Mailserver på Linux (postfix/dovecot)- inkl Spamassassin og Squirrelmail med sasl

Denne guide er skrevet til Linuxfolket der i forvejen har de tekniske foranstaltninger til at kunne læse og forstå denne guide!

Jeg har i sin tid skrevet denne guide på engelsk- så derfor er den også skrevet på dette sprog her!

Skrevet den **04. jan 2011** af **peque** I kategorien **Server / Linux** | ★★★★☆

Installing the Linux is like every other linux installation, but if you haven't tried before, this can be a little tricky in most cases.
Start by downloading the image that fit's into your Computer here:
http://cdimage.ubuntu.com/releases/hardy/
Here you can find all kinds of images for all kinds for all kinds of processors.
After burning the image out on a CD - You're ready to start the installation.

Boot on the CD-rom we just downloadet and burned.
Choose your language that you want to be set as default.
Aterwards you can setup software RAID - if you haven't got a RAID controller onboard.
Normally you'll need to run RAID1 - for mirrorred harddrives - so you'll get the best data security - if you haven't got the options to run RAID6.

The next part is to setup the harddrive - Normally I'll do this:

128mb	/boot	Start partition
4096mb	SWAP	SWAP partition - normally twice the amount of RAM
40GB	/	Root partition for the ROOT system
Rest of the disc	/storage	The place where we'll store the mail!

Normally you'll can make a directory - just for mail - which make it easiere in backup cases - which we'll also do here.
The next is setting up the networking etc (which will be a good place to setup bonding - later on, for making the same IP on both Interfaces - if you got that) .
After starting out with the default question about user settings - etc - we'll be asked for pre-defined services on the server, where we'll need these services:

LAMP	- Linux - Apache - MySQL - PHP
Mailserver	- Postfix - Dovecot
OpenSSH	- OpenSHH - remote administrating

These are the standard services we're needing to build our Mailserver, in the end we're hoping to get it with IMAP, SASL and Dovecot.
So afterwards setting the machine services up - you need to finish the install and reboot. Afterwards log in to the machine, edit /etc/apt/sources.list and remove the # in front of the lines in the buttom the activate all reposities with packages for Ubuntu.
Starting setting up the server:
For setting up the mailserver we need to edit a lot of setups so let's get started with the Apache/PHP/MySQL setup.

For beeing able to see our mailserver from the internet - we're needing to start some where - and we do that with Apache.
Our normal web-directory-root is /var/www which in this case are OK to start out with.
For testing cases we need the following ports to be opened to beeing able to recieve and send mails etc.

Ports:
22 - SSH
25 - SMTP
80 - HTTP
993 - IMAPS
995 - POP3S

So this will need to be forwarded through the network to the server.
In our testcase - we'll use the domain domain.dk for our mailtest domain. This is set at http://www.GratisDNS.dk to point at our public IP - and through the firewall routed towards the testserver.
Be sure THIS is working otherwise we cannot get anything else to work!
In Our testcase here - we'll use:

Domain name:	mail.domain.dk
IP-Address:	123.123.123.123

Apache:
Editing the apache2.conf - remember to add the default ServerName like this:

```
ServerName    webmail.domain.dk
```

After we're getting this to work - We'll need to get the administrating interface to Webmail/mailusers - postfixadmin. So in the root dir - do this:

```
# wget http://switch.dl.sourceforge.net/sourceforge/postfixadmin/postfixadmin-2.2.0-rc3.tar.gz
```

Then unzip it:

```
# tar -xvzf postfixadmin-2.2.0-rc3.tar.gz
```

And last move it to the Apachedir:
```
mv postfixadmin-2.2.0-rc3 /var/www/mail
```

Now go to the dir and edit the DATABASE_MYSQL.TXT file to fit your need regarding to passwords etc.
Now you're ready to import the file into the DB:

```
# mysql -p < DATABASE_MYSQL.TXT
```

Now the Database are build to fit into MySQL.

Now we should edit apache to fit it all. This will be in the /etc/apache2/sites-enabled/000-default.conf and set the root dir to /var/www/mail instead - so the webserver's home dir is in the maildir.
The next part is to edit the /var/www/mail/config.inc.php to fit into your needs - and afterwards remember to delete setup.php - AFTER setting the new superadmin!!!!!
Now your ready to try point a browser to your local server!

We'll make some changes here later on, so this is just for the setup and test!!!

