Set Up Mail Server Documentation

1.0

Nosy

2014 01 23
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>3.1</td>
<td>13</td>
</tr>
<tr>
<td>3.2</td>
<td>13</td>
</tr>
<tr>
<td>3.3</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>17</td>
</tr>
<tr>
<td>5.1</td>
<td>17</td>
</tr>
<tr>
<td>5.2</td>
<td>17</td>
</tr>
<tr>
<td>5.3</td>
<td>17</td>
</tr>
<tr>
<td>5.4</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>19</td>
</tr>
<tr>
<td>6.1</td>
<td>19</td>
</tr>
<tr>
<td>6.2</td>
<td>28</td>
</tr>
<tr>
<td>6.3</td>
<td>32</td>
</tr>
<tr>
<td>6.4 Webmail</td>
<td>36</td>
</tr>
<tr>
<td>6.5</td>
<td>37</td>
</tr>
<tr>
<td>6.6</td>
<td>38</td>
</tr>
<tr>
<td>7</td>
<td>39</td>
</tr>
<tr>
<td>7.1</td>
<td>39</td>
</tr>
<tr>
<td>7.2 SQL</td>
<td>41</td>
</tr>
<tr>
<td>8</td>
<td>43</td>
</tr>
<tr>
<td>8.1</td>
<td>43</td>
</tr>
<tr>
<td>8.2 strategy</td>
<td>43</td>
</tr>
<tr>
<td>8.3</td>
<td>44</td>
</tr>
<tr>
<td>8.4</td>
<td>45</td>
</tr>
<tr>
<td>8.5</td>
<td>45</td>
</tr>
<tr>
<td>8.6 Telnet</td>
<td>46</td>
</tr>
<tr>
<td>8.7 Can postfix receive?</td>
<td>46</td>
</tr>
</tbody>
</table>
One of the most fragile and fragmented services I’ve had to configure on Ubuntu is a mail server. No matter which of the many guides I follow, each time I do it there’s always something not working.

This one is mostly for my benefit, but hopefully it’ll be useful to others, too. I’ve tried to make the guide easy to follow and as short as possible. Please comment if something isn’t clear.

Before we start, I have to give a huge amount of credit to Ivar Abrahamsen for his guide which is, by far, one of the best ones out there.

So let’s kick off . . .

- Postfix is the mail transfer agent (MTA) responsible for accepting new messages and storing them on your server as well as allowing authorised users to send e-mail.
- Courier sits in front of Postfix and provides an IMAP and POP3 interface for clients to connect to.
- SASL with SSL and TLS allows you to authenticate and communicate with the mail server securely.
- SpamAssassin will analyse your e-mails as they arrive and will filter out what it thinks is spam.
- ClamAV will scan e-mails for viruses before delivering it to your inbox.
- Amavis ties SpamAssassin and ClamAV together, and is itself hooked into Postfix.
- MySQL will be used to manage user accounts and e-mail forwarding.
1.2

```
sudo su -
apt-get update
apt-get install -y mysql-server postfix postfix-mysql libsasl2-modules libsasl2-modules-sql libgsasl7
```

During the installation of MySQL you will be prompted for the root user password, as shown:

```
Configuring mysql-server-5.1

While not mandatory, it is highly recommended that you set a password for the MySQL administrative "root" user.

If this field is left blank, the password will not be changed.

New password for the MySQL "root" user:

[password]

<Ok>
```

Enter a secure password, and don’t forget it!

Similarly, during the installation of Courier you will be presented with the following configuration prompts:

Choose No

Choose OK

Choose Internet Site

Enter your mail server name (e.g. replace example.com with your own domain). Make sure you have this subdomain configured in your DNS records.

Choose OK

I won’t walk you through the parameters we’re using when configuring Postfix as I want to keep this guide light. If you’re interested, you can find more information from the man pages.
mv /etc/postfix/main.cf{,.default}
vi /etc/postfix/main.cf

Copy/paste the following (change all instances of mail.example.com):

```
myorigin = /etc/mailname
```

```
smtpd_banner = $myhostname ESMTP $mail_name biff = no append_dot_mydomain = no mydestination = relayhost = mynetworks = 127.0.0.0/8 [:ffff:127.0.0.0]/104 [:1]/128 mynetworks_style = host mailbox_size_limit = 0 virtual_mailbox_limit = 0 recipient_delimiter = + inet_interfaces = all message_size_limit = 0
```

# SMTP Authentication (SASL)

```
smtp_sasl_auth_enable = yes broken_sasl_auth_clients = yes smtpd_sasl_security_options = noanonymous smtpd_sasl_local_domain =
```

# Encrypted transfer (SSL/TLS)

```
smtp_use_tls = yes smtpd_use_tls = yes smtpd_tls_cert_file = /etc/ssl/private/mail.example.com.crt smtpd_tls_key_file = /etc/ssl/private/mail.example.com.key smtpd_tls_session_cache_database = btree:${data_directory}/smtpd_scache smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
```

# Basic SPAM prevention

```
smtpd_helo_required = yes smtpd_delayReject = yes disable_vrfy_command = yes smtpd_sender_restrictions = permit_sasl_authenticated, permit_mynetworks, reject smtpd_recipient_restrictions = permit_sasl_authenticated, permit_mynetworks, reject
```

# Force incoming mail to go through Amavis

```
content_filter = amavis:[127.0.0.1]:10024 receive_override_options = no_address_mappings
```

# Virtual user mappings

```
alias_maps = hash:/etc/aliases alias_database = hash:/etc/aliases virtual_mailbox_base = /var/spool/mail/virtual virtual_mailbox_maps = mysql:/etc/postfix/maps/user.cf virtual_uid_maps = static:5000 virtual_gid_maps = static:5000 virtual_alias_maps = mysql:/etc/postfix/maps/alias.cf virtual_mailbox_domains = mysql:/etc/postfix/maps/domain.cf
```

mv /etc/postfix/master.cf{,.default}
vi /etc/postfix/master.cf

Copy/paste the following (no changes required):

```
1.2.1 #
```

```
# Postfix master process configuration file. For details on the format # of the file, see the master(5) manual page (command: “man 5 master”). # # Do not forget to execute “postfix reload” after editing this file. # #=================================================================smtp service type private unpriv chroot wakeup maxproc command + args # (yes) (yes) (yes) (never) (100) #=================================================================
```

```
smtpd smtps inet n - - - smtpd
-o smtpd_tls_wrappermode=yes
submission inet n - - - smtpd popupf ifo n - - 60 1 pickup
-o content_filter= -o receive_override_options=no_header_body_checks
```

1.2.
cleanup unix n - - - 0 cleanup qmgr fifo n - n 300 1 qmgr tlsmgr unix - - - 1000? 1 tlsmgr rewrite unix - - - - - trivial-rewrite bounce unix - - - - 0 bounce defer unix - - - - 0 bounce trace unix - - - - 0 bounce verify unix - - - - 1 verify flush unix n - 1000? 0 flush proxymap unix - n - - proxymap proxymap write unix - n - 1 proxymap smtp unix - - - smtp 
-o smtp fallback relay=

showq unix n - - - showq error unix - - - - - error retry unix - - - - - error discard unix - - - - - discard local unix - n n - - virtual smtp unix - - - - - smtp anvil unix - - - - - 1 anvil scache unix - - - - 1 scache

# Interfaces to non-Postfix software. Be sure to examine the manual # pages of the non-Postfix software to find out what options it wants. # # Many of the following services use the Postfix pipe(8) delivery # agent. See the pipe(8) man page for information about ${recipient} # and other message envelope options. # 
==================================================================== # # maildrop. See the Postfix MAILDROP_README file for details. # Also specify in main.cf: maildrop_destination_recipient_limit=1 # maildrop unix - n n - - pipe
flags=D Rhu user=vmail argv=/usr/bin/maildrop -d ${recipient}

# # See the Postfix UUCP_README file for configuration details. # uucp unix - n n - - pipe
flags=F qhu user=uucp argv=uux -r -n -z -a$sender - $nexthop!rmail ($recipient)

# # Other external delivery methods. # ifmail unix - n n - - pipe
flags=F user=ftn argv=/usr/lib/ifmail/ifmail -r $nexthop ($recipient)

bsmtp unix - n n - pipe flags=F q. user=bsmtp argv=/usr/lib/bsmtp/bsmtp -t$nexthop -f$sender $recipient

scalemail-backend unix - n n - 2 pipe flags=R user=scalemail argv=/usr/lib/scalemail/bin/scalemail-store ${nex-
thop} ${user} ${extension}

mailman unix - n n - - pipe flags=FR user=list argv=/usr/lib/mailman/bin/postfix-to-mailman.py ${nexthop} ${user}

amavis unix - - - - 2 smtp -o smtp_data_done_timeout=1200 -o smtp_send_xforward_command=yes -o dis-
able_dns_lookups=yes -o max_use=20

127.0.0.1:10025 inet n - - - smtpd -o content_filter= -o localrecipient_maps= -o relay_recipient_maps= -o smtpd_restriction_classes= -o smtpd_delay_reject=no -o smtpd_client_restrictions=permit_mynetworks, reject -o smtpd_helo_restrictions= -o smtpd_sender_restrictions= -o smtpd_recipient_restrictions=permit_mynetworks, reject -o smtpd_data_restrictions=reject_unauth_pipelining -o smtpd_end_of_data_restrictions= -o mynetworks=127.0.0.8 -o smtpd_error_sleep_time=0 -o smtpd_soft_error_limit=1001 -o smtpd_hard_error_limit=1000 -o smtpd_client_connection_limit=0 -o smtpd_client_connection_rate_limit=0 -o receive_override_options=no_header_body_checks,no_unknown_recipient_checks

As all our mail users are going to be virtual (i.e. we’re not going to create physical user accounts for each user), we only need to create one mail directory and one user account.

groupadd virtual -g 5000 useradd -r -g “virtual” -c “Virtual User” -u 5000 virtual mkdir /var/spool/mail/virtual chown virtual:virtual /var/spool/mail/virtual

Now we’ll create the database which will store the mail user configuration and forwarding rules.

mysql -uroot -p

Enter the password you created during the MySQL installation.

Copy/paste the following (change mailuserpassword, example.com and change the admin’s password to something more secure):
CREATE DATABASE mail;

GRANT ALL ON mail.* TO mail@localhost IDENTIFIED BY 'mailuserpassword';

FLUSH PRIVILEGES;
USE mail;

CREATE TABLE IF NOT EXISTS alias ( source varchar(255) NOT NULL, destination varchar(255) NOT NULL default '', enabled tinyint(1) unsigned NOT NULL default '1', PRIMARY KEY (source) ) ENGINE=MyISAM DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS domain ( domain varchar(255) NOT NULL default '', transport varchar(255) NOT NULL default 'virtual:', enabled tinyint(1) unsigned NOT NULL default '1', PRIMARY KEY (domain) ) ENGINE=MyISAM DEFAULT CHARSET=utf8;

CREATE TABLE IF NOT EXISTS user ( email varchar(255) NOT NULL default '', password varchar(255) NOT NULL default '', name varchar(255) default '', quota varchar(255) default NULL, enabled tinyint(1) unsigned NOT NULL default '1', PRIMARY KEY (email) ) ENGINE=MyISAM DEFAULT CHARSET=utf8;

INSERT INTO alias (source, destination, enabled) VALUES ('@localhost', 'admin@example.com', 1);
INSERT INTO alias (source, destination, enabled) VALUES ('@localhost.localdomain', '@localhost', 1);
INSERT INTO domain (domain, transport, enabled) VALUES ('localhost', 'virtual:', 1);
INSERT INTO domain (domain, transport, enabled) VALUES ('localhost.localdomain', 'virtual:', 1);
INSERT INTO domain (domain, transport, enabled) VALUES ('example.com', 'virtual:', 1);
INSERT INTO user (email, password, name, quota, enabled) VALUES ('admin@example.com', ENCRYPT('changeme'), 'Administrator', NULL, 1);

Note that we're encrypting the password. Some guides will recommend storing the password in clear text so that you can configure Postfix to support CRAM-* (e.g. CRAM-MD5) authentication methods. I think it's much more secure to store these passwords encrypted and use SSL/TLS to encrypt your authentication requests. For that reason, we don't need to store clear text passwords and we don't need to provide CRAM-* support.

Now that the database is in place we can create the map files to tell Postfix how to communicate with it.

mkdir /etc/postfix/maps
vi /etc/postfix/maps/alias.cf

Copy/paste the following (change mailuserpassword):

user=mail
password=mailuserpassword dbname=mail table=alias select_field=destination where_field=source hosts=127.0.0.1 additional_conditions=and enabled = 1

vi /etc/postfix/maps/domain.cf

Copy/paste the following (change mailuserpassword):

user = mail
password = mailuserpassword dbname = mail table = domain select_field = domain where_field = domain hosts = 127.0.0.1 additional_conditions = and enabled = 1

vi /etc/postfix/maps/user.cf

Copy/paste the following (change mailuserpassword):

user = mail
password = mailuserpassword
dbname = mail
table = user
select_field = CONCAT(SUBSTRING_INDEX(email,'@',-1),'/',SUBSTRING_INDEX(email,'@',1),'/')
where_field = email
hosts = 127.0.0.1
additional_conditions = and enabled = 1

Set restrictive read permissions as these files contain the MySQL mail user’s password.

chmod 700 /etc/postfix/maps/*
chown postfix:postfix /etc/postfix/maps/*

The final part of configuring Postfix is to configure the authentication mechanism. SASL is a authentication layer that provides the ability to receive a user’s credentials in a variety of formats.

mkdir -p /var/spool/postfix/var/run/saslauthd
mkdir /etc/postfix/sasl
adduser postfix sasl
vi /etc/postfix/sasl/smtpd.conf

Copy/paste the following (change mailuserpassword):

pwcheck_method: saslauthd auxprop_plugin: sql
mech_list: plain
login sql_engine: mysql
sql_hostnames: 127.0.0.1
sql_user: mail
sql_passwd: mailuserpassword
sql_database: mail
sql_select: SELECT password FROM user WHERE email='%u@%r' AND enabled = 1

chmod -R 700 /etc/postfix/sasl/smtpd.conf
mv /etc/default/saslauthd{,.default}

Copy/paste the following (no changes required):

START=yes
DESC="SASL Authentication Daemon” NAME="saslauthd” MECHANISMS="pam” MECH_OPTIONS="”
THREADS=5 OPTIONS="" -r -c -m /var/spool/postfix/var/run/saslauthd"

vi /etc/pam.d/smtp

Copy/paste the following (change all instances of mailuserpassword):

auth required pam_mysql.so user=mail passwd=mailuserpassword host=127.0.0.1
db=mail table=user usercolumn=email passwdcolumn=password crypt=1
account sufficient pam_mysql.so user=mail passwd=mailuserpassword host=127.0.0.1
db=mail table=user usercolumn=email passwdcolumn=password crypt=1

chmod 700 /etc/pam.d/smtp

Now let’s configure Courier.

I like to provide both IMAP and POP3 support, although personally I only use IMAP. In addition, we’ll be provide SSL support for securing authentication requests.

mv /etc/courier/authdaemonrc{,.default}
vi /etc/courier/authdaemonrc

Copy/paste the following (no changes required):

authmodulelist="authmysql"
authmodulelistorig="authuserdb authpam authpgsql authldap authmysql authcustom authpipe”
daemons=5 authdaemonvar=/var/run/courier/authdaemon DEBUG_LOGIN=0 DEFAULTOPTIONS="" LOGGEROPTS=""

mv /etc/courier/authmysqlrc{,.default}
vi /etc/courier/authmysqlrc

Copy/paste the following (change mailuserpassword):

authmodulelist="authmysql"
Set Up Mail Server Documentation, 1.0

MySQL_SERVER localhost

MySQL_USERNAME mail MySQL_PASSWORD mailuserpassword MySQL_PORT 0 MySQL_DATABASE mail MySQL_USER_TABLE user MySQL_CRYPT_PWFIELD password MySQL_UID_FIELD 5000 MySQL_GID_FIELD 5000 MySQL_LOGIN_FIELD email MySQL_HOME_FIELD "/var/spool/mail/virtual" MySQL_MAILDIR_FIELD CONCAT(SUBSTRING_INDEX(email,'@',-1),"/",SUBSTRING_INDEX(email,'@',1),")" MySQL_NAME_FIELD name MySQL_QUOTA_FIELD quota

mv /etc/courier/imapd{,.default}
vi /etc/courier/imapd

Copy/paste the following (no changes required):

ADDRESS=0

PORT=143 MAXDAEMONS=40 MAXPERIP=20 PIDFILE=/var/run/courier/imapd.pid TCPDOPTS="-nodnslookup -noidentlookup" LOGGEROPTS="-name=imapd" IMAP_CAPABILITY=""IMAP4rev1 UIDPLUS CHILDREN NAMESPACE THREAD=ORDEREDSUBJECT THREAD=REFERENCES SORT QUOTA IDLE" IMAP_KEYWORDS=1 IMAP_ACL=1 IMAP_CAPABILITY_ORIG=""IMAP4rev1 UIDPLUS CHILDREN NAMESPACE THREAD=ORDEREDSUBJECT THREAD=REFERENCES SORT QUOTA AUTH=CRAM-MD5 AUTH=CRAM-SHA1 AUTH=CRAM-SHA256 IDLE" IMAP_PROXY=0 IMAP_PROXY_FOREIGN=0 IMAP_IDLE_TIMEOUT=60 IMAP_MAILBOX_SANITY_CHECK=1 IMAP_CAPABILITY_TLS="SIMAP_CAPABILITY AUTH=PLAIN" IMAP_CAPABILITY_TLS_ORIG="SIMAP_CAPABILITY ORIG AUTH=PLAIN" IMAP_DISABLETHREADSORT=0 IMAP_CHECK_ALL_FOLDERS=0 IMAP_OBSOLETE_CLIENT=0 IMAP_UMASK=022 IMAP_ULIMITD=65536 IMAP_USELOCKS=1 IMAP_SHAREDINDEXFILE=/etc/courier/shared/index IMAP_ENHANCEDIDLE=0 IMAP_TRASHFOLDERNAME=Trash IMAP_EMPTYTRASH=Trash:7 IMAP_MOVE_EXPUNGE_TO_TRASH=0 SENDMAIL=/usr/sbin/sendmail HEADERFROM=X-IMAP-Sender IMAPDSTART=YES

mv /etc/courier/imapd-ssl{,.default}
vi /etc/courier/imapd-ssl

Copy/paste the following (change mail.example.com):

SSLPORT=993

SSLADDRESS=0 SSLPIDFILE=/var/run/courier/imapd-ssl.pid SSLLLOGGEROPTS="-name=imapd-ssl" IMAPDSSLSTART=YES IMAPDSTARTTLS=YES IMAP_TLS_REQUIRED=0 COURIERTLS=/usr/bin/couriertls TLS_KX_LIST=ALL TLS_COMPRESSION=ALL TLS_CERTS=X509 TLS_CERTFILE=/etc/ssl/private/mail.example.com.pem TLS_TRUSTCERTS=/etc/ssl/certs TLS_VERIFYPEER=NONE TLS_CACHEFILE=/var/lib/courier/couriersslcache TLS_CACHESIZE=524288 MAILDIRPATH=Maildir

mv /etc/courier/pop3d{,.default}
vi /etc/courier/pop3d

Copy/paste the following (no changes required):

PIDFILE=/var/run/courier/pop3d.pid

MAXDAEMONS=40 MAXPERIP=4 POPAUTH="LOGIN" POPAUTH_ORIG="PLAIN LOGIN CRAM-MD5 CRAM-SHA1 CRAM-SHA256" POPAUTH_TLS="LOGIN PLAIN" POPAUTH_TLS_ORIG="LOGIN PLAIN" POP3_PROXY=0 PORT=110 ADDRESS=0 TCPDOPTS="-nodnslookup -noidentlookup" LOGGEROPTS="-name=pop3d" POP3START=YES MAILDIRPATH=Maildir

mv /etc/courier/pop3d-ssl{,.default}
vi /etc/courier/pop3d-ssl

Copy/paste the following (change mail.example.com):

SSLPORT=995 SSLADDRESS=0 SSLPIDFILE=/var/run/courier/pop3d-ssl.pid SSLLLOGGEROPTS="-name=pop3d-ssl" POP3SSLSTART=YES POP3_STARTTLS=YES POP3_TLS_REQUIRED=0

1.2.
Set Up Mail Server Documentation, 1.0

COURIERTLS=/usr/bin/couriertls TLS_STARTTLS_PROTOCOL=TLS1 TLS_KX_LIST=ALL
TLS_COMPRESSION=ALL TLS_CERTS=X509 TLS_CERTFILE=/etc/ssl/private/mail.example.com.pem
TLS_TRUSTCERTS=/etc/ssl/certs TLS_VERIFYPEER=None TLS_CACHEFILE=/var/lib/courier/couriersslcache
MAILDIRPATH=Maildir

We need to create SSL certificates for Courier to use when authenticating using SSL/TLS. You can either purchase these (to prevent “invalid” certificate warnings) or generate a self-signed certificate which is just as secure, and free.

Run the following (change mail.example.com):

# Remove default certificates
rm -f /etc/courier/imapd.cnf
rm -f /etc/courier/imapd.pem
rm -f /etc/courier/pop3d.cnf
rm -f /etc/courier/pop3d.pem

# Generate a new PEM certificate (valid for 10 years)
openssl req -x509 -newkey rsa:1024 -keyout "/etc/ssl/private/mail.example.com.pem" -out "/etc/ssl/private/mail.example.com.pem" -nodes -days 3650

# Generate a new CRT certificate (valid for 10 years)
openssl req -new -outform PEM -out "/etc/ssl/private/mail.example.com.crt" -newkey rsa:2048 -nodes -keyout "/etc/ssl/private/mail.example.com.key" -keyform PEM -days 3650 -x509

chmod 640 /etc/ssl/private/mail.example.com.*
chgrp ssl-cert /etc/ssl/private/mail.example.com.* You will be prompted to input some information about the certificates you create. You can enter any information you want here except Common Name (CN) which must be your mailname (e.g. mail.example.com).

Next we’ll configure Amavis, the software that ties together SpamAssassin and ClamAV with Postfix.

adduser clamav amavis
cat /dev/null > /etc/amavis/conf.d/15-content-filter-mode
vi /etc/amavis/conf.d/15-content-filter-mode
Copy/paste the following (no changes required):

use strict;

@bypass_virus_checks_maps = (
  %bypass_virus_checks, @bypass_virus_checks_acl, $bypass_virus_checks_re);
@bypass_spam_checks_maps = (
  %bypass_spam_checks, @bypass_spam_checks_acl, $bypass_spam Checks_re);

1;
cat /dev/null > /etc/amavis/conf.d/50-user

vi /etc/amavis/conf.d/50-user
Copy/paste the following (no changes required):

use strict;

@local_domains_acl = qw(.);
$log_level = 1;
$syslog_priority = 'info';
$sa_kill_level_deft = 6.5;
$final_spam_destiny = D_DISCARD;
$pax = 'pax';

1;
mv /etc/default/spamassassin{,.default}

vi /etc/default/spamassassin
Copy/paste the following (no changes required):

ENABLED=1 OPTIONS="--create-prefs --max-children 5 --helper-home-dir" PIDFILE="/var/run/spamd.pid"
CRON=0 dpkg-reconfigure clamav-freshclam
Choose OK
Choose daemon
Choose a mirror closest to you.
Enter your proxy, if required. Usually you will leave this blank.
By default, ClamAV updates every hour. That’s excessive. Bring that down to once a day.
Choose No
Now restart everything.

/etc/init.d/saslauthd restart /etc/init.d/postfix restart /etc/init.d/courier-authdaemon restart /etc/init.d/courier-imap restart /etc/init.d/courier-imap-ssl restart That’s it, you’re done!

You can test your setup by configuring your mail client to connect to your new mail server using admin@example.com as your username and the password you chose (“changeme” in the guide).

Errors will usually show up in /var/log/mail.log or post any problems you’re having in a comment and I’ll try my best to help.

For more information regarding the mail database, testing using Telnet, and more information regarding how all these services are stitched together, please see Flurdy’s guide.
<table>
<thead>
<tr>
<th>Edition</th>
<th>State</th>
<th>Started</th>
<th>Updated</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2nd</td>
<td>Released</td>
<td>2004-02</td>
<td>2004-07</td>
<td>Based on Mandrake 10.x, but valid for all distributions. Very thorough. Includes package description, where to get the sources and binaries, how to build them or which RPMs to use, includes many references, etc etc. Starts off with a basic working server, then advances, extends and tightens it in stages.</td>
</tr>
<tr>
<td>4th</td>
<td>Released</td>
<td>2005-10</td>
<td>2005-12</td>
<td>Based on Breezy Badger, Ubuntu 5.10. Includes Postgrey</td>
</tr>
<tr>
<td>5th</td>
<td>Released</td>
<td>2006-05</td>
<td>2006-11</td>
<td>Based on Dapper Drake, Ubuntu 6.06 LTS.</td>
</tr>
<tr>
<td>6th</td>
<td>Scrapped</td>
<td>2006-11</td>
<td>2007-10</td>
<td>Was to be based on Edgy Eft, Ubuntu 6.10 or 7.04. Includes Domain Key signing. Include my mail admin or my catchall aliases admin.</td>
</tr>
<tr>
<td>7th</td>
<td>Released</td>
<td>2008-04</td>
<td>2009-06</td>
<td>Updated, based on Ubuntu 8.04 LTS Hardy Heron. Using Amazon EC2 as example. (Tested with 8.10 &amp; 9.04 as well)</td>
</tr>
<tr>
<td>8th</td>
<td>Released</td>
<td>2009-05</td>
<td>2009-11</td>
<td>Based on Ubuntu 8.10 (intrepid), then tested with 9.04 (jaunty) &amp; 9.10 (karmic) as well. Using official Ubuntu ec2 as examples.</td>
</tr>
<tr>
<td>9th</td>
<td>Released</td>
<td>2009-11</td>
<td>2010-05</td>
<td>Based on Ubuntu 9.10 (karmic) using Canonical’s cloud images. Added Roundcube webmail option.</td>
</tr>
<tr>
<td>10th</td>
<td>Released</td>
<td>2009-12</td>
<td>2013-01</td>
<td>Based on Ubuntu 10.04 LTS (lucid) using Canonical’s cloud images. Tested on 10.10 (maverick). Tested on 11.04 (natty)</td>
</tr>
<tr>
<td>11th</td>
<td>Draft</td>
<td>2012-11</td>
<td>2013-01</td>
<td>Based on Ubuntu 12.04 (precise). Tested with 12.10 (quantal)</td>
</tr>
</tbody>
</table>

Further details available in the change log and below in the introduction.
3.1

This is a step by step howto guide to set up a mail server on a GNU / Linux system. It is easy to follow, but you end up with a powerful secure mail server. The server accepts unlimited domains and users, and all mail can be read via your favourite clients, or via web mail.

It is secure, traffic can encrypted and it will block virtually all spam and viruses.

3.2

Don't take my word for it! Research others opinions and methods. Look at my references, look at Postfix.org's howtos, read the excellent books available (E.g. Kyle's or Hildebrandt's), search the web or read the proper documentation.

If you refer to this howto in your own document, or find useful links, then let me know.

3.3

If you found this howto very useful, spread the word and help others? If this howto was exceptionally useful why not donate me some beer money? Or buy a postfix book using my amazon affiliate links further down? Or buy a t-shirt from my t-shirt shop? Otherwise send me a Thank You note?
• : Ubuntu Linux
  www.ubuntu.com
  Ah the age old distro argument... Thankfully this set up should work on most distros. I used to base this howto on Mandrake(now Mandriva), and I started this new edition on a Gentoo box. But I don’t have the patience for Gentoo, nor the money to stay with Mandriva Power editions. Why Ubuntu? Its free, simple and slick. As Ubuntu is derived from debian the installations used here will be apt-get based. Please refer to my other editions for details on RPM or source based installations.

• MTAMail Transfer Agent : Postfix
  www.postfix.org
  Simple, free and slick. Yup I am a sucker for anything that works easily. Postfix is powerfull, well established, but not too bloated, and is security concious from the start.

• Pop(Post Office Protocol )/IMAP(Internet Mail Access Protocol ): Courier IMAP
  www.courier-mta.org/imap/
  My first mail server installment was with Courier. I have not found a reason to change this as again it is simple, and free.

• : MySQL
  www.mysql.com
  Although I use Firebird for my application development, (or Hibernate/C-JDBC hybrids), MySQL is well supported for the sort of lookups required in a mail server.

• : Amavisd-new
  www.ijs.si/software/amavisd/
  Easy plug in solution for spam, virus checking etc.

• Anti-Spam(): SpamAssassin
  spamassassin.apache.org
  Powerfull renowned spam fighting tool.

• Anti-Virus(): ClamAV
  www.clamav.net
  Free virus scanner that can be trusted and includes update daemon.

