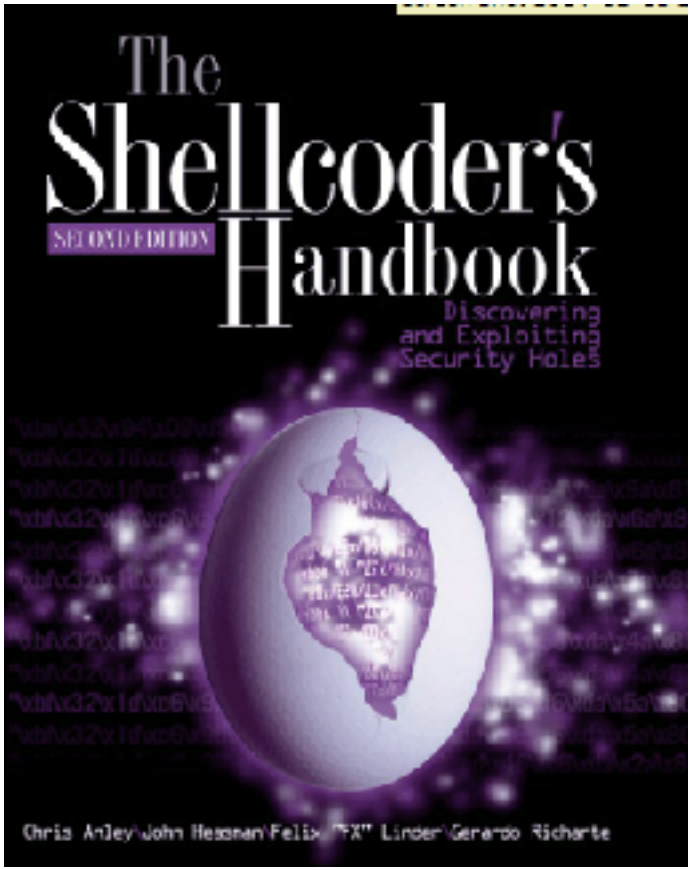
CNIT 127: Exploit Development

Ch 6: The Wild World of Windows

37



Revised 2-23-17

Topics

- Win32 API, DLLs, and PE Files
- Heaps
- Threading
- DCOM
- Exception Handling
- Debuggers

Win32 API, DLLs, and PE Files

Windows API
(Application Programming Interface)

- In Linux, a programmer can talk directly to the kernel with syscalls (INT 0x80)
- But in Windows the kernel is only accessible through the Windows API
- Implemented as a set of DLLs
- Changes with each Windows version and Service Pack

Windows API
(Application Programming Interface)

- Every process using the Windows API must use dynamic linking to the DLLs
- The Windows API changes more often than Linux Syscalls do
- Here's an API call to make a window

```
hwnd = CreateWindowEx(
    WS_EX_CLIENTEDGE,
    g_szClassName,
    "The title of my window",
    WS_OVERLAPPEDWINDOW,
    CW_USEDEFAULT, CW_USEDEFAULT, 240, 120,
    NULL, NULL, hInstance, NULL);
```

DLLs
(Dynamic Link Libraries)

- Pre-compiled library code
- Loaded as needed when executable files run
- You can see loaded DLLs with Process Explorer
  - View, Lower Pane View, DLLs
  - Link Ch 6b

Process Explorer - Sysinternals: www.sysinternals.com [WIN-8LDVLI8QDEN\sam]

File Options View Process Find DLL Users Help

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
System Idle Process	92.37	0 K	8 K	0		
System	0.31	44 K	560 K	4		
Interrupts	0.88	0 K	0 K	n/a	Hardware Interrupts and DPCs	
smss.exe		196 K	204 K	500		
csrss.exe		736 K	1,436 K	592		
csrss.exe	0.02	976 K	3,116 K	656		
wininit.exe		688 K	384 K	668		
services.exe	< 0.01	1,816 K	3,044 K	764		
svchost.exe	0.02	2,924 K	5,948 K	840	Host Process for Windows S...	Microsoft Corporation
WmiPrvSE.exe		1,420 K	6,372 K	5048		

Name	Description	Company Name	Path
advapi32.dll	Advanced Windows 32 Base API	Microsoft Corporation	C:\Windows\System32\advapi32.dll
bcryptprimitives.dll	Windows Cryptographic Primitives Library	Microsoft Corporation	C:\Windows\System32\bcryptprimitives.dll
combase.dll	Microsoft COM for Windows	Microsoft Corporation	C:\Windows\System32\combase.dll
comctl32.dll	User Experience Controls Library	Microsoft Corporation	C:\Windows\WinSxS\x86_microsoft.windows.common-cont...
comdlg32.dll	Common Dialogs DLL	Microsoft Corporation	C:\Windows\System32\comdlg32.dll
cryptbase.dll	Base cryptographic API DLL	Microsoft Corporation	C:\Windows\System32\cryptbase.dll
dwmapi.dll	Microsoft Desktop Window Manager API	Microsoft Corporation	C:\Windows\System32\dwmapi.dll
gdi32.dll	GDI Client DLL	Microsoft Corporation	C:\Windows\System32\gdi32.dll
imm32.dll	Multi-User Windows IMM32 API Client DLL	Microsoft Corporation	C:\Windows\System32\imm32.dll
kernel.appcore.dll	AppModel API Host	Microsoft Corporation	C:\Windows\System32\kernel.appcore.dll
kernel32.dll	Windows NT BASE API Client DLL	Microsoft Corporation	C:\Windows\System32\kernel32.dll
KernelBase.dll	Windows NT BASE API Client DLL	Microsoft Corporation	C:\Windows\System32\KernelBase.dll
locale.nls			C:\Windows\System32\locale.nls
msctf.dll	MSCTF Server DLL	Microsoft Corporation	C:\Windows\System32\msctf.dll
msvcrt.dll	Windows NT CRT DLL	Microsoft Corporation	C:\Windows\System32\msvcrt.dll
notepad.exe	Notepad	Microsoft Corporation	C:\Windows\System32\notepad.exe
notepad.exe.mui	Notepad	Microsoft Corporation	C:\Windows\System32\en-US\notepad.exe.mui
ntdll.dll	NT Layer DLL	Microsoft Corporation	C:\Windows\System32\ntdll.dll
ole32.dll	Microsoft OLE for Windows	Microsoft Corporation	C:\Windows\System32\ole32.dll
oleaut32.dll		Microsoft Corporation	C:\Windows\System32\oleaut32.dll

