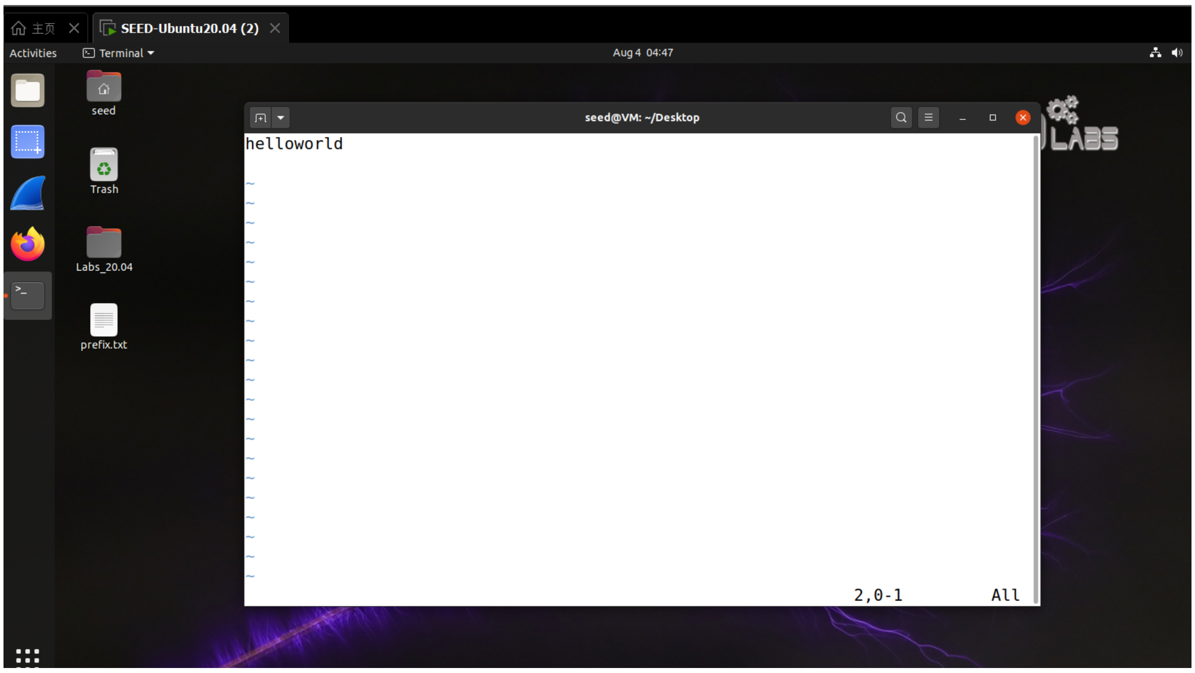
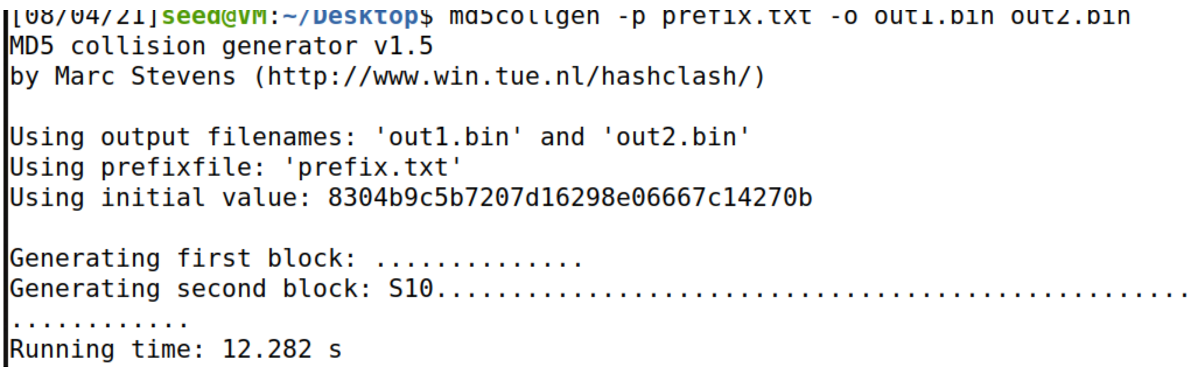
TASK 1

