SMTP and Internet EMAIL

TCP/IP class

Jim Binkley

outline

- intro
- X.400 - a small diversion
- SMTP
  - protocol
  - typical architecture/DNS/SMTP trace
  - sendmail config - not talking about it here
- extending SMTP - MIME, etc.
- secure email

intro

- email is ubiquitious, conventional, essential
- much expected and unexpected can be done
  - conventional email
  - paging, 1-way messages
  - mailing lists: moderated, unmoderated
  - agents filtering NEWS or who knows what
  - attachments of non-plaintext binary stuff including postscript/images/viruses

intro

- SMTP protocol been around for a while - Simple Mail Transport Protocol
- RFC 821, Postel, 1982 specifies SMTP
- RFC 822, Crocker, specifies format of mail message
- **7-bit ASCII characters only**
- recently extended to deal with multi-part MIME which can include encoded binary data
- MIME - Multipurpose Internet Mail Extensions, RFC 1521-1524, 1993. New RFCs appearing as MIME extended

store and forward

- one sometime hears that there are 3 basic comm. mechanisms, circuits, packets, store and forward
- store means Mail Transfer Agents enqueue mail until it can be delivered
- application layer of course, not net/link layers
- reliable xfer over TCP, end to end
- messages may be sent to multiple recipients but that is done as mailing list expansion, multiple TCP connections

mobility

- email is possibly the perfect application to deal with mobile computers
- transfer agent can queue email during periods of disconnection
- send it when connected

X.400 (a little background)

- part of ISO/OSI system, Message Handling System or MHS
- invented after SMTP
- aka CCITT X.400, aka MOTIS, ISO10021, etc.
- 1984 version done before ASN/X.500
- 1988 version uses ASN/X.500 distinguished names
- mail gateway possible between SMTP and 88 X.400, see rfc 1327

functional parts

- end user identifier by O/R (originator/recipient name), assume X.500 DN (distinguished name)
- C=US/ADMD=ATTMAIL/PRMD=DNA6L/ORG=UNISYS/PN=ShelbyFoote
- user agent (UA), sw app that sends/recv/stores messages
- message transfer agent (MTA), stores and forwards the message handed to it by UA to remote MTA
- message transfer system (MTS), MTAs + UAs cooperating together to make a mail system