CPU Usage: 7.63% | Commit Charge: 35.79% | **Processes: 42** | Physical Usage: 68.38%

PE (Portable Executable) Files

- Format used for .EXE and .DLL files
  - And some other extensions (link Ch 6c)
- Can be loaded on every 32-bit (or 64-bit) Windows version
- Contains information about all required DLLs
- Easy to see with PEView (link Ch 6d)

Import Table for Notepad

PEview - C:\Windows\System32\notepad.exe

File View Go Help

- notepad.exe
  - IMAGE_DOS_HEADER
  - MS-DOS Stub Program
  - IMAGE_NT_HEADERS
  - IMAGE_SECTION_HEADER .text
  - IMAGE_SECTION_HEADER .data
  - IMAGE_SECTION_HEADER .rsrc
  - IMAGE_SECTION_HEADER .reloc
  - BOUND IMPORT Directory Table
  - BOUND IMPORT DLL Names
  - SECTION .text
  - SECTION .data
  - SECTION .rsrc
  - SECTION .reloc

pFile	Data	Description	Value
00000278	4791A64B	Time Date Stamp	2008/01/19 Sat 07:27:07 UTC
0000027C	0070	Offset to Module Name	ADVAPI32.dll
0000027E	0000	Number of Module Forwarder Refs	
00000280	4791A76D	Time Date Stamp	2008/01/19 Sat 07:31:57 UTC
00000284	007D	Offset to Module Name	KERNEL32.dll
00000286	0000	Number of Module Forwarder Refs	
00000288	4791A6A5	Time Date Stamp	2008/01/19 Sat 07:28:37 UTC
0000028C	008A	Offset to Module Name	GDI32.dll
0000028E	0000	Number of Module Forwarder Refs	
00000290	4791A773	Time Date Stamp	2008/01/19 Sat 07:32:03 UTC
00000294	0094	Offset to Module Name	USER32.dll
00000296	0000	Number of Module Forwarder Refs	
00000298	4791A727	Time Date Stamp	2008/01/19 Sat 07:30:47 UTC
0000029C	009F	Offset to Module Name	msvcrt.dll
0000029E	0000	Number of Module Forwarder Refs	

Viewing BOUND IMPORT Directory Table

- Windows Server 2008 Version

Sections of a PE File

- .text – instructions to execute
- .data – global variables
- .idata – Import descriptors
- .rsrc – Resources (icons, etc.)
- .reloc – Relocation data

Relocating PE Files

- DLLs have a Base Address
  - This is where they are designed to load
- But two DLLs might have the same Base Address
  - And both be used by the same EXE
- One of them must be moved--"Rebased"
- This process uses the .reloc section

Imports and Exports

- Imports
  - Functions the program needs to use from other code
  - Both EXE and DLL files have imports
  - The imports generally point to DLL's
- Exports
  - Functions this program offers for others to use
  - DLL's have many exports, EXE's don't

Notepad.exe Imports

PEview - C:\Users\Administrator\Desktop\notepad-win10.exe

File View Go Help

- notepad-win10.exe
  - IMAGE_DOS_HEADER
  - MS-DOS Stub Program
  - IMAGE_NT_HEADERS
  - IMAGE_SECTION_HEADER .text
  - IMAGE_SECTION_HEADER .data
  - IMAGE_SECTION_HEADER .idata
  - IMAGE_SECTION_HEADER .rsrc
  - IMAGE_SECTION_HEADER .reloc
  - SECTION .text
  - SECTION .data
  - SECTION .idata
    - IMPORT Address Table
    - IMPORT Directory Table
    - IMPORT Name Table
    - IMPORT Hints/Names & DLL Names
  - SECTION .rsrc
  - SECTION .reloc

pFile	Data	Description	Value
0001AE00	0001FBB0	Hint/Name RVA	0214 OpenProcessToken
0001AE04	0001FBC4	Hint/Name RVA	016F GetTokenInformation
0001AE08	0001FBDA	Hint/Name RVA	00EE DuplicateEncryptionInfoFile
0001AE0C	0001FBF8	Hint/Name RVA	02A6 RegSetValueExW
0001AE10	0001FC0A	Hint/Name RVA	0296 RegQueryValueExW
0001AE14	0001FC1E	Hint/Name RVA	0264 RegCreateKeyW
0001AE18	0001FC2E	Hint/Name RVA	0258 RegCloseKey
0001AE1C	0001FC3C	Hint/Name RVA	0289 RegOpenKeyExW
0001AE20	0001FC4C	Hint/Name RVA	0121 EventSetInformation
0001AE24	0001FC62	Hint/Name RVA	0120 EventRegister
0001AE28	0001FC72	Hint/Name RVA	0122 EventUnregister
0001AE2C	0001FC84	Hint/Name RVA	0128 EventWriteTransfer
0001AE30	0001FC9A	Hint/Name RVA	0197 IsTextUnicode
0001AE34	00000000	End of Imports	ADVAPI32.dll
0001AE38	80000159	Ordinal	0159
0001AE3C	00020D2A	Hint/Name RVA	000C CreateStatusWindowW
0001AE40	00000000	End of Imports	COMCTL32.dll
0001AE44	00020D9E	Hint/Name RVA	0008 FindTextW
0001AE48	00020DBA	Hint/Name RVA	0003 ChooseFontW
0001AE4C	00020D7A	Hint/Name RVA	000E GetSaveFileNameW

