How To Set Up A Linux Mail Hub

A mail hub is a machine that sits on your LAN, acting as the interface between your users and the Internet. It processes all mail, ensuring that messages sent from users get transmitted and that incoming items are distributed to the correct user. Linux, the free version of Unix, allows you to set up a Unix-based mail hub on an old Intel box for minimum outlay and maximum flexibility.

By Paul Dunne

In this article, we will cover setting up a Linux machine to act as a mail hub for the local network, allowing workstations to send and receive Internet mail without themselves being directly connected to the outside world.

Before you try doing this, you should already know how to install Linux and connect that Linux machine to the Internet. [*If you don't, we'll be covering it in a future issue of PCNA - Ed.*]

Sendmail

Although the sendmail utility comes ready-built with almost any Linux distribution you care to name, there are advantages to knowing how to compile it yourself. Firstly, a widely-used, important program such as sendmail inevitably attracts a lot of attention from hackers. Updates to the program incorporating fixes for the latest security holes are regularly being made available. These updates are available as patches against the source code, which you need to add and then re-compile. (You can get sendmail precompiled, of course, but it's safer to compile your own from source code that doesn't appear to have been hacked.)

There is a sendmail Web page at www.sendmail.org, and many sites mirror the source code from there. Once you have the source, extract it into a directory (/usr/src/sendmailx.x.xx) and cd into the src directory therein. In there is a makesendmail shell script that will do all the work for you. It is easiest to do the configuration in the source directory, before actually installing the binary and configuration files.

Configuring

Tackling the sendmail configuration process from scratch is tough. The main configuration file, sendmail.cf, is designed to be easy for the sendmail program to read. Unfortunately, this means that it's somewhat cryptic to mere humans. Fortunately, it is a task that rarely has to be done from the beginning. The sendmail distribution includes several sample sendmail.cf files, one or other of which can be adapted to most configurations with a few changes. Any modern Linux distribution will include these sample files in its sendmail installation.

Recent versions of sendmail have introduced an important simplification into the configuration process, by shifting the user intervention from direct editing of the sendmail configuration file to making changes to files of M4 macros, which is easier and more immediately understandable.

In this section, we will consider use of the M4 macros. The section on configuring a sendmail client that connects to our mail hub will deal directly with the /etc/sendmail.cf file, as that process is sufficiently simple to be easily accomplished without the aid of the M4 macros.

Generating A Config

The sendmail configuration apparatus is in the /cf subdirectory. Within this, the /m4 directory contains support routines, which should not be changed.

The /cf directory contains the configuration files themselves. They have .mc suffixes, and must be run through M4 to become complete. The resulting output should have a .cf suffix.

The /ostype directory contains definitions describing a particular operating system type. These should always be referenced using the OSTYPE macro in the .mc file.

The /domain directory contains definitions describing a particular domain, referenced using the DOMAIN macro in the .mc file. The /mailer directory holds descriptions of mailers, referenced using the MAILER macro in the .mc file.

In the /sh directory are the shell files used when building the .cf file from the .mc file in the cf subdirectory, while /feature holds special features that you might want to include. They should be referenced using the FEATURE macro.

In /hack you'll find local hacks from Berkeley (the home of sendmail), of no more than voyeuristic interest, if that. And in /siteconfig is the site configuration, ie, tables of locally-connected UUCP sites.

Typical File

The root of the configuration process is what might be called the base .mc file. This forms the starting-point for M4 directives which invoke other macro files. All of the definitions in our base .mc file in turn reference other .mc files. Order is important: follow that given here.

In what follows, I shall simply clarify what I have done here. For the full

information, see the README in the cf/ subdirectory.

```
VERSIONID('@(#)yourcompany.mc
8.5 (Berks) 10/8/97')
OSTYPE(mklinux) FEATURE(nouucp)
MAILER(local) MAILER(smtp)
```

The first line is for housekeeping, and puts the version line into the output file, so you can keep track of changes.

OSTYPE

The first macro defines our operating system. You must define an operating system environment, or the configuration file build will fail. For us, the OS is of course Linux; we use the file in ostypes called mklinux.mc. This contains details such as default file locations and other OS-specific material. It should not need to be changed.

FEATURE

The only feature we use is nouucp, which says "don't do anything special with UUCP addresses at all".

