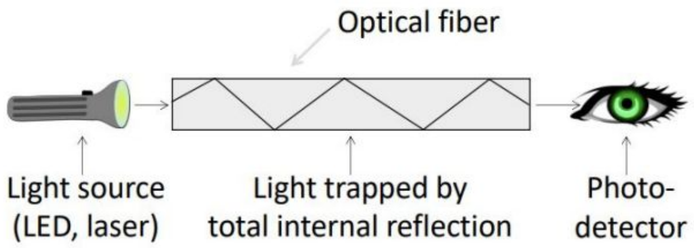
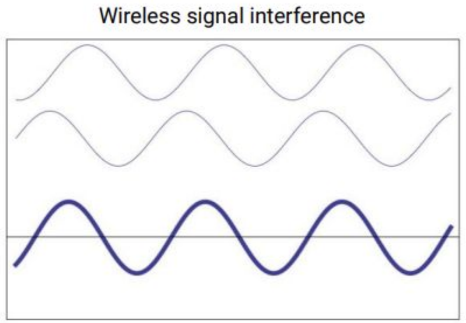
CSE 461 - Final review

Spring 2023

Physical Layer

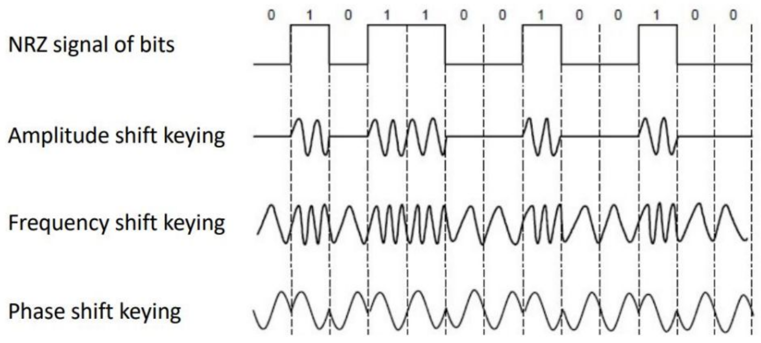
- Media types
  - Wired - twisted pair, coaxial
  - Fiber - long, thin strands of glass
  - Wireless - broadcasted signal
    - Potentially many receivers
    - Nearby signals interfere at a receiver


Optical fiber
Light source (LED, laser)
Light trapped by total internal reflection
Photo-detector



Coding and Modulation

- Physical layer - translate analog signal on wires into digital bits
- How can we send information across a link?
- Coding - directly on a wire
  - Simplest scheme - Non-Return to Zero, high voltage represents 1, low voltage represents 0
  - Better schemes account for clock recovery - 4B/5B coding, Manchester encoding, etc
- Modulation - carrying RF signals by modulating a carrier signal



Fundamental Limits

- Key Channel Properties - B: Bandwidth (hertz), S: Signal strength, N: Noise
- Nyquist Limit
  - Maximum symbol rate is 2B
  - With V signal levels, ignoring noise,
    max bit rate R = 2B log_2(V) bits/sec
- Shannon Capacity
  - Signal-to-Noise ratio (S/N) determines number of distinguishable voltage levels
  - Max carrying rate C = B log_2(1 + S/N) bits/sec
    - Max rate at transmitting without loss over a random channel
    - Increasing signal power gives diminishing returns
    - Increasing bandwidth increases capacity linearly

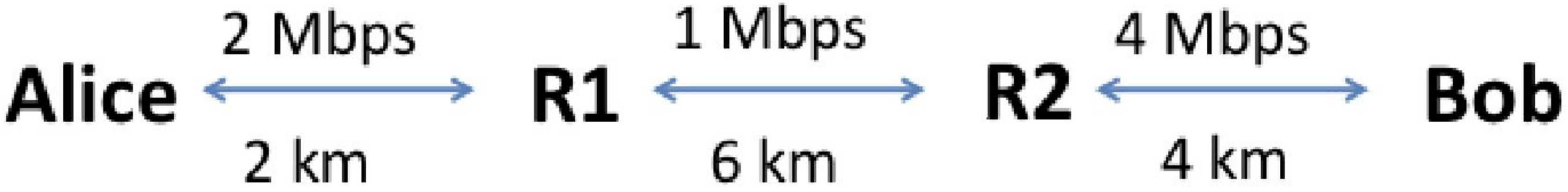
Practice Questions:

Q-1 Imagine that you are in charge of engineering the device responsible for processing a 4B/5B coding scheme and writing it onto a link with bandwidth 8Hz (Hertz). Your device can vary voltage in the range -1V (Volts) to 1V.

