1 Introduction

When financial institutions eventually make encryption a routine procedure, you can begin to feel safe dealing with them by email. Some of them have figured out ways of letting you view your information *via* `https` without sending any easily-compromised details with the email, but most are not anywhere near that yet. In the meantime, you'll just have to practice encrypting whenever you can.

There are many ways of managing email encryption. Mostly people don't. The old-fashioned way was to use digital certificates, an absolute nightmare for the ordinary user to configure. In this talk I shall go through the practicality of using `PGP` in its modern version of `GPG`, instead.

All these examples under `Linux` were installed and run on a `Debian 9 (stretch)` system. The examples under `MacOSX` were installed and run on `version 10.9.5 (maverick)` of that system.

2 Encryption Considerations

Nowadays, you can easily email small attachments, up to 10MB, and large files can be posted on DVD.

```
$ gpg --encrypt --recipient user@emailaddress --output file.gpg infile
$ openssl enc -md md5 -aes-256-cbc -salt -in file -out file.aes -pass pass:"pass phrase"
```

Using the two encryption programs above, my experiments show that encryption of very large files is 5 times faster using `openssl` than `gpg`. Thus we encrypt directly using `gpg`, or encrypt using `openssl` and send the passphrase using `gpg`. Either way, we shall need the `GnuPG` suite of programs.

3 Suitable Email Programs

The site `https://en.m.wikipedia.org/wiki/Comparison_of_email_clients` examines at least 50 email programs. We would like to select one that runs on `Linux, Mac` and `Windows`; gets email using `pop3`; provides security with `ssl/tls`; and uses `pgp/gpg` for encryption and decryption.

This narrows it down to a choice of one of `Alpine`, `Claws`, `Gnus`, `GroupWise`, `InScribe`, `Thunderbird`, `Mulberry`, `Mutt`, `SeaMonkey` or `SquirrelMail`. Of these, I shall examine only two in detail in this talk: `mutt`, because I know it very well, having used it under `Linux` for over 15 years; and `thunderbird`, because I have helped someone use it under `MacOSX` for the past 4 years.

4 Some Notes on Email: `mutt` and `thunderbird`

(For email systems, see `http://www.tldp.org/HOWTO/Mail-Administrator-HOWTO-3.html`)
(See `https://www.neomutt.org/` and see `https://www.mozilla.org/en-US/thunderbird/`)

`Mutt` and `thunderbird` occupy opposite ends of a spectrum.

`Mutt` is designed to read and write emails in a normal `Unix` environment, where there are other programs taking care of transporting email in and out of the computer and displaying various file types. `Mutt` is text-based, can do interactive and non-interactive email tasks, reads email in as text, and has no idea of how to actually get email (it relies on `fetchmail`), or how to send it (it hands it over to `exim`) or how to display images (it calls `xli` or `qiv`), PDFs (it calls `xpdf` or `evince`), or various types of documents (it simply calls `libreoffice`, which you configure it to call (in `~/.mailcap`), and install on your system).

`Thunderbird`, on the other hand, is a rather monolithic program that is starting to suffer from feature-creep; and under `Linux` there is recent evidence of instability. It relies on the plug-in `enigmail` to deal with encryption, but then mostly does everything else itself.

By the way, and *utterly weird*: you can send an email to yourself at your own computer using `mutt`, just by typing `'mutt user@localhost'` — but you cannot send an email to yourself using `thunderbird` !

5 mutt: Configure for Email Encryption

In a normal Linux system, we can happily get by with the packages gnupg, exim4, fetchmail and mutt.

