1 Download and Install GnuPG

Download and install the latest version of GnuPG. Check that you are using GnuPG version $\geq$ 2.2.0.

```
$ gpg --version
gpg (GnuPG) 2.2.9
libgcrypt 1.8.3
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.
   html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: /home/petergibbons/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

2 Inspect Your Keyring

Inspect your keyring; it should be empty to begin with. GnuPG will create a directory (`.gnupg`) to store your keyring and other data. If you want to start over, remove or rename this directory. **Be careful:** if you remove the `.gnupg` directory and you do not have a backup, you have lost all access to the public and private-keys it contained!

```
$ gpg --list-keys
gpg: directory '/home/petergibbons/.gnupg' created
gpg: keybox '/home/petergibbons/.gnupg/pubring.kbx' created
gpg: /home/petergibbons/.gnupg/trustdb.gpg: trustdb created
```

Re-run the command to check that `.gnupg` is created and your keyring is empty.

```
$ gpg --list-keys
# no output
```

3 Generate a New Keypair

Generate a new keypair using your own details. Set the keypair to expire in one year.

```
$ gpg --full-generate-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.

gpg: directory '/home/petergibbons/.gnupg' created
gpg: keybox '/home/petergibbons/.gnupg/pubring.kbx' created
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)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 4096
Requested keysize is 4096 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 Fri 9 Aug 14:07:44 2019 IST
Is this correct? (y/N) y

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

Real name: Peter Gibbons
Email address: peter.gibbons@initech.com
Comment:
You selected this USER-ID:
"Peter Gibbons <peter.gibbons@initech.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: /home/petergibbons/.gnupg/trustdb.gpg: trustdb created
gpg: key 5F4BE2E91CB38808 marked as ultimately trusted
gpg: directory '/home/petergibbons/.gnupg/openpgp-revocs.d' created
```

```
gpg: revocation certificate stored as '/home/petergibbons/.gnupg/openpgp-
   revocs.d/A0E127FBB850FB147872BF5716B52F466063A900.rev'
public and secret key created and signed.

pub   rsa4096 2020-12-07 [SC] [expires: 2021-12-07]
      F21D92E3C709A9F6C0E3D0E75F4BE2E91CB38808
uid                      Peter Gibbons <peter.gibbons@initech.com>
sub   rsa4096 2020-12-07 [E] [expires: 2021-12-07]
```

Check that the new keypair is now in your keyring.

```
$ gpg --list-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 2021-12-07
/Users/martin/.gnupg/pubring.kbx
--------------------------------
pub   rsa4096 2020-12-07 [SC] [expires: 2021-12-07]
      F21D92E3C709A9F6C0E3D0E75F4BE2E91CB38808
uid           [ultimate] Peter Gibbons <peter.gibbons@initech.com>
sub   rsa4096 2020-12-07 [E] [expires: 2021-12-07]
```

4 Generate a Revocation Certificate

Generate a revocation certificate. This will not revoke the keypair immediately, but it will allow you to revoke the keypair in the future even if you lose access to your private-key. The \ symbol indicates that the command wraps onto the next line.

```
$ gpg --output peter.gibbons@initech.com.asc.revoke \
--gen-revoke peter.gibbons@initech.com

sec  rsa4096/5F4BE2E91CB38808 2020-12-07 Peter Gibbons <peter.
   gibbons@initech.com>

Create a revocation certificate for this key? (y/N) y
Please select the reason for the revocation:
0 = No reason specified
1 = Key has been compromised
2 = Key is superseded
3 = Key is no longer used
Q = Cancel
(Probably you want to select 1 here)
Your decision? 0
Enter an optional description; end it with an empty line:
```

```
> Created immediately after keypair generation
>
Reason for revocation: No reason specified
Created immediately after keypair generation
Is this okay? (y/N) y
ASCII armored output forced.
Revocation certificate created.

Please move it to a medium which you can hide away; if Mallory gets
access to this certificate he can use it to make your key unusable.
It is smart to print this certificate and store it away, just in case
your media become unreadable. But have some caution: The print system of
your machine might store the data and make it available to others!
```

5 Publicise Your Public-Key

You can share your public-key with your friends and colleagues.

