Assignment 3.7 Lab: MD5

National Taiwan Normal University CSIE Information Security

60947038S

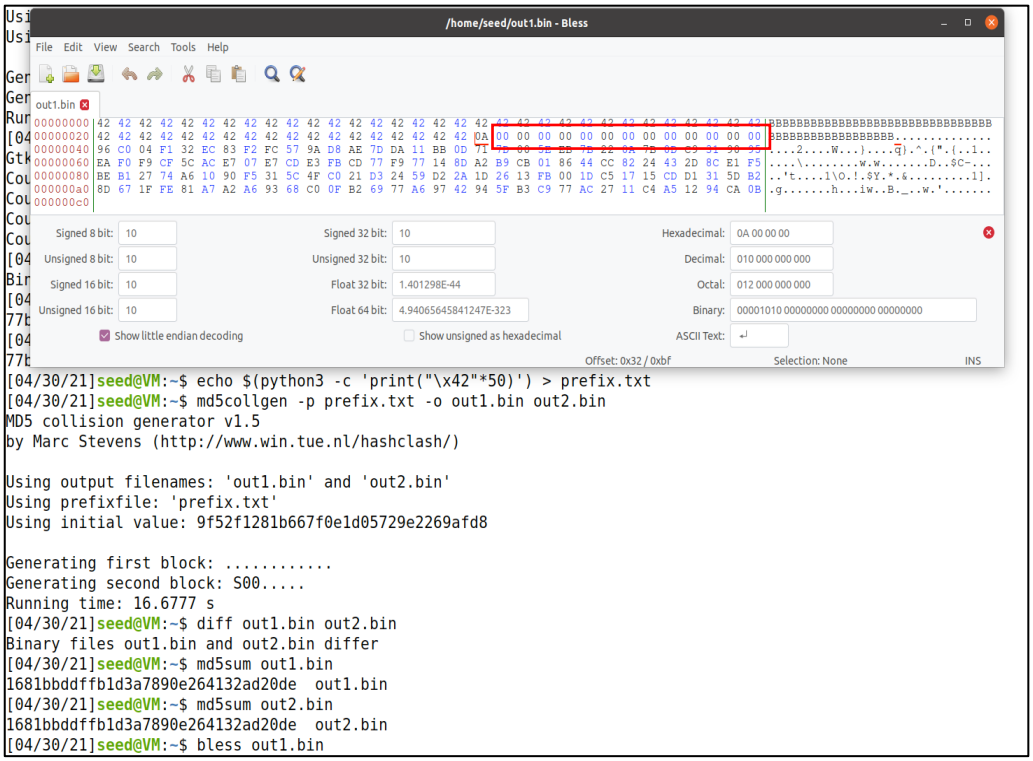
Task 1: Generating Two Different Files with the Same MD5 Hash

I. 首先利用 md5collgen tool 建立兩個不同的 file(out1.bin , out2.bin)，但有著相同的 prefix.txt，和相同的 MD5 hash value，使用 commands: md5collgen -p prefix.txt -o out1.bin out2.bin

II. 為了確認是否生成兩個不同的檔案確有著相同的 MD5 hash value，使用 commands : diff <binfile,binfile>來檢查，透過 md5sum <binfile>來查看 hash value。

III. 不同的 prefix text 長度設定會影響他 padding 的樣子，我們使用 python3 來將不同個數的 B 加到 prefix.txt 生成 out1.bin 和 out2.bin，使用 commands: *echo* $(python3 -c 'print("/x42"*NUMBERS)') > prefix.txt 並用 hex editor(bless)來查看 out1.bin 和 out2.bin 或使用 commands xxd <filename>。

1. 將 50 個 B 加入 prefix.txt 中，用 bless 查看少於 64 bytes 會發生什麼事：可以看到它自動幫我們 padding 0x00 到 64bytes



