Why is networking so complex?
1) Variety of hardware, 2) Variety of software, 3) Variety of protocols/standards, 4) Terminology can be confusing, 5) Wireless / mobility issues add complexity

Network: system for connecting computers using a single transmission technology

Internet: set of networks connected by routers that are configured to communicate among a variety of network transmission tech. / network of networks.

RFC (Request for Comments): Group started repo for researcher comments

Internet Engineering Task Force (IETF): produces docs for internet standards

protocols define: 1) format & order of messages sent & received among network entities. 2) actions taken on message transmission & receipt.

TCP Service:
1) TCP - Transmission Control Protocol, connection-oriented service. 2) handshake: prepare for transfer. Set up "state" in two communicating hosts. 4) reliable, in-order, byte-stream data transfer. 5) flow control: sender won't overwhelm receiver. 6) congestion control: senders slow down sending rate when network is congested

UDP service:
1) UDP – User Data Protocol. connectionless service. 2) light-weight, fast. 3) no handshake. 4) unreliable (best effort) data transfer. 5) no flow control. 6) no congestion control

Applications that use TCP:
o HTTP (Web) / o FTP (file transfer) / o Telnet / o SMTP (email)

Applications that use UDP:
o streaming media / o Teleconferencing / o DNS (Domain Name Service) / o VOIP Internet telephony

circuit switching: dedicated circuit per call: telephone net / packet-switching: data sent through net in discrete "chunks" (packets)

TDM divides BW into blocks of time, during which only 1 user is permitted to transmit / FDM divides BW by frequency, each host

Statistical Multiplexing: allows info from a number of channels to be combined for transmission over a single channel.

End-To-End Delay / Nodal Delay:
Total time from initiating "send" (from source) to completed "receive" (at destination)

Four sources of packet delay
o nodal processing (process packet headers, check bit errors; determine output link) / o queueing delay (time waiting at output link for transmission) / o Transmission delay (L/R) (time outputting info the transmission medium) / o Propagation delay (d/s) (time traveling over physical medium)

Suppose there are 69 packets entering a queue at the same time. Each packet is of size 7 MiB. The link transmission rate is 1.7 Gbps. What is the queueing delay of packet number 3 (in ms). / N = 69
L = 7 MiB = 58720256 bits
R = 1.7 Gbps = 1700000000 bps
2L/R = 2 * 58720256 / 1700000000 = 0.0691 s = 69.1 ms
avg_queuing_delay = L(N-1)/2R = 58720256 (69-1) / (2*1700000000) = 3992977408 / 3400000000 = 1.17440512 s = 1174.40512 ms (this is extra)

What is the total utilization of a circuit-switched network, accommodating 10 users with equal bandwidth, with the following users:
• Four users are utilizing 100% their bandwidth.
• Two users are utilizing 60% of their bandwidth.
• Four users are inactive
4 * (1.00) * 0.10 + 2 * (0.60) * 0.10 + 4 * (0.00) * 0.10 = 0.52 = 52%

How long does it take to send a 8 MiB file from Host A to Host B over a circuit-switched network, assuming:
Total link transmission rate = 47.1 Gbps. Network is FDM, with 7 permitted users, each with an equal bandwidth share. A link connection requires a setup time of 54.8 ms.
L = 8 MiB = 8 * 1024 * 1024 * 8 = 67108864 bits
R = 47.1 Gbps = 4.71 * 10^10 = 47100000000 bps
R/user = 47100000000/7 users = 6728571428.57 bps/user
linksetup = 54.8
d_trans = L/R = 67108864 / 6728571428.57 = 9.9737 ms
total_send_time = d_trans + link_setup = 9.97 + 54.8 = 64.77 ms

Suppose there are 3 routers in sequence between Host A and Host B, all of which use store-and-forward routing. What is the total end-to-end delay for a packet originating from Host A with destination Host B, under the following conditions.
Each of the link transmission rates are 8.4 Mbps. The total distance from Host A to Host B along its path of transmission is 160 km. The speed of propagation through the transmission medium is is 2.7 x 10 m/s. The packet size is 3 KiB. / A--1--2--3--B
R = 8.4 Mbps = 8400000
L = 3 KiB = 3072 B = 24576 bits
d = 160 km = 160000 m
s = 2.7 x 10^8 m/s = 270000000 m/s
d_trans = L/R = 24576 bits/8400000 bps = 0.0029257 s = 2.9257 ms
d_prop = d/s = 160000 m / 270000000 m/s = 0.000592592 s = 0.592593 ms
nodal delay = (3+1 transmissions) * 2.9257 + 0.592593 = 12.2954497 = 12.3 ms

