FULGRIM'S WRITEUP #1

Crackme name: **Easiest**

Author: NullerF

Language: C/C++

Platform: Windows

Hello everyone, this is my first writeup in the field of reverse engineering and cracking (I am novice in this field pivoting from pentesting/system engineering).

Alongside my daily work, I plan to release one writeup per week, whenever possible.

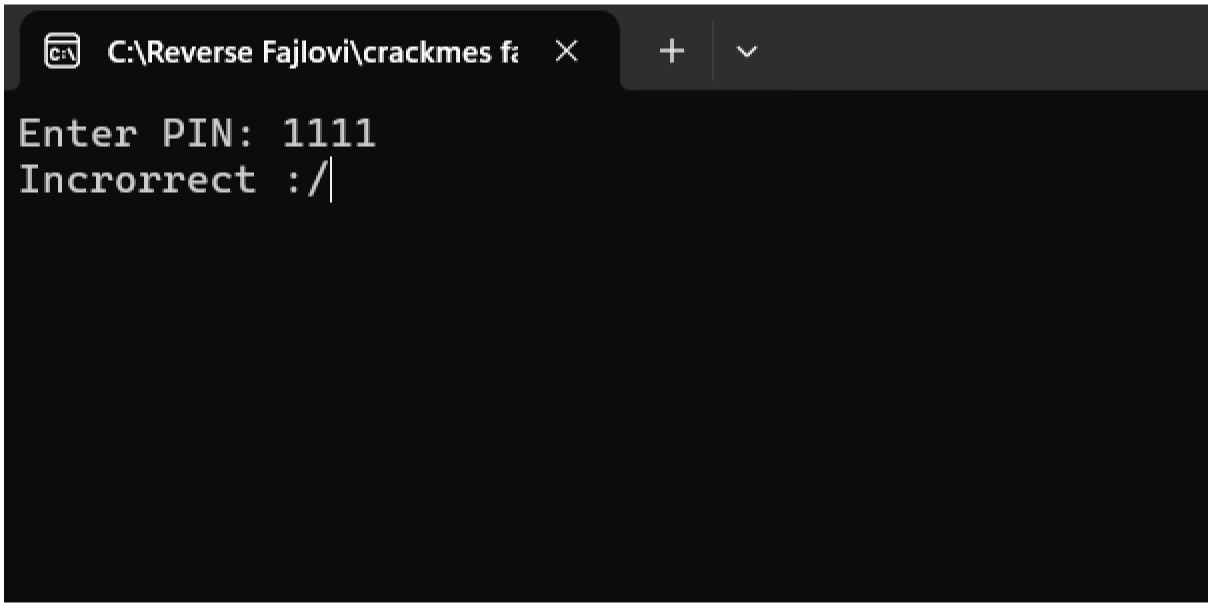
This one in particular is a very easy crackme, perfect for beginners. Hope you enjoy it :)

**Note: Greetings to NullerF for releasing this crackme!*

Analysis

For the sake of this crackme I will be using x64dbg as my preference, if you want to follow along I recommend using the same as me. But feel free to use whatever you are comfortable with.

For the starter, the console asks us to input the correct PIN in order to give us the correct result.

