Into The Core

IN-DEPTH EXPLORATION OF WINDOWS 10 IOT CORE

Paul Sabanal
X-Force Advanced Research
IBM Security

August 3, 2016
Introduction
# Overview

<table>
<thead>
<tr>
<th>Edition</th>
<th>Description</th>
<th>Target Devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows 10 IoT</td>
<td>UWP apps, Win32 apps, desktop shell, x86, advanced</td>
<td>Kiosk, POS, ATM, Medical devices</td>
</tr>
<tr>
<td>Enterprise</td>
<td>lockdown</td>
<td></td>
</tr>
<tr>
<td>Windows 10 IoT</td>
<td>UWP apps, multiuser support, lockdown features</td>
<td>Mobile POS, Industry handheld terminals</td>
</tr>
<tr>
<td>Mobile</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Windows 10 IoT</td>
<td>For low-cost, low-power devices. UWP apps only.</td>
<td>Smart home devices, IoT gateway, digital signage</td>
</tr>
<tr>
<td>Core</td>
<td>ARM and x86</td>
<td></td>
</tr>
</tbody>
</table>
Overview

Raspberry Pi 2 & 3
- ARM
- 32-bit
- On-board Wi-fi and Bluetooth, Ethernet
- 4 x USB 2.0

Minnowboard Max
- x86
- 32-bit
- Ethernet
- 1 x USB 2.0, 1 x USB 3.0

Dragonboard 410c
- ARM
- On-board Wi-fi and Bluetooth
- 2 x USB 2.0
Internals
Internals > FFU

C:\>ImgMount.exe "c:\Program Files (x86)\Microsoft IoT\FFU\MinnowBoardMax\flash.ffu"

WP8 ROM Image Tools v.1.0.204
htc ROM Image Editor (r) 2007-2012 AnDim & XDA-Developers
ImgMount Tool v.1.0.15

(htcRIE) Mounting the image file: 'c:\Program Files (x86)\Microsoft IoT\FFU\MinnowBoardMax\flash.ffu'
Loading .FFU image ... ok
Creating virtual disk ... ok
Mounting MainOS partition as: '\\flash.mnt' ... ok
(htcRIE) Successfully mounted an image file.
## Internals > Partition Layout

<table>
<thead>
<tr>
<th>Partition</th>
<th>File System</th>
<th>Mount Point</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>EFI System Partition</td>
<td>FAT</td>
<td>C:\EFIESP</td>
<td>Boot manager, boot configurations, UEFI applications</td>
</tr>
<tr>
<td>Crash dump partition</td>
<td>FAT32</td>
<td>D:</td>
<td>Crash dump data</td>
</tr>
<tr>
<td>Main OS</td>
<td>NTFS</td>
<td>C:</td>
<td>OS, registry hives, OEM applications</td>
</tr>
<tr>
<td>Data partition</td>
<td>NTFS</td>
<td>U:</td>
<td>Applications, application data, user data</td>
</tr>
</tbody>
</table>
Internals > Boot Process

Device powers on and runs SoC firmware bootloader

Bootloader launched the UEFI environment and UEFI applications

UEFI environment launches Boot Manager (C:\EFI\ESP\EFI\Microsoft\boot\bootmgfw.elf)

Boot Manager launches Windows Boot Loader (C:\Windows\System32\Boot\winload.elf)

Windows Boot Loader launches main OS
Internals > Apps

• Universal Windows Platform (UWP) apps
  – Foreground/default apps
  – Background app

• Console applications
  – Win32 apps
  – No UI
  – C++ only

• Headed/Headless mode
  – UI or no UI
Internals > Security

Windows Defender

Microsoft Passport

Virtualization Based Security (VBS)

Device Guard
Credential Guard
Hypervisor Code Integrity (HVCI)
Internals > Security

- Address Space Layout Randomization (ASLR)
- Data Execution Prevention (DEP)
- Control Flow Guard (CFG)
Internals > Security

