리버스 엔지니어링 보고서

write by 허00

02.exe(abex_level2) 분석&풀이

목차

1.서문

1-1 개요

abex_level_2를 리버싱 하게 되었다. 이 파일은 pe구조가 손상되어서 올리디버거가 작동하지 않았다. 그래서 hxd라는 헥사 에디터를 이용하여 문제풀이를 하게 되었다.

1-2 목표

실행 불가능한 파일에서 바이너리 분석을 이용하여서 시리얼 키를 추출해낼 수 있다.

1-3 목적

실행 불가능한 파일의 분석 방법을 배울 수 있다.

1-4 분석툴

- hxd : 모든 파일, 메모리, 물리디스크를 덤프해서 보여준다.
- peview : pe파일의 정보를 덤프, pe구조를 형식화해서 보여준다.
- ollydbg : 프로그램을 디버깅하는 툴로, 코드, 레지스터 상태, 스택 등을 프로그램의 흐름에 따라 관찰할 수 있다.

2.정적 분석

2-1 외관상 특징

-파일이름 : 02.exe

-파일크기 : 4kb

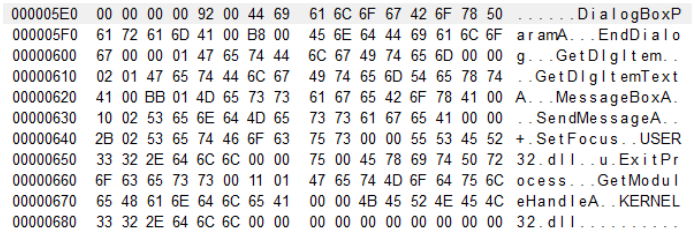
2-1-1 아이콘,확장자

-아이콘 : 없음

-확장자 : exe(pe)

2-2 import, export된 항목

(그림 1)

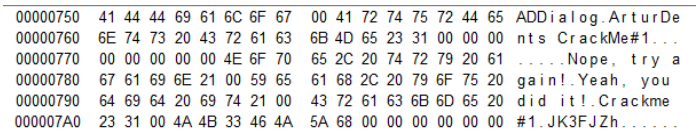


```
000005E0  00 00 00 00 92 00 44 69  61 6C 6F 67 42 6F 78 50  ......DialogBoxP
000005F0  61 72 61 6D 41 00 B8 00  45 6E 64 44 69 61 6C 6F  aramA...EndDialo
00000600  67 00 00 01 47 65 74 44  6C 67 49 74 65 6D 00 00  g...GetDlgItem..
00000610  02 01 47 65 74 44 6C 67  49 74 65 6D 54 65 78 74  ..GetDlgItemText
00000620  41 00 BB 01 4D 65 73 73  61 67 65 42 6F 78 41 00  A...MessageBoxA.
00000630  10 02 53 65 6E 64 4D 65  73 73 61 67 65 41 00 00  ..SendMessageA..
00000640  2B 02 53 65 74 46 6F 63  75 73 00 00 55 53 45 52  +.SetFocus..USER
00000650  33 32 2E 64 6C 6C 00 00  75 00 45 78 69 74 50 72  32.dll..u.ExitPr
00000660  6F 63 65 73 73 00 11 01  47 65 74 4D 6F 64 75 6C  ocess...GetModul
00000670  65 48 61 6E 64 6C 65 41  00 00 4B 45 52 4E 45 4C  eHandleA..KERNEL
00000680  33 32 2E 64 6C 6C 00 00  00 00 00 00 00 00 00 00  32.dll.........
```

pe구조가 깨져서 import항목과 export 항목이 구분 되지 않았다. 그래서 hxd를 이용하여서 사용된 함수의 힌트를 찾아보았다. 메시지 관련 함수가 많은 것을 볼 때 win32 gui프로그램으로 추정된다.