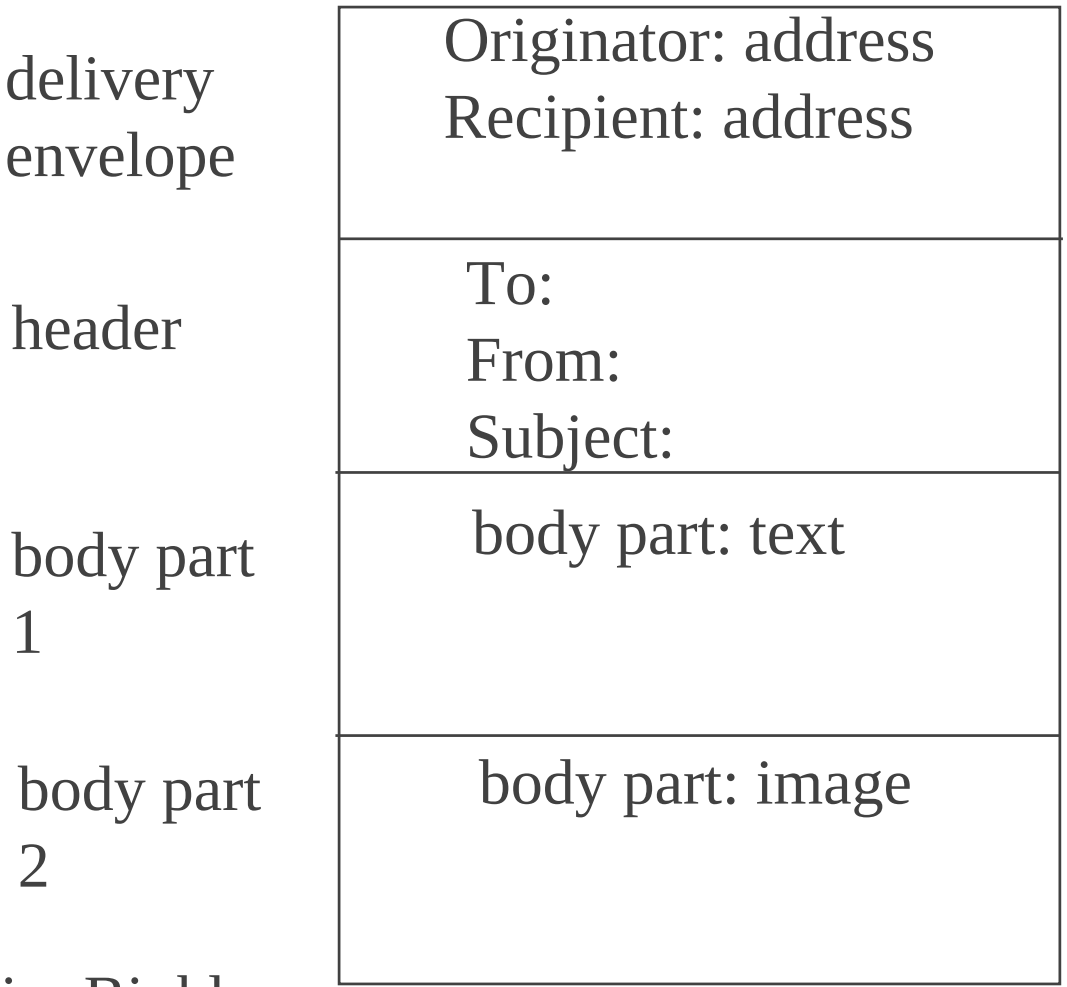
functional parts, cont.

- UA-> MTA via submission and delivery protocol
- MTA->MTA via message transfer protocol

high-level overview items

- reliable and connection-oriented service
- mail may have multiple body parts (text + pictures)
- parts are typed (header/body, header/body)
- mail routing done by embedding info in O/R name or may be obtained from X.500

X.400 mail architecture



SMTP - Simple Mail Xfer Protocol

- very simple protocol, 7-bit ASCII chars
- uses TCP, port 25
- architecture similar to X.400 at this point
- MTA administration can be complex; e.g., UNIX BSD sendmail config files
- typically uses DNS MX records
- MIME extensions allow “multimedia” mail

request/response protocol

- HELO client.dns.name - say hello to other side
  - 250 server-dns “Howdy”
- MAIL From: user@dns-site - id originator
- RCTP To: user@dns-site - id recipient
- DATA
  - text
  - . - DOT in column 1 for EOF
- QUIT - end of transmission

other commands

- RSET - reset connection
- VRFY jrb - verify, expand on server-side
- EXPN mail-list-name - expand mail list
- NOOP
- HELP

- VRFY, EXPN may not be available since they are security holes

SMTP trace

```
% telnet localhost 25
220 rigel.cs.pdx.edu Sendmail 4.1/pdx-...
help
214-Commands:
214-    HELO MAIL  RCPT  DATA RSET
214-    NOOP QUIT  HELP  VRFY EXPN
blah blah
214 End of HELP info
helo localhost
250 rigel.cs.pdx.edu Hello localhost, pleased to meet
        you
rcpt to:jrb
```

SMTP trace, cont.

