Open in app Get started

Michael Galarnyk Follow
Aug 18, 2018 · 5 min read · Listen

Save

Public-key (asymmetric) Cryptography using GPG

GNU Privacy Guard (GPG, also called GnuPG) is a free encryption software you can use to encrypt and decrypt files. While the documentation for GnuPG is excellent, this is a quick cheatsheet on how to get started with GPG.

Install GPG

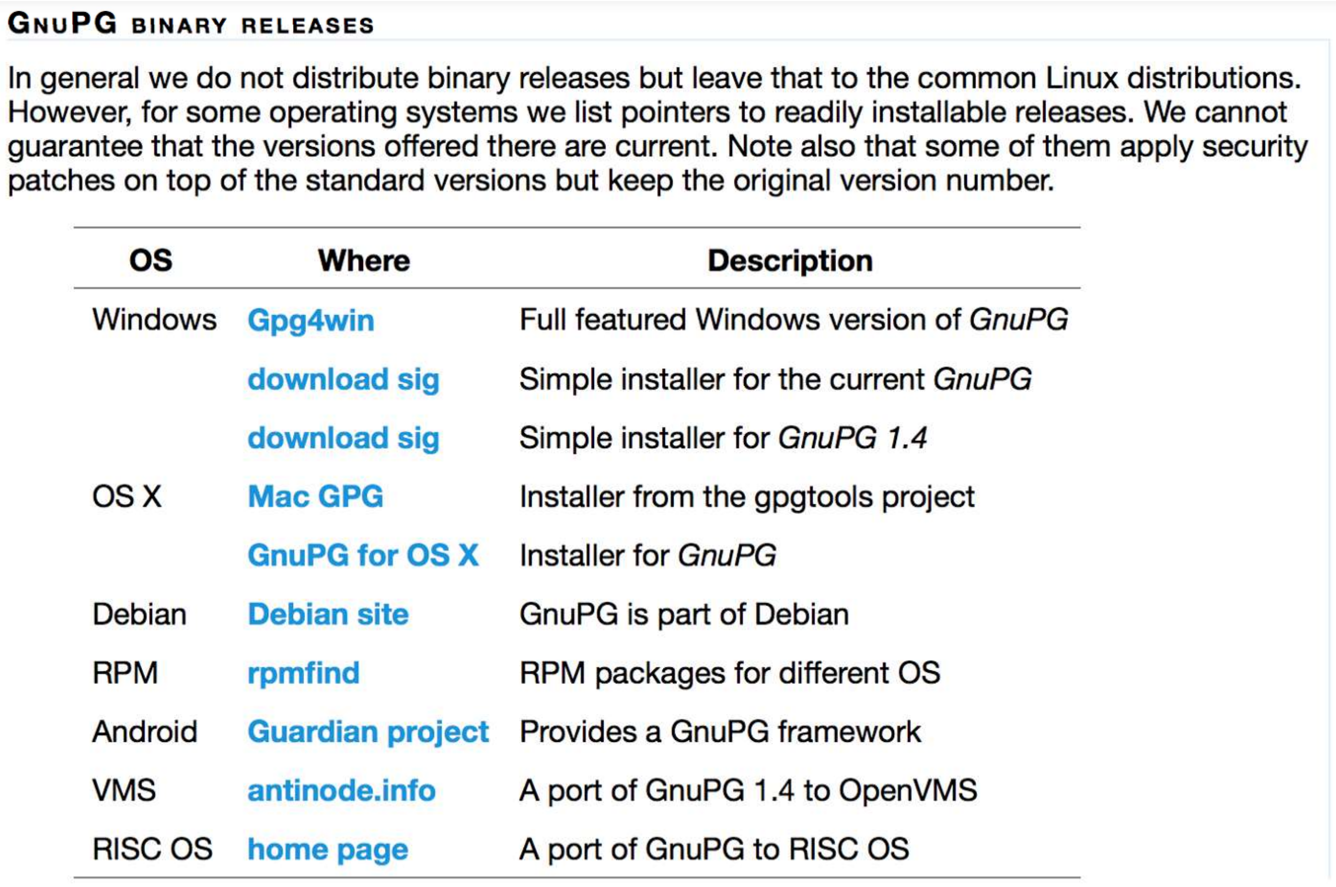
Mac

You need homebrew to be able to install gpg on Mac . If you don't have homebrew installed, you can learn how to do that here. After that, it is a one line command.

```
brew install gnupg
```

Windows

There are many ways to install gpg on windows. Perhaps the easiest way to is to go to GnuPG site and use the simple installer for the current GnuPG.



GnuPG binary releases

In general we do not distribute binary releases but leave that to the common Linux distributions. However, for some operating systems we list pointers to readily installable releases. We cannot guarantee that the versions offered there are current. Note also that some of them apply security patches on top of the standard versions but keep the original version number.

OS	Where	Description
Windows	Gpg4win	Full featured Windows version of *GnuPG*
	download sig	Simple installer for the current *GnuPG*
	download sig	Simple installer for *GnuPG 1.4*
OS X	Mac GPG	Installer from the gpgtools project
	GnuPG for OS X	Installer for *GnuPG*
Debian	Debian site	GnuPG is part of Debian
RPM	rpmfind	RPM packages for different OS
Android	Guardian project	Provides a GnuPG framework
VMS	antinode.info	A port of GnuPG 1.4 to OpenVMS
RISC OS	home page	A port of GnuPG to RISC OS

Red Hat / CentOS

```
yum install gnupg
```

Ubuntu / Debian

If you are using these Linux distributions, you might want to change the commands in this tutorial to `gpg2` after using the command below. You can find more infomation on this here.

```
sudo apt-get install gnupg2
```

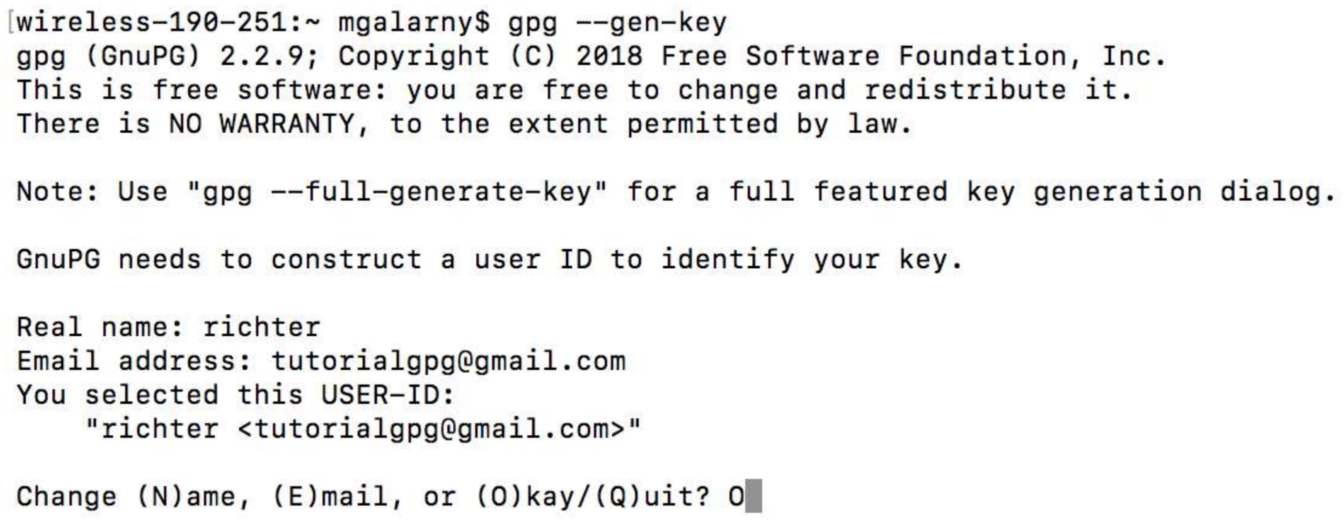
Entire Process

GPG uses a method of encryption known as public key (asymmetric) cryptography, which provides a number of advantages and benefits. In a public key (asymmetric) section just goes through the GPG commands to do this. If you don't understand asymmetric encryption, there is a wonderful youtube video on it here.

