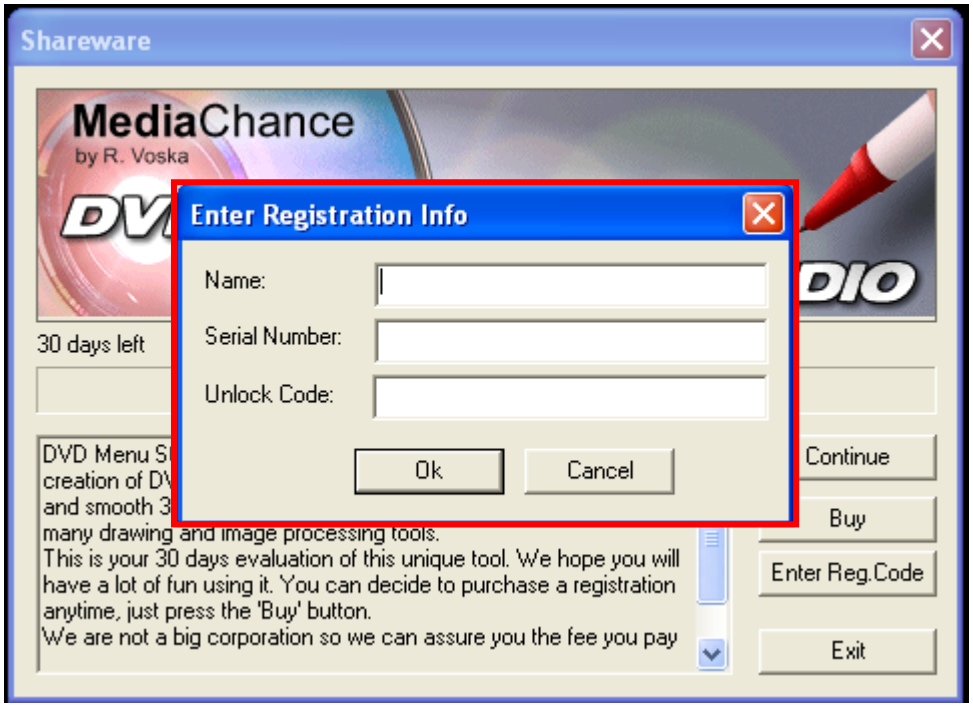
악성코드 분석 보고서

(sand-reversingwithlena-tutorials)

2025.07.17

14

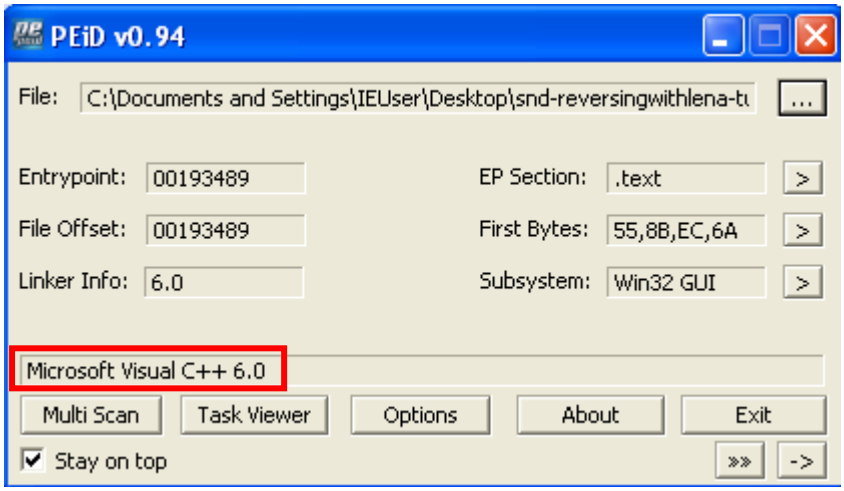
1. 문제



키 입력 창이 발생하는 것을 볼 수 있다. 이걸 없애줘야한다.

Unregistered -> Registered로 변경되어야한다.

