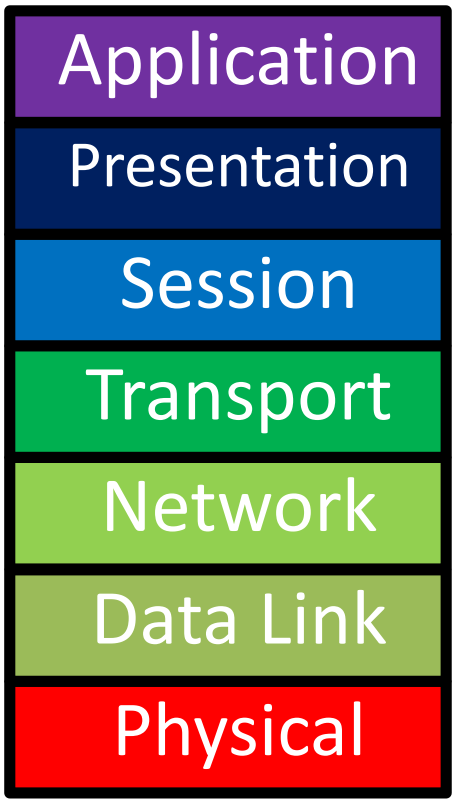
Data Link Layer

Data Link Layer



- Logical Link Control (LLC)
  - Send blocks of data (frames) between physical devices that share a medium
  - Perform error correction
- Medium Access Control (MAC)
  - Regulate access to the (shared) physical media
- Challenges:
  - How to delineate frames?
  - How to detect errors?
  - How to perform media access control (MAC)?
  - How to recover from and avoid collisions?

Example data link layer protocols

- Ethernet (IEEE 802.3)
- WiFi (IEEE 802.11)
- Bluetooth (IEEE 802.15)
- Zigbee

First order of business: How to delineate frames?

- Use preambles for synchronizing the sender and the receiver

- Start of frame delimitor (SFD) after the preamble for byte level synchronization.

Dealing with Noise

- The physical world is inherently noisy
  - Interference from electrical cables
  - Cross-talk from radio transmissions, microwave ovens
  - Solar storms
- How to detect bit-errors in transmissions?
- How to recover from errors?

Parity Bits

- Idea: add extra bits to keep the number of 1s even
  - Example: 7-bit ASCII characters + 1 parity bit

0101001 1 1101001 0 1011110 1 0001110 1 0110100 1

10

- Detects 1-bit errors and some 2-bit errors
- Not reliable against bursty errors
- Adds 14% overhead

Checksums

- Idea:
  - Add up the bytes in the data
  - Include the sum in the frame

- Use ones-complement arithmetic
- Lower overhead than parity: 16 bits per frame
- Used in UDP, TCP, and IP

Cyclic Redundancy Check (CRC)

- Uses field theory to compute a semi-unique value for a given message
- Much better performance than previous approaches
  - Fixed size overhead per frame (usually 32-bits)
  - Quick to implement in hardware
- No need to learn the details of CRC

802.3 Ethernet

Bytes	7	1	6	6	2	0-1500	0-46	4
	Preamble	SF	Source	Dest.	Length	Data	Pad	CRC

- Ethernet frames begin with SFD and ends with CRC.
- Source and destination are MAC addresses
- Minimum packet length of 64 bytes, hence the pad

What happens if packet is corrupted?

- Drop it and let the higher layers take care of it
  - Ethernet
- Retransmit
  - WiFi

Pros and Cons of dropping versus re-transmitting

- Cons:
  - Error free transmission cannot be guaranteed
  - Not all applications want this functionality
  - Error checking adds CPU and packet size overhead
  - Error recovery requires buffering
- Pros:
  - Potentially better performance than app-level error checking
  - Most useful over lossy links. Ethernet does not do anything if there are errors.
  - WiFi, cellular, satellite does use reliability at the link layer.

What is Medium Access?

- Ethernet and WiFi are both multi-access technologies
  - Broadcast medium, shared by many hosts
  - Simultaneous transmissions cause collisions
- Media Access Control (MAC) protocols are required
  - Rules on how to share the medium
  - Strategies for detecting, avoiding, and recovering from collisions

Strategies for Media Access

- Channel partitioning (aka, leave it to the physical layer)
  - Divide the resource into small pieces
  - Allocate each piece to one host
  - Time Division Multi-Access (TDMA), Frequency Division Multi-Access (FDMA), Code Division Multi-Access (CDMA
  - E.g., Cellular
- Taking turns
  - Tightly coordinate shared access to avoid collisions
  - Example: Token ring networks
- Contention
  - Allow collisions, but use strategies to recover
  - Examples: Ethernet, Wifi

WiFi and Ethernet do not use TDMA, CDMA, or FDMA.