Generating Key Pair (Private and Public Keys)

1) Create your keys. This will generate a key pair. One is a private key which you need to keep safe and a public key which you can share with other people.

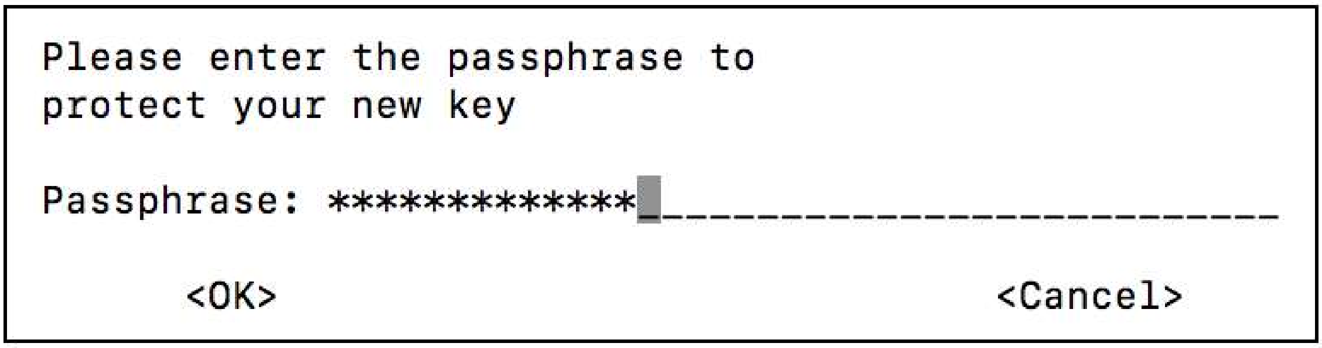
```
gpg --gen-key
```



```
[wireless-190-251:~ mgalarny$ gpg --gen-key
gpg (GnuPG) 2.2.9; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

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

Real name: richter
Email address: tutorialgpg@gmail.com
You selected this USER-ID:
    "richter <tutorialgpg@gmail.com>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit? O
```

Enter name, email address, and O

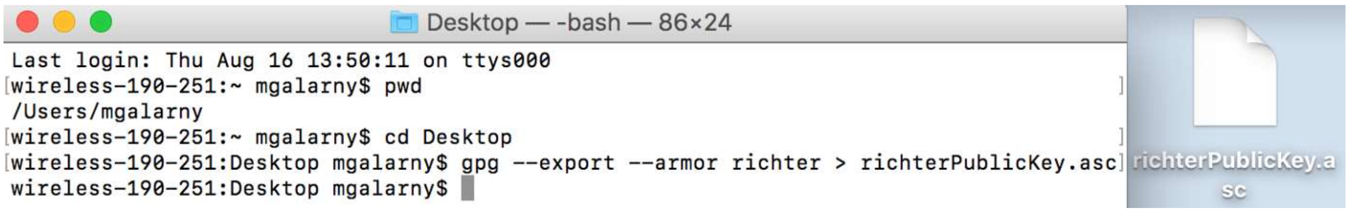
2) You will have to enter a password. Keep it somewhere safe.



```
Please enter the passphrase to
protect your new key

Passphrase: *************_____________________________

      <OK>                                   <Cancel>
```

Enter and re-enter your password

3) Export your public key. In this case, `richter` is the name of my public key. It will be



```
Last login: Thu Aug 16 13:50:11 on ttys000
wireless-190-251:~ mgalarny$ pwd
/Users/mgalarny
wireless-190-251:~ mgalarny$ cd Desktop
wireless-190-251:Desktop mgalarny$ gpg --export --armor richter > richterPublicKey.asc
wireless-190-251:Desktop mgalarny$
```

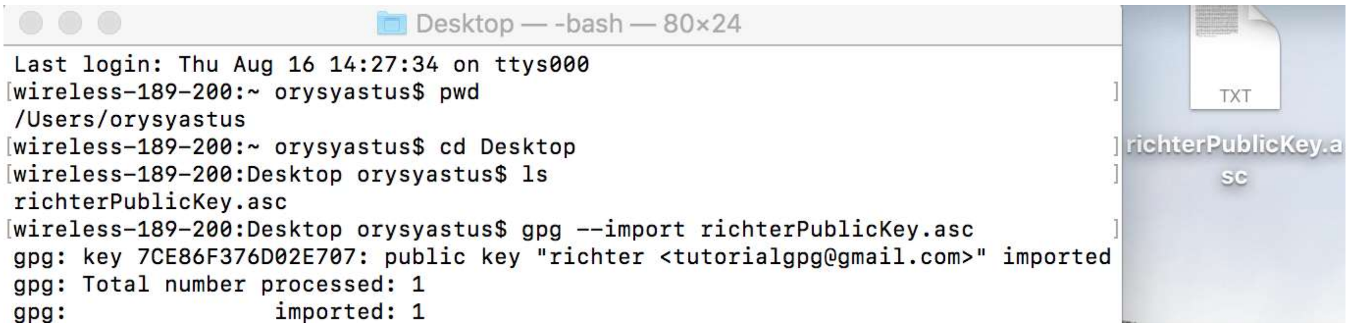
Export your public key

4) Send the public key you exported to another person.

Public Key Holder

1) Import another persons public key. You need to substitute `richterPublicKey` for the public key you wish to import.

```
gpg --import richterPublicKey.asc
```



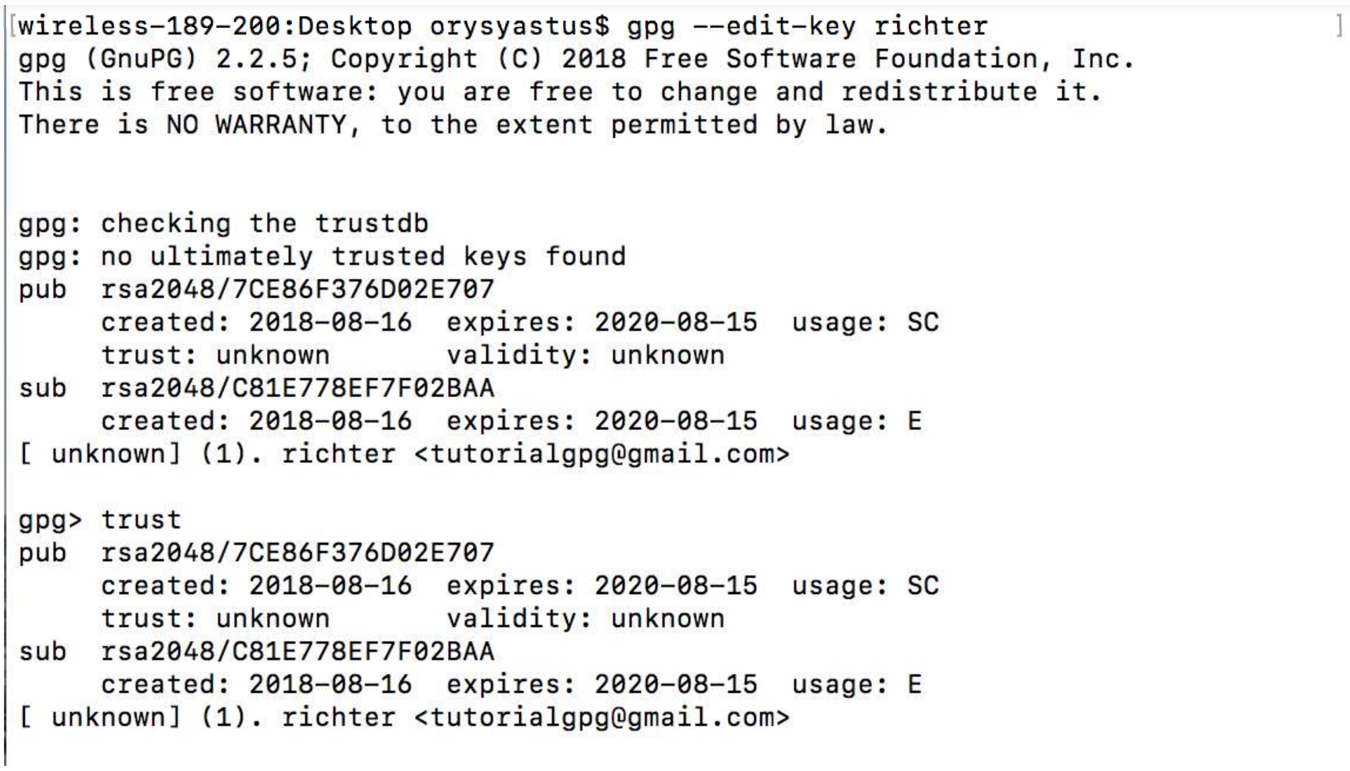
```
Last login: Thu Aug 16 14:27:34 on ttys000
wireless-189-200:~ orysyastus$ pwd
/Users/orysyastus
wireless-189-200:~ orysyastus$ cd Desktop
wireless-189-200:Desktop orysyastus$ ls
richterPublicKey.asc
wireless-189-200:Desktop orysyastus$ gpg --import richterPublicKey.asc
gpg: key 7CE86F376D02E707: public key "richter <tutorialgpg@gmail.com>" imported
gpg: Total number processed: 1
gpg:               imported: 1
```