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firmware TPM</td>
<td>TPM implemented in the SoC</td>
</tr>
<tr>
<td>Discrete TPM</td>
<td>Chip module that can be attached to a board</td>
</tr>
<tr>
<td>Software TPM</td>
<td>Software emulated TPM used in development</td>
</tr>
</tbody>
</table>

- **Secure Boot**
  - Prevents device tampering during boot
  - Stops the system from running unverified binaries
  - Protects against boot kits, rootkits, and other low level malware

- **BitLocker**
  - Lightweight version of BitLocker
  - Encryption of user and system files
Internals > Windows Update

- Automatic forced update
- Check for updates through “Windows Update” tab of Windows Device Portal
- Pro edition allows deferred updates
Attack Surface
Starting Nmap 7.12 (https://nmap.org) at 2016-07-13 01:33 Malay Peninsula Standard Time
Nmap scan report for 10.0.1.108
Host is up (0.020s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
135/tcp  open  msrpc
445/tcp  open  microsoft-ds
8080/tcp open  http-proxy
MAC Address: B8:27:EB:B5:A9:E0 (Raspberry Pi Foundation)

Nmap done: 1 IP address (1 host up) scanned in 3.24 seconds
Attack Surface > Windows Device Portal

- http://<device ip>:8080

- Files can be found in \C:\Windows\WebManagement\www on the device

- User name: Administrator, password: p@ssw0rd

- Built on top of REST APIs
  - http://<device ip>:8080/restdocumentation.htm

<table>
<thead>
<tr>
<th>Utility</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home</td>
<td>Device information, change device name/password, timezone settings</td>
</tr>
<tr>
<td>Apps</td>
<td>Install/uninstall of apps</td>
</tr>
<tr>
<td>App File Explorer</td>
<td>File explorer for installed apps locations</td>
</tr>
<tr>
<td>Processes</td>
<td>Running processes list, process memory usage, and process termination</td>
</tr>
<tr>
<td>Performance</td>
<td>Real time graphical display of CPU and I/O usage</td>
</tr>
<tr>
<td>Debugging</td>
<td>Starting VS remote debugger, downloading of live kernel and process dumps</td>
</tr>
<tr>
<td>ETW</td>
<td>Event tracing</td>
</tr>
<tr>
<td>Perf Tracing</td>
<td>Trace logging of CPU, disk, and memory usage</td>
</tr>
<tr>
<td>Devices</td>
<td>Device manager for peripherals attached to the device</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>Bluetooth device search</td>
</tr>
<tr>
<td>Audio</td>
<td>Device speaker and microphone volume adjustments</td>
</tr>
<tr>
<td>Networking</td>
<td>WiFi configuration</td>
</tr>
<tr>
<td>Windows Update</td>
<td>Last update timestamp, check for updates</td>
</tr>
<tr>
<td>IoT Onboarding</td>
<td>Internet Connection Sharing settings, SoftAP settings, AllJoyn onboarding settings</td>
</tr>
<tr>
<td>TPM Configuration</td>
<td>TPM installation, configuration, and provisioning</td>
</tr>
<tr>
<td>Remote</td>
<td>Enable Windows IoT Remote Server</td>
</tr>
</tbody>
</table>
Attack Surface > Network services
Attack Surface > Windows Device Portal
### Attack Surface > Windows Device Portal

#### App Manager

<table>
<thead>
<tr>
<th>App Name</th>
<th>App Type</th>
<th>Startup</th>
</tr>
</thead>
<tbody>
<tr>
<td>IoTCoreDefaultApp (Default)</td>
<td>Foreground</td>
<td>Default App</td>
</tr>
<tr>
<td>IoTOnboardingTask (Startup)</td>
<td>Background</td>
<td>Remove from Startup</td>
</tr>
<tr>
<td>IoTUAP00E</td>
<td>Foreground</td>
<td>Set as Default App</td>
</tr>
<tr>
<td>ZWave Adapter Headless Host</td>
<td>Background</td>
<td>Add to Startup</td>
</tr>
</tbody>
</table>

