Electronic Mail

COSC301 Laboratory Manual

E-mail is a large and rather complex topic, but is very important. Being such an important part of everyday life, e-mail has a lot of things to consider. We could talk about many aspects of e-mail administration, but we'll stick to the basics.

By the end of this lab, you should have a good idea of what is involved when setting up a very basic e-mail server, how mail is delivered and received, and how such things as mailing lists can be set up. We don't have time today to cover other important e-mail issues such as SPAM and Virus prevention.

1. Preliminary Configuration

1. In our case, we're going to have all our e-mail services run on our Server1, but this needn't be the case.

2. Add an MX record with priority 0, for localdomain, that points to our servers IPv4 and IPv6 address. The second argument of an MX record must not be a name with a CNAME record, but rather something that has an A or AAAA record[1].

3. Add an alias smtp.localdomain which points to the e-mail server, for both IPv4 and IPv6. Remember, this is an alias, so don't add an entry to the reverse zone.

4. Add another alias called pop3.localdomain, in the same way as the previous alias.

5. Make the changes take effect, and test to make sure the mappings are correct and working.

```
$ dig +short -t MX localdomain
0 smtp.localdomain.
$ dig +short -t ANY server1.localdomain
192.168.1.1
fd6b:4104:35ce::1
$ dig +short -t ANY smtp.localdomain
192.168.1.1
fd6b:4104:35ce::1
$ dig +short -t ANY pop3.localdomain
192.168.1.1
fd6b:4104:35ce::1
```

6. Add a usercode for a fictitous user called bob on Server1. Use **adduser bob** for this. We shall be using this user later to test our delivery.

2. Install and Configure Exim

> **Note**
>
> This section will be done entirely on Server1, which in our case is the e-mail server.

Exim is a MTA (Mail Transfer Agent) that is largely Sendmail compatible. Sendmail is a very old (but still actively maintained) MTA that is very powerful. Unfortunately, Sendmail has a

[1] See RFC1912.

spotty security history, and its configuration takes a lot of knowledge to do well. Indeed, its been said to require “black magic” to know how to do well, and is a valuable skill for a Unix systems administrator to add to their CV, as Sendmail is one of the most flexible MTAs on the planet.

Exim has a large user base, as it is the default MTA used in the Debian GNU/Linux distribution. Having a large user base means that its relatively stable, and that there is a large community you can ask for help. Exim is also easy to learn, while still being reasonably flexible.

You should have already installed Exim when **logcheck** was installed. However, if you haven't installed it yet, use **apt-get install exim4** to install.

Reconfigure the service using **sudo dpkg-reconfigure exim4-config**. Respond to its questions with the following answers:

Configuration type
Respond with “internet site; mail is sent and received directly using SMTP”.

Note that networks such as our small domain, that has an invalid domain name (“localdomain”), would typically be better off using “mail sent by smarthost; received via SMTP or fetchmail”. This is a useful configuration for a gateway machine such as ours which doesn’t have a public domain. Any message that gets forwarded will need to have its addresses rewritten, so the invalid domain “localdomain” is not seen. It would be changed to something more appropriate, such as our publically addressable e-mail account.

System mail name — used to qualify addresses without a domain.
In our case, we want to consider all our mail as `user`@`domain`, and our domain is `localdomain`, so respond with `localdomain`. Normally, it would be something like `example.com`, but we only have one DNS label.

IP addresses to listen on
We would like to listen on the inside interface and the loopback interface, both for IPv4 and IPv6. In our case, where we don’t actually accept mail coming in from the internet, only be able to send e-mail, we don’t need to listen on the outside interface.

We are asked to respond with a semi-colon seperated list of addresses. Respond with the following:

```
127.0.0.1 ; ::1 ; 192.168.1.1 ; fd6b:4104:35ce::1
```

Other destinations for which mail is accepted
We don’t operate as a final destination for any other domains, so we don’t generally need to add anything here. We already accept e-mail for `mailbox`@localdomain, but perhaps we might also allow `mailbox`@server1.localdomain. However, possibly because we only have one DNS label in our domain, we also need to add `localdomain`, so respond with `localdomain;server1.localdomain`.

