

Exim Practical

1.) Build and install Exim using pkg_add

```
# pkg_add ftp://41.218.234.1/packages/!ll/xim"4.#$ 3.t%&

i . '(nsid ring t) fact t)at t) packag s ar l(cat d (ns(m r m(t s r* r
*i/ tc/mak .c(nf
+ ! , - E . _ , / - E _ 0 1 E . . / 2 E 3 4 ftp://41.218.234.1/usr/tftp%((t/

5.B: 6 ) n installing fr(m p(rt and supp(rt f(r m7s8l d( as % l(9
# cd /usr/p(rts/mail/ xim
# mak :
6 / - ; _ + < , = > 3 7 s :
6 / - ; _ ' 0 5 - E 5 - _ , ' ! 5 3 7 s :
install cl an
```

2.) . plac , ndmail 9 it) Exim

```
# / tc/rc.d/s ndmail st(p
# *i / tc/rc.c(nf
s ndmail_ na%l 3?5 0 5 E?
s ndmail_su%mit_ na%l 3?5 0?
xim_ na%l 3?<E,?
```

```
Edit / tc/mail/mail r.c(nf and c)ang lin s t( r fl ct lin s as % l(9
```

```
s ndmail /usr/l(cal/s%in/ xim
s nd"mail /usr/l(cal/s%in/ xim
mail8 /usr/l(cal/s%in/ xim "%p
n 9 alias s /usr/%in/tru
```

```
# ,tart Exim 9 it) c(mmands as % l(9
```

```
/usr/l(cal/ tc/rc.d/ xim start
```

```
# /f xim is running /usr/l(cal/ tc/rc.d/ xim status s) (uld r turn a pr(c ss id
```

```
# ' r at / tc/p ri(dic.c(nf and add t)is lin s
```

```
dail7_status_includ _su%mit_mail83?5 0?
dail7_cl an_) (ststat_ na%l 3?5 0?
```

```
# 5(9 d( t) %asic t st % l(9 t( mak sur * r7t)ing is fin
#/usr/s%in/s ndmail "%1
```

<(u s)(uld g t t) sam (utput as running /usr/l(cal/s%in/ xim ""1

2.) - sting Exim

5.B !ll t st s)(uld % d(n using t) us r inst

@irst\ ') ck 9)at Exim 9ill d(9it) l(cal addr ss
B xim ""t inst

-)is t sts t) d li* r7 r(uting f(r a l(cal acc(unt. Examin (utput g(tt n

-r7 t) d li* r7 r(uting f(r a n(n" xist nt l(cal us r acc(unt

B xim ""t CunkCunkkunk

-r7 s(m t)ing t)at is in / tc/alias s

B xim ""t p(stmast r

5.B: Exim d(sn)t all(9 mail d li* r7 t(r((t mail%(x % caus (f s curit7 r as(ns.
; nc 9 n d t(mak r((t an alias f(r t) ,7st ms !dministrat(r mail acc(unt
%7 diting / tc/alias s and unc(mm nting lin % l(9

#r((t: m :m7.d(main

and c)ang it E r m m% ring t(r m(* t) initial D#D) t(:

r((t: inst

n(9 tr7 t)is again

B xim ""t p(stmast r

- st l(cal d li* r7 and pass a m ssag dir ctl7 t(xim 9it)(ut using an + F!

B xim "" "(df inst

t st m ssag .

.

5.B: -) m ssag is t rminat d %7 a lin t)at Cust c(ntains a 2 (t.

-) "*" (pti(n turns (n us r * rificati(n (utput\ 9)ic) s)(9 7(u c(pi s (f Exim)ds

l(g lin s

-) "(df (pti(ns r 8u st Df(r gr(und) d li* r7\ 9)ic) m ans t)at t) xim

c(mmand 9(n)t r turn until t) d li* r7 is c(mpl t .

Check what is in Exim's logs:

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

The panic log should normally be empty, and if nothing has ever been written to it, it will not even exist. *Tip:* 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.

If you get a permission error, make sure that your username is in the 'mail' group, then logout and login again to become a member of that group.

If the delivery succeeded, you should see two lines in the main log, one containing `<=` for the message arriving, and one containing `=>` for the delivery.

Now go check the local user's mailbox:

```
$ ls -l /var/mail/inst
$ cat /var/mail/inst
```

If the delivery didn't succeed, you need to find out why. If the information in the log doesn't help, you can try the delivery again, with debugging turned on:

```
$ exim -d -odf inst
<there will be output from Exim here>
This is another test message.
.
```

The `-d` option turns on debugging, which gives a lot more information than `-v`. You need to be an Exim administrator to use `-d`. If you get a 'Permission denied' error, check that you are a member of the "mail" group.

If you are logged on as `inst`, you can use the `mail` command to read the mail in the usual way. You could also try sending a message from the `mail` command.

The next thing is to test whether Exim can send to a remote host. The speed of this may vary, depending on the state of the network connection. In what follows, replace `user@remote.host` with your home email address.

First, check that Exim can route to the address:

```
$ exim -bt user@remote.host
```

Now send a message to the remote address:

```
$ exim -v -odf user@remote.host
This is a test message.
```

This time, the `-v` option causes Exim to display the SMTP dialogue as well as the log lines. If you can, check that the message arrived safely. If there are problems, see if you can figure out what went wrong and why.