import other person's key

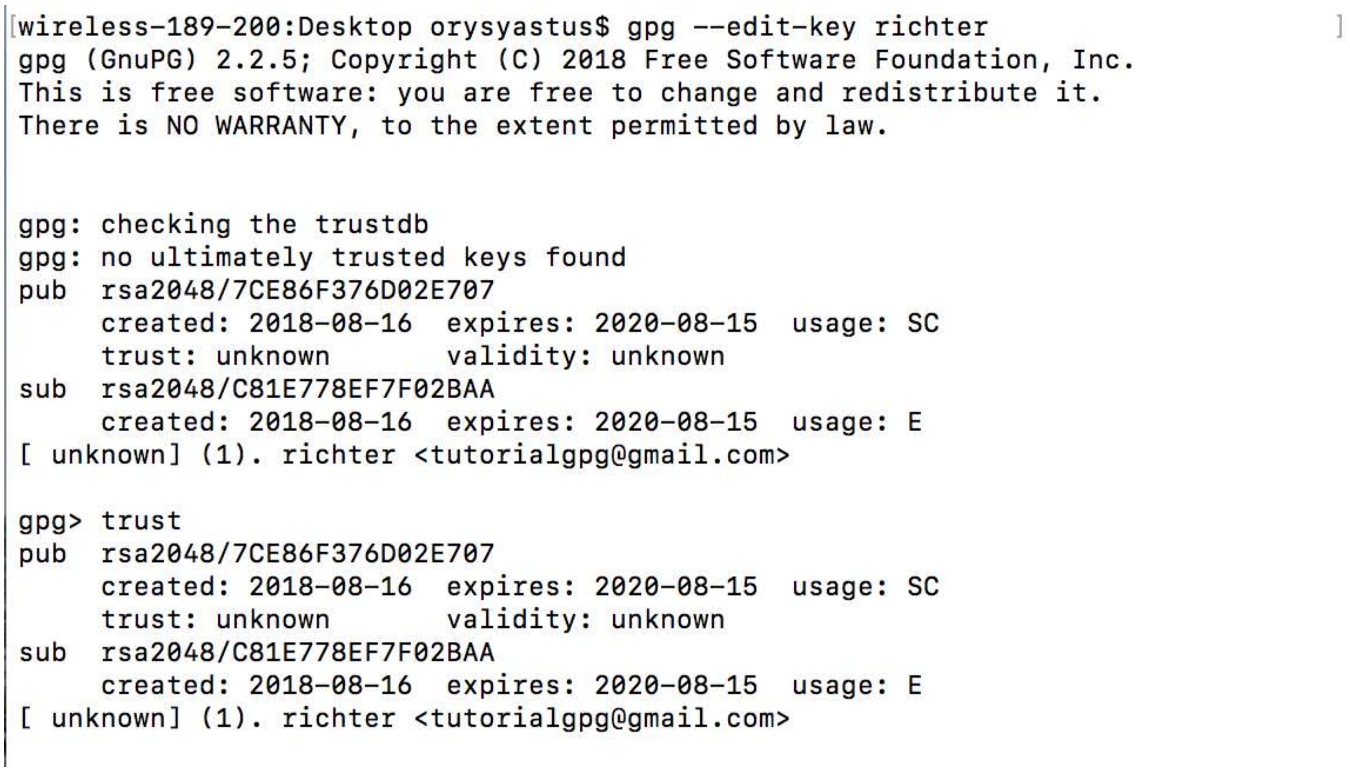
2) Trust the public key. This will prevent GPG from warning you every time you encrypt something with that public key. You need to substitute `richter` with the name of your public key.

```
gpg --edit-key richter
```

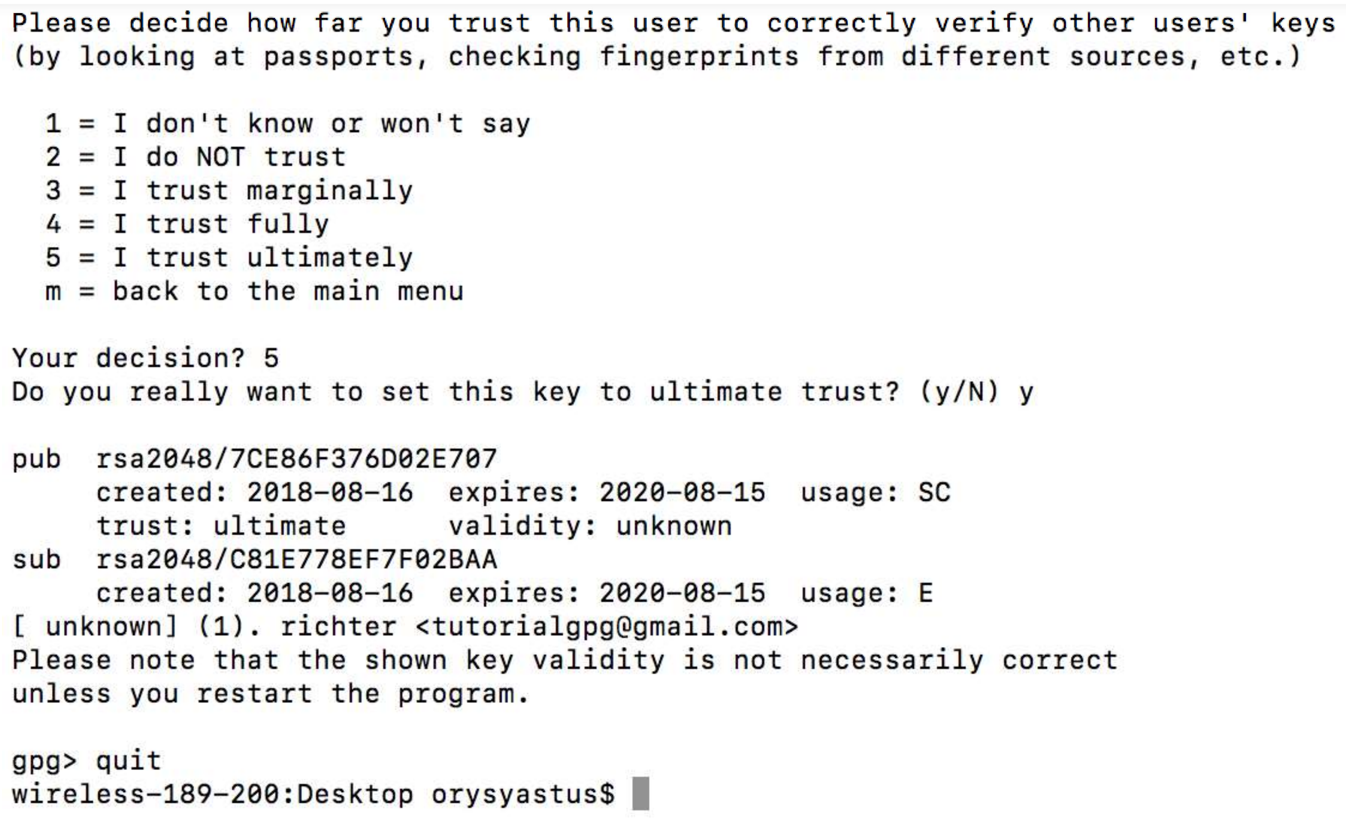
Open in app
Get started


```
wireless-189-200:Desktop orysyastus$ gpg --edit-key richter
gpg (GnuPG) 2.2.5; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.


gpg: checking the trustdb
gpg: no ultimately trusted keys found
pub  rsa2048/7CE86F376D02E707
     created: 2018-08-16  expires: 2020-08-15  usage: SC
     trust: unknown       validity: unknown
sub  rsa2048/C81E778EF7F02BAA
     created: 2018-08-16  expires: 2020-08-15  usage: E
[ unknown] (1). richter <tutorialgpg@gmail.com>

gpg> trust
pub  rsa2048/7CE86F376D02E707
     created: 2018-08-16  expires: 2020-08-15  usage: SC
     trust: unknown       validity: unknown
sub  rsa2048/C81E778EF7F02BAA
     created: 2018-08-16  expires: 2020-08-15  usage: E
[ unknown] (1). richter <tutorialgpg@gmail.com>
```