2. 將 64 個 B 加入 prefix.txt 中，用 bless 查看 prefix text 恰好是 64 bytes 會發生什麼事：沒有任何 padding。

```
[04/30/21]seed@VM:~$ echo $(python3 -c 'print("\x42"*63)') > prefix_64
[04/30/21]seed@VM:~$ md5collgen  -p prefix_64 -o out1_64.bin out2_64.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1_64.bin' and 'out2_64.bin'
Using prefixfile: 'prefix_64'
Using initial value: 1a411e7066d64ae6b60256017d130eda

Generating first block: ..
Generating second block: W..............................
Running time: 3.95876 s
[04/30/21]seed@VM:~$ xxd out1_64.bin
00000000: 4242 4242 4242 4242 4242 4242 4242 4242  BBBBBBBBBBBBBBBB
00000010: 4242 4242 4242 4242 4242 4242 4242 4242  BBBBBBBBBBBBBBBB
00000020: 4242 4242 4242 4242 4242 4242 4242 4242  BBBBBBBBBBBBBBBB
00000030: 4242 4242 4242 4242 4242 4242 4242 420a  BBBBBBBBBBBBBBB.
00000040: 8f32 dac8 f8aa f756 ad51 7e73 ec88 5a3c  .2.....V.Q~s..Z<
00000050: f3c6 061d 4cf9 269b 30be 7951 92cb 3802  ....L.&.0.yQ..8.
00000060: 79f5 01c1 d0a5 9f89 cbfb 4032 174e be6a  y.........@2.N.j
00000070: dabf d82b ba51 7d4a 4108 6668 8f29 c694  ...+.Q}JA.fh.)..
00000080: f280 18bc b917 6323 03bf cb34 97ce 61a4  ......c#...4..a.
00000090: 4f9d 5aa8 480f 81ec 182d 9b76 ca2a fece  O.Z.H....-.v.*..
000000a0: fd8f b7d5 da6f 3688 f656 093a 981c 961f  .....o6..V.:....
000000b0: db74 c87f 7ac4 b33b cf5d 40dd 4df9 6f05  .t..z..;.]@.M.o.
[04/30/21]seed@VM:~$
```

3. 將 128 個 C 加入 prefix_128 中，並用 prefix_128 來生成 md5collgen 的兩個 output: out1_128.bin 和 out2_128.bin，padding 結果會和 64 的整數倍相同，不會有任何 padding，但實際上兩個檔案不同在哪裡？利用 xxd 將 binfile 丟進 txt，使用 diff 來辨別實際上差異的地方。

```
[04/30/21]seed@VM:~$ echo $(python3 -c 'print("\x43"*127)') > prefix_128
[04/30/21]seed@VM:~$ cat prefix_128
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
[04/30/21]seed@VM:~$ md5collgen -p prefix_128 -o out1_128.bin out2_128.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1_128.bin' and 'out2_128.bin'
Using prefixfile: 'prefix_128'
Using initial value: 0dca8edb6993206e503fcbb81e110dac

Generating first block: ...
Generating second block: S10.....
Running time: 2.73359 s
[04/30/21]seed@VM:~$ xxd out1_128.bin > o.txt
[04/30/21]seed@VM:~$ xxd out2_128.bin > p.txt
[04/30/21]seed@VM:~$ diff o.txt p.txt
10,12c10,12
< 00000090: 2859 2b60 db01 fb00 35da 800d 0925 3cdd  (Y+`....5....%<.
< 000000a0: 66d1 b3ea 5ec5 1b92 eb1b d4bc 1c7c 016c  f...^........|.l
< 000000b0: 818a a7df eb44 b7a2 be6c 4e68 8929 a2a0  .....D...lNh.)..
---
> 00000090: 2859 2be0 db01 fb00 35da 800d 0925 3cdd  (Y+.....5....%<.
> 000000a0: 66d1 b3ea 5ec5 1b92 eb1b d4bc 1cfc 016c  f...^..........l
> 000000b0: 818a a7df eb44 b7a2 be6c 4ee8 8929 a2a0  .....D...lN..)..
14,16c14,16
< 000000d0: 2fa8 c677 601b 9321 a9b3 1050 1b60 48d8  /..w`..!...P.`H.
< 000000e0: d755 5858 9f87 81fc 3b44 cc5a 5938 0e61  .UXX....;D.ZY8.a
< 000000f0: ac06 8f24 c93d 4124 f061 947d 98ca 217c  ...$.=A$.a.}..!|
---
> 000000d0: 2fa8 c6f7 601b 9321 a9b3 1050 1b60 48d8  /...`..!...P.`H.
> 000000e0: d755 5858 9f87 81fc 3b44 cc5a 59b8 0d61  .UXX....;D.ZY..a
> 000000f0: ac06 8f24 c93d 4124 f061 94fd 98ca 217c  ...$.=A$.a....!|
[04/30/21]seed@VM:~$
```