Q-1.1 Ignoring noise, what is the maximum bit rate your device can write to the wire if you partition the voltage range into 16 levels? Answer in bps and show your work.

Q 1.2 Now assume that the noise across the link is guaranteed to be strictly less than 0.25V. What is the maximum number of levels you can use within the the voltage range and still tolerate noise (that is, correctly detect the transmitted level at the receiver)? Show your work.

Q-2 Consider a network between Alice and Bob with link lengths and bandwidths as shown. R1 and R2 are routers in between that have negligible packet processing and queuing delay.



a) What is the net propagation delay of packets from Alice to Bob? Assume speed of signal in medium = 2x10^8 m/s

b) What is the throughput experienced by packets from Alice to Bob?

c) What is the total delay experienced by packets from Alice to Bob assuming a packet size of 10 Bytes?

d) Is the link propagation delay limited or throughput limited?

e) At what packet size would the answer to question (4) be “both delay and throughput-limited”?

f) Can you change the speed of exactly one link so that your answer to question (4) be “both delay and throughput-limited”?

e) Assume ACKs have a negligible size. What is the total round-trip delay? Assume a packet size of 10 Bytes as before.

True and False

a) The Internet mandates that if Apple Inc. makes the physical and datalink layer, Apple must implement corresponding the network layer as well.

b) Under no circumstances would one choose to use NRZ encoding over Manchester encoding.

c) Nyquist sampling theorem proves that with a bandwidth of 20 MHz, you cannot send data faster than 40 Mbps.

Aside: Network Security

What could an attacker (in the network) still learn from an HTTPS connection (assuming HTTPS is working as intended and that the cryptography has not been broken)?

- Message Content ❌ Messages sent over an HTTPS connection are encrypted
- Destination IP Address ✓ The destination IP lies in the IP header and is used to route the packet to its destination; it is visible to the attacker.
- Source IP Address ✓ Similarly, the source IP is also visible to the attacker
- Time when the message was sent ✓ Another piece of metadata attackers have access to; attackers can observe timing of packets being sent
- Private key used ❌ Assuming cryptography has not been broken, the private key is encrypted and sent over the network and cannot be learned by the attacker.

Aside: Network Security

A message that is encrypted via symmetric encryption and uses MAC ensures the following:

- detecting whether message contents were changed ✓ As MAC is being used, it can be used to validate whether or not the message has been altered
- message content is kept private ✓ Message is encrypted and its contents are not visible to attackers
- messages cannot be intercepted and redirected ❌ Symmetric encryption alone cannot guarantee that packets are not intercepted/redirected; attackers can modify packet headers and redirect them
- identity of the message's sender is known ❌ The receiver only knows that a host having the same secret key has sent them the message. There is no inherent information on the identity of the sender
- message is fresh / was not replayed ❌ Not guaranteed by symmetric encryption / MAC alone, requires timestamp or nonce

Aside: Network Security

Which statements are true about VPNs with IPSEC (secured VPN):

- VPNs operate at the network layer ✓ VPNs use tunneling and encapsulate IP packets with IP headers
- Provide confidentiality AND authenticity ✓ granted by IPSEC
- Sends a packet through tunnels by rewriting the original packet's IP source/destination ✗ The existing header is maintained; the IP packet is wrapped with another IP header
- If a VPN server is compromised, anonymity is lost ✓ A compromised VPN server means attackers could trace back packets to their original source, identifying the user who has sent them
- No entity can tell whether or not a host is using a VPN ✗ Internet Service Providers (ISPs) can still detect whether or not someone is using a VPN.

Aside: Network Security

Explain the high level-steps (transport and above) taken when making an HTTPS request to Google.com, including receiving and handling. Assume nothing is cached.

- Address Lookup:
  - Browser makes a DNS query to resolve "www.google.com" to local DNS name server (recursive).
  - Local name server performs iterative queries. First queries root name server, then .com, finally obtaining IP address of google.com
  - Local name server caches IP address(es) and sends back response to the client
  - Client receives IP address and proceeds to establish a TCP connection.