Enter `trust`


```
wireless-189-200:Desktop orysyastus$ gpg --edit-key richter
gpg (GnuPG) 2.2.5; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.


gpg: checking the trustdb
gpg: no ultimately trusted keys found
pub  rsa2048/7CE86F376D02E707
     created: 2018-08-16  expires: 2020-08-15  usage: SC
     trust: unknown       validity: unknown
sub  rsa2048/C81E778EF7F02BAA
     created: 2018-08-16  expires: 2020-08-15  usage: E
[ unknown] (1). richter <tutorialgpg@gmail.com>

gpg> trust
pub  rsa2048/7CE86F376D02E707
     created: 2018-08-16  expires: 2020-08-15  usage: SC
     trust: unknown       validity: unknown
sub  rsa2048/C81E778EF7F02BAA
     created: 2018-08-16  expires: 2020-08-15  usage: E
[ unknown] (1). richter <tutorialgpg@gmail.com>
```

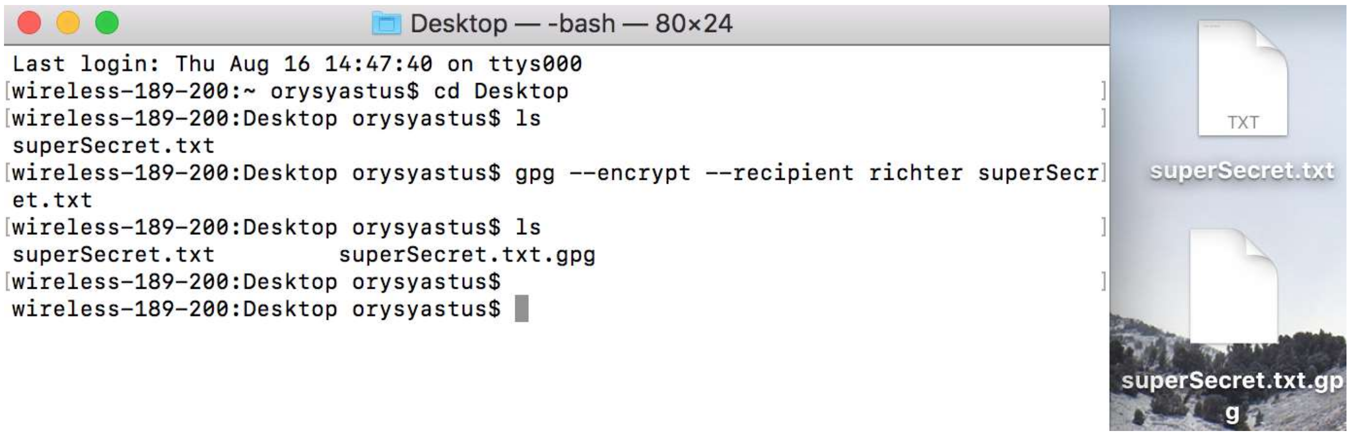
Enter `5` , `y` , and then `quit`



```
Please decide how far you trust this user to correctly verify other users' keys
(by looking at passports, checking fingerprints from different sources, etc.)

  1 = I don't know or won't say
  2 = I do NOT trust
  3 = I trust marginally
  4 = I trust fully
  5 = I trust ultimately
  m = back to the main menu

Your decision? 5
Do you really want to set this key to ultimate trust? (y/N) y

pub  rsa2048/7CE86F376D02E707
     created: 2018-08-16  expires: 2020-08-15  usage: SC
     trust: ultimate      validity: unknown
sub  rsa2048/C81E778EF7F02BAA
     created: 2018-08-16  expires: 2020-08-15  usage: E
[ unknown] (1). richter <tutorialgpg@gmail.com>
Please note that the shown key validity is not necessarily correct
unless you restart the program.

gpg> quit
wireless-189-200:Desktop orysyastus$ 
```

3) This step shows how to encrypt a file (in this case, I encrypted a file superSecret.txt).

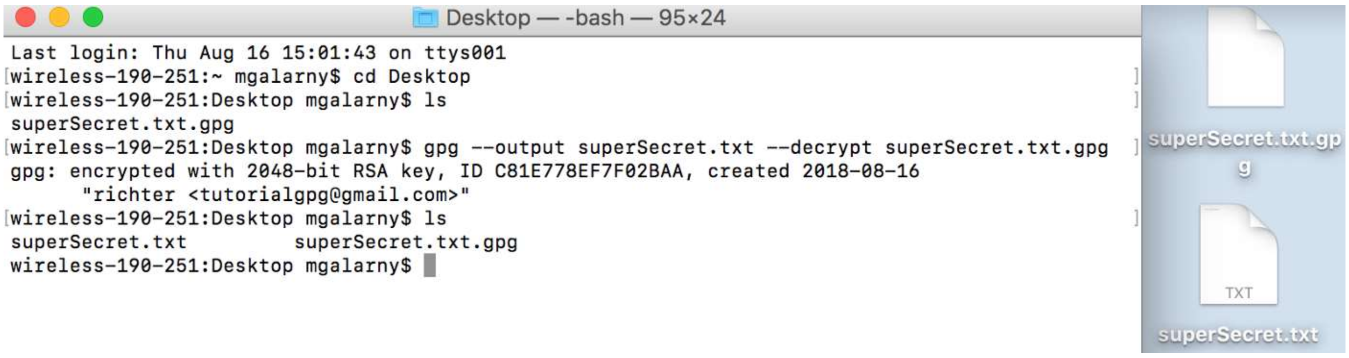
```
gpg --encrypt --recipient richter superSecret.txt
```



```
Last login: Thu Aug 16 14:47:40 on ttys000
[wireless-189-200:~ orysyastus$ cd Desktop                                     ]
[wireless-189-200:Desktop orysyastus$ ls                                       ]
superSecret.txt
[wireless-189-200:Desktop orysyastus$ gpg --encrypt --recipient richter superSecr]
et.txt
[wireless-189-200:Desktop orysyastus$ ls                                       ]
superSecret.txt         superSecret.txt.gpg
[wireless-189-200:Desktop orysyastus$                                          ]
wireless-189-200:Desktop orysyastus$ 
```

4) Transfer the encrypted file to the private key holder.

