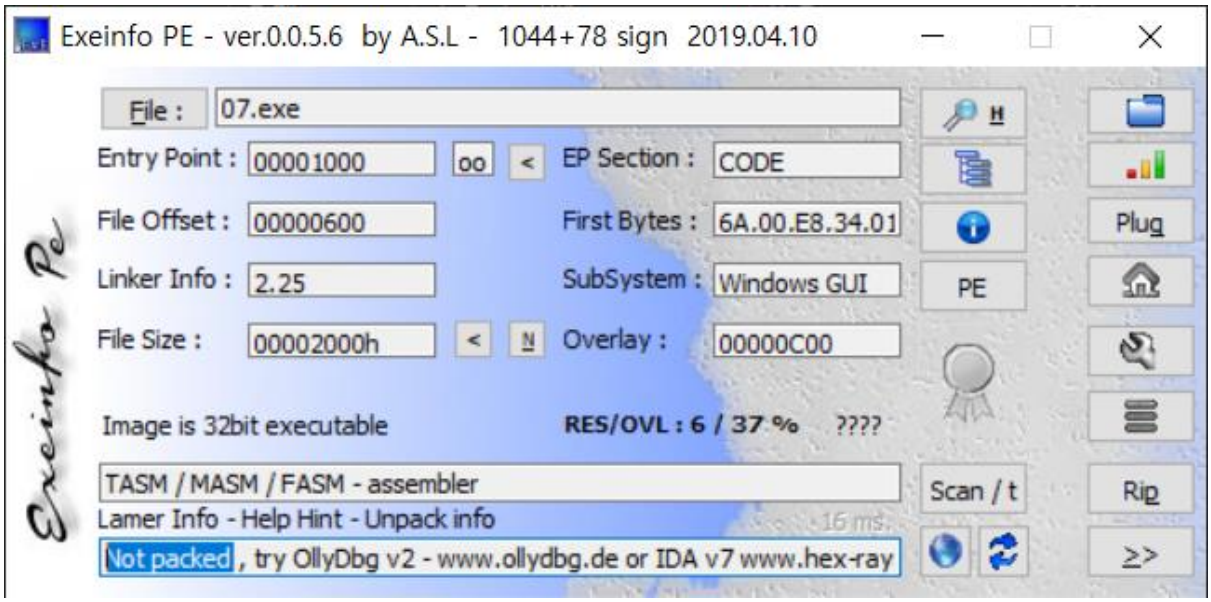
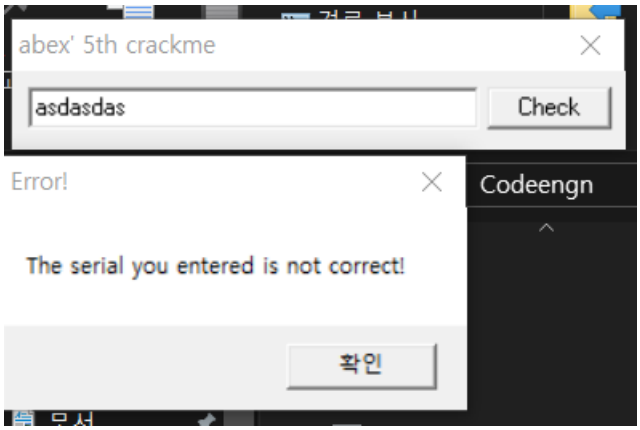
06.exe - 컴퓨터 C 드라이브의 이름이 CodeEngn 일경우 시리얼이 생성될때 CodeEngn은 'ß 어떤것'으로 변경되는가

일단 프로그램을 디버깅 하기 전 PE분석을 해본다.

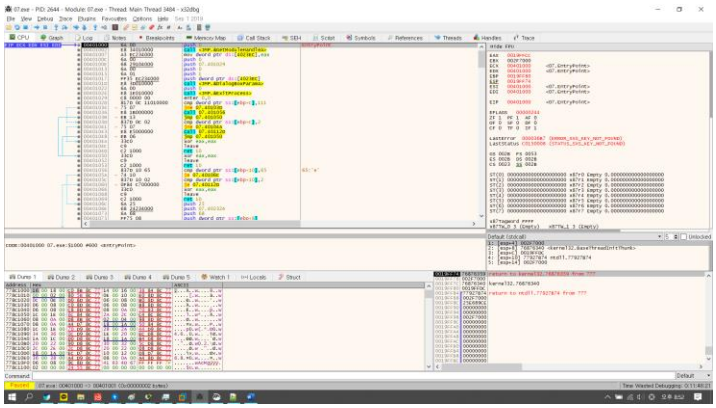


ExeinfoPE를 통해 Not packed가 되었다는 것을 확인함.



