Chapter 2: Application layer

- 2.1 Principles of network applications
- 2.2 Web and HTTP
- 2.3 FTP
- 2.4 Electronic Mail
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 P2P file sharing
- 2.7 Socket programming with TCP
- 2.8 Socket programming with UDP
- 2.9 Building a Web server
HTTP Recap

- Protocol for the World Wide Web
  - Client-server architecture
  - Pull protocol (you request, “pull,” the html file you want, the server does not push it onto you)
  - Steps in an HTTP communication
    - Handshaking and connection set-up
  - Types and format of HTTP messages
    - All in ASCII

- New vocabulary
  - Port number, protocol and processes

- Using telnet

For Today: Electronic mail

- Major elements of email
- Main protocols (and port numbers)
- Types and format of messages
- Steps for email messages to move from sender to receiver, through the Internet
Electronic Mail

Three major components:

1) user agents
2) mail servers
3) SMTP: simple mail transfer protocol
   (and user access protocols)

Scenario: Alice sends message to Bob
Discussion Question

- The textbook states “SMTP does not normally use intermediate mail servers for sending mail…”
- Are devices in the network core used in sending mail? Explain.

Electronic Mail: SMTP

- There are three phases in SMTP
  - handshaking (greeting)
  - transfer of messages
  - closure
- command/response interaction
  - commands: ASCII text
  - response: status code and phrase
- client and server sides of SMTP run on every mail server
  - Use persistent TCP connections (reliable transfer)
  - Use port 25
- messages must be in ASCII
  - No binary data can be send - meaning what!? 
Sample SMTP interaction

- In the following interaction with SMTP, which lines are
  - Handshaking
  - Transfer of message
  - Closure

Sample SMTP interaction

fcapmaster:- jcardell$ telnet smtp.smith.edu 25
Trying 131.229.64.236...
Connected to baton.smith.edu.
Escape character is '^]'.

C: HELO jbc.edu
S: 250 baton.smith.edu Hello [131.229.102.128], pleased to meet you
C: MAIL FROM: <judy@jbc.edu>
S: 250 2.1.0 <judy@jbc.edu>... Sender ok
C: RCPT TO: <jcardell@smith.edu>
S: 250 2.1.5 <jcardell@smith.edu>... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Hello Me
C: This is an email message from me as a user agent via telnet
C: .
S: 250 2.0.0 s8GFb0Q4007216 Message accepted for delivery
C: QUIT
S: 221 2.0.0 baton.smith.edu closing connection

Connection closed by foreign host.
Mail message format

* Example of the actual message - NOT part of the SMTP handshaking process

- header lines, e.g.,
  - To:
  - From:
  - Subject: different from SMTP commands!

- body
  - the "message", ASCII characters only

Try SMTP interaction with Smith accounts
(using telnet so you can be the user agent)

- Send email without using a fancy email client, but with you doing all the tasks your mail reader usually does for you
  - telnet <servername> 25
  - You should receive a '220' reply from the server
  - enter HELO, MAIL FROM:, RCPT TO:, DATA, QUIT commands

- Next, include header lines in the actual message

- In Chrome/gmail, compare the message 'properties' of the two messages
Back to SMTP \(\rightarrow\) shortcomings

... 

- messages must be in ASCII
  - No binary data can be send - meaning what!? 

Message format: multimedia extensions

- MIME: multipurpose internet mail extension
- additional lines in message header define the MIME content

```
From: alice@edf.ch
To: bob@cornell.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data ..... 
........................
......base64 encoded data
```
**Base64 Encoding**

