Link Layer Services
Error Correction and Detection

Based on Computer Networking, 4th Edition by Kurose and Ross

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
  - LANs
- layer-2 packet is a frame, encapsulates datagram
- data-link layer has responsibility of transferring datagram from one node to adjacent node over a link
**Link layer: context**

- Datagram transferred by different link protocols over different links:
  - e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link
- Each link protocol provides different services
  - e.g., may or may not provide rdt over link

**transportation analogy**
- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne
- tourist = datagram
- transport segment = communication link
- transportation mode = link layer protocol
- travel agent = routing algorithm

---

**Link Layer Services**

**Framing, link access:**
- encapsulate datagram into frame, adding header, trailer
- channel access if shared medium
- "MAC" addresses used in frame headers to identify source, dest
  - different from IP address!

**Reliable delivery between adjacent nodes**
- we learned how to do this already (chapter 3!)
- seldom used on low bit error link (fiber, some twisted pair)
- wireless links: high error rates
  - Q: why both link-level and end-end reliability?

**Flow Control:**
- pacing between adjacent sending and receiving nodes

**Error Detection:**
- errors caused by signal attenuation, noise.
- receiver detects presence of errors:
  - signals sender for retransmission or drops frame

**Error Correction:**
- receiver identifies and corrects bit error(s) without resorting to retransmission

**Half-duplex and full-duplex**
- with half duplex, nodes at both ends of link can transmit, but not at same time
Where Is the Link Layer Implemented?

- in each and every host
- link layer implemented in "adaptor" (aka network interface card NIC)
  - Ethernet card, PCMCI card, 802.11 card
  - implements link, physical layer
- attaches into host's system buses
- combination of hardware, software, firmware

Adaptors Communicating

- sending side:
  - encapsulates datagram in a frame
  - adds error checking bits, rdt, flow control, etc.
- receiving side
  - looks for errors, rdt, flow control, etc
  - extracts datagram, passes to rcving node
- adapter is semi-autonomous
- link & physical layers
**Error Detection**

- EDC = Error Detection and Correction bits (redundancy)
- D = Data protected by error checking, may include header fields

Error detection not 100% reliable!
- protocol may miss some errors, but rarely
- larger EDC field yields better detection and correction

---

**Parity Checking**

**Single Bit Parity:** Detect single bit errors

\[ \text{d data bits} \rightarrow \text{parity bit} \]

\[ 0111000110101011 \]

---

**Two Dimensional Bit Parity:** Detect and correct single bit errors

\[ \begin{array}{cccc}
    d_{1,1} & \ldots & d_{1,j} & d_{1,j+1} \\
    d_{2,1} & \ldots & d_{2,j} & d_{2,j+1} \\
    \vdots & \ldots & \vdots & \vdots \\
    d_{i,1} & \ldots & d_{i,j} & d_{i,j+1} \\
    d_{i+1,1} & \ldots & d_{i+1,j} & d_{i+1,j+1} \\
\end{array} \]

- row parity
- column parity

\[ \begin{array}{c}
    101011 \\
    111100 \\
    111101 \\
    001010 \\
\end{array} \]

- no errors

- parity error
- correctable
- single bit error

Stan Kurkovsky
**Internet checksum**

**Goal:** detect “errors” (e.g., flipped bits) in transmitted segment (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?*

---

**Checksumming: Cyclic Redundancy Check**

- view data bits, \( D \), as a binary number
- choose \( r+1 \) bit pattern (generator), \( G \)
- goal: choose \( r \) CRC bits, \( R \), such that
  - \(<D,R>\) exactly divisible by \( G \) (modulo 2)
  - receiver knows \( G \), divides \(<D,R>\) by \( G \). If non-zero remainder: error detected!
  - can detect all burst errors less than \( r+1 \) bits
- widely used in practice (802.11 WiFi, ATM)
**CRC example**

**Want:**

- \( D \cdot 2^r \) XOR \( R = nG \)

**equivalently:**

- \( D \cdot 2^r = nG \) XOR \( R \)

**equivalently:**

if we divide \( D \cdot 2^r \) by \( G \),

want remainder \( R \)

\[ R = \text{remainder}\left(\frac{D \cdot 2^r}{G}\right) \]