Name 林昕銳

ID 41047035S

3.1 Equivalent Assumption

- **Given g^α, g^β, compute $g^{\alpha\beta}$**

  因為 G 群是循環的，可以將 g^α 和 g^β 看作 g 的冪次，根據群的性質:

  $$g^\alpha \times g^\beta = g^{\alpha+\beta}$$

  這裡需要計算的是 $g^{\alpha\beta}$ ，所以計算 $\alpha \times \beta$，然後取模 q (因為群的階是 q)，計算 $g^{(\alpha\times\beta) \mod q}$ 可以在多項式時間內完成 (deterministic poly-time equivalent)，因為 α 和 β 的值都是小於 q 的。

- **Given g^α, compute g^{α^2}**

  應用群的冪次運算規則，計算 α^2 再取模 q，然後計算:

  $$g^{(\alpha^2 \mod q)}$$

  因為指數運算和模運算都可以在多項式時間內完成，這個問題也可以在多項式時間內解開 (deterministic poly-time equivalent)。

- **Given g^α and $\alpha \neq 0$, compute $g^{1/\alpha}$**

  需要找到 a 的模 q 逆元，也就是某個數 β 使得:

  $$\alpha\beta \equiv 1 \pmod q$$

  因為 q 是質數，使用擴展歐幾里得算法可以在多項式時間 (poly-time) 內找到 β，然後計算 g^β 即可。

- **Given g^α, g^β with $\beta \neq 0$, compute $g^{\alpha/\beta}$**

  解開 $\beta \neq 0$ 時的 β 的模 q 逆元 c，使得:

  $$\beta c \equiv 1 \pmod q$$

  計算 $\alpha \times c \mod q$ 後，再計算:

  $$g^{(\alpha\times c \mod q)}$$

  使用擴展歐幾里得算法找逆元和後續的乘法、模運算都是 deterministic poly-time equivalent。

3.2 Implicit certificate

1. The equivalence of Alice's private and public keys. That is, prove $Q_A = \alpha G$.

Alice 的私鑰和公鑰計算方式:

- 私鑰: $a = e'\alpha + s$
- 公鑰: $Q_A = e'\gamma' + Q_{CA}$

e' 是從證書 $Cert$ 重新計算的 hash 值， γ' 是從 $Cert$ 解碼得到， s 和 γ 是由 CA 計算。

根據 CA 的計算過程有 $\gamma = \alpha G + kG$ 且 $s = ek + c$ 。因此:

$$Q_A = e'\gamma' + Q_{CA} = e'(\alpha G + kG) + cG$$

可以重新寫成:

$$Q_A = (e'\alpha + e'k)G + cG = (e'\alpha + s - c)G + cG = (e'\alpha + s)G = \alpha G$$

這裡使用 $e' = e$ 的事實，因為 e 和 e' 都是對同一證書 $Cert$ 的 hash 值。

因此得出 Alice 的公鑰 $Q_A = \alpha G$ ，其中 a 是她的私鑰。

2. Please show that given CA's public key Q_{CA}, without the CA secret key c, it is computationally infeasible to generate a valid certificate. The certificate is valid if $Q_A = \alpha G$.

有效的證書代表每個人都可以從 $Cert$ 和 Q_{CA} 計算出 Q_A 使得:

$$Q_A = \alpha G$$

α 是根據 $Cert$ 和 s 計算出來的私鑰，並且 s 是 $ek + c$ 。

因為 e 和 k 是隨機選擇的，並且 c 是保密的，沒有 c 的知識，外部攻擊者無法計算出正確的 s 。

如果攻擊者不知道 c ， 攻擊者需要解開以下等式找到 s:

$$s = ek + c$$

但因為 e 和 k 是隨機的並且 c 未知，這式子有多個可能的解，而且沒有足夠的資訊確定哪個是正確的。

因此使 $Q_A = \alpha G$ 成立的 $Cert$ 和 s 組合在計算上是不行的。

得出: 除非知道 c ，否則無法生成有效的證書。

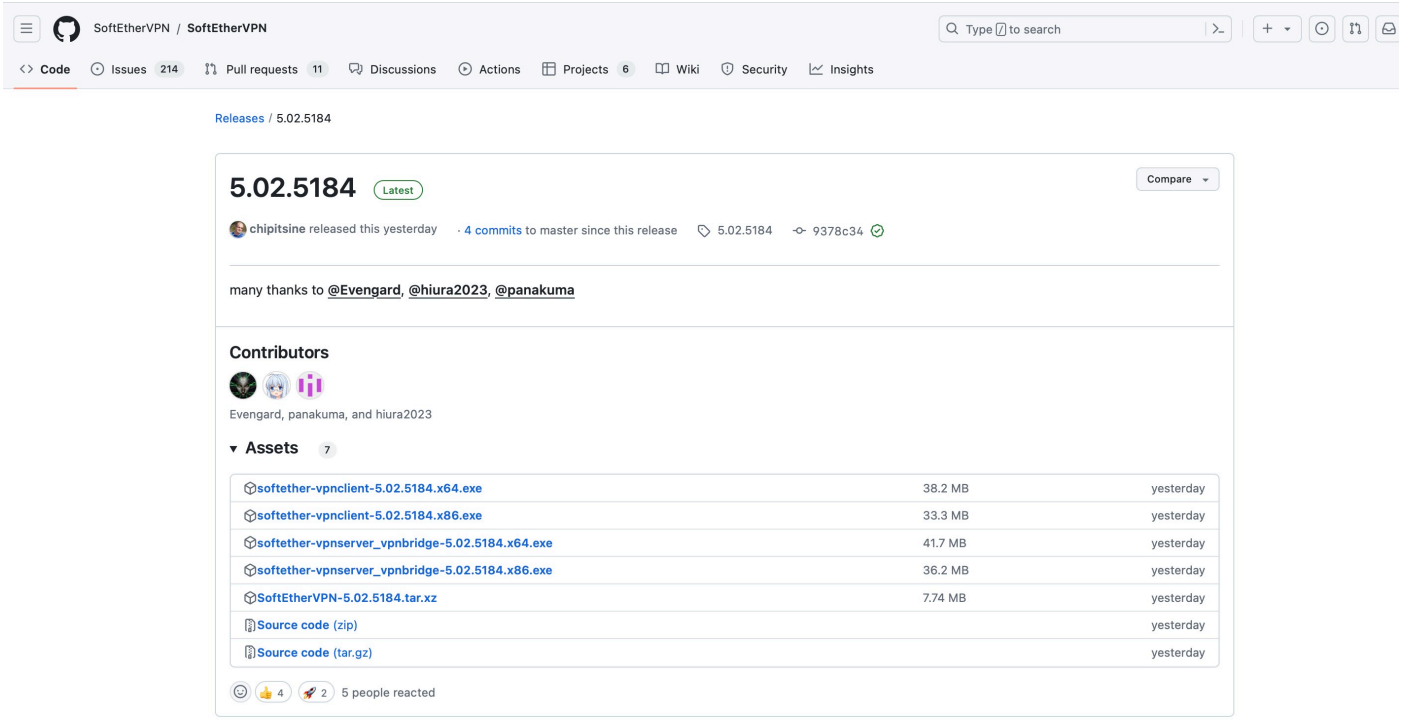
3.3 SoftEther 25

SoftEther VPN 是一款免費的、開源的多協議 VPN 軟體，可以遵循以下包括從 GitHub 下載、安裝到設定 VPN 伺服器和客戶端的一步步步驟 (此教學以 Linux 為例):

步驟一: 下載 SoftEther VPN

在 GitHub 上 Releases 頁面下載:

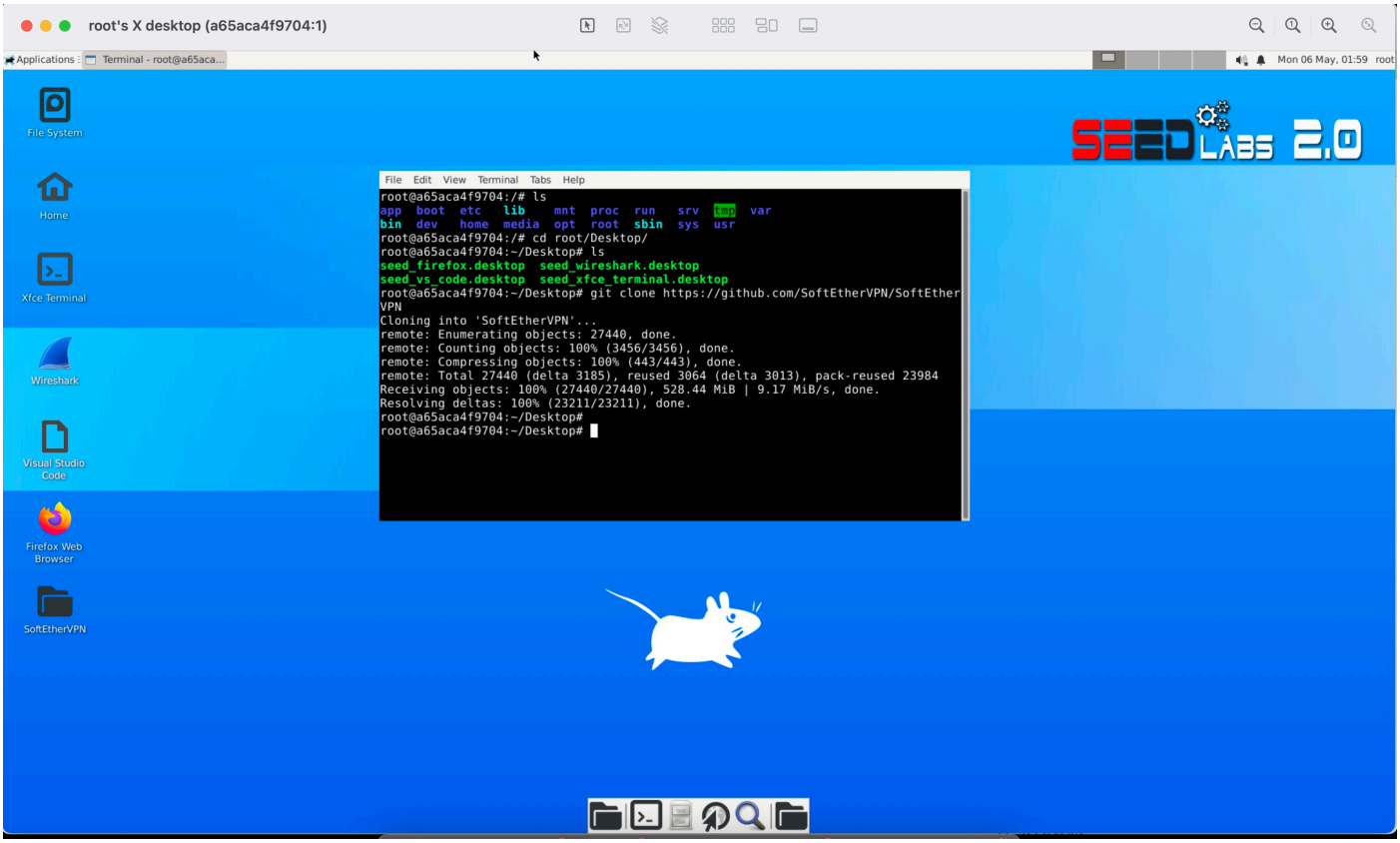
1. 首先進入 SoftEther 的 GitHub 頁面:SoftEther VPN on GitHub
2. 滑到頁面下方，找到「Releases」。
3. 選擇作業系統的版本下載。SoftEther 支援 Windows, Linux, macOS 等多個平台。



使用 Git Clone 下載:

1. 在終端機執行以下指令:

```
git clone https://github.com/SoftEtherVPN/SoftEtherVPN
```



步驟二: 安裝 SoftEther VPN

在 Linux 上安裝:

在 Linux 上安裝 SoftEther VPN 需要先安裝一些依賴函式庫，可以使用以下指令安裝:

```
apt update
apt install cmake
apt install pkg-config
apt install libsodium-dev
apt install libncurses5-dev libncursesw5-dev
apt install libssl-dev
apt install libreadline-dev
```

1. 解壓縮下載的檔案。
2. 通過終端進行安裝。首先切換到解壓縮後的目錄，例如:

   ```
   cd SoftEtherVPN
   ```

3. 執行以下指令進行編譯和安裝:

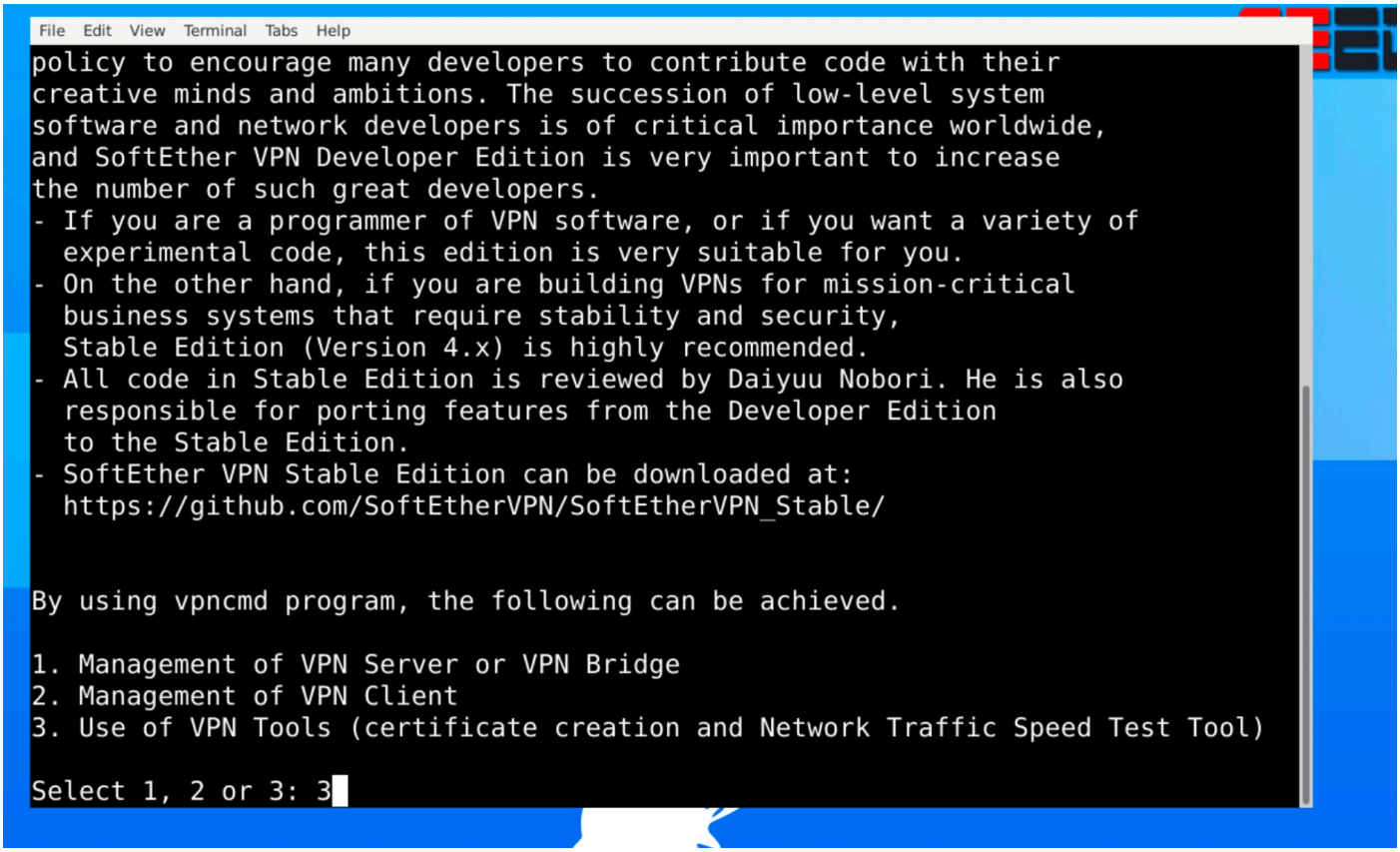
   ```
   cd SoftEtherVPN
   git submodule init && git submodule update
   ./configure
   make -C build
   make -C build install
   ```

