Email

Timothy Daly

March 16, 2004

Abstract

We examine the RFC standards behind electronic mail. We review a mail server, Postfix, in some detail. We give an example implementation of a program to fetch a single message. We give an example implementation of a program to watch a mail queue.
Contents

1 Weekly News 3

2 The basic idea 8

3 Standards 8
  3.1 Simple Mail Transfer Protocol (SMTP) ......................... 8
    3.1.1 MAIL .................................................. 8
    3.1.2 RCPT .................................................. 9
    3.1.3 DATA .................................................. 9
    3.1.4 Example .............................................. 9

4 Post Office Protocol POP3 10
  4.1 A full session example ......................................... 12

5 IMAP4 13
  5.1 Sample session .............................................. 18

6 Servers 19
  6.1 Mail ..................................................... 19
    6.1.1 Sending Mail .......................................... 19
    6.1.2 Receiving Mail ......................................... 19
  6.2 Postfix .................................................. 20
    6.2.1 Getting Postfix ........................................ 20
    6.2.2 Building and Installing Postfix ......................... 21
    6.2.3 Starting Postfix ....................................... 21
    6.2.4 Stopping Postfix ..................................... 21
  6.3 QPopper [21] ............................................ 21
    6.3.1 Getting the source .................................... 22
    6.3.2 Building and installing the executable .................... 22
    6.3.3 Running popper standalone ................................ 22
  6.4 Sendmail ................................................ 22

7 Clients 22
  7.1 Fetchmail ................................................ 22
  7.2 Evolution ................................................ 24
  7.3 Fetch.java .............................................. 24
  7.4 Biff.java ............................................... 26
1 Weekly News

Microsoft facing European sanctions [1]

Microsoft has broken European Union antitrust law and could face sanctions and hundreds of millions of dollars in fines, according to a draft decision expect to win endorsement on Monday from an advisory committee to the 15 EU states.

Essentially there is a struggle going on worldwide with respect to media and entertainment. The issue is who controls the “channel” between the producer (e.g., the musician) and the consumer. The winner in this game will determine the future direction of all entertainment delivery. Microsoft wants that control.

In order to gain that control position Microsoft is using the same tactic it used to kill off Netscape and gain control of the browser market. Microsoft “bundles” their software into their operating system. They give the software away essentially for free because it is covered by the cost of getting the operating system. Other businesses cannot compete with free and cannot convince the consumer to pay for an add-on program that to do what Microsoft already does. Thus, the competition disappears and Microsoft owns the market. It worked with Netscape (and dozens of other programs) and now it is working against Real, the supplier of media player software.

In particular, against Real, they have used a second predatory tactic. The idea is that Microsoft “partners” with the company and gets a license to use the technology. Once they understand the technology they add it to their own software and then break the partnership. Thus the partner company has lost its differentiating features. This tactic was used against Stalk (maker of doublespace) and Sendo (maker of cell phones) and many others.

Once Microsoft has the technology and has given the technology away for free it then claims that the technology is “embedded” into Windows and cannot be separated. They refuse to ship add-on tools like media players separately.

Thus, Microsoft uses the fact that it has a monopoly on the desktop to kill off competition. The tactics are illegal and Microsoft has twice faced U.S. Federal Antitrust trials, neither of which had any effect.

The European Union sees that a U.S. corporation has the ability to kill off competition in the European market and is moving to stop the behavior. They are insisting that Microsoft ship two versions of the operating system, one with Media Player and one without. Microsoft says it cannot do that.

Rumor has it that Microsoft will force a decision where they get to control the outcome (as happened in the U.S.) to their favor. The likely result
will be that the decision allows Microsoft to offer a “broken” version of 
Windows but only if you buy it directly from the company. The “broken” 
version (non-Media version) will not be pre-installed and will be very hard 
to install (since Microsoft will insist that you get special drivers for your 
hardware from the manufacturer rather than bundling them). Microsoft 
has never lost one of these fights and I don’t believe they will lose this 
one.

In any case, whatever the decision is, there will be an appeal which will 
take several years to adjudicate. By the time a decision is reached Real will 
no longer exist and there will be no media competition. Justice delayed is 
justice denied.

**P2P in the Legal Crosshairs** [2]

A draft letter purportedly circulated by Bill Lockyer to fellow 
state attorneys general characterizes P2P software as a “dangerous 
product” and describes the failure of technology makers to 
warn consumers of those danger as a deceptive trade practice.

However, the metadata associated with the Microsoft Word doc-
ument indicates it was either drafted or reviewed by a senior vice 
president of the Motion Picture Association of America. Ac-
cording to the metadata (automatically generated by the Word 
application), the document’s author or editor is “stevensonv.”

Now Vans Stevenson is the MPAA’s senior vice president for state legisla-
tive affairs.

The idea is that if they can get the free peer-to-peer programs classified 
as a dangerous product they can regulate, fine, and remove such programs 
from the internet. Essentially the game is to make file sharing illegal.

The above letter is being delivered this week at a nationwide conference 
attended by all of the states attorneys general. If they can convince all of 
the states to pass legislation to classify P2P software as dangerous they 
can arrest the authors and users who share and send them to prison.

Your elected officials are watching out for your rights. Not.

**Dept. of Homeland Security Chooses Groove, P2P** [19]

BEVERLY, Mass., February, 26 2004 - Groove Networks Inc., a 
leading provider of secure virtual office software that lets teams of 
people work over the network as if they were in the same loc-
tion, today announced that its software is a core component of 
an information-sharing network that Department of Homeland 
Security Secretary Tom Ridge announced Tuesday, calling it "a 
key part of our national homeland security strategy.” A public 
demonstration of the Homeland Security Information Network
(HSIN) will occur Thursday at the AFCEA Homeland Security Conference at the Ronald Reagan International Trade Center in Washington, D.C.

Marking the one-year anniversary of the Department of Homeland Security, Ridge said HSIN will expand upon the Joint Regional Information Exchange System (JRIES) that includes Groove Workspace as a core component for real-time, secure, intra- and inter-agency collaboration. JRIES has been developed by state and local officials in partnership with the federal government. The exchange system allows multiple federal, state and local agencies and emergency operations centers to receive and share the same intelligence and tactical information, giving everyone involved the same “situational awareness.”

Ridge said HSIN will be expanded to “all 50 states, five territories, tribal governments, and 50 major urban areas” and eventually to the private sector so it can coordinate preparedness efforts with government officials.

Groove software’s unique decentralized architecture provides an agile, secure and extensible collaboration infrastructure to support inter-agency decision-making. Information shared within Groove workspaces is encrypted both ‘over the wire’ and on the user’s hard drive. The software provides secure communication across insecure networks, is self-synchronizing, supports on- and off-line use, and employs a “web of trust” user authentication model.

Groove uses P2P to coordinate their file sharing. P2P may be illegal shortly (because it has no non-infringing uses) but that won’t stop the government from using it (for non-infringing purposes).

**HP Starts Pushing Desktop Linux[20]**

HP sells Linux systems based on the Mandrake distribution. They just don’t mention it. Mandrake is currently in bankruptcy proceedings. HP has an SCO binary-only license and is in a strange limbo of being licensed to run binaries-only but forced to ship sources because of the GPL. But they do sell Linux pre-installed on computers.

And it is on desktop machines so HP has entered the desktop wars. The interesting thing is that large companies want to buy from other large companies who will support their products. HP certainly qualifies as a large company. It should be interesting to see what level of support they provide.