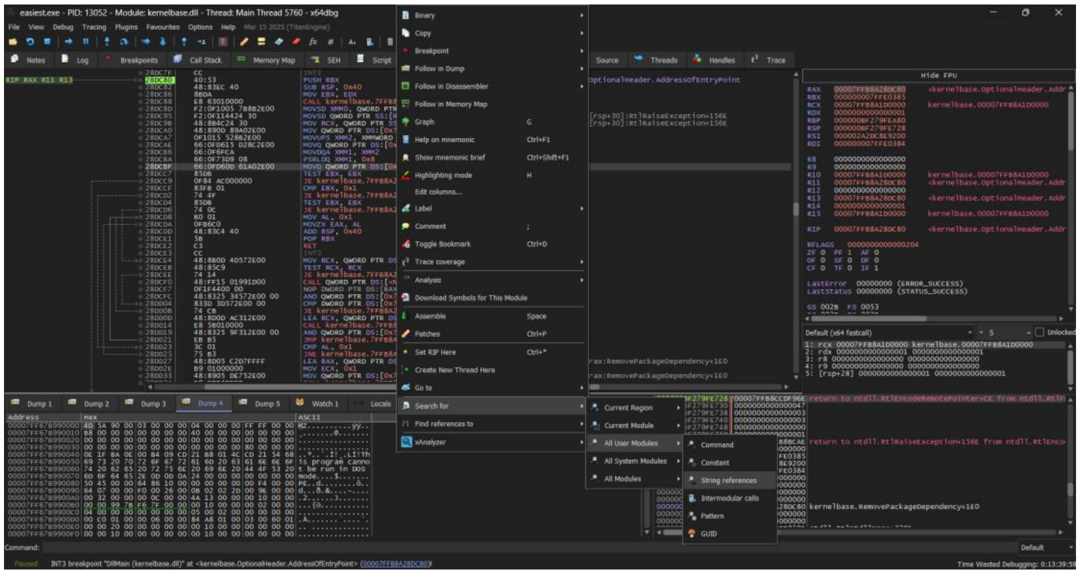


```
Enter PIN: 1111
Incrorrect :/
```

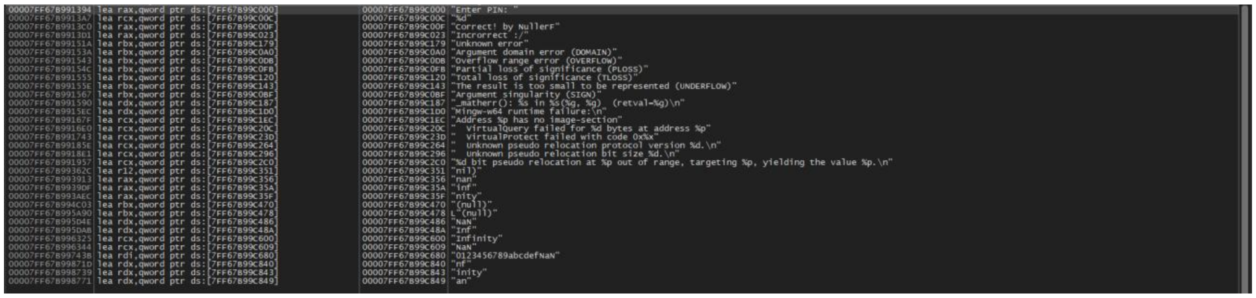
As far as I know, we are not given correct creds (shocking I know) so it’s our responsibility to find it by reversing the .exe.

Next up, let’s try finding the strings in x64dbg by searching the console prompt for the PIN:

When you open x64dbg:
Right click in CPU/Dissassembly window → Search for → All User Modules → String References