• : Cyrus SASL
www.imc.org/ietf-sasl/

Secure and trusted cryptography technology for authentication of SMTP traffic.

- PostGrey(Postfix)
  isg.ee.ethz.ch/tools/postgrey/

  Postgrey is an excellent little script to stop 99% of all spam. All it does is on first contact for specific from-to combinations, tells the sender server to try again in a little while, which most spammers can't afford to do. When proper servers try again after a few minutes it lets it through.

- : TLS
  www.ietf.org/html.charters/tls-charter.html

  Secure and trusted cryptography technology for encryption of SMTP traffic. Not to be confused with client encryption technology like GnuPG and S/MIME. They are covered in the extend section. Formerly referenced as SSL.

- WebMail: SquirrelMail or Roundcube
  www.squirrelmail.org
  www.roundcube.net

  Easy to set up php based web mail client. Extensive plugin selection.

  Ajaxified prettier web mail client. Not quite as solid as SquirrelMail.

- : Amazon ec2
  aws.amazon.com/ec2

  This guide can be installed locally, co-located or in the cloud.

My preference is ec2, and I provide ec2 based examples, however it makes no difference where you install your mail server.

Please see software links appendix for further information about these software packages. In that section there is more links to documentation or forums, and viable alternatives, downloadable packages, versions details etc.

Further software and tweaks are discussed in the extension section. Also review other peoples opinion on these packages via my references.
5.1

This section is different for every distribution and for every version.

This howto is based on Ubuntu and its base of debian which uses apt-get. Therefore this section uses apt packages to its fullest.

For other installation method please refer to previous edition’s software links and your own distribution for the documentation for other ways of installing. My 2nd edition(outdated) has instructions for Mandriva, general RPM and tarball compiling.

To follow the rest of this howto with another distribution, you need to ensure all your packages have been installed with the same modules, E.g MySQL lookup on postfix and sasl, php in apache etc.

I have set up mail servers using the 32bit and 64bit x86 platforms, and if all the packages are available then other, E.g. Mac platforms should work too.

5.2

With installing Ubuntu you have a choice of which base system to install. You may choose server or desktop image or very basic setups. I will assume a server install, but it should not differ.

Or if you have chosen an ec2 based server, you can:

• Base it on a Canonical image
• Apply my ec2 Ubuntu server suggestions
• Use my prebuilt ec2 public images for this postfix mail server

Please note that after a while Ill stop specifying the use of sudo, as it is up to yourselves if you use it or use a privileged user, e.g. root. My advice is to use ‘sudo’.

5.3

For assistance with repositories, refer to this article on ubuntu’s wiki.

A previous edition discussed repository configuration in more detail.
But basically for this you need main and universe, and I also throw in the other “safe” ones: restricted and multiverse (and partner, when available).

```bash
sudo vi /etc/apt/sources.list
```

If main and universe already is listed, this is a quick find and replace to add the others all over: `:%s/main un/main restricted multiverse un/g`

As mentioned in the previous edition you also might want to find a repository closer to your server.

## 5.4

You need to install a whole bunch of packages. We will install them bit by bit. But first check your package sources are correctly pointing to main multiverse restricted universe repositories of your current Ubuntu version.

```bash
sudo vi /etc/apt/sources.list
```

Secondly update your current system:

```bash
sudo apt-get update
sudo apt-get upgrade
```

---

: aptitude is no longer supplied in the base install of Ubuntu. This is due to some concurrency issues. Some part of this document may still refer to aptitude. You should use the original apt-get instead.

---

### 5.4.1

I also install a few other packages that I personally prefer. But nothing todo with the mail server. `sudo aptitude install vim mutt lynx curl git`

### 5.4.2

To find out which packages you may have installed, you can use for example:

```bash
sudo dpkg --list | grep postfix
```

or

```bash
sudo aptitude search postfix
```

---

**EC2 Bundle**

My AMI flurdy-amis/ubuntu-mail-server-clean is based on Canonical’s official Ubuntu with these basic mail server packages installed.
6.1

6.1.1

Shorewall

Not essential for an EC2 image. It is essential for a normal server. UFW is bundled with recent Ubuntu distributions, but I still prefer Shorewall for servers.

```
sudo aptitude install shorewall shorewall-doc
# for earlier ubuntu versions use package shorewall-common shorewall-perl shorewall-doc instead
```

Amazon provides a firewall/ access control for its servers, so not always needed then, but nice to have. And in all others situations; a must have.

Basically at first you want to only allow SSH. Then SMTP and IMAP from your IP only.

When you are confident that the mail server is secure, you can open SMTP to the world. If you prefer you can also open IMAP to the world, unless you have a very small client IP range.

Later you may open web access to the webmail and admin gui. This you may also restrict to specific IPs.

**SSH**

By default Shorewall in Ubuntu has an empty set up. You can find the default values for Shorewall in /usr/share/doc/shorwall/default-config. And examples in /usr/share/doc/shorwall/examples. We will create a basic set up.

First configure which network adapters we are accessing the net.

```
cp /usr/share/doc/shorewall/default-config/interfaces /etc/shorewall/
vi /etc/shorewall/interfaces
net eth0 detect dhcp,tcpflags,logmartians,nosmurfs
```
Then we will configure network zones

cp /usr/share/doc/shorewall/default-config/zones /etc/shorewall/
vi /etc/shorewall/zones

Add the firewall if not there and the internet as a zone.

fw firewall
# loc ipv4
net ipv4

Then if needed to specify hosts you can do it in this file. E.g. If you want to specify what is your home IP etc.

cp /usr/share/doc/shorewall/default-config/hosts /etc/shorewall/
vi /etc/shorewall/hosts
# eth0:192.168.0.0/24

Then set what is the default policy for firewall access.

cp /usr/share/doc/shorewall/default-config/policy /etc/shorewall/
vi /etc/shorewall/policy

$FW net ACCEPT
net $FW DROP info
net all DROP info
# The FOLLOWING POLICY MUST BE LAST
all all REJECT info

For safety in case it goes down.

cp /usr/share/doc/shorewall/default-config/routestopped /etc/shorewall/
vi /etc/shorewall/routestopped
eth0 0.0.0.0 routeback

You may put in a netmask of your IP range if you are more concerned.

Now for the main firewall rules. You can find predetermined macro rules for Shorewall in /usr/share/shorewall.

cp /usr/share/doc/shorewall/default-config/rules /etc/shorewall/
vi /etc/shorewall/rules

SSH/ACCEPT net $FW
Ping/ACCEPT net $FW

# Permit all ICMP traffic FROM the firewall TO the net zone
ACCEPT $FW net icmp

# mail lines
SMTP/ACCEPT net $FW
SMTPS/ACCEPT net $FW
Submission/ACCEPT net $FW
IMAP/ACCEPT net $FW
IMAPS/ACCEPT net $FW

# web
Web/ACCEPT net $FW

Firewall configuring is always risky business, as it is easy to lock yourself out. To test the setup syntax, run
shorewall check
Restart it with
/etc/init.d/shorewall restart

Then to switch it on during boot:
vi /etc/default/shorewall
startup=1

For more details on IP Tables and Shorewall, look up its website.

6.1.2
MySQL

sudo aptitude install mysql-client mysql-server

This will prompt you for a root password. Choose something wise and remember it! For purpose of this tutorial I will set it to rootPASSWORD

Now we will need to create the tables for those lookups just specified. First you need to create a user to use in MySQL for mail only. Then you need to create the database, Take note of your chosen mail username and password. You will need the password you specified for root during MySQL package installation.

# If not already done (in package installation)...
mysqladmin -u root -p password new_password
# log in as root
mysql -u root -p
# then enter password for the root account when prompted

Enter password:

# then we create the mail database
create database maildb;
# then we create a new user: "mail"
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP
ON maildb.* TO 'mail'@'localhost' IDENTIFIED by 'mailPASSWORD';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP
ON maildb.* TO 'mail'@'%' IDENTIFIED by 'mailPASSWORD';
exit;

Obviously replace mailPASSWORD with your chosen password!

Then you will need to create these tables:

- aliases
- domains
- users
We will create more later on for further extensions, but only these are relevant now.

Log in to mysql as the new mail user

```bash
mysql -u mail -p maildb
# enter the newly created password
```

Enter password:

Then run this commands to create the tables:

```sql
CREATE TABLE `aliases` (  
    `pkid` smallint(3) NOT NULL auto_increment,  
    `mail` varchar(120) NOT NULL default '',  
    `destination` varchar(120) NOT NULL default '',  
    `enabled` tinyint(1) NOT NULL default '1',  
    PRIMARY KEY ('pkid'),  
    UNIQUE KEY `mail` ('mail')
) ;

CREATE TABLE `domains` (  
    `pkid` smallint(6) NOT NULL auto_increment,  
    `domain` varchar(120) NOT NULL default '',  
    `transport` varchar(120) NOT NULL default 'virtual:',  
    `enabled` tinyint(1) NOT NULL default '1',  
    PRIMARY KEY ('pkid')
) ;

CREATE TABLE `users` (  
    `id` varchar(128) NOT NULL default '',  
    `name` varchar(128) NOT NULL default '',  
    `uid` smallint(5) unsigned NOT NULL default '5000',  
    `gid` smallint(5) unsigned NOT NULL default '5000',  
    `home` varchar(255) NOT NULL default '/var/spool/mail/virtual',  
    `maildir` varchar(255) NOT NULL default 'blah/',  
    `enabled` tinyint(3) unsigned NOT NULL default '1',  
    `change_password` tinyint(3) unsigned NOT NULL default '1',  
    `clear` varchar(128) NOT NULL default 'ChangeMe',  
    `crypt` varchar(128) NOT NULL default 'sdtrusfX0Jj66',  
    `quota` varchar(255) NOT NULL default '',  
    `procmailrc` varchar(128) NOT NULL default '',  
    `spamsassinrc` varchar(128) NOT NULL default '',  
    PRIMARY KEY ('id'),  
    UNIQUE KEY `id` ('id')
) ;
```

The last few fields in the users table are not required, but useful if you extend later.

```bash
# To visualise the tables created:
describe aliases;
describe domains;
describe users;
# then quit mysql
exit;
```

Next is to edit the MySQL's my.cnf file. In Ubuntu/debian this is created by default. In Mandrake I had to manually create a blank one in /etc. But we need to configure it, so:

```bash
sudo vi /etc/mysql/my.cnf
```

In previous version you needed to comment out this line
#skip-networking

However in todays file the default is to bind the address to localhost, which is fine.

```bash
bind-address = 127.0.0.1
```

It is very useful at the start to log any SQL calls that makes it to MySQL. So enable these lines:

```bash
general_log_file = /var/log/mysql/mysql.log
general_log = 1
```

Then in a few weeks comment it out when everything is working, as it slows mysql down

Restart MySQL to make sure its picking up the new settings.

```
sudo /etc/init.d/mysql restart
```

## 6.1.3 MTA

### Postfix

```
sudo aptitude install postfix postfix-mysql
```

This will prompt you to choose type of email server. Select internet site It will also suggest a server name. Correct this if needed.

You should put the name of your server in this file

```
sudo vi /etc/mailname
```

Could be something like smtp.domain.name, where domain name obviously is replaced with your domain name.

Now will open the main postfix configuration file:

```
sudo vi /etc/postfix/main.cf
```

Debian and Ubuntu already puts in some sensible default values in this file. You may need to comment some of them out if we put the same in as well.

```
# This is already done in /etc/mailname
#myhostname= mail.example.com
```

Next is the origin which is the domain appended to email from this machine, this can be your full servername, or domain name.

```
# myorigin=/etc/mailname
myorigin=example.com
```

Then decide what the greeting text will be. Enough info so it is useful, but not divulge everything to potential hackers.

```bash
smtpd_banner = $myhostname ESMTP $mail_name
```
Next you need to decide whether to send all outgoing mail via another SMTP server, or send them yourself. I send via my ISP’s server, so it has to worry about the queing etc. If you send it yourself then you are not reliant on 3rd party server. But you may risk more exposure and accidentally be blocked by spam blockers. And it is more work for your server. Also many servers block dynamic dns hosts, so you may find your server gets rejected. However choose whichever you are comfortable with.

```bash
# leave blank to do it yourself
relayhost =
# or put it an accessible smtp server
relayhost = smtp.yourisp.com
```

Next is network details. You will accept connection from anywhere, and you only trust this machine

```bash
inet_interfaces = all
mynetworks_style = host
```

Next you can masquerade some outgoing addresses. Say your machine’s name is mail.domain.com. You may not want outgoing mail to come from username@mail.example.com, as you’d prefer username@example.com. You can also state which domain not to masquerade. E.g. if you use a dynamic dns service, then your server address will be a subdomain. You can also specify which users not to masquerade.

```bash
# masquerade_domains = mail.example.com www.example.com !sub.dyndomain.com # masquerade_exceptions = root
```

As we will be using virtual domains, these need to be empty.

```bash
local_recipient_maps =
mydestination =
```

Then will set a few numbers.

```bash
# how long if undelivered before sending warning update to sender
delay_warning_time = 4h
# will it be a permanent error or temporary
unknown_local_recipient_reject_code = 450
# how long to keep message on queue before return as failed.
maximal_queue_lifetime = 7d
# max and min time in seconds between retries if connection failed
minimal_backoff_time = 1000s
maximal_backoff_time = 8000s
# how long to wait when servers connect before receiving rest of data
smtp_helo_timeout = 60s
# how many address can be used in one message.
smtpd_recipient_limit = 16
# how many error before back off.
smtpd_soft_error_limit = 3
smtpd_hard_error_limit = 12
```

Now we can specify some restrictions. Be carefull that each setting is on one line only.

```bash
# Requirements for the HELO statement
smtpd_helo_restrictions = permit_mynetworks, warn_if_reject reject_non_fqdn_hostname, reject_invalid_hostname, permit
# Requirements for the sender details
smtpd_sender_restrictions = permit_mynetworks, warn_if_reject reject_non_fqdn_sender,
```
reject_unknown_sender_domain, reject_unauth_pipelining, permit

# Requirements for the connecting server
smtpd_client_restrictions = reject_rbl_client sbl.spamhaus.org,
    reject_rbl_client blackholes.easynet.nl,
    reject_rbl_client dnsbl.njabl.org
# Requirement for the recipient address
smtpd_recipient_restrictions = reject_unauth_pipelining, permit_mynetworks,
    reject_non_fqdn_recipient, reject_unknown_recipient_domain,
    reject_unauth_destination, permit
smtpd_data_restrictions = reject_unauth_pipelining

Further restrictions:

# require proper helo at connections
smtpd_helo_required = yes
# waste spammers time before rejecting them
smtpd_delay_reject = yes
disable_vrfy_command = yes

Next we need to set some maps and lookups for the virtual domains.

# not sure of the difference of the next two
# but they are needed for local aliasing
alias_maps = hash:/etc/postfix/aliases
alias_database = hash:/etc/postfix/aliases
# this specifies where the virtual mailbox folders will be located
virtual_mailbox_base = /var/spool/mail/virtual
# this is for the mailbox location for each user
virtual_mailbox_maps = mysql:/etc/postfix/mysql_mailbox.cf
# and this is for aliases
virtual_alias_maps = mysql:/etc/postfix/mysql_alias.cf
# and this is for domain lookups
virtual_mailbox_domains = mysql:/etc/postfix/mysql_domains.cf
# this is how to connect to the domains (all virtual, but the option is there)
# not used yet
# transport_maps = mysql:/etc/postfix/mysql_transport.cf

You can (as in my older editions) use a lookup for the uid and gid of the owner of mail files. But I tend to have one
owner(virtual), so instead add this:

virtual_uid_maps = static:5000
virtual_gid_maps = static:5000

You need to set up an alias file. This is only used locally, and not by your own mail domains.

sudo cp /etc/aliases /etc/postfix/aliases
# may want to view the file to check if ok.
# especially that the final alias, eg root goes
# to a real person
sudo postalias /etc/postfix/aliases

Next you need to set up the folder where the virtual mail will be stored. This may have already been done by the
apt-get. And also create the user whom will own the folders.

# to add if there is not a virtual user
sudo mkdir /var/spool/mail/virtual
sudo groupadd --system virtual -g 5000
sudo useradd --system virtual -u 5000 -g 5000
sudo chown -R virtual:virtual /var/spool/mail/virtual
Postfix's MySQL

Next we need to set up the files to access the lookups via the database. We will only set up a few now, and the rest later when/if needed:

Edit(create) how to find the users mailbox location

```
sudo vi /etc/postfix/mysql_mailbox.cf
```

```
user=mail
password=mailPASSWORD
dbname=maildb
table=users
select_field=maildir
where_field=id
hosts=127.0.0.1
additional_conditions = and enabled = 1
```

Create how to find the email alias:

```
sudo vi /etc/postfix/mysql_alias.cf
```

```
user=mail
password=mailPASSWORD
dbname=maildb
table=aliases
select_field=destination
where_field=mail
hosts=127.0.0.1
additional_conditions = and enabled = 1
```

Create how to find the domains:

```
sudo vi /etc/postfix/mysql_domains.cf
```

```
user=mail
password=mailPASSWORD
dbname=maildb
table=domains
select_field=domain
where_field=domain
hosts=127.0.0.1
additional_conditions = and enabled = 1
```

If you specify an ip in hosts, (as opposed to ‘localhost’) then it will communicate over tcp and not the mysql socket. (chroot restriction). Ps. remember to replace the passwords with your chosen mail user password.

6.1.4 Pop/IMAP

Courier IMAP
sudo aptitude install courier-base courier-authdaemon courier-authlib-mysql \
courier-imap courier-imap-ssl courier-ssl

will prompt you about webdirectories. You can say no to this. It will also warn you about the certificate location. Ignore it.

Please refer to previous edition for more explanations. But below is the details of what you need to change.

sudo vi /etc/courier/authdaemonrc

Change to mysql mode.
authmodulelist="authmysql"

Further down enable logging.
DEBUG_LOGIN=2

sudo vi /etc/courier/authmysqlrc

Changed user
MYSQL_USERNAME mail

Changed password to whichever you have chosen
MYSQL_PASSWORD mailPASSWORD

Changed database
MYSQL_DATABASE maildb

Changed users table
MYSQL_USER_TABLE users

Keep commented in crypt pw
MYSQL_CRYPT_PWFIELD crypt

Keep commented out clear pw
# MYSQL_CLEAR_PWFIELD clear

Added maildir
MYSQL_MAILDIR_FIELD concat(home,’/’,maildir)

Added where clause
MYSQL_WHERE_CLAUSE enabled=1

Lastly you can have a look at the imapd file, but no changes is needed.
vi /etc/courier/imapd

6.1.5

You now have a basic mail server!
Before continuing to the advanced and secure mail server you must ensure the basic setup works. This will save you from loads of pain further on.

It is very easy to make typos, miss tiny steps, unclear steps or simple actual errors in this howto.

Insert stub data from data section

Apply advice from test section judiciously

Ensure the mail server can receive email correctly first, then try sending.

Once you are positive the mail has been received, the mail folders have been automatically created, only then you should test if you can actually read the emails before proceeding

Ive created an EC2 bundle for this stage: flurdy-amis/ubuntu-mail-server-simple.

6.2

Now lets extend this setup with more useful content checks, security and user interfaces.

6.2.1 Content Checks (Anti spam & anti virus)

Amavisd-new

Amavisd ties together all the different ways of checking email content for spam and viruses. Install amavisd-new

```
sudo aptitude install amavisd-new
```

The defaults are pretty good and also the ubuntu documentation is pretty clear, and recommended.

Here is a tweaked version of it:

Initially we will not enable spam or virus detection! This is so we can get amavis set up to receive, check and pass on emails before we go on and over-complicate it.

All of amavis' configuration files are in /etc/amavis. They are now spread across several files in conf.d. Debian and Ubuntu defaults are now very sensible and spread into separate files.

```
cd /etc/amavis/conf.d
```

01-debian defaults are fine.

Have a look at

```
less 05-domain_id
```

but dont change anything in it.

Have a look at

```
less 05-node_id
```

but dont change anything in it.

Have a look at

```
less 15-av_scanners
```
but don't change anything in it.

Edit content check file

```bash
sudo vi 15-content_filter_mode
```

Comment out both virus and spam scans. (Default).

```bash
# #@bypass_virus_checks_maps = ( # %bypass_virus_checks, %bypass_virus_checks_acl, %bypass_virus_checks_re);
# @bypass_spam_checks_maps = ( # %bypass_spam_checks, %bypass_spam_checks_acl, %bypass_spam_checks_re);
```

Have a look at

- `less 20-debian_defaults`

and

```
less 21-ubuntu_defaults
```

but don't change anything in them.

25-amavis_helpers defaults are fine.

30-template-localization defaults are fine.

Edit user file

```bash
sudo vi 50-user
```

In the middle insert:

```perl
@local_domains_acl = qw(.);
$log_level = 2;
$syslog_priority = 'debug';
# $sa_tag_level_deflt = 2.0; # add spam info headers if at, or above that level
# $sa_tag2_level_deflt = 6.31; # add 'spam detected' headers at that level
$sa_kill_level_deflt = 8.0; # triggers spam evasive actions
# $sa_dsn_cutoff_level = 10; # spam level beyond which a DSN is not sent
$final_spam_destiny = D_PASS;
# $final_spam_destiny = D_REJECT; # default
# $final_spam_destiny = D_BOUNCE; # debian default
# $final_spam_destiny = D_DISCARD; # ubuntu default, recommended as sender is usually faked
```

We have now setup amavis to scan and pass along incoming email. Next we will setup postfix to talk to amavis.

```bash
vi /etc/postfix/master.cf
```

Append these lines to the end of the file (make sure they are not already present). (Note the -o lines have spaces in front of them.

```
amavis unix - - - - 2 smtp
 -o smtp_data_done_timeout=1200
 -o smtp_send_xforward_command=yes
 -o disable_dns_lookups=yes
 -o max_use=20
127.0.0.1:10025 inet n - - - - smtpd
 -o content_filter=
 -o local_recipient_maps=
 -o relay_recipient_maps=
 -o smtpd_restriction_classes=
 -o smtpd_delay_reject=no
```

6.2.
-o smtpd_client_restrictions=permit_mynetworks, reject
-o smtpd_helo_restrictions=
-o smtpd_sender_restrictions=
-o smtpd_recipient_restrictions=permit_mynetworks, reject
-o smtpd_data_restrictions=reject_unauth_pipelining
-o smtpd_end_of_data_restrictions=
-o mynetworks=127.0.0.0/8
-o smtpd_error_sleep_time=0
-o smtpd_soft_error_limit=1001
-o smtpd_hard_error_limit=1000
-o smtpd_client_connection_count_limit=0
-o smtpd_client_connection_rate_limit=0
-o receive_override_options=no_header_body_checks, no_unknown_recipient_checks

Also add the following two lines immediately below the “pickup” transport service:

- o content_filter=
  - o receive_override_options=no_header_body_checks

and then added to main.cf

sudo vi /etc/postfix/main.cf

content_filter = amavis:127.0.0.1:10024

Enable scanning by ClamAV of amavis’ temporary files.

sudo adduser clamav amavis

This should be it to get amavis working. If emails are picked up by amavis and passed back to postfix then it looks okay. Only when finished testing do you proceed to uncomment the anti virus and anti spam lines in

sudo vi /etc/amavis/conf.d/50-user

@local_domains_acl = qw(.);
$log_level = 1;
$syslog_priority = 'info';
# $sa_tag_level_deflt = 2.0; # add spam info headers if at, or above that level
# $sa_tag2_level_deflt = 6.31; # add ‘spam detected’ headers at that level
$sa_kill_level_deflt = 8.0; # triggers spam evasive actions
# $sa_dsn_cutoff_level = 10; # spam level beyond which a DSN is not sent
# $final_spam_destiny = D_PASS;
# $final_spam_destiny = D_REJECT; # default
# $final_spam_destiny = D_BOUNCE; # debian default
$final_spam_destiny = D_DISCARD; # ubuntu default, recommended as sender is usually faked
6.2.2 Anti-Spam

SpamAssassin

: sudo aptitude install spamassassin spamc The default config of spam assassin is okay. You could refer to previous edition for more configuration options. You do need to tell SpamAssassin to start smapd on boot.

vi /etc/default/spamassassin

ENABLED=1

One configuration option you could tweak is to enable Bayes and auto learning.

vi /etc/spamassassin/local.cf

I read your email

6.2.3 Anti Virus

ClamAV

:

sudo aptitude install clamav-base libclamav6 clamav-daemon clamav-freshclam

(Earlier versions of Ubuntu may use libclamav5) ClamAV does not need setting up. Configuration files are in /etc/clamav, but they are automatically generated, so do not edit.

By default freshclam, the daemon that updates the virus definition database, is run 24 times a day. That seems a little excessive, so I tend to set that to once a day.

sudo dpkg-reconfigure clamav-freshclam

It will also ask if you want it to be daemon (yes) and which server is closest to you.

If needed, the command below will redefine the configuration with a lot of questions. Not needed unless you need to configure.

sudo dpkg-reconfigure clamav-base

6.2.4 Postgrey

:

sudo aptitude install postgrey

postgrey. Postfix.

sudo vi /etc/postfix/main.cf

::

smtpd_recipient_restrictions = reject_unauth_pipelining, permit_mynetworks, permit_sasl_authenticated, reject_non_fqdn_recipient, reject_unknown_recipient_domain, reject_unauth_destination, check_policy_service inet:[:1]:10023, permit
ipv6inet::1:10023 inet:127.0.0.1:10023

/etc/postgrey /etc/default/postgrey. E.g. delay, auto whitelisting, or reject message.

POSTGREY_OPTS="--inet=10023 --max-age=365"

You know have an advanced mail server. You can use this, but I recommend continuing. However this is a good point to test the set up so far and to insert some data in the db. I've created an EC2 bundle for this stage: flurdy-amis/ubuntu-mail-server-spam.

6.3

Stopping hackers, phishers, spammers, your boss and your neighbour from accessing your server or the traffic in between is important, and easily done.

6.3.1

Normal email traffic between clients and servers are in open plain text. That includes passwords and content of emails.

SASL

SASL secures the actual authentication (login), by encoding the passwords so that it can not be easily intercepted. The rest of the emails are however in clear plain text.

SASL can be a royal pain to set up, especially as it does not support storing encrypted passwords by default in Ubuntu. Therefore my previous editions described how to configure SASL using plain text passwords in the database.

Obviously this is not ideal, so there are ways to combine SASL and storing encrypted passwords. In the future the packages that comes with Ubuntu may support the password_format configuration option for SASL. But until then you can configure SASL to ask PAM to compare the passwords:

```
sudo aptitude install libsasl2-modules libsasl2-modules-sql libgsasl7\libauthen-sasl-cyrus-perl sasl2-bin libpam-mysql
```

Enable postfix to access SASL files:

```
sudo adduser postfix sasl
```

Create sasl files accessibly even by chrooted Postfix:

```
sudo mkdir -p /var/spool/postfix/var/run/saslauthd
```

Add SASL configurations to Postfix:

```
sudo vi /etc/postfix/main.cf
```

```
# SASL
smtpd_sasl_auth_enable = yes
# If your potential clients use Outlook Express or other older clients
# this needs to be set to yes
broken_sasl_auth_clients = no
smtpd_sasl_security_options = noanonymous
smtpd_sasl_local_domain =
```
Modify these existing configurations:

```bash
# Add permit_sasl_authenticated to your existing smtpd_sender_restrictions
smtpd_sender_restrictions = permit_sasl_authenticated, permit_mynetworks,
    warn_if_reject reject reject_non_fqdn_sender, reject_unknown_sender_domain,
    reject_unauth_pipelining, permit
# Add permit_sasl_authenticated to your existing smtpd_recipient_restrictions
smtpd_recipient_restrictions = reject_unauth_pipelining, permit_mynetworks,
    permit_sasl_authenticated, reject_non_fqdn_recipient, reject_unknown_recipient_domain,
    reject_unauth_destination, check_policy_service inet:127.0.0.1:10023, permit
```

Change how SASLAUTHD is run:

```bash
sudo vi /etc/default/saslauthd

# Toggle this to yes
START=yes
# Switch this to be under postfix’s spool
# And add -r so that the realm(domain) is part of the username
OPTIONS="-r -c -m /var/spool/postfix/var/run/saslauthd"
```

Tell postfix how to interact with SASL:

```bash
sudo vi /etc/postfix/sasl/smtpd.conf

pwcheck_method: saslauthd
mech_list: plain login cram-md5 digest-md5
log_level: 7
allow_plaintext: true
auxprop_plugin: sql
sql_engine: mysql
sql_hostnames: 127.0.0.1
sql_user: mail
sql_passwd: mailPASSWORD
sql_database: maildb
sql_select: select crypt from users where id='%u@%r' and enabled = 1

(When SASL is working you can remove the log_level line.)
```

Tell the pam how to to authenticate smtp via mysql:

```bash
sudo vi /etc/pam.d/smtp

auth required pam_mysql.so user=mail passwd=aPASSWORD
    host=127.0.0.1 db=maildb table=users usercolumn=id passwdcolumn=crypt crypt=1
account sufficient pam_mysql.so user=mail passwd=aPASSWORD
    host=127.0.0.1 db=maildb table=users usercolumn=id passwdcolumn=crypt crypt=1
```

In addition to tailing var/log/mail.log and /var/log/mysql/mysql.log it is quite useful to tail the auth.log as well when testing SASL.

```bash
tail -f /var/log/auth.log
```

Restart postfix and saslauthd to enable SASL for sending emails.
sudo /etc/init.d/saslauthd restart
sudo /etc/init.d/postfix restart

Imap SASL / Courier

I tend not to have SASL for my courier authentication, as I enforce TLS for all my clients.