```
$ gpg --output pubkey.peter.gibbons@initech.com.gpg \
--export peter.gibbons@initech.com
# no output
```

```
$ gpg --output pubkey.peter.gibbons@initech.com.gpg.asc --armor \
--export peter.gibbons@initech.com
# no output
```

6 Import Bob Porter's Public-Key Into Your Keyring

Suppose you wish to exchange confidential emails with Bob Porter, a member of Initech's H.R. Department. You meet Bob in person and you exchange public-keys and verify the fingerprints. You receive his public-key in a file named `pubkey.bob.porter@initech.com.gpg.asc` which is available in App. A. Import his public-key into your keyring.

```
$ gpg --import pubkey.bob.porter@initech.com.gpg.asc
gpg: key EBC14BEB9460B02E: public key "Bob Porter <bob.porter@initech.com>"
   imported
gpg: Total number processed: 1
gpg:               imported: 1
```

You should now have two public-keys in your keyring.

```
$ gpg --list-keys
/home/petergibbons/.gnupg/pubring.kbx
--------------------------------
pub   rsa4096 2020-12-07 [SC] [expires: 2021-12-07]
      F21D92E3C709A9F6C0E3D0E75F4BE2E91CB38808
uid           [ultimate] Peter Gibbons <peter.gibbons@initech.com>
sub   rsa4096 2020-12-07 [E] [expires: 2021-12-07]

pub   rsa4096 2020-12-07 [SC]
      399CA583929A08FFF20686F8EBC14BEB9460B02E
uid           [ unknown] Bob Porter <bob.porter@initech.com>
sub   rsa4096 2020-12-07 [E]
```

However, you should only have one private-key. You do not have access to Bob Porter's private-key.

```
$ gpg --list-secret-keys
/home/petergibbons/.gnupg/pubring.kbx
--------------------------------
sec   rsa4096 2020-12-07 [SC] [expires: 2021-12-07]
      F21D92E3C709A9F6C0E3D0E75F4BE2E91CB38808
uid           [ultimate] Peter Gibbons <peter.gibbons@initech.com>
ssb   rsa4096 2020-12-07 [E] [expires: 2021-12-07]
```

7 Sign a Public-Key

You are confident that the real name and email address of the person that gave you the public-key were in fact Bob Porter and `bob.porter@initech.com`. You can sign the key using your private-key. Make sure to sign the `keyid` for Bob Porter's public-key (`FFCCBBBE86B8A5BA` in the listing below).

```
$ gpg --keyid-format LONG --list-keys
/home/petergibbons/.gnupg/pubring.kbx
--------------------------------
pub   rsa4096/5F4BE2E91CB38808 2020-12-07 [SC] [expires: 2021-12-07]
      F21D92E3C709A9F6C0E3D0E75F4BE2E91CB38808
uid                 [ultimate] Peter Gibbons <peter.gibbons@initech.com>
sub   rsa4096/1C154C2222D3CCBE 2020-12-07 [E] [expires: 2021-12-07]

pub   rsa4096/EBC14BEB9460B02E 2020-12-07 [SC]
      399CA583929A08FFF20686F8EBC14BEB9460B02E
uid                 [ unknown] Bob Porter <bob.porter@initech.com>
sub   rsa4096/FFCCBBBE86B8A5BA 2020-12-07 [E]
```

```
$ gpg --sign-key FFCCBBBE86B8A5BA

pub  rsa4096/EBC14BEB9460B02E
     created: 2020-12-07 expires: never    usage: SC
     trust: unknown     validity: unknown
sub  rsa4096/FFCCBBBE86B8A5BA
     created: 2020-12-07 expires: never    usage: E
[ unknown] (1). Bob Porter <bob.porter@initech.com>


pub  rsa4096/EBC14BEB9460B02E
     created: 2020-12-07 expires: never    usage: SC
     trust: unknown     validity: unknown
 Primary key fingerprint: 399C A583 929A 08FF F206 86F8 EBC1 4BEB 9460 B02E

     Bob Porter <bob.porter@initech.com>

Are you sure that you want to sign this key with your
key "Peter Gibbons <peter.gibbons@initech.com>" (5F4BE2E91CB38808)

Really sign? (y/N) y
```

Verify that your signature was added to Bob Porter's public-key.