250 jrb... Recipient ok
mail from elmer@cwazy.wabbit.farm
350 elmer@cwzy.wabbit.farm... Sender ok
data
354 Enter mail, end with “.” on a line by itself
I’m going to XXX that cwazy wabbit!!!
.
250 Mail accepted
quit
221 rigel.cs.pdx.edu delivering mail

SMTP trace - the result

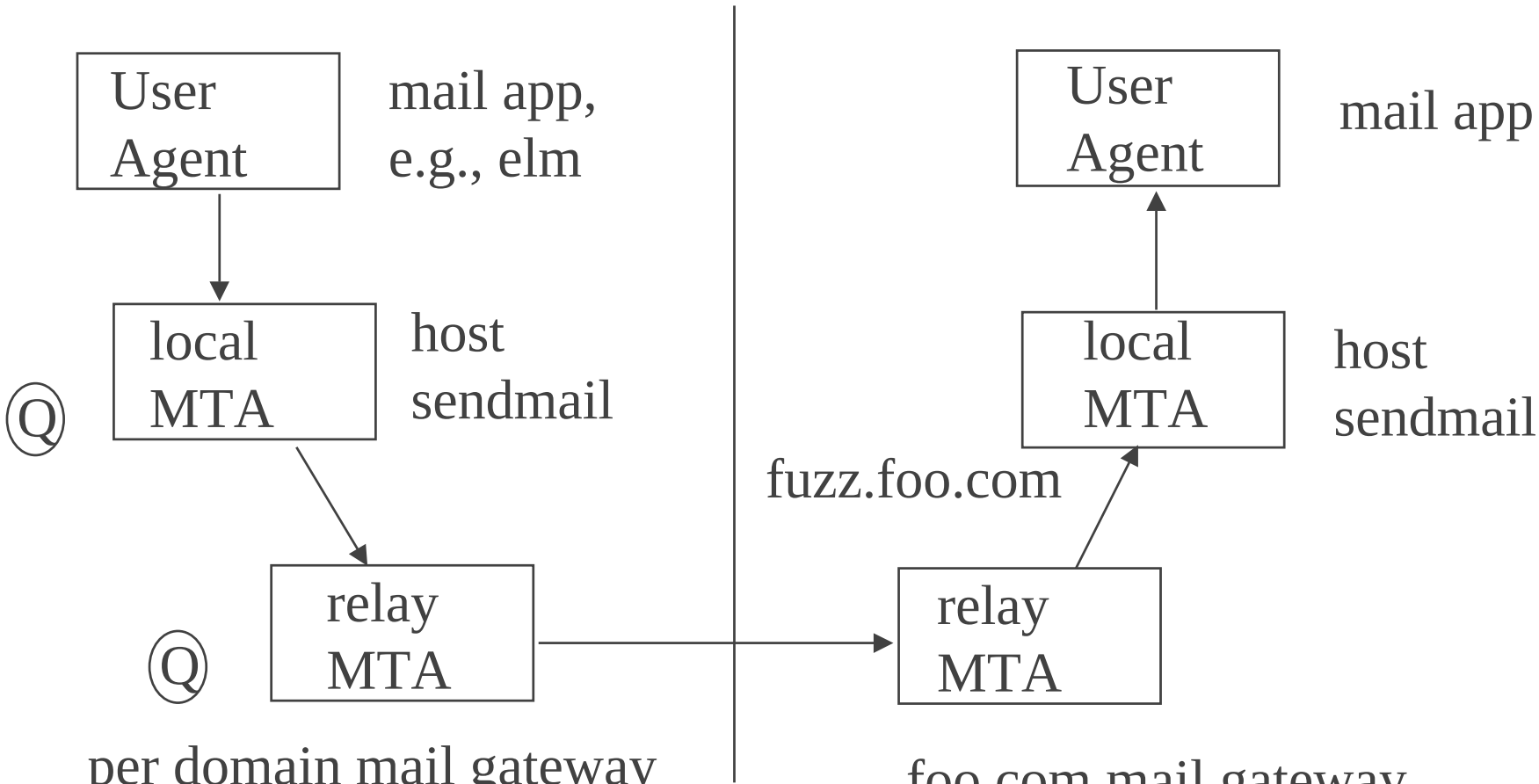
- on UNIX system with MH mail client
- % *show 3*
  Received: from localhost by rigel.cs.pdx.edu
  Date: Sun, 22 Nov 92 11:53:11 PST
  From: elmer@cwazy.wabbit.farm
  Apparently-To: jrb

  I’m going to XXX that cwazy wabbit!!!

