Introduction to Internet Mail

Philip Hazel

Mail agents

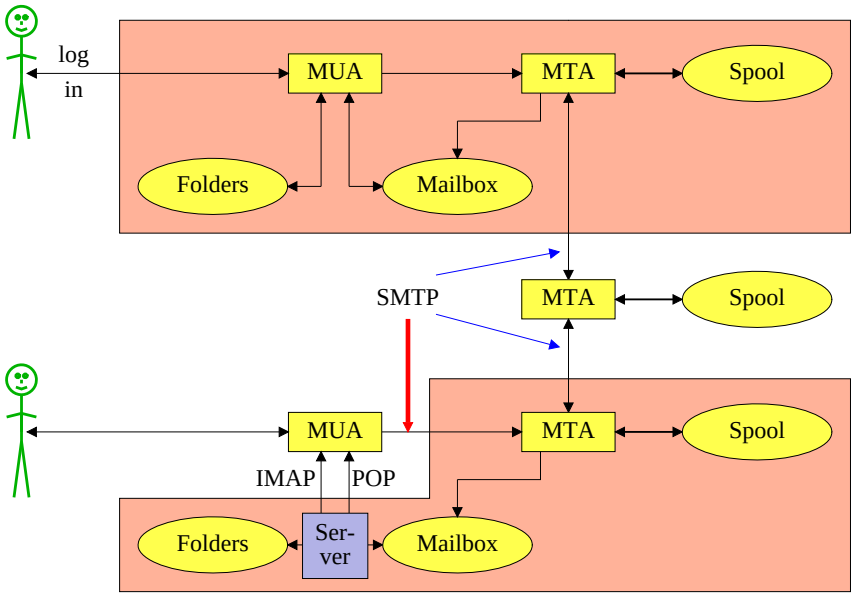
- MUA = Mail User Agent
- Interacts directly with the end user

  Pine, MH, Elm, mutt, mail, Eudora, Mulberry, Pegasus, Outlook, Thunderbird, web browsers ...
- Multiple MUAs on one system – end user choice

- MTA = Mail Transfer Agent
- Receives and delivers messages

  Sendmail, Smail, Exim, qmail, Postfix, ...
- Only one fully active MTA per system – sysadmin choice
- Most MTAs also act as Mail Submission Agents (MSAs)

MUA/MTA Interactions



Authenticating senders

- Embedded MUA uses inter-process call to send to MTA
  - May use pipe, file, or internal SMTP over a pipe
  - MTA knows the identity of the sender
- Freestanding MUA uses SMTP to send mail
  - MUA can point at any MTA whatsoever
  - MTA must distinguish local/remote clients
  - Only “submission” clients are allowed to relay
  - IP identification is no good for roaming clients
  - No authentication in basic SMTP protocol
  - AUTH command in extended SMTP
  - Use of security additions (TLS/SSL)

Message format (1)

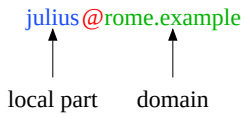
```
From: Philip Hazel <phil@exim.example>
To: Julius Caesar <julius@rome.example>
Cc: Mark Anthony <MarkA@cleo.co.example>
Subject: How Internet mail works

Julius,
  I'm going to be running a course on ...
```

- Format was originally defined by RFC 822 in 1982
  - Now superseded by RFC 2822 (published 2001)
- Message consists of
  - Header lines – some have a well-defined syntax
  - A blank line – terminates the end of the header
  - Body lines
- Notice that a message is defined in terms of **lines**

Message format (2)

- An email address consists of a *local part* and a *domain*



- A basic message body is unstructured ASCII text
- Other RFCs (MIME, 2045) add additional header lines that define structure for the body
- MIME supports attachments of various kinds and in various encodings
- Creating/decoding attachments is really the MUA's job
  - MTAs may have to do it to interface to content scanners

A message in transit (1)

- Headers added by the MUA before sending

```
From: Philip Hazel <phil@exim.example>
To: Julius Caesar <julius@rome.example>
Cc: Mark Anthony <MarkA@cleo.co.example>
Subject: How Internet mail works

Date: Mon, 10 May 2004 11:29:24 +0100 (BST)
Message-ID: <Pine.SOL.3.96.990117111343.
  19032A-100000@taurus.exim.example>
MIME-Version: 1.0
Content-Type: TEXT/PLAIN; charset=US-ASCII

Julius,
  I'm going to be running a course on ...
```


A message in transit (2)