**Privacy Safeguards Deep-Sixed[3]**

Two computer projects designed to preserve the privacy of Americans were quietly killed while Congress was restricting Pentagon
data-gathering research in a widely publicized effort to protect citizens from futuristic anti-terrorism tools.

Admiral John Poindexter created an effort, called Genisys, to scan government and commercial records. He also created an effort, called Bio-ALIRT, to scan medical records. There was a huge data-mining effort called Total Information Awareness which would review all of these records. Both scanning programs were supposed to hide the names of the individuals unless concrete information of a terror link was found. All of these programs were “officially killed” but have been transferred to the intelligence gathering agencies (where work does not get reported) rather than at DARPA (where reports are required).

**Feds Want Wiretap-Ready Net [4]**

Technology companies should be required to ensure that law enforcement agencies can install wiretaps on Internet traffic and new generations of digital communications, the Justice Department says.

The push would effectively expand the scope of the Communications Assistance for Law Enforcement Act, a 1994 law that requires the telecommunications industry to build into its products tools that U.S. investigators can use to eavesdrop on conversations with a court order.

This affects open source programmers in mysterious ways. If you build a program to do VOIP (voice over IP) you may be required to modify the program to make sure the U.S. federal government can monitor your traffic. Thus it MAY be illegal to encrypt the traffic if you don’t provide a way to decrypt it without the user’s consent.

**The SCO-IBM patent battle [5]**

SCO sued IBM for breach of contract. IBM counter-sued for patent violations. IBM has a very wide and deep patent portfolio and selected four patents that apply directly to SCO’s software.

This is really a two-edged sword. IBM uses patents defensively to give it freedom to work. If you try to block IBM they will generally show up with some patents you are violating and then negotiate a settlement that allows them to continue working. Not every company uses patents defensively and there are rumors that Microsoft is amassing a huge software patent portfolio to use against open source.

In general, even though they are being applied to good purpose in this case, I hate patents. I even have a patent and I hate patents.

**The Draft IPR Enforcement Directive – A Threat to Competition and to Liberty**
The EU’s draft Directive on the enforcement of intellectual property rights sets out to make it dramatically easier to enforce copyrights, patents, and trademarks in Europe, and to punish people who tamper with technical mechanisms designed to prevent copying or counterfeiting. The directive has been welcomed by the music and film industries. But it divides the computer industry - Microsoft is for, while Sun is against - and the telecomms industry is strongly opposed. Supermarkets also stand to lose. Resistance is building, for example in the European press. Online liberties are also at risk, as well as commercial interests.

Even the ‘normal’ trademark disputes that arise in business will now become absolutely explosive. An example is the case between Renault and Audi when Renault introduced the ‘Quadra’ and got sued by Audi on the grounds that this model name was too similar to its ‘Quattro’. Audi won (1993). If the directive had been in force then, it would not just have involved some damages and a rebranding exercise: Audi could have forced Renault to withdraw all the vehicles sold in the meantime, then disposed of them by passing them to a charitable organisation or scrapping them, got compensation of Renault’s profits plus double licence fees, etc. This is surely excessive.

Effects of harsher enforcement: In these turbulent waters, the Commission has launched a draft Directive that will criminalise all acts of intellectual property infringement that are carried out deliberately in the course of a business, rather than just serious cases as at present. It will also make generally available some intimidatory techniques that until now have existed only in some jurisdictions - such as the UK’s Anton Piller and Mareva orders, which respectively allow searches and the freezing of bank accounts in civil cases, and a Dutch provision that an infringer can be compelled to recall goods from the market at his own expense. In the UK, where they were invented, Anton Piller orders turned out to be dangerous instruments and open to abuse; as a result, many safeguards have been developed in the UK since its introduction in 1976. The Directive does not compel Member States to enact these safeguards and it is predictable that many will not.

Effects on free software: There are likely victims who cross the boundary between the industrial and the cultural victims. The most obvious of these is the free software community. The main reason that Microsoft is not completely dominant in the operating systems market is the competition from free operating
systems such as BSD and GNU/linux that are maintained by
armies of volunteers. These groups do not really have the re-
sources to defend against large civil suits; a recent action against
Linux by SCO is causing some concern. Until now, Microsoft
has forborne to use its own patent portfolio against its free
competitors, but this may change. Tilting the playing field by
introducing the threat of criminal penalties will make life sig-
nificantly harder for the free software community in the long
term. Many of the developers and maintainers are university
graduate students who treat their work as a training exercise;
however, universities are more risk-averse than commercial ISPs
when faced with the threat of copyright lawsuits (even vex-
atious threats). The elimination of free software would have
serious effects for commercial software based on it (such as Ap-
ple’s OS/X) and would likely result in significant price rises. It
would also threaten large European public investments in soft-
ware based on free platforms.

2 The basic idea

The basic idea of a mail program is that it collect messages from a sender and
forwards them to the receiver in a reliable, time-insensitive way. Since mail is
just a file with an associated destination address the implementation amounts
to file transfer with a few extra commands to specify who the should receive the
file.

3 Standards

3.1 Simple Mail Transfer Protocol (SMTP)

There are three steps to a SMTP transaction. The transaction is started by a
MAIL command which gives the sender identification. Next a series of one or
more RCPT commands follow giving the receiver information. Then a DATA
command gives the mail data. Finally the end of mail data indicator confirms
the transaction.

3.1.1 MAIL

The first step in the procedure is the MAIL command. The

<reverse-path> contains the source mailbox

MAIL<br FROM:<reverse-path><CRLF>

This command tells the SMTP receiver that a new mail transaction is start-
ing.
3.1.2 RCPT
The second step in the procedure is the **RCPT** command

```
RCPT<TO>:<forward-path><CRLF>
```

This command gives a forward-path identifying one recipient. If accepted, the receiver-SMTP returns a 250 OK reply and stores the forward-path. If the recipient is unknown the receiver-SMTP returns a 550 Failure reply. This second step of the procedure can be repeated any number of times.

3.1.3 DATA
The third step in the procedure is the **DATA** command.

```
DATA<CRLF>
```

If accepted, the receiver-SMTP returns a 354 Intermediate reply and considers all succeeding lines to be the message text. When the end of text is received and stored the SMTP-receiver sends a 250 OK reply.

Since the mail data is sent on the transmission channel the end of the mail must be indicated so that the command and reply dialog can be resumed. SMTP indicates the end of the mail data by sending a line containing only a period.

Please note that the mail data includes the memo header items such as Date, Subject, To, Cc, From.

The end of mail data indicator also confirms the mail transaction and tells the receiver-SMTP to now process the stored recipients and mail data. If accepted, the receiver-SMTP returns a 250 OK reply. The **DATA** command should fail only if the mail transaction was incomplete (for example, no recipients), or if resources are not available.

3.1.4 Example

```
S: MAIL FROM: <smith@alpha>
R: 250 OK

S: MAIL FROM: <Jones@beta>
R: 250 OK

S: MAIL FROM: <green@beta>
R: 250 OK

S: MAIL FROM: <brown@beta>
R: 250 OK

S: DATA
R: 354 Start mail input, end with <CRLF>..<CRLF>
S: blah, blah, blah..
```
4 Post Office Protocol POP3

Initially, the server host starts the POP3 service by listening on TCP port 110. When a client host wishes to make use of the service, it establishes a TCP connection with the server host. When the connection is established, the POP3 server sends a greeting. The client and POP3 server then exchange commands and responses (respectively) until the connection is closed or aborted.

