COMP 431
Internet Services & Protocols

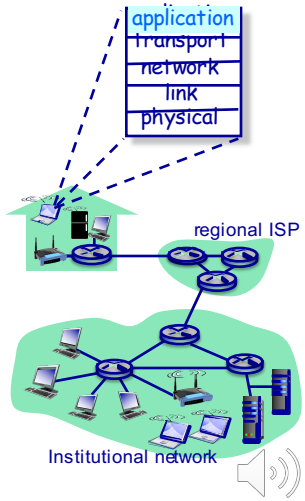
Applications & Application-Layer Protocols: FTP and Email (SMTP & POP)

Jasleen Kaur

February 4, 2020

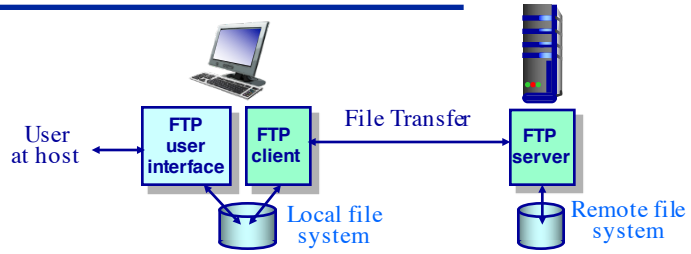
Application-Layer Protocols
Outline

- Example client/server systems and their application-level protocols:
  - » The World-Wide Web (HTTP)
  - » Reliable file transfer (FTP)
  - » E-mail (SMTP & POP)
  - » Internet Domain Name System (DNS)
- Example p2p applications systems:
  - » BitTorrent
- Other protocols and systems:
  - » Streaming media — DASH
  - » Content delivery networks (CDNs)



Application-Layer Protocols

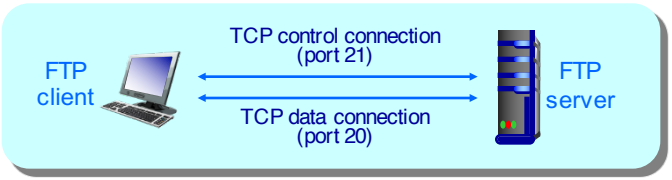
FTP: The Internet file transfer protocol (RFC 959)



- FTP is used to transfer a file to/from remote host
- FTP uses a client/server model
  - » Client: side that initiates transfer (either to/from remote)
  - » Server: remote host
- FTP server listens for connections on port 21

FTP Protocol Design

Control and data sockets



- FTP client contacts FTP server on port 21, using TCP as the transport protocol
- Two parallel TCP connections opened:
  - » A *control* connection for exchanging commands, responses ("out of band control")
  - » *n* data connections for transferring file data to/from server
- FTP server maintains "state"
  - » Remembers current directory, earlier authentication

FTP Protocol Design

FTP commands, responses

- Sample commands:
  - » Sent as ASCII text on control socket

```
USER <username>
PASS <password>
LIST
    Return list of file in current directory
RETR <filename>
    Retrieves (gets) file
STOR <filename>
    Stores (puts) file onto remote host
```

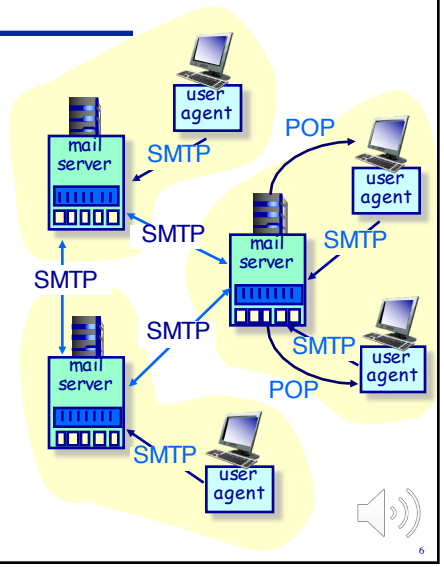
- Sample return codes
  - » Status code and phrase (as in HTTP)

```
331 Username OK, password required
125 data connection already open;
    transfer starting
425 Can't open data connection
452 Error writing file
```

