

# **Reversing Engineering Contest – Hackers’ Dream REVERSING #1,#2**

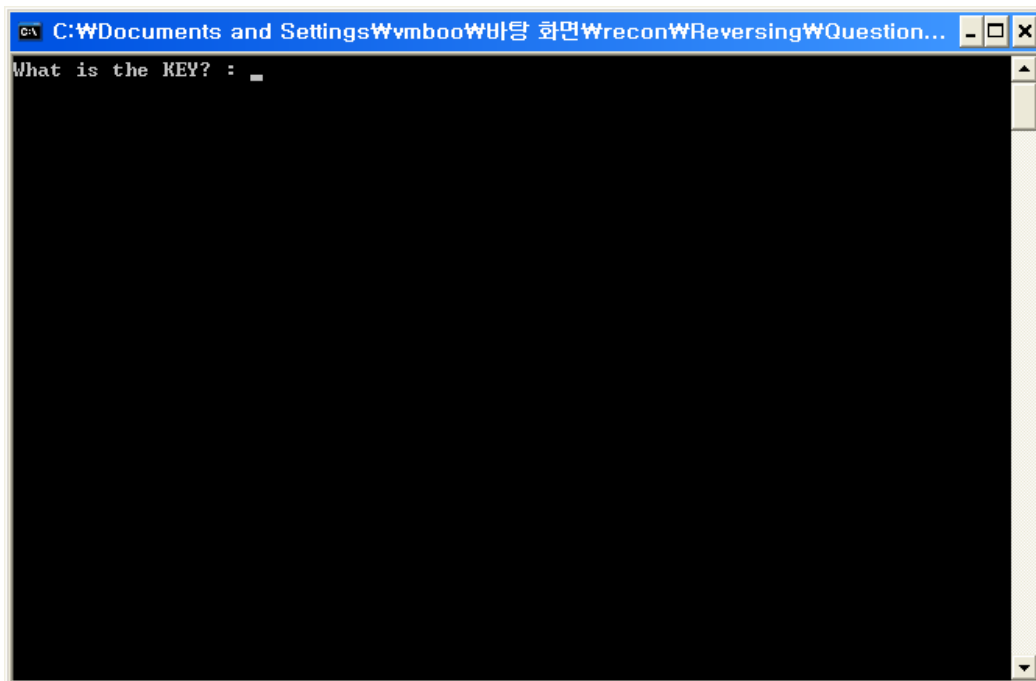
HACKING GROUP “OVERTIME”  
mrboo< [bsh7983@gmail.com](mailto:bsh7983@gmail.com) > 2008.11.12

이번 POC에서 개최한 리버싱 컨테스트중 리버싱부분에 대해 문서를 만들어 보았습니다. 아래 풀이는 활동하고 있는 스터디그룹에서 발표한 내용입니다. 개인적인 문서라 존칭은 생략을 하였습니다.

## REVERSING 1번 문제 풀이

### 1. 프로그램 분석(직접 실행)

우선 분석전에 이 프로그램이 무슨 일을 하는지 파악하기 위해 vmware에서 동작을 시켜본다.



위와 같이 KEY를 물어보는 창이 뜨며, 잘못된 KEY를 넣었을때 "You're wrong !! "이란 메시지가 나온다. 이로 미루어 KEY를 찾는 문제인듯 하다.

### 2. 패킹여부 확인(PE tools)

아래는 PEID툴로 확인해본 화면이다. 섹션이름이나 여러가지로 팩킹되었다는걸 알수 있다. UpolyX v0.5라고 나온다. 전에 했을때는 안나오더니 문서작성하려고 다른 PC서 돌려보니 나온다. PEID버전이 다른가 보다;; google에서 UpolyX에 대한 MUP문서를 구할수 있었으나 패턴이 다른걸 알수 있었다. 뭐 어쨌든 다른 방식으로 MUP를 해보았다.



### <최초 로딩시 화면>

F8로 트레이싱해보면 아래와 같은 코드에서 루프가 발생함을 알 수 있다.

```
0040C09C  B8 00104000    mov eax, Question.00401000
0040C0A1  3D 00704000    cmp eax, <&KERNEL32.Sleep>
0040C0A6  74 06          je short Question.0040C0AE
0040C0A8  8030 A6        xor byte ptr ds:[eax], 0A6
0040C0AB  40             inc eax
0040C0AC  EB F3         jmp short Question.0040C0A1
```

처음에 00401000을 eax에 넣는다.(0040C09C)

401000은 보통 exe파일의 바이너리 text섹션주소이다.