- Headers added by MTAs

```
Received: from taurus.exim.example
  ([192.168.34.54] ident=exim)
  by mauve.csi.example with esmtp
  (Exim 4.30) id 101qxX-00011X-Ab;
  Mon, 10 May 2004 11:50:39 +0100
Received: from phil (helo=localhost)
  by taurus.exim.example with local-smtp
  (Exim 4.31) id 101qin-0005PB-2c;
  Mon, 10 May 2004 11:50:25 +0100

From: Philip Hazel <phil@exim.example>
To: Julius Caesar <julius@rome.example>
Cc: Mark Anthony <MarkA@cleo.co.example>
Subject: How Internet mail works
Date: Mon, 10 May 2004 11:29:24 +0100 (BST)
Message-ID: <Pine.SOL.3.96.990117111343.
  19032A-100000@taurus.exim.example>
MIME-Version: 1.0
...
```

A message in transit (3)

- A message is transmitted with an *envelope*

```
MAIL FROM:<phil@exim.example>
RCPT TO:<julius@rome.example>
```

- The envelope is separate from the RFC 2822 message
- Envelope (RFC 2821) fields need not be the same as the header (RFC 2822) fields (**From:** and **To:**)
- MTAs are (mainly) concerned with envelopes

  Just like the Post Office...

- Error (‘‘bounce’’) messages have null senders

```
MAIL FROM:<>
```

- This is to prevent looping

An SMTP session

```
telnet relay.rome.example 25
220 relay.rome.example ESMTP Exim ...
EHLO taurus.exim.example
250-relay.rome.example ...
250-SIZE 10485760
250-PIPELINING
250 HELP
MAIL FROM:<phil@exim.example>
250 OK
RCPT TO:<julius@rome.example>
250 Accepted
DATA
354 Enter message, ending with "."
Received: from ...
From: ...
etc...
.
250 OK id=10sPdr-00034H-4B
QUIT
221 relay.rome.example closing connection ...
```

SMTP return codes

- 2*xx* OK
  - 220 Service ready
  - 250 Requested mail action okay, completed
- 3*xx* Send more data
  - 354 Start mail input; end with <CRLF>.<CRLF>
- 4*xx* Temporary failure
  - 421 Service not available, closing transmission channel
  - 450 Requested mail action not taken: mailbox unavailable
  - 451 Requested action aborted: error in processing
- 5*xx* Permanent failure
  - 500 Syntax error, command unrecognized
  - 501 Syntax error in parameters or arguments
  - 503 Bad sequence of commands
  - 550 Requested action not taken: mailbox unavailable
  - 554 Transaction failed or no SMTP service here

Email forgery

- It is trivial to forge unencrypted, unsigned mail
- This is an inevitable consequence when the sender and recipient hosts are independent
- Most spam contains forged senders and forged header lines
- Be alert for forgery when investigating
- and ...
- Never send automatic spam or virus warnings!
  - If you do, you are just adding to the problem
  - This is known as ‘‘collateral spam’’ or ‘‘Joe jobs’’

The Domain Name Service

- The DNS is a worldwide, distributed database
- DNS servers are called *name servers*
- There are multiple servers for each DNS *zone*
- Secondary servers are preferably off-site
- Records in the DNS are keyed by type and domain name
- Root servers are at the base of the hierarchy
- Caching is used to improve performance
- Each record has a time-to-live field

Use of the DNS for email (1)

- Three DNS record types are used for routing mail
- *Mail eXchange* (MX) records map mail domains to host names

  They provide a list of hosts, with preferences

```
hermes.cam.ac.uk.  MX 5 green.csi.cam.ac.uk.
                   MX 7 ppsw3.csi.cam.ac.uk.
                   MX 7 ppsw4.csi.cam.ac.uk.
```

- *Address* (A) records map host names to IPv4 addresses

```
green.csi.cam.ac.uk.   A 131.111.8.57
ppsw3.csi.cam.ac.uk.   A 131.111.8.38
ppsw4.csi.cam.ac.uk.   A 131.111.8.44
```

- IPv6 addresses use AAAA (“quad A”) records

```
ahost.csi.cam.ac.uk.   AAAA 2001:630:200:...
```

Use of the DNS for email (2)

- MX records were added to the DNS after its initial deployment
- Backwards compatibility rule

  If no MX records are found

  - Look for an address record
  - If found, treat it as an MX with 0 preference (most preferred)
- MX records were invented for gateways to other mail systems

  They are now used for handing generic (e.g. corporate) mail domains
- SRV (service) records can also be used for email routing

  This feature is not widely deployed

Other DNS records

- The PTR record type maps IP addresses to names
- The IP address is inverted, then looked up in *in-addr.arpa*

```
57.8.111.131.in-addr.arpa.  PTR  green.csi.cam.ac.uk.
```

- PTR and address records do not have to be one-to-one

```
cam.ac.uk.      MX 7  mx.cam.ac.uk.
mx.cam.ac.uk.   A     131.111.8.33

33.8.111.131.in-addr.arpa. PTR ppsw-4m.csi.cam.ac.uk.
```