First you start out in the **AUTHORIZATION** state. In this state you can enter one of 3 commands:

- **USER**
- **PASS**
- **QUIT**

Once your userid and password are recognized the server will access your mailbox and you enter the **TRANSACTION** state. In this state you can enter one of these commands:

- **STAT** – returns the number of message and the number of octets
- **LIST [msg]** – returns the list of waiting messages

```plaintext
C: LIST
S: +OK 2 messages (320 octets)
S: 1 120
S: 2 200
S: .
...
C: LIST 2
S: +OK 2 200
...
C: LIST 3
S: -ERR no such message, only 2 messages in maildrop
```

- **RETR msg** types out the given message

```plaintext
C: RETR 1
S: +OK 120 octets
S: <the POP3 server sends the entire message here>
S: .
```

- **DELE msg** – deletes the message
C: DELE 1
S: +OK message 1 deleted
...
C: DELE 2
S: -ERR message 2 already deleted

- NOOP – do nothing, resets the timeout counter

C: NOOP
S: +OK

- RSET – unmarks deleted messages (undeletes them)

C: RSET
S: +OK maildrop has 2 messages (320 octets)

You can also enter the QUIT command which will move you to the UPDATE state. In the UPDATE state the server will remove messages that were deleted. Note that if the connection crashes the server will not enter the update state and, thus, will not delete the messages.

C: QUIT
S: +OK dewey POP3 server signing off (maildrop empty)
...
C: QUIT
S: +OK dewey POP3 server signing off (2 messages left)
...

There are some optional POP3 commands:

- TOP msg n – type out “n” lines of msg number

C: TOP 1 10
S: +OK
S: <the POP3 server sends the headers of the message, a blank line, and the first 10 lines of the body of the message>
S: ...
...
C: TOP 100 3
S: -ERR no such message

- UIDL [msg] – unique ID listing numbers

C: UIDL
S: +OK
S: 1 whqtwso000W5w418f9t5JxYwZ
S: 2 QhdPYR:00WBw1Ph7x7
S: ...
C: UIDL 2
S: +OK 2 QhdPYR:00WBw1Ph7x7
   ...
C: UIDL 3
S: -ERR no such message, only 2 messages in maildrop

- **USER name** - identify the name of the mailbox

   C: USER frated
   S: -ERR sorry, no mailbox for frated here
   ...
   C: USER mrose
   S: +OK mrose is a real hoopy frood

- **PASS string** - give a password associated with the mailbox

   C: USER mrose
   S: +OK mrose is a real hoopy frood
   C: PASS secret
   S: -ERR maildrop already locked
   ...
   C: USER mrose
   S: +OK mrose is a real hoopy frood
   C: PASS secret
   S: +OK mrose’s maildrop has 2 messages (320 octets)

- **APOP name digest** - a mailbox name and an MD5 digest string

   S: +OK POP3 server ready <1896.6971709520@dbc.mtview.ca.us>
   C: APOP mrose cac9334bac560ecc979e58001b3e22fb
   S: +OK maildrop has 1 message (369 octets)
      In this example, the shared secret is the string ‘tanstaaf’.
      Hence, the MD5 algorithm is applied to the string
      <1896.6971709520@dbc.mtview.ca.us>tanstaaf
      which produces a digest value of
      cac9334bac560ecc979e58001b3e22fb

### 4.1 A full session example

   S: <wait for connection on TCP port 110>
   C: <open connection>
   S: +OK POP3 server ready <1896.6971709520@dbc.mtview.ca.us>
   C: APOP mrose cac9334bac560ecc979e58001b3e22fb
S: +OK mrose's maildrop has 2 messages (320 octets)
C: STAT
S: +OK 2 320
C: LIST
S: +OK 2 messages (320 octets)
S: 1 120
S: 2 200
S: .
C: RETR 1
S: +OK 120 octets
S: <the POP3 server sends message 1>
S: .
C: DELE 1
S: +OK message 1 deleted
C: RETR 2
S: +OK 200 octets
S: <the POP3 server sends message 2>
S: .
C: DELE 2
S: +OK message 2 deleted
C: QUIT
S: +OK dewey POP3 server signing off (maildrop empty)
C: <close connection>
S: <wait for next connection>

5 IMAP4

- CAPABILITY – requests a listing of capabilities the server supports

  C: abcd CAPABILITY
  S: * CAPABILITY IMAP4
  S: abcd OK CAPABILITY completed

- NOOP – does nothing but reset the inactivity timer

  C: a002 NOOP
  S: a002 OK NOOP completed
  . . .
  C: a047 NOOP
  S: * 22 EXPUNGE
  S: * 23 EXISTS
  S: * 3 RECENT
  S: * 14 FETCH (FLAGS (\Seen \Deleted))
  S: a047 OK NOOP completed

- LOGOUT – tell the server you are leaving
C: A023 LOGOUT
S: * BYE IMAP4 Server logging out
S: A023 OK LOGOUT completed

• AUTHENTICATE – choose an authentication mechanism

S: * OK KerberosV4 IMAP4 Server
C: A001 AUTHENTICATE KERBEROS_V4
S: + AmFYig==
C: BACQUESUkXxLiNVS5FRFUa0CAsho84kLN3/IJmrMG+25a4DT
+nZImj3nhTfJUtC2AA+o0KPKfHESafS9a3CL50ebeydHJtYWFD
WwQ1MWiy6IesKvL5rL9WjXUb9MwT9bp0bYL60KiQh
S: + or//EoAADZI=
C: DiAF5A4gaG0o01ALuBkAMw==
S: A001 OK Kerberos V4 authentication successful
Note: the line breaks in the first client answer are for
editorial clarity and are not in real authenticators.

• LOGIN username – select the mailbox for access

C: a001 LOGIN SMITH SESAME
S: a001 OK LOGIN completed

• SELECT mailboxName – choose a mailbox

C: A142 SELECT INBOX
S: * 172 EXISTS
S: * 1 RECENT
S: * OK [UNSEEN 12] Message 12 is first unseen
S: * OK [UIDVALIDITY 3857529045] UIDs valid
S: * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
S: * OK [PERMANENTFLAGS (\Draft) \) Limited
S: A142 OK [READ-WRITE] SELECT completed

• EXAMINE mailboxName – select a mailbox in read-only mode

C: A932 EXAMINE blurdybloop
S: * 17 EXISTS
S: * 2 RECENT
S: * OK [UNSEEN 8] Message 8 is first unseen
S: * OK [UIDVALIDITY 3857529045] UIDs valid
S: * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
S: * OK [PERMANENTFLAGS ()] No permanent flags permitted
S: A932 OK [READ-ONLY] EXAMINE completed

• CREATE mailboxName – create a mailbox with the given name
C: A003 CREATE owatagusiam/
S: A003 OK CREATE completed
C: A004 CREATE owatagusiam/blurdybloop
S: A004 OK CREATE completed

• DELETE mailboxName – remove a mailbox

  C: A683 DELETE blurdybloop
  S: A683 OK DELETE completed

• RENAME oldMailbox newMailbox – rename a mailbox

  C: Z4S9 RENAME blurdybloop owatagusiam
  S: Z4S9 OK RENAME completed

• SUBSCRIBE mailbox – adds the mailbox to active mailbox list

  C: A002 SUBSCRIBE #news.comp.mail.mime
  S: A002 OK SUBSCRIBE completed

• UNSUBSCRIBE mailbox – removes the mailbox from the active list

  C: A002 UNSUBSCRIBE #news.comp.mail.mime
  S: A002 OK UNSUBSCRIBE completed

• LIST reference name – list mailboxes that match the wildcard name

  For example, here are some examples of how references and mailbox names might be interpreted on a UNIX-based server:

<table>
<thead>
<tr>
<th>Reference</th>
<th>Mailbox Name</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>smith/Mail</code></td>
<td>foo.*</td>
<td><code>smith/Mail/foo.*</code></td>
</tr>
<tr>
<td><code>archive/</code></td>
<td>%</td>
<td><code>archive/%</code></td>
</tr>
<tr>
<td>#news.</td>
<td>comp.mail.*</td>
<td>#news.comp.mail.*</td>
</tr>
<tr>
<td><code>smith/Mail</code></td>
<td>/usr/doc/foo</td>
<td>/usr/doc/foo</td>
</tr>
<tr>
<td><code>archive/</code></td>
<td><code>fred/Mail/*</code></td>
<td><code>fred/Mail/*</code></td>
</tr>
</tbody>
</table>

  C: A002 LIST "~/Mail/ "%"
  S: * LIST (\Noselect) "~/Mail/foo
  S: * LIST () "~/Mail/meetings
  S: A002 OK LIST completed

• LSUB reference name – list subset of names that are active
C: A002 LSUB "#news." "comp.mail.*"
S: * LSUB () "." #news.comp.mail.mime
S: * LSUB () "." #news.comp.mail.misc
S: A002 OK LSUB completed

- APPEND mailboxName (list) (string) literal – append literal to msg

  C: A003 APPEND saved-messages (\Seen) {310}
  C: Date: Mon, 7 Feb 1994 21:52:25 -0800 (PST)
  C: From: Fred Foobar <foobar@Blurdybloop.COM>
  C: Subject: afternoon meeting
  C: To: mooch@owatagu.siam.edu
  C: Message-ID: <B27397-01000000@Blurdybloop.COM>
  C: MIME-Version: 1.0
  C: Content-Type: TEXT/PLAIN; CHARSET=US-ASCII
  C:
  C: Hello Joe, do you think we can meet at 3:30 tomorrow?
  C:
  S: A003 OK APPEND completed

- CHECK – checkpoint the currently selected mailbox

  C: FXXZ CHECK
  S: FXXZ OK CHECK Completed

- CLOSE – close the currently selected mailbox and delete tagged msgs

  C: A341 CLOSE
  S: A341 OK CLOSE completed

- EXPUNGE – delete tagged msgs

  C: A202 EXPUNGE
  S: * 3 EXPUNGE
  S: * 3 EXPUNGE
  S: * 5 EXPUNGE
  S: * 8 EXPUNGE
  S: A202 OK EXPUNGE completed

- SEARCH charset criteria – search mailboxes for matching msg

  C: A282 SEARCH FLAGGED SINCE 1-Feb-1994 NOT FROM "Smith"
  S: * SEARCH 2 84 882
  S: A282 OK SEARCH completed

- FETCH msgset msgdata – retrieve data associate with msg

16
C: A654 FETCH 2:4 (FLAGS RFC822.HEADER.LINES (DATE FROM))
S: * 2 FETCH ....
S: * 3 FETCH ....
S: * 4 FETCH ....
S: A003 OK FETCH completed

- **PARTIAL** seqno data position number – fetch part of a message

C: A005 PARTIAL 4 RFC822 1 1024
S: * 1 FETCH (RFC822 {1024})
S: Return-Path: <gray@cac.washington.edu>
S: ...
S: .......... FLAGS (\Seen))
S: A005 OK PARTIAL completed

- **STORE** msgset data value – change a message

C: A003 STORE 2:4 +FLAGS (\Deleted)
S: * 2 FETCH FLAGS (\Deleted \Seen)
S: * 3 FETCH FLAGS (\Deleted)
S: * 4 FETCH FLAGS (\Deleted \Flagged \Seen)
S: A003 OK STORE completed

- **COPY** – msgset mailbox – copy a message to the destination mailbox

C: A003 COPY 2:4 MEETING
S: A003 OK COPY completed

- **UID** cmdname cmdargs – use unique identifiers

C: A003 UID FETCH 4827313:4828442 FLAGS
S: * 23 FETCH (FLAGS (\Seen UID 4827313)
S: * 24 FETCH (FLAGS (\Seen UID 4827943)
S: * 25 FETCH (FLAGS (\Seen UID 4828442)
S: A999 UID FETCH completed

- **Xatomz** – experimental command

C: a441 CAPABILITY
S: * CAPABILITY IMAP4 XPIG-LATIN
S: a441 OK CAPABILITY completed
C: A442 XPIG-LATIN
S: * XPIG-LATIN ow-nay eaking-spay ig-pay atin-lay
S: A442 OK XPIG-LATIN ompleted-cay
5.1 Sample session

The following is a transcript of an IMAP4 session. A long line in this sample is broken for editorial clarity.

S:  * OK IMAP4 Service Ready
C: a001 login mrc secret
S: a001 OK LOGIN completed
C: a002 select inbox
S:  * 18 EXISTS
S:  * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
S:  * 2 RECENT
S:  * OK [UNSEEN 17] Message 17 is the first unseen message
S:  * OK [UIDVALIDITY 3857529046] UIDs valid
S: a002 OK [READ-WRITE] SELECT completed
C: a003 fetch 12 full
S:  * 12 FETCH (FLAGS (\Seen) INTERNALDATE "14-Jul-1993 02:44:25 -0700"
RFC822.SIZE 4282 ENVELOPE ("Wed, 14 Jul 1993 02:23:25 -0700 (PDT)"
"IMAP4 WG mtg summary and minutes"
("("Terry Gray" NIL "gray" "cac.washington.edu"))
("("Terry Gray" NIL "gray" "cac.washington.edu"))
("("Terry Gray" NIL "gray" "cac.washington.edu"))
((NIL NIL "imap" "cac.washington.edu"))
((NIL NIL "minutes" "CNRI.Reston.VA.US")
("(John Klensin" NIL "KLENSIN" "INFOODS.MIT.EDU")
NIL NIL
"<B27397-01000000@cac.washington.edu>")
BODY ("TEXT" "PLAIN" ("CHARSET" "US-ASCII") NIL NIL "7BIT" 3028 92))
S: a003 OK FETCH completed
C: a004 fetch 12 rfc822.header
S:  * 12 FETCH (RFC822.HEADER {346}
S: Date: Wed, 14 Jul 1993 02:23:25 -0700 (PDT)
S: From: Terry Gray <gray@cac.washington.edu>
S: Subject: IMAP4 WG mtg summary and minutes
S: To: imap@cac.washington.edu
S: cc: minutes@CNRI.Reston.VA.US, John Klensin <KLENSIN@INFOODS.MIT.EDU>
S: Message-Id: <B27397-01000000@cac.washington.edu>
S: MIME-Version: 1.0
S: Content-Type: TEXT/PLAIN; CHARSET=US-ASCII
S:
S:
S: )
S: a004 OK FETCH completed
C: a005 store 12 +flags \deleted
S: * 12 FETCH (FLAGS (\Seen \Deleted))
S: a005 OK +FLAGS completed
C: a006 logout
S: * BYE IMAP4 server terminating connection

18
6 Servers

6.1 Mail

6.1.1 Sending Mail

Mail can be sent to a local user thus:

echo "it works" | mail class -s "it works"

which will create a piece of mail and save it in "/var/spool/postfix/maildrop".

Note that mail you send will not actually arrive until some program processes
the mail in the maildrop file. There are programs that read the maildrop file
and distribute the outgoing mail to user’s mailboxes. They also listen to the
network and receive incoming mail. Such programs are called “Mail Transfer
Agents” and we mention a few below.