### Install app

1. **App package**
   - Choose File: No file chosen
   - Dependency
     - Add dependency

2. **Certificate**
   - Choose File: No file chosen

3. **Deploy**
   - Go
   - Reset
Attack Surface > Windows Device Portal

Running Processes

Run command

<table>
<thead>
<tr>
<th>PID</th>
<th>NAME</th>
<th>USER NAME</th>
<th>SESSION ID</th>
<th>CPU</th>
<th>PRIVATE WORKING</th>
<th>WORKING SET</th>
<th>COMMIT SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>System Idle Process</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>77.13%</td>
<td>8.0 KB</td>
<td>8.0 KB</td>
<td>N/A</td>
</tr>
<tr>
<td>4</td>
<td>System</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>0.00%</td>
<td>8.0 KB</td>
<td>60.0 KB</td>
<td>N/A</td>
</tr>
<tr>
<td>268</td>
<td>smss.exe</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>0.00%</td>
<td>136.0 KB</td>
<td>796.0 KB</td>
<td>N/A</td>
</tr>
<tr>
<td>460</td>
<td>csrss.exe</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>0.00%</td>
<td>332.0 KB</td>
<td>19.0 MB</td>
<td>N/A</td>
</tr>
<tr>
<td>520</td>
<td>wininit.exe</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>0.00%</td>
<td>492.0 KB</td>
<td>3.3 MB</td>
<td>N/A</td>
</tr>
<tr>
<td>564</td>
<td>services.exe</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>0.00%</td>
<td>1.4 MB</td>
<td>4.1 MB</td>
<td>N/A</td>
</tr>
<tr>
<td>580</td>
<td>lsass.exe</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>0.00%</td>
<td>2.5 MB</td>
<td>9.9 MB</td>
<td>N/A</td>
</tr>
<tr>
<td>664</td>
<td>dwm.exe</td>
<td>Window Manager\DWM-0</td>
<td>0</td>
<td>0.00%</td>
<td>5.5 MB</td>
<td>16.2 MB</td>
<td>10.1 MB</td>
</tr>
<tr>
<td>672</td>
<td>svchost.exe</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>0.00%</td>
<td>2.1 MB</td>
<td>10.8 MB</td>
<td>3.6 MB</td>
</tr>
<tr>
<td>724</td>
<td>svchost.exe</td>
<td>NT AUTHORITY\NETWORK</td>
<td>0</td>
<td>0.00%</td>
<td>1.5 MB</td>
<td>5.3 MB</td>
<td>2.2 MB</td>
</tr>
<tr>
<td>740</td>
<td>sihost.exe</td>
<td>kapre\DefaultAccount</td>
<td>0</td>
<td>0.00%</td>
<td>2.1 MB</td>
<td>13.9 MB</td>
<td>3.4 MB</td>
</tr>
<tr>
<td>848</td>
<td>svchost.exe</td>
<td>NT AUTHORITY\SYSTEM</td>
<td>0</td>
<td>0.00%</td>
<td>7.1 MB</td>
<td>25.5 MB</td>
<td>9.3 MB</td>
</tr>
</tbody>
</table>
Attack Surface > Windows Device Portal
Attack Surface > Network services

• SSH
  – Enabled by default
  – Starts at boot

• Windows File Sharing
  – Enabled by default
  – Starts at boot

• Windows IoT Remote Server
  – Remote UI client installed from Windows Store
  – Can be enabled through the Remote tab in the Windows Device Portal
  – No authentication
  – NanoRDPServer.exe
Windows IoT Remote Server

- Enable Windows IoT Remote Server

Windows IoT Remote Server is enabled at boot and currently running.