5.1 Encryption and decryption (gnupg)

Refer to https://www.gnupg.org/gph/en/manual/c14.html
Install gpg: `# apt-get install gnupg`
Identify version: `$ /usr/bin/gpg --version`
```
gpg (GnuPG) 2.1.18
```
Now to generate your public and private keys for asymmetric encryption and decryption, you may refer to the procedure in https://linuxlsga.net/gpg.pdf on our site.
Run: `$ gpg --full-generate-key`

5.2 Establish your email system (exim4)

Refer to https://wiki.debian.org/Exim
Install exim: `# apt-get install exim4`
Identify version: `$ /usr/sbin/exim4 --version`
```
Exim version 4.89 #2 built 14-Jun-2017 05:03:07
```
Get a simple basic answer file linuxlsga.net/ee.exim.conf from our site, and edit it for your situation. Assume that your computer is named tux and is on your domain home.net and that your internet service provider mail computer is isp.mail. Adjust these assumpions as appropriate in what follows.
Configure it — run: `# dpkg-reconfigure exim4-config`
and you'll see these questions and can use the suggested answers:

```
exim4: simple configuration for home use
we assuming your computer is named 'tux' and is on your domain 'home.net'
and that your internet service provider mail computer is 'isp.mail'
adjust these assumpions as appropriate in what follows.
the '#' symbol means that you are running as the administrator of your computer.
run: # dpkg-reconfigure exim4-config
you'll see these questions and can use the suggested answers.

  mail sent by smarthost; received via SMTP or fetchmail        (select this one)
  system mail name:  home.net
  IP-addresses to listen on for incoming SMTP connections:      127.0.0.1
  Other destinations for which mail is accepted: tux.home.net;  tux
  Machines to relay mail for:                                   (leave this blank)
  IP address or host name of the outgoing smarthost:            isp.mail
  Hide local mail name in outgoing mail?                        yes
  Visible domain name for local users:                          home.net
  Keep number of DNS-queries minimal (Dial-on-Demand)?          yes
  Delivery method for local mail:                               mbox format in /var/mail/
  Split configuration into small files?                         no
```

5.3 Fetching email from your ISP (fetchmail)

Refer to http://newbiedoc.sourceforge.net/networking/fetchmail.html
Install fetchmail: `# apt-get install fetchmail`
Identify version: `$ /usr/bin/fetchmail -V`
```
This is fetchmail release 6.3.26+GSS+NTLM+SDPS+SSL-SSLv3+NLS+KRB5.
```

Configure `fetchmail` by getting the simple basic config file `linuxlsga.net/ee.fetchmailrc` from our site shown below. Edit it for your situation: replace `john` with your username on your computer; replace `john.smith@isp.net` with your-email-address account; insert your actual account password; replace `securemail.isp.net` with your ISP email computer. Then save it in a file named `.fetchmailrc` in your home directory and run: `$ chmod 0700 ~/.fetchmailrc.`

```
set postmaster "john"
set bouncemail
set no spambounce
set properties ""

poll securemail.isp.net with proto POP3 port 995
user 'john.smith@isp.net' there with password 'RdcjuHjddewWpOK' is 'john' here
options ssl
```

To get email: `$ /usr/bin/fetchmail`

5.4 Reading and writing email (mutt)

Refer to `https://www.neomutt.org/`
Install mutt: `# apt-get install mutt`
Identify version: `$ /usr/bin/mutt -v`
`NeoMutt 20170113 (1.7.2)`
Configure `mutt` by copying a simple basic config file `linuxlsga.net/ee.muttrc` from our site into a file named `.muttrc` in your home directory. Assume that your username on your computer is `john` and your email address is `john.smith@isp.net` and edit these lines below for that data and shove them into `$HOME/.muttrc`

```
set editor=vi     or    set editor=nano      (if you prefer nano)
set folder=~/Mail
set copy=yes
set postponed="~/Mail/postponed"
set spoolfile=/var/spool/mail/john
set mbox=~/mbox
my_hdr From: "john" <john.smith@isp.net>
set sort=threads
set edit_headers
set record="=sentmail"
unset write_bcc
unset bounce_delivered
```

5.5 Encryption options (mutt)

Refer to `http://codesorcery.net/old/mutt/mutt-gnupg-howto`
Because `mutt` is a curses-based application, you shall probably not want to use the `X-Window` version of the `pinentry` program — you expect to sometimes be logged in to a server which is not running `X`. So we shall use the `pinentry-curses` program instead.
Install it — run: `# apt-get install pinentry-curses`
and run: `# apt-get install gnupg-agent`
and then make sure that the file `$HOME/.gnupg/gpg-agent.conf` contains the line
`pinentry-program /usr/bin/pinentry-curses`

