've done:
  - Processes
  - Threads
  - Synchronization
  - Virtual Memory
  - File Systems
  - Disks
- We have our *almost* all our building blocks!
- One more: networks
- And then… combinations:
  - Parallel and Distributed Computing
  - Distributed and Cluster File Systems
  - Security

This Time

- Introduction to Networks
- What they are
  - Hardware structure
  - Logical structure
- Network communication
  - Protocols, naming, routing
- TCP/IP congestion control mechanisms

Networks

Networks

- A system of lines or channel that interconnect
  - railroads, telephones
- Computer networks
  - hierarchical systems of boxes and wires
    - usually concerned with providing efficient, correct, and robust message passing between two separate nodes (which are usually computers)
    - organized by geographical proximity

Networks: OS View

OS see the network as just another device

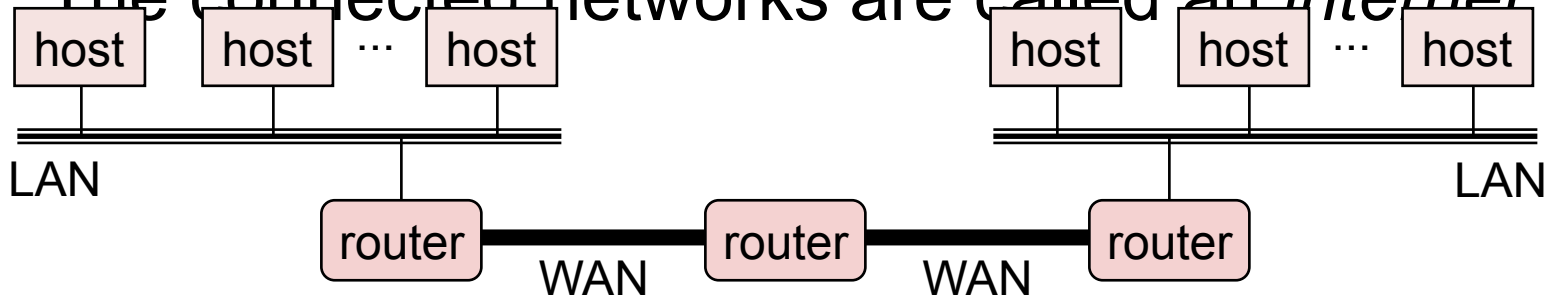
- Network Interface Controller (NIC) is added to the bus
- Transfer data to/from memory to NIC through DMA or memory-mapped I/O

Levels of Networks

- System Area Network (SAN)
  - Connects cluster or machine room
  - e.g., Quadrics
- Local Area Network (LAN)
  - Connects nodes in a single building
  - Needs to be fast and reliable
  - e.g., Ethernet
- Wide Area Network (WAN)
  - Connects nodes across the state, country, or planet
  - Typically slower and less reliable than LAN
  - Often high-speed point-to-point phone lines

Next Level: internets

- Multiple incompatible LANs/WANs can be physically connected by specialized computers called *routers*
  - More on routers in a minute
- The connected networks are called an *internet*



- The Global IP Internet is the most famous example of an internet

Next Level: internets
Plain Text

- Multiple incompatible LANs/WANs can be physically connected by specialized computers called routers
  - More on routers in a minute
- The connected networks are called internet
- The Global IP Capital ‘I’ Internet is the most famous example of an internet (spelled with a lower-case ‘i’)
- Physical connections
  - LAN connects to hosts directly
  - WAN lines connect routers
  - Routers connect LAN and WAN together

Three Steps to Network Communication

- **Protocols** to encode data
- **Naming** the machine with which you are communicating
- **Routing** information from a source to a destination

Layers of the Network

Open Source Interconnection 7 Layer Model

Provides common vocabulary for network engineers (generally the developers stay out of this level of detail).

Layer 1 is Hardware
Layer 2 is Switching
Layer 3 is Routing
Layer 4 is TCP
Layer 5 is often OS/Library
Layer 6 is often OS/Library
Layer 7 is Application

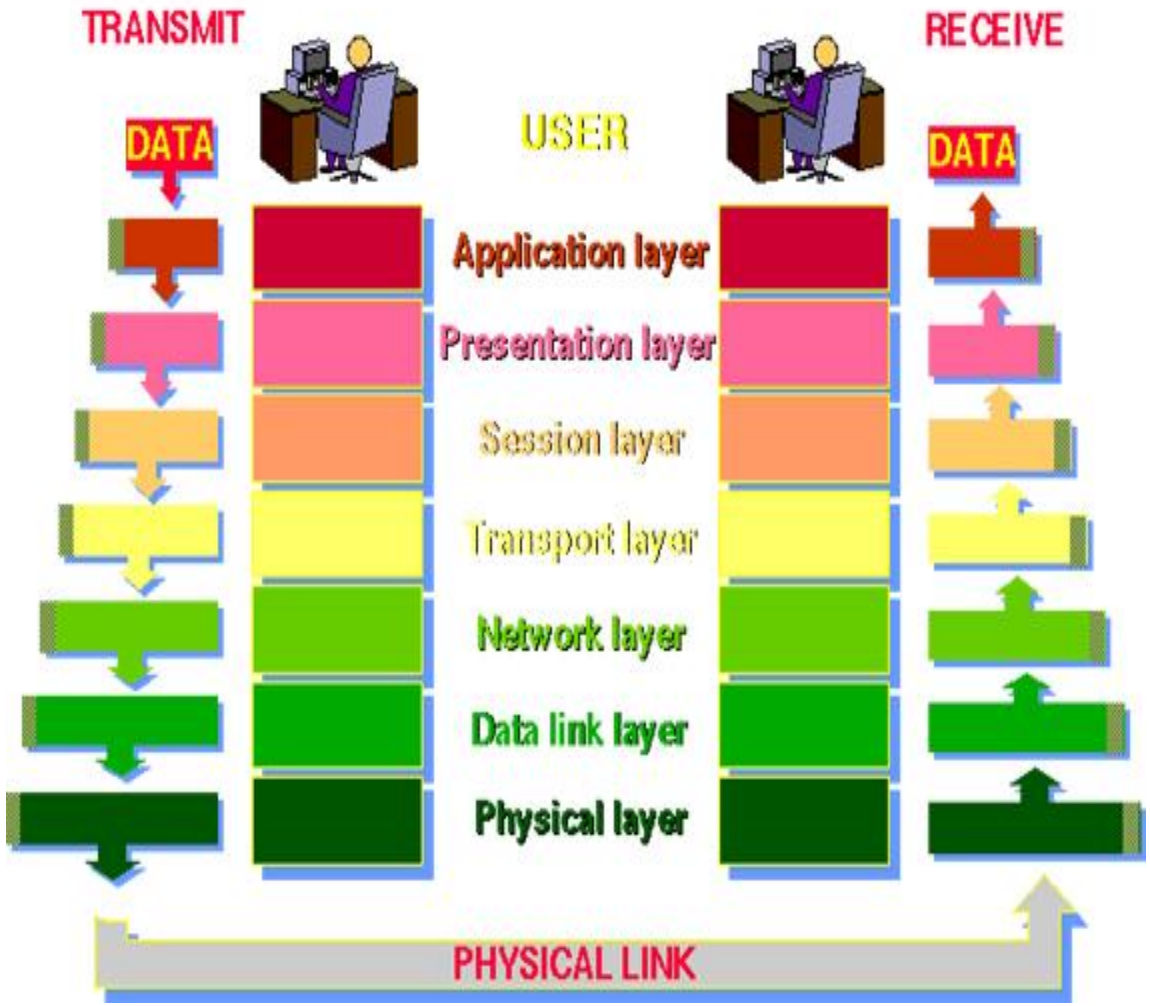
OSI (Open Source Interconnection) 7 Layer Model