Windows IoT Remote Server enables users to connect from another computer over a network connection to this device using a client app. To access Windows IoT Core device, install the remote client app from [https://www.microsoft.com/store/apps/9nblggh5mnx](https://www.microsoft.com/store/apps/9nblggh5mnx)
Attack Surface > Device drivers

• Drivers for built-in or external peripherals

• Drivers for wireless adapters
  – Wifi
  – Bluetooth
  – ZigBee
  – Z-Wave

• Successful exploitation often results in kernel level privilege
Attack Surface > Malware

- Password guessing/brute forcing of login credentials
- Vulnerabilities in the network services
- Lateral infection coming from other machines

C:\>mimikatz.exe

.#####. mimikatz 2.1 (x64) built on Jul 11 2016 00:32:57
.## ^ ##. "A La Vie, A L'Amour"
## / ## /* * */
## \\ ## Benjamin DELPY `gentilkiwi` (benjamin@gentilkiwi.com)
'## v ##' http://blog.gentilkiwi.com/mimikatz (oe.oe)
'#####' with 20 modules * * */
mimikatz # privilege::debug
Privilege '20' OK
mimikatz # sekurlsa::ssp

Authentication Id : 0 ; 247557 (00000000:0003c705)
Session : Interactive from 1
User Name : polsab
Domain : DESKTOP-39HUL88
Logon Server : (null)
Logon Time : 7/20/2016 6:15:59 PM
SID : S-1-5-21-4294090806-594742593-2658599142-1001

ssp :
[00000000]
* Username : Administrator
* Domain : 10.0.1.108
* Password : diwata
Hacking
Hacking > Device Discovery

My devices

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>IP Address</th>
<th>Settings</th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>cliwata</td>
<td>Raspberry Pi 3</td>
<td>10.0.1.108</td>
<td></td>
<td>10.0.14376.0</td>
</tr>
<tr>
<td>kapre</td>
<td>Raspberry Pi 3</td>
<td>10.0.1.110</td>
<td></td>
<td>10.0.14376.0</td>
</tr>
</tbody>
</table>
Hacking > Device Discovery

Offset | Description
--- | ---
0 | Device name
0x42 | IP address
0x64 | MAC address
0x96 | Board serial number
0xe6 | Device Type
0x14a | OS version
0x1ae | Device architecture
### Hacking > PowerShell

- **Remote device administration and configuration**

- **Built-in and 3rd party tools for penetration testing and reversing. Ex:**
  - CimSweep
  - Autoruns

```powershell
PS C:\WINDOWS\system32> $CimSessionPi2 = New-CimSession -ComputerName 10.0.1.110 -Credential Administrator
PS C:\WINDOWS\system32> Get-CSRegistryAutoStart -CimSession $CimSessionPi2

<table>
<thead>
<tr>
<th>Path</th>
<th>AutoRunEntry</th>
<th>ImagePath</th>
<th>Category</th>
<th>PSComputerName</th>
</tr>
</thead>
<tbody>
<tr>
<td>HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon</td>
<td>Shell</td>
<td>IotShell.exe</td>
<td>Logon</td>
<td>10.0.1.110</td>
</tr>
<tr>
<td>HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon</td>
<td>Userinit</td>
<td>userinit.exe</td>
<td>Logon</td>
<td>10.0.1.110</td>
</tr>
<tr>
<td>HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon</td>
<td>VMApplet</td>
<td>SystemPropertiesPerformance.exe /pagefile</td>
<td>Logon</td>
<td>10.0.1.110</td>
</tr>
<tr>
<td>HKLM\SYSTEM\CurrentControlSet\Control\Session Manager</td>
<td>BootExecute</td>
<td>autocheck autochk</td>
<td>BootExecute</td>
<td>10.0.1.110</td>
</tr>
</tbody>
</table>
```

<snip>
Hacking > Static analysis

- UWP apps can be found in Data partition (U:\, also linked with C:\Data)

