CSE 422 Notes, Set 5

- These slides contain materials provided with the text: Computer Networking: A Top Down Approach, various editions, by Jim Kurose and Keith Ross, Addison-Wesley.
- Additional figures are repeated, with permission, from Computer Networks, 2nd through 4th Editions, by A. S. Tanenbaum, Prentice Hall.
- The remainder of the materials were developed by Philip McKinley at Michigan State University.

Link Layer

- Introduction and services
- Error detection
- Link-layer Addressing
- Multiple access protocols
- Ethernet operation

Reading assignment:
   6.1, 6.2, 6.3.1, 6.3.2, 6.4.1, 6.4.2
Link Layer: Introduction

Some terminology:
- hosts and routers are **nodes**
- communication channels that connect adjacent nodes along communication path are **links**
  - wired links
  - wireless links
- layer-2 packet is a **frame**, encapsulates datagram

**data-link layer** has responsibility for transferring datagram (in a frame) from one node to an adjacent node over a link

Link Layer Services

- **Responsibilities:**
  - Framing: encapsulate datagram into frame, adding header, trailer
  - Access control: channel access if shared medium
  - Addressing: “MAC” addresses used in frame headers to identify source, dest (different from IP address!)
  - Error control

- **Reliable delivery between adjacent nodes?**
  - We learned already how to build a reliable protocol.
    - sequence numbers on packets (frames) and ACKs
  - However, 100% reliability typically **not** implemented at link level
  - Instead, **enhanced** reliability (limited number of retransmissions)
    - Seldom used on low bit-error link (fiber, copper)
    - More common on wireless links due to high error rates
Where is the link layer implemented?

- in each and every node
- link layer implemented in “adapter” (a.k.a. network interface card -- NIC) and its associated driver
  - Ethernet card, 802.11 card, Ethernet chipset
  - implements link, physical layer
- attaches into host’s system buses
- combination of hardware, software, firmware

Adapters Communicating

- **sending side:**
  - encapsulates datagram in frame
  - adds checksum (and possibly reliability and flow control info)
- **receiving side**
  - detects/corrects errors, rdt, flow control, etc
  - extracts datagram, passes to upper layer at receiving side
Link Layer

- Introduction and services
- Error detection
- Multiple access protocols
- Link-layer Addressing
- Ethernet operation

Errors in communication

- Examples sources of errors
  - thermal noise
  - impulse noise: e.g., lightning
  - signal distortion
  - crosstalk
  - sender, receiver losing synchronization

- Even a single bit error can cause major damage. Why?
Two ways of handling errors...

- Error detection
  - error is detected in frame
  - frame is retransmitted
  - ex. parity checks, cyclic redundancy codes

- Error correction
  - sender transmits redundant information
  - receiver uses this to correct errors
  - not as widely used in data communication as error detection
  - ex. Hamming code, other bit-error codes, block erasure codes

- Why not always use correction??

Internet checksum?

**Goal:** detect “errors” (e.g., flipped bits) in transmitted packet (note: used at transport layer only)

**Sender:**
- treat segment contents as sequence of 16-bit integers
- checksum: addition (1’s complement sum) of segment contents
- sender puts checksum value into UDP checksum field

**Receiver:**
- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - NO - error detected
  - YES - no error detected.
  
  But maybe errors nonetheless?
  - Can we do better?
Cyclic Redundancy Codes (CRC)

- Basic idea: treat string of bits as coefficients of a polynomial, using modulo 2 arithmetic according to rules of algebraic field theory.
- Ex. 101001 represents \( x^5 + x^3 + 1 \)
- Addition and subtraction are both equivalent to exclusive-or

  \[
  \begin{array}{c}
  \begin{array}{c}
  10011011 \\
  +11001010 \\
  \end{array}
  \end{array}
  \]

Checksumming: Cyclic Redundancy Check

- View data bits, \( D(x) \), as a (long) binary number
- Choose \( r+1 \) bit pattern (generator), \( G(x) \)
- Goal: compute \( r \) CRC bits, \( R(x) \), such that
  - \( \langle D, R \rangle \) exactly divisible by \( G \) (modulo 2)
  - Receiver knows \( G \), divides \( \langle D, R \rangle \) by \( G \). If non-zero remainder: error detected!
  - Can detect all burst errors less than \( r+1 \) bits
- Widely used in practice (Ethernet, 802.11 WiFi, ATM)

\[
D \cdot 2^r \text{ XOR } R
\]
(Small) CRC Example

Frame: $1010001101$
$G(x): 110101$

When Will A CRC Code Fail?