Mysql
For beeing able to see the different request in the databasefile - Please remove the # in front of the

following line in /etc/mysql/my.cnf:

```
#log            = /var/log/mysql/mysql.log
```

And also if you want your MySQL server to be able to accept request from other machines. You can edit the line Bind! But for stopping filling up your /var/log/ then remember after setting it all up - to undo those setting

OpenSSL (Certificat)
You'll need this for making a secure connection between the client and the server - so we'll make our own in first case. For that case we'll need to make a Cacert first and afterwards sign the certificate to make it work on the server.

```
cd /tmp
openssl genrsa -des3 -rand /etc/hosts -out smtpd.key 1024
```

Sign the cert request:

```
openssl req -new -key smtpd.key -out smtpd.csr
openssl x509 -req -days 3650 -in smtpd.csr -signkey smtpd.key -out smtpd.crt
```

Sign out the certificate:

```
openssl rsa -in smtpd.key -out smtpd.key.unencrypted
mv -f smtpd.key.unencrypted smtpd.key
openssl req -new -x509 -extensions v3_ca -keyout cakey.pem -out cacert.pem -days 3650
```

Move the Certs to /etc/postfix/certs
In the end we'll have 1 cert - 1 key - 1 CAcert placed in /etc/postfix/certs/*
Those certs will be used for postfix,dovecot etc!!! To make our connection encrypted! As you can see in the postfix configuration, we're using those certs to make our clients accepting the certicate without prompting the validation of the certificate each time.

Editing Postfix Configuration:
Start by going to the directory - /etc/postfix
The next part is getting the rigth packages for getting the server up and running to the sasl authentication:

```
# apt-get install postfix-mysql postfix-tls libsasl2-modules-sql libsasl2-modules sasl2-bin
```

Which install all packages needing for the SASL authentication.
Now we shall edit the main.cf file which is the main configuration file for postfix:

```
# See /usr/share/postfix/main.cf.dist for a commented, more complete version



# Debian specific:  Specifying a file name will cause the first
# line of that file to be used as the name.  The Debian default
# is /etc/mailname.
#myorigin = /etc/mailname

smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
biff = no

# appending .domain is the MUA's job.
append_dot_mydomain = no

# Uncomment the next line to generate "delayed mail" warnings
#delay_warning_time = 4h
```

```
readme_directory = no

# TLS parameters
smtpd_tls_cert_file=/etc/postfix/certs/mailcert.pem
smtpd_tls_key_file=/etc/postfix/certs/mailkey.pem
smtpd_tls_CAfile = /etc/postfix/certs/cacert.pem
smtpd_use_tls=yes
smtp_use_tls = yes
smtpd_tls_session_cache_database = btree:${queue_directory}/smtpd_scache
smtp_tls_session_cache_database = btree:${queue_directory}/smtp_scache
smtpd_tls_loglevel = 1
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s

tls_random_exchange_name = /var/run/prng_exch
tls_random_source = dev:/dev/urandom
tls_smtp_use_tls = yes
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth-client

# See /usr/share/doc/postfix/TLS_README.gz in the postfix-doc package for
# information on enabling SSL in the smtp client.

myhostname = xxxx.dk
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
myorigin = /etc/mailname
mydestination =
relayhost =
mynetworks = 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
mailbox_command = procmail -a "$EXTENSION"
mailbox_size_limit = 0
recipient_delimiter = +
inet_interfaces = all

##### SASL bits #####
smtpd_sasl_auth_enable = yes
smtpd_sasl_local_domain =
smtpd_sasl_security_options=noanonymous
smtpd_recipient_restrictions =
   permit_sasl_authenticated,
   reject_unauth_destination,
   permit_mynetworks
   check_relay_domains
smtpd_delay_reject = yes
broken_sasl_auth_clients = yes

########## Virtual User Configurations ###################
virtual_alias_maps     = mysql:/etc/postfix/mysql_virtual_alias_maps.cf
virtual_mailbox_domains = mysql:/etc/postfix/mysql_virtual_mailbox_maps.cf
virtual_mailbox_maps    = mysql:/etc/postfix/mysql_virtual_mailbox_maps.cf
virtual_mailbox:limit    = 512000000
virtual_minimum_uid    = 5000
virtual_uid_maps    = static:5000
virtual_gid_maps    = static:5000
```

```
virtual_mailbox_base    = /storage
virtual_transport    = virtual

#### Additional for Quota Support
virtual_create_maildirsize = yes
virtual_mailbox_extended = yes
virtual_mailbox_limit_maps = mysql:/etc/postfix/mysql_virtual_mailbox_limit_maps.cf
virtual_mailbox_limit_override = yes
virtual_maildir_limit_message = Sorry, the your maildir has overdrawn your diskspace quota, please free up some of spaces of your mailbox try again.
virtual_overquota_bounce = yes
```

