CodeEngn Basic RCE

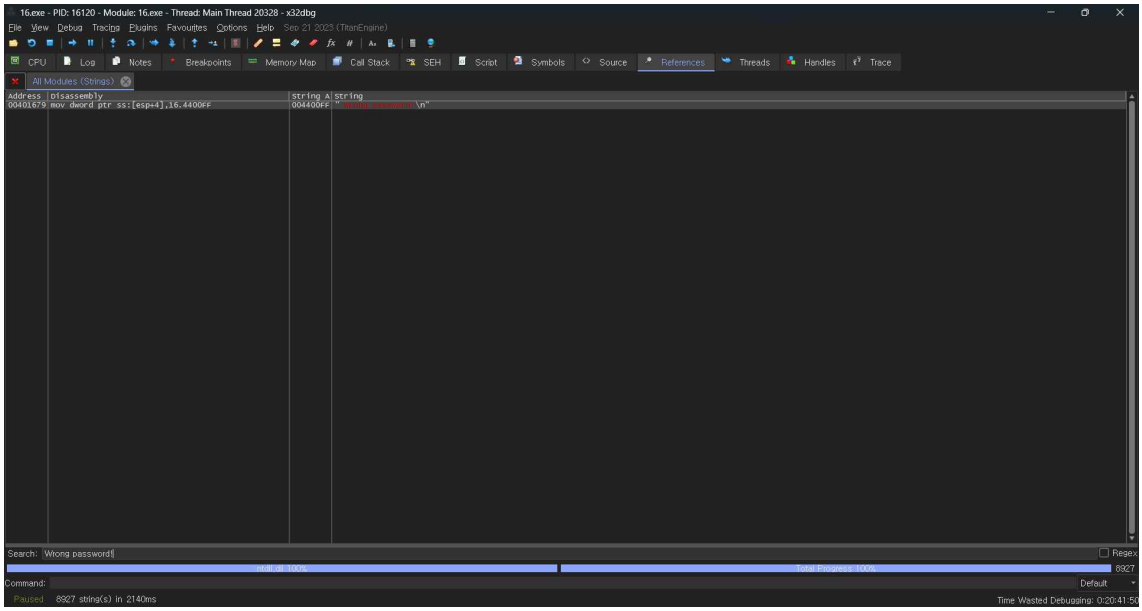
강 보 성

[16]

풀이 : 디버깅을 하기에 앞서 프로그램을 실행하여 임의의 값을 시리얼 값으로 입력해본다. 그러면 Wrong password! 라는 메시지를 확인할 수 있다.

