

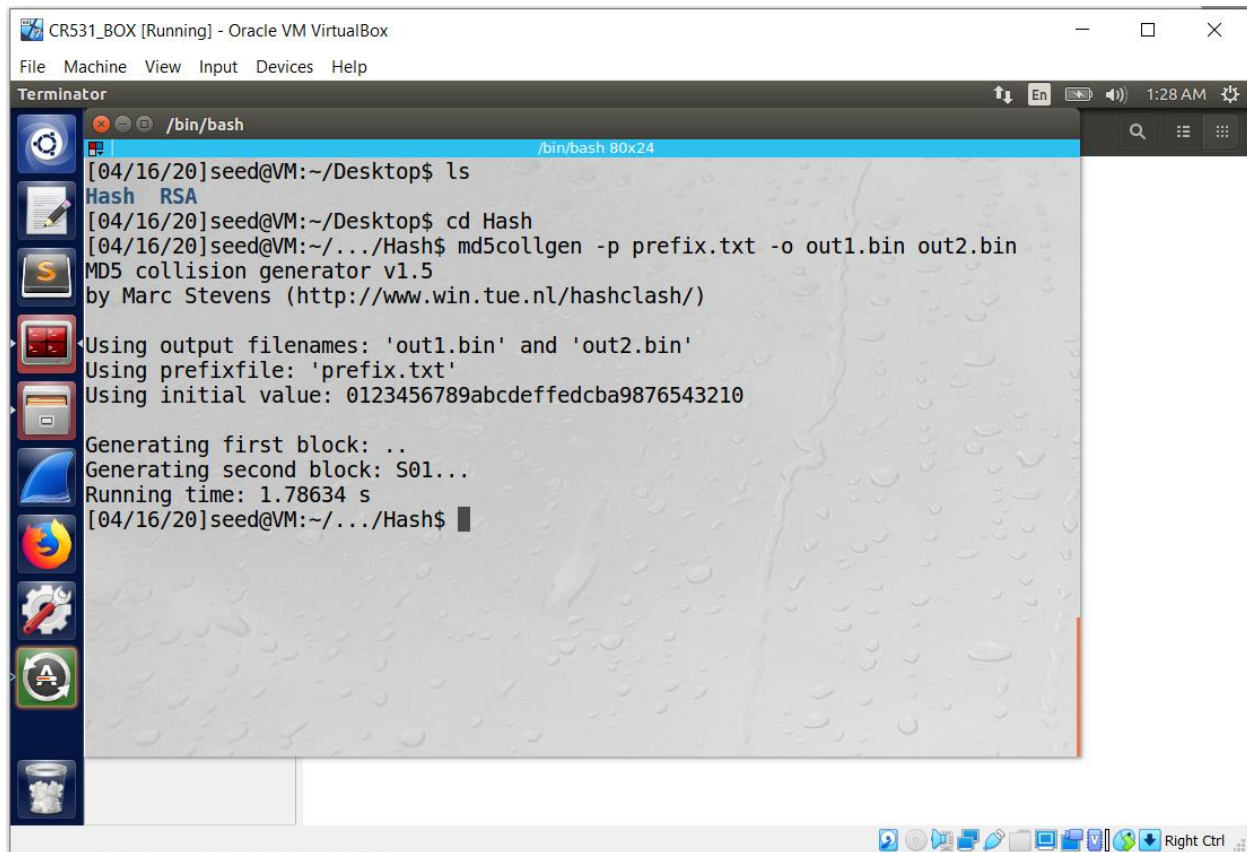
CS-767 FOUNDATIONS NETWORK SECURITY

HASH LAB

Chakradhar Reddy Donuri

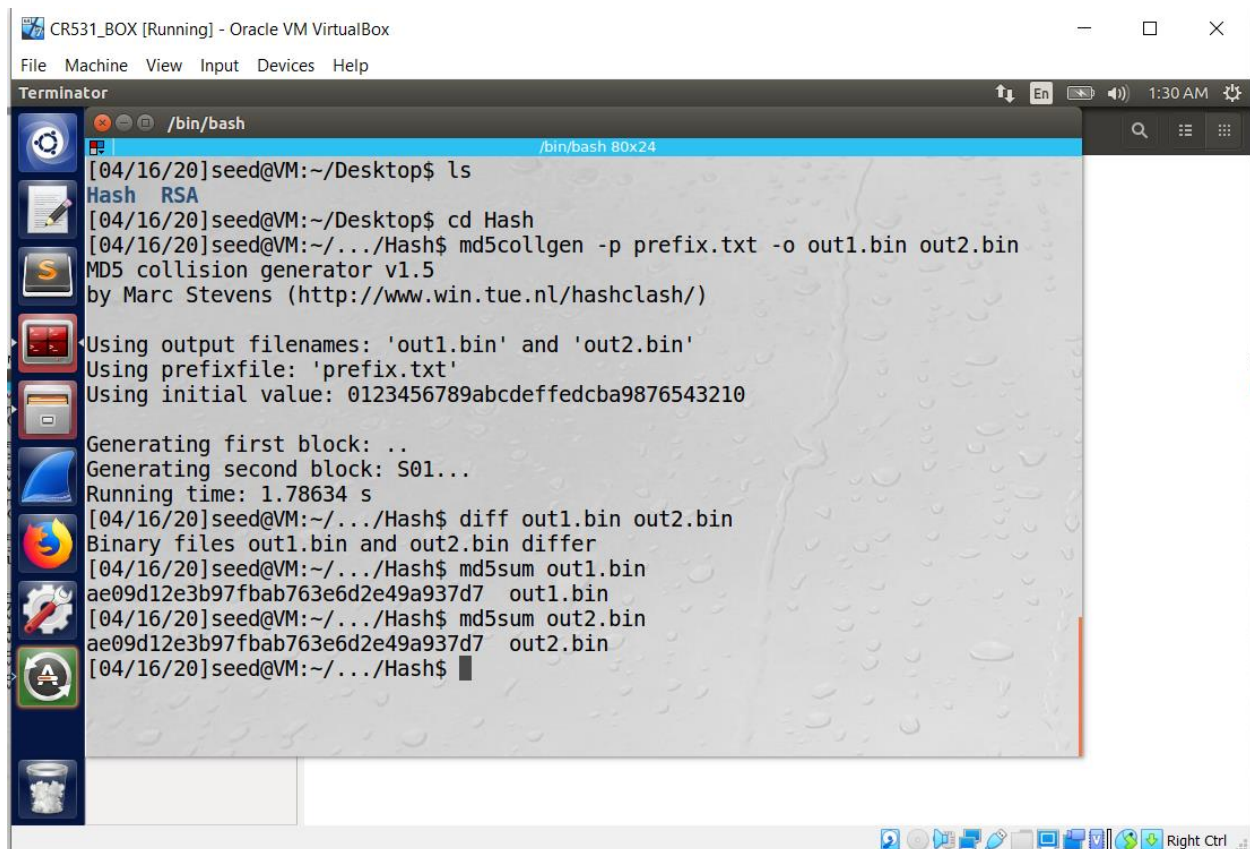
E949F496

Task-1: The following command generates two output files, out1.bin and out2.bin, for a given a prefix file prefix.txt: `$ md5collgen -p prefix.txt -o out1.bin out2.bin`



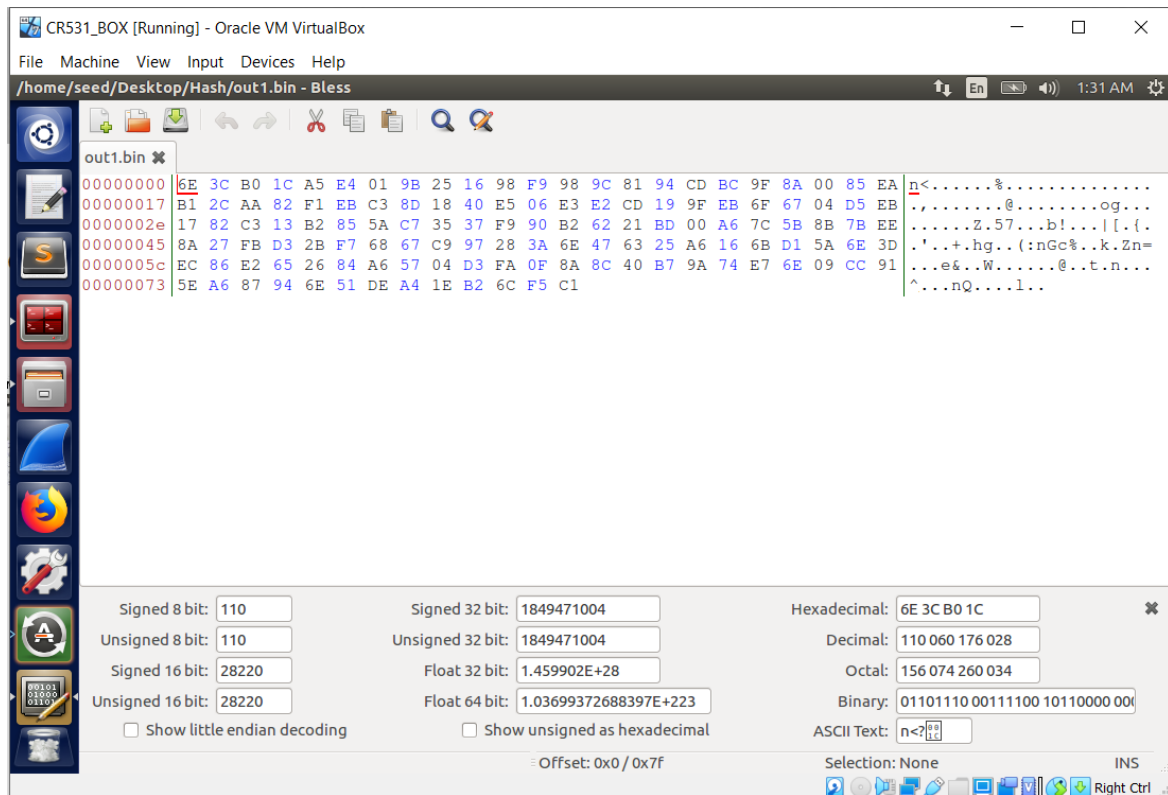
```
CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
/bin/bash
[04/16/20]seed@VM:~/Desktop$ ls
Hash RSA
[04/16/20]seed@VM:~/Desktop$ cd Hash
[04/16/20]seed@VM:~/../Hash$ md5collgen -p prefix.txt -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)
Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'prefix.txt'
Using initial value: 0123456789abcdeffedcba9876543210
Generating first block: ..
Generating second block: S01...
Running time: 1.78634 s
[04/16/20]seed@VM:~/../Hash$
```

