Email as (real) files

Francisco J Ballesteros
Laboratorio de Sistemas
Universidad Rey Juan Carlos

ABSTRACT

Understanding mail is a complex task. There are many standards involved and many formats for headers, text, and attachments. On the other hand, reading mail is simple: There is some text shown and one or more files included as attachments. This paper describes a mail system built by considering this. It stores mails in the file system as shown in a mail reader, and attachments decoded as regular files. Thus making file tools become mail handling programs.

Introduction

Long ago someone said regarding mail in Plan 9:

> *Seems our user base is growing larger than main memories of our imap servers. Anyone have any ideas to keep systems from running out of memory?*

By that time Plan B was using the file server program (Plan 9's *fossil*) also as a mail server program. To do so, the only requirement was to keep mail undecoded and stored in the file system as any other document would be. Today Nupas [1] can do the job, although we keep on using the tools described here.

A mail including some text along with several PDF documents is not different from a note made on a file along two PDF files. And that is the format used by the mail tools described here.

The editor used (be it *Acme* or *O/live*) becomes a mail reader as soon as there is a convenient way of producing mail listings. That can be done using *grep*, although it is more convenient to use a program built for such task. A mouse click on a mail name (a mail path) opens it for reading. The same happens to any attachment.

Furthermore, attachments using weird names may be renamed using *mv*, or perhaps deleted using *rm* if they are of no interest. This program also becomes the ultimate spam processing tool. As another benefit, *Venti* coalesces storage for multiple copies of the same attachment, if present on different mails.

The same approach can be applied for sending mail. A program may collect files written by the user into an agreed-upon directory. These files are simple text files in the format used by *Acme*'s *Mail*. Only that now any editor is able to compose mail for delivery, without requiring an specific tool.

Considering that files in the file server are remotely available, mail becomes remotely available as well. For small remote terminals (such as phones) *upas/fs* may be instructed to use the new mailbox format, making IMAP available as well.

Overall, we think this approach is a good strategy for handling mail. The approach consists mostly on avoiding the need for software to handle mail. If there is no software, it will hardly run slow or out of memory.

Of course this is feasible only after having decoded mail messages, which means that indeed we require software for the task. Its job is now to unpack a Plan 9 mailbox into an already decoded set of files. This is described in the rest of the paper.

Mail box format

In Plan B, mails for users are parsed and decoded first, and then stored in a file hierarchy where these and other tools can be used to process them. The main tool is *mail2fs*, which performs this processing. A mailbox is a directory, usually under `/mail/box/$user/`, that contains one directory per month (e.g., `200603/` for mails processed on March 2006). In these directories there is one directory per message.

The convention is that message (directory) names starting with ''`a.`'' correspond to archived messages not to be usually shown to the user. Names starting with ''`s.`'' correspond to messages that seem to be spam (not usually shown either). Names starting with ''`d.`'' correspond to deleted messages not yet removed from the file system. Any other rune can be used instead of `a`, `s` or `d` as a convenience (the meaning would be up to the user). But for this optional prefix, messages use a serial message number as their directory name.

The directory for a message contains at least two files: `text` and `raw`. The `text` file has the mail headers and body already processed for reading. Its contents are similar to what *Acme*'s *Mail* would show for the message. The file `raw` has the original mail headers without any processing, including the UNIX header line. This file is kept both for debugging and also for obtaining message ids when replying to mails.

Any attachment in the mail is kept stored in a separate file (possibly with the file name indicated in the MIME header) ready to be used. That is, decoded. When the attachment is a mail, the message is stored in a subdirectory following the same conventions stated above. For mails with attachments, the `text` file contains additional text indicating the relative path names (from the mail's directory) that can be used to open the attachments. This is convenient to *plumb*(1) them while reading.

A Plan B mail box also contains two files: `seq` and `digest`. Messages are given sequence numbers while they are added to the mail box. The file `seq` contains the sequence number for the last message (or zero) and is `DMEXCL` to provide locking for multiple programs using the mail box. The file `digest` contains digests for mails added to the mailbox using *mail2fs* (but not for those added by hand using file tools). When a message has a digest that was already seen in the past the message is silently discarded as a duplicate.

Virtual mail folders may be created by storing text files with mail lists that contain a mail description per line starting with the path for each mail. Copying the text shown for some messages in a mail listing into another text file would ''save'' such messages into that file. The program *mlist* writes to standard output a clean listing for messages with paths found in the standard input.

Other programs in the suite (most of them scripts) provide tools as a convenience for reading mail on *Acme* and *O/live*.

Examples

Move all mails from the Plan 9 mailbox to the Plan B one, and create the latter if it does not exist:

```
; mail2fs -c
```

List mails:

```
; mails
200910/1153/text     nemo          Re: FDP: Ámbito
200910/1152/text     enrique.sor   FDP: Ámbito
200910/1151/text     paurea        Re: [9fans] clarification on man 9p
200910/1150/text     esoriano      Re: cómo traducís contention?
200910/1149/text     paurea        Re: cómo traducís contention?
```

Plumb all PDF attachments from Glenda:

```
; for (f in '{mails | grep Glenda | awk '{print $1}'}^/*.pdf)
        plumb $f
```

Create a virtual folder for messages in 9fans:

```
; mails | grep 9fans > 9fans
```

Of course we may use *sort* to present the list of mails in the order desired:

```
; mails | grep '^[0-9]' | sort -t/ +0nr | sort -t/ +1r | mlist
```

Most of the times there are convenience scripts to process mail and there is no need to execute complex command lines. These scripts can be used within *Acme* or *O/live* and do their job for a panel showing a single message, for messages named in the standard input, or for messages given as argument.

For example, within *O/live*, executing ''`!Mails`'' at `/mail/box/$user/msgs` produces an initial list of mails. This list can be refreshed by executing ''`,<Mails`'' for the panel containing the mail list. To read a mail we just click (button-3) on the mail path. To remove mails from a virtual folder with cut them from the list.

To select mails according to text shown in the mail index we use the Sam command language. For example, ''`,x/9fans/+-p`'' produces a mail index for mails coming from *9fans*. Should we want a mail folder for just those messages, we may simply write the text shown in the panel to a text file. That file becomes a virtual mail box.

To archive a set of mails we send their index text as standard input to *Arch*. For example, ''`.>Arch`'' archives all mails selected in the panel. In the same way, *Spam* flags mails as spam. In both cases, *mv* is the underlying tool; it renames the directories to start with ''`a.`'' and ''`s.`'' respectively. (*Mails* lists all mail, archived or not, if given the `-a` flag).

Locate mails about *mail2fs*:

```
; looktags nemo msgs mail2fs
/mail/box/nemo/msgs/200801/a.9955/text
/mail/box/nemo/msgs/200801/a.9955/raw
/mail/box/nemo/msgs/200801/a.9937/text
/mail/box/nemo/msgs/200801/a.9937/raw
/mail/box/nemo/msgs/200801/a.9937/1.mai2fs.c
 ...
```

Here, *looktags* is a general purpose file search tool. Since mails are included in the file server as regular files, they are also indexed by such tool.

References

1. E. Quanstrom, Nupas: Scaling upas., *Intl. Workshop on Plan 9*, 2008.