```
$ gpg --list-sigs FFCCBBBE86B8A5BA
gpg: checking the trustdb
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: depth: 0 valid:  1  signed:  1  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: depth: 1 valid:  1  signed:  0  trust: 1-, 0q, 0n, 0m, 0f, 0u
gpg: next trustdb check due at 2021-12-07
pub   rsa4096 2020-12-07 [SC]
      399CA583929A08FFF20686F8EBC14BEB9460B02E
uid           [  full  ] Bob Porter <bob.porter@initech.com>
sig 3       EBC14BEB9460B02E 2020-12-07 Bob Porter <bob.porter@initech.com>
sig         5F4BE2E91CB38808 2020-12-07 Peter Gibbons <peter.gibbons@initech
    .com>
sub   rsa4096 2020-12-07 [E]
sig         EBC14BEB9460B02E 2020-12-07 Bob Porter <bob.porter@initech.com>
```

If you wish, you can export the signed version of Bob Porter's public-key and return it to Bob Porter. He can then publicise the fact that you verified his identity.

8 Update Your Trust

Even though you trust that the public-key in `pubkey.bob.porter@initech.com.gpg.asc` belongs to Bob Porter, you may not trust his ability verify the ownership of other public-keys. This is a private, subjective decision that only impacts your local setup.

```
$ gpg --update-trustdb
gpg: marginals needed: 3 completes needed: 1 trust model: pgp
gpg: depth: 0 valid:  1  signed:  1  trust: 0-, 0q, 0n, 0m, 0f, 1u
No trust value assigned to:
pub   rsa4096 2020-12-07 [SC]
      "Bob Porter <bob.porter@initech.com>"
 Primary key fingerprint: 399C A583 929A 08FF F206 86F8 EBC1 4BEB 9460 B02E

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
  s = skip this key
  q = quit

Your decision? 2
gpg: depth: 1 valid:  1  signed:  0  trust: 0-, 0q, 1n, 0m, 0f, 0u
gpg: next trustdb check due at 2021-12-07
     399CA583929A08FFF20686F8EBC14BEB9460B02E
```

9 Sign and Encrypt a Message

You want to send a message to Bob Porter. You want to sign the message to prove that it originated from you and you want to encrypt the message so that only Bob Porter can read it. The message is stored in a file called `message.txt` and is available in App. B.

```
$ gpg --output secret-message.txt --armor --encrypt --sign \
--recipient FFCCBBBE86B8A5BA message.txt
# no output
```

The command does not produce any visible output but it should create a file called `secret-message.txt`. Bob Porter will be able to decrypt the file and verify the signature using `gpg --decrypt --verify`. **Of course, you cannot do this since you do not have access to Bob Porter's private-key!**

```
$ # This will only work for Bob!
$ gpg --decrypt secret-message.txt
gpg: encrypted with 4096-bit RSA key, ID FFCCBBBE86B8A5BA, created
    2020-12-07
      "Bob Porter <bob.porter@initech.com>"
Hi Bob,

The IBAN for my Swiss bank account is:

CH93 1234 4567 8910 2345 6

Please transfer the funds to that account.

Regards,
Peter.
gpg: Signature made Mon 7 Dec 16:17:40 2020 GMT
gpg:              using RSA key F21D92E3C709A9F6C0E3D0E75F4BE2E91CB38808
gpg: Can't check signature: No public key
```

10 Verify a Signature

Bob Porter responds with a file called `signed-reply.txt`. He signed the file but did not encrypt its contents using `gpg --clearsign`. The contents of the signed file are available in App. C. You can verify the signature to ensure that Bob Porter did indeed send the message.

