OpenPGP Email Encryption Workshop

by Alex Chesnakov

About me

- Alex Chesnakov 
- Senior Engineer, Security Manager @ Health Gorilla
- Security
- Linux, AWS, CTFs

- You may see me sailing or flying kites at Shoreline Lake

All opinions are my own and not those of my employer.

Goals

- Introduction to OpenPGP encryption
- Create OpenPGP keys
- Setup OpenPGP encryption in Thunderbird email client
- Exchange encrypted email messages with 2-3 recipients
- Encrypt, Decrypt and Sign text / files

Encryption

Encryption Algorithms:

- Symmetric (AES256, encrypted zip file)
- Asymmetric (OpenPGP)

Usage:

- In transit (http**s**://, WIFI encryption)
- At Rest (Mac and Windows disk encryption)
- End-to-End Encryption (OpenPGP)

Why

- Before 1996 - Illegal to export
- Encryption ban proposal in the UK
- Encryption ban proposal in EU
- Encryption ban proposal in the US

- I want to encrypt a few files
- No encryption necessary, I just want to be sure email indeed comes from the person
- I found a 0-day but company insists I use their OpenPGP to encrypt email before sending
- I downloaded .iso or program, they suggest to check signature, but I don't know how

Terminology

- PGP - Pretty Good Privacy Software. Proprietary, PGP Corp -> Symantec -> Broadcom
- GPG - GNU Privacy Guard Software. Free and open-source
- OpenPGP – Standard. PGP and GPG are OpenPGP-compliant

- Plaintext - "*Secret Message*" or binary file, like photo
- Ciphertext - "*Frperg Zrffntr*" encrypted text
- Cipher - algorithm for performing encryption/decryption. ROT13: A-N, B-O, C-P
- Hash - function map data of arbitrary size to fixed-size value output
  - Hash always has the same length for same hash algorithm (MD5, SHA256)
  - Change one bit in input, 50% different bits in hash
  - The same input always gives the same hash
  - One way function - cannot recover plaintext from a hash

Goals of PGP Cryptography

- **Confidentiality** - message contents remain private
- **Integrity** - refers to keeping a message unchanged
- **Nonrepudiation** - person cannot deny signing a particular message

- **Authenticity** - All these effects occurring simultaneously.

Main Usage Examples

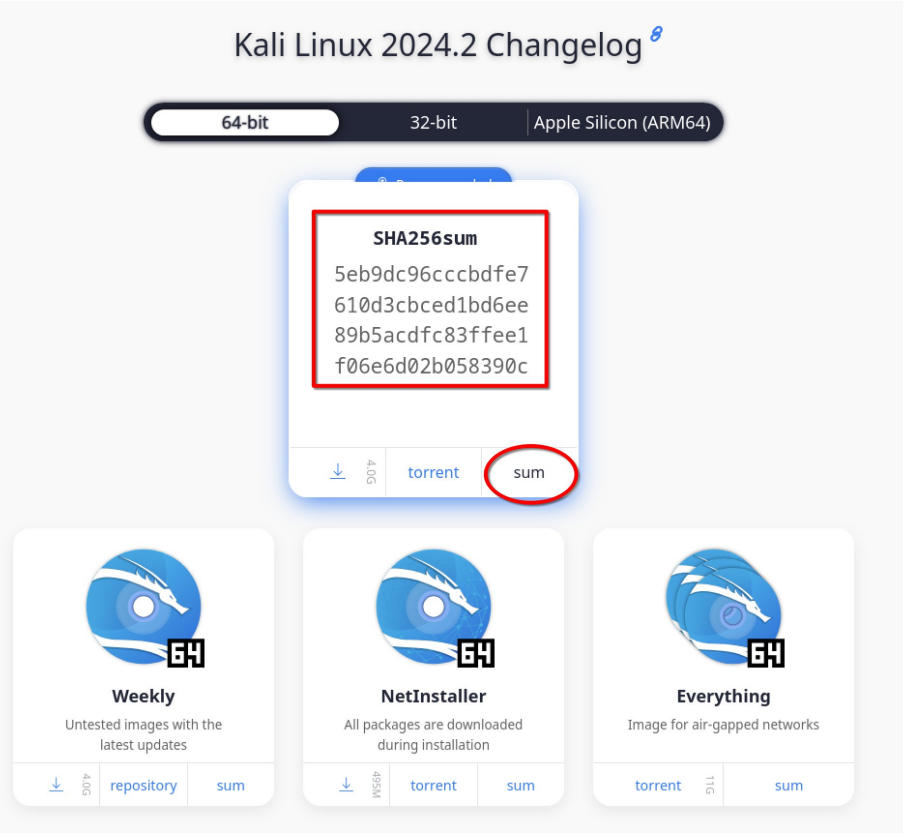
Goal	How
I want anyone who reads this message to be sure I sent it and it was not changed (**Nonrepudiation, Integrity**)	Digitally sign the message with your private key
I want to verify the sender's identity	Verify signature with the sender's public key.
I want to send a message that only my intended recipient can read (**Confidentiality**)	Encrypt the message with the recipient's public key
I want to decrypt a message that I received	Decrypt the message with your private key
I want my message to be readable only by recipient, and I want recipient to be sure message came from me (**Authenticity**)	Encrypt the message with the recipient's public key and digitally sign the message with your private key
I want to decrypt and verify a message that includes a digital signature	Decrypt the message with your private key and verify the signature with the sender's public key

Confidentiality

Confidentiality means that the message contents remain private. The *plaintext* cannot be viewed by anyone who doesn't have the necessary keys.
In many cases, you cannot prevent someone from viewing the *ciphertext*, because every message over the Internet can be viewed by a large number of people, just as postcards can be viewed by a postman or a neighbor.

Integrity

Integrity refers to keeping a message unchanged. By using OpenPGP, you can confirm that a message has not been tampered with during transmission.



Nonrepudiation

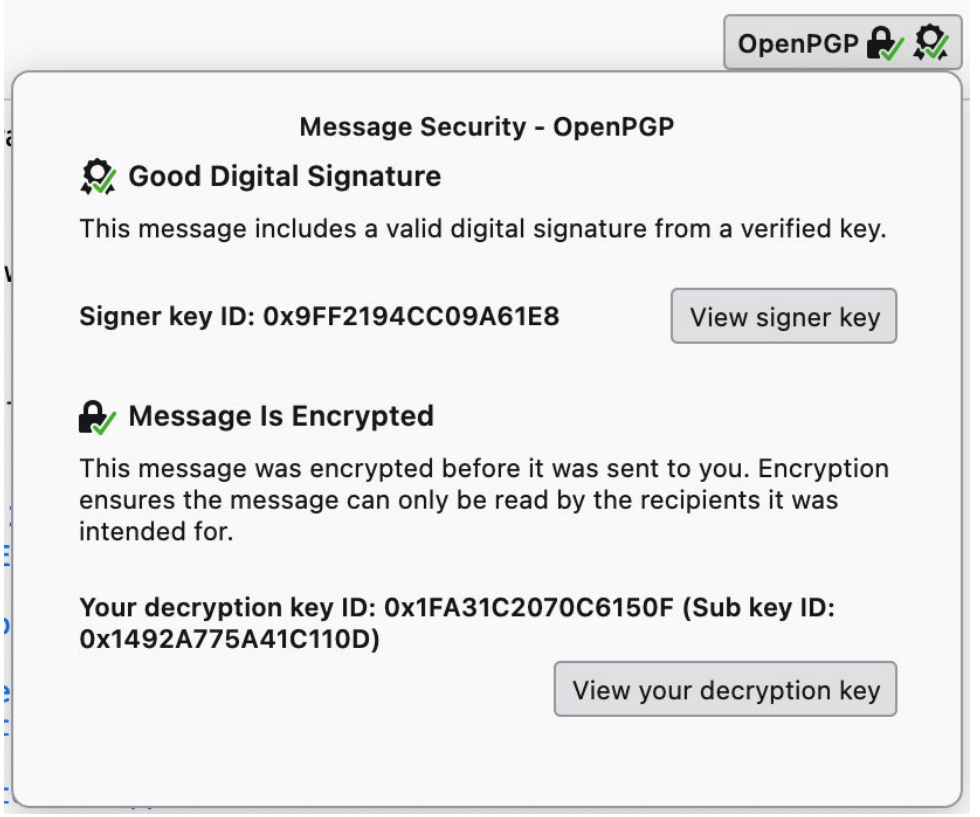
Nonrepudiation means that a person cannot deny signing a particular message, which is especially important in the context of email.
Example of nonrepudiation:

- Software vendors **Sign** program, drivers, .iso files
- DocuSign

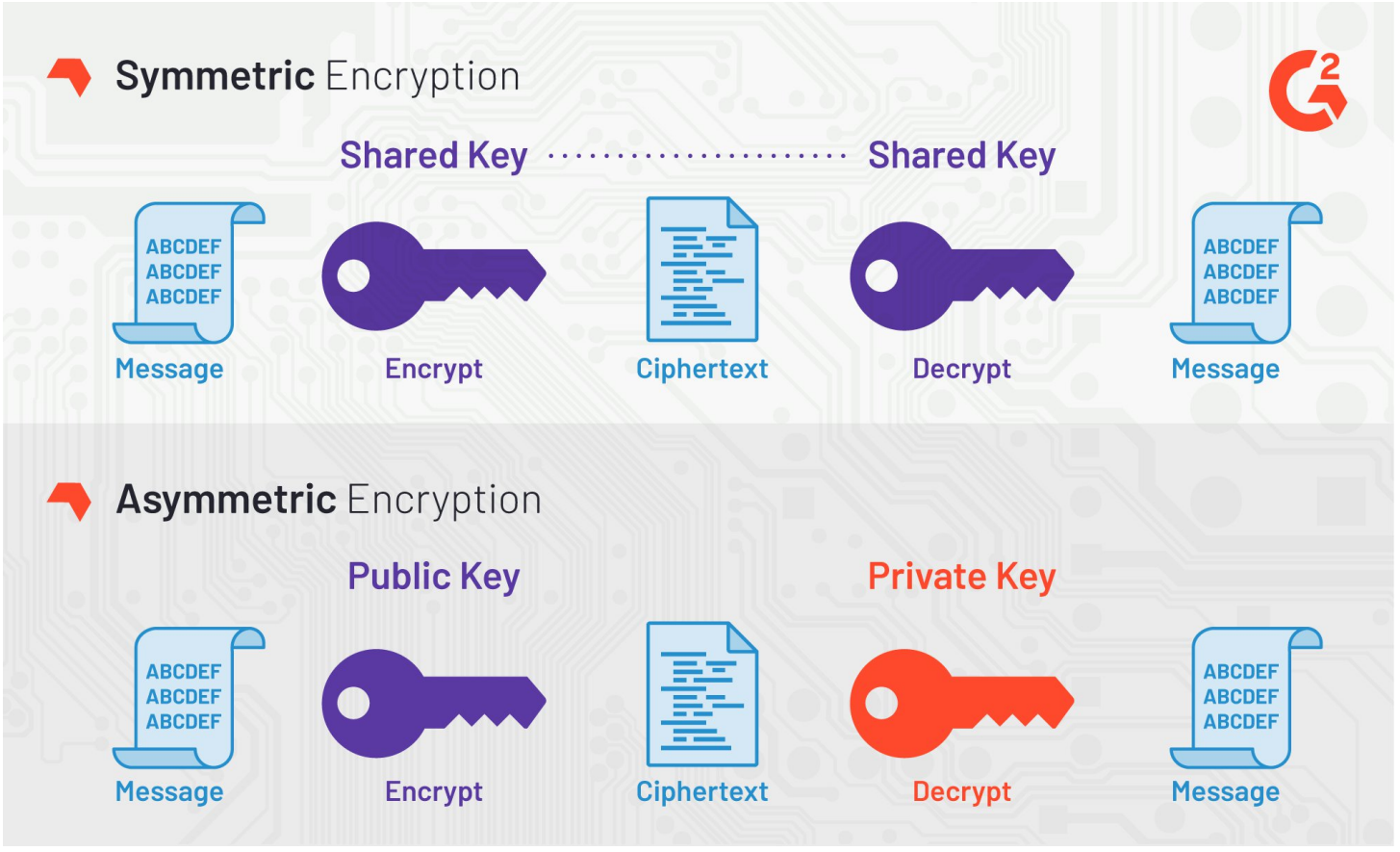
DocuSign®

Authenticity

When you receive an email that has been **encrypted** and **signed** with OpenPGP, you know that the contents of the message have been **concealed** from any eavesdroppers. You know that the content of the message has **not** been **changed**. You also know that the message **comes** from the **person** who has the right to send such a message in the sender's name



Symmetric vs Asymmetric Encryption



Asymmetric Cryptography

- **Pair of keys**
  Encrypt with one key, decrypt with another key.
  You can encrypt with public key and decrypt only with private key
  You can encrypt with private key and decrypt only with public key
- **Public key** – share the key with anyone – Public-Key Cryptography
- **Private key** – do not share, keep private
- **Keyserver** – server that provides directory of OpenPGP keys

Thunderbird – easy way to start with OpenPGP

Thunderbird - https://www.thunderbird.net/

Mac:

```
brew install --cask thunderbird
```

Kali Linux (Thunderbird is probably installed already):

```
sudo apt update
sudo apt install thunderbird
```

Windows:

https://www.thunderbird.net/ Click Download. Install

Thunderbird: Add Account

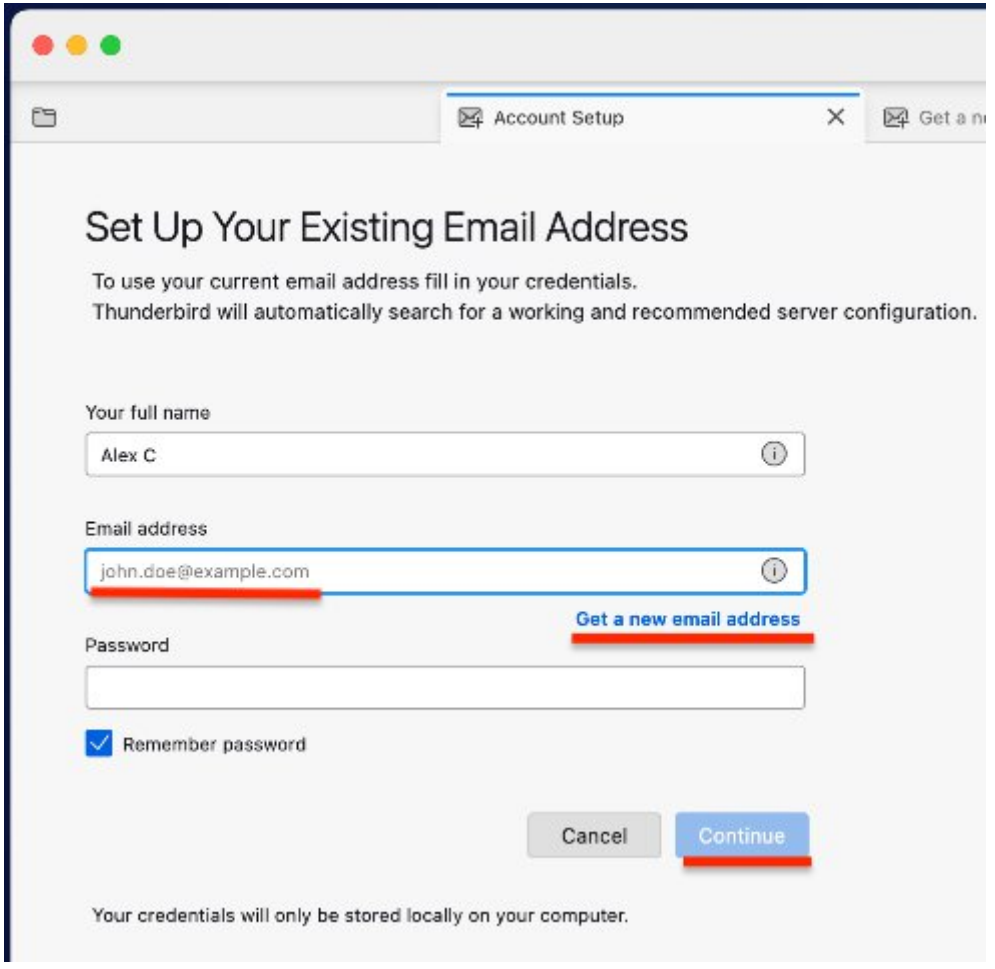
Sign in with your existing email

NOTE: For Google and Outlook
do not enter password, Click Continue

OR

Get a new email address

NOTE: The new address by will work only two weeks in Thunderbird (IMAP protocol) but you can keep on using Mailfence in Thunderbird with a paid subscription



Thunderbird: Generate new Key Pair

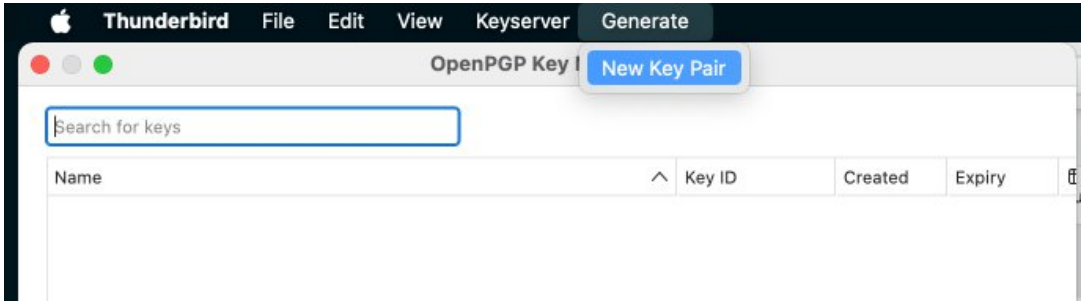
In Thunderbird:

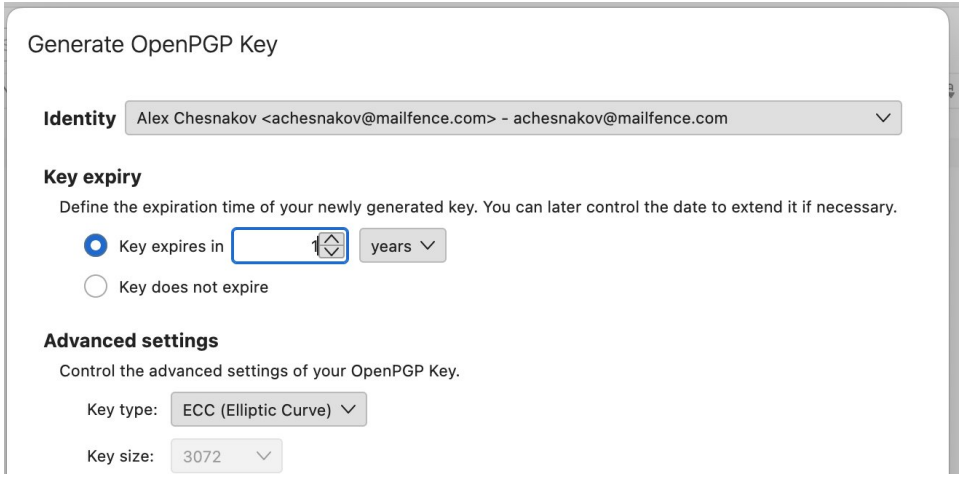
Tools -> OpenPGP Key Manager

Generate -> New Key Pair

Defaults will work

You may Expire your first keys in 1 year and use ECC Key (much shorter keys)