6.1.2 Receiving Mail

The mail program is a general purpose mail handler. You can ask what com-
mands it supports:

"/var/spool/mail/root": 1 message 1 new

> N 1 root@localhost.local Tue Mar 16 20:38 14/473 "foo"
& ?

Mail Commands

t <message list> type messages

n goto and type next message
e <message list> edit messages

f <message list> give head lines of messages
d <message list> delete messages

s <message list> file append messages to file

u <message list> undelete messages

R <message list> reply to message senders

r <message list> reply to message senders and all recipients

pre <message list> make messages go back to /usr/spool/mail

m <user list> mail to specific users

q quit, saving unresolved messages in mbox

x quit, do not remove system mailbox

h print out active message headers

! shell escape

cd [directory] chdir to directory or home if none given

A <message list> consists of integers, ranges of same, or user names separated
by spaces. If omitted, Mail uses the last message typed.

A <user list> consists of user names or aliases separated by spaces. Aliases are defined in .mailrc in your home directory.

It allows you to manipulate mail that has arrived in your mailbox. Mail tells you the actual file that holds your mailbox:

"/var/spool/mail/root": 1 message 1 new

This can be configured in different places by different sites.

You can read a piece of email once it is delivered thus:

& t 1
Message 1:
From root@localhost.localdomain Tue Mar 16 20:38:27 2004
X-Original-To: root
Delivered-To: root@localhost.localdomain
To: root@localhost.localdomain
Subject: foo
Date: Tue, 16 Mar 2004 20:38:14 -0500 (EST)
From: root@localhost.localdomain (root)

foo

&

6.2 Postfix

Postfix is a “Mail Transfer Agent”. It receives email from the local machine or the network (or itself if it bounces mail).

Postfix is structured as a series of small programs that serve limited roles.

Stage 1, the pickup stage, is made up of two programs, pickup which reads mail from the maildrop file and smtpd which reads mail from the internet.

Stage 2, the cleanup stage, rewrites the email to a standard form, handles virtual addresses and bounced mail.

Stage 3, the queue stage, handles a queue of incoming mail and places it into the active or, if there is a problem, the deferred queue.

Stage 4, the delivery stage, resolves addresses and either places the mail in a local mailbox or sends it out to another node on the internet.

6.2.1 Getting Postfix

wget ftp://postfix.cloud9.net/official/postfix-2.0.19.tar.gz
6.2.2 Building and Installing Postfix

tar -zxvf postfix-2.0.19.tar.gz
cd postfix-2.0.19
./configure
make
make install

6.2.3 Starting Postfix

postfix start

6.2.4 Stopping Postfix

postfix stop

6.3 QPopper [21]

To quote from the man page:

The Qpopper server is a single program (called popper) that is
launched by inetd when it gets a service request on the POP TCP
port. (The official port number specified in RFC 1939 for POP ver-
sion 3 is port 110. However, some POP3 clients attempt to contact
the server at port 109, the POP version 2 port. Unless you are
running both POP2 and POP3 servers, you can simply define both
ports for use by the POP3 server.

The qpopper program initializes and verifies that the peer IP address
is registered in the local domain (unless the -R command-line option
is used), logging a warning message when a connection is made with a
client whose IP address does not have a canonical name. For systems
using BSD 4.3 bind, it also checks to see if a canonical name lookup
for the client returns the same peer IP address, logging a warning
message if it does not.

The server enters the authorization state, during which the client
must correctly identify itself by providing a valid Unix userid and
password on the server’s host machine (or successfully authenticate
using APOP or AUTH). No other exchanges are allowed during this
state (other than a request to quit.) If authentication fails, a warning
message is logged and the session ends.

Once the user is identified, qpopper changes its user and group ids to
match that of the user and enters the transaction state. The server
makes a temporary copy of the user’s maildrop which is used for
all subsequent transactions (unless running in server mode). These
include the bulk of POP commands to retrieve mail, delete mail,
undelete mail, and so forth.
6.3.1 Getting the source
Use wget to transfer the tar gzipped file from the host:

```
wget ftp://ftp.qualcomm.com/eudora/servers/unix/popper/popper4.0.5.tar.gz
```

6.3.2 Building and installing the executable

```
tar -zxvf popper4.0.5.tar.gz
cd popper4.0.5
./configure --enable-standalone
make
make install
```

6.3.3 Running popper standalone

```
popper -d
```

6.4 Sendmail

7 Clients

7.1 Fetchmail

Fetchmail is a full-featured, robust, well-documented remote-mail retrieval and forwarding utility intended to be used over on-demand TCP/IP links (such as SLIP or PPP connections). It supports every remote-mail protocol now in use on the Internet: POP2, POP3, RPOP, APOP, KPOP, all flavors of IMAP, ETRN, and ODMR. It can even support IPv6 and IPSEC.

Fetchmail retrieves mail from remote mail servers and forwards it via SMTP, so it can then be read by normal mail user agents such as mutt, elm(1) or BSD Mail. It allows all your system MTA’s filtering, forwarding, and aliasing facilities to work just as they would on normal mail.

Fetchmail offers better security than any other Unix remote-mail client. It supports APOP, KPOP, OTP, Compuserve RPA, Microsoft NTLM, and IMAP RFC1731 encrypted authentication methods including CRAM-MD5 to avoid sending passwords en clair. It can be configured to support end-to-end encryption via tunneling with ssh, the Secure Shell.

Fetchmail can be used as a POP/IMAP-to-SMTP gateway for an entire DNS domain, collecting mail from a single drop box on an ISP and SMTP-forwarding it based on header addresses. (We don’t really recommend this, though, as it may lose important envelope-header information. ETRN or a UUCP connection is better.)
Fetchmail can be started automatically and silently as a system daemon at boot time. When running in this mode with a short poll interval, it is pretty hard for anyone to tell that the incoming mail link is not a full-time "push" connection.

Fetchmail is easy to configure. You can edit its dotfile directly, or use the interactive GUI configurator (fetchmailconf) supplied with the fetchmail distribution. It is also directly supported in linuxconf versions 1.16r8 and later.

Fetchmail is fast and lightweight. It packs all its standard features (POP3, IMAP, and ETRN support) in 196K of core on a Pentium under Linux.

fetchmail is a mail-retrieval and forwarding utility; it fetches mail from remote mail servers and forwards it to your local (client) machines delivery system. You can then handle the retrieved mail using normal mail user agents such as mutt(1), elm(1) or Mail(1). The fetchmail utility can be run in a daemon mode to repeatedly poll one or more systems at a specified interval.

The fetchmail program can gather mail from servers supporting any of the common mail-retrieval protocols: POP2, POP3, IMAP2bis, IMAP4, and IMAPrev1. It can also use the ESMTPT ETRN extension and ODMR. (The RFCs describing all these protocols are listed at the end of this manual page.)

While fetchmail is primarily intended to be used over on-demand TCP/IP links (such as SLIP or PPP connections), it may also be useful as a message transfer agent for sites which refuse for security reasons to permit (sender-initiated) SMTP transactions with sendmail.

As each message is retrieved fetchmail normally delivers it via SMTP to port 25 on the machine it is running on (localhost), just as though it were being passed in over a normal TCP/IP link. The mail will then be delivered locally via your systems MDA (Mail Delivery Agent, usually sendmail(8) but your system may use a different one such as small, mmq, exim, or qmail). All the delivery-control mechanisms (such as .forward files) normally available through your system MDA and local delivery agents will therefore work.