Task 2: Understanding MD5’s Property

```
[04/30/21]seed@VM:~$ echo "ChengPoSheng" > prefix_t2
[04/30/21]seed@VM:~$ md5collgen -p prefix_t2 -o t2_1 t2_2
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 't2_1' and 't2_2'
Using prefixfile: 'prefix_t2'
Using initial value: 350afad547aef02777454b8a43b91311

Generating first block: ..
Generating second block: W..
Running time: 4.94848 s
[04/30/21]seed@VM:~$ md5sum t2_1
8895134948b0f2488724af291649a1ae  t2_1
[04/30/21]seed@VM:~$ md5sum t2_2
8895134948b0f2488724af291649a1ae  t2_2
[04/30/21]seed@VM:~$ diff t2_1 t2_2
Binary files t2_1 and t2_2 differ
[04/30/21]seed@VM:~$ echo "60947038S" > suffix_t2
[04/30/21]seed@VM:~$ cat t2_1 suffix_t2 > t2_1_done
[04/30/21]seed@VM:~$ cat t2_2 suffix_t2 > t2_2_done
[04/30/21]seed@VM:~$ md5 t2_1_done

Command 'md5' not found, did you mean:

  command 'mdl' from snap mdl (0.11.0)
  command 'cd5' from deb cd5 (0.1-4)
  command 'mdp' from deb mdp (1.0.15-1)
  command 'mdu' from deb mtools (4.0.24-1)

See 'snap info <snapname>' for additional versions.

[04/30/21]seed@VM:~$ md5sum t2_1_done
107a30cda23b731846d83715b9503fcf  t2_1_done
[04/30/21]seed@VM:~$ md5sum t2_2_done
107a30cda23b731846d83715b9503fcf  t2_2_done
[04/30/21]seed@VM:~$ diff t2_1_done t2_2_done
Binary files t2_1_done and t2_2_done differ
[04/30/21]seed@VM:~$
```

分別創建一個 prefix_t2(ChengPoSheng)和 suffix_t2(60947038S)，md5collgen 利用 prefix_t2 創建 t2_1 和 t2_2，測試兩個是不同的檔案確有相同的 md5 hash value，分別實驗將 t2_1 和 suffix_t2 串再一起與將 t2_2 和 suffix_t2 串再一起，產生的 t2_1_done 和 t2_2_done 經過檢測是不同檔案但卻帶有相同的 md5 hash value，所以此 property 在 md5 上也成立。

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

建立一個 task3.c，我們要將 xyz char array 存放 200 個 D，並執行此 task3.c，開啟 bless task3 查看結果以及 200 個 D 所在的 offset。

```c
#include <stdio.h>
unsigned char xyz[200] = {
0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x4
/* The actual contents of this array are up to you */
};
int main()
{
int i;
for (i=0; i<200; i++){
printf("%x", xyz[i]);
}
printf("\n");
}
```