Domains to relay mail for
Leave this empty, as this list is for *other* networks which we forward (relay) mail for. We might do this if we were an e-mail gateway or acting as a backup MX (mail exchanger) for someone else’s domain. In other words, what goes here is the list of networks that we may accept mail for, but are not the final destination.

Machines to relay mail for
Generally, this would be set to your internal network, because client machines generally shouldn't be able to send e-mail without going through the mail-server[2], so respond with the following:

```
192.168.1.0/24 ; fd6b:4104:35ce::/64
```

Keep number of DNS-queries minimal?
No

Delivery method for local mail:
Accept the default, as that is what a lot of local tools expect. There are good reasons for using the alternative however.

Split files?
Yes

Before we go on and have a look at what we've just done, let's setup the system aliases so common e-mail addresses, such as postmaster and hostmaster exist. Also, it gives us a way of redirecting system accounts, such as root, to a real person, who should regularly check their e-mail. The right hand side of an alias can either be a qualified or unqualified e-mail address, such as `person`, or `person@example.com`. You could even have multiple right hand sides, useful if there are a few administrators, of if you want to make a small, manually managed mailing list.

Make all the system mail (ie. all the mail for `root` and any system accounts) eventually reach your account (ie. `mal`) on the mailserver. Open `/etc/aliases` and add the following line (it may be already there):

```
root: mal
```

In some versions of Ubuntu (and other Linux versions) it may already be there by default

Note the mailboxes ("users") called postmaster, hostmaster, webmaster, abuse, security etc. These are well-known aliases that are associated with e-mail, DNS, WWW, and security administrators. There are expected to be monitored e-mail accounts so any problems can be properly reported. You will recall that we have used the hostmaster address when defining our DNS zone files.

If these entries don't exist, then create them now. Eventually the mail is sent to `mal`, the administrator.

Okay, now that we've dealt with our aliases for the moment, let's check what happened when we answered all those questions previously. Debian manages the contents of the Exim configuration file for us, to enable as much management to be done via package management drop-ins, and the Debian Configuration (Debconf) system. It was Debconf that was asking us those questions via the menu system earlier. **Read from section 2 to the end of section 2.1 of `/usr/share/doc/exim4-base/README.Debian.gz`; use zless to view it.** You need to understand how the configuration files can be managed as there are a number of choices with different tradeoffs.

Write down the three ways that the Exim configuration can be managed under Debian (hint: one of the ways is to not have Debconf manage the configuration at all).

[2]This enables egress scanning for viruses, etc. To implement it, a firewall would be required to ensure only the e-mail server can create outgoing e-mail connections.

Have a look at the autogenerated file that Exim is given as its configuration file: `/var/lib/exim4/config.autogenerated`. You will notice the file makes heavy use of conditional configuration macros, which makes the file harder to read. Unfortunately, these conditions[3] are interpreted by Exim itself, and not some preprocessor, so it is harder for us to see the file after the all of the macros have been evaluated. For the sake of comparison, have a look at the example configuration file (that doesn't use macros) in `/usr/share/doc/exim4-base/examples/example.conf.gz`, to get a feel for what the end-result looks like.

Thankfully, we can also ask Exim to dump its configuration elements to stdout using the `-bP` command-line option. Run **exim4 -bP | less** and have a look.

Like most packages that contain system services, the Exim package contains a startup script that will be run at boot-time. But now we need to restart Exim to affect our changes. However, before we do this, it is a good practice to check our configuration file (if such a facility is provided). Check the file, and then restart Exim using the following commands. You will notice that the startup script executes **update-exim4.conf** to (re)-generate the configuration file and store it in the correct place.

```
# /etc/init.d/exim4 stop
$ pstree
Ensure that there are no exim4 processes, if there are…
# killall exim4

# /etc/init.d/exim4 start
$ pstree
Check there is ONE exim4 process

# lsof -Pni | grep exim4
exim4 … IPv4 … TCP 127.0.0.1:25 (LISTEN)
exim4 … IPv6 … TCP [::1]:25 (LISTEN)
exim4 … IPv4 … TCP 192.168.1.1:25 (LISTEN)
exim4 … IPv6 … TCP [fd6b:4104:35ce::1]:25 (LISTEN)
```

Exercise

As an exercise make sure the output of **lsof** is as expected.