新建文件prefix.txt并修改文件内容为helloworld；



```
helloworld
~
~
~
                                          2,0-1         All
```

生成加密文件out1.bin out2.bin

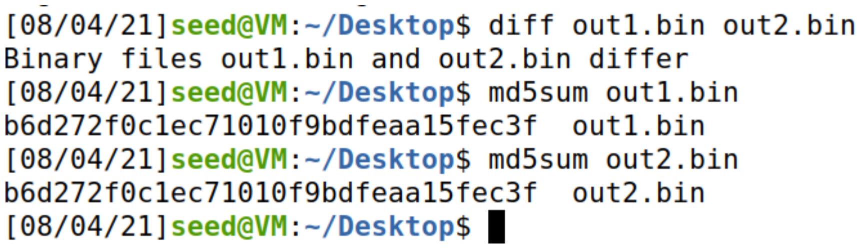


```
[08/04/21]seed@VM:~/Desktop$ md5collgen -p prefix.txt -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'prefix.txt'
Using initial value: 8304b9c5b7207d16298e06667c14270b

Generating first block: ..............
Generating second block: S10.................................................
............
Running time: 12.282 s
```

比较不同



```
[08/04/21]seed@VM:~/Desktop$ diff out1.bin out2.bin
Binary files out1.bin and out2.bin differ
[08/04/21]seed@VM:~/Desktop$ md5sum out1.bin
b6d272f0c1ec71010f9bdfeaa15fec3f  out1.bin
[08/04/21]seed@VM:~/Desktop$ md5sum out2.bin
b6d272f0c1ec71010f9bdfeaa15fec3f  out2.bin
[08/04/21]seed@VM:~/Desktop$ 
```

> **Question 1** If the length of your prefix file is not multiple of 64, what is going to happen?

补零至 64 Byte。

> **Question 2** Create a prefix file with exactly 64 bytes, and run the collision tool again, and see what happens.

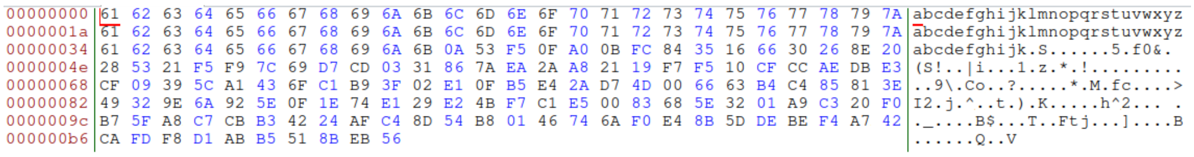
将 prefix.txt 改为

```
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijk
```

这里一共 63 个字母，加上文件结束符 `0A` 正好 64 Byte。

```
md5collgen -p prefix.txt -o out1.bin out2.bin
```

生成文件后如下所示



```
00000000 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A abcdefghijklmnopqrstuvwxyz
0000001a 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A abcdefghijklmnopqrstuvwxyz
00000034 61 62 63 64 65 66 67 68 69 6A 6B 0A 53 F5 0F A0 0B FC 84 35 16 66 30 26 8E 20 abcdefghijk.S......5.f0&.
0000004e 28 53 21 F5 F9 7C 69 D7 CD 03 31 86 7A EA 2A A8 21 19 F7 F5 10 CF CC AE DB E3 (S!..|i...1.z.*.!.........
00000068 CF 09 39 5C A1 43 6F C1 B9 3F 02 E1 0F B5 E4 2A D7 4D 00 66 63 B4 C4 85 81 3E ..9\.Co..?.....*.M.fc....>
00000082 49 32 9E 6A 92 5E 0F 1E 74 E1 29 E2 4B F7 C1 E5 00 83 68 5E 32 01 A9 C3 20 F0 I2.j.^..t.).K.....h^2... .
0000009c B7 5F A8 C7 CB B3 42 24 AF C4 8D 54 B8 01 46 74 6A F0 E4 8B 5D DE BE F4 A7 42 ._.....B$...T..Ftj...]....B
000000b6 CA FD F8 D1 AB B5 51 8B EB 56                                                 ......Q..V
```