Check out that the virtual mailbox base are marked for /storage, which will be our base for maildir's.
We need to make that directory manually and afterwards change the permission for the directory:

```
mkdir /storage
chown -R postfix:postfix /storage
chmod -R 777 /storage
```

Virtual MySQL files:
Those files are used for connecting to the MySQL database for virtual users.
The first file is mysql_virtual_alias_maps.cf

```
user        = postfix
password        = xxxxxx
hosts        = 127.0.0.1
dbname        = postfix
table         = alias
select_field        = goto
where_field        = address
```

mysql_virtual_domains_maps.cf

```
user        = postfix
password        = xxxxxx
hosts        = 127.0.0.1
dbname        = postfix
table        = domain
select_field        = domain
where_field        = domain
additional_conditions = AND backupmx = '0' AND active = '1'
```

mysql_virtual_mailbox_limit_maps.cf

```
user                    = postfix
password                = xxxxxx
hosts                   = 127.0.0.1
dbname                  = postfix
table                   = mailbox
select_field          = quota
where_field            = username
additional_conditions = AND active = '1'
```

mysql_virtual_mailbox_maps.cf

```
user                    = postfix
```

```
password             = xxxxxx
hosts                = 127.0.0.1
dbname               = postfix
table                = mailbox
select_field         = maildir
where_field          = username
additional_conditions = AND active = '1'
```

And this file if you want support for relaying
mysql_relay_domains_maps.cf

```
user                 = postfix
password             = xxxxxx
hosts                = 127.0.0.1
dbname               = postfix
table                = domain
select_field         = domain
where_field          = domain
additional_conditions = AND backupmx = '1'
```

Now the permission for creating the Directory is made (otherwise we can add the user through the webinterface)

Enabling SASL-authentication:
For validating our clients along the way - and especially NOT make an OPEN RELAY mailserver, we're needing to use SASL-authentication. AS you can see in the main.cf file - we have enabled the setting - so we're need to make a file : /etc/postfix/sasl/smtpd.conf

```
pwcheck_method: auxprop
mech_list: plain login cram-md5 digest-md5
auxprop_plugin: sql
sql_engine: mysql
sql_hostnames: 127.0.0.1
sql_user: postfix
sql_database: postfix
sql_passwd: XXXXXXXXX
sql_select: SELECT password FROM mailbox WHERE username = '%u@%r'
```

That's all we need to start out with the SASL authentication.

Editing Dovecot
To start with we shall edit the two files:

/etc/dovecot/dovecot.conf
/etc/dovecot/dovecot-sql.conf

To start with the Dovecot.conf:
Change the following lines to fit into your needs:

```
Protocols = imaps pop3s    (Only secure connections)
Listen = *
Ssl_listen = *
ssl_cert_file = /etc/postfix/certs/mailcert.pem
ssl_key_file = /etc/postfix/certs/mailkey.pem
```

```
ssl_key_password = XXXXXXX
ssl_ca_file = /etc/postfix/certs/cacert.pem
verbose_ssl = yes

login_process_size = 64
login_greeting = domain Dovecot ready.
mail_location = maildir:/storage/%u/
namespace private {
   separator = .
   inbox = yes
   hidden = yes
}
Mail_access_groups = postfix

For testing case:
mail_debug = yes
verbose_proctitle = yes
first_valid_uid = 5000
first_valid_gid = 5000
mbox_read_locks = fcntl
mbox_write_locks = dotlock fcntl

protocol imap {
   imap_client_workarounds = delay-newmail outlook-idle netscape-eoh tb-extra-mailbox-sep
}
Protocol pop3 {
   pop3_uidl_format = %08Xu%08Xv
   pop3_client_workarounds = outlook-no-nuls oe-ns-eoh
}

auth_default_realm = domain.dk
auth_username_chars = abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ01234567890.-_@

auth_verbose = yes
auth_debug = yes
auth_debug_passwords = yes

auth default {
   mechanisms = plain login digest-md5 cram-md5
     passdb sql {
         # Path for SQL configuration file
         Args = /etc/dovecot/dovecot-sql.conf
     }

       userdb sql {
         args = /etc/dovecot/dovecot-sql.conf
}
User = root
               # It's possible to export the authentication interface to other programs:
               socket listenn {
      #master {
        # Master socket provides access to userdb information. It's typically
        # used to give Dovecot's local delivery agent access to userdb so it
        # can find mailbox locations.
        #path = /var/run/dovecot/auth-master
        #mode = 0600
        # Default user/group is the one who started dovecot-auth (root)
        #user =
       #group =
       #}
       client {
        # The client socket is generally safe to export to everyone. Typical use
        # is to export it to your SMTP server so it can do SMTP AUTH lookups
        # using it.
        path = /var/spool/postfix/private/auth-client
        mode = 0660
        user = postfix
        group = postfix
       }
     }
```