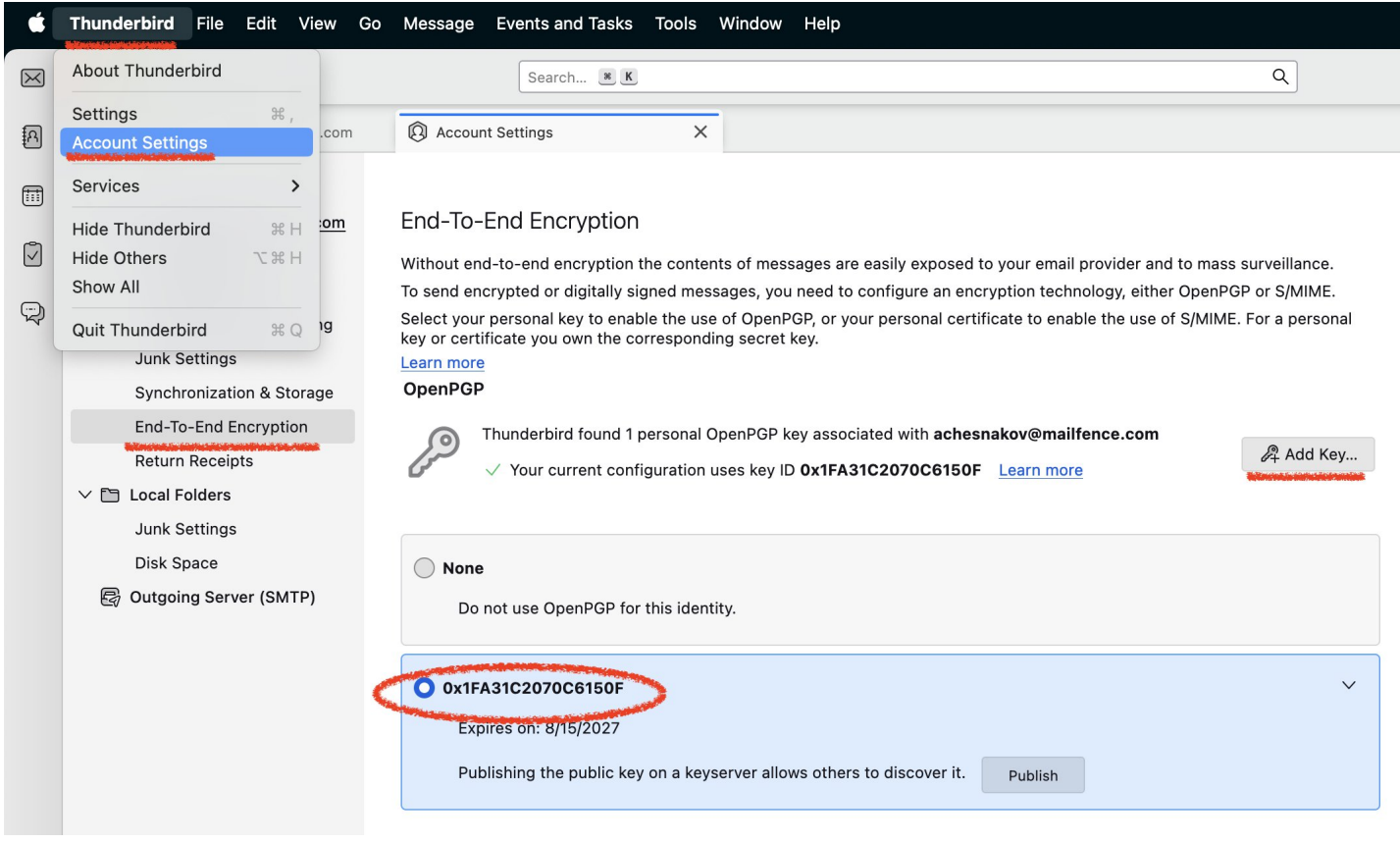


Thunderbird: use the OpenPGP Keys

In Thunderbird:

Thunderbird ->
Account Settings ->
End-To-End Encryption ->
Choose your Keys

May need to Add Key
(if keys do not shows up)

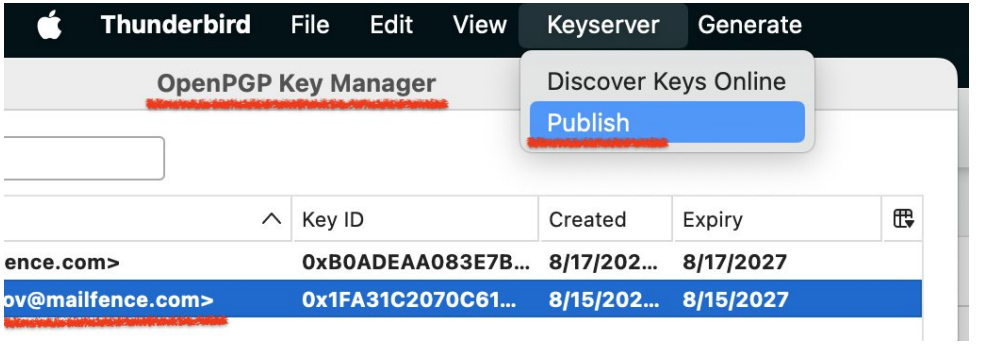


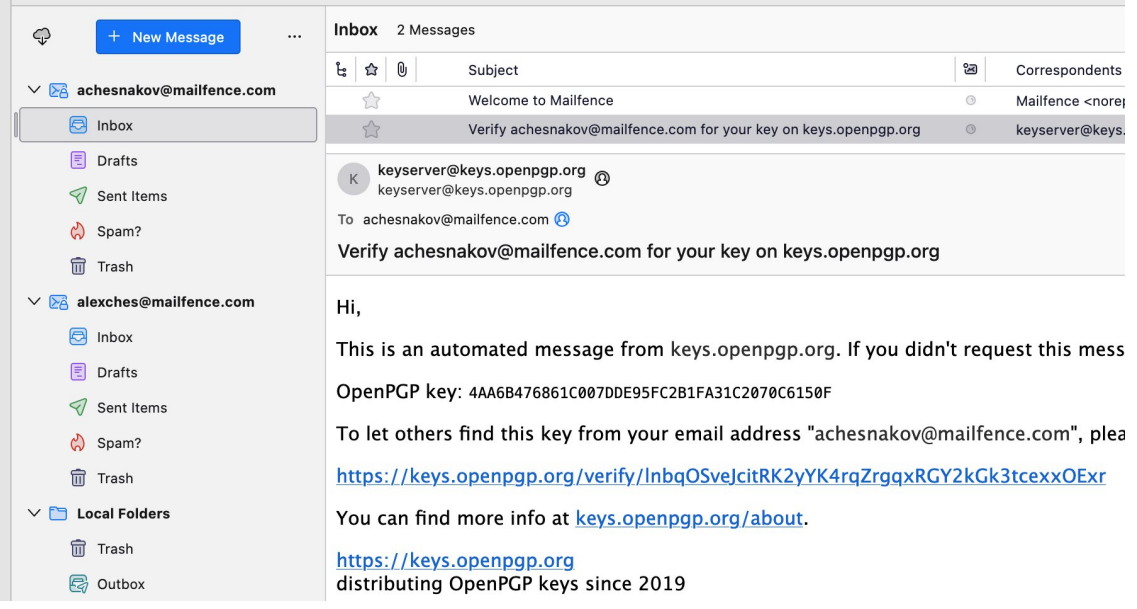
Publish Keys (Optional)

You may Publish the key
This will help others find your OpenPGP keys
but gives away info about you (privacy)

You should receive an email and confirm that you the owner of this email account

This will make the key discoverable through the search at https://keys.openpgp.org/

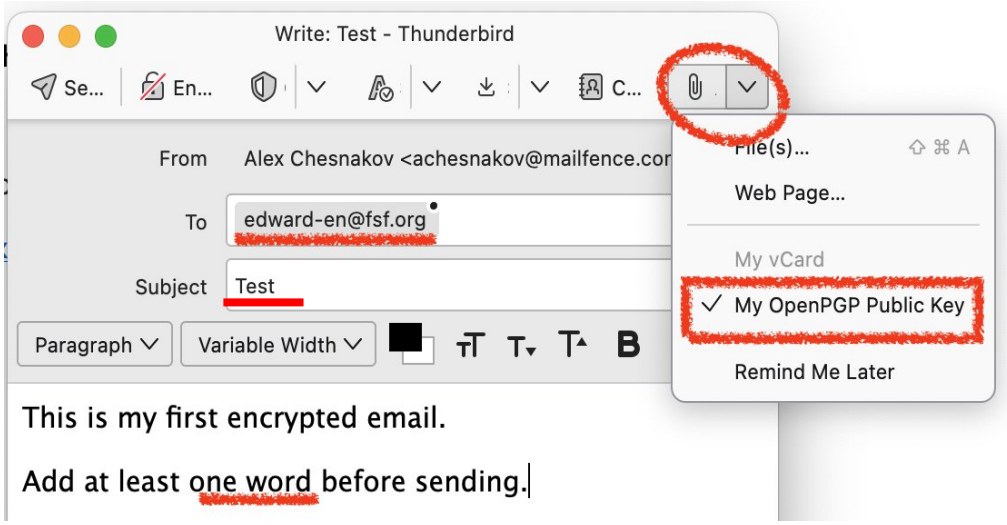




Email Edward your PubKey

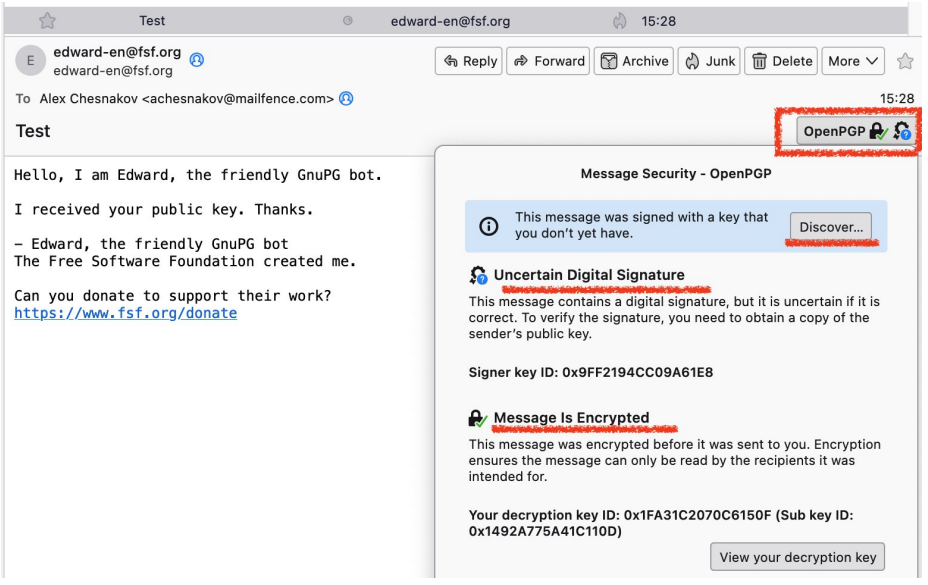
Free Software Foundation (FSF) project to help start with OpenPGP

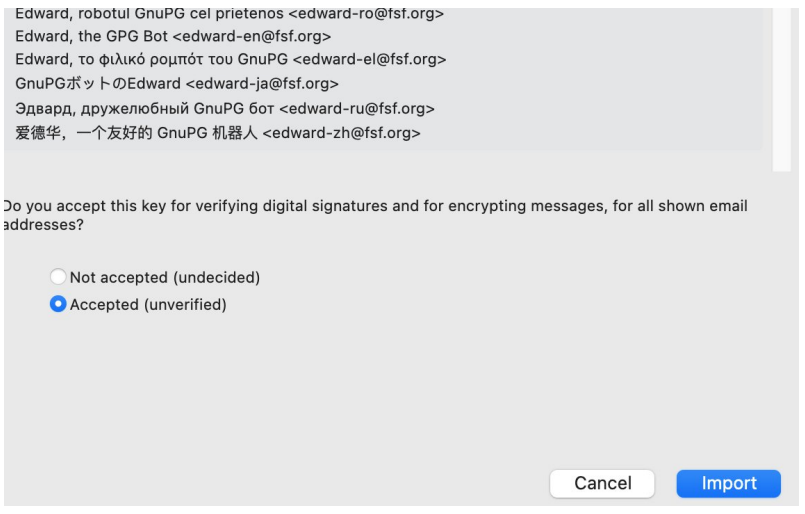
- Create email and attach Public Key
- Add one word to **Subject** & **Body**
- Sent to **`edward-en@fsf.org`**



