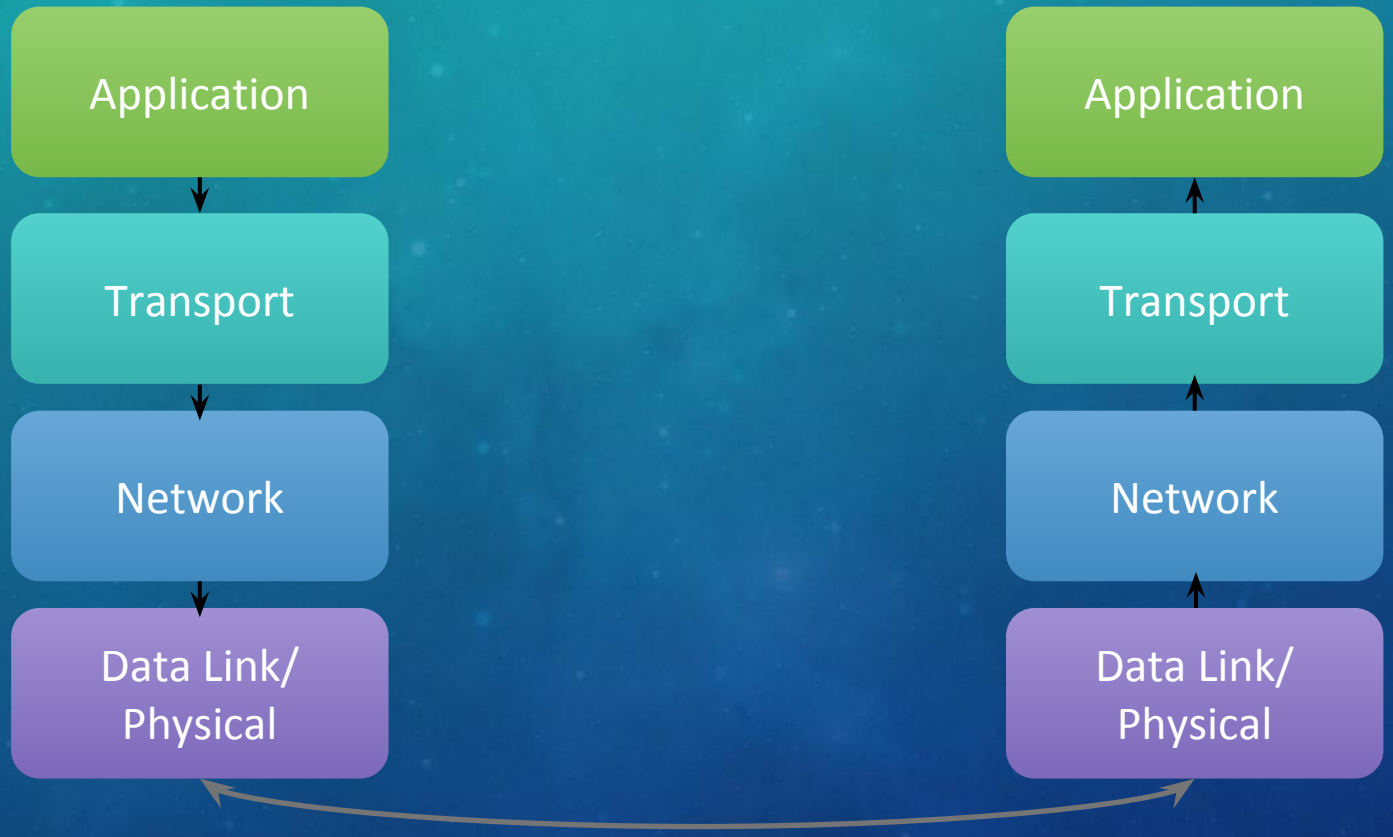
CSE 461

MIDTERM REVIEW

NETWORK LAYERS & ENCAPSULATION



APPLICATION LAYER

- Used by applications
- Protocol is arbitrary

TRANSPORT LAYER

- Involves packaging of data for transport
- UDP/TCP and ports

NETWORK LAYERS & ENCAPSULATION

- Handles issues related to routing on the network
- Data treated as packets

DATA LINK/PHYSICAL LAYERS

- Data link layer
  - Puts data onto the actual line
  - Error-correcting codes to account for line noise are in the data link layer
    - At this level, data consists of frames
- Physical layer
  - Actual electrical or wireless oscillations