There's another feature, nullclient. This could be used to do what we will do later on by hand, ie, generate a stripped-down configuration file that does nothing but forward all mail to a central hub via a local SMTP-based network. The argument is the name of that hub.

MAILER

The MAILER macros use macro files to specify rules to handle one or more mailers. Here, we invoke definitions for a local mailer and an SMTP mailer. There are fewer mailers supported in this version than the previous version, owing mostly to a simpler world. As a general rule, put the MAILER definitions last in your .mc file, and always put MAILER(smtp) before MAILER(uucp) - several features and definitions will modify the definition of mailers, and the SMTP mailer modifies the UUCP mailer.

The "local" parameter specifies the local and prog mailers. You will almost always need these; the only exception is if you relay all your mail to another site. This mailer is included automatically.

SMTP is the Simple Mail Transport Protocol mailer. This does not hide hosts behind a gateway or another other such hack; it assumes a world where everyone is running the name server. This file actually defines four mailers: "smtp" for regular (old-style) SMTP to other servers, "esmtp" for extended SMTP to other servers, "smtp8" to do SMTP to other servers without converting 8-bit data to MIME (essentially, this is your statement that you know the other end is 8-bit clean even if it doesn't say so), and "relay" for transmission to our RELAY_HOST, LUSER_RELAY or MAILER_HUB.

sendmail.cf

The final sendmail configuration file is produced by invoking M4 with the .mc file given above as its argu-

```
### Defined Macros (1)
# The name of the mail hub
DRwotan.dunne.com
# The hub as it is known to the outside world
DHtiny1.demon.co.uk
# The local official domain name
Dj$w
# Our domain name
DDdunne.com
# Identity of the error message sender
DnMailer-Daemon
# Look of the Unix From line
DlFrom $g  $d
# The characters that separate address components
Do.:%@!^=/[]
# Default form for the sender's address
Dq<$g>

### Defined Classes (2)
# All possible names for local machine
Cw localhost donner

### Options (3)
# default delivery mode (in background)
Odbackground
# temporary file permissions—-0600 for secure mail
OF0600
# default UID & GID
Ou1
Og1
# level at which to syslog errors
OL9
# Wait for SMTP replies.
Or1h
# default messages to old style
OoTrue
# Replace unquoted spaces with a dot
OB.

### Header Declarations (4)
HFrom: $q
HReceived: by $j id $i; $b
H?x?Full-Name: $?x$x$.
H?D?Date: $a
H?M?Message-Id: <$t.$i@$j>

### Priorities (5)
Pspecial-delivery=100
Pfirst-class=0
Plist=-30
Pbulk=-60
Pjunk=-100

### Mailer Delivery Agent Definitions (6)
# Mailer to forward all mail to the hub machine
Mhub, P=[IPC], S=10, R=0, F=xmDFMuCX, A=IPC $h
# Sendmail requires these, but we won't use them
Mlocal,    P=/bin/mail,  S=0,  R=0,  F=lsDFMShP, A=deliver $u
Mprog,     P=/bin/sh,    S=0,  R=0,  F=lsDFMeu,  A=sh -c $u

### The Rules Sets (7)
S0 select delivery agent
R@$+       $#error       $: Missing user name
R$+        $#hub         $@$R $:$1   forward to hub

S3 preprocessing for all rule sets
R$*<>$*                  $n          handle <> error addresses
R$*<$*<$*>$*>$*          $2<$3>$4    de-nest brackets
R$*<$*>$*                $2          basic RFC822 parsing

S10 rewrite the sender for the hub
R$-                      $@$1@$H     user -> user@hub
R$-@$w                   $@$1@$H     user@local -> user@hub
R$-@$=w                  $@$1@$H     user@othernames -> user@hub
R$-@$=w.$D               $@$1@$H     user@domain -> user@hub

S1 dummy ruleset 1 (unused)
```

Figure 1 - A sample sendmail.cf file.

Linux Mail Hub

ment. The command looks like this:

m4 m4/cf.m4 config.mc > config.cf

where config.mc is the macro file we've developed above, and config.cf the output - the sendmail configuration file - which will end up as /etc/sendmail.cf.