Viewing IMPORT Address Table

- Windows 10 Version

Advapi32.dll Exports

PEview - C:\Windows\System32\advapi32.dll

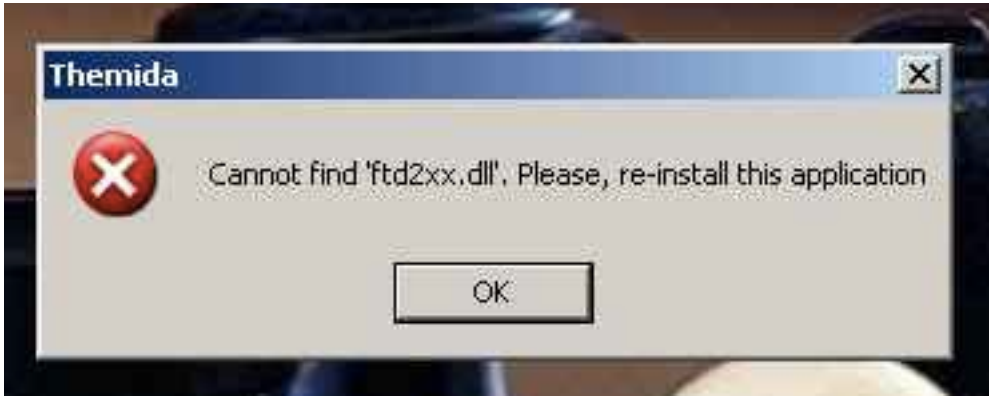
File View Go Help

- advapi32.dll
  - IMAGE_DOS_HEADER
  - MS-DOS Stub Program
  - IMAGE_NT_HEADERS
  - IMAGE_SECTION_HEADER .text
  - IMAGE_SECTION_HEADER .data
  - IMAGE_SECTION_HEADER .idata
  - IMAGE_SECTION_HEADER .didat
  - IMAGE_SECTION_HEADER .rsrc
  - IMAGE_SECTION_HEADER .reloc
  - SECTION .text
    - IMAGE_DEBUG_DIRECTORY
    - DELAY IMPORT DLL Names
    - IMAGE_LOAD_CONFIG_DIRECTORY
    - IMAGE_DEBUG_TYPE_CODEVIEW
    - IMAGE_DEBUG_TYPE_
    - DELAY IMPORT Descriptors
    - DELAY IMPORT Name Table
    - DELAY IMPORT Hints/Names
    - IMAGE_EXPORT_DIRECTORY
    - EXPORT Address Table
    - EXPORT Name Pointer Table
    - EXPORT Ordinal Table
    - EXPORT Names
  - SECTION .data
  - SECTION .idata
  - SECTION .didat
  - SECTION .rsrc
  - SECTION .reloc
  - CERTIFICATE Table

pFile	Data	Description	Value
000621A8	00038A70	Function RVA	03E8
000621AC	000475E0	Function RVA	03E9 I_ScGetCurrentGroupStateW
000621B0	00064EFE	Forwarded Name RVA	03EA A_SHAFinal -> NTDLL.A_SHAFinal
000621B4	00064F19	Forwarded Name RVA	03EB A_SHAInit -> NTDLL.A_SHAInit
000621B8	00064F35	Forwarded Name RVA	03EC A_SHAUpdate -> NTDLL.A_SHAUpdate
000621BC	00044070	Function RVA	03ED AbortSystemShutdownA
000621C0	000440E0	Function RVA	03EE AbortSystemShutdownW
000621C4	00030530	Function RVA	03EF AccessCheck
000621C8	0002DFC0	Function RVA	03F0 AccessCheckAndAuditAlarmA
000621CC	00030470	Function RVA	03F1 AccessCheckAndAuditAlarmW
000621D0	00030510	Function RVA	03F2 AccessCheckByType
000621D4	0002E0C0	Function RVA	03F3 AccessCheckByTypeAndAuditAlarmA
000621D8	00030490	Function RVA	03F4 AccessCheckByTypeAndAuditAlarmW
000621DC	000304F0	Function RVA	03F5 AccessCheckByTypeResultList
000621E0	0002E1D0	Function RVA	03F6 AccessCheckByTypeResultListAndAuditAl
000621E4	0002E2E0	Function RVA	03F7 AccessCheckByTypeResultListAndAuditAl
000621E8	000304B0	Function RVA	03F8 AccessCheckByTypeResultListAndAuditAl
000621EC	000304D0	Function RVA	03F9 AccessCheckByTypeResultListAndAuditAl
000621F0	0001E9B0	Function RVA	03FA AddAccessAllowedAce
000621F4	0001E940	Function RVA	03FB AddAccessAllowedAceEx
000621F8	00030550	Function RVA	03FC AddAccessAllowedObjectAce
000621FC	00030590	Function RVA	03FD AddAccessDeniedAce
00062200	00030570	Function RVA	03FE AddAccessDeniedAceEx
00062204	000305B0	Function RVA	03FF AddAccessDeniedObjectAce
00062208	000162A0	Function RVA	0400 AddAce
0006220C	000305F0	Function RVA	0401 AddAuditAccessAce
00062210	000305D0	Function RVA	0402 AddAuditAccessAceEx
00062214	00030610	Function RVA	0403 AddAuditAccessObjectAce
00062218	00047C90	Function RVA	0404 AddConditionalAce

Viewing EXPORT Address Table

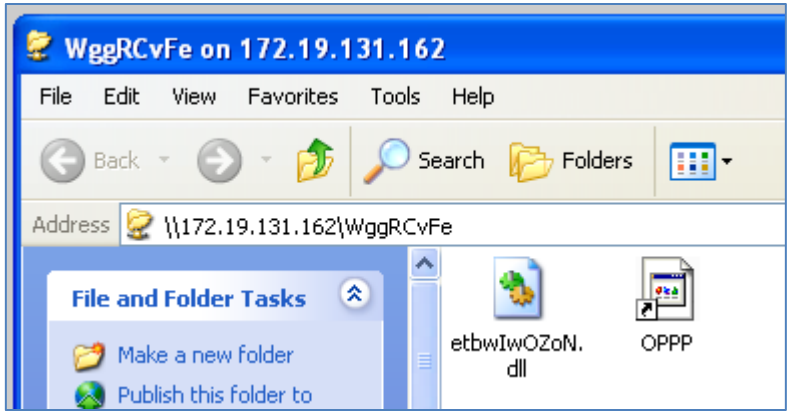
DLL Loading



- When an EXE launches, Windows hunts for the required DLLs
  - Looking first in the current working directory
- This allows a developer to include a DLL version other than the one in C:\Windows\System32
  - Leads to DLL Hell; users may need to adjust PATH to resolve DLL version conflicts

Stuxnet: LNK 0day

- Loaded a DLL from a USB thumbdrive
- Took over the machine as soon as the icons appear
  - Link Ch 6h



Relative Virtual Address (RVA)

- Windows EXE processes are loaded into 0x00400000 by default
  - This is a Virtual Address, only visible to each process
  - Error on page 113 of textbook, too many zeroes in 0x00400000
- RVA is used to aid in rebasing DLLs
  - Loading them in non-preferred locations

Example of VA (Virtual Address)

For example, a possible physical memory address (visible by the CPU):

```
0x00300000 on physical memory has process A's main
0x00500000 on physical memory has process B's main
```

And the OS may have a mapping table:

```
process A's 0x00400000 (VA) = physical address 0x00300000
process B's 0x00400000 (VA) = physical address 0x00500000
```

Then when you try to read 0x004000000 in process A, you'll get the content which is located on 0x00300000 of physical memory.