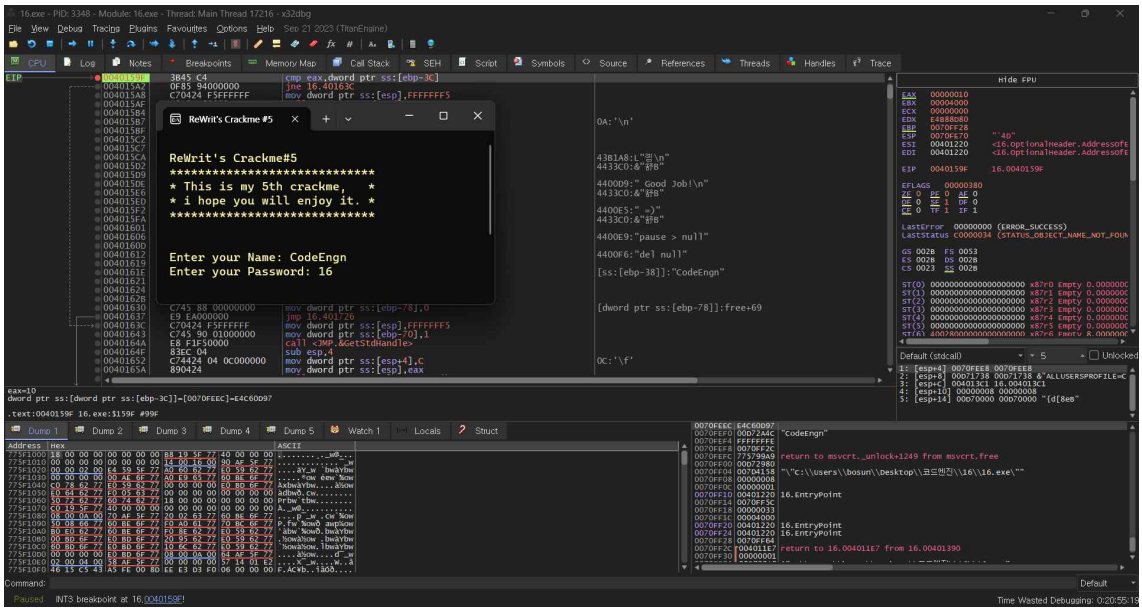
<프로그램 실행>

16.exe를 x64dbg로 열어서 Wrong password! 라는 문자열을 검색하여 해당 주소로 이동한다.



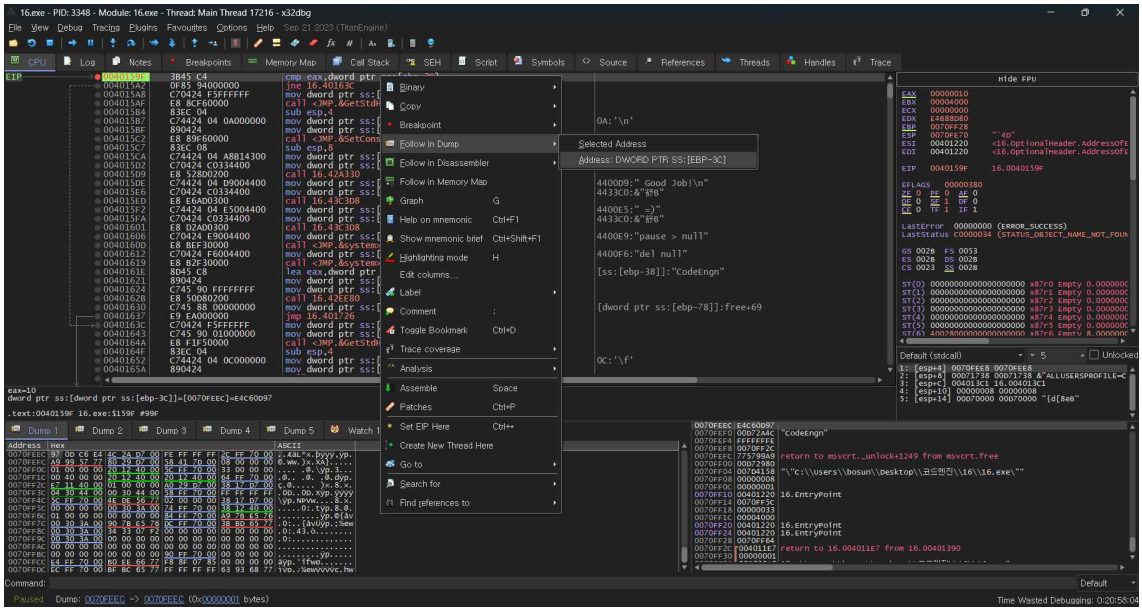
<“Wrong password!” 문자열 검색>

해당 주소로 이동하면 주변에서 성공 메시지를 발견할 수 있으며 그 위로 eax와 ss:[ebp-3C] 데이터를 비교하는 cmp 명령어를 발견할 수 있다. *(**ss**란 세그먼트 레지스터 중 하나로, 스택 영역을 의미한다. **dword ptr**은 메모리에 있는 데이터를 나타낸다. 이때 나타내는 데이터의 크기는 4바이트이다.)* 해당 명령어에 브레이크 포인트를 걸어두고 다시 실행하여 시리얼 값으로 16을 입력한다. 그러면 eax에 10이 저장됨을 확인할 수 있다.