ADDRESSING

- MAC addresses
- IP addresses
- Ports
- Sockets (file descriptors)

MAC ADDRESSES

- 48-bit
- Identify instance of specific network interface hardware

IP ADDRESSES

- 32-bit (in IPv4) or 128-bit (in IPv6)
- Identify a host on a network
- Can change dynamically

PORTS

- 16-bit
- Identify communication channels on a specific host
- Often map to applications

SOCKETS

- Programming interface for networking
- Most common implementation is Berkeley sockets
- Allows data to be sent with file descriptor-like structures

UDP VS. TCP

UDP	TCP
Unreliable	Reliable
Connection-less	Connection-oriented
No acknowledgements	Acknowledgements
No flow control	Sliding window
No sequence numbers	Sequence numbers

TELNET VS. FTP

Telnet	FTP
Used for sending text data; originally for remote login into a server	Used for sending files
Data transfer and control on same channel	Separate channels for control and data transfer; control channel uses Telnet
Uses TCP	Also uses TCP
Once connected, server and client essentially the same	Server and client behave very differently
Not secure; largely replaced by SSH	Not secure; somewhat replaced by SFTP

METRICS

- Bandwidth
- Latency
- Throughput, goodput
- Channel utilization
- Shannon’s theorem
- Nyquist rate

FREQUENCY & BANDWIDTH

- Frequency: rate of an oscillation
- Bandwidth: measures the width of a range of frequencies
- Bandwidth = $freq_{upper}$ - $freq_{lower}$
- Human hearing bandwidth: ~20kHz (20kHz - 20 Hz)
- “Bandwidth” and “bitrate” are often used interchangeably; this is a different definition
- **Bonus Question**: what’s the frequency range and bandwidth of 802.11 b/g?
  - 2.4 GHz to 2.5 GHz; 100 MHz

LATENCY

- Time between source and destination
- Shortest possible latency bounded by *c*
- Ping can measure round-trip latency

THROUGHPUT & GOODPUT

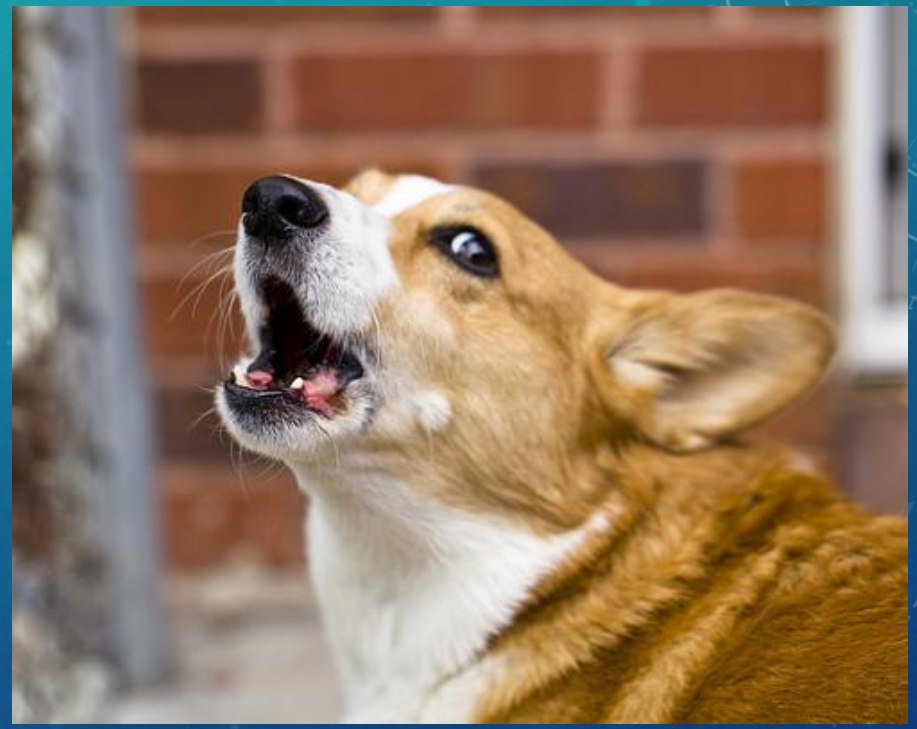
- Throughput: measures how much data can be sent in a given time period (a.k.a. bitrate)
- E.g., 100 Gbps
- Bits that you can send (i.e., put onto the wire) per amount of time
- Goodput: excludes protocol bits and retransmitted data packets
- What factors might cause goodput < throughput?
  - Protocol overhead
  - Dropped or corrupted packets
  - Flow control