2. 해결 방법

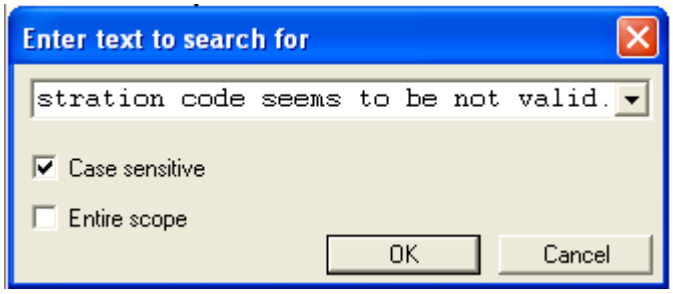


PEID로 확인 시 C++로 만든 파일이라는 것을 알 수 있다.

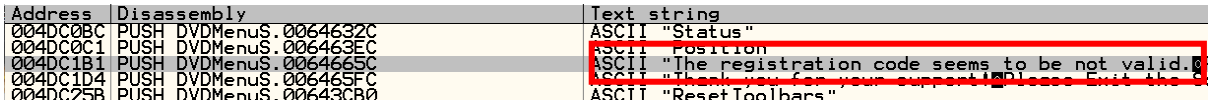
키 입력 칸에

아무것도 입력 안 할 시 -> Please enter your name, 이름만 입력할 시, 시리얼 넘버 특정 개수 넘어가지 않을 시 -> Please check Serial Number...

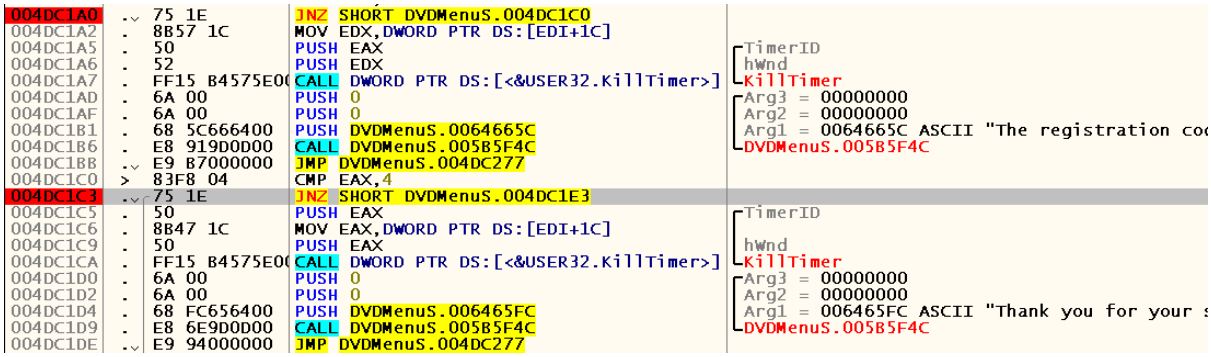
시리얼 넘버 특정 개수 넘어가고 Unlock Code 적어도 -> The registration code seems to be not valid. Please check if you didn't made any mistake. 창이 생기는 것을 볼 수 있다,



Search for > All referenced text strings 눌러서 '-> The registration code seems to be not valid. ' 해당 구문을 찾아준다.

