This documentation describes gpgmailencrypt 2.2.x

1 What is gpgmailencrypt

gpgmailencrypt is an encrypting e-mail gateway that can encrypt e-mails.

It supports

* PGP/Inline

* PGP/Mime

* SMime

* encrypted PDF

It can be used normally as a script doing everything on command line or in daemon mode, where gpgmailencrypt acts as an encrypting smtp server.

It takes e-mails and if a encryption key exists for this user it will return the e-mail encrypted to another e-mail server.

The encryption method can be selected per user.

1.1 Prerequisites

The following software needs to installed

- python3.x
- gnupg (I recommend version 2)
- openssl

For encrypted-PDF you need to install:

- email2pdf (https://github.com/andrewferrier/email2pdf)
- wkhtml2pdf (http://wkhtmltopdf.org)
- pdftk
- 7zip

2 Installation

2.1 General

There are 2 different ways to install gpgmailencrypt. Choose the one that suits you best.In both cases you need to have installed python3 setuptools (on Ubuntu it's 'sudo apt-get install python3-setuptools')

2.1.1 Using pip

Install pip first (on Ubuntu its 'sudo apt-get install python3-pip')

The most easy way to install gpgmailencrypt is via pip:

```
sudo pip3 install gpgmailencrypt
```

2.1.2 Installing the archive

After you've downloaded the package from
https://github.com/gpgmailencrypt/gpgmailencrypt/releases

```
unzip the zip-package with unzip gpgmailencrypt-xxx.zip
or untar it
tar -xzf gpgmailencrypt-xxx.tar.gz
```

Then change in the directory and start installation with

```
sudo python3 ./setup.py install
```

2.2 Daemon

1. Make the gpgmailencrypt init file writeable

   ```
   sudo chmod 755 /etc/init.d/gpgmailencrypt
   ```

2. Create a user (gpg-mailencrypt) under which the daemon should run

   ```
   sudo adduser gpg-mailencrypt
   ```

3. You can set the user in the file /etc/default/gpgmailencrypt. It should contain

   ```
   USER="gpg-mailencrypt"
   DIR=”/usr/local/bin”
   ```

3 Configuration /etc/gpgmailencrypt.conf

To create a default configuration file create the conf file with:

```
gme.py -x >~/gpgmailencrypt.conf
sudo cp ~/gpgmailencrypt.conf /etc
```

```
sudo chown gpg-mailencrypt.root /etc/gpgmailencrypt.conf
sudo chmod 640 /etc/gpgmailencrypt.conf
```

3.1 General

```
[default]
prefered_encryption = pgpinline          #valid values are
                                         #'pgpinline','pgpmime' or 'smime'
```

```
add_header = no                           #adds a X-GPGMailencrypt header to
                                          #the mail
domains =                                 #comma separated list of domain
                                          #names, that should be encrypted,
                                          #empty is all
spamsubject =***SPAM                      #Spam recognition string, spam will
                                          #not be encrypted
output=mail                               #valid values are 'mail'or 'stdout'
locale=en                                 #DE|EN|ES|FR|IT|NL|PL|PT|RU|SE
mailtemplatedir=/usr/share/gpgmailencrypt/mailtemplates
                                          #directory where mail
                                          #templates are stored
systemmailfrom=gpgmailencrypt@localhost   # e-mail address used when sending
                                          #system mails
alwaysencrypt=False                       #if True e-mails will be sent
                                          #encrypted, even if there is no key.
                                          #Fallback encryption is encrypted pdf
```

3.2 PGP specific configuration

```
[gpg]
keyhome = /var/lib/gpgmailencrypt/.gnupg        #home directory of public
                                                #gpgkeyring
gpgcommand = /usr/bin/gpg2
allowgpgcomment = yes                           #allow a comment string in the
                                                #GPG file
```