Now We can check whether the output files are distinct or not using the diff command. We can also use the md5sum command to check the MD5 hash of each output file

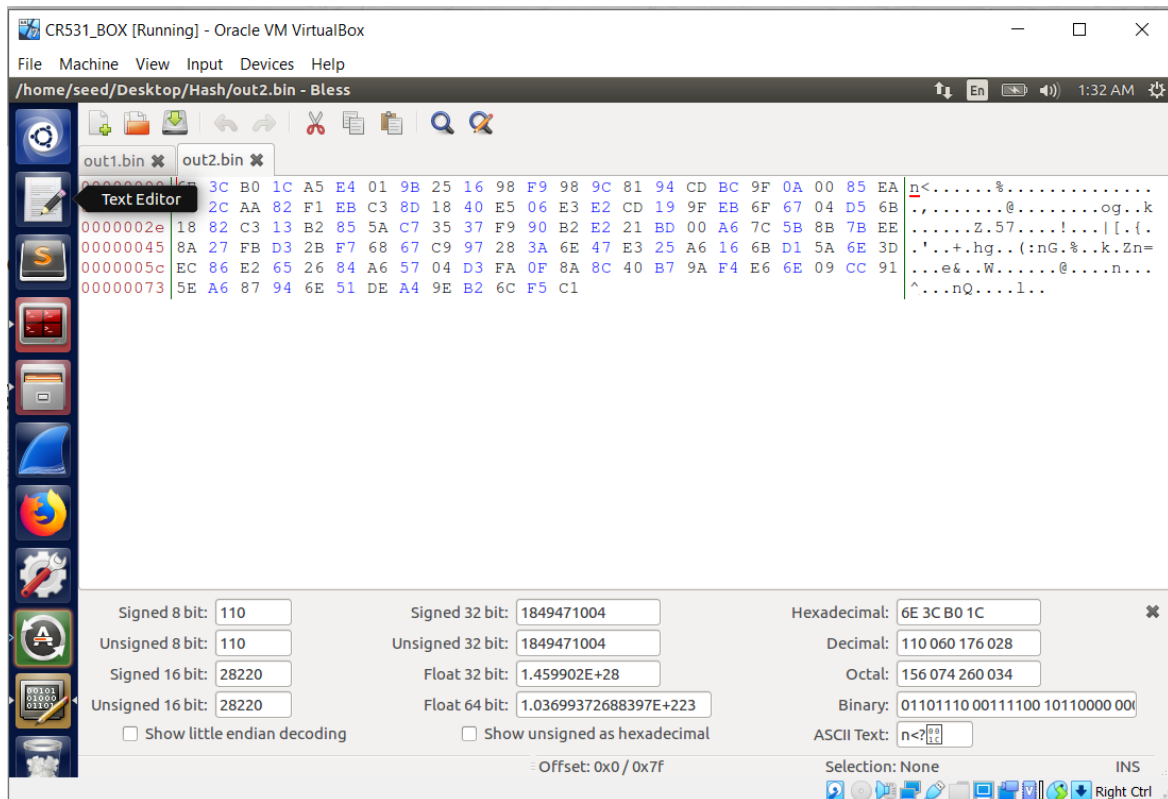


```
[04/16/20]seed@VM:~/Desktop$ ls
Hash  RSA
[04/16/20]seed@VM:~/Desktop$ cd Hash
[04/16/20]seed@VM:~/../Hash$ md5collgen -p prefix.txt -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)
Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'prefix.txt'
Using initial value: 0123456789abcdeffedcba9876543210
Generating first block: ..
Generating second block: S01...
Running time: 1.78634 s
[04/16/20]seed@VM:~/../Hash$ diff out1.bin out2.bin
Binary files out1.bin and out2.bin differ
[04/16/20]seed@VM:~/../Hash$ md5sum out1.bin
ae09d12e3b97fbab763e6d2e49a937d7  out1.bin
[04/16/20]seed@VM:~/../Hash$ md5sum out2.bin
ae09d12e3b97fbab763e6d2e49a937d7  out2.bin
[04/16/20]seed@VM:~/../Hash$
```

Out1.bin in Bless Editor



Out2.bin in Bless editor



1. Suppose that you use an arbitrary prefix file that has a byte length that is not a multiple of 64 as your input to md5collgen. What will md5collgen do in terms of padding?

Ans: Extra zeros are padded

out1.bin ✖	out2.bin ✖
00000000	6E 3C B0 1C A5 E4 01 9B 25 16 98 F9 98 9C 81 94 CD BC 9F 8A 00 85 EA B1 2C AA 82 F1 EB C3
0000001e	8D 18 40 E5 06 E3 E2 CD 19 9F EB 6F 67 04 D5 EB 17 82 C3 13 B2 85 5A C7 35 37 F9 90 B2 62
0000003c	21 BD 00 A6 7C 5B 8B 7B EE 8A 27 FB D3 2B F7 68 67 C9 97 28 3A 6E 47 63 25 A6 16 6B D1 5A
0000005a	6E 3D EC 86 E2 65 26 84 A6 57 04 D3 FA 0F 8A 8C 40 B7 9A 74 E7 6E 09 CC 91 5E A6 87 94 6E
00000078	51 DE A4 1E B2 6C F5 C1

2. Suppose that you a prefix file that is 64 bytes long as input to md5collgen again. What happens in regards to padding in this case?

Ans: If it is exact 64 bytes then no extra zeros are padded.