- Link Ch 6g

OllyDbg: Code Starts Near 0x400000

```
OllyDbg - Lab09-01.exe - [CPU - main thread, module Lab09-01]
File  View  Debug  Plugins  Options  Window  Help

00401000  $ 55            PUSH EBP
00401001  . 8BEC          MOV EBP,ESP
00401003  . 83EC 08       SUB ESP,8
00401006  . 8D45 F8       LEA EAX,DWORD PTR SS:[EBP-8]
00401009  . 50            PUSH EAX                          pHandle
0040100A  . 68 3F000F00   PUSH 0F003F                       Access = KEY_ALL_ACCESS
0040100F  . 6A 00         PUSH 0                            Reserved = 0
00401011  . 68 40C04000   PUSH Lab09-01.0040C040            Subkey = "SOFTWARE\Microsoft \XPS"
00401016  . 68 02000080   PUSH 80000002                     hKey = HKEY_LOCAL_MACHINE
0040101B  . FF15 20B04000 CALL DWORD PTR DS:[<&ADVAPI32.RegOpenKe  RegOpenKeyExA
00401021  . 85C0          TEST EAX,EAX
00401023  . 74 04         JE SHORT Lab09-01.00401029
```

Heaps

Many Heaps

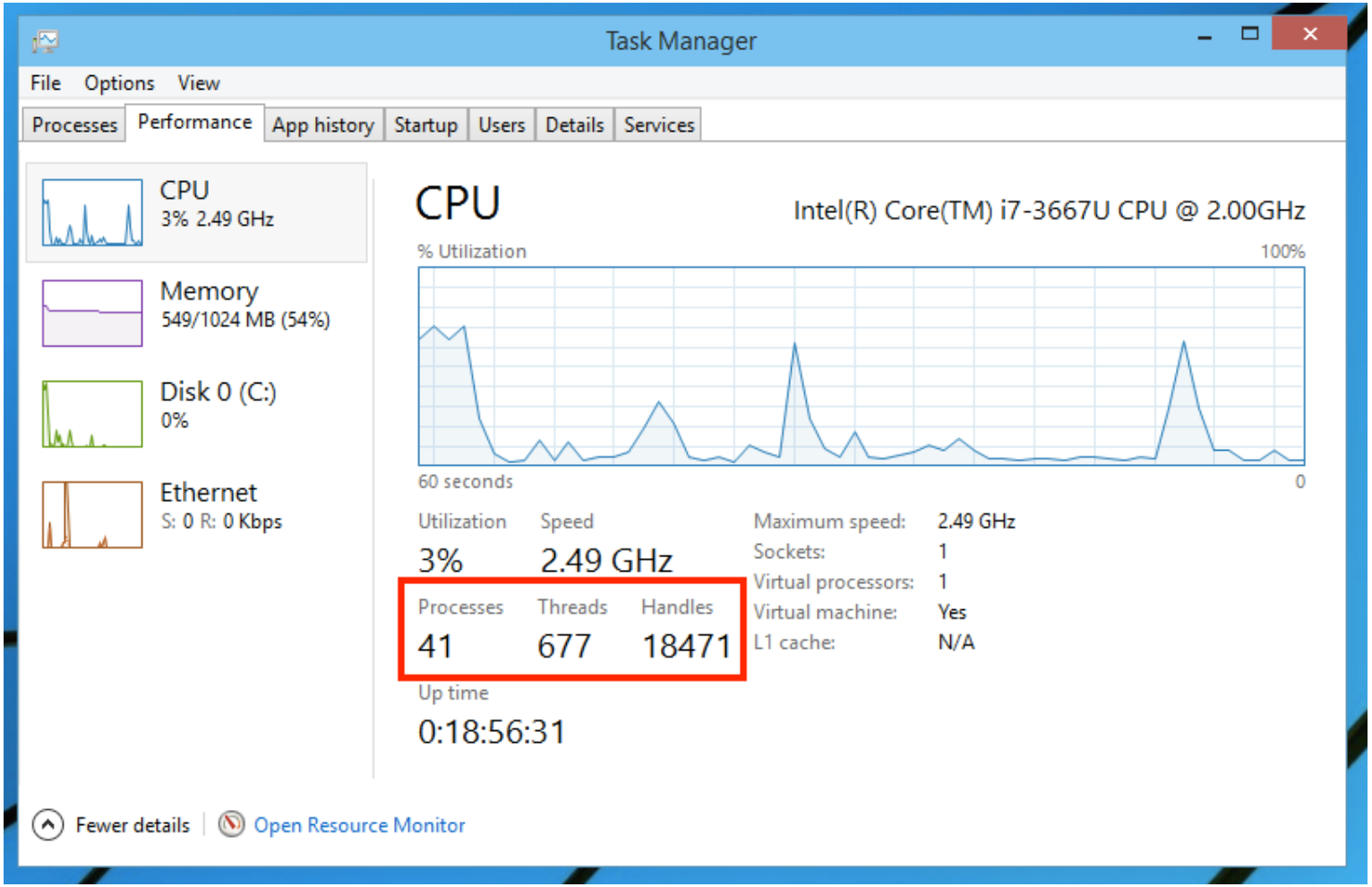
- Heap is used for temporary storage of data
  - Via malloc() and free()
- Linux uses one heap, but Windows uses many heaps
- Each DLL that loads can set up its own heap
- Heap corruption attacks are very confusing

Threading

One Process, Many Threads

- Each process is subdivided into **threads**
- Processor time slices are allocated to threads, not processes
- This allows a single process to operate more efficiently
  - If one thread is waiting for something, other threads can keep moving

Threads in Task Manager



Handles

- Handles are pointers to objects like open files
- Each thread has many handles
- You can view details about every thread with Process Explorer

Process Explorer - Sysinternals: www.sysinternals.com [WIN-8LDVLI8QDEN\sam]