프로그램을 실행시 시리얼 키를 입력 받고 확인을 하는 프로그램으로 추정됨.

X32dbg로 실행하여 분석을 시작한다.



일단 Check 버튼을 눌렀을 때 하드디스크의 이름 정보를 가져오는 함수가 작동될 것으로 예상된다.

이때 keyword(volume, disk, information, get 등)을 가지고 인터럽트를 검색해본다.

Address	Disassembly	Destination
00401002	call <JMP.&GetModuleHandleA>	<kernel32.GetModuleHandleA>
0040101D	call <JMP.&DialogBoxParamA>	<user32.DialogBoxParamA>
00401024	call <JMP.&ExitProcess>	<kernel32.ExitProcess>
00401078	call <JMP.&GetDlgItemTextA>	<user32.GetDlgItemTextA>
00401099	call <JMP.&GetVolumeInformationA>	<kernel32.GetVolumeInformationA>
004010A8	call <JMP.&lstrcatA>	<kernel32.lstrcatA>
004010D9	call <JMP.&lstrcatA>	<kernel32.lstrcatA>
004010E8	call <JMP.&lstrcatA>	<kernel32.lstrcatA>
004010F7	call <JMP.&lstrcmpiA>	<kernel32.lstrcmpiA>
00401110	call <JMP.&MessageBoxA>	<user32.MessageBoxA>
00401126	call <JMP.&MessageBoxA>	<user32.MessageBoxA>
00401132	call <JMP.&EndDialog>	<user32.EndDialog>

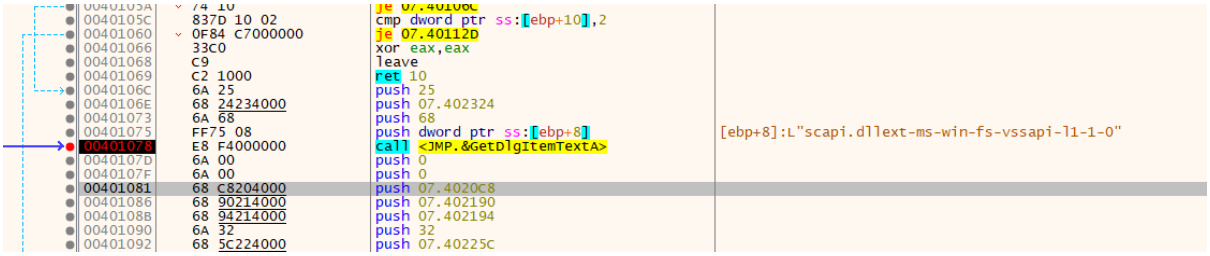
이때 "GetVolumeInformationA" 함수를 발견하게 되는데 이 함수를 통해 볼륨의 이름을 가져오는 것으로 추정이 된다.

aph | Log | Notes | Breakpoints | Memory Map | Call Stack | SEH | Script | Symbols | References | Threads