2-3 문자열

(그림 2)



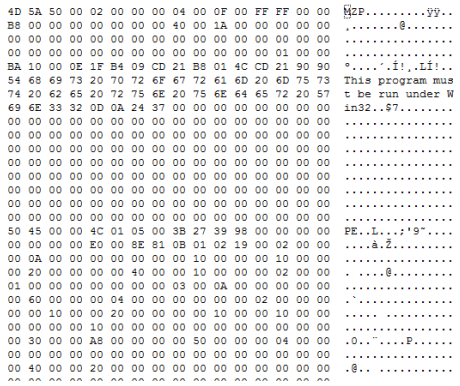
```
00000750  41 44 44 69 61 6C 6F 67  00 41 72 74 75 72 44 65  ADDialog.ArturDe
00000760  6E 74 73 20 43 72 61 63  6B 4D 65 23 31 00 00 00  nts CrackMe#1...
00000770  00 00 00 00 00 4E 6F 70  65 2C 20 74 72 79 20 61  .....Nope, try a
00000780  67 61 69 6E 21 00 59 65  61 68 2C 20 79 6F 75 20  gain!.Yeah, you
00000790  64 69 64 20 69 74 21 00  43 72 61 63 6B 6D 65 20  did it!.Crackme
000007A0  23 31 00 4A 4B 33 46 4A  5A 68 00 00 00 00 00 00  #1.JK3FJZh......
```

메시지 박스에 출력될 것으로 추정되는 문자열과 마지막에 정답으로 추정되는 “JK3FJZh”가 보인다.

2-4 PE구조상 특이사항

헤더를 분석해 보니 정상 파일에 비해 헤더가 심하게 손상되어 있었다.

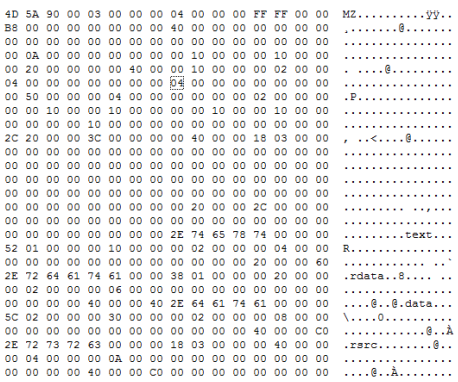
(그림 3)



```
4D 5A 50 00 02 00 00 00 04 00 0F 00 FF FF 00 00  MZP.........ÿÿ..
B8 00 00 00 00 00 00 00 40 00 1A 00 00 00 00 00  ,.......@.......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 01 00 00 00  ................
BA 10 00 0E 1F B4 09 CD 21 B8 01 4C CD 21 90 90  º....´.Í!¸.LÍ!..
54 68 69 73 20 70 72 6F 67 72 61 6D 20 6D 75 73  This program mus
74 20 62 65 20 72 75 6E 20 75 6E 64 65 72 20 57  t be run under W
69 6E 33 32 0D 0A 24 37 00 00 00 00 00 00 00 00  in32..$7........
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
50 45 00 00 4C 01 05 00 3B 27 39 98 00 00 00 00  PE..L...;'9˜....
00 00 00 00 E0 00 8E 81 0B 01 02 19 00 02 00 00  ....à.Ž.........
00 0A 00 00 00 00 00 00 00 10 00 00 00 10 00 00  ................
00 20 00 00 00 00 40 00 00 10 00 00 00 02 00 00  . ....@.........
01 00 00 00 00 00 00 00 03 00 0A 00 00 00 00 00  ................
00 60 00 00 00 04 00 00 00 00 00 00 02 00 00 00  .`..............
00 00 10 00 00 20 00 00 00 00 10 00 00 10 00 00  ..... ..........
00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00  ................
00 30 00 00 A8 00 00 00 00 50 00 00 00 04 00 00  .0..¨....P......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 40 00 00 20 00 00 00 00 00 00 00 00 00 00 00  .@.. ...........
```

“abex level 1“의 헤더 부분이다. NT_DOS_HEADER부터, PE시그너쳐 등이 관찰된다.

