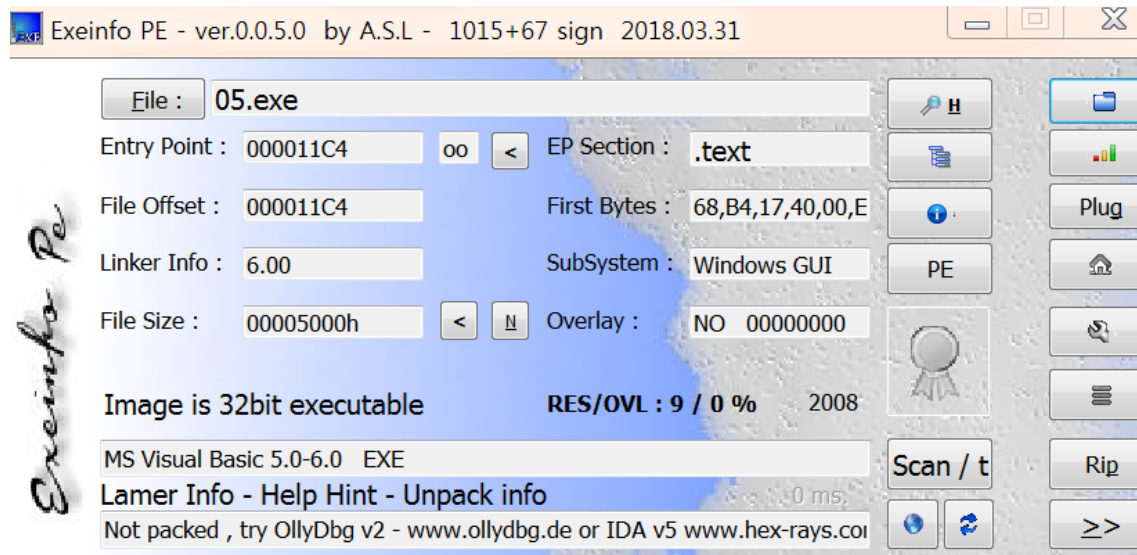
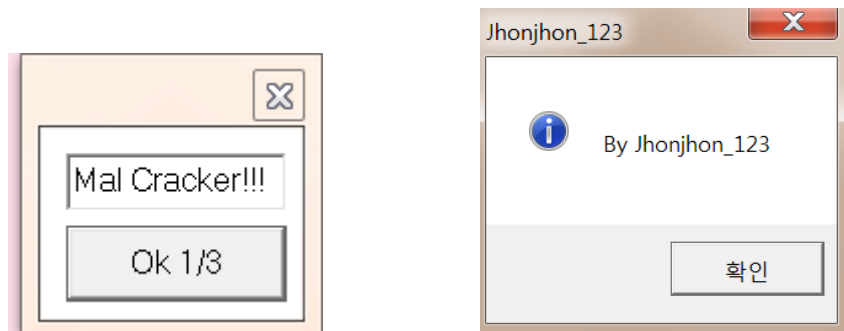


코드 엔진 Challenges: Advance 05

Author: Pass Corta
Korean: Serial 은 무엇인가

문제를 확인하고 프로그램을 실행시키자 빈 칸이 나와 '5'를 입력하고 ok 버튼을 누르자 다음과 같은 메시지가 나오고 입력값 5는 'Mal Cracker!!' 라는 메시지로 바뀌었다.
아무래도 틀린 시리얼 값을 입력하면 "MAL Cracker!!"라는 메시지로 바뀌는 것 같다.



PEID로 패킹 여부를 확인해보자 아무런 정보를 얻을 수 없었다. 올리디버거로 분석해보자.

문제가 Serial을 찾는 것이고 지금까지 Serial을 찾을 때 사용자의 입력값과 원래의 시리얼을

비교한 후 에 점프하여 메시지를 출력하는 형식이므로 search for를 통해 cmp 관련 함수를 찾아보자.

```

004023F7 CALL DWORD PTR DS:[&MSVBVM60.__vbaObjSet MSVBVM60.__vbaObjSet
00402438 CALL DWORD PTR DS:[&MSVBVM60.__vbaObjSet MSVBVM60.__vbaObjSet
00402625 CALL DWORD PTR DS:[&MSVBVM60.__vbaObjSet MSVBVM60.__vbaObjSet
00402670 CALL DWORD PTR DS:[&MSVBVM60.__vbaObjSet MSVBVM60.__vbaObjSet
004026BA CALL DWORD PTR DS:[&MSVBVM60.__vbaObjSet MSVBVM60.__vbaObjSet
00402476 CALL DWORD PTR DS:[&MSVBVM60.__vbaStrCmp MSVBVM60.__vbaStrCmp
00402372 CALL DWORD PTR DS:[&MSVBVM60.__vbaStrVarVal MSVBVM60.__vbaStrVarVal
004022AD CALL DWORD PTR DS:[&MSVBVM60.__vbaVarCat MSVBVM60.__vbaVarCat
00402319 CALL DWORD PTR DS:[&MSVBVM60.__vbaVarCat MSVBVM60.__vbaVarCat