note: email relay considered harmful

- in sendmail versions prior to 8.9.0 was ON by default
  - sender launder through 3rd party (2 victims including relay site and receiver)
  - could be turned off thou via various hacks
- in sendmail versions >= 8.9.0 relay is configured off and must be turned on
  - need it for pop but can confine to site IP addresses/names only

SMTP architecture (generalized)



mail “roles”

- User Agent - send/recv mail, store in “folders”, handle MIME attachments
- local MTA - store incoming mail in queue, send outgoing mail via SMTP to local mail gateway or direct to remote MTA
- relay (gateway) MTA - act as **centralized** mail gateway for a domain, talk to remote MTAs, local MTAs
  - true gateway function to X.400, UUCP, etc.

MTAs - some operation details

- queue mail
- retry after 30 minutes
- shouldn’t give up for 4-5 days
- goal: extreme reliability (nevermind rain, sleet, snow, pit bulls, etc...)
- MX record can be used for mail gateway on Internet, gateway can then forward to hosts behind a firewall

mail structure

- 3 parts (barring MIME)
- envelope, MTAs use it, see RFC 821
  Mail From: jrb@cs.pdx.edu
  Rcpt To: tulp@cvitoa.ns.nl
- header (some, not all possible fields)
  From:
  To:
  Subject:
  X-means user-agent defined field. Otherwise see RFC 822
- body, blank line after header. Line < 1000 bytes.

User Agents

- UNIX
  - MH Mail, many small commands including
    - » inc, comp, show, rmm (read my mail :->)
  - elm
  - pine
- PC
  - eudora (only talking about SMTP here)
  - per vendor mail apps

MTAs

- smail/qmail - replacement for
- sendmail
  - /etc/sendmail.cf - config file is grammar
  - hard to setup, see LARGE ORA book, or Nemeth
  - typically aim workstations at one gateway so that said workstations all have same simple sendmail config file, gateway complex
- PCs may use Post Office Protocol (POP) to allow login to local UNIX server to process recv. mail

MTA and mail forwarding

- DNS maintains MX record as “alias” for mail gateway
- e.g., map jrb@foo.com to realbox.foo.com
- MTA will take address, 1st try MX lookup
- fallback on normal lookup (jrb@a.foo.com)
- eventually make TCP connection to port 25 and use SMTP to send mail

SMTP extensions

- uses NVT ASCII, 7-bit character code, bit 8 must be 0
- how to extend and send binary data?
- answer: encode in 7-bit ASCII encoding...

RFC 1425 - ESMTP

- RFC 1425, 1993, defines framework for extending SMTP (Extended SMTP)
- client uses EHLO instead of HELO, server responds with 250 if it can do it
- can now use SIZE to verify that server can handle large mail letter as opposed to running out of disk storage (one way to lose mail)
- 8BITMIME - can negotiate use of 8 bit chars

RFC 1522 - header extensions

- specifies how to put non-ASCII chars in 822 headers
- 2 encodings
  - Q encoding - quoted-printable, intended for Latin character sets, intended for mostly ASCII with a few special chars
  - **base 64**, 3 bytes of text encoded as 4 6-bit values (3*8=4*6)

MIME - Multipurpose Internet Mail Extensions

- RFC 1521, 1993 - defines extensions that allow the body to deal with non-ASCII binary data
- thus we can include images/audio data in email
- justs adds new header info, body still transmitted in NVT ASCII since gateways require that
- previous extensions like SIZE useful but not required
- email can be multi-part, e.g., text/image/sound all in one envelope. clients must support it

MIME header

- header includes following for MIME mail:

  Mime-version: 1.0
  Content-Type: TEXT/PLAIN; charset=US-ASCII
  Content-Transfer-Encoding:
  Content-ID:
  Content-Description:

- e.g., image:

  Content-type: image/gif

- image might follow in base64 encoding

sample MIME types

text/plain
text/richtext - simple formatting, similar to HTML
multipart/mixed - multiple body parts
multipart/alternative - all parts have same semantic content
message/rfc822 - encapsulated mail
message/external-body - “pointer” to external message
application/postscript
image/jpeg
image/gif
audio/basic - sound file
video/mpeg