Application-Layer Protocols

Electronic mail

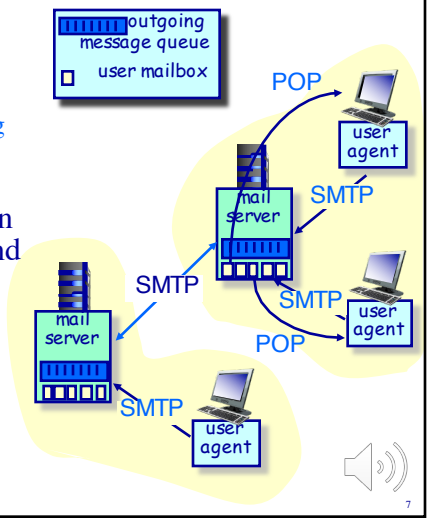
- Major components:
  - » User agents
  - » Mail servers
  - » Mailboxes
- Protocols:
  - » Simple Mail Transfer Protocol (SMTP) delivers mail to servers
    - ❖ From clients to local mail server
    - ❖ Inter-mail server delivery
  - » Post Office Protocol (POP) for user access to delivered email
    - ❖ (Also IMAP! More later…)



Electronic Mail

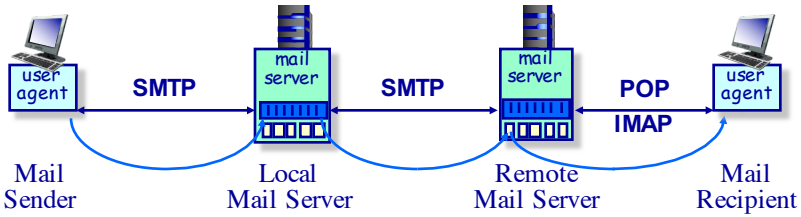
Mail servers

- Servers maintain:
  - » A message queue of outgoing email messages
  - » A mailbox containing incoming messages for each user
- SMTP protocol is run between mail agents and servers to send email messages
  - » Client — the sending mail server or agent
  - » Server — the receiving mail server



Electronic Mail

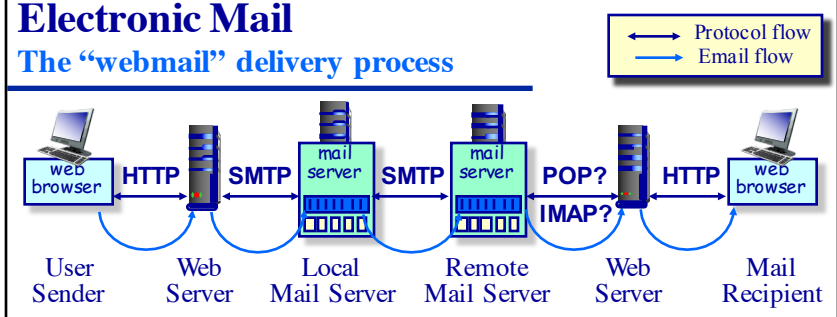
The email delivery process



- User's mail agent contacts its local mail server
- Local mail server contacts the destination mail server(s)
- Destination mail server places the mail into the appropriate user's mailbox
- User retrieves mail via a mail access protocol

Electronic Mail

The "webmail" delivery process



- User's browser sends components of email message via HTTP to a "webmail" server
- Web server is either also an SMTP server or it contacts its local mail server
- To read mail, the mail access protocol is ultimately HTTP

The Email Delivery Process

SMTP [RFC 821]

- SMTP uses a TCP socket on port 25 to transfer email reliably from client to server
- Email is temporarily stored on the local server and eventually transferred directly to receiving server
  - » Intermediate relay is a special case
- Three phases of the protocol:
  - » Handshaking ("greeting")
  - » Transfer of messages
  - » Closure
- Client/server interaction follows a command/response paradigm
  - » commands: ASCII text <CRLF>
  - » response: status code and phrase <CRLF>
  - » Command and response lines terminated with **CRLF**
- messages must be in 7-bit ASCII

The Email Delivery Process

Sample SMTP interaction

- SMTP client establishes TCP connection to server hamburger.edu at port 25
  - » (SMTP is non-standard in that the server "talks first")

```
Server: 220 hamburger.edu
Client: 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
```

Line with single '.' is the message delimiter

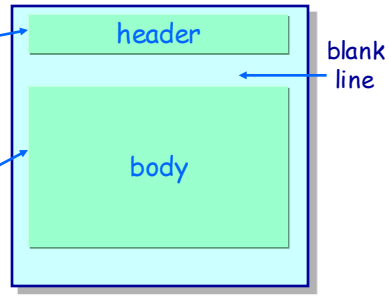
Electronic Mail

Mail message format (RFC 822)

- Header lines, e.g.,
  - » From:
  - » To:
  - » Subject:

these are different from SMTP commands!