If no port 25 listener is available, but your fetchmail configuration was told about a reliable local MDA, it will use that MDA for local delivery instead. At build time, fetchmail normally looks for executable procmail(1) and sendmail(1) binaries.

If the program fetchmailconf is available, it will assist you in setting up and editing a fetchmailrc configuration. It runs under X and requires that the language Python and the Tk toolkit be present on your system. If you are first setting up fetchmail for single-user
mode, it is recommended that you use Novice mode. Expert mode
provides complete control of fetchmail configuration, including the
multidrop features. In either case, the Autoprobe button will tell you
the most capable protocol a given mailserver supports, and warn you
of potential problems with that server.

The behavior of fetchmail is controlled by command-line options and
a run control file, /.fetchmailrc, the syntax of which we describe in
a later section (this file is what the fetchmailconf program edits).
Command-line options override /.fetchmailrc declarations.

Each server name that you specify following the options on the com-
mand line will be queried. If you don’t specify any servers on the
command line, each poll entry in your /.fetchmailrc file will be
queried.

To facilitate the use of fetchmail in scripts and pipelines, it returns an
appropriate exit code upon termination – see EXIT CODES below.

7.2 Evolution

7.3 Fetch.java

// Fetch reads a particular mail number off a POP(3) server.
// It does not delete the message.
// example command line:
// java Fetch mailnumber

// to use this code you must change these lines to the correct values
// and recompile the class:
// String user = "yourname";
// String pass = "yourpass";
// String host = "yourhost";

import java.awt.*;
import java.awt.event.*;
import java.net.*;
import java.io.*;
import java.util.*;
import java.util.Vector;

public class Fetch {

    public static int checkmail(int i) throws Exception
    {
        Socket talk=null;
        BufferedReader inp=null;
        DataOutputStream outs=null;
        StringTokenizer st;

        ...
StringBuffer header;
int num_msgs=-1;
String toss, line;

String user = "yourname";
String pass = "yourpass";
String host = "yourhost";

try
{ talk=new Socket(host,110);
inp=new BufferedReader(new InputStreamReader(talk.getInputStream()));
outs=new DataOutputStream(talk.getOutputStream());
PrintWriter p = new PrintWriter(talk.getOutputStream());
inp.readLine();
p.println("user "+user); p.flush();
inp.readLine();
p.println("pass "+pass); p.flush();
inp.readLine();
p.println("top " + i + " 99999\n"); p.flush();
while(true)
{ line=inp.readLine();
  if(line.equals(".") || line == null)
    break;
  System.out.println(line);
}
outs.writeBytes("QUIT\n");
outs.flush();
outs.close();
outs=null;
}
catch (Exception e)
{ if(outs != null)
{ outs.writeBytes("QUIT\n");
  outs.flush();
  outs.close();
  outs=null;
}
  throw e;
}
finally
{ if(outs != null)
{ outs.writeBytes("QUIT\n");
  outs.flush();
  outs.close();
  outs=null;
}
if(talk != null)
    talk.close();
}
return num_msgs;
}

public static void main(String args[])
{ try
    { checkmail(Integer.decode(args[0]).intValue());
    }
catch(Exception e)
    { e.printStackTrace();
    }
}

7.4 Biff.java

//Java-based version of xbiff. Biff reads its data off a POP(3)
//server. When Biff is started it regularly contact the POP server and
//beeps when new mail has arrived. It shows the number of messages
//currently on the POP server at the bottom of the
//window. Control-clicking in the window shows a configuration panel
//which allows to change the user, host, password and
//frequency. Single-clicking shows a box with all headers.
//
// - indication of number of mails waiting on POP server
// - listing of headers by single-clicking on the Biff window
// - extension (MailResource) to allow access to a number of other mail
//   resources (e.g. file, IMAP)
//
//Arguments can be seen by starting 'java Biff -help'.
//
//The subjects of all mail currently on the pop server can be looked at
//by clicking with the mouse on Biff's window. A control-mouse-click
//displays a configuration dialog.
//
// example command line:
//java Biff -user daly -pass burns -host pop.mail.net -delay 1000

import java.awt.*;
import java.awt.event.*;
import java.net.*;
import java.io.*;
import java.util.*;
import java.util.Vector;

interface MailListener
{
    public void NewMail(int new_msgs);
    public void Reset();
    public void Error(String msg);
    public void OkAgain();
}

interface MailResource
{
    public void AddMailListener(MailListener l);
    public void RemoveMailListener(MailListener l);
    public void StartPollingForMail();
    public void StopPollingForMail();
    public Vector GetHeaders() throws Exception;
    public long GetSleepPeriod();
    public void SetSleepPeriod(long msecs);
    public String GetHost();
    public void SetHost(String new_host);
    public String GetUser();
    public void SetUser(String new_user);
    public String GetPass();
    public void SetPass(String new_pass);
}

class PopServer implements MailResource, java.langRunnable
{
    private Vector listeners=new Vector();
    private long sleep_period=60000;
    private int counter=0;
    private String uid="";
    private String pw="";
    private String pop_host="popsrv";
    private int num_msgs=-1;
    private Vector headers=new Vector();
    private boolean in_error=false;
    private Thread poller=null;

    public PopServer(long l, String userid, String password, String popsrv)
    { sleep_period=l;
        uid=userid;
        pw=password;
        pop_host=popsrv;
        ResetHeaders();
    }

    public long GetSleepPeriod() {return sleep_period;}
}
public void SetSleepPeriod(long msecs)
{ if(msecs > 0)
    sleep_period=msecs;
}

public String GetHost() {return pop_host;}
public void SetHost(String new_host) {pop_host=new_host;}
public String GetUser() {return uid;}
public void SetUser(String new_user) {uid=new_user;}
public String GetPass() {return pw;}
public void SetPass(String new_pass) {pw=new_pass;}

public void AddMailListener(MailListener l)
{ listeners.addElement(l);
}

public void RemoveMailListener(MailListener l)
{ listeners.removeElement(l);
}

public void StartPollingForMail()
{ if(poller == null)
    { poller=new Thread(this);
        poller.start();
    }
}

public synchronized void StopPollingForMail()
{ if(poller != null)
    { poller.stop();
        poller=null;
    }
}

public Vector GetHeaders() throws Exception {return headers;}

private void ResetHeaders()
{ headers.removeAllElements();
    headers.addElement(" --- No messages available --- ");
}

public void run()
{ MailListener l;
    while(true)
    { try
{ num_msgs=CheckMail();
    if(in_error == true)
    { in_error=false;
        for(int i=0; i < listeners.size(); i++)
            ((MailListener)listeners.elementAt(i)).OkAgain();
    }
    if(num_msgs == 0)
    { if(counter > 0)
        { for(int i=0; i < listeners.size(); i++)
            ((MailListener)listeners.elementAt(i)).Reset();
                counter=0;
                ResetHeaders();
        }
    }
    else
    { if(num_msgs > counter)
        { for(int i=0; i < listeners.size(); i++)
            ((MailListener)listeners.elementAt(i)).NewMail(num_msgs);
                counter=num_msgs;
        }
    }
    catch(Exception error)
    { for(int i=0; i < listeners.size(); i++)
        { in_error=true;
            l=(MailListener)listeners.elementAt(i);
            l.Error("POP server error: " + error);
        }
    }
    try
    { Thread.currentThread().sleep(sleep_period);
    }
    catch(java.lang.InterruptedIOException ex)
    { break;
    }
}