MIME example - multipart

Mime-Version: 1.0
Content-Type: Multipart/Mixed; Boundary="NextPart"
To: IETF-Announce

--NextPart

boring ASCII text.... blah blah yadda yadda

-- NextPart
Content-Type: Multipart/Alternative Boundary="
OtherAccess"

MIME example - fetch from mail server

--OtherAccess
Content-Type: Message/External-body;
access-type="mail-server";
server="mailserv@ds.internic.net"

Content-Type: text/plain
Content-ID: <19951114160051.I-D@CNRI.Reston.VA.US>

ENCODING mime
FILE /internet-drafts/draft-ietf/idmr-traceroute-ipm-00.txt

MIME example - ftp access

--OtherAccess
Content-Type: Message/External-body;
name="draft-ietf-idmr-traceroute-ipm-00.txt";
site="ds.internic.net";
access-type="anon-ftp";
directory="internet-drafts";

Content-Type: text/plain
Content-ID: <199... etc>

--OtherAccess
--NextPart--

MIME example - summary

- 2 parts
  - ASCII summary
  - way to fetch the file itself
- the 2nd part is “alternative”, two ways to fetch the same file
  - email-server
  - anonymous ftp

richtext?

- defined in MIME rfc 1341
- attempt to allow limited, simple formatting -- improvement on plain text
- precursor to HTML
- uses SGML tags
- <BOLD> bold </BOLD>
- <CENTER> centered! </CENTER>

UNIX metamail - MIME starter kit

- traditional UNIX mail apps like MH/elm are text-based, although X-based versions may be available
- metamail from Bellcore exists as mechanism to try and “simply” add MIME capabilities to such mail agents
- might also support MIME in news readers too
- metamail = mailcap configuration file plus a set of utilities
- on other systems, or UNIX, 3rd-party mail app may exist that offers tightly integrated MIME support
- metamail system -- loosely integrated at best

metamail operation - reception

- mailcap configuration file - matches up mime types with commands that should be executed when data of that sort shows up
- mail app can just call
  % metamail 822-message
- metamail(1) will consult the mailcap config file and carry out an action on the file
- might play audio file, run mpeg movie

mailcap file - /etc/mailcap

- sample mailcap entries:

```
audio/*;  showaudio %s
image/*; xv %s
application/postscript; lpr %s

OR

application/postscript;  ghostview %s
```

metamail app set (some, not all)

- *metamail 822-msg* - carry out actions on file
- *audiosend* - read audio and mail it
- *mailto* - BSD-like mail app, can do MIME things too
- *metasend* - program to take files, tack on MIME image and send as MIME message
- *mmencode* - encode/decode files, default is base64, use -u switch to decode
- *richtext* - display richtext
- *showaudio* - “show”, really play audio on Sun /dev/audio
- *showpicture* - display image

secure SMTP email

- two systems commonly mentioned:
  - **PEM - privacy enhanced mail**, IETF RFCs 1421-1423
  - **PGP - pretty good privacy**, Phillip Zimmerman
- both feature use of **public-key encryption**
- major difference is attitude towards **KEY distribution**
- **how to distribute KEYS?** (how to trust who you get KEY from?) (not just email problem)

private-key encryption

- or **symmetric crytography**
- encrypt(key, plaintext msg) -> cypertext
- decrypt(key, cyphertext msg) -> plaintext
- key length is important, DES 56 bits
  - # of atoms in earth, 2 ** 170
- example algorithms, DES (56), IDEA(128)
- “key” problem, how does key get to both sides (over the phone?) - so called **“shared secret”**
  - you don’t send it plain over the net...