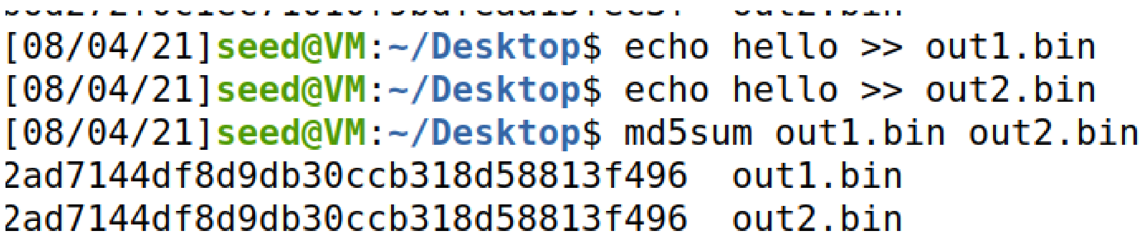
可以看到没有补零了。

> **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.

例如第一个例子中，有 3 个 Byte 不同。经过多次尝试发现，这些不同的数量和位置不固定。

task2

对刚刚的两个 md5 相同的文件分别加上一个后缀，然后查看它们的 md5



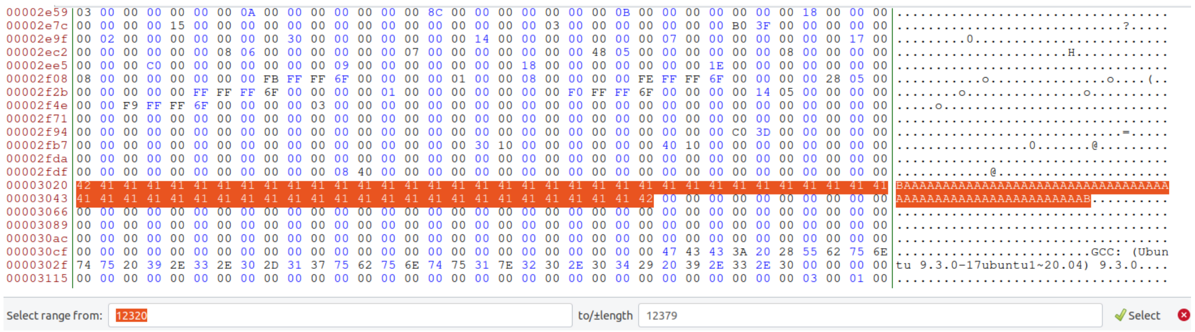
```
[08/04/21]seed@VM:~/Desktop$ echo hello >> out1.bin
[08/04/21]seed@VM:~/Desktop$ echo hello >> out2.bin
[08/04/21]seed@VM:~/Desktop$ md5sum out1.bin out2.bin
2ad7144df8d9db30ccb318d58813f496  out1.bin
2ad7144df8d9db30ccb318d58813f496  out2.bin
```

task3

新建 pro.c

```
#include <stdio.h>
unsigned char xyz[200] = {
    'B', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'B'
};
int main()
{
    int i;
    for (i=0; i<200; i++){
        printf("%x", xyz[i]);
    }
    printf("\n");
}
```