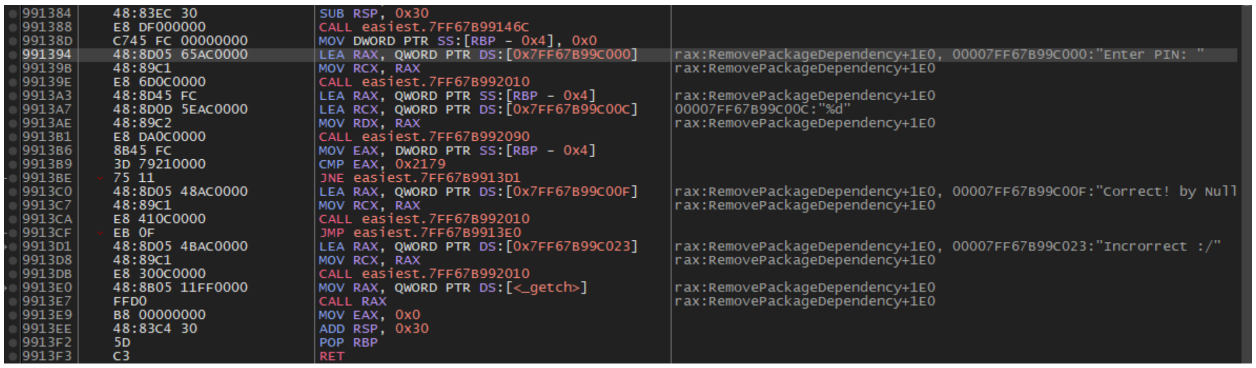


You will be prompted by strings x64dbg could find in disassembly view:



```
00007FF67B991394 lea rax,qword ptr ds:[7FF67B99C000]      00007FF67B99C000 "Enter PIN: "
00007FF67B9913A7 lea rcx,qword ptr ds:[7FF67B99C00C]      00007FF67B99C00C "%d"
00007FF67B9913C0 lea rax,qword ptr ds:[7FF67B99C00F]      00007FF67B99C00F "Correct! by NullerF"
00007FF67B9913D1 lea rax,qword ptr ds:[7FF67B99C023]      00007FF67B99C023 "Incrorrect :/"
00007FF67B99151A lea rbx,qword ptr ds:[7FF67B99C179]      00007FF67B99C179 "Unknown error"
00007FF67B99153A lea rbx,qword ptr ds:[7FF67B99C0A0]      00007FF67B99C0A0 "Argument domain error (DOMAIN)"
00007FF67B991543 lea rbx,qword ptr ds:[7FF67B99C0DB]      00007FF67B99C0DB "Overflow range error (OVERFLOW)"
00007FF67B99154C lea rbx,qword ptr ds:[7FF67B99C0FB]      00007FF67B99C0FB "Partial loss of significance (PLOSS)"
00007FF67B991555 lea rbx,qword ptr ds:[7FF67B99C120]      00007FF67B99C120 "Total loss of significance (TLOSS)"
00007FF67B99155E lea rbx,qword ptr ds:[7FF67B99C143]      00007FF67B99C143 "The result is too small to be represented (UNDERFLOW)"
00007FF67B991567 lea rbx,qword ptr ds:[7FF67B99C0BF]      00007FF67B99C0BF "Argument singularity (SIGN)"
00007FF67B991590 lea rdx,qword ptr ds:[7FF67B99C187]      00007FF67B99C187 "_matherr(): %s in %s(%g, %g)  (retval=%g)\n"
00007FF67B9915EC lea rdx,qword ptr ds:[7FF67B99C1D0]      00007FF67B99C1D0 "Mingw-w64 runtime failure:\n"
00007FF67B99167F lea rcx,qword ptr ds:[7FF67B99C1EC]      00007FF67B99C1EC "Address %p has no image-section"
00007FF67B9916E0 lea rcx,qword ptr ds:[7FF67B99C20C]      00007FF67B99C20C "  VirtualQuery failed for %d bytes at address %p"
00007FF67B991743 lea rcx,qword ptr ds:[7FF67B99C23D]      00007FF67B99C23D "  VirtualProtect failed with code 0x%x"
00007FF67B99185E lea rcx,qword ptr ds:[7FF67B99C264]      00007FF67B99C264 "  Unknown pseudo relocation protocol version %d.\n"
00007FF67B9918E1 lea rcx,qword ptr ds:[7FF67B99C296]      00007FF67B99C296 "  Unknown pseudo relocation bit size %d.\n"
00007FF67B991957 lea rcx,qword ptr ds:[7FF67B99C2C0]      00007FF67B99C2C0 "%d bit pseudo relocation at %p out of range, targeting %p, yielding the value %p.\n"
00007FF67B99362C lea r12,qword ptr ds:[7FF67B99C351]      00007FF67B99C351 "nil)"
00007FF67B993913 lea rax,qword ptr ds:[7FF67B99C356]      00007FF67B99C356 "nan"
00007FF67B9939DF lea rax,qword ptr ds:[7FF67B99C35A]      00007FF67B99C35A "inf"
00007FF67B993AEC lea rax,qword ptr ds:[7FF67B99C35F]      00007FF67B99C35F "nity"
00007FF67B994C03 lea rbx,qword ptr ds:[7FF67B99C470]      00007FF67B99C470 "(null)"
00007FF67B995A90 lea rbx,qword ptr ds:[7FF67B99C478]      00007FF67B99C478 L"(null)"
00007FF67B995D4E lea rdx,qword ptr ds:[7FF67B99C486]      00007FF67B99C486 "NaN"
00007FF67B995DAB lea rdx,qword ptr ds:[7FF67B99C48A]      00007FF67B99C48A "Inf"
00007FF67B996325 lea rcx,qword ptr ds:[7FF67B99C600]      00007FF67B99C600 "Infinity"
00007FF67B996344 lea rcx,qword ptr ds:[7FF67B99C609]      00007FF67B99C609 "NaN"
00007FF67B99743B lea rdi,qword ptr ds:[7FF67B99C680]      00007FF67B99C680 "0123456789abcdefNaN"
00007FF67B99871D lea rdx,qword ptr ds:[7FF67B99C840]      00007FF67B99C840 "nf"
00007FF67B998739 lea rdx,qword ptr ds:[7FF67B99C843]      00007FF67B99C843 "inity"
00007FF67B998771 lea rdx,qword ptr ds:[7FF67B99C849]      00007FF67B99C849 "an"
```

