How to set up a mail server on a GNU / Linux system

Step by step guide to install Postfix

Ubuntu + Postfix + Courier IMAP + MySQL + Amavisd-new + SpamAssassin + ClamAV + SASL + TLS + Roundcube + Postgrey

Easy to follow howto on setting up a mail server with unlimited users and domains, with IMAP/Pop access, anti-spam, anti-virus, secure authentication, encrypted traffic, web mail interface and more.

Based on an Ubuntu distribution platform, but instructions are distro generic. Examples are run on Amazon AWS ec2, but only for demonstration purposes.

12th edition
Author Ivar Abrahamsen
License: Respect (CC by-sa)
Last Update: 2014-10-28
Contact / Discuss / Contribute

Contents

I More

Search

Return to top.

Editions

Edition	State	Started	Updated	Description
1st	Released (outdated)	2004-01	2004-02	Based on Mandrake 9.1.
2nd	Released (outdated)	2004-02	2004-07	Based on Mandrake 10.x. Very thorough with advanced server sections.
3rd	Released (outdated)	2005-05	2005-11	Based on Ubuntu 5.04, Hoary Hedgehog. Now includes SASL & TLS integration.
4th	Released (outdated)	2005-10	2005-12	Based on Breezy Badger, Ubuntu 5.10. Includes Postgrey.
5th	Released (outdated)	2006-05	2006-11	Based on Ubuntu 6.06 LTS, Dapper Drake.
~~6th~~	Scrapped	2006-11	2007-10	Was to be based on Edgy Eft, Ubuntu 6.10 or 7.04. include Domain Key signing. include my mail admin or my catchall aliases admin.
7th	Released (outdated)	2008-04	2009-06	Updated, based on Ubuntu 8.04 LTS Hardy Heron. Using Amazon EC2 as example. (Tested with 8.10 & 9.04 as well)
8th	Released (outdated)	2009-05	2009-11	Based on Ubuntu 8.10 (intrepid), then tested with 9.04 (jaunty) & 9.10 (karmic) as well. Using official Ubuntu ec2 as examples.
9th	Released (outdated)	2009-11	2010-05	Based on Ubuntu 9.10 (karmic) using Canonical's cloud images. Added Roundcube webmail option.
10th	Released	2009-12	2013-01	Based on Ubuntu 10.04 LTS (lucid) using Canonical's cloud images. Tested on 10.10 (maverick). Tested on 11.04 (natty)
11th	Released	2012-11	2014-05	Based on Ubuntu 12.04 LTS (precise). Tested with 12.10 (quantal) and 13.04 (raring)
12th (this)	Released	2014-05	2014-10	Based on Ubuntu 14.04 LTS (trusty).

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

Return to top.

Introduction

Aim

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.

Return to top.

Research

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.

Donate

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?

UK

US

EU

Return to top.

Software

What software packages have/will I use and why.

- **OS: 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.

- **MTA: Postfix**

  www.postfix.org

  Simple, free and slick. Yup I am a sucker for anything that works easily. Postfix is powerful, well established, but not too bloated, and is security concious from the start.

- **Pop/IMAP: Courier IMAP**

  www.courier-mta.org/imap/

  My first mail server installation was with Courier. I have not found a reason to change this as again it is simple, and free.

- **Database: 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.

- **Content Check: Amavisd-new**

  www.ijs.si/software/amavisd/

  Easy plug in solution for spam, virus checking etc.

- **Anti-Spam: SpamAssassin**

  spamassassin.apache.org

  Powerful renowned spam fighting tool.

- **Anti-Virus: ClamAV**

  www.clamav.net

  Free virus scanner that can be trusted and includes update daemon.

- **Authentication: Cyrus SASL**

  www.imc.org/ietf-sasl/

  Secure and trusted cryptography technology for authentication of SMTP traffic.

- **PostGrey**

  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 cant afford to do. When proper servers try again after a few minutes it lets it through.

- **Encryption: TLS**

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

  Secure and trusted cryptography technology for encryption of SMTP traffic. Not too 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

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

  www.roundcube.net

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

- **Platform: 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.

Return to top.

Installation

Distribution

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 documention 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.

Base Install

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.