For the following question, proper hexadecimal format is (0xYYYY) where Y will range in (0-9) or (A-F). Only proper formats will be accepted. Suppose that we send a DNS request with ID #12162. | What is the little-endian represent (hex)? 0x822F | What is the big-endian represent (hex)? 0x2F82 | Which represent is required for network communication?
(Enter "1" or "2" w/o quotes) 2

d_nodal = d_proc + d_queue + d_trans + d_prop

A client in a network with a proxy server requests a 7 MiB file from an internet server, fakeservername.com. The network's proxy server has a 2.11 Mbps connection to fakeservername.com. The average response time between the network's proxy server and the internet origin server (including RTT) is 1.5 seconds for a small "header-only" HTTP request/response. The file requested by the client is currently in the proxy server cache, but the proxy server relays the client's request to the internet server with "if-modified since". Assume that transmissions between the proxy and the origin servers are stream (not packets) at full bandwidth, with negligible propagation delay. How much time is saved if the file has not been modified (in ms)?
L = 7 MiB = 7 * 1024 * 1024 * 8 = 58720256 bits
R = 2.11 Mbps = 2110000 bps | t = 1.5 s
L/R = 58720256 / 2110000 = 27.8295 s

A client's browser sends an HTTP request to a website. The website responds with a handshake and sets up a TCP connection. The connection setup takes 2 sec, including the RTT. The browser then sends the request for the website's index file. The index file references 4 additional images, which are to be requested/downloaded by the client's browser. How many requests (including the initial request) must be sent by the browser…
1. With non-persistent HTTP? (2n+2) = 2*4+2=10 requests
2. With persistent HTTP? (n+2)=4+2=6 requests

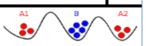
00-conn_init	00-conn_init
00-conn_accept	00-conn_ack
01-req conn	01-req conn
02-req index	02-req index
03-req conn	03-req img1
04-request img1	04-req img2
05-req conn	05-req img3
06-request img2	06-req img4
07-req conn	How much longer does non-persistent HTTP take than persistent HTTP?
08-request img3	
09-req conn	
10-request img4	(10-6)req * 2sec/req = 8 sec

Email Example:
1) Alice uses her user agent to compose message and send to bob@someschool.edu 2) Alice's user agent 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 uses his user agent to read message

POP3	IMAP
1) uses "download and delete" mode. 2) Bob cannot re-read e-mail if he changes client 3) "Download-and-keep": copies of messages on different clients 4) POP3 is stateless across sessions	1) Keep all messages in one place: the server 2) Allows user to organize messages in folders 3) IMAP keeps user state across sessions

What is reliable data transfer?: The ability to guarantee that a transmitted message was received, and it was received as it was sent.

Two-Generals Problem: Positions of the armies.



Armies A1 and A2 need to communicate but their messengers may be captured by army B.
1) Cannot guarantee that message will be received
2) Cannot guarantee that received message has no errors

What about RDT is indicated by the two-generals problem? If there is any aspect of the communications channel, it is impossible to guarantee 100% reliable data transfer.

What are some aspects of reliable data transfer? Error Detection, Receipt Acknowledgement, Timing, Fairness, Message Sequencing, Usage Fairness,

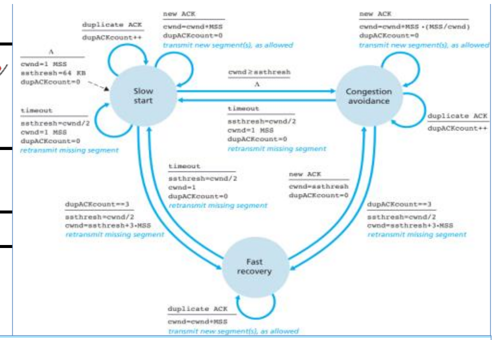
What is the maximum number of bytes that can be carried in the "application data" section of a UDP segment? 65,527 bytes. The "length" field of the UDP header is 16-bits, so the largest value it can hold is 65,535. The header is 8 bytes, which leaves 65,527 bytes for the "applicationdata". It would not be practical to send a segment of this size through the internet.