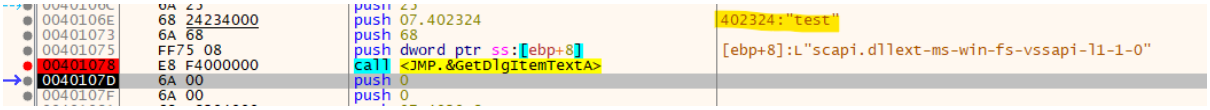
```
0040105A   74 10          je 07.40106C
0040105C   837D 10 02     cmp dword ptr ss:[ebp+10],2
00401060   0F84 C7000000  je 07.40112D
00401066   33C0           xor eax,eax
00401068   C9             leave
00401069   C2 1000        ret 10
0040106C   6A 25          push 25
0040106E   68 24234000    push 07.402324
00401073   6A 68          push 68
00401075   FF75 08        push dword ptr ss:[ebp+8]
00401078   E8 F4000000    call <JMP.&GetDlgItemTextA>
0040107D   6A 00          push 0
0040107F   6A 00          push 0
00401081   68 C8204000    push 07.4020C8
00401086   68 90214000    push 07.402190
0040108B   68 94214000    push 07.402194
00401090   6A 32          push 32
00401092   68 5C224000    push 07.40225C
00401097   6A 00          push 0
00401099   E8 B5000000    call <JMP.&GetVolumeInformationA>
0040109E   68 F3234000    push 07.4023F3                    4023F3:"4562-ABEX"
004010A3   68 5C224000    push 07.40225C
004010A8   E8 94000000    call <JMP.&lstrcatA>
004010AD   B2 02          mov dl,2
004010AF   8305 5C224000 01  add dword ptr ds:[40225C],1
004010B6   8305 5D224000 01  add dword ptr ds:[40225D],1
004010BD   8305 5E224000 01  add dword ptr ds:[40225E],1
004010C4   8305 5F224000 01  add dword ptr ds:[40225F],1
004010CB   FECA           dec dl
004010CD   75 E0          jne 07.4010AF
004010CF   68 FD234000    push 07.4023FD                    4023FD:"L2C-5781"
004010D4   68 00204000    push 07.402000
004010D9   E8 63000000    call <JMP.&lstrcatA>
004010DE   68 5C224000    push 07.40225C
004010E3   68 00204000    push 07.402000
004010E8   E8 54000000    call <JMP.&lstrcatA>
004010ED   68 24234000    push 07.402324
```

이제 함수를 호출하는 주소로 들어가 보면 위와 같이 inputbox에서 입력 값을 받아오고 어떠한 문자열과 "lstcatA" 함수를 호출하는 것을 볼 수가 있다.

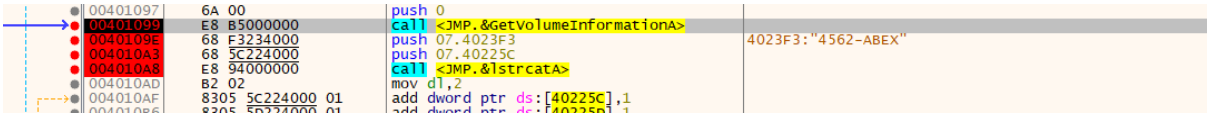
이때 어떠한 작동을 하는지에 대해 분석하기 위해 함수를 호출하는 위치에 Break Point를 걸고 관찰을 한다.



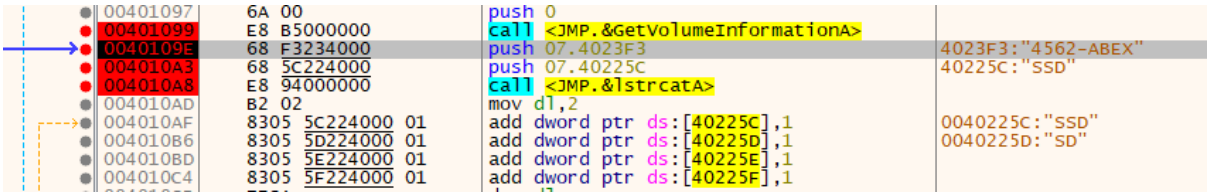
일단 첫번째 Break Point에서 이제 여기서 F8를 눌러 Call를 한다.



실행 결과 402324에 "test" 라는 문자열이 저장된 것을 볼 수 있다.

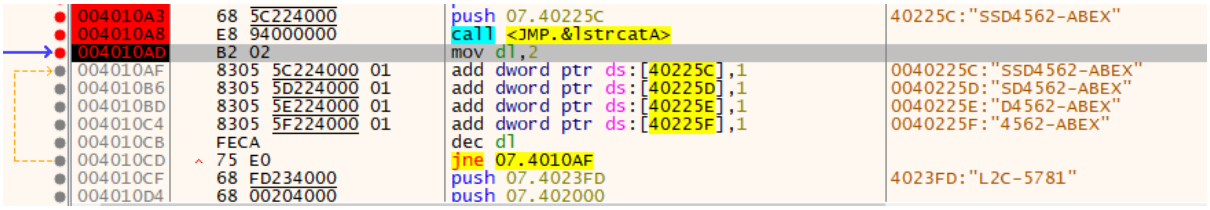


두번째 Break Point에서 F8를 눌러 Call를 하면 어떠한 변화가 생기는지 관찰을 한다.