If you have chosen an ec2 based server you should follow my ec2 suggestions first.

I strongly suggest choosing the latest LTS version of Ubuntu, not the versions in between. Once this is set up you will tinker very little with it, and it will quickly be annoying to upgrade distributions once a year.

Ps. 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'.

Repositories

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

I would recommend find a repository archive close to your server's location. For example a country specific one or if hosted on *AWS EC2* an archive in your AWS region. Remember these are highly security sensitive so choose one you trust.

You need the *main* and *universe* repositories. The *multiverse*, *restricted* and *partner* can be added but are not needed. Do not add *backports*.

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

Uncomment the lines that have commented out *universe*. E.g. here are mine for ec2 in Europe:

```
deb http://eu-west-1.ec2.archive.ubuntu.com/ubuntu/ trusty universe
deb-src http://eu-west-1.ec2.archive.ubuntu.com/ubuntu/ trusty universe
deb http://eu-west-1.ec2.archive.ubuntu.com/ubuntu/ trusty-updates universe
deb-src http://eu-west-1.ec2.archive.ubuntu.com/ubuntu/ trusty-updates universe
```

```
deb http://security.ubuntu.com/ubuntu trusty-security universe
deb-src http://security.ubuntu.com/ubuntu trusty-security universe
```

Note the *security* repository always have to go to the non-mirrored server.

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

Packages

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.

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

Secondly update your current system:

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

Note: *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.

Additional packages

I also install a few other packages that I personally prefer. But nothing todo with the mail server.

```
sudo apt-get install vim lynx curl git
```

Mutt is a very usefull command line mail client that I always install but I usually do that at the end when testing so that it doesn't install its dependency on Postfix before I am ready.

```
sudo apt-get install mutt
```

Package status

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

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

And to find which are available:

```
apt-cache search postfix
```

Return to top.

Configuration

- **Core/Simple**
  - Firewall (Shorewall)
  - MTA (Postfix)
  - Database (MySQL)
  - Pop/IMAP (Courier)
- **Advanced**
  - Content Checks (amivisd-new)
    - Anti-Spam(SpamAssassin)
    - Anti-Virus (ClamAV)
    - Policy Check (PostGrey)
- **Secure**
  - Authentication (SASL)
  - Encryption (TLS)
- Webmail (SquirrelMail)
- Administration (phpMyAdmin)
- **External**
  - Domain name
  - DNS

Simple mail server

Now lets configure a simple mail server using some of the packages installed.

Firewall

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.

Installation

```
sudo apt-get 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.

Configuration

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 only

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

First configure which network adapters we are accessing the net.

```
sudo cp /usr/share/doc/shorewall/default-config/interfaces /etc/shorewall/;
sudo vi /etc/shorewall/interfaces
```

```
net     eth0            detect          dhcp,tcpflags,logmartians,nosmurfs
```

Then we will configure network zones

```
sudo cp /usr/share/doc/shorewall/default-config/zones /etc/shorewall/;
sudo 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 wanto specify what is your home IP etc.

```
sudo cp /usr/share/doc/shorewall/default-config/hosts /etc/shorewall/;
sudo vi /etc/shorewall/hosts
```

```
# loc   eth0:192.168.0.0/24
```

Then set what is the default policy for firewall access.

```
sudo cp /usr/share/doc/shorewall/default-config/policy /etc/shorewall/;
sudo 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.

```
sudo cp /usr/share/doc/shorewall/default-config/routestopped /etc/shorewall/;
sudo 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*.

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

```
SSH/ACCEPT      net             $FW
```

Please note SSH is these days a known attack vector for brute force attacks. Especially if you allow connection from anywhere on the internet and on the standard SSH port (22).

Open for business

Once your server is working come back to this step and open up SMTP and Web access to others.

```
vi /etc/shorewall/rules
```

```
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.

Return to top.

Database

MySQL

Installation

```
sudo apt-get 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***

Configuration

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

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

Then run this commands to create the tables:

```
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(1) NOT NULL default '1',
`change_password` tinyint(1) NOT NULL default '1',
`clear` varchar(128) NOT NULL default 'ChangeMe',
`crypt` varchar(128) NOT NULL default 'sdtrusfX0Jj66',
`quota` varchar(255) NOT NULL default '',
PRIMARY KEY  (`id`),
UNIQUE KEY `id` (`id`)
) ;
```

```
# 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:

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

In previous version you needed to comment out this line

```
#skip-networking
```

However in today's file the default is to bind the address to localhost, which is fine.