위 루프를 돌면서 Olly에서 보여주는 내용을 보면 401000부터 407000(KERNEL32.Sleep주소가 407000으로 되었음)까지 1바이트씩 증가를 시키며 0A6과 xor시킨다.

그리고 다음코드를 보면

```
0040C0AE  B8 00804000    mov eax, Question.00408000
0040C0B3  3D 00B04000    cmp eax, Question.0040B000
0040C0B8  74 06          je short Question.0040C0C0
0040C0BA  8030 A2        xor byte ptr ds:[eax], 0A2
0040C0BD  40             inc eax
0040C0BE  ^ EB F3        jmp short Question.0040C0B3
```

이역시 408000부터 40B000까지 하나씩 주소를 증가시키며 주소가 가리키는 1byte문자를 0A2와 xor시키게 된다.

위의 두가지 패턴은 패킹된 영역을 복호화시켜주는 코드임을 알 수 있다.

복호화를 시켜주고 retn을 한다. retn명령어는 현재 스택에 최상위에 있는 주소를 EIP에 pop을 하고 JMP EIP를 하기 때문에

retn상태서 스택의 최상위를 보면 401170이 들어있는걸 알 수 있다.

이 401170이 OEP이다.

덤프툴을 이용하여 덤프를 뜨고 IAT복구를 하게 되면 언패킹을 할 수 있을 것이다.

언패킹된 바이너리를 다시 올리로 로딩시켜보고 search for -> all referenced text strings를 선택하여 text 스트링을 보면 프로그램실행시켜서 처음 나오는 "What is the KEY??"와 "You're wrong!!"이외에도 "Good job!!"이란 스트링이 있는게 보인다.

그곳으로 이동을 해보자

```
004010A8  |. 85C0          test eax, eax
004010AA  |. 75 1A         jnz short Question.004010C6
004010AC  |. 68 40804000   push Question.00408040          ; ASCII "Good
Job !! "
004010B1  |. E8 89000000   call Question.0040113F
004010B6  |. 83C4 04       add esp, 4
004010B9  |. 68 10270000   push 2710                      ; /Timeout =
10000. ms
004010BE  |. FF15 00704000 call near dword ptr ds:[<&KERNEL32.Sleep>; WSleep
```

```

004010C4 |. EB 18          jmp short Question.004010DE
004010C6 |> 68 30804000     push Question.00408030          ; ASCII "You'r
Wrong !!"
004010CB |. E8 6F000000    call Question.0040113F
004010D0 |. 83C4 04        add esp, 4
004010D3 |. 68 10270000     push 2710                          ; /Timeout =
10000. ms
004010D8 |. FF15 00704000   call near dword ptr ds:[<&KERNEL32.Sleep>; WSleep

```

위처럼 eax가 0인지 아닌지를 판단하여 0이면 Good으로 아니면 Wrong으로 가게끔  
프로그램되어있는것을 알수 있다.

