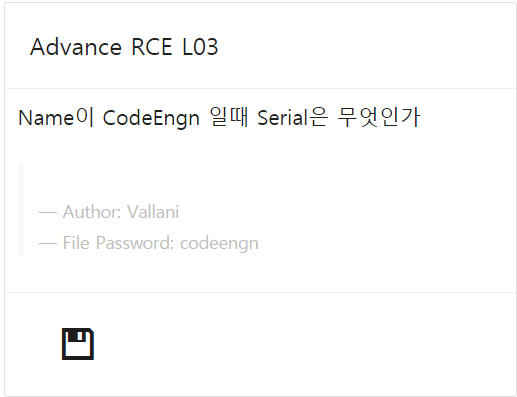
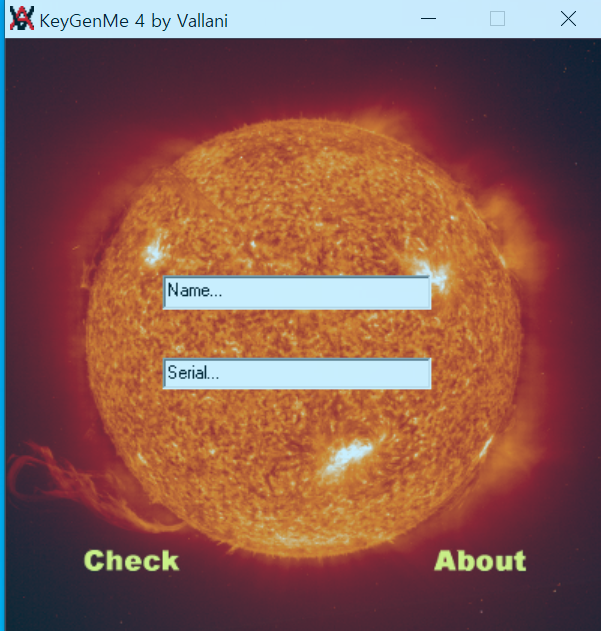
[L03]

1. 문제 확인





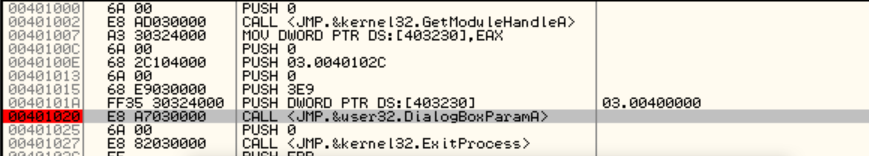
2. 과정

해당 exe 파일을 열어봤을 때 다음과 같이 Name과 Serial을 입력하는 키가 있었다.

두 값이 답에 일치하면 해결할 수 있는 것 같다.

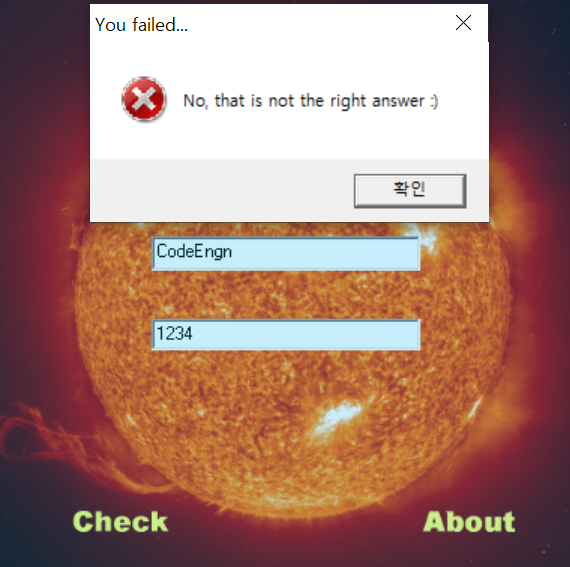
우선 앞서 계속 사용했던 올리디버거로 파일을 열어보았다.

F8을 눌러 코드를 한 줄씩 실행시켜보았다.



그러다가 call jmp 코드에 가면 문제 파일이 열리면서 name과 시리얼 값을 받아 가는 것 같다.

일단 name(=CodeEngn)과 시리얼 값(내가 지정한 1234)를 입력했다.



다음과 같이 체크를 누르면 정답이 아니라는 alert 창이 뜨는 것을 확인할 수 있는데

문제는 check를 클릭하면 올리디버거의 코드가 다른 곳으로 이동할 줄 알았는데

아무런 반응이 없다.