<table>
<thead>
<tr>
<th>Value</th>
<th>Char</th>
<th>Value</th>
<th>Char</th>
<th>Value</th>
<th>Char</th>
<th>Value</th>
<th>Char</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A</td>
<td>16</td>
<td>Q</td>
<td>32</td>
<td>g</td>
<td>48</td>
<td>w</td>
</tr>
<tr>
<td>1</td>
<td>B</td>
<td>17</td>
<td>R</td>
<td>33</td>
<td>h</td>
<td>49</td>
<td>x</td>
</tr>
<tr>
<td>2</td>
<td>C</td>
<td>18</td>
<td>S</td>
<td>34</td>
<td>i</td>
<td>50</td>
<td>y</td>
</tr>
<tr>
<td>3</td>
<td>D</td>
<td>19</td>
<td>T</td>
<td>35</td>
<td>j</td>
<td>51</td>
<td>z</td>
</tr>
<tr>
<td>4</td>
<td>E</td>
<td>20</td>
<td>U</td>
<td>36</td>
<td>k</td>
<td>52</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>F</td>
<td>21</td>
<td>V</td>
<td>37</td>
<td>l</td>
<td>53</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>G</td>
<td>22</td>
<td>W</td>
<td>38</td>
<td>m</td>
<td>54</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>H</td>
<td>23</td>
<td>X</td>
<td>39</td>
<td>n</td>
<td>55</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>I</td>
<td>24</td>
<td>Y</td>
<td>40</td>
<td>o</td>
<td>56</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>J</td>
<td>25</td>
<td>Z</td>
<td>41</td>
<td>p</td>
<td>57</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>K</td>
<td>26</td>
<td>a</td>
<td>42</td>
<td>q</td>
<td>58</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>L</td>
<td>27</td>
<td>b</td>
<td>43</td>
<td>r</td>
<td>59</td>
<td>7</td>
</tr>
<tr>
<td>12</td>
<td>M</td>
<td>28</td>
<td>c</td>
<td>44</td>
<td>s</td>
<td>60</td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>N</td>
<td>29</td>
<td>d</td>
<td>45</td>
<td>t</td>
<td>61</td>
<td>9</td>
</tr>
<tr>
<td>14</td>
<td>O</td>
<td>30</td>
<td>e</td>
<td>46</td>
<td>u</td>
<td>62</td>
<td>+</td>
</tr>
<tr>
<td>15</td>
<td>P</td>
<td>31</td>
<td>f</td>
<td>47</td>
<td>v</td>
<td>63</td>
<td>/</td>
</tr>
</tbody>
</table>

4-18

**Base64 Encoding**

- **Original (binary) bit stream**
  
  100110111010001011101001
  
  100110 ... 111010 ... 001011 ... 101001

- **Which corresponds to the 6-bit values**
  
  38, 58, 11 and 41

- **Which are encoded as**
  
m6Lp
**Mail access protocols**

- SMTP is a 'PUSH' protocol
- So how do we 'PULL' messages off the mail server?

**SMTP: compared to HTTP**

- HTTP: pull (you pull info from a server when desired)
- SMTP: push; POP, IMAP, (HTTP): pull
- both have ASCII command/response interaction, status codes
- SMTP: multiple objects sent in one message, using encoding as needed
  - SMTP requires message (header & body) to be in ASCII
- HTTP: each object encapsulated in its own response message
Brief Glimpse: FTP elements

- File transfer protocol
- Two connections
  - Control connection
  - Data connection
- “Out of band”
- The control connection maintains state information

FTP: separate control & data connections

The FTP Client:
- Contacts the FTP server at port 21
- This is a control connection, used to log in
- Commands for file transfer are over this control connection
  - List/Change directory
  - Request to send or receive files ...
**FTP: separate control & data connections**

The Server:
- Listens on port 21 for an incoming connection request
- When server receives a request, the server opens a separate **data connection** to client

**Port Numbers**
- Can google for list of assigned port numbers:
  - HTTP – 80
  - FTP – 20 & 21
  - SMTP – 25
  - POP – 110
  - IMAP – 143
  - (DNS – 53 over UDP)
Summary

- **New protocols**
  - SMTP – email delivery and storage
  - Mail access protocols
    - POP3, IMAP, HTTP
- **Using telnet to spoof being**
  - an HTTP client agent
  - an email client agent
- **Glimpse of FTP – 2 channels**