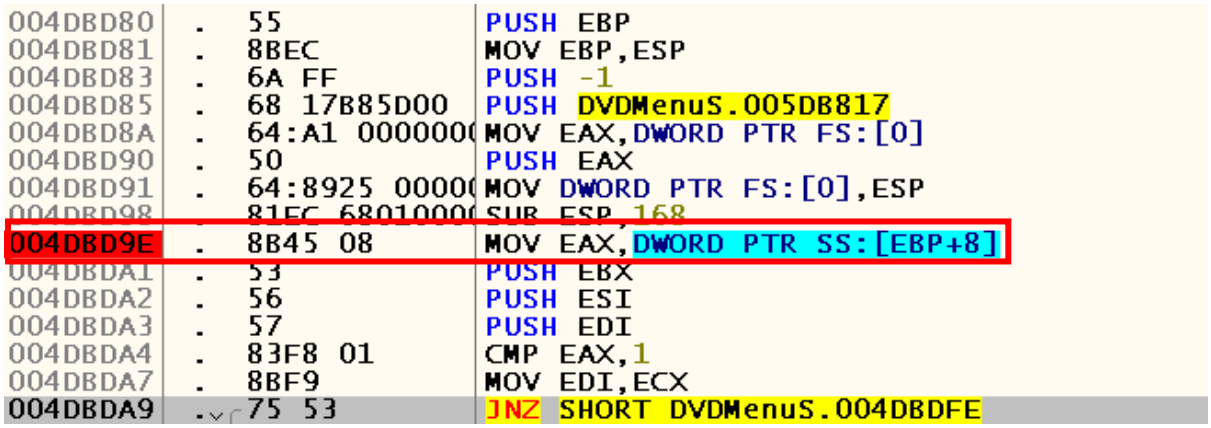


들어가서 주변 분기문을 찾아주고 브레이크를 걸어준다.

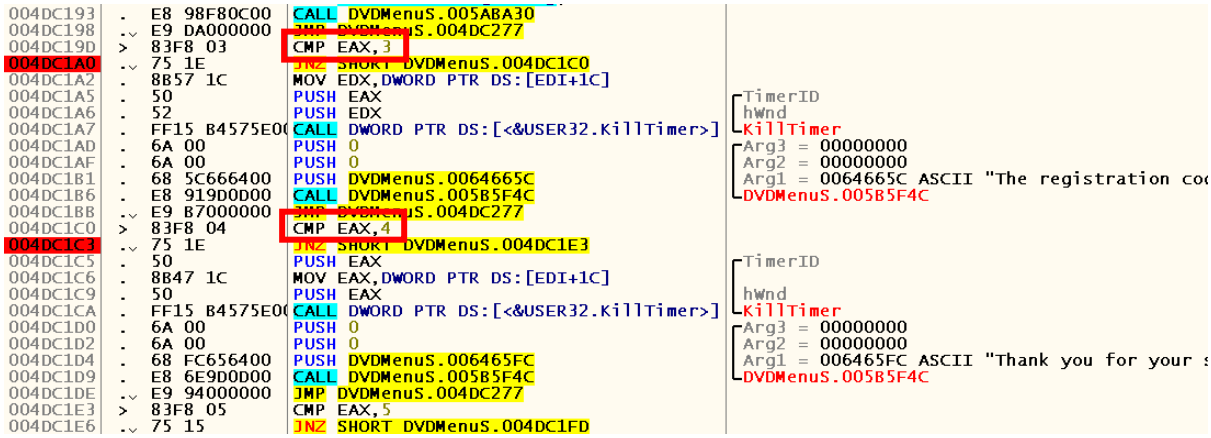


근처에 성공했을 시 나오는 구문을 찾을 수 있다. 그 분기문에 브레이크를 걸어준다.

분기문부터 함수까지 반복하는 것을 볼 수 있고 함수 호출 후 0x004DC277로 점프하는 것을 볼 수 있다.



EAX를 정해주는 함수를 찾기 위해 위로 올라가보면 0x004DBD9E에서 EAX 값을 정해주는 것을 볼 수 있다.



성공했을 때와 실패했을 때 코드를 보면 분기문 위해 EAX를 비교하는 코드가 있는 걸 볼 수 있다.

EAX = 4일 경우 성공했다는 창이 나오고 EAX = 3일 경우 실패했다는 창이 나온다.

그럼 EAX가 무조건 4가 나오게 해야한다.

```
004DBD80  .   55            PUSH EBP
004DBD81  .   8BEC          MOV EBP,ESP
004DBD83  .   6A FF         PUSH -1
004DBD85  .   68 17B85D00   PUSH DVDMenuS.005DB817
004DBD8A  .   64:A1 0000000 MOV EAX,DWORD PTR FS:[0]
004DBD90  .   50            PUSH EAX
004DBD91  .   64:8925 00000 MOV DWORD PTR FS:[0],ESP
004DBD98  .   81EC 6801000( SUB ESP,168
004DBD9E  .   8B45 08       MOV EAX,DWORD PTR SS:[EBP+8]
004DBDA1  .   53            PUSH EBX
004DBDA2  .   56            PUSH ESI
004DBDA3  .   57            PUSH EDI
004DBDA4  .   83F8 01       CMP EAX,1
004DBDA7  .   8BF9          MOV EDI,ECX
004DBDA9  .v  75 53         JNZ SHORT DVDMenuS.004DBDFE
```