3.3 SMIME specific configuration

```
[smime]
keyhome = ~/.smime                        #home directory of S/MIME public key
                                          #files
opensslcommand = /usr/bin/openssl
defaultcipher = DES3                      #DES3|AES128|AES192|AES256
extractkey= no                            #automatically scan e-mails and
                                          #extract smime public keys to
                                          #'keyextractdir'
keyextractdir=~/.smime/extract

[smimeuser]
smime.user@domain.com = user.pem[,cipher] #public S/MIME key file [,used cipher,
                                          #see defaultcipher]
```

3.4 PDF specific configuration

The e-mail text will be stored in an encrypted pdf, attachments in an encrypted zip-file with the same password.

```
[pdf]
email2pdfcommand=/usr/bin/email2pdf       #path where to find email2pdf (needed
                                          #for creating pdfs, see
            #https://github.com/andrewferrier/email2pdf)
```

```
pdftkcommand=/usr/bin/pdftk              #path where to find pdftk (needed for
                                         #encrypting pdf files
pdfdomains=localhost                     #a comma separated list of sender
                                         #domains, which are allowed to use
                                         #pdf-encrypt
passwordlength=10                        #Length of the automatic created
                                         #password
passwordlifetime=172800                  #lifetime for autocreated passwords in

                                         #seconds. Default is 48 hours
```

3.5 Zip-specific configuration

The zip functionality is mostly used with the pdf feature. Nevertheless you can use it indepently to zip all attachments (zipatttachments=True).

```
[zip]
7zipcommand=/usr/bin7za                  #path where to find 7za
defaultcipher=ZipCrypto                  #ZipCrypto|AES128|AES256
compressionlevel=5                       #1,3,5,7,9  with 1:lowest compression,
                                         #but very fast, 9 is
                                         # highest compression, but very slow,
                                         #default is 5
securezipcontainer=False                 #attachments will be stored in an
                                         #encrypted zip file. If this option is
                                         #true,
                                         #the directory will be also encrypted
zipattachments=False                     #if True all attachments will be
                                         #zipped, independent from the
                                         #encryption method
```

3.6 Daemon specific configuration

```
[daemon]
host = 127.0.0.1                         #smtp host
port = 10025                             #smtp port
smtps = False                            #use smtps encryption
starttls = False                         #use starttls encryption
forcetls = False                         #communication (e.g. authentication)
                                         #will be only possible after STARTTLS
sslkeyfile = /etc/gpgsmtp.key            #the x509 certificate key file
sslcertfile = /etc/gpgsmtp.crt           #the x509 certificate cert file
authenticate = False                     #users must authenticate
smtppasswords = '/etc/gpgmailencrypt.pw' #use smtps encryption
```

The gpgmailencrypt.pw has the following format (for the authenticate option see chapter 6.3.):

```
user1=password1
user2=password2
```

Don't forget to make the file readable only for the gpgmailencrypt user!

How the x509 certificate files can be created see:

```
https://www.e-rave.nl/create-a-self-signed-ssl-key-for-postfix
```

The ssl certificates will be needed for “starttls” and “smtps” mode.

What's the difference between starttls and smtps?

Both create and use a ssl encrypted communication channel. Smtps is the old version, where both sides use ssl from the start. Starttls connections first start unencrypted, then both sides switch to ssl after sending the “startTLS” command.

In doubt use the starttls variant.

If you set stattls and smtps at the same time in your configuration, then SMTPS will be used.

4 Key Management

The following commands have to be used as the user, that is running gpgmailencrypt. Remember that in daemon mode this user is 'gpg-mailencrypt'. So for daemon mode you first have to change the user

```
sudo su – gpg-mailencrypt
```

4.1 PGP

Add a PGP key to the public key ring

```
gpg --import publickey.gpg
```

4.2 SMIME

Smime keys are stored in the directory ~/.smime per default. You have to create it if it does not exist. Each key is stored in a single file in pem-format.

Usually you get the smime.key file in a different format. To convert it use

```
openssl pkcs7 -print_certs -inform DER -in smime.p7s -out smime.pem
```

Let's say you get the smime.p7s from agentj@mib.