However if you have a more lenient access policy which is wise if you have many users, then you may want SASL in Courier as well:

sudo vi /etc/courier/imapd

This may already be available as a commented out line. If not replace the current line by adding UTH=CRAM-MD5 AUTH=CRAM-SHA1 so it resembles something like this: (Again on one line)

IMAP_CAPABILITY="IMAP4rev1 UIDPLUS CHILDREN NAMESPACE THREAD=ORDEREDSUBJECT THREAD=REFERENCES SORT QUOTA AUTH=CRAM-MD5 AUTH=CRAM-SHA1

sudo /etc/init.d/courier-authdaemon restart;
sudo /etc/init.d/courier-imap restart

6.3.2

TLS

Encrypting the traffic stops anyone else listening in on your email communications. And is very recommended. There are different types of communication to encrypt: The data traffic between your email applications and the server when you read emails or when you send emails, and communication between other email servers and your server.

For the encryption of reading emails, it is Courier you need to configure. For sending, and between server encryption it is Postfix.

TLS in Postfix

To encrypt you need certificates. Ubuntu creates some for you for which you can use while setting up the server. However before you go live, it is recommended to create your own with your proper domain name etc. Please refer to previous edition for more detail.

vi /etc/postfix/main.cf

There are already some TLS settings in the default debian/ubuntu version of this file. I moved these to the end, for clarity, but that is up to you.

# TLS parameters
# smtp_use_tls = no
smtp_tls_security_level = may
# smtpd_use_tls=yes
smtpd_tls_security_level = may
# smtpd_TLS_auth_only = no
smtp_tls_note_starttls_offer = yes
smtpd_tls_loglevel = 1
smtpd_tls_received_header = yes
smtpd_TLS_session_cache_timeout = 3600s
tls_random_source = dev:/dev/urandom
smtpd_TLS_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
smtpd_TLS_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
smtpd_TLS_session_cache_database = btrees:$(data_directory)/smtpd_scache
smtpd_TLS_session_cache_database = btrees:$(data_directory)/smtpd_scache
smtpd_TLS_CAfile = /etc/ssl/certs/ca-certificates.crt
Next we have a look at the master.cf file.

```bash
vi /etc/postfix/master.cf
```

By default only the normal smtp service is enabled, which is fine. But I prefer to enable submission (port 587), so that clients can use it, and I can restrict them to TLS only. Also enabled smtsp service (port 465), for some computeability with some older clients (outlook express etc).

```bash
submission inet n - n - - smtpd
 -o smtpd_sasl_auth_enable=yes
 # if you do not want to restrict it encryption only, comment out next line<
 -o smtpd_tls_auth_only=yes
 # -o smtpd_tls_security_level=encrypt
 # -o header_checks=<
 -o smtpd_client_restrictions=permit_sasl_authenticated,reject_unauth_destination,reject
 -o smtpd_sasl_security_options=noanonymous,noplaintext
 -o smtpd_sasl_tls_security_options=noanonymous
 # -o milter_macro_daemon_name=ORIGINATING<
 smtps inet n - - - - smtpd
 -o smtpd_tls_wrappermode=yes
 -o smtpd_sasl_auth_enable=yes
 -o smtpd_tls_auth_only=yes
 -o smtpd_client_restrictions=permit_sasl_authenticated,reject
 -o smtpd_sasl_security_options=noanonymous,noplaintext
 -o smtpd_sasl_tls_security_options=noanonymous
 # -o milter_macro_daemon_name=ORIGINATING
```

**TLS in Courier**

Again Ubuntu has created a certificate for you, but if you want to create your own, especially for a properly named server, then do this.

```bash
cd /etc/courier
openssl req -x509 -newkey rsa:1024 -keyout imapd.pem
        -out imapd.pem -nodes -days 999
```

For more details review an earlier edition.

Then you need to edit

```bash
vi /etc/courier/imapd-ssl
```

By default Ubuntu already points to you certificate

```
TLS_CERTFILE=/etc/courier/imapd.pem
```

Modify this if needed.

Also you if want to restrict IMAP users to SSL/TLS only toggle this setting to 1.

```
IMAP_TLS_REQUIRED=1
```

For maximum compatibility it is not wise to restrict to TLS only for the traffic between servers. As this means not all valid emails sent by others can reach your server. However enabling them the option to encrypt is a good idea.

Be aware that the emails are not encrypted on your machine, nor on the server. For this type of client encryption, please refer to previous edition for more on GnuPG.

In some situations SASL and TLS do not play well together. Those situations are in combinations of storing encrypted passwords, using MD5 authentication over encrypted traffic. I recommend, insisting on TLS traffic with your authenticating clients, which then negates the need for SASL.
You now have an advanced secure mail server. Now is another good point to test the setup so far and to insert some data in the db.

I've created an EC2 bundle for this stage: flurdy-amis/ubuntu-mail-server-secure.

### 6.4 Webmail

#### 6.4.1 Web

You may need to enable web access in the firewall. Check the firewall configuration if necessary.

#### 6.4.2 SquirrelMail

This howto in previous editions used to have SquirrelMail as the webmail client. It is more mature with a longer testing record. It has a large library of various plugins. Please read the SquirrelMail extension further down on how to install it instead if preferred.

#### 6.4.3 Roundcube Webmail

To install Roundcube sudo aptitude install roundcube roundcube-mysql roundcube-plugins It will ask you if you want to configure its database access, answer yes, then select mysql. Then it will ask for the root mysql uses password, which it will create a roundcube mysql user and ask for its desired password. This will create a symlink in /etc/apache2/conf.d/ to /etc/roundcube/apache.conf. Edit this file.

```
sudo vi /etc/roundcube/apache.conf
```

Depending on your setup you may want to move those Alias commands at the top to your virtual hosts configuration, or for this example enable them here for all hosts.

```
# Uncomment them to use it or adapt them to your configuration
Alias /roundcube/program/js/tiny_mce/ /usr/share/tinymce/www/
Alias /roundcube /var/lib/roundcube
```

Next edit the configuration file

```
sudo vi /etc/roundcube/main.inc.php
```

Modify these lines for added security and ease of log in:

```php
<?php
$rcmail_config[‘default_host’] = ‘ssl://localhost:993’;
$rcmail_config[‘smtp_server’] = ‘ssl://localhost’;
$rcmail_config[‘smtp_port’] = 465;
# keep as default or change to your mail server name
$rcmail_config[‘smtp_helo_host’] = ‘mail.example.com’;
$rcmail_config[‘create_default_folders’] = TRUE;
```

There are other tweaks and security features you can enable such as:
Set Up Mail Server Documentation, 1.0

```php
$rcmail_config['sendmail_delay'] = 1;
```

But perhaps concentrate on getting the basics working first... Save, exit and reload Apache to enable these aliases for Roundcube to work

```bash
:sudo /etc/init.d/apache2 reload
```

Then go to your roundcube installation depending where and how you modified those Aliases, e.g. at http://mail.example.com/roundcube.

That should be it

You can obviously modify and tweak further. One thing that may be useful is to have the Roundcube Apache Alias on different virtual hosts, and configure username_domain in main.inc.php to append different email addresses, or configure the default_host to different mail server depending on virtual host... More details on the Roundcube Wiki.

6.5

6.5.1 Web

You may need to enable web access in the firewall. Check the firewall configuration if this necessary.

6.5.2 phpmyadmin

```bash
sudo aptitude install phpmyadmin
```

Enter Yes to set it up, enter root mysql password, enter a phpmyadmin mysql user password twice. Accept apache2 as the web server.

You may choose to restrict phpMyAdmin to a specific virtual host. If so you need to, edit

```bash
sudo vi /etc/apache2/conf.d/phpmyadmin.conf
```

and comment out the alias.

```bash
#Alias /phpmyadmin /usr/share/phpmyadmin
```

And insert the alias instead into a virtual host configuration in /etc/apache2/sites-available/. For this example we are not, and for testing we keep the Alias uncommented.

Reload apache to activate changes. First test if ok.

```bash
sudo apache2ctl -t
```

Then reload it.

```bash
sudo /etc/init.d/apache2 reload
```

You can now go to http://yourdomain.com/phpmyadmin/, and login with the mail user. You can use it as it is, but I recommend securing it a bit more.

One simple way is adding apache’s .htaccess login requirement.

Further restrictions can be restricting to a specific virtual host. Or renaming the folder. Purely obfuscating, but simple.

Or using the example in the webmail section, and adding SSL requirement to the connection. Or disable mysql root’s access via phpMyAdmin.

6.5.
Please refer to previous edition for example on .htaccess, and mysql user restriction.

6.6

Before making any changes you need to have done a few steps externally. (Or at least before you start testing).

6.6.1

You need a domain name to use with your email server. This may be one you purchased, or a subdomain of an existing one, or a dynamic one e.g. dyndns.org.

6.6.2 DNS

You will also need to configure the MX details for the DNS of this server. This is done via your domain registrar, or sometimes an external nameserver(DNS) provider. You can also host your own DNS via packages such as Bind.

Your provider might let you do this through a GUI, but in this is technically how a the configuration should look like:

domain.tld IN MX 10 yourmailserver.domain.tld

(Replace domain.tld with your domain name, and yourmailserver.domain.tld with the full name of your mail server).

Repeat this for each domain that you want the server to handle.

Further mx entries are possible in the same file, if there are subdomains. And also if you have backup MX servers. Refer to my backup MX section if interested.

: Some other mail systems will check via reverse DNS for a match between IP and mail server name, as part of their spam scoring.

If people need suggestions for domain registrars or dns providers then let me know

You know have a finished mail server. This is as far as the main guide goes. Hope it was clear enough to follow.

Now it is time to insert data, and to test how it works.

Feel free to extend it with my suggestions further down.

Ive created an EC2 bundle for this stage: flurdy-amis/ubuntu-mail-server-webmail.
7.1

So we got a fully set up mail server... Well no, there is no users, domains, no nothing!

Okay, first you need add some default data, some which are required, some which make sense. Then we’ll add your own users and domains.

7.1.1

# Use phpMyAdmin or command line mysql

```
INSERT INTO domains (domain) VALUES
    ('localhost'),
    ('localhost.localdomain');
```

Then some default aliases. Some people say these are not needed, but I’d include them.

```
INSERT INTO aliases (mail,destination) VALUES
    ('postmaster@localhost','root@localhost'),
    ('sysadmin@localhost','root@localhost'),
    ('webmaster@localhost','root@localhost'),
    ('abuse@localhost','root@localhost'),
    ('root@localhost','root@localhost'),
    ('@localhost','root@localhost'),
    ('@localhost.localdomain','@localhost');
```

```
INSERT INTO users (id,name,maildir,crypt) VALUES
    ('root@localhost','root','root/',encrypt('apassword') );
```

7.1.2 Domains users

Now lets add some proper data. Say you want this machine to handle data for the fictional domains of “blobber.org”, “whopper.nu” and “lala.com”. Then say this machine’s name is “mail.blobber.org”. All email to lala.com is to be forwarded to whupper.nu.

```
INSERT INTO domains (domain) VALUES
    ('blobber.org'),
    ('mail.blobber.org'),
    ('whopper.nu'),
    ('lala.com');
```
You also have two users called “Xandros” and “Vivita”.

```
INSERT INTO users (id,name,maildir,crypt) VALUES
('xandros@blobber.org','xandros','xandros/','encrypt('apassword') ),
('vivita@blobber.org','vivita','vivita/',' encrypt('anotherpassword') );
```

You want all mail for whooper.nu to go to xandros (catchall).

```
INSERT INTO aliases (mail,destination) VALUES
('@whopper.nu','xandros@blobber.org');
```

There is also a “Karl” user, but he does want all mail forwarded to an external account.

```
INSERT INTO aliases (mail,destination) VALUES
('karl@blobber.org','karl.vovianda@gmail.com');
```

So what does each of these lines actually do? Well the domains are pretty straight forward.

The users are as well, it requires four fields. ID is the email address of the user, and also its username when log- in in, described later on. NAME is optional description of the user. MAILDIR is the name of the folder inside /var/spool/mail/virtual. It must end in a /, otherwise it wont be used as a unix maildir format. CRYPT is the encrypted text password to use.

The aliases are the interesting part. Lets start from a top down view to see how emails get delivered: Say an email arrives addressed to “john@whopper.nu”.

Postfix looks up domains and say whopper.nu is an domain it listens to.

First lookup does not find this user, but the next finds the catchall “@whopper.nu”. None does so it next searches for “@whopper.nu”, which is the way to specify catch all others for that domain.

It finds one row and its destination is “xandros@blobber.org”.

It then searches for “xandros@blobber.org” and finds one, which destination is the same as the mail, therefore it is the final destination.

It then tries to deliver this mail. The look up says blobber.org is a local mail so it looks up users for a matching id and delivers it to its maildir.

Lets try “julian.whippit@lala.com”.

Postfix looks up domains and it is an domain it listens to.

First lookup does not find this user, but the next finds the catchall “@lala.com”. But its destination is another catchall, “@blobber.org”.

This means Postfix will look for “julian.whippit@blobber.org”. This address is not found either, nor is a catchall for blobber.org. Therefore this address is not valid and the message will be bounced.
Any mail arriving for “karl@blobber.org” or “karl@lala.com”, gets forward to an external address of “karl.vovianda@gmail.com”. So forwarding is simple. I tend to use a subdomain for all my friends addresses as easily I forget what their real addresses are, and I use different email clients all the time.

I also added the required aliases of postmaster and abuse to blobber.org and whopper.nu. The catchall for lala.com means they are not required for that domain.

Another useful alias to add is root, as often you get admin mail from e.g cron jobs within those domains etc. Other often used aliases are info, sysadmin, support, sales, webmaster, mail, contact and all. But they are also honeypots for spam, so just include the ones you think you will need.

### 7.1.3

So to add a new domain to the system, You do this, replacing the italics with relevant data:

```sql
INSERT INTO domains (domain) VALUES ('domain.tld');
INSERT INTO aliases (mail,destination) VALUES
   ('@domain.tld','email@address'),
   ('postmaster@domain.tld','email@address'),
   ('abuse@domain.tld','email@address');
```

And to add a new user to the system, do this:

```sql
INSERT INTO users (id,name,maildir,crypt) VALUES
   ('email@address','short description','foldername/',encrypt('password') );
INSERT INTO aliases (mail,destination) VALUES
   ('email@address','email@address');
```

### 7.2 SQL

A selection of useful sql statements, if you are not using an admin/manager program to maintain your email domains and users.

**Find domains without a catchall**

```sql
SELECT al.*
FROM aliases al
LEFT JOIN domains dom
   ON dom.domain = SUBSTRING(al.mail,LOCATE('@',al.mail)+1)
WHERE dom.domain is null
ORDER BY al.mail ASC
```

**non local destination aliases**

```sql
SELECT al.*
FROM aliases al
LEFT JOIN domains dom
   ON dom.domain = SUBSTRING(al.destination,LOCATE('@',al.destination)+1)
WHERE dom.domain is null
ORDER BY al.enabled, al.destination ASC, al.mail ASC
```

**aliases for a certain domain**

```sql
```
SELECT al.*
FROM aliases al
WHERE SUBSTRING(al.mail,LOCATE('@',al.mail)+1) = 'domain.tld'
ORDER BY al.enabled, al.mail ASC

aliases for a certain domains, checking if enabled for both domain and alias

select *
from domains d
join aliases a
  on a.mail like concat( '%','@',d.domain)
  and a.enabled = 1
where d.enabled = 1
and d.domain like '%foobar%'
order by d.domain,a.mail
8.1

- Missed a step
  
  If you mistakenly or intentionally skipped past sections, you may have missed an important step in your configuration, which my guide presumes you have followed.

- Typo
  
  99% of all problems is spelling errors or typos you entered while following this guide. Sorry, but it just happens. Often it can be trivial, such as a space at the end of the configuration line which was not expected etc. Or not understanding my example where it is a multi line entry.

- Typo by me
  
  Yes, I make a lot of mistakes. Nothing wrong in that, but I hope I have corrected most over time. Any new sections are however at risk... :)

- Different application or configurations
  
  It is obviously entirely up to you how you set up your system. But the more you deviate from this guide, the more likely incompatibilities or confusion will arise.

- Distribution/version differences
  
  If you run a different version or even distribution to this guide, then some things will be different. Small issues, such as default values and significant things such as path differences etc. Some sections in this guide are not always thoroughly tested with every new release of Ubuntu, but these differences gets pointed out by people for me.

- Walking before crawling
  
  Don’t try the full blown mail server before the basics are working.

- Gamma rays and little goblins
  
  Got to blame it on something or someone.

8.2 strategy

What steps to think of when testing.
8.2.1

It is very helpful to test early in this set up, to establish if the first sections are working as expected.

So when you only have your very basic postfix and mysql up: Test it!

That way you know that bit worked and can rule it out of any future problems.

Don’t wait till you complicated and muddied the water after amavis, courier etc is added.

By constantly testing if you can send and receive you can tick off and black box each section as working, and immediately spot issues.

8.2.2

Testing how things work is often about isolating the problem first. So by using the steps of testing early above, you can see which step caused the problem.

Also if you can’t log into your webmail it is often nothing to do with the webmail section that is causing the problem. Often postfix itself is broken etc.

8.2.3 Test in order

As part of the isolating the problem rule, you most of the time test in order, and test each section thus isolating the problem. This would then quickly isolate the problem when e.g. such as above issues of reading emails via the webmail. This would be in order:

1. Access: Can I get(ssh) to the box, and is there a firewall issue?
2. Database: Is the database up, do my application reach it?
3. Postfix: Can I send email by command line, do I receive emails via telnet?
4. Content checks: Do they cause a problem?
5. Courier: Can I read emails?
6. Webmail: And last but not least does the web integration work?

8.2.4

Assisting in isolating the problem, you often have to disable options and applications. Such as turn off postgrey or content checks to make sure emails get delivered.

Previous editions do have some more detail on how to achieve this

8.3

Essential to monitor what actually happens, and tailing specifically the mail and mysql log.

- /var/log/system.log
- /var/log/mail.log
- /var/log/mysql/mysql.log
- /var/log/apache2/access.log
In one window:
And in another window:
In a third or more do your actual configuration or testing.

8.4

previous edition 1
previous edition 2
The previous editions has detail on switching services off untill time to test them.
It also details locking down your server from spammers untill finished testing.

8.5

Shorewall
You can also switch on more messages for when the firewall is rejecting connections. Add info to all REJECT, BOUNCE and DROP policies.

```
sudo vi /etc/shorewall/policy
```
such as:
```
net all DROP info
```

MySQL
There is no point in tailing the mysql log if query debugging is not turned one.
By default it is not. However in this guide I do switch it on, in case that was missed switch it on now:

```
sudo vi /etc/mysql/my.cnf
```

Make sure this is not commented out
```
log = /var/log/mysql/mysql.log
```

Courier
As mentioned in the setup, switching on debugging for Courier is easy:

```
sudo vi /etc/courier/authdaemonrc
```
```
DEBUG_LOGIN=2
```

Amavis
You can also debug amavis:

```
sudo vi /etc/amavis/conf.d/50-user
```
And perhaps bump it up if already debugging:
```
$log_level = 2;
```
8.6 Telnet

When testing a mail server, telnet is alpha & omega. You use it to simulate real mail servers to test responses by your mail server.

First you test it on the server to exclude firewall and network issues.

Then you test it from another machine to simulate an actual other mail server.

Once these are working you can use proper email clients, however 99% I just use mutt locally when I need to test if a server is working.

8.7 Can postfix receive?

Lets assume: You have followed my guide up to basic configuration at least You have entered data into the database The services MySQL and Postfix are running. If testing a fuller stack, then amavis, postgrey, clamav-daemon, spamassassin etc must also be running. Try this locally on the server first, then try from another machine once it is working locally. Lets try and send a message to xandros@example.org (replace with your own user in this setup, or use postmaster@localhost) from you@example.com (again replace with a real email address you use that is not associated with this server.)

telnet localhost 25
#
# Open the hand shake with ehlo and the server name you are connecting from...
#
# Change mail.example.com to something valid eg your servername
EHLO mail.example.com
#
# The mail server will then dump out some details about its capabilities, e.g.
>250-mail.flurdy.net
>250-PIPELINING
>....
>....
#
# then say who is the sender of this email
MAIL FROM: <your@example.com>
> 250 Ok
#
# then say who the mail is for
RCPT TO: <xandros@example.org>
> 250 Ok
#
# then enter the keyword data
data
#
# enter message bodyand end with a line with only a full stop.
blah blah blah
more blah
,
> 250 Ok; queued as QWKJDKASAS
#
# end the connection with
quit
> 221 Bye

If while you were doing this you were tailing the /var/log/mail.log you would see some activities and if any errors occured. (You should probably get some complaints about missing headers as we skipped most...)

If while you were doing this you were tailing the /var/log/mysql/mysql.log as well you really should have seen some activity otherwise you have a problem.

If you see any errors (or worse no activity) in these log files, this is what you need to fix! For common problems and solutions check the previous edition.
However if no errors popped up, and the folder /var/mail/virtual/xandros now exists then your server can receive emails!

8.8 Can postfix send?

You need to make sure you can first receive emails as above

The services MySQL and Postfix are running.

Basically you just tested that above, but we need double check if it can send out to other servers. Again we will first test locally, which should work, then remotely which introduces many possible problems.

telnet localhost 25
# Open the hand shake with ehlo and the server name you are connecting from...
# This time it has to be the name of your server
EHLO mail.example.org
# The mail server will then dump out some details about its capabilities, e.g.
>250-mail.flurdy.net
>250-PIPELINING
>....
>....
# then say who is the sender of this email, which is a local user
MAIL FROM: <xandros@example.org>
> 250 Ok
# then say who the mail is for which is an external address e.g. gmail etc.
RCPT TO: <you@example.com>
> 250 Ok
# then enter the keyword data
data
> 354 End data with <CR><LF>.<CR><LF></LF></LF></LF></CR></LF>
# enter message bodyand end with a line with only a full stop.
blah blah blah
more blah
.
> 250 Ok; queued as QWKJDKASAS
# end the connection with
quit
> 221 BYE

We have to assume receiving works above so no need to tail mysql’s logs. However if any rejection errors occured in the mail.log then you have an error.

However if no errors occured and you see in the log something like this:

Dec 17 10:25:45 servername postfix/smtp[12345]: 12345678: to=<you@example.com>, relay=127.0.0.1[127.0.0.1]:10024, delay=15, delays=15/0.01/0.02/0.11, dsn=2.0.0, status=sent (250 2.0.0 Ok, id=12345-09, from MTA([127.0.0.1]:10025): 250 2.0.0 Ok: queued as 1234567)

Then the sending emails work!

8.9 Can courier read emails

You need to make sure you can first receive emails as above

You need to make sure you can send emails as above
You need to make sure you have received an email and the folder /var/mail/virtual/xandros exists. The services MySQL, courier-authdaemon and courier-imap are running.

There is not too much you can test via telnet for courier. But you can check if it is up and you can connect to it.

telnet 127.0.0.1 143
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.

The rest you would have to test via a proper email IMAP client.

Can amavis check and pass emails along?

You need to make sure you can first receive emails as above

You need to make sure you can send emails as above

You need to make sure you have received an email and the folder /var/mail/virtual/xandros exists

You can check if the service is responding:

telnet 127.0.0.1 10024
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
220 [127.0.0.1] ESMTP amavisd-new service ready

Then just tail /var/log/mail.log for any problems.
Brief hints if you receive a ready setup machine (or EC2 AMI), and what then to check and to customize it to your setup.

- Stop services
- Restrict firewall
- Change passwords
- Check configurations
- Set machine name
- Certificates
- Start and test services
- Insert data
- Reload postfix
- Open firewall
- Test

9.1

First stop services so they won’t accidentally do something.

```bash
sudo service spotfix stop
sudo service courier-imap-ssl stop
sudo service courier-imap stop
sudo service courier-authdaemon stop
sudo service mysql stop
sudo service amavisd stop
sudo service spamassassin stop
sudo service clamav stop
```

9.2

Check what the firewall rules are.
vi /etc/shorewall/rules

Refer to the firewall settings. Restrict to just SSH access for now.

9.3

Next the passwords needs to be changed. For both the system and mysql.

System passwords
Check which users are defined on the system.
cat /etc/passwd

Apart from all the system ones, there should probably be none (if EC2 AMI) or just your user if it is a standard Ubuntu install. If there are some users, you need to change their passwords.

SSH Access
Next we check whom got SSH access. If there was any users defined, check their home folders for ssh keys.
cat /home/username/.ssh/auth*

Remove any you do not expect to be there. Next check if and which specific users has been defined for SSH access in
vi /etc/ssh/sshd

Usually this is fine.

9.4 MySQL

First you need to change the root mysql user. If none has been set do this
mysqladmin -u root password new_password

Otherwise do this and you will be prompted for the old password
mysqladmin -u root password new_password -p

Then the default mail user as well. If you know the old password
mysqladmin -u mail password new_password -p

Otherwise log into mysql as root:
mysql -u root -p

Enter new root password specified above, then:
update mysql.user set password=PASSWORD('apassword') where user='mail';
flush privileges;

You may need to revisit the top of MySQL section to re-grant the mail use rights on the database.
If you do not know the old root password, you have to restart mysql without grant rights. Google it... :)
Update postfix mysql configuration files with the new password.
sudo vi /etc/postfix/mysql*
password=apassword

Update courier’s authmysql file with the new password as well.
sudo vi /etc/courier/authmysqlrc
MYSQL_PASSWORD apassword

If SASL is set up, then you need to update its passwords. First in postfix SASL file:
sudo vi /etc/postfix/sasl/smtpd.conf
sql_passw: aPASSWORD

Then on both lines in:
sudo vi /etc/pam.d/smtp
passwd=aPASSWORD

9.5

You should scan the postfix, courier, etc. configurations to check if they match what you expect.

9.6

Now you need to define your machine name, e.g. something like smtp.yourdomain.com. You need to define it in
sudo vi /etc/mailname

And then your domain name in
sudo vi /etc/postfix/main.cf
under the mydomain setting
myorigin=yourdomain.com

It could also be smart to check what the unix hostname is specified as
hostname
This can be reset by
sudo hostname smtp.yourdomain.com.

All though this does not have to be the same as your postfix mail server name. You may want to speficiy some hosts in hosts file as well,
sudo vi /etc/hosts
127.0.0.1 localhost.localdomain localhost
127.0.0.1 smtp.yourdomain.com smtp
9.7

You could go along with the generated certificates (if they are there, default for Ubuntu). Or if you could create new ones with the correct machine name in them. Especially if this a mail server used by many, and authenticity is important. Follow the TLS certificate instructions for Postfix and Courier.

9.8

Next you need to start your mail services and test them.

```
sudo /etc/init.d/mysql start
sudo /etc/init.d/spamassassin start
sudo /etc/init.d/clamav start
sudo /etc/init.d/amavisd start
sudo /etc/init.d/postfix start
sudo /etc/init.d/courier-imap-ssl start
sudo /etc/init.d/courier-imap start
sudo /etc/init.d/courier-authdaemon start
```

So test tjenestene via testing section.

9.9

Insert your mail domains, aliases and users using the data section.

Some times there are test data already in the database. Remove them. E.g.;

```
mysql -u mail -papassword maildb

delete from domains where domain = 'bar.com';
delete from aliases where mail = 'foo@bar.com';
```

9.10

Then open up the firewall, follow the world access bit in the firewall configuration. Voila. Up and running. Well we hope.
Please refer to previous edition for how and why you can extend this mail server.

By now you should have a fully working system. No point extending and complicating it untill then. What next? There are many ways to extend the server, to create your own powerful customised version.

Some of these sections can be brief as they are not core to this howto.

10.1 MX

With MX backup losing emails are unlikely.

Normally if someone sends an email destined for you, their server will try and connect to your server. If it can’t reach your server for whatever reason (it is down, dns issues, there is network problems, or just too busy), the other server will back off and try again in a bit. How many and for how long it will try again is determined by the sending server. Some of them are not very patient, and it will report undelivered after only a few attempts. So you would have lost that email.