3. Create an arbitrary prefix file and generate two binaries with the same MD5 sum as explained above. The files generated by md5collgen are different. Which bytes are different and what are their values? You can use bless and/or cmp to find the answer.

Ans: From the Bless editor screenshots attached above it is observed that

Bytes that are different (20,46,47,60,84,110,111,124)

Values at those positions in out1.bin are (8A, EB, 17, 62, 63, 74, E7, 1E)

Values at those positions in out2.bin are (0A, 6B, 18, E2, E3, F4, E6, 9E)

4. The output files are different but they have the same MD5 sum. Identify the cryptographic hash property that is not met by the MD5 hash.

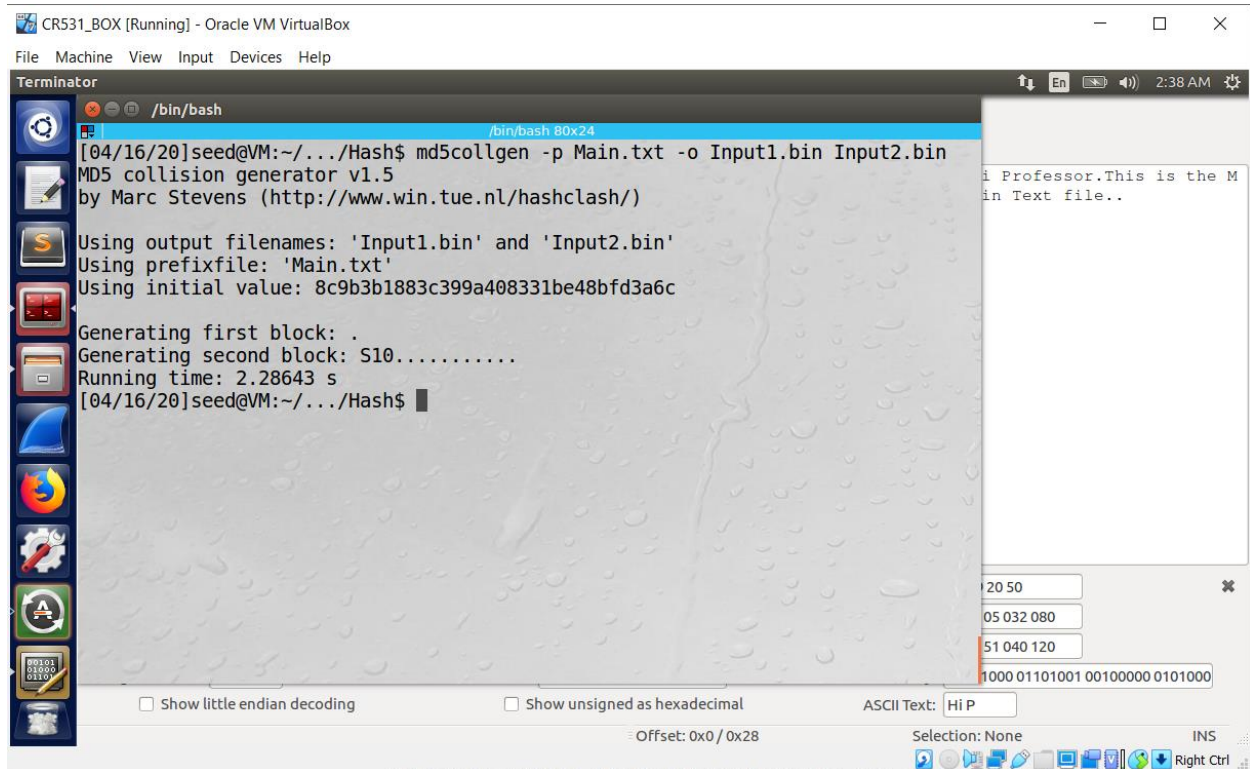
Ans: Two different messages or file with same MD5 sum This property is sometimes referred to as strong collision resistance.

5. Identify an attack that can be launched due to the vulnerability that you identified above

Ans: Collision attack on a cryptographic hash that tries to find two inputs files or messages producing the same hash value, i.e. a hash collision.

Task-2: Adding a suffix to two files that have the same hash value

Consider 2 input files names Input1.bin and Input2.bin and a file that is used to concatenate is Main.txt

