

## GPG (GNU Privacy Guard)

GPG 是一個加密軟體，主要用來做 **加解密檔案** 還有 **數位簽章 ( Sign )**，今天要用到的就是他簽章的功能

如果沒聽過數位簽章也沒關係，就把它想成現實生活中的 **蓋章** 就好。想想當你去銀行開戶時，銀行除了看你的身份證之外，通常還會要求你**蓋章**。日後如果你還要辦一些有的沒的，銀行就會要你**蓋同一個章**，因為那個章就代表你本人

數位簽章也是如此，只不過電腦上的簽章是用 **RSA** 非對稱加密來做的，藉此保證你的簽章不會被偽造

### 安裝 GPG

在各個 package manager 上都可以找到 GPG，跑個腳本就裝好了～

```
$ brew install gpg           # macOS
$ sudo apt-get install gnupg # Debian, Ubuntu
$ sudo yum install gnupg     # CentOS
```

安裝完跑 `gpg --version` 看到版本就沒問題了

```
Larry-Pro ~ > gpg --version
gpg (GnuPG) 2.2.20
libgcrypt 1.8.5
Copyright (C) 2020 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.
```

## 產生一組 RSA 公私鑰 ( 刻印章 )

接著就是要去 刻一個屬於自己的印章，只不過不用出門，只需要下個指

令 `gpg --generate-key`，然後輸入名字跟信箱 ( 不一定要跟 Github 一樣 )

```
Larry-Pro ~ > gpg --generate-key
gpg (GnuPG) 2.2.20; Copyright (C) 2020 Free Software Foundation, Inc.
GnuPG needs to construct a user ID to identify your key.

Real name: Larry Lu
Email address: pudding850806@gmail.com
You selected this USER-ID:
    "Larry Lu <pudding850806@gmail.com>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit? 0
```

接著他會問你要不要設密碼，我自己是都沒在設 XD，一直 Enter 就可以了

結束後下 `gpg --list-keys` 就可以看到現在有一組新的 RSA 公私鑰，這組

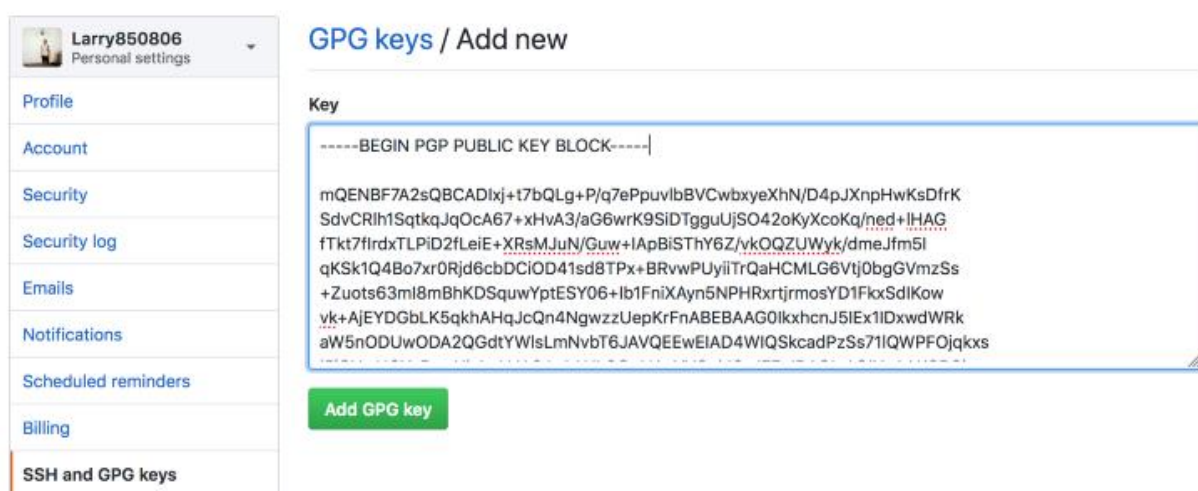
Key 就是你剛刻好的印章，而 Key 的 ID 是 `A471A74FCD.....6C9798821F`

```
Larry-Pro ~ > gpg --list-keys
/Users/larry/.gnupg/pubring.kbx
-----
pub      rsa2048 2020-05-17 [SC] [expires: 2022-05-17]
         A471A74FCD2B3BD654163C53A3AA4C6C9798821F
uid           [ultimate] Larry Lu <pudding850806@gmail.com>
sub      rsa2048 2020-05-17 [E] [expires: 2022-05-17]
```

## 把公鑰給 Github ( 給銀行看你的印章 )

生出公私鑰之後，接著要讓 Github 把你的 **公鑰** 記起來，那要怎麼拿到你的公鑰呢？

下指令 `gpg --armor --export <KEY_ID>` 噴出的那一長串東西就是你的**公鑰**，要一字不漏複製起來，包括開頭的-----BEGIN PGP...BLOCK-----，然後整串新增到 [Github 個人設定裡面的 GPG keys](#)




完成後就會看到下圖的畫面，代表 Github 已經把你的公鑰記起來了

因為 RSA 的公私鑰是一組的，所以往後只要在自己電腦上 **用私鑰簽章**，push 上去 Github 就會用公鑰去確認是不是你本人簽的

## GPG keys

[New GPG key](#)

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.