```
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:

```
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
```

Return to top.

MTA

Postfix

Installation

```
sudo apt-get 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.

Configuration

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.

First specify the name of your server.

```
# 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.

```
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 sometimes send via my ISP's server, so it has to worry about the queuing 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.

```
# leave blank to do it yourself
relayhost =
```

```
# or point it to an accessible smtp server
relayhost = smtp.yourisp.com
```

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

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

I currently restrict my servers to ip4 and avoid ip6 at the moment. (Or rather avoid flooding my logs with ip6 errors). Hopefully soon enough servers that they interact with will support 6 and I can support both.

```
inet_protocols=ipv4
```

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.

```
# 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.

```
local_recipient_maps =
mydestination =
```

Then will set a few numbers.

```
# 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.
# some have 3 days, I have 16 days as I am backup server for some people
# whom go on holiday with their server switched off.
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.
# effective stopper to mass spammers, accidental copy in whole address list
# but may restrict intentional mail shots.
smtpd_recipient_limit = 16
# how many error before back off.
smtpd_soft_error_limit = 3
# how many max errors before blocking it.
smtpd_hard_error_limit = 12
```

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

```
# 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
# 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
```

Note: If using Amazon ec2 you may want to move the mail spool to /mnt or an EBS location. You will need to symlink correctly afterwards.

Return to top.

Postfix's MySQL configuration

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.

Return to top.

Pop/IMAP

A normal mail server will need IMAP, however if your server is a backup or redirection only server then Courier is not needed.

Courier IMAP

Installation:

```
sudo apt-get 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
```

Return to top.

Summary

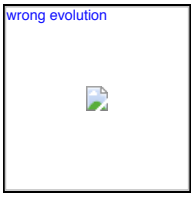
You now have a basic mail server!

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 judicously
- 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 proceding

Return to top.

Advanced mail server



Somewhere, something went terribly wrong

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

Content Checks (Anti spam & anti virus)

Amavisd-new

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

Install amavids-new

```
sudo apt-get 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/amavisd*. They are now spread across several files in *conf.d*. Debian and Ubuntu defaults are now very sensible and spread into seperate 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 dont change anything in it.

Edit content check file

```
sudo vi 15-content_filter_mode
```

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

```
# #@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 dont change anything in them.

25-amavis_helpers defaults are fine.

30-template-localization defaults are fine.

Edit user file

```
sudo vi 50-user
```

In the middle insert:

```
@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 incomming email. Next we will setup postfix to talk to amavis.

```
sudo 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
        -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
```

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 proced to uncomment the anti virus and anti spam lines in

```
sudo vi 15-content_filter_mode
```

```
@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);
```

But do that after the next section (SpamAssassin).

When things are working we will turn down logging level, and start bouncing/discarding spam.

```
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
```

Return to top.

Anti-Spam

SpamAssassin

Installation:

```
sudo apt-get 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.

```
sudo vi /etc/default/spamassassin
```

```
ENABLED=1
```

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

```
sudo vi /etc/spamassassin/local.cf
```

Return to top.

Anti Virus

ClamAV

Installation:

```
sudo apt-get install clamav clamav-base libclamav6 clamav-daemon clamav-freshclam
```

(Earlier vesions 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
```

Enable scanning by ClamAV of amavis' temporary files.

```
sudo adduser clamav amavis
```

Return to top.

Postgrey

Installation:

```
sudo apt-get install postgrey
```

The default config of postgrey is okay. However you need to tell Postfix to use it.

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

And then edit the 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
```

You can tweak whitelisting in ***/etc/postgrey***. You can tweak postgrey configuration by tweaking ***/etc/default/postgrey***. E.g. delay, auto whitelisting, or reject message.

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

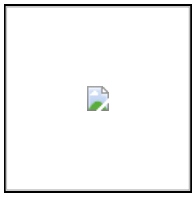
Return to top.

You know have an advanced mail server. You can use this, but Id recommend continuing. However this is a good point to test the set up so far and to insert some data in the db.