0x004DBD9E를 변경해줘야하는데 MOV EAX, 4로 해주면 오버라이트 되서 바로 변경은 안된다.

```
8B45 08       MOV EAX,DWORD PTR SS:[EBP+8]
B8 04000000   MOV EAX,4
```

왜냐하면 MOV EAX,DWORD PTR SS:[EBP+8]는 8B45 08로 총 3byte인데

MOV EAX, 4는 B8 04000000로 총 5byte로 오버라이트된다. 그러면 총 2byte가 오버라이트가 돼서 그 밑에 코드인 PUSH EBX (1bye), PUSH ESI(1byte)가 없어진다.

그러므로 인라인패치(코드를 수정하고 싶은데 직접 접근해서 수정하기 어려울 때 사용하는 우회 방법)를 이용해서 코드를 수정해준다.

밑에 끝까지 내려가 보면 비어있는 공간이 보인다. 거기서 수정을 해준다.

우선 그 공간에 가기 위해서는 0x004DBD9E에서 점프를 해줘야한다.

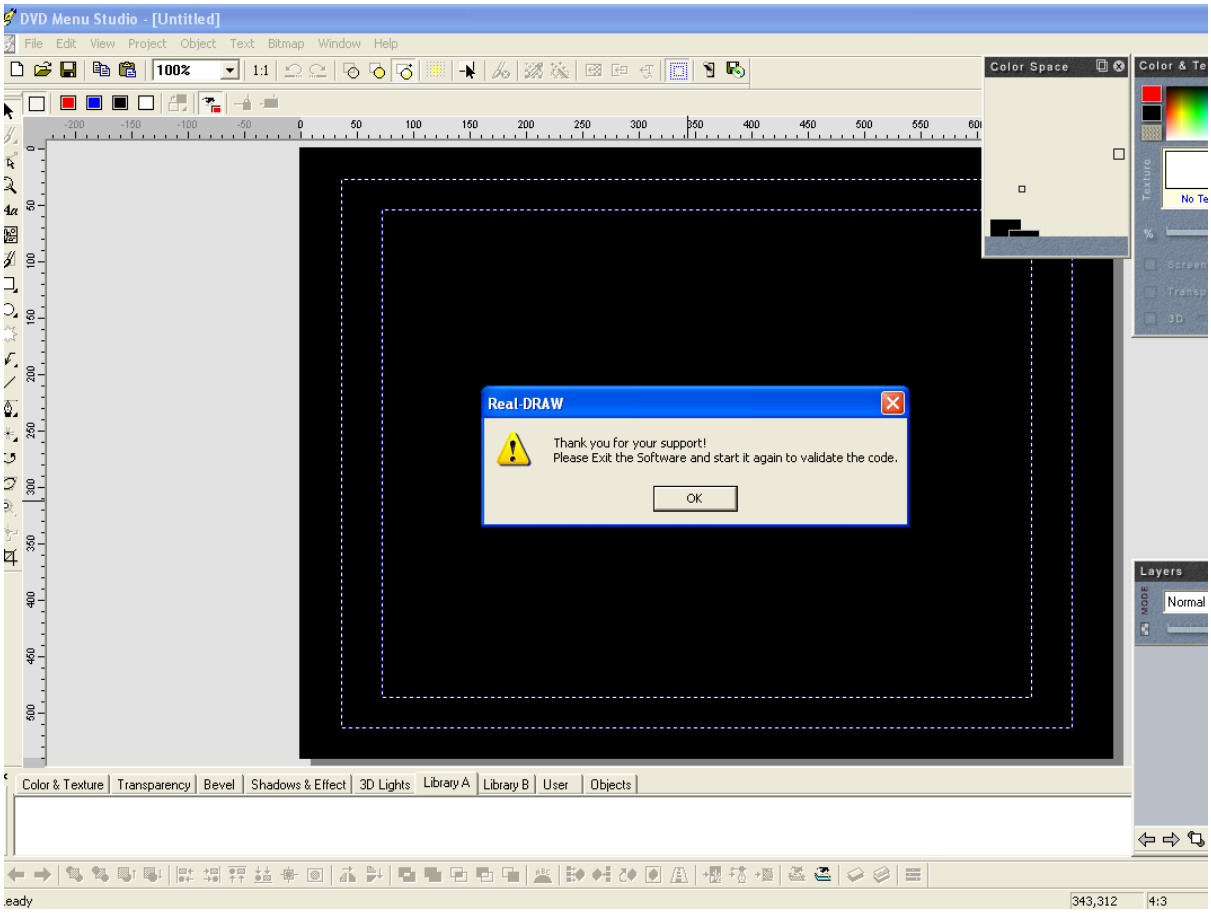
```
004DBD98  .   81EC 6801000( SUB ESP,168
004DBD9E  .v  E9 0C921000   JMP DVDMenuS.005E4FAF
004DBDA3  .   57            PUSH EDI
004DBDA4  .   83F8 01       CMP EAX,1
004DBDA7  .   8BF9          MOV EDI,ECX
004DBDA9  .v  75 53         JNZ SHORT DVDMenuS.004DBDFE
```