Update the configuration — run: `$ gpg-connect-agent reloadagent /bye`.
Further configure `mutt` by getting a simple basic config file `linuxlsga.net/ee.mutt.gpg` from our site, as listed below. It contains directives for `mutt` to use in any `gpg` encryption.Add it to the end of the file named `.muttrc` that is already in your home directory.

```
set pgp_decode_command="gpg %?p?--passphrase-fd 0? --no-verbose --batch \
                --output - %f"
set pgp_verify_command="gpg --no-verbose --batch --output - --verify %s %f"
set pgp_decrypt_command="gpg --passphrase-fd 0 --no-verbose --batch --output - %f"
set pgp_sign_command="gpg --no-verbose --batch --output - --passphrase-fd 0 \
                --armor --detach-sign --textmode %?a?-u %a? %f"
set pgp_clearsign_command="gpg --no-verbose --batch --output - --passphrase-fd 0 \
                --armor --textmode --clearsign %?a?-u %a? %f"
set pgp_encrypt_only_command="pgpewrap gpg --batch --quiet --no-verbose --output - \
                --encrypt --textmode --armor --always-trust -- -r %r -- %f"
set pgp_encrypt_sign_command="pgpewrap gpg --passphrase-fd 0 --batch --quiet \
                --no-verbose --textmode --output - --encrypt --sign %?a?-u %a? \
                --armor --always-trust  -- -r %r -- %f"
set pgp_import_command="gpg --no-verbose --import -v %f"
set pgp_export_command="gpg --no-verbose --export --armor %r"
set pgp_verify_key_command="gpg --no-verbose --batch --fingerprint --check-sigs %r"
set pgp_list_pubring_command="gpg --no-verbose --batch --with-colons --list-keys %r"
set pgp_list_secring_command="gpg --no-verbose --batch --with-colons --list-secret-keys %r"
set pgp_replyencrypt=yes
set pgp_timeout=1800
set pgp_good_sign="^gpg: Good signature from"
```

Notice the use of the program `pgpewrap` which refers to the program `/usr/lib/mutt/pgpewrap`, which gets installed with the `mutt` package. Also note that you can supress the `GPGME unavailable` message by altering `set crypt_use_gpgme=yes` in the file `/etc/Muttrc.d/gpg.rc` to `set crypt_use_gpgme=no`.

6 `mutt`: Use for Email Encryption

In what follows, I shall show you how to master these four steps:
(1) ensure that you and your correspondent have exchanged your public keys;
(2) send an encrypted message plus attachments to a person;
(3) send a signed and encrypted message plus attachments to a person;
(4) receive and decode an encrypted message from a person.

6.1 Exchange your public keys

This explains how you and your correspondent can exchange your public keys.
Ensure that you have at least one public key available or the mutt program hangs when trying to exchange them. Start a message to them:
`$ mutt them@their-email-address`
When finished your message about your public key being attached, press `'ESC+k'` to attach a public key for them. You'll see `Please enter the key ID:`, which would normally be your email address. You'll see a list of possible public keys; use the up/down arrows to select the right one and press `ENTER`; you'll see `Invoking PGP ...`; press `'y'` to send the email.

Now it is vital that they confirm that your public key really came from you by telephoning you and asking you to read out the result of performing this hash function on your public key:

`$ gpg --export --armour me@my-email | sha512sum`
and compare it with the result that they obtain by doing the same on their computer.
Ensure that the person has sent you their public key for replying:
Get new mail: `$ fetchmail`
Read emails: `$ mutt`
Highlight their message using the up/down arrows and open their email by pressing `'ENTER'`. You shall see some attachments. View them by pressing `'v'`. Select their public key using the up/down arrows; it should be the one containing their email address. When highlighted press `'CTRL+k'` to import it into your computer. Now it is vital that you confirm that their public key really came from them by telephoning the person and asking them to read out the result of performing this hash function on their public key:
`$ gpg --export --armour them@their-email | sha512sum`
and compare it with the result that you obtain by doing the same on your computer.