그 결과 40225C에 저의 하드 볼륨 이름인 SSD가 들어가게 되는 것을 볼 수가 있다.

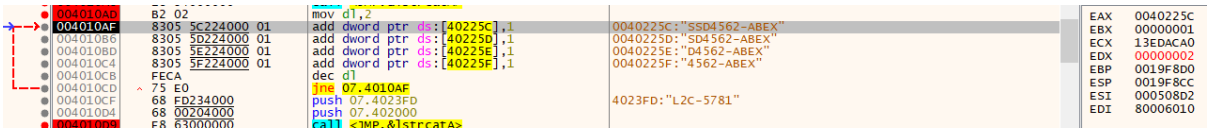
그리고 Call "lstrcatA"라는 함수를 호출하는데 어떠한 변화가 생기는지 관찰을 한다.



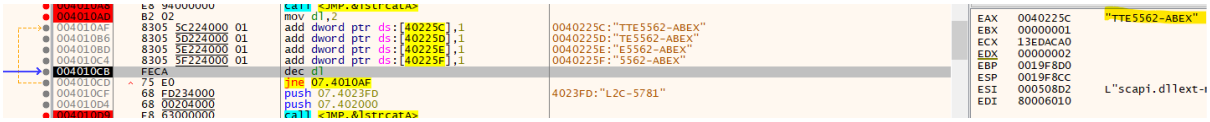
Call를 한 결과 40225C에 시리얼 키로 추정이 되는 문자열과 결합이 되어 "SSD4562-ABEX"가 된 것을 볼 수 있다.

그리고 DL을 2로 설정하고 문자열에 1을 더하는 작업을 여러 번 한 뒤 DI를 1감소시켜 DL이 0이 될 때 까지 반복을 하고 있는 것을 볼 수 있다.

이제 어떻게 문자열이 변화하는지 관찰을 한다.



일단 DL은 2로 설정이 된 것을 볼 수 있다.



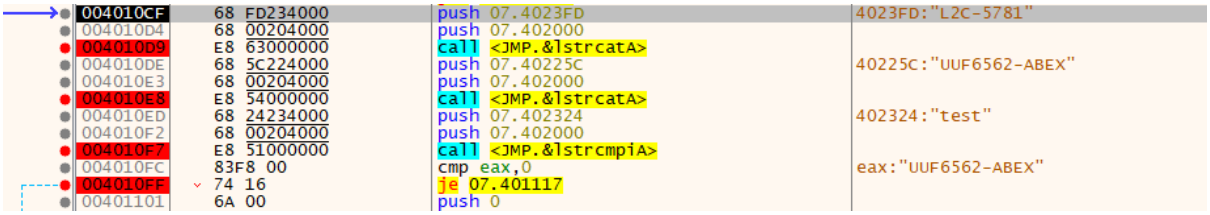
그리고 앞문자부터 하나씩 1씩 증가하는 것을 4번 하여 TTE5562-ABEX가 되는 것을 볼 수 있다.

이제 DL이 0이 되어 반복문이 끝날 때까지 어떻게 변화하는지 관찰하여 본다.

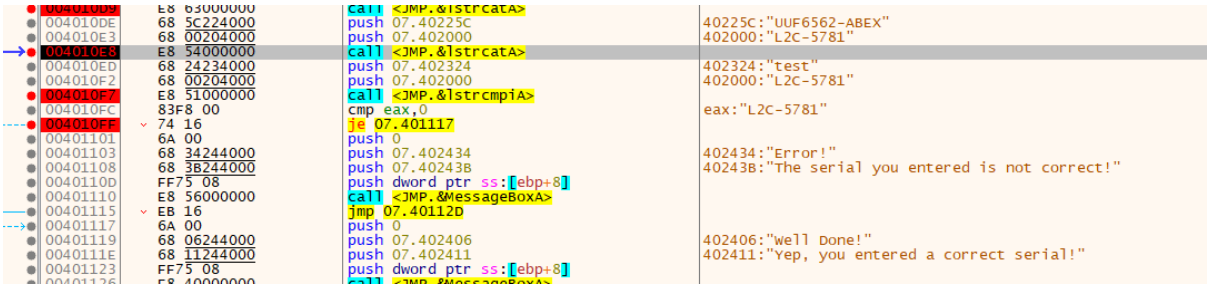
```
0040225C        "UUF6562-ABEX"
```

최종적으로 UUF6562-ABEX가 되는 것을 볼 수 있다.