Aside: Network Security

Explain the high level-steps (transport and above) taken when making an HTTPS request to Google.com, including receiving and handling. Assume nothing is cached.

- Transport Connection w/ Encryption
  - Client establishes a TCP connection via three-way handshake
  - TLS handshake begins:
    - Client and google.com negotiate encryption parameters (algorithm to use among other things)
    - Google.com sends its certificate for the client to verify. Client verifies certificate against list of Certificate Authorities
    - Client generates symmetric session key, encrypts it with google.com’s public key and sends it
    - Google.com decrypts message using its private key and obtains session key
    - Future messages are now encrypted/decrypted using session key

Aside: Network Security

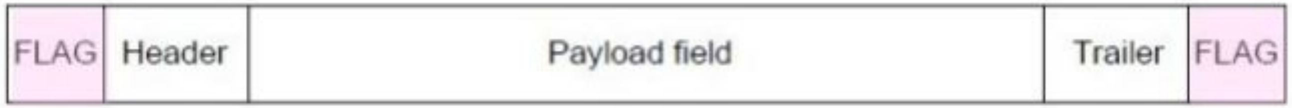
Explain the high level-steps (transport and above) taken when making an HTTPS request to Google.com, including receiving and handling. Assume nothing is cached.

- HTTP Connection/Data Transfer
    - Client sends HTTP request(s) (GET/POST/other, sets appropriate headers) via secure HTTP connection. Connection is persistent and can pipeline multiple requests.
    - Google.com receives requests, processes them, generates appropriate HTTP response
    - Client receives HTTP response, processes it by parsing content and/or fetching embedded resources, then renders result
    - Client cleans up TCP connections once idle

Link Layer

- Goal: Transfer frames over one or more connected links

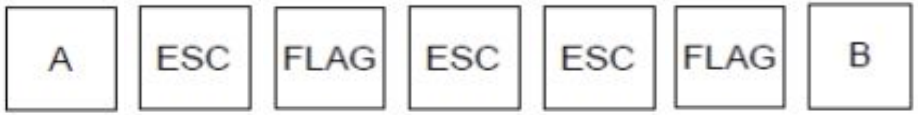
- Framing - how do we interpret a stream of bits as a sequence of frames?
    - Fixed-size frames (motivation), wasteful for small payloads
    - Byte count (motivation), store length of frame as header for frame
    - Byte stuffing/bit stuffing, encode flag at byte/bit level at start and end of frame



Byte Stuffing Exercise

Assume you have the following bytes to be sent over the link. What are the bytes after stuffing?

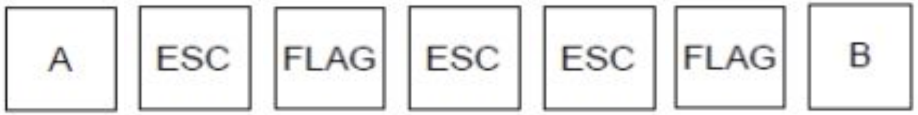
Original packets:



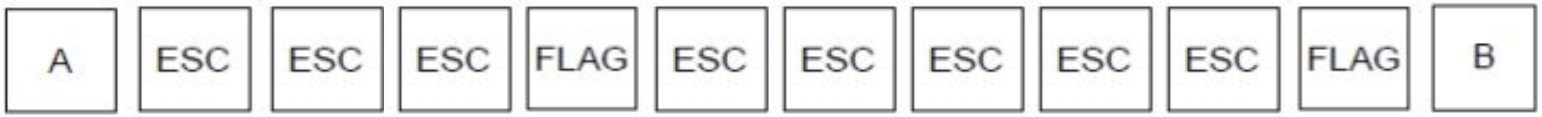
Byte Stuffing Exercise

Assume you have the following bytes to be sent over the link. What are the bytes after stuffing?

Original bytes:



After stuffing:



Error Detection, Correction

- Noise may flip some received bits in message

- Key idea: error detection/correction codes
  - Add check bits to the message to let some errors be detected (and more required for correction)

- Tradeoffs:
  - Adding check bits -> larger bandwidth requirements
  - Need modest computation to detect/correct error

Error Detection, Correction cont.