Instead of 'smime.pem” you should use a unique name for the file and copy it in ~/.smime/

```
cp smime.pem ~/.smime/agentj@mib.pem
```

For this user you need also an entry in /etc/gpgmailencrypt.conf

```
[smimeuser]
agentj@mib = agentj@mib.pem
```

4.3 Using PDF encryption

Let's say you want to send an e-mail with confidential content to somebody, but you don't have the gpg or smime key of the receiver. So PDF encryption is for you.

PDF encryption is -in contrary to gpg and smime encryption – a symmetric encryption method. Gpgmailencrypt is converting the e-mail text to a PDF file and puts all attachments in a ZIP file.

Both are encrypted with a password.

How to use PDF encryption?

Simply add **#encrypt** at the beginning of the subject of your e-mail (at least one whitespace after #encrypt).

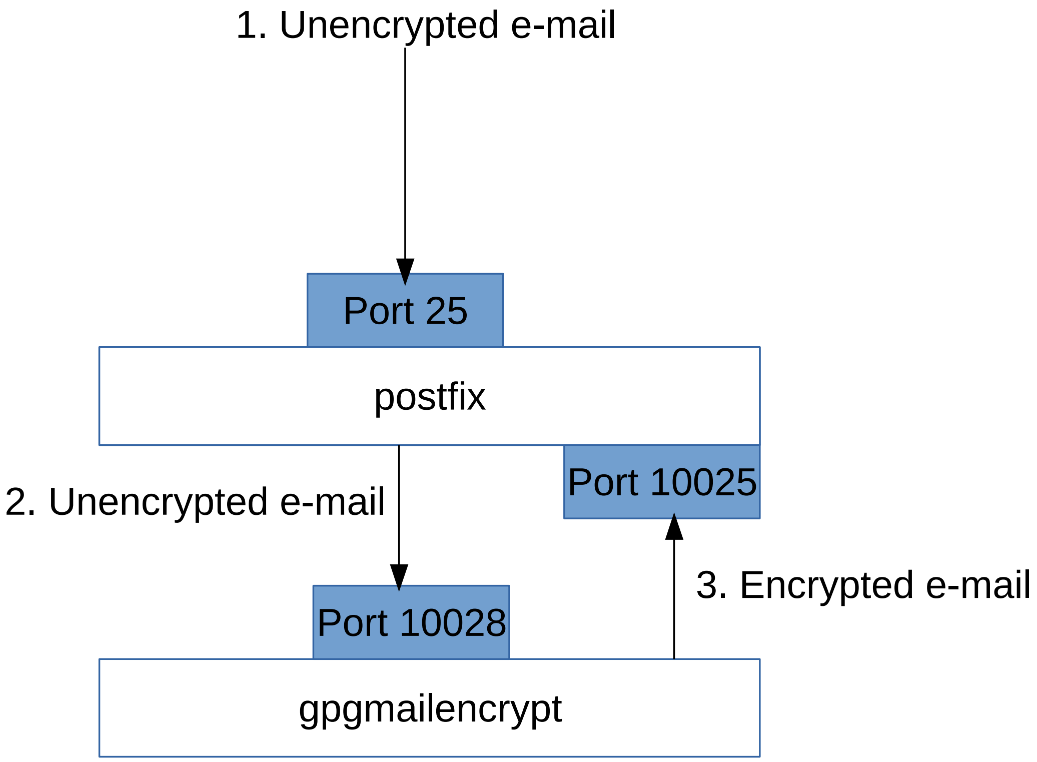
That's all.

When you send your e-mail you will receive an automatic e-mail with the password. Give the password to the e-mail recipient via a different communication channel (e.g. phone). Then he can read the e-mail and the attachments with standard tools like a pdf-reader and any zip-application.

If the recipient has already a gpg or smime key, then this will be used.