private String readIn(BufferedReader dis) throws IOException
{ String incoming;
    StringTokenizer st;
    incoming=dis.readLine();
    return incoming;
}

private boolean popError(String incoming) throws Exception
public synchronized int CheckMail() throws Exception
{
    Socket talk=null;
    BufferedReader inp=null;
    DataOutputStream outs=null;
    StringTokenizer st;
    StringBuffer header;
    int num_msgs=-1;
    String toss, line;
    try
    {
        talk=new Socket(pop_host,110);
        inp=new BufferedReader(new InputStreamReader(talk.getInputStream()));
        outs=new DataOutputStream(talk.getOutputStream());
        PrintWriter p = new PrintWriter(talk.getOutputStream());
        popError(readLn(inp));
        Thread.currentThread().sleep(1000);
        p.println("user "+uid); p.flush();
        popError(readLn(inp));
        p.println("pass "+pw); p.flush();
        popError(readLn(inp));
        p.println("stat"); p.flush();
        toss=readLn(inp);
        popError(toss);
        st=new StringTokenizer(toss);
        st.nextToken();
        num_msgs=Integer.parseInt(st.nextToken());
        if(num_msgs > counter)
        {
            headers.removeAllElements();
            for(int i=1; i < num_msgs+1; i++)
            {
                header=new StringBuffer();
                outs.writeBytes("top " + i + " 0\n");
                popError(readLn(inp));
                while(true)
                {
                    line=readLn(inp);
                    if(line.equals(".") || line == null)
                        break;
                    if(line.indexOf("From:")!= -1)
                    {
                        header.append(line);
                        continue;
                    }
                }
            }
        }
    }
}
if(line.indexOf("Subject:")) != -1)
    { header.append(" "+line);
      continue;
    }
} 
if(header.length() > 0)
  headers.addElement(header.toString());
}
}
outs.writeBytes("QUIT\n");
outs.flush();
outs.close();
outs=null;
}
catch (Exception e)
{ if(outs != null)
  { outs.writeBytes("QUIT\n");
    outs.flush();
    outs.close();
    outs=null;
  }
  throw e;
}
finally
{ if(outs != null)
  { outs.writeBytes("QUIT\n");
    outs.flush();
    outs.close();
    outs=null;
  }
  if(talk != null)
    talk.close();
  if(headers.size() == 0)
    ResetHeaders();
}
return num_msgs;
}
}

class Headers extends Window 
{
  public Headers(Frame parent, Vector vec)
  { super(parent);
    java.awt.List headers=new java.awt.List();
    for(int i=0; i < vec.size(); i++)
      { headers.add((String)vec.elementAt(i));

31
headers.addMouseListener(
    new MouseListener()
    { public void mouseClicked(MouseEvent e) {Close();}
    public void mouseEntered(MouseEvent e) {}
    public void mouseExited(MouseEvent e) {}
    public void mousePressed(MouseEvent e) {}
    public void mouseReleased(MouseEvent e) {}}
    );
add(headers);
setSize(500,300);
Point parent_loc=parent.getLocation();
parent_loc.x=parent_loc.x-430;
setLocation(parent_loc);
setFont(new Font("TimesRoman",Font.BOLD,12));
show();
}

public void Close() {dispose();}

}

class bCanvas extends Canvas
{
    Image one=null, two=null;
    Toolkit tk;
    Biff jb;
    Graphics offgraph;
    Image offscrn;

    public bCanvas(Biff jb)
    { super();
      File f1test, f2test;
      this.jb = jb;
      try
      { f1test = new File(this.jb.mail);
        f2test = new File(this.jb.nomail);
        tk=this.getToolkit();
        if(f1test.exists())
          one=tk.getImage(this.jb.mail);
        if(f2test.exists())
          two=tk.getImage(this.jb.nomail);
      }
      catch (NullPointerException e)
      { System.err.println("File is null");
      }

      32
public void paint(Graphics g) {
    Dimension dm;
    FontMetrics fm;
    dm = jb.size();
    offscrn = createImage(this.size().width, this.size().height);
    offgraph = offscrn.getGraphics();
    offgraph.setColor(Color.lightGray);
    if (one != null)
        offgraph.draw3DRect(0, 0, this.size().width, this.size().height, true);
    offgraph.setColor(Color.black);
    if (jb.ErrorOccurred())
    { offgraph.setColor(Color.red);
        offgraph.setFont(new Font("TimesRoman", Font.BOLD, 12));
        fm = getFontMetrics(new Font("TimesRoman", Font.BOLD, 12));
        offgraph.drawString(" Error ",
            (this.size().width - fm.stringWidth(" Error "))/2,
            ((this.size().height) - (fm.getHeight()/2)));
    g.drawImage(offscrn, 0, 0, this);
    return;
    }
    if (jb.num_msgs > 0)
    { if (one != null)
        { offgraph.drawImage(one, 0, 0, dm.width, dm.height - 50, this);
            offgraph.setFont(new Font("TimesRoman", Font.BOLD, 12));
            fm = getFontMetrics(new Font("TimesRoman", Font.BOLD, 12));
            offgraph.drawString(" + jb.num_msgs + " msg",
                (this.size().width - fm.stringWidth(" "))/2-10,
                ((this.size().height) - (fm.getHeight()/2))+5);
        }
    else
    { offgraph.setColor(Color.blue);
        offgraph.fillRect(0, 0, dm.width, dm.height);
        offgraph.setColor(Color.white);
        offgraph.setFont(new Font("TimesRoman", Font.BOLD, 12));
        fm = getFontMetrics(new Font("TimesRoman", Font.BOLD, 12));
        offgraph.drawString(" - jb.num_msgs + " msg",
            (this.size().width - fm.stringWidth(" "))/2,
            ((this.size().height) - (fm.getHeight()/2)));
    }
    }
    else
    { offgraph.setColor(Color.black);
        if (two != null)
            { offgraph.drawImage(two, 0, 0, dm.width, dm.height, this);
    }
public void update(Graphics g) {paint(g);}

public Dimension preferredSize() {return new Dimension(70,70);}

public Dimension minimumSize() {return preferredSize();}

class bInfoBox extends Frame
{
    Biff caller;

    public bInfoBox(String title, String message, Biff caller)
    { super(title);
        this.caller = caller;
        add("Center", new Label(message, Label.CENTER));
        Button okay = new Button("Okay");
        add("South", okay);
        pack();
        show();
    }

    public boolean handleEvent(Event evt)
    { if (evt.target instanceof Button)
        { this.dispose();
            caller.killJberror();
            return true;
        }
        return super.handleEvent(evt);
    }
}

class bSetup extends Frame
{
    TextField mailhost, user, pass, freq;
    Button close, reset, quit;
    Biff jb;

bInfoBox info;

public bSetup(Biff jb)
{
    super("Biff Configure");
    this.jb = jb;
    SetLayout(new GridLayout(5,1,10,10));
    Panel line1 = new Panel();
    Panel line2 = new Panel();
    Panel line3 = new Panel();
    Panel line4 = new Panel();
    Panel line5 = new Panel();
    line1.setLayout(new GridLayout(1,2,5));
    line2.setLayout(new GridLayout(1,2,5));
    line3.setLayout(new GridLayout(1,2,5));
    line4.setLayout(new GridLayout(1,3,5));
    line5.setLayout(new GridLayout(1,2,5));
    line1.add(new Label("Mailhost:",Label.RIGHT));
    mailhost = new TextField(jb.getHost(),30);
    line1.add(mailhost);
    line2.add(new Label("User Name:",Label.RIGHT));
    user = new TextField(jb.getUser(),30);
    line2.add(user);
    line3.add(new Label("Password:",Label.RIGHT));
    pass = new TextField(jb.getPass(),15);
    pass.setEchoCharacter('*');
    line3.add(pass);
    line5.add(new Label("Frequency (milliseconds) ",Label.RIGHT));
    freq = new TextField(Long.toString(jb.getDelay()),2);
    line5.add(freq);
    close = new Button("Set Values");
    quit = new Button("Quit");
    line4.add(close);
    line4.add(quit);
    add(line1);
    add(line2);
    add(line3);
    add(line5);
    add(line4);
    pack();
}