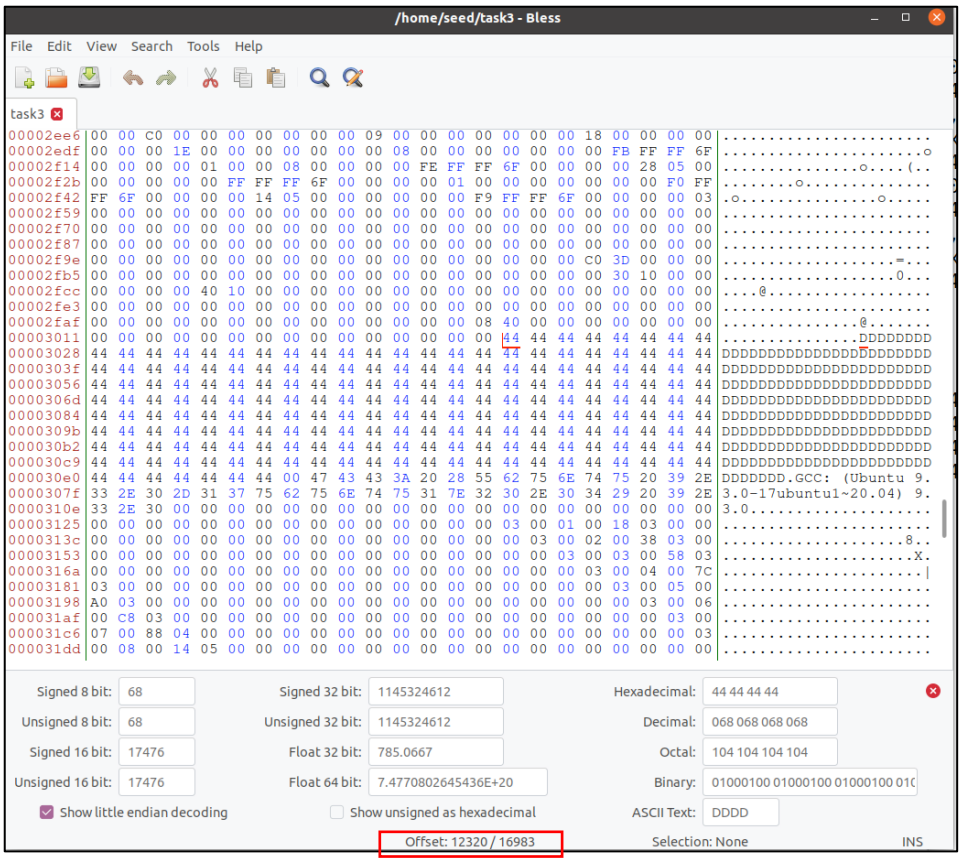
```
[04/30/21]seed@VM:~$ echo $(python3 -c 'print("0x44,"*199)') > D.txt
[04/30/21]seed@VM:~$ cat D.txt
0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x
44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44
,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0
x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x4
4,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,
0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x
44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44
,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0
x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x4
4,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,
0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,
[04/30/21]seed@VM:~$ gedit task3.c
[04/30/21]seed@VM:~$ gcc -o task3 task3.c
[04/30/21]seed@VM:~$ ./task3
44444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444
44444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444
44444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444
44444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444
4444444444444444444444444444440
[04/30/21]seed@VM:~$ bless task3
```

I. 從 bless 上可以看到第一個 D 所在的 offset 為 12320，根據題目所示，因為 prefix 長度必須是 64 的整數倍，經過計算 12320 mod 64 = 32，把 12320 +32=12352 湊出 prefix_t3，又 128 個 D 之後是 suffix 所以 suffix 的 offset 為 12352+128=12480。

II. 當我們完成 prefix_t3 和 suffix_t3 後，我們將用 prefix_t3 來生成 task3_1.bin 和 ask3_2.bin，生成之後我們各別取他們尾端的 128 bytes 來當作我們的 p 和 q

III. 這時只要將 prefix_t3 p/q suffix_t3 串再一起得到的兩個檔案就是我們所要的，驗證他的 md5 hash value 以及透過 chmod +x <file> 來執行他們，實際確認他們的差別。

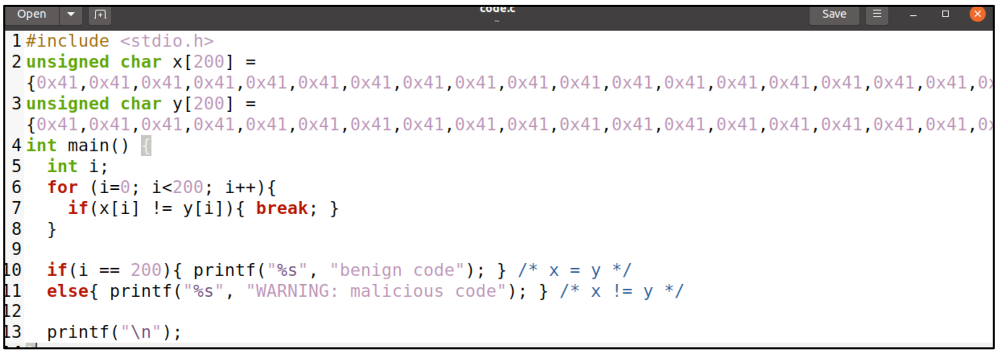
Problem:如果我們在第一步所做的切割沒做好可能會發生在第三步執行的結果會發生 Segmentation fault。

