

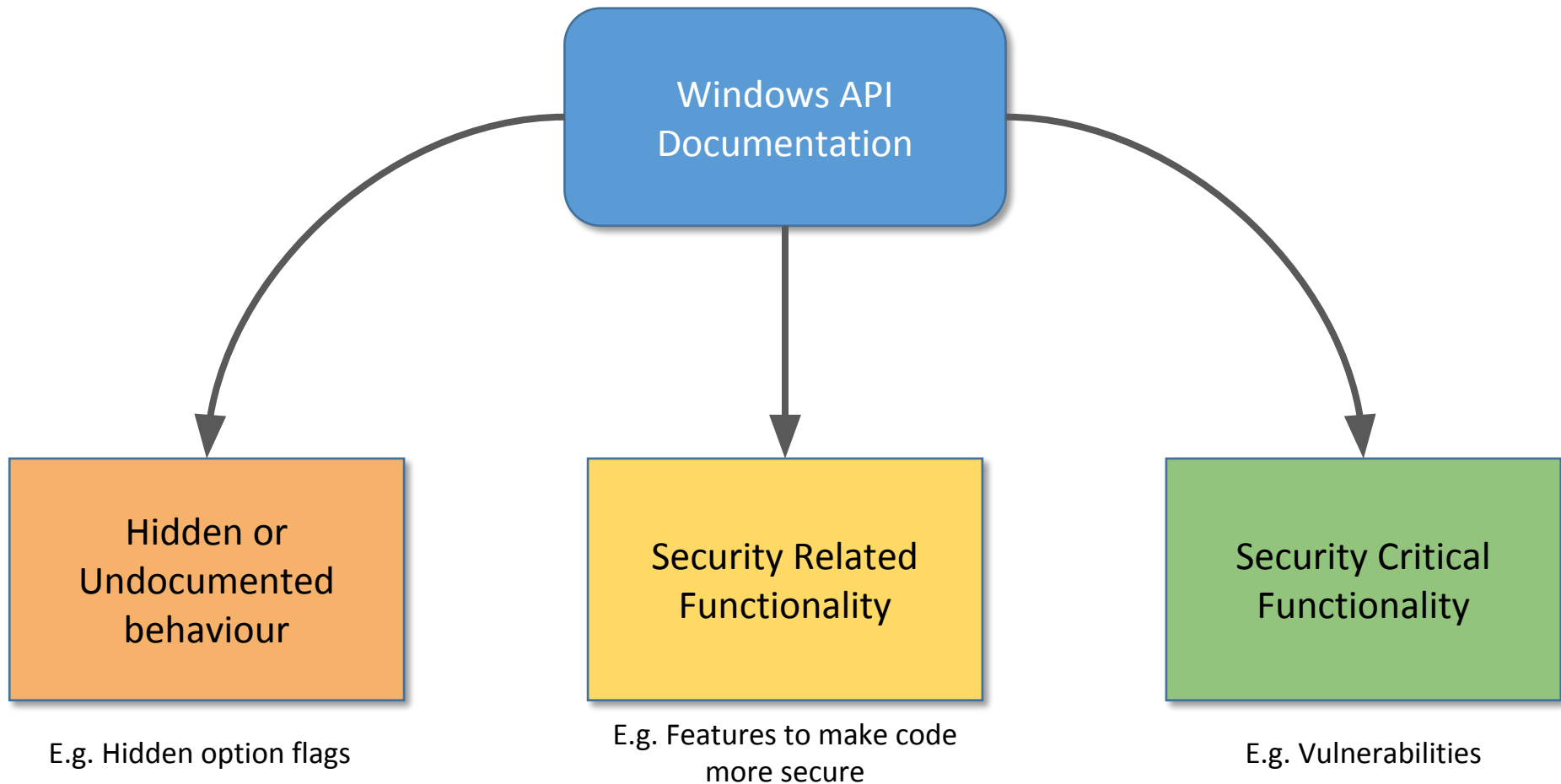


Documented to Fail

How I Stopped Worrying and Learned to Love the MSDN

James Forshaw - @tiraniddo
Ruxcon 2016

What I'm Going to be Talking About



The Problem of Documentation

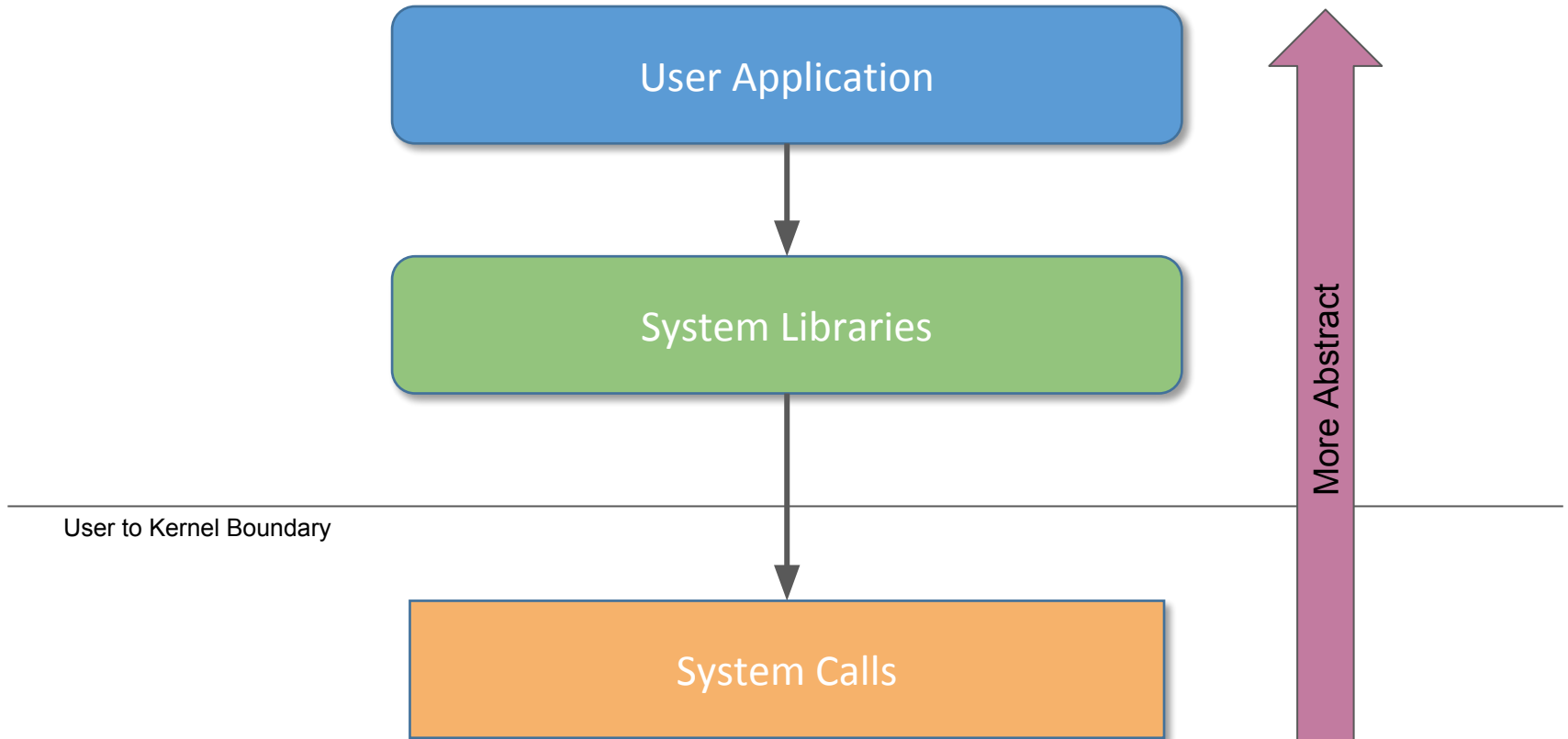
Limited space

Not up to date

Hides
Implementation
Details

Wildly Inaccurate

System Layering and Abstraction



Typical POSIX Application

```
// C library Open File API  
FILE* fopen(const char* pathname, const char* mode);
```



```
// POSIX Open File API (System Call)  
int open(const char* pathname, int flags, mode_t mode);
```

Typical POSIX Application

```
// C library Open File API  
FILE* fopen(const char* pathname, const char* mode);
```

Libc Mode to
System Call
Arguments

```
// POSIX Open File API (System Call)  
int open(const char* pathname, int flags, mode_t mode);
```

Win32 API Complexity

```
// C library Open File API  
FILE* fopen(const char* pathname, const char* mode);
```



```
// Win32 Open File APIs  
HANDLE CreateFile(  
    LPCTSTR                lpFileName,  
    DWORD                  dwDesiredAccess,  
    DWORD                  dwShareMode,  
    LPSECURITY_ATTRIBUTES lpSecurityAttributes,  
    DWORD                  dwCreationDisposition,  
    DWORD                  dwFlagsAndAttributes,  
    HANDLE                 hTemplateFile  
);
```

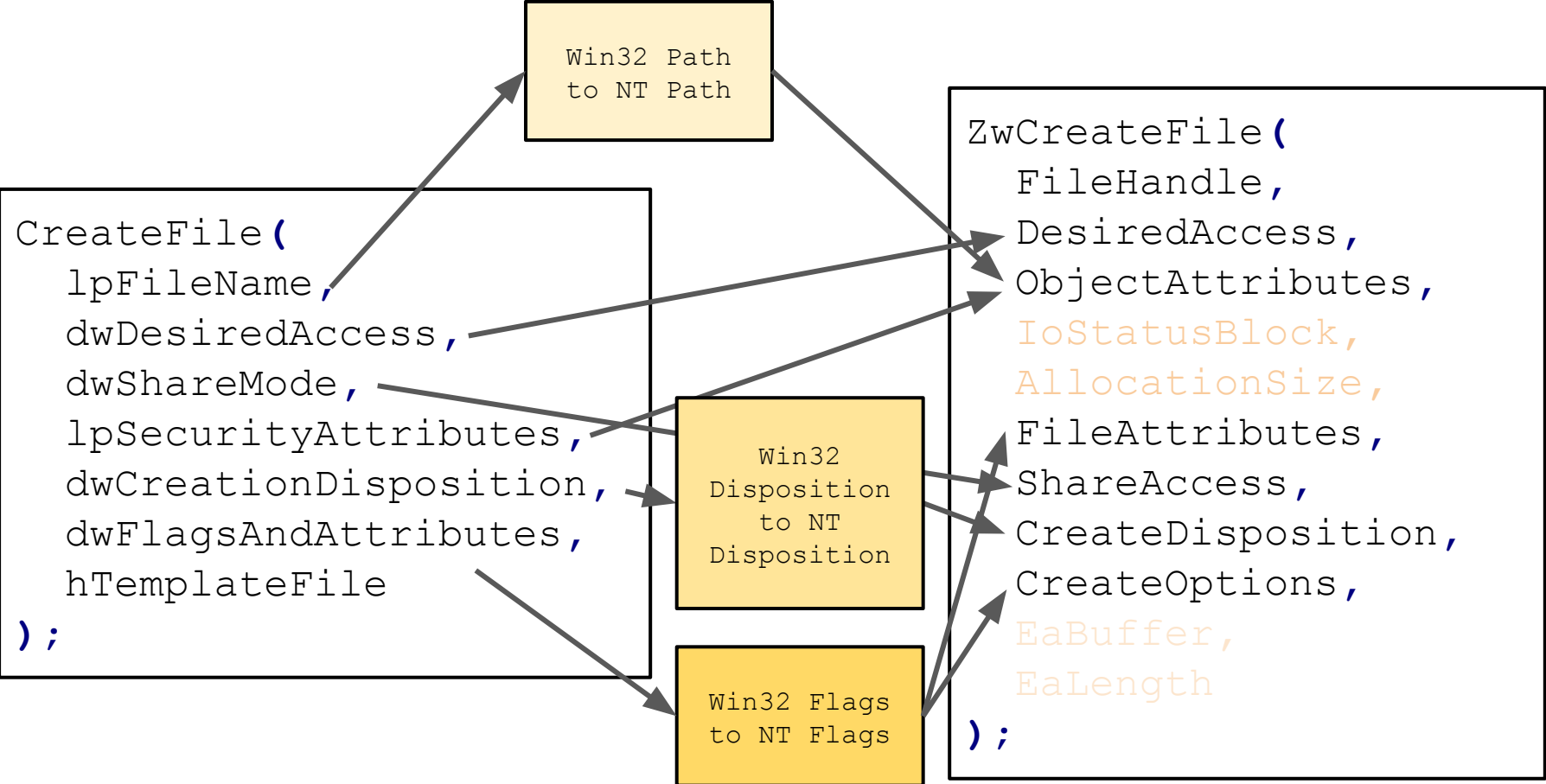
Win32 API Complexity

```
// C library Open File API  
FILE* fopen(const char* pathname, const char* mode);
```

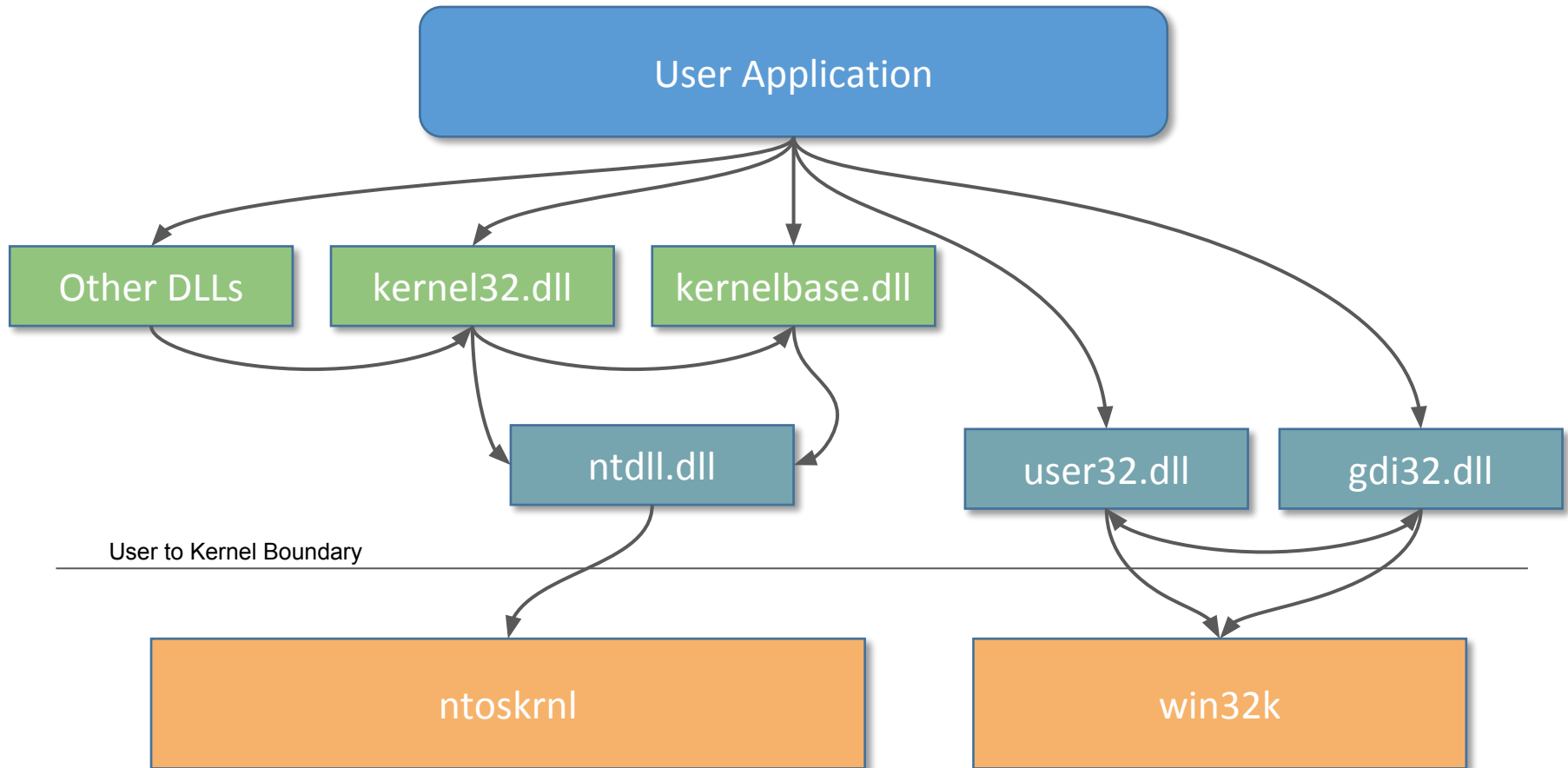
```
// Win32 Open File APIs  
HANDLE CreateFile(  
    LPCTSTR          lpFileName,  
    DWORD            dwDesiredAccess,  
    DWORD            dwShareMode,  
    LPSECURITY_ATTRIBUTES lpSecurityAttributes,  
    DWORD            dwCreationDisposition,  
    DWORD            dwFlagsAndAttributes,  
    HANDLE           hTemplateFile  
);
```

Libc Mode to
Win32 Call
Arguments

Native System Calls Can Be Even Worse



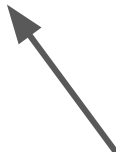
Windows API Layering



Historical Example

- Old IE11 Preview Bug Bounty sandbox escape
 - Allowed you to create arbitrary registry keys by abusing symbolic links

```
BOOL IESetProtectedModeRegKeyOnly (REGISTRYKEY* key) {  
    // ...  
    RegCreateKeyEx (  
        HKEY_CURRENT_USER,  
        "Software\\LowRights\\NewKey",  
        0,  
        NULL,  
        0,  
        KEY_ALL_ACCESS,  
        NULL,  
        &hKey,  
        &dwDisposition  
    );  
}
```



We can replace NewKey with a symbolic link and create arbitrary keys.