```
$ gpg --verify signed-reply.txt
gpg: Signature made Mon 7 Dec 16:20:33 2020 GMT
gpg:              using RSA key 399CA583929A08FFF20686F8EBC14BEB9460B02E
gpg: Good signature from "Bob Porter <bob.porter@initech.com>" [full]
```

Appendices

A pubkey.bob.porter@initech.com.gpg.asc

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBF/OU+sBEAC4/932o0gF5OGCngVChJIvvoos7zOQgg4N65Epc9LHGSHxvgbt
hgNI9dWLNtVCb5yPOiZxKZLOwxwLIZtiISjzovsO3NCrq2jpVWrOwmZhaWNXgWSV
vV3jo5QsUmBzRXG7EvDAEOEXs3KthmCTKPIaglUHZuORbutugCxJTtaWaxGaHxwy
/KyDe975BroQc3OSWdSmfeWKpS53oBVX6fZZ/cDgRhd83nYzgW9f4ri9/Lkg/Ct0
```

```
1bh/k6HZ3NwIyNomsynklCXCsJo6taofoTGW0ztca67tyqL2XHYlVAjnibzdA86H
eeGcLWuqaegJ0MiY00ywAtBHaPxNLZztrcBfw3fJY5w9SpBvaezrtvY89E4YuWjA
IRcj767kK/2tSfvSU2VlTWTlc6ZCMkoId4u7v7IRKuf/qI2OGF6L8fOe8TuE1rZP
Jre5TuoEkrO+N11JykHOU3/k4mma6sr7bBj/5AAoTSWs69Pzyy9x87IqDSLPtZJ5
nF3Jr1eM+mpkTPRGVFlvwTJ8T4KKD+nwl5i/K4LF7/3HGUj81zaT6+PzaJtBCzFt
doaryzHojkfOHhf3VkWAzu0CQSn7YgVmyUxMrmvZq9Z4pUFRjEWY/kaCREs/lM1U
gbiTHEbSAzbGTOtFYzULgNc+NTDs38SDy8q0Xh6mwsYDJlW8G8uJC5HuFQARAQAB
tCNCb2IgUG9ydGVyIDxib2IucG9ydGVyQGluaXRlY2guY29tPokCTgQTAQgAOBYh
BDmcpYOSmgj/8gaG+OvBS+uUYLAuBQJfzlPrAhsDBQsJCAcCBhUKCQgLAgQWAgMB
Ah4BAheAAAoJEOvBS+uUYLAuzSsP+wTm8MSpeafPaEY7zzVE2QXlZpV147Dkgv6E
rZ7USQAyq81S8iF7EMc/d1Q1Of7od3vjFI80oyngIAhlBZBVT4EzgydEju3oXmcy
VxCKKungdjpLZNI9v37tXWxrGOzq7cBdiVyT+X7gRAXGTewvud5OuRsJX0RLxuCM
mz8jX7yxu2yFg31VBX8WHHSuSOoWUIMEobKXnVurEt3b1tK/BGhiKrN9eivxLxDi
LEBHRGFolZDYY90ib6nF4JP2zJmxhnZlHaxb/XS4o5kWokKpzwOp/ZdYk0j8FTc0
k0pGVuyksO3Nwq4mshxedIK84Yzb67zEiebwpiTSHUyj89CkuN+9IJwjgm64ca7Z
rfkT8poZ9jxXXZNOuZHq+mpa0I99WgOmDj2GIZu6zy1iE716r5NkayRX30nRdods
85ibTbNP3JI9rPe0r2tHEBGcmC/jdFCJUTaLohLxfztrAJTfiPHEzmH4NHX/ywkl
sQCrdnxB/kY2xbq/Hf6iDWolmPcDtg0YMmmg4fo91TWhGhCnXRKCLnddTcyCeMK3
Qkjx+7v8QPyigpzF0p42bcVeDu2HiICxcAUIu70V140hKrwRPZ8pTC/qKbqOOHyQ
PT/1BFIx1+ZUoLDnOIM8moHCECcN9NZXOLf8lZDxLGxMZD7NNFlncQPbq1chX9qp
tfy2CR6XuQINBF/OU+sBEADbhirTWwD420fQ0I+zTn0NpuCJkBrXvv6y3t5bWF9x
WtbR9JejkUOqk/nbLB3FDjUu9sYG0yLZDbYoIjlxugfpSUpR89jroTTvP1S5LCCs
Mvy7XbkiHASGiiPC7PpYHoEZqn1Tli/EFIkV+yMm/wk2bTGyZbPGjdzp6UfnO0q+
A8NLfb8e0OwweZKUfI7SyPvc6tMgmIOfmmMCcEQc4cT2Nesbuvqp1SBatHzYmVra
Sf1GPJWSEOi1rfxCyoO4dDrIVkhpGox30AbDtoK5rXyqLLYLXdLcij8FBCX1SbYJ
FUCirEUUMrqmI/UN00707rV8MfOdQwn6AgzQ+MSqXoEjTXGbKpAdr2tGYIMzPY33
LEGPZHBHIGUuj06i0z1Kb2UziLC6ZjdycpoVxoln7b7UZpB/b8ooNFfhgKwZpQ9m
vt319lqScWYYeH5Wld4fkEXRxYhyjDCTFxaM0pSQJ6Cj765dJUd8OFqPuZImNbQl
qCw1Xyu7qCPUMeWQ1ZZz5jilZri63jvGJICZgPkoojfqLhbEzGheOZ/ANBJjeahT
5Or4NSwmSeftmI2iutNOPmOkOtKSJgwap7BkNaiRk8NbMf8V0rfgUhXG2uotXTv8
NprrS1Hwhlmz2z+gHHCpikuCJ74XNW1g+/A3tUm0RdQFinRZke1f+T/1MwsCR89z
1QARAQABiQI2BBgBCAAgFiEEOZylg5KaCP/yBob468FL65RgsC4FAl/OU+sCGwwA
CgkQ68FL65RgsC4GCA/+KglfpjrOVJI5xhOkR/Fz45HmTksgwLAOR0hPPw8kDSkx
rs4/f9CSWz7QGen46qTUB3B2yHuI1Qr8UDBiQErmKZDTmupW2kIZuOTcGIQcYlWR
a2/TnHGsKJBGR+xldjjew3QTyG74op0LiUMAR+ZrH1TNJdEdXyALT8qpcapCYRbq
T1HXI7vgOOooVhDP8mPs0B9Uqf/QDPFfS6xhz1EB+LBRlcW/+Gh/g8Dhg/KVQVkl
mS76JuUDG+1ERIoTRpRhJzW8RzXCBlsdNGLQiKjgyRW/sLgrr16nfya/1Xp3EI+8
ftWlyfuDh2vQBF1tubOw3XgwZR+kNrgocO5jTWPoiwsL7mCnmxtUlIlFkYeDxUpm
Szv39h3EasCwQEo4Hk8O9G4vMTv1sfW9WuxduKqGIF+cPLmleCTj/LixBSfktHQJ
1JKjUayDTv8JHaGIUQHPpoHTjfj5A9RZXSLFv26cnFNpqfE+ubLtUO0vQaHrM2xU
ZtFYTEfOlfYn2L3T1Vt50D0ziXE9AF2YrQgiAKEOWqiYthWbz3dv32QwYTm4bs7f
rZO1yHB20z6m9QVSOdUApgNqtkwjy2s56XlNggZTCuJaZ+qPAyzRgqNE6vIt++7Z
LlkFiJRb5PLdQS4/QYjFBt/X1CscVs3wRJP+3/jMXqp2gvJ7hMWXx3OWiBcTDcE=
=3Hgh
-----END PGP PUBLIC KEY BLOCK-----
```

