Codeengn Challenges Advance RCE LEVEL4 풀이

Reverse2 L04 Start

Author : LibertyorDeath

Korea :

Name이 CodeEngn 일때 Serial은 무엇인가

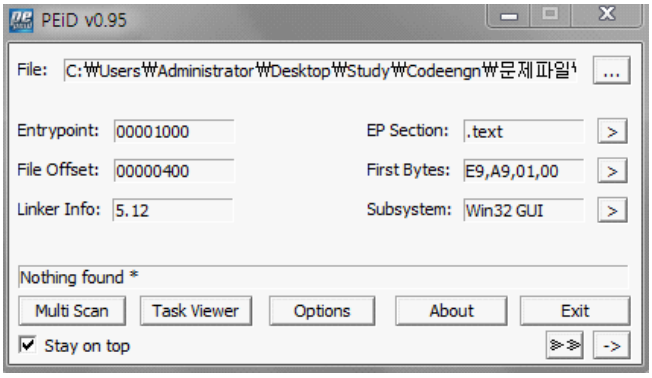
English :

Find the Serial when the Name is CodeEngn

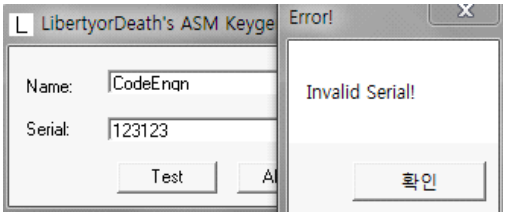
Down

PEID로 프로그램에 대한 정보를 확인해보니, 아무것도 확인되지 않았다,

파일 크기가 무지 작은걸 보니 아마 자체 패킹 된 것 같다.

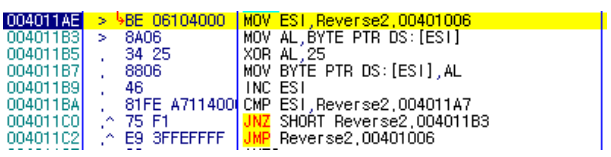


프로그램을 실행시켜 어떤 프로그램인지 확인해보니 이전 문제와 같이 Name과 Serial을 입력해 확인을 하는 프로그램 이었다.



이제 올리로 분석해 보자 .

처음부터 004011AE 주소 부분으로 점프를 해주고있다. 그 주소로 점프를 하면,



ESI 주소 시작부분(00401006)부터 한바이트씩 25와 xor해 언패킹을 하고있는 것을 볼 수가있다.
저 루프가 다 끝난 후 401006 지점에 가보면

```
Address   Hex dump     Disassembly                          Comment
00401000  $- E9 A9010000 JMP Reverse2.004011AE
00401005     90        NOP
00401006  > 6A 00      PUSH 0
00401008  ? E8 C1010000 CALL <JMP.&kernel32.GetModuleHandleA>
0040100D  ? A3 BC304000 MOV DWORD PTR DS:[4030BC],EAX
00401012  ? 6A 00      PUSH 0
00401014  ? 68 36104000 PUSH Reverse2.00401036
00401019  ? 6A 00      PUSH 0
0040101B  . 68 E9030000 PUSH 3E9
00401020  ? FF35 BC3040 PUSH DWORD PTR DS:[4030BC]
00401026  ? E8 B5010000 CALL <JMP.&user32.DialogBoxParamA>
0040102B  ? A3 24314000 MOV DWORD PTR DS:[403124],EAX
00401030  ? 50          PUSH EAX
00401031  ? E8 92010000 CALL <JMP.&kernel32.ExitProcess>
00401036  ? 55          PUSH EBP
00401037  ? 8BEC        MOV EBP,ESP
00401039  . 817D 0C 1001 CMP DWORD PTR SS:[EBP+C],110
```

이전에는 볼 수 없었던 API들이 보이기 시작한다.

여기서 감을 잡을 수 없어서 맨 밑에 문자열을 비교하는 jmp lstrcmpA 부분에 bp를 걸어놓고 프로그램을 진행해 Nanme에 CodeEngn과 Serial에 123123을 넣고 Check를 누른 후 스택부분을 조사하였다.