Layer	Application/Example		Central Device/ Protocols			DOD4 Model
Application (7) Serves as the window for users and application processes to access the network services.	**End User layer** Program that opens what was sent or creates what is to be sent Resource sharing • Remote file access • Remote printer access • Directory services • Network management		**User Applications** SMTP		**GATEWAY** Can be used on all layers	Process
Presentation (6) Formats the data to be presented to the Application layer. It can be viewed as the "Translator" for the network.	**Syntax layer** encrypt & decrypt (if needed) Character code translation • Data conversion • Data compression • Data encryption • **Character Set Translation**		JPEG/ASCII EBDIC/TIFF/GIF PICT			
Session (5) Allows session establishment between processes running on different stations.	**Synch & send to ports** (logical ports) Session establishment, maintenance and termination • Session support - perform security, name recognition, logging, etc.		**Logical Ports** RPC/SQL/NFS NetBIOS names			
Transport (4) Ensures that messages are delivered error-free, in sequence, and with no losses or duplications.	**TCP** Host to Host, Flow Control Message segmentation • Message acknowledgement • Message traffic control • Session multiplexing	PACKET FILTERING	TCP/SPX/UDP			Host to Host
Network (3) Controls the operations of the subnet, deciding which physical path the data takes.	**Packets** ("letter", contains IP address) Routing • Subnet traffic control • Frame fragmentation • Logical-physical address mapping • Subnet usage accounting		**Routers** IP/IPX/ICMP			Internet
Data Link (2) Provides error-free transfer of data frames from one node to another over the Physical layer.	**Frames** ("envelopes", contains MAC address) [NIC card — Switch — NIC card] (end to end) Establishes & terminates the logical link between nodes • Frame traffic control • Frame sequencing • Frame acknowledgment • Frame delimiting • Frame error checking • Media access control		**Switch** **Bridge** **WAP** PPP/SLIP	Land Based Layers		Network
Physical (1) Concerned with the transmission and reception of the unstructured raw bit stream over the physical medium.	**Physical structure** Cables, hubs, etc. Data Encoding • Physical medium attachment • Transmission technique - Baseband or Broadband • Physical medium transmission Bits & Volts		**Hub**			

Layers of the Network: Plain Text

- Open Source Interconnection 7 layer model (OSI Model)
- Provides common vocabulary for network engineers (generally the developers stay out of this level of detail)
- Layer 1 is Hardware
  - Is physical: concerned with the transmission and reception of the unstructured raw bit stream over the physical medium
  - Physical structure: cables, hubs etc.
    - Data encoding, physical medium attachment, transmission technique :baseband or broadband, physical medium transmission bits and volts
  - Central device/protocols: hub and land-based layers
  - DOD4 Model:: Network
- Layer 2 is Switching
  - Data Link - provides error-free transfer of data frames from one node to another over the physical layer
  - Frames - “envelopes”, contains MAC address
    - NIC card - switch - NIC card, is end to end
    - Establishes and terminates the logical link between nodes, frame traffic control, frame sequencing, frame acknowledgement, frame delimiting, frame error checking, media access control
  - Central device/protocols: switch bridge WAP, PPP/SLIP, and land-based layers
  - DOD4 Model: Network
- Layer 3 is Routing
  - Network: controls the operation of the subnet, deciding which physical path the data takes
  - Packets: “letter”, contains IP address
    - Routing, subnet traffic control, frame fragmentation, logical-physical address mapping, subnet usage accounting
    - Packet filtering
  - Central device/protocols - routers, IP/IPX/ICMP
  - DOD4 Model: internet
- Layer 4 is TCP
  - Transport: ensures that message are delivered error-free, in sequence, and with no losses or duplications
  - TCP: host-to-host, flow control
    - Message segmentation, message acknowledgement, message traffic control, session multiplexing
    - Packet filtering
  - Central device/protocols - TCP/SPX/UDP
  - DOD4 Model: host-to-host
- Layer 5 is often the OS/Library
  - Session: allows session establishment between processes running on different stations
  - Synch and send to ports - logical ports
    - Session establishment, maintenance and termination, session support, perform security, name recognition, logging etc
  - Central devices/protocols: logical parts, RPC/SQL/NFS, NetBIOS names
  - DOD4 Model: process
- Layer 6 is often the OS/Library
  - Presentation: formats the data to be presented to the application layer, it can be viewed as the “Translator” for the network
  - Syntax layer: encrypt and decrypt (if needed)
    - character code translation, data conversion, data compression, data encryption, character set translation
  - Central devices/ protocols - JPEG/ASCII/EBDIC/TIFF/GIF/PICT
  - DOD4 Model: process
- Layer 7 is Application
  - Application: serves as window for users and application processes to access the network services
  - End User Layer: program that opens what was sent or creates what is to be sent
    - Resource sharing, remote file access, remote printer access, directory services, network management
  - Central device/protocols: user applications, SMTP
  - DOD4 model: process
- NOTE: All layers have the Gateway as part of Central Device/Protocols

