Exim Practical

1. Objectives

- Install Exim
- Test standard installation and default configuration
- Inspect and manage the mail queue
- Modify the runtime configuration
- Check and configure relaying
- Demonstrate retry mechanisms
- Process log data
- Configure and test address rewriting
- Add a system filter
- Re-configure as for a large installation
- Set up POP
- ... and anything else which suggests itself ...

2. Installation

Make a directory in which to build Exim, say **/usr/exim**. Copy the source

```
mv /usr/local/source/exim-3.14.tar.gz /usr/exim
```

Go to **/usr/exim** and unzip and untar the source. You should get a directory called **/usr/exim/exim-3.14**. Make that the current directory.

Create a new directory **/usr/exim/exim-3.14/Local**. Copy the file **src/EDITME** to **Local/Makefile**. Then edit that file, following the instructions inside it. You will need to set

```
EXIM_UID
EXIM_GID
SPOOL_DIRECTORY
```

as a minimum. Then obey

```
touch Local/eximon.conf
```

to create a configuration for the monitor (taking all the defaults). Now you can run

```
make
make install
```

You should end up with the Exim binaries in **/usr/exim/bin** and a default configuration file in **/usr/exim/configure**. Test by running

```
/usr/exim/bin/exim -bV
```

which should tell you Exim's version number.

3. Test standard installation and configuration

Substitute a real local use name for 'localuser'. Check what Exim will do with a local address:

```
exim -bt localuser
```

This tests the delivery routing for a local account. Try something that is in **/etc/aliases**:

```
exim -bt postmaster
```

If there isn't an alias for **postmaster**, create one. Normally there is also an alias for **root** – it is usual to treat root as an ordinary user. See what happens if you test an invalid address:

```
exim -bt invalid
```

Try a real local delivery:

```
exim -d localuser
This is a test message.
.
```

The `-d` flag turns on debugging output; you can increase the amount by putting a number after `-d`, up to `-d9` for general information. `-d10` adds tracing of filter files and `-d11` adds debugging output from DNS lookups. If the delivery succeeded, go and check the local user's mailbox. (If not, try to figure out why.) Check out Exim's logs:

```
cat /var/spool/exim/log/mainlog
cat /var/spool/exim/log/paniclog
```

The panic log should normally be empty. On a live system it is helpful to set up a **cron** job that mails you a warning if it ever finds a non-empty panic log.

Now would be a good time to fire up the Exim Monitor, to keep an eye on Exim's activity:

```
/usr/exim/bin/eximon
```

The upper window shows a 'tail' of the main log; the lower window shows the messages that are waiting in the queue. Expect both to be empty to start with.

If a **sendmail** daemon is running, kill it off. Use `ps` with `grep` to find it.

```
ps -xa | grep sendmail
kill process-id
```

Now you should be able to start an Exim daemon:

```
/usr/exim/bin/exim -bd -q15m
```

Look at the log in the monitor window to confirm that it has started successfully. The `-bd` option causes the daemon to listen for incoming SMTP calls, while the `-q15m` option causes it to start a queue runner process every 15 minutes. On a live system, this should happen automatically on a reboot as a result of the symbolic link from **/usr/sbin/sendmail**. You might need to edit **/etc/rc.conf** to set the queue run interval you want.

At this point it might be a good idea to change **/usr/sbin/sendmail** to be a symbolic link pointing to **/usr/exim/bin/exim**.

```
rm /usr/sbin/sendmail
ln -l /usr/exim/bin/exim /usr/sbin/sendmail
```

Exim is now fully installed.

Send a message to a remote address:

```
exim -d user@remote.host
This is a test message.
.
```

The `-d` flag causes it to display the SMTP dialogue. Now check that a remote host can send a message to your host.

4. Queue management

There are several command line options (and equivalent menu items in the monitor) for doing things to messages. To put a message on the queue without its being delivered, run

```
exim -odq address1 address2 ...
Test message.
.
```

The message will stay there until a queue runner process notices it. List the messages on the queue:

```
exim -bp
```

Freeze a message, for example:

```
exim -Mf 12jRno-0004fx-00
```

Do a manual queue run, with minimal debugging output:

```
exim -d -q
```

The frozen message gets skipped. Force a delivery of one message, with minimal debugging:

```
exim -d -M 12jRno-0004fx-00
```

This overrides any retry delay and also overrides freezing.

Other type of manual queue run (options starting with **-q**, **-R** and **-s**, and manual controls for messages (all starting **-M**) are listed in the manual.