Those lines with the red letters are only for Debugging cases!
The Next part is editing /etc/dovecot/dovecot-sql.cnf

```
driver =  mysql
default_pass_scheme = MD5-CRYPT
connect = dbname=postfix user=postfix  host=localhost password=xxxxxxx
password_query = SELECT password FROM mailbox WHERE username = '%u' AND active ='1'
user_query = SELECT maildir,5000 AS uid,5000 AS gid FROM mailbox WHERE username  =
    '%u' AND active = '1'
```

Those lines are need to be edited in that File!

When this is up and running and testet that'll we can go on with integration with the spam and Virus filter.

Spamassassin and ClamAV
These programs are the ones for scanning for Virus and Spam, so we'll need those not to get a whole lot of spammail and Virus into our system.
Start with installing the different packages:

```
apt-get install amavisd-new spamassassin clamav-daemon clamav-freshclam libnet-dns-perl libmail-spf-query-perl pyzor razor file arc gzip bzip2 cabextract zip unzip unrar-free cpio tar zoo arj lzop nomarch pax unzoo
```

These tools are used for scanning also inside archive files etc.
Clamav
To start with the Clamav integration, we will start with editing the configuration for that.
Since we're ruardng Ubuntu Hardy - the user amavis is already created during the install, but we'll need to add the user to the group of clamav. So add the text amavis to clamav in the file /etc/group:

```
clamav:x:117:amavis
amavis:x:118:clamav
```

This will make our daemon for clamav (clamd) accessable for the user amavisd! And visaversa!

The next part is to activate spam and antivirus detection in the file /etc/amavisd/conf.d/15-content_filter_mode:

```
use strict;
```

```
# You can modify this file to re-enable SPAM checking through spamassassin
# and to re-enable antivirus checking.

#
# Default antivirus checking mode
# Uncomment the two lines below to enable it back
#

@bypass_virus_checks_maps = (
  \%bypass_virus_checks, \@bypass_virus_checks_acl, \$bypass_virus_checks_re);


#
# Default SPAM checking mode
# Uncomment the two lines below to enable it back
#

@bypass_spam_checks_maps = (
  \%bypass_spam_checks, \@bypass_spam_checks_acl, \$bypass_spam_checks_re);

1;  # ensure a defined return
```

That's it for clamav

Spamassassin
Spamassassin will automaticly scan for optional components and use them if available. A few of these, which we'll use, are the dcc-client,pyzor and razor(as installed earlier) These will not need to be configured.
Edit /etc/default/spamassassin to activate the daemon. Change ENABLED=0 to ENABLED=1, and set CRON=1 for enabling a dailæy update of the rules:

```
# Change to one to enable spamd
ENABLED=1

# Cronjob
# Set to anything but 0 to enable the cron job to automatically update
# spamassassin's rules on a nightly basis
CRON=1
```

That's the default configuration for Spamassassin

Integrating the filters into postfix
For postfix integration, you should only add in /etc/postfix/main.cf the following line:

```
content_filter=smtp-amavis:[127.0.0.1]:10024
```

Edit the /etc/postfix/master.cf, and add these læines in the bottom:

```
smtp-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
```

Add these lines immediatly following the "pickup" transport service:

```
        -o content_filter=
        -o receive_override_options=no_header_body_checks
```

This will help stop marking messages, reporting spam, as Spam.
Reload postfix, and now content filtering with spam and virus detection is enabled!