5 Integrating gpgmailencrypt in postfix

5.1 Direct way from postfix to gpgmailencryt



Install and configure gpgmailencrypt as daemon.

/etc/gpgmailencrypt.conf

```
[mailserver]
host = 127.0.0.1
port = 10025
[daemon]
host = 127.0.0.1
port = 10028
```

/etc/postfix/main.cf

```
content_filter=gpgmailencrypt:[127.0.0.1]:10028
```

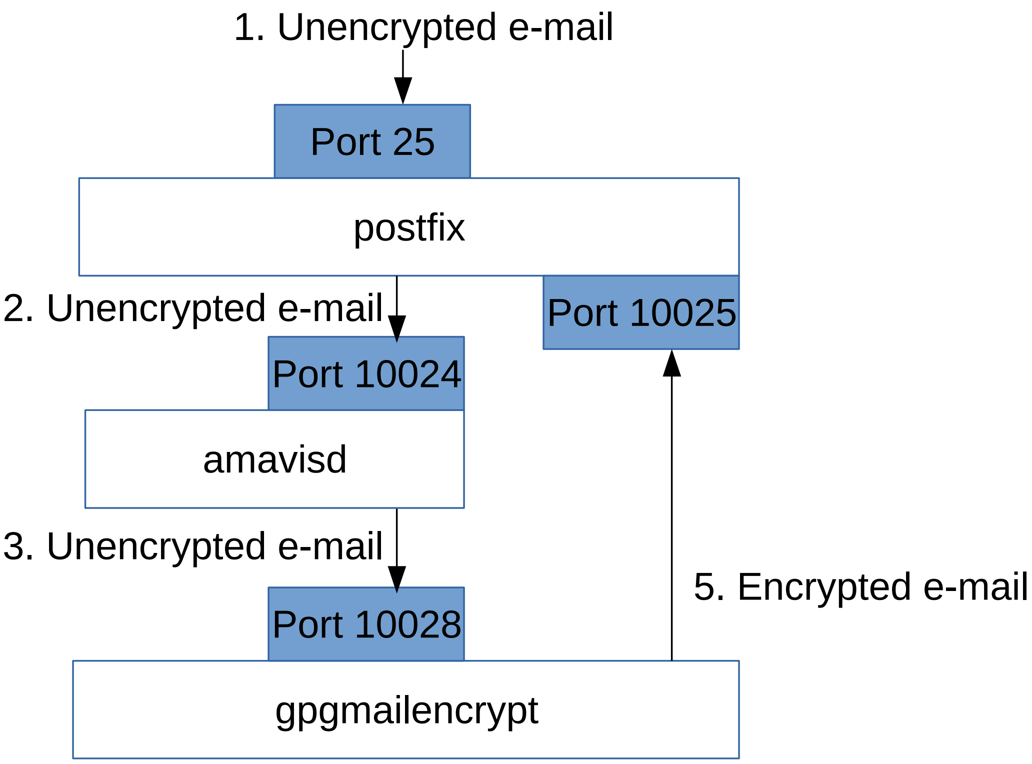
/etc/postfix/master.cf

```
localhost:10025 inet n - n - -  smtpd
        -o content_filter=
        -o mynetworks=127.0.0.0/8
        -o receive_override_options=no_unknown_recipient_checks
        -o smtpd_recipient_restrictions=permit_mynetworks,reject_unauth_destination
        -o smtpd_authorized_xforward_hosts=127.0.0.0/8

gpgmailencrypt unix - - n - 2 smtp
        -o smtp_data_done_timeout=1800
```

5.2 From Postfix to amavisd to gpgmailencrypt

This is to check for viruses and spam before encrypting the mail



This has to be configured as before, except:

/etc/postfix/main.cf

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

/etc/postfix/master.cf

```
amavis  unix - - n - 2 smtp
       -o smtp_data_done_timeout=1800
```

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

```
[...]
$notify_method  = 'smtp:[127.0.0.1]:10028';
$forward_method = 'smtp:[127.0.0.1]:10028';
```

Of course you have to configure amavisd to suite your needs.