Return to top.

Secure mail server

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



No, I will not fix your computer

Authentication

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:

Installation

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

Configuration

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:

```
# Add permit_sasl_authenticated to you existing  smtpd_sender_restrictions
smtpd_sender_restrictions = permit_sasl_authenticated, permit_mynetworks,
                warn_if_reject reject_non_fqdn_sender, reject_unknown_sender_domain,
                reject_unauth_pipelining, permit
# Add permit_sasl_authenticated to you 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:

```
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:

```
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.)
(*Note*: While *sql_passw* is the original parameter name (without the *d*), a more obvious *sql_passwd* will also work in later versions)

Tell the pam how to to authenticate smtp via mysql:

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

These must be on 2 lines only, but I have broken them up for easier to read.

```
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 usefull to tail the auth.log as well when testing SASL.

```
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 avaiable 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 IDLE"
```

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

Return to top.

Encryption

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 beetwen 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 = btree:${data_directory}/smtpd_scache
# smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
smtp_tls_CAfile = /etc/ssl/certs/ca-certificates.crt
```

Next we have a look at the master.cf file.

```
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 *smtps* service (port 465), for some compatebility with some older clients (outlook express etc).

```
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 body_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.

```
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

```
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.

HTTPS

You probably also want to insist on https connections over tls if you below add webmail that are exposed to the public. Securing a web server is out of scope for this howto, but will not be a lot different than the mail server tls settings.

You know have an advanced secure mail server. Now is another good point to test the set up so far and to insert some data in the db.

Return to top.

Webmail

Enable web access

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

Alternative: 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.

Roundcube webmail client

To install Roundcube

```
sudo apt-get 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 symblink 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:

```
$rcmail_config['default_host'] = 'ssl://localhost';

$rcmail_config['default_port'] = '993';

$rcmail_config['imap_force_ns'] = true;

$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:

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

But perhaps concentrate on getting the basics working firtst...

Save, exit and reload Apache to enable these aliases for Roundcube to work

```
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

If you have enabled session encryption then also enable the *mcrypt* library

```
sudo ln -s /etc/php5/mods-available/mcrypt.ini /etc/php5/apache2/conf.d/20-mcrypt.ini
```

You can obviously modify and tweak further.

- One thing that may be useful is to have the Roundcube Apache *Alias* on different virtual hosts.
- Other is to 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
- For security enforcing https for the webmail is probably smart.
- Adding your own skin logo etc to customise the look is likely. *($rcmail_config['skin_logo'] = 'yourlogo.png';)*
- If you webmail is not on the same server as your MTA (Postfix), then the MTA might not have the webmail server's IP listed in permit_mynetworks, so you might want to log into the smtp when sending email:

  ```
  $rcmail_config['smtp_user'] = '%u';
  $rcmail_config['smtp_pass'] = '%p';
  $rcmail_config['smtp_auth_type'] = 'login';
  ```

-

More details on the Roundcube Wiki.

Return to top.

Administration

Enable web access

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

Install phpmyadmin

```
sudo apt-get 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 spefic virtual host. If so you need to, edit

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

and comment out the alias.

```
#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.

```
sudo apache2ctl -t
```

Then reload it.

```
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 ubfuscating, but simple.

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

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

Return to top.

External changes

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

Domain name

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.

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.

Note: 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.

Return to top.

Data

Add users and domains

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.

Required domains and users

First the required domains for local mail

```
# 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');
```

Then a root user.

```
INSERT INTO users (id,name,maildir,crypt) VALUES
        ('root@localhost','root','root/',encrypt('apassword', CONCAT('$5$', MD5(RAND())))  );
```

Note: this uses the encrypt function with a random salt per user and prefixed with 5 which instructs it to use SHA-256 encryption hashing.

You can alternatively use the plain encrypt('apassword') function, however it is then unsalted and only considers the first 8 character of the password.
This may be required if you use other software that needs to interact with the users authentication. However most will support the SHA-256 crypt() call.

Domains and 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');

INSERT INTO aliases (mail,destination) VALUES
        ('@lala.com','@whupper.nu'),
        ('@mail.blobber.org','@blobber.org'),
        ('postmaster@whopper.nu','postmaster@localhost'),
        ('abuse@whopper.nu','abuse@localhost'),
        ('postmaster@blobber.org','postmaster@localhost'),
        ('abuse@blobber.org','abuse@localhost');
```

