코드엔진 Basic RCE L20

문제

CodeEngn.com [코드엔진]

코드엔진은 국내 리버스엔지니어링 정보공유를 위해 2007년 부터 리버스엔지니어링 컨퍼런스 및 세미나, 워크숍을 현업 실무자들과 함께 운영하고 있는 비영리 커뮤니티입니다.

https://ch.codeengn.com/

의 Basic RCE L20

해결 과정

- 실행파일을 ollyDbg에 올려 분석을 시작합니다.
- 코드를 보면 ‘crackme3.key’ 파일이 있는지 확인하는 함수와 파일이 있을 경우 파일에 있는 데이터의 길이를 구하는 함수를 볼 수 있습니다.
- 코드를 아래로 내려보면 key파일에 있는 데이터의 길이가 0x12(=18)일때 그 다음 코드로 넘어갈 수 있는 비교코드를 확인할 수 있습니다.
- 따라서, key파일에 있는 데이터의 길이를 18로 만들어 첫 번째 비교로직을 통과합니다.
- 그 다음을 보면 두개의 값을 비교해서 그 두 값이 같아야 다음 코드로 넘어갈 수 있는 비교 로직을 볼 수 있습니다.
- 이 두개의 값 중 하나는 key파일에 있는 데이터중 마지막부터 시작하여 길이가 4인 데이터이고 나머지 하나는 특정한 값을 가진 데이터입니다.
- 따라서 key파일에 있는 데이터 중 마지막부터 시작하여 길이가 4인 데이터를 특정한 값을 가진 데이터와 똑같이 만들면 두 번째 비교 로직을 통과할 수 있습니다.
- 마지막은 성공 메시지 박스에 ‘Cracked by: CodeEngn!’ 문자열을 출력하는 것인데 코드를 보면 Cracked by: !은 이미 출력되고 있고 :과 !사이에는 key파일에 있는 데이터 중 처음부터 시작하여 길이가 14인 데이터를 출력합니다.
- 단, 출력되는 데이터는 그 전에 코드에 의해 변형이 이루어진 데이터로 그 내용은 0x41부터 시작하여 0x4E까지의 값 각각과 key파일에 있는 데이터의 한 바이트 각각씩 총 14번 XOR 연산한 후 처음부터 시작하여 key파일에 있는 데이터의 각각 한 바이트 위치에 저장한 데이터 입니다.

- 따라서 key파일에 있는 데이터의 변형을 완료했을때 CodeEngn + 공백의 아스키코드 값을 가지면 되는 것으로 XOR연산의 특성을 이용하여 CodeEngn + 공백의 각 문자의 아스키코드 값과 0x41부터 0x49까지 XOR 연산을 수행하고 나머지 6바이트는 00으로 설정해 key파일에 넣습니다.
- 마지막으로 실행파일을 실행해 보면 원하는 문자열이 성공 메시지 박스에 출력되는 것을 확인할 수 있습니다.

실행 구조

- 프로그램을 실행하면 crackme3.key파일이 있는지 확인합니다.
- 있으면 다음 로직으로 이동하고 없으면 빈 메시지박스를 출력합니다.
- key파일에 있는 데이터의 길이가 18인지 확인합니다.
- 18이면 다음 로직으로 이동하고 아니면 빈 메시지박스를 출력합니다.
- key파일에 있는 데이터에 해결과정에서 쓴 방법대로 변형을 가하고 맨 마지막 길이가 4인 데이터와 특정한 값을 비교합니다.
- 같으면 다음 로직으로 이동하고 다르면 빈 메시지박스를 출력합니다.
- 마지막으로 ‘Cracked by: !’에서 :과 !사이에 변형시킨 14길이의 key파일 데이터를 넣어줍니다.
- 그런 후 변형시킨 14길이의 key파일 데이터를 :과 !사이에 넣은 ‘Cracked by: !’ 문자열과 추가적인 다른 문자열을 보여주는 성공 메시지 박스를 출력합니다.