Amavis and SSL

Amavis does not support SSL encryption of any kind or authentication. So you can't use these features of gpgmailencrypt in combination with amavis.

5.3 Using authentication

For using the authentication add the following to gpgmailencrypt section in /etc/postfix/master.cf

```
        -o smtp_sasl_auth_enable=yes
        -o smtp_sasl_password_maps=hash:/etc/postfix/gpgmailencrypt_passwd
```

With /etc/postfix/gpgmailencrypt_passwd having the following structure

```
localhost user:password
```

Then use the following commands

```
sudo chmod 640 /etc/postfix/gpgmailencrypt_passwd
sudo postmap /etc/postfix/gpgmailencrypt_passwd
```

5.4 Using starttls

Add the following to gpgmailencrypt section in /etc/postfix/master.cf:

```
      -o smtp_use_tls = yes
      -o smtp_tls_security_level = encrypt
```

5.5 Using smtps

To use the gpgmailencrypt smtps feature with postfix 2.x you need to install stunnel (in Ubuntu the package is called stunnel4)

Create the file /etc/stunnel/gpgmailencrypt.conf

```
[gpgmailencrypt-smtps]
    accept = 10000
    client = yes
    connect = localhost:10028
```

And change /etc/default/stunnel4

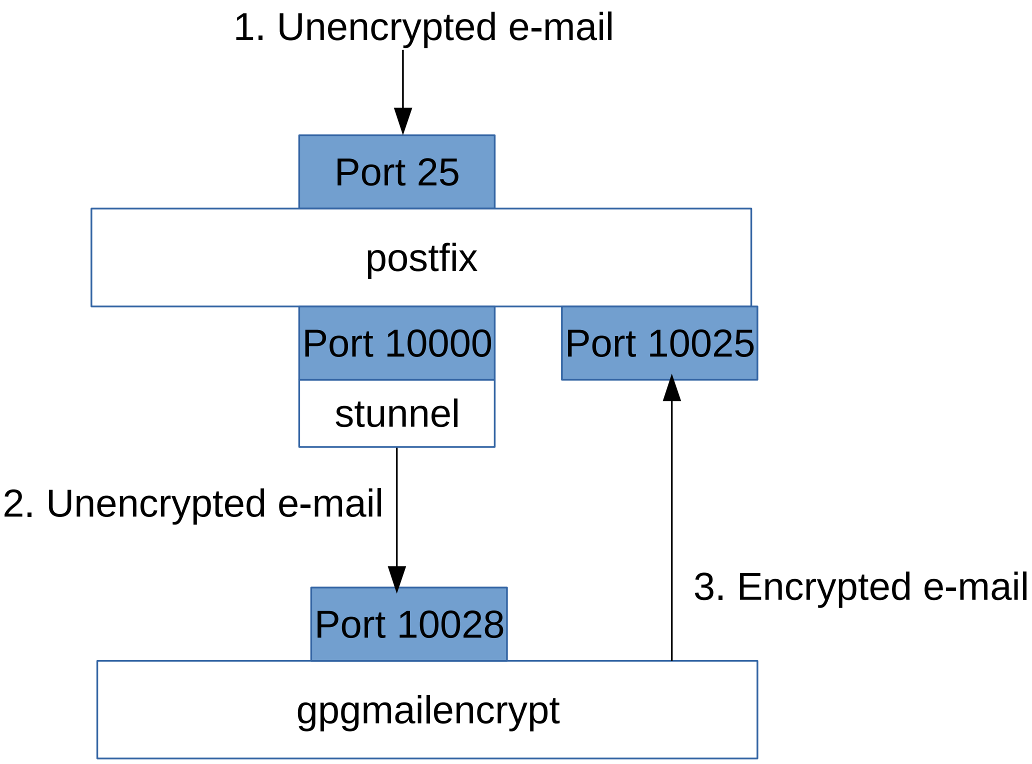
```
ENABLED=1
```

Then start stunnel with

```
/etc/init.d/stunnel4 start
```

/etc/postfix/main.cf should be changed to

```
content_filter=gpgmailencrypt:[127.0.0.1]:10000
```



6 The admin console

6.1 The admin user – the hen and egg problem

First you have to create a normal user (which only the admin user can do), then you can give him administration rights.