The Fix

```
BOOL IESetProtectedModeRegKeyOnly(REGISTRYKEY* key)
{
    if (!RegOpenKeyEx(
        HKEY_CURRENT_USER,
        "Software\\LowRightsKey\\NewKey",
        8, ← uOptions Parameter set to 8
        KEY_ALL_ACCESS,
        &hKey)) {
        // Key already created (e.g. a symbolic link)
        return FALSE;
    }

    RegCreateKeyEx(
        // ...
    );
}
```

RegOpenKeyEx on MSDN (What it used to look like)

lpSubKey [in, optional]

The name of the registry subkey to be opened.

Key names are not case sensitive.

If this parameter is **NULL** or a pointer to an empty string, the function will open a new handle to the key identified by the *hKey* parameter.

For more information, see [Registry Element Size Limits](#).

ulOptions [in]

This parameter is reserved and must be zero.

Really? It *MUST* be Zero?

RegOpenKeyEx on MSDN (What it now looks like)

ulOptions [in]

Specifies the option to apply when opening the key. Set this parameter to zero or the following:

Value	Meaning
REG_OPTION_OPEN_LINK	The key is a symbolic link. Registry symbolic links should only be used when absolutely necessary.

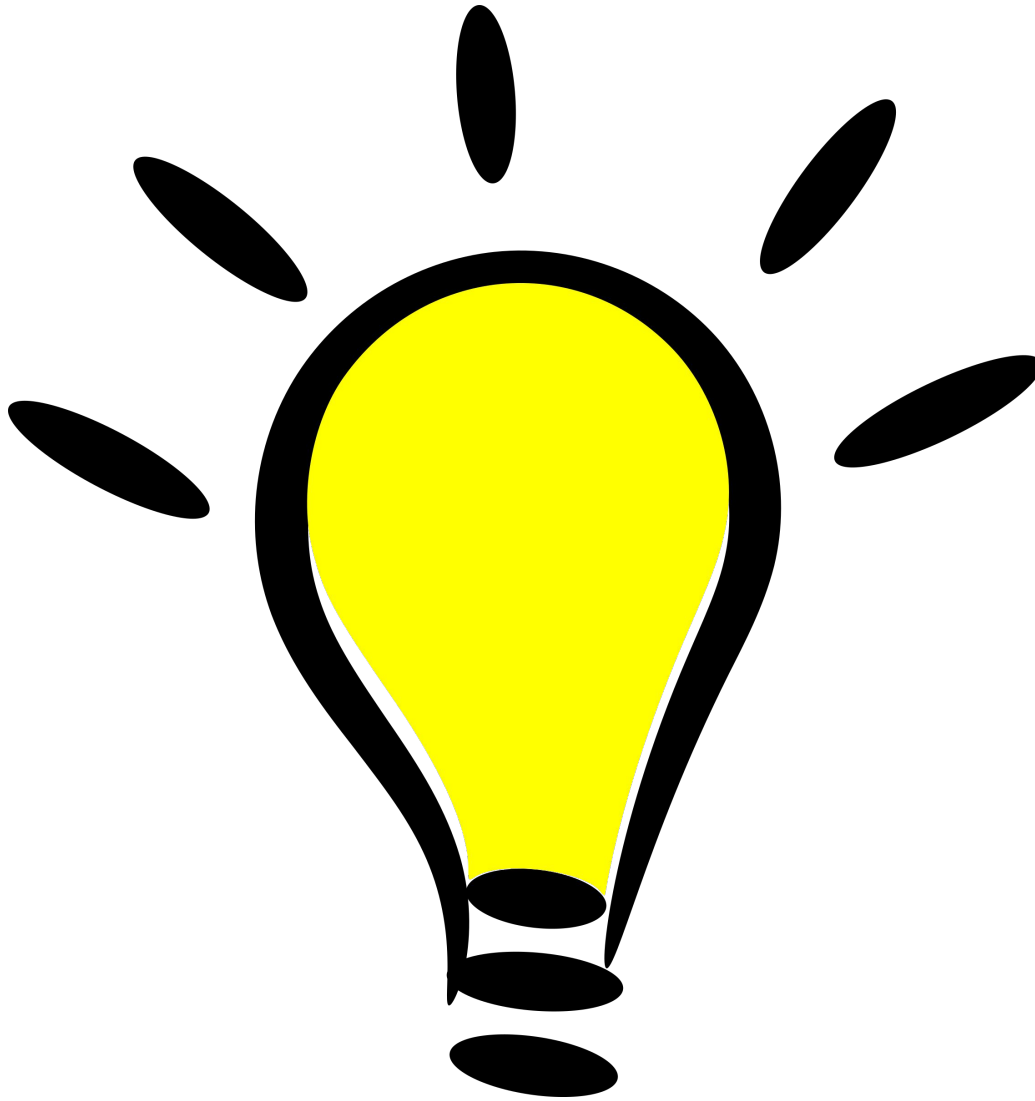
They Forgot RegOpenKeyTransacted and RegCreateKeyEx

dwOptions [in]

This parameter can be one of the following values.

Value	Meaning
REG_OPTION_BACKUP_RESTORE 0x00000004L	If this flag is set, the function ignores the <i>samDesired</i> parameter and attempts to open the key with the access required to backup or restore the key. If the calling thread has the SE_BACKUP_NAME privilege enabled, the key is opened with the ACCESS_SYSTEM_SECURITY and KEY_READ access rights. If the calling thread has the SE_RESTORE_NAME privilege enabled, beginning with Windows Vista, the key is opened with the ACCESS_SYSTEM_SECURITY, DELETE and KEY_WRITE access rights. If both privileges are enabled, the key has the combined access rights for both privileges. For more information, see Running with Special Privileges .
REG_OPTION_CREATE_LINK 0x00000002L	Note Registry symbolic links should only be used for application compatibility when absolutely necessary. This key is a symbolic link. The target path is assigned to the L"SymbolicLinkValue" value of the key. The target path must be an absolute registry path.
REG_OPTION_NON_VOLATILE 0x00000000L	This key is not volatile; this is the default. The information is stored in a file and is preserved when the system is restarted. The RegSaveKey function saves keys that are not volatile.
REG_OPTION_VOLATILE 0x00000001L	All keys created by the function are volatile. The information is stored in memory and is not preserved when the corresponding registry hive is unloaded. For HKEY_LOCAL_MACHINE , this occurs only when the system initiates a full shutdown. For registry keys loaded by the RegLoadKey function, this occurs when the corresponding RegUnLoadKey is performed. The RegSaveKey function does not save volatile keys. This flag is ignored for keys that already exist. Note On a user selected shutdown, a fast startup shutdown is the default behavior for the system.

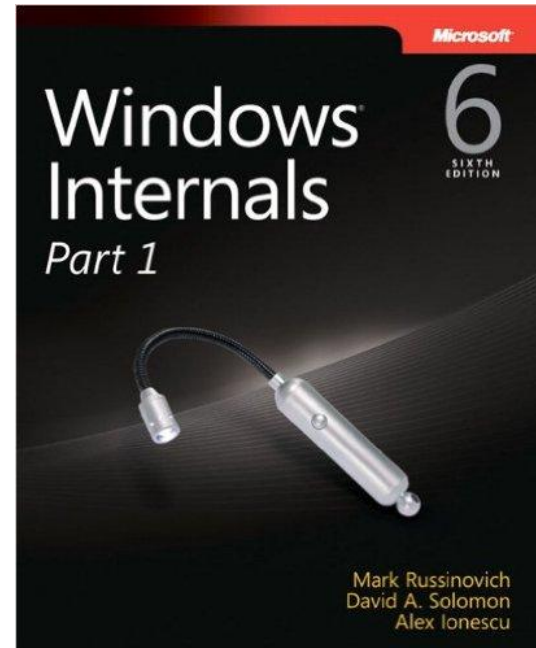
An Idea Forms



Sources of API Documentation



The horse's mouth

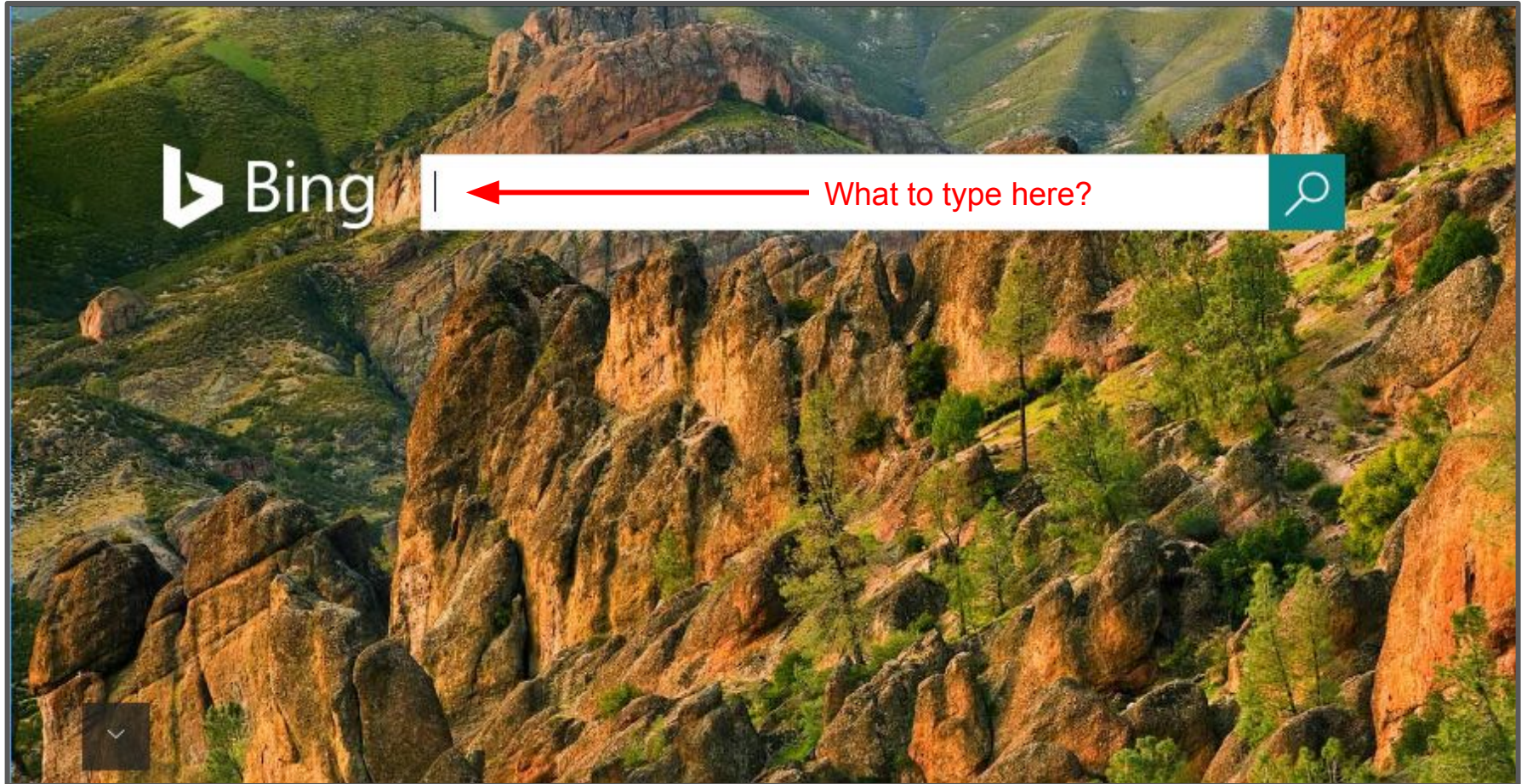


The horse's ass?

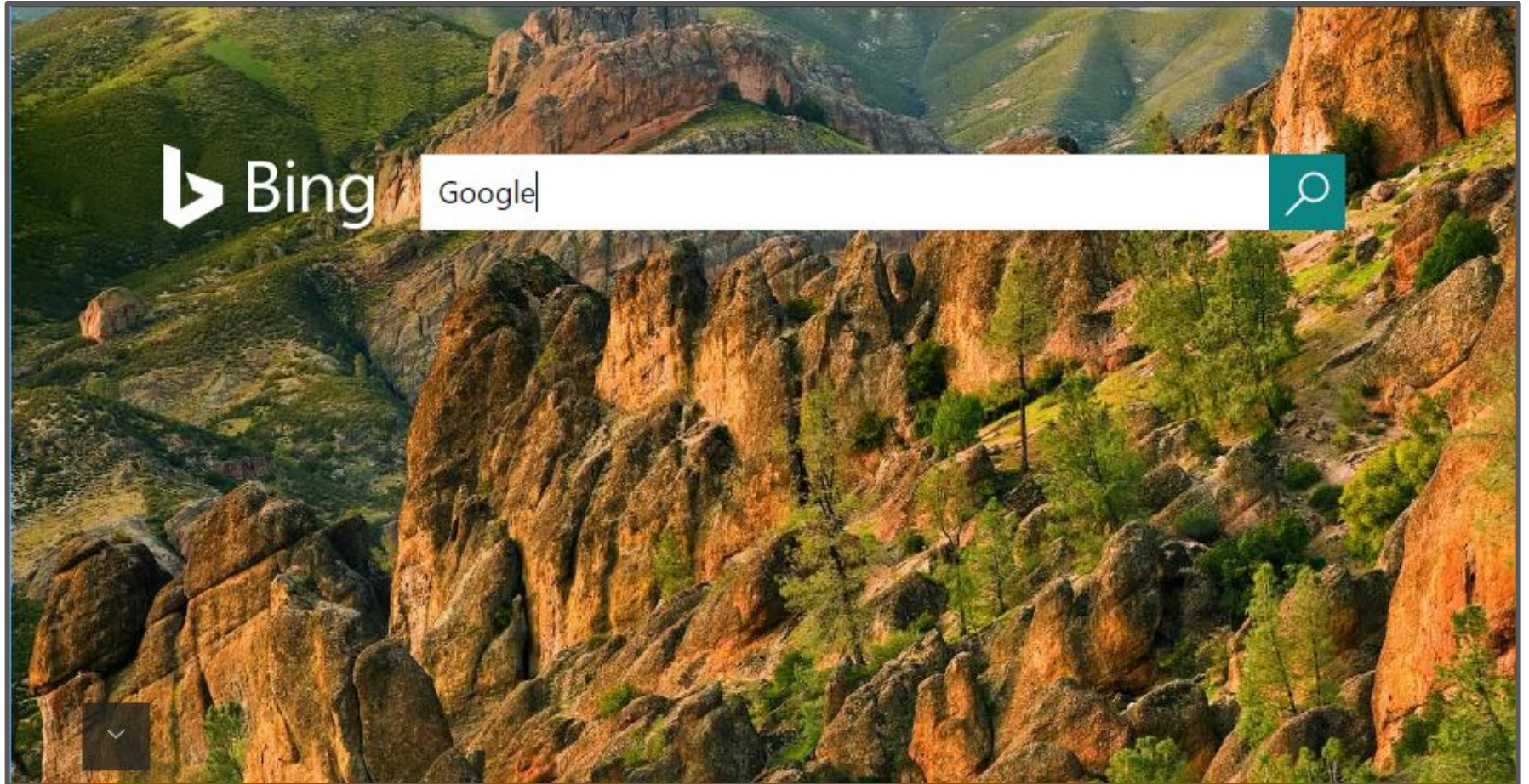
```
BOOL MyLovelyHorse(_In_z_ const char* lpNeigh,  
                  _In_ BOOL bTail,  
                  _Out_ DWORD* dwSpeed)  
{  
    // ....  
}
```

Legs perhaps? I'm confused.

