

CodeEngn Basic RCE L09

: StolenByte를 구하시오 Ex) 75156A0068352040

\*STOLENBYTE란?

- 훔친 바이트란 의미로 프로그램의 한부분의 코드를 훔쳐내어 다른 부분으로 옮겨진 코드를 말합니다.

주로 옮겨지는 코드는 엔트리 포인트위의 몇개의 코드들이며 옮겨진 코드들은 OEP 주소로 점프하기 전에 위치에서 PUSH 됩니다.

이러한 StolenByte 는 주로 패커가 프로그램을 패킹할때 볼수있습니다. 이렇게 옮겨진 코드들은 할당된 메모리 공간에서 실행됩니다.

이때문에 패킹된 프로세스가 덤프될때 StolenByte를 복구하지 못하면 프로그램은 정상적으로 작동하지 못하게 됩니다.

출처: <https://stih.tistory.com/67> [STIH]

엔트리 포인트 위 몇개의 코드들 & OEP 주소로 점프하기전 위치에서 PUSH 인 걸로 보아, POPAD로 OEP코드를 찾고 그 위에 있는 코드들을 살펴보면 되겠다는 생각이 듭니다 !!

Address	Disassembly
004071F0	PUSHAD
004071F1	MOV ESI,09.004
004071F6	LEA EDI,DWORD
004071FC	PUSH EDI
004071FD	OR EBP,FFFFFFFF
00407200	JMP SHORT 09.0
00407202	NOP
00407203	NOP
00407204	NOP
00407205	NOP
00407206	NOP
00407207	NOP
00407208	MOV AL,BYTE PTR
0040720A	INC ESI
0040720B	MOV BYTE PTR D
0040720D	INC EDI
0040720E	ADD EBX,EBX
00407210	JNZ SHORT 09.0
00407212	MOV EBX,DWORD

올리디버거로 틀면 PUSHAD가 나오고, 패킹되어 있으니깐 POPAD를 찾으면 하나의 POPAD가 존재하는 걸 볼 수 있습니다.

00407357	. 8D87 1F020000	LEA EAX,DWORD PTR DS:[EDI+21F]
0040735D	. 8020 7F	AND BYTE PTR DS:[EAX],7F
00407360	. 8060 28 7F	AND BYTE PTR DS:[EAX+28],7F
00407364	. 58	POP EAX
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 09.00402000
00407375	. 68 12204000	PUSH 09.00402012
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 09.0040737E
00407384	. 83EC 80	SUB ESP,-80
00407387	. -E9 809CFFFF	JMP 09.0040100C
0040738C	. 00	DB 00
0040738D	. 00	DB 00

POPAD로 이동한 화면

0040737E ~ 00407382의 어셈블리어를 보면 JNZ 명령으로 계속 LOOP를 돌고 있는 것을 볼 수 있습니다.

그렇다면 OEP로 이동하는 주소는 루프를 빠져나와 점프하게 되는 주소인 0040100C인 것을 알 수 있어요 !

0040736D	. 61	POPAD	
0040736E	. 6A 00	PUSH 0	
00407370	. 68 00204000	PUSH 09.00402000	ASCII "abex" 3rd crackme"
00407375	. 68 12204000	PUSH 09.00402012	ASCII "Click OK to check for the keyfile."
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 09.0040737E	
00407384	. 83EC 80	SUB ESP,-80	
00407387	. -E9 809CFFFF	JMP 09.0040100C	
0040738C	. 00	DB 00	
0040738D	. 00	DB 00	

STOLENBYTE는 그 위에 PUSH되는 것들이라 했으니 PUSH와 OEP이동코드에 BP를 걸어 주었고, F9로 실행 시 ASCII코드가 나옵니다.

00401000	90	NOP	
00401001	90	NOP	
00401002	90	NOP	
00401003	90	NOP	
00401004	90	NOP	
00401005	90	NOP	
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 09.0040109F	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 80000080	
00401025	68 B9204000	PUSH 09.004020B9	ASCII "abex.l2c"
0040102A	E8 5E000000	CALL 09.0040108D	JMP to KERNEL32.CreateFileA
0040102F	A3 CA204000	MOV DWORD PTR DS:[4020CA],EAX	
00401034	CC	CMP EAX, 1	

OEP 주소로 이동해서 자동으로 UNPACK을 해주었고, ASCII코드를 보면 EXE파일이 대충 어떤 문자열을 출력하는 지 볼 수 있어요!

OEP 시작 부분 바로 밑에 MessageBox가 있는데 이 함수는 파라미터가 4개인데, 위 쪽을 보면 PUSH 0 이외에 다 NOP으로 채워져 있습니다.

따라서 총 3개의 파라미터가 부족하고, 이전에 OEP 주소 위쪽에 있던 3개가 STOLEN BYTE 인것을 알 수 있습니다 ㅎㅎ

00407360	. 61	POPAD	
00407361	. 6A 00	PUSH 0	
00407370	. 68 00204000	PUSH 09.00402000	ASCII "abex' 3rd crackme"
00407375	. 68 12204000	PUSH 09.00402012	ASCII "Click OK to check for the keyfile."
0040737A	. 8D4424 80	LEA EAX, DWORD PTR SS:[ESP-80]	
0040737E	> 6A 00	PUSH 0	
00407380	. 39C4	CMPL ESP, EAX	
00407382	. ^75 FA	JNZ SHORT 09.0040737E	
00407384	. 83EC 80	SUB ESP, -80	
00407387	. E9 809CFFFF	JMP 09.0040100C	
0040738C	00	DB 00	
0040738D	00	DB 00	

STOLEN BYTE는 6A00680020400006812204000 이네요 :)