Being a high-volume service, e-mail servers often do their own logging, rather than using syslog. Exim will log to files in `/var/log/exim4/`. The `mainlog` file is the standard file you will want to look at. There may be a `paniclog` file; it is generated if the server fails to start or crashes for some reason.

For the remainder of this lab, you may like to run **tail -f /var/log/exim4/mainlog** in a spare virtual console to keep an eye on your logs. The `-f` option causes **tail** to not exit, and keep outputting any new entries in the file.

Exercise

As an exercise check your logs to ensure the server started cleanly. This will show that you are able to find logs, which are a valuable resource.

[3] The macros such as `.ifndef`, which read as "if not defined"

3. Testing Submission by Hand Using SMTP

In this section, we're going to send some test messages to `bob@localdomain`, from the client machine, which should appear in his mailspool file `/var/spool/mail/bob` on the server. This file is in *mbox* format, which is the common form on Unix systems. Bob (and you) will read your e-mail on the server using the MUA called **mutt**, which you will first need to install on the server:

```
# apt-get install mutt
```

Use **telnet** from Client1 to connect to port 25 (the SMTP port), on the mailserver for localdomain. Here is a transcript of what you can type, this is speaking the SMTP protocol. This is what a MTA would do when sending an e-mail. As you do this, keep an eye on the logs being generated on the server.

```
mal@client1:~$ dig +short -t MX localdomain
0 smtp.localdomain.
mal@client1:~$ telnet smtp.localdomain 25
Trying fd6b:4104:35ce::1...
Connected to smtp.localdomain.
Escape character is '^]'.
220 server1.localdomain ESMTP Exim …
EHLO client1.localdomain
250-server1.localdomain Hello client1.localdomain [fd6b:4104:35ce::a00:27ff:fe28:370e]
250-SIZE 52428800
250-PIPELINING
250 HELP
MAIL FROM:mal@localdomain
250 OK
RCPT TO:bob@localdomain
250 Accepted
DATA
354 Enter message, ending with "." on a line by itself
From: mal@localdomain
To: bob@localdomain
Subject: Hi Bob
Blank line seperates header from body.
How are you today?
.
250 OK id=195bKY-0000f3-Uq
QUIT
221 server1.localdomain closing connection
Connection closed by foreign host.
```

If you look at your server logs, you should now see that the delivery succeeded; the log enty will look similar to the following:

```
timestamp 1OEATZ-00003WL-3M <= mal@localdomain H=client1.localdomain ↵
   [fd6b:4104:35ce:0:a00:27ff:fe28:370e] P=esmpt S=size
timestamp 1OEATZ-00003WL-3M => bob <bob@localdomain> R=local_user ↵
   T=mail_spool
timestamp 1OEATZ-00003WL-3M Completed
```

Now log into Server1 as Bob, and have a look at the users spool file. This file is in mbox format, and stores all the messages in the users Inbox.

```
bob@server1:~$ cd /var/mail/
```

```
bob@server1:/var/mail$ ls
bob   mal
bob@server1:/var/mail$ less bob
From mal@localdomain Tue Jan 06 16:40:33 2015
Return-path: <mal@localdomain>
Envelope-to: bob@localdomain
Delivery-date: Tue, 06 Jan 2015 16:40:33 +1200
Received: from client1.localdomain ([192.168.1.11])
        by smtp.localdomain with esmtp (Exim 4.44)
        id 195bKY-0000f3-Uq
        for bob@localdomain; Tue, 06 Jan 2015 16:40:33 +1200
From: mal@localdomain
To: bob@localdomain
Subject: Hi Bob
Message-Id: <E195bKY-0000f3-Uq@smtp.localdomain>
Date: Tue, 06 Jan 2015 16:40:33 +1200

How are you today?
Use q to exit less
```

Exercise

As an exercise make sure at least you can see one message as above (if you did it correctly the first time there will be only one message). Notice the line that starts with `From`, but doesn't have a colon after it. That is part of the mbox format, and is there to serve as a message seperator.

If it didn't work, consult the Section 4, "Troubleshooting" for some troubleshooting advice.

On Unix systems, once the MUA has closed the spool file, it removes any seen messages into the user's mail folder in their home directory, such as `~/mail/` or `~/Mail/`. However, a lot of systems are migrating away from the mbox format to a more robust Maildir format, so we don't go over it much here. Most users won't read their e-mail this way anyway.