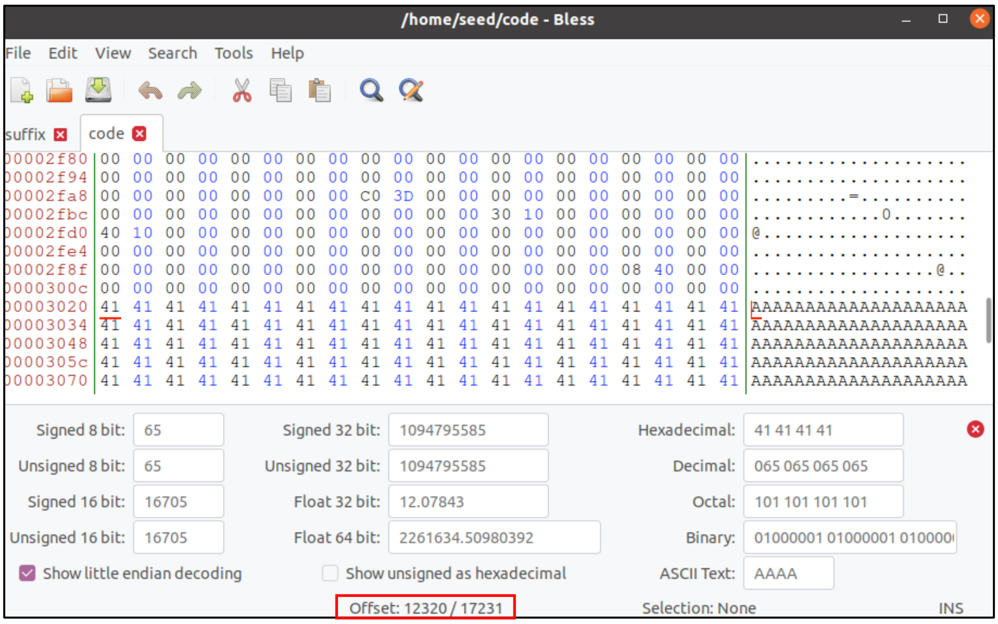


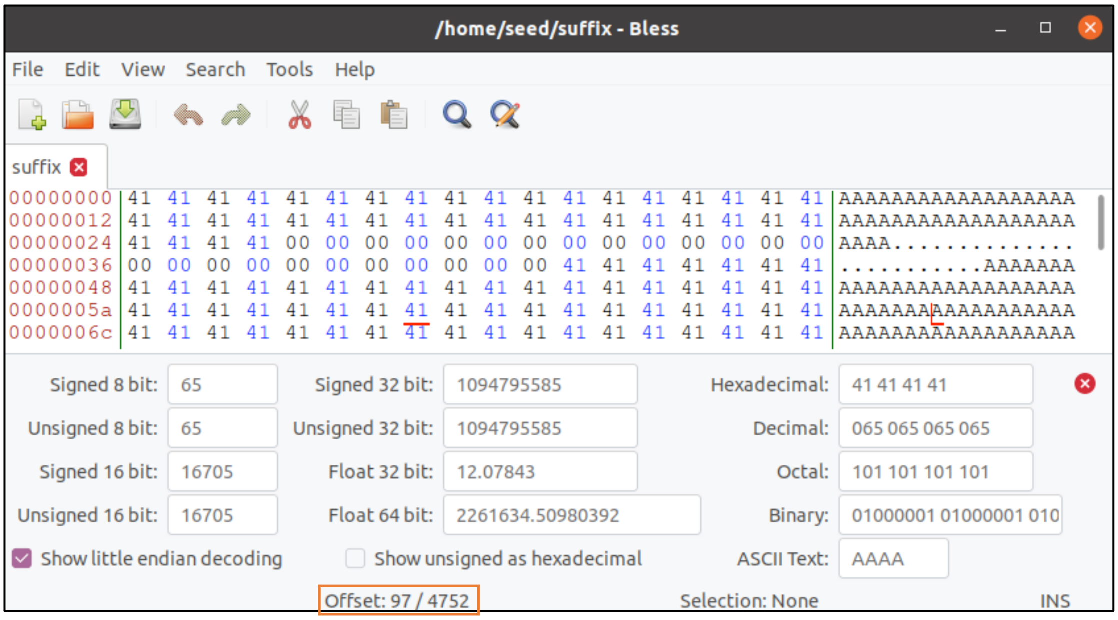

```
[04/30/21]seed@VM:~$ head -c 12352 task3 > prefix_t3
[04/30/21]seed@VM:~$ tail -c +12480 task3 > suffix_t3
[04/30/21]seed@VM:~$ md5collgen -p prefix_t3 -o task3_1.bin task3_2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'task3_1.bin' and 'task3_2.bin'
Using prefixfile: 'prefix_t3'
Using initial value: 1b9ce8f65b339765c5158f11cc5e8be2

Generating first block: ................
Generating second block: W...
Running time: 15.2787 s
[04/30/21]seed@VM:~$ tail -c 128 task3_1.bin > p
[04/30/21]seed@VM:~$ tail -c 128 task3_2.bin > q
[04/30/21]seed@VM:~$ cat prefix_t3 p suffix_t3 > t3_1_done
[04/30/21]seed@VM:~$ cat prefix_t3 q suffix_t3 > t3_2_done
[04/30/21]seed@VM:~$ md5sum t3_1_done
ca789cd0ce72113e79f59e08fd170a2f  t3_1_done
[04/30/21]seed@VM:~$ md5sum t3_2_done
ca789cd0ce72113e79f59e08fd170a2f  t3_2_done
[04/30/21]seed@VM:~$ chmod +x t3_1_done