Listing 1: pubkey.bob.porter@initech.com.gpg.asc

B message.txt

```
Hi Bob,

The IBAN for my Swiss bank account is:

CH93 1234 4567 8910 2345 6

Please transfer the funds to that account.

Regards,
Peter.
```

Listing 2: message.txt

C signed-reply.txt

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA256

Hi Peter,

All done!

Regards,
Bob.
-----BEGIN PGP SIGNATURE-----

iQIzBAEBCAAdFiEEOZylg5KaCP/yBob468FL65RgsC4FAl/OVlEACgkQ68FL65Rg
sC7TgxAAjX1usB6uMOMz4eF6WzlzCWh0R5HJUI1qILGF0GQaDGswW2yZgAbiDT7v
OC2q6lO9PItX5OeLcOyksWwARX1LmFhQoFx0j1UAI9GQOxZqtCVWkolEfBDDN1oZ
vGJRa7bRsboD7oTf6XEEwAvp/G66RfsUs6BuziAC8YTdxzeaGngeKBuu1XcxY4UV
XtWNB57JDmSnO91JjnNFTr9Mdm3ntR6iF0Wb2Rtn5hn3ezjzCEnVbKD/t7RJfXgy
3v5rstd3eEVfJTk1yaeOAna9Qjhlv1t1vXssEci+yApAM1VRf4JogiweJ9BbPk6I
PQcx6MjWhfVZx0fu7ysr5Y3F96m3PB0hZMWONDp4pUezS17nbHa8JvUHiV4yxX92
pqHWwEzgyJRxPfmHoVKEWjG4XzfqfDQVLshmutEBJYoEBTzpfKxb+hvUs82Ybn9U
8onXGjEr1pFHFQL0nGg//lK35LFsUtIP7uE33YF0i3O8t/IjQ8gyE/uGN+3K+L+E
IAAa5FDgo+Y2T6Jc8H1o4ANqbhtUIATuNqYq9JieMuT4HqgvnsVIox91PdWeZCYD
1vGHSNKOOinEsnQCUlFJOxVqlIwyxIgkH1VvkOko+PzhYPnKBCH/0aw+cCR+zVqg
mCie9SYpZqzBAXCCWJx9K7HFMhS5Y8wpt6iNcfc0VdPM5/vjYcg=
=jUAI
-----END PGP SIGNATURE-----
```

Listing 3: signed-reply.txt