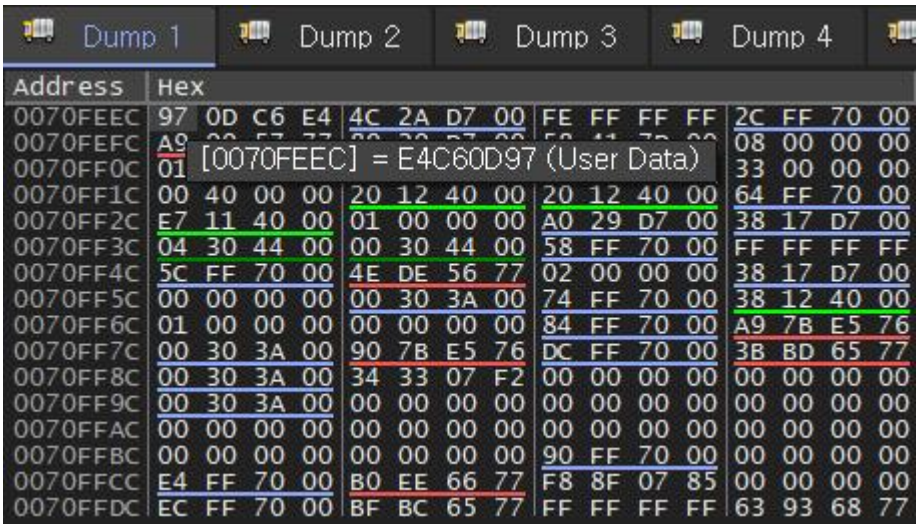


<시리얼 값으로 16을 입력 후, 변화된 eax 값>

내가 입력한 값을 16진수로 변환시키고 ss:[ebp-3C] 데이터와 비교한 뒤 같으면 성공 메시지를 출력하는 것으로 추측된다. ss:[ebp-3C] 데이터를 확인하기 위해 ss:[ebp-3C]의 덤프를 따라간다. 그러면 해당 데이터가 0xE4C60D97임을 알 수 있다.



<ss:[ebp-3C] 덤프 따라가기>



<ss:[ebp-3C] 데이터 확인(리틀엔디안)>

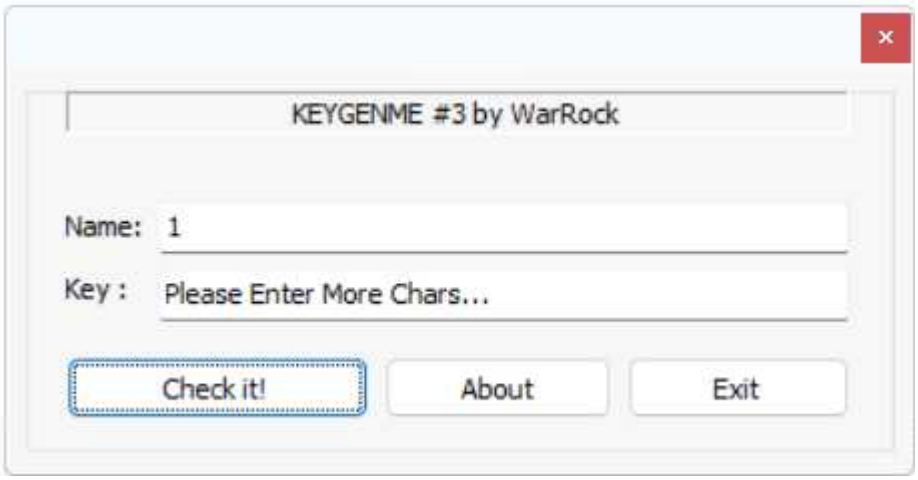
0xE4C60D97을 10진수로 변환하여 시리얼 값으로 입력해보면 성공 메시지를 확인할 수 있다.

<성공 메시지 출력>

답 : 3838184855

[17]