Our point of interest is “Enter PIN:” (the starter) let’s double click it to see where it would reference us.

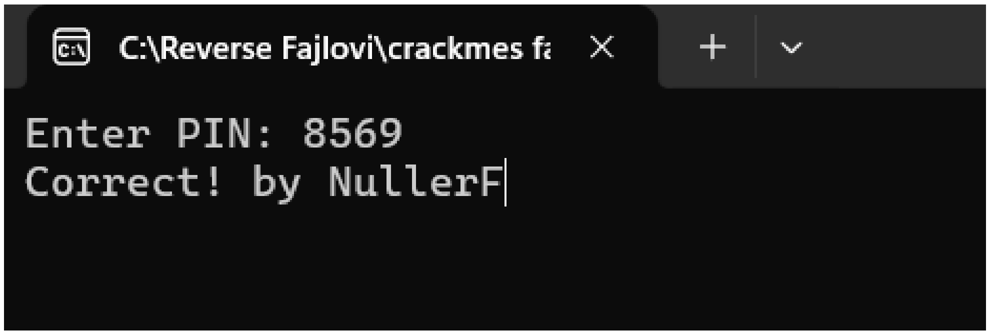


```
991384   48:83EC 30          SUB RSP, 0x30
991388   E8 DF000000         CALL easiest.7FF67B99146C
99138D   C745 FC 00000000    MOV DWORD PTR SS:[RBP - 0x4], 0x0
991394   48:8D05 65AC0000    LEA RAX, QWORD PTR DS:[0x7FF67B99C000]   rax:RemovePackageDependency+1E0, 00007FF67B99C000:"Enter PIN: "
99139B   48:89C1             MOV RCX, RAX                             rax:RemovePackageDependency+1E0
99139E   E8 6D0C0000         CALL easiest.7FF67B992010
9913A3   48:8D45 FC          LEA RAX, QWORD PTR SS:[RBP - 0x4]        rax:RemovePackageDependency+1E0
9913A7   48:8D0D 5EAC0000    LEA RCX, QWORD PTR DS:[0x7FF67B99C00C]   00007FF67B99C00C:"%d"
9913AE   48:89C2             MOV RDX, RAX                             rax:RemovePackageDependency+1E0
9913B1   E8 DA0C0000         CALL easiest.7FF67B992090
9913B6   8B45 FC             MOV EAX, DWORD PTR SS:[RBP - 0x4]
9913B9   3D 79210000         CMP EAX, 0x2179
9913BE   75 11               JNE easiest.7FF67B9913D1
9913C0   48:8D05 48AC0000    LEA RAX, QWORD PTR DS:[0x7FF67B99C00F]   rax:RemovePackageDependency+1E0, 00007FF67B99C00F:"Correct! by Null
9913C7   48:89C1             MOV RCX, RAX                             rax:RemovePackageDependency+1E0
9913CA   E8 410C0000         CALL easiest.7FF67B992010
9913CF   EB 0F               JMP easiest.7FF67B9913E0
9913D1   48:8D05 4BAC0000    LEA RAX, QWORD PTR DS:[0x7FF67B99C023]   rax:RemovePackageDependency+1E0, 00007FF67B99C023:"Incrorrect :/"
9913D8   48:89C1             MOV RCX, RAX                             rax:RemovePackageDependency+1E0
9913DB   E8 300C0000         CALL easiest.7FF67B992010
9913E0   48:8B05 11FF0000    MOV RAX, QWORD PTR DS:[<_getch>]         rax:RemovePackageDependency+1E0
9913E7   FFD0                CALL RAX                                 rax:RemovePackageDependency+1E0
9913E9   B8 00000000         MOV EAX, 0x0
9913EE   48:83C4 30          ADD RSP, 0x30
9913F2   5D                  POP RBP
9913F3   C3                  RET
```

