

# CodeEngn Basic RCE L04 Writeup



Daniel Smith

Feb 18 · 3 min read

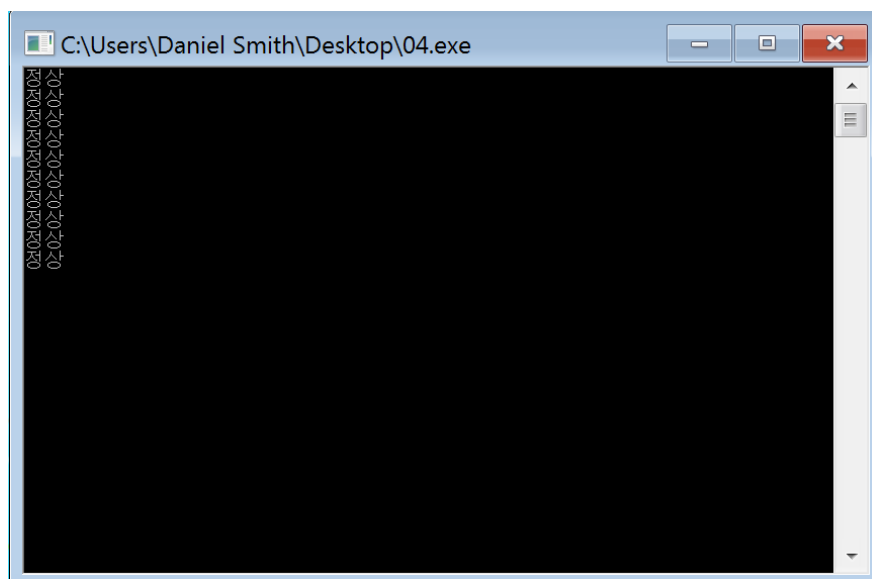
Let's start RCE

Filename: 04.exe

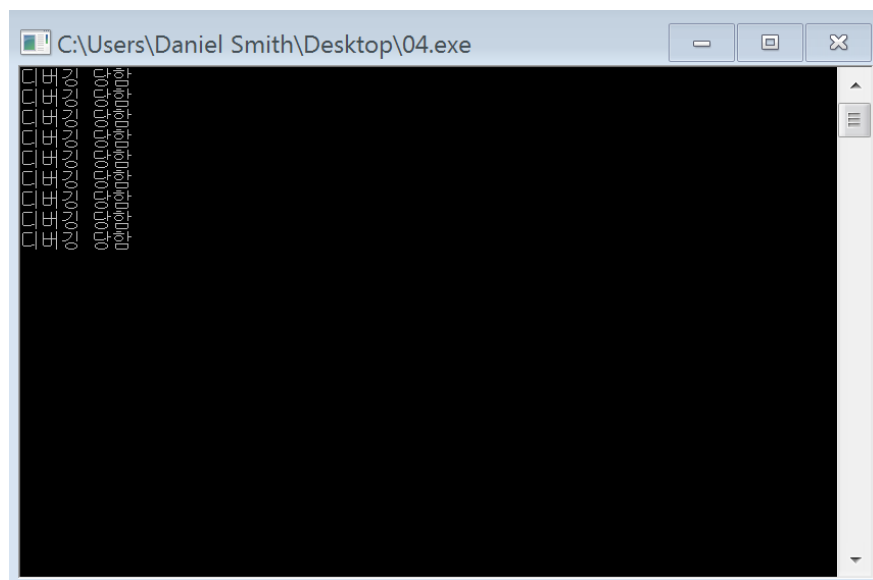
Description: 이 프로그램은 디버거 프로그램을 탐지하는 기능을 갖고 있다. 디버거를 탐지하는 함수의 이름은 무엇인가

Author: CodeEngn

## Run



Normal



If got debugged

## Analysis

```
0040104F CALL NEAR DWORD PTR DS:[<&KERNEL32.Sleep,kerne132.Sleep
0040105E CALL NEAR DWORD PTR DS:[<&KERNEL32.IsDel kerne132.IsDebuggerPresent
00407D05 CALL NEAR DWORD PTR DS:[<&KERNEL32.Multi kerne132.MultiByteToWideChar
```

프로그램상에서 호출하는 API 들을 살펴보았을 때, 디버거를 탐지 하는 함수는 IsDebuggerPresent 이라는 것을 알 수 있다.

분석하기 전에, MSDN에서 IsDebuggerPresent 함수의 반환 값을 알아 보자.

# IsDebuggerPresent function

Determines whether the calling process is being debugged by a user-mode debugger.

## Syntax

C++

```
BOOL WINAPI IsDebuggerPresent(void);
```

## Parameters

This function has no parameters.

## Return value

If the current process is running in the context of a debugger, the return value is nonzero.

If the current process is not running in the context of a debugger, the return value is zero.