- $D(x)$ represents the data
- $G(x)$ represents the generator polynomial
- $T(x)$ represents transmitted bits
- $R(x)$ represents received bits
- $E(x)$ represents error bits
- Know: $R(x) = T(x) + E(x)$
- What is the relationship between $G(x)$ and $R(x)$ for the code to fail?

- Hence, what is the relationship between $G(x)$ and $E(x)$ for the code to fail?
Example of CRC Failure

\[ G(x) = x^5 + x^4 + x^2 + 1 \]

Frame: 1010001101

T  1 0 1 0 0 0 1 1 0 1 0 1 1 1 0
R  1 0 1 0 0 0 1 0 1 0 0 1 1 1 1

Detecting Burst Errors

- Recall burst error of length \( n \)
  - errors in a frame are limited to \( n \)-bit range
  - some bits are correct, some are flipped

- A CRC with generator \( G(x) \) of degree \( r \)
  - can detect all burst errors of length \( r \) or less. Why?

  - Will fail to detect burst errors of length \( r + 1 \) with probability \( 1/2^{r-1} \). Why?

  - Will fail to detect burst errors of length \( r + 2 \) or more with probability \( 1/2^r \). Why?
**Example Generators**

- **CRC-12** = \( x^{12} + x^{11} + x^{3} + x^{2} + x^{1} + 1 \)
  - Used when character length is 6 bits. Why?
- **CRC-16** = \( x^{16} + x^{15} + x^{2} + 1 \)
- **CRC-CCITT** = \( x^{16} + x^{12} + x^{5} + 1 \)
  - Catch all burst errors of length 16 or less.
  - Catch 99.997% of burst errors of length 17.
  - Catch 99.998% of all burst errors of length 18 or more.

---

**More...**

- **CRC-32**
  \( x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1 \)
  detects 99.99999997% of all burst errors of length 34 or more.

- **CRC-64-ISO**
  \( x^{64} + x^{4} + x^{3} + x + 1 \)

- **CRC-64-ECMA-182**
  \( x^{64} + x^{62} + x^{57} + x^{55} + x^{54} + x^{53} + x^{52} + x^{47} + x^{46} + x^{45} + x^{40} + x^{39} + x^{38} + x^{37} + x^{35} + x^{33} + x^{32} + x^{31} + x^{29} + x^{27} + x^{24} + x^{23} + x^{22} + x^{21} + x^{19} + x^{17} + x^{13} + x^{12} + x^{10} + x^{9} + x^{7} + x^{4} + x + 1 \)
How effective is a 64-bit CRC?

- $1/2^{64} \approx 5 \times 10^{-20}$
- Estimated Internet traffic for 2019:
  - ~ 2 zettabytes ($2 \times 10^{21}$ bytes)
- Assume mean packet length is 500 bytes
- Assume 5% (1/20) of packets experience errors
  - Probably the percentage is far lower

How about 128-bit CRC?

- But, Internet traffic is increasing
- What if it increases by a factor of 1,000,000?

- Would a 128-bit CRC be sufficient?
- $1/2^{128} \approx 3 \times 10^{-39}$
- Same calculations indicates this code will fail to detect an error every (how many?) years.
Link Layer

- Introduction and services
- Error detection
- Link-layer Addressing
- Multiple access protocols
- Ethernet operation

IP/MAC Addresses and ARP

- 32-bit IP address:
  - network-layer address
  - used to get datagram to destination IP subnet

- Physical address
  - Function: *get frame from one interface to another physically-connected interface (same network)*
  - On a local network, typically a MAC address
  - E.g., 48 bit MAC address (for most LANs)
    - burned in NIC ROM, also sometimes software settable
LAN Addresses and ARP

Each adapter on LAN has unique LAN address

Broadcast address = FF-FF-FF-FF-FF-FF

ARP: Address Resolution Protocol

Question: how to determine MAC address of B knowing B’s IP address?