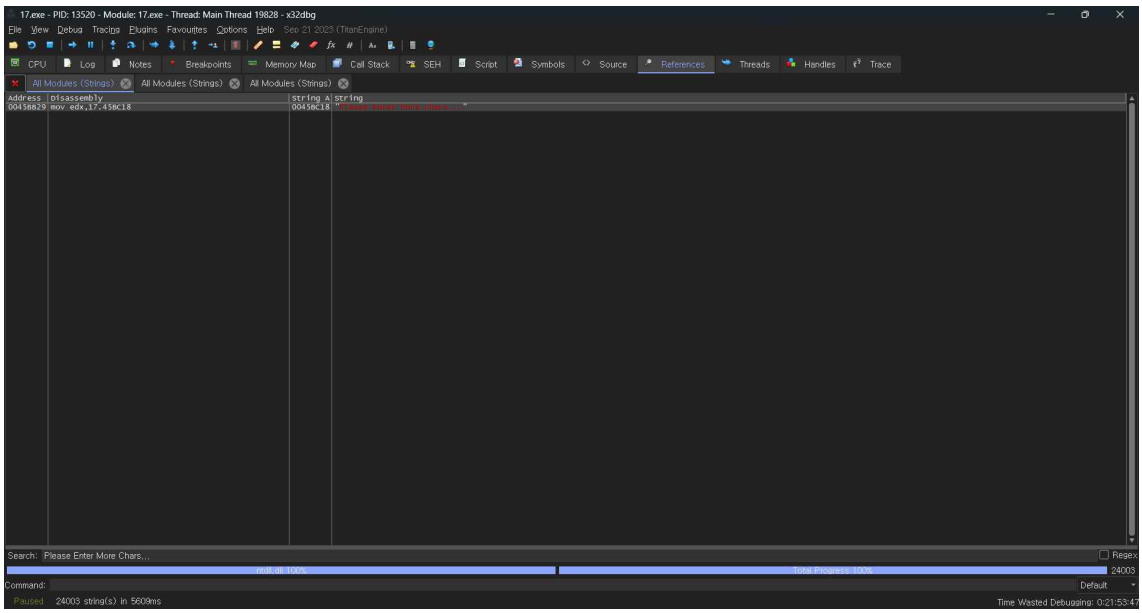
풀이 : 프로그램을 실행하여 임의의 Name을 입력해본다. 힌트에서 이름이 한자리라고 했으니 1부터 넣어보겠다.



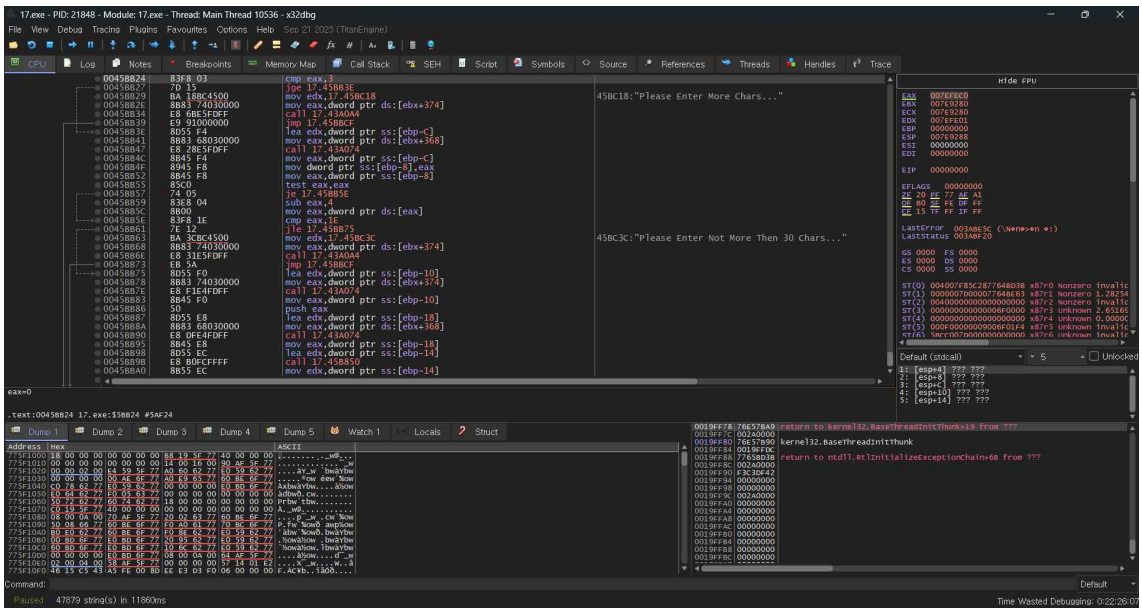
<Name으로 1을 입력한 결과>

힌트에선 Name이 한자리라고 했으나 프로그램은 더 긴 값을 요구한다. 일단 x64dbg로

디버깅을 하며 “Please Enter More Chars...”라는 문자열 데이터를 찾아본다.