Reliable Data Transfer Steps:
1) Error detection (compute checksum for both client-server)
2) Acknowledgement
3) Sequencing
4) Timing (flow/congestion control)
5) Retransmission
6) Fairness

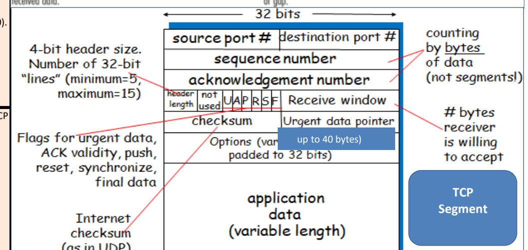
Compute the sum with carry-wraparound (sometimes called the one's complement sum) of the following two numbers. Give answer in 8-bit binary, zero-padded to 8 bits if necessary, with no spaces (e.g. 00101000). Please note this is different than the checksum calculation.
```
  10000011
  10000000
=100000011 = 00000100 (8-bit sum)
```

Assume a TCP sender is continuously sending 1,250-byte segment. If a TCP receiver advertises a window size of 7,251 bytes, and with a link transmission rate 45 Mbps an end-to-end propagation delay of 39.3 ms, what is the utilization? Assume no errors, no processing or queueing delay, and ACKs transmit instantly. Also assume the sender will not transmit a non-full segment. Give answer in percent (1 dec)
NumSegments = 7251 / 1250 = 5.8008 segments = 5 segments
L = 1250 bytes = 1250 * 8 = 10000 bits
R = 45 Mbps = 45000000 bps = 45000 bpms (bits per ms)
t_prop = 39.3 ms
t_trans = L/R = 10000 bits / 45000 bpms = 0.2222 ms
RTT = t_prop * 2 = 39.3 * 2 = 78.6 ms
delay per packet = RTT + t_trans = 78.6 + 0.2222 = 78.8222 ms
utilization = (segments * t_trans)/(delay per packet) = 5 * 0.2222 / 78.8222 = 0.0141 * 100 = 1.4096 % = 1.4 %

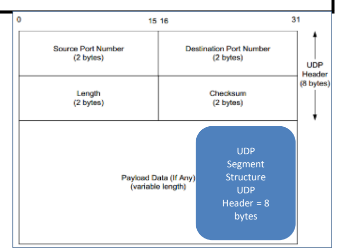
What are some tradeoffs for the implementation of RDT in TCP
1) Additional overhead. 2) Possible delays due to waiting for acknowledgements. 3) More packets on the network (because of ACKs) – leading to congestion issues



Event	TCP Receiver Action
Arrival of in-order segment with expected sequence number. All data up to expected sequence number already acknowledged.	Delayed ACK. Wait up to 500 msec for arrival of another in-order segment. If next in-order segment does not arrive in this interval, send an ACK.
Arrival of in-order segment with expected sequence number. One other in-order segment waiting for ACK transmission.	Immediately send single cumulative ACK, ACKing both in-order segments.
Arrival of out-of-order segment with higher-than-expected sequence number. Gap detected.	Immediately send duplicate ACK, indicating sequence number of next expected byte (which is the lower end of the gap).
Arrival of segment that partially or completely fills in gap in received data.	Immediately send ACK, provided that segment starts at the lower end of gap.



TCP Retransmission Scenarios: 1) segment processing error, 2) segment loss in transmission, 3) ACK lost in transmission, 4) ACK delayed



What are three important functions of a packet-switched network?
1) Packet Construction, 2) Transmission, 3) Interpretation

avg_queuing_delay for packet N = (N-1)L/R
avg_queuing_delay (for N packets) = L(N-1)/2R

IP Protocol Stack/OSI model:
7) Application (supporting network applications; FTP, SMTP, HTTP)
6) Presentation (applications interpret meaning of data; encryption, compression)
5) Session (synchronization, checkpointing, recovery of data exchange)
4) Transport (process-process data transfer; TCP, UDP)
3) Network (routing datagrams from src to dst; IP, routing protocols)
2) Link (data transfer between neighboring network elements; PPP, Ethernet)
1) Physical (carries signals over medium; cable,