File Options View Process Find

Process

- wininit.exe
  - services.exe
    - svchost.exe
      - ApplicationFrameHost...
      - dllhost.exe
    - svchost.exe
    - svchost.exe
      - taskhostex.exe
      - taskhostw.exe
    - svchost.exe
    - svchost.exe
      - audiodg.exe
    - svchost.exe
      - dasHost.exe
    - svchost.exe
    - spoolsv.exe
    - svchost.exe
    - MsMpEng.exe
    - vmtoolsd.exe
    - svchost.exe
    - svchost.exe
    - dllhost.exe
    - msdtc.exe
    - NisSrv.exe
    - SearchIndexer.exe
      - SearchProtocolHost.e...
      - SearchFilterHost.exe
    - armsvc.exe
    - svchost.exe
  - lsass.exe
- csrss.exe
- winlogon.exe
  - dwm.exe
- explorer.exe
  - vmtoolsd.exe
  - SkyDrive.exe
  - procexp.exe

vmtoolsd.exe	0.12	14,756 K	18,664 K	3960 VMware Tools Core Service	VMware, Inc.
SkyDrive.exe	0.08	4,412 K	3,976 K	4068 Microsoft OneDrive	Microsoft Corporation
procexp.exe	6.30	23,192 K	39,608 K	5740 Sysinternals Process Explorer	Sysinternals - www.sysinter...

explorer.exe:3128 Properties

Image | Performance | Performance Graph | GPU Graph | Threads | TCP/IP | Security | Environment | Strings

Count: 56

TID	CPU	Cycles Delta	Start Address
3216	0.08	1,997,600	SHCORE.dll!Ordinal250+0xb80
3292	< 0.01	62,138	twinui.appcore.dll!DllGetClassObject+0x4b3e0
2116			ntdll.dll!RtlSizeHeap+0x1640
3252			SHCORE.dll!Ordinal250+0xb80
3284			windows.immersiveshell.serviceprovider.dll!DllCanUnloadNow+0x16e0
3280			twinui.appcore.dll+0x24a70
3364			SHCORE.dll!Ordinal250+0xb80
3760			SHCORE.dll!Ordinal250+0xb80
5996			SHCORE.dll!Ordinal250+0xb80
3196			combase.dll!GetErrorInfo+0x110
3392			SHCORE.dll!Ordinal250+0xb80
3524			ntdll.dll!RtlSizeHeap+0x1640
2212			ntdll.dll!RtlSizeHeap+0x1640
308			ntdll.dll!RtlSizeHeap+0x1640
6072			ntdll.dll!RtlSizeHeap+0x1640
4264			ntdll.dll!RtlSizeHeap+0x1640

Stack | Module

Thread ID:	3132		
Start Time:	1:54:11 PM 12/12/2014		
State:	Wait:WrUserRequest	Base Priority:	8
Kernel Time:	0:00:01.984	Dynamic Priority:	10
User Time:	0:00:01.718	I/O Priority:	Normal
Context Switches:	18,178	Memory Priority:	5
Cycles:	9,729,467,628	Ideal Processor:	0

Permissions | Kill | Suspend

OK | Cancel

The Genius and Idiocy of the **DCOM** (Distributed Common Object Model) and **DCE-RPC** (Distributed Computing Environment / Remote Procedure Calls)

Follow the Money

- Microsoft's business model is to distribute binary packages for money
- You can build a complex application by purchasing third-party COM modules from vendors
  - And tying them together with Visual Basic

COM Objects

- Can be written in any supported language
- Interoperate seamlessly
- BUT a C++ integer is not the same as a Visual Basic integer
- So you need to define the input and outputs with an IDL (Interface Description Language) file

DCOM Interface Description Language (IDL) File

```
[ uuid(e33c0cc4-0482-101a-bc0c-02608c6ba218),
  version(1.0),
  implicit_handle(handle_t rpc_binding)
] interface ???
{
  typedef struct {
    TYPE_2 element_1;
    TYPE_3 element_2;
  } TYPE_1;
...

  short Function_00(
        [in] long element_9,
        [in] [unique] [string] wchar_t *element_10,
        [in] [unique] TYPE_1 *element_11,
        [in] [unique] TYPE_1 *element_12,
        [in] [unique] TYPE_2 *element_13,
        [in] long element_14,
        [in] long element_15,
       [out] [context_handle] void *element_16
  );
```