Receive First Encrypted Email

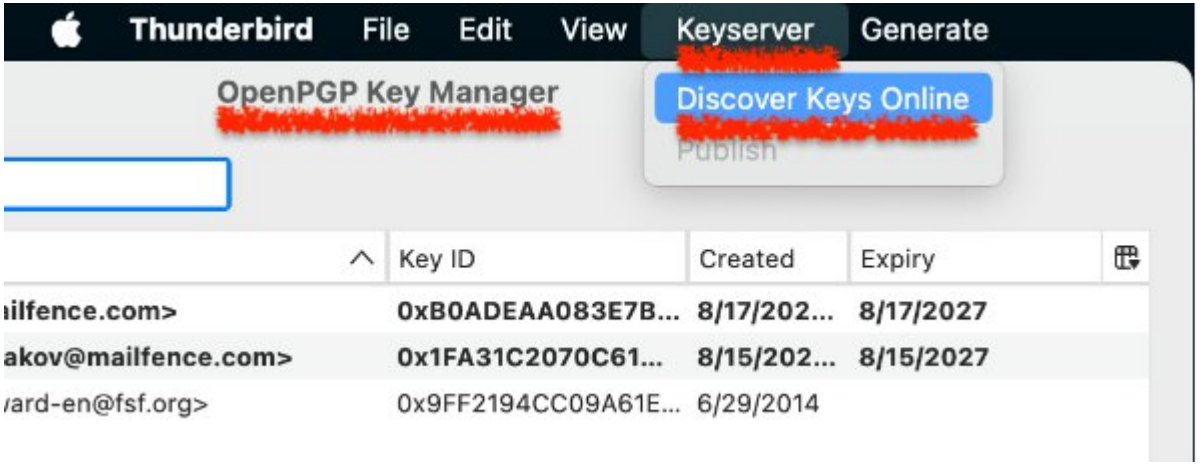
- Congrats, you received Encrypted Email
- Edward has our Public Key and can encrypt message with our PubKey
- We haven’t added Edward’s PubKey so we are not sure if Edward is Edward
- We can Click Discover
- We initiated the email, we can Import and choose Import Accepted





Discover PubKeys Online

Thunderbird->
Tools->
OpenPGP Key Manager->
Keyserver->
Discover Keys Online

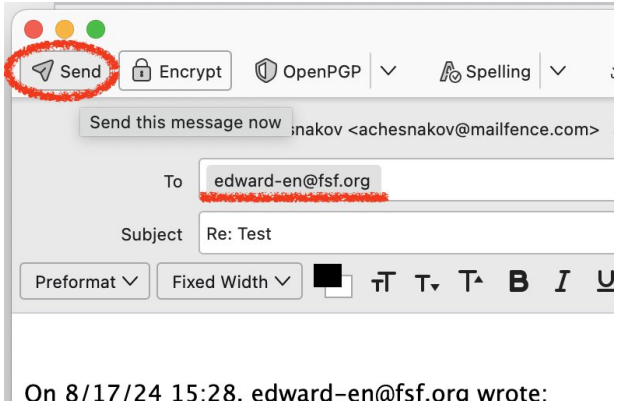


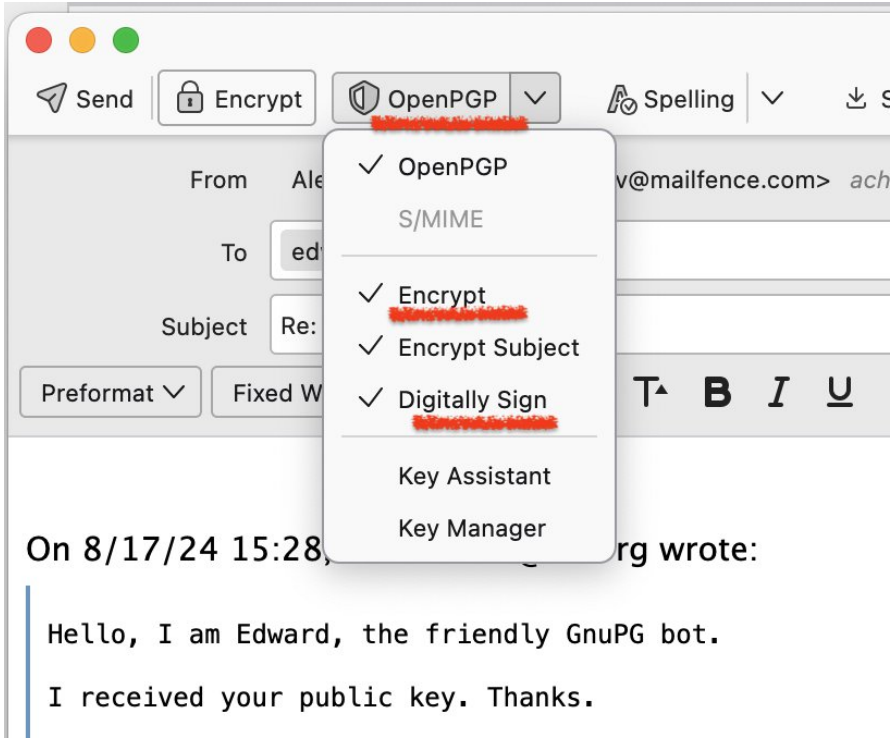
Try to find and Import

- edward-en@fsf.org
- achesnakov@mailfence.com

Send First Encrypted Email

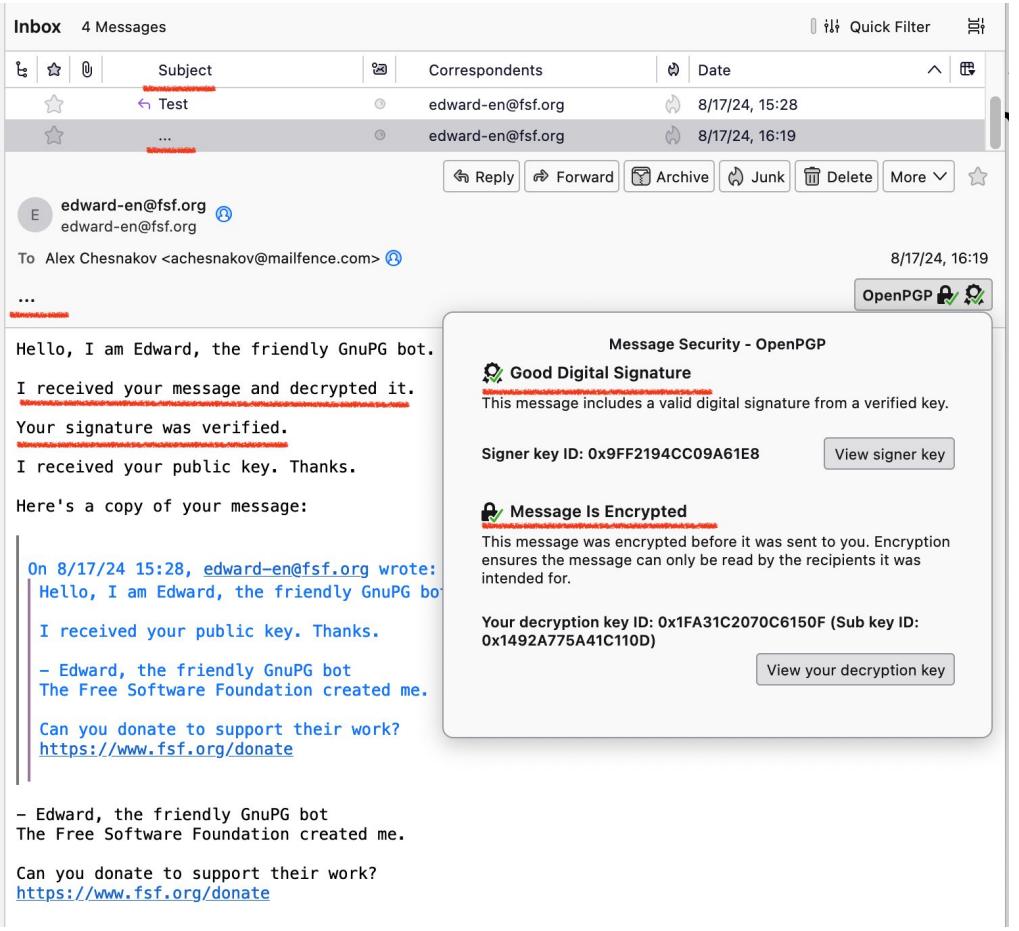
- Reply to Edward and be sure to:
  - Encrypt
  - Digitally Sign
- Click Send
- Receive a response message
- Notice three dots instead of Subject