그래서 breakpoint를 걸어 두고 다시 시도해보았지만 똑같이 반응을 보이지 않았고

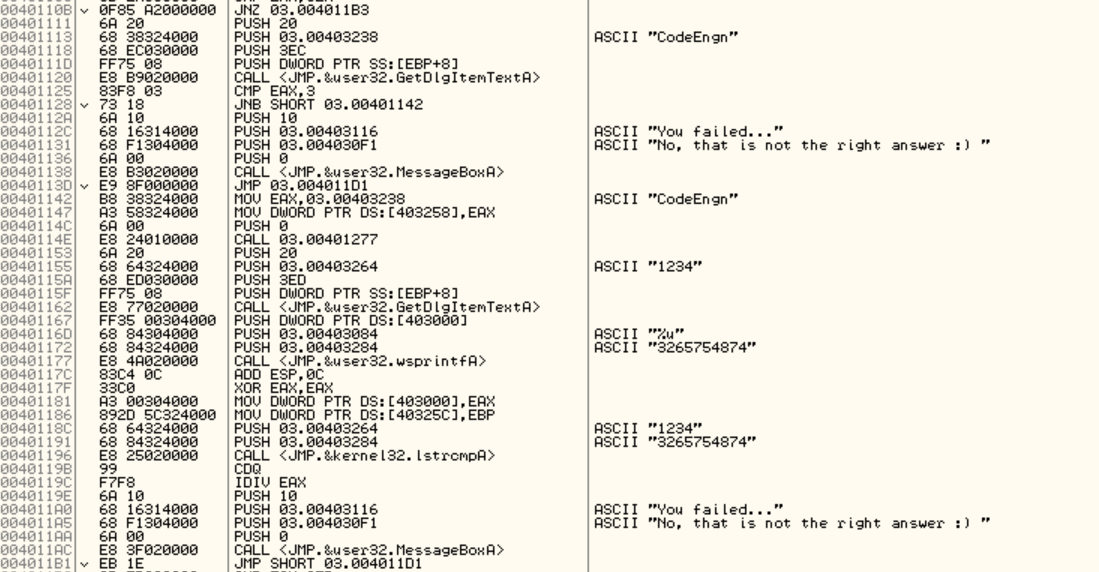
아직까지 해결법을 찾질 못했다.

그래서 다른 writeup 풀이를 살펴보며 문제를 해결해보았다.

다시 Name과 시리얼을 입력하고 401111 주소로 이동해보면

아래와 같이 성공 문자열과 실패 문자열을 확인할 수 있는데

내가 입력한 CodeEngn과 1234값이 대입된 것을 볼 수 있다.



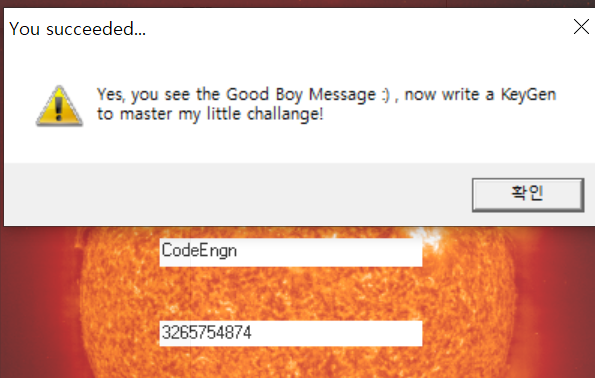
여기서 40118C를 확인해보면 PUSH 값으로 1234와 3265754874를 넣고

strcmpA 함수를 사용하는 것을 확인할 수 있는데

strcmp는 두 값을 비교하는 함수인 것을 알 수 있다.

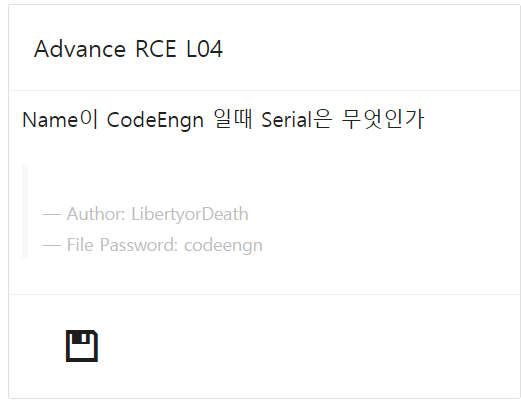
이 두 값을 비교해 일치하면 성공 문자열을 불러올 것이라는 생각이 들었다.