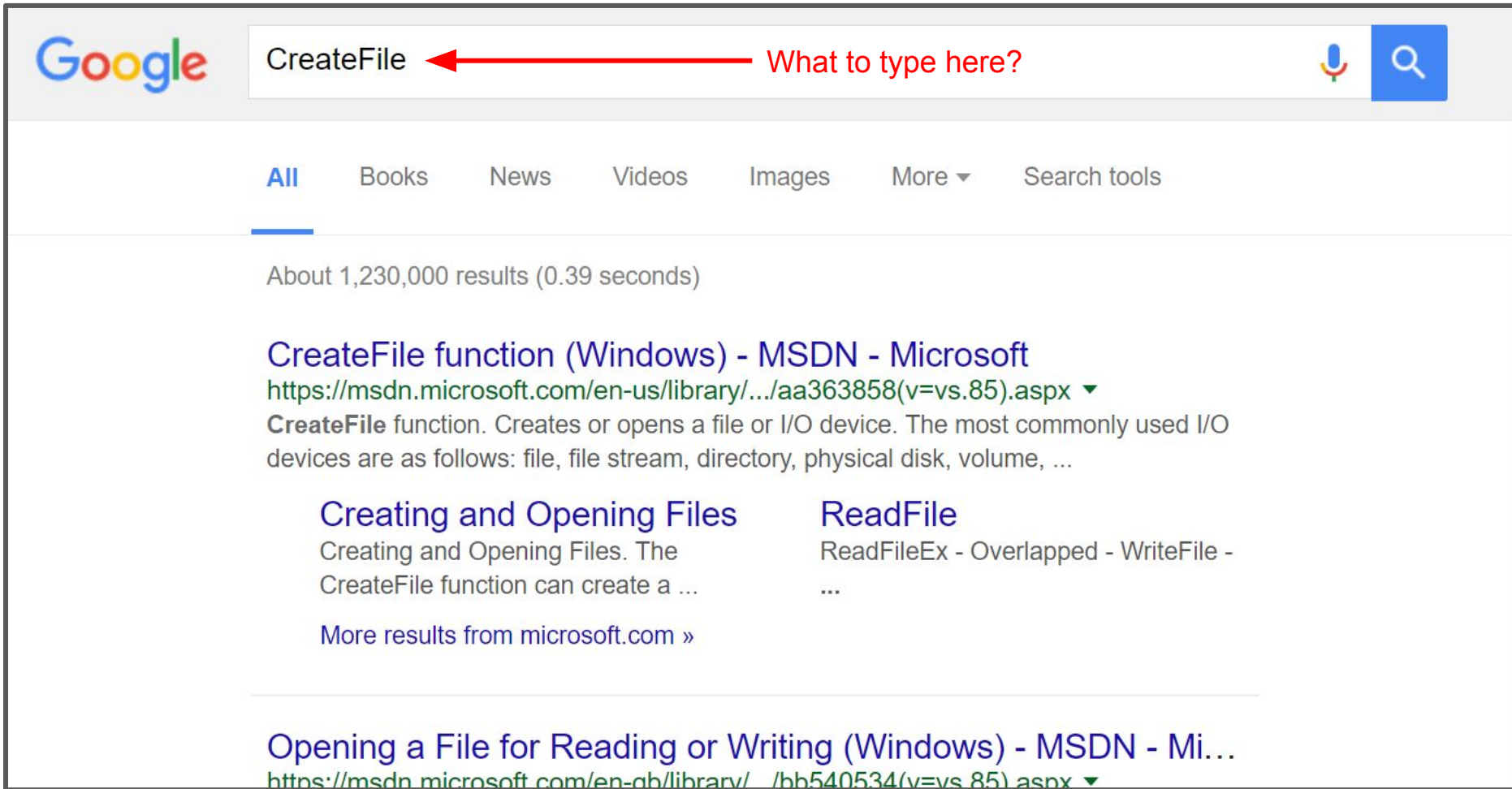
Bootstrapping the Research



Bootstrapping the Research



Bootstrapping the Research



Google

CreateFile ← What to type here?

All Books News Videos Images More Search tools

About 1,230,000 results (0.39 seconds)

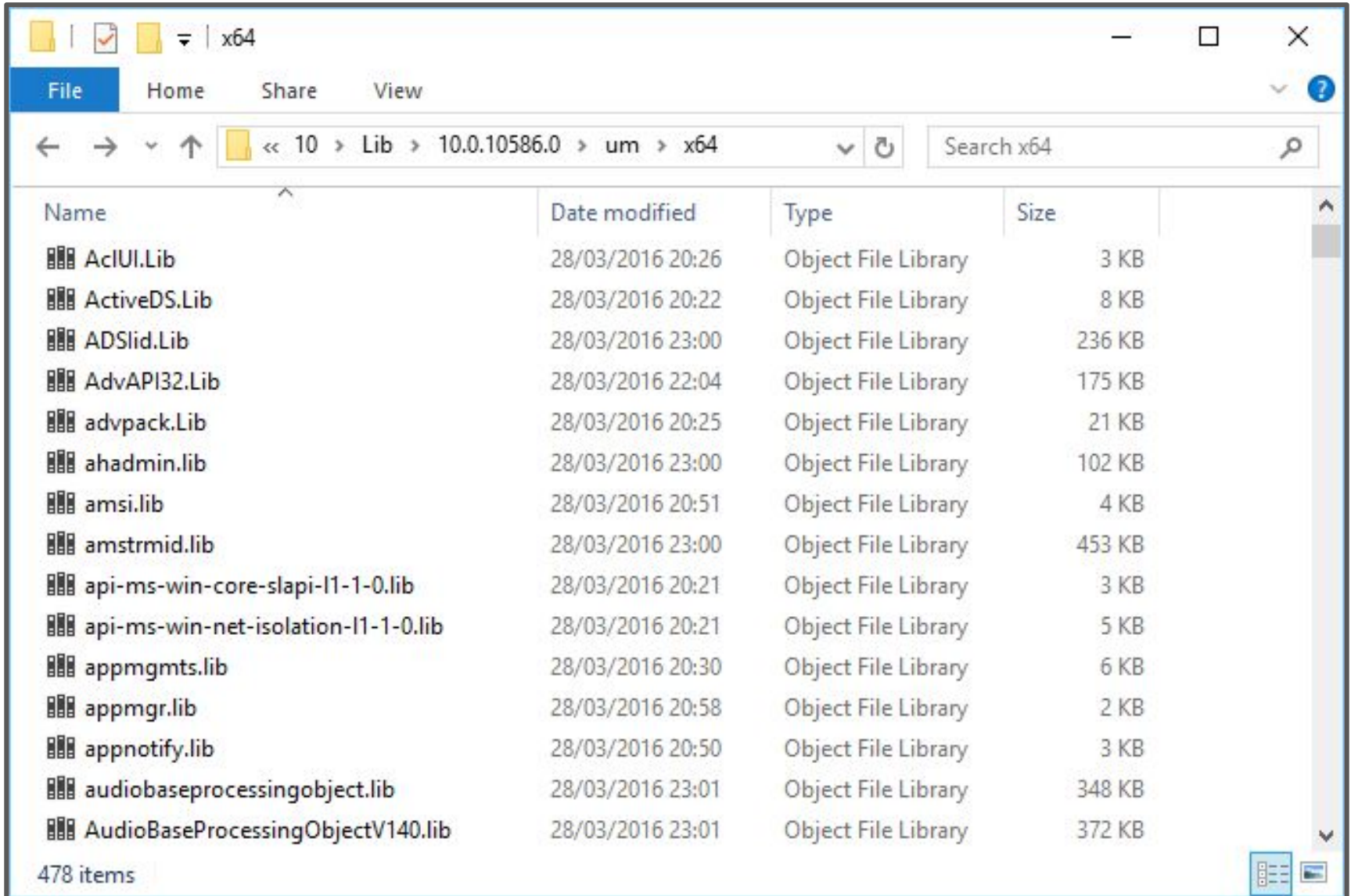
CreateFile function (Windows) - MSDN - Microsoft
[https://msdn.microsoft.com/en-us/library/.../aa363858\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/.../aa363858(v=vs.85).aspx) ▾
CreateFile function. Creates or opens a file or I/O device. The most commonly used I/O devices are as follows: file, file stream, directory, physical disk, volume, ...

Creating and Opening Files Creating and Opening Files. The CreateFile function can create a ...	ReadFile ReadFileEx - Overlapped - WriteFile - ...
---	--

[More results from microsoft.com »](#)

Opening a File for Reading or Writing (Windows) - MSDN - Mi...
[https://msdn.microsoft.com/en-gb/library/.../bb540534\(v=vs.85\).aspx](https://msdn.microsoft.com/en-gb/library/.../bb540534(v=vs.85).aspx) ▾

DLL Import Libraries



The screenshot shows a Windows File Explorer window with the following details:

- Address bar: \ll 10 > Lib > 10.0.10586.0 > um > x64
- Search bar: Search x64
- Table of files:

Name	Date modified	Type	Size
AclUI.Lib	28/03/2016 20:26	Object File Library	3 KB
ActiveDS.Lib	28/03/2016 20:22	Object File Library	8 KB
ADSIid.Lib	28/03/2016 23:00	Object File Library	236 KB
AdvAPI32.Lib	28/03/2016 22:04	Object File Library	175 KB
advpack.Lib	28/03/2016 20:25	Object File Library	21 KB
ahadmin.lib	28/03/2016 23:00	Object File Library	102 KB
amsi.lib	28/03/2016 20:51	Object File Library	4 KB
amstrmid.lib	28/03/2016 23:00	Object File Library	453 KB
api-ms-win-core-slapi-l1-1-0.lib	28/03/2016 20:21	Object File Library	3 KB
api-ms-win-net-isolation-l1-1-0.lib	28/03/2016 20:21	Object File Library	5 KB
appmgmts.lib	28/03/2016 20:30	Object File Library	6 KB
appmgr.lib	28/03/2016 20:58	Object File Library	2 KB
appnotify.lib	28/03/2016 20:50	Object File Library	3 KB
audiobaseprocessingobject.lib	28/03/2016 23:01	Object File Library	348 KB
AudioBaseProcessingObjectV140.lib	28/03/2016 23:01	Object File Library	372 KB

478 items

Documented Format (pecoff_v83.docx section 7)

Import Library Format

....

Import Header

The import header contains the following fields and offsets.

Offset	Size	Field	Description
0	2	Sig1	Must be IMAGE_FILE_MACHINE_UNKNOWN. For more information, see section 3.3.1, "Machine Types."
2	2	Sig2	Must be 0xFFFF.
4	2	Version	The structure version.
6	2	Machine	The number that identifies the type of target machine. For more information, see section 3.3.1, "Machine Types."

Parsing the Libraries

```
string signature =
    Encoding.ASCII.GetString(reader.ReadBytes(8));
if (signature != "!<arch>\n")
    throw new EndOfStreamException("Invalid signature");

List<ArchiveHeader> headers = new List<ArchiveHeader>();
long file_length = reader.BaseStream.Length;
while (reader.BaseStream.Position < file_length) {
    headers.Add(new ArchiveHeader(reader));
}
```

End Result

```
ADVAPI32.dll!AccessCheckByType
ADVAPI32.dll!AccessCheckByTypeAndAuditAlarmA
ADVAPI32.dll!AccessCheckByTypeAndAuditAlarmW
ADVAPI32.dll!AccessCheckByTypeResultList
ADVAPI32.dll!AddAccessAllowedAce
ADVAPI32.dll!AddAccessAllowedAceEx
ADVAPI32.dll!AddAccessAllowedObjectAce
ADVAPI32.dll!AddAccessDeniedAce
ADVAPI32.dll!AddAccessDeniedAceEx
ADVAPI32.dll!AddAccessDeniedObjectAce
ADVAPI32.dll!AddAce
...
```

Imports extracted = 26719
Filtered to valid C function names = 20374

Don't Do This - Pulling from MSDN Online

The screenshot shows a Windows File Explorer window with the address bar set to `dev > MSDNParser > MSDNParser > output`. The file list contains 16 HTML files, each with a size ranging from 102 KB to 167 KB. The status bar at the bottom indicates `8,993 items` and `8,993 items selected`.

An open Properties dialog box is overlaid on the right side of the window. The title bar reads `ADVAPI32.dll!AccessCheckByTypeResultListAndAuditAl...`. The `General` tab is active, showing a folder icon and the text `8,993 Files, 0 Folders`. The `Details` tab is also visible.

The Properties dialog box displays the following information:

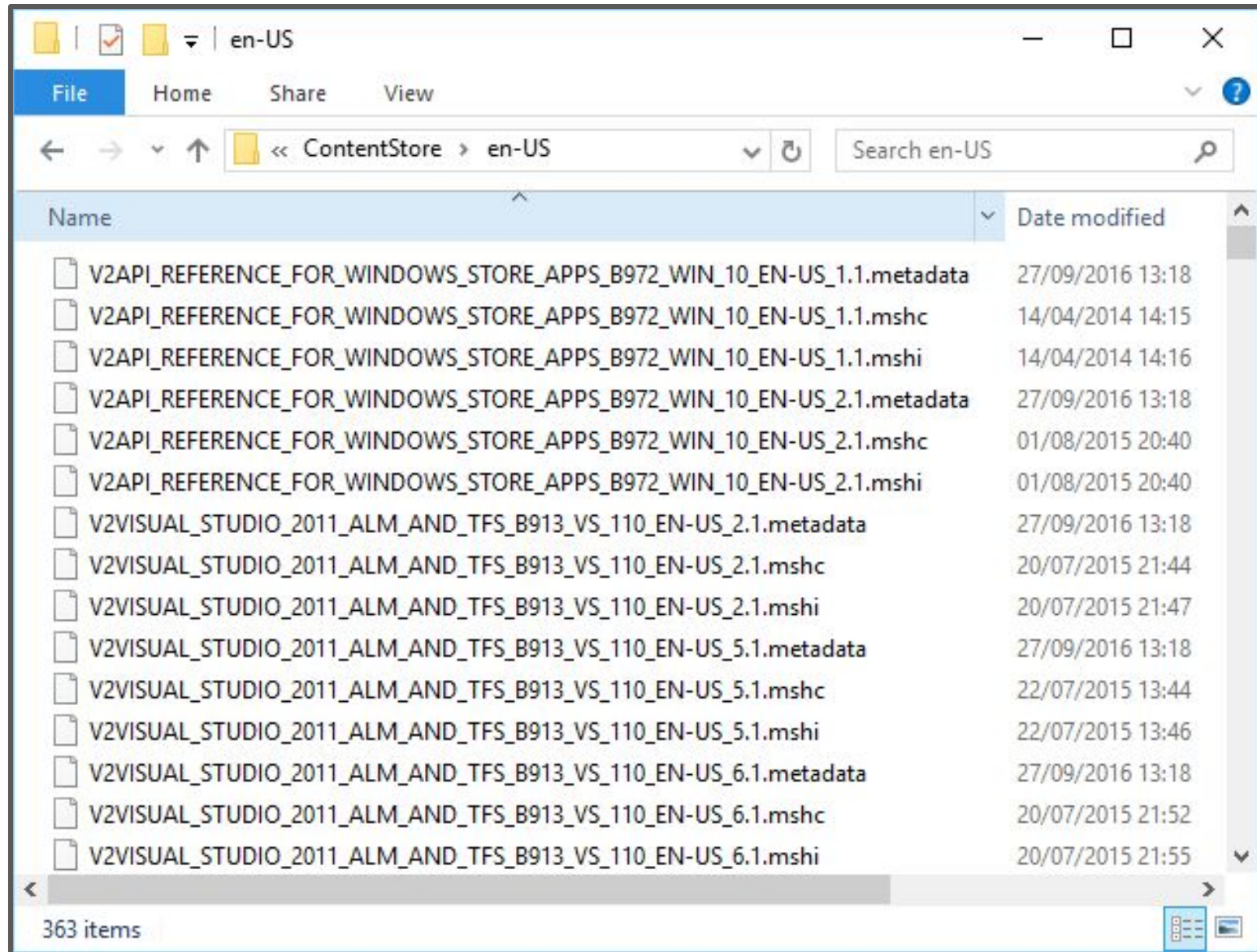
- Type:** Multiple Types
- Location:** All in D:\dev\MSDNParser\MSDNParser\output
- Size:** 1.48 GB (1,590,051,046 bytes)
- Size on disk:** 1.49 GB (1,608,810,496 bytes)

The `Attributes` section shows `Read-only` and `Hidden` checkboxes, both of which are unchecked. An `Advanced...` button is located to the right of the checkboxes. At the bottom of the dialog box, there are `OK`, `Cancel`, and `Apply` buttons.