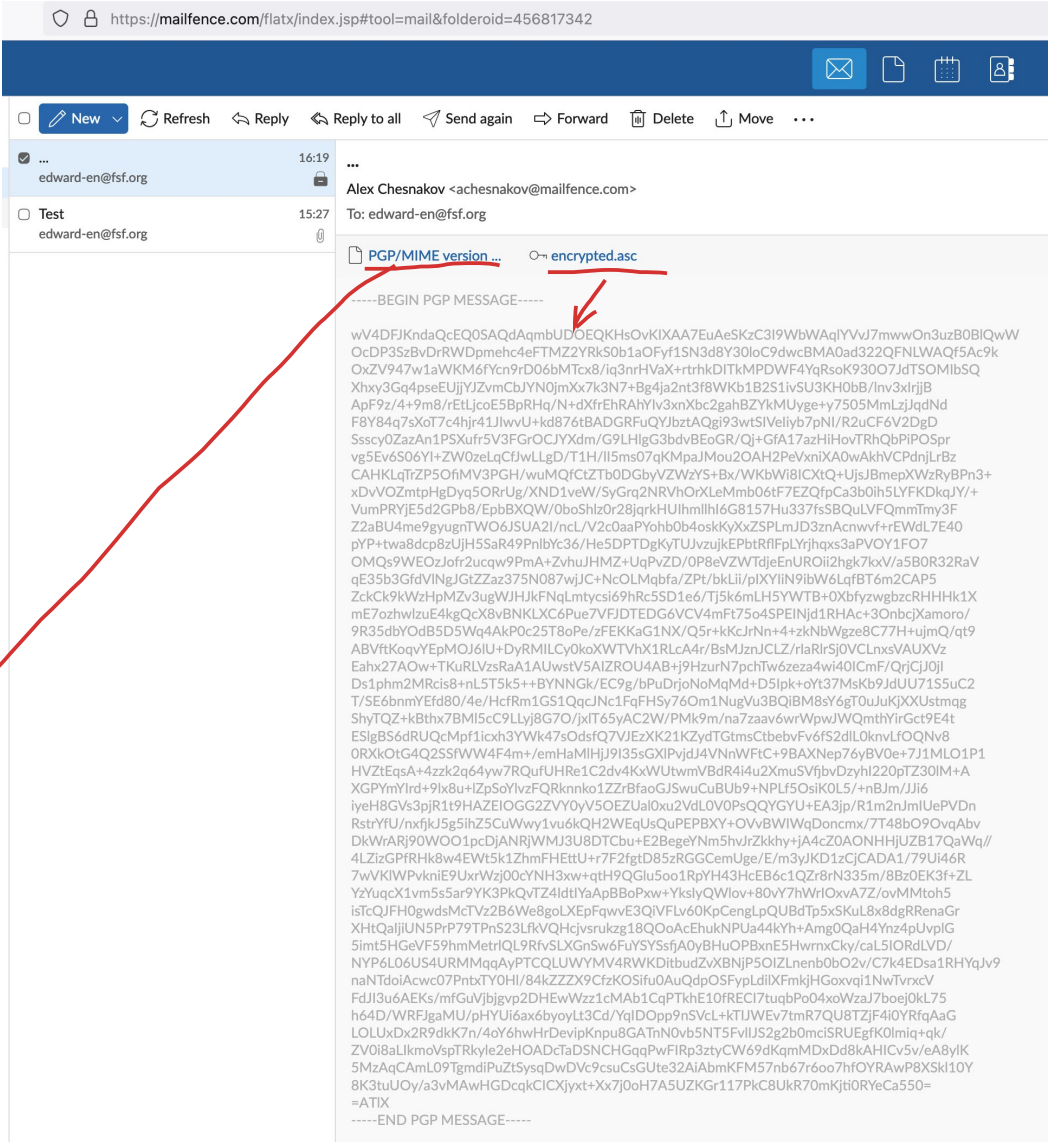
Receive an Encrypted Email from Edward

- Receive a response message
- Notice three dots instead of Subject



Check if email is encrypted

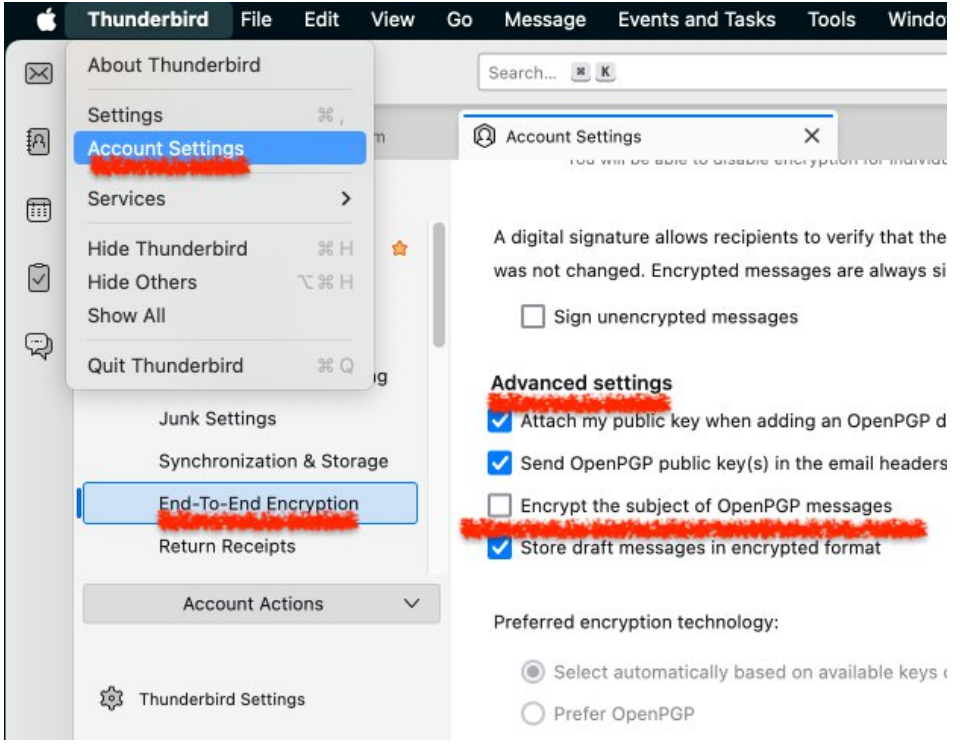
- Log into email provider web portal
- See how the message is looks like
- Two Files
  - encrypted.asc
  - PGP/MIME version identification
- Mailfence will offer to decrypt, but you have to upload your PRIVATE Key



Subject Encryption

- OpenPGP do not offer Encryption for Subject
- Thunderbird encrypts Subject by default
- They use non-standard implementation
- Works fine between Thunderbird clients
- Do not work between everything else
- Choose you subject (“Hello”, “Test”)

- **Do not encrypt subject**
- Go to Account Settings->
  End-To-End Encryption->
  Advanced Settings->
- Uncheck Encrypt Subject

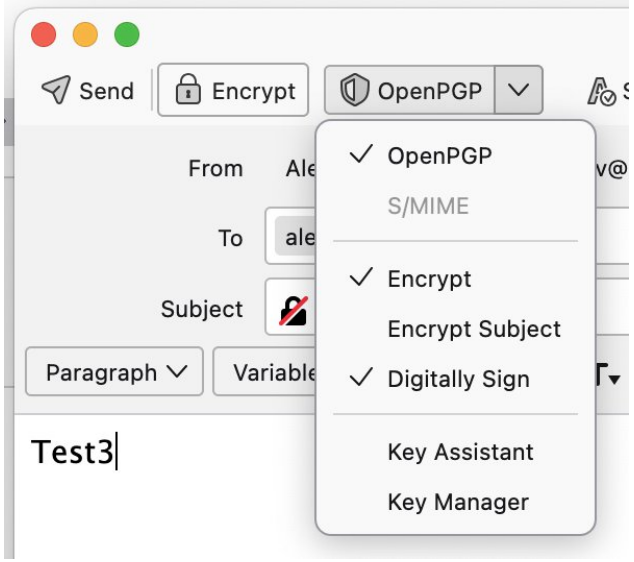


Exchange encrypted email with a person

- Ask somebody around for their email that has OpenPGP keys
- As a backup, you can send email to me - achesnakov@mailfence.com
- Search for their keys at Thunderbird's OpenPGP Key Manager or at https://keys.openpgp.org/
- Import keys
- Click on the keys and Accept keys with "Yes, I've verified in person..."
- Create a new email to this person
- Be sure you Encrypt and Digitally sign the message

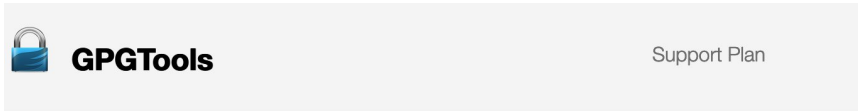
Other person:

- Receive the email
- Import Keys
- Click Reply to the email
- Encrypt and Digitally Sign the email before Sending



Advanced use of GnuPG (GPG)

Mac



Windows

Gpg4win Components

Gpg4win is an installer for Windows and contains several Free Software components:

GnuPG
The backend; this is the actual encryption tool.

Kleopatra
A certificate manager for OpenPGP and X.509 (S/MIME) and common crypto dialogs.

GpgOL
An add-in for Microsoft Outlook providing email cryptography. Supports both IMAP/SMTP and MS Exchange servers.

GpgEX
A plugin for Microsoft Explorer (file encryption).

Install GnuPG (CLI) and GUI tools

GnuPG - https://gnupg.org/

Mac. GPG Suite. Includes CLI and GUI:

```
brew install --cask gpg-suite
```

Kali Linux. GnuPG and Kleopatra (gpg is pre-installed):

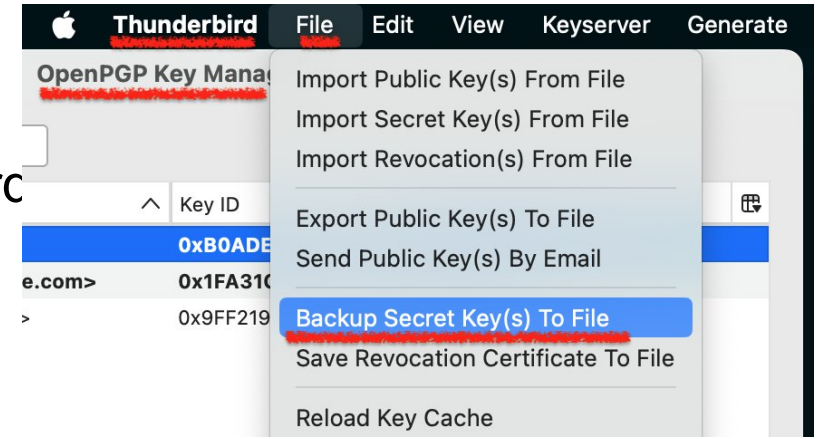
```
sudo apt update
sudo apt install gpg kleopatra
```

Windows. Gpg4win. Includes GnuPG and Kleopatra:

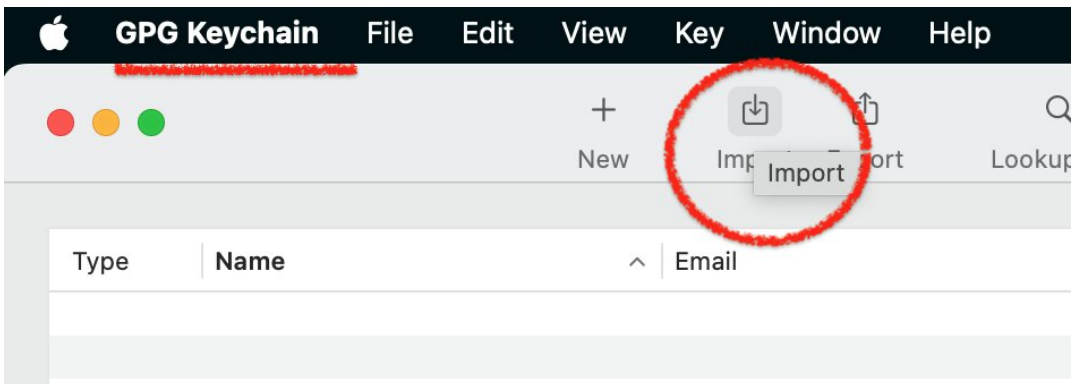
https://gpg4win.org/ Click Download. Install

Export Keys from Thunderbird to GnuPG

- Thunderbird can’t sync keys with GnuPG
- Export Secret (Private) Keys from Thunderbird
- Thunderbird will export Private and Pub Keys
- Choose Password for Private (Secret) Keys