[04/30/21]seed@VM:~$ ./t3_1_done
44444444444444444444444444444444444444444444444444444444444444446751437312397f3045f63665f14a
f18b9d9f48c8522b69f7d979b21214a9214115f7c4ee233260fe0d22121729cd4f767261b4f796c604be5d6ebc3a
5375e2b92de96661278fc8d36a214cdd0e93f35946c5e7d4d14c926d7ca983d8ec5473abed718d274289d9718b48
6a3e16755f19db5835f7f58ab62ef2d36ed4444444444444444444444444444444444444444444444444444444444444444
444444444444444444444
[04/30/21]seed@VM:~$ chmod +x t3_2_done
[04/30/21]seed@VM:~$ ./t3_2_done
44444444444444444444444444444444444444444444444444444444444444446751437312397f3045f63665f14a
f18b9d9f4c8522b69f7d979b21214a9214115f7c4ee233260fe0d22121f29cd4f767261b4f796c604be5deebc3a5
375e2b92de96661278fc8d36a214cdd0e93f359c6c5e7d4d14c926d7ca983d8ec5473abed718d274289d9718bc86
93e16755f19db5835f7f58abe2ef2d36ed4444444444444444444444444444444444444444444444444444444444444444
444444444444444444444
[04/30/21]seed@VM:~$
```

