

[2] Execute the target PE file properly. How did you do that?

- 힌트로 **GetModuleFileName**, **GetFileTitle**이 있다. 제발 이전 과제와 비슷하게 파일의 이름을 바꾸면 실행되는 문제면 좋겠다.
- 먼저 FLOSS로 파일의 문자열들을 검색해서 나온 문자열 중 .exe 형식을 가진 KU-ReverseMe2.exe를 발견해서 이름을 바꿔봤다. 하지만 답이 아니었다..

1. GetModuleFileName (KERNEL.32.DLL)



```
Listing: KU-ReverseMe2-(x64)-(smrt224).exe

*****
*          POINTER to EXTERNAL FUNCTION          *
*****
DWORD  __fastcall GetModuleFileNameW(RMODULE hModule, LPW...
EAX:4   <RETURN>
RCX:8   hModule
RDX:8   lpFilename
R8D:4   nSize
637  GetModuleFileNameW <<not bound>>
PTR_GetModuleFileNameW_14001f238          XREF[4]:  FUN_140007b80:140007cc9(R),
                                           FUN_140007fb0:1400080fa(R),
                                           FUN_140009170:1400091f0(R),
                                           _configure_wide_argv:140012cd4(R...)

14001f238 4a bd 02      addr  KERNEL32.DLL::GetModuleFileNameW
00 00 00
00 00

*****
*          POINTER to EXTERNAL FUNCTION          *
*****
DWORD  __fastcall GetModuleFileNameA(RMODULE hModule, LPS...
EAX:4   <RETURN>
RCX:8   hModule
RDX:8   lpFilename
R8D:4   nSize
636  GetModuleFileNameA <<not bound>>
PTR_GetModuleFileNameA_14001f240          XREF[2]:  FUN_140001940:14000198d(R),
                                           FUN_14000aa00:14000aad3(R)

14001f240 34 bd 02      addr  KERNEL32.DLL::GetModuleFileNameA
00 00 00
00 00
```

- 일단 **GetModuleFileNameA**, **GetModuleFileNameW**의 XREF를 모아보면 아래와 같다.
 - FUN_140007b80:140007cc9
 - FUN_140007fb0:1400080fa
 - FUN_140009170:1400091f0
 - FUN_140001940:14000198d
 - FUN_14000aa00:14000aad3

2. GetFileTitle (COMDLG32.DLL)



- GetFileTitleA의 XREF는 `FUN_140001940:1400019a5` 뿐이다.