<“Please Enter More Chars...” 문자열 검색>

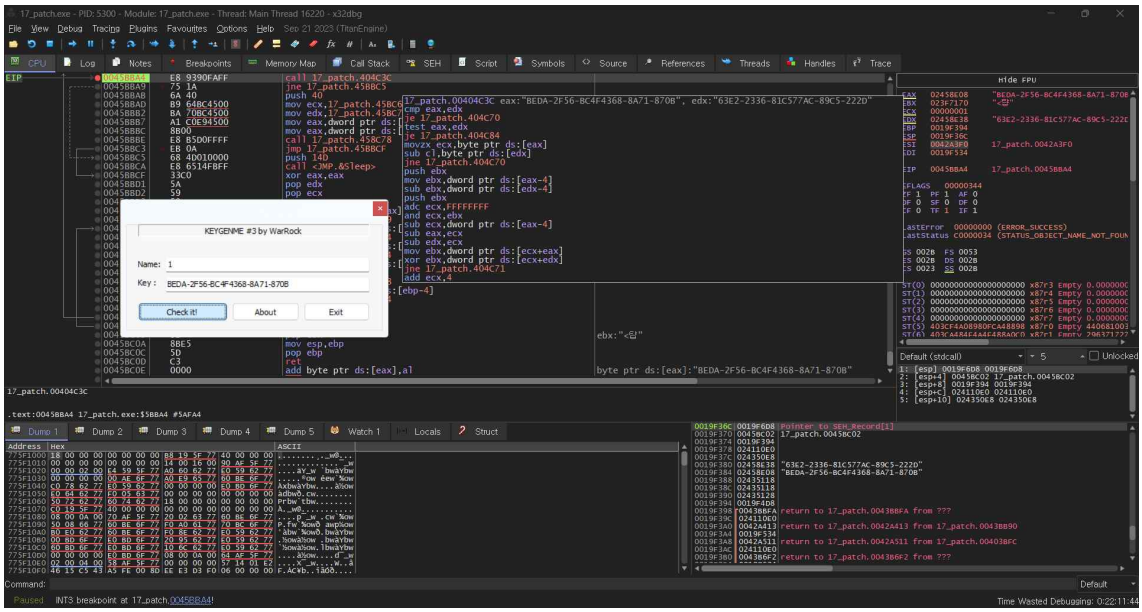


<‘cmp eax, 3’ 이후 메시지 출력>

해당 문자열 데이터 위로 ‘cmp eax, 3’이라는 코드가 보인다. 내가 입력한 값이 3자리 미만이라면 “Please Enter More Chars...”라는 메시지를 출력하는 것으로 추측된다. 정답으로 요구하는 Name 값은 한자리이므로 해당 코드를 cmp eax, 1로 수정 후 patch 한다.

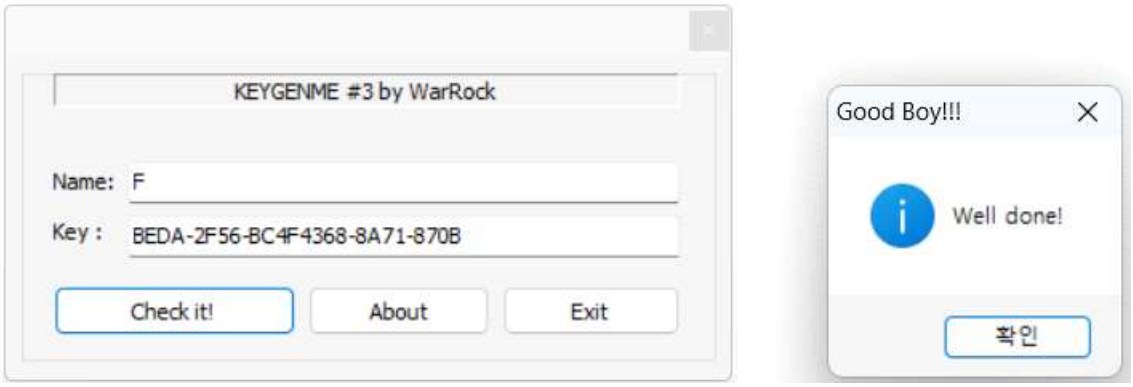
<'cmp eax, 1'로 코드 수정>

그러면 한자리 값을 입력해도 정상적으로 코드가 진행됨을 확인할 수 있다. 코드를 내려보며 분석하다 보면 성공 메시지가 보인다. 해당 메시지 위로 call 명령어가 보인다. 해당 명령어가 호출하는 함수를 살펴보면 eax와 edx를 비교하는 것을 확인할 수 있다. call 명령어에 브레이크 포인트를 걸어두고 다시 실행해 본다. 그러면 eax와 edx에 시리얼 값이 저장됨을 확인할 수 있다. 이때 eax 값은 내가 입력한 시리얼 값이고 edx 값은 Name 값에 따라 생성되는 것으로 추측된다.