- App installed in U:\Programs \WindowsApps

- Lib DLLs and XBF (binary XAML)

- Assets folder
  - Images
  - Fonts
  - etc

<table>
<thead>
<tr>
<th>Filename</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;app_name&gt;.exe</td>
<td>App startup stub</td>
</tr>
<tr>
<td>&lt;app_name&gt;.dll</td>
<td>App code</td>
</tr>
<tr>
<td>AppManifest.xml</td>
<td>UWP app package manifest</td>
</tr>
<tr>
<td>AppBlockMap.xml</td>
<td>Cryptographic block hashes for files in package</td>
</tr>
<tr>
<td>AppxSignature.p7x</td>
<td>App package digital signature file</td>
</tr>
</tbody>
</table>
Hacking > Static analysis

```plaintext
; Section 4. (virtual address 00006000)
; Virtual size : 00000010 ( 16.)
; Section size in file : 00000200 ( 512.)
; Offset to raw data for section: 00003600
; Flags 60020020: Text Executable Readable
; Alignment : default

; Segment type: Pure code
AREA .text, CODE, ALIGN=4
; ORG 0x406000
CODE16

EXPORT start
start
MOV R12, #RHBinder__ShimExeMain
LDR.W PC, [R12]
; End of function start
```
Hacking > Kernel debugging
Hacking > Kernel debugging

```bash
# Enable serial debugging
bcdedit -dbgsettings serial
# Turn on debugging
bcdedit -debug on
```

```powershell
Get-WMIObject Win32_pnpentity | ? Name -like "*Serial*COM*"
```

```plaintext
__GENUS : 2
__CLASS : Win32_PnPEntity
__SUPERCLASS : CIM_LogicalDevice
__DYNASTY : CIM_ManagedSystemElement
__RELPATH : Win32_PnPEntity.DeviceID="FTDIBUS\VID_0403+PID_6014+5&3278CBC5&0&3\0000"
__PROPERTY_COUNT : 26
__DERIVATION : {CIM_LogicalDevice, CIM_LogicalElement, CIM_ManagedSystemElement}
__SERVER : DESKTOP-39HUL88
__NAMESPACE : root\cimv2
__PATH : \\DESKTOP-39HUL88\root\cimv2:Win32_PnPEntity.DeviceID="FTDIBUS\VID_0403+PID_6014+5&3278CBC5&0&3\0000"
Availability : 
Caption : USB Serial Port [COM3]
ClassGuid : {4d36e978-e325-11ce-bfc1-08002be10318}
CompatibleID : 
```
Hacking > Kernel debugging

# PORT is the COM port number used by your USB-to-serial adapter
windbg.exe -k com:port=<PORT>,baud=921600

Microsoft (R) Windows Debugger Version 10.0.10586.567 X86
Copyright (c) Microsoft Corporation. All rights reserved.
Opened \\.\com3
Waiting to reconnect...

Connected to Windows 10 14393 ARM (NT) Thumb-2 target at (Sun Jul 24 19:32:43.111 2016 (UTC + 8:00)), ptr64 FALSE
Kernel Debugger connection established.
Symbol search path is: srv*
Executable search path is:
*** ERROR: Symbol file could not be found. Defaulted to export symbols for ntkrnlmp.exe -
Windows 10 Kernel Version 14393 MP (1 procs) Free ARM (NT) Thumb-2
Built by: 14393.0.armfre.rs1_release.160715-1616
Machine Name:
Kernel base = 0x80c1b000 PsLoadedModuleList = 0x80e07c78
System Uptime: 0 days 0:00:00.000
Break instruction exception - code 80000003 (first chance)
*******************************************************************************
*                                                                             *
*   You are seeing this message because you pressed either                    *
*       CTRL+C (if you run console kernel debugger) or,                       *
*       CTRL+BREAK (if you run GUI kernel debugger),                          *
*   on your debugger machine's keyboard.                                      *
*                                                                             *
*                   THIS IS NOT A BUG OR A SYSTEM CRASH                       *
*                                                                             *
* If you did not intend to break into the debugger, press the "g" key, then   *
* press the "Enter" key now. This message might immediately reappear. If it *
* does, press "g" and "Enter" again.                                          *
*******************************************************************************
*** ERROR: Symbol file could not be found. Defaulted to export symbols for ntkrnlmp.exe -
nt!DbgBreakPointWithStatus: 80c40d90 defe __debugbreak
Hacking > User mode debugging