A Sendmail Installation

Having completed the configuration, log in as root and type "make install" to install the new sendmail - having remembered to back up your old program in case anything should be out of order. The following are the files installed by sendmail.

/usr/sbin/sendmail

This is the actual sendmail program. There may be a symbolic link in sendmail's historic location, /usr/lib, pointing here, but /usr/sbin/ is now the actual location.

/etc/sendmail.cf

This is the configuration file that we generated.

/usr/bin/newaliases

This is a symbolic link to /usr/sbin/sendmail. When invoked by this name, sendmail will rebuild the aliases database.

/var/spool/mqueue

This is the post office, where in-going and outgoing mail is kept awaiting delivery. It should have mode 700, to prevent inquisitive users from peeking at other users' mail.

/etc/aliases

This is the systemwide aliases file.

/usr/lib/sendmail.hf

This is the help file for sendmail.

/etc/sendmail.st

This optional file can be used by sendmail to record statistics.

/usr/bin/mailq

This is a symbolic link to /usr/sbin/sendmail. When invoked under this name, sendmail prints the contents of the mail queue.

Starting Sendmail

You will most likely want to have the sendmail daemon started every time the machine boots up. This is done by adding a line to the appropriate rc file in /etc/rc.d. The exact configuration of /etc/rc.d varies between Linux distributions (I suppose if you are conservative, you may just have a simple BSD-style set-up, with only /etc/rc and /etc/rc.local), so I can't give exact instructions that are guaranteed to be applicable. The easiest thing is probably to put it in /etc/rc.d/rc.local. The line should look like this:

```
if [ -x /usr/sbin/sendmail ]
echo "sendmail"
/usr/sbin/sendmail -bd -q1h
fi
```

This checks to see if the file is there, then tells the system console what it's doing, and starts up sendmail in daemon mode (-bd), and sets it to process the mail queue every hour (-q1h).

The Mail Queue

The mail queue lives in /var/spool/mqueue (unless you have changed the default, which you really should not). All mail messages are held as two files here, one file being named dfXXXnnnnn, the other qfXXXnnnnn, where XXX is a three-letter sequence, nnnnn a five-number sequence, both being used simply to give every message a unique identifier. The qf file is the queue control file, containing the email message header and various processing information; the df file is the data file, and contains the body of the email message. There are other files, but they are transient and usually of interest only to sendmail.

Logging

Sendmail uses the syslog(8) facility. Usually, this is set up to log all sendmail messages to /var/log/maillog, which by default will record all mail that passes through sendmail.

Security

Sendmail has a reputation as a security nightmare, but this is largely undeserved, particularly with version 8, which solved a lot of the problems that previous versions did have. Much of sendmail's security is down to the system administrator. Some specific points to watch for are:

- Make sure the aliases file isn't writable except by trusted system personnel. This includes both the text and database version.
- Make sure that other files that sendmail reads, such as the mailertable, are only writable by trusted system personnel.
- The queue directory should not be world writable.

Clients

Providing email service to other machines in the network can be done in two ways. The first is to use SMTP to act as a mail hub that sends and receives Internet (and optionally local) mail on behalf of the other machines. Secondly, a POP service can be set up, where local users use client software on their computers to collect their mail via the POP3 protocol, and send mail via SMTP to the server.

/etc/aliases

Users on the local network must be identifiable by the sendmail process running on the server machine. In the case of POP mailboxes, this is done by creating a normal user account. In the case of Linux clients collecting mail through sendmail themselves, this is done by adding the appropriate alias to /etc/aliases. For example, on my local network, any mail arriving at my mail server (tiny1.demon.co.uk) for "bob" is sent on to bob@donner.dunne.com on my Internet network by the following line in /etc/aliases:

bob: bob@donner.dunne.com

The simplest way to make sure that mail comes back to the right place is to set the Reply-To header in all outgoing mail to point to the account on the mail hub, not the originating machine. This can be done in the options settings of your POP3 mailer, or will be handled for you by sendmail on a Linux client.

The Sendmail.cf File

Now, let's examine the sendmail.cf file itself - it's listed in Figure 1. This file can be kept quite simple when its only task is to relay all mail to another machine for further processing. Here is a minimal sendmail.cf file for installing on Linux boxes that talk to the mail hub. There isn't the space for a full run-down on the syntax of the file. For the full gen, consult the irreplaceable book *Sendmail*, published by O'Reilly & Associates.

