Crack me 문제 보고서

|  |  |
| --- | --- |
| 문제명 | CodeEngn Reverse2 L01 |
| 파일명 | Advance 01.exe |
| 풀이자 | 서치원(2011003155) |
| 풀이 기간 | 2016/02/06 ~ 2017/02/12 |
| 문제 내용 | \* 문제 프로그램이 실행 후 몇 초 후에 종료되는지 알아내시오 |

문제 풀이:

문제 파일을 실행 시 창 하나가 뜨게됩니다.



이 창은 생성 후 몇 초 후에 자동으로 사라지게 됩니다. 이 사라지는 시간을 알아내는 것이 목표입니다.



프로그램은 UPX로 packing 되어 있습니다.



ollydbg로 실행한 화면입니다. PUSHAD가 있으므로 stack 주소에 break를 걸어 POPAD 구간을 찾아 unpacking 합니다.



unpacking 후의 시작 주소 화면입니다. 아직은 루틴을 알기 어려우므로 계속 진행해봅니다.



진행을 하면 저번과 같이 debugger 탐색에 걸리게 됩니다. intermodular call를 통해 isDebuggerPresent 메소드를 호출하는 부분을 찾습니다.



호출 구간을 찾아 NOP 처리를 해주면 걸리지 않고 진행이 될 것입니다.

진행하다보면 주소 40EA13 에서 멈춘 후 창이 나온 후 사라지고 다음 주소로 넘어가게 됩니다. 여기서 이 주소에서 호출된 메소드가 창을 생성하고 없애는 역할을 하는 것을 알 수 있습니다.

또 메소드 내부로 들어가 라인 마다 진행하면 40B21A 주소에서 멈추게 되므로 그 내부로 들어갑니다.



40B21A 주소에서 호출되는 메소드 내부 루틴입니다. break가 걸려있는 EAX를 CALL 할때 창이 뜨고 없어지고 난 후에 다음 주소로 진행되므로 EAX에 저장되는 값을 확인해야 할 것입니다. 이 때의 EAX 값은 45DF07입니다.

이 주소에서 진행하면 이번엔 45E05F 주소에서 멈추게 됩니다.



호출되는 내부 메소드로 들어가보면 MessageBoxW 메소드가 보이는데 444DB8을 호출할 경우 창이 뜨고 그 후 꺼집니다. 지금까진 몇 초를 경과시키는 코드가 보이지 않으므로 전에 사용된 TimeGetTime 메소드가 호출되는지를 확인해봅니다.



검색결과 함수가 호출되는 부분이 있어 break를 걸고 진행합니다.



진행해보면 분기점을 만나게 되고 거기서 EAX값과 비교하는데 EAX는 TimeGetTime의 값이고 444C71 지점은 sleep 부분이므로 여기서 EBX + 4의 값이 시간이 걸리는 값일 것입니다.



비교 당시의 값입니다. 스택의 값 337B(=13179ms)이 창이 존속하는 시간입니다.