If you had specified a backup MX, this email may not have been lost. Upon first failure to connect to your server, the sender would see if there is any alternative server to send to. So it connects to your backup mx server. This server spools and queues your message and will try at intervals to send the message to you. It too will though eventually give up.

What is the difference? Simple, you (or whoever controls the backup mx) is in control how long and often to try connecting to your machine. So if you have a reasonable values and your server is not down for weeks, no mail is lost.

How to implement it? First edit the DNS records again, and add a backup mx with a higher value.

```
# your server details
domain.tld  IN  MX  10  your.mailserver.name.tld
# new backup server
domain.tld  IN  MX  20  your.backupserver.name.tld
```

Now presuming the other backup mx is a postfix server identical to this, or you are backing up someone else’s server; Go into mysql and create this tables:

```
CREATE TABLE `backups` (  
`pkid` smallint(6) NOT NULL auto_increment,  
`domain` varchar(128) NOT NULL default '',  
`transport` varchar(128) NOT NULL default '[]',  
`enabled` smallint(6) NOT NULL default '1',  
PRIMARY KEY (`pkid`),
```
Then still on the backup server, edit main.cf and add these:

```plaintext
relay_domains = mysql:/etc/postfix/mysql_backups.cf
transport_maps = mysql:/etc/postfix/mysql_transport.cf
```

You may choose to have this as the last line in the file, as you may use small cron jobs to modify this ip address, if you don’t have a permanent static address. It should contain your IP address, hence if you do not have a very static IP address, that you need to automatic editing if the postfix file.

```plaintext
proxy_interfaces = 1.2.3.4
```

If someone comes with a better way, then let me know.

Next create this file /etc/postfix/mysql_backups.cf

```plaintext
user=mail
password=apassword
dbname=maildb
table=backups
select_field=domain
where_field=domain
hosts=127.0.0.1
additional_conditions = and enabled = 1
```

Next create this file /etc/postfix/mysql_transport.cf

```plaintext
user=mail
password=apassword
dbname=maildb
table=backups
select_field=transport
where_field=domain
hosts=127.0.0.1
additional_conditions = and enabled = 1
```

You noticed I added a transport lookup. This is a field in both the domain and the backup tables. In domains it is used to determine how to deliver the email, ie either virtual (correct) or local (not used in this howto). When backing up servers, your also need to specify in the transport field how to connect to the correct servers.

Say you are backup for a friends server, mail.friend.com, for the domains of friend1.com and friend2.com. So you should insert this into your backup table.

```plaintext
INSERT INTO backups (domain,transport)
VALUES ('friend1.com', ':[mail.friend.com]'),
('friend2.com', ':[mail.friend.com]');
```

The :[] tells to connect directly to this server, not doing any more look ups for valid MX servers.

This shouls now work fine. Further tweaking of the queue values, review these and modify as appropriate. Shorter warning times are good for the sender, so that they realise the email has not arrived yet, but may also be annoying. Tradeoffs.. Look in the first main.cf configurations for ways to do so.

### 10.1.1

Unfortunately spammers are using backup mx as a way to saturate the networks with invalid emails, known as backscatter mail.
They simply lookup a domain’s MX servers and connect directly to one of the lower priority servers whom may be just a backup mx. This server if configured as above will not check for valid addresses aliases but will accept and queue all emails for the domain’s it is configured as a backup mx for. These will then be delivered by the server later to the primary MX server, whom will then maybe reject them as the aliases are not valid. However the sender addresses are often invalid and a long trail of reject messages to and forth around the net follows...

To avoid this you can enable relay recipient lookup in Postfix.

Edit /etc/postfix/main.cf and add:

```
relay_recipient_maps = mysql:/etc/postfix/mysql_relays.cf
```

Then create a new file /etc/postfix/mysql_relays.cf

```
user=mail
password=apassword
dbname=maildb
table=relays
select_field=recipient
where_field=recipient
hosts=127.0.0.1
additional_conditions = and enabled = 1
```

Then add the following MySQL table:

```sql
CREATE TABLE `relays`
(`pkid` smallint(6) NOT NULL auto_increment,
 `recipient` varchar(120) NOT NULL default '',
 `enabled` tinyint(1) NOT NULL default '1',
 `status` varchar(10) NOT NULL default 'OK',
 PRIMARY KEY (`pkid`),
 UNIQUE KEY `recipient` (`recipient`)
);
```

If the relay_recipient_maps setting is set, then postfix will reject all email addresses not specified in this table. As with many postfix lookups, it will recursively search for a match from the full address. In the following examples, emails to john@example.com are the only emails that will be accepted for the whole example.com domain. However for @example.org all emails will be accepted for backup, except any for support@example.org which will be rejected.

```
insert into relays (recipient,status) values
('john@example.com','OK'),
('support@example.org','REJECT'),
('@example.org','OK');
```

### 10.2

Here is rough backup script to backup your configurations and mail folders. You may want to backup the folders separately as they can quickly grow to GBs. Adding this to a cronjob automates this process. Be aware that you should stop postfix and courier while backing up the mail folders. And that if they have grown large, that this may take some time.

```
tar czf mail-config.xxxxx.tgz /etc/postfix /etc/courier /etc/spamassassin /etc/clamav /etc/amavis /etc/mysql/my.cnf

tar czf mail-fold.xxxx.tgz /var/spool/mail/virtual

mysqldump -u mail -papassword -t maildb > data.sql 
mysqldump -u mail -papassword -d maildb > schema.sql

tar czf mail-data.xxxx.tgz schema.sql data.sql

tar cf mail.xxxxx.tar mail-*.*.xxxxx.tgz
```
You may combine a full backup with an intermediate update of what has changed recently only.

```
tar --newer-mtime "2005-01-01"
```

## 10.3 IDSPF

Further security features is using Microsoft’s Sender ID or Pobox’s SPF. I’d use SPF as there is much argument over Sender ID.

spf.pobox.com

www.microsoft.com/mscorp/safety/technologies/senderid/

SPF should limit who can send mail on behalf of your domains, and is an open design. I do recommend SPF, with some reservations, detailed below.

While Microsoft is not always entirely evil, as sometimes they do nice things and make some useful software, I would prefer not to be locked into their Sender ID technology.

### SPF configuration

The pobox site has some nice SPF generation tools to setup your SPF configuration. Probably best to use theirs.

But the way I have my setup, is generally one domain with detailed SPF, then all other domains just with an SPF alias to it. e.g:

**Main domain DNS TXT field:**

```
“v=spf1 a mx a:myserver.example.com include:aspmx.googlemail.com include:gmail.com ~all”
```

The important elements are:

- I list the mail servers and websites associated with this domain (the a and mx bit).
- I then specifically list the name of a server I may send mail from applications automatically using addresses within this domain.
- As you can see I also use Google Apps with this domain, thus tell SPF to also allow all mail servers associated with google mail.
- Then for most of the other domains I would use this DNS TXT field:

```
“v=spf1 a mx include:example.com ~all”
```

The important elements are:

- I list the mail servers and websites associated with this domain
- Then I tell SPF to also allow all mail servers associated with my main domain (example.com).
- And for all these I use ~all!

Ps. Some domains I have added an even stricter SPF, as these are domains that will never send an email.

### SPF problem

It is worth noting about SPF, that you should leave the decision to whether to reject or allow the email to the mail servers. Therefore using -all instead of ~all is not a good choice. Leave it to the SPAM scoring by the receiving server, like SpamAssasin does it. You then minimise the risk of false positives.

One of the reason I do discourage -all use, is that SPF has a distinct problem:

It does not like email forwarding or use of backup MX!
Consider this: Your address of lulu@hoopa.com sends a joke email to a few friends. One of these is trixie@bellbell.org.

Trixie’s email address is actually an alias and forwards the email to her private webmail account on hotmailnot.com.

Now if your domain, hoopa.com, have a strict SPF set up, which only allows emails to be sent by its mail server. And you/the mail admin has added -all to the SPF, which tells other server to reject emails not from your server. This you think makes sense, spammers can not use your domain for spoof emails.

So what happens: bellbell.org receives the email from lulu, and possible checks the SPF, which is OK, and forwards it on to hotmailnot.com.

However if hotmailnot.com also checks SPF, it will receive the email from bellbell.org, check the SPF to see bellbell.org’s mail server is allowed to send emails on behalf hoopa.com. SPF will say No!, and with the -all, hotmailnot.com email server will reject the email!

2nd scenario if lulu email trixie directly at hotmailnot.com, but hotmailnot.com main mail server was down, and email was sent to the backup mx server. When the main server came online again, and the backup spooled the email back to it, the SPF would again fail as the hoopa.com’s SPF would not mention hotmailnot.com backup mx as an allowed mail server.

Solution:

Of course you can not list all possible forwarding / backup mx email server that your domain’s users might at some point email!

I simple just use the ~all option. Which simple say it is not the expected server, but probably ok.

And if this is added to a scoring by the receiver, then the accumulated spam score might be enough to reject dodgy emails.

10.4
todo

Reporting spam to Pyzor, Razor and SpamCop, for collaboration in spam fighting.

More detail on SpamCop is here.

pyzor.sourceforge.net
razor.sourceforge.net

10.5 /
todo

You can implement white and black lists to explicitly allow or block domains and users.

You have already visited the option of a blackhole list of known open relays in the postfix configuration.

You can implement further lists inside Postfix or SpamAssassin. Amavisd-new already has a few well known white/black listed items in its config files. SpamAssassin also as a feture to automatically learn white lists.
10.6 PGP & S/MIME

Adding support for GnuPG and S/MIME increases individual security.

This is not implemented on the postfix server side, as this totally a client side option.

However SquirrelMail has a GnuPG option. It is a plugin that can be downloaded from their website. Which can then be enabled via SquirrelMail’s config script.

Here is how to create a GnuPG key pair.

```
# check you have not already got a key
gpg --list-keys
# then create one

gpg --gen-key
```

To import GnuPG into Evolution; in your settings/preferences edit your account settings and add you private key under the security tab. The private key is found via listing the GnuPG keys as above, then it is the 8 characters after the “sub 1024g/” bit of you key.

To use GnuPG with Thunderbird you need to install EnigMail.

S/MIME is another way to encrypt and/or sign messages. You can create you own certificate or use known organizations like Thawte. (Thawte was originally set up by the Ubuntu founder)

10.7

If people change addresses, a bounced message stating so if people send email to the old address is quite useful. To implement this in postfix, first create a lookup table in the database.

```
CREATE TABLE 'relocated' ( 
    'pkid' smallint(6) NOT NULL auto_increment,
    'oldadr' varchar(128) NOT NULL default '',
    'newadr' varchar(128) NOT NULL default '',
    'enabled' tinyint(1) NOT NULL default '1',
    PRIMARY KEY ('pkid'),
    UNIQUE KEY 'oldadr' ('oldadr')
) ;
```

Then add this to /etc/postfix/main.cf

```
relocated_maps = mysql:/etc/postfix/mysql_relocated.cf
```

The create this file /etc/postfix/mysql_relocated.cf

```
user=mail
password=apassword
dbname=maildb
table=relocated
select_field=newadr
where_field=oldadr
hosts=127.0.0.1
```

Then if pete@domain1.com has changed address to pete.jones@another.org:

```
INSERT INTO relocated (oldadr,newadr) VALUES
    ('pete@domain1.com','pete.jones@another.org');
```
If anyone sends an email to pete@domain.com, they will get a message back stating he has changed address to pete.jones@another.org.

10.8 Pop-before-SMTP

If SASL didn’t work, or you are using clients which don’t support it, the Pop-Before-SMTP is an easy way around that issue, so that people externally can still securely send mail via your server.

Refer to my 2nd edition on Pop-before-SMTP setup.

10.9
todo

Trying out a few admin software might make you life easier, if phpMyAdmin gets to crude. Quick search More to come later.

10.10

10.11

If you use catch alls, which are useful for some domains, then eventually some addresses will be target for spam. You can then either stop the catch all, or stop individual addresses.

By implementing a lookup and adding this restriction to smtpd_recipient_restrictions accomplishes this.

check_recipient_access mysql:/etc/postfix/mysql_block_recip.cf,

smtpd_recipient_restrictions = permit_mynetworks, permit_sasl_authenticated, \
check_recipient_access mysql:/etc/postfix/mysql_block_recip.cf, \
reject_non_fqdn_recipient, reject_unauth_destination, \
check_relay_domains

Beware of the order is important here, if any options says ok before check_recipient_access it will ignore it.

Next create mysql_block_recip.cf to lookup addresses. Either create a another table, or add a blocked field to aliases table.

10.12

10.13

Rich Brown has written a howto on adding Mailman, a mail list program, to my howto. Click here to read it.

Do note it is not part of my howto, so do not contact me regarding it. And although I think it is fine, I can’t guarantee it will work.

If you do need assistance or need to talk about it, contact Rich via his howto or use the forums for this howto.

If you want a simple mailing list, it can be implemented by simply separating aliases in the destination field in the aliases table with a comma.
Set Up Mail Server Documentation, 1.0

```
INSERT INTO aliases (mail,destination) VALUES ('listof@domain.com', 'john@ppp.com,vic@domain.com,jj@somewhere.tld');
```

10.14 Google Apps / GMail

I have for various reasons integrated some Google Apps hosted domains into my mail server. And you can still have good control over the addresses by using your server with Google Apps.

More information on Google Apps.

Some already have their domain’s email hosted with Google.

Some people prefer Google’s web based interface.

Temporary Migrations.

Include Google’s security features on top of yours.

Options

The easiest and simples solution is not to have a domain MXed to your server, and simply alias email to those domains. eg All email to joeblogs.co.uk hosted on your server are forwarded to joeblogs.com hosted with google.

You may set up your own server to simple be a mail server backup (mx) for a domain hosted with google. If you are the first priority in the MX details of the DNS, you still have some control, but not all will obey the priority listing. E.g. spammers, but some valid senders as well.

However the one I use and the option where you are most in control is to keep you server as the only MX server in the DNS. And only forward certain aliases onto Google after all your servers checks. Other aliases and user can just use your mail server if you prefer. I will explain how to do this in the next steps.

DNS

You only put your mail server as the mx for the domain in question. Google will complain about this, as it will not be able to verify that email is setup correctly. Ignore this as it will still accept emails.

MySQL tables

You setup you aliases as normal. However you domain table needs tweaking. This is because otherwise your server will just forward the email to itself. You can actually specify aliases in the domain table.

Example

If for example:joe@bloggs.com wants to use gmail. mary@bloggs.com does not.

If not already configured as a backup mx:

Add a transport lookup to your /etc/postfix/main.cf file:

```
transport_maps = mysql:/etc/postfix/mysql_transport.cf
```

Then create /etc/postfix/mysql_transport.cf file:

```
user=mail
password=apassword
dbname=maildb
table=backups
select_field=transport
where_field=domain
hosts=127.0.0.1
additional_conditions = and enabled = 1
```
Assuming there are no bloggs.com data in any tables (domain, alias, users, relays, backups):

```sql
insert into domains (domain, transport) values
('joe@bloggs.com', 'smtp: [aspmx.l.google.com]:587'),
('bloggs.com', 'virtual:');
insert into aliases (mail, destination) values
('joe@bloggs.com', 'joe@bloggs.com'),
('mary@bloggs.com', 'mary@bloggs.com');
insert into users (id, name, maildir, crypt) values
('mary@bloggs.com', 'mary', 'bloggs.com/mary', encrypt('maryspassword'));
```

The domains insert is the interesting one. The transport map lookup checks recursively for an alias match and will first look for user@domain before it looks at the general bloggs.com for which transport to use. The square brackets around aspmx.l.google.com indicates that this server will not lookup for mx settings for this domains DNS, but instead connect directly. (This can avoid never ending recursive lookups/relays)

Note if you have backup mx configured and chosen to enable relay recipient lookup to avoid backscatter mail spam, then you need to add your Google Apps users to the relays table:

```sql
insert into relays (recipient, status) values ('joe@bloggs.com', 'OK');
```

Refer to backup mx section for creation of this table.

**TLS certificate**

If you have set your server up to prefer TLS then you should add Google’s signing authority to your server’s root certificate list. Google used to use Thawte but now use Equifax.

Download the Equifax Secure Certificate Authority certificate from their website (the base-64 encoded):

```bash
wget http://www.geotrust.com/resources/root_certificates/certificates/Equifax_Secure_Certificate_Authority.cer;
```

You need to fix the line endings in this file by either using sed:

```bash
sed -i 's/$// Equifax_Secure_Certificate_Authority.cer;
```

Or install a tiny util:

```bash
sudo aptitude install tofrodos;
```

```bash
fromdos Equifax_Secure_Certificate_Authority.cer;
```

Put it into your certificate root folder:

```bash
sudo chown root:root Equifax_Secure_Certificate_Authority.cer;
sudo mv Equifax_Secure_Certificate_Authority.cer /usr/share/ca-certificates/mozilla/;
cd /etc/ssl/certs;
sudo ln -s /usr/share/ca-certificates/mozilla/Equifax_Secure_Certificate_Authority.cer .;
```

And then append it to the root list that postfix knows about:

```bash
sudo su;
cat Equifax_Secure_Certificate_Authority.cer >> ca-certificates.crt;
exit;
sudo service postfix restart;
```

**Issues**

There are some items you should consider when integrating Google Apps.

**Privacy**

First there is the privacy issue. This is the same as if you were using Google Apps only or GMail. Google can and will read your email. However probably not a person, but they will use it for commercial reasons, E.g. showing relevant
ads. Some people really hate this part and refuse to use Google’s mail products. However I trust them a little bit, and do use it.

Spam
If you forward spam, then consider your own servers reputation. Should be okay though.

SPF
If you use SPF for your domain, consider that both your server and google will receive and send mail on behalf of that domain. Adding include:_spf.google.com

should cover it.

Google internally
Be aware Google think they host you domain. So if others inside google, or using google hosted apps or GMail, if they email you, the email may not go via your email server, but directly to the Google Apps for your domain. That could be an issue if not all aliases you have use Google Apps. This needs to be tested more though. Especially as it may only be an issue if Google’s servers are part of your domains MXs. It may be worth adding aliases in your Google Apps admin for the non google apps addresses to some user whom can handle these?

Maildrop, spam folder and vacation messaging
Villu have documented swapping in Maildrop for virtual transport and automatically deliverin spam to a spam folder. (And links to a post about vacation messaging) Please read his post here.

10.15 Squirrel Mail


You need to copy a SquirrelMail configuration to apache.

```bash
sudo cp /etc/squirrelmail/apache.conf /etc/apache2/sites-available/squirrelmail
```

And enable with this:

```bash
sudo ln -s /etc/apache2/sites-available/squirrelmail /etc/apache2/sites-enabled/500-squirrelmail
```

Or as Florent recommends, use:

```bash
sudo a2ensite squirrelmail
```

You may accept the default apache configuration where squirrelmail is folder in all sites. But I prefer virtual hosting. But you dont need to do these next steps.

```
sudo vi /etc/apache2/sites-available/squirrelmail
```

Comment out the alias.

```
# alias /squirrelmail /usr/share/squirrelmail
```

Uncomment the virtual settings., and insert your servers name.

```
# users will prefer a simple URL like http://webmail.example.com
DocumentRoot /usr/share/squirrelmail
ServerName webmail.example.com
```

If you have apache SSL enabled in apache, then you can also uncomment the mod_rewrite section for further security. Reload apache to activate changes. First test if ok.
sudo apache2ctl -t

Then reload it.
sudo /etc/init.d/apache2 reload

You can now go to yourdomain.com/squirrelmail/ or mail.yourdomain.com if you chose virtual host. This should show a squirrel mail page. Log in won't work yet though.

Start configuring squirrel mail.
sudo squirrelmail-configure

Initially change nothing. You can customize more afterwards. You can browse, and exit sub menus by typing R.
Type 2 to edit server settings. Type A to edit IMAP settings.
Type 8 to edit server software. Enter courier.
Now they say using TLS over localhost is a waste of time. But I do anyway. Type 7 to edit secure IMAP and Y to enable it.
Type 5 to edit IMAP port. Enter 993
Type S to save your changes. Hit Enter.
Type Q to exit.

You can now go to yourdomain.com/squirrelmail/ or mail.yourdomain.com if you chose virtual host. This should show a squirrel mail page. Log in will now work. (Except you may not have defined users, check data section. And they may not have received an email which also means you can not view any IMAP info.)

Please refer to previous edition for more detail. E.g. creating address books and user preferences.

10.16

Preventing Brute Force attack on your server.
First line of defence is the firewall. If they can't get to your server then they can't hack in. However to be a useful internet based server you have to expose some services, e.g. SMTP and maybe SSH.
Below are two widely used ways to protect yourself.

10.17

Deny hosts is an effective tool to protect your SSH service from brute force attack. To install:
sudo aptitude install denyhosts

Tweak in /etc/denyhosts.conf. Perhaps whitelist your ips to prevent accidentally locking yourself out... You do this via /etc/hosts.allow.

Read more in this thread for tweaks.

To protect your server against distributed attack, read about DenyHosts' synchronisation feature.
10.18

fail2ban protects against a multitude of brute force attacks. Relevant to this guide is the protection for SMTP and IMAP. Follow this guide for how to install & configure it.

10.19

If you have any suggestions to other ways of extending a postfix server, then fire off a mail to me via the contact form further down.

(Or rather, I’d prefer that you write down the extension, and let me know the link! :))
Impressions

Easy to use. Anyone can use, not just big companies. Very useful. Tools are command line but simple. Firefox extensions work well. Recommended.

I find it very usefull. Basically it is a colo hosting environment. Some may use it as for Saas, ie single scalable application in the cloud, but I use it as a hosting environment for complete servers.

ec2 introduction, tips and howtos

I have made a separate tips and howto on the use of ec2 for general server needs. Hope it will be useful for people.

How I use it for my mail servers

Different images to launch for different needs. Good way to scale backup MXs if needed. Can script backup to S3 of mail dirs etc.

Using EC2 with this howto

If you plan to use EC2 to follow this howto, then familiarise yourself with EC2 first. Check the links further down, e.g. my tips.

Once competent enough on EC2, launch the latest official Ubuntu ec2 image or one of Eric Hammond images. You can cheat by using my other images, but you should really know how the whole server was built by starting from the bottom.

When using EC2 images, be aware of security groups as they restricts access to your server on top of the firewall. Initially you will need SSH (22) access, quite soon you will need SMTP and IMAP ports opened, 25,143,465,587 and 993, and eventually webserver ports of 80 and 443. Read here for tips on securing AMIs.

Also do not terminate your instances without backing up your machine. This you can do by either create your own image. Or backup certain data if you got an image to instantiate from. Back up to S3 or your local machine. Create images only now and then. Backup configurations, database, maildirs more regularly.

Once launched, follow my Initialize section.

1st note: Spamhaus.org lists amazons ec2 ip ranges as dynamic, thus many mail servers will reject emails from it. (Including other people using this howto.) But Spamhaus has a simple web page to remove ips, which they link to in rejection messages. Simple look in your logs, click on the link on follow the instrucions: basically fill in your ip, email and state its for a mail server. Then Spamhaus will remove your IP from their database.

2nd Note: Amazon has extended this spam limitations, so if you have a busy mail server, follow their FAQ entry for removing mail throttling.

3rd Note: This fix needs to applied to the instances buildt on an early 8.04 base. This is not a problem with the later Hammond or any Canonical based images.
4th Note: Check I have not left my SSH key in the root or ubuntu user.

5th Note: Please let me know if I’ve been silly enough to leave a bash history or anything like that?

Amazon EC2 Images: AMIs

Public AMIs to use as base:

Current AMIs:

<table>
<thead>
<tr>
<th>AMI Description</th>
<th>S3 Name</th>
<th>Ubuntu version</th>
<th>Extended from</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ubuntu's official server AMI</td>
<td>ubuntu-images/ubuntu-lucid-10.04-i386-server-20100427.1</td>
<td>10.04 LTS</td>
<td>Lucid</td>
<td>OK US</td>
</tr>
<tr>
<td>My base Ubuntu server AMI</td>
<td>ami-57f28d05</td>
<td>10.04 LTS</td>
<td>Lucid</td>
<td>ami-2d4aa444 Ubuntu’s official base OK EU</td>
</tr>
<tr>
<td>Basic Postfix server, with MySql, amavis, TLS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Postfix not configured.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flurdy-amis/ubuntu-lucid-mailserver_base-20110701-01</td>
<td>ami-9c947cf5</td>
<td>10.04 LTS</td>
<td>Lucid</td>
<td>ami-f8a64e91 flurdy’s base Testing US</td>
</tr>
<tr>
<td>Basic Postfix Testing US</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Older AMIs:

( Images with strike through them are no longer recommended. Their are fine for experimenting and testing, but should not be used for permanent “live” servers )

<table>
<thead>
<tr>
<th>AMI Description</th>
<th>S3 Name</th>
<th>Ubuntu version</th>
<th>Extended from</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canonical’s Official Ubuntu 9.10 Karmic</td>
<td></td>
<td>9.10 Karmic</td>
<td>ami-5059be39 Base install: Canonical’s Official Ubuntu 8.10 Intrepid 32bit US</td>
</tr>
<tr>
<td>Canonical’s Official Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Canonical’s Official Ubuntu 8.10 Intrepid 32bit US</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alestic Ubuntu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
12.1

Ivar Abrahamsen, an IT Senior Consultant from Norway. Specialising in developing and integrating middleware application systems. Mainly open source and Java based technology stack. Recently moved back to UK (Hampshire) after 5 years in Oslo and before that 15 years in Manchester.

12.2

Remember I have stood on the shoulders of giants. I just ended up with a system that worked for me, and decided to document its evolution.

12.3

Before contacting, have you?: * Read this document properly? Followed it step by step?

(While we can not insist on the same setup for everyone, assistance is easier and more likely if less customised)

• Have tested properly? Applied the solutions provided in the test section?
• Read the forums for solutions already posted?
• Read the FAQ?
• Tailed the mail.log? It usually tells you what the problem is!
• Tailed the mysql.log? If nothing happens there it should indicate something...
• Include a short dump of the mail.log with your post. (Remember to anonymise the servernames, email adresses & definitly passwords!)

12.4

Use the Ubuntu forums! :)

• Here is a thread on this specific mail server howto.
• And another one by me which is also used..
• Or alternatiely use Server Fault.

Please participate in the forums.

• If you see an issue you also have, contribute with more information.
• And even better if it something you may know how to solve, please let people know.
• And especially, if you post a problem, then solve it, let people know what the solution was! (and not just that you solved it...)

I am rubbish in replying to emails, and the forums are read and answered by people whom know a lot more about Postfix than me. Questions sent to me directly may not be answered for a while or at all unfortunately.

12.5

Not to be rude, but in attempt to reduce the volume of emails I receive please consider the following:

• If you simply want advice please use the forum or Server Fault.
• My references and the people whom follow the forums and the postfix mailing lists are much more knowledgeable than me.
• People whom follow the forums and the postfix mailing lists are much more likely to reply than me.
• I am a firm believer in: Give a man a fish; you have fed him for today. Teach a man to fish; and you have fed him for a lifetime. (Playing far too much Civ was not all wasted..) Or rather my version: "Teach a man to fish, share the teaching, and you have fed many others for a lifetime”.

**So if you have any questions, problems with using this guide, or any other issues with this guide, please use the forum.** Then it is also an available archive for others to find solutions in the future.

• Any technical difference of opinion, please use the forum.
• Interested in Postfix, or got a technical query about it? Considered the postfix mailing lists.
• If you made / found an extension / alternative / translation to this tutorial, fantastic! Please let me know, and I’ll link to it. And shout at me, if I am slow in doing so. :)
• Any clear technical mistakes by me in this guide, then let me know, but perhaps discuss it first in the forums?
• **If you find any spelling mistakes or broken links, please let me know.** Even after 6-7 years since the first version people are still finding typos every week. So my initial versions must have contained only typos! :0
• Thank you messages are very appreciated however! Actually it makes my day. :)

(People whom buy a T-shirt of me makes my week.. ;p )

If you still want my help:

• If you still need advice you may hire me as a consultant via www.eray.co.uk.
• You may hire me to set up a complete email server for you via www.eray.co.uk.
• However if you:
  – are an individual or non commercial organisation
  – **have tried all the steps mentioned above: test section, FAQ, forums, mailing lists.** then you can contact me via flurdy.com/contact
10 years after the initial version of this document I think roughly 95% of the people that do contact me directly, have not bothered to test properly (command line first, not straight to GUI...) nor read the whole guide and common solutions & forums.

They just want me do to their work for them. For free.

The chances of me replying to their emails are miniscule, I much prefer to spend time with my daughter instead :) However many people contribute with really good stuff in the forums, send me typos(frequently..), or thank you notes. You guys are great!

12.6

Why your own mail server

Main reason: Because you can.