public boolean handleEvent (Event evt)
{
    if (evt.target == close)
    {
        jb.setUser(user.getText());
        jb.setPass(pass.getText());
        jb.setHost(mailhost.getText());
    }
}
try
{ jb.SetDelay(Integer.parseInt(freq.getText()));
}
catch (NumberFormatException e) {}
setVisible(false);
return true;
}
if (evt.target == reset)
{ jb.jbcanvas.repaint();
  return true;
}
if (evt.target == quit)
{ System.exit(0);
  return true;
}
return false;
}

public class Biff extends Frame implements MailListener, WindowListener
{
  bSetup jbsetup;
  bCanvas jbcanvas;
  String nomail,mail;
  bInfoBox jberror;
  int num_msgs=0;
  MailResource mail_resource=null;
  Headers headers=null;
  boolean error_occurred=false;

  public Biff(String title, String m, String n, MailResource res)
  { super(title);
    mail = m;
    nomail = n;
    mail_resource = res;
    jbsetup = new bSetup(this);
    jbcanvas = new bCanvas(this);
    add("Center", jbcanvas);
    pack();
    setSize(70,90);
    mail_resource.AddMailListener(this);
    mail_resource.StartPollingForMail();
    addWindowListener(this);
    show();
  }

  public long GetDelay() {return mail_resource.GetSleepPeriod();}
}
public void SetDelay(long msecs) { mail_resource.SetSleepPeriod(msecs); }
public String GetHost() { return mail_resource.GetHost(); }
public void SetHost(String new_host) { mail_resource.SetHost(new_host); }
public String GetUser() { return mail_resource.GetUser(); }
public void SetUser(String new_user) { mail_resource.SetUser(new_user); }
public String GetPass() { return mail_resource.GetPass(); }
public void SetPass(String new_pass) { mail_resource.SetPass(new_pass); }
public boolean ErrorOccurred() { return error_occurred; }
public void killjerror(){jerror=null;}

public boolean handleEvent (Event evt)
{ if (evt.id == Event.MOUSE_DOWN)
{ if(evt.controlDown())
   { jsetup.show();
      return true;
   }
else
{ try
   { headers=new Headers(this, mail_resource.GetHeaders());
   }
      catch(Exception e) {} 
      return true;
   }
return super.handleEvent(evt);
}

public void NewMail(int num_new_msgs)
{ error_occurred=false;
  Toolkit.getDefaultToolkit().beep();
  num_msgs=num_new_msgs;
  jbcanvas.repaint();
}

public void Reset()
{ error_occurred=false;
  num_msgs=0;
  jbcanvas.repaint();
}

public void Error(String msg)
{ error_occurred=true;
  System.err.println(msg);
  jbcanvas.repaint();
}
public void OkAgain()
{ error_occurred=false;
  jbcanvas.repaint();
}

terminal

public void windowOpened(WindowEvent e) {}
public void windowClosing(WindowEvent e)
{ mail_resource.RemoveMailListener(this);
  mail_resource.StopPollingForMail();
  System.exit(1);
}
public void windowClosed(WindowEvent e) {}
public void windowIconified(WindowEvent e) {}
public void windowDeiconified(WindowEvent e) {}
public void windowActivated(WindowEvent e) {}
public void windowDeactivated(WindowEvent e) {}

public static void main(String args[])
{
  Biff master;
  long d=60000;
  String m="";
  String n="";
  String p="";
  String u="user";
  String h="host";
  for(int i = 0; i < args.length; i++)
  {
    if (args[i].equals("-mail")) m = args[i+1];
    if (args[i].equals("-nomail")) n = args[i+1];
    if (args[i].equals("-pass")) p = args[i+1];
    if (args[i].equals("-user")) u = args[i+1];
    if (args[i].equals("-host")) h = args[i+1];
    if (args[i].equals("-delay")) d = new Long(args[i+1]).longValue();
    if (args[i].equals("-help"))
      System.out.println("\nBiff -- Biff for Java");
      System.out.println("Mouse click -- see headers");
      System.out.println("Control-Mouse click -- change parameters/quit");
      System.out.println("-mail filename ");
      System.out.println("-nomail filename ");
      System.out.println("-user username");
      System.out.println("-pass password");
      System.out.println("-host mailhost");
      System.out.println("-delay milliseconds");
      System.out.println("--help\n\n");
      System.exit(0);
  }
}

MailResource res=new PopServer(d, u, p, h);
master = new Biff("Biff",m,n,res);
Dimension dim=Toolkit.getDefaultToolkit().getScreenSize();
master.setLocation(dim.width-70, 65);
}
}
References

[1] Reuters,  
“Microsoft facing European sanctions”,  

[2] Jardin, Xeni,  
“P2P in the Legal Crosshairs”,  

[3] Associated Press,  
“Privacy Safeguards Deep-Sixed”,  

[4] Associated Press,  
“Feds Want Wiretap-Ready Net”,  

Foundation for Information Policy Research,  
“The Draft IPR Enforcement Directive – A Threat to Competition and to Liberty”,  
“http://www.fipr.org/copyright/draft-ipr-enforce.html”

[6] Nelson, R.,  

[7] Myers, J., Rose, M.,  

[8] Gellens, R., et. al,  
“POP3 Extension Mechanism”,  
“http://asg.web.cmu.edu/rfc/rfc2449.html”

[9] Klenzsin, J.,  
“Simple Mail Transfer Protocol”,  
“http://asg.web.cmu.edu/rfc/rfc2821.html”

[10] Klenzsin, J.,  
“SMTP Service Extensions”,  
“http://asg.web.cmu.edu/rfc/rfc1869.html”

[11] Partridge, C.,  
“Main routing and the domain system”,  
“http://asg.web.cmu.edu/rfc/rfc974.html”
[12] Postel, J.
“Simple Mail Transfer Protocol”,
“http://asg.web.cmu.edu/rfc/rfc821.html”

[13] Postel, J.
“Simple Mail Transfer Protocol”,
“http://asg.web.cmu.edu/rfc/rfc788.html”

[14] Sluzer, S., Postel, J.,
“Mail Transfer Protocol”,
“http://asg.web.cmu.edu/rfc/rfc780.html”

[15] Sluzer, S., Postel, J.,
“Mail Transfer Protocol”,
“http://asg.web.cmu.edu/rfc/rfc772.html”

[16] Crispin, M.,
“Internet Message Access Protocol - Version 4”,
“http://asg.web.cmu.edu/rfc/rfc1730.html”

[17] Myers, J., Rose, M.,

[18] Venema, Wietse,
“Postfix”,
“www.postfix.org”

[19] Groove Networks Press Release,
“Groove Networks Announces Role In Newly Announced Homeland Security Information Network”,

[20] O’Gara, Maureen,
“HP Starts Pushing Desktop Linux”,
“http://www.linuxworld.com/story/44054.htm”

[21] Qualcomm,
“QPopper 0.5.tar.gz”,