This is the accepted version of a paper published in . This paper has been peer-reviewed but does not include the final publisher proof-corrections or journal pagination.

Citation for the original published paper (version of record):

An Internal Gate for a Local Email System
*Sys Admin Magazine - The Journal for UNIX Systems Administrators, V4*(1)

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

Permanent link to this version:
http://urn.kb.se/resolve?urn=urn:nbn:se:lnu:diva-77948
An Internal Gate for a Local Email System

Lars Magnusson

Some years ago, I implemented a simple mail server -- qdms, "the quick and dirty mail server," from alt.sources -- to show my students what can be done with shell scripts. The main feature of the server was the use of a UNIX mailbox as queuing system.

Since then I have reused this concept in several other applications, as a small-scale netnews distribution system, a fax gate, and an Internet gate for a MS-Mail system. In this article I describe the concept, with reference to the Internet gate.

The Problem

A former colleague, now working as a regional technical manager in the state-owned Swedish Employment Training Cooperation, needed to get email out on to Internet through the organization's newly acquired MS-Mail system. MS-Mail is supposed to contact the Internet and X400 through an email switch, Softswitch EMX. The problem was that this gate, used with MS-Mail, allowed internal mail users only to reply to external mail or to send mail to external correspondents, who had earlier sent mail in through the gate. Users could not send mail out to recipients not registered in the switch.

The Cause

MS-Mail addresses normally have a format of three fields with a maximum of ten letters each. Internet and X400 addresses must be registered as a prefix and a shortname, with ten or fewer letters each. The shortname is often a machine-generated identity string.
The result is that some MS-Mail to SMTP- or UUCP-gates, such as the Softswitch EMX, have serious problems in dealing with new recipient addresses. Such addresses must be registered by the system administrator before they can be used.

The Solution

Because he needed of immediate contact with people on Internet, my former colleague challenged me to come up with a quick fix for this problem. Since I knew that I was able to reach registered addresses on Internet, my solution was simple: Send the letter to a fixed address, a mail server, where the extra receiver address can be extracted and the letter rewritten and resent to the correct recipient.

At the local level, outgoing mail would have a fixed first address line. This line would specify the correct Internet address.

A UNIX mail machine still on the Internet was designated as the holder of the fixed address. This was where the Internet mail would be queued and processed.

To ensure that the gate could distinguish the fixed address line from ordinary lines in a message, we chose the following format: "To userid@domain". The MS-Mail system would send letters to a shortname tied to the mailbox on the UNIX system. This mailbox would be read by the application, the mailgate. By reading the Internet-address on the first address line of the letter, the mailgate could rebuild the letter and resend it to the correct recipient, as if it had come directly from the sender. The receiver would notice the detour only if he/she read the receiver lines in the header of the letter.

When a reply is received, it goes directly back to the original sender through the EMX email switch. This eliminates the use of the extra gate in future contacts, since the external mail ID now gets automatically registered with the reply.

The Implementation

The implementation begins on the UNIX gateway machine, with the creation of an ID associated with the gate's function, like inet, internet, or some other suitable name. The ID is registered in the MS-Mail gate under a similar name, in this case INTERNET-internet, where the first part designates the EMX gate and the second the UNIX mailbox.

Mail sent by users is routed to the UNIX mailbox, where it is queued until it can be processed for retransmission out on Internet.

The gateway process is a shell script, inetmail (Listing 1), that is run by cron, once a minute. It reads the first letter in the box, saving it in a temporary file in /tmp. A read from the queue looks like this (under BSD):

```
/usr/5bin/echo "s 1 /tmp/infile.$$ \n q \n" I \
/usr/ucb/Mail -N -u mailbox-id > /dev/null 2>&1
```

The request is now saved as the tmp file infile.$$, which will be used for subsequent processing.
At the same time the script steps the queue one position forward by deleting the letter from the queue with qn. Then simulates a <RETURN> from the keyboard, a functionality apparently available only with the SVRx echo (the BSD echo's that I tried did not have this capacity). If your system doesn't have a SVRx addendum, you might try saving the two commands as two lines without the "\n" and then using cat to supply them to the mail program (either Mail or mailx), like this:

```bash
cat mailcmd Mail -N -u mailbox-id 2>&1 > /dev/null
```

A major advantage to using the mailer as a queueing system is that you don't have to worry about concurrency problems during the processing. If new requests come in during processing, the mailer system takes care of them independently, as when we read our own mail. The result is a fairly decent and flexible queueing system, with the only major drawback being that it is not as fast as a custom-made queuer.