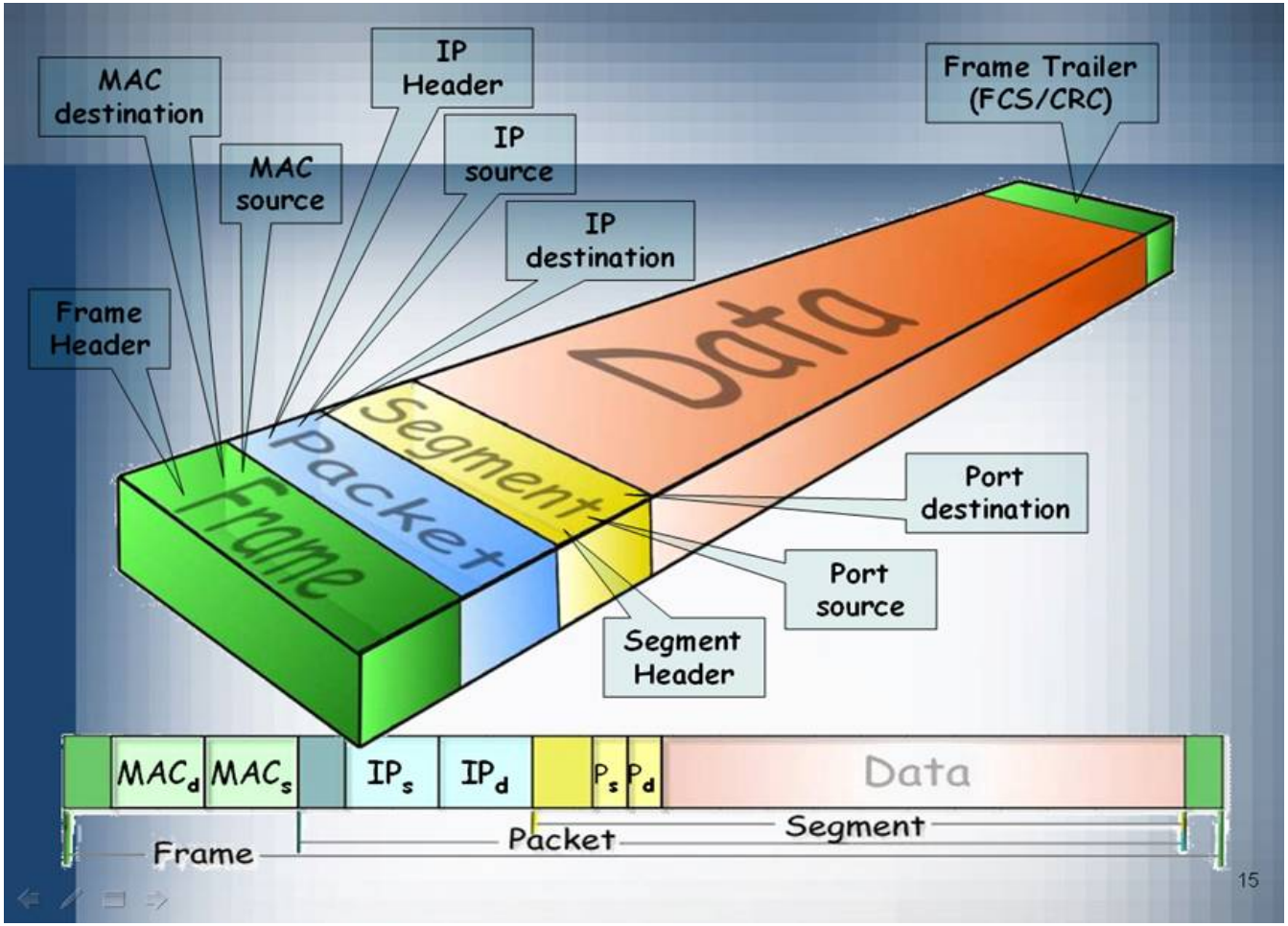
THE 7 LAYERS OF OSI



The 7 Layers of OSI: Text Description

- Data Transmission Layer order
  - application, presentation, session, transport, network, data link, physical
- Between transmission and receiving of data there is a physical link
- Data Receiving Layer Order
  - physical, data link, network, transport, session, presentation, application

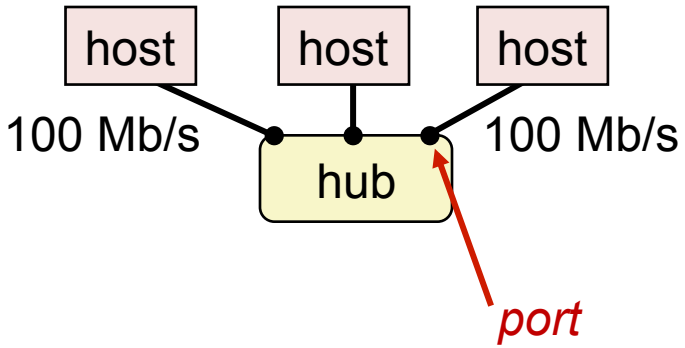
Structure of Packet



Structure of Packet: Text Description

- NOTE: in order from beginning to end of packet
- Frame
  - Header includes: MAC destination, MAC source
  - Packet is its payload
  - Trailer includes: FCS/CRC (error checking information)
- Packet
  - Header includes: IP source, IP destination
  - Segment is its payload
- Segment
  - Header includes: Port source, Port destination
  - Data included here

Layer 2 Example: Ethernet

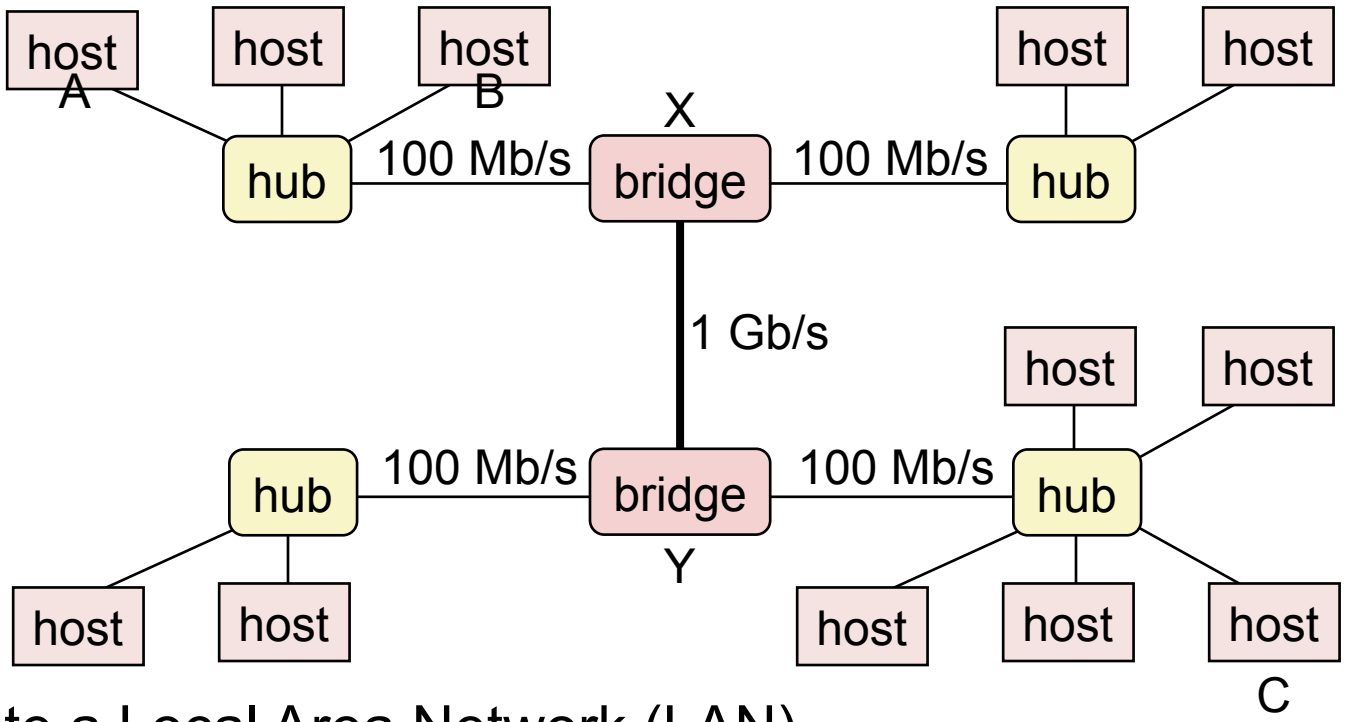


- Invented Xerox PARC, 1973
  - co-invented by Bob Metcalfe, who is now at UT!
- Connects collection of *hosts* connected by wires (twisted pairs) to a *hub*
- Hub slavishly copies each bit from each port to every other port
  - Every host sees every bit
- Each Ethernet adapter (on each host) has a unique 48-bit address (**M**edium **A**ccess **C**ontrol address)
  - e.g., 00:16:ea:e3:54:e6
- Bandwidth from 3MB/s to 1 GB/s