步驟三: 確認 SoftEther VPN 的狀態

1. 可以使用以下指令:

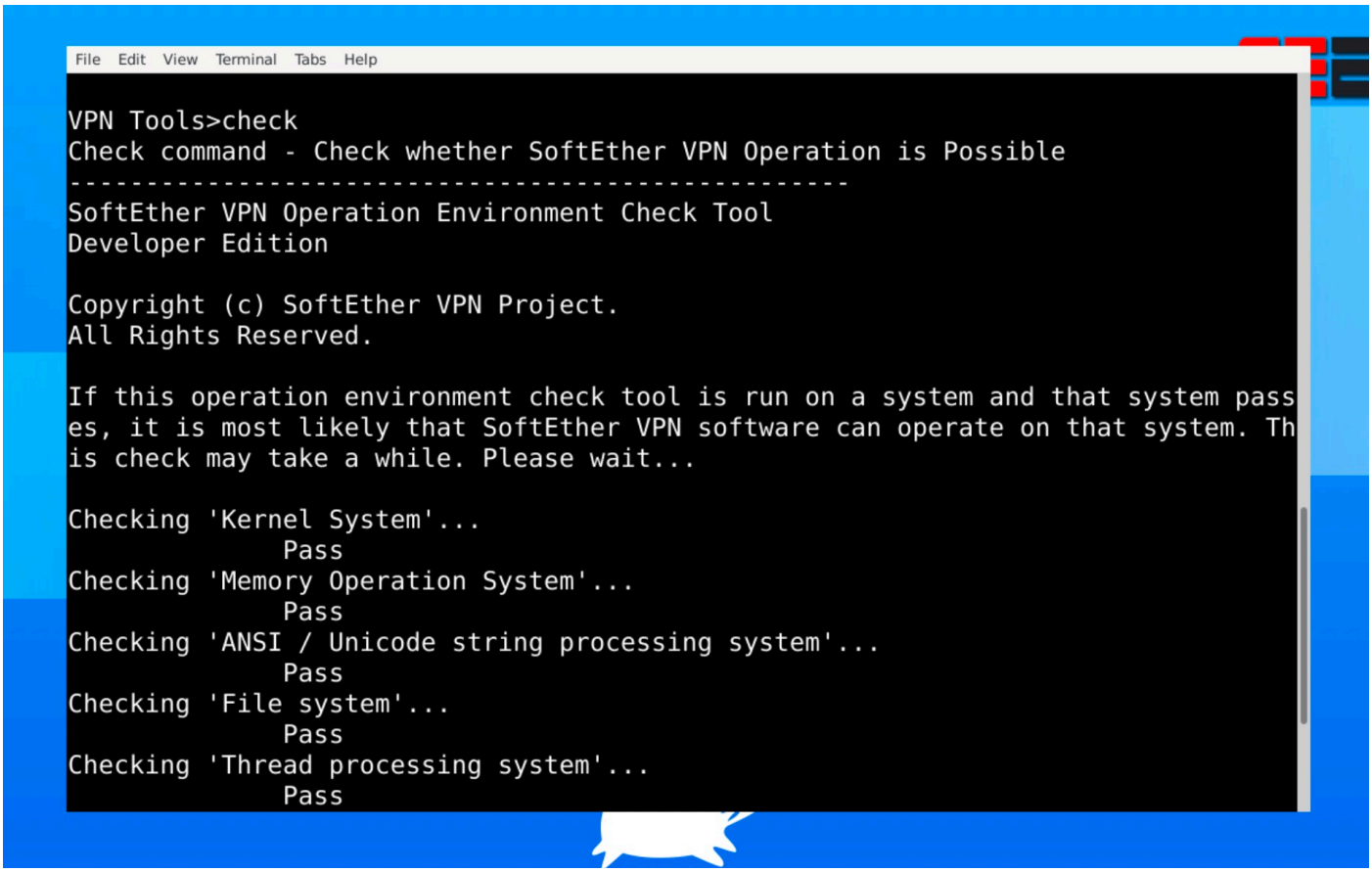
   ```
   vpncmd
   ```

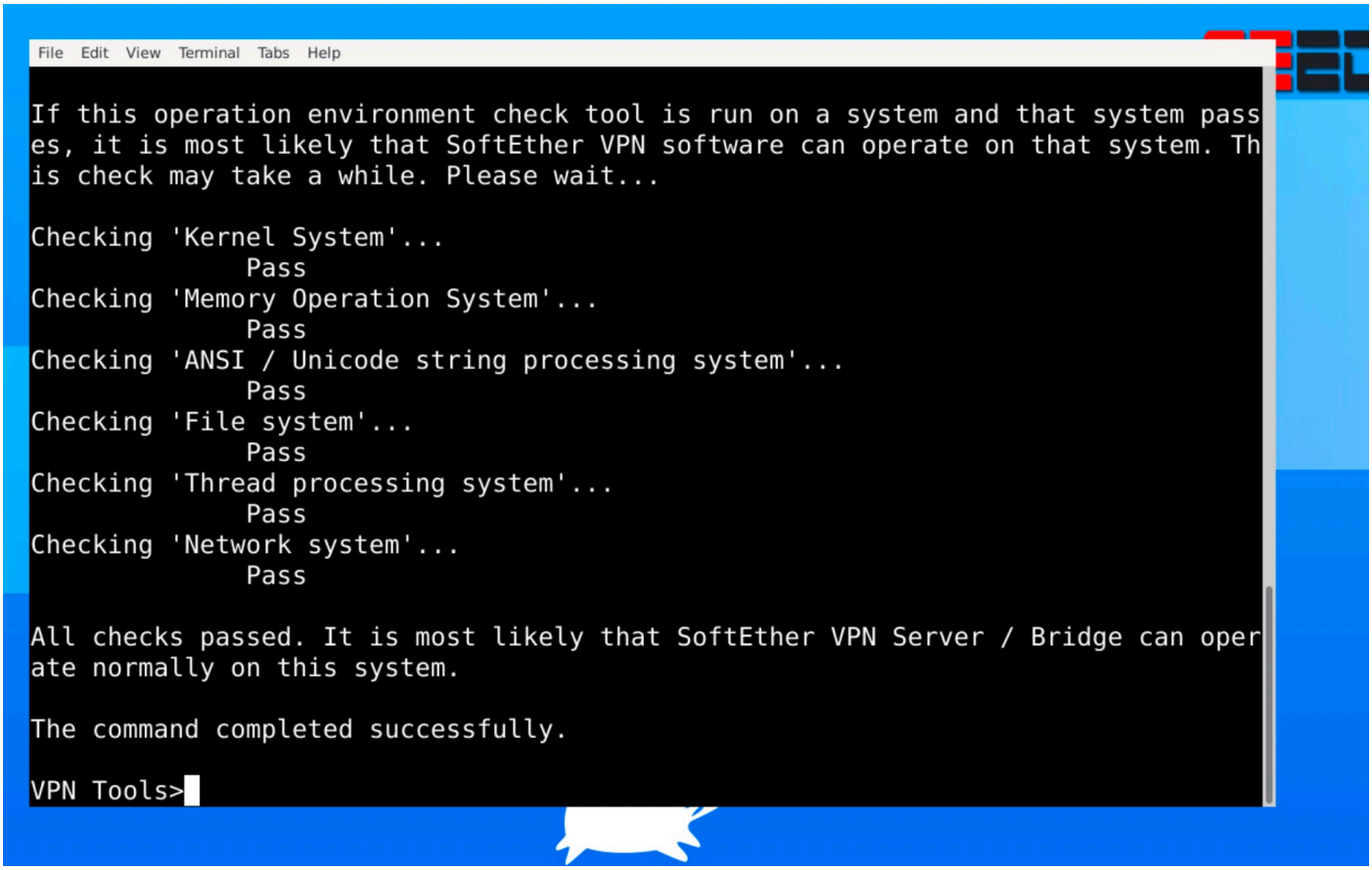
2. 選擇 3. Use of VPN Tools:



3. 使用以下指令，可以看到 All pass

```
check
```

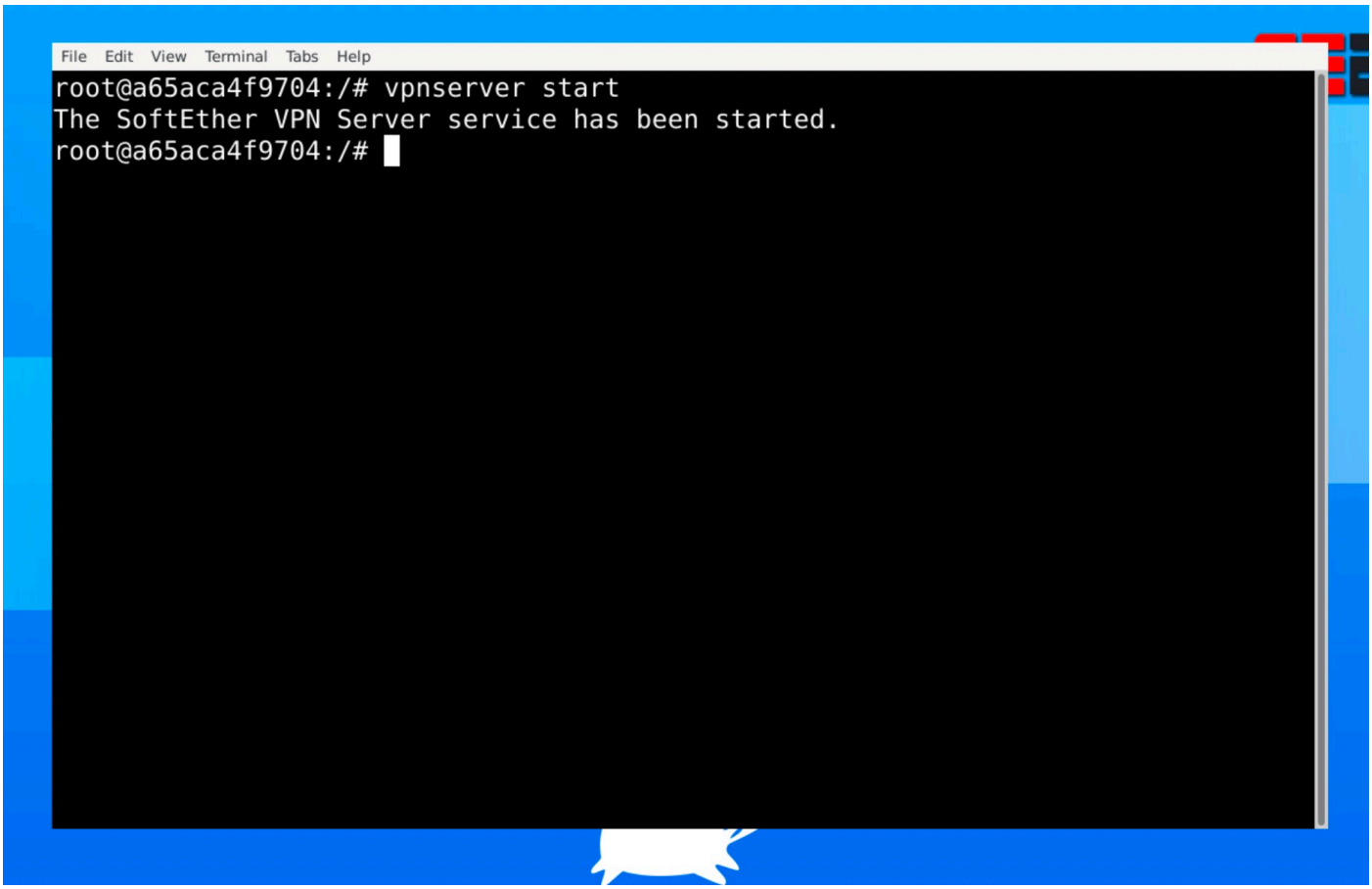




步驟四:設定 SoftEther VPN Server

1. 啟動 SoftEther VPN Server，可以用以下指令:

```
vpnserver start
```



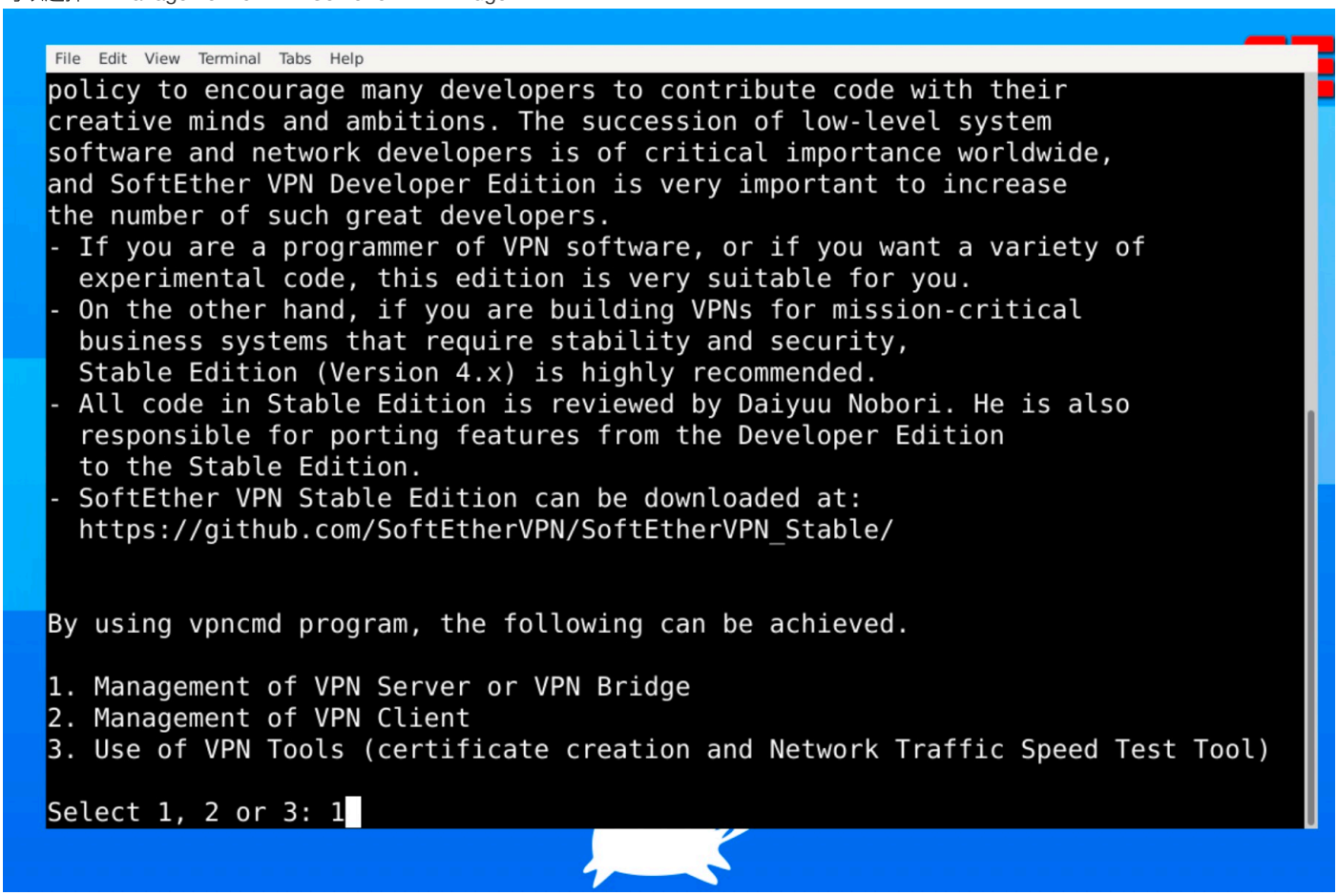
如果要停止伺服器，可以使用以下指令:

```
vpnserver stop
```

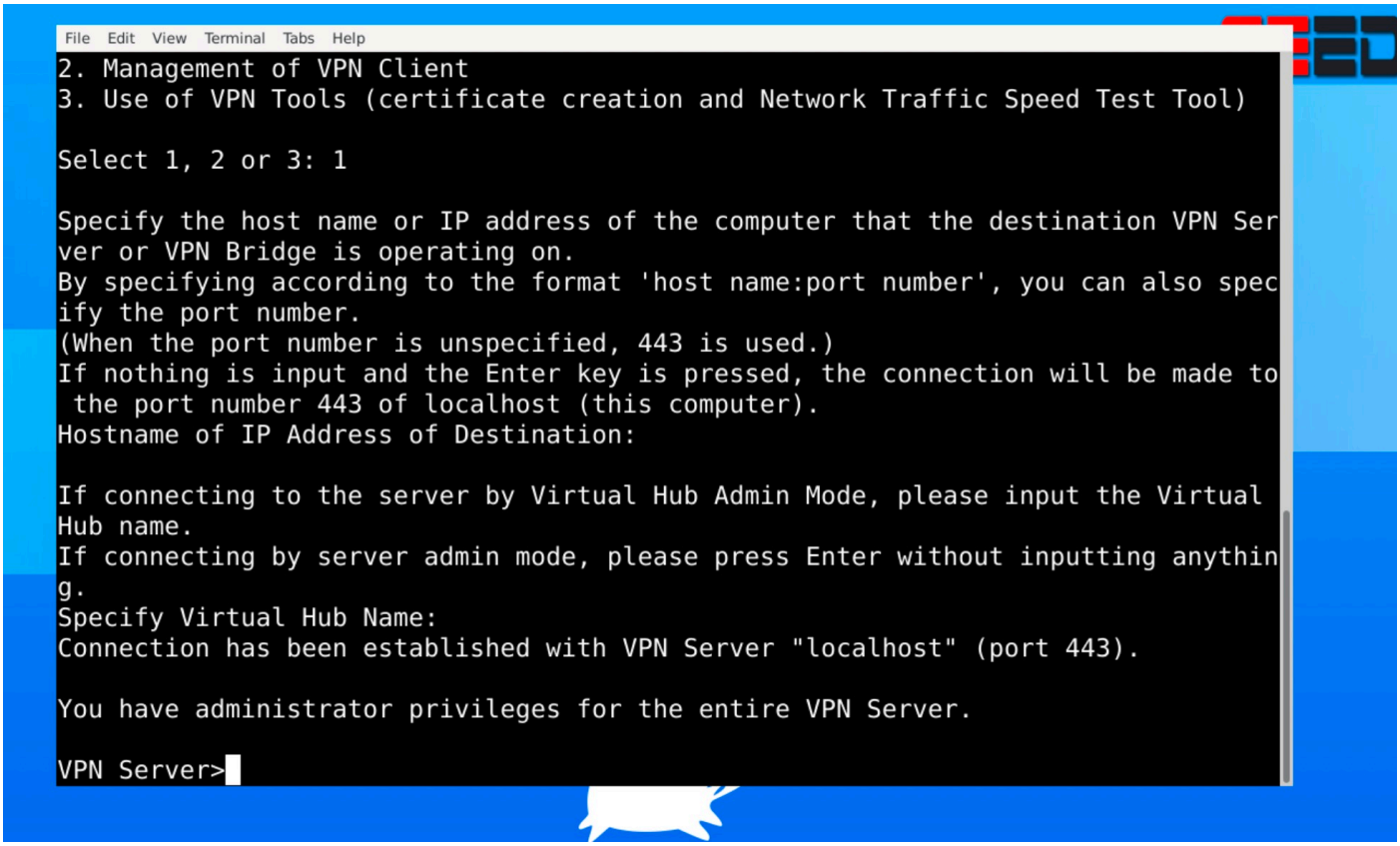
2. VPN 伺服器設定可以使用以下指令:

```
vpncmd
```

3. 可以選擇 1. Management of VPN Server or VPN Bridge:

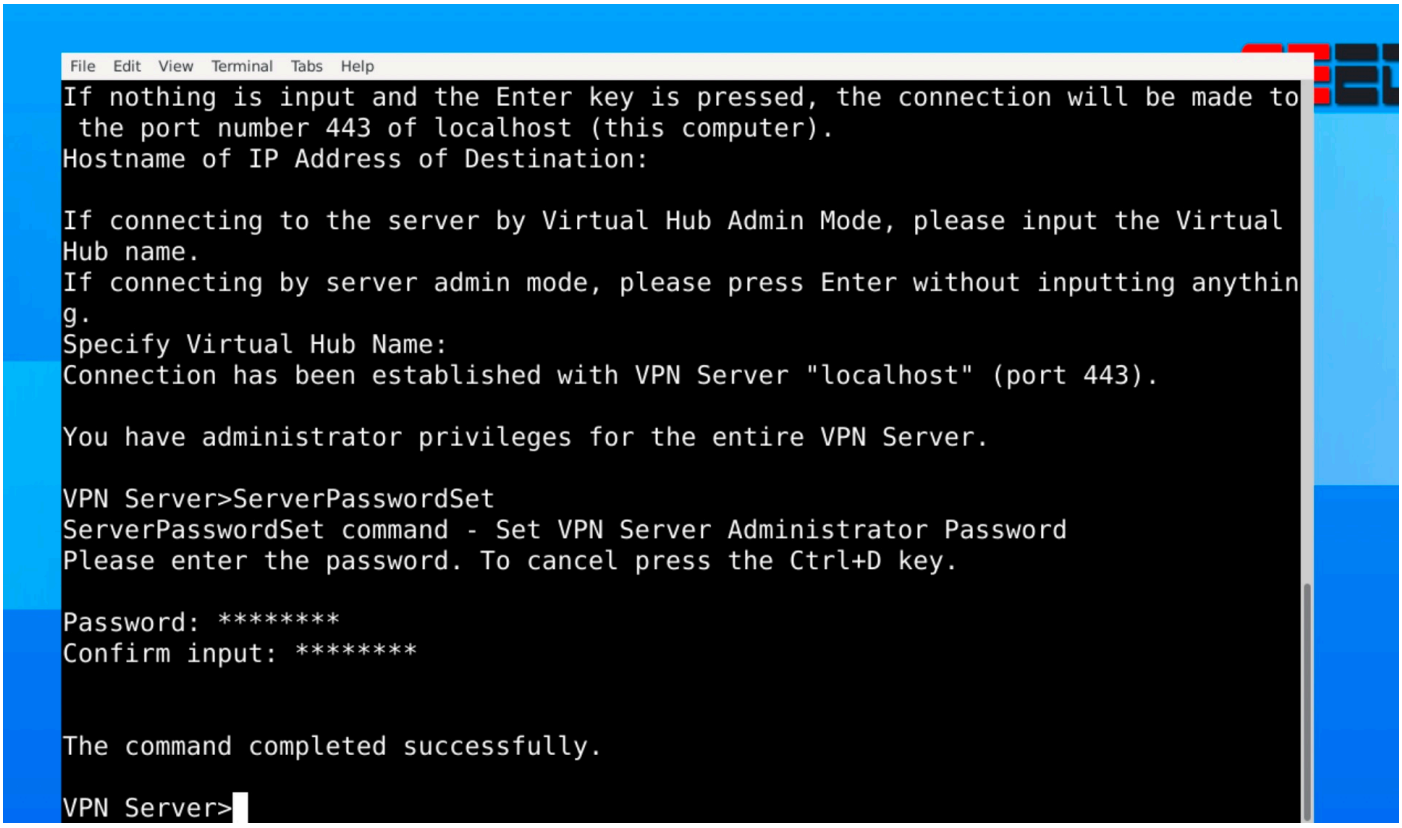


4. 連續按兩次 Enter 鍵，直到出現以下畫面:



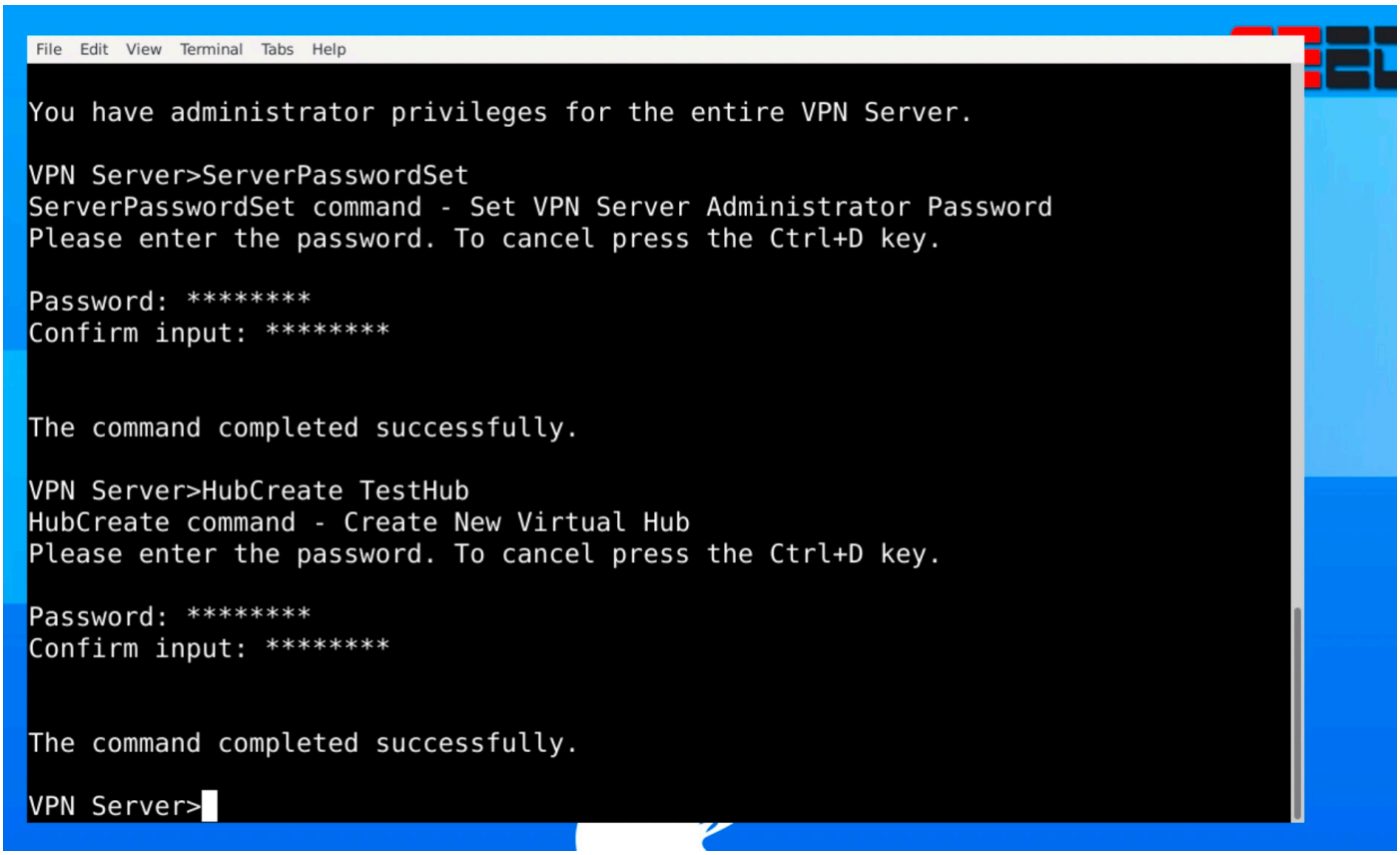
5. 設定 VPN Server 的密碼，輸入密碼並確認:

```
ServerPasswordSet
```