Squirrelmail (for webaccess to mailbox):
We'll use squirrelmail for given the employee's a chance to lccess their mailbox through the internet. So start by installing the needed packages:

```
apt-get install squirrelmail squirrelmail-decode squirrelmail-locales
```

After installing those packages, change your path to /etc/squirrelmail.Start by running the configurationscript for squirrelmail in the squirrelmail directory:

```
./conf.pl
```

Then we should see a menu - for all settings regarding to the Squirrelmail.

```
Menu 1:
1.Organization Name        : Firmanavn
2.Organization Logo        : ../images/insalogo.jpg
3.Org: Logo Width/height        : (900/300)
4.Organization Title        : Firmanavn Webmail
5.Signout page            :
6.Top Frame            : _top
7.Provider Link            : http://www.domain.dk
8.Provider Name        : domain
Menu 2:
1.Domain            : domain.dk
2.Invert time            : false
3.Sendmail or SMTP        : SMTP
SMTP Settings:
4. SMTP server            : localhost
```

```
5.SMTP port            : 25
6.POP before SMTP        : false
7. SMTP Authentication         : login(with IMAP username and password)
8. Secure SMTP (TLS)        : false
9 Header encryption key        :
IMAP Settings:
4. IMAP server            : localhost
5. IMAP Port            : 993
6. Authentication Type        : login
7. Secure IMAP (TLS)        : true
8. Server Software        : dovecot
9. Delimiter            : .

Menu 3:
1.  Default Folder Prefix                :
2.  Show Folder Prefix Option        : false
3.  Trash Folder                         : Trash
4.  Sent Folder                          : Sent
5.  Drafts Folder                        : Drafts
6.  By default, move to trash            : true
7.  By default, move to sent              : true
8.  By default, save as draft            : true
9.  List Special Folders First           : true
10. Show Special Folders Color        : true
11. Auto Expunge                          : true
12. Default Sub. of INBOX            : false
13. Show 'Contain Sub.' Option        : false
14. Default Unseen Notify                : 2
15. Default Unseen Type                  : 1
16. Auto Create Special Folders      : true
17. Folder Delete Bypasses Trash      : false
18. Enable /NoSelect folder fix      : false

Menu 4:
1.  Data Directory                      : /var/local/squirrelmail/data/
2.  Attachment Directory                : /var/local/squirrelmail/attach/
3.  Directory Hash Level                : 0
4.  Default Left Size                 : 150
5.  Usernames in Lowercase              : false
6.  Allow use of priority              : true
7.  Hide SM attributions              : false
8.  Allow use of receipts              : true
9.  Allow editing of identity          : true
    Allow editing of name              : true
    Remove username from header     : false
10. Allow server thread sort            : false
11. Allow server-side sorting          : false
12. Allow server charset search         : true
13. Enable UID support                  : true
14. PHP session name                    : SQMSESSID
15. Location base                       :

Menu 5:
1.  Change Themes
    Plain Blue                                  Deep Ocean
```

Purple	Ice
Blue Steel	High Contrast
Servery	BluesNews
Blue Grey	Methodical
In The Pink (Changes)	Monostochastic (Changes)
Spice of Life (Changes)	Spice of Life - Dark (Changes)
Darkness (Changes)	Midnight
Dark Green	Minimal BW
Net Style	Simple Green
Bluesome	Simple Purple
Autumn 2	Classic Blue
Powder Blue	Turquoise

2. CSS File :

Menu 6:
1. Change LDAP Servers
2. Use Javascript Address Book Search : false
3. Global file address book :
4. Allow writing into global file address book : false
5. Allow listing of global file address book : true
6. Allowed address book line length : 2048

Menu 7:
Editing the MOTD of the day.

Menu 8:
If any plugins are installed - they're are here

Menu 9:
1. DSN for Address Book :
2. Table for Address Book : address

3. DSN for Preferences :
4. Table for Preferences : userprefs
5. Field for username : user
6. Field for prefs key : prefkey
7. Field for prefs value : prefval

8. DSN for Global Address Book :
9. Table for Global Address Book : global_abook
10. Allow writing into Global Address Book : false
11. Allow listing of Global Address Book : false

Menu 10:
1. Default Language : da_DK
2. Default Charset : iso-8859-1
3. Enable lossy encoding : false

Those directory's for using Squirrelmail - aren't created at installationtime - so you'll ned to manually create them and give them the rigth permission:

```
mkdir -p /var/local/squirrelmail/data
mkdir -p /var/local/squirrelmail/attach
chmod -R 777 /var/local/squirrelmail
```

These are the most important Files to edit for Squirrelmail, but when we're installing the server for real - I'll

properly use the directly package from http://www.squirrelmail.org instead of the one i APT.
Although we'll need to make our Webserver connection secure for this Case.