MOV EAX,DWORD PTR SS:[EBP+8] -> JMP SHORT DVDMenuS. 005E4FAF로 바꿔준다. 근데 JMP명령어도 총 5byte를 차지하기 때문에 PUSH EBX, PUSH ESI가 없어진다.

그럼 005E4FAF에서 PUSH EBX, PUSH ESI를 작성해줘야한다.

```
B8 04000000   MOV EAX,4
53            PUSH EBX
56            PUSH ESI
E9 E86DEFFF   JMP DVDMenuS.004DBDA3
```

다시 0x004DBD9E 돌아가야하기 때문에 JMP SHORT DVDMenuS. 004DBDA3도 작성해준다.



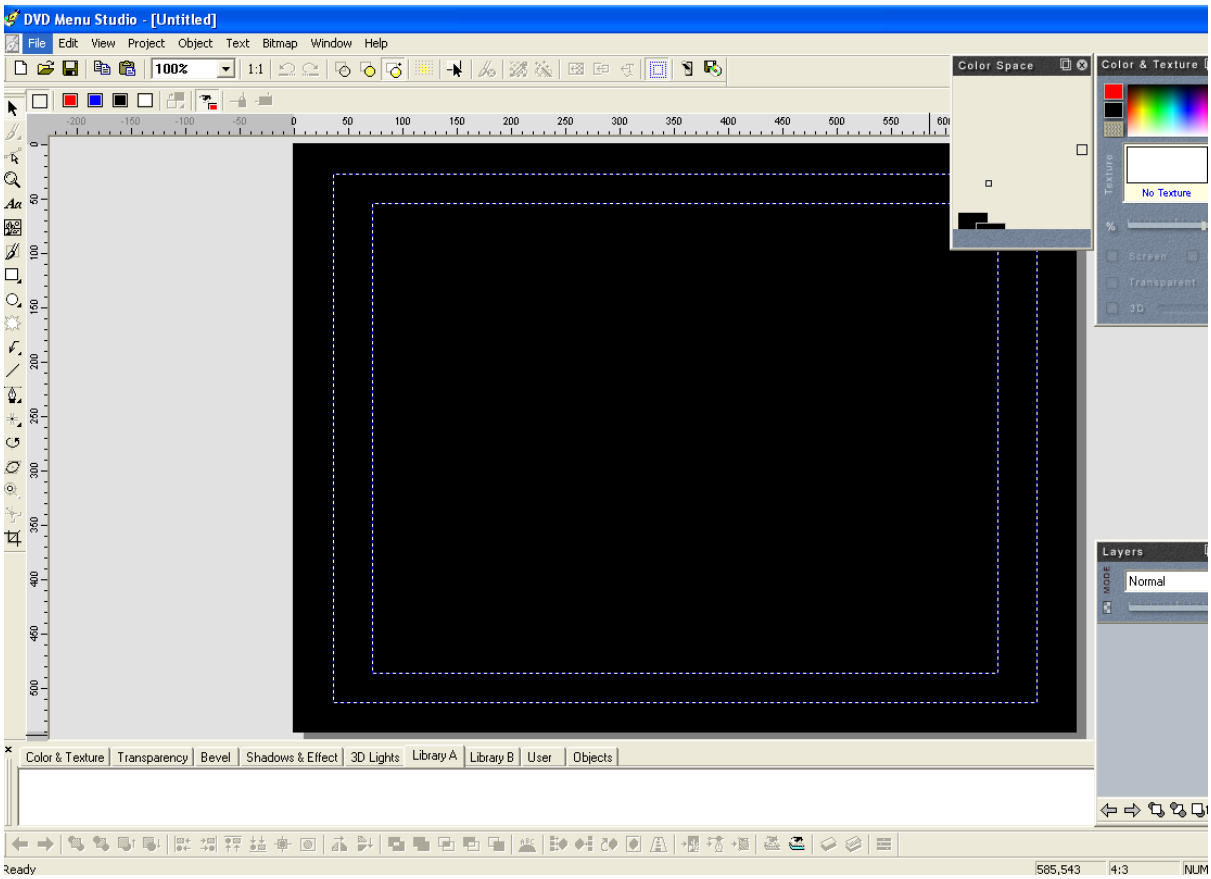
실행해보면 성공했다는 창이 무한 반복되는 걸 볼 수 있다. 왜냐하면 이 프로그램을 실행 할 때 그 분기문을 지나가기 때문에 무한루프처럼 계속 뜨는 것을 볼 수 있다, 그럼 EAX = 4가 되면 안된다.

```
004DBDA4   .  83F8 01      CMP EAX,1
004DBDA9   .  75 53        JNZ SHORT DVDMenuS.004DBDFE
004DBDAB   .  50           PUSH EAX                          ; TimerID
004DBDAC   .  8B47 1C      MOV EAX,DWORD PTR DS:[EDI+1C]
004DBDAF   .  50           PUSH EAX                          ; hWnd
004DBDB0   .  FF15 B4575E0 CALL DWORD PTR DS:[<&USER32.KillTimer>]  ; KillTimer
004DBDB6   .  E8 8E5C0E00  CALL DVDMenuS.005C1A49
```