3. FUN_140001940

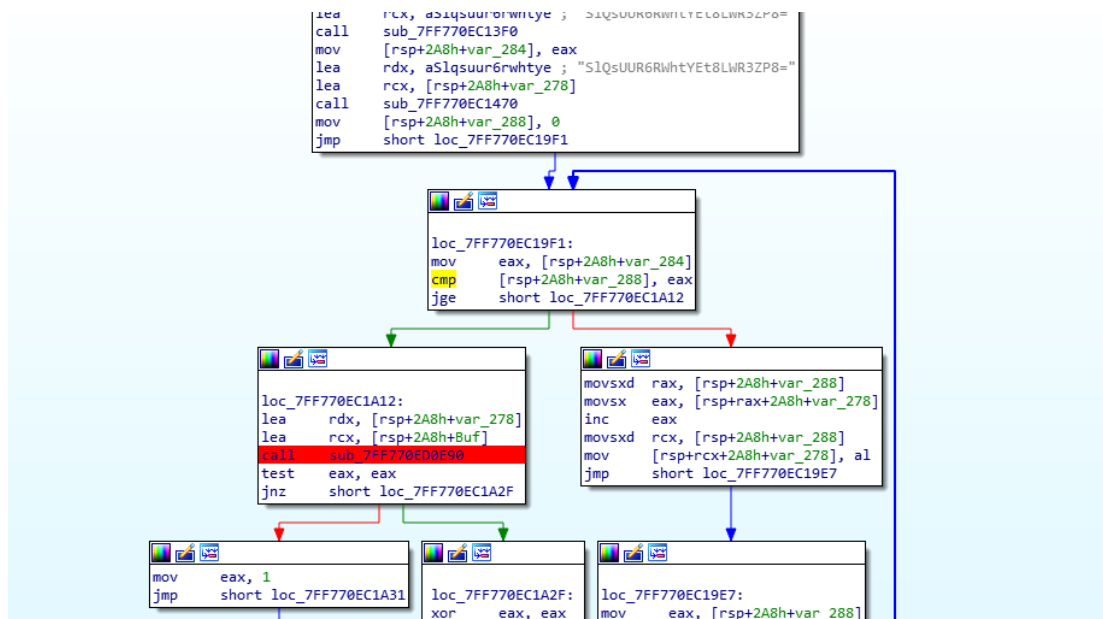
- GetModuleFileName, GetFileTitle의 공통된 함수는 `FUN_140001940` 이므로 Ghidra로 디컴파일해보겠다.

```
void FUN_140001940(void)
{
    longlong lVar1;
    CHAR *pCVar2;
    byte *pbVar3;
    undefined auStack_2a8 [32];
    int local_288;
    int local_284;
    byte local_278 [64];
    CHAR local_238 [272];
    CHAR local_128 [272];
    ulonglong local_18;

    local_18 = DAT_14002d080 ^ (ulonglong)auStack_2a8;
    pCVar2 = local_238;
    for (lVar1 = 0x104; lVar1 != 0; lVar1 = lVar1 + -1) {
        *pCVar2 = '\0';
        pCVar2 = pCVar2 + 1;
    }
    pCVar2 = local_128;
    for (lVar1 = 0x104; lVar1 != 0; lVar1 = lVar1 + -1) {
        *pCVar2 = '\0';
        pCVar2 = pCVar2 + 1;
    }
    GetModuleFileNameA((HMODULE)0x0, local_238, 0x104); //현재 실행 경로를 local_238에 담는다.
    GetFileTitleA(local_238, local_128, 0x104); // 파일의 이름을 local_128에 담는다.
    pbVar3 = local_278;
    for (lVar1 = 0x40; lVar1 != 0; lVar1 = lVar1 + -1) {
        *pbVar3 = 0;
        pbVar3 = pbVar3 + 1;
    }
    local_284 = FUN_1400013f0((byte *)s_SlQsUUR6RWhYEt8LWR3ZP8=_14002d000);
    FUN_140001470(local_278, (byte *)s_SlQsUUR6RWhYEt8LWR3ZP8=_14002d000);
    for (local_288 = 0; local_288 < local_284; local_288 = local_288 + 1) {
        local_278[local_288] = local_278[local_288] + 1;
    }
    strcmp(local_128, (char *)local_278);
    FUN_14000b800(local_18 ^ (ulonglong)auStack_2a8);
}
```

```
return;
}
```

- `local_278`=파일 경로, `local_128`=파일 이름
- 마지막 부분에서 `local_128` 과 `local_278` 을 비교하는 것을 보아, 이 값이 0이 되면 파일이 실행되는 것이 아닐까?
 - 그래서 일단 `local_278` 이 뭔지를 알아야겠다.
 - 따라서 IDA로 `strcmp`부분에 중단점을 걸고 디버깅을 해보겠다.



General registers	
RAX	0000000000000013
RBX	000000000000000A
RCX	000000B1918FFBA0 → Stack[0000041C]:000000B1918FFBA0
RDX	000000B1918FFA50 → Stack[0000041C]:000000B1918FFA50

- `RCX=0XB1918FFBA0`

```
000000B1918FFBA0 4B 55 2D 52 65 76 65 72 73 65 4D 65 32 2D 28 78 KU-ReverseMe2-(x
000000B1918FFB80 36 34 29 2D 28 73 6D 72 74 32 32 34 29 2E 65 78 64)-(smrt224).ex
000000B1918FFBC0 65 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 e.....
```

- `RDX=0XB1918FFA50`

```
000000B1918FFA50 4B 55 2D 52 45 78 46 69 6E 61 4C 7D 2E 65 78 65 KU-RE{Final}.exe
```

- `RCX` 는 현재 파일의 이름, `RDX` 는 비교할 문자열이다. 따라서 현재 파일의 이름을 `KU-RE{Final}.exe`로 바꿔보겠다. 성공했다!