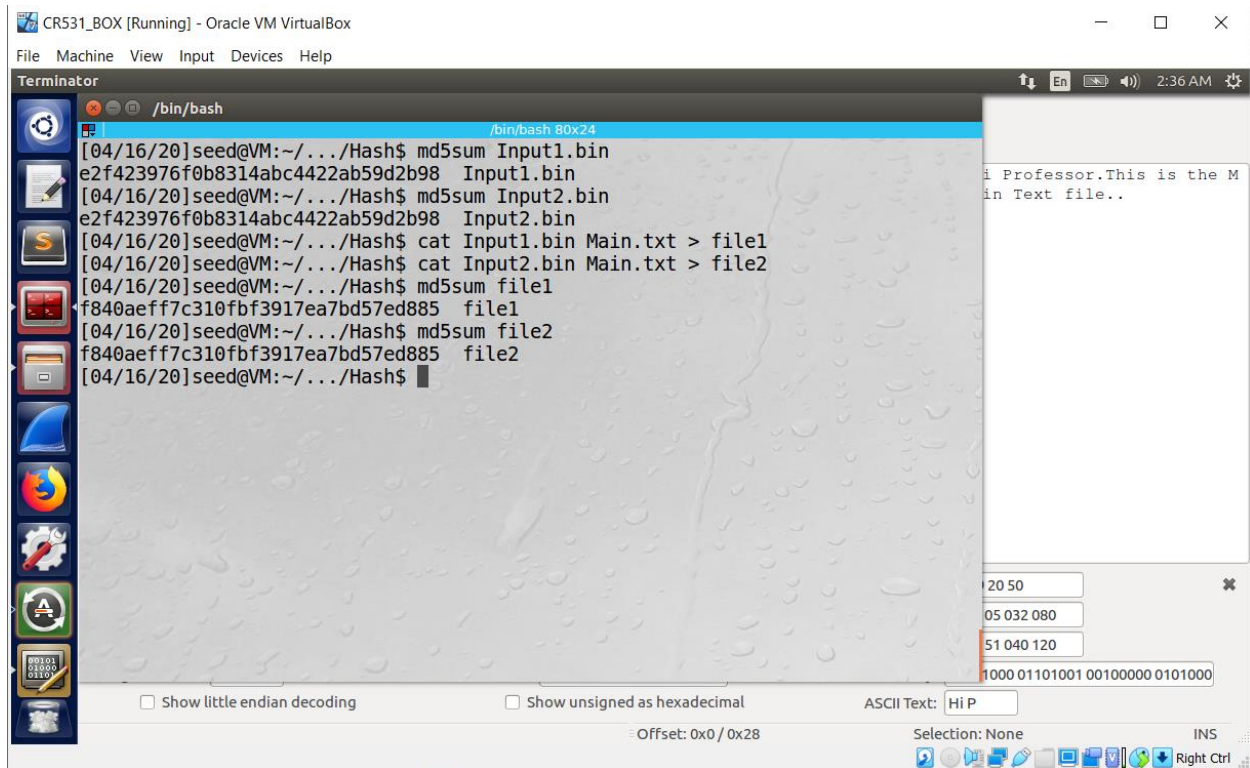


```
CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
/bin/bash
[04/16/20]seed@VM:~/.../Hash$ md5collgen -p Main.txt -o Input1.bin Input2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)
Using output filenames: 'Input1.bin' and 'Input2.bin'
Using prefixfile: 'Main.txt'
Using initial value: 8c9b3b1883c399a408331be48bfd3a6c
Generating first block: .
Generating second block: S10.....
Running time: 2.28643 s
[04/16/20]seed@VM:~/.../Hash$
```

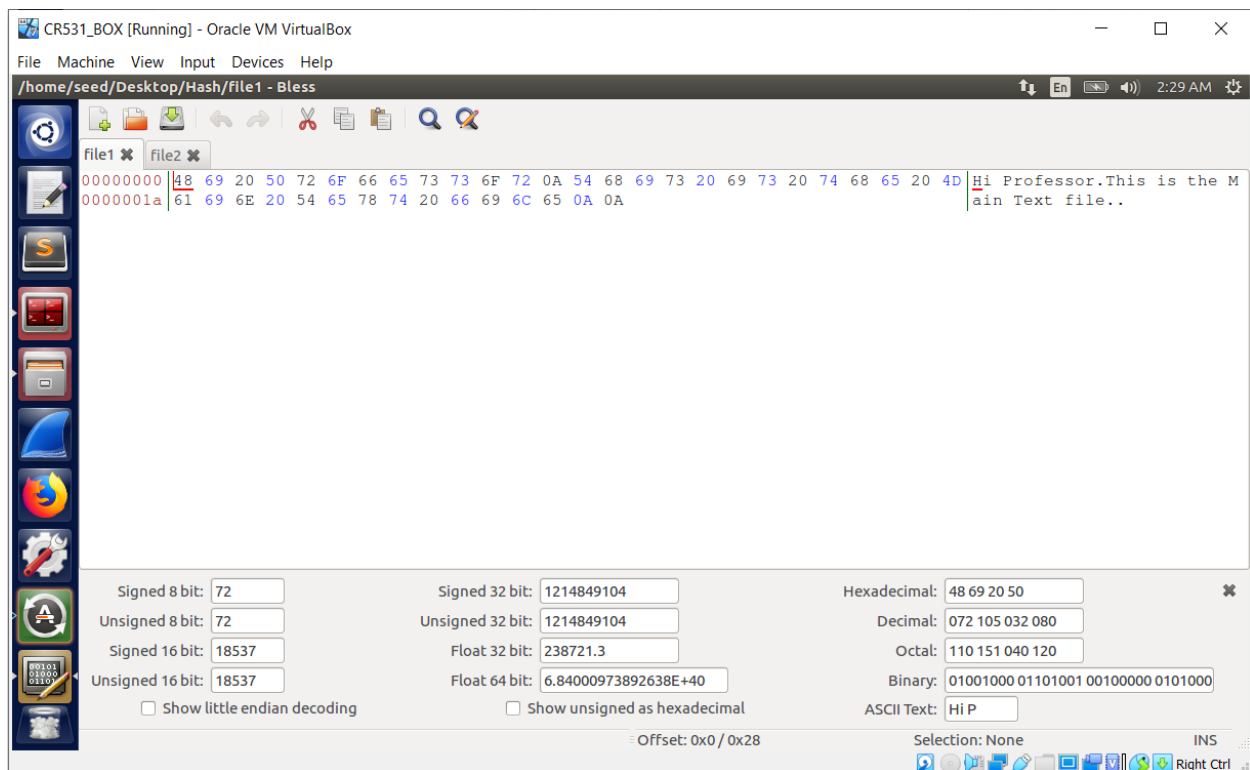
i Professor.This is the M
in Text file..