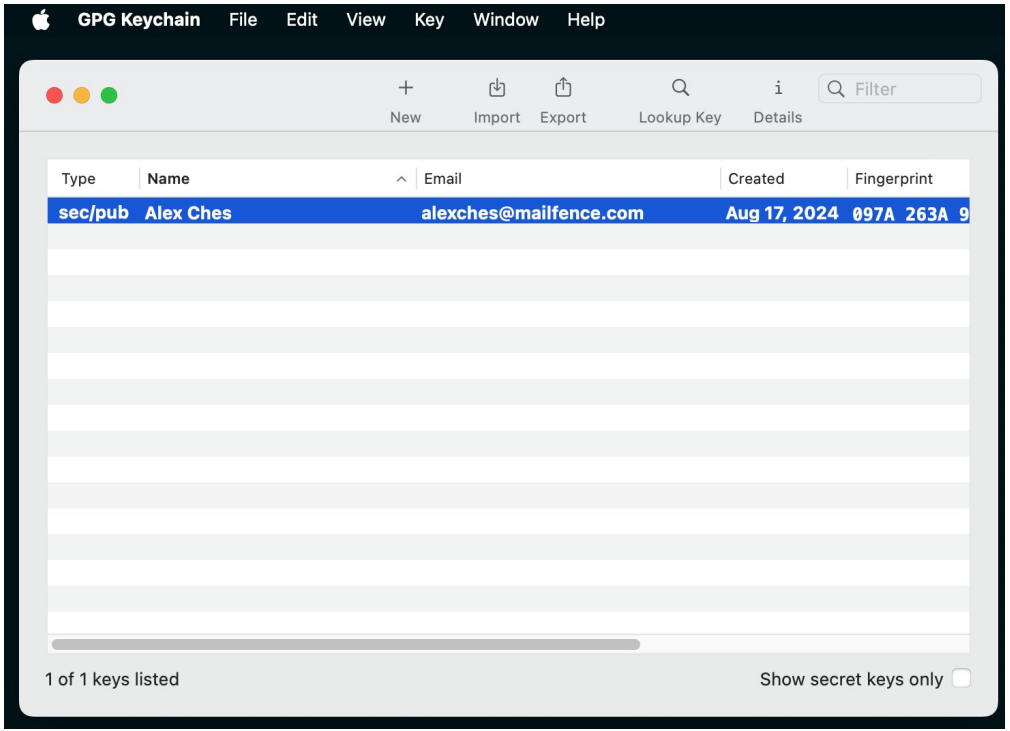


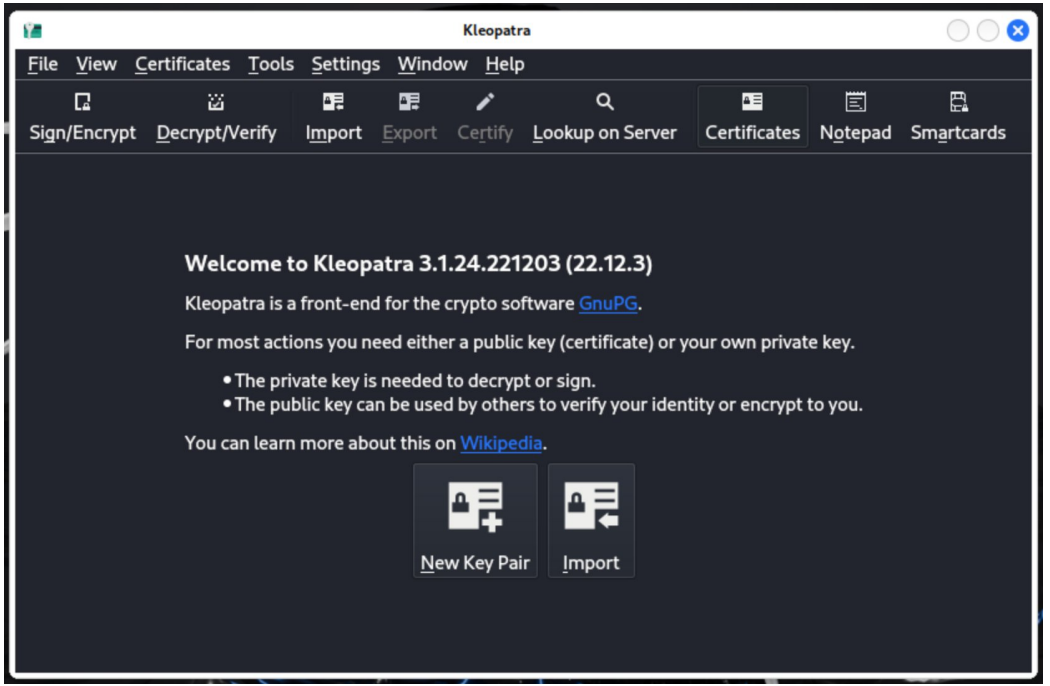
- Import keys to GPG Keychain or Kleopatra



CLI vs GUI

- CLI version offers full functionality
- For simple use both CLI and GUI are good
- CLI version can be used in scripts and automation





List Keys

- **`gpg --list-keys`** – List Public Keys
- **`gpg --list-secret-keys`** – List Private Keys

```
20:02 alexc@Alexs-MBP /Users/alexc
% gpg --list-keys
/Users/alexc/.gnupg/pubring.kbx
------------------------------
pub   ed25519 2024-08-17 [SC] [expires: 2027-08-17]
      097A263A90BF7916CFC41014B0ADEAA083E7BE88
uid           [ultimate] Alex Ches <alexches@mailfence.com>
sub   cv25519 2024-08-17 [E] [expires: 2027-08-17]

pub   ed25519 2024-08-16 [SC] [expires: 2027-08-16]
      4AA6B476861C007DDE95FC2B1FA31C2070C6150F
uid           [ unknown] Alex Chesnakov <achesnakov@mailfence.com>
sub   cv25519 2024-08-16 [E] [expires: 2027-08-16]

pub   rsa2048 2014-06-29 [SC]
      F357AA1A5B1FA42CFD9FE52A9FF2194CC09A61E8
uid           [ unknown] Edward, the GPG Bot <edward-en@fsf.org>
uid           [ unknown] Edward the GPG Bot <edward@fsf.org>
uid           [ unknown] GnuPGボットの Edward <edward-ja@fsf.org>
uid           [ unknown] Edward, l'amichevole bot GnuPG <edward-it@fsf.org>
uid           [ unknown] Edward, le gentil robot de GnuPG <edward-fr@fsf.org>
uid           [ unknown] Edward, el simpático robot GnuPG <edward-es@fsf.org>
uid           [ unknown] Edward, o amigo robô de GnuPG <edward-pt-br@fsf.org>
uid           [ unknown] Edward, robotul GnuPG cel prietenos <edward-ro@fsf.org>
uid           [ unknown] Edward, arkadaş canlısı GnuPG botu <edward-tr@fsf.org>
uid           [ unknown] Edward, der freundliche GnuPG Roboter <edward-de@fsf.org>
uid           [ unknown] Эдвард, дружелюбный GnuPG бот <edward-ru@fsf.org>
uid           [ unknown] Edward, το φιλικό ρομπότ του GnuPG <edward-el@fsf.org>
uid           [ unknown] 爱德华，一个友好的 GnuPG 机器人 <edward-zh@fsf.org>
sub   rsa2048 2014-06-29 [E]
```

```
20:03 alexc@Alexs-MBP /Users/alexc
% gpg --list-secret-keys
/Users/alexc/.gnupg/pubring.kbx
------------------------------
sec   ed25519 2024-08-17 [SC] [expires: 2027-08-17]
      097A263A90BF7916CFC41014B0ADEAA083E7BE88
uid           [ultimate] Alex Ches <alexches@mailfence.com>
ssb   cv25519 2024-08-17 [E] [expires: 2027-08-17]

sec   ed25519 2024-08-16 [SC] [expires: 2027-08-16]
      4AA6B476861C007DDE95FC2B1FA31C2070C6150F
uid           [ unknown] Alex Chesnakov <achesnakov@mailfence.com>
ssb   cv25519 2024-08-16 [E] [expires: 2027-08-16]
```

Export Keys

```
gpg --export-secret-keys --armor [keyID] > my_pgpkey.private.asc
gpg --export --armor [keyID] > my_pgpkey.public.asc
```

`[keyID]` – **Last 8** or **16** digits of Fingerprint
may use Email Address
or Full Fingerprint