<사용자 입력 후, 변화된 eax 값과 edx 값>

이제부터는 한자리 알파벳이나 숫자를 무작위로 Name에 입력하여 문제에서 제시된 시리얼 값을 생성하는 Name 값을 찾는다. (브루트 포스) Name으로 F를 입력했을 때, 성공 메시지가 출력됨을 확인할 수 있다.



<성공 메시지 출력>

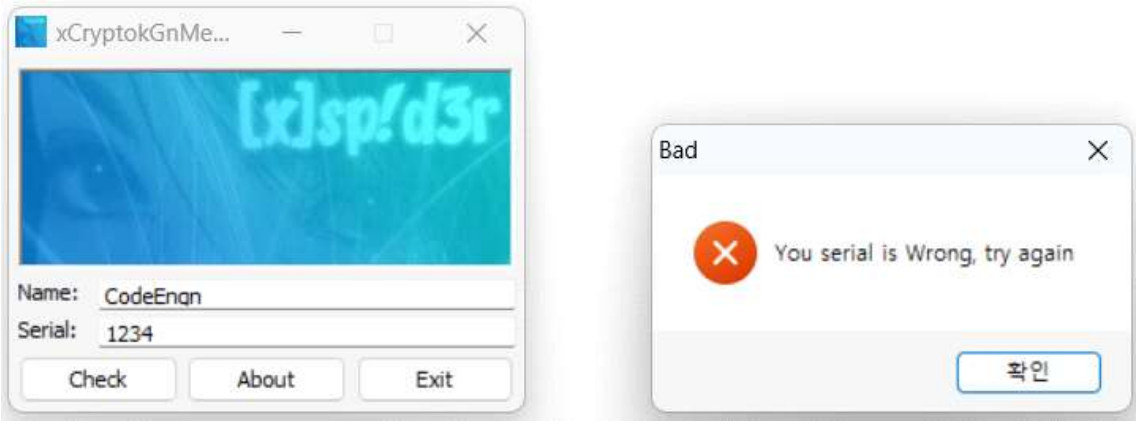
<MD5 해시 생성기에 F 대입>

F의 MD5 해시값은 8006189430253l5f869e4e1f09471012이다.

답 : 8006189430253l5f869e4e1f09471012

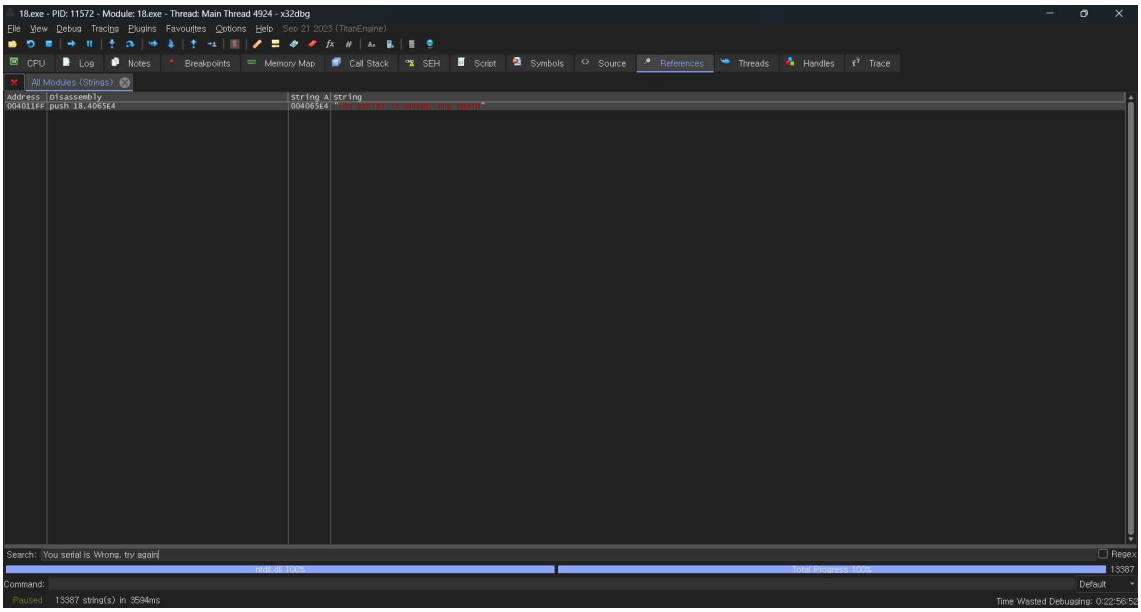
[18]