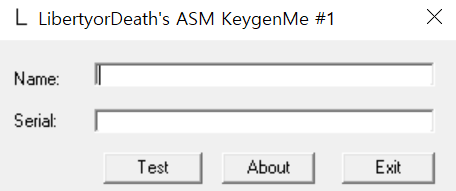
3. 결과



[L04]

1. 문제 확인





2. 과정

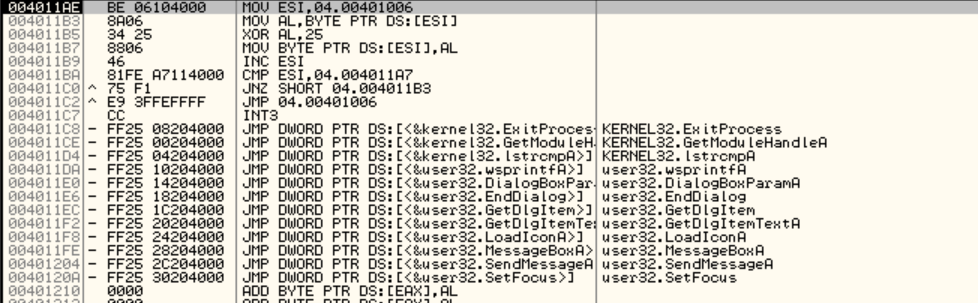
아까 문제처럼 name과 시리얼 값을 받는데

해결 방법도 비슷할 것 같다는 느낌이 든다.

올리디버거로 문제를 열어보았다.



처음 코드부터 jmp로 시작하면서 다른 곳으로 이동하게 된다.

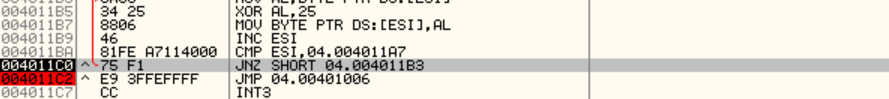


해당 주소로 들어와보니 다음과 같았고, 코드를 쓱 보니

벌써 눈에 strcmp 함수가 눈에 들어왔다.

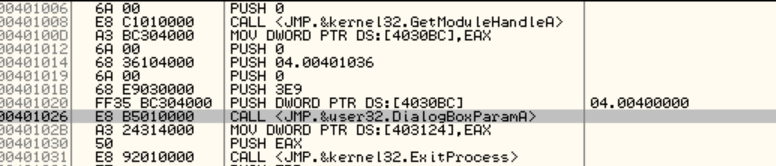
저게 중요 포인트일 것 같았다.

일단 계속 F8을 눌러 실행시켜보았다.



다음과 같은 루프를 탈출하면 루프아래 JMP로 인해 끝나는 지점으로 이동하기 때문에

Breakpoint를 걸어 두고 탈출했다.

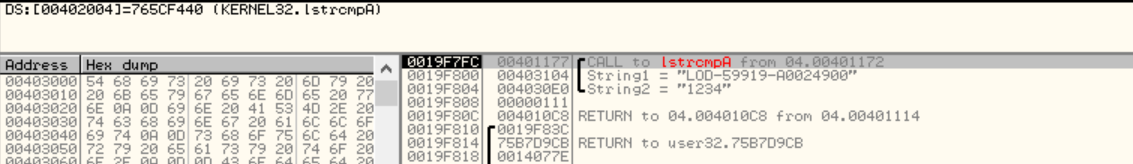


위 코드 (Call jmp)로 실행하면 name 값과 시리얼 값을 요구하는데

여기에 일단 CodeEngn(주어진 값)과 1234를 입력해 전송했다.

그러면 해당 문제가 종료포인트로 이동하게 된다.

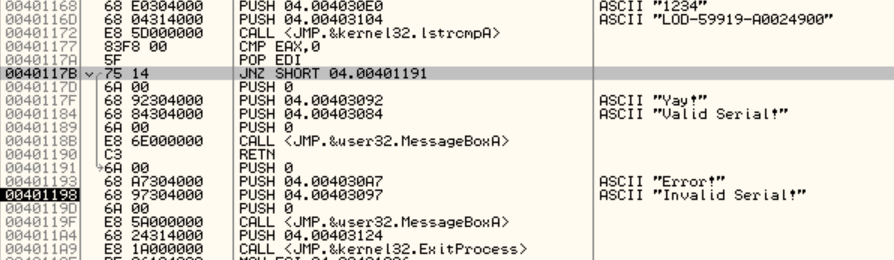
다시 돌아와 문제의 strcmp 코드 지점에 breakpoint를 일단 주고 다시 F8을 눌러 실행해보았다.



해당 strcmp 코드에서 멈춘 뒤 아래를 살펴보니 내가 입력한 값이 String2의 값으로 들어가 있는 것을 볼 수 있다.

아마도 String1과 String2를 비교한 뒤 성공 및 실패 문자열을 출력하는 것 같았다.

F7을 눌러 자세히 알아보며 코드를 실행해보았는데



역시나 문자열을 출력하는 곳으로 이동하는 것을 알 수 있다.

난 여기서 정답을 입력하지 않았기 때문에 Error! 문자열 쪽으로 이동했다.

이제 답을 알았으니 CodeEngn과 LOD-59919-A0024900를 입력해보았다.

3. 결과