CHANNEL UTILIZATION

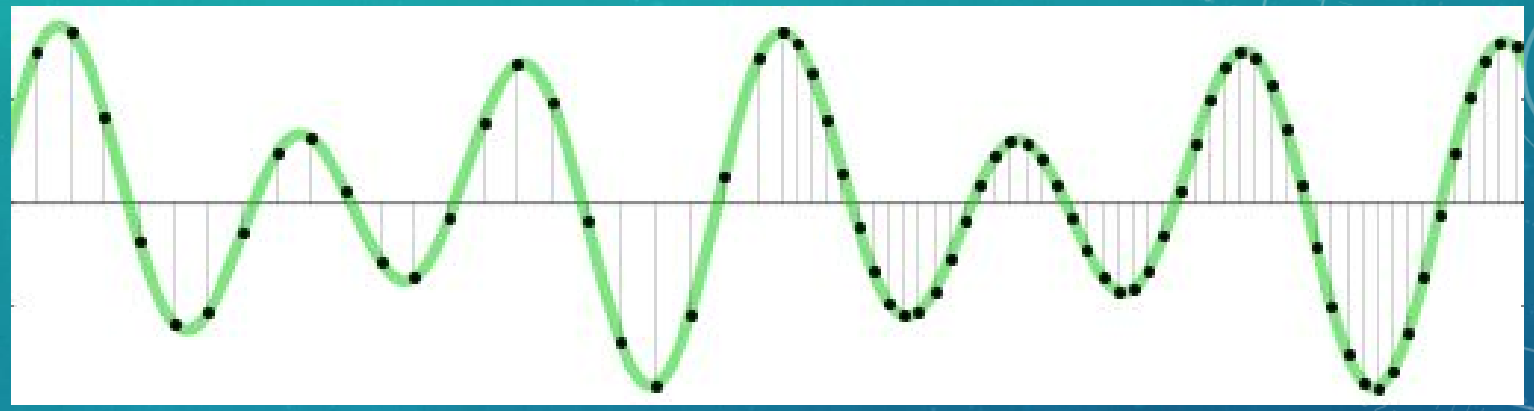
- Calculates how much of the channel is being used
- Percent of the time the channel is in use
- (sent data size) / ((channel bitrate) * (round-trip latency))

SHANNON THEOREM

- Tells about maximum bitrate in the presence of noise
- Capacity = bandwidth * $\log_2(1 + \text{signal/noise})$
- $C = B \log_2(1 + S/N)$
- What are the implications of this?



NYQUIST RATE



- To recover a waveform, the sampling rate must be at least two times the highest frequency
- Telephone sampling rate is 8kHz; what are the implications of this?
- What sampling rate would be required to recover all frequencies audible by humans? (Up to 20kHZ)
    - Audio CDs use 44.1kHz sampling rates for this reason