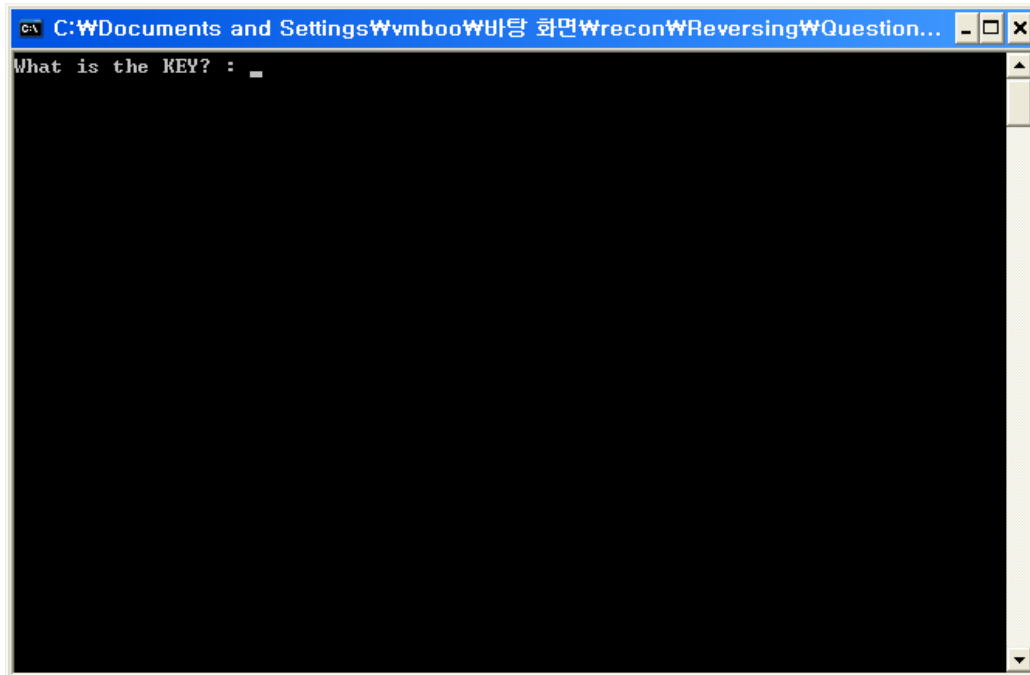
우리는 여기서 간단히 jnz를 jz 또는 jmp로 바꾸어 볼수도 있을것이다.

Question 1은 패스워드는 찾지 않았다. 여기서는 간단히 패치만 할것이다.

## REVERSING 2번 문제 풀이

### 1. 프로그램 분석(직접 실행)

우선 분석전에 이 프로그램이 무슨 일을 하는지 파악하기 위해 vmware에서 동작을 시켜본다.



question1번 문제와 동일함을 알수 있다.

### 2. 패킹여부 확인(PE tools)

아래는 PEID툴로 확인해본 화면이다. question1과 동일하다. 패킹된것으로 의심해 본다.

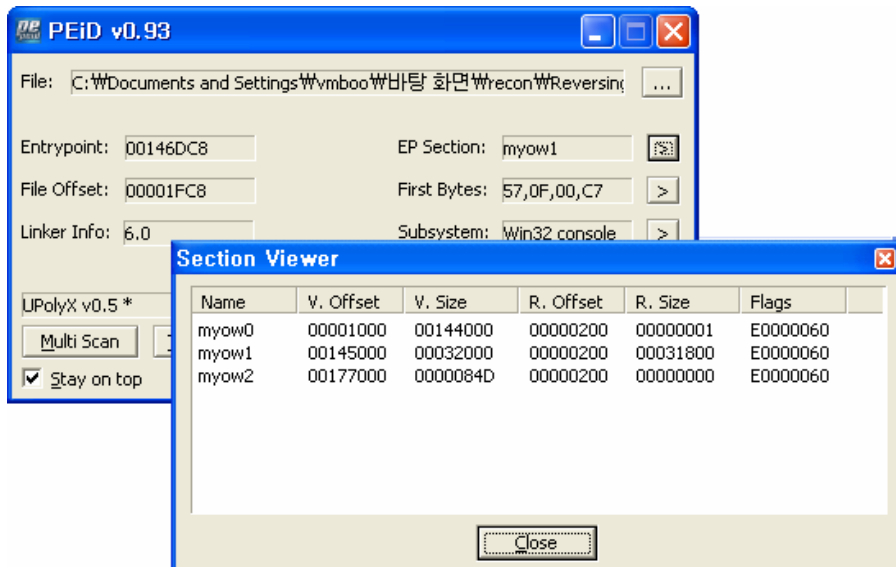
그리고 역시 전에는 안나왔던 UPolyX v0.5로 팩을 했다고 나온다.

아직 UPolyX에 대해서는 확인하지 않았다. 나중에 하겠다....

