

CS 640 Introduction to Computer Networks

Lecture 19

CS 640

Today's lecture

- E-mail
 - Overview
 - Message format
 - SMTP
 - IMAP/POP

CS 640

Internet apps: application, transport protocols

<u>Application</u>	<u>Application layer protocol</u>	<u>Underlying transport protocol</u>
e-mail	smtp [RFC 821]	TCP
remote terminal access	telnet [RFC 854]	TCP
Web	http [RFC 2068]	TCP
file transfer	ftp [RFC 959]	TCP
streaming multimedia	proprietary (e.g. RealNetworks)	TCP or UDP
remote file server	NFS	TCP or UDP
Internet telephony	proprietary (e.g., Vocaltec)	typically UDP

CS 640

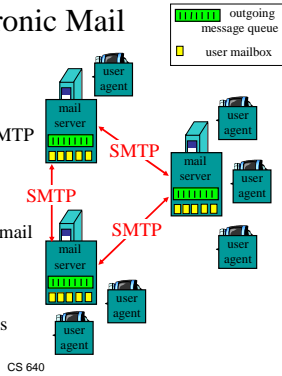
Electronic Mail

Three major components:

- user agents
- mail servers
- simple mail transfer prot.: SMTP

User Agent

- a.k.a. "mail reader"
- composing, editing, reading mail messages
- e.g., Eudora, Outlook, pine, Netscape Messenger
- outgoing, incoming messages stored on server

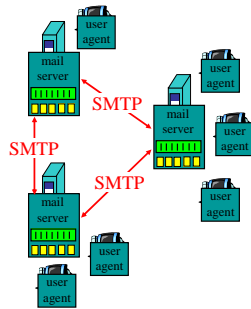


CS 640

Electronic Mail: mail servers

Mail Servers

- mailbox contains incoming messages (yet to be read) for user
- message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages
 - client: sending mail server
 - "server": receiving mail server



CS 640

Today's lecture

- E-mail
 - Overview
 - Message format
 - SMTP
 - IMAP/POP

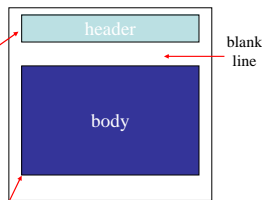
CS 640

Mail message format

SMTP: protocol for exchanging email messages

RFC 822: standard for text message format:

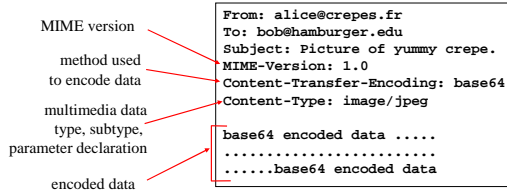
- header lines, e.g.,
 - To:
 - From:
 - Subject:*different from SMTP commands!*
- body
 - the "message", ASCII characters only



CS 640

Message format: multimedia extensions

- MIME: email format extension, RFC 2045, 2056
- additional lines in msg header declare MIME content type



CS 640

MIME types

Content-Type: type/subtype; parameters

Text

- example subtypes: **plain**, **html**

Image

- example subtypes: **jpeg**, **gif**

Audio

- example subtypes: **basic** (8-bit mu-law encoded), **32kadpcm** (32 kbps coding)

Video

- example subtypes: **mpeg**, **quicktime**

Application

- other data that must be processed by reader before "viewable"
- example subtypes: **mword**, **octet-stream**

CS 640

Multipart Type

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary=98766789
```

```
--98766789
Content-Transfer-Encoding: quoted-printable
Content-Type: text/plain

Dear Bob,
Please find a picture of a crepe.
--98766789
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.....base64 encoded data
--98766789--
```

CS 640

Today's lecture

- E-mail
 - Overview
 - Message format
 - SMTP
 - IMAP/POP

CS 640

Electronic Mail: SMTP [RFC 821]

- uses TCP to reliably transfer email messages from client to server, port 25
- direct transfer: sending server to receiving server
- three phases of transfer
 - handshaking (greeting)
 - transfer of messages
 - closure
- command/response interaction
 - commands: ASCII text
 - response: status code and phrase
- messages must be in 7-bit ASCII

CS 640

Sample SMTP interaction

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C:   How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

CS 640

Try SMTP interaction for yourself:

- `telnet servername 25`
 - see 220 reply from server
 - enter HELO, MAIL FROM, RCPT TO, DATA, QUIT commands
- above lets you send email without using email client (reader)

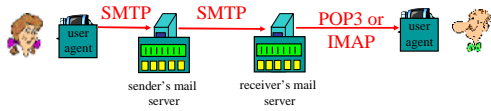
CS 640

SMTP: final words

- SMTP uses persistent connections
 - SMTP requires message (header & body) to be in 7-bit ASCII
 - certain character strings not permitted in msg (e.g., CRLF.CRLF). Thus msg has to be encoded (usually into either base-64 or quoted printable)
 - SMTP server uses CRLF.CRLF to determine end of message
- Comparison with http:**
- http: pull
 - email: push
 - both have ASCII command/response interaction, status codes
 - http: each object encapsulated in its own response msg
 - smtp: multiple objects sent in multipart msg

CS 640

Mail access protocols



- SMTP: delivery/storage to receiver's server
- Mail access protocol: retrieval from server
 - POP: Post Office Protocol [RFC 1939]
 - authorization (agent <->server) and download
 - IMAP: Internet Mail Access Protocol [RFC 1730]
 - more features (more complex)
 - manipulation of stored msgs on server
 - HTTP: Hotmail , Yahoo! Mail, etc.

CS 640

POP3 protocol

authorization phase

- client commands:
 - **user**: declare username
 - **pass**: password

- server responses
 - **+OK**
 - **-ERR**

transaction phase, client:

- **list**: list message numbers
- **retr**: retrieve message by #
- **dele**: delete
- **quit**

```

S: +OK POP3 server ready
C: user alice
S: +OK
C: pass hungry
S: +OK user successfully logged on
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
    
```

CS 640

Spam (a.k.a. unsolicited bulk email)

- Spam filters (e.g. spamassassin)
 - Build statistical model of good & bad messages from user feedback
 - Filter out the bad ones
 - Avoid false positives at all costs (whitelists of senders)
- Blacklisting mail servers sending spam (spamhaus.org)
 - Often distributed through DNS
 - If your server relays any message, the spammers will exploit it
 - SMTP servers can authenticate senders of messages
- Spammers rent networks of zombies from malicious hackers
 - Prefixes with DSL, dialup, cable generally blacklisted
- Spam from hijacked prefixes advertised from hacked routers

CS 640