(그림4)



```
4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ.........ÿÿ..
B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ,.......@.......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 0A 00 00 00 00 00 00 00 10 00 00 00 10 00 00  ................
00 20 00 00 00 00 40 00 00 10 00 00 00 02 00 00  . ....@.........
04 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00  ................
00 50 00 00 00 04 00 00 00 00 00 00 02 00 00 00  .P..............
00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 00  ................
00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00  ................
2C 20 00 00 3C 00 00 00 00 40 00 00 18 03 00 00  , ..<....@......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 20 00 00 2C 00 00 00 00  ........ ..,....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00  .........text...
52 01 00 00 00 10 00 00 00 02 00 00 00 04 00 00  R...............
00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60  ............ ..`
2E 72 64 61 74 61 00 00 38 01 00 00 00 20 00 00  .rdata..8.... ..
00 02 00 00 00 06 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 40 00 00 40 2E 64 61 74 61 00 00 00  ....@..@.data...
5C 02 00 00 00 30 00 00 00 02 00 00 00 08 00 00  \....0..........
00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 C0  ............@..À
2E 72 73 72 63 00 00 00 18 03 00 00 00 40 00 00  .rsrc........@..
00 04 00 00 00 0A 00 00 00 00 00 00 00 00 00 00  ................
00 00 00 00 40 00 00 C0 00 00 00 00 00 00 00 00  ....@..À........
```

(그림 3)과 대조적으로 헤더부분에 많은 항목들이 누락되어있다.

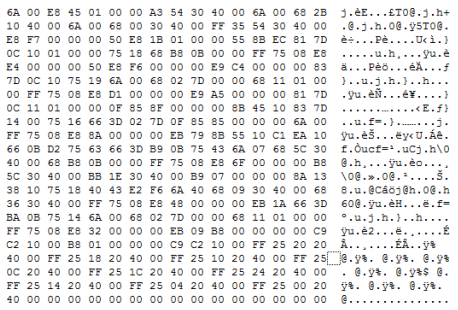
3.동적 분석

3-1 패킹여부

올바른 pe파일이 아니여서, 패킹 여부가 판단되지 않았다.

3-2 코드 분석

(그림 5)

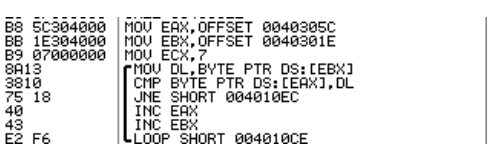


```
6A 00 E8 45 01 00 00 A3 54 30 40 00 6A 00 68 2B  j.èE...£T0@.j.h+
10 40 00 6A 00 68 00 30 40 00 FF 35 54 30 40 00  .@.j.h.0@.ÿ5T0@.
E8 F7 00 00 00 50 E8 1B 01 00 00 55 8B EC 81 7D  è÷...Pè....U‹ì.}
0C 10 01 00 00 75 18 68 B8 0B 00 00 FF 75 08 E8  .....u.h¸...ÿu.è
E4 00 00 00 50 E8 F6 00 00 00 E9 C4 00 00 00 83  ä...Pèö...éÄ...ƒ
7D 0C 10 75 19 6A 00 68 02 7D 00 00 68 11 01 00  }..u.j.h.}..h...
00 FF 75 08 E8 D1 00 00 00 E9 A5 00 00 00 81 7D  .ÿu.èÑ...é¥....}
0C 11 01 00 00 0F 85 8F 00 00 00 8B 45 10 83 7D  ..........‹E.ƒ}
14 00 75 16 66 3D 02 7D 0F 85 85 00 00 00 6A 00  ..u.f=.}.……....j.
FF 75 08 E8 8A 00 00 00 EB 79 8B 55 10 C1 EA 10  ÿu.èŠ...ëy‹U.Áê.
66 0B D2 75 63 66 3D B9 0B 75 43 6A 07 68 5C 30  f.Òucf=¹.uCj.h\0
40 00 68 B8 0B 00 00 FF 75 08 E8 6F 00 00 00 B8  @.h¸...ÿu.èo...¸
5C 30 40 00 BB 1E 30 40 00 B9 07 00 00 00 8A 13  \0@.».0@.¹....Š.
38 10 75 18 40 43 E2 F6 6A 40 68 09 30 40 00 68  8.u.@Câöj@h.0@.h
36 30 40 00 FF 75 08 E8 48 00 00 00 EB 1A 66 3D  60@.ÿu.èH...ë.f=
BA 0B 75 14 6A 00 68 02 7D 00 00 68 11 01 00 00  °.u.j.h.}..h....
FF 75 08 E8 32 00 00 00 EB 09 B8 00 00 00 00 C9  ÿu.è2...ë.¸.....É
C2 10 00 B8 01 00 00 00 C9 C2 10 00 FF 25 20 20  Â..¸....ÉÂ..ÿ%
40 00 FF 25 18 20 40 00 FF 25 10 20 40 00 FF 25  @.ÿ%. @.ÿ%. @.ÿ%
0C 20 40 00 FF 25 1C 20 40 00 FF 25 24 20 40 00  . @.ÿ%. @.ÿ%$ @.
FF 25 14 20 40 00 FF 25 04 20 40 00 FF 25 00 20  ÿ%. @.ÿ%. @.ÿ%.
40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  @...............
```