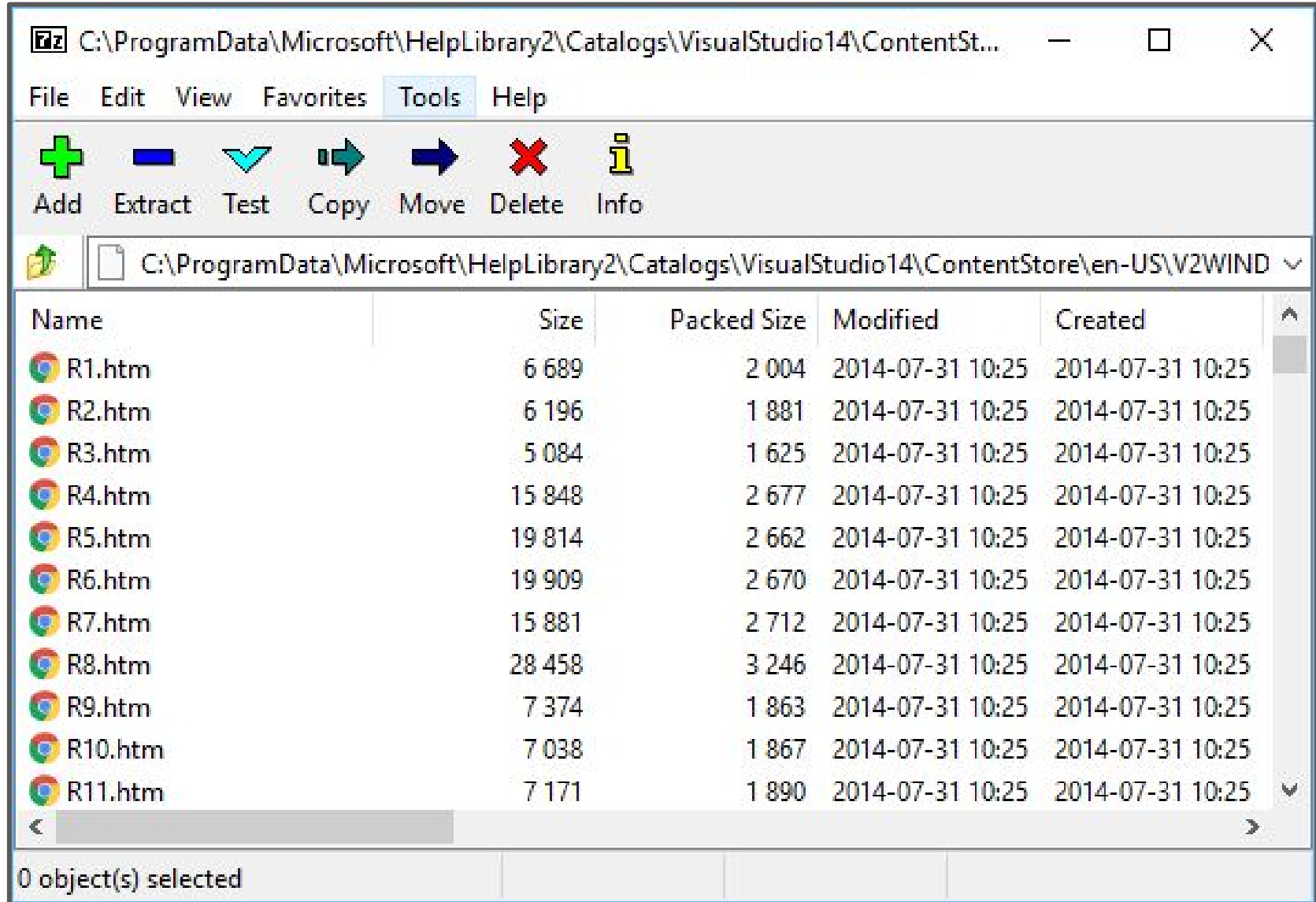
Pretty Much Unusable XHTML

```
<!DOCTYPE html>
<html dir="ltr" xmlns="http://www.w3.org/1999/xhtml" lang="en">
<head><link rel="canonical" href="https://msdn.microsoft.com/en-us/library/windows/desktop/aa374831" />
  <title>AccessCheckByTypeAndAuditAlarm function (Windows)</title>
<meta name="DCS.dcsuri" content="/en-us/library/windows/desktop/aa374831(d=default,l=en-us,v=vs.85)"/>
<meta name="NormalizedUrl" content="https://msdn.microsoft.com/en-us/library/windows/desktop/aa374831" />
<meta name="ms.normalizedurl" content="https://msdn.microsoft.com/en-us/library/windows/desktop/aa374831" />
<meta name="AmbientContext" content="{&quot;layout.limit_max_width&quot;:true,&quot;AmbientContextId&quot;:1}"/>
<meta name="VotingContextUrl" content="https://msdn.microsoft.com/en-us/library/windows/desktop/aa374831" />
<meta name="MN" content="FB361393-3:01:18 AM" />
<meta name="Search.ShortId" content="aa374831" />
<meta name="ms.shortidmsdn" content="aa374831" />
<meta name="ms.locale" content="en-us" />
<meta name="ms.prodver" content="VS.85" />
<meta name="ms.contentlang" content="EN" />
<meta name="ms.lang" content="EN" />
<meta name="ms.loc" content="US" />
<meta name="ms.sitever" content="2016.09.18.1" />
<meta name="ms.assetid" content="ea14fd55-e0e4-4bf2-b20e-5874783c16c3" />
<meta name="ms.auth" content="0" />
<meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <script type="text/javascript" data-do-not-move="true">
    //<![CDATA[
      var AmbientContext = null;
      (function(){
        var root = (function(){return this;}).call(null);
        var GetCookie = function(name){
          var cookies = root.document.cookie ? root.document.cookie.split('; ') : [];
          for (var i = 0; i < cookies.length; i++) {
            var pos = cookies[i].indexOf('=');
            if (name === root.decodeURIComponent(cookies[i].slice(0, pos)))
              {
                var cookie = cookies[i].slice(pos + 1).replace(/\+/g, ' ');
                cookie = root.decodeURIComponent(cookie);
              }
          }
        }
      })();
    ]></script>
  </head>
</html>
```

Forgot about Offline MSDN



MSHC Just Zip File Archives with HTML



The screenshot shows a Windows Explorer window titled "C:\ProgramData\Microsoft\HelpLibrary2\Catalogs\VisualStudio14\ContentSt...". The address bar displays the path "C:\ProgramData\Microsoft\HelpLibrary2\Catalogs\VisualStudio14\ContentStore\en-US\V2WIND". The main area shows a list of files in a table format:

Name	Size	Packed Size	Modified	Created
R1.htm	6 689	2 004	2014-07-31 10:25	2014-07-31 10:25
R2.htm	6 196	1 881	2014-07-31 10:25	2014-07-31 10:25
R3.htm	5 084	1 625	2014-07-31 10:25	2014-07-31 10:25
R4.htm	15 848	2 677	2014-07-31 10:25	2014-07-31 10:25
R5.htm	19 814	2 662	2014-07-31 10:25	2014-07-31 10:25
R6.htm	19 909	2 670	2014-07-31 10:25	2014-07-31 10:25
R7.htm	15 881	2 712	2014-07-31 10:25	2014-07-31 10:25
R8.htm	28 458	3 246	2014-07-31 10:25	2014-07-31 10:25
R9.htm	7 374	1 863	2014-07-31 10:25	2014-07-31 10:25
R10.htm	7 038	1 867	2014-07-31 10:25	2014-07-31 10:25
R11.htm	7 171	1 890	2014-07-31 10:25	2014-07-31 10:25

The status bar at the bottom indicates "0 object(s) selected".

Ignore Index, Just Parse Out of HTML

```
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>CreateFile function</title>
    <meta name="Microsoft.Help.F1"
          content="CreateFile" />
    <meta name="Microsoft.Help.F1"
          content="CreateFileA" />
    <meta name="Microsoft.Help.F1"
          content="CreateFileW" />
    ...
  </head>
```

API names and keywords used when hitting F1

Filtered to valid C function names = 20374
Found Unique Documented Functions = 8616

The Laziest Possible Approach

- Search the XHTML text nodes for keywords
- Based on the experience with RegOpenKeyEx how about looking for the word “reserved”.
- Suspiciously high number of matching help files :(

The Laziest Possible Approach

- Search the XHTML text nodes for keywords
- Based on the experience with RegOpenKeyEx how about looking for the word “reserved”.
- Suspiciously high number of matching help files :(



problem?

```
<p>© 2015 Microsoft Corporation.  
All rights reserved.</p>
```

Parameter Documentation

```
<h2>Parameters</h2>
<dl>
  <dt><em>lpFileName</em> [in]</dt>
  <dd><p>The name of the file or device to
be created or opened. You may use either
forward slashes (/) or backslashes (\) in
this name.</p>
  ...
</dd>
  <dt><em>dwDesiredAccess</em> [in]</dt>
  <dd>...</dd>
</dl>
```


Suitable Parameters for Inspection

Flags [in]

This parameter is **reserved** for future use.

dwReserved [in]

This value is reserved and must be set to 0.

pfnArray [in, optional]

Unused. Set to **NULL**.

We'll also limit length of description, if > 100 characters it's probably rambling.

Found Unique Documented Functions = 8616

Functions with Matching Parameters = 562

Source-code Annotation Language (SAL)

- A set of annotations created by Microsoft to document functional behaviour (current version 2.0)
- Used with Visual Studio Code Analysis to find bugs in your programs
- Can also be used to detect security issues such as buffer overflows
- Can apply to:
 - Parameters
 - Return Types
 - Function behaviours (such as caller/callee relationships)
 - Locking

Common Parameter Annotations

<i>Annotation</i>	<i>Description</i>
<code>_In_</code>	Parameter or buffer is an input to the function
<code>_Out_</code>	Buffer is an output from the function
<code>_Inout_</code>	Buffer is both an input and output
<code>_In_z_</code>	Buffer is a zero terminated string
<code>_Reserved_</code>	The parameter is reserved

Note: I couldn't seem to find an online source for the SAL 2.0 `_Reserved_` annotation. Ironic?

Behaviour of `_Reserved_` Annotation

```
void MyFunction(_In_z_ LPCSTR lpName,  
               _Reserved_ DWORD dwFlags) {  
    cout << lpName << endl;  
}  
  
int main() {  
    MyFunction("NoReserved", 0);  
    MyFunction("Reserved", 1);  
    return 0;  
}
```

The screenshot shows the 'Error List' window in Visual Studio. The status bar at the top indicates 'Entire Solution' with '0 Errors', '1 Warning', and '0 Messages'. The search bar is empty. The table below lists the error details:

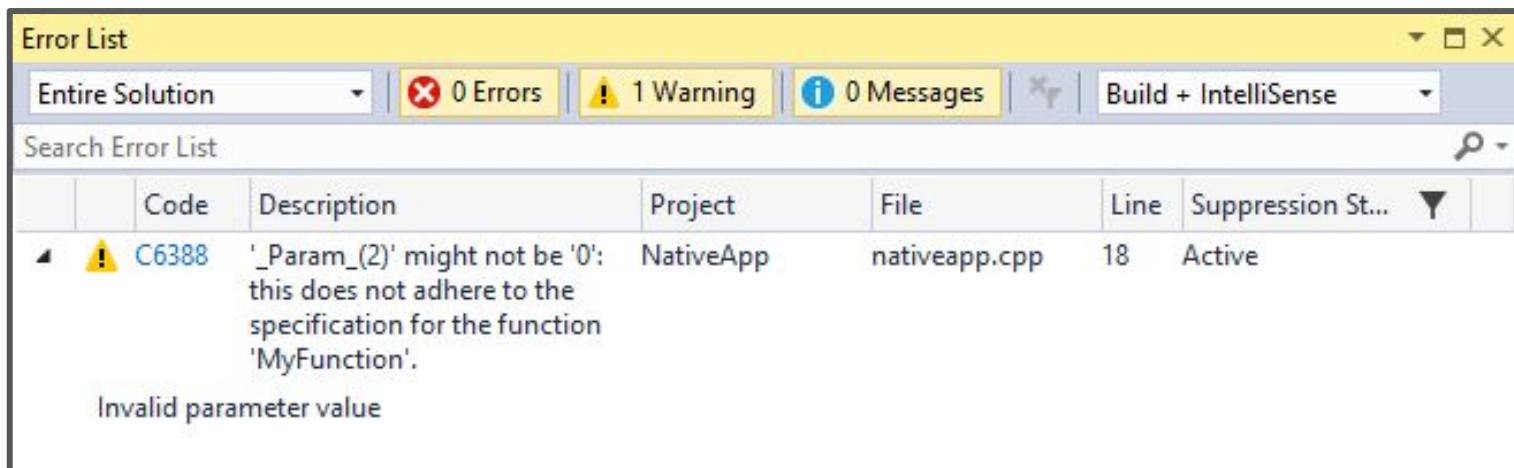
	Code	Description	Project	File	Line	Suppression St...
▲	! C6388	'_Param_(2)' might not be '0': this does not adhere to the specification for the function 'MyFunction'.	NativeApp	nativeapp.cpp	18	Active

Invalid parameter value

Behaviour of `_Reserved_` Annotation

```
void MyFunction(_In_z_ LPCSTR lpName,
               _Reserved_ DWORD dwFlags) {
    cout << lpName << " " << (dwFlags == 1) << endl;
}

int main() {
    MyFunction("NoReserved", 0);
    MyFunction("Reserved", 1);
    return 0;
}
```



The screenshot shows the 'Error List' window in Visual Studio. The window title is 'Error List'. The status bar at the top indicates 'Entire Solution' with '0 Errors', '1 Warning', and '0 Messages'. The search bar is empty. The table below shows a single warning entry:

	Code	Description	Project	File	Line	Suppression St...
⚠	C6388	'_Param_(2)' might not be '0': this does not adhere to the specification for the function 'MyFunction'.	NativeApp	nativeapp.cpp	18	Active

Below the table, the text 'Invalid parameter value' is displayed.

Combining the Two Data Sets

- Add any functions we've found with `_Reserved_` parameter which didn't get picked up earlier
 - Found another 40 functions, where the documentation wasn't found or used alternative keywords (such as “not used” `_(ツ)_/`)
- For interest would be worth correlating what parameters are not marked with `_Reserved_` but are documented as reserved.
 - Approximately 100 functions were documented as reserved but not marked in the headers as such.

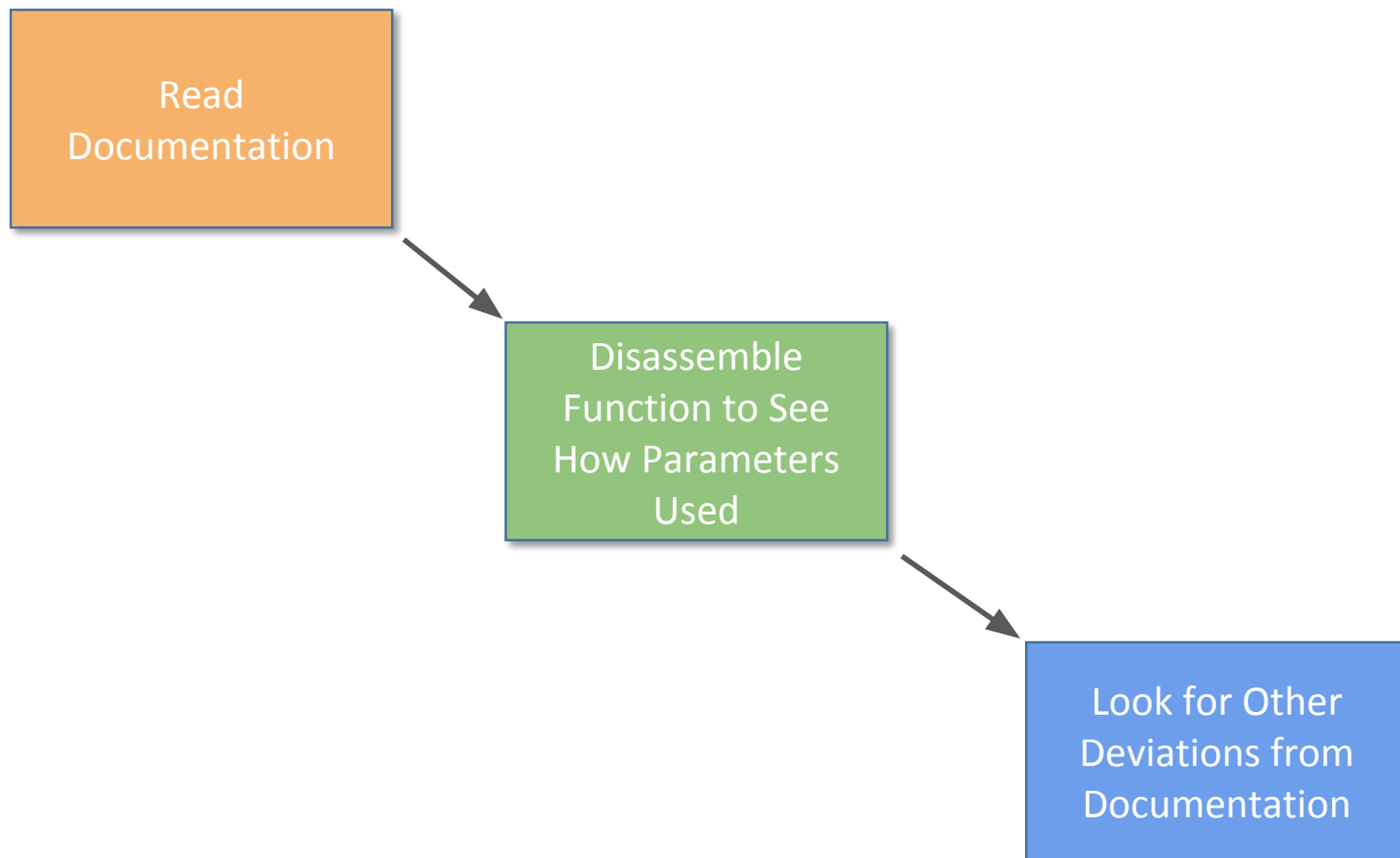
Running Total: 602

Code Analysis

- We've got our collection of functions. Around 600 is probably too many to do manual analysis.
- While could do automated analysis for this, would prefer the personal touch.
- Limit to only those functions in some core libraries:
 - KERNEL32/KERNELBASE
 - USER32
 - GDI32
 - NTDLL
 - ADVAPI32
 - SHELL32