Exercise

Note the headers that were added to the message by the MTA. Now, as bob on the mailserver, start **mutt** (you installed it previously) as an exercise and navigate your way to your inbox. View the message (**Enter**), and exit (**q**, **q**, **y** to fully quit). Look at the raw file again, you should notice that some new headers have been added. What do you think the Status header is used for?

4. Troubleshooting

This section is useful if you are having trouble getting Exim to deliver messages correctly. It addresses inspecting the mail queue, viewing the messages in the mail queue, and what to do when you think you may have fixed the problem.

Beware restarting

When reconfiguring Exim, experience has shown that it pays to explicitly stop the Exim process, reconfigure it, stop it again, check that all Exim processes have stopped, then start it. We have found in the past a simple restart can leave the old process in place, as well as launching new processes, which means your old (possibly broken) configuration might still be in force. The new instance might hang around in the background complaining that it can't bind to the SMTP port.

Managing the mail queue

Normally, the mail queue will be empty, so if you use **sudo mailq** to inspect the queue, you would normally expect the command to produce no output, in which case the queue is empty.

On the other hand, the queue could have data in it. The queue is simply the messages that have yet to be delivered. Having some messages in the queue on a production mail-server is to be expected. Some mail-servers might be too busy to process incoming messages, so they have to be held a while on the sending mail-server. These messages will be retried later. Here is some sample output showing two messages in the queue, including how long they have been in the queue, and the *message ID*. The **mailq** command would be run on Server1.

```
No output means the queue is empty.
# mailq
27m   450 1Au2JH-0000ex-Pw <root@localdomain>
          staff@localdomain

25m   443 1Au2Kz-0000fD-Kb <root@localdomain>
          staff@localdomain
```

We can force Exim to attempt a delivery attempt immediately, instead of waiting according to its normal retry timers. We do this using the message IDs.

```
# exim4 -M 1Au2JH-0000ex-Pw 1Au2Kz-0000fD-Kb
# mailq
No ouput, queue is now empty, everything worked.
```

Other messages, however, might end up as *** `frozen` ***, in which case the mail server can neither send the message, and neither can it send a bounce. This is often because of a misconfiguration. Frozen messages will remain in the queue awaiting manual intervention.

Diagnosing what is wrong with a message can be done by inspecting parts of the message. We could inspect the message headers, the message body (which we generally wouldn't want to do out of respect for the privacy of the parties involved), or the logs that are associated with the message delivery.

```
Display message logs
# exim4 -Mvl message-id
…
Message headers
# exim4 -Mvh message-id
…
Message body (last resort, consider privacy)
# exim4 -Mvb message-id
…
```

We can use the command **exiqgrep -zi** to output the message IDs of all frozen messages in the mail queue. We can feed this into various **exim4 -Mxx** commands, which operate on messages given a list of message IDs as arguments. Here is a list of examples:

```
Thaw all frozen messages and attempt delivery on them
# exiqgrep -zi | xargs exim4 -M
If they are still frozen, the problem likely isn't solved yet.

Remove all frozen messages (assuming you know they are all pointless)
# exiqgrep -zi | xargs exim4 -Mrm
```

5. Self-assessment

1. You must be able to send a message using **telnet** as a client, and read the message.

2. Make sure that you can inspect the message log.

6. Inspecting Headers

Ever wondered why an e-mail addressed to Joe Smith ended up in your Inbox? The envelope header isn't necessarily the same as the message header.

Consider the following message being sent from the e-mail address zeek@zebedee.com, to the members of a mailing list nohomers@hodum.com. The user bigtim99@ezisp.net is a member of this list. What is shown is a diagram of the communication between the SMTP client (in this case, the server responsible for the mailing list, which is postie.hodum.com), and the SMTP server (in this case, the server for the recipients mail-box, mailhub1.ezisp.net).

```
postie.hodum.com connects to mailhub1.ezisp.net
S→C 220 mailhub1.ezisp.net ESMTP Exim 4.44 Mon, ↵
    14 Mar 2005 16:43:12 +1200
C→S EHLO postie.hodum.com
S→C 250-mailhub1.ezisp.net Hello postie.hodum.com ↵
    [123.123.123.123]
...
C→S MAIL FROM:zeek@zebedee.com   Envelope header
S→C OK
C→S RCPT TO:bigtim99@ezisp.net   Envelope header
S→C OK
C→S DATA
S→C 354 Enter message, ending with "." on a line by itself
C→S From: zeek@zebedee.com       Message header
C→S To: nohomers@hodum.com       Message header
C→S Subject: Minutes of the last meeting...
...
```

The `To:` header in the message might say something different compared to the envelope (`RCPT TO:`). However, this can be normal and legitimate. In the case of mailing-list the `To:` header would be the mailing-list address, not the individual recipient this copy of message is being sent to.