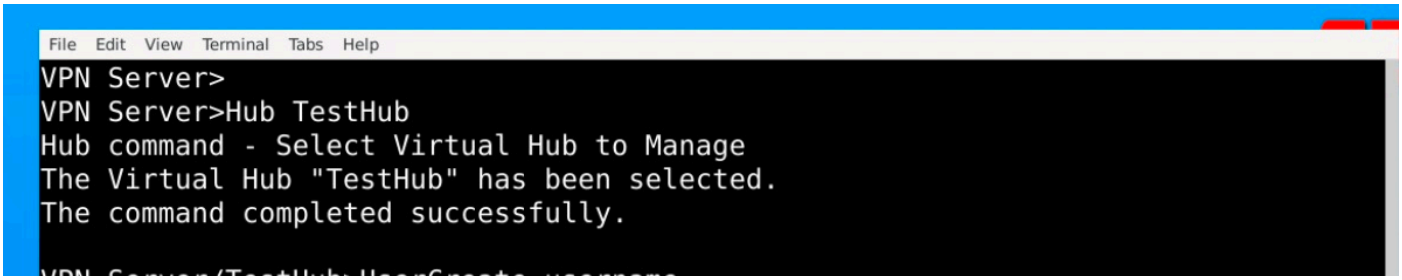
6. 設定 SoftEther VPN Server 的 Hub，輸入密碼並確認:，可以使用以下指令:

```
HubCreate TestHub
```



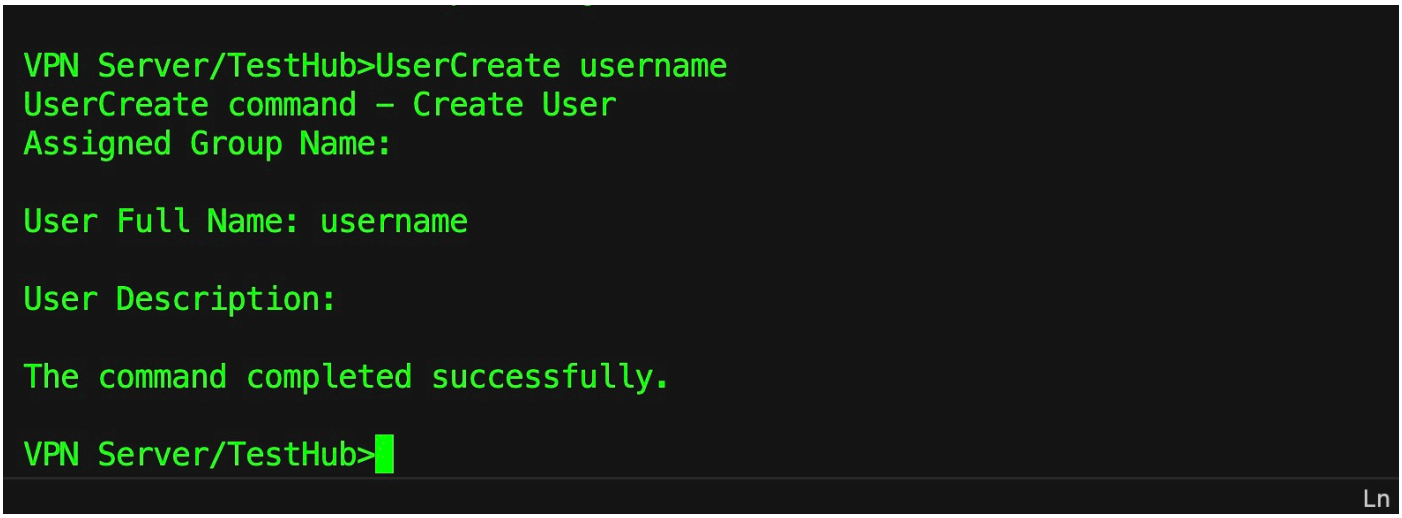
7. 進入 Hub，可以使用以下指令:

```
Hub TestHub
```



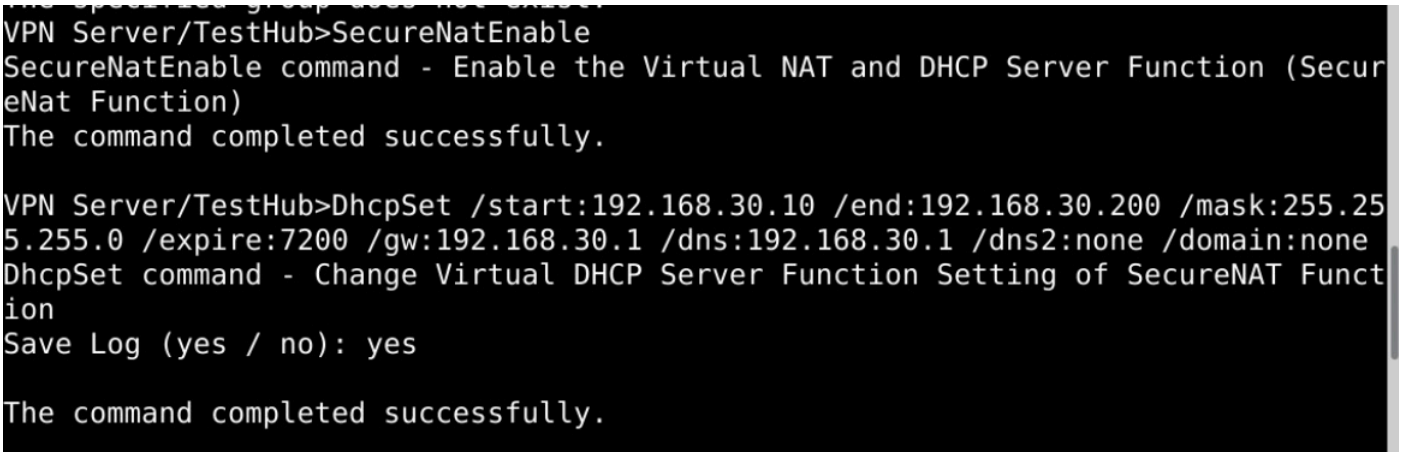
8. 創建一個新的用戶:

```
UserCreate username
```



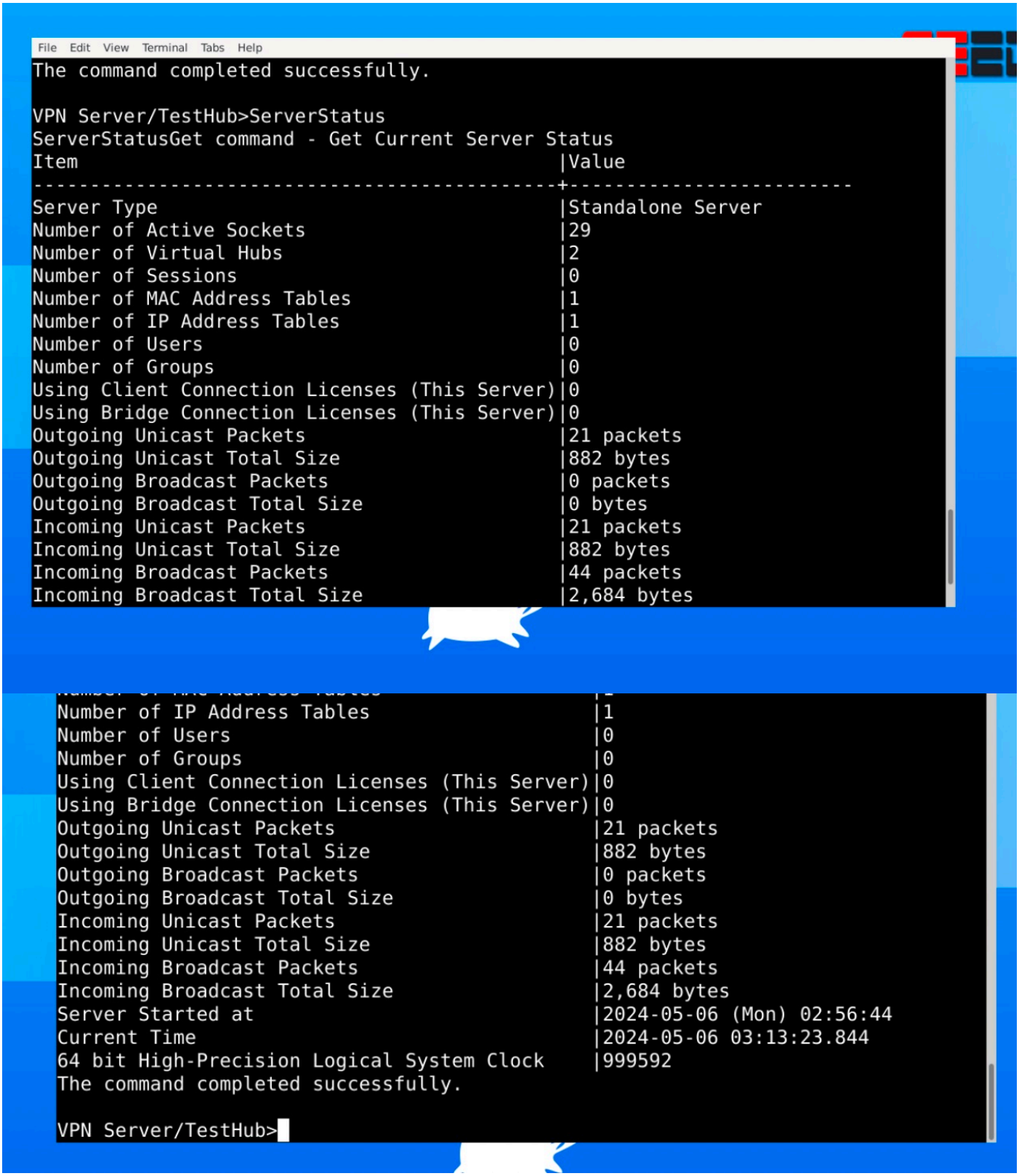
9. 啟用虛擬 NAT 和 DHCP 功能:

```
SecureNatEnable
DhcpSet /start:192.168.30.10 /end:192.168.30.200 /mask:255.255.255.0 /expire:7200 /gw:192.168.30.1 /dns:192.168.30.1
/dns2:none /domain:none
```



10. 確認伺服器狀態:

```
ServerStatus
```



步驟五:設定 SoftEther VPN Client 並連接到 Server

1. 啟動 SoftEther VPN Client，可以用以下指令:

```
vpnclient start
```

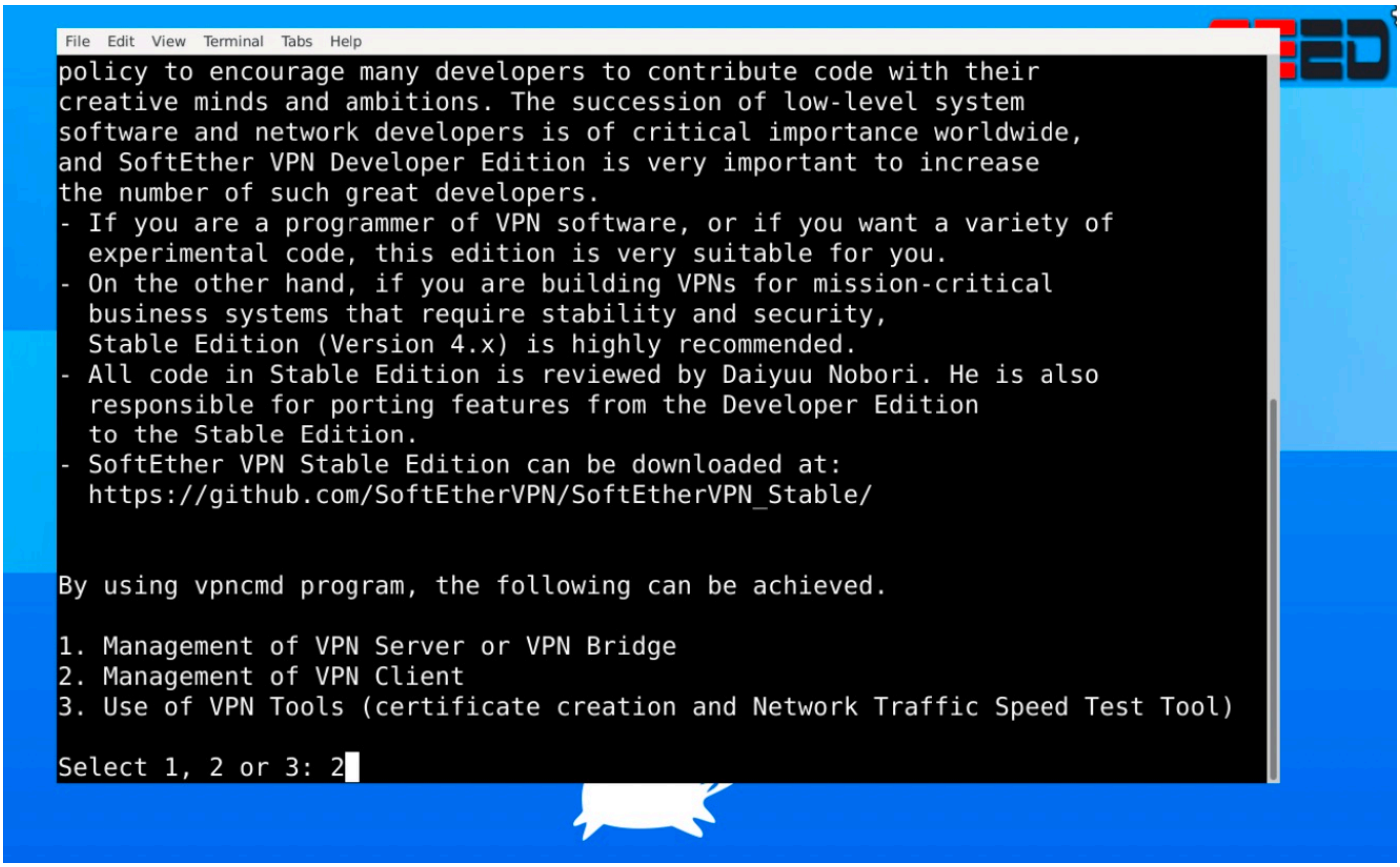
如果要停止客戶端，可以使用以下指令:

```
vpnclient stop
```