Application Layer Responsibilities:
1) Determine destination IP address
2) Support network applications
3) Decide which data which will transit the internet

Types of security threats: 1) malware, spyware / 2) denial of service / 3) packet sniffing / 4) address spoofing

Application Architectures: 1) Client-server (eg. Web surfing), 2) Peer-to-peer (P2P) (eg. Bittorrent), 3) Hybrid of client-server and

Application layer protocol defines: 1) Types of messages exchanged, 2) Message syntax, 3) Message semantics, 4) Rules for when and how processes send & respond to msgs

What are some reasons for the layering of network protocols?
1) Protocols can be tested independently of one another, 2) When maintenance is required at one level, changes do not affect other layers, 3) Can update a the inner workings of a protocol, as long as input/output remain the same, 4) The complication of dealing with the intermediated types of hosts and data is lessened

TCP services: 1) connection-oriented: setup required between client and server processes, 2) reliable, in-order transport bet/ client-server, 3) flow control: sender won't overwhelm receiver, 4) congestion control: throttle sender when network overloaded, 5) does not provide: timing, minimum bandwidth guarantees

[VOIP] Given the attached image, and:
- Host A converts analog to digital at a = 38 Kbps | - Link transmission rate R = 1.4 Mbps | - Host A groups data into packets of length L = 79 bytes | - Distance to travel d = 843.2 km | - Propagation speed s = 2.5 x 10 m/s | - Host A sends each packet to Host B as soon as it gathers a whole packet. | - Host B converts back from digital to analog as soon as it receives a whole packet. | How much time elapses from when the first bit starts to be created until the conversion back to analog begins?
analog-digital = 38 Kbps = 38000 bps
R = 1.4 Mbps = 1400000 bps
L = 79 B = 632 bits
d = 843.2 km = 843200 m
s = 2.5 * 10^8 = 250000000 m/s
store-and-forward
d_convert = L/R = 632 / 38000 = 16.6316 ms
d_trans = L/R = 632 / 1400000 = 0.4514 ms
d_prop = d/s = 843200 / 250000000 = 3.3728 ms
d_nodal = d_convert + d_trans + d_prop = 16.6316 + 0.4514 + 3.3728 = 20.4558 ms = 20.46 ms

Given a link with a maximum transmission rate of 82.6 Mbps. Only two computers, X and Y, wish to transmit starting at time t = 0 seconds. Computer X sends fileX (17 MiB) and computer Y sends fileY (184 KiB), both starting at time t = 0.
- Statistical multiplexing is used, with details as follows:
Packet Payload Size = 1000 Bytes / Packet Header Size = 24 Bytes (overhead)
- Ignore Processing and Queueing delays
Assume partial packets (packets consisting of less than 1000 Bytes of data) are padded so that they are the same size as full packets.
At what time (t = ?) would FileX finish transmitting?
Lx = 17 MiB = 17 * 1024 * 1024 * 8 = 142606336 bits
Ly = 184 KiB = 184 * 1024 * 8 = 1507328 bits
R = 82.6 Mbps = 82600000 bps
NumPacketsX = 142606336 / 8000 = 17825.792 = 17826
NumPacketsY = 1507328 / 8000 = 188.416 = 189
NumPacketsWhenYFinishes = 189 * 2 = 378
TotalNumPackets = 17826 + 189 = 17826 + 189 = 18015
numPacketsWhenFileYFinishes = 189 * 2 = 378
Total Packet Size = Payload + Header = 1024 B = 8192 bits/packet
Tx = TotalPacketSize * TotalNumPackets / R = (8192 * (17826 + 189)) / 82600000 bps = 1.79 s
Ty = TotalPacketSize * numPacketsWhenFileYFinishes / R = (8192 * (189 * 2)) / 82600000 bps = 0.03749 s = 37.5 ms

Given a transmission medium with propagation rate of 2.5x10 m/sec. Multiple computers share this link using packet switching.
- Assume that queuing is necessary only for transmission.
- Assume that there is no other traffic on the medium, no switches between the link and the destination.
- At least 20 packets arrive in the queue at time t = 0.
Each packet's size is 1000 bytes.
- The distance to the destination is 1500 km.
Answer the following questions for each of the given transmission rates (in ms).