풀이 : 먼저 프로그램을 실행시키고 임의의 시리얼 값을 입력한다. 그러면 “You serial is Wrong, try again”이라는 메시지를 확인할 수 있다.

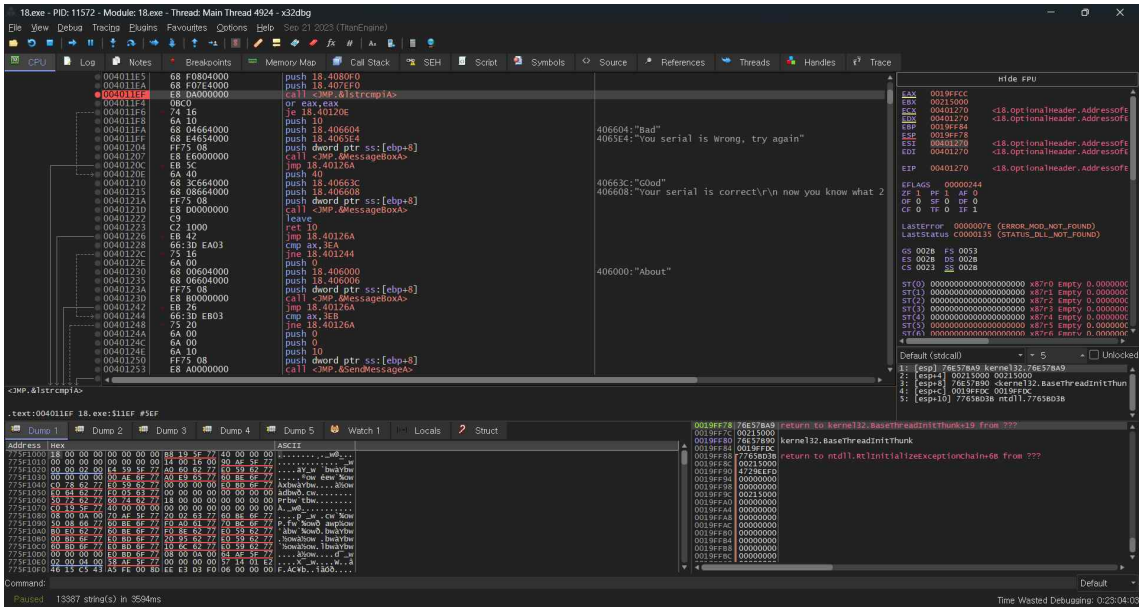


<프로그램 실행>

프로그램을 x64dbg로 디버깅하며 “You serial is Wrong, try again”이라는 문자열을 검색하여 해당 주소로 이동한다.