This is it! The author didn’t specify how the crack had to be made, so we will not go by stupid/plain patching, we will try to find the value instead and change it.

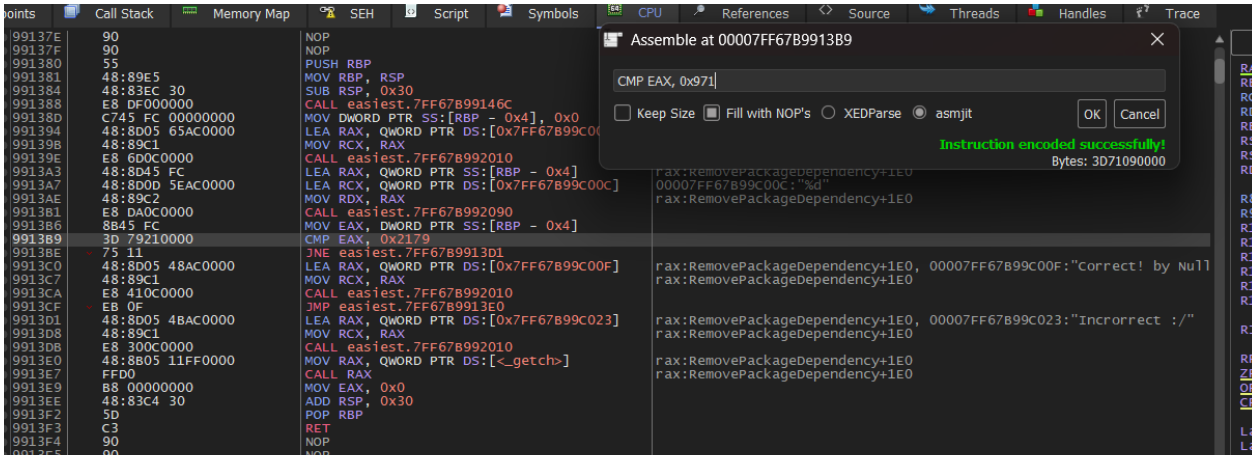
I believe if you had experience with crackmes that you already have spotted where our point of interest is.