You also have two users called *"Xandros"* and *"Vivita"*.

```
INSERT INTO users (id,name,maildir,crypt) VALUES
        ('xandros@blobber.org','xandros','xandros/',encrypt('apassword', CONCAT('$5$', MD5(RAND())) ),
        ('vivita@blobber.org','vivita','vivita/', encrypt('anotherpassword', CONCAT('$5$', MD5(RAND())) );

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

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 loggin 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 alises 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.
- Postfix then looks up aliases and searches for a row where the mail field matches *"john@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.

Adding template

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

```
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:

```
INSERT INTO users (id,name,maildir,crypt) VALUES
        ('email@address','short description','foldername/',encrypt('password', CONCAT('$5$', MD5(RAND())))  );
INSERT INTO aliases (mail,destination) VALUES
        ('email@address','email@address');
```

Return to top.

Common 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

```
#Remember some might be disabled
SELECT dom.domain
FROM domains dom
LEFT JOIN aliases al
        ON CONCAT( '@', dom.domain ) = al.mail
WHERE al.mail is null
OR al.enabled = 0
ORDER BY dom.domain ASC
```

Find aliases for an invalid domain

```
SELECT al.*
FROM aliases al
LEFT JOIN domains dom
        ON dom.domain = SUBSTRING(al.mail,LOCATE('@',al.mail)+1)
WHERE dom.domain is null
OR dom.enabled = 0
ORDER BY al.mail ASC
```

Find all non local destination aliases

```
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
```

Find all aliases for a certain domain

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

Find all 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
```

Return to top.

Test

Common problems

- **Missed a step**

  If you mistakenly or intentially skipped past sections, you may have missed an important step in your configuration, which my guide pressumes 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 incompatibilties or confusion will arrise.

- **Distrobution/version differences**

  If you run a different version or even distrobution 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 thouroughly 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.

Check the FAQ for common specific problems.

Return to top.

Test strategy

What steps to think of when testing.

Test early and frequently

It is very helpfull 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 mudded 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 immidietly spot issues.

Isolate the problem

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.

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?

Simplify the system

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

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

Return to top.

Tail, tail and tail again

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:

```
tail -f /var/log/mail.log
```

And in another window:

```
tail -f /var/log/mysql/mysql.log
```

In a third or more do your actual configuration or testing.

Return to top.

Switch off services

previous edition 1

previous edition 2

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

Return to top.

Switch debug on

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;
```

Return to top.

Telnet is your friend

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

1. First you test it on the server to exclude firewall and network issues.
2. Then you test it from another machine to simulate an actual other mail server.
3. 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.

Return to top.

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
> 354 End data with <CR><LF>.<CR><LF></LF></CR></LF></CR>
# 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!

Return to top.

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></CR></LF></CR>
# 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!

Return to top.

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 '^]'.
* OK [CAPABILITY IMAP4rev1 UIDPLUS CHILDREN NAMESPACE THREAD=ORDEREDSUBJECT THREAD=REFERENCES SORT
  QUOTA IDLE ACL ACL2=UNION STARTTLS LOGINDISABLED] Courier-IMAP ready. Copyright 1998-2008
  Double Precision, Inc.  See COPYING for distribution information.
```

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.

Return to top.

Initialize

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

Stop services

First stop services so they wont accidentally do something.

```
sudo /etc/init.d/postfix stop
```

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

Restrict firewall

Check what the firewall rules are.

```
vi /etc/shorewall/rules
```

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

Change passwords

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.

MySQL passwords

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
```

Check configurations

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

Set machine name

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
```

Certificates

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 authenticiy is important. Follow the TLS certificate instructions for Postfix and Courier.

Start and test services.

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.

Insert data

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';
```

Open firewall

Then open up the firewall, follow the world access bit in the firewall configuration. Voila. Up and running. Well we hope.

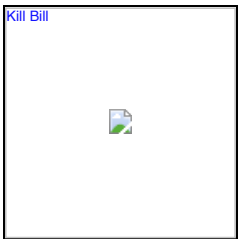
Return to top.