실력을 늘리려면 MUP를 해봐야....할거 같다. 어렵지만 ㅠ

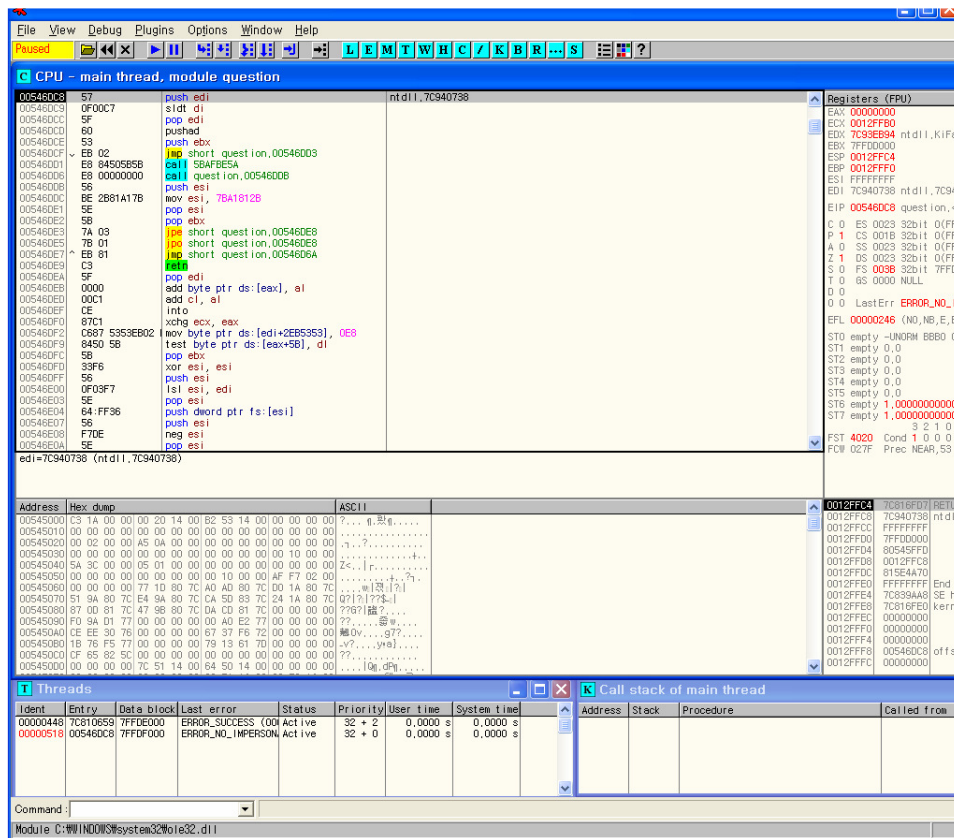
혹시라도 MUP에 관심이 있다면 art of unpacking 문서를 읽어보세요..

정말 방법도 여러가지임을 알수 있을 것이다...



### 3. 디버깅(OLLYDBG)

그럼 다음엔 올리로 이 프로그램을 로딩해보자.



### <최초 로딩시 화면>

F8로 트레이싱 해보자..

그러면

```
00546E16 313E xor dword ptr ds:[esi], edi ; ntdll.7C940738
```

여기서 exception이 발생한것을 알수 있다.

디버깅 하면서 당황스런 부분도 역시 문서에 남기도록 하겠다.

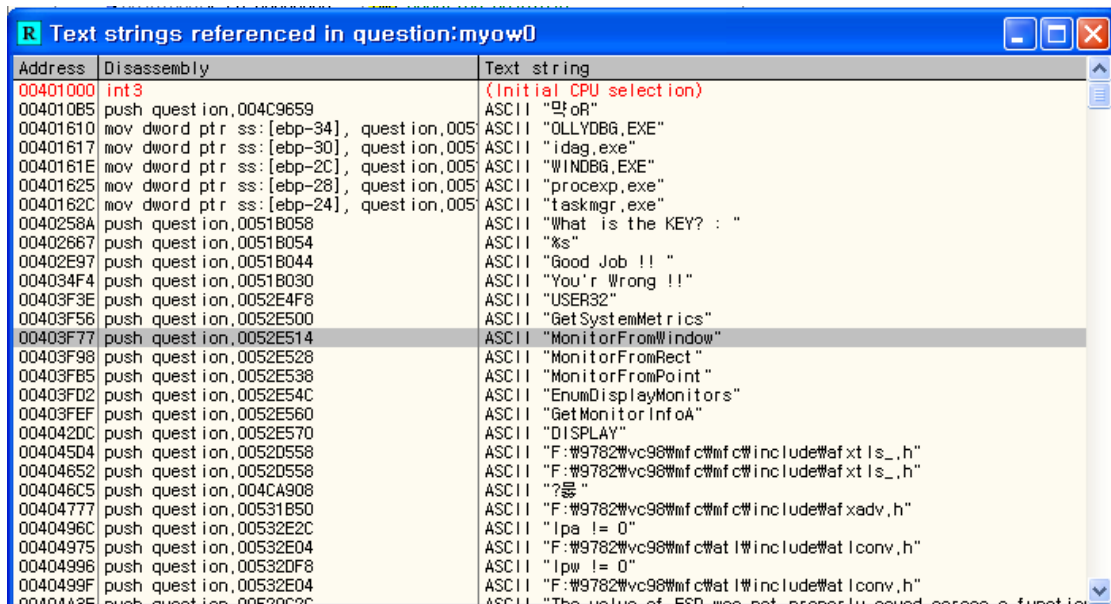
첫번째로 당황스런 부분이다.

