

CodeEngn Basic RCE L11

: OEP를 찾으시오. Ex) 00401000 / Stolenbyte 를 찾으시오.

Ex) FF35CA204000E84D000000 정답인증은 OEP+ Stolenbyte

Ex) 00401000FF35CA204000E84D000000

1. OEP 구하기

004071F0	50	PUSHAD
004071F1	BE 00704000	MOV ESI,11.00407000
004071F6	8DBE 00A0FFFF	LEA EDI,DWORD PTR DS:[ESI+FFFFA000]
004071FC	57	PUSH EDI
004071FD	83CD FF	OR EBP,FFFFFFFF
00407200	EB 10	JMP SHORT 11.00407212
00407202	90	NOP
00407203	90	NOP
00407204	90	NOP
00407205	90	NOP
00407206	90	NOP
00407207	90	NOP
00407208	8A06	MOV AL,BYTE PTR DS:[ESI]
0040720A	46	INC ESI
0040720B	8B07	MOV BYTE PTR DS:[EDI],AL
0040720D	47	INC EDI
0040720E	01DB	ADD EBX,EBX
00407210	75 07	JNZ SHORT 11.00407219
00407212	8B1E	MOV EBX,DWORD PTR DS:[ESI]
00407214	83EE FC	SUB ESI,-4
00407217	11DB	ADC EBX,EBX
00407219	72 ED	JB SHORT 11.00407208
0040721B	B8 01000000	MOV EAX,1
0040721D	01DB	ADD EBX,EBX

Address	Hex dump	ASCII
00408000	00 00 00 00 DC 36 E5 37 00 00 00 00 00 00 01 00??.....0.
00408010	10 00 00 00 18 00 00 00 00 00 00 00 DC 36 E5 37t.....??
00408020	00 00 00 00 00 00 01 00 01 00 00 00 30 00 00 000.0.....*
00408030	00 00 00 00 DC 36 E5 37 00 00 00 00 00 00 01 00??.....0.
00408040	00 00 00 00 48 00 00 00 EC 8A 00 00 8C 02 00 00H.....?..

올리디버거에 올려보았더니 바로 pushad가 보인다.

OEP코드를 찾기 위해 PUSHAD의 짝꿍 POPAD를 찾아준다.

00407365	50	PUSH EAX
00407366	54	PUSH ESP
00407367	50	PUSH EAX
00407368	53	PUSH EBX
00407369	57	PUSH EDI
0040736A	FFD5	CALL EBP
0040736C	58	POP EAX
0040736D	61	POPAD
0040736E	6A 00	PUSH 0
00407370	68 00204000	PUSH 11.00402000
00407375	68 12204000	PUSH 11.00402012
0040737A	8D4424 80	LEA EAX,DWORD PTR SS:[ESP-80]
0040737E	6A 00	PUSH 0
00407380	39C4	CMP ESP,EAX
00407382	75 FA	JNZ SHORT 11.0040737E
00407384	83EC 80	SUB ESP,-80
00407387	E9 809CFFFF	JMP 11.0040100C
0040738C	00	DB 00
0040738D	00	DB 00

조금만 내려봤더니 POPAD 명령어가 존재해서, 앞뒤 부분의 코드를 살펴보면 OEP로 점프하는 명령문을 찾아보겠다.

해당 주소로 점프하는 명령어는 JMP와 JNZ가 있고,
 이 둘 중 하나가 OEP코드가 있는 곳으로 이동하라는 명령을 지시한다고 추측된다.
 그럼 반복되지 않는 부분이 OEP코드로 이동하는 부분이기 때문에,
 각각의 명령어를 분석해 볼 필요가 있다.

JNZ = JUMP NOT ZERO

0이 아니면 점프!이기 때문에 JNZ 바로 위 명령어인

CMP ESP, EAX는 ESP-EAX를 명령하고,

바로 밑에 JNZ는 ESP와 EAX가 같아야 이후 명령어인 SUB ESP, -80을 수행할 수 있으니깐
 JNZ명령어에서 ESP와 EAX가 같지 않아서 계속 0040737E PUSH 0 으로 이동한다는 점을
 알 수 있다.

따라서 OEP 코드가 있는 곳은 그 밑에 있는 JMP 명령어에 있는 0040100C이라는 것을 알 수
 있다.

2. Stolenbyte 구하기

stolenbyte는 OEP로 이동하기 전에 PUSH 되는 것들입니다.

아까 OEP코드를 구할 때 총 3개의 인자가 푸시된다는 것을 알 수 있었는데, 이것이 훔쳐진
 바이트를 구하는데 핵심이니 기억해두자.

각각 PUSH인자와 점프 명령어에 BP를 걸어주고 F9를 눌러 실행 시켜주었더니 아스키 코드
 문자열이 보였고, 위에서 구한 OEP주소로 이동했다.

00401006	90	NOP	
00401007	90	NOP	
00401008	90	NOP	
00401009	90	NOP	
0040100A	90	NOP	
0040100B	90	NOP	
0040100C	6A 00	PUSH 0	
0040100E	E8 8C000000	CALL 11.0040109F	JMP to USER32.MessageBoxA
00401013	6A 00	PUSH 0	
00401015	68 80000000	PUSH 80	
0040101A	6A 03	PUSH 3	
0040101C	6A 00	PUSH 0	
0040101E	6A 00	PUSH 0	
00401020	68 00000080	PUSH 80000000	
00401025	68 B9204000	PUSH 11.004020B9	ASCII "abex.l2c"
0040102A	E8 5E000000	CALL 11.0040108D	JMP to KERNEL32.CreateFileA
0040102F	A3 CA204000	MOV DWORD PTR DS:[4020CA],EAX	
00401034	83F8 FF	CMF EAX,-1	
00401037	74 3C	JE_SHORT 11.00401075	

PUSH 0 위로 NOP으로 쪽 채워진 것을 확인할 수 있는데,

그 밑에 호출하는 함수 메세지 박스는 총 4개의 파라미터를 필요로 한다.

그래서 3개의 인자가 부족하고,

이전에 OEP로 이동하기 전 3개의 PUSH된 부분이 stolenbyte라는 것을 알 수 있다.

00407365	. 50	PUSH EAX
00407366	. 54	PUSH ESP
00407367	. 50	PUSH EAX
00407368	. 53	PUSH EBX
00407369	. 57	PUSH EDI
0040736A	. FF05	CALL EBP
0040736C	. 58	POP EAX
0040736D	. 61	POPAD
0040736E	. 6A 00	PUSH 0
00407370	. 68 00204000	PUSH 11.00402000
00407375	. 68 12204000	PUSH 11.00402012
0040737A	. 8D4424 80	LEA EAX,DWORD PTR SS:[ESP-80]
0040737E	> 6A 00	PUSH 0
00407380	. 39C4	CMP ESP,EAX
00407382	. ^75 FA	JNZ SHORT 11.0040737E
00407384	. 83EC 80	SUB ESP,-80
00407387	. -E9 809CFFFF	JMP 11.0040100C
0040738C	. 00	DB 00
0040738D	. 00	DB 00

6A0068002040006812204000 이라는 것을 알 수 있다.

따라서 11번의 키는 0040100C6A0068002040006812204000 이다.