- ***Hamming distance*** of a coding - minimum error distance between any pair of codewords that cannot be detected
    - e.g. 1-dimensional parity has Hamming distance of 2, since 2 bit-flips required to get to next valid codeword
    - **1 0** 0 0 | 1 -> **0 1** 0 0 | 1, both pass error detection

- Error detection:
    - For a coding of distance d + 1, up to d errors will always be detected
- Error correction:
    - For a coding of distance 2d + 1, up to d errors can always be corrected by mapping to closest valid keyword

- Larger messages use ***checksums*** for error detection

Hamming distance cont.

- Given an unknown algorithm that has Hamming Distance of 9, What is the maximum number of errors it can reliably detect?

- What is the maximum number of errors it can reliably fix?

Hamming distance cont.

- Given an unknown algorithm that has Hamming Distance of 9, What is the maximum number of errors it can reliably detect?

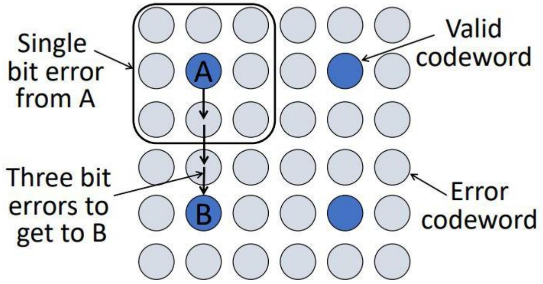
  8. As per the formula, $d + 1 = 9$, so $d = 8$ errors that can always be detected

- What is the maximum number of errors it can reliably fix?

  5. As per the formula, $2d + 1 = 9$, $d = 4$ errors that can always be corrected

Error Detection, Correction cont.

- Why is error correction harder?
  - Need to narrow down position of the error, but errors can also occur in the check bit
- With a code with a Hamming distance of at least 3,
  - Single bit errors will be closest to a unique valid codeword



- Error correction:
  - Needed when errors are expected
  - Or no time for retransmissions
  - Used in physical layer (802.11, DVB, WiMAX), application layer (FEC)
- Error detection:
  - More efficient when errors not expected
  - And errors are large when they do occur
  - Used with retransmission in link layer and above layers for residual errors

Multiple Access

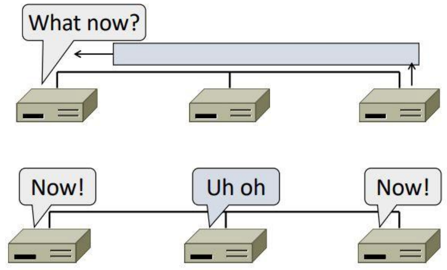
- Multiplexing shared link usage
    - Time Division Multiplexing (TDM) - users take turns on a fixed schedule
    - Frequency Division Multiplexing (FDM) - users placed on different frequency bands
    - Tradeoffs?

- Centralized vs. Distributed Multiple Access
    - Centralized, use a “scheduler” to choose who transmits and when
        - E.g. cellular networks (tower coordinates)
        - Scales well + efficient, but long setup and management
    - Distributed, have participants “figure it out” somehow
        - E.g. WiFi networks
        - Operates well in low load and easy setup, but hard to scale

Distributed (random) Access

- How do nodes share a single link? Who sends when?
    - Aloha - just send it whenever
    - CSMA (Carrier Sense Multiple Access) - listen for activity before we send
    - CSMA/CD (with Collision Detection) - receiver detects collision and aborts (jams)

- CSMA "Persistence"
    - Each node waits 2 * delay seconds to hear of a collision
    - Problem - multiple waiting nodes will queue up, then collide
        - Ideally want N senders to send with probability 1/N
        - Solution: Binary Exponential Backoff (BEB)
            - 1st collision, wait 0 or 1 frame times
            - 2nd collision, wait from 0 to 3 times
            - 3rd collision, wait from 0 to 7 times ....