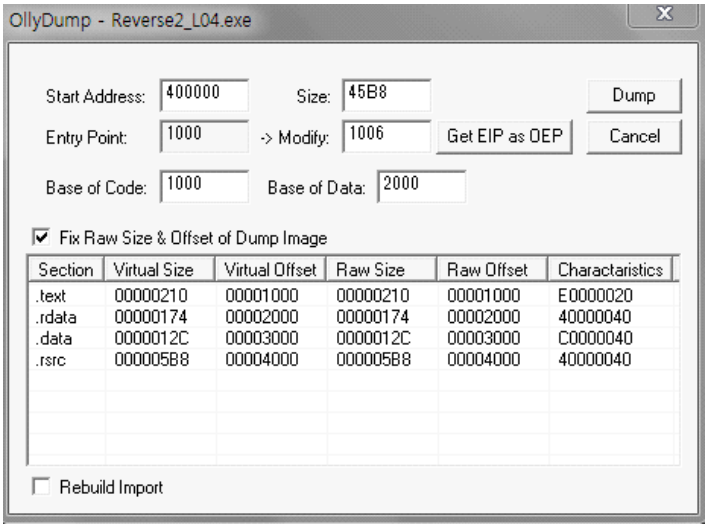
```
004011C8  ,- FF25 0820400( JMP DWORD PTR DS:[<&kernel32.ExitProces> kernel32.ExitProcess
004011CE  ,- FF25 0020400( JMP DWORD PTR DS:[<&kernel32.GetModuleHa> kernel32.GetModuleHandleA
00401104  ,- FF25 0420400( JMP DWORD PTR DS:[<&kernel32.lstrcmpA>]  kernel32.lstrcmpA
004011DA  ,- FF25 1020400( JMP DWORD PTR DS:[<&user32.wsprintfA>]   user32.wsprintfA
004011E0  ,- FF25 1420400( JMP DWORD PTR DS:[<&user32.DialogBoxPar> user32.DialogBoxParamA
004011E6  ,- FF25 1820400( JMP DWORD PTR DS:[<&user32.EndDialog>]   user32.EndDialog
004011EC  ,- FF25 1C20400( JMP DWORD PTR DS:[<&user32.GetDlgItem>]  user32.GetDlgItem
004011F2  ,- FF25 2020400( JMP DWORD PTR DS:[<&user32.GetDlgItemTe> user32.GetDlgItemTextA
004011F8  ,- FF25 2420400( JMP DWORD PTR DS:[<&user32.LoadIconA>]   user32.LoadIconA
004011FE  ,- FF25 2820400( JMP DWORD PTR DS:[<&user32.MessageBoxA>] user32.MessageBoxA
00401204  ,- FF25 2C20400( JMP DWORD PTR DS:[<&user32.SendMessageA> user32.SendMessageA
0040120A  ,- FF25 3020400( JMP DWORD PTR DS:[<&user32.SetFocus>]    user32.SetFocus
```

```
0012FA80  0012FA94
0012FA84  0012FAA4
0012FA88  00401165 Reverse2.00401165
0012FA8C  00403104 ASCII "L0D-59919-A0024900"
0012FA90  00401177 CALL to lstrcmpA
0012FA94  00403104 String1 = "L0D-59919-A0024900"
0012FA98  004030E0 String2 = "123123"
0012FA9C  0012FB20
```

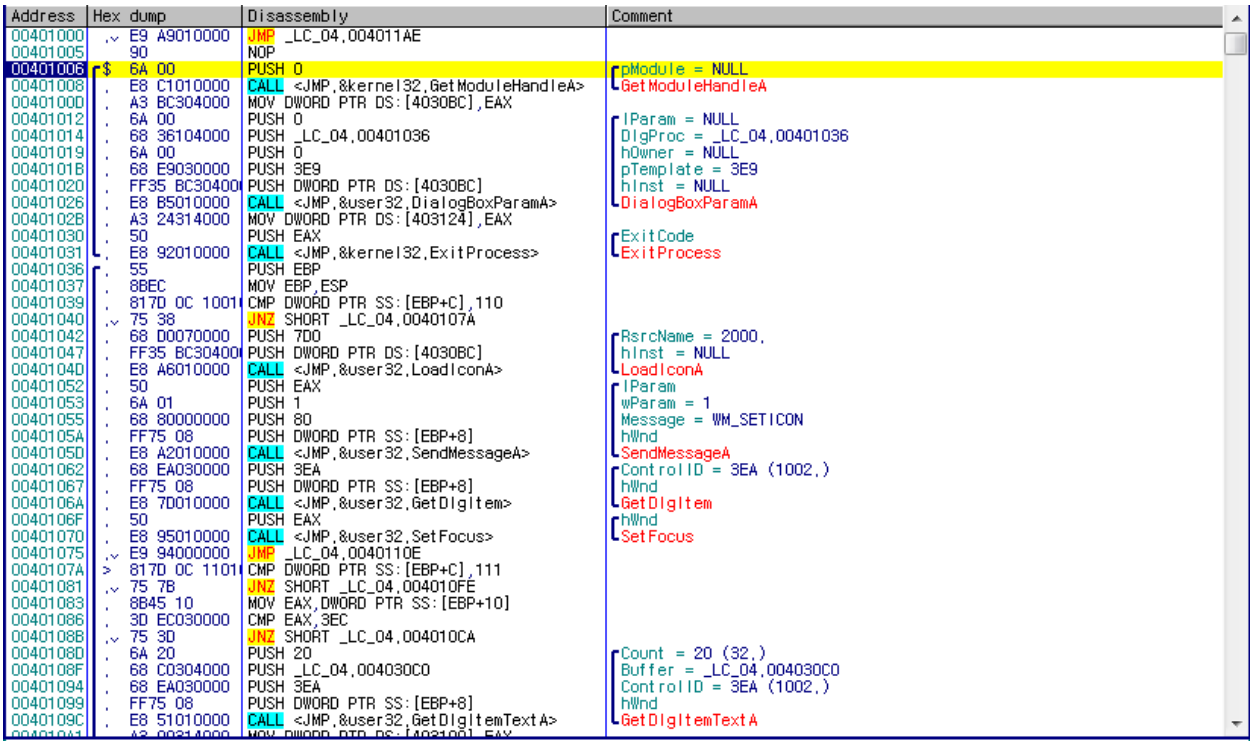
내가 넣어준 123123과 어떤 문자열을 서로 비교하는 것을 볼수가 있었다.
나는 저기서 답을 얻었다,