Other good reasons: Basically it leaves you in complete control, to expand, customize and tweak your mail server to your needs. You are not dependant on 3rd party providers, limited by their technology contraints or your budgets. With your own mail server you can add as many aliases, users and domain as you’d like, be as restrictive or open about security, virus, spam, file sizes etc as you prefer. And is it is well known, frequently updated, open source application stack, you can also trust the software you use.

Why I wrote this howto

When I set up my first email server I used a mix of other howtos on the net. And they were so helpfull that I though I would contribute back with my experience. And it has been useful as a recipe script for myself every time I need to install/update a server.

A less angelic reason is that back in 2003 I was setting up mail server for a few friends and collegues. Soon I was getting more request, and being a lazy programmer, I thought.. “Why don’t I write a howto and let them do it themselves...” Soon it was listed on postfix.org and I was getting thousends of hits and lots of emails. (blessing in disguise)

12.7

- Postfix howtos
- Kyle’s book
- John Locke on TechRepublic
- Hildebrandt’s book
- Hildebrandt’s website
- List-Petersen
- Genco Yilmaz
- Christop Haas
- Nenzel & Peet
- Peters
- Matthews
- Stepanov
- Andy “Besy”
• Meta Consultancy
• Postfix TLS
• Postfix main.cf doc
• saslauthd
• Bypassing amavisd
• Ubuntu Help: Squirrelmail

12.8

Please refer to the previous edition for a list of urls and suitable downloads. However most are unnecessary with decent package manager.

12.9 Difference between Ubuntu versions

I used to distinguish differences, which were available in a previous edition.

In the last few editions I have not. I base it on the latest LTS version, and test it with the minor releases in between.

12.10

Please refer to the previous edition for a complete list of downloads that are available. However all the required software are available as .deb packages direct from Ubuntu repositories.

12.11

Brief list of latest changes.

• 2013-01-10: Made Roundcube default webmail client.
• 2011-11-18: Added fail2ban.
• 2011-10-10: (Re)Added domain and DNS section
• 2010-06-09: Improved Google Apps integration. Added backup relay recipient lookup. Update phpmyadmin section.
• 2010-02-15: Redid SASL secure authentication section.
• 2009-12-16: Expanded test section with text from older editions and new babble.
• 2009-11-25: Bumped to edition 9! And added Roundcube as webmail client.
• 2009-11-11: Updated to work with 9.10 Karmic Koala.
• 2009-06-04: made basic server image available on ec2. Based canonical’s official ec2 ami.
• 2009-06-02: made clean server image available on ec2. Based canonical’s official ec2 ami.
• 2009-05-29: changed contact section.

• 2009-05-29: started 8th edition Used to refer to all changes, but got too long. A previous edition contains such a list.

12.12 Todo

• Spell check!

• Remove uid and guid

Please refer to the previous edition for some old todos....

12.13 FAQ

There is not yet an extensive FAQ.

But please, most of the frequent questions have been asked and answered in the forums. Most are also unnecessary as following the test section will have solved them.

Some question that frequently get sent to me, which first of all should have been asked in the forums and has been answered there many times, which then I tend to ignore are:

• Squirrelmail does not allow me to log in This is due to many things. Most are due to skipping too fast forward, ignoring test sections etc. Answers:
  – Does postfix work? No point trying to run before you can crawl. Send emails to recipients on your server, tail mail.log to see if everything is okay. Often mysql is not configured properly, check the mysql logs for activity.
  – Have they ever received an email? If not they can not log into squirrelmail as the email folders will not yet exist.
  – Does Courier work? If it doesn’t then you have still got some more setup to do.
  – If all above is okay, then it may be a problem with your Squirrelmail setup. Check empty spaces in squirrelmail mysql setup. More details in test section.

• Email folders do not exist Mentioned many times in this guide and forums. Answers:
  – Have they received an email? If not they can not log into squirrelmail as the email folders will not yet exist. When receiving their first email, postfix will create all the neccessary folders. If it does not your postfix setup is broken.
  – There is a program that creates the folders for you. I do not recommend it, as basically your postfix setup is broken if no folders are created, and you better fix it instead.

• SASL authentication does not work All lot of people have issues with SASL in certain setups. There are quite a few messages in the forum is regarding this. SASL works for me, but I can not tell my configuration apart from other people’s server where it does not. Workarounds or alternatives:
  – Do you need external SMTP & IMAP access? Or will webmail interface be sufficient for external dynamic ip clients? Then you do not need SASL. Restrict by IP and TLS is enough.
  – Do you need SASL at all? If you accept TLS only and restrict by IP addresses in mynetworks it may not be neccessary. Not an option with road warriors and too random dynamic IPs unfortunately.
  – Could POP-before-SMTP be an alternative?
– There are a few suggestions in the forum. Note however most people assume SASL is the problem when another factor that is the root cause.

• (!)file(1) utility (/usr/bin/file) FAILED: run_command: can’t fork: Cannot allocate memory at /usr/sbin/amavisd-new line 3077.

Running postfix on a micro instance on Amazon ec2? It only has 600Mb of ram and amavisd+clamav can grow too big over time. Restart these services, and even some times reboot...
13.1 mailman

13.1.1

Please note that the information on this page may be out of date. Check for the latest installation information on the Mailman wiki.

GNU Mailman works on most POSIX-based systems such as Unix, MacOSX, or GNU/Linux. It does not currently work on Windows. You must have a mail server that you can send messages to, and a web server that supports the CGI/1.1 API. Apache makes a fine choice for web server, and mail servers such as Postfix, Exim, Sendmail, and qmail should work just fine.

To install Mailman from source, you will need an ANSI C compiler to build Mailman’s security wrappers. The GNU C compiler gcc works well.

You must have the Python interpreter installed somewhere on your system. As of this writing, Python 2.4.4 is recommended, but see the wiki page above for the latest information.

Before installing Mailman, you need to prepare your system by adding certain users and groups. You will need to have root privileges to perform the steps in this section.

Mailman requires a unique user and group name which will own its files, and under which its processes will run. Mailman’s basic security is based on group ownership permissions, so it’s important to get this step right. Typically, you will add a new user and a new group, both called mailman. The mailman user must be a member of the mailman group. Mailman will be installed under the mailman user and group, with the set-group-id (setgid) bit enabled.

If these names are already in use, you can choose different user and group names, as long as you remember these when you run configure. If you choose a different unique user name, you will have to specify this with configure’s –with-username option, and if you choose a different group name, you will have to specify this with configure’s –with-groupname option.

On Linux systems, you can use the following commands to create these accounts. Check your system’s manual pages for details:
% groupadd mailman
% useradd -c "GNU Mailman" -s /no/shell -d /no/home -g mailman mailman

Typically, Mailman is installed into a single directory, which includes both the Mailman source code and the run-time list and archive data. It is possible to split the static program files from the variable data files and install them in separate directories. This section will describe the available options. The default is to install all of Mailman to /usr/local/mailman. You can change this base installation directory (referred to here as $prefix) by specifying the directory with the –prefix configure option. If you’re upgrading from a previous version of Mailman, you may want to use the –prefix option unless you move your mailing lists.

You cannot install Mailman on a filesystem that is mounted with the nosuid option. This will break Mailman, which relies on setgid programs for its security. If this describes your environment, simply install Mailman in a location that allows setgid programs.

Make sure the installation directory is set to group mailman (or whatever you’re going to specify with –with-groupname) and has the setgid bit set. You probably also want to guarantee that this directory is readable and executable by everyone. For example, these shell commands will accomplish this:

% cd $prefix
% chgrp mailman .
% chmod a+rx,g+ws .

The installation directory, $prefix, cannot be the same directory that the source tarball has been unpacked to and in which you run configure, but it can, if you wish, be a subdirectory, e.g., $prefix/src.

You are now ready to configure and install the Mailman software.

Before you can install Mailman, you must run configure to set various installation options your system might need.

Take special note of the –with-mail-gid and –with-cgi-gid options below. You will probably need to use these.

You should not be root while performing the steps in this section. Do them under your own login, or whatever account you typically use to install software. You do not need to do these steps as user mailman, but you could. However, make sure that the login used is a member of the mailman group as that that group has write permissions to the $prefix directory made in the previous step. You must also have permission to create a setgid file in the file system where it resides (NFS and other mounts can be configured to inhibit setgid settings).

If you’ve installed other GNU software, you should be familiar with the configure script. Usually you can just cd to the directory you unpacked the Mailman source tarball into, and run configure with no arguments:

% cd mailman-<version>
% ./configure
% make install

---

1 This is the default for Mailman 2.1. Earlier versions of Mailman installed everything under /home/mailman by default.
2 BSD users should see the 15.2 section for additional information.
mailman:

- **–prefix=dir**
  
  Standard GNU configure option which changes the base directory that Mailman is installed into. By default $prefix is /usr/local/mailman. This directory must already exist, and be set up as described in 2.2.

- **–exec-prefix=dir**
  
  Standard GNU configure option which lets you specify a different installation directory for architecture dependent binaries.

- **–with-var-prefix=dir**
  
  Store mutable data under dir instead of under the $prefix or $exec_prefix. Examples of such data include the list archives and list settings database.

- **–with-python=/path/to/python**
  
  Specify an alternative Python interpreter to use for the wrapper programs. The default is to use the interpreter found first on your shell’s $PATH.

- **–with-username=username-or-uid**
  
  Specify a different username than mailman. The value of this option can be an integer user id or a user name. Be sure your $prefix directory is owned by this user.

- **–with-groupname=groupname-or-gid**
  
  Specify a different groupname than mailman. The value of this option can be an integer group id or a group name. Be sure your $prefix directory is group-owned by this group.

- **–with-mail-gid=group-or-groups**
  
  Specify an alternative group for running scripts via the mail wrapper. group-or-groups can be a list of one or more integer group ids or symbolic group names. The first value in the list that resolves to an existing group is used. By default, the value is the list mailman, other, mail, and daemon.

  : This is highly system dependent and you must get this right, because the group id is compiled into the mail wrapper program for added security. On systems using sendmail, the sendmail.cf configuration file designates the group id of sendmail processes using the DefaultUser option. (If commented out, it still may be indicating the default...)

Check your mail server’s documentation and configuration files to find the right value for this switch.

- **–with-cgi-gid=group-or-groups**
  
  Specify an alternative group for running scripts via the CGI wrapper. group-or-groups can be a list of one or more integer group ids or symbolic group names. The first value in the list that resolves to an existing group is used. By default, the value is the list www, www-data, and nobody.

  : The proper value for this is dependent on your web server configuration. You must get this right, because the group id is compiled into the CGI wrapper program for added security, and no Mailman CGI scripts will run if this is incorrect.

If you’re using Apache, check the values for the Group option in your httpd.conf file.

- **–with-cgi-ext=extension**
  
  Specify an extension for cgi-bin programs. The CGI wrappers placed in $prefix/cgi-bin will have this extension (some web servers require an extension). extension must include the leading dot.

- **–with-mailhost=hostname**
Specify the fully qualified host name part for outgoing email. After the installation is complete, this value can be overridden in $prefix/Mailman/mm_cfg.py.

- --with-urlhost=hostname

Specify the fully qualified host name part of urls. After the installation is complete, this value can be overridden in $prefix/Mailman/mm_cfg.py.

- --with-gcc=no
  gcc, ., cc $PATH.

**configure, make, make install** Mailman.

After you’ve run make install, you should check that your installation has all the correct permissions and group ownerships by running the **check_perms** script. First change to the installation (i.e. $prefix) directory, then run the bin/check_perms program. Don’t try to run bin/check_perms from the source directory; it will only run from the installation directory.

If this reports no problems, then it’s very likely <wink> that your installation is set up correctly. If it reports problems, then you can either fix them manually, re-run the installation, or use bin/check_perms to fix the problems (probably the easiest solution):

- You need to become the user that did the installation, and that owns all the files in $prefix, or root.
- Run bin/check_perms -f
- !

Mailman, ., (o-x). However, the web server process must be able to follow the symbolic link in public directory, otherwise your public Pipermail archives will not work. To set this up, become root and run the following commands:

```
# cd <prefix>/archives
# chown <web-server-user> private
# chmod o-x private
```

You need to know what user your web server runs as. It may be www, apache, httpd or nobody, depending on your server’s configuration.

**web**

Mailman. To get everything running you need to hook Mailman up to both your web server and your mail system.

If you plan on running your mail and web servers on different machines, sharing Mailman installations via NFS, be sure that the clocks on those two machines are synchronized closely. You might take a look at the file Mailman/LockFile.py; the constant CLOCK_SLOP helps the locking mechanism compensate for clock skew in this type of environment.

This section describes some of the things you need to do to connect Mailman’s web interface to your web server. The instructions here are somewhat geared toward the Apache web server, so you should consult your web server documentation for details.
You must configure your web server to enable CGI script permission in the $prefix/cgi-bin to run CGI scripts. The line you should add might look something like the following, with the real absolute directory substituted for $prefix, of course:

```
Exec /mailman/* $prefix/cgi-bin/*
```

or:

```
ScriptAlias /mailman/ $prefix/cgi-bin/
```

: You want to be very sure that the user id under which your CGI scripts run is not in the mailman group you created above, otherwise private archives will be accessible to anyone.

Copy the Mailman, Python, and GNU logos to a location accessible to your web server. E.g. with Apache, you've usually got an icons directory that you can drop the images into. For example:

```
% cp $prefix/icons/*.{jpg,png} /path/to/apache/icons
```

You then want to add a line to your $prefix/Mailman/mm_cfg.py file which sets the base URL for the logos. For example:

```
IMAGE_LOGOS = '/images/'
```

The default value for IMAGE_LOGOS is /icons/. Read the comment in Defaults.py.in for details.

Configure your web server to point to the Pipermail public mailing list archives. For example, in Apache:

```
Alias /pipermail/ $varprefix/archives/public/
```

where $varprefix is usually $prefix unless you’ve used the –with-var-prefix option to configure. Also be sure to configure your web server to follow symbolic links in this directory, otherwise public Pipermail archives won’t be accessible. For Apache users, consult the FollowSymLinks option.

If you’re going to be supporting internationalized public archives, you will probably want to turn off any default charset directive for the Pipermail directory, otherwise your multilingual archive pages won’t show up correctly. Here’s an example for Apache, based on the standard installation directories:

```
<Directory "/usr/local/mailman/archives/public/">
  AddDefaultCharset Off
</Directory>
```

This section describes some of the things you need to do to connect Mailman’s email interface to your mail server. The instructions here are different for each mail server; if your mail server is not described in the following subsections, try to generalize from the existing documentation, and consider contributing documentation updates to the Mailman developers.

Under rare circumstances or due to mis-configuration, mail to the owner(s) of the ‘mailman’ site-list (see section 8) can bounce. In order to prevent a mail loop this mail is sent with envelope from mailman-loop which is normally aliased as

```
mailman-loop: $varprefix/data/owner-bounces.mbox
```

but which can be aliased to any, always deliverable, local address or file. If you are using the Postfix MTA integrated as described in section 6.1, this alias will be generated automatically. In all other cases, you should install this alias along with your normal system aliases.
Set Up Mail Server Documentation, 1.0

Postfix

Mailman should work pretty much out of the box with a standard Postfix installation. It has been tested with various Postfix versions up to and including Postfix 2.1.5.

In order to support Mailman’s optional VERP delivery, you will want to disable luser_relay (the default) and you will want to set recipient_delimiter for extended address semantics. You should comment out any luser_relay value in your main.cf and just go with the defaults. Also, add this to your main.cf file:

```plaintext
recipient_delimiter = +
```

Using “+” as the delimiter works well with the default values for VERP_FORMAT and VERP_REGEXP in Defaults.py.

When attempting to deliver a message to a non-existent local address, Postfix may return a 450 error code. Since this is a transient error code, Mailman will continue to attempt to deliver the message for DELIVERY_RETRY_PERIOD - 5 days by default. You might want to set Postfix up so that it returns permanent error codes for non-existent local users by adding the following to your main.cf file:

```plaintext
unknown_local_recipient_reject_code = 550
```

Finally, if you are using Postfix-style virtual domains, read the section on virtual domain support below.

You can integrate Postfix and Mailman such that when new lists are created, or lists are removed, Postfix’s alias database will be automatically updated. The following are the steps you need to take to make this work.

In the description below, we assume that you’ve installed Mailman in the default location, i.e. /usr/local/mailman. If that’s not the case, adjust the instructions according to your use of configure’s –prefix and –with-var-prefix options.

Note: If you are using virtual domains and you want Mailman to honor your virtual domains, read the 6.1 section below first! Add this to the bottom of the $prefix/Mailman/mm_cfg.py file:

```plaintext
MTA = 'Postfix'
```

The MTA variable names a module in the Mailman/MTA directory which contains the mail server-specific functions to be executed when a list is created or removed.

Look at the Defaults.py file for the variables POSTFIX_ALIAS_CMD and POSTFIX_MAP_CMD command. Make sure these point to your postalias and postmap programs respectively. Remember that if you need to make changes, do it in mm_cfg.py. Run the bin/genaliases script to initialize your aliases file.

```plaintext
% cd /usr/local/mailman
% bin/genaliases
```

Make sure that the owner of the data/aliases and data/aliases.db file is mailman, that the group owner for those files is mailman, or whatever user and group you used in the configure command, and that both files are group writable:

```plaintext
% su
% chown mailman:mailman data/aliases*
% chmod g+w data/aliases*
```

Hack your Postfix’s main.cf file to include the following path in your alias_maps variable:

```plaintext
/usr/local/mailman/data/aliases
```

Note that there should be no trailing .db. Do not include this in your alias_database variable. This is because you do not want Postfix’s newaliases command to modify Mailman’s aliases.db file, but you do want Postfix to consult aliases.db when looking for local addresses.

You probably want to use a hash: style database for this entry. Here’s an example:
alias_maps = hash:/etc/postfix/aliases,
hash:/usr/local/mailman/data/aliases

When you configure Mailman, use the --with-mail-gid=mailman switch; this will be the default if you configured
Mailman after adding the mailman owner. Because the owner of the aliases.db file is mailman, Postfix will execute
Mailman’s wrapper program as uid and gid mailman.

That’s it! One caveat: when you add or remove a list, the aliases.db file will updated, but it will not automatically run
postfix reload. This is because you need to be root to run this and suid-root scripts are not secure. The only effect of
this is that it will take about a minute for Postfix to notice the change to the aliases.db file and update its tables.

This section describes how to integrate Mailman with Postfix for automatic generation of Postfix virtual_alias_maps
for Mailman list addresses. Mailman’s support of virtual domains is limited in that list names must be globally unique
within a single Mailman instance, i.e., two lists may not have the same name even if they are in different domains.

Postfix 2.0 supports “virtual alias domains”, essentially what used to be called “Postfix-style virtual domains” in
earlier Postfix versions. To make virtual alias domains work with Mailman, you need to do some setup in both Postfix
and Mailman. Mailman will write all virtual alias mappings to a file called, by default, /usr/local/mailman/data/virtual-
mailman. It will also use postmap to create the virtual-mailman.db file that Postfix will actually use.

First, you need to set up the Postfix virtual alias domains as described in the Postfix documentation (see Postfix’s
virtual(5) manpage). Note that it’s your responsibility to include the virtual-alias.domain anything line as described
manpage (in recent Postfix this is not required if the domain is included in virtual_alias_domains in main.cf); Mailman
will not include this line in virtual-mailman. You are highly encouraged to make sure your virtual alias domains are
working properly before integrating with Mailman.

Next, add a path to Postfix’s virtual_alias_maps variable, pointing to the virtual-mailman file, e.g.:

```
virtual_alias_maps = <your normal virtual alias files>,
hash:/usr/local/mailman/data/virtual-mailman
```

assuming you’ve installed Mailman in the default location. If you’re using an older version of Postfix which doesn’t
have the virtual_alias_maps variable, use the virtual_maps variable instead.

Next, in your mm_cfg.py file, you will want to set the variable POSTFIX_STYLE_VIRTUAL_DOMAINS to the list
of virtual domains that Mailman should update. This may not be all of the virtual alias domains that your Postfix
installation supports! The values in this list will be matched against the host_name attribute of mailing lists objects,
and must be an exact match.

Here’s an example. Say that Postfix is configured to handle the virtual domains dom1.ain, dom2.ain, and dom3.ain,
and further that in your main.cf file you’ve got the following settings:

```
myhostname = mail.dom1.ain
mydomain = dom1.ain
mydestination = $myhostname, localhost.$mydomain
virtual_alias_maps =
    hash:/some/path/to/virtual-dom1,
    hash:/some/path/to/virtual-dom2,
    hash:/some/path/to/virtual-dom2
```

If in your virtual-dom1 file, you’ve got the following lines:

```
dom1.ain IGNORE
@dom1.ain @mail.dom1.ain
```

this tells Postfix to deliver anything addressed to dom1.ain to the same mailbox at mail.dom1.com, its default destina-

13.1. mailman
In this case you would not include dom1.ain in POSTFIX_STYLE_VIRTUAL_DOMAINS because otherwise Mailman will write entries for mailing lists in the dom1.ain domain as:

```
mylist@dom1.ain  mylist
mylist-request@dom1.ain  mylist-request
# and so on...
```

The more specific entries trump your more general entries, thus breaking the delivery of any dom1.ain mailing list.

However, you would include dom2.ain and dom3.ain in mm_cfg.py:

```
POSTFIX_STYLE_VIRTUAL_DOMAINS = ['dom2.ain', 'dom3.ain']
```

Now, any list that Mailman creates in either of those two domains, will have the correct entries written to /usr/local/mailman/data/virtual-mailman.

As above with the data/aliases* files, you want to make sure that both data/virtual-mailman and data/virtual-mailman.db are user and group owned by mailman.

Fil fil@rezo.net has an alternative approach based on virtual maps and regular expressions, as described at:

(French) http://listes.rezo.net/comment.php

(English) http://listes.rezo.net/how.php

This is a good (and simpler) alternative if you don’t mind exposing an additional hostname in the domain part of the addresses people will use to contact your list. I.e. if people should use mylist@lists.dom.ain instead of mylist@dom.ain.

Exim

Note: This section is derived from Nigel Metheringham’s ‘HOWTO - Using Exim and Mailman together’, which covers Mailman 2.0.x and Exim 3. It has been updated to cover Mailman 2.1 and Exim 4. The updated document is here: http://www.exim.org/howto/mailman21.html and is recommended over the information in the subsections below if you are using Exim 4. There is no Mailman configuration needed other than the standard options detailed in the Mailman install documentation. The Exim configuration is transparent to Mailman. The user and group settings for Mailman must match those in the config fragments given below.

```
Exim
```

The Exim configuration is built so that a list created within Mailman automatically appears to Exim without the need for defining any additional aliases.

The drawback of this configuration is that it will work poorly on systems supporting lists in several different mail domains. While Mailman handles virtual domains, it does not yet support having two distinct lists with the same name in different virtual domains, using the same Mailman installation. This will eventually change. (But see below for a variation on this scheme that should accommodate virtual domains better.)

The configuration file excerpts below are for use in an already functional Exim configuration, which accepts mail for the domain in which the list resides. If this domain is separate from the others handled by your Exim configuration, then you’ll need to:

add the list domain, “my.list.domain” to local_domains add a “domains=my.list.domain” option to the director (router) for the list (optional) exclude that domain from your other directors (routers) Note: The instructions in this document should work with either Exim 3 or Exim 4. In Exim 3, you must have a local_domains configuration setting; in Exim 4, you most likely have a local_domains domainlist. If you don’t, you probably know what you’re doing and can adjust accordingly. Similarly, in Exim 4 the concept of “directors” has disappeared - there are only routers now. So if you’re using Exim 4, whenever this document says “director”, read “router”. Whether you are using Exim 3 or Exim 4, you will need to add some macros to the main section of your Exim config file. You will also need to define one new transport. With Exim 3, you’ll need to add a new director; with Exim 4, a new router plays the same role.
Finally, the configuration supplied here should allow co-habiting Mailman 2.0 and 2.1 installations, with the proviso that you’ll probably want to use mm21 in place of mailman - e.g., MM21_HOME, mm21_transport, etc.

First, you need to add some macros to the top of your Exim config file. These just make the director (router) and transport below a bit cleaner. Obviously, you’ll need to edit these based on how you configured and installed Mailman.

```bash
# Home dir for your Mailman installation -- aka Mailman's prefix
# directory.
MAILMAN_HOME=/usr/local/mailman
MAILMAN_WRAP=MAILMAN_HOME/mail/mailman

# User and group for Mailman, should match your --with-mail-gid
# switch to Mailman's configure script.
MAILMAN_USER=mailman
MAILMAN_GROUP=mailman
```

**Exim 3** Add this to the transports section of your Exim config file, i.e. somewhere between the first and second “end” line:

```bash
mailman_transport:
  driver = pipe
  command = MAILMAN_WRAP \
    '${if def:local_part_suffix \n      {$sg{$local_part_suffix}{-\w+}{\+.}\(\d\)?}\(\$1\)} \n      \n    \$local_part
  current_directory = MAILMAN_HOME
  home_directory = MAILMAN_HOME
  user = MAILMAN_USER
  group = MAILMAN_GROUP
```

**Exim 3** If you’re using Exim 3, you’ll need to add the following director to your config file (directors go between the second and third “end” lines). Also, don’t forget that order matters - e.g. you can make Mailman lists take precedence over system aliases by putting this director in front of your aliasfile director, or vice-versa.

```bash
# Handle all addresses related to a list 'foo': the posting address.
# Automatically detects list existence by looking
# for lists/$local_part/config.pck under MAILMAN_HOME.
mailman_director:
  driver = smartuser
  require_files = MAILMAN_HOME/lists/$local_part/config.pck
  suffix_optional
  suffix = -bounces : -bounces+* :
  -confirm+* : -join : -leave :
  -owner : -request : -admin
  transport = mailman_transport
```

**Exim 4** In Exim 4, there’s no such thing as directors - you need to add a new router instead. Also, the canonical order of the configuration file was changed so routers come before transports, so the router for Exim 4 comes first here. Put this router somewhere after the “begin routers” line of your config file, and remember that order matters.

```bash
mailman_router:
  driver = accept
  require_files = MAILMAN_HOME/lists/$local_part/config.pck
```

13.1. mailman
local_part_suffix_optional
local_part_suffix = -admin : -bounces : -bounces+* : \
     -confirm : -confirm+* : \
     -join : -leave : \
     -owner : -request : \
     -subscribe : -unsubscribe
transport = mailman_transport

Exim 4  The transport for Exim 4 is the same as for Exim 3 (see 6.2; just copy the transport given above to somewhere under the “begin transports” line of your Exim config file.

Exim should be configured to allow reasonable volume - e.g. don’t set max_recipients down to a silly value - and with normal degrees of security - specifically, be sure to allow relaying from 127.0.0.1, but pretty much nothing else. Parallel deliveries and other tweaks can also be used if you like; experiment with your setup to see what works. Delay warning messages should be switched off or configured to only happen for non-list mail, unless you like receiving tons of mail when some random host is down.

Mailman will send as many MAIL FROM/RCPT TO as it needs. It may result in more than 10 or 100 messages sent in one connection, which will exceed the default value of Exim’s smtp_accept_queue_per_connection value. This is bad because it will cause Exim to switch into queue mode and severely delay delivery of your list messages. The way to fix this is to set Mailman’s SMTP_MAX_SESSIONS_PER_CONNECTION (in $prefix/Mailman/mm_cfg.py) to a smaller value than Exim’s smtp_accept_queue_per_connection. Mailman should ignore Exim delay warning messages, even though Exim should never send this to list messages. Mailman 2.1’s general bounce detection and VERP support should greatly improve the bounce detector’s hit rates. List existence is determined by the existence of a config.pck file for a list. If you delete lists by foul means, be aware of this. If you are getting Exim or Mailman complaining about user ids when you send mail to a list, check that the MAILMAN_USER and MAILMAN_GROUP match those of Mailman itself (i.e. what were used in the configure script). Also make sure you do not have aliases in the main alias file for the list.

Exim’s receiver verification feature is very useful - it lets Exim reject unrouteable addresses at SMTP time. However, this is most useful for externally-originating mail that is addressed to mail in one of your local domains. For Mailman list traffic, mail originates on your server, and is addressed to random external domains that are not under your control. Furthermore, each message is addressed to many recipients - up to 500 if you use Mailman’s default configuration and don’t tweak SMTP_MAX_RCPTS.

Doing receiver verification on Mailman list traffic is a recipe for trouble. In particular, Exim will attempt to route every recipient addresses in outgoing Mailman list posts. Even though this requires nothing more than a few DNS lookups for each address, it can still introduce significant delays. Therefore, you should disable recipient verification for Mailman traffic.