When the message to a mailing-list (or in general, any alias) gets received by the mail server, the server will *explode* the message, so one message will go to each member of the list, with the envelope header set to that invidual's e-mail address, but the message header (`To:`) set to the mailing-list address.

It is very difficult to trust any headers in an e-mail message. As an administrator, it is imperative that you understand the concept of forging headers, as it is one of the common things that spammers and fraudsters do. Envelope (`MAIL FROM:`) and message headers (both `From:` and `To:`) are commonly forged to try and hide their tracks. Received headers are also routinely manipulated to make it harder for people to trace the spam back to where it came from. For this reason, be polite if complaining to the who you think might be responsible for the sending network, as it's not uncommon to be mistaken.

Using **telnet**, try to forge a message into Bob's account that says it's from someone else. Try to make it look like it really did come from that person. As bob on the mailserver (Server1), read the message, inspect the headers, and see if some of your peers can spot if it is genuine

or fake. Sites should have a monitored e-mail address abuse@domain, that e-mail messages about abuses such as spam or intrusion activity should be sent to.

To view all headers in Mutt, use **h** when viewing a message. All mailers have (or at least should have) the ability to view all headers.

Lets say that Bob received a message that was supposedly from his boss telling him to let in a technician into the computer room. Lets assume that Bob doesn't question this. The technician is really an intruder, using *social engineering* to try to get Bob to let him in, so he can steal business secrets, which he sells to their competition.

How can incidents like this, and others, be prevented? Come up with two suitable answers: one technical, and one social.

7. POP3 Server

Install the software used in this section on the mail server. **apt-get install dovecot-pop3d**

Setting up a POP3 server is very easy. We only need to install a suitable package, such as Dovecot. There are other alternatives, but we will not cover the topic much in this lab. Dovecot POP3d will run as a standalone service, although POP3 servers on small networks could run from inetd because they don't take long to start up. There are other things we could configure on the POP3 server, such as enabling the use of encryption for sending data over the network; but for now, we shall keep it very simple.

Ensure that `/etc/dovecot/dovecot.conf` contains only pop3 (we aren't going to setup POPS --- the SSL-wrapped POP3 as it adds more complexity that we need at the moment). Save and exit, then restart the dovecot service, using the techniques you should by now be familiar with.

Verify that something is listening for POP3 connections:

```
# lsof -i :pop3
COMMAND    PID    USER   FD   TYPE DEVICE SIZE NODE NAME
dovecot   5200    root    5u  IPv4  13788       TCP *:pop3 (LISTEN)
```

Hmmm, seems its only IPv4. Edit the configuration file again and search for IPv6. You should see a descriptive comment documenting the `listen` configurable. Set it as suggested to enable IPv4 and IPv6 access:

```
listen = *, ::
```

Exercise

Save and exit the configuration file, restart Dovecot (you should be able to figure out how to do this by now). As an exercise check, using **lsof**, that Dovecot's processes are IPv6 enabled. Find out what is listening on the POP3 port (TCP port 110) using the command below.

```
# lsof -Pni TCP:110
…
```

Beware that POP3 (as well as SMTP) is generally a clear text protocol, which means passwords can be captured fairly easily. Because this is a Bad Thing, the protocol has been extended to use other authentication protocols instead of just plaintext username and password. A discussion of this is beyond this humble lab though.

The Dovecot server is able to get e-mail from many different places, and so we need to tell it where it can find the messages for each user (it can do auto-detection, but this can fail for some users). Search the Dovecot configuration files (in `conf.d`), and ensure the `mail_location` variable is set to:

```
mail_location = mbox:~/mail:INBOX=/var/mail/%u
```

Restart Dovecot.