# PORT is the local port you want dbgsrv to listen on
C:\Windows\System32\Debuggers\dbgsrv.exe -t tcp:port=<PORT>
Microsoft (R) Windows Debugger Version 10.0.10586.567 X86
Copyright (c) Microsoft Corporation. All rights reserved.

*** wait with pending attach
Symbol search path is: srv*
Executable search path is:
ModLoad: 01110000 011db000 C:\windows\system32\WebManagement.exe
ModLoad: 77400000 77565000 C:\windows\SYSTEM32\ntdll.dll
ModLoad: 77270000 773fe000 C:\windows\System32\KERNELBASE.dll

(69c.280): Break instruction exception - code 80000003 (first chance)
ntdll!DbgBreakPoint: 77422740 defe __debugbreak
0:005> !peb

InheritedAddressSpace: No
ReadImageFileExecuteOptions: No
BeingDebugged: Yes
ImageBaseAddress: 01110000
Ldr 774eb9e0
Ldr.Initialized: Yes
Ldr.InInitializationOrderModuleList: 00c41738 . 00c4fcd0
Ldr.InLoadOrderModuleList: 00c41810 . 00c4fcc0
Ldr.InMemoryOrderModuleList: 00c41818 . 00c4fcc8

Base TimeStamp Module
11100000 57898ebe Jul 16 09:32:46 2016 C:\windows\system32\WebManagement.exe
77400000 57898ba5 Jul 16 09:19:33 2016 C:\windows\SYSTEM32\ntdll.dll
77270000 57898c4c Jul 16 09:22:20 2016 C:\windows\System32\KERNELBASE.dll

0:005> u $exentry
WebManagement+0xa6631:
011b6630 e92d4800 push {r11,lr}
011b6634 46eb mov r11,sp
011b663e f000fb65 bl WebManagement+0xa6d04 (011b6d04)
011b664a e8bd4800 pop {r11,lr}
011b665e f7ffbf25 b.w WebManagement+0xa648c (011b648c)
011b6664 0000 movs r0,0
011b666e f24c6c64 mov r12,#0xC664
011b6678 f2c01c1c movt r12,#0x11C
Hacking > Crash Dump

- Start Visual Studio Remote Debugger
  - Start
  - Run as Default Account

- Live kernel dumps
  - Download live kernel dump
  - Bugcheck dumps on device

- Live process dumps
  - Refresh list
    - PID | NAME
    - 0   | System Idle Process
    - 4   | System
    - 268 | smss.exe
    - 102 |
Microsoft (R) Windows Debugger Version 10.0.10586.567 X86
Copyright (c) Microsoft Corporation. All rights reserved.

Loading Dump File [d:\winiot\WebManagement.exe-LiveUM-2016-07-24-12-36-09.dmp]
User Mini Dump File: Only registers, stack and portions of memory are available

Symbol search path is: srv*
Executable search path is:
Windows 10 Version 14376 MP (4 procs) Free ARM (NT) Thumb-2
Product: WinNt, suite: SingleUserTS
Built by: 10.0.14376.0 (rs1_release.160624-1700)
Machine Name: 
Debug session time: Mon Jul 25 03:36:09.000 2016 (UTC + 8:00)
System Uptime: not available
Process Uptime: 1 days 4:48:37.000

.........................

....