one-way keyed “hash” function

- important for a number of reasons including: **authentication - making sure a message is from a particular user**
- f(msg) -> bit string (digest) OR shared secret
- f(secret key, msg) -> bit string (digest)
- append bit string to msg and send it
- easy to compute, but digest unique per msg (can’t reverse it)
- MD5, or “message digest 5”, has result 128 bits

hash functions and keys

- if we have a key as a shared secret, then both sides can guarantee that the message was indeed not tampered with
- otherwise it is just a checksum
- sometimes useful for distinguishing objects like remote files (2 ** 128 is a lot of bits)
- used with public-key crypto where 1. compute hash over msg, 2. sign hash

public-key crypto

- two keys, public+private
- can’t deduce private from public
- RSA (one algorithm) owns technology (patents) in USA until year 2000
- sometimes called asymmetric cryptography
- RSA algorithm can also do **digital signature** in addition to encryption
- 4 ops: encrypt/decrypt/sign/verify signature
  - authentication is thus a capability too

how to use it with email

- assume party A and party B (alice, bob)
- alice wants to send a secure message to bob
- she *somehow* obtains bob's public key
  - she CANNOT use his private key ...
- encrypts message with bob's public key, sends it
- bob recvs and use private key to decrypt
- pro: no shared secret
- con: still must somehow give out public keys in secure manner without 3rd party saying "here is bob's keys" when it is really "evil fred's" key

encryption, cont.

- may use bulk encryption algorithm like DES for body of message
- use RSA to encrypt DES key which is included
- computational overhead of public key crypto is high - use of DES for efficiency

digital signature

- alice takes private key and produces hash from message input, appends signature value
- sends message (msg, signature)
- bob can use alice's public key and verify that the msg is authentic
  - bob must 1st "somehow" obtain alice's public key FROM ALICE !!!
- public key may be signed in turn by 3rd party
  - recursive verification process

certificate

- signed (by 3rd party “authority) public key
- roughly: (public key, name, 3rd party signature)
- in order to verify you must somehow obtain 3rd party public key
  - network protocol or floppy disk
- msg could then be (data, public key certificate, (my) signature)
- assumption being that you 1st verify authority signature of certificate, and then use user pub key

stumbling block: key distribution centers

- could acquire KEYS from database, but database has to be trusted (keys might be symmetric or public)
- what if I substitute my public KEY for your desired party -- then I can read his messages
- assume “his” == “bob”
- **Key Distribution Center** could sign public key for “bob” with their private key
- when get message, acquire KDC public key and verify signature
- real issues are social, legal, AND technical too

PEM basics - Privacy Enhanced Mail

- RFCS 1421-1424
- PEM message is always authenticated, may be encrypted
- encryption done with DES-CBC
- either public/private key encryption can be used
- however if public key mgmt used - RSA algorithm +
  - assume certificates and certificate hierarchy
  - authentication hash of message signed with sender's private key
  - if encryption used, done with DES and session key which is encrypted with recv public key

more PEM

- assumption is that private-key only PEM won’t be used
- message encryption done with private-key for reasons of efficiency (key is enclosed and encrypted)
- PEM mail is ASCII-only and not very readable either
- certification authority is racy assumption

Certificate Authorities

- with PEM, you could invoke non-existent certification protocols to talk to non-existent CA server hierarchy
- CA hierarchy supposed to use X.500 naming
- have super-secure servers at top, run by whom?
- or you can include the certificates
- or have them already
- or implementation could be told to not use them

a word from Ancient Rome

“Sed quis custodiet ipsos custodes?”

Juvenal

not: “who cleans up after the custodians” ...

(thanks to Dave Aucsmith)

PEM history ?

- stuck on certificate problem currently
- TIS - PEM (Trusted Information Systems)
- RIPEM - Mark Riorden, does not implement certificates

PGP - Pretty Good Privacy

- bypasses CA problem by assuming there isn't one, and you get the key somehow
  - telephone (out of band)
  - finger (throw caution to the winds)
  - trusted 3rd party (or 3rd party human network)
- uses IDEA for encryption, RSA for key management, MD5 as one-way hash function
- less information exposed in header compared to PEM, Pretty Good Engineering...

Phil's Quote

"If privacy is outlawed, only outlaws will have privacy"

P.Z.