MSDN

IsDebuggerPresent 함수는 해당 프로세스가 디버거에서 실행되고 있을 때는 0 이 아닌 값을 반환하고, 그렇지 않을 경우에는 0 을 반환한다

<pre> 00401041 . B8 CCCCCCCC MOV EAX,CCCCCCCC 00401046 . F3:AB REP STOS DWORD PTR ES:[EDI] 00401048 &gt; 8BF4 MOV ESI,ESP 0040104A . 68 E8030000 PUSH 3E8 0040104F . FF15 68B14300 CALL NEAR DWORD PTR DS:[&lt;&amp;KERNEL32.Sleep] Sleep 00401055 . 3BF4 CMP ESI,ESP 00401057 . E8 B4710000 CALL 04.00408210 0040105C . 8BF4 MOV ESI,ESP 0040105E . FF15 64B14300 CALL NEAR DWORD PTR DS:[&lt;&amp;KERNEL32.IsDebuggerPresent] IsDebuggerPresent 00401064 . 3BF4 CMP ESI,ESP 00401066 . E8 A5710000 CALL 04.00408210 0040106B . 85C0 TEST EAX,EAX </pre>	<p>Registers (FPU)</p> <pre> EAX 00000000 ECX 75B5189F KERNELBA.75B5189F EDX 77B46C04 ntdll.KiFastSystemCallRet EBX 7FFD9000 ESP 0012FEFC EBP 0012FF48 ESI 0012FEFC EDI 0012FF48 EIP 0040105E 04.0040105E C 0 ES 0023 32bit 0(FFFFFFFF) </pre>
---	--

IsDebuggerPresent 호출 이후 레지스터들의 변화를 관찰하기 위해 해당 함수에 BP 를 걸었다

```

00401048 > 8BF4 MOV ESI,ESP
0040104A . 68 E8030000 PUSH 3E8 [Timeout = 1000. ms]
0040104F . FF15 68B14300 CALL NEAR DWORD PTR DS:[<&KERNEL32.Sleep Sleep]
00401055 . 3BF4 CMP ESI,ESP
00401057 . E8 B4710000 CALL 04.00408210
0040105C . 8BF4 MOV ESI,ESP
0040105E . FF15 64B14300 CALL NEAR DWORD PTR DS:[<&KERNEL32.IsDebuggerPresent IsDebuggerPresent]
00401064 . 3BF4 CMP ESI,ESP
00401066 . E8 A5710000 CALL 04.00408210
0040106B . 85C0 TEST EAX,EAX
0040106D . 74 0F JE SHORT 04.0040107E
0040106F . 68 24104300 PUSH 04.00431024 [Arg1 = 00431024]
00401074 . E8 17710000 CALL 04.00408190 04.00408190
00401079 . 83C4 04 ADD ESP,4
0040107C . EB 0D JMP SHORT 04.0040108B
0040107E > 68 1C104300 PUSH 04.0043101C [Arg1 = 0043101C]
00401083 . E8 08710000 CALL 04.00408190 04.00408190
00401088 . 83C4 04 ADD ESP,4
0040108B > EB BB JMP SHORT 04.00401048

```

Registers (FPU)	
EAX	00000001
ECX	75B5189F KERN
EDX	77B46C04 ntdl
EBX	7FFD9000
ESP	0012FEFC
EBP	0012FF48
ESI	0012FEFC
EDI	0012FF48
EIP	0040106D 04.0
C 0	ES 0023 32bi
P 0	CS 001B 32bi
A 0	SS 0023 32bi
Z 0	DS 0023 32bi
S 0	FS 003B 32bi
T 0	GS 0000 NULL
D 0	
O 0	LastErrr ERRO

0x0040106D 까지 실행 해본 결과, 예상대로 반환값은 1이 되었고, test(AND 연산 수행) 명령으로 인해 ZF 가 unset 되었고 결과적으로 현재 JE 명령에서 점프를 안할 것이다

디버깅을 당했을때 “디버깅 당함” 을 출력하므로, 0x0040106D 에서 점프하는곳(0x0040107E)이 “정상” 문자열을 출력 하는 구간일것이다

즉, EAX 를 0으로 바꾸거나, test 명령에서 0 과 0 을 비교한다면 ZF 가 unset 되지 않아 “정상” 문자열을 출력하는 곳으로 점프하게 될것이다

EAX 를 0으로 바꿔보자

```

0040105E . FF15 64B14300 CALL NEAR DWORD PTR DS:[<&KERNEL32.IsDebuggerPresent IsDebuggerPresent]
00401064 . 33C0 XOR EAX,EAX
00401066 . E8 A5710000 CALL 04.00408210
0040106B . 85C0 TEST EAX,EAX
0040106D . 74 0F JE SHORT 04.0040107E

```

Patched

IsDebuggerPresent 호출 이후 아래 CMP 명령에서 레지스터의 변화가 없어서 CMP 명령을 XOR EAX, EAX 로 패치하였다

```

0040105E . FF15 64B14300 CALL NEAR DWORD PTR DS:[<&KERNEL32.IsDebuggerPresent IsDebuggerPresent]
00401064 . 33C0 XOR EAX,EAX
00401066 . E8 A5710000 CALL 04.00408210
0040106B . 85C0 TEST EAX,EAX
0040106D . 74 0F JE SHORT 04.0040107E
0040106F . 68 24104300 PUSH 04.00431024

```

이제 디버거에서 실행시켜도 “디버깅 당함” 이 아닌 “정상” 이란 문자열을 출력한다