이제 밑에는 있는 부분을 이어서 분석을 해본다.



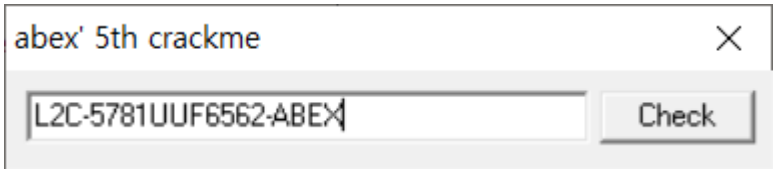
새로운 키조합에 사용되는 것으로 예상이 되는 문자열 L2C-5781를 stack에 올린다.

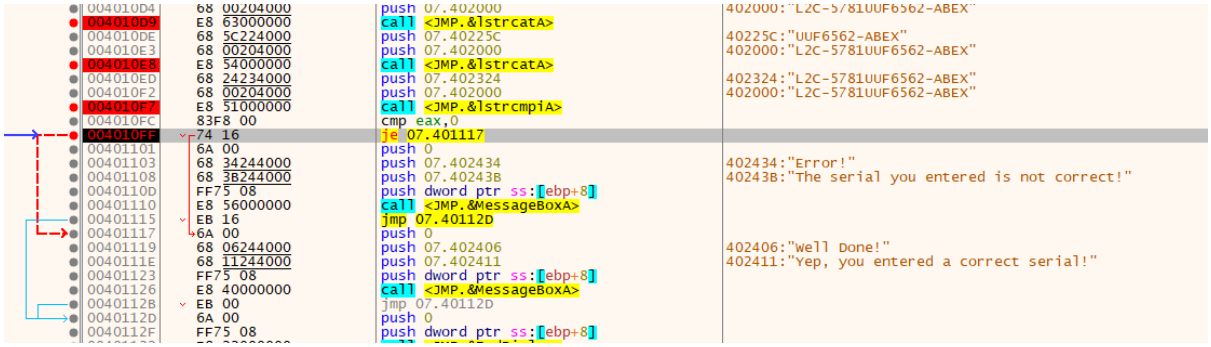


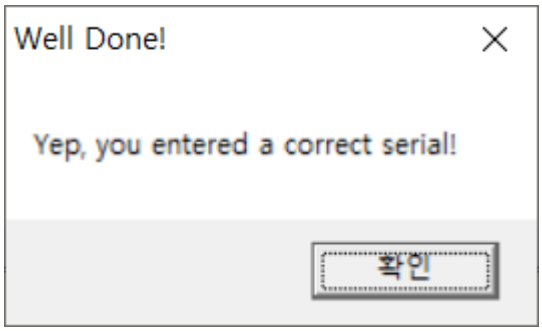
또한 그 이후 "lstrcatA" 함수를 통해 L2C-5781와UUF6562-ABEX를 합치는 것을 볼 수 있다.

그리고 "lstrcmpiA" 함수를 통해 비교한 결과를 ZF 받고 메시지를 출력하는 것을 볼 수가 있다.

이때 생성된 시리얼키값 생성 알고리즘을 분석한 것이 정확한지 직접 넣어서 확인해본다.







이러한 결과를 토대로 시리얼 키를 생성하는 알고리즘은 드라이브의 볼륨이름을 가져와 4562-ABEX와 결합을 하고 앞 4글자에 1씩더하는 것을 2번 하여서 L2C-5781와 결합하여 최종 시리얼 키를 생성하는 것을 파악할 수 있다.

이제 문제로 돌아와 C드라이브 이름이 CodeEngn일때의 시리얼 키는 "L2C-5781EqfgEngn4562-ABEX" 라는 것을 알 수가 있다. 그리고 CodeEngn은 EgfgEngn으로 변화하는 것을 알 수 있다.

정답: EgfgEngn