Under Exim 3, put this in your main configuration section:
receiver_verify_hosts = !127.0.0.1

Under Exim 4, this is probably already taken care of for you by the default recipient verification ACL statement (in the RCPT TO ACL):
accept domains = +local_domains
endpass
message = unknown user
verify = recipient

which only does recipient verification on addresses in your domain. (That’s not exactly the same as doing recipient verification only on messages coming from non-127.0.0.1 hosts, but it should do the trick for Mailman.)
SMTP Callback  Exim’s SMTP callback feature is an even more powerful way to detect bogus sender addresses than normal sender verification. Unfortunately, lots of servers send bounce messages with a bogus address in the header, and there are plenty that send bounces with bogus envelope senders (even though they’re supposed to just use an empty envelope sender for bounces).

In order to ensure that Mailman can disable/remove bouncing addresses, you generally want to receive bounces for Mailman lists, even if those bounces are themselves not bounceable. Thus, you might want to disable SMTP callback on bounce messages.

With Exim 4, you can accomplish this using something like the following in your RCPT TO ACL:

```plaintext
# Accept bounces to lists even if callbacks or other checks would fail
warn  message = X-WhitelistedRCPT-nohdrfromcallback: Yes
  condition = \n  ${if and {{match{$local_part}{(.*)-bounces\+.*}} \n    {exists {MAILMAN_HOME/lists/$1/config.pck}}} \n    {yes}{no}}

accept condition = \n  ${if and {{match{$local_part}{(.*)-bounces\+.*}} \n    {exists {MAILMAN_HOME/lists/$1/config.pck}}} \n    {yes}{no}}

# Now, check sender address with SMTP callback.
deny !verify = sender/callout=90s
```

If you also do SMTP callbacks on header addresses, you’ll want something like this in your DATA ACL:

```plaintext
deny  !condition = $header_X-WhitelistedRCPT-nohdrfromcallback:
  !verify = header_sender/callout=90s
```

EximMailmanVERP  VERP will send one email, with a separate envelope sender (return path), for each of your subscribers - read the information in $prefix/Mailman/Defaults.py for the options that start with VERP. In a nutshell, all you need to do to enable VERP with Exim is to add these lines to $prefix/Mailman/mm_cfg.py:

```plaintext
VERP_PASSWORD_REMINDERS = Yes
VERP_PERSONALIZED_DELIVERIES = Yes
VERP_DELIVERY_INTERVAL = Yes
VERP_CONFIRMATIONS = Yes
```

(The director (router) above is smart enough to deal with VERP bounces.)

One approach to handling virtual domains is to use a separate Mailman installation for each virtual domain. Currently, this is the only way to have lists with the same name in different virtual domains handled by the same machine.

In this case, the MAILMAN_HOME and MAILMAN_WRAP macros are useless - you can remove them. Change your director (router) to something like this:

```plaintext
require_files = /virtual/${domain}/mailman/lists/${lc:$local_part}/config.pck
```

and change your transport like this:

```plaintext
command = /virtual/${domain}/mailman/mail/mailman \n  ${if def:local_part_suffix \n    ({sg{$local_part_suffix}{-(\w+)(\+.*)?}{\$1}}) \n    {post}} \n  $local_part
```
This is how a set of address tests for the Exim lists look on a working system. The list in question is `quixote-users@mems-exchange.org`, and these commands were run on the mems-exchange.org mail server (“% ” indicates the Unix shell prompt):

% exim -bt quixote-users
quixote-users@mems-exchange.org
  router = mailman_main_router, transport = mailman_transport

% exim -bt quixote-users-request
quixote-users-request@mems-exchange.org
  router = mailman_router, transport = mailman_transport

% exim -bt quixote-users-bounces
quixote-users-bounces@mems-exchange.org
  router = mailman_router, transport = mailman_transport

% exim -bt quixote-users-bounces+luser=example.com
quixote-users-bounces+luser=example.com@mems-exchange.org
  router = mailman_router, transport = mailman_transport

If your exim -bt output looks something like this, that’s a start: at least it means Exim will pass the right messages to the right Mailman commands. It by no means guarantees that your Exim/Mailman installation is functioning perfectly, though!

Originally written by Nigel Metheringham postmaster@exim.org. Updated by Marc Merlin marc_soft@merlins.org for Mailman 2.1, Exim 4. Overhauled/reformatted/clarified/simplified by Greg Ward gward@python.net.

### Sendmail

| You may be tempted to set the DELIVERY_MODULE configuration variable in mm_cfg.py to ‘Sendmail’ when using the Sendmail mail server. Don’t. The Sendmail.py module is misnamed - it’s really a command line based message handoff scheme as opposed to the SMTP scheme used in SMTPDirect.py (the default). Sendmail.py has known security holes and is provided as a proof-of-concept only . If you are having problems using SMTPDirect.py fix those instead of using Sendmail.py, or you may open your system up to security exploits. |

---

| In fact, in later versions of Mailman, this module is explicitly sabotaged. You have to know what you’re doing in order to re-enable it. |

| Sendmail “smrsh” Many newer versions of Sendmail come with a restricted execution utility called “smrsh”, which limits the executables that Sendmail will allow to be used as mail programs. You need to explicitly allow Mailman’s wrapper program to be used with smrsh or Mailman will not work. If mail is not getting delivered to Mailman’s wrapper program and you’re getting an “operating system error” in your mail syslog, this could be your problem. |

| One good way of enabling this is: |

| Find out where your Sendmail executes its smrsh wrapper |

% grep smrsh /etc/mail/sendmail.cf
Figure out where smrsh expects symlinks for allowable mail programs. At the very beginning of the following output you will see a full path to some directory, e.g. /var/adm/sm.bin or similar:

```
% strings $path_to_smrsh | less
```

cd into /var/adm/sm.bin, or where ever it happens to reside on your system - alternatives include /etc/smrsh, /var/smrsh and /usr/local/smrsh.

```
% cd /var/adm/sm.bin
```

Create a symbolic link to Mailman’s wrapper program:

```
% ln -s /usr/local/mailman/mail/mailman mailman
```

David Champion has contributed a recipe for more closely integrating Sendmail and Mailman, such that Sendmail will automatically recognize and deliver to new mailing lists as they are created, without having to manually edit alias tables.

In the contrib directory of Mailman’s source distribution, you will find four files:

- mm-handler.readme - an explanation of how to set everything up
- mm-handler - the mail delivery agent (MDA)
- mailman.mc - a toy configuration file sample
- virtusertable - a sample for RFC 2142 address exceptions

One of the surest performance killers for Sendmail users is when Sendmail is configured to synchronously verify the recipient’s host via DNS. If it does this for messages posted to it from Mailman, you will get horrible performance. Since Mailman usually connects via localhost (i.e. 127.0.0.1) to the SMTP port of Sendmail, you should be sure to configure Sendmail to not do DNS verification synchronously for localhost connections.

**Qmail**

There are some issues that users of the qmail mail transport agent have encountered. None of the core maintainers use qmail, so all of this information has been contributed by the Mailman user community, especially Martin Preishuber and Christian Tismer, with notes by Balazs Nagy (BN) and Norbert Bollow (NB).

You might need to set the mail-gid user to either qmail, mailman, or nofiles by using the `--with-mail-gid` configure option. BN: it highly depends on your mail storing policy. For example if you use the simple `~alias/.qmail-*` files, you can use `id -g alias`. But if you use `/var/qmail/users`, the specified mail gid can be used.

If you are going to be directing virtual domains directly to the mailman user (using `virtualdomains` on a list-only domain, for example), you will have to use `--with-mail-gid` gid of mailman user’s group. This is incompatible with having list aliases in `~alias`, unless that alias simply forwards to mailman-listname*.

If there is a user mailman on your system, the alias mailman-owner will work only in `~mailman`. You have to do a touch `~mailman-owner` in `~mailman` directory to create this alias. NB: An alternative, IMHO better solution is to chown root `~mailman`, that will stop qmail from considering mailman to be a user to whom mail can be delivered. (See `man 8 qmail-getpw`.)

In a related issue, if you have any users with the same name as one of your mailing lists, you will have problems if list names contain the identical listname.

```
```

where in this case the listname is e.g. zooper-users.

NB: Alternatively, you could host the lists on a virtual domain, and use the `/var/qmail/control/virtualdomains` file to put the mailman user in charge of this virtual domain.

BN: If inbound messages are delivered by another user than mailman, it’s necessary to allow it to access `~mailman`. Be sure that `~mailman` has group writing access and setgid bit is set. Then put the delivering user to mailman group, and
you can deny access to ~mailman to others. Be sure that you can do the same with the WWW service. By the way the best thing is to make a virtual mail server to handle all of the mail. NB: E.g. make an additional ‘A’ DNS record for the virtual mailserver pointing to your IP address, add the line lists.kva.hu:mailman to /var/qmail/control/virtualdomains and a lists.kva.hu line to /var/qmail/control/rcpthosts file. Don’t forget to HUP the qmail-send after modifying ‘virtualdomains’! Then every mail to lists.kva.hu will arrive to mail.kva.hu’s mailman user.

Then make your aliases:

```
.qmail    => mailman@...’s letters
.qmail-owner => mailman-owner’s letters
```

For list aliases, you can either create them manually:

```
.qmail-list => posts to the ‘list’ list .qmail-list-admin => posts to the ‘list’s owner .qmail-list-request => requests to ‘list’ etc
```

or for automatic list alias handling (when using the lists.kva.hu virtual as above), see contrib/qmail-to-mailman.py in the Mailman source distribution. Modify the ~mailman/qmail-default to include:

```
|preline /path/to/python /path/to/qmail-to-mailman.py
```

and new lists will automatically be picked up.

You have to make sure that the localhost can relay. If you start qmail via inetd and tcpenv, you need some line the following in your /etc/hosts.allow file:

```
tcp-env: 127. 10.205.200. : setenv RELAYCLIENT
```

where 10.205.200. is your IP address block. If you use tcpserver, then you need something like the following in your /etc/tcp.smtp file:

```
10.205.200.:allow,RELAYCLIENT=""
127.:allow,RELAYCLIENT=""
```

BN: Bigger /var/qmail/control/concurrencyremote values work better sending outbound messages, within reason. Unless you know your system can handle it (many if not most cannot) this should not be set to a value greater than 120. More information about setting up qmail and relaying can be found in the qmail documentation. BN: Last but not least, here’s a little script to generate aliases to your lists (if for some reason you can/will not have them automatically picked up using contrib/qmail-to-mailman.py):

This script is for the Mailman 2.0 series:

```
#!/bin/sh if [ $# = 1 ]; then
    i=$1 echo Making links to $i in the current directory... echo “|preline /home/mailman/mail/mailman post $i” > .qmail-$i echo “|preline /home/mailman/mail/mailman mailowner $i” > .qmail-$i-admin echo “|preline /home/mailman/mail/mailman mailowner $i” > .qmail-$i-owner echo “|preline /home/mailman/mail/mailman mailowner $i” > .qmail-owner-$i echo “|preline /home/mailman/mail/mailman mailcmd $i” > .qmail-$i-request
fi
```

Note: This is for a new Mailman 2.1 installation. Users upgrading from Mailman 2.0 would most likely change /usr/local/mailman to /home/mailman. If in doubt, refer to the –prefix option passed to configure during compile time.

```
#!/bin/sh if [ $# = 1 ]; then
    i=$1 echo Making links to $i in the current directory... echo “|preline /usr/local/mailman/mail/mailman post $i” > .qmail-$i echo “|preline /usr/local/mailman/mail/mailman admin $i” > .qmail-$i-admin echo “|preline /usr/local/mailman/mail/mailman bounces $i” > .qmail-$i-bounces
    # The following line is for VERP # echo “|preline /usr/local/mailman/mail/mailman
```
VERP  You will note in the alias generating script for 2.1 above, there is a line for VERP that has been commented out. If you are interested in VERP there are two options. The first option is to allow Mailman to do the VERP formatting. To activate this, uncomment that line and add the following lines to your mm_cfg.py file:

```
VERP_FORMAT = '%(bounces)s-+%(mailbox)s=%(host)s'
VERP_REGEXP = r'^(?P<bounces>.*?)-+(?P<mailbox>[^=]+)=(?P<host>[^@]+)@.*$'
```

The second option is a patch on SourceForge located at:

http://sourceforge.net/tracker/?func=detail&atid=300103&aid=645513&group_id=103

This patch currently needs more testing and might best be suitable for developers or people well familiar with qmail. Having said that, this patch is the more qmail-friendly approach resulting in large performance gains.

As mentioned in the 6.4 section for a virtual mail server, a patch under testing is located at:


Again, this patch is for people familiar with their qmail installation.

You might be interested in some information on modifying footers that Norbert Bollow has written about Mailman and qmail, available here:

http://mailman.cis.to/qmail-verh/

Mailman has a large number of site-wide configuration options which you should now review and change according to your needs. Some of the options control how Mailman interacts with your environment, and other options select defaults for newly created lists. There are system tuning parameters and integration options.

The full set of site-wide defaults lives in the $prefix/Mailman/Defaults.py file, however you should never modify this file! Instead, change the mm_cfg.py file in that same directory. You only need to add values to mm_cfg.py that are different than the defaults in Defaults.py, and future Mailman upgrades are guaranteed never to touch your mm_cfg.py file.

The Defaults.py file is documented extensively, so the options are not described here. The Defaults.py and mm_cfg.py are both Python files so valid Python syntax must be maintained or your Mailman installation will break.

Do not change the HOME_DIR or MAILMAN_DIR variables. These are set automatically by the configure script, and you will break your Mailman installation if you change these.

You should make any changes to mm_cfg.py using the account you installed Mailman under in the 3 section.

---

3 In general, changing the list defaults described in this section will not affect any already created lists. To make changes after a list has been created, use the web interface or the command line scripts, such as bin/withlist and bin/config_list.
After you have completed the integration of Mailman and your mail server, you need to create a "site-wide" mailing list. This is the one that password reminders will appear to come from, and it is required for proper Mailman operation. Usually this should be a list called mailman, but if you need to change this, be sure to change the MAILMAN_SITE_LIST variable in mm_cfg.py. You can create the site list with this command, following the prompts:

% bin/newlist mailman

Now configure your site list. There is a convenient template for a generic site list in the installation directory, under data/sitelist.cfg which can help you with this. You should review the configuration options in the template, but note that any options not named in the sitelist.cfg file won’t be changed.

The template can be applied to your site list by running:

% bin/config_list -i data/sitelist.cfg mailman

After applying the sitelist.cfg options, be sure you review the site list’s configuration via the admin pages.

You should also subscribe yourself to the site list.

cron

Several Mailman features occur on a regular schedule, so you must set up cron to run the right programs at the right time.

If your version of crontab supports the -u option, you must be root to do this next step. Add $prefix/cron/crontab.in as a crontab entry by executing these commands:

% cd $prefix/cron
% crontab -u mailman crontab.in

If you used the –with-username option, use that user name instead of mailman for the -u argument value. If your crontab does not support the -u option, try these commands:

% cd $prefix/cron
% su - mailman
% crontab crontab.in

: If you accepted the defaults for the –with-username option and for the name of the site list, and one of the cron jobs ever encounters an error, the cron daemon will mail the error output to the ‘mailman’ user and it will most likely be delivered to the ‘mailman’ site list and possibly not be accepted. For this reason it is a good idea to insert MAILTO=user@example.com

or

MAILTO=mailman-owner

at the beginning of crontab.in before installing it to cause this output to be mailed to a real user or to the owner of the site list or to configure the site list (see section 8) to accept this mail.

Mailman qrunner

Mailman depends on a process called the “qrunner” to delivery all email messages it sees. You must start the qrunner by executing the following command from the $prefix directory:

---

\(^4\) Note that if you’re upgrading from a previous version of Mailman, you’ll want to install the new crontab, but be careful if you’re running multiple Mailman installations on your site! Changing the crontab could mess with other parallel Mailman installations.
You probably want to start Mailman every time you reboot your system. Exactly how to do this depends on your operating system. If your OS supports the chkconfig command (e.g. RedHat and Mandrake Linuxes) you can do the following (as root, from the Mailman install directory):

% cp scripts/mailman /etc/init.d/mailman
% chkconfig --add mailman

Note that /etc/init.d may be /etc/rc.d/init.d on some systems.

On Gentoo Linux, you can do the following:

% cp scripts/mailman /etc/init.d/mailman
% rc-update add mailman default

On Debian, you probably want to use:

% update-rc.d mailman defaults

For Unixes that don’t support chkconfig, you might try the following set of commands:

% cp scripts/mailman /etc/init.d/mailman
% cp misc/mailman /etc/init.d
% cd /etc/rc.d/rc0.d
% ln -s ../init.d/mailman K12mailman
% cd ../rc1.d
% ln -s ../init.d/mailman K12mailman
% cd ../rc2.d
% ln -s ../init.d/mailman S98mailman
% cd ../rc3.d
% ln -s ../init.d/mailman S98mailman
% cd ../rc4.d
% ln -s ../init.d/mailman S98mailman
% cd ../rc5.d
% ln -s ../init.d/mailman S98mailman
% cd ../rc6.d
% ln -s ../init.d/mailman K12mailman

There are two site-wide passwords that you can create from the command line, using the bin/mmsitepass script. The first is the “site password” which can be used anywhere a password is required in the system. The site password will get you into the administration page for any list, and it can be used to log in as any user. Think root for a Unix system, so pick this password wisely!

The second password is a site-wide “list creator” password. You can use this to delegate the ability to create new mailing lists without providing all the privileges of the site password. Of course, the owner of the site password can also create new mailing lists, but the list creator password is limited to just that special role.
To set the site password, use this command:

% $prefix/bin/mmsitepass <your-site-password>

To set the list creator password, use this command:

% $prefix/bin/mmsitepass -c <list-creator-password>

It is okay not to set a list creator password, but you probably do want a site password.

For more detailed information about using Mailman, including creating and configuring mailing lists, see the Mailman List Administration Manual. These instructions provide a quick guide to creating your first mailing list via the web interface:

Start by visiting the url http://my.dom.ain/mailman/create.

Fill out the form as described in the on-screen instructions, and in the “List creator’s password” field, type the password you entered in section 7. Type your own email address for the “Initial list owner address”, and select “Yes” to notify the list administrator.

Click on the “Create List” button.

Check your email for a message from Mailman informing you that your new mailing list was created.

Now visit the list’s administration page, either by following the link on the confirmation web page or clicking on the link from the email Mailman just sent you. Typically the url will be something like http://my.dom.ain/mailman/admin/mylist.

Type in the list’s password and click on “Let me in...”.

Click on “Membership Management” and then on “Mass Subscription”.

Enter your email address in the big text field, and click on “Submit Your Changes”.

Now go to your email and send a message to mylist@my.dom.ain. Within a minute or two you should see your message reflected back to you via Mailman.

Congratulations! You’ve just set up and tested your first Mailman mailing list. If you had any problems along the way, please see the 14 section.

If you encounter problems with running Mailman, first check the question and answer section below. If your problem is not covered there, check the online help, including the FAQ and the community FAQ wiki.

Also check for errors in your syslog files, your mail and web server log files and in Mailman’s $prefix/logs/error file. If you’re still having problems, you should send a message to the mailman-users@python.org mailing list 5; see http://mail.python.org/mailman/listinfo/mailman-users for more information.

Be sure to including information on your operating system, which version of Python you’re using, and which version of Mailman you’re installing.

Here is a list of some common questions and answers:

: All Mailman web pages give a 404 File not found error.

: Your web server has not been set up properly for handling Mailman’s CGI programs. Make sure you have:

5 You must subscribe to this mailing list in order to post to it, but the mailing list’s archives are publicly visible.
configured the web server to give permissions to /prefix/cgi-bin restarted the web server properly. Consult your web server's documentation for instructions on how to do check these issues.

: All Mailman web pages give an “Internal Server Error”.

: The likely problem is that you are using the wrong user or group for the CGI scripts. Check your web server's log files. If you see a line like

    Attempt to exec script with invalid gid 51, expected 99

you will need to reinstall Mailman, specifying the proper CGI group id, as described in the 3 section.

: I send mail to the list, and get back mail saying the list is not found!

: You probably didn’t add the necessary aliases to the system alias database, or you didn’t properly integrate Mailman with your mail server. Perhaps you didn’t update the alias database, or your system requires you to run newaliases explicitly. Refer to your server specific instructions in the 6 section.

: I send mail to the list, and get back mail saying, “unknown mailer error”.

you will need to reinstall Mailman, specifying the proper mail group id as described in the 3 section.

: I use Postfix as my mail server and the mail wrapper programs are logging complaints about the wrong GID.

: I use Sendmail as my mail server, and when I send mail to the list, I get back mail saying, “sh: mailman not available for sendmail programs”.

: Your system uses the Sendmail restricted shell (smrsh). You need to configure smrsh by creating a symbolic link from the mail wrapper ($prefix/mail/mailman) to the directory identifying executables allowed to run under smrsh.

Some common names for this directory are /var/admin/sm.bin, /usr/admin/sm.bin or /etc/smrsh.

Note that on Debian Linux, the system makes /usr/lib/sm.bin, which is wrong, you will need to create the directory /usr/admin/sm.bin and add the link there. Note further any aliases newaliases spits out will need to be adjusted to point to the secure link to the wrapper.

: I messed up when I called configure. How do I clean things up and re-install?

: 

    % make clean
    % ./configure --with-the-right-options
    % make install

Generally, Mailman runs on any POSIX-based system, such as Solaris, the various BSD variants, Linux systems, MacOSX, and other generic Unix systems. It doesn’t run on Windows. For the most part, the generic instructions given in this document should be sufficient to get Mailman working on any supported platform. Some operating systems have additional recommended installation or configuration instructions.

GNU/Linux

Linux seems to be the most popular platform for running Mailman. Here are some hints on getting Mailman to run on Linux:

If you are getting errors with hard link creations and/or you are using a special secure kernel (secure-linux/openwall/grsecurity), see the file contrib/README.check_perms_grsecurity in the Mailman source distribution.

Note that if you are using Linux Mandrake in secure mode, you are probably concerned by this.
Apparently Mandrake 9.0 changed the permissions on gcc, so if you build as the mailman user, you need to be sure mailman is in the cctools group.

If you installed Python from your Linux distribution’s package manager (e.g. .rpms for Redhat-derived systems or .deb for Debian), you must install the “development” package of Python, or you may not get everything you need.

For example, using Python 2.2 on Debian, you will need to install the python2.2-dev package. On Redhat, you probably need the python2-devel package.

If you install Python from source, you should be fine.

One symptom of this problem, although for unknown reasons, is that you might get an error such as this during your install:

```
Traceback (most recent call last):
  File "bin/update", line 44, in ?
    import paths
ImportError: No module named paths
make: *** [update] Error 1
```

If this happens, install the Python development package and try configure and make install again. Or install the latest version of Python from source, available from http://www.python.org.

This problem can manifest itself in other Linux distributions in different ways, although usually it appears as ImportError.

**BSD**

Vivek Khera writes that some BSDs do nightly security scans for setuid file changes. setgid directories also come up on the scan when they change. Also, the setgid bit is not necessary on BSD systems because group ownership is automatically inherited on files created in directories. On other Unixes, this only happens when the directory has the setgid bit turned on.

To install without turning on the setgid bit on directories, simply pass in the DIRSETGID variable to make, after you’ve run configure:

```
% make DIRSETGID=: install
```

This disables the chmod g+s command on installed directories.

**MacOSX**

Many people run Mailman on MacOSX. Here are some pointers that have been collected on getting Mailman to run on MacOSX.

Jaguar (MacOSX 10.2) comes with Python 2.2. While this isn’t the very latest stable version of Python, it ought to be sufficient to run Mailman 2.1.

David B. O’Donnell has a web page describing his configuration of Mailman 2.0.13 and Postfix on MacOSX Server. http://www.afp548.com/Articles/mail/python-mailman.html


Panther server (MacOSX 10.3) comes with Mailman; Your operating system should contain documentation that will help you, and Apple has a tech document about a problem you might encounter running Mailman on Mac OS X Server 10.3:
http://docs.info.apple.com/article.html?artnum=107889

Terry Allen provides the following detailed instructions on running Mailman on the ‘client’ version of OSX, or in earlier versions of OSX:

Mac OSX 10.3 and onwards has the basics for a successful Mailman installation. Users of earlier versions of Mac OSX contains Sendmail and those users should look at the Sendmail installation section for tips. You should follow the basic installation steps as described earlier in this manual, substituting as appropriate, the steps outlined in this section.

By default, Mac OSX 10.3 ‘client’ version does not have a fully functional version of Postfix. Setting up a working MTA such as Postfix is beyond the scope of this guide and you should refer to http://www.postfix.org for tips on getting Postfix running. An easy way to set Postfix up is to install and run Postfix Enabler, a stand-alone tool for configuring Postfix on Mac OSX, available from http://www.roadstead.com/weblog/Tutorials/PostfixEnabler.html.

Likewise, Mac OSX ‘client’ version from 10.1 onwards includes a working Apache webservice. This is switched on using the System Preferences control panel under the ‘Sharing tab’. A useful tool for configuring the Apache on Mac OSX is Webmin, which can be obtained from http://www.webmin.com.

Webmin can also perform configuration for other system tasks, including Postfix, adding jobs to your crontab, adding user and groups, plus adding startup and shutdown jobs.

In a stock installation of OSX, the requirement for Mailman is to have Python installed. Python is not installed by default, so it is advised that you install the developer’s tools package, which may have been provided with your system. It can also be downloaded from the Apple developer site at http://connect.apple.com. Not only is the developer tools package an essential requirement for installing Mailman, but it will come in handy at a later date should you need other tools. The developer’s tools are also know by the name XCode tools.

As a minimum, the Python version should be 2.2, but 2.3 is recommended.

If you wish to add a user and group using the command line in OSX instead of via Webmin or another GUI interface, open your terminal application and follow the commands as indicated below - do not type the comments following the “#” since they are just notes:

    sudo tcsh
    niutil -create / /users/mailman
    niutil -createprop / /users/mailman name mailman
    # Note that xxx is a free user ID number on your system
    niutil -createprop / /users/mailman uid xxx
    niutil -createprop / /users/mailman home /usr/local/mailman
    mkdir -p /usr/local/mailman
    niutil -createprop / /users/mailman shell /bin/tcsh
    passwd mailman
    # To prevent malicious hacking, supply a secure password here
    niutil -create / /groups/mailman
    niutil -createprop / /groups/mailman name mailman
    # Note that xxx is a free group ID number on your system
    niutil -createprop / /groups/mailman gid xxx
    niutil -createprop / /groups/mailman passwd ‘*’
    niutil -createprop / /groups/mailman users ‘mailman’
    chown mailman:mailman /usr/local/mailman
    cd /usr/local/mailman
    chmod a+rx,g+ws .
    exit
    su mailman

For setting up Apache on OSX to handle Mailman, the steps are almost identical and the configuration file on a stock Mac OSX Client version is stored in the nearly standard location of /etc/httpd/httpd.conf.

The AFP548.com site has a time-saving automated startup item creator for Mailman, which can be found at http://www.afp548.com/Software/MailmanStartup.tar.gz

13.1. mailman
To install it, copy it into your `/Library/StartupItems` directory. As the root or superuser, from the terminal, enter the following:

```
gunzip MailmanStartup.tar.gz

tar xvf MailmanStartup.tar
```

It will create the startup item for you so that when you reboot, Mailman will start up.

---

13.1.2

This document is intended to help the members of a Mailman 2.1 mailing list learn to use the features available to them. It covers the use of the web and email interfaces for subscribing and unsubscribing, changing member options, getting password reminders and other subscriber-level tasks. It also answers some common questions of interest to Mailman list members.

Information for list and site administrators is provided in other documents.

This document need not be read in order. If you are simply looking for an answer to a specific question, jump to the appropriate place and references to other sections will be provided if necessary or potentially helpful.

Note

For the purposes of this document, we assume that the reader is familiar with common terms related to email (eg: Subject line, body of the message) and web sites (eg: drop-down box, button) or can look them up. We also assume that the reader can already use his or her email program and web browser well enough that instructions such as “send email to this address” or “visit this web page” or “fill in the form provided” are clear. If you are not familiar with these actions, you may want to consult other documentation to learn how to do these things with your particular setup.

Sections of this document have been borrowed from the List Administrator Manual found in Mailman CVS, which was written by Barry A. Warsaw, and from the in-line help for Mailman 2.1.

The rest of this manual has been written by Terri Oda, and (hopefully soon) edited and added to by members of the mailman community via the mailman wiki at [http://wiki.list.org](http://wiki.list.org). Please see the wiki changelogs for more information.

Terri has been maintaining mailing lists since the year she attained voting age in Canada, although the two are not related. She currently oversees the mailing lists at Linuxchix.org, as well as several smaller servers. In the world outside of list administration, Terri is doing work with an artificial life spam detector, and is actually more of a programmer than technical writer.

Proofreading thanks go to Margaret McCarthy and Jason Walton.
Please feel free to use this document in any way that will help you with your installation of Mailman. You are expressly permitted to distribute copies to your users and make modified copies that reflect your specific setup (for example, replacing all the instances of WEBSERVER and DOMAIN with the appropriate actual values). You can also get printable versions of this document in HTML or printable versions of this document in PDF (currently requires a wiki login, which you can create for free).