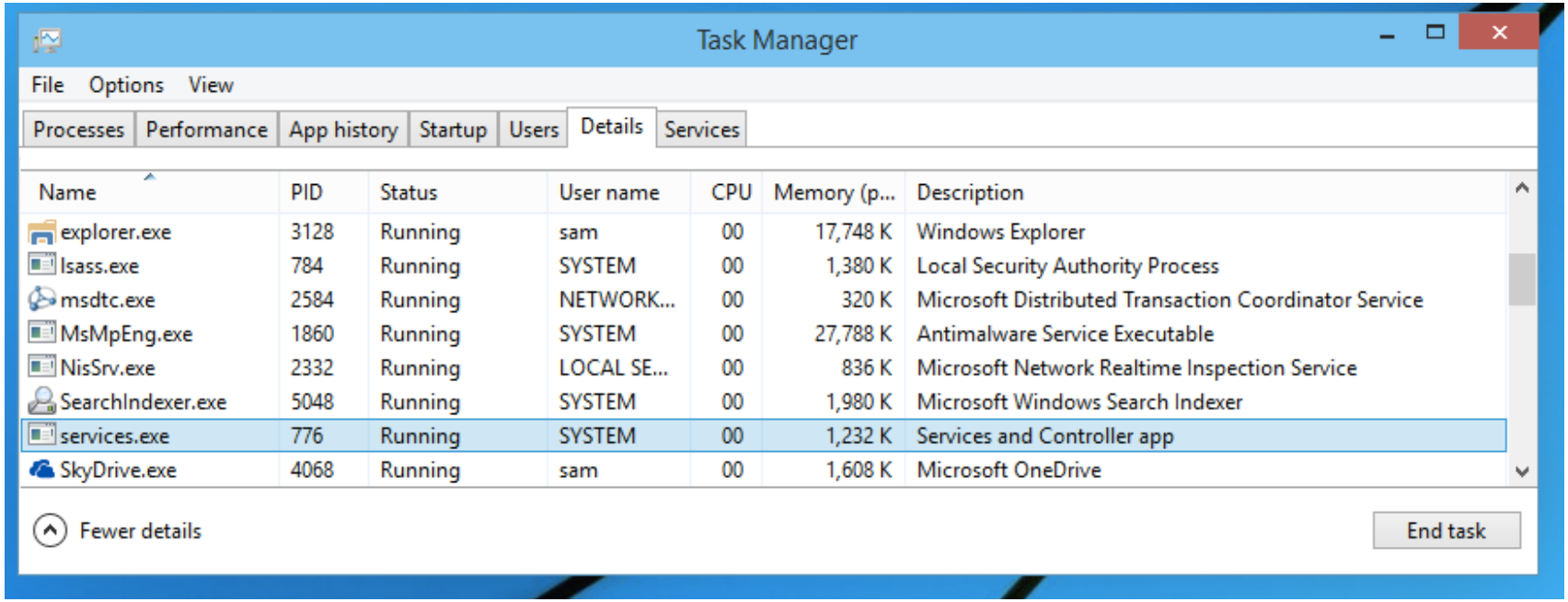
DCOM IDL File

- Specifies arguments and return values for a particular function
  - In a particular interface defined by UUID, also called a GUID
  - GUID is 128 bits long; 32 hex characters

Two Ways to Load a COM Object

- Load directly into process space as a DLL
- Launch as a service
  - By the Service Control Manager (services.exe)
- Running as a service is more stable and secure
  - But much slower
- In-process calls are 1000 times faster than calling a COM interface on the same machine but in a different process

Service Control Manager (SCM)



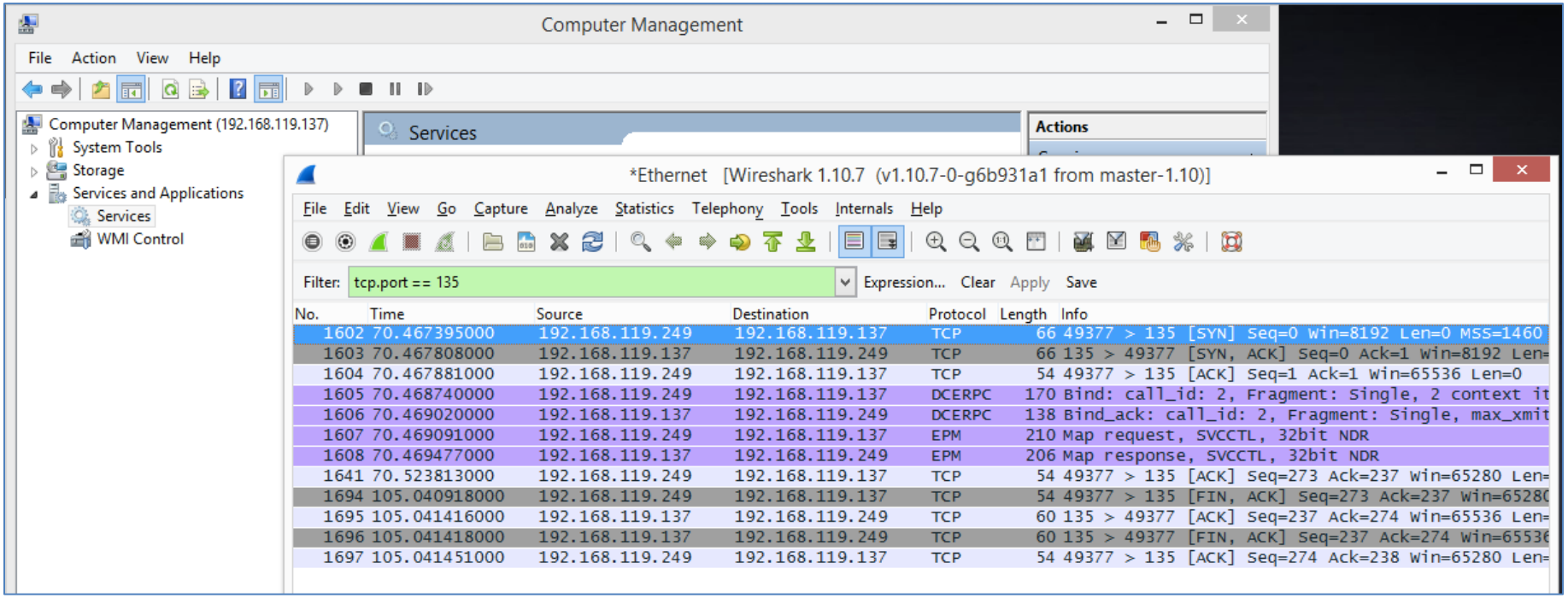
- Appears in Task Manager as services.exe

DCOM Calls

- Microsoft's priority: make it easy for developers to write software
- A simple registry or parameter change tells a program to use a different process
  - Or even a different machine
- A process can call a COM interface on a different machine on the LAN
  - 10x slower than calling a COM interface on the same machine