Layer 2 Example: Ethernet Plain Text

- Example physical structure
  - Hub connects to hosts via ports
  - Information may travel at speeds up to 100 Mb/s
- Invented Xerox PARC, 1973
  - Co-invented by Bob Metcalfe, who is now at UT
- Connects collection of hosts connected by wires (twisted pairs) to a hub
- Hub slavishly copies each bit from each port to every other port
  - Every host sees every bit
- Each Ethernet adapter (on each host) has a unique 48-bit address
  - Medium Access Control (MAC) address
  - e.g. 00:16:ea:e3:54:e6
- Bandwidth from 3 MB/s to 1 GB/s

Example: Connected Ethernet Segments



- Create a Local Area Network (LAN)
- Bridges cleverly learn which hosts are reachable from which ports and then selectively copy frames from port to port
- Becoming cheap enough to replace hubs
- BUT… Ethernet has its own protocol. How do we communicate across network types? (Ethernet, wireless, DSL …)

Example: Connected Ethernet Segments Plain Text

- Create a Local Area Network (LAN)
- Example LAN physical structure
  - has 2 bridges - bridge X and bridge Y
    - are connected by 1 Gb/s connection
  - each bridge connected to 2 hubs with 100 Mb/s connection
  - hubs connected to differing numbers of hosts
- Bridges cleverly learn which hosts are reachable from which ports and then selectively copy frames from port to port
- Becoming cheap enough to replace hubs
- BUT… Ethernet has its own protocol. How do we communicate across network types? (Ethernet, wireless, DSL…)

The Notion of an internet Protocol

How is it possible to send bits across incompatible LANs and WANs?

- Protocol software runs on each host and router
- Smooths out the differences between the different networks
- Implements an *internet protocol* (Layer 3)
  - Lower-case ‘i’ internet.

internet Protocol

- Provides a *naming scheme*
  - An internet protocol defines a uniform format for *host addresses*
  - Each host (and router) is assigned at least one of these internet addresses that uniquely identifies it
- Provides a *routing mechanism*

internet Protocol

- Provides a *delivery mechanism*
  - Governs how hosts and routers should cooperate when they transfer data from network to network
  - Defines a standard transfer unit (*packet*)
  - Packet consists of *header* and *payload*
    - Header contains info such as packet size, source and destination addresses
    - Payload contains data bits sent from source host
- Most computers today speak IP
- TCP/IP is the protocol for the global IP Internet (but it's Layers 3 and 4…)

Naming

- Every host node has a unique string name and a 32-bit Internet number called its host-id (or thinks it does…)
- Host-ids are numbers (128.83.130.33); one for each level of an address (sloth.cs.utexas.edu)
  - Symbolic names have the most significant field first; numeric names have the least significant field first
- The OS needs to map logical addresses (*machine name, application*) to the host-id and **port** for the application
- Each machine has a set of local names and the *Gateway* to which it sends messages directly
- Gateway routes message to recipient
  - More in a minute

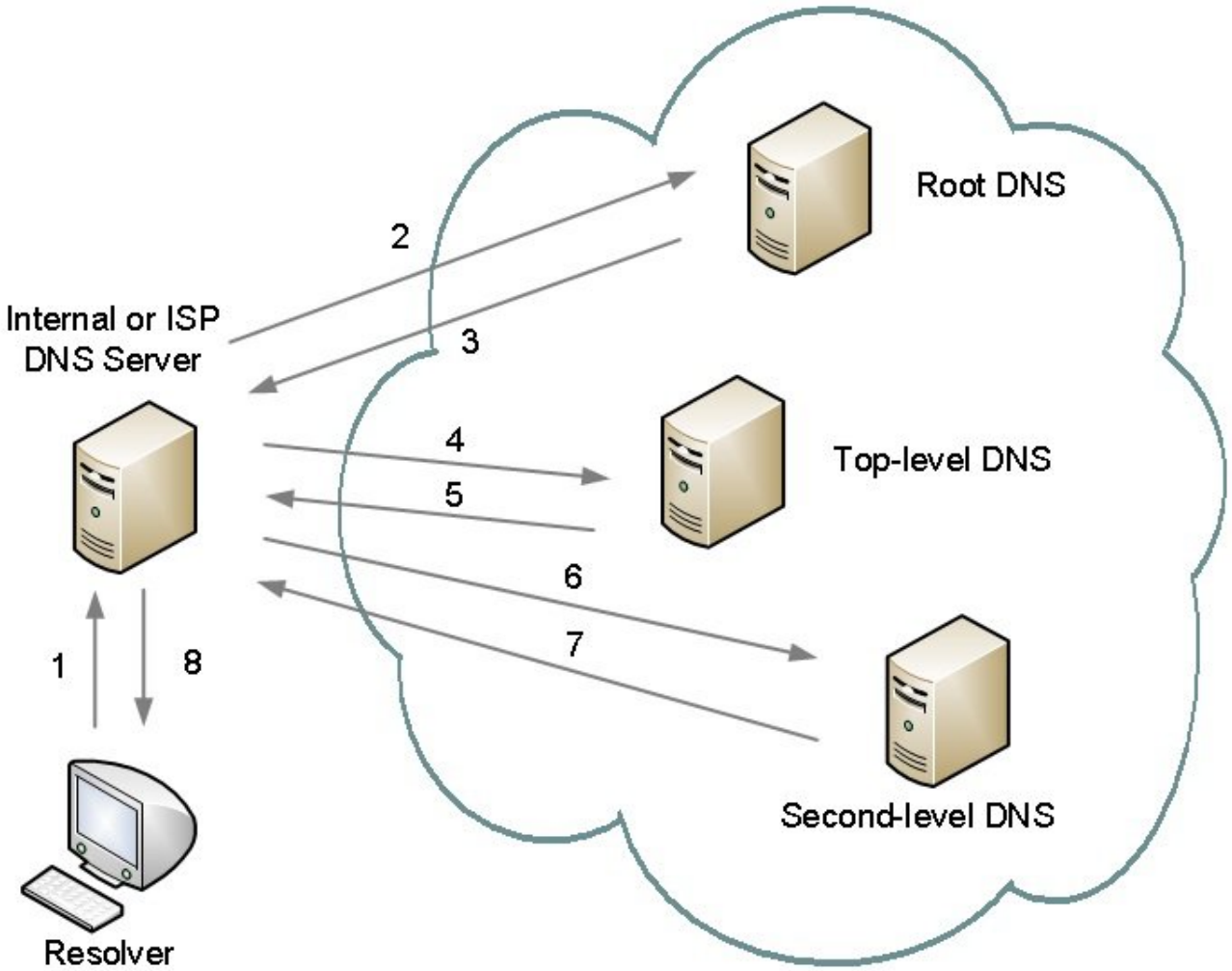
Routing

- Application sends a message to a name, routing software turns name into an Internet address and send the message on its way
- Packets are routed based on destination IP address
  - Address space is structured to make routing practical at global scale
  - For example, 128.83.*.* goes to UT
  - So packets need MAC address and IP addresses (is that it?)
- Also need to send message to right process on the right host
  - A process can create a port on the host
    - UTCS web server: 128.83.120.39:80

Domain Name Service

- The Internet maintains a mapping between IP addresses and domain names in a huge worldwide distributed database called *DNS*
  - Sender finds IP address of intended receiver here
  - Distributed hierarchical database
- To reduce lookup overhead, the Gateway node (which is being used in our example) stores *routing tables* which contain a cache of internet addresses