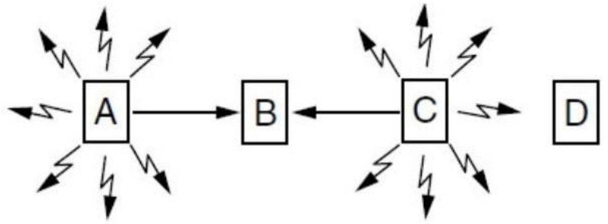


Wireless Complications

1. Media is infinite - can’t Carrier Sense
2. Nodes (usually) can’t hear while sending - can’t Collision Detect

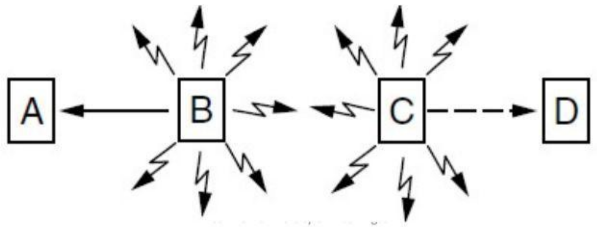
Hidden Terminal Problem:

- A and C can’t hear each other, but collide at B



Exposed Terminal Problem:

- B and C can hear each other, but don’t collide at A and D



MACA

- Uses short handshake instead of CSMA

- Protocol rules:
  1. A sender node transmits a RTS (Request-To-Send, with frame length)
  2. The receiver replies with a CTS (Clear-To-Send, with frame length)
  3. Sender transmits the frame while nodes hearing the CTS stay silent

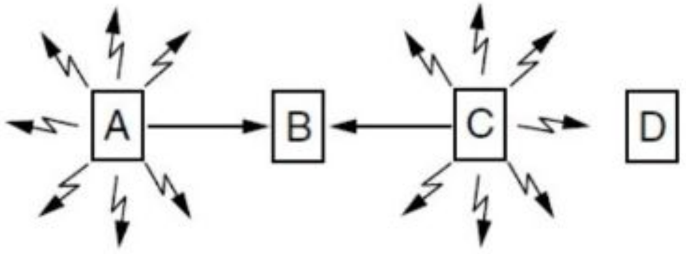
MACA

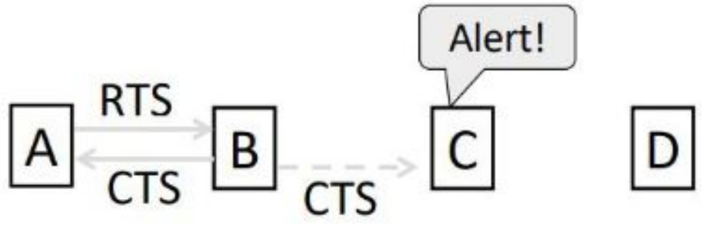
Hidden Terminal Problem:

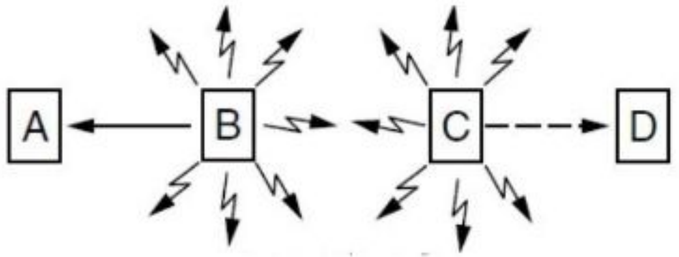
- A and C can’t hear each other, but collide at B

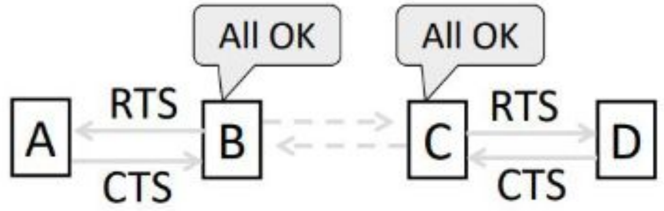
Exposed Terminal Problem:

- B and C can hear each other, but don’t collide at A and D









Switching

- Modern Ethernet uses switches instead of multiple access
  - Convenience running wires to one location
  - More reliable - wire cut is not a single point of failure

- Switch Forwarding
  - Switch needs to find right output port for destination address
  - Backward learning
    - To fill table, looks at source address of input frames
    - To forward, looks up address in table or broadcasts to all ports if not found

- Issue - forwarding loops
  - Solution, each switch runs distributed ***spanning tree*** algorithm
  - Finds subset of links w/ no loops and reaches all switches