Private Key Holder



```
Last login: Thu Aug 16 15:01:43 on ttys001
wireless-190-251:~ mgalarny$ cd Desktop
wireless-190-251:Desktop mgalarny$ ls
superSecret.txt.gpg
wireless-190-251:Desktop mgalarny$ gpg --output superSecret.txt --decrypt superSecret.txt.gpg
gpg: encrypted with 2048-bit RSA key, ID C81E778EF7F02BAA, created 2018-08-16
      "richter <tutorialgpg@gmail.com>"
wireless-190-251:Desktop mgalarny$ ls
superSecret.txt         superSecret.txt.gpg
wireless-190-251:Desktop mgalarny$
```

Keep in mind that you can also decrypt multiple files using the following command.

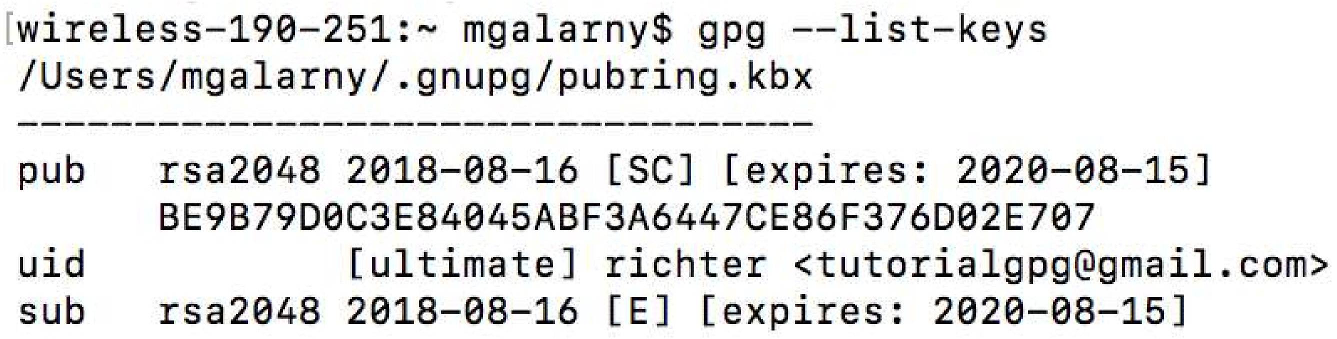
```
gpg --decrypt-files *.gpg
```

List Keys in your Keyring

Public Keys

You can view a list of public keys in your keyring as well as the name and email address associated with each key

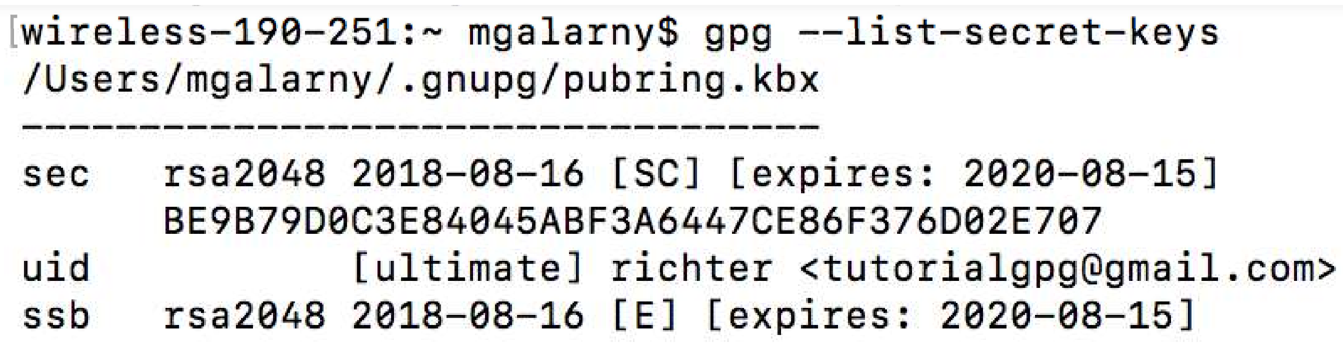
```
gpg --list-keys
```



```
wireless-190-251:~ mgalarny$ gpg --list-keys
/Users/mgalarny/.gnupg/pubring.kbx
----------------------------------
pub   rsa2048 2018-08-16 [SC] [expires: 2020-08-15]
      BE9B79D0C3E84045ABF3A6447CE86F376D02E707
uid           [ultimate] richter <tutorialgpg@gmail.com>
sub   rsa2048 2018-08-16 [E] [expires: 2020-08-15]
```

Private Keys

The following command will list the private keys in your keyring. This will show the

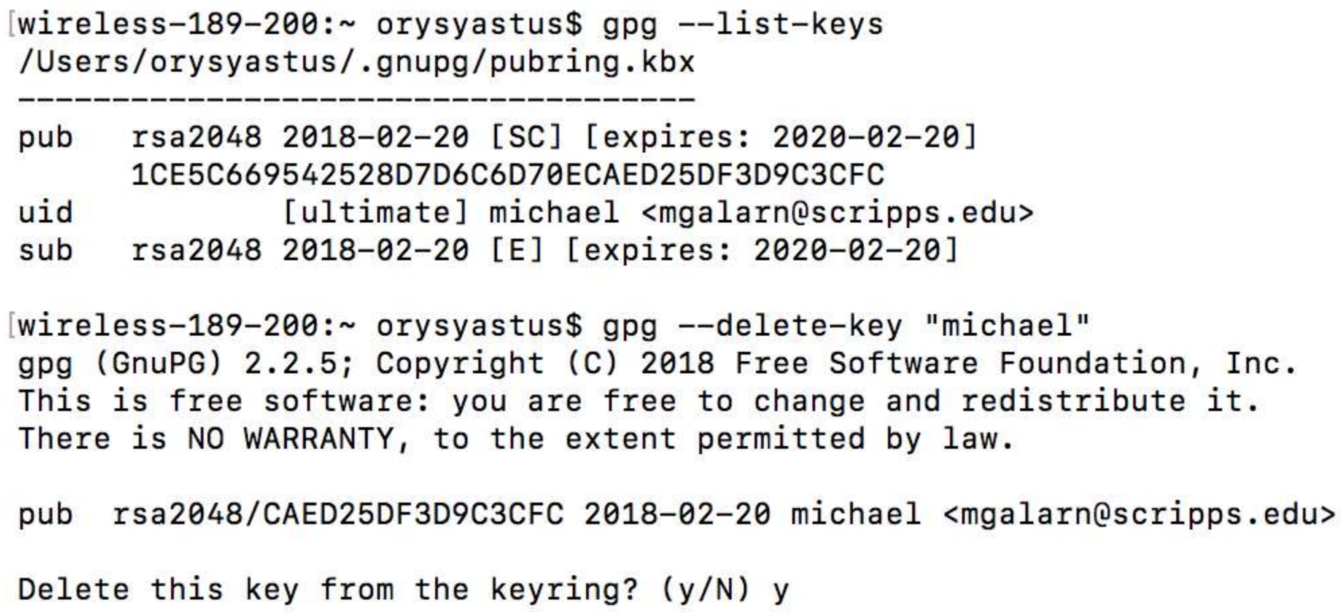

```
[wireless-190-251:~ mgalarny$ gpg --list-secret-keys
/Users/mgalarny/.gnupg/pubring.kbx
----------------------------------
sec   rsa2048 2018-08-16 [SC] [expires: 2020-08-15]
      BE9B79D0C3E84045ABF3A6447CE86F376D02E707
uid           [ultimate] richter <tutorialgpg@gmail.com>
ssb   rsa2048 2018-08-16 [E] [expires: 2020-08-15]
```