나같은 경우 546DFD에서 xor esi, esi로 esi를 초기화 시켰으므로 이 프로그램은 당연히 무조건 exception이 발생하는 구조라고 생각을 하고 혹시 OEP로 가기전 system 로딩부분에서 다른 길로 빠져나가는 부분이 있을까라는 생각으로 올리의 디버깅옵션중 system탭에 Entry point of main module 이 곳이 아니고 System breakpoint로 설정을 해서 디버깅을 해봤으나 결국은 위의 초기화 화면으로 넘어오게 됨을 알고 디버깅을 어케 해야하나 고민을 했었다.

어떤 패커의 경우는 main 엔트리로 가기전에 실행되는 TLS영역에 코딩을 하는 경우도 있다. 그래서 요게 그건가 했는데 아닌거 같다. PEID로 TLStable부분에 내용이 없는것으로도 확인이 가능하다.

그래서 처음 올리로 로딩을 시키지 말고 프로그램을 실행을 시켜 KEY를 물어보는 창이 뜰때 attach로 로딩을 시켜보았다.

KEY??문자열이 보인다는 것은 프로그램이 메모리상에 언패킹이 되었다는 뜻이고 이 상태에서 attach를 해서 보면 언패킹된 코드를 볼수 있어 이 방식으로 했다. 우선 쉽기 때문에... 그래서 문자열을 검색(search for..)해보니 익숙한 문자가 처음 부분에 보였다.



Address	Disassembly	Text string
00401000	int 3	(Initial CPU selection)
004010B5	push question,004C9659	ASCII "막오"
00401610	mov dword ptr ss:[ebp-34], question,005	ASCII "OLLYDBG.EXE"
00401617	mov dword ptr ss:[ebp-30], question,005	ASCII "idag.exe"
0040161E	mov dword ptr ss:[ebp-2C], question,005	ASCII "WINDBG.EXE"
00401625	mov dword ptr ss:[ebp-28], question,005	ASCII "procexp.exe"
0040162C	mov dword ptr ss:[ebp-24], question,005	ASCII "taskmgr.exe"
0040258A	push question,0051B058	ASCII "What is the KEY? : "
00402667	push question,0051B054	ASCII "%s"
00402E97	push question,0051B044	ASCII "Good Job !! "
004034F4	push question,0051B030	ASCII "You'r Wrong !!"
00403F3E	push question,0052E4F8	ASCII "USER32"
00403F56	push question,0052E500	ASCII "GetSystemMetrics"
00403F77	push question,0052E514	ASCII "MonitorFromWindow"
00403F98	push question,0052E528	ASCII "MonitorFromRect"
00403FB5	push question,0052E538	ASCII "MonitorFromPoint"
00403FD2	push question,0052E54C	ASCII "EnumDisplayMonitors"
00403FEF	push question,0052E560	ASCII "GetMonitorInfoA"
004042DC	push question,0052E570	ASCII "DISPLAY"
004045D4	push question,0052D558	ASCII "F:\9782\vc98\mf\include\afx\ls_.h"
00404652	push question,0052D558	ASCII "F:\9782\vc98\mf\include\afx\ls_.h"
004046C5	push question,004CA908	ASCII "?문"
00404777	push question,00531B50	ASCII "F:\9782\vc98\mf\include\afxadv.h"
0040496C	push question,00532E2C	ASCII "lpa != 0"
00404975	push question,00532E04	ASCII "F:\9782\vc98\mf\include\atlconv.h"
00404996	push question,00532DF8	ASCII "lpw != 0"
0040499F	push question,00532E04	ASCII "F:\9782\vc98\mf\include\atlconv.h"
00404A3E	push question,00532C2C	ASCII "The value of ESP was not properly saved across a function call"