You won't be able to receive messages from a remote host until you start the Exim daemon:

```
# exim -bd -q30m
```

The `-bd` option causes the daemon to listen for incoming SMTP calls, and the `-q30m` option causes it to start a queue runner process every 30 minutes. On a live system, starting the daemon should happen automatically on a reboot, by putting the following entry in `/etc/rc.conf`:

```
exim_enable="YES"
```

Once you've done this, you can use `/usr/local/etc/rc.d/exim {start|stop|status}` as usual.

Use telnet to check that the daemon is accepting SMTP calls:

```
$ telnet localhost 25
```

You should see an Exim greeting message. Use QUIT to exit.

Now check that a remote host can send a message to your host, and see how Exim logs what happens. If that succeeds, you have a working basic installation correctly installed.

Try sending to an invalid address from a remote host, and see what error message you get, and how Exim logs this case. Look in both **mainlog** and **rejectlog**.

Queue management tests

There are several command line options 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 stays on the queue until a queue runner process notices it.

List the messages on the queue:

```
$ exim -bp
```

Do a manual queue run, with minimal verification output:

```
$ exim -v -q
```

(Without `-v` you won't see any output at all on the terminal, but there will be entries in the log.)

Checking relay control

To demonstrate that Exim will relay by default via the loopback interface, try the following sequence of SMTP commands. Wait for Exim to respond to each command before typing the next one. Substitute the number of your pc for **X**:

```
$ telnet 127.0.0.1 25
ehlo localhost
mail from:<inst@pcX.ghe0.dns.gh>
rcpt to:<inst@pcX.ghe0.dns.gh>
rcpt to:<inst@some.remote.domain>
```

You should get an OK response to all the SMTP commands. Type ``quit'` to end the SMTP session without actually sending a message.

Now try the same thing, but use your host's IP address instead of 127.0.0.1.

```
$ telnet xx.xx.xx.nn 25
ehlo localhost
mail from:<inst@pcX.ghe0.dns.gh>
rcpt to:<inst@pcX.ghe0.dns.gh>
rcpt to:<user@some.remote.domain>
```

In this case, you should get the error message

```
550 relay not permitted
```

for the second `RCPT` command, which is the one that is trying to relay. The first `RCPT` command should be accepted, because it specifies a local delivery. You could also try telnetting from an external host and running the same check.

Processing log data

Run **exigrep** to extract all information about a certain message, or a certain user's messages, or messages for a certain domain. For example:

```
$ exigrep inst /var/log/exim/mainlog
```

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

To extract simple statistics from a log, run

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

There are options for selecting which bits you don't want. Details are in the manual. If you have time, experiment with the options for outputting the statistics as HTML.

Part 3. Changing the configuration

To change Exim's runtime configuration, you must edit `/usr/local/etc/exim/configure` and then "HUP" the Exim daemon (as root) - that is, send it a HUP (HangUP) signal. The daemon stores its process id (pid) in a file, in order to make this easy. Using this signal is less disruptive than completely stopping and starting the daemon. You can use these commands to send the signal:

```
# cat /var/run/exim.pid
# kill -HUP nnnn
```

where *nnnn* is the pid from the previous line. Alternatively, you can use the startup script installed by the port to do this for you:

```
# /usr/local/etc/rc.d/exim reload
```

You can confirm that the daemon has restarted by checking the main log.

The following sections contain some suggestions for configuration modifications that you can try, just to get a feel for how the configuration file works. You do not have to stick rigidly to these examples; use different domain names or user names if you want to.

Adding more local domains

Edit the configuration, and change the `local_domains` setting so that it looks like this:

```
domainlist local_domains = @ : pcX.ghe0.dns.gh : bubu.ghe0.dns.gh
```

where **X** is the number of your host. Remember to HUP the daemon afterwards. Now you have a new local domain. Try sending it some mail:

```
$ mail inst@pcX.ghe0.dns.gh
```

Check that it arrives in your mailbox.

If you want to add a lot of domains, or if you want to keep changing them, it is easier to keep the list of domains in a file instead of in the Exim configuration. (You can also keep them in several different kinds of database, such as LDAP or MySQL, but we don't cover that in this workshop.)

4. Relaying from another host

In section 2 above, there is test to demonstrate that relaying is blocked if you connect to your host's IP address.

We are now going to remove this block by changing a line in the configuration to let all the classroom hosts relay through your host. Change this line:

```
hostlist relay_from_hosts = 127.0.0.1
```

to

```
hostlist relay_from_hosts = 127.0.0.1 : 41.218.234.0/24
```

where *41.218.234.0/24* is the classroom network. (Don't forget to HUP the daemon.) Then try the telnet test from section 2 again. This time it should accept the request to relay. Ask one of the other students to try relaying through your host -- it should work. If you can, telnet from a host outside the classroom network, and confirm that relaying is still blocked.

Allowing relaying to specific domains

The default configuration contains the line

```
domainlist relay_to_domains =
```

This defines domains to which your host will relay, wherever the message comes from. As you can see, the default list is empty, so no domains match. Add some domains to this line. For example, add the domain of your home email where I've put "example.com"

```
domainlist relay_to_domains = example.com
```

Now we need to test that Exim will indeed relay to those domains (but not to others) from a host that does not match **relay_from_hosts**. Exim has a testing facility that lets you simulate an SMTP call from a remote host. Run it like this:

```
$ exim -bh 192.168.1.1
```

You will see some debugging output, and then an SMTP greeting line. Now type SMTP commands, waiting for a response between each one:

```
ehlo testhost
mail from:<inst@pcX.ghe0.dns.gh>
rcpt to:<user@somewhere.com>
rcpt to: user@some.other.domain
quit
```