编译,定位到刚刚的字符串存储在 `0x3020` 位置



```
00002e59 03 00 00 00 00 00 00 0A 00 00 00 00 00 00 00 8C 00 00 00 00 00 00 00 0B 00 00 00 00 00 00 00 18 00 00 00 ...................................
00002e7c 00 00 00 00 15 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 00 00 00 00 00 00 00 B0 3F 00 00 00 00 00 .............................?.....
00002e9f 00 02 00 00 00 00 00 00 00 30 00 00 00 00 00 00 00 14 00 00 00 00 00 00 00 07 00 00 00 00 00 00 00 17 00 .........0.........................
00002ec2 00 00 00 00 00 00 08 06 00 00 00 00 00 00 07 00 00 00 00 00 00 00 48 05 00 00 00 00 00 00 08 00 00 00 00 ......................H............
00002ee5 00 00 00 C0 00 00 00 00 00 00 00 09 00 00 00 00 00 00 00 18 00 00 00 00 00 00 00 1E 00 00 00 00 00 00 00 ...................................
00002f08 08 00 00 00 00 00 00 00 FB FF FF 6F 00 00 00 00 01 00 00 08 00 00 00 00 FE FF FF 6F 00 00 00 00 28 05 00 ...........o..............o.....(..
00002f2b 00 00 00 00 00 FF FF FF 6F 00 00 00 00 01 00 00 00 00 00 00 00 F0 FF FF 6F 00 00 00 00 14 05 00 00 00 00 ........o................o.........
00002f4e 00 00 F9 FF FF 6F 00 00 00 00 03 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....o.............................
00002f71 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
00002f94 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 C0 3D 00 00 00 00 00 .............................=.....
00002fb7 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 30 10 00 00 00 00 00 00 40 10 00 00 00 00 00 00 00 00 .................0.......@.........
00002fda 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
00002fdf 00 00 00 00 00 00 00 00 00 00 00 08 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ............@......................
00003020 42 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 BAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
00003043 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 42 00 00 00 00 00 00 00 00 00 00 AAAAAAAAAAAAAAAAAAAAAAAAB..........
00003066 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
00003089 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
000030ac 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
000030cf 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 47 43 43 3A 20 28 55 62 75 6E .........................GCC: (Ubun
0000302f 74 75 20 39 2E 33 2E 30 2D 31 37 75 62 75 6E 74 75 31 7E 32 30 2E 30 34 29 20 39 2E 33 2E 30 00 00 00 00 tu 9.3.0-17ubuntu1~20.04) 9.3.0....
00003115 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 00 01 00 ...................................
```

Select range from: 12320 to/±length 12379

我们不妨截取到 12340 位置

```
head -c 12340 pro > prefix
```

计算得到在 12320 到 12379 范围内，12352 为 64 的倍数，因此我们把12352 后面的截取出来

```
tail -c +12353 pro > suffix
```

然后对 prefix 生成 md5 相同的两个文件

```
md5collgen -p prefix -o prefix1 prefix2
```

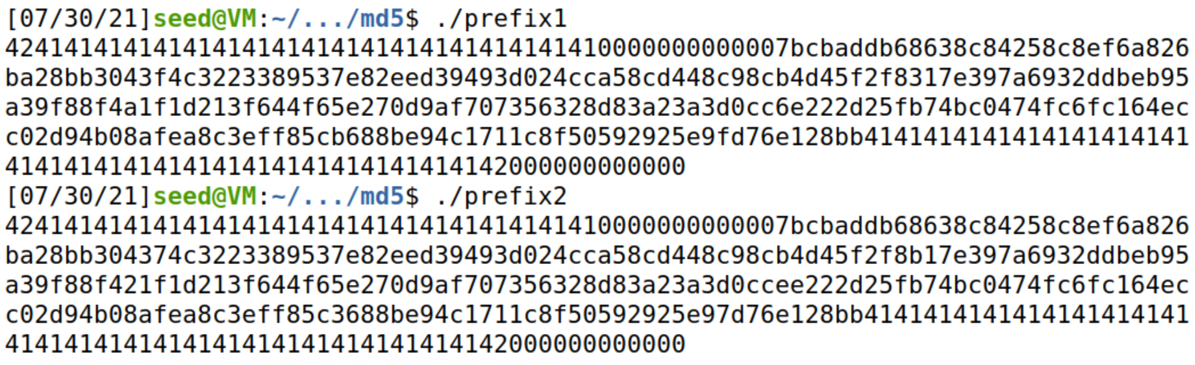
把刚刚的尾巴接到这两个文件后面

```
cat suffix >> prefix1
cat suffix >> prefix2
```