Delete Keys from Keyring

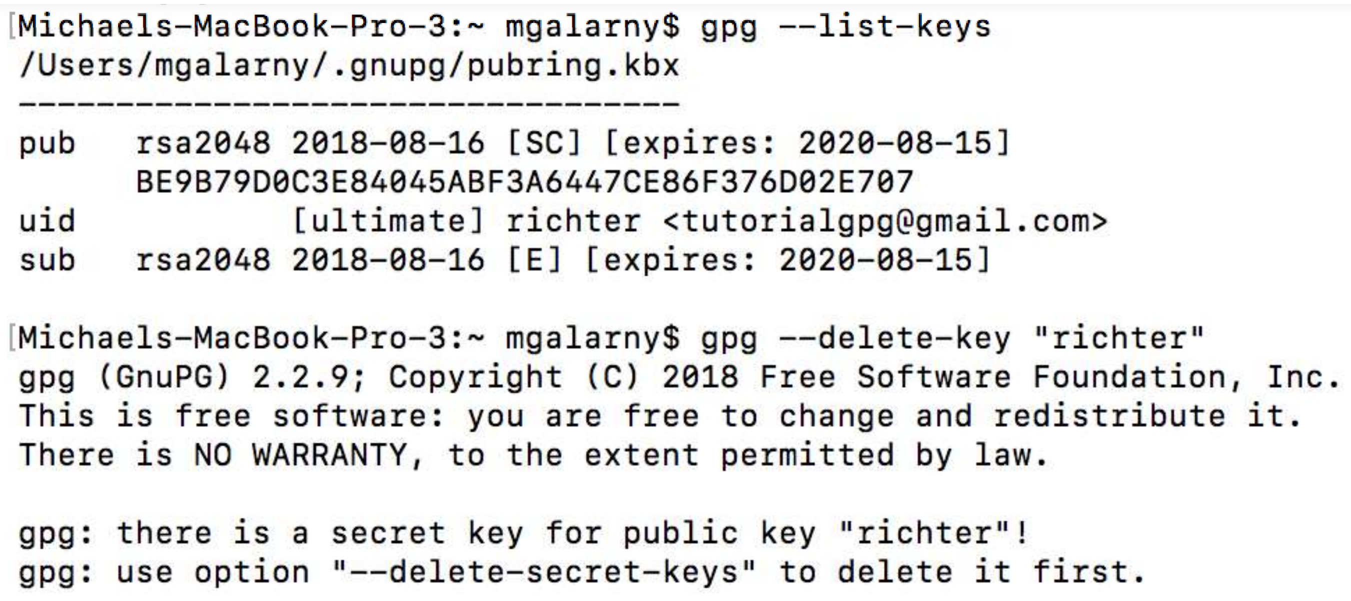
You can also delete keys from your keyring.

Remove Public Key

```
gpg --delete-key "User Name"
```


```
[wireless-189-200:~ orysyastus$ gpg --list-keys
/Users/orysyastus/.gnupg/pubring.kbx
------------------------------------
pub   rsa2048 2018-02-20 [SC] [expires: 2020-02-20]
      1CE5C669542528D7D6C6D70ECAED25DF3D9C3CFC
uid           [ultimate] michael <mgalarn@scripps.edu>
sub   rsa2048 2018-02-20 [E] [expires: 2020-02-20]

[wireless-189-200:~ orysyastus$ gpg --delete-key "michael"
gpg (GnuPG) 2.2.5; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

pub  rsa2048/CAED25DF3D9C3CFC 2018-02-20 michael <mgalarn@scripps.edu>

Delete this key from the keyring? (y/N) y
```

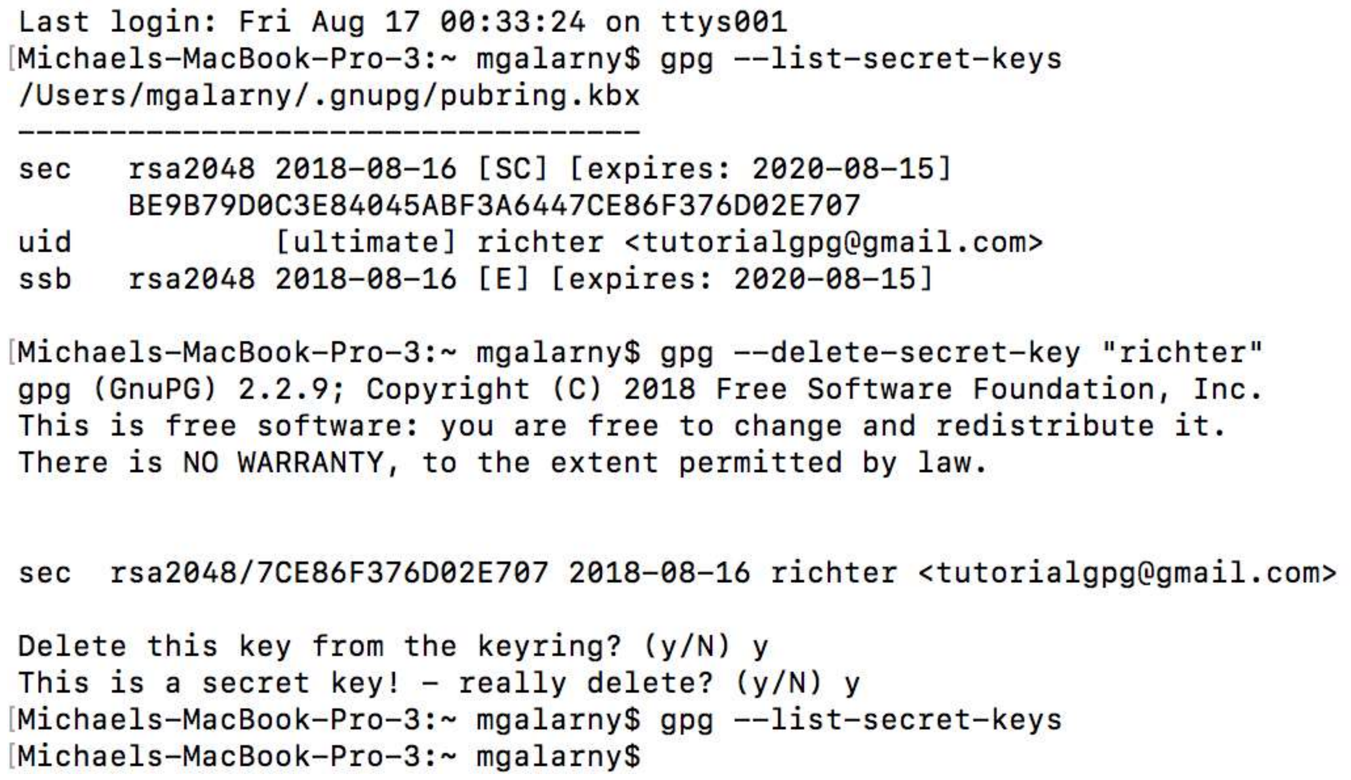
Note that if you try to delete a public key when you have its associated private key you will run into an error.



```
[Michaels-MacBook-Pro-3:~ mgalarny$ gpg --list-keys
/Users/mgalarny/.gnupg/pubring.kbx
----------------------------------
pub   rsa2048 2018-08-16 [SC] [expires: 2020-08-15]
      BE9B79D0C3E84045ABF3A6447CE86F376D02E707
uid           [ultimate] richter <tutorialgpg@gmail.com>
sub   rsa2048 2018-08-16 [E] [expires: 2020-08-15]

[Michaels-MacBook-Pro-3:~ mgalarny$ gpg --delete-key "richter"
gpg (GnuPG) 2.2.9; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: there is a secret key for public key "richter"!
gpg: use option "--delete-secret-keys" to delete it first.
```