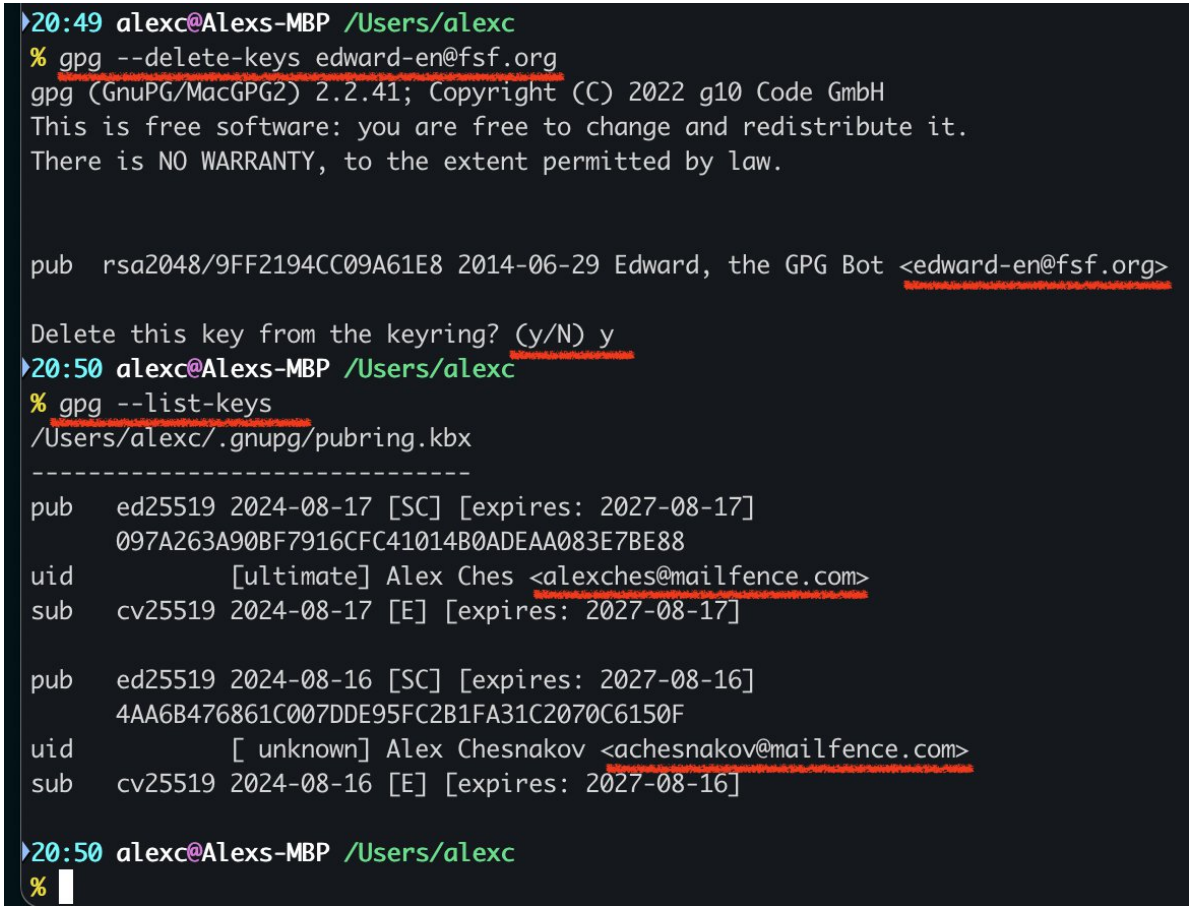
`--armor` – Export as ASCII text

```
20:36 alexc@Alexs-MBP /Users/alexc
% gpg --list-keys edward-en@fsf.org
pub   rsa2048 2014-06-29 [SC]
      F357AA1A5B1FA42CFD9FE52A9FF2194CC09A61E8
uid           [ unknown] Edward, the GPG Bot <edward-en@fsf.org>
uid           [ unknown] Edward the GPG Bot <edward@fsf.org>
uid           [ unknown] GnuPGボットのEdward <edward-ja@fsf.org>
uid           [ unknown] Edward, l'amichevole bot GnuPG <edward-it@fsf.org>
uid           [ unknown] Edward, le gentil robot de GnuPG <edward-fr@fsf.org>
uid           [ unknown] Edward, el simpático robot GnuPG <edward-es@fsf.org>
uid           [ unknown] Edward, o amigo robô de GnuPG <edward-pt-br@fsf.org>
uid           [ unknown] Edward, robotul GnuPG cel prietenos <edward-ro@fsf.org>
uid           [ unknown] Edward, arkadaş canlısı GnuPG botu <edward-tr@fsf.org>
uid           [ unknown] Edward, der freundliche GnuPG Roboter <edward-de@fsf.org>
uid           [ unknown] Эдвард, дружелюбный GnuPG бот <edward-ru@fsf.org>
uid           [ unknown] Edward, το φιλικό ρομπότ του GnuPG <edward-el@fsf.org>
uid           [ unknown] 爱德华，一个友好的 GnuPG 机器人 <edward-zh@fsf.org>
sub   rsa2048 2014-06-29 [E]

20:36 alexc@Alexs-MBP /Users/alexc
% gpg --export --armor C09A61E8 > edward-en_fsf.org.public.asc
20:36 alexc@Alexs-MBP /Users/alexc
% gpg --export --armor edward-en@fsf.org > edward-en_fsf.org.public.asc
20:37 alexc@Alexs-MBP /Users/alexc
% gpg --export --armor F357AA1A5B1FA42CFD9FE52A9FF2194CC09A61E8 > edward-en_fsf.org.public.asc
20:38 alexc@Alexs-MBP /Users/alexc
% cat edward-en_fsf.org.public.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBFOwfzoBCADpwK6sGC3EUzgD7IW1X5ZDR1nC5/rcXacAJLarPpvQBEz4pwjT
joAzATM7F9RwIzJ3hJTZHiYaQY4cfiGlKSnrd8GPC8A4QkxXIaQ0hLpcsBSbtZJp
o2iOzL2fRHmW2ZlnSHXPKbDwx5p0NcdQfjL9i2Yo31aLIO/Chhn5uyvIznOjaCSC
/O6x2C4m81Lu+B4UTDpl8y6ChtphUxyFGd7RXDXmkYQrxVqJbXKuSVmNMhM09myG
```

Delete Keys

```
gpg --delete-secret-keys [keyID]
gpg --delete-keys [keyID]
```

`[keyID]` – **Last 8** or **16** digits of Fingerprint
may use Email Address
or Full Fingerprint



Import Keys

```
gpg --import my_pgpkey.private.asc
gpg --import my_pgpkey.public.asc
```

```
20:59 alexc@Alexs-MBP /Users/alexc
% ls -lah ed*
-rw-r--r--@ 1 alexc  staff   8.0K Aug 18 20:38 edward-en_fsf.org.public.asc
20:59 alexc@Alexs-MBP /Users/alexc
% gpg --import edward-en_fsf.org.public.asc
gpg: key 9FF2194CC09A61E8: public key "Edward, the GPG Bot <edward-en@fsf.org>" imported
gpg: Total number processed: 1
gpg:               imported: 1
20:59 alexc@Alexs-MBP /Users/alexc
%
```

Keyservers

```
gpg --keyserver hkps://keys.openpgp.org --search edward-en@fsf.org
```

We can **skip `--keyserver`** `hkps://keys.openpgp.org`
if we are using **default** `keys.openpgp.org` server,
not custom, like `pgp.mit.edu`

```
gpg --search achesnakov@mailfence.com
```

```
Email address may have multiple Keys,
need to specify specific Key
gpg --recv-keys 1FA31C2070C6150F
```

```
gpg --send-keys 1FA31C2070C6150F
```

```
21:13 alexc@Alexs-MBP /Users/alexc
% gpg --search achesnakov@mailfence.com
gpg: data source: https://keys.openpgp.org:443
(1)     Alex Chesnakov <achesnakov@mailfence.com>
          256 bit EDDSA key 1FA31C2070C6150F, created: 2024-08-16
Keys 1-1 of 1 for "achesnakov@mailfence.com".  Enter number(s), N)ext, or Q)uit > n
21:14 alexc@Alexs-MBP /Users/alexc
% gpg --recv-keys achesnakov@mailfence.com
gpg: "achesnakov@mailfence.com" not a key ID: skipping
21:14 alexc@Alexs-MBP /Users/alexc
% gpg --recv-keys 1FA31C2070C6150F
gpg: key 1FA31C2070C6150F: "Alex Chesnakov <achesnakov@mailfence.com>" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
21:14 alexc@Alexs-MBP /Users/alexc
% gpg --send-keys 1FA31C2070C6150F
gpg: sending key 1FA31C2070C6150F to hkps://keys.openpgp.org
21:15 alexc@Alexs-MBP /Users/alexc
%
```

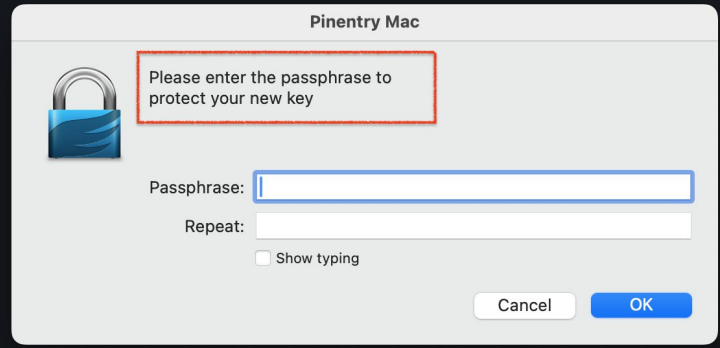
Generate Keys

`gpg --full-generate-key`

Revocation Certificate:

- Save it in a safe place
- Upload to keyserve
  if you lost your Private Key
- Decryption will still work
  after you upload Rev.Cert.
- Somebody need to search keyserver to notice that the key was revoked

```
21:46 alexc@Alexs-MBP /Users/alexc
% gpg --full-generate-key
gpg (GnuPG/MacGPG2) 2.2.41; Copyright (C) 2022 g10 Code GmbH
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072)
Requested keysize is 3072 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 1y
Key expires at Tue Aug 19 21:46:58 2025 PDT
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: Alex Ches
Email address: alexches@mailfence.com
Comment:
You selected this USER-ID:
    "Alex Ches <alexches@mailfence.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: directory '/Users/alexc/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/Users/alexc/.gnupg/openpgp-revocs.d/6FADBEB92358625468E0D9CE6125468E82CF4320.rev'
public and secret key created and signed.

pub   rsa3072 2024-08-20 [SC] [expires: 2025-08-20]
      6FADBEB92358625468E0D9CE6125468E82CF4320
uid                      Alex Ches <alexches@mailfence.com>
sub   rsa3072 2024-08-20 [E] [expires: 2025-08-20]

21:48 alexc@Alexs-MBP /Users/alexc
%
```



Sign the message / file

`gpg --sign file`
file.**gpg** (input file + signature, binary)

`gpg --clear-sign file`
file.**asc** (original message + ascii signature)

`gpg --detach-sign file`
file.**sig** (separate signature, binary)

`gpg --armor --local-user 70C6150F --detach-sig file`
file.**asc** (specify private key, separate signature, ascii)

```
% cat my_message.txt
This is my message.
22:31 alexc@Alexs-MBP /Users/alexc
% gpg --sign my_message.txt
22:31 alexc@Alexs-MBP /Users/alexc
% cat my_message.txt.gpg
??????!?a????u???H?=??M-.NLO?+?(I;???Y?D??
???R1Y1E?*5?    ?+??????????)i~???sq??ﺩ???b?g6?
???Wr?M<?EcCX?f3#â<ز5
    °?p??U??l?%6~??~??g/3%
```

```
% gpg --clear-sign my_message.txt
22:33 alexc@Alexs-MBP /Users/alexc
% cat my_message.txt.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256

This is my message.
-----BEGIN PGP SIGNATURE-----

iHUEARYIAB0WIQQJeiY6kL95Fs/EEBSwreqgg+e+iAUCZsQqugAKCRCwreqgg+e+
iFdSAQCSMcDIIXEUYCYAmK16/mS/HzKNiXP+pSZKPUKSY1SnyQD9G/W88/STb00d
8qU94jTwxsMtkr0xHYYyvpqiNw6AHwU=
=NyDr
-----END PGP SIGNATURE-----
```

```
% gpg --armor --local-user 70C6150F --detach-sig my_message.txt
23:10 alexc@Alexs-MBP /Users/alexc
% cat my_message.txt.asc
-----BEGIN PGP SIGNATURE-----

iHUEABYIAB0WIQRKprR2hhwAfd6V/CsfoxwgcMYVDwUCZsQzTAAKCRAfoxwgcMYV
D8iaAQD7x8ufLH93ZFErcftj8eq1cHQSzmnFQ7nxw50csExFaQD9F2gmXDpLQLHF
aVhQQw02k6xVzy6xcgrxDv1IRkNqCgE=
=XdoA
-----END PGP SIGNATURE-----
```

```
% gpg --detach-sign my_message.txt
22:48 alexc@Alexs-MBP /Users/alexc
% cat my_message.txt.sig
?!      z&:??y????G羈 f?.7
        ??G羈 =??*??±??  S4??r9??Z#?h??8?}WO???`?????g?VwKP#???IM{y??H?N?%
```