- Low utilization
  - TDMA and FDMA have low utilization
  - Just like a circuit switched network
  - CDMA is just too complex
- Not distributed
  - Multiple hosts that cannot directly coordinate
  - No fancy code distribution schemes
  - No fancy (complicated) token-passing schemes

Instead, use random access MAC

- Aloha (first protocol used)
- CSMA/CD (Ethernet)
- CSMA/CA (WiFi)

Randomized access: ALOHA

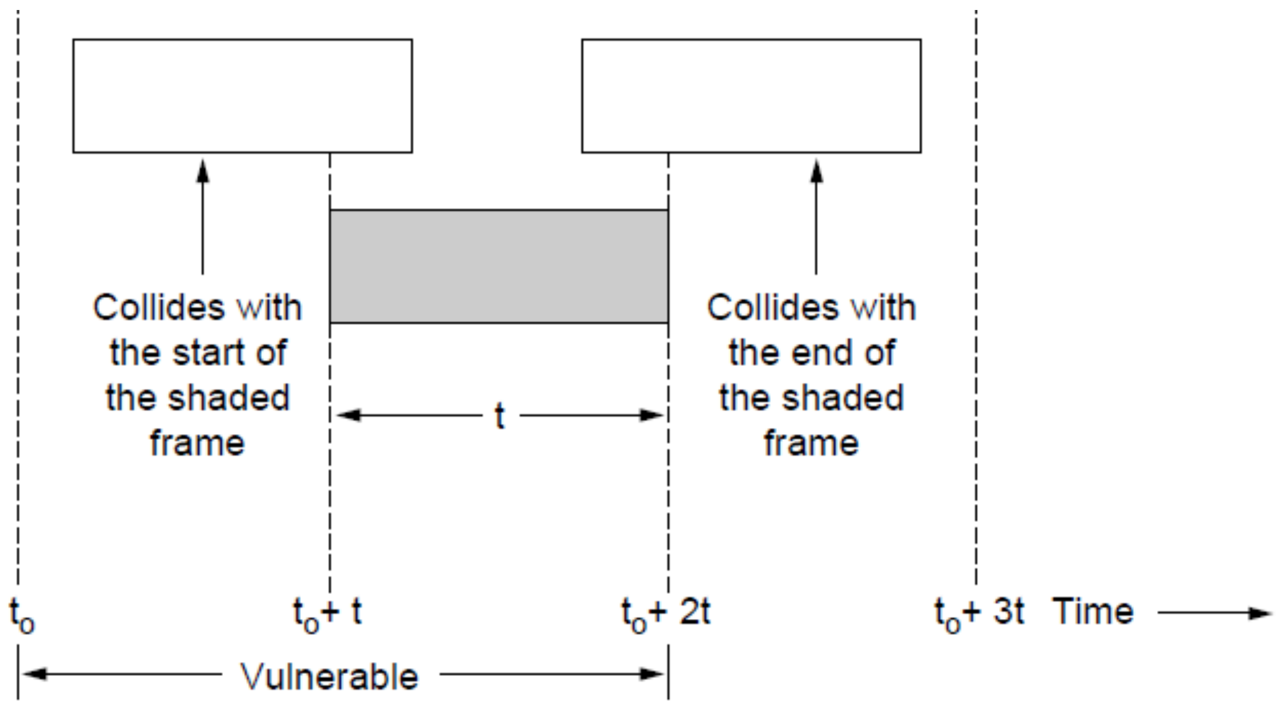
- Wireless links between the Hawaiian islands in the 70s

- Aloha protocol:
  - Just send when you have data!
  - There will be some collisions of course …
  - Detect error frames and retransmit a random time later
- Simple, decentralized and works well for low load