Remove Private Key

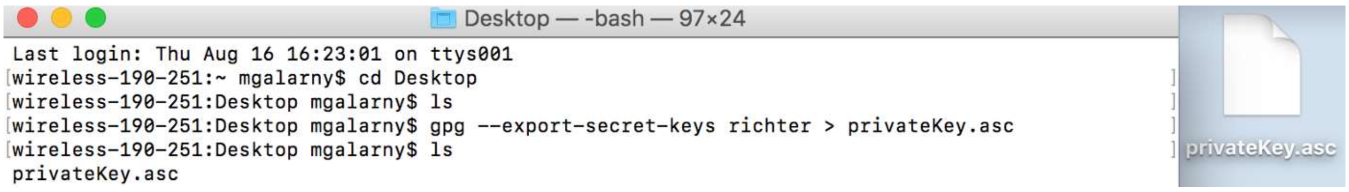
```
gpg --delete-secret-key "User Name"
```



```
Last login: Fri Aug 17 00:33:24 on ttys001
[Michaels-MacBook-Pro-3:~ mgalarny$ gpg --list-secret-keys
/Users/mgalarny/.gnupg/pubring.kbx
----------------------------------
sec   rsa2048 2018-08-16 [SC] [expires: 2020-08-15]
      BE9B79D0C3E84045ABF3A6447CE86F376D02E707
uid           [ultimate] richter <tutorialgpg@gmail.com>
ssb   rsa2048 2018-08-16 [E] [expires: 2020-08-15]

[Michaels-MacBook-Pro-3:~ mgalarny$ gpg --delete-secret-key "richter"
gpg (GnuPG) 2.2.9; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.


sec  rsa2048/7CE86F376D02E707 2018-08-16 richter <tutorialgpg@gmail.com>

Delete this key from the keyring? (y/N) y
This is a secret key! - really delete? (y/N) y
[Michaels-MacBook-Pro-3:~ mgalarny$ gpg --list-secret-keys
[Michaels-MacBook-Pro-3:~ mgalarny$
```

How to Export and Import a Secret Key

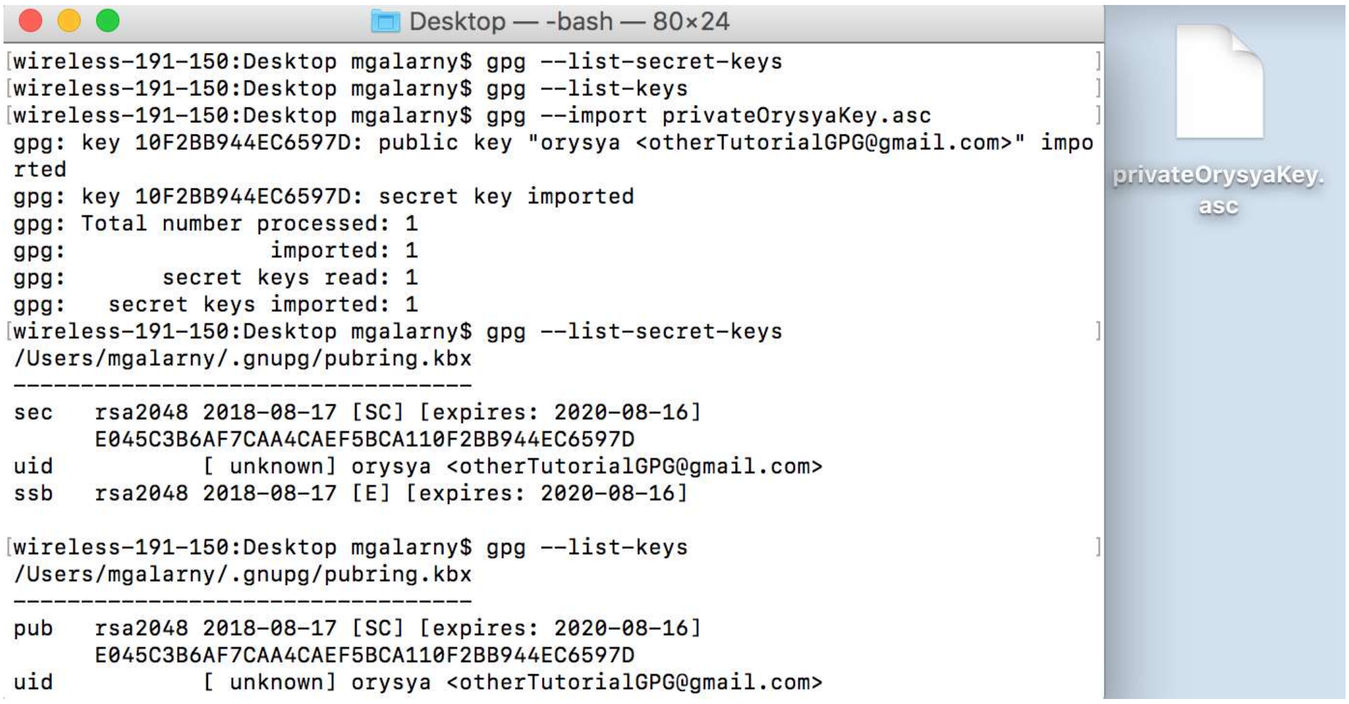
```
gpg --export-secret-keys richter > privateKey.asc
```



```
Last login: Thu Aug 16 16:23:01 on ttys001
wireless-190-251:~ mgalarny$ cd Desktop
wireless-190-251:Desktop mgalarny$ ls
wireless-190-251:Desktop mgalarny$ gpg --export-secret-keys richter > privateKey.asc
wireless-190-251:Desktop mgalarny$ ls
privateKey.asc
```

Import Secret Key

```
gpg --import privateKey.asc
```



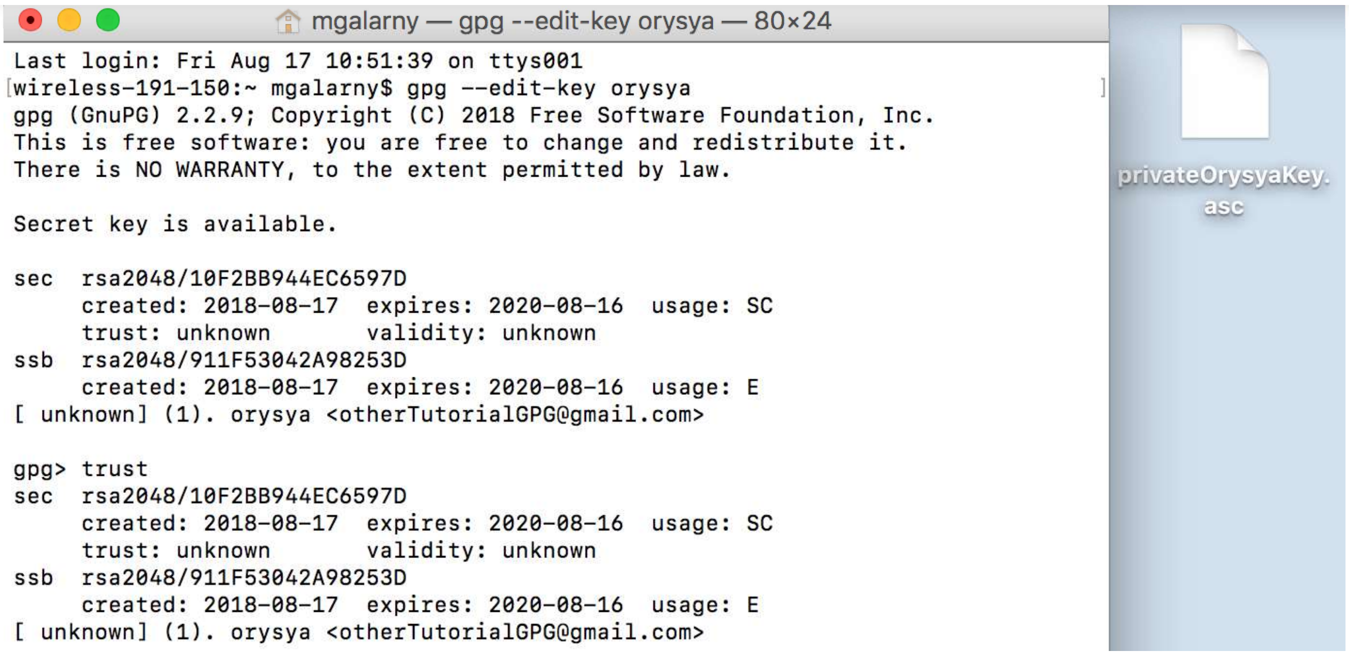
```
wireless-191-150:Desktop mgalarny$ gpg --list-secret-keys
wireless-191-150:Desktop mgalarny$ gpg --list-keys
wireless-191-150:Desktop mgalarny$ gpg --import privateOrysyaKey.asc
gpg: key 10F2BB944EC6597D: public key "orysya <otherTutorialGPG@gmail.com>" impo
rted
gpg: key 10F2BB944EC6597D: secret key imported
gpg: Total number processed: 1
gpg:               imported: 1
gpg:       secret keys read: 1
gpg:   secret keys imported: 1
wireless-191-150:Desktop mgalarny$ gpg --list-secret-keys
/Users/mgalarny/.gnupg/pubring.kbx
----------------------------------
sec   rsa2048 2018-08-17 [SC] [expires: 2020-08-16]
      E045C3B6AF7CAA4CAEF5BCA110F2BB944EC6597D
uid           [ unknown] orysya <otherTutorialGPG@gmail.com>
ssb   rsa2048 2018-08-17 [E] [expires: 2020-08-16]

wireless-191-150:Desktop mgalarny$ gpg --list-keys
/Users/mgalarny/.gnupg/pubring.kbx
----------------------------------
pub   rsa2048 2018-08-17 [SC] [expires: 2020-08-16]
      E045C3B6AF7CAA4CAEF5BCA110F2BB944EC6597D
uid           [ unknown] orysya <otherTutorialGPG@gmail.com>
```