HTTP

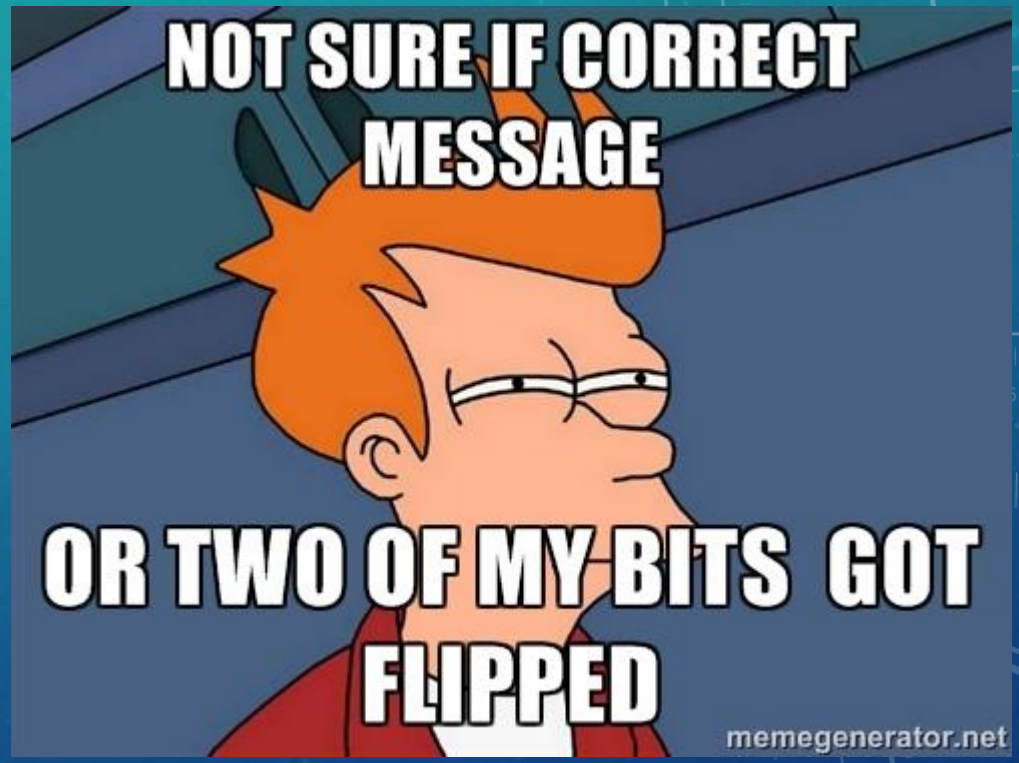
- HTTP 1.0
  - Initial connection over TCP acts as a preamble
  - Content-length can designate payload end
    - Bad for streaming
    - Put the content-length in the end of the payload ⟶ hard to cache on the receiver side
  - Caching used heavily
- HTTP 1.1
  - Data comes as a stream, chunked into defined lengths (tokenized)
  - Connections are reused, reducing overhead
  - Some pipelining possible, but limited
- HTTP 2.0
  - Reduces latency through header compression
  - Allows asynchronous sending/multiplexing over a single TCP connection
  - Fixed the head-of-line blocking problem in HTTP 1.x

ERROR HANDLING / Integrity

- Parity bits
- Checksums
- CRCs

PARITY BITS

- Bits check parity on a set of bits
- Even parity: bits add to 0
- Odd parity: bits add to 1
- Multiple parity bits (on odd bits/ on even bits, etc.) can increase effectiveness



CHECKSUMS & CRCS

- Checksums:
  - Adds all words in data as unsigned numbers, allowing to overflow
  - Sum was then compared to check data integrity
- CRCS:
  - Specific type of checksum that uses polynomial division
- Both are integrity checks using a fixed size of data

OTHER TOPICS

- DNS
  - Find the IP address for a URL
  - UDP is typically used when packet size is smaller than 512 bytes
  - Typosquatting: register a wrong address for phishing www.yaho.com
  - Denial of service: overloading DNS servers → unable to resolve URLs
  - Registrar hacking: company's name servers that supply IP info are hacked → traffic redirected
  - DNS hijacking: DNS servers maliciously return incorrect translations
  - DNS cache poisoning: caches are supposed to expire. Altering the cache data returns incorrect translations

ADDITIONAL STUDY SUGGESTIONS

- Read through your project 0 & 1 code and diagram what it's doing
- Review HW problems; do similar problems
- Watch David Wetherall's Coursera course videos (link)
- Review old midterms
  - Available on CSE site (but cover somewhat different material than what ours will)

ANY QUESTIONS?