It is highly recommended that you provide a link to the latest version of this document, which can be found at

http://wiki.list.org/display/DOC/Mailman+2.1+Members+Manual

And if you happen to fix any typos or add sections that may be of interest to all Mailman users, please consider contributing them back to the community by editing the wiki version of this document.

A mailing list is simply a list of addresses to which the same information is being sent. If you were a magazine publisher, you would have a list of the mailing addresses of all the subscribers to the magazine. In the case of an electronic mailing list, we use a list of email addresses from people interested in hearing about or discussing a given topic.

Two common types of email mailing lists are announcement lists and discussion lists.

Announcement lists are are used so that one person or group can send announcements to a group of people, much like a magazine publisher’s mailing list is used to send out magazines. For example, a band may use a mailing list to let their fan base know about their upcoming concerts.

A discussion list is used to allow a group of people to discuss topics amongst themselves, with everyone able to send mail to the list and have it distributed to everyone in the group. This discussion may also be moderated, so only selected posts are sent on to the group as a whole, or only certain people are allowed to send to the group. For example, a group of model plane enthusiasts might use a mailing list to share tips about model construction and flying.

Some common terms:

A “post” typically denotes a message sent to a mailing list. (Think of posting a message on a bulletin board.) People who are part of an electronic mailing list are usually called the list’s “members” or “subscribers.” “List administrators” are the people in charge of maintaining that one list. Lists may have one or more administrators. A list may also have people in charge of reading posts and deciding if they should be sent on to all subscribers. These people are called list moderators. Often more than one electronic mailing list will be run using the same piece of software. The person who maintains the software which runs the lists is called the “site administrator.” Often the site administrator also administers individual lists.

GNU Mailman

GNU Mailman is software that lets you manage electronic mailing lists. It supports a wide range of mailing list types, such as general discussion lists and announce-only lists. Mailman has extensive features which make it good for list subscribers, such as easy subscription and unsubscription, privacy options, and the ability to temporarily stop getting posts from the list. The list member features are covered in this document.

Mailman also has many features which make it attractive to list and site administrators. These features are covered in the list and site administrator manuals.
Often, it’s easier to simply give an example than explain exactly how to find the address for your specific list. As such, we’ll frequently give examples for a fictional list called LISTNAME@DOMAIN whose list information page can be found at http://WEBSERVER/mailman/listinfo/LISTNAME.

Neither of these are real addresses, but they show the form of a typical list address. The capital letters used for the list-specific parts of each address should make it easier to see what should be changed for each list. Although specific list configurations may be different, you will probably be able to just replace the words given in capital letters with the appropriate values for a real list:

LISTNAME The name of your list. DOMAIN The name of the mail server which handles that list. WEBSERVER The name of the web server which handles the list web interface. This may be the same as DOMAIN, and often refers to the same machine, but does not have to be identical. As a real-life example, if you are interested in the mailman-users list, you’d make the following substitutions: LISTNAME=mailman-users, DOMAIN=python.org, WEBSERVER=mail.python.org. As such, for the mailman-users@python.org mailing list, the list information page can be found at the URL http://mail.python.org/mailman/listinfo/mailman-users. (These, unlike most of the examples given in this document, are real addresses.)

Most lists will have this information stored in the List-* headers. Many mail programs will hide these by default, so you may have to choose to view full headers before you can see these informational headers.

Mailman

Mailman has two different interfaces for the list subscriber: the web interface and the email interface. Most discussion list subscribers use the email interface, since this includes the email address you use to send mail to all the subscribers of that list.

The interface you use for changing options is largely a matter of preference, since most (but not all) of the options which can be changed from the web interface can also be changed by email. Usually it is easier to use the web interface for changing options, since the web interface provides instructions as you go, but there are times when people may prefer the email interface, so both are provided.

Web

The web interface of Mailman is its selling point for many administrators, since it makes it much easier for subscribers and administrators to see which options are available, and what these options do.

Every mailing list is also accessible by a number of web pages. Note that the exact URLs are configurable by the site administrator, so they may be different than what’s described below. We’ll describe the most common configuration, but check with your site administrator or hosting service for details.

List information (listinfo) page

Usually found at http://WEBSERVER/mailman/listinfo/LISTNAME (for example, http://lists.example.com/mailman/listinfo/mylist) The listinfo page is the starting point for the subscriber interface. As one would assume from the name it’s given, it contains information about the LISTNAME list. Usually all the other subscriber pages can be accessed from this point, so you really only need to know this one address. Member options page

Usually found at http://WEBSERVER/mailman/options/LISTNAME/EMAIL (For example, http://lists.example.com/mailman/options/mylist/kathy@here.com) This page can also be accessed by going to the listinfo page and entering your email address into the box beside the button marked “Unsubscribe or Edit Options” (this is near the bottom of the page). The member options page allows you to log in/out and change your list settings, as well as unsubscribe or get a copy of your password mailed to you. To log in to your member options page: If you are not already logged in, there will be a box near the top for you to enter your password. (If you do not know your
password, see Section 6.1: How do I get my password? for more information on getting your password.) Enter your password in the box and press the button. Once you are logged in, you will be able to view and change all your list settings. List Archives

Usually found at http://WEBSERVER/pipermail/LISTNAME if the list is publicly archived, and http://WEBSERVER/mailman/private/LISTNAME if the list is privately archived. (For example, http://lists.example.com/pipermail/mylist or http://lists.example.com/mailman/private/mylist) The list archive pages have copies of the posts sent to the mailing list, usually grouped by month. In each monthly group, the posts are usually indexed by author, date, thread, and subject. Note: Pipermail is the name of the default archiver that comes with Mailman. Other archive programs are available. If the archive is private, you will need to supply your subscribed email address and your password to log in. (See Section 6.1: How do I get my password? for more information on getting your password.)

email

Every mailing list has a set of email addresses to which messages can be sent. There’s always one address for posting messages to the list, one address to which bounces are sent, and addresses for processing email commands. For a fictional mailing list called mylist@example.com, you’d find these addresses:

mylist@example.com this is the email address people should use for new postings to the list. mylist-join@example.com by sending a message to this address, a new member can request subscription to the list. Both the Subject: header and body of such a message are ignored. Note that mylist-subscribe@example.com is an alias for the -join address. mylist-leave@example.com by sending a message to this address, a member can request unsupscription from the list. As with the -join address, the Subject: header and body of the message is ignored. Note that mylist-unsubscribe@example.com is an alias for the -leave address. mylist-request@example.com This address reaches the list owner and list moderators directly. This is the address you use if you need to contact the person or people in charge. This address reaches a mail robot which processes email commands that can be used to set member subscription options, as well as process other commands. A list of members’ email commands is provided in Appendix A. mylist-bounces@example.com This address receives bounces from members whose addresses have become either temporarily or permanently inactive. The -bounces address is also a mail robot that processes bounces and automatically disables or removes members as configured in the bounce processing settings. Any bounce messages that are either unrecognized, or do not seem to contain member addresses, are forwarded to the list administrators. mylist-confirm@example.com This address is another email robot, which processes confirmation messages for subscription and unsubscription requests. There’s also an -admin address which also reaches the list administrators, but this address only exists for compatibility with older versions of Mailman.

For changing options, we use the LISTNAME-request address (for example, mylist-request@example.com).

Commands can appear in the subject line or the body of the message. Each command should be on a separate line. If your mail program automatically appends a signature to your messages, you may want to put the word “end” (without the quotes) on a separate line after your other commands. The end command tells Mailman not to process the email after that point.

The most important command is probably the “help” command, since it makes Mailman return a message full of useful information about the email commands and directions to the web interface.

Quick references to the subscriber commands have been provided in Appendices A and B. (These have been slightly adapted from the output of the help command.)

|

| 13.1. mailman | 97 |
remember that many mailing list administrators are volunteers who are donating their spare time to run the list, and they may be very busy people.

This list administrator email address is in the form LISTNAME-owner@DOMAIN, where LISTNAME is the name of the list (eg: mailman-users) and DOMAIN is the name of the server (eg: python.org). This email address, along with the email addresses of specific administrators, is given on the bottom of the list information pages. See Section 3.1: The web interface for more information on finding the list information page for your list.

Since subscribing (joining) and unsubscribing (leaving) lists are often the only things a list member needs to know, these can both be done without requiring you to know a password.

? ()

There are two common ways you can subscribe to a Mailman mailing list.

Using the web interface:

1. Go to the list information page for the list you want to join. (This will probably be something like http://WEBSERVER/mailman/listinfo/LISTNAME)
2. Look for the section marked “Subscribing to LISTNAME” and fill in the boxes. You can fill in the following:
   - You must enter your email address. You may choose to supply your real name. You may choose a password. If you do not choose one, Mailman will generate one for you.
   - Do NOT use a valuable password, since this password may be mailed to you in plain text. If the list supports more than one language, you may be able to choose your preferred language. Note: This setting does not affect posts to the list, only pre-prepared Mailman texts such as your member options page.
3. Press the subscribe button. A new page should appear telling you that your request has been sent.

Using the email interface:

1. Open a mail program which sends mail from the address you want to subscribe.
2. Send a mail to the list subscription address, which will be in the form LISTNAME-join@DOMAIN. The subject and body of the message will be ignored, so it doesn’t matter what you put there. You may also use LISTNAME-subscribe@DOMAIN.

After following one of these sets of instructions (you don’t need to do both!), there are a few possible outcomes depending upon the settings for that list.

You may receive an email message asking for confirmation that you really want to be subscribed to the list. This is to prevent anyone from subscribing you to lists without your permission. Follow the instructions given in the message to confirm your wish to be subscribed. A moderator may also need to confirm your subscription if you are subscribing to a limited list. Or you may have to wait for a moderator and follow the instructions in the confirmation mail. Once this is done, you will likely receive another message welcoming you to the list. This message contains some useful information including your list password and some quick links for changing your options, so you may want to save it for later reference.

Note: Subscribing can be done in other ways as well. See Appendix A for more advanced email subscribing commands.

? ()

Don’t want to be on a list any more? If you’re just going on vacation or are too busy to read mails and want to temporarily turn them off, you may want to stop mail delivery rather than unsubscribing. This means you keep your
password and other settings so you can, for example, still have access to private list archives. If this is what you’d prefer, see Section 7.1 for instructions on disabling mail delivery temporarily.

If you actually want to leave the list, there are two common ways you can unsubscribe from a Mailman mailing list.

Using the web interface:

1. Go to the list information page for the list you want to leave. (This will probably be something like http://WEBSERVER/mailman/listinfo/LISTNAME)
2. Look for the section marked “LISTNAME subscribers” (usually found near the bottom of the page).
3. There should be a button marked “Unsubscribe or Edit Options.” Enter your email address in the box beside this button and press the button.
4. You should be brought to a new page which has an “Unsubscribe” button. Press it to unsubscribe and follow the instructions given.

Using the email interface:

1. Open a mail program which sends mail from the address you want to unsubscribe.
2. Send a mail to the list unsubscribe address, which will be of the form LISTNAME-leave@DOMAIN. The subject and body of this message will be ignored, so it doesn’t matter what you put there. You may also use LISTNAME-unsubscribe@DOMAIN.

After following one of these sets of instructions (you don’t need to do both!), you will be sent a confirmation mail and must follow the instructions given in that mail to complete the unsubscription. This is to stop people from unsubscribing you without your permission. In addition, a moderator may need to approve your unsubscription.

If you do not receive this confirmation mail with instructions, make sure that you typed your email address correctly (if you were using the web interface to unsubscribe) and that the address you tried to unsubscribe is, indeed, actually subscribed to that list. For security reasons, Mailman generates the same member options page regardless of whether the address entered is subscribed or not. This means that people cannot use this part of the web interface to find out if someone is subscribed to the list, but it also means that it’s hard to tell if you just made a typo.

Once your unsubscription has been processed, you will will probably receive another message confirming your unsubscribing from the list, and at that point you should stop receiving messages.

If you wish to skip the confirmation process (for example, you might be unsubscribing an address which no longer works), it is possible to bypass it by using your password instead and either logging in to your options page using it (See Section 3.1), or sending it with your email commands to LISTNAME-request (See Appendix A for advanced email unsubscription commands). See Section 6.1 for more information on getting your password.

Your password was either set by you or generated by Mailman when you subscribed. You probably got a copy of it in a welcome message sent when you joined the list, and you may also receive a reminder of it every month. It is used to verify your identity to Mailman so that only the holder of the password (you!) and the administrators can view and change your settings.

Do NOT use a valuable password for Mailman, since it can be sent in plain text to you.

If you’ve forgotten your password and haven’t saved the welcome message or any reminder messages, you can always get a reminder through the web interface:

Go to the list information page for the list from which you wish to get your password (This will probably be something like http://WEBSERVER/mailman/listinfo/LISTNAME) Look for the section marked “LISTNAME subscribers” (this
section is usually found near the bottom of the page). There should be a button marked “Unsubscribe or Edit Options.” Enter your email address in the box beside this button and press the button. You should be brought to a new page which has an “Password Reminder” section. Press the “Remind” button to have your password emailed to you. If you do not receive the password reminder email after doing this, make sure that you typed your email address correctly and that the address you used is, indeed, actually subscribed to that list. For security reasons, Mailman generates the same member options page regardless of whether the address entered is subscribed or not. This means that people cannot use this part of the web interface to find out if someone is subscribed to the list, but it also means that it’s hard to tell if you just made a typo.

You can also get a reminder using the email interface,

Send a mail to LISTNAME-request@DOMAIN with the command password Commands can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.)

If you are not sending mail from your subscribed address, you can also specify this address by sending the command password address=$<$ADDRESS$>$.

Do NOT use a valuable password, since this password may be mailed to you in plain text. From the web interface:

Log in to your member options page. (See Section 3.1 for instructions on how to do this.) Look for the password changing boxes on the right-hand side of the page and enter your new password in the appropriate boxes, then press the button marked “Change My Password.” This can also be changed for multiple lists at the same time if you are subscribed to more than one list on the same domain. See Section 10.1 for information about changing settings globally.

From the email interface:

Send a mail to LISTNAME-request@DOMAIN with the command password <OLDPASSWORD> <NEWPASSWORD>. Commands can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.)

If you are not sending mail from your membership address, you can also specify this address with address=<ADDRESS> after <NEWPASSWORD>.

For example, if kathy@here.com wanted to change her mylist password from zirc to miko, but she was sending mail from her work address kathy@work.com, she could send a message to mylist-request@example.com with the subject set to password zirc miko address=kathy@here.com.

If you don’t wish to the reminder email including your password every month, you can disable it from the member options page. (You can always get the password mailed out when you actually want it. See Section 6.1 for instructions.)

Using the web interface:

Log in to your member options page. (See Section 3.1 for instructions on how to do this.) Look for the section marked “Get password reminder email for this list?” and change the value accordingly. This can also be changed for multiple lists at the same time if you are subscribed to more than one list on the same domain. See Section 10.1 for information about changing settings globally.

Using the email interface:

Send a mail to LISTNAME-request@DOMAIN with the command set reminders on or set reminders off. Commands can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.) Set it to “on” to receive reminders, and “off” to stop receiving reminders.
You may wish to temporarily stop getting messages from the list without having to unsubscribe. If you disable mail delivery, you will no longer receive messages, but will still be a subscriber and will retain your password and other settings.

This can be handy in a many different cases. For example, you could be going on vacation or need a break from the list because you’re too busy to read any extra mail. Many mailing lists also allow only subscribers to post to the list, so if you commonly send mail from more than one address (eg, one address for at home and another for when you’re travelling), you may want to have more than one subscribed account, but have only one of them actually receive mail. You can also use this as a way to read private archives even on a list which may be too busy for you to have sent directly to your mailbox. All you need to do is subscribe, disable mail delivery, and use your password and email to log in to the archives.

To disable/enable mail delivery using the web interface:

Log in to your options page. (See Section 3.1 for instructions.) Go down to the section marked “Mail delivery” and select “Disabled” to stop receiving mail, and “Enabled” to start receiving mail. This can also be changed for multiple lists at the same time if you are subscribed to more than one list on the same domain. See Section 10.1 for information about changing settings globally.

To disable/enable mail delivery using the email interface:

Send a mail to LISTNAME-request@DOMAIN with the command set delivery off or set delivery on. Commands can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.) Set it to “off” to stop receiving posts, and “on” to start receiving them again.

Mailman can’t completely stop you from getting duplicate messages, but it can help. One common reason people get multiple copies of a mail is that the sender has used a “group reply” function to send mail to both the list and some number of individuals. If you want to avoid getting these messages, Mailman can be set to check and see if you are in the To: or CC: lines of the message. If your address appears there, then Mailman can be told not to deliver another copy to you.

To turn this on or off using the web interface:

Log in to your member options page. (See Section 3.1 for more details on how to do this.) Scroll down to the bottom of the page to the section marked “Avoid duplicate copies of messages?” and change the value accordingly. This can also be changed for multiple lists at the same time if you are subscribed to more than one list on the same domain. See Section 10.1 for information about changing settings globally.

To turn this on or off using the email interface:

Send a mail to LISTNAME-request@DOMAIN with the command set duplicates on or set duplicates off. Commands can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.) Set it to “on” to receive list copies of messages already sent to you, set it to “off” to avoid receiving these duplicates.

To change your subscription address,

Log in to your member options page. (See Section 3.1 for more details on how to do this.) In the section marked “Changing your LISTNAME membership information,” enter your new address. If you wish to change your address,
for all subscriptions using the old address, select the “Change globally” box. If you have subscriptions under another
address or for lists on a different domain, these will have to be done separately. See Section 10.1 for more information
about changing settings globally. There is no special way to do this from the email interface, but you can subscribe
and unsubscribe for more or less the same effect. (See Sections 5.1 and 5.2 for more information on subscribing and
unsubscribing.)

/ (myposts option)

By default in Mailman, you get a copy of every post you send to the list. Some people like this since it lets them
know when the post has gone through and means they have a copy of their own words with the rest of a discussion,
but others don’t want to bother downloading copies of their own posts.

This option has no effect if you are receiving digests. You may also want to see Section 7.5, which discusses acknowl-
edgement emails for posts sent to the list.

To set this using the web interface:

Log in to your member options page. (See Section 3.1 for more details on how to do this.) Look for the section marked
“Receive your own posts to the list?” Set it to “Yes” to receive copies of your own posts, and “No” to avoid receiving
them. To set this using the email interface:

Send a mail to LISTNAME-request@DOMAIN with the command set myposts on or set myposts off. Commands
can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail
commands.) Set it to “on” to receive copies of your own posts, and “off” to avoid receiving them.

(ack option)

On most lists, you will simply receive a copy of your mail when it has gone through the list software, but if this is
disabled (See Section 7.4), your list mail delivery is disabled (See Section 7.1), you are not subscribed to that topic
(See Section 9.2) or you simply want an extra acknowledgement from the system, this option may be useful to you.

Note: If you are not subscribed to the list, this option cannot be used. You must either check the archives yourself (if
the list has public archives), ask someone who is subscribed to the list, or subscribe to use this option.

To set this using the web interface:

Log in to your member options page. (See Section 3.1 for more details on how to do this.) Look for the section marked
“Receive acknowledgement mail when you send mail to the list?” Set it to “Yes” to receive a mail letting you
know your post has been received, and “No” to avoid receiving such an acknowledgement. To set this using the email
interface:

Send a mail to LISTNAME-request@DOMAIN with the command set ack on or set ack off. Commands can appear in
either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.)
Set it to “on” if you wish to receive mail letting you know your post has been received, and “off” to avoid receiving
such an acknowledgement.

There are a few common reasons for this:

No one has sent any mail to the list(s) you’re on for a little while. To check if this is the case, try visiting the archives
of the list (assuming that the list has archives). If the list has no archives, you may have to ask another subscriber. (See
Section 3.1 for help in finding the list archives.)

Note: Generally, it is considered impolite to send test messages to the entire list. If you feel a need to test that the list
is working and for some reason you cannot simply compose a regular message to the list, it is less disruptive to send
You were bouncing mail and have had mail delivery (temporarily) disabled by the list software. If your mail provider “bounces” too many messages (that is, it tells Mailman that the message could not be delivered) Mailman eventually stops trying to send you mail. This feature allows Mailman to gracefully handle addresses which no longer exist (for example, the subscriber has found a new internet service provider and forgot to unsubscribe the old address), as well as addresses which are temporarily out-of-service (for example, the subscriber has used up all of the allotted space for his or her email account, or the subscriber’s mail provider is experiencing difficulties).

Even if you are unaware of any difficulties with your mail provider, it is a good idea to check this. Some popular webmail providers and internet servers are not as reliable as one might assume, nor is the internet as a whole. You may want to also send yourself a test message from another account or ask a friend to send you a test message to make sure your subscribed address is working.

To check if this may be the reason you are not receiving messages, log in to the your options page (See Section 3.1 for more details on how to do this) and look at your options. There should be one marked “Mail Delivery” - if it is set to “Disabled,” set it to “Enabled” to start receiving mail again. (For more instructions on disabling or enabling mail delivery, see Section 7.1.)

Note: Even if you have not been disabled at the time you check, you could be bouncing messages and not have reached the threshold for your subscription to be disabled. You may need to check again.

There is a delay or break in the networks between you and the list server. No matter what many of us would like, the internet is not 100% reliable, nor is it always fast. Sometimes, messages simply take a long time to get to you. Try to be patient, especially if the server is far (in terms of networks, not geography, although often one implies the other) from your internet service provider.

To check if this might be causing your problem, you can try pinging the list server or tracing the route between you and it. (Instructions on how to do this varies from platform to platform, so you may want to use a search engine to find those appropriate for you.)

The Mailman installation on the list server is not functioning or not functioning properly. To test if this is a case, try visiting the list’s web interface and try sending a message to LISTNAME-request@DOMAIN with the command “help” (without the quotes) in the Subject:. If neither of these works after a reasonable length of time, this may be the problem. You may wish to contact either the list or site administrator(s).

8.1 How can I start or stop getting the list posts grouped into one big email? (digest option)

Groups of posts are called “digests” in Mailman. Rather than get messages one at a time, you can get messages grouped together. On a moderately busy list, this typically means you get one email per day, although it may be more or less frequent depending upon the list.

You may also want to look at Section 8.2 which discusses MIME and plain text digests.

To turn digest mode on or off using the web interface,

Log in to your member options page. (See Section 3.1 for more details on how to do this.) Look for the section marked “Set Digest Mode.” Set it to “On” to receive messages bundled together in digests. Set it to “Off” to receive posts separately.

To turn digest mode on or off using the email interface,

Send a mail to LISTNAME-request@DOMAIN with the command set digest plain or set digest mime or set digest off. Commands can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.) Set it to “off” if you wish to receive individual posts separately, and to “plain” or “mime” to receive posts grouped into one large mail. See Section 8.2 for more information on plain versus MIME digests.
8.2 What are MIME and Plain Text Digests? How do I change which one I get? (digest option)

MIME is short for Multipurpose Internet Mail Extensions. It is used to send things by email which are not necessarily simple plain text. (For example, MIME would be used if you were sending a picture of your dog to a friend.)

A MIME digest has each message as an attachment inside the message, along with a summary table of contents.

A plain text digest is a simpler form of digest, which should be readable even in mail readers which don’t support MIME. The messages are simply put one after the other into one large text message.

Most modern mail programs do support MIME, so you only need to choose plain text digests if you are having trouble reading the MIME ones.

Note: This option has no effect if you are not receiving mail bunched as digests. (See Section 8.1 for more information on receiving mail as digests.)

To set your digest type using the web interface:

Log in to your member options page. (See Section 3.1 for more details on how to do this.) Look for the section marked “Get MIME or Plain Text Digests?.” Set it to “MIME” to receive digests in MIME format, or “Plain text” to receive digests in plain text format. This can also be changed for multiple lists at the same time if you are subscribed to more than one list on the same domain. See Section 10.1 for information about changing settings globally.

To set your digest type using the email interface,

Send a mail to LISTNAME-request@DOMAIN with the command set digest plain or set digest mime. Commands can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.) Set it to “plain” to get posts bundled into a plain text digest, or “mime” to get posts bundled together into a MIME digest.

Some lists are set up so that different topics are handled by Mailman. For example, the courses list on Linuxchix.org is a discussion list for courses being run by linuxchix members, and often there are several courses being run at the same time. (eg: Networking for beginners, C programming, LaTeX document mark up.) Each of the courses being run is a separate topic on the list so that people can choose only to receive the course they want to take.

These topics must be configured by the list administrator, but it is the responsibility of each poster to make sure that their post is put with the correct topic. Usually, this means adding a tag of some type to the subject line (eg: [Networking] What type of cables do I need?) or making sure the Keywords: line has the right information. (By default, you can put a Keywords: section in the beginning of the body of your message, but this can be configured by your list administrator.) Note that these tags are case-insensitive.

9.1 How do I make sure that my post has the right topic?

When a list administrator defines a topic, he or she sets three things:

a topic name a regular expression (regexp) a description You can view this information by logging in to your member options page. (See Section 3.1 for more details on how to do this.) and clicking on the “details” link for any topic that interests you.

To post on a given topic, you need to make sure that the Keywords: or Subject: headers in a message match the regular expression for that topic. Regular expressions can actually be fairly complex, so you may want to just ask the list administrator if you don’t know how to make heads or tails of the expression given.

Most Mailman topic expressions will be fairly simple regular expressions, so in this document we will simply give you some common examples. Regular expressions are a bit too complex to teach in a few lines here, so if you really
want to understand how the regular expressions work, you should find a tutorial or reference elsewhere. (For example, DevShed has a decent tutorial at http://www.devshed.com/Server_Side/Administration/RegExp/)

Here are some examples of possible regular expressions and matching lines:

Regular expression Matching lines zuff Keywords: zuff zuff Keywords: ZUFF zuff Keywords: Zuff zuff Keywords: amaryllis, zuff, applesauce zuff Subject: [zuff] Do you have the right stuff for zuff? zuff Subject: Do you have the right stuff for zuff? zuff Subject: What is zuff?

Keywords: [zuff]  [zuff]

Subject: [zuff] Do you have the right stuff?  [zuff]

Subject: Online zuff tutorials (was Re: [zuff] What is zuff?) A few notes:

The matching is case-insensitive, so if zuff matches, so will ZUFF, zuFF, and any other variations in capitalization. Some characters have special meaning in a regular expression, so to match those characters specifically, they must be “escaped” with a backslash (). As you can see in the above example, [ and ] are such characters. (Others include “.”, “?”, “*”, and “^”). The backslash is also used for other things (I wasn’t kidding about regular expressions being complex: consult other documentation for details about other uses of the backslash character), but this is the most likely use in a topic expression.

9.2 How do I subscribe to all or only some topics on a list?

If topics have been set up by your mailing list administrator, you can choose to subscribe to only part of a list by selecting the topics you want to receive.

If you wish to get all messages sent to the list, make sure you are not subscribed to any topics.

Log in to your member options page. (See Section 3.1 for more details on how to do this.) Look for the section marked “Which topic categories would you like to subscribe to?” If any topics are defined, you can select those you wish. If you do not select any topics of interest, you will receive all posts sent to the list.

You probably also want to look at Section 9.3 which discusses changing your settings for messages where no topic is set.

9.3 How do I get or avoid getting messages with no topic set?

If you wish to get all messages sent to the list, make sure you are not subscribed to any specific topic. (See Section 9.2.)

If you are only subscribed to some topics, you can either choose to either receive or not receive messages with no topic set, much the way you can choose to subscribe only to certain topics.

To change this setting,

Log in to your member options page. (See Section 3.1 for more details on how to do this.) Look for the section marked “Do you want to receive message that do not match any topic filter?” If you wish to receive messages with no topic set, select “Yes.” If you do not wish to receive such messages, choose “No.”

This setting has no effect if you are not subscribed to any topics.
10.1 Change Globally? Set Globally? What does that mean?

For some of the options given in your member options page, there is a tick-box which says “Change Globally” or “Set Globally.” This means that if you change this option, you can also have the change made for all your other list subscriptions with the same address to lists on the same domain. This can be handy if, for example, you want to make sure all your passwords are the same, or you are going on vacation and want to turn off mail delivery from all the lists.

To change your subscription name,

1. Log in to your member options page. (See Section 3.1 for more details on how to do this.)
2. In the section marked “Changing your LISTNAME membership information,” enter your new name in the appropriate box.

This can also be changed for multiple lists at the same time if you are subscribed to more than one list on the same domain. See Section 10.1 for information about changing settings globally.

Note: You do not need to have a subscription name set.

Mailman is available with many different languages. (For a complete listing see http://mailman.sourceforge.net/i18n.html.) This means that, if your list has other languages enabled, you may be able to have the web interface, etc. in a language of your choice.