DNS


Root DNS
2
3
Internal or ISP
DNS Server
4
5
Top-level DNS
6
7
1
8
Second-level DNS
Resolver

DNS: Text Description

Connections from and back to the resolver (computer making the request)
Example: Assume wants to access www.ox.ac.uk

1. From resolver to internal or ISP DNS Server
   - Do you know the IP address for www.ox.ac.uk?
2. Internal or ISP DNS Server to Root DNS
   - Asking root…
3. Root DNS back to Internal or ISP DNS Server
   - No, but here is how you reach the uk server
4. Internal or ISP DNS Server to Top-level DNS
   - UK server, do you know the IP address for www.ox.ac.uk?
5. Top-Level DNS back to Internal or ISP DNS Server
   - No, but here is how you reach the ac.uk server
6. Internal or ISP DNS Server to Second-level DNS
   - ac.uk server, do you know the IP address for www.ox.ac.uk?
7. Second-level DNS back to Internal or ISP DNS Server
   - Sure! It's 129.67.242.155 (or that's one of them…)
8. Internal or ISP DNS Server back to resolver
   - Use 129.67.242.155

Domain Naming System (DNS)

- Conceptually, programmers can view the DNS database as a collection of millions of *host entry structures*:

```
/* DNS host entry structure */
struct hostent {
   char   *h_name;        /* official domain name of host */
   char   **h_aliases;    /* null-terminated array of domain names */
   int    h_addrtype;     /* host address type (AF_INET) */
   int    h_length;       /* length of an address, in bytes */
   char   **h_addr_list;  /* null-terminated array of in_addr structs */
};
```

- Functions for retrieving host entries from DNS:

  `gethostbyname()`: query key is a DNS domain name.

  `gethostbyaddr()`: query key is an IP address.

Domain Naming System (DNS): Plain Text

- Conceptually, programmers can view the DNS database as a collection of millions of host entry structures

```
struct hostent {
    char *h_name; /*official domain name of host*/
    char **h_aliases; /*null-terminated array of domain names*/
    int h_addrtype; /*host address type (AF_INET)*/
    int h_length; /*length of an address, in bytes*/
    char **h_addr_list; /*null-terminated array of in_addr structs*/ };
```

- Functions for retrieving host entries from DNS

  gethostbyname(): query key in DNS domain name

  gethostbyaddr(): query key is an IP address

Properties of DNS Host Entries

- Each host entry is an equivalence class of domain names and IP addresses
- Each host has a locally defined domain name `localhost` which always maps to the *loopback address* `127.0.0.1`
- Different kinds of mappings are possible:
  - Simple case: one-to-one mapping between domain name and IP address:
    - **`sloth.cs.utexas.edu`** maps to **`128.83.130.33`**
  - Multiple domain names mapped to the same IP address:
    - **`eecs.mit.edu`** and **`cs.mit.edu`** both map to **`18.62.1.6`**
  - Multiple domain names mapped to multiple IP addresses:
    - **`google.com`** maps to multiple IP addresses
  - Some valid domain names don't map to any IP address:
    - for example: **`ics.cs.cmu.edu`**


There's no
place like
127.0.0.1

TCP/IP Protocol Family

- IP (Internet protocol, Layer 3) :
  - Provides *basic naming scheme* and **unreliable** *delivery capability* of packets (datagrams) from host-to-host
- UDP (User Datagram Protocol, Layer 4)
  - Uses IP to provide unreliable datagram delivery from *process-to-process*
- TCP (Transmission Control Protocol, Layer 4)
  - Uses IP to provide *reliable* byte streams from process-to-process over connections

Note that IP is the base protocol for both UDP and TCP.

UDP and TCP

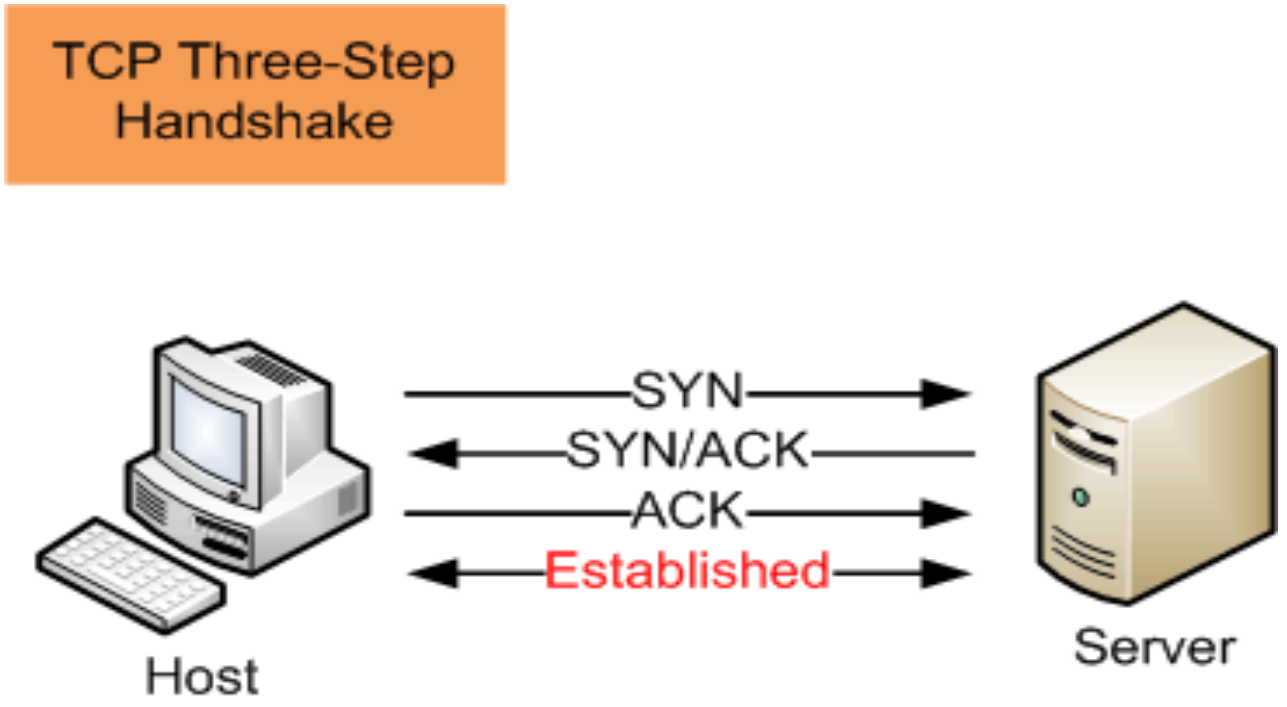
- UDP: User Datagram Protocol
  - Sent packets may be dropped, reordered, or duplicated
  - These issues are exposed to the application
- TCP: Transmission Control Protocol
  - Provides illusion of a reliable “pipe” between two processes on two different machines
  - Masks lost & reordered packets so applications don’t have to worry
  - Handles congestion and flow control
- UDP and TCP are the most popular protocols on IP
  - Both use 16-bit *port* number as well as 32-bit IP address
  - Applications *bind* a port and receive traffic on that port

Unreliable Components

- Network does not deliver packets reliably
- Several types of error can affect packet delivery
  - Bit errors (electrical interference, cosmic rays…)
  - Packet loss
  - Link and node failure
- Properly delivered frames may be delayed, reordered, or duplicated