흠..OLLYDBG.EXE,idag.exe,WINDBG.EXE,procexp.exe,taskmgr.exe.....  
 우연히 올리를 띄운상태에서 question2번 문제를 실행시켜보니 올리가 종료되는것을 보았다.  
 안티디버깅 기법을 사용한것을 알수 있다. 올리디버그,IDA,WinDBG등....  
 그렇다고 올리디버거의 실행파일(ollydbg.exe)의 이름을 수정을 하여 해봐도 똑같다. ㅋㅋ  
 파일이름말고도 수정할게 있기 때문이다. EPROCESS에 클래스이름이 아마 변하지 않아서  
 그런듯 싶다. 예전에 공부했던 내용을 다시 확인하려 했지만 귀차니즘 때문에 그냥 넘어갔다.  
 이걸 EPROCESS를 한번 찾아보면 아마 답이 나오지 않을까 싶다....나도 이것저것 책만 읽고  
 정리가 안되서리;; 언제 정리를 해야되는데..

그래서 다른 디버거를 사용을 해봤으나 초기화면은 피할수 없었다.  
 열심히 방법을 찾던 중 asprotect의 mup하는 과정을 보게되었다.  
 그리고 두번째 문제에 적용을 해봤는데 너무 비슷하게 넘어가는 것이다...  
 방법은 exception이 발생을 해도 무시를 하고 shift+F9로 넘겨주어라  
 그리고 계속 exception이 발생해도 계속 s+F9로 넘겨주고 언젠가 프로그램이 쪽날때가  
 있을것이다.  
 이때를 기억해서 몇번 s+F9를 눌렀는지 기억을 하고 마지막 누르기 전까지 한번 가봐라

```
0054547E    F7F2                div edx
00545480    64:8F05 00000000>pop dword ptr fs:[0]
00545487    83C4 04             add esp, 4
```

54547E까지 왔다.  
 이때 edx는 0이므로 div(나누기)를 하여 exception이 발생함을 알수 있다.  
 그럼 이제 그 아래 545480번지에 브포를 지정하고 F9 -> s+F9를 누르자  
 그리고 계속 F8로 트레이싱을 해보자  
 조금만 실행하다 보면 search for에 text를 보면 위처럼 OLLYDBG.exe가 나오는걸 볼수 있다.  
 거의 다왔다...OEP만 찾으면 된다.  
 나 같은 경우 언패킹 패턴을 대입해봤다. 모든 플래그와 레지스터를 복구하는 popad,popfd가  
 순차적으로 나오는 OP코드를 검색을 해보았다.  

```
005457AC    61                popad
005457AD    9D                popfd
005457AE    - E9 BDFBEBFF      jmp question.00405370
```

 이 문서로는 깔끔하게 jmp문이 OEP라는 것을 확인할수 있지만 이걸 찾으려고 엄한곳에  
 덤프를 떠보구,,암튼 노가다짓을 많이 했다.ㅠㅠ  
 그렇습니다;; 위 405370으로 덤프를 뜨고 리빌딩을 시키면 프로그램이 제대로 뜸을 확인할수  
 있다.  
 또는 OEPFINDER라는 툴을 이용하면 OEP찾는데 도움이 많이 될거 같다.

#### 4. PWD찾기

이번 question2는 숨겨진 패스워드를 찾아보겠다.

언팩된 프로그램을 올리로 로드하고 F8로 트레이싱을 한다.

```
00405467 E8 ADBBFFFF call 21.00401019
```

이부분에서 프로그램이 종료가 됨을 볼수 있다.

이부분에 브포를 걸구 F7(step into)로 함수에 들어간다.

그럼 jmp문을 만날것이며 점프를 하게 되면 드디어 이곳에서 디버깅 체크를 하는 곳으로 넘어가게 된다.

이때 나는 OLLYDBG.EXE가 있는 데이터영역에 바이트를 00으로 모두 수정을 하여 이 루틴을 넘어가게 했다.

그리고 dump창에는 미리 12EF4C 주소로 이동하여 어떻게 데이터가 변하는지 확인해보자

계속 트레이싱을 하면 또 이곳에서 프로그램이 종료가 된다.

```
004016B5 E8 64F9FFFF call 21.0040101E
```

그럼 다시 이부분에 브포를 걸구 F7로 함수진입을 한다.