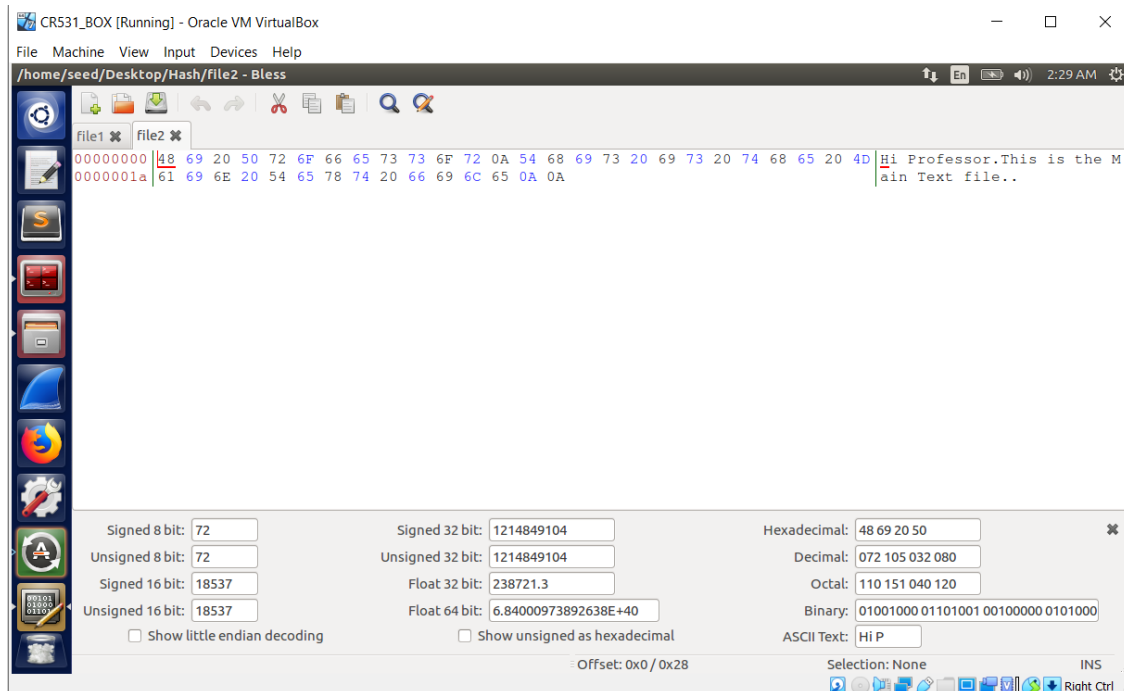
20 50
05 032 080
51 040 120
1000 01101001 00100000 0101000

Show little endian decoding Show unsigned as hexadecimal ASCII Text: Hi P
Offset: 0x0 / 0x28 Selection: None INS
Right Ctrl



File1 and File2 In Bless Editor





From the above experiment, we can observe that this property holds for MD5 i.e. Input files have same hash value and adding the same Main.txt file to each Input file resulted in two outputs that have same hash value.

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

1) The ending byte position of your prefix and the starting byte position of your suffix in the original C program?

Ans) Now, we head -c 64 a.out > pre and tail -c +4320 program > suf

Ending byte of pre in bless : 1040

Starting byte of suf in bless : 10e0

2) A screen shot showing that the output of md5sum for both of your programs (i.e., the ones that are built with the two outputs from md5collgen) is the same.

Ans)

[illegible]

1) The C code that implements the pseudo-code. 2) The benign program. 3) The malicious program.

CR531_BOX [Running] - Oracle VM VirtualBox

[illegible]

The screenshot displays the Oracle VM VirtualBox interface with the CR531_BOX [Running] window. The main display area shows a hex dump of the file 'a.out'. The hex dump is organized into columns: address, hex, and ASCII. The hex dump shows various data including '00000f8a', '00000fa4', '00000fb2', etc. The bottom status bar shows 'Offset: 0x1040 / 0x1f87' and 'Selection: None'. The top menu bar includes 'File', 'Machine', 'View', 'Input', 'Devices', 'Help'.

CR531_BOX [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Terminator

```
/bin/bash
[04/16/20]seed@VM:~/.../Hash$ head -c +193 a.out > end
[04/16/20]seed@VM:~/.../Hash$ head -c 64 a.out > prefix
[04/16/20]seed@VM:~/.../Hash$ md5collgen -p prefix -o Mgen Ngen
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'Mgen' and 'Ngen'
Using prefixfile: 'prefix'
Using initial value: 5bb555f1d6ecd128df2d6ed5024eadf5

Generating first block: .....
Generating second block: S10.....
Running time: 5.01239 s
[04/16/20]seed@VM:~/.../Hash$
```

Save

C Tab Width: 8 Ln 13, Col 1 INS Right Ctrl

4) Screen shot of the output of md5sum for the binaries of the benign and malicious programs