So for the very first time you have to start the following python program as the user, that runs the gpgmailencrypt daemon. Replace “admin1” and “secreet” with the values you like

```
#!/usr/bin/python3
import gpgmailencrypt
user=”admin1”
password=”secret”
with gpgmailencrypt.gme() as g:
      g.adm_set_user(user,password)
```

Now the user is created.

Now change the /etc/gpgmailencrypt.conf to the following value

```
[daemon]
admins=admin1                       #comma separated list of admins, that can
use the admin console
```

Now restart the daemon, that's it.

6.2 Using the admin console

Start

```
gme_admin.py localhost 10025
```

Replace 'localhost' and the port number with the values of your daemon

Now login as an admin user. You will see the following screen

```
Try to connect to localhost:10025 ...
gpgmailencrypt admin console
============================
User: admin1
Password:
OK
Authentication successful.
Welcome. Enter 'HELP' for a list of commands
> _
```

The following commands are available:

flush	tries to re-send deferred e-mails
debug true/false	sets the debug mode valid options are TRUE|FALSE, ON|OFF, YES|NO
deluser	deletes a user example: 'deluser john'
help	Shows all available commands
messages	shows all systemwarnings and -errors
quit	leave the console
reload	reloads the configuration file
resetstatistics	sets all statistic values to 0
setuser	adds a new user or changes the password for an existing user example: 'setuser john johnspassword'
statistics	print statistic information
users	print users

6.3 Non-admin users

Users, that don't have the admin privilege can be used to login to the server, when the daemon need authentication, set with

/etc/gpgmailencrypt.conf

```
[daemon]
authenticate = True                 #users must authenticate
```

7 Using gpgmailencrypt from the command line

Start

```
gme.py
```

The command line options are

```
Usage:

gme.py [options] receiver@email.address < Inputfile_from_stdin

Options:

-a --addheader:   adds X-GPGMailencrypt header to the mail
-c f --config f:  use configfile 'f'. Default is /etc/gpgmailencrypt.conf
-d --daemon :     start gpgmailencrypt as smtpserver
```

```
-e pgpinline :    preferred encryption method, either 'pgpinline','pgpmime' or
                  'smime'
-f mail :         reads email file 'mail', otherwise from stdin
-h --help :       print this help
-k f --keyhome f: sets gpg key directory to 'f'
-l t --log t:     print information into _logfile, with valid types 't'
                  'none','stderr','syslog','file'
-n domainnames:   sets the used domain names (comma separated lists, no space),
                  which should be encrypted, empty is all
-m mailfile :     write email file to 'mailfile', otherwise email will be sent
                  via smtp
-o p --output p:  valid values for p are 'mail' or 'stdout', alternatively you
                  can set an outputfile with -m
-x --example:     print example config file
-v --verbose:     print debugging information into _logfile
-z --zip:         zip attachments
```

8 Using gpgmailencrypt as a module

```
import gpgmailencrypt
help(gpgmailencrypt)
```

The most important function is

```
gpgmailencrypt.encrypt_mails(self,mailtext,receiver)
```

An example:

```
import gpgmailencrypt

#reads the e-mail from a file
f=open(‶myemail.eml)
mail=f.read()
f.close()

#sends an encrypted e-mail
with gpgmailencrypt.gme() as g:
      g.encrypt_mails(mailtext=mail,recipient=”agentj@mib”)
      #the following sends the e-mail to 2 recipients
      g.encrypt_mails(mailtext=mail,recipient=[”agentj@mib”,”agentk@mib”])
```