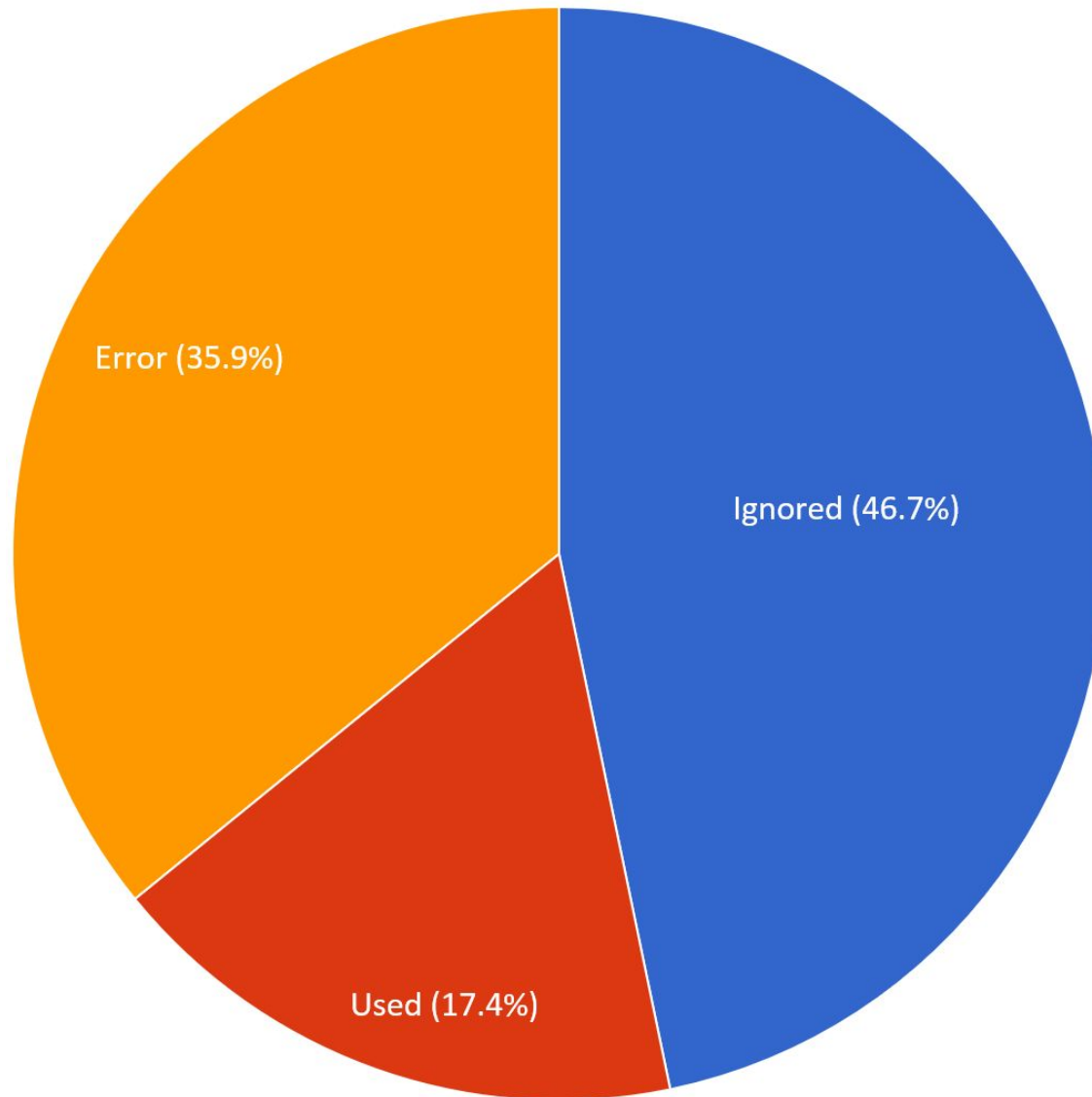
Final Total: 94 Unique Functions

Review Strategy




What You've All Been Waiting For
Results of Analysis

Graph Time!

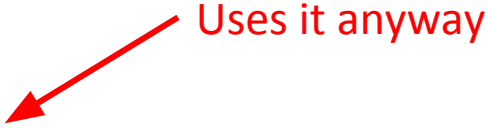


CreateHardLink Weirdness

```
BOOL CreateHardLinkW(  
    _In_      LPCWSTR lpFileName,  
    _In_      LPCWSTR lpExistingFileName,  Reserved parameter  
    _Reserved_ LPSECURITY_ATTRIBUTES lpSecAttr) {  
    OBJECT_ATTRIBUTES ObjA;  
    HANDLE FileHandle;  
  
    if (!lpFileName || !lpExistingFileName) {  
        RtlSetLastWin32Error(ERROR_INVALID_PARAMETER);  
        return FALSE  
    }  
  
    if (lpSecAttr)  
        ObjA.SecurityDescriptor = lpSecAttr->lpSecurityDescriptor;  
  
    NtOpenFile(&FileHandle,  
        SYNCHRONIZE | FILE_WRITE_ATTRIBUTES, &ObjA, ...);  
    // ...  
}
```

CreateHardLink Weirdness

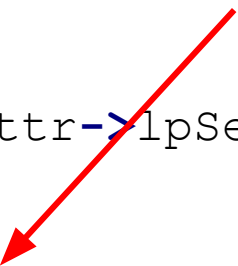
```
BOOL CreateHardLinkW(  
    _In_      LPCWSTR lpFileName,  
    _In_      LPCWSTR lpExistingFileName,  
    _Reserved_ LPSECURITY_ATTRIBUTES lpSecAttr) {  
    OBJECT_ATTRIBUTES ObjA;  
    HANDLE FileHandle;  
  
    if (!lpFileName || !lpExistingFileName) {  
        RtlSetLastWin32Error(ERROR_INVALID_PARAMETER);  
        return FALSE  
    }  
  
    if (lpSecAttr)  
        ObjA.SecurityDescriptor = lpSecAttr->lpSecurityDescriptor;  
  
    NtOpenFile(&FileHandle,  
        SYNCHRONIZE | FILE_WRITE_ATTRIBUTES, &ObjA, ...);  
    // ...  
}
```



Uses it anyway

CreateHardLink Weirdness

```
BOOL CreateHardLinkW(  
    _In_      LPCWSTR lpFileName,  
    _In_      LPCWSTR lpExistingFileName,  
    _Reserved_ LPSECURITY_ATTRIBUTES lpSecAttr) {  
    OBJECT_ATTRIBUTES ObjA;  
    HANDLE FileHandle;  
  
    if (!lpFileName || !lpExistingFileName) {  
        RtlSetLastWin32Error(ERROR_INVALID_PARAMETER);  
        return FALSE  
    }  
  
    if (lpSecAttr)  
        ObjA.SecurityDescriptor = lpSecAttr->lpSecurityDescriptor;  
  
    NtOpenFile(&FileHandle,  
              SYNCHRONIZE | FILE_WRITE_ATTRIBUTES, &ObjA, ...);  
    // ...  
}
```



Ignores write access

Some Other Weird Ones

- KERNEL32.dll!CreateConsoleScreenBuffer
- KERNEL32.dll!UpdateProcThreadAttribute
- ntdll.dll!ZwCreateTransaction
- ntdll.dll!ZwNotifyChangeKey
- ntdll.dll!ZwReadFile
- ntdll.dll!ZwSetValueKey
- ntdll.dll!ZwWriteFile

CreateBoundaryDescriptor

```
HANDLE CreateBoundaryDescriptor (  
    _In_ LPCTSTR Name,  
    _In_ ULONG     Flags  
);
```

Name [in]

The name of the boundary descriptor.

Flags [in]

This parameter is **reserved** for future use.

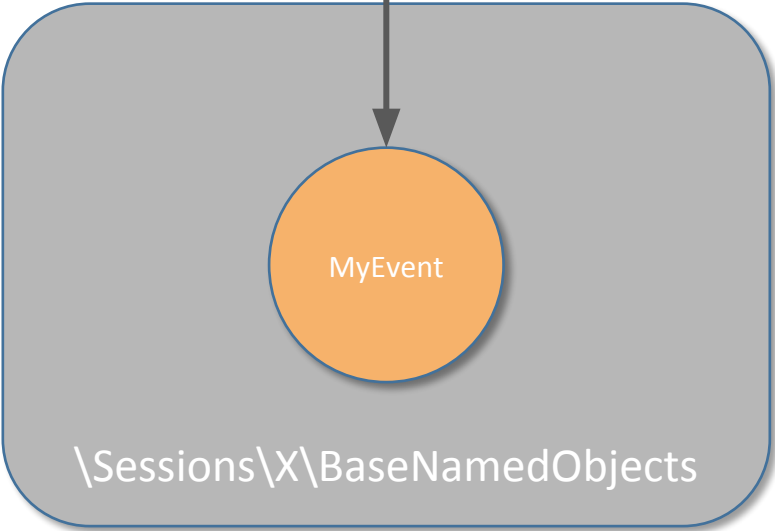
Under the Hood

```
HANDLE CreateBoundaryDescriptorW(LPCWSTR Name,
                                ULONG Flags) {
    UNICODE_STRING name_str;
    RtlInitUnicodeString(&name_str, Name);
    return RtlCreateBoundaryDescriptor(&name_str,
                                      Flags);
}
```

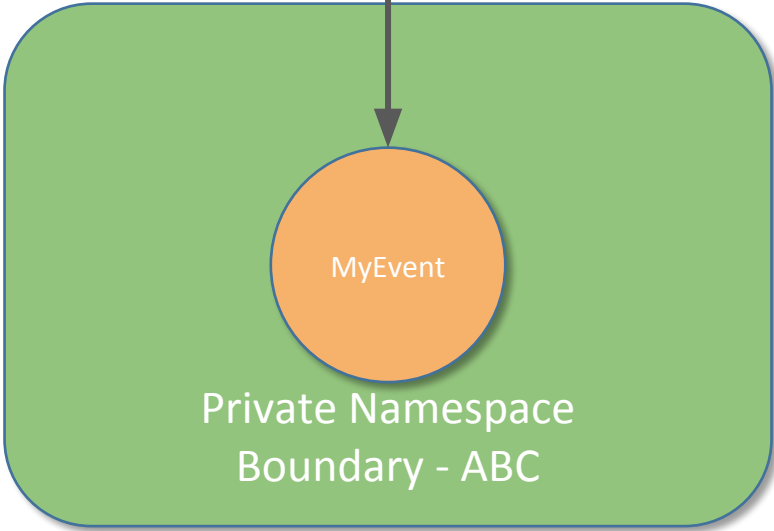
```
void* RtlCreateBoundaryDescriptor(PUNICODE_STRING Name,
                                 ULONG Flags) {
    if (Flags & ~1) ← Flag '1' is valid
        return NULL;
    BOUNDARY_DESC* desc = RtlAllocateHeap();
    if (Flags & 1)
        desc->Flags = 1; ← Set '1' to
                          descriptor flags
    // ...
    return desc;
}
```


Private Namespaces

```
CreateEvent (NULL,  
            FALSE,  
            FALSE,  
            L"MyEvent")
```



```
CreateEvent (NULL,  
            FALSE,  
            FALSE,  
            L"ABC\\MyEvent")
```



So What is the Boundary Descriptor Used For?

```
HANDLE WINAPI CreatePrivateNamespace(  
    _In_opt_ LPSECURITY_ATTRIBUTES lpNamespaceAttributes,  
    _In_     LPVOID                 lpBoundaryDescriptor,  
    _In_     LPCTSTR                lpAliasPrefix  
);
```

lpBoundaryDescriptor [in]

A descriptor that defines how the namespace is to be isolated. The caller must be within this boundary. The

CreateBoundaryDescriptor function creates a boundary descriptor.

What does it mean to be within?

Security Feature?

Object Namespaces

An *object namespace* protects named objects from unauthorized access. Creating a private namespace enables applications and services to build a more secure environment.

A process can create a private namespace using the **CreatePrivateNamespace** function. This function requires that you specify a *boundary* that defines how the objects in the namespace are to be isolated. **The caller must be within the specified boundary** for the create operation to succeed. To specify a boundary, use the **CreateBoundaryDescriptor** and **AddSIDToBoundaryDescriptor** functions.

You keep using that word!

Maybe This Will Clear Things Up?

```
BOOL WINAPI AddSIDToBoundaryDescriptor(  
    _Inout_ HANDLE *BoundaryDescriptor,  
    _In_     PSID   RequiredSid  
);
```

BoundaryDescriptor [in, out]

A handle to the boundary descriptor. The

CreateBoundaryDescriptor function returns this handle.

RequiredSid [in]

A pointer to a **SID** structure.

Remarks


The **AddSIDToBoundaryDescriptor** function must be called once for each SID to be added to the boundary descriptor.

Nope no help whatsoever!

Best Documentation is an Article from 2007!


The screenshot shows the CodeGuru website interface. At the top left is the CodeGuru logo with the tagline "The number one developer site!". To the right is a search bar. Below the logo is a navigation menu with links for "Visual C++ / C++", ".NET / C#", "Visual Basic", "Others", "Forums", "Videos", and "Submit". A breadcrumb trail shows the path: "codeguru > .NET / C# > .NET > Framework > Microsoft Namespace". Below the breadcrumb is a link "Read More in Microsoft Namespace »". A grey box contains the text: "Understand how to handle big data and improve organizational agility to support demands of a dynamic enterprise. Read our eBook today." Below this are social sharing icons for "Post a comment", "Email Article", "Print Article", and "Share Articles". The article title is "Vista Improves Security Through Private Object Namespaces". It is posted by "Nick Wienholt" on "February 12th, 2007". There are five stars for rating and a "Vote!" button. Below the title are social sharing buttons for "Tweet", "Like" (with a count of 0), "Share", and "G+1". The article text begins with: "You can protect Windows kernel objects such as events and mutexes by using security descriptors, but prior to Windows Vista you couldn't secure the actual names of the kernel objects. Building on the concept of namespaces that are used to separate Terminal Service sessions, Windows Vista allows you to define, secure, and use custom namespaces to prevent malicious applications from denying access to kernel object functionality."

Digging into the Kernel

```
NSTATUS ObpCaptureBoundaryDescriptor (BOUNDARY_DESC* input,
                                      BOUNDARY_DESC** out) {
    // ...
    if (input->Flags & 1) {  Check Flag
        SECURITY_SUBJECT_CONTEXT ctx;
        DWORD is_app_container;
        SeCaptureSubjectContext (&ctx);
        PACCESS_TOKEN token = SeQuerySubjectContextToken (&ctx);
        SeQueryInformationToken (token, TokenIsAppContainer,
                                &is_app_container);

        if (is_app_container) {
            SeQueryInformationToken (token, TokenAppContainerSid,
                                    &app_container);

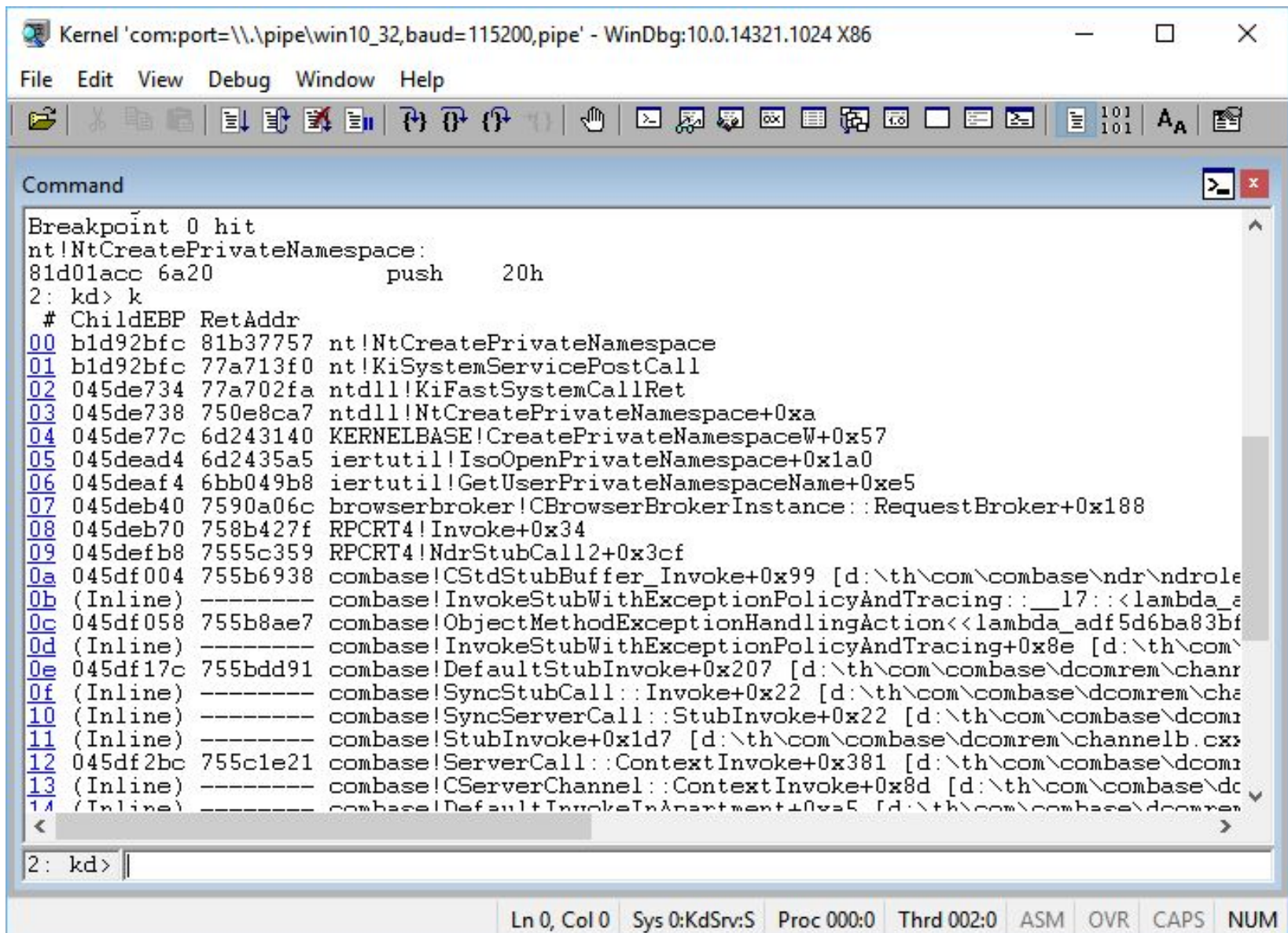
            // Add AC SID to boundary descriptor
        }
    }
    // ...
}
```

 Add AppContainer
SID to Boundary

Boundary Descriptor Used as an Access Check

```
NTSTATUS ObpVerifyCreatorAccessCheck(  
    BOUNDARY_DESC* desc) {  
    SECURITY_SUBJECT_CONTEXT ctx;  
    for(PSID sid : desc->Sids) {  
        PSECURITY_DESCRIPTOR sd = MakeSDForSid(sid);  
        if (!SeAccessCheck(sd, &ctx, ...)) {  
            return STATUS_ACCESS_DENIED;  
        }  
    }  
  
    if (desc->ACSid) {  
        // Check current user is also in the AC.  
    }  
}
```

Do Anyone Use Them?