Task 4: Making the Two Programs Behave Differently

I. 建立一個 code.c，裡面做判斷，如果 Array X 和 Y 完全一樣時，執行的是 benign code 否則為 malicious code。



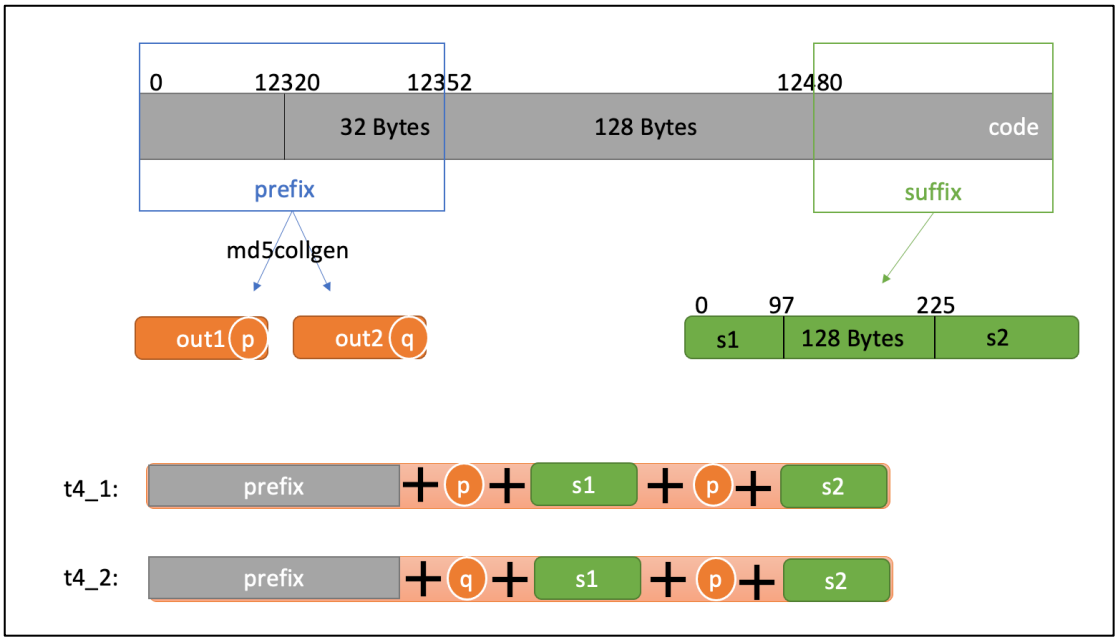


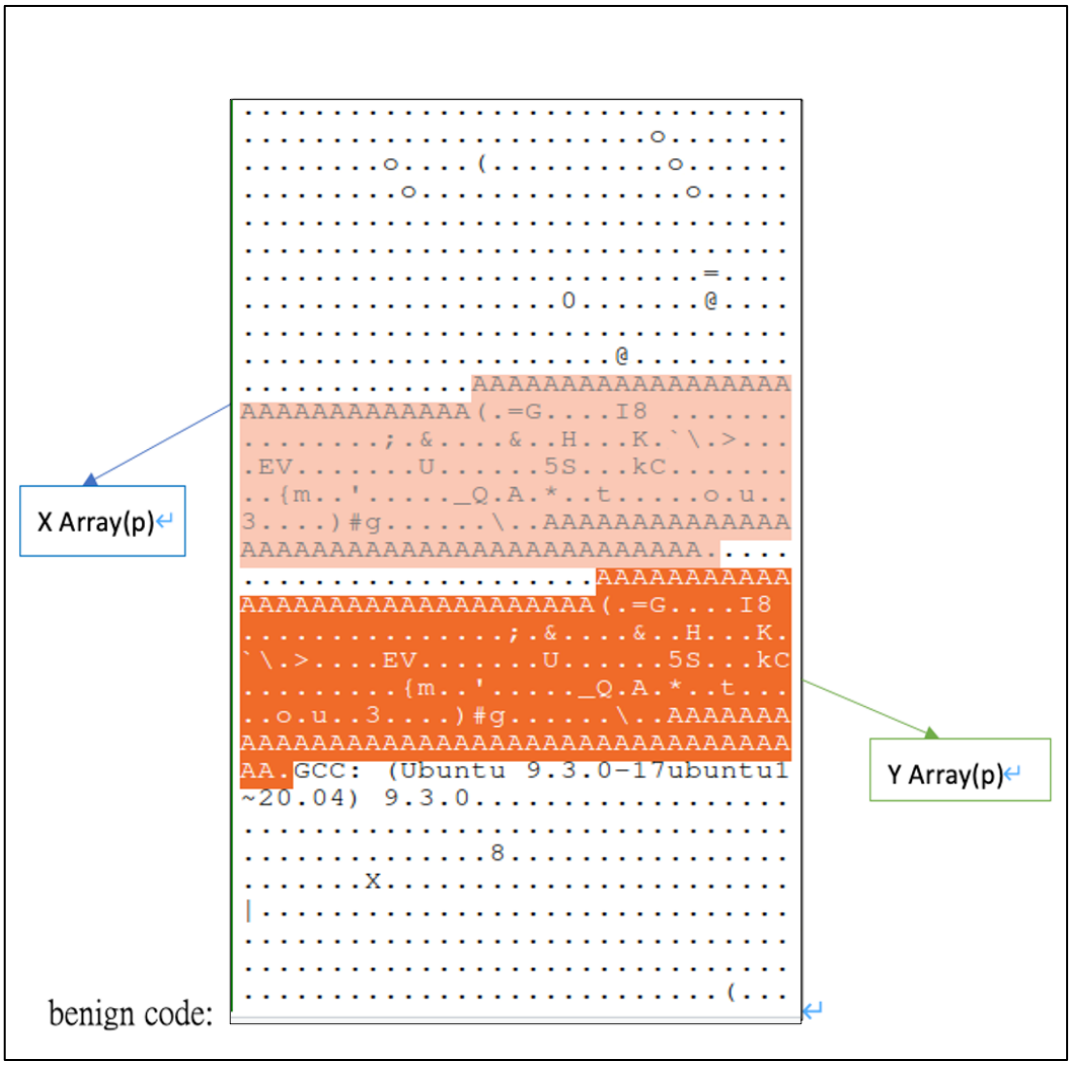


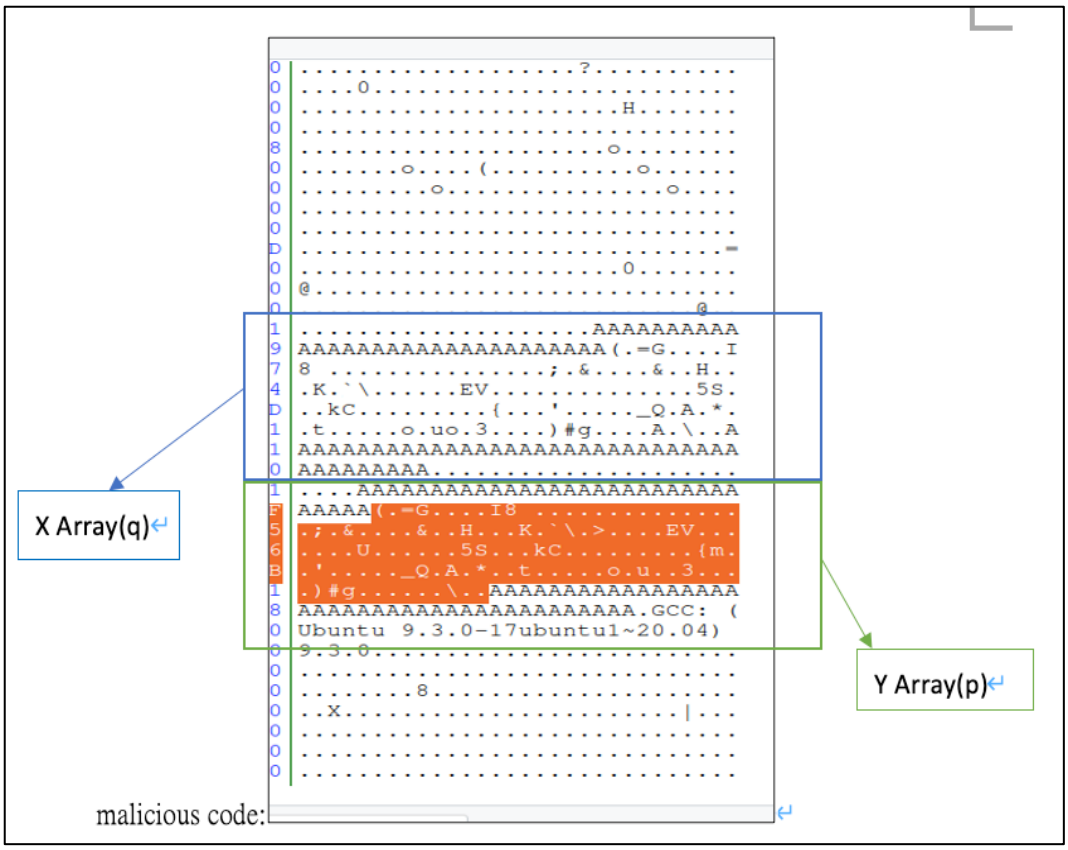
II. 目標是切成這樣：

| 12352 | p | 64 | p | 4512 | > task4_1

| 12352 | q | 64 | p | 4512 | > task4_2








```
[05/01/21]seed@VM:~$ head -c 12351 code > prefix
[05/01/21]seed@VM:~$ tail -c +12480 code > suffix
[05/01/21]seed@VM:~$ md5collgen -p prefix -o out1 out2
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1' and 'out2'
Using prefixfile: 'prefix'
Using initial value: 4dcfaeb9b0a401ded87aaf3261645348

Generating first block: .
Generating second block: S11.................
Running time: 2.45297 s
[05/01/21]seed@VM:~$ tail -c 128 out1 > p
[05/01/21]seed@VM:~$ tail -c 128 out2 > q
[05/01/21]seed@VM:~$ head -c 97 suffix > s1
[05/01/21]seed@VM:~$ tail -c +225 suffix > s2
[05/01/21]seed@VM:~$ cat prefix p s1 p s2 > t4_1
[05/01/21]seed@VM:~$ cat prefix q s1 p s2 > t4_2
[05/01/21]seed@VM:~$ chmod +x t4_1
[05/01/21]seed@VM:~$ chmod +x t4_2
[05/01/21]seed@VM:~$ ./t4_1
benign code
[05/01/21]seed@VM:~$ ./t4_2
WARNING: malicious code
[05/01/21]seed@VM:~$
```