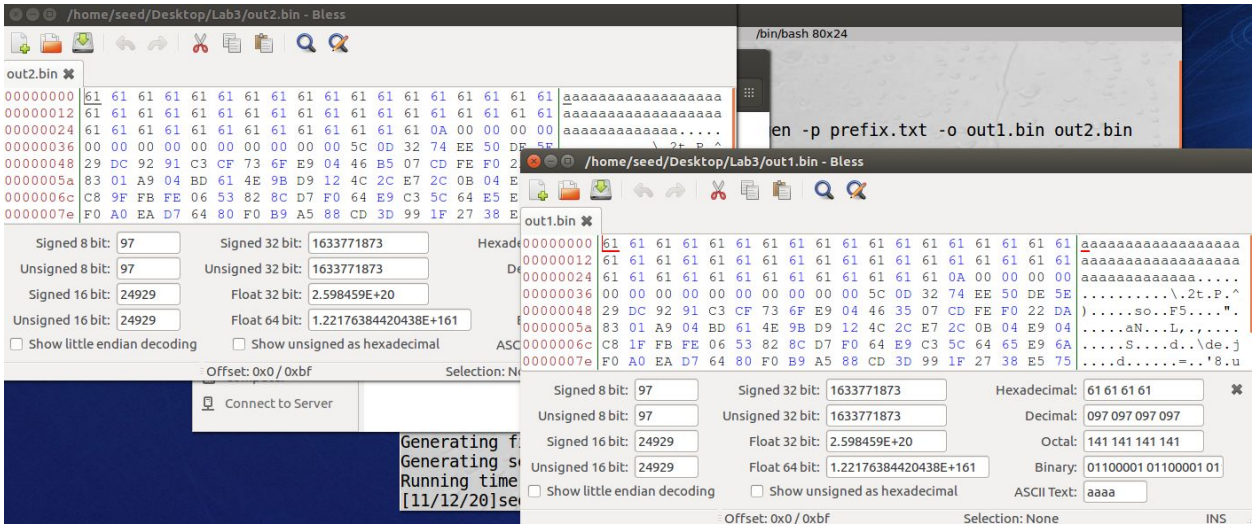
-Group 5
Computer Cryptography
Vlad Smirnov, Ryan Gumbayan, Christian Fecteau

MD5 Collision Attack Lab

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

We are told to create a .txt file and use the MD5 hash file on both. As you can see here, they seem to be the same when looking at the bits in the Bless Hex Editor.



Although when running the **diff** command to check the files the terminal displays that the binary files do differ.

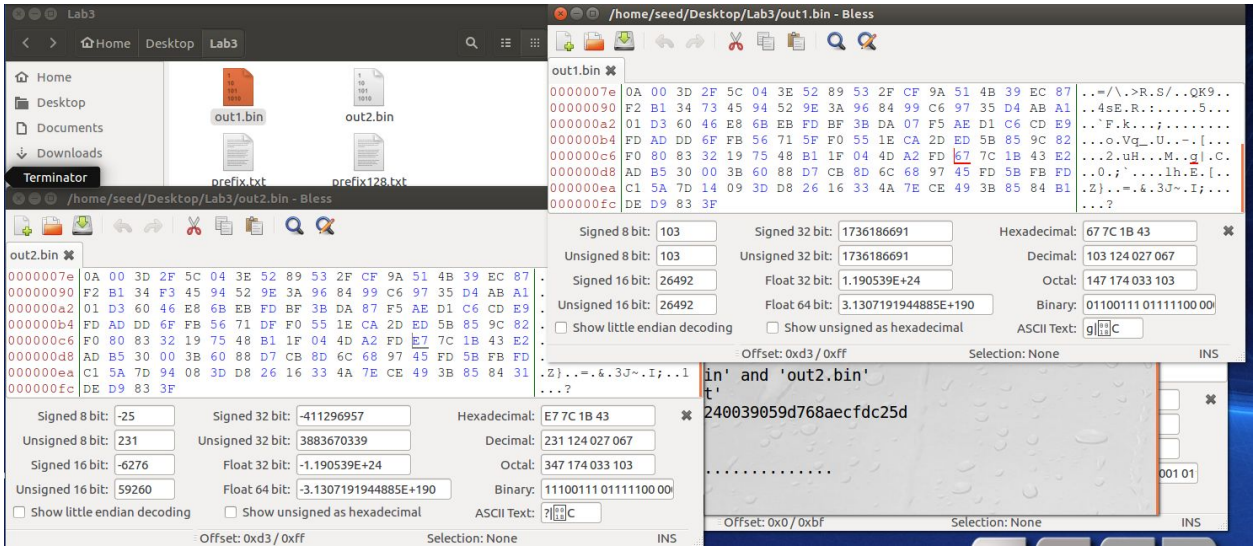
```
[11/12/20]seed@VM:~/.../Lab3$ md5collgen -p prefix.txt -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'prefix.txt'
Using initial value: 74df719f9932eed5cf2a402e2cf89a20

Generating first block: ..
Generating second block: S00.
Running time: 0.24719 s
[11/12/20]seed@VM:~/.../Lab3$ diff out1.bin out2.bin
Binary files out1.bin and out2.bin differ
[11/12/20]seed@VM:~/.../Lab3$
```

According to question 1, if the length of the prefix file is not a multiple of 64, it will be padded by zeros, as seen in the Bless Hex Editor screenshot above. Likewise, for question 2, if the file is a multiple of 64, there will be no padding, as the bits will take up each space in the file. In question 3, it seems that the change of bits between the two files is not consistent, and changes with each trial. In the screenshot below, you can see

that on line c6 of the Hex Editor, the two bits are different in each file.