Extend

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 powerfull customized version.

- Remote MX mail backup



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

Remote MX mail backup

With MX backup loosing 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 patience, 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 backuing 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` tinyint(1) NOT NULL default '1',
        PRIMARY KEY  (`pkid`),
        UNIQUE KEY `domain` (`domain`)
);
```

Then still on the backup server, edit main.cf and add these:

```
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 addres, hence if you do not have a very static IP address, then you need some way of automatic edit the postfix main.cf file.

```
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

```
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

```
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.

```
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 should 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.

Relay recipient lookup

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:

```
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 specificed 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');
```

Return to top.

Local file backup

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.xxx.tgz schema.sql data.sql
tar cf mail.xxxxx.tar  mail-*.xxxxx.tgz
```

You may combine a full backup with a intermediate update of what has changed recently only.

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

Return to top.

Sender ID & SPF

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 include:_spf.google.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.

Return to top.

Spam reporting

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

Return to top.

White/Black Lists

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. SpamAssissin also as a feture to automaticly learn white lists.

Return to top.

PGP & S/MIME

Adding support for GnuPG and S/MIME increases indiviual 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)

Return to top.

Relocation notice

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, frst 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.

Return to top.

Pop-before-SMTP

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

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

Return to top.

Admin software

Trying out a few admin software might make you life easier, if command line or phpMyAdmin gets to crude. Quick search for postfix admin will find many options.

I have made an application Sorting Office, github.com/flurdy/sortingoffice. It is for managing data in the database(s), ie domains, users, aliases etc and not for configuration. You can look at a demo at sortingoffice-demo.herokuapp.com.

Return to top.

Auto Reply

todo

Postfix have now features to auto reply to an email, while still delivering it to its alias.

Return to top.

Block Addresses

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 indivdual addresses.

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

Return to top.

Throttle Output

todo

For some users with restrictions on bandwidth, you may wish to control how much mail is sendt out. Postfix has long refused to implement these features, out of ideolocial beliefs that mail servers should not be restricted. However there are some ways around this. More to come later.

Return to top.

Mail Lists

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 mailling list, it can be implemented by simply seperating aliases in the destination field in the aliases table with a comma.

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

Return to top.

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.

Why

- 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.

How

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):

```
insert into backups (domain,transport) values
                ('joe@bloggs.com','smtp:[aspmx.l.google.com]:587'),
insert into domains (domain,transport) values
                ('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', CONCAT('$5$', MD5(RAND())) );
```

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:

```
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. On the latest Ubuntu releases this is no longer needed as it is already included.

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

```
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:

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

Or install a tiny util:

```
sudo apt-get install tofrodos;
fromdos Equifax_Secure_Certificate_Authority.cer;
```

Put it into your certificate root folder:

```
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:

```
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. Including Google's spfs in your domain's dns txt settings should cover it.

```
include:_spf.google.com
```

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 is only an issue if not all aliases you have use Google Apps.

One possible solution is to forward all emails for the people that do not want to use gmail to another domain hosted by you that then again on your server aliases to the original email address.

Return to top.

Maildrop, spam folder and vacation messaging

Villu have documented swapping in Maildrop for virtual transport and automatically delivering spam to a spam folder. (And links to a post about vacation messaging)

Please read his post here.

Return to top.

Squirrel Mail

Using among others the https://help.ubuntu.com/community/Squirrelmail as an updated reference.

You need to copy a SquirrelMail configuration to apache.

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

And enable with this:

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

Or as Florent recommends, use:

```
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 wont 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.

```
courier
```

Now they say using TLS over localhost is a waste of time. But I do anyway. Type ***7*** to edit secure IMAP. Type ***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.

Return to top.

Brute force

Preventing Brute Force attack on your server.

First line of defence is the firewall. If they cant get to your server then they cant hack in. However to be a useful internet based server you have to expose some services, e.g. SMTP.

However SSH and IMAP can be restricted. You should limit it to your own IP ranges only. Another trick is to not use the standard ports.

This however can cause issues with tools expecting it to be on the standard port and which can not be configured. Or if the SSH port is not documented and standardised in your organisation the correct port may be forgotten.