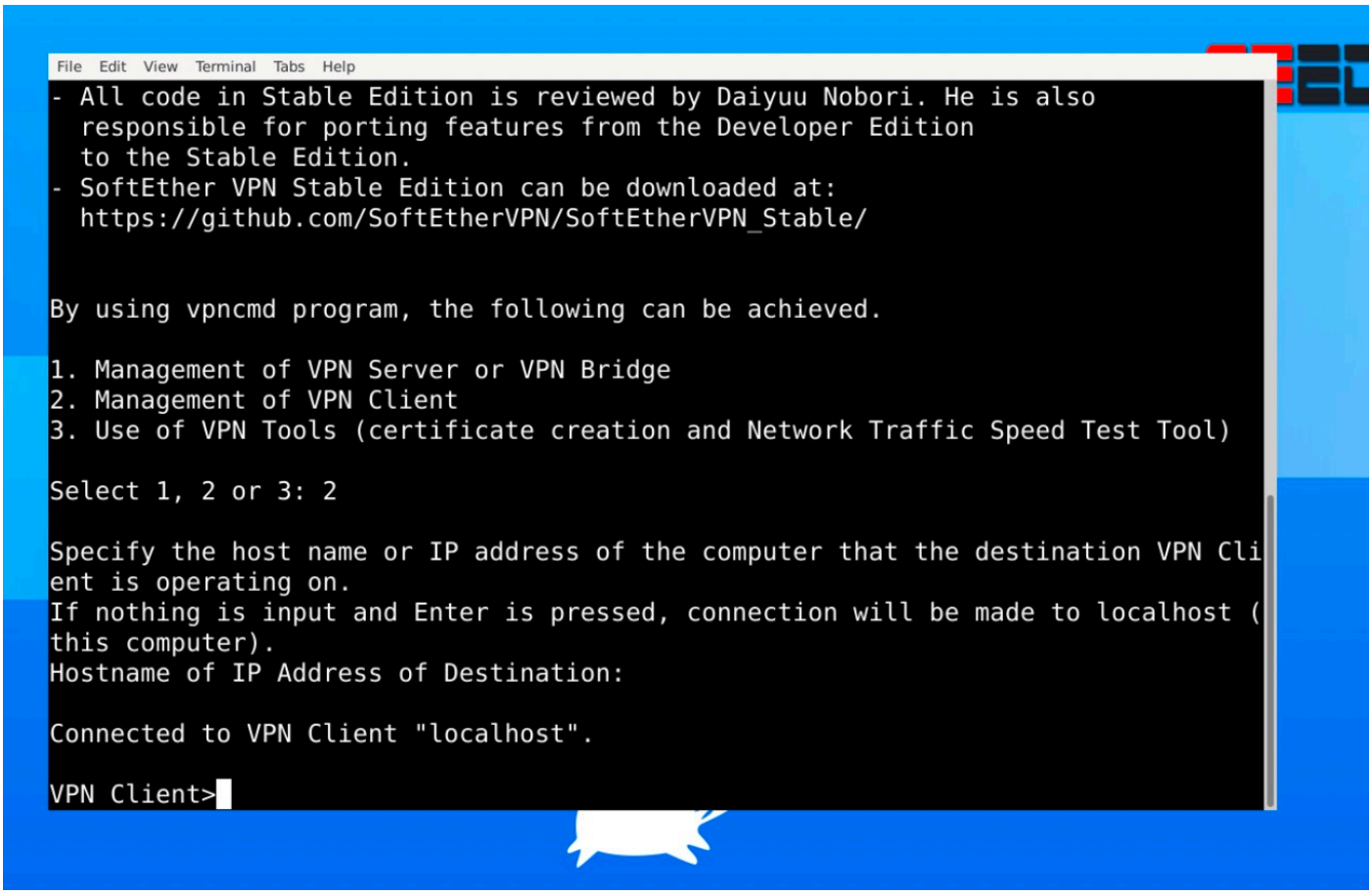
2. VPN 伺服器設定可以使用以下指令:

```
vpncmd
```

3. 可以選擇 2. Management of VPN Client:

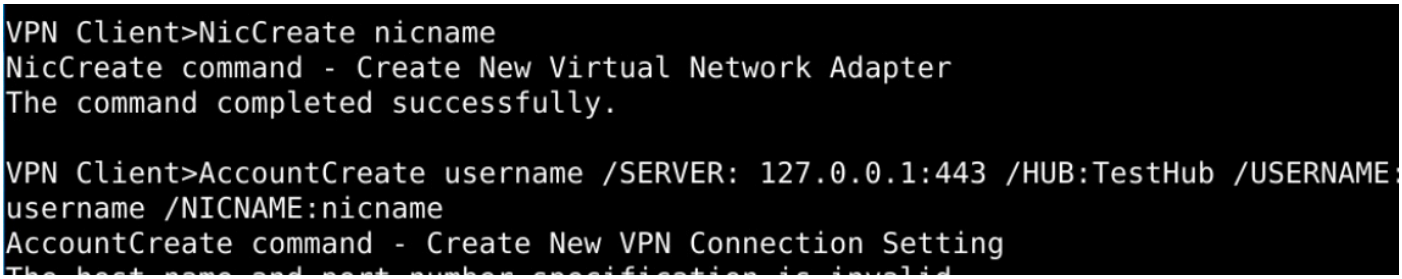


4. 按一次 Enter 鍵，直到出現以下畫面:



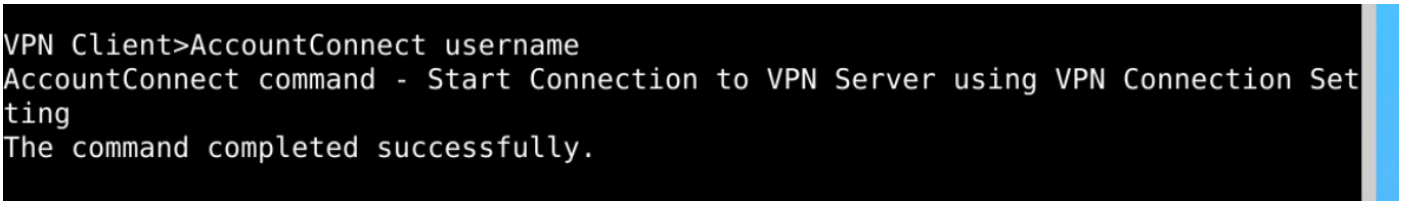
5. 創建一個新的 NIC 並且 創建一個連接設定:

```
NicCreate nicname
AccountCreate username /SERVER:127.0.0.1:443 /HUB:TestHub /USERNAME:username /NICNAME:nicname
```



6. 啟動 VPN 連接:

```
AccountConnect username
```



7. 確認連接狀態:

```
AccountStatusGet username
```

```
VPN Client>AccountStatusGet username
AccountStatusGet command - Get Current VPN Connection Setting Status
Item                                    |Value
----------------------------------------+-----------------------
VPN Connection Setting Name             |username
Session Status                          |Retrying
Connection Started at                   |2024-05-06 (Mon) 03:39:19
First Session has been Established since|-
Number of Established Sessions          |0 Times
The command completed successfully.
```

結論

SoftEther VPN 的功能很多，支援各種高級設定和調整，如果遇到問題可以參考官方文件或社區論壇尋找答案。

參考資料

- SoftEther VPN on GitHub
- SoftEther UNIX Tutorial

3.4 Random Number Generator in Linux Kernel

在 Linux 作業系統中，`/dev/random` 和 `/dev/urandom` 的行為最近的近年進行調整，讓兩者在多數情況下變得幾乎相同，這變更主要是它們如何處理熵（隨機性來源）的細節。

在之前 `/dev/random` 可能會在熵耗盡時阻塞輸出，而 `/dev/urandom` 則不會阻塞，即使熵耗盡也會繼續提供數據，讓 `/dev/urandom` 在多個情況下都常用到，特別是在需要大量隨機數據的情況。

但最近的 Linux kernel 更新中，這兩者行為被設計為更相同。現在不管是 `/dev/random` 或是 `/dev/urandom`，它們都連接到同個內部的隨機數生成器（Linux Random Number Generator, LRNG）並共享相同的輸入熵池。這樣會在大多情況下，不管讀取兩者之一所得到的隨機性應該會是相同的，此外最新的設計也讓它們共享輸出熵池。

這樣主要是為了簡化系統行為，減少開發者在選擇使用 `/dev/random` 還是 `/dev/urandom` 時的困惑，確保在熵充足的情況下，不管選擇哪個 random，程式都能獲得更好的隨機數據，因此現在選擇使用 `/dev/urandom` 是更佳的，就算在安全性要求極高的程式中也是。

額外參考資料

- Unix & Linux Stack Exchange
- Information Security Stack Exchange

3.5 Lab: MD5 Collision Attack Lab

alias md5collgen 指令:

```
alias md5collgen='/path/to/md5collgen'
```

Task 1: Generating Two Different Files with the Same MD5 Hash

用 `md5collgen` 生成兩個不同的文件，但是 MD5 hash 值相同:

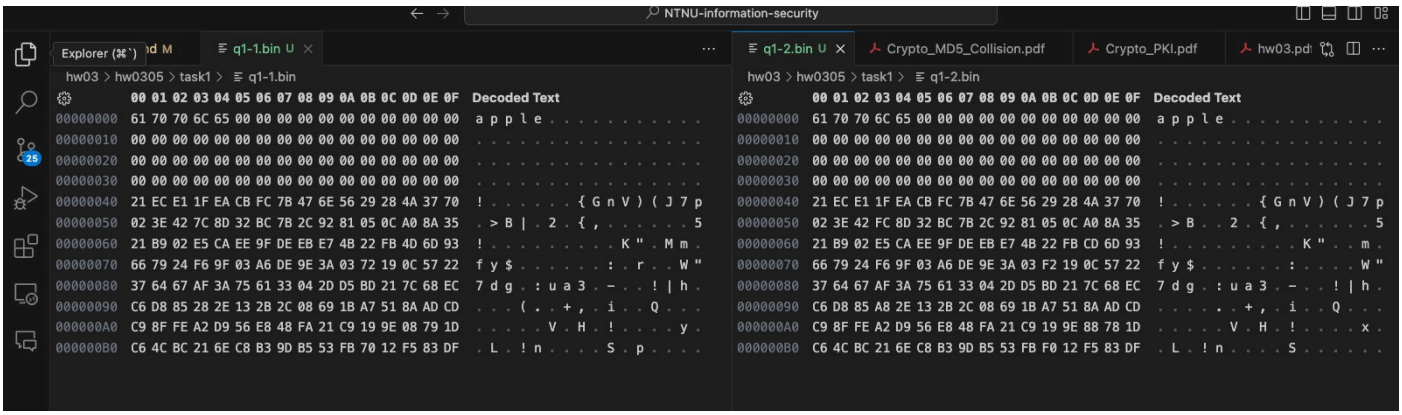
```
md5collgen -p prefix.txt -o out1.bin out2.bin
diff out1.bin out2.bin
md5sum out1.bin; md5sum out2.bin
```

```
root@8e36e1c80b89:/app/hw03/hw0305/task1# md5collgen -p prefix.txt -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'prefix.txt'
Using initial value: 125bb928c696df30dcedcdd6c1a4ba05

Generating first block: ..........................
Generating second block: S10.
Running time: 32.4605 s
root@8e36e1c80b89:/app/hw03/hw0305/task1# diff out1.bin out2.bin
Binary files out1.bin and out2.bin differ
root@8e36e1c80b89:/app/hw03/hw0305/task1# md5sum out1.bin; md5sum out2.bin
b8d0c7702c7a3e63deb568a8777cd0b1  out1.bin
b8d0c7702c7a3e63deb568a8777cd0b1  out2.bin
root@8e36e1c80b89:/app/hw03/hw0305/task1#
```

- Question 1. If the length of your prefix file is not multiple of 64, what is going to happen?
  使用 `truncate` 指令將 `q1.txt` 的長度截斷為 10 bytes，然後使用 `md5collgen` 生成兩個文件:

```
truncate -s 10 q1.txt
md5collgen -p q1.txt -o q1-1.bin q1-2.bin
```

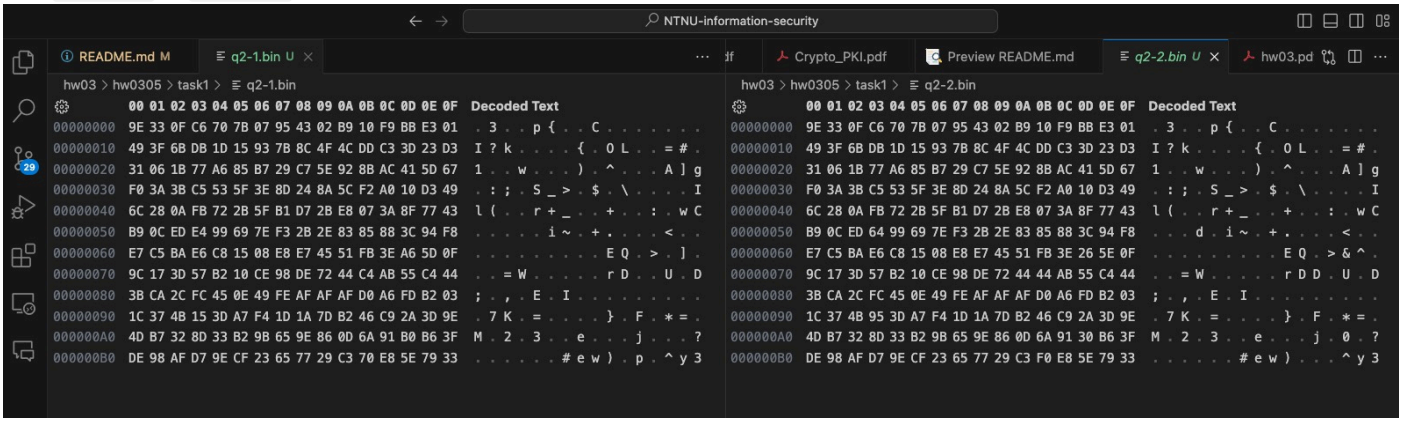
  比對 `q1-1.bin` 和 `q1-2.bin` 的差異，可以看到兩個文件的不足 64 bytes 的部分都以 0x00 填充，因為 MD5 hash 是 64 bytes 的 block hash，所以不足 64 bytes 的部分會被填充為 0x00，結果如下:



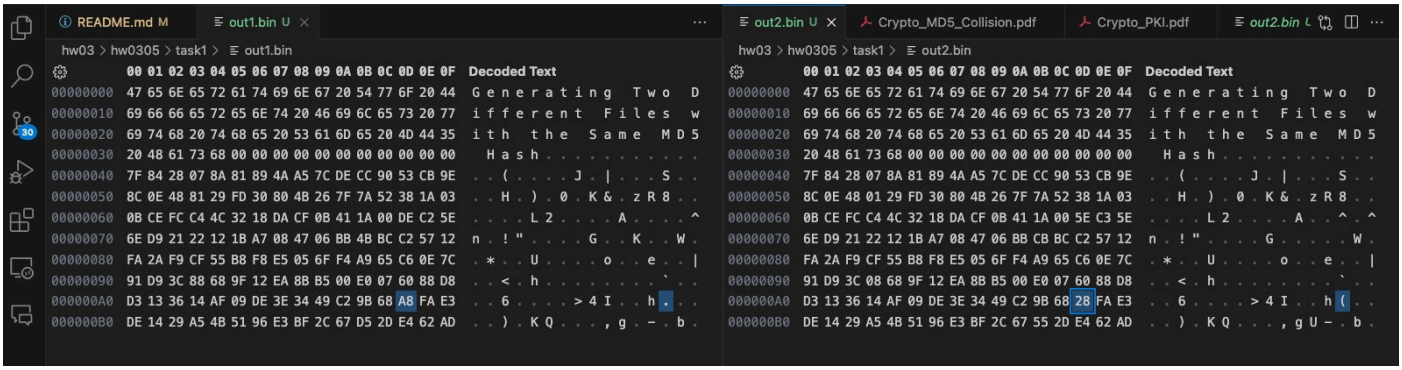
- Question 2. Create a prefix file with exactly 64 bytes, and run the collision tool again, and see what happens.
  使用 `dd` 指令生成 64 bytes 的文件 `q2.txt`，然後使用 `md5collgen` 生成兩個文件:

```
dd if=/dev/urandom of=q2.txt bs=64 count=1
md5collgen -p q2.txt -o q2-1.bin q2-2.bin
```

  比對 `q2-1.bin` 和 `q2-2.bin` 的差異，可以看到兩個文件的 MD5 hash 值相同，結果如下:



- Question 3. Are the data (128 bytes) generated by md5collgen completely different for the two output files? Please identify all the bytes that are different.
  比對 `out1.bin` 和 `out2.bin` 的差異，不是所有 128 bytes 的數據都相同，並且在多次執行 `md5collgen` 後，生成的文件的差異部分也不同，如下標記的部分是不同的:



Task 2: Understanding MD5's Property

先使用 `md5collgen` 生成兩個文件:

```
md5collgen -p file1.txt -o file1-1.bin file1-2.bin
```

然後將以下數據合併到 `file3-1.bin` 和 `file3-2.bin`:

```
cat file1-1.bin file2.txt > file3-1.bin
cat file1-2.bin file2.txt > file3-2.bin
```

比對以下文件的 MD5 hash 值，可以看到 `file1-1.bin` 和 `file1-2.bin` 的 MD5 hash 值相同，而 `file3-1.bin` 和 `file3-2.bin` 的 MD5 hash 值也相同:

```
md5sum file1-1.bin; md5sum file1-2.bin; md5sum file3-1.bin; md5sum file3-2.bin;
```

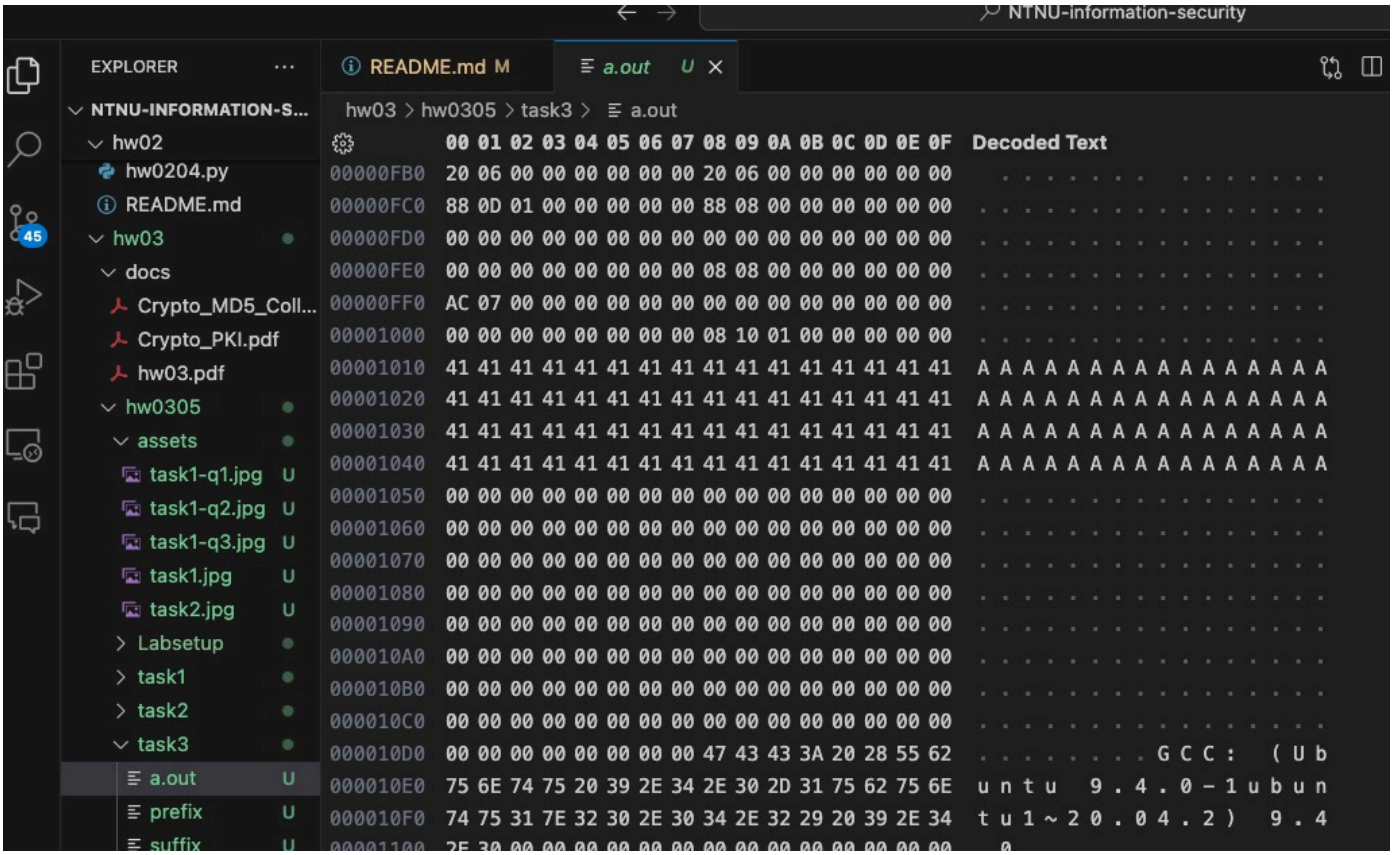
```
root@8e36e1c80b89:/app/hw03/hw0305/task2# md5sum file1-1.bin; md5sum file1-2.bin; md5sum file3-1.bin; md5sum file3-2.
bin;
d9bad79a45e4d97e6cb334e0d9d714f4  file1-1.bin
d9bad79a45e4d97e6cb334e0d9d714f4  file1-2.bin
8fd2ba9989d04d1d7f31109d874ee9d4  file3-1.bin
8fd2ba9989d04d1d7f31109d874ee9d4  file3-2.bin
root@8e36e1c80b89:/app/hw03/hw0305/task2#
```

Task 3: Generating Two Executable Files with the Same MD5 Hash

使用 `gcc` 編譯 `task3.c`:

```
gcc task3.c -o a.out
```

查看 `a.out` 的 bytes，可以看到 prefix 是在 `0x1010` 的位置:



取 `0x1010` 64 (0x40) bytes 的數據 `0x1050`，十進制為 4176，

```
head -c 4176 a.out > prefix.out
md5collgen -p prefix.out -o prefix1.out prefix2.out
```

將 4176 + 128 + 1 bytes 的數據生成到 `suffix.bin`:

```
tail -c +4305 a.out > suffix.out
```

合併以下檔案分別生成 `s1.out` 和 `s2.out`，然後設置執行權限:

```
cat prefix1.out suffix.out > s1.out
cat prefix2.out suffix.out > s2.out
chmod +x s1.out s2.out
```

追加 `suffix.out` 到 `prefix1.out` 和 `prefix2.out`，然後設置執行權限:

```
cat suffix.out >> prefix1.out
cat suffix.out >> prefix2.out
chmod +x prefix1.out prefix2.out
```

執行 `s1.out` 和 `s2.out`:

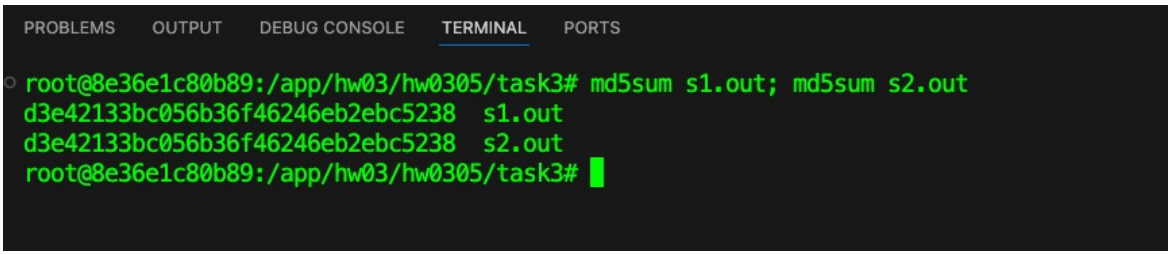
```
./s1.out
./s2.out
```

比對 `s1.out` 和 `s2.out` 的 MD5 hash 值，可以看到兩個文件的 MD5 hash 還是有所不同:

```
echo $(./s1.out) | md5sum;echo $(./s2.out) | md5sum
```

比對 `s1` 和 `s2` 的 MD5 hash 值，可以看到兩個文件的 MD5 hash 值相同:

```
md5sum s1.out; md5sum s2.out
```