TASK 2: Understanding MD5's Property

In task 2, we are told that inputs that have the same hash will keep that same hash value no matter if they are both changed in the same way. In the below screenshot, we can see that two files are created using the MD5 hash. After creating the output files, we append a random string using the echo command. Right after appending the string, we check both hash values, and can see that both still have the same hash value.

```
Using output filenames: 'hi1' and 'hi2'
Using prefixfile: 'hi.txt'
Using initial value: 03080eb00787f81b42b40efbe1306bd1

Generating first block: ................d.sssssssssssssss.ssss.ssssss..........
.....
Generating second block: W.............
Running time: 27.7089 s
[11/14/20]seed@VM:~/.../Lab3$ md5sum hi1 hi2
06ec4e75865d6f4eb9f54cb896e78a06  hi1
06ec4e75865d6f4eb9f54cb896e78a06  hi2
[11/14/20]seed@VM:~/.../Lab3$ echo hi >> hi1
[11/14/20]seed@VM:~/.../Lab3$ echo hi >> hi2
[11/14/20]seed@VM:~/.../Lab3$ md5sum hi1 hi2
899d4efb2d1ca3a5a9a988cc5497e8d0  hi1
899d4efb2d1ca3a5a9a988cc5497e8d0  hi2
[11/14/20]seed@VM:~/.../Lab3$ cat hi1 hi2 > hi3
```

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

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

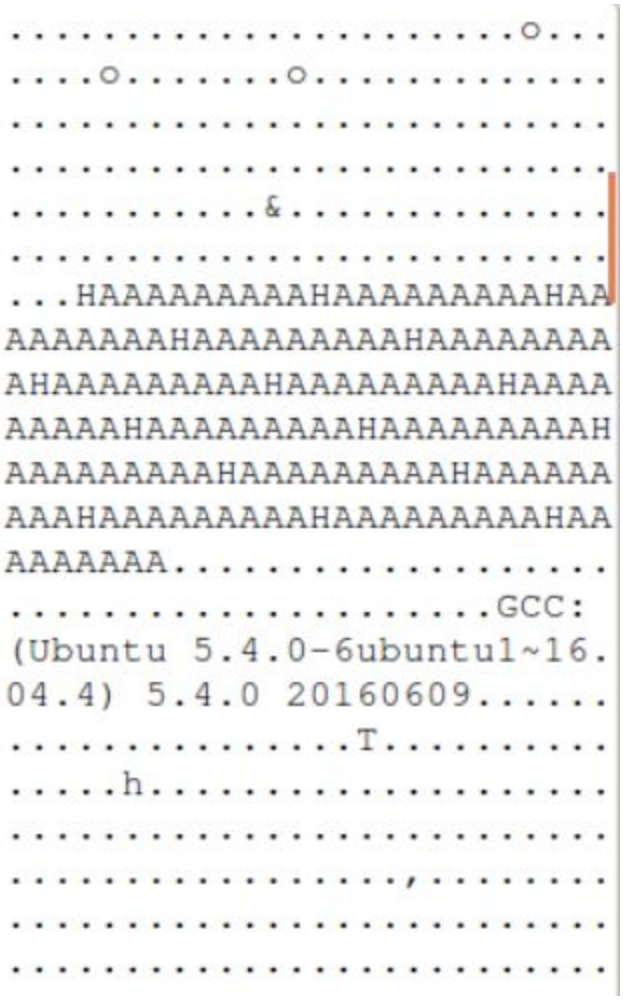
Compiled the code above into program.out . We can find the location value of the array that is stored in the program. We will separate the program.out file into three and save the first 3200 bytes to a prefix file. Then we will save the last 100 bytes of program.out to suffix file. Then we will save 3300th byte to the end of suffix file.

```
[11/16/20]seed@VM:~/Desktop$ head -c 3200 program.out > prefix
[11/16/20]seed@VM:~/Desktop$ tail -c 100 program.out > suffix
[11/16/20]seed@VM:~/Desktop$ tail -c +3300 program.out > suffix
```