POP3 servers often have a DNS alias that makes pop3 a valid hostname in the DNS. We've already done this earlier in this lab, so pop3.localdomain should resolve to be Server1's addresses:

```
$ host pop3.localdomain
pop3.localdomain has address 192.168.1.1
pop3.localdomain has IPv6 address fd6b:4104:35ce::1
```

Now we better check to see if our POP service works. Since POP is used for pulling mail off the mail server, send a couple of simple messages to your mailbox on the server (using either **mail**[4] or **mutt**), so you have something interesting to look at. Now lets speak POP to the server. Do this from Client1.

```
$ telnet pop3.localdomain 110
Trying fd6b:4104:35ce::1...
Connected to pop3.localdomain.
Escape character is '^]'.
+OK Dovecot ready.
USER bob
-ERR Plaintext authentication disallowed on non-secure (SSL/TLS) connections.
QUIT
+OK Logging out
Connection closed by foreign host.
```

Ah, Dovecot has disabled plain-text authentication by default, which is a good policy (secure by default: if we want to have to have a less secure system, we have to make it that way deliberately).

For our purposes though, we want to have plain-text authentication enabled, to show how it all works. Search the Dovecot configuration files again and uncomment the `disable_plaintext_auth` line and set it to `no` explicitly.

Now restart Dovecot, and try connecting again from the client.

```
$ telnet pop3.localdomain pop3
Trying fd6b:4104:35ce::1...
Connected to pop3.localdomain.
Escape character is '^]'.
+OK Dovecot ready.
USER bob
+OK
PASS bob's password
+OK Logged in.
LIST
+OK 3 messages:
1 486
2 486
3 673
```

[4] use **apt-get install mailx** if not available.

```
.
RETR 3
+OK 673 octets
Return-path: <mal@localdomain>
Envelope-to: bob@localdomain
Delivery-date: Thu, 06 Jan 2015 23:15:24 +1200
Received: from mal by localdomain with local (Exim 4.60)
        (envelope-from <mal@localdomain>)
        id 1HkIEW-0001N5-Pq
        for bob@localdomain; Sat, 05 May 2007 23:15:24 +1200
Date: Thu, 6 Jan 2015 23:15:24 +1200
To: bob@localdomain
Subject: Test from mutt
Message-ID: <20070505111524.GA5271@localdomain>
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Disposition: inline
User-Agent: Mutt/1.5.11
From: "Miss A. Laneous" <mal@localdomain>

Hi Bob, its me, your old friend muttley.

Goodbye.
>From mal

.
RETR 2
…
.
DELE 2
+OK Marked to be deleted.
DELE 3
+OK Marked to be deleted.
LIST
+OK 1 messages:
1 486
.
QUIT
+OK Logging out, messages deleted.
Connection closed by foreign host.
```

The command structure should be obvious, you should be able to guess what `USER` `PASS` and `LIST` do. `RETR` retrieves a message (using the number output by `LIST`), and `DELE` deletes a message from the server.

As with all clear-text protocols containing private data, it would be wise to use some sort of secure tunnel, such as SSL. Some people use **ssh** to set up a secure tunnel to the server. We'll cover the use of **ssh** for port forwarding in a later lab.