"FF25" IAT패턴을 바탕으로 코드영역을 찾아 보았다.

다른 정상 프로그램에 코드를 이식하여서 올리디버거로 돌려보니 아래와 같이 키를 대조하는 루틴이 등장했다.

(그림6)



```
B8 5C304000  MOV EAX,OFFSET 0040305C
BB 1E304000  MOV EBX,OFFSET 0040301E
B9 07000000  MOV ECX,7
8A13         MOV DL,BYTE PTR DS:[EBX]
3810         CMP BYTE PTR DS:[EAX],DL
75 18        JNE SHORT 004010EC
40           INC EAX
43           INC EBX
E2 F6        LOOP SHORT 004010CE
```

eax와 ebx를 이용하여서 문자열을 비교하고 있다. "mov ecx 7을 통해 7자리 키임을 짐작할 수 있다." 깨진 파일이라서 각각의 문자열 오프셋을 분석하는 데에는 실패했다.

3-3 문제 해결 방법&풀이

위의 과정을 통하여 문제의 답은 7자리 키이며 (그림 2)의 키 "JK3FJZh" 까지는 유추할 수 있었지만, 실제로 키 값이 맞는지 아니면 시드(seed) 값으로 사용되고 있는지는 증명할 수 없었다. 지금은 추정을 통해서 키라고 간주 했지만, 나중에는 꼭 증명해 보고 싶다.

4.마치며

4-1 발견한 노하우

1. 손상된 pe에서 정보 추출하기
2. 다양한 가정과 추론을 통한 문제 풀이

4-2 참신했던 부분

문제 자체가 정말 건질 것이 많아서 참신했다. 깨진 파일에서 단순히 문자열을 보고 정답이라고 생각할 수 있었지만, 파일에 분명히 존재하는 코드영역을 보고 분석을 결심했다.

그리고 몇 번의 시행착오 끝에 올리디버거로 코드를 읽어 내는데 성공 했다. 문자열 오프셋을 대칭시키는 데에는 실패했지만, 예전에 이론만 공부하고, 실전에 뛰어든 2번째 문제 치고는 장족의 발전인 듯 해 흐뭇하다. 다음에 꼭 파일을 완전 복구해서 다시 도전해 보고 싶다.

4-3 연구, 필요 했던 부분

1. pe구조
2. 바이너리 분석 방법