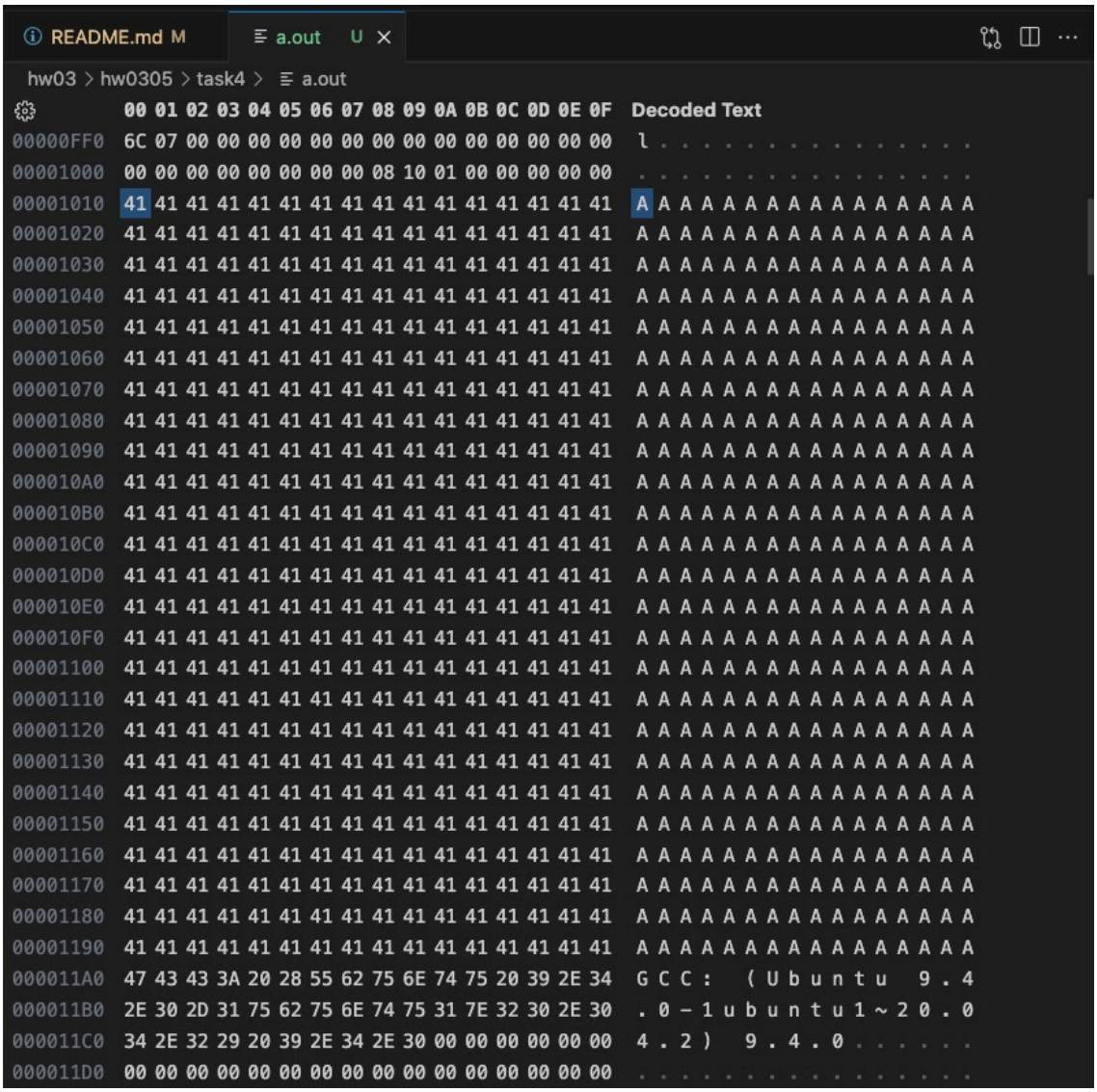


Task 4: Making the Two Programs Behave Differently

使用 `gcc` 編譯 `task4.c`:

```
gcc task4.c -o a.out
```

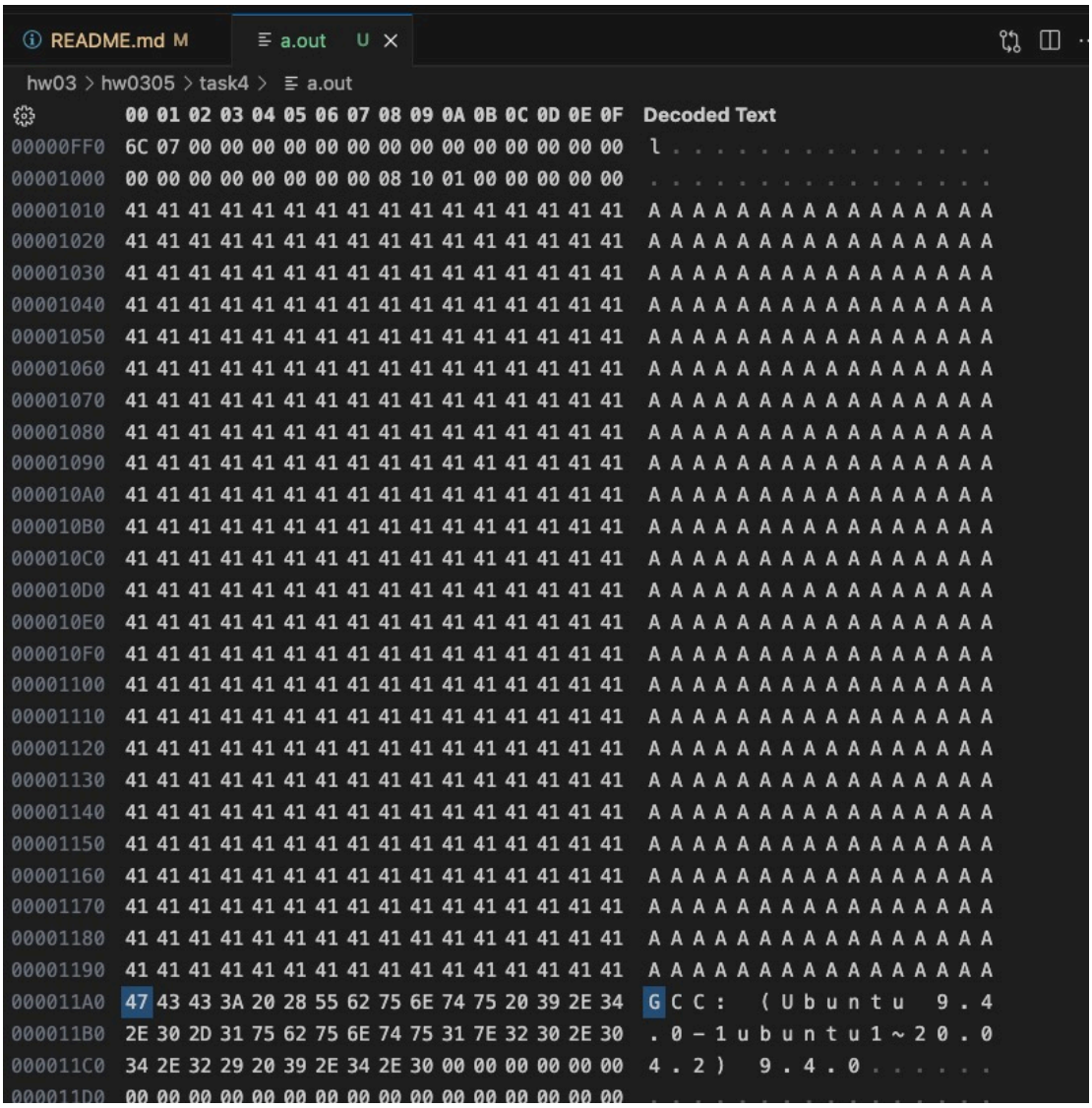
查看 `a.out` 的 bytes 的 prefix，可以看到 A array 是在 `0x1010` 的位置，十進位 4112:



4112 + 48 = 4160 bytes (整除 64) 的數據生成到 `prefix.out`:

```
head -c 4160 a.out > prefix.out
md5collgen -p prefix.out -o prefix1.out prefix2.out
```

查看 `a.out` 的 bytes 的 suffix，可以看到是在 `0x11A0` 的位置，加一十進位後為 4513:



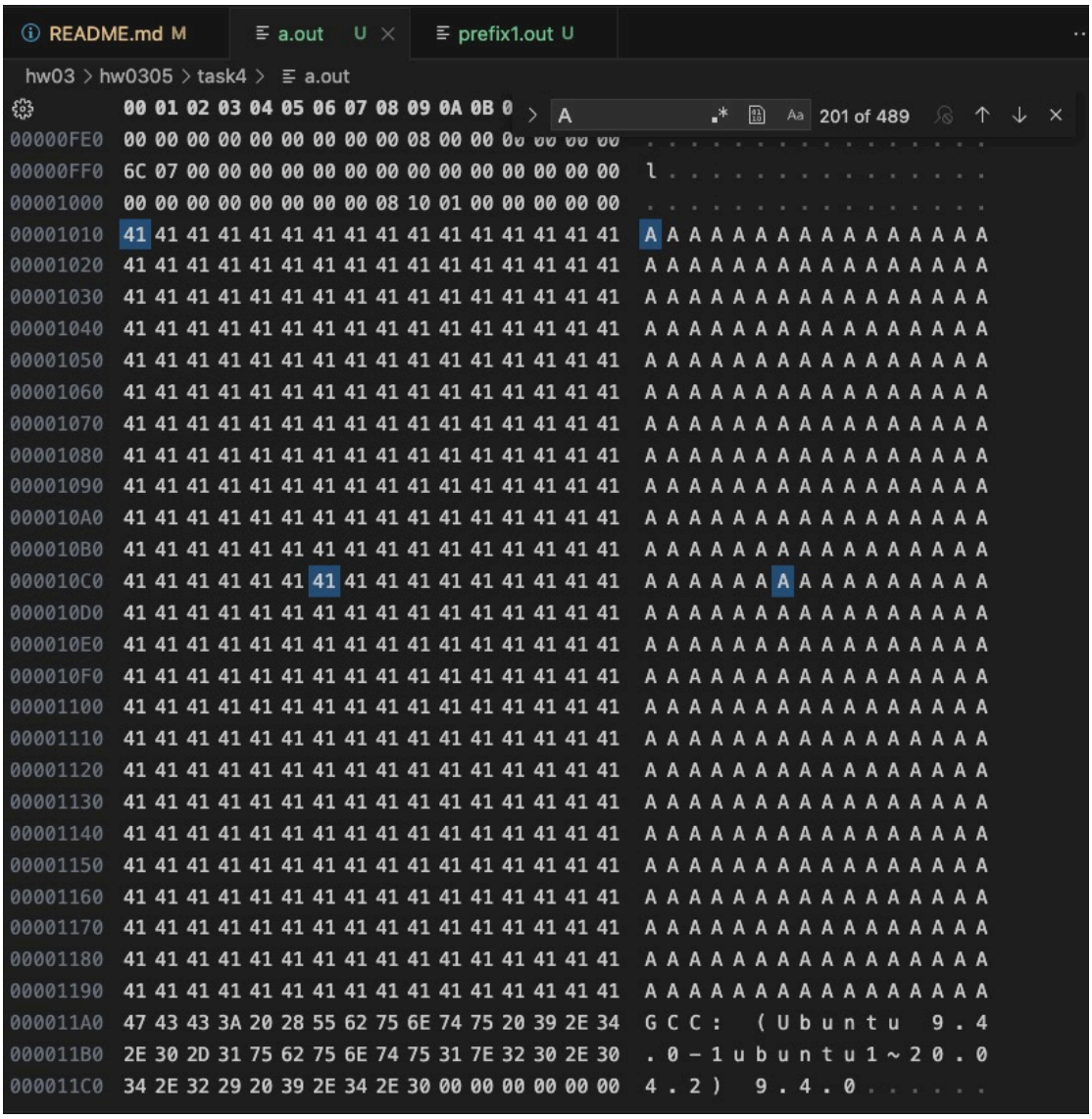
取 `4513` bytes 的數據生成到 `suffix.out`:

```
tail -c +4513 a.out > suffix.out
```

將 48 + 128 = 176 bytes 的數據生成到 `middle.out`:

```
tail -c 176 prefix1.out > middle.out
```

X array 是在 `0x1010` 的位置，Y array 是在 `0x10C0` 的位置，:



0x10C0 - 0x1010 = 176 bytes 的數據，離 200 差 24 bytes，並且 200 - 176 = 24 bytes 的數據生成到 `m24.bin`

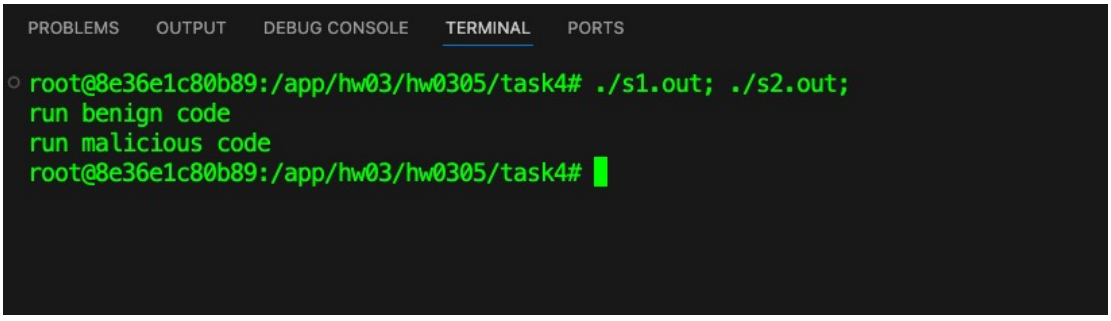
```
python3 -c "print('\x00'*40)" > tmp
head -c 24 tmp > m24.bin
rm tmp
```

合併以下檔案分別生成 `s1.out` 和 `s2.out`，然後設置執行權限:

```
cat prefix1.out m24.bin middle.out m24.bin suffix.out > s1.out
cat prefix2.out m24.bin middle.out m24.bin suffix.out > s2.out
chmod +x s1.out s2.out
```

查看 `s1.out` 和 `s2.out` 的運行結果是不同的:

```
./s1.out; ./s2.out;
```



但是比對 `s1.out` 和 `s2.out` 的 MD5 hash 值，可以看到兩個文件的 MD5 hash 值相同:

```
md5sum s1.out; md5sum s2.out;
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

root@8e36e1c80b89:/app/hw03/hw0305/task4# md5sum s1.out; md5sum s2.out;
cfb4dbf6fbab7d5f7442f4b18eb1953c  s1.out
cfb4dbf6fbab7d5f7442f4b18eb1953c  s2.out
root@8e36e1c80b89:/app/hw03/hw0305/task4#
root@8e36e1c80b89:/app/hw03/hw0305/task4#
```

3.6 Lab: PKI Lab

將 Container 啟動:

```
cd LabSetup
dcoker-compose build
docker-compose up
```

Task 1: Becoming a Certificate Authority (CA)

DNS 設定增加 /etc/hosts 文件內容:

```
10.9.0.80 www.bank32.com
10.9.0.80 www.smith2020.com
```

創建一個新的 CA 並生成 CA 的私鑰和證書:

```
mkdir demoCA
mkdir demoCA/certsare
mkdir demoCA/crl
mkdir demoCA/newcerts
touch demoCA/index.txt
echo 1000 > demoCA/serial
cp /usr/lib/ssl/openssl.cnf .
openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -keyout ca.key -out ca.crt
```

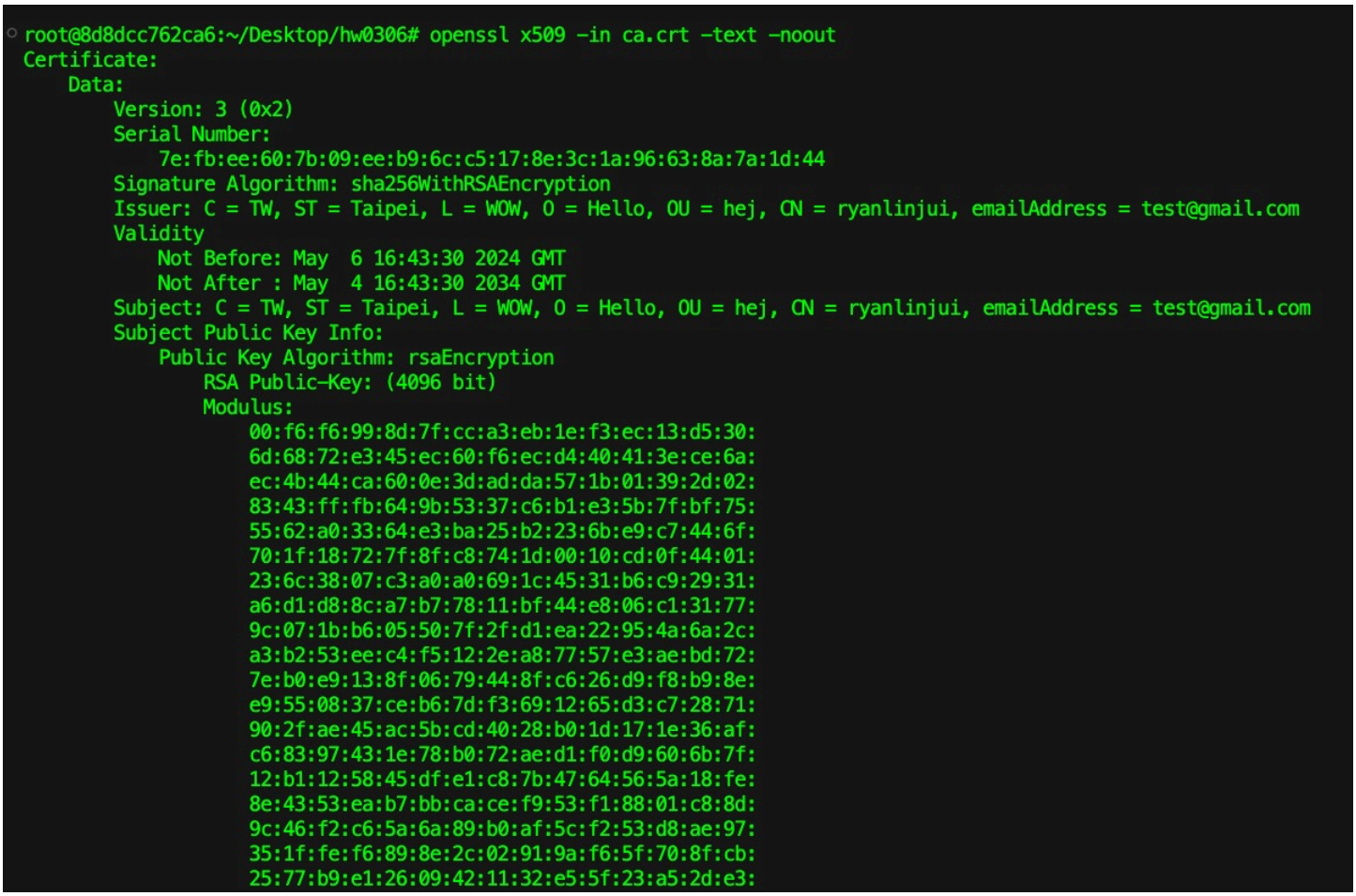
你可以依照以下圖片的方式填寫資訊(密碼: `0000`):

```
root@8d8dcc762ca6:~/Desktop/hw0306# openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -keyout ca.key -out ca.crt
Generating a RSA private key
..............++++
.......................++++
writing new private key to 'ca.key'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:TW
State or Province Name (full name) [Some-State]:Taipei
Locality Name (eg, city) []:WOW
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Hello
Organizational Unit Name (eg, section) []:hej
Common Name (e.g. server FQDN or YOUR name) []:ryanlinjui
Email Address []:test@gmail.com
root@8d8dcc762ca6:~/Desktop/hw0306#
```