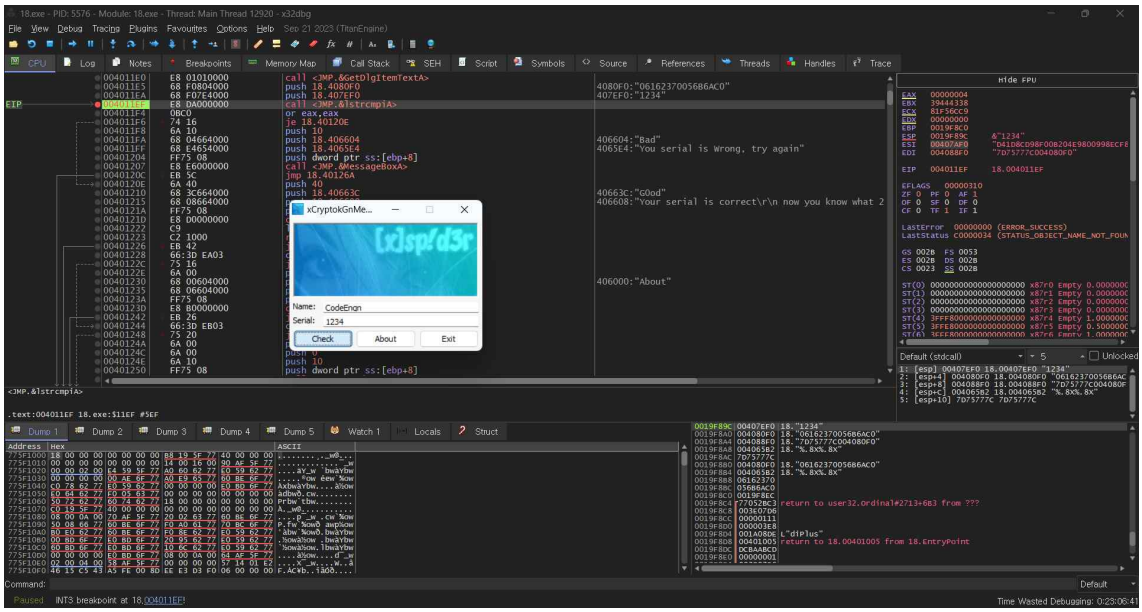


<“You serial is Wrong, try again” 문자열 검색>



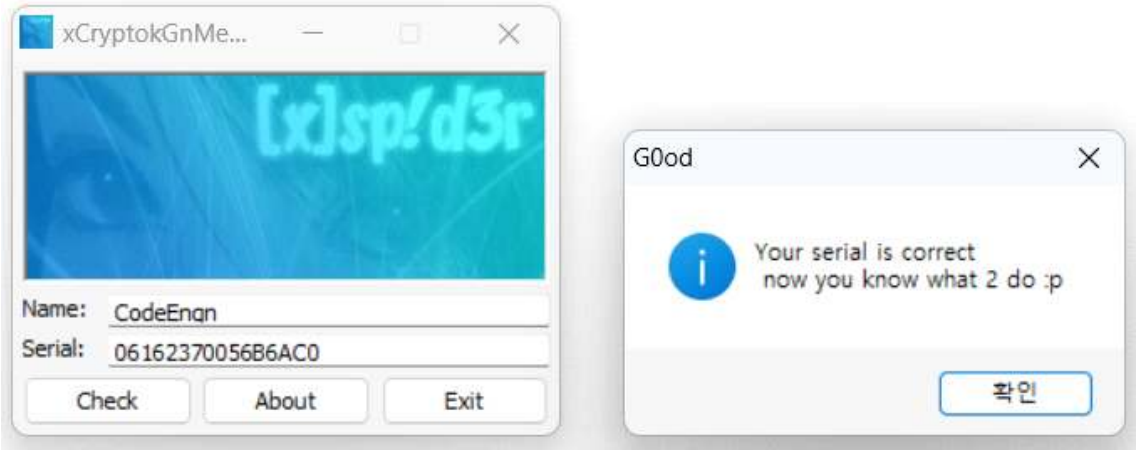
<주변 코드 분석>

해당 문자열 위쪽으로 성공 메시지와 실패 메시지로 이동하는 분기문이 보이며, 그 위로 데이터를 두 번 push하고 두 값을 비교하는 것으로 추정되는 함수 호출문이 보인다. 해당 호출문에 브레이크 포인트를 걸고 프로그램을 다시 실행시킨다.



<함수 호출 전, 두 번의 push>

앞서 확인한 바와 같이 함수를 호출하기 전에 두 번의 push가 이루어졌는데, 첫 번째로는 “06162370056B6AC0”라는 값이, 두 번째로 나의 입력값이 push 되었다. 첫 번째로 push 된 값이 시리얼 값으로 추정된다. 따라서 프로그램을 실행하고 해당 값을 입력해보면, 성공 메시지가 출력됨을 확인할 수 있다.



<성공 메시지 출력>

답 : 06162370056B6AC0