Enter TCP…

- Guaranteed delivery
- Session-oriented



Enter TCP…: Plain Text

- Guaranteed delivery
- Session-oriented
- TCP 3-step handshake sets up the session
  - Host sends SYN to server
  - Server replies SYN/ACK
  - Host replies ACK
- When the handshake is complete, a connection is established

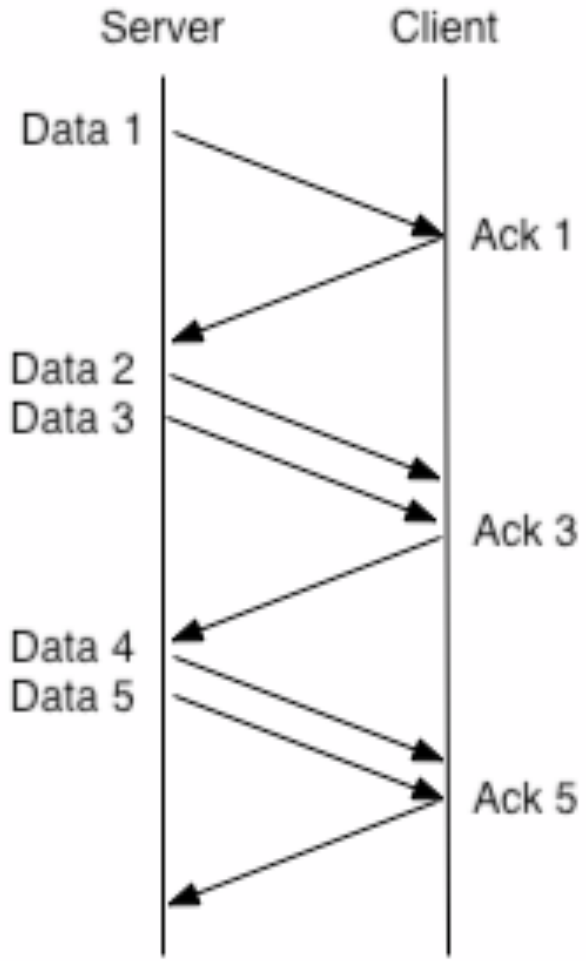
Getting Reliability from Unreliable Components

- Solution 1: use acks
  - Sender sends a message and sets a timer
  - Receiver receives a message and sends an *ack* (acknowledgement)
  - Sender receives ack and clears timer
  - (If timer goes off, start again)
- Problems
  - Bandwidth is 1 packet per round trip time (RTT)

Getting Reliability from Unreliable Components

- Solution 2: Pipeline Solution 1
  - Multiple packets in flight
  - Resend un-ack-ed packets after timeout
- Optimizations
  - Cumulative acks
    - ack i acks packets 1 to i
    - Immediate resend on nack
  - Delayed acks
    - For bidirectional communication, use application response as implicit ack

TCP Example



TCP Example: Text Description

- Sever sends data segment 1 to client
- Client responds with ack 1
- Server sends data segments 2 and 3 to client
- Client responds with ack 3
- Server sends data segments 4 and 5 to client
- Client responds with ack 5

Uses of TCP

- Most applications use TCP
  - Easier interface (gives you reliability!)
  - Automatically avoids congestion

TCP and the OS

OS provides reliability for TCP. Thus, it must:

- Track unacknowledged data
  - Keep a copy, keep timer
  - Receiver re-orders out of order segments
  - Where does it keep all this information?
    - The PCB… *Protocol* Control Block!
- Must ACK received segments very quickly
- Must decide when to wake processes on receipt
  - Is there more data coming? Should it wait?
- Must decide when to send data
  - If send is small, should it wait for more?

Some more about TCP

Implementation Details:

- Buffers at sender, receiver, and router
- Data is acked at the receiver as it is read from the buffer by the application
  - Until that point, packets could still be dropped

Goals:

- Want to save network from congestive collapse
  - Can be indicated by packet loss
- Want multiple outstanding packets at a time

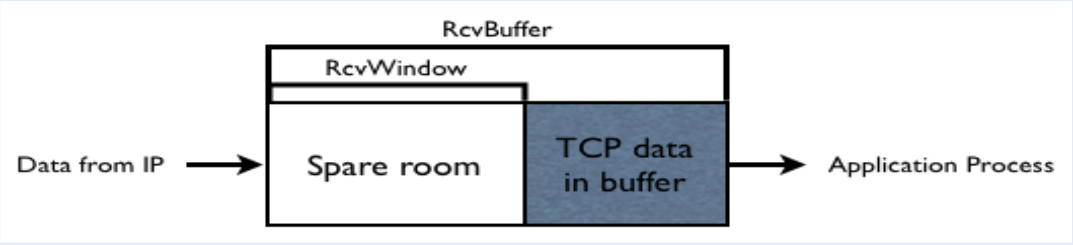
TCP Window

- Maximum number of bytes that can be sent without acknowledgement (ACK)
  - Prevents sender from overflowing receiver's buffer
- Dynamically sized to the network environment
  - Increases in size until loss occurs
- Implications on performance
  - Larger windows suited to high-latency networks
  - Smaller windows suited to high-loss networks

TCP Flow Control

- Sender maintains
  - RcvWindow: estimate of buffer space at receiver
  - Estimated using LastByteSent, LastByteAcked
- Receiver maintains
  - RcvBuffer
  - Size of data in buffer determined by LastByteRead, LastByteRcvd
- To avoid overflowing
  - LastByteRcvd-LastByteRead ≤ RcvBuffer
- Hence, sender ensures
  - LastByteSent - LastByteAcked ≤ RcvWindow

    where RcvWindow = RcvBuffer - (LastByteRcvd - LastByteRead)

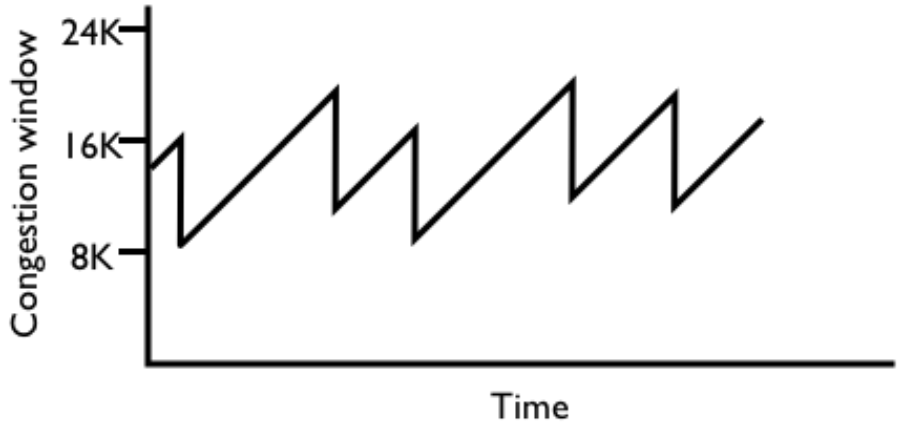


TCP Congestion Control

- Both sides of a connection keep track of congestion window (CongWin)
  - LastByteRcvd-LastByteRead ≤ min(CongWin, RcvBuffer)
    - send rate: CongWin/RTT (assuming negligible retransmission and loss)
- If acks are received regularly, CongWin grows
- If ack loss is detected, CongWin shrinks
  - ack loss detected as timeout or receipt of 3 duplicate acks
- Congestion Control algorithm has five major components
  - additive increase, multiplicative decrease
  - slow start
  - reaction to timeout events
  - round trip variance estimation
  - exponential retransmit timer backoff

