Last Lecture: Network Layer

1. Design goals and issues
2. Basic Routing Algorithms & Protocols
3. Addressing, Fragmentation and reassembly
4. Internet Routing Protocols and Inter-networking
5. Router design
   1. Short History + Architectures
   2. Switching fabrics
   3. Address lookup problem ✓
6. Congestion Control, Quality of Service
7. More on the Internet’s Network Layer
This Lecture: Data Link Layer

1. Design goals and issues ✔
2. (More on) Error Control and Detection ✔
3. Multiple Access Control (MAC)
4. Ethernet, LAN Addresses and ARP
5. Hubs, Bridges, Switches
6. Wireless LANs
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
Message, Segment, Packet, Frame

HTTP message

TCP segment

IP packet

Ethernet frame

SONET frame

router

host

HTTP

TCP

IP

Ethernet interface

host

HTTP

TCP

IP

Ethernet interface

host

HTTP

TCP

IP

Ethernet interface
Link Layer for Each Hop

- *IP packet transferred over multiple hops*
  - Each hop has a link layer protocol
  - May be different on different hops

- *Analogy: trip from Buffalo to New York*
  - Limo: Buffalo to BNI Airport
  - Plane: BNI to JFK
  - Train: JFK to Hotel

- *Refining the analogy*
  - Tourist == packet
  - Transport segment == communication link
  - Transportation mode == link-layer protocol
  - Travel agent == routing algorithm
Where Does Link Layer “Happen”? 

- **Link layer implemented in adaptor (net. interface card)**
  - Ethernet card, PCMCIA card, 802.11 card

- **Sending side:**
  - Encapsulates datagram in a frame
  - Adds error checking bits, flow control, etc.

- **Receiving side:**
  - Looks for errors, flow control, etc.
  - Extracts datagram and passes to receiving node
Link Layer Services

Basic services:
- Framing and encoding
- Error detection, correction

Access services:
- Sharing a broadcast channel: multiple access
- Link layer addressing

Performance services:
- Reliable data transfer, flow control: done!
Link Layer Basic Services

- **Encoding**
  - Representing the 0s and 1s

- **Framing**
  - Encapsulating packet into frame, adding header, trailer
  - Using MAC addresses, rather than IP addresses

- **Error detection**
  - Errors caused by signal attenuation, noise.
  - Receiver detecting presence of errors

- **Error correction**
  - Receiver correcting errors without retransmission
Message = [1 0 1 1]

Noise “flips” the 3\textsuperscript{rd} bit

The Problem
Aoccdrнig to rscheearch at an Elingsh uinervtisy, it deosn’t mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt тihng is that the frist and lsat ltteer are at the rght pclae. The rset can be a toatl mses and you can sitll raed it wouthit a porblem. Тihs is bcuseae we do not raed ervey lteter by it slef but the wrod as a wlohe.
Principles of Error Detecting/Correcting Codes

- **Messages**: vectors of length \( m \), i.e. \( \{0, 1\}^m \)

- **Encoding function**: \( f : \{0, 1\}^m \rightarrow \{0, 1\}^n \)  
  \((n > m \text{ to add redundancy})\)

- Given message \( x \), send \( y = f(x) \)

- Receiver receive \( y' \) (possibly different from \( y \))

- **Decoding**: get back \( x \) from \( y' \)

**The Solution**
Efficiency of the System

- How much extra redundancy added?
  - $n/m$ is the code rate, want to keep near 1

- How many errors can the system detect, correct?
  - Say, it can detect $e$ bit-errors, want it to be large

- Natural tradeoff between rate and error detection capability
  - Small $n/m$ implies small $e$
What Shannon + Hamming Taught Us

\[ C = \{ f(x) \mid x \in \{0, 1\}^m \} \]

Is called the set of codewords

The minimum distance of \( C \) is

\[ \Delta(C) = \min_{c_1 \neq c_2 \in C} (\text{Hamming-Distance}(c_1, c_2)) \]

We can always detect \( \Delta(C) - 1 \) errors

We can always correct \( \left\lfloor \frac{\Delta(C) - 1}{2} \right\rfloor \) errors
**Examples We’ve Seen: Parity Checking**

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

- d data bits
- 0111000110101011 0

---

**Two Dimensional Bit Parity:**
Detect 3 bit-errors and correct single bit errors

- d<sub>1,1</sub> ... d<sub>1,j</sub>
- d<sub>2,1</sub> ... d<sub>2,j</sub>
- ... ... ...
- d<sub>i,1</sub> ... d<sub>i,j</sub>
- d<sub>i+1,1</sub> ... d<sub>i+1,j</sub>

---

<table>
<thead>
<tr>
<th>101011</th>
<th>101011</th>
</tr>
</thead>
<tbody>
<tr>
<td>111100</td>
<td>101100</td>
</tr>
<tr>
<td>011101</td>
<td>011101</td>
</tr>
<tr>
<td>001010</td>
<td>001010</td>
</tr>
</tbody>
</table>

*no errors*

---

*parity error*

---

*correctable single bit error*
CRC Code: More Sophisticated Error Detection

- 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!
- *Widely used in practice* (Ethernet, 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 \( R \)

\[ R = \text{remainder}[\frac{D \cdot 2^r}{G}] \]
CRC in terms of Polynomials

- **Message** $M$ length $k$ (110011)
  - $M(x) = x^5 + x^4 + x + 1$

- **$G$** is given as a *Generator Polynomial* of degree $r$
  - $CRC-12 = x^{12} + x^{11} + x^3 + x^2 + x^1$
  - $CRC-16$, $CRC-CCITT$, $CRC-32$

- Arithmetic Modulo 2 is now done in terms of these polynomials
  - $M(x) x^r = G(x)Q(x) + R(x)$
  - $R(x)$ represent the bits to be added to message

- *In practice*: use circuit consisting of XOR-gates and shift registers $\rightarrow$ very fast
CRC Can Detect

- All single-bit errors
- All double-bit errors, as long as G has at least 3 1’s
- Any odd number of errors, as long as G contains a factor \((x+1)\) (why?)
- Any burst error of length \(n\) or less
- Most larger burst errors
- Probability of undetected \((n+1)\)-burst error is \(\frac{1}{2^{n-1}}\)
- Probability of undetected longer burst error is \(\frac{1}{2^n}\)