Kernel 'com:port=\\.\pipe\win10_32,baud=115200,pipe' - WinDbg:10.0.14321.1024 X86

File Edit View Debug Window Help

Command

```
Breakpoint 0 hit
nt!NtCreatePrivateNamespace:
81d01acc 6a20          push     20h
2: kd> k
# ChildEBP RetAddr
00 b1d92bfc 81b37757 nt!NtCreatePrivateNamespace
01 b1d92bfc 77a713f0 nt!KiSystemServicePostCall
02 045de734 77a702fa ntdll!KiFastSystemCallRet
03 045de738 750e8ca7 ntdll!NtCreatePrivateNamespace+0xa
04 045de77c 6d243140 KERNELBASE!CreatePrivateNamespaceW+0x57
05 045dead4 6d2435a5 iertutil!IsoOpenPrivateNamespace+0x1a0
06 045deaf4 6bb049b8 iertutil!GetUserPrivateNamespaceName+0xe5
07 045deb40 7590a06c browserbroker!CBrowserBrokerInstance::RequestBroker+0x188
08 045deb70 758b427f RPCRT4!Invoke+0x34
09 045defb8 7555c359 RPCRT4!NdrStubCall12+0x3cf
0a 045df004 755b6938 combase!CStdStubBuffer_Invoke+0x99 [d:\th\com\combase\ndr\ndrole
0b (Inline) ----- combase!InvokeStubWithExceptionPolicyAndTracing::__17::<lambda_e
0c 045df058 755b8ae7 combase!ObjectMethodExceptionHandlingAction<<lambda_adf5d6ba83bf
0d (Inline) ----- combase!InvokeStubWithExceptionPolicyAndTracing+0x8e [d:\th\com\
0e 045df17c 755bdd91 combase!DefaultStubInvoke+0x207 [d:\th\com\combase\dcomrem\chanr
0f (Inline) ----- combase!SyncServerCall::Invoke+0x22 [d:\th\com\combase\dcomrem\cha
10 (Inline) ----- combase!SyncServerCall::StubInvoke+0x22 [d:\th\com\combase\dcomr
11 (Inline) ----- combase!StubInvoke+0x1d7 [d:\th\com\combase\dcomrem\channelb.cxx
12 045df2bc 755c1e21 combase!ServerCall::ContextInvoke+0x381 [d:\th\com\combase\dcomr
13 (Inline) ----- combase!CServerChannel::ContextInvoke+0x8d [d:\th\com\combase\dc
14 (Inline) ----- combase!DefaultInvokeInApartment+0xa5 [d:\th\com\combase\dcomre
```

2: kd> |

Ln 0, Col 0 Sys 0:KdSrv:S Proc 000:0 Thrd 002:0 ASM OVR CAPS NUM

Edge/Internet Explorer Boundary Descriptor

Entry
Type = Name

Entry Size

Version

Entry
Count

Total
Bytes

Flags

```
kd> dc poi(@esp+10) Lpoi(poi(@esp+10)+8) /4
0359cfa0 00000001 00000001 000000a0 00000000 .....
0359cfb0 00000001 0000008c 00450049 00730055 .....I.E.U.s.
0359cfc0 00720065 0053005f 0031002d 0035002d e.r._.S.-.1.-.5.
0359cfd0 0032002d 002d0031 00310031 00340030 -.2.1.-.1.1.0.4.
0359cfe0 00330033 00370037 002d0033 00380031 3.3.7.7.3.-.1.8.
0359cff0 00330038 00320036 00340036 00340035 8.3.6.2.6.4.5.4.
0359d000 0033002d 00350036 00300033 00320036 -.3.6.5.3.0.6.2.
0359d010 00370034 002d0034 00300031 00310030 4.7.4.-.1.0.0.1.
0359d020 004d005f 00630069 006f0072 006f0073 _M.i.c.r.o.s.o.
0359d030 00740066 00640045 00650067 00000000 f.t.E.d.g.e.....
```

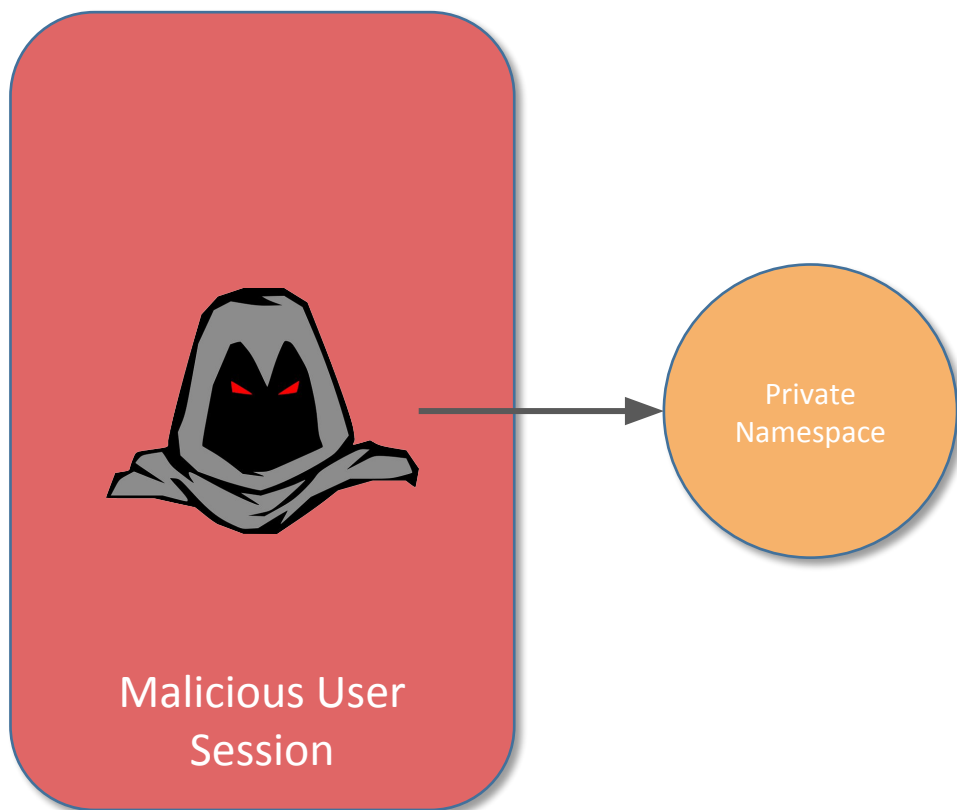
Name - IEUser_SID_MicrosoftEdge

Edge Private Namespace Object Security Descriptor

<i>Type</i>	<i>User</i>	<i>Access</i>
Allowed	Administrators	GENERIC_ALL
Allowed	OWNER RIGHTS	GENERIC_ALL
Allowed	ALL APPLICATION PACKAGES	GENERIC_ALL
Allowed	IE App Container SID	GENERIC_ALL
Allowed	Current User	GENERIC_ALL
Mandatory Label	Low Mandatory Level	

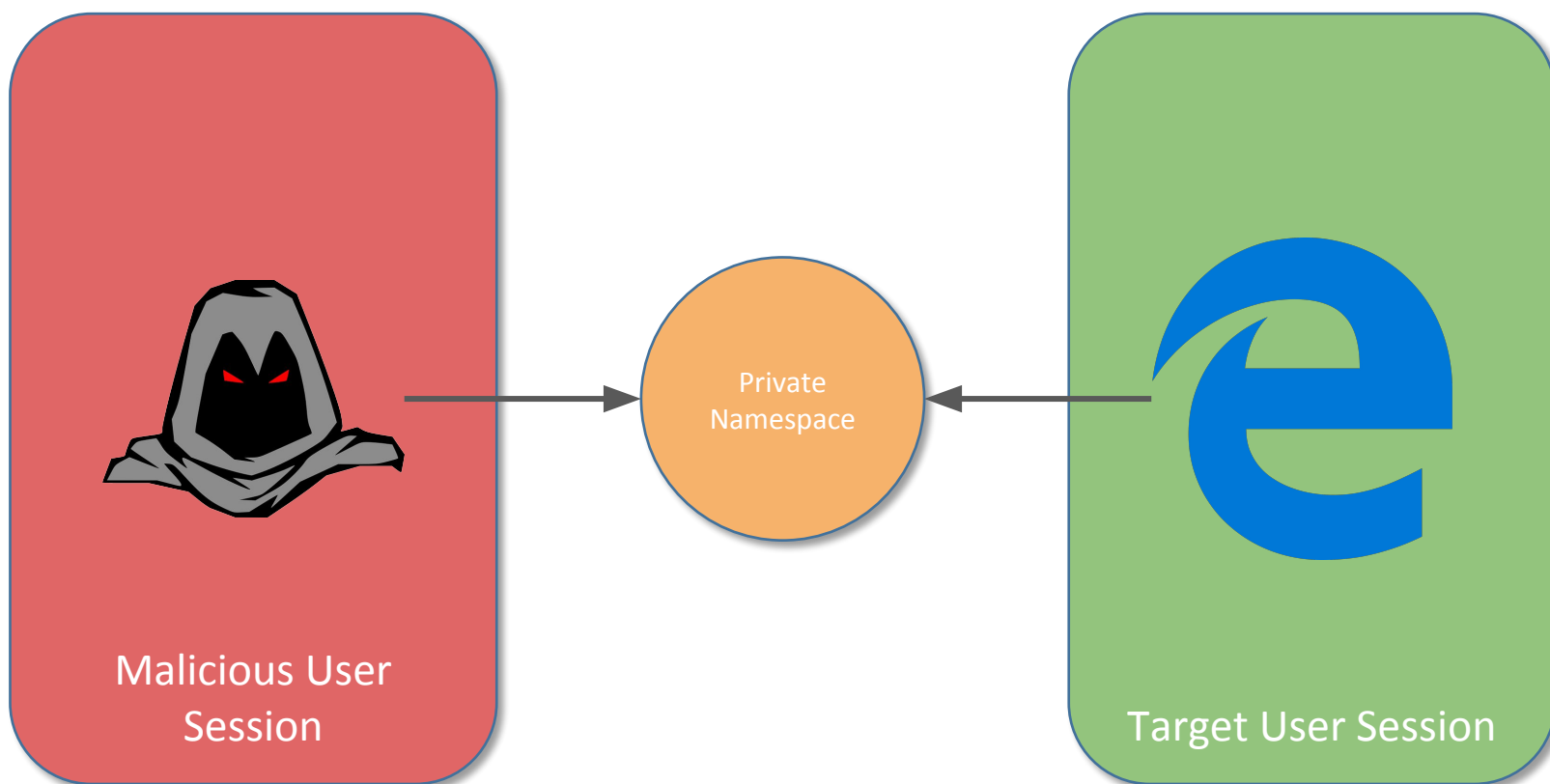
CVE-2016-3387 - Project Zero Issue 878

- The boundary descriptor doesn't limit the namespace to the current user.



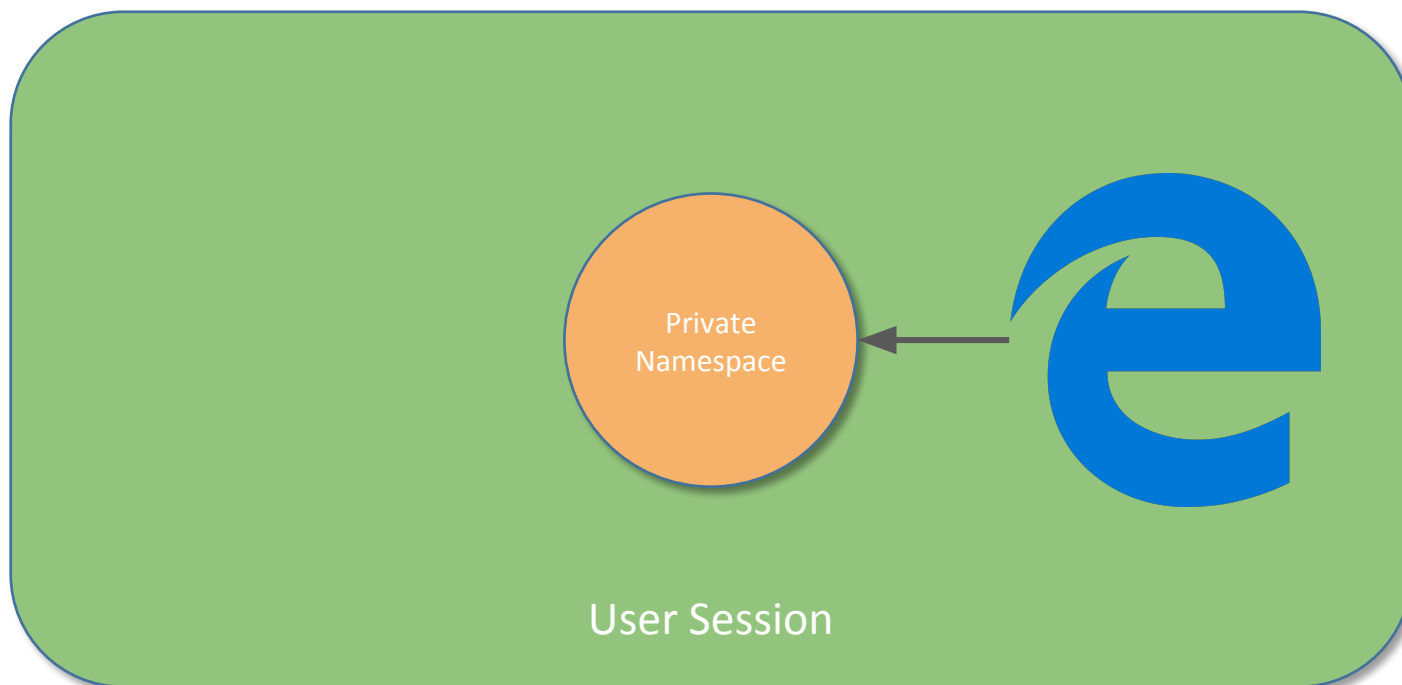
CVE-2016-3387 - Project Zero Issue 878

- The boundary descriptor doesn't limit the namespace to the current user.