모든 EAX를 찾아서 실행해보자. 우선 EAX = 1일 경우로 해본다.

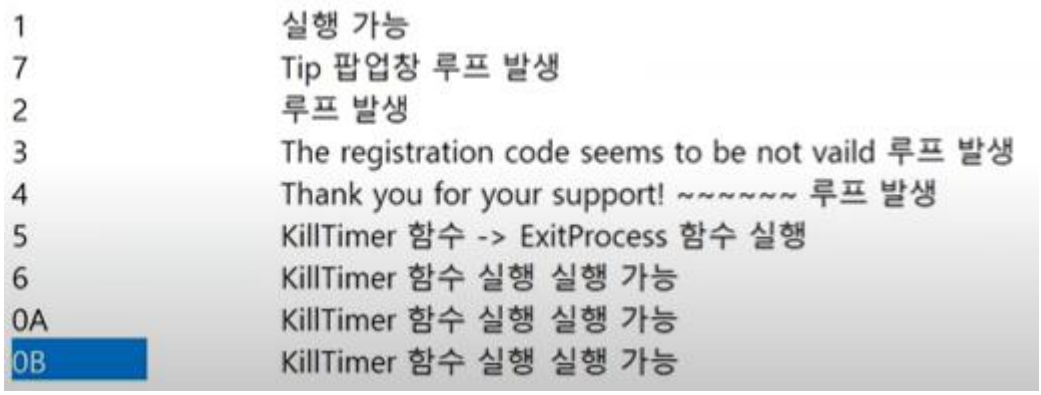
```
B8 01000000   MOV EAX,1
53            PUSH EBX
56            PUSH ESI
E9 E86DEFFF   JMP DVDMenuS.004DBDA3
```

바꿔주고 저장해고 실행해본다.