Additive Increase, Multiplicative Decrease

- Multiplicative decrease
  - Half CongWin after loss event
  - Until it reaches 1 Maximum Segment Size (MSS)
- Additive increase
  - Increase CongWin by 1 MSS every roundtrip



Additive Increase, Multiplicative Decrease: Plain Text

- Multiplicative decrease
  - Half CongWin after loss event
  - Until it reaches 1 Maximum Segment Size (MSS)
- Additive increase
  - Increase CongWin by 1 MSS every round trip
- Graph with Congestion Window on y axis and Time on the x axis
  - Congestion window goes up with a gradual positive slope because of the additive increase
  - Congestion window goes down in a vertical line, because of the multiplicative decrease

Slow Start

- At start, CongWin set to 1 MSS
- Too slow to grow linearly
  - During slow start phase, exponential growth...
    - CongWin grows by 1 MSS for each received ack
    - Results in doubling every RTT
  - Until first loss event occurs

Reaction to Timeout Events

- TCP actually treats loss events differently
  - On receipt of three duplicate acks
    - Multiplicative decrease, additive increase
  - On a timeout
    - Drop congestion window to 1 MSS
    - Slow start mode until CongWin reaches a threshold (typically half of CongWin before loss)
    - Additive increase after threshold reached
- Why the difference?
  - Receipt of acks, even if duplicate, shows that some packets get to destination
    - fast recovery eliminates slow start phase (TCP Reno)

RTT Variance and Retransmit Backoff

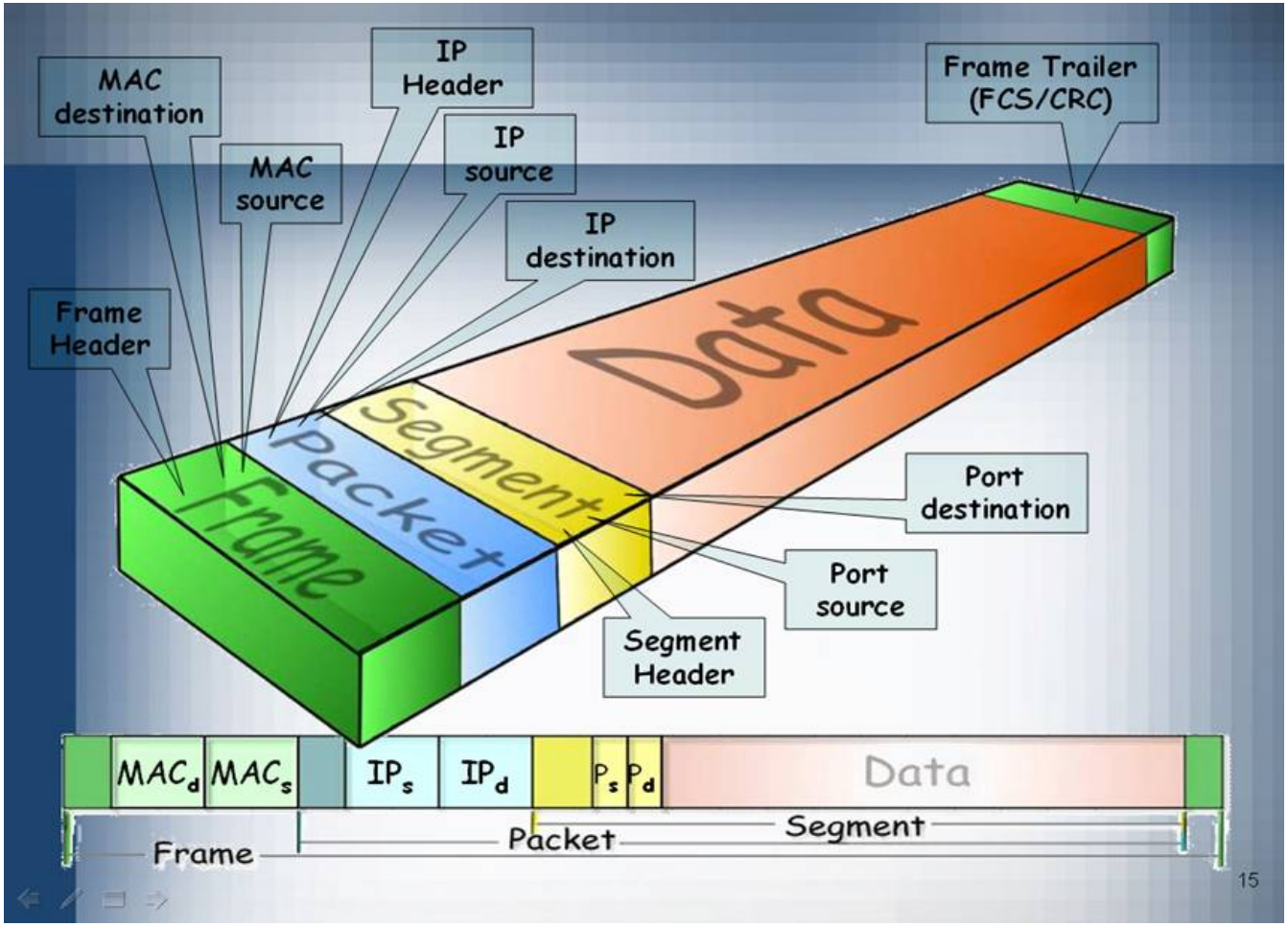
- Original TCP set timeout to twice the estimated RTT
  - But with high load (above 75%) RTT can vary by 16X
    - Lots of unnecessary sends under load (bad)
- Current versions set timeout to
  - Estimated RTT + 4 x MeanDev(RTT)
- Retransmit backoff is exponential
  - Provably needed for stability
  - This is why web browser stalls for 5 sec, then for 10 then...
    - Hint: hit reload if page no there after 5 sec

Network Costs

- Overhead: CPU time to put the data on or pull it off the wire
- Latency: time for one byte to go from one place to another
  - Latency from New York to San Francisco:
    - 3000 miles * 1 sec/186,282 miles= 15 ms
- Throughput: Maximum bytes per second (bandwidth)

- *Note: Bandwidth is not the whole story!*
- Key to good performance:
  - LAN: minimize overhead
  - WAN: keep the pipeline full!