- Each IP node (host, router) on LAN has ARP table
- ARP table: IP/MAC address mappings for some LAN nodes
  - IP address; MAC address; TTL
    - TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)
ARP protocol: Same LAN (network)

- A wants to send datagram to B, and B’s MAC address not in A’s ARP table.
- A broadcasts ARP query packet, containing B’s IP address
  - dest MAC address = FF-FF-FF-FF-FF
  - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B’s) MAC address
  - frame sent to A’s MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - soft state: information that times out (goes away) unless refreshed
- ARP is “plug-and-play”: nodes create their ARP tables without intervention from net administrator

Multiple Access Links and Protocols

Two types of “links”:
- point-to-point
  - PPP for dial-up access
  - point-to-point link between Ethernet switch and host
- broadcast (shared wire or medium)
  - old-fashioned Ethernet
  - 802.11 wireless LAN
  - upstream shared channels (cable, satellite)
Multiple Access protocols

- single shared broadcast channel
- two or more simultaneous transmissions by nodes: interference
  - collision if node receives two or more signals at the same time

multiple access protocol

- distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- communication about channel sharing must use channel itself!
- No separate channel for coordination

Ideal Multiple Access Protocol

Broadcast channel of rate R bps

1. when one node wants to transmit, it can send at rate R.
2. when M nodes want to transmit, packets are transmitted first come, first served
3. simple...
Problem with static allocation

- Why not give each node its own subchannel?
- One channel at C bps vs. N subchannels, each at C/N bps
- L bits per packet
- Aggregate arrival rate is lambda
- Delivery time?

Dynamic Channel Allocation

- bandwidth allocated on a demand basis
- must be concerned about fairness
- question: how do nodes “agree” on which node should send when
Random Access Protocols

- When node has packet to send
  - transmit at full channel data rate R.
  - no a priori coordination among nodes
- two or more transmitting nodes → “collision”,
- random access MAC protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)
- Examples of random access MAC protocols:
  - ALOHA
  - Slotted ALOHA
  - CSMA, CSMA/CD, CSMA/CA

ALOHA

- Pure Aloha
  - when frame is ready, just send it.
  - stations listen while they send, and therefore detect collisions
  - when collisions occur, stations retransmit after a random period of time
- Slotted Aloha
  - As above but frames are sent on “slot” boundaries
  - One node can send a “beacon” signal to define slots
  - Improves performance compared to Pure Aloha
Carrier Sense Multiple Access

- What an idea! Listen before sending!

- 1-persistent CSMA
  - Station listens to channel. When the channel becomes idle, the frame is transmitted with probability 1.0.
  - If a collision occurs, the station waits random amount of time and repeats.
  - The longer the propagation delay, the worse the performance of the protocol. Why?
  - Even if the propagation delay is 0, there will still be collisions. Why?

CSMA (cont.)

- non-persistent CSMA
  - Station senses channel. If idle, send. If busy, waits a random amount of time and repeat.
  - Better channel utilization and longer delays than 1-persistent CSMA. Why?
CSMA (cont.)

- **p-persistent CSMA**
  - Station ready to send senses the channel.
  - If channel is idle, transmit with probability $p$. With probability $q = 1 - p$, wait one time unit (typically propagation delay) and repeat procedure.
  - If busy, listen until idle and repeat.

**ALOHA and CSMA Performance**

![Graph comparing channel utilization versus load for various random access protocols.]