7.1. Self-assessment

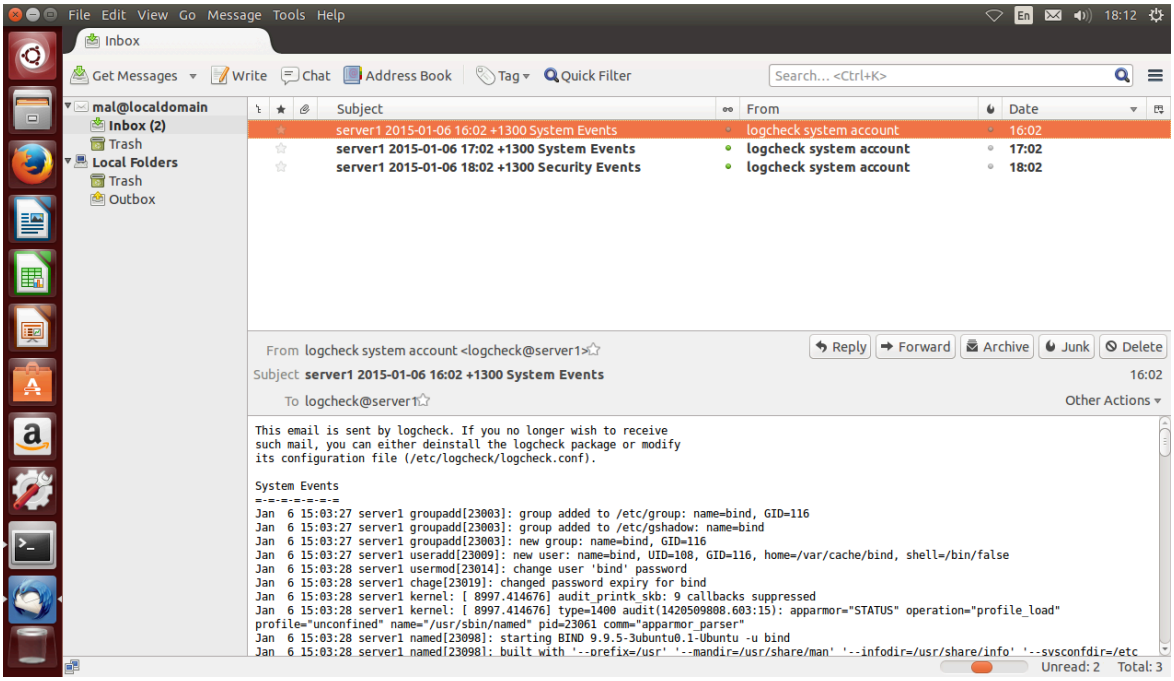
1. Make sure your terminal session above showing that you have successfully spoken POP3.

2. Do some research into IMAP and POP3, and create a short list of the biggest differences between them.

3. Configure a typical GUI e-mail program, such as Thunderbird (you can access it from the menu system), on Client1 to access the POP3 and SMTP services on Server1 (use the DNS aliases, don't say server1.localdomain in the configuration settings). Test that you can send and receive messages. It should end up looking something like Figure 1, "Using Thunderbird to access Email via POP3 and SMTP":

Considering the options you saw when setting up the account, what further inprovements could be made?

Figure 1. Using Thunderbird to access Email via POP3 and SMTP



An account has been setup using Thunderbird to access the server using SMTP and POP3; a test message has been sent to mal@localdomain from the same account.

8. [Optional] A Simple Open Mailing List

Running short on time?

This section is optional, you needn't do it if you are running low on time, but by doing it you will get a much better appreciation of how e-mail is processed and routed.

Most mail servers host a handful of mailing lists as well, such as one for all the staff, or a large organisation might have lists set up for, say, all the system administrators. Naturally, a list can be anything you decide it to be. With Exim, setting up a mailing list is really quite easy, and involves just a few main steps. We shall be setting up a staff@localdomain mailing list.

Although most lists that you can easily subscribe to use a management engine such as Majordomo, or GNU Mailman, we shall not go into the use of such tools, rather we shall have statically configured mailing lists, which means a human will need to update the list of subscribers.

The administrator of a list will often be reachable as `listname`-owner, with request for subscriptions often going to `listname`-request. Both of these should be aliases.

On the mail server, create a new file in `/etc/exim4/conf.d/router/950_local_mailing_lists`. Into the file, put the following text. You should think about why I gave it the number 950 at the beginning of the filename. To find out what some of these lines mean, look them up in the Exim Specification (consult the index).

```
lists:
  driver = redirect
  file = /etc/lists/$local_part
  forbid_pipe
  forbid_file
  errors_to = $local_part-owner@localdomain
  no_more
```

As you might have guessed, if mail comes in addressed to staff@localdomain, it will (assuming no user or alias called staff exists), redirect it to all the e-mail addresses listed in `/etc/lists/staff`. This is true for all the files in `/etc/lists/`.

You will need to add users to the list, so create the `/etc/lists/` directory, and add a file called `staff`, which should be a simple list (one local user or address per line), of the addresses that are part of this mailing list (just `mal` and `bob`).

You should also create aliases for staff-owner and staff-request in `/etc/aliases` if you haven't already. On many Sendmail-like systems, you need to use the **newaliases** command after editing the **aliases** file, although you don't need it for Exim. Restart **exim** and check it loads correctly.