After running the commands, below we get the two separate files that contain the data from the array.

```
.......................o...
....o.......o.............
..........................
..........................
...........&..............
..........................
...HAAAAAAAAAHAAAAAAAAAHAA
AAAAAAAHAAAAAAAAAHAAAAAAA
AHAAAAAAAAAHAAAAAAAAAHAAA
AAAAAHAAAAAAAAAHAAAAAAAAH
AAAAAAAAAHAAAAAAAAAHAAAAA
AAAHAAAAAAAAAHAAAAAAAAAHA
AAAAAAA...................
....................GCC:
(Ubuntu 5.4.0-6ubuntu1~16.
04.4) 5.4.0 20160609......
..............T...........
.....h....................
..........................
................,.........
..........................
..........................
```

Prefix

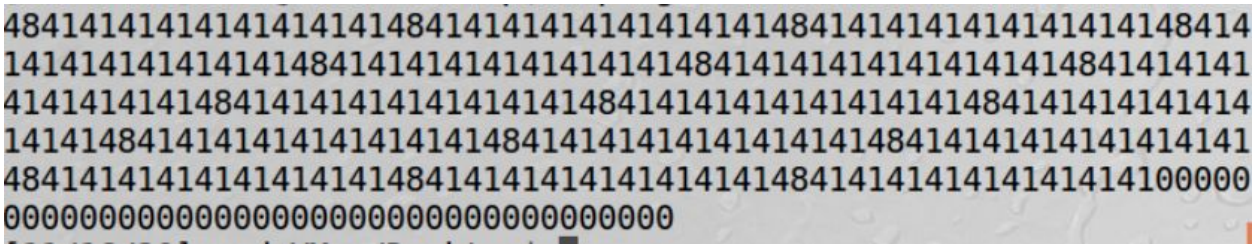


Suffix

Concatenating each file with a common end from program.out and running each individual program with the commands ./prefix and ./suffix gave us the following outputs below:

```
48414141414141414141484141414141414141414841414141414141414148414
14141414141414148414141414141414141484141414141414141484141414141
41414141414841414141414141414148414141414141414148414141414141414
14141484141414141414141484141414141414141484141414141414141414141
48414141414141414141484141414141414141484141414141414141414100000
0000000000000000000000000000000000000
```

Suffix output



Prefix output

We then stored the output of each file into a text file with argument ./prefix > prefixoutp and ./suffix > suffixoutp and compared the content. Comparing the output on each file suffix and prefix results that each output is different. Although the content outputted looks similar, we ran the command diff -q on the two output files. This returned an argument of:

Files suffixout and prefixout differ

TASK 4: Making the Two Programs Behave Differently

In task 4 we are tasked with creating two programs which have the same MD5 sum, but both act differently. I was not able to make them act differently, and did not completely understand how to make both programs different. However, I was able to separate the parts of the binary files. As seen below, I tried to concatenate different parts of the code, but was not successful.

```
Generating first block: .
Generating second block: S01.................................................
Running time: 2.84768 s
[11/16/20]seed@VM:~/.../Lab3$ bless legen kegen
Unexpected end of file has occurred. The following elements are not closed: pref, preferences. Line
22, position 36.
Directory '/home/seed/.config/bless/plugins' not found.
Directory '/home/seed/.config/bless/plugins' not found.
Directory '/home/seed/.config/bless/plugins' not found.
Could not find file "/home/seed/.config/bless/export_patterns".
Could not find file "/home/seed/.config/bless/history.xml".
Document does not have a root element.
Sharing violation on path /home/seed/.config/bless/preferences.xml
Sharing violation on path /home/seed/.config/bless/preferences.xml
Sharing violation on path /home/seed/.config/bless/preferences.xml
Document does not have a root element.
[11/16/20]seed@VM:~/.../Lab3$ md5sum legen kegen
49e8690ac65d035fd74ce7673a3a10fc  legen
49e8690ac65d035fd74ce7673a3a10fc  kegen
[11/16/20]seed@VM:~/.../Lab3$ head -c +4353 a.out > end
[11/16/20]seed@VM:~/.../Lab3$ tail -c +321 end > commonend
[11/16/20]seed@VM:~/.../Lab3$ cp pgen benigncode
cp: cannot stat 'pgen': No such file or directory
```

```
[11/16/20]seed@VM:~/.../Lab3$ cp legen benigncode
[11/16/20]seed@VM:~/.../Lab3$ cp kegen maliciouscode
[11/16/20]seed@VM:~/.../Lab3$ cat middle >> benigncode
cat: middle: No such file or directory
[11/16/20]seed@VM:~/.../Lab3$ md5sum benigncode maliciouscode
49e8690ac65d035fd74ce7673a3a10fc  benigncode
49e8690ac65d035fd74ce7673a3a10fc  maliciouscode
[11/16/20]seed@VM:~/.../Lab3$ chmod +x benigncode maliciouscode
[11/16/20]seed@VM:~/.../Lab3$ ./benigncode
This is where malicious code would be run
[11/16/20]seed@VM:~/.../Lab3$ ./maliciouscode
This is where malicious code would be run
[11/16/20]seed@VM:~/.../Lab3$
```