赋予执行权限

```
chmod +x prefix1
chmod +x prefix1
```

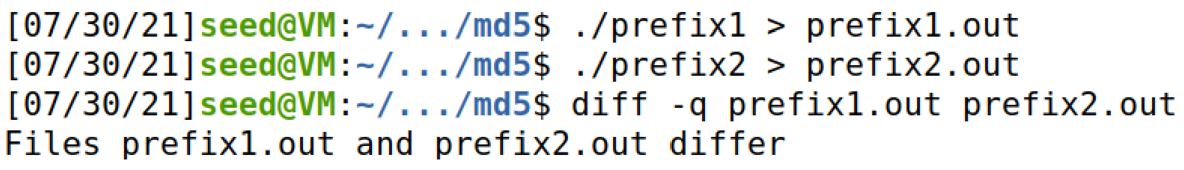
运行



```
[07/30/21]seed@VM:~/.../md5$ ./prefix1
4241414141414141414141414141414141414141414141410000000000007bcbaddb68638c84258c8ef6a826
ba28bb3043f4c3223389537e82eed39493d024cca58cd448c98cb4d45f2f8317e397a6932ddbeb95
a39f88f4a1f1d213f644f65e270d9af707356328d83a23a3d0cc6e222d25fb74bc0474fc6fc164ec
c02d94b08afea8c3eff85cb688be94c1711c8f50592925e9fd76e128bb4141414141414141414141
41414141414141414141414141414141414200000000000000
[07/30/21]seed@VM:~/.../md5$ ./prefix2
4241414141414141414141414141414141414141414141410000000000007bcbaddb68638c84258c8ef6a826
ba28bb304374c3223389537e82eed39493d024cca58cd448c98cb4d45f2f8b17e397a6932ddbeb95
a39f88f421f1d213f644f65e270d9af707356328d83a23a3d0ccee222d25fb74bc0474fc6fc164ec
c02d94b08afea8c3eff85c3688be94c1711c8f50592925e97d76e128bb4141414141414141414141
41414141414141414141414141414141414200000000000000
```

可以看到，两个输出是不同的

```
./prefix1 > prefix1.out
./prefix2 > prefix2.out
diff -q prefix1.out prefix2.out
```



```
[07/30/21]seed@VM:~/.../md5$ ./prefix1 > prefix1.out
[07/30/21]seed@VM:~/.../md5$ ./prefix2 > prefix2.out
[07/30/21]seed@VM:~/.../md5$ diff -q prefix1.out prefix2.out
Files prefix1.out and prefix2.out differ
```

Task 4: Making the Two Programs Behave Differently

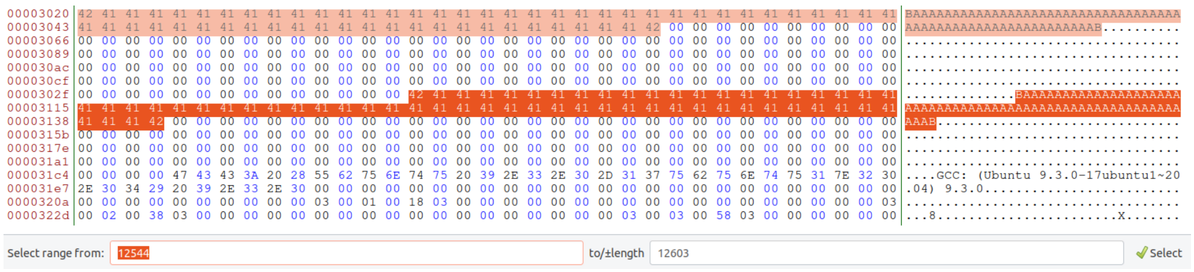
构造 `origin.c` 如下

```
#include <stdio.h>
unsigned char a[200] = {
    'B', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'B'
};
unsigned char b[200] = {
    'B', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A',
    'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'A', 'B'
};
int main()
{
    int i;
    int isSame=1;
    for(i = 0; i < 200; i++)
    {
        if(a[i]!=b[i])
            isSame=0;
    }
    if(isSame)
        printf("run benign code\n");
    else
        printf("run malicious code\n");
}
```

