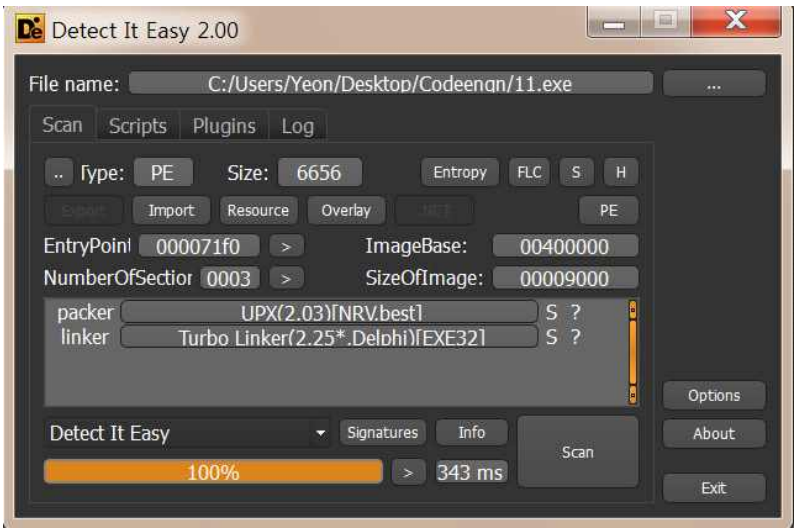
코드 엔진 Challenges: Basic 11

Author:abex
Korean: OEP를 찾으시오 . EX)00401000
stolenbyte를 찾으시오 . Ex)FF35CA204000E84000000
정답인증은 OEP+StolenByte
Ex)00401000FF35CA204000E84000000

문제를 보면 StolenByte가 쓰이고 패킹되어있다는 것을 알 수 있다. 먼저 DE를 통해 확인해보자.



UPX로 패킹되어 있는 것을 알 수 있다. StolenByte되어있으니 UPX를 이용해 언패킹을 해도 오류가 날것이다 .오류를 해결하기 위해 먼저 StolenByte부터 찾아보자.

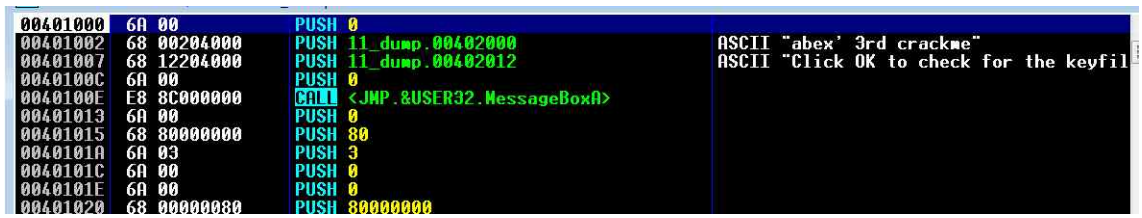
```
0040736C   . 58              POP EAX
0040736D   . 61              POPAD
0040736E   . 6A 00           PUSH 0
00407370   . 68 00204000     PUSH 11.00402000                ASCII "abex' 3rd crackme"
00407375   . 68 12204000     PUSH 11.00402012                ASCII "Click OK to check for the keyfile."
0040737A   . 8D4424 80       LEA EAX,DWORD PTR SS:[ESP-80]
0040737E   > 6A 00           PUSH 0
00407380   . 39C4            CMP ESP,EAX
00407382   .^75 FA           JNZ SHORT 11.0040737E
00407384   . 83EC 80         SUB ESP,-80
00407387   .-E9 809CFFFF     JMP 11.0040100C
```

이전 10번 문제에서 사용했던 ESP 주소에 BP를 설정하는 방식을 이용해서 OEP 분기전으로 이동해보았다. 이동을하니 분기문이 보이고 OEP는 0040100C라는 것을 알 수 있다.
또 POPAD 아래에 12Byte가 Stolenbyte라는 것을 알 수 있다.

```
00401008   90              NOP
00401009   90              NOP
0040100A   90              NOP
0040100B   90              NOP
0040100C   6A 00           PUSH 0
0040100E   E8 8C000000     CALL 11.0040109F                JMP to USER32.MessageBoxA
00401013   6A 00           PUSH 0
00401015   68 80000000     PUSH 80
0040101A   6A 03           PUSH 3
```

확인을 위해서 이동해보니 위와 같이 메시지 박스함수를 출력하는 것도 알 수 있고 예상했던대로 OEP 위의 코드가 비어있는 것 또한 알 수 있다. 살펴보니 MessageBoxA 함수에 필요한 인자들이 비어있는 것도 확인 할 수 있다. 비어있는 코드를 입력해보자.

여기서 훔쳐진 코드를 입력해주기었기에 OEP 코드도 00401000이 된다.



정상적으로 언패킹된 것을 확인할 수 있다.

문제의 답읍 004010006A00680020400068122040000이다