- CNAME records provide a general aliasing facility

```
pelican.cam.ac.uk.  CNAME  redshank.csx.cam.ac.uk.
```

DNS lookup tools

- *host* is easy to use for simple queries

```
host demon.net
demon.net   A  193.195.224.1
host 193.195.224.1
Name: finch-staff-1.server.demon.net
Address: 193.195.224.1
host -t mx demon.net
demon.net  MX  10 lon1-relay-1.mail.thus.net
demon.net  MX  5 lon1-hub-internal.mail.demon.net
demon.net  MX  5 anchor-hub-internal.mail.demon.net
```

- *nslookup* is more verbose in both input and output

```
nslookup bt.net
nslookup 192.168.34.135
nslookup -querytype=mx bt.net
```

- *dig* is the ultimate nitty-gritty tool

```
dig bt.net
dig -x 192.168.34.135
dig energis.net mx
```


DNS mysteries

- Sometime primary and secondary name servers get out of step
- When mystified, check for server disagreement
- A second argument for *host* specifies a name server

```
host -t ns xxx.ac.uk
xxx.ac.uk  NS  mentor.xxx.ac.uk
xxx.ac.uk  NS  ns0.ja.net

host harvey.xxx.ac.uk  mentor.xxx.ac.uk
harvey.xxx.ac.uk  A  192.168.1.3

host harvey.xxx.ac.uk  ns0.ja.net
harvey.xxx.ac.uk has no A record at ns0.ja.net
  (Authoritative answer)
```

Common DNS errors

- Final dots missing on RHS host names in MX records
- MX records point to aliases instead of canonical names
  - This should work, but is inefficient and deprecated
- MX records point to non-existent hosts
- MX records contain IP addresses (not host names) on the right-hand side
  - Unfortunately some MTAs accept this
  - Also, some name server software conspires to support this
- MX records do not contain a preference value

Routing a message

- Process locally handled addresses
  - Alias lists
  - Forwarding files
  - Local mailboxes
- Recognize special remote addresses
  - For example, those for local client hosts
- Look up MX records for remote addresses
- If ourself (the current host) is in the list with preference *P*
  - Discard MX records whose preference is greater than or equal to *P*
  - This logic is for secondary MX servers
- For each remaining MX record, get the host's IP address(es)

Delivering a message

- Perform local delivery
- For each remote delivery
  - Try to connect to each remote host until one succeeds
  - If it accepts or permanently rejects the message, that's it
- After temporary failures, try again at a later time
- Time out after deferring too many times
- Avoid sending multiple copies of the same message to the same host
  - The RFCs recommend single copies with multiple recipients
  - Sometimes single copies are necessary

Checking incoming senders

- A lot of messages are sent with bad envelope senders
  - Misconfigured mail software
  - Unregistered domains
  - Misconfigured name servers
  - Forgeries – probably the biggest cause nowadays
- Many MTAs check the domain of the sender address
- It is harder to check the local part
  - A reverse SMTP "callout" is needed
  - Uses more resources and can be quite slow
  - Controversial when used indiscriminately
- Bounce messages have no envelope sender; no check is possible

Checking incoming recipients

- Some MTAs check each local recipient during the SMTP transaction
  - Rejections are handled by the sending MTA
  - The receiving MTA avoids problems with bad senders
- Other MTAs accept messages without checking local recipients
  - The checks happen later
  - Errors are handled by the receiving MTA
  - More detailed error messages can be generated ...
    - ... but not necessarily delivered
    - ... or delivered to an innocent 3rd party (collateral spam)
- Checking at SMTP time is nowadays very common (because of forgeries)
  - Reduces collateral spam because ratware does not generate bounces

Relay control

- Incoming: From any host to specific domains
  - Example: incoming gateway or backup MTA
- Outgoing: From specific hosts to anywhere
  - Example: outgoing gateway on local network
- From SMTP-authenticated hosts to anywhere
  - Example: travelling employee or customer using a remote network
- Encryption can be used for password protection during authentication
- Authentication can also be done using certificates
- Any other relaying is ‘‘open’’, and is a Bad Thing

Policy controls on incoming mail

- Block known miscreant hosts and networks
  - Spamhaus project, Realtime Blackhole List (RBL), etc...
- Block known miscreant senders
  - Not as effective as it once was
- Reject SMTP protocol violations
  - Catches some ‘‘pump and dump’’ ratware
- Greylisting – temporarily reject unknown senders
  - Has to be used in conjunction with black and white lists
  - Requires continuous management – not that simple...
- Refuse malformed messages
- Refuse virus-laden messages
- Try to recognize unwanted messages (spam)
  - Discard (danger of false positives)
  - Annotate (let the end user decide)