When this is up and running - testet etc - you're able to edit /etc/apache2/sites-available/squirrelmail to fit your needs for webaccess. We'll like to edit it to get only https access to the webmail.

Editing Apache and Securing Webmail access:
First start by editing /var/www/index.html - caurse we wan't all directly traffic on this server redirected to our main website:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title>domain Redirect</title>
<meta http-equiv="REFRESH" content="0;url=http://www.domain.com"></HEAD>
<BODY>
You'll be redirected to www.xxxxxx.com
</BODY>
</HTML>
```

This little htmlfile will redirect all incomming request on www.domain.dk to www.domain.com
The next file to edit is /etc/apache2/sites-available/default and remove the word "mail" that we during our Apache configuration earlier added to the file. This will put our webserver's root Directory back to /var/ww - where the index.html is the page that the root hits and redirect the browser to our company mainpage!
When this is done - We'll start editing /etc/apache2/sites-available/squirrelmail to look like this:

```
<IfModule mod_ssl.c>

   <VirtualHost *:443>
      DocumentRoot /usr/share/squirrelmail
      ServerName webmail.xxxxxx.dk
      SSLEngine on
      SSLCertificateFile /etc/apache2/certs/wwwcert.pem
      SSLCertificateKeyFile /etc/apache2/certs/wwwkey.pem
      SSLCACertificateFile /etc/apache2/certs/Cacert.pem

   </VirtualHost>
   <VirtualHost *:80>
      DocumentRoot /usr/share/squirrelmail
      ServerName webmail.domain.dk
      RewriteEngine on
      RewriteRule ^/(.*) https://webmail.domain.dk/$1
   </VirtualHost>
</IfModule>
```

Make a symlink from the file - to the active sites-enabled:

```
ln -sf /etc/apache2/sites-available/squirrelmail /etc/apache2/sites-enabled/
```

Next part is enabling to 2modules that'll be used for redirecting the from http to https !
```
a2enmod rewrite
a2enmod ssl
```
At the same time We'll need to make and default virtualsite conf and SSL default conf. Starting with the default 000default:
```
NameVirtualHost *:80
```

```
<VirtualHost *:80>
   ServerAdmin webmaster@localhost
   DocumentRoot /var/www
   RewriteEngine on
         RewriteRule ^/(.*) http://wwww.domain.com/$1

   <Directory />
      Options FollowSymLinks
      AllowOverride None
   </Directory>
   <Directory /var/www>
      Options Indexes FollowSymLinks MultiViews
      AllowOverride None
      Order allow,deny
      allow from all
   </Directory>
   ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
   <Directory "/usr/lib/cgi-bin">
      AllowOverride None

      Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
      Order allow,deny
      Allow from all
   </Directory>
   ErrorLog /var/log/apache2/error.log
   # Possible values include: debug, info, notice, warn, error, crit,
   # alert, emerg.
   LogLevel warn
   CustomLog /var/log/apache2/access.log combined
   ServerSignature On

   Alias /doc/ "/usr/share/doc/"
   <Directory "/usr/share/doc/">
      Options Indexes MultiViews FollowSymLinks
      AllowOverride None
      Order deny,allow
      Deny from all
      Allow from 127.0.0.0/255.0.0.0 ::1/128
   </Directory>
 </VirtualHost>
```

And the SSLdefault:

```
NameVirtualHost *:443
<VirtualHost *:443>
   ServerAdmin webmaster@domain.dk
   ServerName  webmail.domain.dk
SSLEngine On
SSLCertificateFile /etc/apache2/ssl/apache.pem
   DocumentRoot /usr/share/squirrelmail
   <Directory />
      Options FollowSymLinks
      AllowOverride None
   </Directory>
```

