Link Layer
Link Layer

• Transfer **frames** over one or more connected links
  • Frames are messages of limited size
  • Builds on the physical layer which moves stream of **bits**
In terms of layers ...

**Network**

- Sending machine
  - Packet

- Receiving machine
  - Packet

**Link**

**Physical**

- Actual data path
In terms of layers ...

Network

Sending machine
Packet

Receiving machine
Packet

Link

Frame

Virtual data path

Physical

Header Payload field Trailer

Actual data path
Typical Implementation of Layers (2)
Topics we’ll cover

1. Framing
   • Delimiting start/end of frames
2. Error detection and correction
   • Handling errors
3. Retransmissions
   • Handling loss
4. Multiple Access
   • 802.11, classic Ethernet
5. Switching
   • Modern Ethernet
Framing
Delimiting start/end of frames
Framing: Problem

• How do we interpret a stream of bits as a sequence of frames?
Framing Methods

1. Fixed-size frames (motivation)
2. Byte count (motivation)
3. Byte stuffing
4. Bit stuffing

- In practice, the physical layer often helps to identify frame boundaries
  - E.g., Ethernet, 802.11
1. Fixed-size frames

• Make every frame a fixed number of bits
  • Pad smaller frames

• Problems?
  • Wasted transmissions for small frames
2. Byte Count

• Start each frame with a length field

• Problems?
2. Byte Count: Problem

• Difficult to re-synchronize after framing error
  • Want a way to scan for a start of frame
3. Byte Stuffing

- A special **flag** byte value for start/end of frame
  - Replace ("stuff") the flag with an escape code

- Problems?
3. Byte Stuffing: Problem

- Must escape the escape code too! Rules:
  - Replace each FLAG in data with ESC FLAG
  - Replace each ESC in data with ESC ESC

- Now any unescaped FLAG denotes frame start/end
You see:

1. Solitary FLAG? -> Start or end of packet
2. Solitary ESC? -> Bad packet!
3. ESC FLAG? -> remove ESC and pass FLAG through
4. ESC ESC FLAG? -> removed ESC and then start of end of packet
5. ESC ESC ESC FLAG? -> pass ESC FLAG through
6. ESC FLAG FLAG? -> pass FLAG through then start of end of packet
4. Bit Stuffing

• Can stuff at the bit level too
  • Call a flag six consecutive 1s
  • On transmit, after five 1s in the data, insert a 0
  • On receive, a 0 after five 1s is deleted
Link Example: PPP over SONET

• PPP is Point-to-Point Protocol
• Widely used for link framing
  • E.g., it is used to frame IP packets that are sent over SONET optical links
Link Example: PPP over SONET (2)

• Think of SONET as a bit stream, and PPP as the framing that carries an IP packet over the link.

Protocol stacks

PPP frames may be split over SONET payloads
Link Example: PPP over SONET (3)

- Framing uses byte stuffing
  - **FLAG** is 0x7E and **ESC** is 0x7D

<table>
<thead>
<tr>
<th>Bytes</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>1 or 2</th>
<th>Variable</th>
<th>2 or 4</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flag</td>
<td>01111110</td>
<td>Address</td>
<td>11111111</td>
<td>Control</td>
<td>00000011</td>
<td>Protocol</td>
<td>Payload</td>
</tr>
</tbody>
</table>

CSE 461 University of Washington 19
Link Example: PPP over SONET (4)

• Byte stuffing method:
  • To stuff (unstuff) a byte
    • add (remove) ESC (0x7D)
    • and XOR byte with 0x20
  • Removes **FLAG** from the contents of the frame