6.2 Send an encrypted message

This is how you can send an encrypted message plus its attachments to a person:
Construct your message: `$ mutt person@email.address`
Enter the subject and message as usual. When done, exit the editing program `'vi'` or `'nano'`.
Add attachments by pressing `'a'` and entering the path to the files. You'll see a list of attachments.
To encrypt, press `'p'` (for `'PGP'`). Along the bottom you'll see:
`PGP (e)ncrypt, (s)ign, sign (a)s, (b)oth, s/(m)ime or (c)lear?`
Press `'e'` to encrypt. You'll see another line appear in the header area:
`Security: Encrypt (PGP/MIME)`
When you are ready to send the email, press `'y'`. Along the bottom you'll see:
`Enter keyID for person@emailaddress:`
It is asking for a unique way of selecting the public key to which to encrypt. There might be other keys owned by the same person. Just enter the person's email address, followed by `'ENTER.'`
You'll see a list of all possible keys containing that name. Move up/down with the arrows and select the right one by pressing `'ENTER'`. The whole email will be encrypted and sent, and `mutt` exits.

6.3 Send a signed and encrypted message

In addition to encryption of an email message and its attachments, digitally signing an email helps to assure: authentication (it came from me), non-repudiation (I cannot deny I sent it) and integrity (it has not been tampered with on the way).

This explains how you can send a signed and encrypted message plus attachments to a person.
First, create a message to them: `$ mutt person@email.address`
Enter the subject and message as usual. When done, exit the editing program `'vi'` or `'nano'`.
Add attachments by pressing `'a'` and entering the path to the files. You'll see a list of attachments.
To encrypt, press `'p'` (for `'PGP'`). Along the bottom you'll see:
`PGP (e)ncrypt, (s)ign, sign (a)s, (b)oth, s/(m)ime or (c)lear?`
This time, press `'b'` to both encrypt and sign. You'll see more lines appear in the header area:
`Security: Encrypt (PGP/MIME)`
`Security: Sign, Encrypt (PGP/MIME)`
When you are ready to send the email, press `'y'`. Along the bottom you'll see:
`Enter keyID for person@email.address:`
It is asking for a unique way of selecting the public key to which to encrypt. There might be other keys owned by the same person. Just enter the person's email address, followed by `'ENTER.'`
You'll see a list of all possible keys containing that name. Move up/down with the arrow keys and select the right one by pressing `'ENTER'`. The whole email will be encrypted, signed and sent, and `mutt` exits.

6.4 Receive an encrypted message

This explains how you can receive an encrypted message and encrypted attachments from a person.
Look at your email: `$ mutt`
Scroll down to the email using the arrows. The flags will show `'NP+'` next to the message to indicate it is `'new'` and encrypted with `'PGP'` and is only for you.
Select it by pressing `'ENTER'` while it is highlighted. Along the bottom you'll see:
`Invoking PGP...`
and a `pin-entry` window comes up asking for the passphrase that protects your private key. Type it into the box and click `'OK'`, and you'll see the message in the clear, with, along the bottom:
`PGP message successfully decrypted.`
Finally press `'v'` to view the attachments, `'q'` to exit `mutt`.

7 `thunderbird`: Configure for Email Encryption

You may conveniently configure `thunderbird` for encryption by installing `gpg` and the add-on `enigmail`. Refer to the site `https://securityinabox.org/en/guide/thunderbird/mac/` for detailed use of `thunderbird` encryption: exchanging keys, encrypting, signing, sending, receiving, and other topics. This article is quite comprehensive and I shall not repeat it here, just follow it and prepare your system.

8 `thunderbird`: Use for Email Encryption

Once again, refer to the excellent `https://securityinabox.org/en/guide/thunderbird/mac/` for detailed use of `thunderbird` encryption: exchanging keys, encrypting, signing, sending, receiving, *etc.*

9 Conclusion

If you want some *practice* using these tools, just come along to our Friday afternoon Linux Learners Lounge between 2pm-4pm, and we can try out all this stuff with you until we get it right.

See our website for more information.
`linuxlsga.net`