编译

```
gcc -o origin origin.c
```

找到两个数组位置

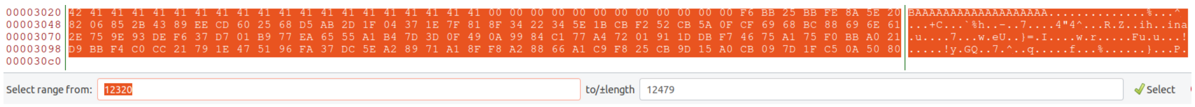


```
00003020 42 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 BAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
00003043 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 42 00 00 00 00 00 00 00 00 00 00 AAAAAAAAAAAAAAAAAAAAAAAAB..........
00003066 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
00003089 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
000030ac 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
000030cf 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
0000302f 00 00 00 00 00 00 00 00 00 00 00 00 00 00 42 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 ..............BAAAAAAAAAAAAAAAAAAAA
00003115 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
00003138 41 41 41 42 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 AAAB...............................
0000315b 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
0000317e 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
000031a1 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...................................
000031c4 00 00 00 00 47 43 43 3A 20 28 55 62 75 6E 74 75 20 39 2E 33 2E 30 2D 31 37 75 62 75 6E 74 75 31 7E 32 30 ....GCC: (Ubuntu 9.3.0-17ubuntu1~20
000031e7 2E 30 34 29 20 39 2E 33 2E 30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .04) 9.3.0.........................
0000320a 00 00 00 00 00 00 00 00 00 00 03 00 01 00 18 03 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 ...................................
0000322d 00 02 00 38 03 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03 00 03 00 58 03 00 00 00 00 00 00 ...8.......................X.......
```

与上一个 task 同样的方法，我们构造两个 md5 相同的文件

```
head -c 12340 origin > prefix
md5collgen -p prefix -o prefix1 prefix2
```

查看 prefix1



```
00003020 42 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 00 00 00 00 00 00 00 00 00 00 F6 BB 25 BB FE 8A 5E 20 BAAAAAAAAAAAAAAAAAAA.............%...^
00003048 82 06 85 2B 43 89 EE CD 60 25 68 D5 AB 2D 1F 04 37 1E 7F 81 8F 34 22 34 5E 1B CB F2 52 CB 5A 0F CF 69 68 BC 88 69 6E 61 ...+C...`%h..-..7....4"4^...R.Z..ih..ina
00003070 2E 75 9E 93 DE F6 37 D7 01 B9 77 EA 65 55 A1 B4 7D 3D 0F 49 0A 99 84 C1 77 A4 72 01 91 1D DB F7 46 75 A1 75 F0 BB A0 21 .u....7...w.eU..}=.I....w.r.....Fu.u...!
00003098 D9 BB F4 C0 CC 21 79 1E 47 51 96 FA 37 DC 5E A2 89 71 A1 8F F8 A2 88 66 A1 C9 F8 25 CB 9D 15 A0 CB 09 7D 1F C5 0A 50 80 .....!y.GQ..7.^..q.....f...%......}...P.
000030c0
```

Select range from: 12320 to/±length 12479

截取

```
tail -c +12320 prefix1 > middle # 截取生成的字符串
tail -c +12768 origin > suffix # 截取第二个字符串（不含）后面的内容
head -c 12543 origin > tmp1 # 截取第二个字符串（不含）前面的内容
```

分别运行并检查 md5

```
[07/30/21]seed@VM:~/.../md5$ ./prefix1
run benign code
[07/30/21]seed@VM:~/.../md5$ ./prefix2
run malicious code
[07/30/21]seed@VM:~/.../md5$ md5sum prefix1 prefix2
82d3b9f2e3110eeb62d5a029acef283c  prefix1
82d3b9f2e3110eeb62d5a029acef283c  prefix2
```

可以看到，它们运行了不同的代码，但 md5 是相同的。