On the client, log in as mal (they don't have to be part of the group however), and send a mail (using your program you set up in the last section) to the mailing list.

Make sure everyone in the list can receive the message to staff@localdomain, and that the aliases staff-owner@localdomain and staff-request@localdomain work as expected.

List-Id Header

A lot of power users tend to be subscribed to a lot of mailing lists. Some could subscribe to at least 20. You can imagine how chaotic one's life were if all this ended up in their inbox. To sort mail, people generally use some mail-sorting functionality in their e-mail client[5]. You can either look for the mailing list address (which may not always be the same) in the To or Cc headers. A lot of mailing lists[6] add headers to assist you, a fairly typical mailing list header is `List-Id` (there are a lot of other `List-*` headers as well).

Let's add a `List-Id` header to our mailing lists so our users can more easily sort their e-mail. If you're using mailing list software to manage your list, this will probably already be done for you. To do this, we need to add a simple entry in the e-mail router we added before. The entry is as follows. Note that another common header is `X-List-Id`. You should remove any headers of the same type, unless there are supposed to be multiple entries (such as `Received` headers). Here is the what the entry should now contain. The bold lines are new.

```
lists:
  driver = redirect
  file = /etc/lists/$local_part
  forbid_pipe
  forbid_file
  errors_to = $local_part-owner@localdomain
```

[5] I don't want to say MUA here, because most clients are more than just a MUA these days, as they also send mail (speak SMTP), provide advanced message filtering and may also deal with groupware functionality.
[6] The GNU Mailman software is well-known for this.

```
  headers_remove = List-Id
  headers_add = List-Id: $local_part@localdomain
  no_more
```

> **Exercise**
> As an exercise restart Exim, and send a message to the list. Make sure you get the header added. You can then view the full headers of the message.

Reply-To Munging

Let's say that you are subscribed to a rather large mailing list. Let's say you post a question (which could be a reply to someone elses message). You are listed in the From header for that message. People could respond in one of two ways: Reply, or Reply All.

In the case of Reply, the message would go only to yourself, not the mailing list. This can annoy a lot of people, because it means that others can participate in the discussion (and locks you into the discussion).

On the other had, if Reply All was used, the reply would go to yourself, and the mailing list, and you would end up with two copies of the message. This also annoys a lot of people.

The `Reply-To` header is a header that an e-mail client application may use to decide who to send a response to when replying (or using reply-all), instead of using whichever address is listed in the `From` header.

Many mailing lists have an option to automatically set the Reply-To on all messages, such that all messages go back to the mailing list. This is a controversial thing to do, with some people saying its a good idea [http://www.metasystema.org/essays/reply-to-useful.mhtml] and others insisting that it's a bad idea [http://www.unicom.com/pw/reply-to-harmful.html]. It has its place in some lists, depending largely on the user population.

We can implement `Reply-To` munging the same way we did the `List-Id` header, with the `header-add` parameter in the message router. This example builds on the `List-Id` example above.

```
lists:
  driver = redirect
  file = /etc/lists/$local_part
  forbid_pipe
  forbid_file
  errors_to = $local_part-owner@localdomain
  headers_remove = List-Id:Reply-To
  headers_add = List-Id: $local_part@localdomain\n\
        Reply-To: $local_part@localdomain
  no_more
```

Restart Exim, and send another test message. Check that `Reply-To` has been set. Reply to the message: you should either be asked if you want to reply to staff@localdomain (ie. honour Reply-To) or not. Read the webpages listed in the footnotes below about the sensibilities of `Reply-To` munging.

9. Last Words

There is so much more we could go into on the subject of e-mail. For instance, there are the rather significant topics of spam and viruses, cryptographic issues, and virtual hosting. All of

this would be good to teach you, but we just don’t have the time in one semester. However, you should know that the practical aspects of these topics is similar to the processes we have already used. Here are some topics and tools the eager student may want to read.

RFC1855: Netiquette Guidelines [http://ftp.rfc-editor.org/in-notes/rfc1855.txt]
How to maintain suitable behaviour on a network.

Procmail
A very powerful mail filtering tool.

Spam Assassin
A very useful spam filter.

The case of the 500-mile email [http://www.ibiblio.org/harris/500milemail.html]
A funny anecdote, worth reading.