CMP EAX, 0x2179 → if this is false the program jumps to incorrect section, since the 0x2179 is written in hex let’s convert it to dec and try to run the exe. (0x2179 in dec is 8569)



```
C:\Reverse Fajlovi\crackmes fa
Enter PIN: 8569
Correct! by NullerF
```

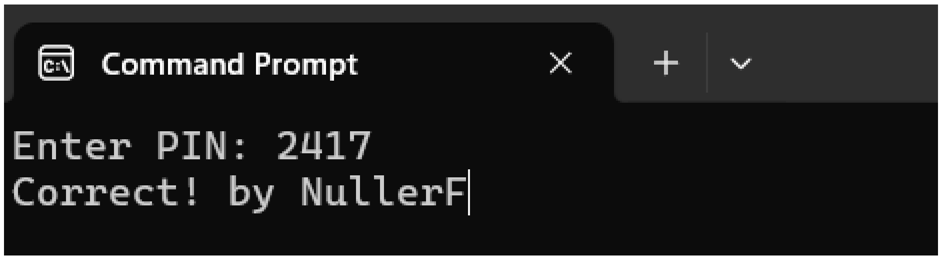
We did it! But – let's try to change the value in disassembly to 0x971.



```
Assemble at 00007FF67B9913B9
CMP EAX, 0x971
[ ] Keep Size  [■] Fill with NOP's  ( ) XEDParse  (●) asmjit
Instruction encoded successfully!
Bytes: 3D71090000
```

```
99137E   90               NOP
99137F   90               NOP
991380   55               PUSH RBP
991381   48:89E5          MOV RBP, RSP
991384   48:83EC 30       SUB RSP, 0x30
991388   E8 DF000000      CALL easiest.7FF67B99146C
99138D   C745 FC 00000000 MOV DWORD PTR SS:[RBP - 0x4], 0x0
991394   48:8D05 65AC0000 LEA RAX, QWORD PTR DS:[0x7FF67B99C0
99139B   48:89C1          MOV RCX, RAX
99139E   E8 6D0C0000      CALL easiest.7FF67B992010
9913A3   48:8D45 FC       LEA RAX, QWORD PTR SS:[RBP - 0x4]      rax:RemovePackageDependency+1E0
9913A7   48:8D0D 5EAC0000 LEA RCX, QWORD PTR DS:[0x7FF67B99C00C] 00007FF67B99C00C:"%d"
9913AE   48:89C2          MOV RDX, RAX                           rax:RemovePackageDependency+1E0
9913B1   E8 DA0C0000      CALL easiest.7FF67B992090
9913B6   8B45 FC          MOV EAX, DWORD PTR SS:[RBP - 0x4]
9913B9   3D 79210000      CMP EAX, 0x2179
9913BE   75 11            JNE easiest.7FF67B9913D1
9913C0   48:8D05 48AC0000 LEA RAX, QWORD PTR DS:[0x7FF67B99C00F] rax:RemovePackageDependency+1E0, 00007FF67B99C00F:"Correct! by Null
9913C7   48:89C1          MOV RCX, RAX                           rax:RemovePackageDependency+1E0
9913CA   E8 410C0000      CALL easiest.7FF67B992010
9913CF   EB 0F            JMP easiest.7FF67B9913E0
9913D1   48:8D05 4BAC0000 LEA RAX, QWORD PTR DS:[0x7FF67B99C023] rax:RemovePackageDependency+1E0, 00007FF67B99C023:"Incrorrect :/"
9913D8   48:89C1          MOV RCX, RAX                           rax:RemovePackageDependency+1E0
9913DB   E8 300C0000      CALL easiest.7FF67B992010
9913E0   48:8B05 11FF0000 MOV RAX, QWORD PTR DS:[<_getch>]       rax:RemovePackageDependency+1E0
9913E7   FFD0             CALL RAX                               rax:RemovePackageDependency+1E0
9913E9   B8 00000000      MOV EAX, 0x0
9913EE   48:83C4 30       ADD RSP, 0x30
9913F2   5D               POP RBP
9913F3   C3               RET
9913F4   90               NOP
```

That would be 2471 in dec.



```
Command Prompt
Enter PIN: 2417
Correct! by NullerF
```

That's it. Next thing would be to try plain patching whereas the program would prompt correct regardless of the input. But I will leave that to you. :)

Feel free to reach out on discord.

Discord: fu1gr1m