CR531_BOX [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Terminator

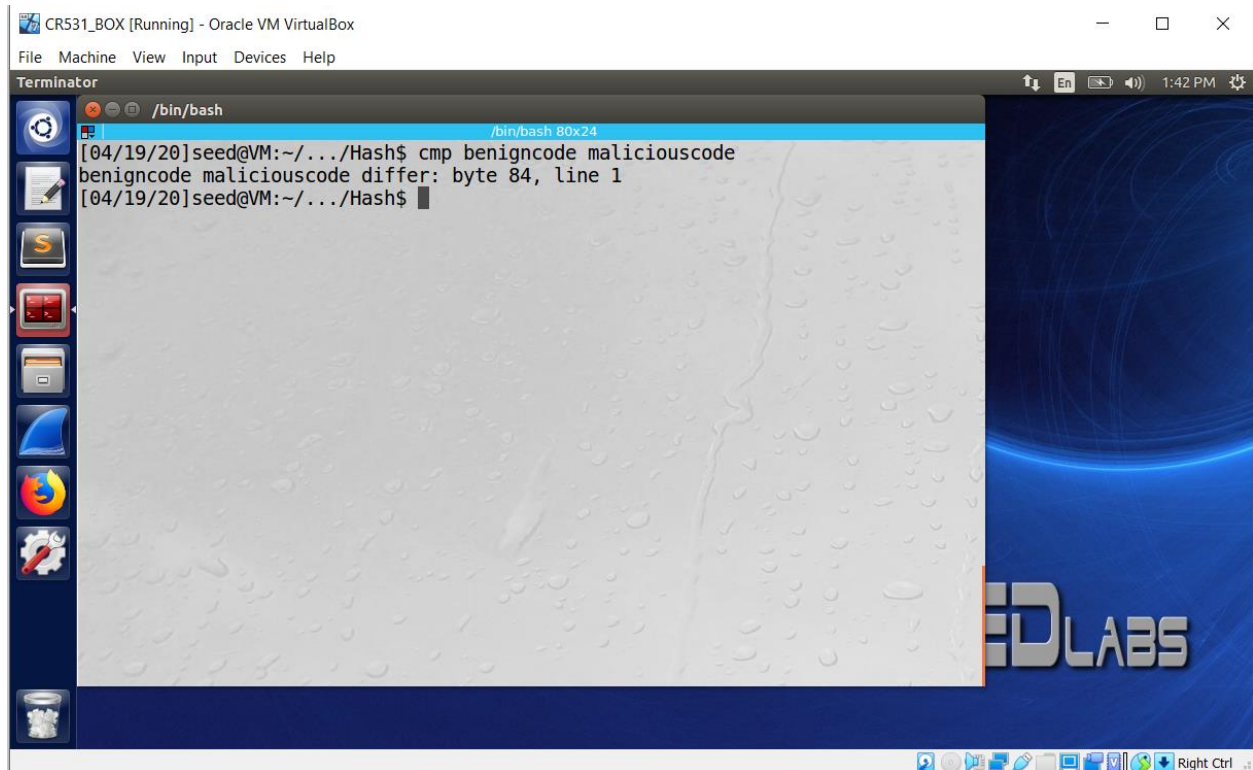
```
/bin/bash
[04/16/20]seed@VM:~/.../Hash$ tail -c +321 end > commonend
[04/16/20]seed@VM:~/.../Hash$ cp Mgen benigncode
[04/16/20]seed@VM:~/.../Hash$ cp Ngen maliciouscode
[04/16/20]seed@VM:~/.../Hash$ cat end >> benigncode
[04/16/20]seed@VM:~/.../Hash$ cat end >> maliciouscode
[04/16/20]seed@VM:~/.../Hash$ cat p >> benigncode
[04/16/20]seed@VM:~/.../Hash$ cat p >> maliciouscode
[04/16/20]seed@VM:~/.../Hash$ cat commonend >> benigncode
[04/16/20]seed@VM:~/.../Hash$ cat commonend >> maliciouscode
[04/16/20]seed@VM:~/.../Hash$ md5sum benigncode
5d11e3cc62cd52917ec7d8b5f38ff17f benigncode
[04/16/20]seed@VM:~/.../Hash$ md5sum maliciouscode
5d11e3cc62cd52917ec7d8b5f38ff17f maliciouscode
[04/16/20]seed@VM:~/.../Hash$
```