4. 더 자세히 보기 FUN_1400018990

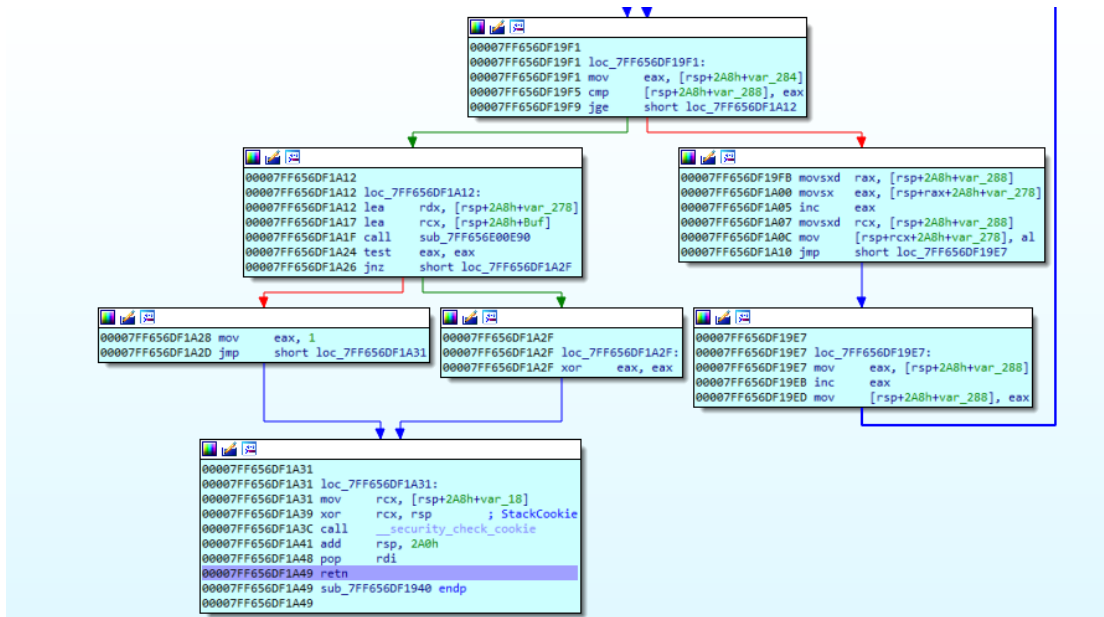
- 파일의 이름을 바꾸면 된다는 것은 알게 되었다. 그런데 정확한 알고리즘은 어떻게 되는 것일까?
- `FUN_140001940` 의 앞뒤를 살펴본 결과, `FUN_140001890` 이 **XREF** 함수임을 알게 됐다.
- 따라서 이 **XREF** 함수를 들어가보자.

```

C:\Decompile: FUN_140001890 - (KU-RE{Final}.exe)
1
2 undefined8 FUN_140001890(undefined8 param_1)
3
4 {
5     int iVar1;
6     HWND pHVar2;
7     undefined8 local_48 [9];
8
9     iVar1 = FUN_140001940();
10    if (iVar1 == 0) {
11        FUN_140010934(1);
12    }
13    CoInitialize((LPVOID)0x0);
14    FUN_140003990(4);
15    FUN_140004bd0((longlong)&DAT_14002df00,0,param_1,(undefined *)0x0);
16    FUN_140001fb0(local_48);
17    pHVar2 = GetActiveWindow();
18    FUN_1400045a0((longlong)local_48,pHVar2,0);
19    FUN_140002640((longlong)local_48);
20    FUN_140007730((longlong)&DAT_14002df00);
21    CoUninitialize();
22    return 0;
23}
24

```

- if문에서 `FUN_140001940` 의 반환값이 0이면, `FUN_140010934` 를 실행시키고 있다.
- 여기서 `FUN_140001940` 의 반환값은 어떻게 결정이 될까?



FUN_140001940의 부분

- 맨 위 cmp과정에서 `eax ≥ 0` 이 되면, 둘째 줄 왼쪽으로 내려간다.
- 내려가고서는 `eax = 0` 이여야 셋째줄 맨 왼쪽으로 내려간다.
- 그리고, `eax = 1`을 넣고, 맨 아래로 내려가 결국 `FUN_140001940`의 반환값으로 1을 반환하게 된다.
- 즉, cmp결과인 `eax = 0` 이면 1을 반환, cmp결과인 `eax > 0` 이면 0을 반환하게 된다.
- 따라서 우리는 cmp 결과가 0이므로, `FUN_140001940`의 반환값으로 1을 반환한다.
- 이제, `FUN_140001890`에서 `iVar1 = FUN_140001940 = 1`임을 알게 되었고, if문을 뛰어넘어서 `FUN_140010934`를 실행하지 않게 되었다.
- 따라서 만약 파일의 이름이 **KU-RE{Final}**이 아니면, `FUN_140010934`이 실행되면서, exe파일 실행 과정에 문제가 생기지 않았을까 추측해본다.
- `FUN_140010934`에 자세히 들어가보니, exit함수들이 실행되는 것을 발견했다.
- 따라서 파일의 이름이 제대로 되지 않으면, exit함수가 이전에 실행되면서 메시지박스가 띄워지지 않는 것이었다.

결과

파일의 이름을 **KU-RE{Final}.exe**로 바꾸고, 실행시키면 메시지박스가 뜬다.

KU-ReverseMe2

Name

Student ID

Find a valid flag

Flag

Unverified