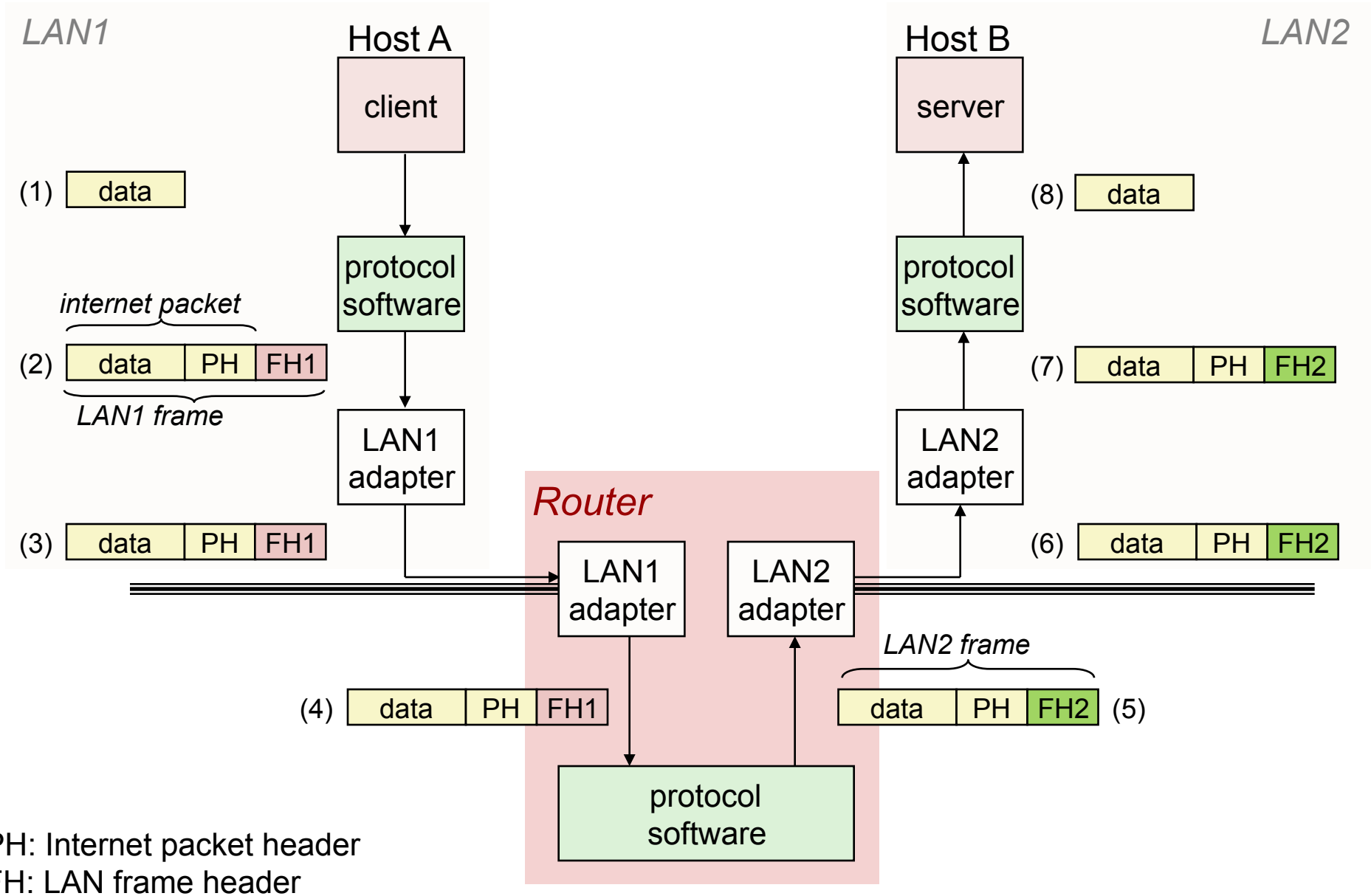
Packets, Again



Packets, Again: Text Description

- NOTE: in order from beginning to end of packet
- Frame header
  - MAC destination
  - MAC source
- Packet
  - IP header
  - IP source
  - IP destination
- Segment
  - segment header
  - port source
  - port destination
- Data
- Frame trailer - FCS/CRC

Transferring Data Over an internet



PH: Internet packet header
FH: LAN frame header

Transferring Data Over an internet: Text Description

- Example has 2 LANs connected by a router
- Terminology:
  - PH: Internet packet header
  - FH: LAN frame header
- Steps to transfer:

  1: Data is sent from Host A's client application to protocol software

  2. The PH and FH1 are appended to the data and sent along to the LAN1 adapter

  3. LAN1 adapter sends Data + PH + FH1 along to the router's LAN1 adapter

  4. Data + PH + FH1 sent to router's protocol software, which strips off FH1 and creates FH2

  5. Data + PH + FH2 sent to router's LAN2 adapter

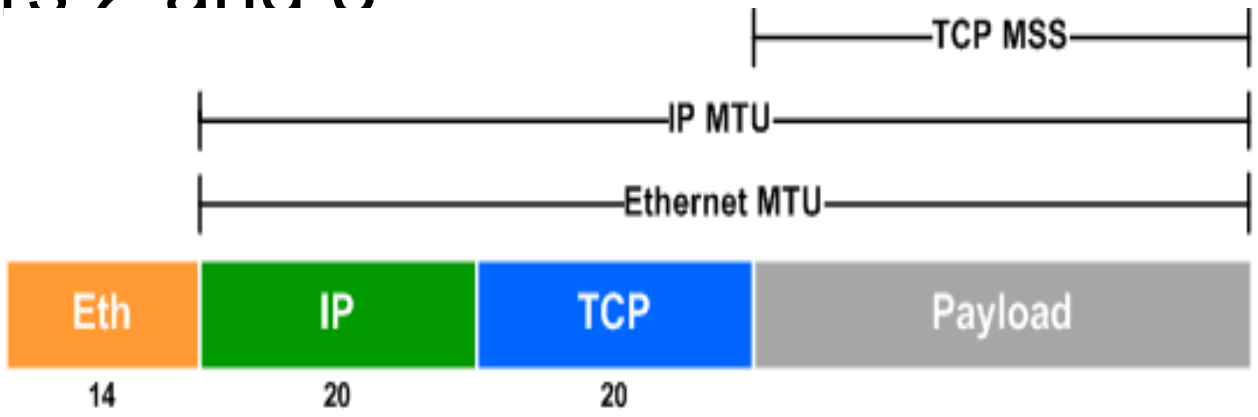
  6. Data + PH + FH2 sent to LAN2 adapter in LAN2

  7. Data + PH + FH2 sent along to protocol software where the PH and FH2 are stripped off

  8. Data goes along to Host B's server application

MTU vs. MSS

- MSS is Maximum Segment Size, a Layer 4 concept
- Often Confused with Maximum Transmission Unit (MTU), a concept of Layers 2 and 3



MTU vs. MSS: Plain Text

- MSS is Maximum Segment Size, a Layer 4 concept
- Often confused with maximum transmission unit (MTU), a concept of Layers 2 and 3
- Parts of a frame:
  - Ethernet MTU: IP, TCP and Payload
  - IP MTU: IP, TCP and Payload
  - TCP MSS: Payload

iClicker Question

The ack in the TCP protocol implies:

A. Message received

B. Segment arrived safely

C. Transmission is complete

D. Nothing

Summary

- Data sent into the network is chopped into *segments*, the network's basic transmission unit
- Frames are sent through the network
- (But we call everything packets)
- Computers at the switching points control the packet flow
- Shared resources can lead to contention
- TCP/IP is careful to avoid bringing down the network through congestive failure
- Unless otherwise stated, assume a reliable end-to-end message delivery
- Networks make tradeoffs between speed, reliability, and expense


I TCP/IP
but mostly IP

Announcements

- Exams are mostly graded
  - Will be returned in discussion section on Friday
- Project 3 due Friday, 4/17
- Project 4 out Friday
  - Will discuss in discussion section NEXT week
- Homework 9 is posted, due Friday 8:45a