- **What can we conclude?**
CSMA/CD (Collision Detection)

CSMA/CD: carrier sensing, deferral as in CSMA
  - collisions detected within short time
  - colliding transmissions aborted, wasting less channel capacity

Collision detection:
  - easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - difficult in wireless LANs: received signal strength overwhelmed by local transmission strength

Link Layer

- Introduction and services
- Error detection and correction
- Multiple access protocols
- Link-layer Addressing
- Ethernet operation
IEEE 802.3 (Ethernet)

- 1-persistent CSMA/CD
- Derives from original Ethernet network
  - built at Xerox, Metcalfe, Boggs et al
  - 2.94 Mbps,
  - later raised to 10 Mbps
  - 100 workstations, 1-km cable,
  - 802.3 covered a whole family of 1-persistent CSMA/CD systems, running at from 1 to 10 Mbps

Cabling Options
Frame structure

- preamble (7 bytes of 10101010)
- start-of-frame byte (10101011)
- destination address (allows multicast, broadcast)
- source address
- Type/length
- data
- pad (keeps minimum frame length at 64 bytes)
  Why?
- 32-bit checksum

802.3 Backoff Algorithm

- slot time set to 512 bit times
- after ith collision (up to 10) choose a random number of slots between 0 and $2^i - 1$ (helps minimize delay)
- failure after 16 collisions
- reason for this approach?
**Example**

**Ethernet CSMA/CD operation**

- **Typical behavior**
  - Successful transmission
  - Followed by contention period
  - Repeat...

![Diagram of Ethernet CSMA/CD operation with time line and frames]

**Diagram Details:**
- Time line (TIME) indicating the progression of the frames.
- Frames labeled with Transmission Period, Contention Period, Contention Slots, and Idle Period.
802.3 Performance

Delay Comparison

CSE 422 - McKinley  
Link Layer
Star topology

- bus topology popular through mid 90s
  - all nodes in same collision domain (can collide with each other)
- today: star topology prevails
  - active switch in center
  - each “spoke” runs a (separate) Ethernet protocol (nodes do not collide with each other)

802.3 Ethernet Standards: Link & Physical Layers

- many different Ethernet standards
  - common MAC protocol and frame format
  - different speeds: 2 Mbps, 10 Mbps, 100 Mbps, 1Gbps, 10G bps and beyond
  - different physical layer media: fiber, cable
  - different encoding methods
“Fast” Ethernet

- Fast Ethernet (802.3u) operates exactly as does regular 10Base-T Ethernet, except at a bit rate of 100 Mbps. That is, all fast Ethernet systems are based on hubs.
- Given the improvements in clocks and the short distances, Manchester encoding is not needed.
- Either hubs or switches may be used, and most handle mixes of 10/100 Mbps traffic.

Gigabit Ethernet (802.3z)

- Two modes
  - Switched full duplex: frames are buffered, no contention possible
  - Hub-based half duplex: emulates CSMA/CD
    - minimum packet length increased to 512 bytes, enabling 100 meter segments
    - frame bursting: chain together small frames to reach 512 bytes
    - now obsolete - don’t worry about contention
- Achieving gigabit speed on twisted pairs
  - use four twisted pairs in cable, and five voltage levels (00,01,10,11 plus special control/framing signal)
  - clock at 125MHz, with 8 bits per clock
Even Higher Rates…

- 10Gb, 40/100Gb, currently up to 400Gbit Ethernet (!)
- Key target domain: data centers
- Combinations of:
  - Low error rate media, specifically, optical fibers
  - Limited distance, depending on medium
  - Full duplex links (no contention!)
  - Complex modulation schemes, e.g., PAM4
  - High baud rates (symbols per second)
  - Forward error correction! (Reed-Solomon codes)
- Stay tuned through your careers!

Link Layer Summary

- With regard to error detection…
  Cyclic Redundancy Codes Rule!
- The beauty of universal (IP) addressing is somewhat tarnished by the need for dynamic address translation
  - But, frankly, ARP is pretty simple
- Shared channels require a contention protocol
  - Simple protocols (Aloha) work well under low loads
  - CSMA/CD is great under all conditions!
- Ethernet was initially just a cable, connectors and the CSMA/CD protocol
  - It continues to evolve, and dominate!