**Email address:** pudding850806@gmail.com

**Key ID:** A3AA4C6C9798821F

**Subkeys:** ACB4F009626EE755

Added on 17 May 2020

Delete

如果還是覺得有點抽象的話，就把這個步驟想成是**第一次**去銀行開戶時要蓋印章，一但蓋下去以後銀行以後就認那個章，只要你的印章沒有弄丟，那別人就不可能假冒你的身份

## 開啟簽章功能

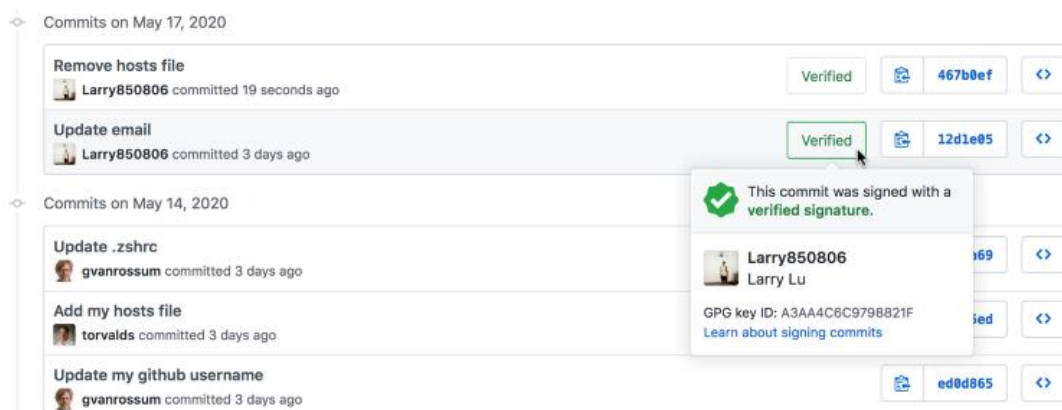
最後一步就是要在本地端把簽章的功能開起來，可以想成是告訴你的 git 每次 commit 時都要蓋印章，設定完之後就可以收工了～

```
$ git config --global commit.gpgsign true # 開啟簽章功能
```

```
$ git config --global user.signingkey <KEY_ID> # 設定簽章用的私鑰 (印章)
```

以後你在本地端 `git commit` 時 git 就會自動用 **私鑰** 幫你的 commit 簽章。因為 Github 已經有你的 **公鑰** 了，如果 Github 能用公鑰解開你的 commit，就代表你是用同一組私鑰簽的，也就能證明是你本人

一旦 Github 確定你是使用者本人而不是偽造的，他就會在你的 commit 後面標上一個超帥的 **Verified** 記號，看起來就超厲害



## 私鑰有可能被偽造嗎？

安全性方面，因為 GPG 預設會產生長度 2048 bit 的公私鑰，以現在的計算速度就算是用超級電腦也不可能在有生之年破解，所以只要你沒有把自己的私鑰洩漏出去，就不可能被偽造身份

如果有一天不小心把私鑰洩漏出去了，那也不用著急，就先到 Github 把之前的公鑰刪掉（報失印章），然後重新生一組 Key 就好了

## GPG keys

New GPG key

This is a list of GPG keys associated with your account. Remove any keys that you do not recognize.



**Email address:** pudding850806@gmail.com  
**Key ID:** A3AA4C6C9798821F  
**Subkeys:** ACB4F009626EE755  
Added on 17 May 2020

Delete

這樣即便某路人甲有你的私鑰，Github 也不會承認他，因為對應的公鑰已經被拔掉了

## 總結

今天講了怎麼在 Github 上偽造身份，雖然感覺很鬧但說不定你真的會用到，也許你很想為 [avbook](#) 專案盡一份心力，但又覺得被同事發現很不好意思，這時就可以直接用同事的名義去貢獻，超實用的吧～

另一方面，為了防止自己的身份被冒用，這邊也建議大家都使用 GPG 幫自己的 commit 簽章，畢竟整個設定的過程還滿簡單的

而且如果有天隔壁座位的正妹同事真的問你「你怎麼會貢獻到 avbook，難不成你有在用嗎？」，你就可以光明正大的說出「我的 commit 都有經過 GPG 簽章，那個沒有 verified 的 commit 不可能是我的」，以保持你正人君子的好形象～

<https://medium.com/starbugs/how-to-fake-the-author-of-git-commit-f44453b70afc>