Sendmail commands are usually one letter in length, and must be at the beginning of a line. Generally, there is no space between a command and its arguments.

The first part, Macros, shows variables (macros in sendmail parlance) being defined by use of the D, Define Macro, command. All the macros defined here are explained by comments on the line before them - a wise practise which should not be confined to example files.

The second section, Classes, is for a special type of variable, a class, that can hold multiple values. The command here is C. The class we are defining is W, which holds a list of alternative host names for the machine (that is, other than the FQDN).

The third section specifies sendmail options. These can be given on the command line, but as there are rather a lot, it makes more sense to have them in the file.

The fourth part specifies what headers must be in every mail message. These are the headers that sendmail will add if the MUA has not already done so.

The fifth section is a set of priority settings. Sendmail will by default process the mail in its queue in order of decreasing priority, beginning with "special-delivery". The level of priority is set by the MUA with the "Precedence:" header.

The sixth section defines a set of mailers that sendmail will use to actually deliver mail. Remember, sendmail is a Mail Transport Agent - it doesn't do the delivery itself. These lines all begin with M. Local and prog are mandatory. The real work here is done by the special mailer [IPC], which invokes internal sendmail routines rather than an external mailer program, to send all mail to a "smart host" using SMTP.

The seventh part is the heart of sendmail, the rule sets. These define the re-writing of addresses. The basic idea here is that there are two sides, a right-hand side and a left-hand side, where the RHS is a pattern to match against input, and the LHS is the transformation to effect upon the input if a match is made. LHS and RHS are separated by tabs; comments are in the third column.

Windows

Under Windows I use Eudora Lite, which is a freeware, cut-down version of Eudora Pro. It is a fine email client in its own right and is available from www.qualcomm.com. Setting it up is simply a matter of pointing it at the mail hub, then telling it the POP user name and password. One glitch I found was that not all options are saved to EUDORA.INI. Specifically, I had to set UseWinSock=1 and UseDialup=0 by editing the ini file, as changing these options from the menu had no effect.

Editing /etc/sendmail.cf By Hand

So, you have an existing sendmail installation and don't want to go to the bother of fussing about with getting the sendmail source, figuring out M4, etc? Well, while I can't hope to cover all the details of the sendmail configuration file syntax here, I can tell you the minimum changes you need to make to transform a generic sendmail.cf into one you can use.

Because of the complexity of this file, I will list here only the things that are absolutely essential to change. I will presume that there is already a suitable sendmail.cf on the machine (provided either by the Linux distribution, or from the sendmail sources).

The W macro contains any other names that this host is known by, besides the FQDN. For example:

Cwlocalhost wotan.dunne.com

The S macro can contain the name of a smart relay host, to which all non-local mail is forwarded without further ado. Some sites can deliver mail to the local network, but cannot look up hosts on the Internet with DNS. Usually such sites are connected to the outside world with UUCP. To ensure delivery of all mail, such sites need to forward all non-local mail over the UUCP link to a smart (or well-connected) host.

These are all the changes you absolutely have to make. The other parameters change how sendmail behaves, but should work on your site without modification.

PCNA

The Author

Paul Dunne (paul@tiny1.demon.co.uk) is a freelance writer and Unix/Internet consultant.

New Reviews from Tech Support Alert

Anti-Trojan Software Reviews

A detailed review of six of the best anti trojan software programs. Two products were impressive with a clear gap between these and other contenders in their ability to detect and remove dangerous modern trojans.

Inkjet Printer Cartridge Suppliers

Everyone gets inundated by hundreds of ads for inkjet printer cartridges, all claiming to be the cheapest or best. But which vendor do you believe? Our editors decided to put them to the test by anonymously buying printer cartridges and testing them in our office inkjet printers. Many suppliers disappointed but we came up with several web sites that offer good quality cheap inkjet cartridges with impressive customer service.

Windows Backup Software

In this review we looked at 18 different backup software products for home or SOHO use. In the end we could only recommend six though only two were good enough to get our “Editor’s Choice” award

The 46 Best Freeware Programs

There are many free utilities that perform as well or better than expensive commercial products. Our Editor Ian Richards picks out his selection of the very best freeware programs and he comes up with some real gems.