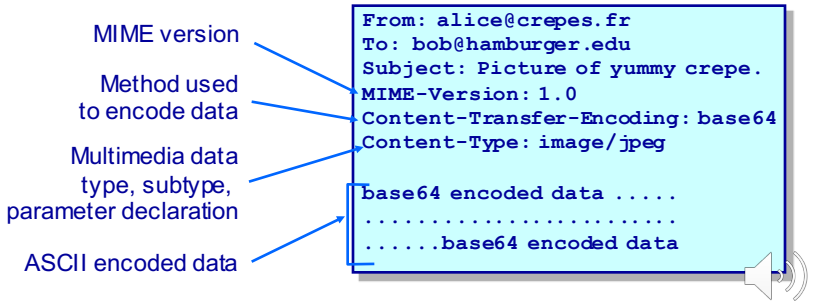
- Body
  - » The "message", ASCII characters only



Mail Message Format

MIME — Multimedia mail extensions (RFC 2045, 2056)

- SMTP requires all data to be 7-bit ASCII characters
  - » All non-ASCII data must be encoded as ASCII strings
- Additional lines in the message header declare MIME content type



MIME Multimedia Mail Extensions

MIME types

```
Content-Type:  <type>/<subtype>[; <parameters>]
```

```
Content-Type: text/plain; charset=us-ascii
```

```
Content-Type: application/pdf; filename=foo.pdf
```

- Text
  - » Subtypes: `plain`, `html`
- Image
  - » Subtypes: `jpeg`, `gif`
- Audio
  - » Subtypes: `basic` (8-bit μ-law encoded), `32kadpcm` (32 kbps ADPCM)
- Video
  - » Subtypes: `mpeg`, `quicktime`
- Application
  - » Other data that must be processed by reader before it is "viewable"
  - » Subtypes: `msword`, `octet-stream`

MIME Types
Multipart Type

MIME version

Header/Body separators

```
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--
```


MIME Types
Multipart Type

MIME version

Indicates multiple, mixed types

Defines parts separator

1 part

1 part

Encoding & type specified for each part

```
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--
```

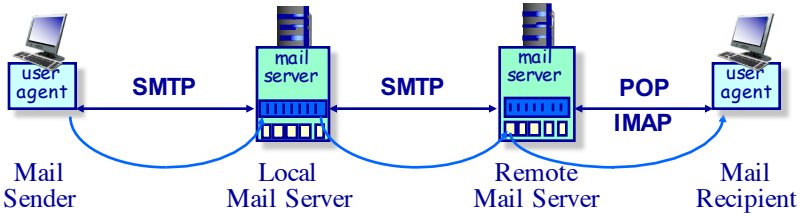
Electronic Mail

SMTP notes

- SMTP uses persistent connections
- SMTP is a "push" protocol
- SMTP requires that message (header & body) be in 7-bit ASCII
  - » All binary objects must be ASCII encoded
  - » Certain character strings are not permitted in a message
  - » Message has to be encoded if these strings are used
- With MIME extensions, multiple objects can be sent in a single multipart message
- SMTP server uses **`CRLF.CRLF`** to determine end of message

Electronic Mail

Mail access protocols



- SMTP: Delivery to receiver's server
- Mail access protocol: Retrieval from server by a user
  - » POP [RFC 1939] — Authorization and download
  - » IMAP (Internet Mail Access Protocol) [RFC 1730]
    - ❖ More features (more complex)
    - ❖ Manipulation of stored messages on server
  - » HTTP: Hotmail , Yahoo! Mail, Gmail, *etc*.

Mail Access Protocols

The POP-3 protocol

- Authorization phase
  - » Client commands:
    - ❖ **`user:`** declare username
    - ❖ **`pass:`** password
  - » Server responses
    - ❖ **`+OK`**
    - ❖ **`-ERR`**
- Transaction phase
  - » **`list:`** list message numbers
  - » **`retr:`** retrieve message by number
  - » **`dele:`** delete
  - » **`quit`**

```
S: +OK POP3 server ready
C: user alice
S: +OK
C: pass hungry
S: +OK

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
```

Application-Layer Protocols

HTTP *v.* SMTP

- HTTP is a "pull" protocol (mostly), SMTP is a "push" protocol
- Persistence:
  - » SMTP uses persistent connections
  - » HTTP may or may not
- Message/object content:
  - » Both have ASCII command/response interaction and status codes
  - » SMTP requires that messages be in 7-bit ASCII — multiple objects message sent in a multipart message
  - » HTTP can transfer anything —each object is encapsulated in its own response headers