Verify Signature

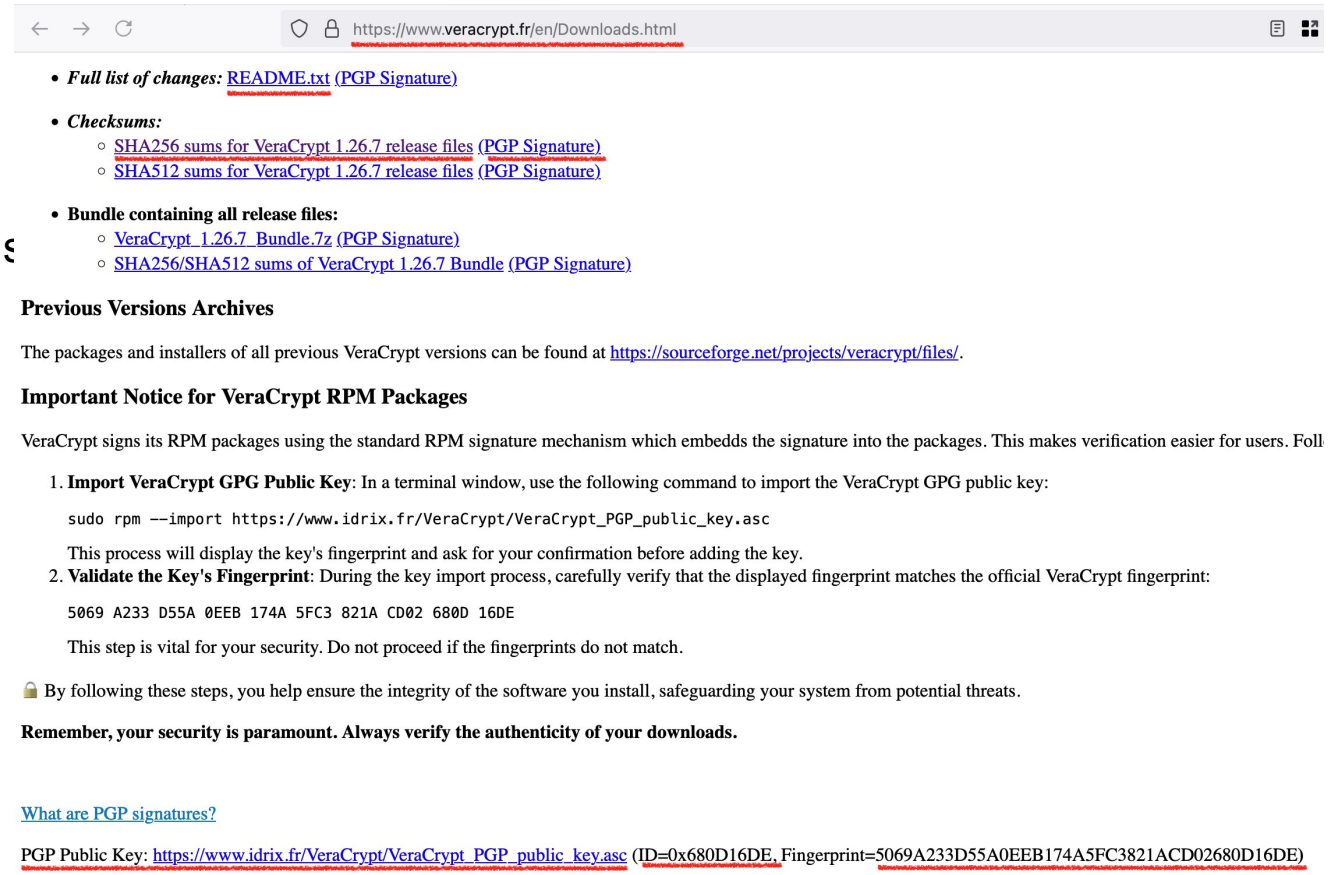
```
gpg --verify my_message.txt.sig
```

```
% gpg --verify my_message.txt.sig
gpg: assuming signed data in 'my_message.txt'
gpg: Signature made Mon Aug 19 22:48:39 2024 PDT
gpg:                using EDDSA key 097A263A90BF7916CFC41014B0ADEAA083E7BE88
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   2  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 2u
gpg: next trustdb check due at 2025-08-20
gpg: Good signature from "Alex Ches <alexches@mailfence.com>" [ultimate]
```

Verifying Files from the Internet

Go to https://www.veracrypt.fr/en/Downloads.html

Download
- README.TXT
- veracrypt-1.26.7-sha256sum.txt
- veracrypt-1.26.7-sha256sum.txt.s



Verifying Files from the Internet (Continue)

- Check sha256sum hash
- Compare with hash on web site
- To be sure that hash on web site wasn't spoofed, check signature
- You can `--import` Key and **trust** this Key to avoid WARNING

```
21:27 alexc@Alexs-MBP /Users/alexc
% head -1 veracrypt-1.26.7-sha256sum.txt
5e8ce0dd03aa1a9728157b224a36b622007f27d4ebb04395fb73f23a19ac6114  README.TXT
21:27 alexc@Alexs-MBP /Users/alexc
% shasum -a 256 README.TXT
5e8ce0dd03aa1a9728157b224a36b622007f27d4ebb04395fb73f23a19ac6114  README.TXT
21:28 alexc@Alexs-MBP /Users/alexc
% gpg --verify veracrypt-1.26.7-sha256sum.txt.sig
gpg: assuming signed data in 'veracrypt-1.26.7-sha256sum.txt'
gpg: Signature made Mon May 20 13:13:41 2024 PDT
gpg:                using RSA key 5069A233D55A0EEB174A5FC3821ACD02680D16DE
gpg: key 821ACD02680D16DE: public key "VeraCrypt Team (2018 - Supersedes Key ID=0x54DDD393) <veracrypt@idrix.fr>" imported
gpg: Total number processed: 1
gpg:               imported: 1
gpg: Good signature from "VeraCrypt Team (2018 - Supersedes Key ID=0x54DDD393) <veracrypt@idrix.fr>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 5069 A233 D55A 0EEB 174A  5FC3 821A CD02 680D 16DE
```

Encrypt / Decrypt Files

```
gpg -e -s -a -u achesnakov@mailfence.com -r achesnakov@mailfence.com message.txt
```

```
gpg -d message.txt.asc
```

```
-e, --encrypt
-s, --sign
-a, --armor
-d, --decrypt

-u, --local-user USER-ID
  use USER-ID to sign, encr, decr
-r, --recipient USER-ID
 encrypt for USER-ID
```

In this case I encrypted for myself
You can encrypt for different recipient

```
% gpg -e -s -a -u achesnakov@mailfence.com -r achesnakov@mailfence.com message.txt
22:30 alexc@Alexs-MBP /Users/alexc
% cat message.txt
This is my message.
22:30 alexc@Alexs-MBP /Users/alexc
% gpg --decrypt message.txt.asc
gpg: encrypted with 256-bit ECDH key, ID 1492A775A41C110D, created 2024-08-16
      "Alex Chesnakov <achesnakov@mailfence.com>"
This is my message.
gpg: Signature made Tue Aug 20 22:30:16 2024 PDT
gpg:                using EDDSA key 4AA6B476861C007DDE95FC2B1FA31C2070C6150F
gpg:                issuer "achesnakov@mailfence.com"
gpg: Good signature from "Alex Chesnakov <achesnakov@mailfence.com>" [ultimate]
22:30 alexc@Alexs-MBP /Users/alexc
% cat message.txt.asc
-----BEGIN PGP MESSAGE-----

hF4DFJKndaQcEQ0SAQdAaRSq0EfYSeh0+jqcThOC31ntDgLLilFfZtZHfPylGhEw
l4ruMI/KwC9orEv7aHrWmDnewbLayMFQu4juicNQRSbnNKTHpEhzxUvUJoCc7qgJ
0sAoAUOxsWSWESjl5bi/PuO/iWx/aWZmTQvtFWjV1OjAOq7yP3hGECk1STHmBpFK
wldmBAlMLVVFB1L5kWvQkwsWNZTvpyUbvLkLchERYZfSI7f8R1m+PX8ChSxErt5z
X0bKHnUMja+O9vtctXwyS83kaj2ML/eYV3hZ7FazFHUozYiEZzfXOKIoH42o8w7h
MA0LV39ce72AzfwIzsTaGLqateIy4BPs1xAVeM11j6qppsokfc+Q5ioI9ERaxrzc
tCLS4Ey8Jj03I3hgBGfMKAQc6okyAfAvsWCT4c4356g5LyOvvNynTDqcpg==
=qD4q
-----END PGP MESSAGE-----
```

Not Covered

- Web of Trust
- Traditional OpenPGP keyservers
- Key Signing Parties
- Photo IDs in OpenPGP keys
- Sub-Keys
- Edit Keys (`gpg --edit-key`)
- Physical Keys: YubiKeys, Smartcards

Q&A

- Q: SSL/TLS and SSH use public / private keys as well. What the difference?
  A: The same technology, different info in the certificates

- Q: Can we use ProtonMail with the OpenPGP?
  A: You have to use their Bridge software to use with the Thunderbird.
  The better way to use Proton's built-in PGP that offer all the functionality.

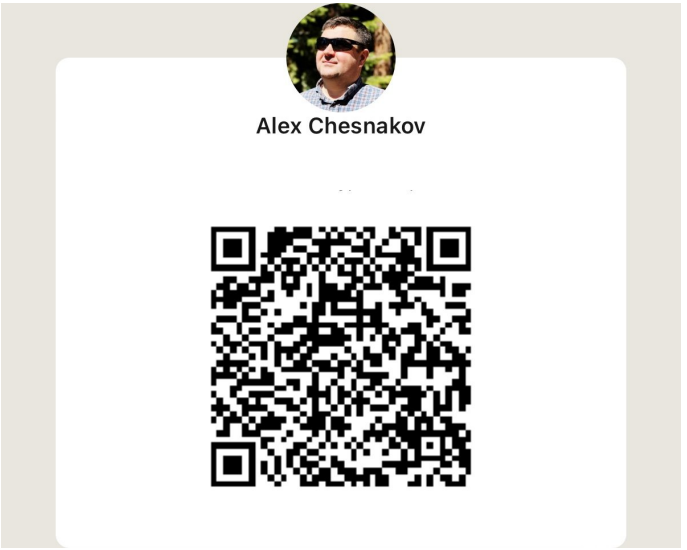
- Q: Can we use OpenPGP on a phone?
  A: I found a few mobile apps, but haven't evaluated them.
  I use ProtonMail on Desktop and Mobile with OpenPGP encryption.

Resources:

- https://emailselfdefense.fsf.org/en/
- https://www.gnupg.org/gph/en/manual.html
- https://www.oreilly.com/library/view/pgp-gpg/9781457185212/

Free O'REILLY access with **San Jose Public Library**
https://www.sjpl.org/resource/oreilly/
Free LinkedIn Learning access with **San Jose Public Library**
https://www.sjpl.org/resource/linkedin-learning/

Thanks



https://github.com/olexandrch

https://www.linkedin.com/in/alex-chesnakov/