After the letter is saved in the tmpfile, it must be rebuilt with new headers to prevent "Apparently-To:" headers from showing up. To accomplish this, I use grep, sed, awk, and cut, storing the results in shell variables that are later used to construct the new header.

To prevent the fixed address line from appearing in the letter, I created a sed filter that just reads everything after the fixed address line into the new letter. Because of the way ksh and sed seem to jointly handle shell variables, I had to dump the sed script in a new tmpfile and run it from there.

When the letter is rebuilt, I run an error test on the extra "To:" header, to see if the sender had forgotten it. The test could be placed before the rebuilding of the letter, but since the load on the system from this gate is marginal, I didn't bother about the order. The test is a simple one and could be enhanced in different ways to cover other errors. The most important aspect is that the sender receives a message warning that his/her letter didn't make it through the gate and telling how to correct the problem.

Once the letter is rebuilt, I use sendmail, with the -f flag set to the original sender, to resend it:

```bash
sendmail -fsender receiver < letter
```

The letter now looks as if it had come directly from sender.

One warning note: as you may know, this is how some fake Internet mail has been generated, so you must run the process as a trusted user, but not as root. I chose to run it as daemon, and as far as I can see, it should not be possible for this implementation to compromise the security within the local system.

Since I want to monitor the performance of the gate, I also perform some logging on the outgoing messages. In many organizations this is unnecessary, but if you are billing your departments, you could use the logging mechanism for accounting purposes. In this case, though, your users would have to send all their mail through the gate, which would in turn make cron's "every minute" much too long a period for coping with the queue.
Conclusion

The gate has worked for some time now as a semiofficial gate for an organization of 5,000 employees. Some bugs have been ironed out -- what the present version lacks most significantly is some form of control if the system goes down during processing.

Since the gate was intended as a temporary measure (until the vendor solved this problem) and since it is monitored by the local manager, we felt that building such a control was not worth the extra effort. However, there are several different ways you could approach this issue. I would probably choose to resolve it by separating the reading of the letter into the tmpfile and the deletion of the letter from the queue into two steps, so that the letter would be deleted from the queue after it had been sent. Then, if the system were to go down, the letter would still be in the queue for processing. This approach does risk sending a letter twice, if the system should crash between sending and deleting -- which you may find unacceptable. If so, a lockfile solution would give better security.

Another possible problem can be the filtering of headers in /usr/lib/mail.rc. If the filtering becomes extensive -- for example, if you remove the ordinary "To" header -- the script would have to be reworked to get the make_request function to work properly. In the case of a missing "To:" header, I would replace the present script lines with the following:

```
TO='grep "^To: " $INFILE.$$ I awk '{print $0; exit }' I cut -d " -f2'
```

Why not just remove the awk section? Well, the user might have included an old letter, with an extra, unexpected "To:" header. And if this isn't "disarmed" with some "include" char, then your script will crash. The same would apply to the LINE shell variable.

As a safety measure, when you've installed your queuescript, run it manually with sh -x scriptname a couple of times. For some shells you might need to add a set -x at the beginning of the individual functions. I found this to be the case when I wrote a script with the help of MKS Toolkit's DOS-ksh, while SUN's sh doesn't seem to need it.

About the Author

Lars Magnusson has a B.Sc. in Geology from Gothenburg Univ. in Sweden (1980) and an M.Sc. in Petroleum Exploration from Chalmers Univ. of Technology, also in Sweden (1984). He worked as a mining geologist/geophysist in Sweden and Greenland in the early 1980s, but since 1985 has been a system administrator, system manager, and a teacher of UNIX-related topics in Greenland, Denmark, and Sweden. He is currently a consultant specializing in UNIX system administration, sendmail, and DNS configuration.