Problems with Aloha

Collisions time is twice the frame time

- Synchronizing senders to slots can reduce collisions



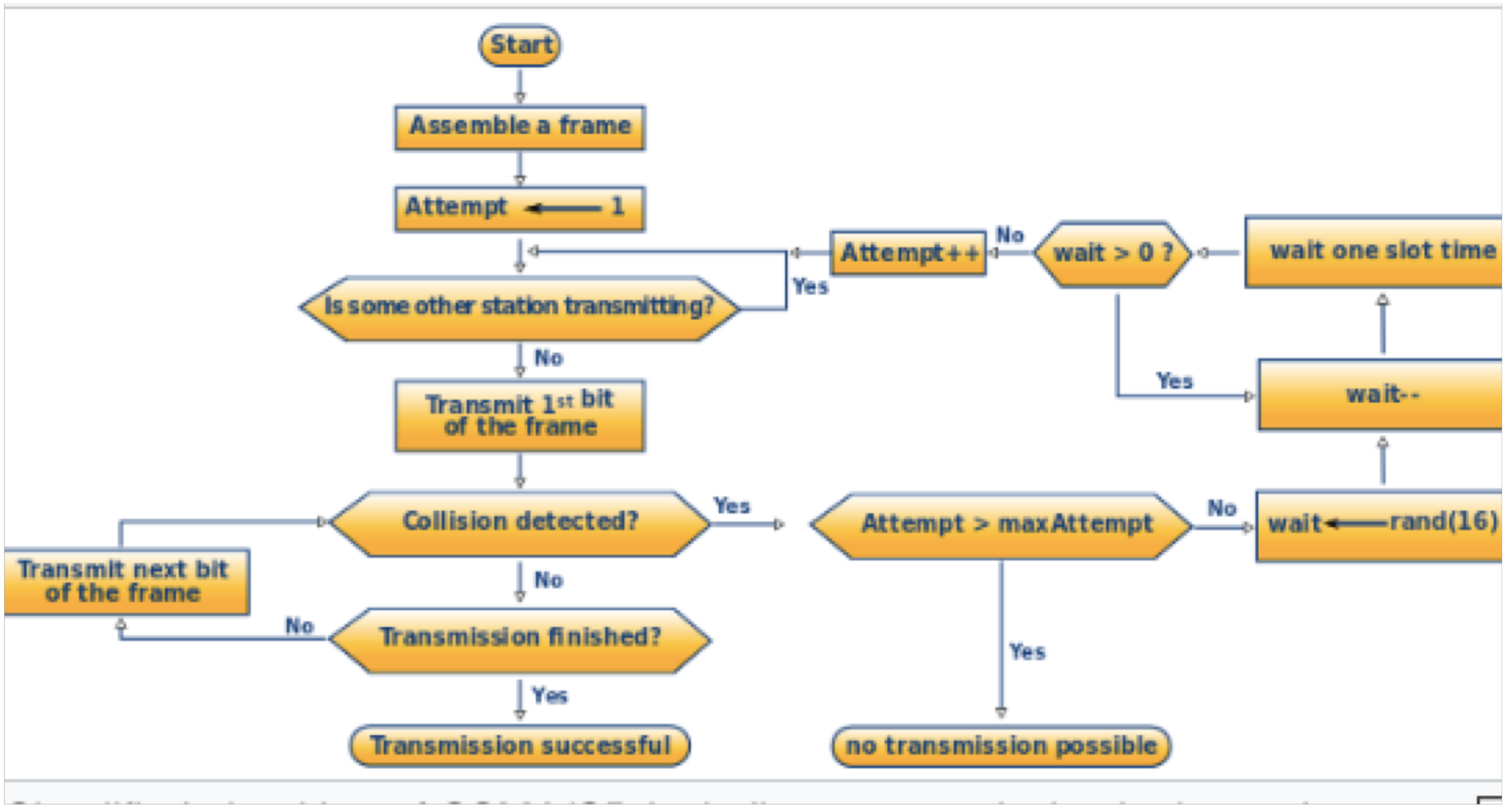
Slotted Aloha

- Divide time into slots
- Each transmission can start only at the beginning of the slot
- Thus, frames either collide completely, or not at all
  - But, hosts must have synchronized clocks

Ethernet: Use random access, but detect collision and backoff CSMA/CD

- Carrier sense multiple access with collision detection
- Key insight: wired protocol allows us to sense the medium
- Algorithm
  1. Sense for carrier
  2. If carrier is present, wait for it to end
     - Sending would cause a collision and waste time
  3. Send a frame and sense for collision
  4. If no collision, then frame has been delivered
  5. If collision, abort immediately
     - Why keep sending if the frame is already corrupted?
  6. Try to retransmit again.

CSMA/CD is one slide



With exponential backoff, the wait->rand(16) is replaced by increasing random numbers

What if the Channel is Busy?

- 1-persistent CSMA
  - Wait until idle then go for it
  - Blocked senders can queue up and collide---too greedy
- Instead, use exponential backoff

Exponential Backoff

- When contention is detected, back off.
- Exponential backoff
  - Select $k \in [0, 2^n - 1]$, where n = number of collisions
  - n is capped at 10, frame dropped after 16 collisions
- Backoff time is divided into contention slots

Exponential backoff example

- Node sends first frame and senses for collision (n=1)
  - If collision occurs, it chooses k = 1 or k = 0 each with probability 0.5
  - IF k=1, it waits 512 bit times, and then senses the channel
  - If k = 0, it immediately senses the channel
- If channel is free, node sends frame (n=2). If collision occurs again,
  - K is chosen from {0,1,2,3}
- n is capped to 10, where n is the number of collisio