```
    <Directory /usr/share/squirrelmail>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>
    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
    <Directory "/usr/lib/cgi-bin">
        AllowOverride None
        Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
        Order allow,deny
        Allow from all
    </Directory>
    ErrorLog /var/log/apache2/SSL_error.log
    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn
    CustomLog /var/log/apache2/access.log combined
    ServerSignature On
          Alias /doc/ "/usr/share/doc/"
    <Directory "/usr/share/doc/">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order deny,allow
        Deny from all
      Allow from 127.0.0.0/255.0.0.0 ::1/128
    </Directory>
</VirtualHost>
```

Then we need to make a certificate to Apache - just like we did to Dovecot , but instead of calling the files mail - we'll call'em Apache.
Then our webserver is validating and forcing the request to go using a secured line - https !
But this is the tricky part about validating our Webserver´s security Zone, so getting this to work is the best part of it all.

The last part is to make our Mailadministration accessable from a browser - So need to make a file called mailadmin.conf in /etc/apache2/sites-available/ and make a symlink to sites-enabled Directory, with these lines in:

```
# Virtual hostfile for Mailadministrating Interface

<VirtualHost *:80>
    ServerName        mailadmin.domain.dk
    ServerAdmin        webmaster@domain.com
    DocumentRoot        /var/www/mail
    ErrorLog /var/log/apache2/mailadmin.log
    <Directory "/var/www/mail/">
            Options Indexes MultiViews FollowSymLinks
            AllowOverride None
                    Order deny,allow
                    Deny from all
                    Allow from 172.16.50.0/23
    </Directory>
</Virtualhost>
```

This is what we're needing for setting up the server - so now it's ready to go to the next step.!!

Certificate import:

Certificate import:
To make Outlook accept our certificates for the mailserver - we need to do the following procedure to make it work:
cd /etc/postfix/certs
Copy the smtpd.crt to your windows Computer.

Afterwards we need to import these certs into the Windows machine, To make our client accept the connections without complaining or asking about if we trust the server.
Client certificate:
Start with doubleclicking on the smtpd.crtfile to import this into the Internet Explorer. Follow the guide to the end - You'll need to add the certificate to the known trusted company's!

Server Certificate:

To make this work -. we'll need to make our selfsigned certs - trusted. Which means the signauthor should also be a trusted company - So we'll import the Cacert into the Internet explorer as an trusted signauthority.
Start Internet Explorer - Go to Preferences - content manager.
There a options in the middle og the windows for importing a new trusted CA-organization. Follow the guide - and afterwards restart your windows - and you're ready to go!

Vacation message:

If you're needing a vacation(out of office reply) you'll need to make some more changes.
The requirements for vacation:
- Perl5
- Perl DBI
- Either Perl DBD::mysql OR Perl DBD::pgsql - depending on DB backend.
- Email::Valid
- Mail::Sendmail

First create the user and group for the vacation user. Should look like this:
```
#/etc/passwd
vacation:*:65501:65501::0:0:Virtual Vacation:/nonexistent:/sbin/nologin

#/etc/group
vacation:*:65501:
```

Add these lines in the end of /etc/postfix/main.cf:
```
transport_maps = hash:/etc/postfix/transport

# Vacation rules
vacation_destination_reciepients_limit = 1
```

And in the end of /etc/postfix/master.cf
```
# Vacation definations
vacation    unix  -  n  n  -  -  pipe
flags=Rq user=vacation argv=/var/spool/vacation/vacation.pl -f ${sender} -- ${recipient}
```

Afterwards you shall create the dir for vacation:

```
mkdir /var/spool/vacation
cp /var/www/mail/VIRTUAL_VACATION/vacation.pl /var/spool/vacation
```
Edit vacation.pl to fit your need for db - user passwd and DBname!

Last you'll edit the file /etc/postfix/transport to fit:
```
autoreply.domain.dk    vacation:
```

And afterwards run the command
```
postmap /etc/postfix/transport
```

Add the vacation user to the DB - and remember to grant access to the postfix DB!

Restart your machine and everything should work now!

Import from old Mailserver:
Integration from POP3 to IMAP - Or converting from mbox to Maildir-
References:
My main references are:
http://www.ubuntu.com
https://help.ubuntu.com/community/
http://www.linuxin.dk
http://www.apache.org

Kommentar af ohhelpme d. 15. jan 2011 | 1

Perfect =).. især mail delen som jeg aldrig har fået til at virke i de 5 -6 år jeg har kørt ubuntu server & ubuntu desktop