To add a new SSH port to the firewall rule:

```
sudo vi /etc/shorewall/rules
```

```
# We will keep the old port here until we can safely switch it off
SSH(ACCEPT)     net             $FW
# Please pick another port number than 1092!
ACCEPT          net             $FW             tcp             1092
```

```
sudo service shorewall check
```

```
sudo service shorewall safe-restart
```

Open the SSH configuration:

```
sudo vi /etc/ssh/sshd_config
```

And change the port from 22

```
Port 1092
```

```
sudo service ssh restart
```

Now from another machine test if the new port works

```
ssh -p 1092 yourusername@yourserver
```

We can then remove the old port from the firewall rules or add it as reject.

```
sudo vi /etc/shorewall/rules
```

```
SSH(DROP)       net             $FW
```

```
sudo service shorewall check
```

```
sudo service shorewall safe-restart
```

To simplify your life add the new port to your personal SSH configuration.

Note if you have physical firewall in front of your mail server you will need to update it as well. E.g. if you use ec2 you will need to add the new port to the security group and remove the old port.

I would suggest IP range restrictions to be on the external firewall to avoid a maintenance hell of updating x servers if a new range needs adding/modifying.

Below are two widely used ways to protect yourself further.

DenyHosts

Deny hosts is an effective tool to protect your SSH service from brute force attack. However DenyHosts has now been removed from the main repositories due to lack of updates.

To install on an older version of Ubuntu:

```
sudo apt-get install denyhosts
```

Tweak in */etc/denyhosts.confs*. 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.

fail2ban

fail2ban protects against a multitude of brute force attacks. Relevant to this guide is the protection for SSH, SMTP, MySQL and IMAP.

```
sudo apt-get install fail2ban
```

Follow this guide for how to configure it.

Return to top.

Suggestions?

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, Id prefer that you write down the extension, and let me know the link! :))

Return to top.

Elastic Compute Cloud

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. It could do with some updating.

How I use it for my mail servers

Different images to launch for different needs. One for plain MX backup, one just for aliases, one to store email only and provide imap, one just for webmail. Good way to also 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.

Instance size

I run my own instances on several AWS ec2 micro size as my traffic is low. However the memory is too low so I also add an 2GB EBS as swap file, detailed in my Ubuntu ec2 guide.

However for a medium company I recommend the small size. High traffic servers needs perhaps larger instances.

Security and backup

I also recommend using EBS for slight recovery of the mail spools if the server goes down.

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 web server 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 instructions: 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 AWS do have ec2 based email server limitations, so if you have a busy mail server, follow their FAQ entry for removing mail throttling. AWS will then also add a reverse IP lookup for you elastic IP to your server, which also helps in the spam scoring and delivery.

Amazon EC2 Images: AMIs

I used to provide ec2 AMIs. But these quickly got outdated so I have removed them.

Vagrant

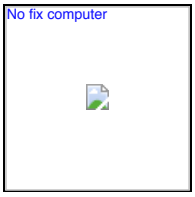
One very useful way to test images without the cost of ec2 is to use Vagrant locally first. This way you could potentially automate your fabric scripts, Puppet, etc as well.

EC2 Links

- Amazon web services (AWS)
- Elastic Computing Cloud (EC2)
- Simple Storage Service (S3)
- AWS Cost Calculator
- EC2 Resource Centre
- EC2 Starter Guide
- EC2 Firefox extension: Elasticfox
- Elasticfox for Firefox 3
- S3 Firefox extension: S3Fox
- EC2 to S3 Admin Scripts
- alestic, ubuntu ec2 images
- Ubuntu ec2
- Ubuntu Cloud
- Ubuntu ec2 Starter Guide
- My ec2 Tips and Howtos

Return to top.

Appendix



About author

Ivar Abrahamsen, an IT Consultant from Norway currently living in Hampshire, UK. Specialising in leading teams that integrate middleware applications, payment systems and data processing systems using mainly Java and Scala technology stacks.

Return to top.

Contact

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.

Forums

The internet is full of people much more knowledgeable than me. And definitely more likely to reply.