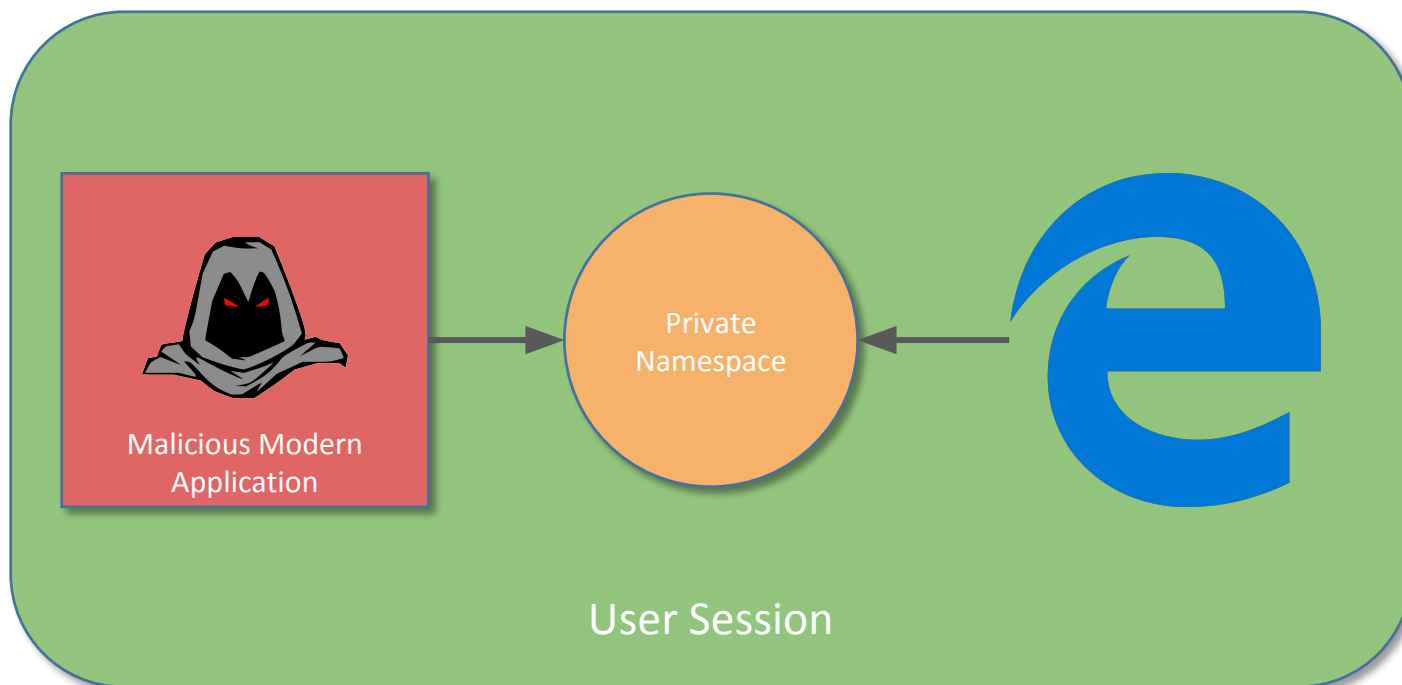
CVE-2016-3387 - Project Zero Issue 879

- The security descriptor on the namespace allows any other App Container process to access Edge's private namespace with *Full Control*



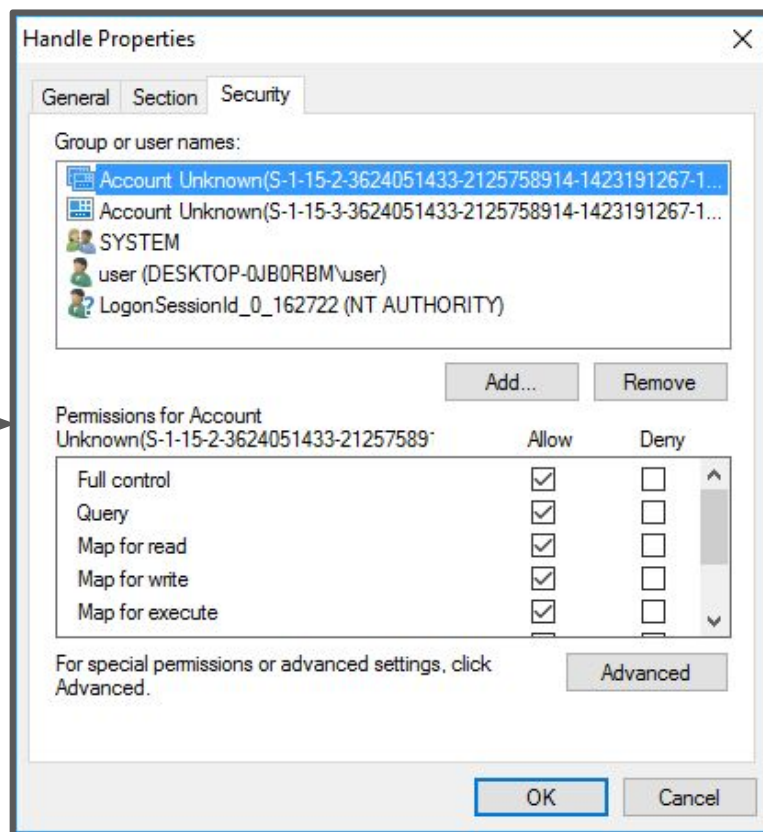
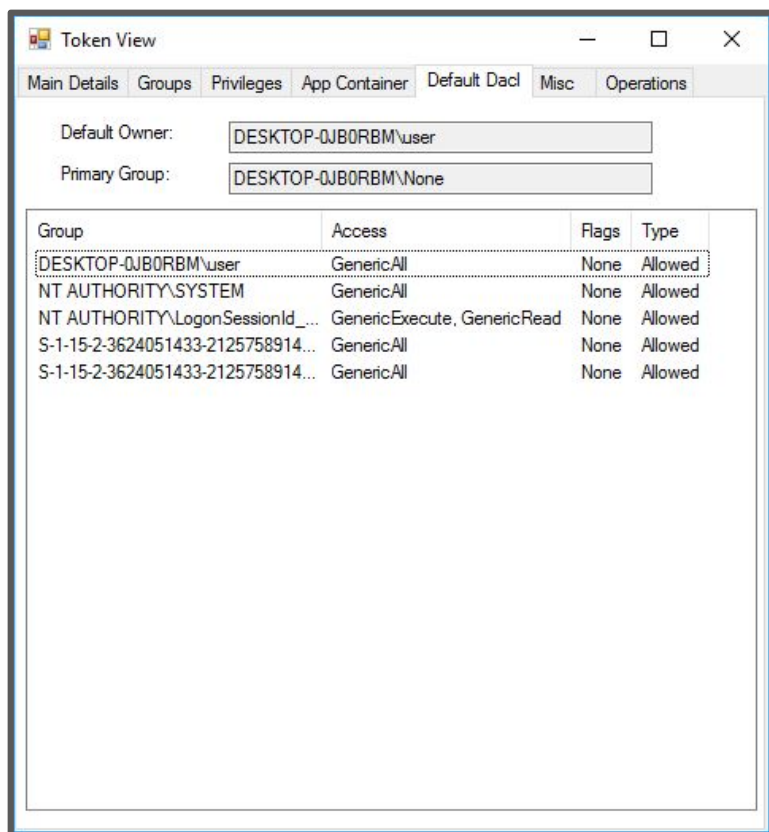
CVE-2016-3387 - Project Zero Issue 879

- The security descriptor on the namespace allows any other App Container process to access Edge's private namespace with *Full Control*



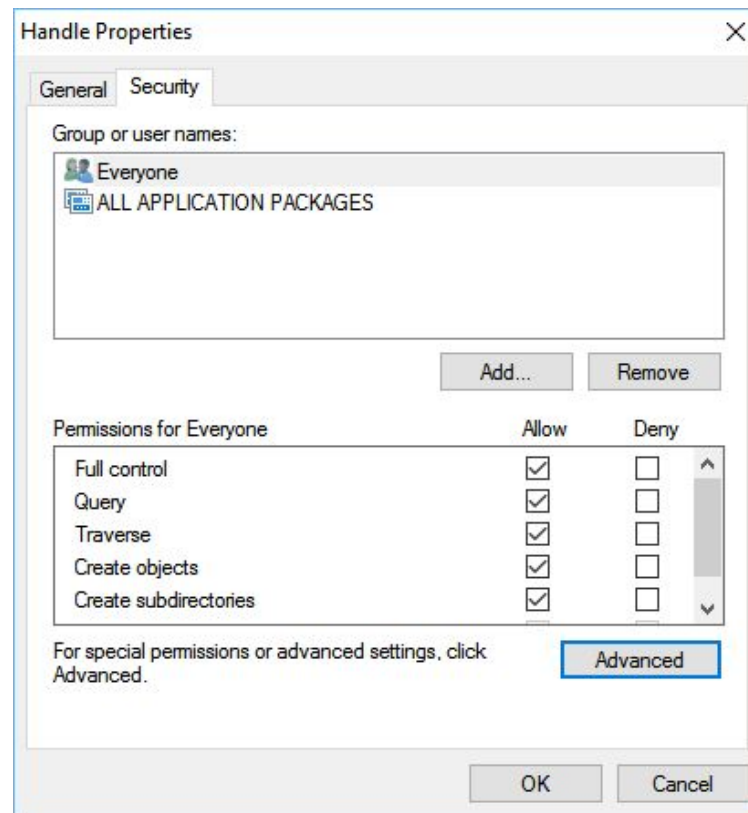
One Final Complication

- Access to Directory != Access to Resource Inside
- New objects uses the default DACL of the creator's token



Inheritable ACEs

- Modify DACL to add inheritable DACL and Mandatory Label ACEs



D: (A;OICIIO;GA;;;WD) (A;OICIIO;GA;;;AC) S: (ML;OICI;;;;S-1-16-0)

Documentation is at Fault

- Doesn't define the security importance of the Boundary Descriptor.
- *CreateBoundaryDescriptor* remarks: "A new boundary descriptor must have at least one security identifier (SID)". But this is not enforced by the Kernel or APIs.
- Doesn't make the clear distinction between a Namespace's *Security Descriptor* and the *Boundary Descriptor*.
 - I.e. Only SD access is needed to open a namespace, not access to the boundary
- Can still find buggy applications, Microsoft fixes were only for Edge/IE.

DEMO TIME

RegLoadAppKey

```
LONG WINAPI RegLoadAppKey(  
    _In_      LPCTSTR lpFile,  
    _Out_     PHKEY   phkResult,  
    _In_      REGSAM  samDesired,  
    _In_      DWORD   dwOptions,  
    _Reserved_ DWORD   Reserved  
);
```

dwOptions [in]

If this parameter is REG_PROCESS_APPKEY, the hive cannot be loaded again while it is loaded by the caller. This prevents access to this registry hive by another caller.

Reserved


This parameter is reserved.

Is the Parameter Used?

```
LONG RegLoadAppKeyW(LPCWSTR lpFile, PHKEY phkResult,
    REGSAM samDesired, DWORD dwOptions, DWORD Reserved) {
if (Reserved ||
    dwOptions && dwOptions != REG_PROCESS_APPKEY) {
return ERROR_INVALID_PARAMETER;
}
RtlInitUnicodeString(&file_name, lpFile);
BuildRootKeyName(&root_key);
InitializeObjectAttributes(&file_obj, &file_name, ...);
InitializeObjectAttributes(&key_obj, &root_key, ...);
NtLoadKeyEx(&key_obj, &file_obj,
    (dwOptions ? KEY_EXCLUSIVE : 0) | KEY_APP_HIVE,
    samDesired, phkResult);
// ...
}
```

Back to Documentation

Unlike `RegLoadKey`, `RegLoadAppKey` **does not load the hive under `HKEY_LOCAL_MACHINE` or `HKEY_USERS`**. Instead, the hive is **loaded under a special root that cannot be enumerated**. As a result, **there is no way to enumerate hives currently loaded by `RegLoadAppKey`**. All operations on hives loaded by `RegLoadAppKey` have to be performed relative to the handle returned in *phkResult*.

- 
- 1) Does not load hive under `\Registry\Machine` or `\Registry\User`
 - 2) Cannot enumerate existing loaded hives from where it does load them

Root Key Name

```
LONG RegLoadAppKeyW(LPCWSTR lpFile, PHKEY phkResult,
    REGSAM samDesired, DWORD dwOptions, DWORD Reserved) {
    if (Reserved ||
        dwOptions && dwOptions != REG_PROCESS_APPKEY) {
        return ERROR_INVALID_PARAMETER;
    }
    RtlInitUnicodeString(&file_name, lpFile);
    BuildRootKeyName(&root_key);
    InitializeObjectAttributes(&file_obj, &file_name, ...);
    InitializeObjectAttributes(&key_obj, &root_key, ...);
    NtLoadKeyEx(&key_obj, &file_obj,
        (dwOptions ? KEY_EXCLUSIVE : 0) | KEY_APP_HIVE,
        samDesired, phkResult);
    // ...
}
```

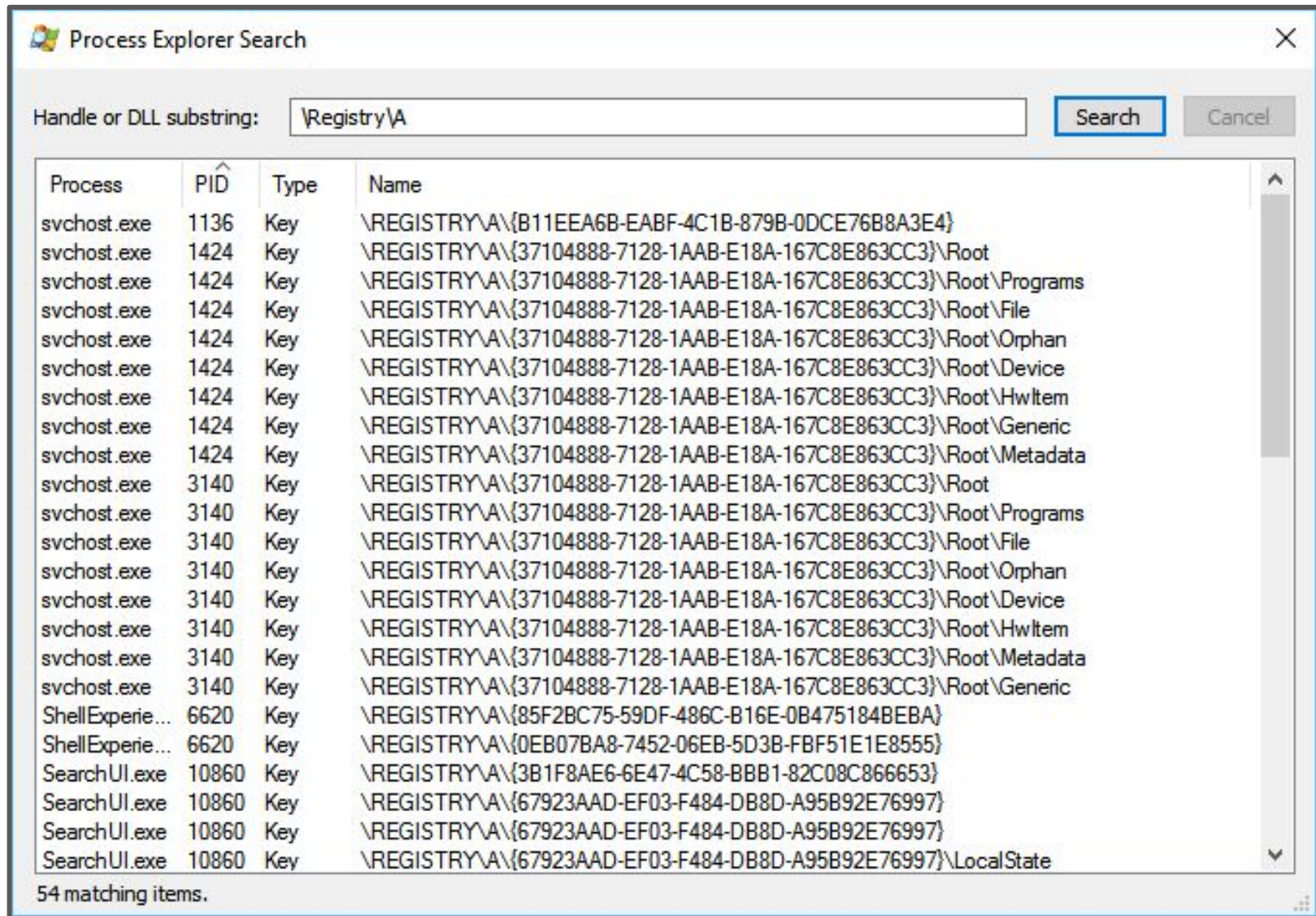
Building Root Key Name

```
NTSTATUS BuildRootKeyName(PUNICODE_STRING KeyName) {
    UNICODE_STRING GuidString;
    GUID Guid;

    RtlRandomEx(&Guid, sizeof(Guid));
    RtlStringFromGUID(&Guid, &GuidString);
    KeyName.Length = GuidString.Length + 26;
    KeyName.MaximumLength = KeyName.Length;
    KeyName.Buffer = RtlAllocateHeap(KeyName.Length);
    RtlAppendUnicodeToString(KeyName, "\\REGISTRY\\A\\");
    RtlAppendUnicodeStringToString(KeyName, &GuidString);
    return STATUS_SUCCESS;
}
```

Root Key Name of From: **\\Registry\\A\\{RANDOM GUID}**

Verify Location



Process Explorer Search

Handle or DLL substring:

Process	PID	Type	Name
svchost.exe	1136	Key	\REGISTRY\A\{B11EEA6B-EABF-4C1B-879B-0DCE76B8A3E4}
svchost.exe	1424	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root
svchost.exe	1424	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Programs
svchost.exe	1424	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\File
svchost.exe	1424	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Orphan
svchost.exe	1424	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Device
svchost.exe	1424	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\HwItem
svchost.exe	1424	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Generic
svchost.exe	1424	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Metadata
svchost.exe	3140	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root
svchost.exe	3140	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Programs
svchost.exe	3140	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\File
svchost.exe	3140	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Orphan
svchost.exe	3140	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Device
svchost.exe	3140	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\HwItem
svchost.exe	3140	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Metadata
svchost.exe	3140	Key	\REGISTRY\A\{37104888-7128-1AAB-E18A-167C8E863CC3}\Root\Generic
ShellExperie...	6620	Key	\REGISTRY\A\{85F2BC75-59DF-486C-B16E-0B475184BEBA}
ShellExperie...	6620	Key	\REGISTRY\A\{0EB07BA8-7452-06EB-5D3B-FBF51E1E8555}
SearchUI.exe	10860	Key	\REGISTRY\A\{3B1F8AE6-6E47-4C58-BBB1-82C08C866653}
SearchUI.exe	10860	Key	\REGISTRY\A\{67923AAD-EF03-F484-DB8D-A95B92E76997}
SearchUI.exe	10860	Key	\REGISTRY\A\{67923AAD-EF03-F484-DB8D-A95B92E76997}
SearchUI.exe	10860	Key	\REGISTRY\A\{67923AAD-EF03-F484-DB8D-A95B92E76997}\LocalState

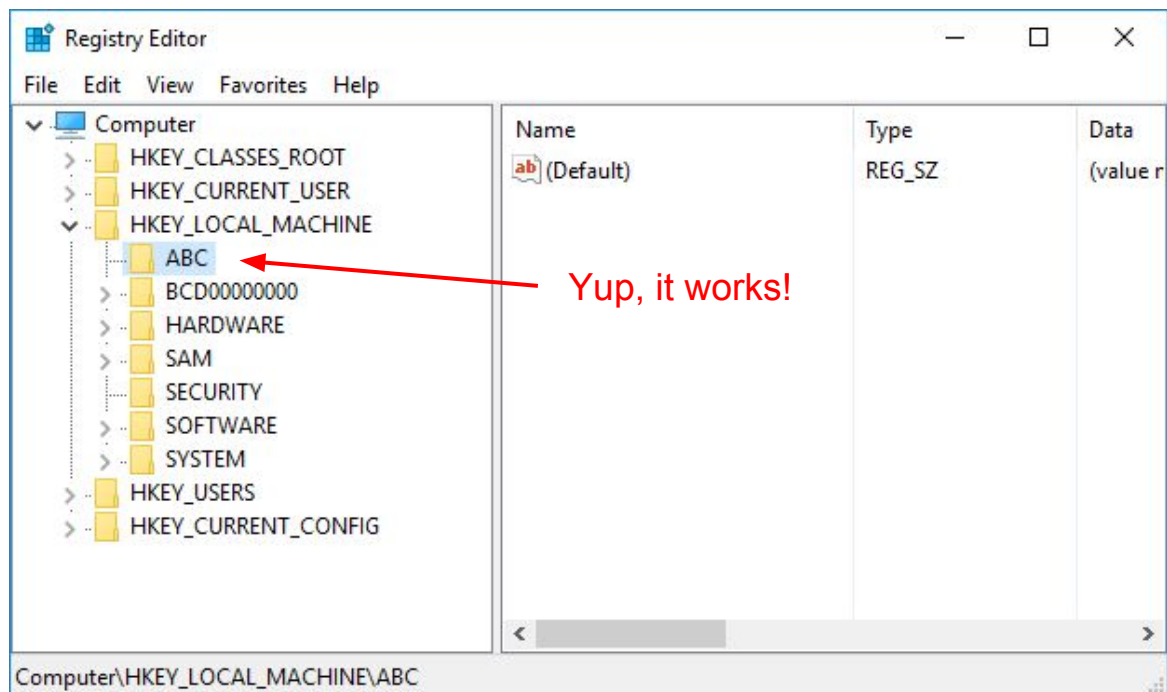
54 matching items.

Can You Load Hive Elsewhere?

```
RtlInitUnicodeString(&file_name, lpFile);  
RtlInitUnicodeString(&key_name, "\REGISTRY\MACHINE\ABC");  
InitializeObjectAttributes(&file_obj, &file_name, ...);  
InitializeObjectAttributes(&key_obj, &root_key, ...);  
NtLoadKeyEx(&key_obj, &file_obj,  
            KEY_APP_HIVE, samDesired, phkResult);
```

Can You Load Hive Elsewhere?

```
RtlInitUnicodeString(&file_name, lpFile);  
RtlInitUnicodeString(&key_name, "\REGISTRY\MACHINE\ABC");  
InitializeObjectAttributes(&file_obj, &file_name, ...);  
InitializeObjectAttributes(&key_obj, &root_key, ...);  
NtLoadKeyEx(&key_obj, &file_obj,  
            KEY_APP_HIVE, samDesired, phkResult);
```



Fixed as CVE-2016-3371
(Project Zero Issue 865)

Testing the Claim of Not Enumerable

- If we can't enumerate the keys then random GUID would prevent brute-force.
- Can we open the keys at all?

<code>OpenKey("\\Registry\\A\\{RANDOM GUID}")</code>	▶	<code>STATUS_ACCESS_DENIED</code>
<code>OpenKey("\\Registry\\A")</code>	▶	<code>STATUS_ACCESS_DENIED</code>

Testing the Claim of Not Enumerable

- If we can't enumerate the keys then random GUID would prevent brute-force.
- Can we open the keys at all?

OpenKey("\\Registry\\A\\{RANDOM GUID}")	▶	STATUS_ACCESS_DENIED
OpenKey("\\Registry\\A")	▶	STATUS_ACCESS_DENIED
OpenKey("\\Registry")	▶	STATUS_SUCCESS

Testing the Claim of Not Enumerable

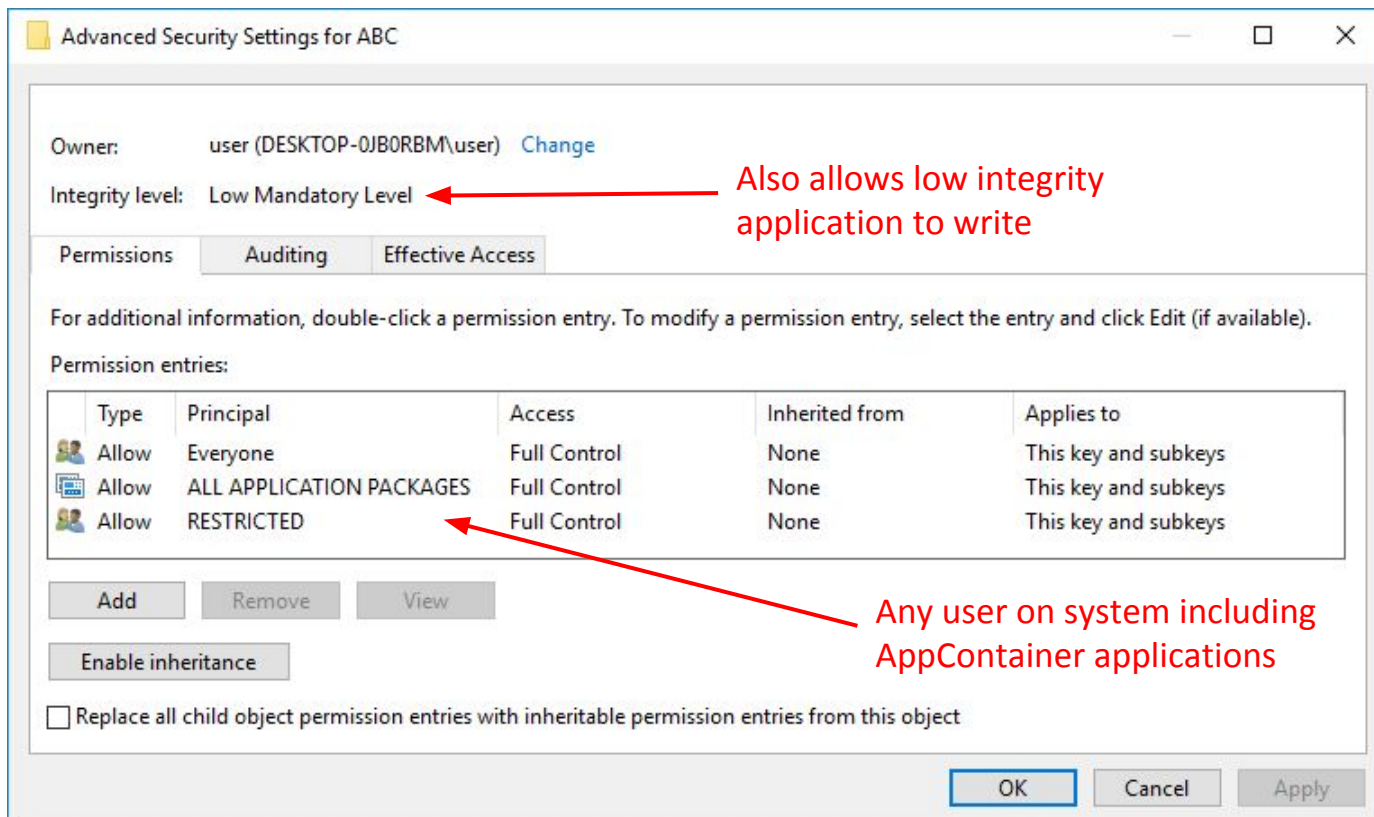
- If we can't enumerate the keys then random GUID would prevent brute-force.
- Can we open the keys at all?

OpenKey("\\Registry\\A\\{RANDOM GUID}")	▶	STATUS_ACCESS_DENIED
OpenKey("\\Registry\\A")	▶	STATUS_ACCESS_DENIED
OpenKey("\\Registry")	▶	STATUS_SUCCESS
OpenKey("\\Registry").OpenKey("A")	▶	STATUS_SUCCESS :-)

- Fixed in CVE-2016-3373 (Project Zero Issue 870)


No Control on New Hive Security

- Not documented what the default security descriptor of a new registry hive is. This makes the enumeration vulnerability *a lot* worse.



Any Interesting Options?

```
LONG RegLoadAppKeyW(LPCWSTR lpFile, PHKEY phkResult,
    REGSAM samDesired, DWORD dwOptions, DWORD Reserved) {
    if (Reserved ||
        dwOptions && dwOptions != REG_PROCESS_APPKEY) {
        return ERROR_INVALID_PARAMETER;
    }
    RtlInitUnicodeString(&file_name, lpFile);
    BuildRootKeyName(&root_key);
    InitializeObjectAttributes(&file_obj, &file_name, ...);
    InitializeObjectAttributes(&key_obj, &root_key, ...);
    NtLoadKeyEx(&key_obj, &file_obj,
        (dwOptions ? KEY_EXCLUSIVE : 0) | KEY_APP_HIVE,
        samDesired, phkResult);
    // ...
}
```



KEY_EXCLUSIVE = 0x20
KEY_APP_HIVE = 0x10

Read Only Flag

```
HRESULT CmpCmdHiveOpen(HANDLE *KeyHandle,
                      PUNICODE_STRING HiveFile,
                      HIVE_OPEN_FLAGS Flags) {
    BOOLEAN OpenReadOnly = FALSE;
    if (Flags & KEY_READ_ONLY)
        OpenReadOnly = TRUE;

    NTSTATUS status = CmpInitHiveFromFile(
        KeyHandle, HiveFile, OpenReadOnly);
    if (status == STATUS_ACCESS_DENIED) {
        RtlImpersonateSelfEx(SecurityImpersonation, 0, NULL);
        CmpInitHiveFromFile(KeyHandle, HiveFile, FALSE);
        PsRevertToSelf();
    }
    return status;
}
```


Read Only Flag

```
NTSTATUS CmpInitHiveFromFile(HANDLE *KeyHandle,
                           PUNICODE_STRING HiveFile,
                           BOOLEAN OpenReadOnly) {

    // ...
    HANDLE FileHandle;
    NTSTATUS status = CmpOpenHiveFile(&FileHandle, NULL,
                                     HiveFile, OpenReadOnly);

    if (status < 0)
        return status;

    if (!OpenReadOnly) {
        HANDLE LogHandle;
        RtlAppendUnicodeStringToString(HiveFile, L".LOG");
        status = CmpOpenHiveFile(&LogHandle, FileHandle,
                                HiveFile, FALSE);
    }
    return status;
}
```

Read Only Flag

```
NTSTATUS CmpOpenHiveFile(HANDLE* FileHandle,
                      HANDLE ParentFile,
                      PUNICODE_STRING FileName,
                      BOOLEAN OpenReadOnly) {
    PSECURITY_DESCRIPTOR security_desc = NULL;
    if (ParentFile)
        CmpQuerySecurityDescriptor(ParentFile, &security_desc);

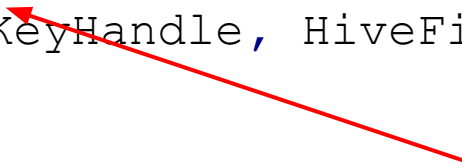
    ACCESS_MASK FileAccess = FILE_READ_DATA;
    if (!OpenReadOnly)
        FileAccess |= FILE_WRITE_DATA;

    InitializeObjectAttributes(&obj_attr, FileName, ...,
                              security_desc);
    return ZwCreateFile(FileHandle, FileAccess, &obj_attr,
                       OpenReadOnly ? FILE_OPEN : FILE_OPEN_IF);
}
```

Back to the Start

```
HRESULT CmpCmdHiveOpen(HANDLE *KeyHandle,
                      PUNICODE_STRING HiveFile,
                      HIVE_OPEN_FLAGS Flags) {
    BOOLEAN OpenReadOnly = FALSE;
    if (Flags & KEY_READ_ONLY)
        OpenReadOnly = TRUE;

    NTSTATUS status = CmpInitHiveFromFile(
        KeyHandle, HiveFile, OpenReadOnly);
    if (status == STATUS_ACCESS_DENIED) {
        RtlImpersonateSelfEx(SecurityImpersonation, 0, NULL);
        CmpInitHiveFromFile(KeyHandle, HiveFile, FALSE);
        PsRevertToSelf();
    }
    return status;
}
```




If Access Denied try again using
the process token

Back to the Start

```
HRESULT CmpCmdHiveOpen(HANDLE *KeyHandle,
                      PUNICODE_STRING HiveFile,
                      HIVE_OPEN_FLAGS Flags) {
    BOOLEAN OpenReadOnly = FALSE;
    if (Flags & KEY_READ_ONLY)
        OpenReadOnly = TRUE;

    NTSTATUS status = CmpInitHiveFromFile(
        KeyHandle, HiveFile, OpenReadOnly);
    if (status == STATUS_ACCESS_DENIED) {
        RtlImpersonateSelfEx(SecurityImpersonation, 0, NULL);
        CmpInitHiveFromFile(KeyHandle, HiveFile, FALSE);
        PsRevertToSelf();
    }
    return status;
}
```



Forgets to set the Read Only flag

DCOM Activation Service

Event Properties window showing event details:

- Date: 10/17/2016 2:56:26.4360549 PM
- Thread: 5528
- Class: File System
- Operation: CreateFile
- Result: SUCCESS
- Path: 3.0_x86__8wekyb3d8bbwe\ActivationStore\ActivationStore.dat
- Duration: 0.0000517

Desired Access: Read Attributes
Disposition: Open
Options: Open Reparse Point
Attributes: n/a
ShareMode: Read, Write, Delete
AllocationSize: n/a
Impersonating: DESKTOP-0JB0RBM\user
OpenResult: Opened

Impersonating User
Will revert to SYSTEM for file access

Event Properties window showing process details:

Image: Host Process for Windows Services, Microsoft Corporation

Name: svchost.exe
Version: 10.0.10586.0 (th2_release.151029-1700)

Path: C:\Windows\system32\svchost.exe
Command Line: C:\Windows\system32\svchost.exe -k DcomLaunch

PID: 720 Architecture: 32-bit
Parent PID: 616 Virtualized: False
Session ID: 0 Integrity: System
User: NT AUTHORITY\SYSTEM
Auth ID: 00000000:000003e7
Started: 10/17/2016 1:56:52 PM Ended: (Running)

Module	Address	Size	Path
svchost.exe	0x30000	0xc000	C:\Windows\system32\svchost.exe
execmodelproxy...	0x66680000	0xc000	C:\Windows\system32\execmodelproxy.exe

Fixed as CVE-2016-0079 (Project Zero Issue 871)

DEMO TIME

Wrap Up

- Using documentation to find bugs seemed to work surprisingly well.
- Plenty more things which could be done with documentation
 - Matching documented behaviour with SAL in the headers
 - Found some functions where MSDN says a parameter is IN/OUT and SAL says only OUT
 - Static analysis of exported functions
 - Static analysis of a binary's use of APIs w.r.t. SAL annotations
- APIs I've mentioned still have documentation problems which mean exploitable conditions could happen again

Questions?