RPC Endpoint Mapper

- Listening on port TCP 135
  - An RPC request in Wireshark



Maps to UUID Values

- Map request shows available RPC functions
  - Link Ch 6m for details

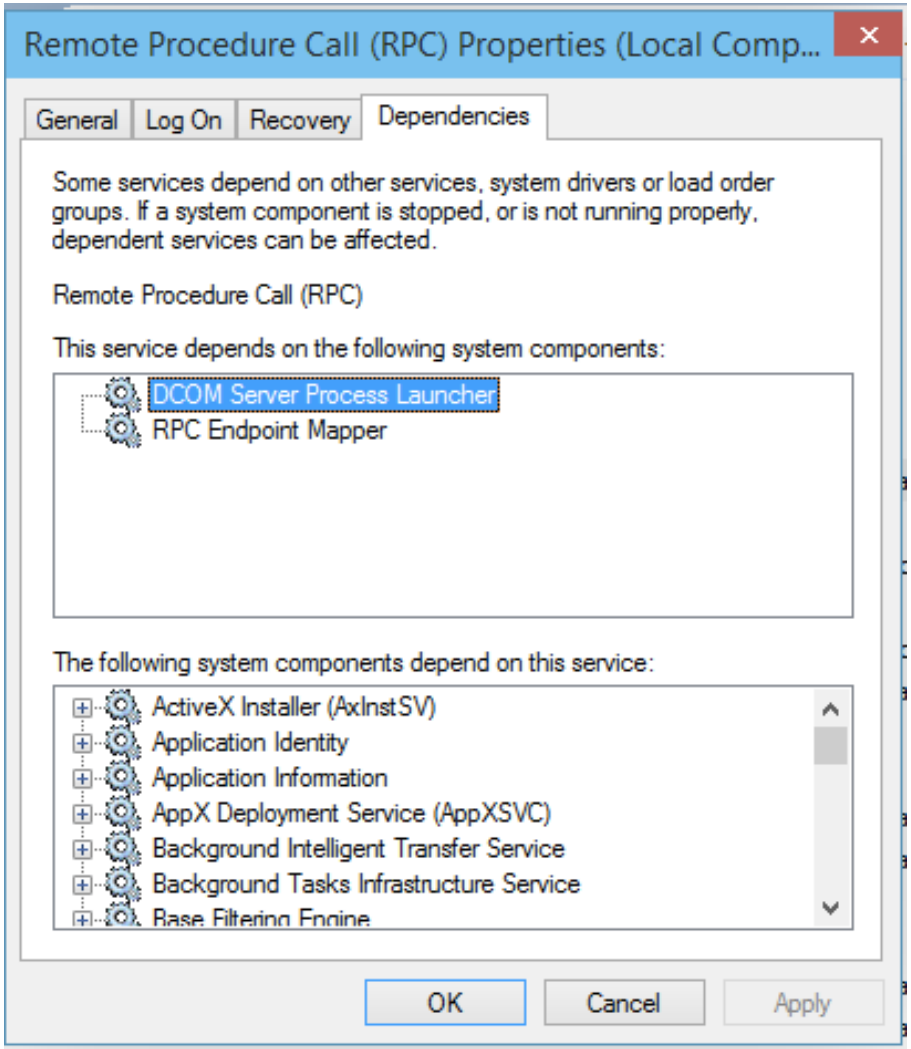
Filter: tcp.port == 135 | Expression... | Clear | Apply | Save

No.	Time	Source	Destination	Protocol	Length	Info
1605	70.468740000	192.168.119.249	192.168.119.137	DCERPC	170	Bind: call_id: 2, Fragment: Single, 2 context
1606	70.469020000	192.168.119.137	192.168.119.249	DCERPC	138	Bind_ack: call_id: 2, Fragment: Single, max_xr
1607	70.469091000	192.168.119.249	192.168.119.137	EPM	210	Map request, SVCCTL, 32bit NDR
1608	70.469477000	192.168.119.137	192.168.119.249	EPM	206	Map response, SVCCTL, 32bit NDR
1641	70.523813000	192.168.119.249	192.168.119.137	TCP	54	49377 > 135 [ACK] Seq=273 Ack=237 Win=65280 Le

```
    Number of floors: 5
  ⊟ Floor 1 UUID: SVCCTL
      LHS Length: 19
      Protocol: UUID (0x0d)
      UUID: SVCCTL (367abb81-9844-35f1-ad32-98f038001003)
      Version 2.0
      RHS Length: 2
      Version Minor: 0
  ⊟ Floor 2 UUID: 32bit NDR
      LHS Length: 19
      Protocol: UUID (0x0d)
      UUID: 32bit NDR (8a885d04-1ceb-11c9-9fe8-08002b104860)
      Version 2.0
      RHS Length: 2
      Version Minor: 0
```

Components that Depend on RPC

- Open Services
- Double-click "Remote Procedure Call"



Security Implications

- Code can be designed to run in a trusted environment
  - Calling DLLs that are included in your application, or Microsoft DLLs
- And easily adapted to run in an untrusted environment
  - Listening on a network port

DEC-RPC Exploitation

- Recon, fuzz, and exploit with Dave Aitel's SPIKE and other tools

www.immunitysec.com/resources/

SPIKE

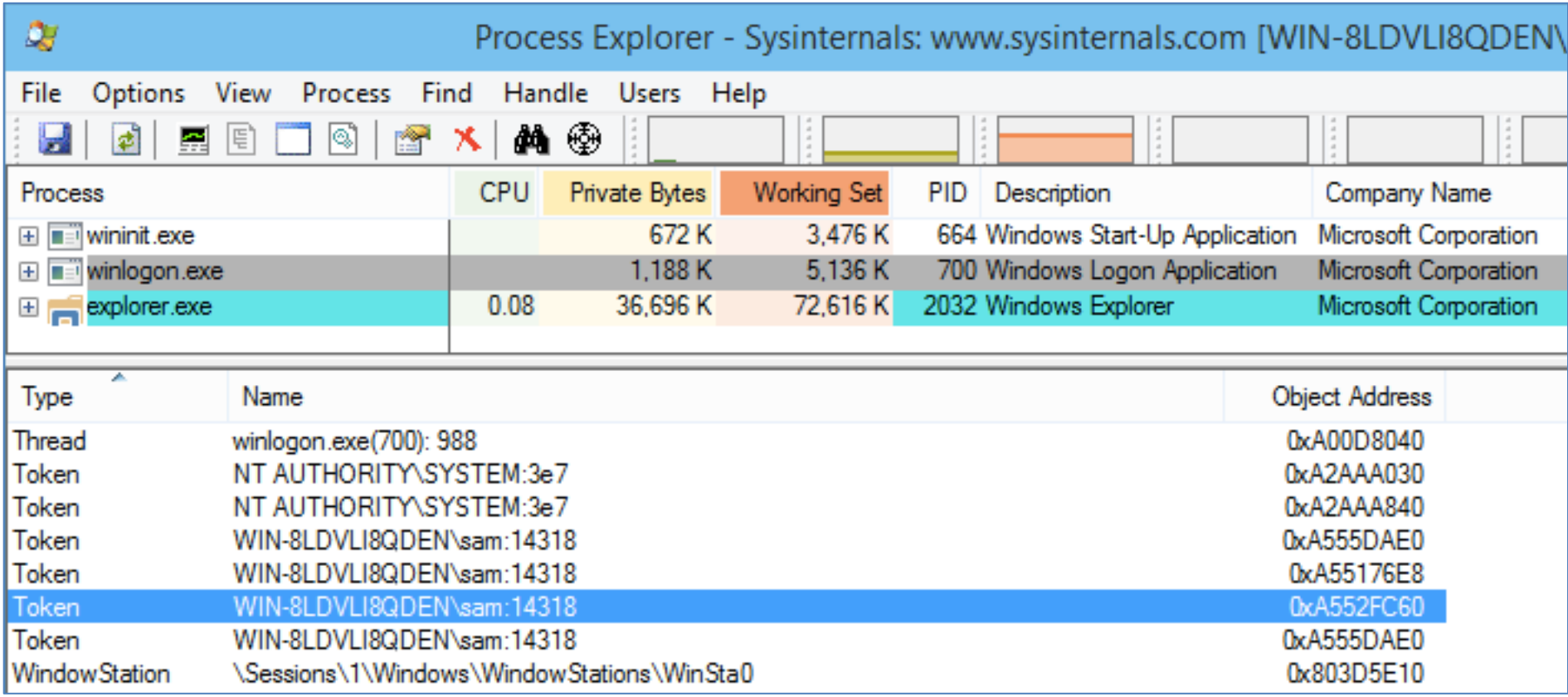
When you need to analyze a new network protocol for buffer overflows c
the SPIKE is the tool of choice for professionals. While it requires a stro
use, it produces results second to none in the field. SPIKE is available
only.