This does NOT necessarily mean that all the posts sent to the list will be in the language you selected. Only the pre-prepared texts presented by Mailman will be affected by this setting. Posts are in whatever language the poster uses. Your preferred language is set when you subscribe (see Section 5.1), and can be changed later if the list supports more than one language.

To change your preferred language in Mailman,

1. Log in to your member options page. (See Section 3.1 for instructions on how to do this.)
2. Go to the section marked “What language do you prefer?” and choose the appropriate language from the drop-down list. If there is no drop-down list of languages, the list you are on probably only supports one language.

If your list does not support the language you would prefer to use, you may contact the list administrator (LISTNAME-owner@DOMAIN) to see if it can be added, but remember that this may mean some work that the list and/or site administrator(s) do not have time or the ability to do.

If your language of choice is not available because no translation exists for Mailman, please consider volunteering your time as a translator. For more information you may want to consult the mailman-i18n mailing list at http://mail.python.org/mailman/listinfo/mailman-i18n. (i18n is a common short-hand for “internationalization” because the word starts with an i, ends with an n, and has 18 letters in between. If you mumble a bit, i18n even sounds a bit like “internationalization.”)

10.4 How do I avoid having my name appear on the subscribers list? (the hide option)

If you do not want to have your email address show up on the subscriber list for any reason, you can opt to have it concealed.
Common reasons for doing this include avoiding unsolicited bulk email (spam). By default, the subscribers list is obscured to hinder spam harvesters, but if you feel this is insufficient it’s easy enough to remove address from the subscriber list given in the information pages or by email request. (Note that this does not conceal your address from the list administrators.) You may wish to see Section 11.2 for more information on what Mailman can do to help avoid spam.

To change this setting using the web interface:

1. Log in to your member options page. (See Section 3.1 for instructions on how to do this.)
2. Go to the section marked “Conceal yourself from subscriber list?” and choose “Yes” to hide your name from the list, or “No” to allow your name to appear on the list.

To change this setting using the email interface:

1. Send a mail to LISTNAME-request@DOMAIN with the command set hide on or set hide off.

Commands can appear in either the body or the subject of the message. (See Section 3.2 for more information about sending mail commands.)
2. Set it to “on” to conceal your email address from the membership list, or “off” to stop concealing your address.

If the list has archives, they can be viewed by going to a web page address. This address usually linked from the list information page and can be found in the List-Archive: of every list message unless your list administrator has disabled these headers. (Many mail programs hide the List-Archive: mail header, so you may have to tell your mail program to allow you to view full headers before you will be able to see it.)

Public archives usually have addresses of the form http://WEBSEVER/pipermail/LISTNAME/ and private archives usually have addresses of the form http://WEBSEVER/mailman/private/LISTNAME.

See Section 3.1 for more information on finding the addresses of a list.

A technical list’s archives may include answers to a range of different questions. Often, the people who have posted these answers would be happy to help someone who doesn’t quite understand the answer, and don’t mind giving their address out for that purpose. But although it would be wonderful if everyone could contact each other easily, we also want to make sure that the list and list archives are not abused by people who send spam.

To make a range of options available to list administrators, Mailman allows a variety of configurations to help protect email addresses. Many of these settings are optional to the list administrator, so your particular list may be set up in many ways. List administrators must walk a fine line between protecting subscribers and making it difficult for people to get in touch.

Subscriber lists o The list administrator can choose to have the subscriber list public, viewable only to list members, or viewable only to list administrators. o The subscriber list is shown with the addresses obscured to make it difficult for spam harvesters to collect your address. o You can choose to have your address hidden from the subscriber list. (See Section 10.4 for more information.) o Note: The entire subscriber list is always available to the list administrators.
List archives o The list administrator can choose for the archives to be public, viewable only to members (private), or completely unavailable. o The HTML archives which are created by Pipermail (the archiving program which comes default with Mailman) contain only obscured addresses. Other archiving programs are available and can do different levels of obfuscation to make addresses less readable. o If you wish to be more sure, you can set the mail header X-no-archive: and Mailman will not archive your posts.
This does not stop other members from quoting your posts, possibly even including your email address. Limited posting to the lists. The list administrator can choose who can post to the list. Most lists are either moderated (a moderator or administrator reviews each posting), set so only subscribers may post to the list, or allow anyone to post to the list. By allowing only subscribers to post to a list, Mailman often blocks all spam and some viruses from being sent through the list. As such, this is a fairly common setting used by list administrators. Anonymous lists can also be made fully anonymous: all identifying information about the sender is stripped from the header before the message is sent on. This is not typically used for anti-spam measures (it has other uses), but it could be used in that way if desired. Of course, many address-obscuring methods can be circumvented by determined people, so be aware that the protections used may not be enough.

Email commands quick reference
Member options quick reference

13.1.3

This is one of a set of three documents for the various people involved in running and using a mailing list. For the members of mailing lists (the subscribers), we have the Mailman 2.1 Members Manual which covers things like subscribing and unsubscribing. For the people who install and set up Mailman, we will eventually have the Mailman 2.1 Site Administration Manual, which has not yet been written.

This document covers the people who run mailing lists: the list administrators (who can change all the list options, as well as authorize postings that have been held for moderation), and the list moderators (who cannot modify list settings).

This document does not need to be read in order, so if you just want an answer to a specific question, jump to the appropriate place and references to other sections will be provided when necessary or potentially helpful.

The bulk of this document has been written by Terri Oda. Terri used to get a lot of spam, so she did research into neat evolutionary anti-spam solutions. And then she thought it would be nice if her grandfather could use the web without worrying about bad stuff getting into his computer, so she started doing web security. All of this is a roundabout way of saying that her day job is as a security researcher, and she actually doesn’t consider herself a document writer, but she was tired of Mailman not having documentation so she figured it was time to do something about it.

The original site admin docs were written by Barry Warsaw. Thanks also go to the rest of the Mailman team who provided the inline help upon which much of this documentation is based.

This document is now stored in the Mailman wiki http://wiki.list.org, and may be edited and added to by members of the Mailman community. Please see the wiki changelogs for more information.

Mailman’s

Mailman3:

• Web – used by most people, this is the main interface most users will see
• email – used as an alternative to the web interface
• – used mostly by site admins, this is very powerful but perhaps not so user-friendly
Web

The most commonly used interface of Mailman is the web interface. Almost all the options can be set from this interface, and it also provides inline help and descriptions of each option. More importantly, the day-to-day tasks of running a mailing list are handled through this interface.

Assuming a fairly standard configuration, the configuration interface has a URL like this:

http://WEBSERVER/mailman/admin/LISTNAME

And the moderation interface has a very similar URL as follows:

http://WEBSERVER/mailman/admindb/LISTNAME

The web domain (WEBSERVER) can’t be set from the web, since it’s a bit too easy to render your web interface unusable by making a typo. As such, this setting is usually changed from the command line. The fix_url script is provided for this purpose. It is used in conjunction with withlist as follows:

bin/withlist -l -r fix_url listname [options] In many cases, this is the only interface a list administrator will use, and may be the only interface the list administrator has permissions to use.

email

There is also an email interface for Mailman, allowing list admins to adjust settings or handle caught messages without requiring them to open up a web browser.

Since this interface is not as commonly used as the web interface, it is highly likely that this part of the documentation will be among the last pieces written. If you wish to speed this up, please feel free to help fill it in!

The web interface for moderation can be found at a URL similar to the following, depending on your mailman config:

http://WEBSERVER/mailman/admindb/LISTNAME

From here, you will see a list of anything awaiting your decision. This is largely messages that have been caught for some reason, but may also include subscription/unsubscription requests if you require admin authorization before someone can join or leave the list.

?  

Messages are “caught” in the moderation queue depending upon your list settings. There are a number of rules that allow you to choose which messages require moderation, but basically you want things which might be inappropriate to be caught by these filters so a human can choose whether they get sent to all subscribers or not.

For example, as an anti-spam measure, many lists only accept messages from their members. Any non-member who sends a message will have it caught in the moderation queue. If the message turns out to be spam, it can be discarded, but if it turns out to be from someone posting from their work address, you may wish to send it out anyhow.

FIXME

Put a link to the relevant section here
For each message displayed in the queue, Mailman will give a summary of the message. This includes the following:
The sender’s address The subject of the message The size The reason the message was held for moderation (e.g., “Post by non-member to a members-only list”) The date/time the message was received A link is provided so you can read the entire message if you want to.

There are 4 choices when it comes to moderating a message:

Defer This option allows you to put off the choice until later. It leaves the message in the queue and does nothing to it. Accept This accepts the message, allowing it to be sent to all of your subscribers. Reject This rejects the message, letting the sender know that it was rejected. You may also include a message explaining why. The reject option is usually used for messages which are inappropriate for that list for some reason – they may have attachments which are too large, be off-topic, or from someone who is not allowed to post to that list. Rejected messages are not sent to the list. Discard This discards the message, ensuring that it is not sent to the list. In this case, no message is sent to the sender. This option is most often used for spam messages. In addition, there is a checkbox (near the submit button) which allows you to “Discard all messages marked Defer” – this allows you to bulk-discard spam messages and can be a very handy option. It only applies to the messages displayed in the queue page as you've loaded it.

This is a setting chosen when the list was set up. If you do not wish to moderate subscription/unsubscription requests, you can change your list config.

The interface for moderating members joining and leaving is similar to that for messages, only instead of a message summary we have the email address and name of the person under consideration.

The options for subscriptions and unsubscriptions are the same as those for messages:

- Defer
  - Decide later.
- Accept
  - Allow this person to subscribe or unsubscribe
- Reject
  - Reject this person’s request, letting them know and optionally giving them a reason for the rejection.
- Discard
  - Discard this person’s request silently, dropping it from the queue without notifying the person.
Removing/deleting mailing lists is performed with the command line tool rmlist. rmlist has 2 modes of operating:

Only remove the list from the admin interface. This retains archives on file system (that is, postings) but makes the list completely inaccessible to users, this removes: The list itself from the web admin interface as well as for users of the list List meta data: Name, description, etc. Policies in place (for example, list moderation settings) List members Like #1 above but also removes the list archives If list archives are retained (default behavior), the archive files are left on the file system. The Mailman administrator can still access the archives through web interface (if the URL has been bookmarked) but list users (and the list administrator) will not have access unless another web service is making the list archives accessible (for example through a search engine).

NOTE removing the list completely requires manual editing of the mail aliases file, that is, /etc/aliases

Once a list has been removed, it may be re-added with the newlist command line tool, however all information about the list (for example, description and previous members) will be missing. However the archives will then be available to all users via the web interface.

It is also possible to allow mailing list removal with the option/variable OWNERS_CAN_DELETE_THEIR_OWN_LISTS in mm_cfg.py, by default this option is disabled.

13.1.4

this is actually a scratch-pad for the site admin docs, which have not yet been written.

Tentative table of contents entries:

Links to the installation guides The $MAILMAN/bin commands how to use withlist Merging archives
sendmail

1

This method uses the mm-handler. See also Method 2 which does not use the mm-handler

The following has been tested using the following configuration
Mailman 2.1.9 Sendmail 8.13.1 CentOS release 4.5 (Final) Attached mm-handler (most noticeable change is writing logs to syslog) You are going to be using the server only for mailing lists and not for delivery to local addresses

For the purpose of this document the server domain shall be called mailman.foo.com and it is a ‘A’ name entry in the DNS.

virtusertable  add the following in your /etc/mail/virtusertable

virtusertable
root@mailman.foo.bar admin@foo.bar

this shall redirect any email to root on the server to be redirected to the email admin on the mail server.

local.m4  local mailer is used by sendmail to deliver mails; however since we are not using the server to host any emails we don’t need it. We can change it to pass all the mails to mm-handler. Edit /usr/share/sendmail-cf/mailer/local.m4

local.m4
at the end of the file look for
Mlocal, ...
and replace it with
Mlocal, P=/etc/mail/mm-handler, F=rDFMhlqSu, S=EnvFromL, R=EnvToL/HdrToL, T=DNS/RFC822/X-Unix, U=mailman:mail, A=mm-handler $h -j $j -d $u

: Make sure that there is a tab after Mlocal, and not spaces. Otherwise it might not work

: This Mlocal is modified so that sendmail passes the hostname using -j parameter. Also a -d parameter has been added before $u.

This assumes that mm-handler is installed in /etc/mail/mm-handler. You change it to the location where mm-handler is installed/copied.
U=mailman:mail shall run mm-handler as user:mailman, group:mail. Change it to one that your mailman installation expects.

mm-handler  Copy the file into /etc/mail/ directory (or where ever your installation of sendmail is).

Make sure that the path in your local.m4 is the same as the one where you are copying the mm-handler file chmod ug+x mm-handler
This shall make the script executable by user and group.
chown root:mail mm-handler; This changes the group to mail. However you should make it that specified in local.m4.

Now edit mm-handler and make sure that.

perl path is correct. $MMWRAPPER is referring to right mailman configuration. $MMLISTDIR is referring to the correct directory. $SEND_MAIL is referring to correct program.

Warning

$SEND_MAIL is specific to the mm-handler attached to this document After configuring the variables in mm-handler run it on the console. This should generate log entries in your syslog. In this case look for /var/log/maillog or /var/log/messages or wherever your system is configured to log these subsystems (refer to syslog.conf). Month&nbs; DD HH:MM:SS mailman mm-handler[32284]: running as root:root Month&nbs; DD HH:MM:SS mailman mm-handler[32284]: running as root. For security reasons you should configure sendmail.mc to run the mailer as non-root Month&nbs; DD HH:MM:SS mailman mm-handler[32284]: #ARGV = -1 ARGV = &nbsp; Month&nbs; DD HH:MM:SS mailman mm-handler[32284]: No arguments passed, doing nothing

If any of the configuration parameters are not correct i.e. correct file is not there or directory doesn’t exist then you should see error messages like Month DD HH:MM:SS mailman mm-handler[3402]: can not find /path/to/mailman.&nbsp; Please configure

Month DD HH:MM:SS mailman mm-handler[3408]: can not find /path/to/lists.&nbsp; Please configure

Once mm-handler can run with no errors. Now you can try to send emails the server and see how it behaves. You can see the log output to verify that emails are being handled properly. Once you are satisfied with the configuration, you can change the syslog.conf and change the log level of mail to level 4 e.g mail.4 /path/to/maillog

This will log error and warning messages to your log file that can be used for administration purposes.

2

This method uses a ‘Postfix’ workaround.

See also Integrating Mailman with Sendmail - Method 1 which uses mm-handler.

To be read in conjunction with the mailman installation instructions at &lt;http://www.list.org/mailman-install/index.html&gt; and the post by Ed Greenberg at &lt;http://mail.python.org/pipermail/mailman-users/2004-June/037518.html&gt;

1 - Installation requirements  I’m using Apache2, Mailman 2.19 and FreeBSD4 and Python 2.4

2 - Set Up Your System  As root

#adduser I used -&gt; user: mailman, group:mailman, password: n

3 - Build and Install Mailman  create Installation Directory (as root)

su# cd /usr/local/ su# mkdir mailman su# chown mailman mailman su# cd mailman su# chgrp mailnull . su# chmod a+rx,g+ws . Now, go to the directory where you have downloaded mailman

su# cd /usr/home/xxxxx/mailman-2.1.9 Change back to user root

su# su root You can ascertain the correct option for –with-mail-gid from /etc/ mail/sendmail.cf

su# grep “DefaultUser” /etc/mail/sendmail.cf => #O DefaultUser=mailnull So use mailnull

su# su root su# make clean su# configure –with-mail-gid=mailnull I’m on FreeBSD so use this command
su# make DIRSETGID=: install Now check permissions
su# cd /usr/local/mailman su# bin/check_perms -f My webserver runs as nobody (check httpd.conf on your system to confirm)
su# grep “User ” /usr/local/apache2/conf/httpd.conf => User nobody
su# cd archives su# chown nobody private su# chmod o-x private 4 - Check your installation
su# cd /usr/local/mailman su# bin/check_perms -f 5 - Set up your webserver
Add this to your httpd-vhosts.conf, or httpd.conf depending on which version of Apache you are using


6 - Integrating sendmail and mailman
Integrating sendmail and mailman

mm-handler would not work for me (after considerable amount of time trying) - I think because by server was medicine.net.au and the address I wanted to use was practiceimprovement.org.au, even though practiceimprovement.org.au was correctly set up on dns to be delegated to the right server.

So I used Ed Greenberg’s clever approach at <http://mail.python.org/pipermail/mailman-users/2004-June/037518.html> which seems better to me anyway.

Note: On Freebsd I first had to build sudo!

create the file /usr/sbin/mailman.aliases

su# pico /usr/bin/mailman.aliases containing the lines below =>

/bin/cp /usr/local/mailman/data/aliases /etc/mail/mailman.aliases /usr/bin/newaliases Note: I spent a lot of time bug testing as I used ‘cp’ in the mailman.aliases script rather than /usr/cp (it worked from the command line but not the web interface)

make it executable

su# chmd 755 /usr/sbin/mailman.aliases Change your sendmail.cf file to include the new alias file (for me, add this to medicine.net.au.mc) define(‘ALIAS_FILE’, ‘etc/mail/aliases/etc/mail/mailman.aliases’) (and then as root)

su# /etc/mail/make install restart Update your sudoers file (‘nobody’ is the user apache runs under - check your httpd.conf)

su# visudo add=> nobody ALL= NOPASSWD: /usr/local/sbin/mailman.aliases mailman ALL= NOPASSWD: /usr/local/sbin/mailman.aliases Also, if you have

Defaults requiretty in the sudoers file, you need to remove or comment it.

7 - Review your site defaults
my mm_cfg.py has these added


Steps 8 - 15 Follow the rest of the instructions from 8 on at -> <http://www.gnu.org/software/mailman/mailman-install/index.html>
When you create a list using /bin/newlist, you should find that the file /etc/mail/mailman.aliases has been created / updated. (I had some permissions to sort out in a few directories before it all worked smoothly) The error log is helpful at /usr/local/mailman/logs/error

These commands can be helpful for troubleshooting, after you have successfully created a testlist and subscribed yourself to that list

To test mailman

su# echo "From: tony@pi.com.au To: testlist@pi.com.au Subject: Happy New Year
test mail body " | /usr/local/mailman/mail/mailman post testlist If that works, use this to test your sendmail/mailman integration

su# echo "From: tony@pi.com.au To: testlist@pi.com.au Subject: Merry Xmas
test mail body " | /bin/sendmail -tony@pi.com.au testlist@pi.com.au Hope this helps someone and would welcome corrections or improvements.

**postfix**

This instruction set is written for those who wish to integrate Mailman with the postfix MTA. The documentation currently available does not cover in detail how to use virtual hosts and MySQL-based aliases;

Before beginning, postfix should be installed and working correctly. If you are using or plan to use virtual hosts, set this up prior to installing and configuring Mailman.

**postfix**

If you do not have any subdomains or do not plan on using virtual hosts within postfix, follow the steps below to integrate Mailman with your system.

$prefix/Mailman/mm_cfg.py:

MTA = 'Postfix'

: that depending on your specific distribution, this setting may be set within $prefix/Mailman/Defaults.py.

mailman:

$prefix/bin/genaliases

: aliases aliases.db. $prefix/data ..//var/lib/mailman/data .

mailman

**postfix**

Let’s assume you have several domains (in this case example1.com and example2.net) and you plan on hosting mailman on lists.example1.com and lists.example2.net. You will have to select a default domain you want to host mailman on. It doesn’t really matter which one you choose but remember that it will be the one where everyone will come to administer their lists. In this case, I will select lists.example1.com
1. \$prefix/Mailman/mm.cfg.py, MTA:

   POSTFIX_STYLE_VIRTUAL_DOMAINS = ['example1.com', 'example2.net']
   VIRTUAL_MAILMAN_LOCAL_DOMAIN = 'localhost'

2. DEFAULT_EMAIL_HOST DEFAULT_URL_HOST:

   #-------------------------------------------------------------
   # Default domain for email addresses of newly created MLs
   DEFAULT_EMAIL_HOST = 'lists.mikesoh.com'
   #-------------------------------------------------------------
   # Default host for web interface of newly created MLs
   DEFAULT_URL_HOST = 'lists.mikesoh.com'

3. mm.cfg.py \$prefix/bin/genaliases, taking care to make sure that the resulting alias files are readable by postfix.

4. postfix mailman . main.cf, :

   virtual_alias_maps = mysql:/etc/postfix/virtual/existing_aliases.cf,
   hash:/var/lib/mailman/data/virtual-mailman
   alias_maps = hash:/etc/aliases, hash:/var/lib/mailman/data/aliases

   : The above additions to virtual_alias_maps and alias_maps are not used by postfix_to_mailman.py and can potentially cause conflicts within Postfix over delivery of mail to Mailman.
   The GNU-Mailman project recommends the methods outlined in section 6.1.1 and section 6.1.2 of the Installation Manual for integrating Postfix and Mailman with virtual domains.

5. main.cf . master.cf, If not already included, add the mailman service to the end of the file:

   #==================================================================
   # service type private unpriv chroot wakeup maxproc command + args
   # (yes) (yes) (yes) (never) (100)
   #==================================================================
   mailman unix - n n - - pipe
   flags=FR user=list argv=/usr/lib/mailman/bin/postfix-to-mailman.py
   ${nexthop} ${user}

6. master.cf . transport, By default, this file should be located within the same directory as your main.cf and master.cf files. Add the following lines:

   example1.com mailman:
   example2.net mailman:

7. transport postmap -v.

   postmap -v /etc/postfix/transport

8. mailman postfix

web. \$prefix/bin/newlist.

13.2 phplist
Variable envelope return path (VERP) is a technique used by some [[electronic mailing list]] software to enable automatic detection and removal of undeliverable [[e-mail address]]es. It works by using a different [[return path]] (also called “envelope sender”) for each recipient of a message.

14.1

Any long-lived mailing list is going to eventually contain addresses that can’t be reached. Addresses that were once valid can become unusable because the person receiving the mail there has switched to a different [[Internet service provider|provider]]. In another scenario, the address may still exist but be abandoned, with unread mail accumulating until there is not enough room left to accept any more.

When a message is sent to a mailing list, the mailing list software re-sends it to all of the addresses on the list. The presence of invalid addresses in the list results in [[bounce message]]s being sent to the owner of the list. If the mailing list is small, the owner can read the bounce messages and manually remove the invalid addresses from the list. With a larger mailing list, this is a tedious, unpleasant job, so it is desirable to automate the process.

Unfortunately, most bounce messages have historically been designed to be read by human users, not automatically handled by software. They all convey the same basic idea (“the message from X to Y could not be delivered because of reason Z”) but with so many variations that it would be nearly impossible to write a program to reliably interpret the meaning of every bounce message. RFC 1894 (obsoleted by RFC 3464) defines a standard format to fix this problem, but support for the standard is far from universal. Having said that, there are several common formats (e.g., RFC 3464, [[qmail]]’s [http://cr.yp.to/proto/qsbmf.txt qsbmf], and Microsoft’s DSN format for [[Microsoft Exchange Server|Exchange]]) that cover large proportion of bounces.

Microsoft Exchange can sometimes bounce a message without providing any indication of the address to which the original message was sent. When Exchange knows the intended recipient, but is not willing to accept email for him, it omits his address. If a message is sent to <code>joe@example.com</code> and the server knows that this is <code>Joe User</code>, it will bounce the message saying that the message to <code>Joe User</code> could not be delivered, leaving out the <code>joe@example.com</code> address altogether. VERP is the only viable way to handle such bounces correctly.

14.2 VERP

The hard part of bounce handling is matching up a bounce message with the undeliverable address that caused the bounce. If the mailing list software can see that a bounce resulted from an attempt to send a message to <tt>user@example.com</tt> then it doesn’t need to understand the rest of the information in the bounce. It can simply
count how many messages were recently sent to <tt>user@example.com</tt>, and how many bounces resulted, and if the proportion of bounced messages is too high, the address is removed from the list.

While bounce message formats in general vary wildly, there is one aspect of a bounce message that is highly predictable: “the address to which it will be sent”. VERP takes full advantage of this. In a mailing list that uses VERP, a different sender address is used for each recipient.

The mailing list manager knows that it sent a message from X to Y, so if a bounce message is received at address X, it can only be because address Y was undeliverable, because nothing was sent from X to any other destination. Thus the important information has been extracted from the bounce message, without any need to understand its contents, which means the person in charge of the list does not need to deal with it manually.

### 14.3

The first serious advocate of this solution, and the originator of the term “‘VERP’” to describe it, was [[Daniel J. Bernstein]], who first put the idea into practice in his [[qmail]] [[Mail transfer agent|MTA]] and [[ezmlm]] mailing list manager. See [http://cr.yp.to/proto/verp.txt his original explanation], dated February 1, 1997.

### 14.4

Assume there is mailing list called <code>wikipedians@example.net</code> and that an individual, <code>bob@example.org</code> has subscribed to it, but later on, Bob has left example.org, so his address is no longer valid. Consider what happens when someone sends a message to the list.

#### 14.5 Without VERP

Without VERP, the mailing list manager might send a message with the following characteristics: * envelope sender: <code>wikipedians-owner@example.net</code> * recipient: <code>bob@example.org</code>

This would result in a bounce, generated by the MTA of either example.net or example.org, with the following characteristics:

- envelope sender: ‘empty’
- recipient: <code>wikipedians-owner@example.net</code>
- contents: example.org was unable to deliver the following message to bob: ...

The mailing list manager can’t be expected to understand the contents of this bounce, and can’t deduce anything from the recipient address because hundreds of other people besides Bob were also sent messages from <code>wikipedians-owner@example.net</code>.

#### 14.6 With VERP

With VERP, the original message would be different:

- envelope sender: <code>wikipedians-owner+bob@example.org@example.net</code>
- recipient: <code>bob@example.org</code>

The bounce, then, will be more useful:

- envelope sender: ‘empty’
• recipient: `<code>wikipedians-owner+bob@example.org@example.net</code>`
• contents: example.org was unable to deliver the following message to bob: ...

From this bounce message the mailing list manager can deduce that a message to `<code>bob@example.org</code>` must have failed.

This example shows the simplest possible method of matching a VERP to a list subscriber: the entire recipient address is included within the return path, with the at sign replaced by an equals sign because a return path with two at signs would be invalid. Other encoding schemes are possible.

### 14.7 VERP

- [[ezmlm]]
- [[GNU Mailman]]
- [[exim]], using a specialized Router/Transport combination
- [[Moodle]]
- [[postfix (software)]]
- [[qmail]]
- [[Sendmail]], with a ruleset.
- [[Mercury Mail Transport System]]
- [[Zimbra]]
- [[StrongMail]]
- [[STEdb]]
- [[Courier Mail Server]]
- [[Sympa]]
- [[mlmmj]]

### 14.8

The use of VERP requires each message to be sent once for every recipient, instead of once to each receiving [[SMTP]] server. This is because of a limitation of SMTP, which allows multiple recipient addresses to be specified in a single transaction, but only one sender address. When there are many subscribers in the same [[domain name]] domain, a mailing list that is not using VERP can combine multiple deliveries into a single transaction. It connects to the appropriate [[Server (computing)]] server] for the domain, gives the single sender address, the recipient addresses, and then sends the message contents only once.

A mailing list using VERP, on the other hand, must send the entire message body repeatedly, which leads to an overall increase in [[Bandwidth (computing)]] usage. This inefficiency is usually not considered a big problem, especially by [[qmail]] users, since qmail always sends messages once per recipient, even when VERP is not being used. Some packages mitigate the impact of VERP by applying it selectively, for example a mailing list manager might only use VERP on 1 in 10 mailings. This way you can gain much of VERP’s tight bounce control and accurate feedback without incurring the processing and network overhead every time.

Another problem with VERP (and with any automatic bounce handling scheme) is that there will always be some recalcitrant [[Mail transfer agent]] MTA]s on the Internet that fail to follow even the simplest standards. VERP depends on the recipients’ MTAs following the rule that bounces are sent to the [[envelope sender]]. This has been a
standard requirement since the dawn of SMTP in 1982 (see RFC 821), but still there are MTAs that get it wrong, usually by bouncing to the address in the `<header>From:</header>` ([header (information technology)]) (See [http://jamesthornton.com/writing/imail-envelope-sender.html Imail] for an example).

Systems that implement [[greylisting]] work fine with VERP if the envelope sender follows the above mentioned format. However, some VERP implementations use message number or random key as part of VERP, which causes each post to the mailing list to be delayed unless the greylisting system treats “similar” sender addresses as being equivalent.

### 14.9

- [[Bounce message]]
- [[Bounce Address Tag Validation]] (BATV) - for bounces from [[backscatter (e-mail)]]
- [[Sender Rewriting Scheme]] (SRS) - for bounces from e-mail forwarding and [[Sender Policy Framework|SPF]]
- [[Simple Mail Transfer Protocol]] (SMTP)
CHAPTER 15

Indices and tables

• genindex
• modindex
• search