Not done yet, you still need to ultimately trust a key.

You will need to make sure that you also ultimately trust a key.

```
gpg --edit-key orysya
```

enter `trust`



```
Last login: Fri Aug 17 10:51:39 on ttys001
wireless-191-150:~ mgalarny$ gpg --edit-key orysya
gpg (GnuPG) 2.2.9; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Secret key is available.

sec  rsa2048/10F2BB944EC6597D
     created: 2018-08-17  expires: 2020-08-16  usage: SC
     trust: unknown       validity: unknown
ssb  rsa2048/911F53042A98253D
     created: 2018-08-17  expires: 2020-08-16  usage: E
[ unknown] (1). orysya <otherTutorialGPG@gmail.com>

gpg> trust
sec  rsa2048/10F2BB944EC6597D
     created: 2018-08-17  expires: 2020-08-16  usage: SC
     trust: unknown       validity: unknown
ssb  rsa2048/911F53042A98253D
     created: 2018-08-17  expires: 2020-08-16  usage: E
[ unknown] (1). orysya <otherTutorialGPG@gmail.com>
```

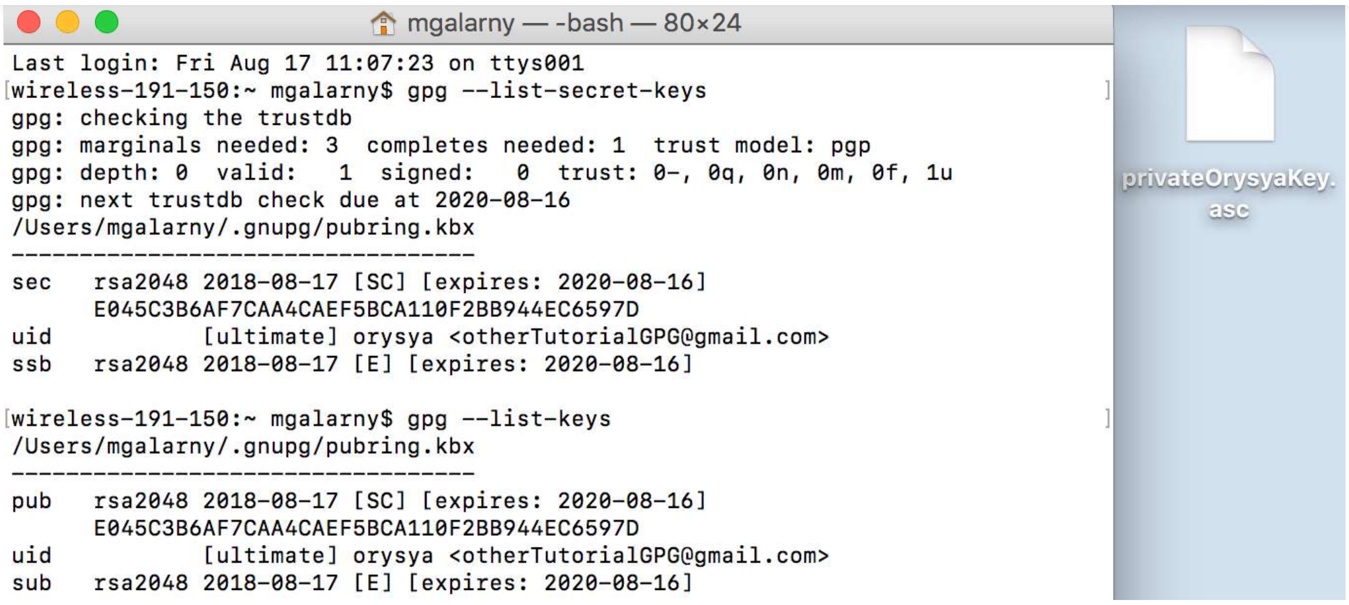
Enter `5` , `y` , and then `quit`

```
Please decide how far you trust this user to correctly verify other users' keys
(by looking at passports, checking fingerprints from different sources, etc.)

  1 = I don't know or won't say
  2 = I do NOT trust
  3 = I trust marginally
  4 = I trust fully
  5 = I trust ultimately
  m = back to the main menu

Your decision? 5
Do you really want to set this key to ultimate trust? (y/N) y

sec  rsa2048/10F2BB944EC6597D
     created: 2018-08-17  expires: 2020-08-16  usage: SC
     trust: ultimate      validity: unknown
ssb  rsa2048/911F53042A98253D
     created: 2018-08-17  expires: 2020-08-16  usage: E
[ unknown] (1). orysya <otherTutorialGPG@gmail.com>
Please note that the shown key validity is not necessarily correct
unless you restart the program.

gpg> quit
```

429 1

You can check this by using the command

```
gpg --list-secret-keys
```



```
Last login: Fri Aug 17 11:07:23 on ttys001
wireless-191-150:~ mgalarny$ gpg --list-secret-keys
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2020-08-16
/Users/mgalarny/.gnupg/pubring.kbx
----------------------------------
sec   rsa2048 2018-08-17 [SC] [expires: 2020-08-16]
      E045C3B6AF7CAA4CAEF5BCA110F2BB944EC6597D
uid           [ultimate] orysya <otherTutorialGPG@gmail.com>
ssb   rsa2048 2018-08-17 [E] [expires: 2020-08-16]

wireless-191-150:~ mgalarny$ gpg --list-keys
/Users/mgalarny/.gnupg/pubring.kbx
----------------------------------
pub   rsa2048 2018-08-17 [SC] [expires: 2020-08-16]
      E045C3B6AF7CAA4CAEF5BCA110F2BB944EC6597D
uid           [ultimate] orysya <otherTutorialGPG@gmail.com>
sub   rsa2048 2018-08-17 [E] [expires: 2020-08-16]
```

Keep in mind that you could also automate the trusting process.

```
expect -c "spawn gpg --edit-key {KEY} trust quit; send \"5\ry\r\";
expect eof"
```

Conclusion

I hope you find this tutorial useful. If you any questions or thoughts on the tutorial, feel free to reach out in the comments below or through Twitter.

Open in app

Get started