What is the transmission delay for each packet?
What is the propagation delay for each bit?
What is the queueing delay for the 4th packet?
What is the end-to-end delay for the 5th packet?
What is the avg queueing delay for the first 10 packets?

p = 20 packets
L = 1000 bytes * 8 = 8000 bits
R1 = 10 Mbps = 10000000 bps
R2 = 100 Mbps = 100000000 bps
d = 1500 km = 1500000 m/s
s = 2.5x10^8 m/s = 250000000 m/s
10 Mbps:
T_trans = L/R = 0.0008 s = 0.8 ms
T_prop = d/s = 0.006 s = 6 ms
T_queue4 = (N-1)L/R = 3 * L / R = 3 * 8000 / 10000000 = 0.0024 s = 2.4 ms
T_e2e5 = L/R + d/s + (5-1)L/R = 8000 / 10000000 + 1500000 / 250000000 + 4 * 8000 / 10000000 = 0.0008 + 0.006 + 0.0032 = 0.01 s = 10 ms
T_avg10 = L(N-1) / (2R) = 8000 * (10-1) / (2 * 10000000) = 0.0036 s = 3.6 ms
100 Mbps:
T_trans = L/R = 0.00008 s = 0.08 ms
T_prop = d/s = 0.006 s = 6 ms
T_queue4 = (N-1)L/R = 3 * L / R = 3 * 8000 / 100000000 = 0.00024 s = 0.24 ms
T_e2e5 = L/R + d/s + (5-1)L/R = 8000 / 100000000 + 1500000 / 250000000 + 4 * 8000 / 100000000 = 0.00008 + 0.006 + 0.00032 = 0.0064 s = 6.4 ms
T_avg10 = L(N-1) / (2R) = 8000 * (10-1) / (2 * 100000000) = 0.00036 s = 0.36 ms

With a stop-and-wait implementation, how long would it take to send five 1500-byte packets, subject to the following conditions:
• R = 20Mbps | • ACK packet L = 20 byte | • Network end-to-end time (after initial transmission) = 20 ms
Initial Transmit = L/R = (1500 * 8) bits / 20,000,000 bps = 0.0006 s = 0.6 ms
Initial Transmit Propagation delay = 20 ms
ACK transmit = (20 header bytes * 8) bytes / 20,000,000 bps = 0.008 ms
ACK Propagation delay = 20ms
First packet total = 0.6 + 20 + 0.008 + 20 = 40.608ms (Just the first packet)
Send 5 total packets = 40.608ms * 5 = 203.04ms
What is the utilization? Time spent transmitting = numPackets * (INITtrans + ACKtrans) = 5 * (0.6 + 0.008) = 3.04ms
Total time until finished = 203.04ms (from above)
utilization = total_t_trans / total_t_nodal = 3.04ms / 203.04ms = 1.5 %

HostA is sending fixed-size packets to remote HostB on a 100 Mbps link. Each packet's length is 1200 bytes. The propagation delay is 1 ms.; the processing/queuing delays are negligible. Given a sliding window of 12,000 bytes, and assuming no pipeline errors, what is HostA's utilization?
t_trans = L/R = (1200 x 8 bits) / 108 bps = 0.000096 sec = 0.096 ms/packet
t_trans for 10 packets = 0.096 * 10 = 0.96 ms
RTT = 1 * 2 = 2 ms. (i.e., propagation delay x 2)
Total time before we can send packet # 11 = 2.096 ms.
Utilization (Percentage of time actually transmitting) = 0.96 / 2.096 = 45.8%

What are the minimum and maximum sizes (in bytes) of a TCP header?
20 bytes min(five 4-byte rows). 60 bytes max ("header length" field of TCP header is 4-bits, largest value it can hold is 15. Number of 4-byte "lines" in the header (4 * 15 = 60).

