CS 43: Computer Networks
Email

Kevin Webb
Swarthmore College
February 15, 2022
Electronic Mail – Major Components

• Mail user agent (MUA)
  • "email client"
  • the software users interact with to send/receive emails
  • takes many forms (Gmail web interface, Thunderbird, phone app, etc.)

• Mail transfer agent (MTA)
  • "email server"
  • takes custody of messages, delivers them

• Simple mail transfer protocol (SMTP)
  • defines message format and transfer procedure
Electronic Mail – Major Components

Mail user agent (MUA)  
"mail client"

Alice
Electronic Mail – Major Components

Mail user agent (MUA)
"mail client"

Mail transfer agent (MTA)
"mail server"

Outgoing queue:

Alice
Electronic Mail – Major Components

Mail user agent (MUA)  
"mail client"

Mail transfer agent (MTA)  
"mail server"

Simple mail transfer protocol  
SMTP

Outgoing queue:
Electronic Mail – Major Components

- Mail user agent (MUA) "mail client"
- Mail transfer agent (MTA) "mail server"
- Outgoing queue:
- Mailboxes:
  - Alice:
  - Bob:
Mail Servers: Ever Vigilant

• Always on, because they always need to be ready to accept mail.

• Usually owned by ISP
  • You use the email server for Swarthmore College (outsourced to Google / Gmail)
Same organization, one server

Mail user agent (MUA)
"mail client"

Mail transfer agent (MTA)
"mail server"

Outgoing queue:

Mailboxes:

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Same organization, one server

Mail user agent (MUA)  
"mail client"

Mail transfer agent (MTA)  
"mail server"

Simple mail transfer protocol  
SMTP

Outgoing queue:

Mailboxes:
Alice: [ ] [ ] [ ] [ ]
Bob: [ ] [ ] [ ] [ ]
Same organization, one server

Mail user agent (MUA) "mail client"

Mail transfer agent (MTA) "mail server"

Outgoing queue:

Mailboxes:

Alice:

Bob:
Same organization, one server

- Mail user agent (MUA) "mail client"
- Mail transfer agent (MTA) "mail server"

Outgoing queue: (NOT SMTP)

Mailboxes:
- Alice: [ ] [ ] [ ] [ ]
- Bob: [ ] [ ] [ ] [ ]

Other protocols

Alice
Bob
What should happen if Alice sends an email to Bob, but they're in different organizations (different mail servers)?

A. Her mail **client** sends a message to his mail **server**

B. Her mail **server** sends a message to his mail **server**

C. Her mail **server** sends a message to his mail **client**

D. Her mail **client** sends a message to his mail **client**
Different organizations, two servers

(1) Alice composes message in mail client.
(2) Alice's client transfers message to her server, via SMTP, where it goes in a queue.
Different organizations, two servers

(3) Alice's server uses SMTP to transmit message to Bob's server.

If this fails for some reason, Alice's server will try again (details depend on server configuration).

Mail user agent (MUA) "mail client"

Mail transfer agent (MTA) "mail server"

Mailboxes:

Alice:  

Outgoing queue:

SMTP

Mailboxes:

Bob:  

Outgoing queue:
Different organizations, two servers

(4) Bob's server places the message in Bob's mailbox, waiting for him to retrieve it.

(5) Eventually, Bob comes along asking for new messages.
Simple Mail Transfer: SMTP [RFC 2821]

• Uses TCP to reliably transfer email message from client to server, port 25
• Direct transfer: sending server to receiving server (no intermediate servers)
• Three phases of transfer
  • handshaking (greeting)
  • transfer of messages
  • closure
• Command/response interaction (like HTTP, FTP)
  • commands: ASCII text
  • response: status code and phrase
• Messages must be in 7-bit ASCII
SMTP Message Format

RFC 822: standard for text message format:

- header lines, e.g.,
  - To:
  - From:
  - Subject: different from SMTP MAIL FROM, RCPT TO: commands!
- Body: the “message”
  - ASCII characters only
  - Signal EOM with “\r\n\r\n”
SMTP messages

• HELO <domain> (or) EHLO <domain>
  • (greeting) identify the sender's domain name

• MAIL FROM:<address>
  • Identify sender's address

• RCPT TO:<address> (can repeat)
  • Identify recipient address(es)

• DATA
  • (message goes here)

• CRLF.CRLF
  • End of SMTP phase, message data comes next

• CRLF.CRLF
  • End of message data (done)
RFC 822: standard for text message format:

• header lines, e.g.,
  • To:
  • From:
  • Subject:
    different from SMTP MAIL FROM, RCPT TO: commands!

• Body: the “message”
  • ASCII characters only
  • Signal EOM with “\r\n.\r\n”
Try SMTP interaction for yourself:

- `telnet allspice.cs.swarthmore.edu 25`
- You should see a 220 reply from the server.
- enter HELO, MAIL FROM, RCPT TO, DATA, QUIT commands

(lets you send email without using email client (MUA))
Demo
Sample SMTP interaction

$ telnet allspice.cs.swarthmore.edu 25
Trying 130.58.68.9...
Connected to allspice.cs.swarthmore.edu
220 allspice.cs.swarthmore.edu ESMTP Postfix
HELO cs.swarthmore.edu
250 allspice.cs.swarthmore.edu
MAIL FROM:<kwebb@cs.swarthmore.edu>
250 2.1.0 OK
RCPT TO:<kwebb@cs.swarthmore.edu>
250 2.1.5 OK
DATA
354 End data with <CR><LF>.<CR><LF>
To: Kevin Webb <kwebb@cs.swarthmore.edu>
From: Kevin Webb <kwebb@cs.swarthmore.edu>
Subject: Telnet test message

This is a test message, via telnet, to myself.
.