EDLABS

Right Ctrl

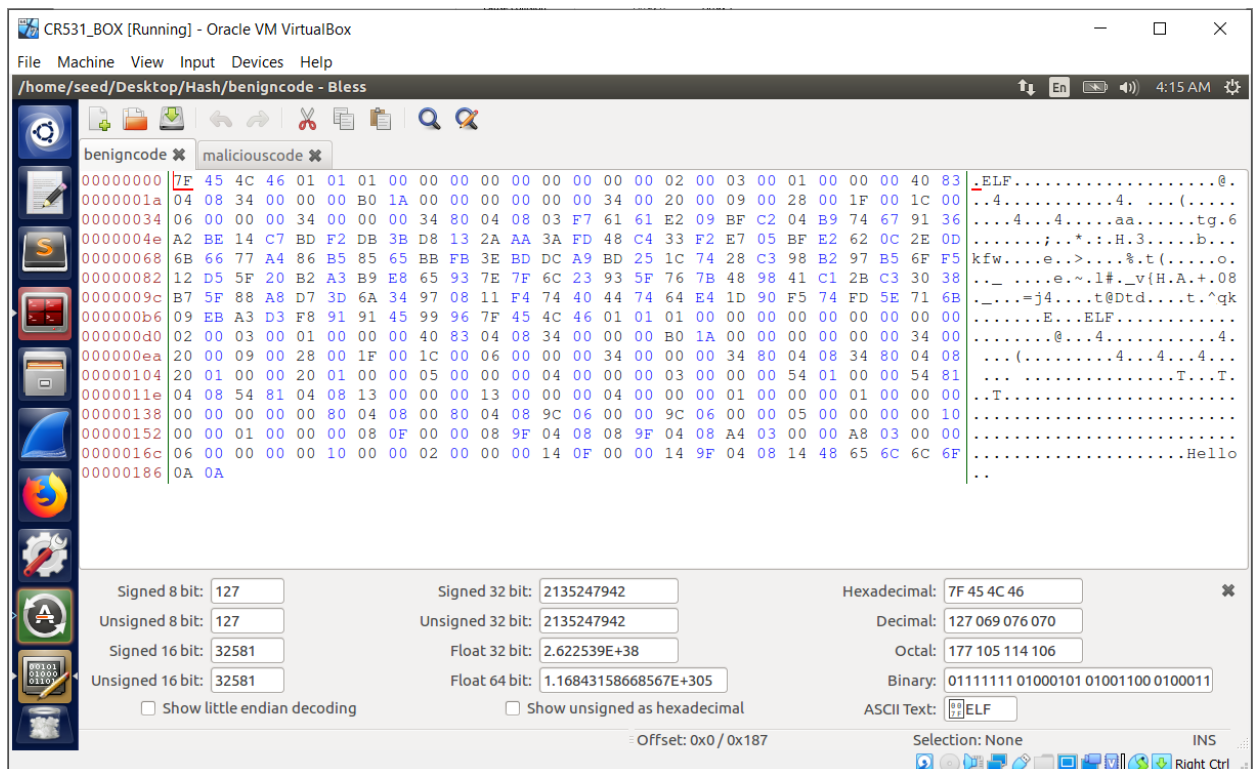
They have same Hash values.

Screen shot of the output of cmp to compare the contents of the malicious and benign binaries



```
CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
/bin/bash
[04/19/20]seed@VM:~/../Hash$ cmp benigncode maliciouscode
benigncode maliciouscode differ: byte 84, line 1
[04/19/20]seed@VM:~/../Hash$
```

Benigncode and maliciouscode in Bless editor



```
CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
/home/seed/Desktop/Hash/benigncode - Bless
benigncode x maliciouscode x
00000000 7F 45 4C 46 01 01 00 00 00 00 00 00 00 02 00 03 00 01 00 00 00 40 83 .ELF.....@.
0000001a 04 08 34 00 00 00 B0 1A 00 00 00 00 00 34 00 20 00 09 00 28 00 1F 00 1C 00 ..4.....4. ... (....
00000034 06 00 00 00 34 00 00 00 34 80 04 08 03 F7 61 61 E2 09 BF C2 04 B9 74 67 91 36 ....4...4.....aa.....tg.6
0000004e A2 BE 14 C7 BD F2 DB 3B D8 13 2A AA 3A FD 48 C4 33 F2 E7 05 BF E2 62 0C 2E 0D ...../...*.H.3.....b..
00000068 6B 66 77 A4 86 B5 85 65 BB FB 3E BD DC A9 BD 25 1C 74 28 C3 98 B2 97 B5 6F F5 kfw....e..>...%.t(.....o
00000082 12 D5 5F 20 B2 A3 B9 E8 65 93 7E 7F 6C 23 93 5F 76 7B 48 98 41 C1 2B C3 30 38 ..._.~.e..l#.._v{H.A.+08
0000009c B7 5F 88 A8 D7 3D 6A 34 97 08 11 F4 74 40 44 74 64 E4 1D 90 F5 74 FD 5E 71 6B ..._.=j4....t@Dtd....t.^qk
000000b6 09 EB A3 D3 F8 91 91 45 99 96 7F 45 4C 46 01 01 01 00 00 00 00 00 00 00 00 .....E...ELF.....
000000d0 02 00 03 00 01 00 00 00 40 83 04 08 34 00 00 00 B0 1A 00 00 00 00 00 34 00 .....@....4.....4.
000000ea 20 00 09 00 28 00 1F 00 1C 00 06 00 00 00 34 00 00 00 34 80 04 08 34 80 04 08 ...(. ....4...4...4...
00000104 20 01 00 00 20 01 00 00 05 00 00 00 04 00 00 00 03 00 00 54 01 00 00 54 81 ... ..T...T...
0000011e 04 08 54 81 04 08 13 00 00 00 13 00 00 00 04 00 00 00 01 00 00 01 00 00 00 ...T.....
00000138 00 00 00 00 00 80 04 08 00 80 04 08 9C 06 00 00 9C 06 00 00 05 00 00 00 10 .....
00000152 00 00 01 00 00 08 0F 00 00 08 9F 04 08 08 9F 04 08 A4 03 00 00 A8 03 00 00 .....
0000016c 06 00 00 00 00 10 00 00 02 00 00 00 14 0F 00 00 14 9F 04 08 14 48 65 6C 6C 6F .....Hello
00000186 0A 0A
```

Signed 8 bit: 127 Signed 32 bit: 2135247942 Hexadecimal: 7F 45 4C 46
Unsigned 8 bit: 127 Unsigned 32 bit: 2135247942 Decimal: 127 069 076 070
Signed 16 bit: 32581 Float 32 bit: 2.622539E+38 Octal: 177 105 114 106
Unsigned 16 bit: 32581 Float 64 bit: 1.16843158668567E+305 Binary: 01111111 01000101 01001100 01000111
☐ Show little endian decoding ☐ Show unsigned as hexadecimal ASCII Text: ELF
Offset: 0x0 / 0x187 Selection: None INS