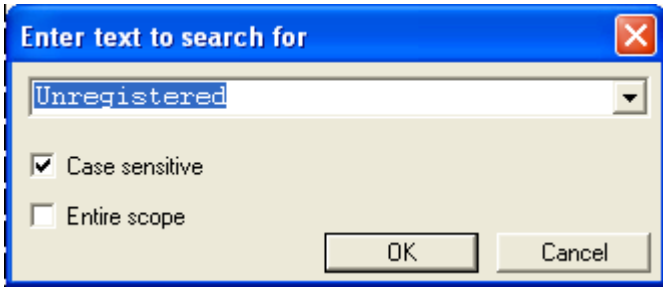


실행해보면 아무 창도 안뜨고 잘 실행되는게 보인다. EAX = 1로 바꿔야 잘 작동되는 걸 알 수 있다.

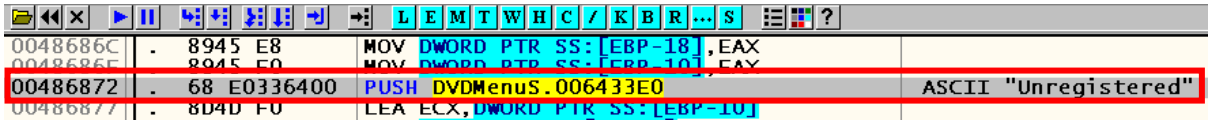
* 참고



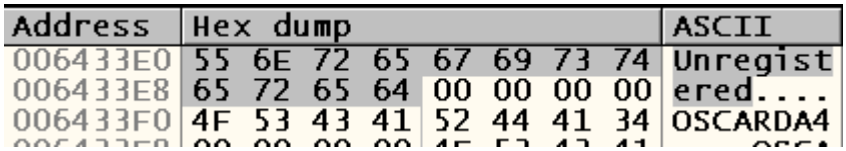
안바꼈으므로 바꿔준다.



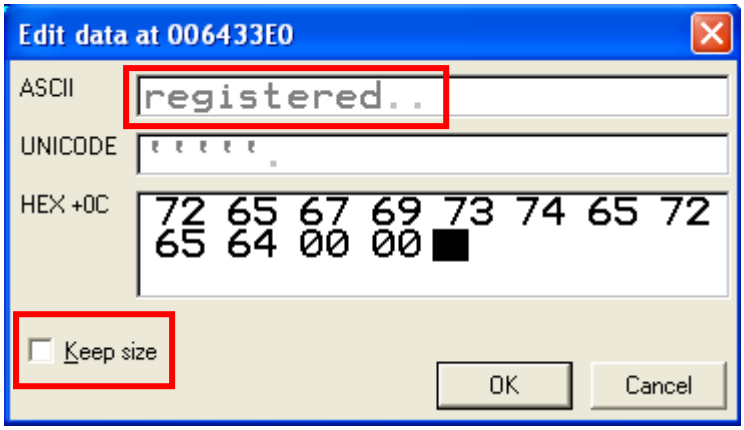
Search for > All referenced text strings 눌러서 Unregistered 해당 구문을 찾아준다.



이걸 참조하고 있는 곳으로 간다. 덤프 창에서 0x006433E0검색해준다.



여기서 'Ctrl + e'누른 후 수정해준다.



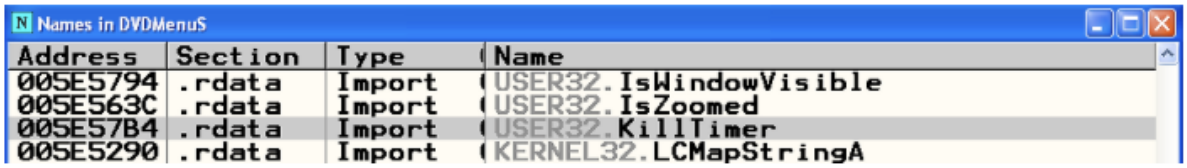
Keep size 꺼주고 앞에 'Un' 삭제하고 뒤에 00을 추가해준 후 다시 저장한다.

바뀐 걸 볼 수 있다.

문제 해결!

-> Lena는 30일의 타이머가 있어서 KillerTimer API로 다 브레이크 걸어서 수정해줬다.

프로그램이 30일 제약이 걸려있기 때문에 시간과 관련된 API를 사용할 것으로 추정된다.
KillTimer API를 찾는다.



set breakpoint on every reference를 눌러서 breakpoint를 걸어준다.
48개의 breakpoint가 걸린다.