Sample SMTP interaction

$ telnet allspice.cs.swarthmore.edu 25
Trying 130.58.68.9...
Connected to allspice.cs.swarthmore.edu
220 allspice.cs.swarthmore.edu ESMTP Postfix
HELO cs.swarthmore.edu
250 allspice.cs.swarthmore.edu
MAIL FROM:<kwebb@cs.swarthmore.edu>
250 2.1.0 OK
RCPT TO:<kwebb@cs.swarthmore.edu>
250 2.1.5 OK
DATA
354 End data with <CR><LF>.<CR><LF>
To: Kevin Webb <kwebb@cs.swarthmore.edu>
From: Kevin Webb <kwebb@cs.swarthmore.edu>
Subject: Telnet test message

This is a test message, via telnet, to myself.
.
End of message.
What keeps us from entering a fake information (e.g., FROM address)?

A. Nothing.

B. The MTA checks that the FROM is valid.

C. We enter a name/password logging into the MTA.
Fun Demo
Wait...why does this work?

(3) Alice's server uses SMTP to transmit message to Bob's server. If this fails for some reason, Alice's server will try again (details depend on server configuration).
This seems too horrible to be true. Surely we can prevent header forging? (How or why not?)

A. Yes

B. No
Message Signing

1. Sender creates cryptographic public/private key pair, publishes public key to the world

2. Sender uses private key to sign messages

3. Receiver can verify*, using published public key, that only the holder of the corresponding private key could have sent the message

* With very high probability.
Message Signing: Challenges

• Disseminating public keys
  • How do you trust that the published public key isn’t also a lie?

• It’s more work, can’t be bothered...
  • Adoption is very low
Server to Server Verification

• Published lists of "bad" IP addresses for blocklists

• LOTS of other techniques
  • See: https://en.wikipedia.org/wiki/Anti-spam_techniques

• Big problem: false positives
SMTP versus HTTP

• HTTP: pull
• SMTP: push

• Both have ASCII command/response interaction, status codes
• HTTP: each object encapsulated in its own response message
• SMTP: multiple objects sent in multipart message
SMTP: final words

• SMTP uses persistent connections
  • Can send multiple emails in one session

• SMTP requires message (header & body) to be in 7-bit ASCII

• SMTP server uses CRLF.CRLF to determine end of message
If SMTP only allows 7-bit ASCII, how do we send pictures/videos/files via email?

A. We encode these objects as 7-bit ASCII

B. We use a different protocol instead of SMTP

C. We’re really sending links to the objects, rather than the objects themselves
Base 64

• Designed to be an efficient way to send binary data as a string

• Uses A-Z, a-z,0-9, “+” and “/” as digits

• A number with digits $d_n d_{n-1} \ldots d_1 d_0 =$
  $64^n d_n + 64^{n-1} d_{n-1} + \ldots + 64 d_1 + d_0$

• Recall from CS 31: Other non-base-10 number systems (binary, octal, hex).
Multipurpose Internet Mail Extensions (MIME)

• Special formatting instructions

• Indicated in the header portion of message (not SMTP)
  • SMTP does *not* care, just looks like message data

• Supports
  • Text in character sets other than ASCII
  • Non-text attachments
  • Message bodies with multiple parts
  • Header information in non-ASCII character sets
MIME

• Adds optional headers
  • Designed to be compatible with non-MIME email clients
  • Both clients must understand it to make sense of it

• Specifies content type, other necessary information

• Designates a boundary between email text and attachments
Mail Access Protocols

• Post Office Protocol: authorization, download

• Internet Mail Access Protocol (IMAP): more features, including manipulation of stored messages on server

• Webmail: gmail, Hotmail, Yahoo! Mail, etc.

• Proprietary protocols (Microsoft Exchange)
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 number
- **dele**: delete
- **quit**

---

S: +OK POP3 server ready
C: user bob
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
More about POP3

• Previous example uses “download and delete” mode
  • Bob cannot re-read e-mail if he changes client

• POP3 “download-and-keep”: copies of messages on different clients

• POP3 is stateless across sessions

• Limitations:
  • Can’t retrieve just the headers
  • Can’t impose structure on messages
**IMAP**

- Keeps all messages in one place: at server

- Allows user to organize messages in folders

- Keeps user state across sessions:
  - names of folders and mappings between message IDs and folder name

- Can request pieces of a message (e.g., text parts without large attachments)
Webmail

• Uses a web browser

• Sends emails using HTTP rather than POP3 or IMAP

• Mail is stored on the (often 3rd party) webmail servers
Summary

• Three main parts to email:
  • Mail User Agent (mail client): read / write for humans
  • Mail Transfer Agent: server that accepts / sends messages
  • SMTP protocol used to negotiate transfers

• No SMTP support for fraud detection
  • Admins do many anti-spam things in practice, worry about false positives

• Extensions (MINE) and encodings (Base64) for sending non-text data