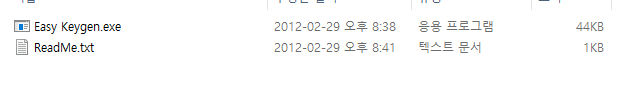
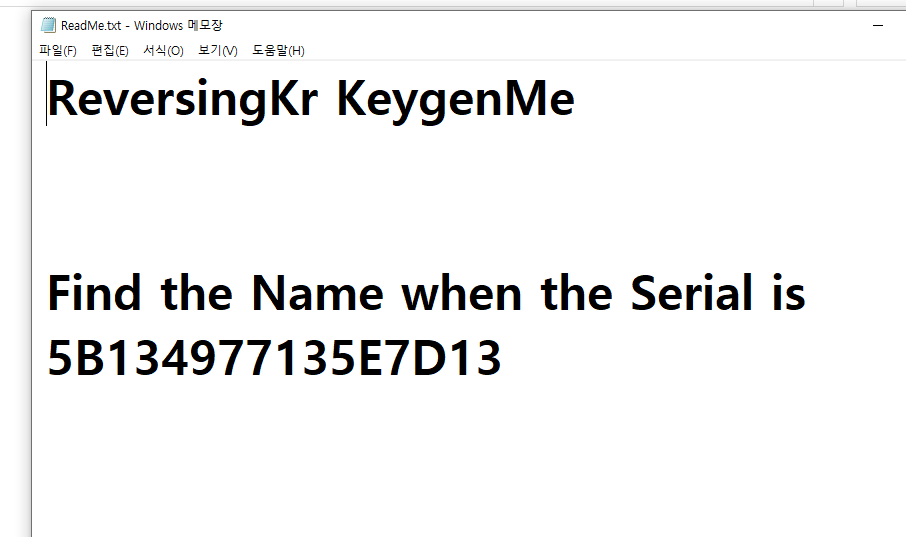
Reversing.kr – Easy Keygen

주어진 파일



ReadMe.txt

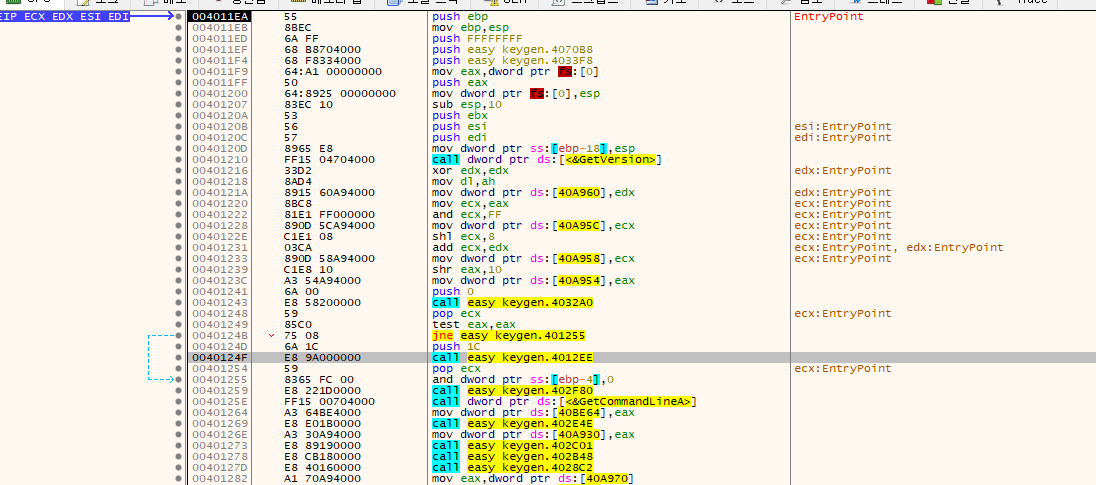


Serial 값이 주어진 값으로 나오게 하는 입력 문자열을 찾는 것이 목표.

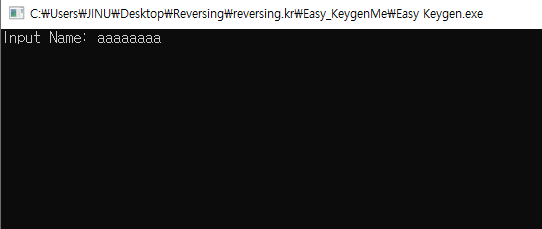
주어진 serial 값을 ascii값으로 의심되어 두 글자 씩 나누어 표시

**5B 13 49 77 13 5E 7D 13**

X32dbg로 주어진 실행 파일 분석



입력 받는 루틴에 bp 설정 후 주어진 serial 값이 8자리 이므로 8자리 문자열 입력



주어진 프로그램 구조

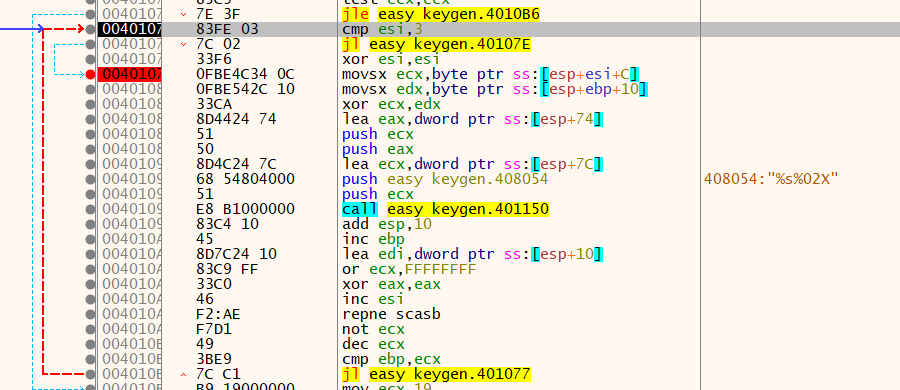
“Name” 입력 -> 입력한 문자열을 특정 규칙에 따라 새로운 문자열 생성

-> “Serial” 입력 -> 생성된 문자열과 입력한 “Serial” 값 비교 후 성공 여부 판단.

=> 두번째 과정인 새로운 문자열 생성 과정에서 적용되는 규칙 파악 후 주어진 Serial 값을 통해 역산하여 요구하는 Name 값 유추 -> Flag 값.

“aaaaaaaa”를 Name값으로 입력 시



새로운 문자열을 생성하는 부분에 bp 설정 후 초기 입력값 “abcdefgh”로 디버깅 재시도 

해당 루틴에서 명령어movsx ecx, byte ptr ss:[esp+esi\*1+0xC] 의 [esp+esi+0xC] 에

“0x10 0x20 0x30” 이 저장 되어 있으며 이 값을 순서대로 돌아가며 입력한 문자열의 문자와 xor연산을 실행한다.

Ex) 첫 번째 문자 ^ 0x10 = 새로운 문자열의 첫 번째 문자

두 번째 문자 ^ 0x20 = 새로운 문자열의 두 번째 문자

세 번째 문자 ^ 0x30 = 새로운 문자열의 세 번째 문자

네 번째 문자 ^ 0x10 = 새로운 문자열의 네 번째 문자

이 후 입력 받은 Serial 값과 비교하는 연산을 거치므로 주어진 문자열

**5B 13 49 77 13 5E 7D 13**

을 위 과정을 다시 거치면

**4B 33 79 67 33 6E 6D 33**

이고 이를 다시 ascii문자열로 바꾸면

**K3yg3nm3**

이며 이 입력값으로 프로그램을 재 실행 시키면 결과는 다음과 같다.



정상 종료 되므로

Flag = “K3yg3nm3”