그럼 이부분에서 종료가 되었단걸 알수 있다.

```
0040152C . CD 2D int 2D
```

int 2D 명령은 디버거 디택션을 하는 코드로 anti reversing의 한 방법으로

[http://www.openrce.org/reference\\_library/anti\\_reversing](http://www.openrce.org/reference_library/anti_reversing) 이곳을 참고하기 바란다.

해결방법은 그 아래 코드

```
0040152E E9 db E9
```

이곳에 브포를 걸구 F9로 실행을 한다. 그럼 브포에 걸리고 다시 트레이싱을 하면 된다.

또 종료, 00401560 . E8 1B3A0000 call 21.00404F80 이부분에 브포걸구 F7로 계속 디버깅..

또 종료 또 브포걸구 F7...

```
00401773 . 50 push eax ; /Arg1 =
```

```
0051B0AC
```

```
00401774 . E8 AFF60400 call 21.00450E28 ; W21.00450E28
```

이부분에 OLLYDBG.EXE를 체크하여 ??

계속 이렇게 하면 끝이 안 날꺼 같다...

방법을 달리해보자

우선 question2문제를 실행하여 KEY를 물어보는 상태까지 확인을 하고

올리로 attach를 하여 보자..어차피 패스워드를 풀기위한 부분만 리버싱하는게 목적이므로..

그리고는 Alt+F9로(excute till user code) 유저에서 입력받을때까지 기다리고

KEY묻는 창에서 1313131을 적는다.

그럼 브포가 걸리면서 이제부터는 Ctrl+F9(excute till return)를 눌러 유저코드가 있는곳까지

계속 실행을 해보자

계속 트레이싱을 해보면 아!! 1313130이 보이게 된다. ㅋㅋ

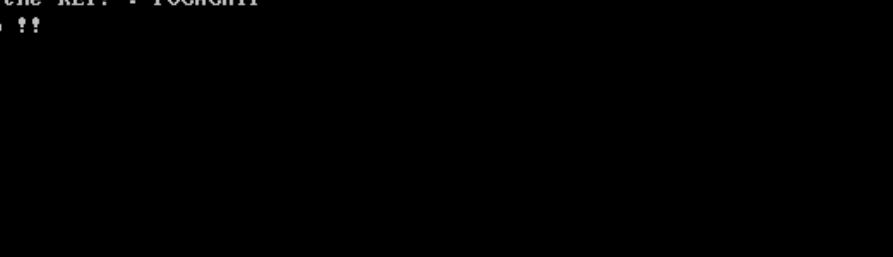
이제부터 눈이 빠지게 131313 언제나오는지 확인하며

특히 1313130이 하나씩 줄어들며 루프를 도는 행동을 하는지 신경써서 트레이싱 해보자

```
00402C58 3B8C95 0CF0FFFF CMP ECX,DWORD PTR SS:[EBP+EDX*4-FF4]
```

이게 패스워드이다.

Address	Hex	dump	ASCII
0012EF4C	CC CC CC CC	CC CC CC CC	CC CC CC CC
0012EF5C	CC CC CC CC	01 CC CC CC	10 EF 12 00
0012EF6C	50 00 00 00	4E 00 00 00	41 00 00 00
0012EF7C	47 00 00 00	4D 00 00 00	4F 00 00 00
0012EF8C	50 00 00 00	4F 00 00 00	43 00 00 00
0012EF9C	43 00 00 00	43 00 00 00	49 00 00 00
0012EFAC	01 00 00 00	01 00 00 00	01 00 00 00
0012EFBC	01 00 00 00	01 00 00 00	01 00 00 00
0012EFC	01 00 00 00	01 00 00 00	01 00 00 00
0012EFD	01 00 00 00	01 00 00 00	01 00 00 00
0012EFE	01 00 00 00	01 00 00 00	01 00 00 00
0012EFF	01 00 00 00	01 00 00 00	01 00 00 00
0012F00	01 00 00 00	01 00 00 00	01 00 00 00



C:\Documents and Settings\WboonMRW바탕 화면\Wrecon\WReversing\WQuestion...

What is the KEY? : POCACHIP

Good Job !!

Good Job!!^^