Loading unloaded module list

Cannot read PEB32 from WOW64 TEB32 ffffffff - Win32 error 0n30
Unable to load image C:\Windows\System32\ntdll.dll, Win32 error On2
** WARNING: Unable to verify timestamp for ntdll.dll
ntdll!NtWaitForSingleObject+0x6:
** WARNING: Unable to verify timestamp for KERNELBASE.dll
77320ab6 4770     bx          lr {KERNELBASE!WaitForSingleObjectEx+0xc0 (76fedf30)}
0:000> |
0:000> !peb
PEB at 032f8000
InheritedAddressSpace:   No
ReadImageFileExecOptions: No
BeingDebugged:            No
ImageBaseAddress:         00a00000
Ldr                       773eb9e0
Ldr.Initialized:          Yes
Ldr.InInitializationOrderModuleList: 034a1730 . 034ae758
Ldr.InLoadOrderModuleList:  034a1808 . 034ae748
Ldr.InMemoryOrderModuleList: 034a1810 . 034ae750
Base TimeStamp          Module
a000000 576dee48 Jun 25 10:36:56 2016 C:\windows\system32\WebManagement.exe
77300000 576deb18 Jun 25 10:23:20 2016 C:\windows\SYSTBM32\ntdll.dll
76f20000 576debe7 Jun 25 10:26:47 2016 C:\windows\System32\KERNELBASE.dll
770b0000 576debda Jun 25 10:26:34 2016 C:\windows\System32\combase.dll
76ce0000 576deb16 Jun 25 10:23:18 2016 C:\windows\System32\ucrtbase.dll
76e30000 576ded32 Jun 25 10:32:18 2016 C:\windows\System32\RPCRT4.dll
76ed0000 576deeeb Jun 25 10:36:11 2016 C:\windows\System32\kernel32legacy.dll
76d90000 576deea6 Jun 25 10:38:34 2016 C:\windows\System32\bcryptPrimitives.dll
Hacking > Fuzzing

• Current Approach
  – Old school
  – REST APIs to control device

• Future Approach
  – Corpus driven fuzzing
  – WinAFL
Recommendations
**Recommendations**

**Segment your network**
- Mitigates lateral infection
- Incident isolation and cleanup

**Protect network services**
- Use built-in firewall
- Disable unnecessary services

**Change default Administrator password**
-Eliminates most malware infection attempts today

**Use devices supporting TPM**
- Minnowboard + Dragonboard
- Raspberry Pi + Discrete TPM

**Take advantage of available security features**
- Enable Secure Boot
- Enable BitLocker
Conclusion
Conclusion

Windows 10 IoT Core’s features makes it an attractive alternative to today’s IoT OS

Attack surface is smaller than other computing devices, but if IoT services are factored in, will be bigger

Vendors/makers should be careful about mis-configurations

More security research needed/encouraged
Questions?
THANK YOU

FOLLOW US ON:

- ibm.com/security
- securityintelligence.com
- xforce.ibmcloud.com
- @ibmsecurity
- youtube/user/ibmsecuritysolutions

© Copyright IBM Corporation 2016. All rights reserved. The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM’s sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. IBM, the IBM logo, and other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.

Statement of Good Security Practices: IT system security involves protecting systems and information through prevention, detection and response to improper access from within and outside your enterprise. Improper access can result in information being altered, destroyed, misappropriated or misused or can result in damage to or misuse of your systems, including for use in attacks on others. No IT system or product should be considered completely secure and no single product, service or security measure can be completely effective in preventing improper use or access. IBM systems, products and services are designed to be part of a lawful, comprehensive security approach, which will necessarily involve additional operational procedures, and may require other systems, products or services to be most effective.

IBM DOES NOT WARRANT THAT ANY SYSTEMS, PRODUCTS OR SERVICES ARE IMMUNE FROM, OR WILL MAKE YOUR ENTERPRISE IMMUNE FROM, THE MALICIOUS OR ILLEGAL CONDUCT OF ANY PARTY.