For TCP, please select whether the service is implemented, and indicate how it is implemented.
Connection-Oriented? Yes, implemented by a 3-way handshake at start of connection.
Full Duplex Service? Yes, for example Telnet, which echoes back every character.
In-order delivery? Yes, via sequence numbers in TCP header information
Pipelining? Yes, though pipeline is limited by congestion and flow control
Flow Control? Yes, receiver advertises a window size, and sender limits output accordingly.
Congestion Control? Yes, implemented by restricting pipeline size based on receipt of acknowledgements (or lake thereof)
Bandwidth Guarantee? No – no bandwidth guarantees.
Reliable Delivery(mostly)? Yes – Acknowledgement for delivery of packets
Jitter Threshold? No – no jitter threshold

For UDP, please select whether the service is implemented, and indicate how it is implemented.
Connection-Oriented? No - Connectionless
Full Duplex Service? No – One way service
In-order delivery? No – Packets may be delivered out of order
Pipelining? No pipeline, BUT it is fire and forget (so you can transmit all you want)
Flow Control? No – Sender may overwhelm receiver
Congestion Control? No – Ignores possibility of congestion in the internet.
Bandwidth Guarantee? No – no bandwidth guarantees.
Reliable delivery(mostly)? No – "Fire and Forget" service ("Send it and FORGET IT!")
Jitter Threshold? No – no jitter threshold

The transport layer resides on/in the network edge and manages communications from process to process

The network layer resides on/in the network edge/core and manages communications from host to host.

What organization manages the .org TLD?
PIR=Public Interest Registry

What is the DNS application-layer protocol used for? DNS resolves aliased, text web addrs to IP addresses. What transport-layer protocol does it make use of? DNS uses UDP.

For de-multiplexing, how is a TCP socket identified?
• Source IP Address, Source Port Number
• Destination IP Address, Destination Port Number
For de-multiplexing, how is a UDP socket identified?
• Destination IP Address, Destination Port Number

HTTP is "stateless" (server maintains no information about past client requests)

Setup UDP socket = SOCK_DGRAM
Setup TCP socket = SOCK_STREAM
socket-bind-listen-accept-recv/send

Four Cookie components for Websites:
1) cookie header line of HTTP response message | 2) cookie header line in HTTP request message | 3) cookie file kept on user's host, managed by user's browser | 4) back-end database at Web site

Why use UDP?
1) no connection establishment (which can add delay)
2) simple: no connection state at sender, receiver
3) small header size
4) no congestion control: UDP can blast away as fast as

The TCP/UDP protocol uses a 16-bit, 1's complement of the 1's complement sum to compute the checksum

endpoint = socket = IP:PORT

To get the 1's complement sum of two binary numbers:
1. Add them
2. Wrap any overflow bit around and add it to the LSB (least significant bit)
3. You're done right here if we ask for the 1's complement sum (summary exercises)
4. To get a checksum to store in your TCP or UDP header, you would take the 1's complement of your answer in #3. That just means you flip all the bits.
5. Then on the receiving end, you would again calculate the 1's complement sum and add that to the checksum. The result will be all 1's if the data is ok (with a tiny possibility for error).
6. Modern CPU's can calculate 1's complement sums and 1's complements for 64 bit numbers in one operation. Which is why these checksums are used.
7. Why are you still reading this? You got the answer back at #3!
8. So most of the confusion comes from the definition of the checksum, which technically is the "1's complement of the 1's complement sum". Total word salad!
9. But you don't care about that! The answer was way back at #3 ok? Just do steps 1 and 2.

Port	Name	Description
7	echo	Echo input back to sender
11	systat	System statistics
13	daytime	Time of day (ASCII)
17	quote	Quote of the day
20,21	ftp	File Transfer Protocol
22	ssh	Secure Shell remote login
23	telnet	Telnet
37	time	System time (seconds since 1970)
53	domain	Domain Name Service (DNS)
80	web, http	World Wide Web (HTTP)
123	ntp	Network Time Protocol (NTP)
161	snmp	Simple Network Management Protocol (SNMP)

URL=http://www.oregon.gov/SiteCollectionImages/branding/portal/bigfoot.png
Host Name=.gov | Path Name=/SiteCollectionImages/branding/portal/bigfoot.png
TLD (Top-Level Domain)=.gov

transport layer: logical communication between process to process
network layer: logical communication between host to host

DevRTT = (1–B) • DevRTT + B•| SampleRTT – EstimatedRTT |, B=0.25