查看 CA 的證書:

```
openssl x509 -in ca.crt -text -noout
```

示意圖如下:



查看 CA 的私鑰:

```
openssl rsa -in ca.key -text -noout
```

示意圖如下(密碼: `0000`):

```
root@8d8dcc762ca6:~/Desktop/hw0306# openssl rsa -in ca.key -text -noout
Enter pass phrase for ca.key:
RSA Private-Key: (4096 bit, 2 primes)
modulus:
    00:f6:f6:99:8d:7f:cc:a3:eb:1e:f3:ec:13:d5:30:
    6d:68:72:e3:45:ec:60:f6:ec:d4:40:41:3e:ce:6a:
    ec:4b:44:ca:60:0e:3d:ad:da:57:1b:01:39:2d:02:
    83:43:ff:fb:64:9b:53:37:c6:b1:e3:5b:7f:bf:75:
    55:62:a0:33:64:e3:ba:25:b2:23:6b:e9:c7:44:6f:
    70:1f:18:72:7f:8f:c8:74:1d:00:10:cd:0f:44:01:
    23:6c:38:07:c3:a0:a0:69:1c:45:31:b6:c9:29:31:
    a6:d1:d8:8c:a7:b7:78:11:bf:44:e8:06:c1:31:77:
    9c:07:1b:b6:05:50:7f:2f:d1:ea:22:95:4a:6a:2c:
    a3:b2:53:ee:c4:f5:12:2e:a8:77:57:e3:ae:bd:72:
    7e:b0:e9:13:8f:06:79:44:8f:c6:26:d9:f8:b9:8e:
    e9:55:08:37:ce:b6:7d:f3:69:12:65:d3:c7:28:71:
    90:2f:ae:45:ac:5b:cd:40:28:b0:1d:17:1e:36:af:
    c6:83:97:43:1e:78:b0:72:ae:d1:f0:d9:60:6b:7f:
    12:b1:12:58:45:df:e1:c8:7b:47:64:56:5a:18:fe:
    8e:43:53:ea:b7:bb:ca:ce:f9:53:f1:88:01:c8:8d:
    9c:46:f2:c6:5a:6a:89:b0:af:5c:f2:53:d8:ae:97:
    35:1f:fe:f6:89:8e:2c:02:91:9a:f6:5f:70:8f:cb:
    25:77:b9:e1:26:09:42:11:32:e5:5f:23:a5:2d:e3:
    0c:fe:0f:59:3a:0e:3f:2f:f0:1d:79:25:3d:57:90:
    bb:1a:bb:0d:1f:46:ac:8b:27:94:9b:4e:ea:ff:eb:
    d8:f2:d0:e3:5d:2a:95:29:44:e0:54:68:50:36:3d:
    c2:84:60:2b:0c:45:7f:0c:00:97:21:d8:a2:22:6b:
    4d:97:5a:d3:e5:c0:0f:d1:bc:c3:69:e8:19:fe:f7:
    cd:36:b9:8e:f7:c0:47:c8:9f:c5:05:2a:9b:6a:27:
    ab:c7:cc:89:aa:d0:c6:72:a5:e8:a3:a4:ef:90:77:
    3e:c5:e1:6a:dc:1f:dd:99:d4:2f:e8:1d:a2:de:e6:
    d3:88:d8:7b:f4:26:c5:9c:4c:54:53:cd:c2:5e:71:
    89:c5:bf:4a:1f:0c:21:ad:2a:f2:a1:d9:b3:e1:1a:
    f5:fb:16:2f:fe:e1:be:d5:01:1d:95:b5:be:a4:6b:
    f6:7f:20:7e:ed:14:9a:b5:72:2e:cb:c6:9b:01:e1:
```

Task 2: Generating a Certificate Request for Your Web Server

生成一個新的私鑰和證書請求，並且在證書請求中添加主機名稱:

```
openssl req -newkey rsa:2048 -sha256 -keyout server.key -out server.csr -subj "/CN=www.bank32.com/O=Bank32 Inc./C=US" -passout
pass:dees -addext "subjectAltName = DNS:www.bank32.com, DNS:www.bank32A.com, DNS:www.bank32B.com"
```

查看證書請求:

```
openssl req -in server.csr -text -noout
```

示意圖如下:

```
root@8d8dcc762ca6:~/Desktop/hw0306# openssl req -in server.csr -text -noout
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: CN = www.bank32.com, O = Bank32 Inc., C = US
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:e2:72:7a:4b:d7:44:40:6f:7e:cf:69:4c:b1:2d:
                    63:ce:5c:6d:46:fe:58:63:ab:2f:cb:be:44:ae:6c:
                    5c:50:4b:89:7b:2a:eb:a6:88:c3:8c:d0:61:68:3d:
                    0b:59:16:df:38:d5:00:f3:29:56:fd:03:87:5f:cd:
                    1f:09:01:a1:2b:54:d7:fb:50:3b:6e:1b:8e:56:dc:
                    ff:62:13:45:2a:d0:e2:a6:b9:ff:7b:3c:fd:8f:f5:
                    1e:05:4d:30:92:5a:7f:39:6e:4b:bd:c5:cd:98:13:
                    23:9f:c7:35:e0:f3:9a:68:13:72:dd:c5:2e:f7:12:
                    6e:d9:e7:b5:f6:1e:4a:0a:98:79:c3:a5:bc:6d:b6:
                    62:5c:89:15:db:01:73:0d:b7:44:04:60:c1:8b:37:
                    ec:92:65:5a:c3:cb:7a:f1:04:75:75:67:1c:7e:be:
                    43:9f:62:f0:c5:20:7d:84:09:d2:97:4c:94:f2:15:
                    62:37:6f:4f:c9:ad:3b:cd:a2:ee:ec:97:5c:38:0b:
                    41:ec:67:fc:6b:a7:03:7c:d9:4c:a9:75:13:09:d5:
                    5f:9c:32:0a:03:bf:41:09:76:b5:0e:16:76:0c:bb:
                    5a:31:e5:86:5d:30:c3:bf:0f:63:7c:28:2f:dc:91:
                    4c:fe:f9:1a:45:d6:60:1b:6a:77:31:cb:a4:cf:10:
                    23:65
                Exponent: 65537 (0x10001)
        Attributes:
        Requested Extensions:
            X509v3 Subject Alternative Name:
                DNS:www.bank32.com, DNS:www.bank32A.com, DNS:www.bank32B.com
    Signature Algorithm: sha256WithRSAEncryption
         c6:7d:2a:8c:d8:af:bb:ec:62:d0:b3:63:69:57:b6:76:b3:4d:
         5e:f0:d2:95:bb:8c:90:6d:71:70:e2:8b:1a:89:46:51:36:a4:
```

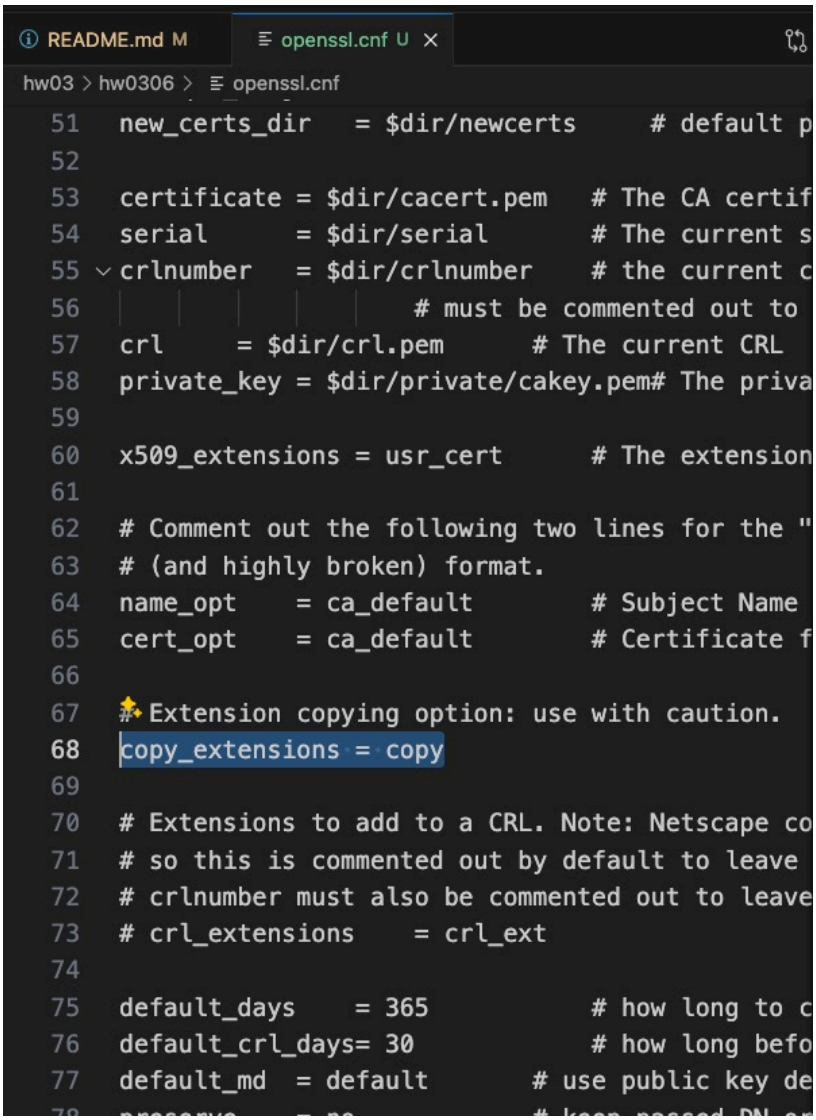
Task 3: Generating a Certificate for your server

生成一個新的證書，並且使用 CA 的私鑰簽名:

```
openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days 3650 -in server.csr -out server.crt -batch -cert ca.crt -keyfile ca.key
```

```
root@8d8dcc762ca6:~/Desktop/hw0306# openssl ca -config openssl.cnf -policy policy_anything -md sha256 -days 3650 -in
server.csr -out server.crt -batch -cert ca.crt -keyfile ca.key
Using configuration from openssl.cnf
Enter pass phrase for ca.key:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4096 (0x1000)
        Validity
            Not Before: May  6 17:01:08 2024 GMT
            Not After : May  4 17:01:08 2034 GMT
        Subject:
            countryName               = US
            organizationName          = Bank32 Inc.
            commonName                = www.bank32.com
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                C1:9B:D3:9D:36:9A:29:03:A1:45:41:2A:A9:9E:FD:DA:02:09:F4:88
            X509v3 Authority Key Identifier:
                keyid:52:80:64:A8:24:66:2A:45:A9:23:BF:98:43:7A:5F:C0:D4:03:68:A6

Certificate is to be certified until May  4 17:01:08 2034 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
root@8d8dcc762ca6:~/Desktop/hw0306#
```

把證書的 `copy_extensions` 註解拿掉:



查看證書:

```
openssl x509 -in server.crt -text -noout
```

示意圖如下:

```
root@8d8dcc762ca6:~/Desktop/hw0306# openssl x509 -in server.crt -text -noout
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 4096 (0x1000)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = TW, ST = Taipei, L = WOW, O = Hello, OU = hej, CN = ryanlinjui, emailAddress = test@gmail.com
        Validity
            Not Before: May  6 17:01:08 2024 GMT
            Not After : May  4 17:01:08 2034 GMT
        Subject: C = US, O = Bank32 Inc., CN = www.bank32.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:e2:72:7a:4b:d7:44:40:6f:7e:cf:69:4c:b1:2d:
                    63:ce:5c:6d:46:fe:58:63:ab:2f:cb:be:44:ae:6c:
                    5c:50:4b:89:7b:2a:eb:a6:88:c3:8c:d0:61:68:3d:
                    0b:59:16:df:38:d5:00:f3:29:56:fd:03:87:5f:cd:
                    1f:09:01:a1:2b:54:d7:fb:50:3b:6e:1b:8e:56:dc:
                    ff:62:13:45:2a:d0:e2:a6:b9:ff:7b:3c:fd:8f:f5:
                    1e:05:4d:30:92:5a:7f:39:6e:4b:bd:c5:cd:98:13:
                    23:9f:c7:35:e0:f3:9a:68:13:72:dd:c5:2e:f7:12:
                    6e:d9:e7:b5:f6:1e:4a:0a:98:79:c3:a5:bc:6d:b6:
                    62:5c:89:15:db:01:73:0d:b7:44:04:60:c1:8b:37:
                    ec:92:65:5a:c3:cb:7a:f1:04:75:75:67:1c:7e:be:
                    43:9f:62:f0:c5:20:7d:84:09:d2:97:4c:94:f2:15:
                    62:37:6f:4f:c9:ad:3b:cd:a2:ee:ec:97:5c:38:0b:
                    41:ec:67:fc:6b:a7:03:7c:d9:4c:a9:75:13:09:d5:
                    5f:9c:32:0a:03:bf:41:09:76:b5:0e:16:76:0c:bb:
                    5a:31:e5:86:5d:30:c3:bf:0f:63:7c:28:2f:dc:91:
                    4c:fe:f9:1a:45:d6:60:1b:6a:77:31:cb:a4:cf:10:
                    23:65
                Exponent: 65537 (0x10001)
        X509v3 extensions:
```

Task 4: Deploying Certificate in an Apache-Based HTTPS Website

查看 Container 的 ID:

```
docker ps -a
```

進入 Container:

```
docker exec -it <container_id> /bin/bash
```

進入 Container 後，啟動 Apache 服務:

```
service apache2 start
```

Task 5: Launching a Man-In-The-Middle Attack

Task 6: Launching a Man-In-The-Middle Attack with a Compromised CA