- Download Current Tarball
- Papers on SPIKE

Tokens and Impersonation

Token

- A token is a 32-bit integer like a file handle
- Defines user rights



Exploiting Token Handling

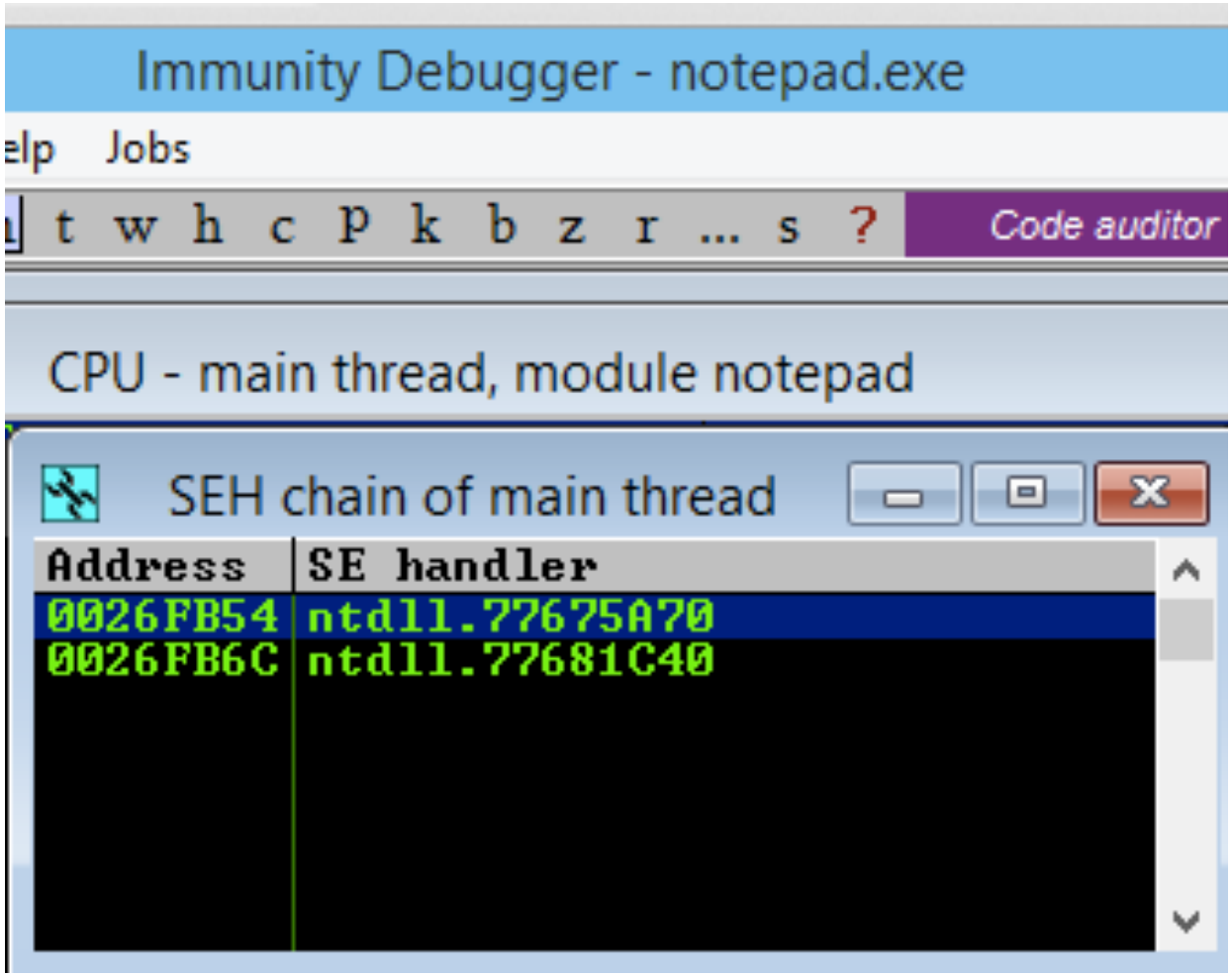
- Attacker can create threads and copy any available token to them
- There are typically tokens available for any user that has recently authenticated

Exception Handling

Structured Exception Handler (SEH)

- When an illegal operation occurs, such as
  - Divide by zero
  - Attempt to execute non-executable memory
  - Attempt to use invalid memory location
- The processor sends an **Exception**
- The OS can handle it, with an error message or a Blue Screen of Death
- But the application can specify custom **exception handlers**

SEH in Immunity Debugger



Exploiting the SEH

- Overwrite the pointer to the SEH chain
- Overwrite the function pointer for the handler on the stack
- Overwrite the default exception handler

Debuggers

Three Options

- SoftICE
  - Old, powerful, difficult to install
- WinDbg
  - Use by Microsoft
  - Can debug the kernel, using a serial cable and two computers
    - Or Ethernet, for Win 8 or later
    - Or LiveKD and one machine
  - UI is terrible
- OllyDbg
  - Very popular but apparently abandoned

OllyDbg

- OllyDbg version 1.10 is very nice
- OllyDbg 2.x is terrible, giving false results, and useless
- No later version seems to be available

Immunity Debugger

Immunity Debugger

- Based on OllyDbg
- Still alive and under development
- Used by many exploit developers

Immunity Debugger