```

예상했던 대로 vbaStrcmp 함수가 보인다. 이 함수의 코드 부분으로 들어가 분기문이 있는지 확인해보자 .

```

00402446 . 52      PUSH EDI
00402447 . 53      PUSH EBX
00402448 . 8B0B    MOV ECX,DWORD PTR DS:[EBX]
0040244A . FF91 A0000000 CALL DWORD PTR DS:[ECX+A0]
00402450 . 3BC7    CMP EAX,EDI
00402452 . DBE2    JGE ECX
00402454 . 7D 12    JGE SHORT 05.00402468
00402456 . 68 00000000 PUSH 0
0040245B . 68 4C1F4000 PUSH 05.00401F4C
00402460 . 53      PUSH EBX
00402461 . 50      PUSH EAX
00402462 . FF15 20104000 CALL DWORD PTR DS:[&MSVBVM60.__vbaHResult MSVBVM60.__vbaHResultCheck0bj
00402468 . 8B85 64FFFFFF MOV EAX,DWORD PTR SS:[EBP-9C]
0040246E . 8B8D 68FFFFFF MOV ECX,DWORD PTR SS:[EBP-98]
00402474 . 50      PUSH EAX
00402475 . 51      PUSH ECX
00402476 . FF15 44104000 CALL DWORD PTR DS:[&MSVBVM60.__vbaStrCmp MSVBVM60.__vbaStrCmp

```

분기문역시 보인다 .이 부분에 BP를 걸고 분석을 해보도록 하자. *JGE (>=)

```

00402446 . 52      PUSH EDI
00402447 . 53      PUSH EBX
00402448 . 8B0B    MOV ECX,DWORD PTR DS:[EBX]
0040244A . FF91 A0000000 CALL DWORD PTR DS:[ECX+A0]
00402450 . 3BC7    CMP EAX,EDI
00402452 . DBE2    JGE ECX
00402454 . 7D 12    JGE SHORT 05.00402468
00402456 . 68 00000000 PUSH 0
0040245B . 68 4C1F4000 PUSH 05.00401F4C
00402460 . 53      PUSH EBX
00402461 . 50      PUSH EAX
00402462 . FF15 20104000 CALL DWORD PTR DS:[&MSVBVM60.__vbaHResultCheck0bj MSVBVM60.__vbaHResultCheck0bj
00402468 . 8B85 64FFFFFF MOV EAX,DWORD PTR SS:[EBP-9C]
0040246E . 8B8D 68FFFFFF MOV ECX,DWORD PTR SS:[EBP-98]
00402474 . 50      PUSH EAX
00402475 . 51      PUSH ECX
00402476 . FF15 44104000 CALL DWORD PTR DS:[&MSVBVM60.__vbaStrCmp MSVBVM60.__vbaStrCmp
0040247E . 8B0B    MOV EBX,ECX
00402484 . F7DB    NEG EBX
00402486 . 1BDB    SBB EBX,EBX
00402488 . 0085 64FFFFFF LEA EAX,DWORD PTR SS:[EBP-9C]
0040248E . 52      PUSH EDI
0040248F . 43      INC EBX
00402490 . 50      PUSH EAX
00402491 . 6A 02    PUSH 2
00402493 . F7DB    NEG EBX
EAX=00224E6C, (UNICODE "677345")

```

-00402454에 bp를 걸고 실행을 한 결과 EAX와 EDI의 값을 비교후에 >= 으므로 00402468로 분기함

-00402468 EBP-9C의 4바이트 값을 복사해 EAX에 저장함 (사용자가 입력한 값)

```

Stack SS:[0012F401]=00264E6C, (UNICODE "677345")
EAX=00000000
Jump from 00402454

```

-0040246E EBP-98의 4바이트 값을 복사해 ECX에 저장함 (Serial값)

```

Stack SS:[0012F404]=00265F7C, (UNICODE "677345")
ECX=778D6570 (ntdll.778D6570)

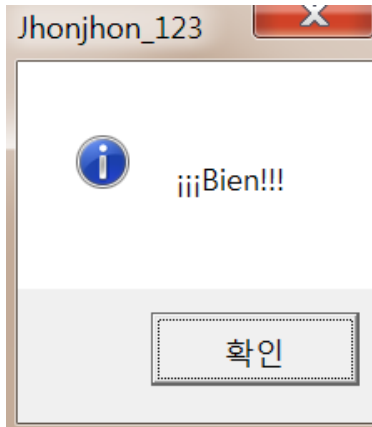
```

-00402474 EAX 값 push

-00402475 ECX 값 push

-00402476 vbaStrCmp 함수를 이용하여 EAX값과 ECX 값 비교 한다 값이 같으면 계속 진행

을 하다가 00402544에서 성공 메시지 값 출력



답 :677345