5. Modifying the runtime configuration

To change the runtime configuration, you must edit **/usr/local/etc/exim/configure** and then HUP the Exim daemon:

```
kill -HUP `cat /var/spool/exim/exim-daemon.pid`
```

Confirm that it has restarted by checking the main log.

Here are some suggestions for configuration modifications:

- Add the options

  ```
  sender_verify
  receiver_verify
  ```

  and then try sending to an invalid address *from another host*.

- Limit the size of messages that Exim will handle:

  ```
  message_size_limit = 500K
  ```

  and check that it enforces the limit.

- Deliver into different mailbox files. For example,

  ```
  file = /home/$local_part/inbox
  ```

  to deliver into home directories.

- Convert **/etc/aliases** to a dbm file using the `exim_dbmbuild` utility:

  ```
  exim_dbmbuild /etc/aliases /etc/aliases.db
  ```

  Change the file name and search type in the **aliasfile** director so that it uses the new file. Check it out using **-bt** with **-d2**.

- Set up a number of different local domains:

  ```
  local_domains = a.example : b.example : ...
  ```

- Set up different alias files for each domain, e.g.

  ```
  /etc/aliases-a.example
  ```

  and reconfigure Exim to use them. Check that it is working correctly, including the case of unknown local parts in each domain.

- Set up a number of different local domains in a dbm file and get Exim to look them up by a setting such as

  ```
  local_domains = dbm;/usr/exim/local_domains.db
  ```

- Add a smartuser director that sends all undeliverable mail to postmaster.

6. Relay control

To demonstrate that Exim does not relay by default, even from the local host, try the following sequence of SMTP commands:

```
telnet 127.0.0.1 25
helo localhost
mail from:<root@local.domain>
rcpt to:<user@some.remote.domain>
```

This should cause a 550 error response. Type ‘quit’ to end the SMTP session. Edit the runtime configuration and add

```
host_accept_relay = 127.0.0.1
```

and HUP the demon. Now try the test again; this time it should accept the recipient. However, if you telnet to the host’s real IP address instead of 127.0.0.1, it will still refuse to relay.

Relay (and other policy) controls can be tested by running a command such as

```
exim -bh 192.168.23.45
```

This runs a fake SMTP session *as if* it had received a call from the host 192.168.23.45. You can enter into an SMTP dialogue, and as it proceeds, Exim outputs the tests it is making and the log lines it would write if this were for real. By this means, you can see if your policy settings are working the way you expect.

7. Demonstrating retry mechanisms

The easiest way to demonstrate what happens when Exim cannot deliver a message is to edit the configuration, and add to the **remote_smtp** transport the line

```
port = 3456
```

This make it try port 3456 instead of the SMTP port (25) when delivering, causing the remote host to refuse the connection (assuming you’ve chosen an unused port!) Send a message to a remote address and see what happens. Start a queue run

```
exim -q
```

and see what happens and what gets logged. Have a look at the message’s own **msglog** file. Use **exinext** to see when it is next scheduled to deliver:

```
exinext remote.domain
```

Change the retry rule to a shorter time; force a delivery attempt with **-M** or **-qf** and check the new retry time.

Remember to remove the setting of `port` when you have finished playing with retries.

8. Processing log data

To extract all information about a certain message, or a certain user’s messages, or messages for a certain domain, use, for example:

```
exigrep root /var/spool/exim/log/mainlog
```

That extracts all the log information for all messages that have any log line containing ‘root’. It’s a Perl pattern match, so you can use Perl regular expressions.

To extract simple statistics from a log, run

```
eximstats /var/spool/exim/log/mainlog | more
```

There are options for selecting which bits you don't want.

9. Configure and test address rewriting

Add to the rewriting section of the configuration

```
othername@otherdomain.com   postmaster@your.domain
```

Then send a message to **othername@otherdomain.com** and see what happens. You can test rewriting rules with -brw. Have a look at the chapter on rewriting and experiment with other forms of rule.

10. Add a system filter

Add a system filter file to the configuration:

```
message_filter = /etc/exim.system.filter
```

The filter file can contain things like

```
# Exim filter
if $h_subject: is "spam" then save /dev/null endif
```

(The first line is important.) Send a message with the subject ‘spam’ and see what happens.

11. Re-configure for a large installation

Change the local mailboxes so that they aren't all in one directory (see example on slide copies). Set

```
split_spool_directory
```

and observe its effect on the **/var/spool/exim** directory. Check that the monitor, and commands like

```
exim -bp
```

still work correctly. Also, if you are keen, verify that Exim transitions between split and unsplit spool without losing any existing messages.

12. POP

Install a POP server from a FreeBSD port and check it out.

* * *