이 문제를 풀고 다른 사람의 풀이법을 봤는데 OLLY DUMP와 import 함수 복구 툴을 이용해 OEP를 바꾸어서 언패킹 하여 문제를 푼사람도 있었었다.

그 방법은 다음과 같았다.
먼저 1006 지점까지 프로그램을 실행시켜 언팩을 진행시킨 후



OLLY DUMP를 이용해 Rebuild Import를 해제 한 후 DUMP를 뜨고,
Import 복구 툴로 import를 복구한 결과물을 다시 올리로 열어보면,



```
Address   Hex dump          Disassembly                          Comment
00401000  .v E9 A9010000    JMP _LC_04,004011AE
00401005    90              NOP
00401006  r$ 6A 00          PUSH 0                               pModule = NULL
00401008  .  E8 C1010000    CALL <JMP,&kernel32,GetModuleHandleA>  GetModuleHandleA
0040100D  .  A3 BC304000    MOV DWORD PTR DS:[4030BC],EAX
00401012  .  6A 00          PUSH 0                               lParam = NULL
00401014  .  68 36104000    PUSH _LC_04,00401036                 DlgProc = _LC_04,00401036
00401019  .  6A 00          PUSH 0                               hOwner = NULL
0040101B  .  68 E9030000    PUSH 3E9                             pTemplate = 3E9
00401020  .  FF35 BC30400(  PUSH DWORD PTR DS:[4030BC]           hInst = NULL
00401026  .  E8 B5010000    CALL <JMP,&user32,DialogBoxParamA>   DialogBoxParamA
0040102B  .  A3 24314000    MOV DWORD PTR DS:[403124],EAX
00401030  .  50             PUSH EAX                             ExitCode
00401031  L. E8 92010000    CALL <JMP,&kernel32,ExitProcess>     ExitProcess
00401036  r. 55             PUSH EBP
00401037  .  8BEC           MOV EBP,ESP
00401039  .  817D 0C 1001(  CMP DWORD PTR SS:[EBP+C],110
00401040  .v 75 38          JNZ SHORT _LC_04,0040107A
00401042  .  68 D0070000    PUSH 7D0                             RsrcName = 2000,
00401047  .  FF35 BC30400(  PUSH DWORD PTR DS:[4030BC]           hInst = NULL
0040104D  .  E8 A6010000    CALL <JMP,&user32,LoadIconA>         LoadIconA
00401052  .  50             PUSH EAX                             lParam
00401053  .  6A 01          PUSH 1                               wParam = 1
00401055  .  68 80000000    PUSH 80                              Message = WM_SETICON
0040105A  .  FF75 08        PUSH DWORD PTR SS:[EBP+8]            hWnd
0040105D  .  E8 A2010000    CALL <JMP,&user32,SendMessageA>      SendMessageA
00401062  .  68 EA030000    PUSH 3EA                             ControlID = 3EA (1002,)
00401067  .  FF75 08        PUSH DWORD PTR SS:[EBP+8]            hWnd
0040106A  .  E8 7D010000    CALL <JMP,&user32,GetDlgItem>        GetDlgItem
0040106F  .  50             PUSH EAX                             hWnd
00401070  .  E8 95010000    CALL <JMP,&user32,SetFocus>          SetFocus
00401075  .v E9 94000000    JMP _LC_04,0040110E
0040107A  > 817D 0C 1101(  CMP DWORD PTR SS:[EBP+C],111
00401081  .v 75 7B          JNZ SHORT _LC_04,004010FE
00401083  .  8B45 10        MOV EAX,DWORD PTR SS:[EBP+10]
00401086  .  3D EC030000    CMP EAX,3EC
0040108B  .v 75 3D          JNZ SHORT _LC_04,004010CA
0040108D  .  6A 20          PUSH 20                              Count = 20 (32,)
0040108F  .  68 C0304000    PUSH _LC_04,004030C0                 Buffer = _LC_04,004030C0
00401094  .  68 EA030000    PUSH 3EA                             ControlID = 3EA (1002,)
00401099  .  FF75 08        PUSH DWORD PTR SS:[EBP+8]            hWnd
0040109C  .  E8 51010000    CALL <JMP,&user32,GetDlgItemTextA>   GetDlgItemTextA
```

다음과 같이 더 분석 하기 쉬운 프로그램의 상태를 볼 수가 있다.