Announcements

- Homework 8 due today.
- Final Project:
  - Code snapshot due today or Thursday.
  - Aim for working solitaire game before lab meeting next week.
Today's Plan

- TCP in detail!
- Codes for error detection and correction.
Networks have Layers!

- Application Layer
  - HTTP
  - POP
  - SMTP
  - XMPP

- Transport Layer
  - TCP
  - UDP
  - RTP

- Internet Layer
  - IP

- Network Layer
  - Packet Switching
  - Ethernet
  - WiFi

- Data Link Layer
  - Start bits
  - Packet Formats

- Physical Layer
  - On Off Keying ...
**Senders: Complete This Section**

- Complete items 1, 2, and 3. Also complete item 4 if Restricted Delivery is desired.
- Print your name and address on the reverse so that we can return the card to you.
- Attach this card to the back of the mailpiece, or on the front if space permits.

1. Article Addressed to:
   - Dennis Dillon
   - Nassau County District Attorney
   - 262 Old Country Rd
   - Mineola, NY 11501

2. Article Number
   - (Transfer from service label)
   - 7003 3110 0001 1629 5492

**Complete This Section on Delivery**

<table>
<thead>
<tr>
<th>A. Signature</th>
<th>B. Received by (Printed Name)</th>
<th>C. Date of Delivery</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

D. Is delivery address different from item 1?  
   - Yes
   - If YES, enter delivery address below:  

8. Service Type
   - Certified Mail  
   - Express Mail  
   - Registered  
   - Return Receipt for Merchandise  
   - Insured Mail  
   - C.O.D.

4. Restricted Delivery? (Extra Fee)  
   - Yes

---

**United States Postal Service**

- Sender: Please print your name, address, and ZIP+4 in this box
...inside Ethernet packet!

<table>
<thead>
<tr>
<th>PREAMBLE</th>
<th>TO</th>
<th>FROM</th>
<th>LENGTH/TYP</th>
<th>ERROR CHECK</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>48</td>
<td>48</td>
<td>16</td>
<td>32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IP version</th>
<th>Hdr Len</th>
<th>Service class</th>
<th>Packet Length</th>
<th>Packet Number</th>
<th>Fragment Num</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TTL</th>
<th>Protocol</th>
<th>Error Check</th>
<th>From Addr</th>
<th>To Addr</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source Port</th>
<th>Destination Port</th>
<th>Sequence Number</th>
<th>Acknowledgement Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hdr Len</th>
<th>Flags</th>
<th>Receiver Window</th>
<th>Urgent Pointer</th>
<th>DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Error Check</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
One by One?

DATA

ACKNOWLEDGMENT
Maintaining Transmission Efficiency

Segment 1 -> Ack 1
Segment 2 -> Ack 2
Segment 3 -> Ack 3
Segment 4 -> Ack 4
Segment 5 -> Ack 5
Segment 6 -> Ack 6
Segment 7 -> Ack 7
Segment 8 -> Ack 8
Segment 9 -> Ack 9
Segment 10 -> Ack 10
Segment 11 -> Ack 11
Segment 12 -> Ack 12
Cumulative Acknowledgments

Segment 1
Segment 2
Segment 3
Segment 4
Segment 5
Segment 6
Segment 7
Segment 8
Segment 9
Segment 10
Segment 11
Segment 12

Ack 1
Ack 2
Ack 3
Ack 4
Ack 5
Ack 6
Ack 7
Ack 8
Ack 9
Ack 10
Ack 11
Ack 12
Cumulative Acknowledgments

Segment 1
Segment 2
Segment 3
Segment 4
Segment 5
Segment 6
Segment 7
Segment 8
Segment 9
Segment 10
Segment 11
Segment 12
Segment 8

Acknowledgments:

Ack 1
Ack 2
Ack 3
Ack 4
Ack 5
Ack 6
Ack 7
Ack 7
Ack 7
Ack 7

Ack 12
Byte Number Sequencing

BYTE 1

BYTE 21

BYTE ???

Length = 50

Length = 20

Length = 100

Length = 50

Ack 21

Ack 121

Ack ???
Byte Number Sequencing

BYTE 1
Length = 20
Ack 21

BYTE 21
Length = 100
Ack 121

BYTE 121
Length = 50
Ack 171
Slow Segments == Confused Conversations

GET /index.html 1
Ack 1
Contents of index.html 1
Retransmission of
GET /index.html 1
Ack 1
Contents of index.html 1
GET /main.html 1
Ack 1
The Connection Start 3-Way Handshake

SYN (seq # = x)

Ack y+1

GET /index.html x+1

Ack x+1 & SYN (seq # = y)

Ack x+2

Contents of index.html (seq = y+1)
### TCP Segment Format

<table>
<thead>
<tr>
<th></th>
<th>Source Port</th>
<th>Destination Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Sequence (Segment) Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Acknowledgment Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Receiver window</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Flags</th>
<th>Urgent Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hdr Len</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>DATA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Flags</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>(nothing)</td>
<td>Here’s some data</td>
</tr>
<tr>
<td>SYN</td>
<td>Connect</td>
</tr>
<tr>
<td>FIN</td>
<td>Disconnect</td>
</tr>
<tr>
<td>ACK</td>
<td>I got segment #___</td>
</tr>
<tr>
<td>RESET</td>
<td>I’m confused and want to hang up</td>
</tr>
<tr>
<td>URG</td>
<td>By the way, …</td>
</tr>
<tr>
<td>PUSH</td>
<td>Send data now</td>
</tr>
</tbody>
</table>
Connection Termination

FIN (seq # = z)

Ack W+1

Ack z+1 & FIN (seq # = W)
TCP Demo
“I was very surprised by a picture my brothel posted on Facebook yesterday”

ASCII codes for r and l:

- \( l = 01101100 \)
- \( r = 01101010 \)

“He posted it in response to Trump’s comments about the Civil War”
One Bad Bit

if( safe ) . . .

==\> 01101001 01100110 00101000 00100000 01110011
  01100001 01100110 01100101 00100000 00101001 . . .

==\> 01101001 01100110 00101000 00100001 01110011
  01100001 01100110 01100101 00100000 00101001 . . .

if(!safe ) . . .
Random Bit Errors

is delivered as

0 1 0 1 0 0 0 1

or

0 1 0 0 0 0 0 1 or 0 1 0 1 1 1 0 0 1

or

0 1 0 1 1 0 0 1 or 0 0 0 1 0 0 0 1

or

1 1 0 1 0 0 0 1 or 0 1 0 0 0 0 0 0

with small \(10^{-12} - 10^{-3}\), independent probability \(p_e\) of an error in any given bit
Burst Errors
Burst Errors

becomes

where each ? is randomly replaced by a 0 or 1 with equal probability.
Burst Errors

Given the 5 bits of random interference

1 0 1 0 0

the transmitted message

0 1 0 1 0 0 0 1

becomes

0 1 ? ? ? ? ? 1

or

0 1 1 0 1 0 0 1
Burst Errors

Given the 5 bits of random interference

0 1 0 1 0

the transmitted message

0 1 0 1 0 0 0 1

becomes

0 1 ?? ?? ?? 1

or

0 1 0 1 0 1 0 1
Burst Errors

0 1 1 0 1 0 0 1 0 1 0 1 0 0 0 1 0 1 0 0 0 0 1 1

becomes