- **Here is a forum thread on this specific mail server howto**.
- **And another one by me which is also used.**.
- **Alternatively many solutions are available on Server Fault.**
- **If more Postfix specific, considered the postfix mailing lists.**

If you find or know of other sources then let me know.

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 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 Civilization in my youth 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"**.

Checklist

Before posting in the forums, ServerFault, etc, 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 FAQ**?
- Read the previous posts in the forums and archives for solutions already found? And at least some forum searching?
- Tailed the mail.log? It usually **tells you what the problem is**!
- Tailed the mysql.log? If nothing happens there it should indicate something...

When you do post remember to include a short dump of the mail.log.
(Remember to anonymise the server names, email addresses & definitely passwords!)

Return to top.

Consultancy and advice

I do run a consultancy service, eray by flurdy. However I do not offer email server consultancy, as it is not really my expertise nor interest despite this document. It was a long time ago when I write most of this stuff.

I do though offer a complete AWS based email server set up by me. (price)

Contact me

- 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 or at least others know.
  It may be a known problem, or simple difference of technical opinion, but better to surface more than miss some.
- If you find any spelling mistakes or broken links, please let me know.
  Even after 10 years since the first version people are still finding typos every week.
- Any technical difference of opinion, please use the forum.
- Please do **not** contact me if you have set up problems. I am terrible procrastinator of replying to all emails, even the very polite ones where I might even know the simple solution to. :(
  Note that:
  - 90% of all emails I receive are solved by testing properly. Or by applying the common solutions.
  - A further 9.9% are problems already solved in the forums.
  - 0.01% are real problems I then forget to respond to.
- You can contact me via flurdy.com/contact as long as you are sure it complies with the *fishing* analogy.
- **Thank you messages** are very appreciated however! It makes my day! :)

Many people contribute with really good stuff in the forums, send me typos (frequently), or thank you notes. You guys are great!

Return to top.

Contribute

Ways to contribute towards this document:

Participate

Participate and respond to questions in forums, such as the Ubuntu ones, or specific sites for Postfix, IMAP, etc.

Correct typos

If you spot any typos either contact me about it, or preferably fork github.com/flurdy/flurdy.com-docs and send me a pull request.

Minor enhancements

If you any minor changes that will improve this document, or update it with a more update to date information: Please fork github.com/flurdy/flurdy.com-docs and send me a pull request.

Larger enhancements

For extensive changes to this document, please contact me about it first. Together we may then decide:

- it is a good idea, and you should send me a pull request
- or it is independent enough to host it elsewhere and just send me a pull request with the link instead
- or decide it is not beneficial and have a beer instead

Related HOWTOs

If you have written a document yourself or found a very relevant one, instead of including it inside this document I prefer to just link to a version hosted by you instead.

Please contact me about your document or even better send me a pull request with the link added to this document.

If you require assistance with hosting it, please contact me for suggestion or assistance.

Return to top.

Why

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 dependent on 3rd party providers, limited by their technology constraints 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.

And its your data. You and your data is not a product of another company. In theory should make your data more secure and keep your privacy.

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 helpful 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 one mail server for a myself and few friends and colleagues. Soon I was getting more request for more servers, 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 thousands of hits and lots of emails. (blessing in disguise)

Return to top.

References

- 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

New references

- Postfix TLS
- Postfix main.cf doc
- saslauthd
- Bypassing amavisd
- Ubuntu Help: Squirrelmail

Return to top.

Software Links

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

Change log

Brief list of latest changes.

- 2014-08-06: Added Sorting Office link
- 2014-07-25: Tested every step with Ubuntu Trusty 14.04, and updated differences.
- 2014-05-31: Removed ec2 AMIs
- 2013-06-30: Added contribute section.
- 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.
- 2010-06-07: Updated for Ubuntu 10.04 LTS Lucid Lynx Modified mysql log option. Removed dynamic uid & gid in postfix.
- 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.

Return to top.

Todo

- Spell check!
- Remove uid and guid

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

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 you can not log into squirrelmail as the email folders will not yet exist. When receiving their first email, postfix will create all the necessary 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 necessary. 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...

Other's solutions

In addition to the 100s of solutions in the forum threads, some have published their issues and solutions as well:

- Sandro's Postfix MTA configuration problems and solutions Ubuntu 12.04

Return to top.