

Digital Forensics

Digital Acquisition

and Rapid Triage





GCFE

INCIDENT RESPONSE & THREAT HUNTING

SANS DFIR CURRICULUM

(w) @SANSForensics (b) dfir.to/DFIRCast (c) dfir.to/LinkedIn

DIGITAL FORENSICS

Windows Forensic Analysis



GIME

Enterprise

Memory Forensics

In-Depth

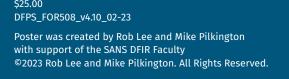
Malware: Advanced

Code Analysis



Mac and iOS Forensic Smartphone Forensic Analysis & Incident Response Analysis In-Depth GASF

Hunt Evil



dfir.sans.org

Process Hacker

Hacker View Tools Users Help

Advanced Incident Response, Threat Hunting & **Digital Forensics**



Cyber Threat Intelligence



Enterprise Cloud Forensics & Incident Response



Enterprise-Class Incident Response & Threat Hunting



for Incident Responders



Reverse-Engineering REM: Malware Analysis Tools & Techniques



Advanced Network Forensics: Threat Hunting, Analysis & **Incident Response**



Hacker Tools, Techniques & Incident Handling **GCIH**

Find Evil – Know Normal

Knowing what's normal on a Windows host helps cut through the noise to quickly locate potential malware. Use the information below as a reference to know what's normal in Windows and to focus your attention on the outliers.

System

Image Path: N/A for system.exe - Not generated from an executable image

Parent Process: None Number of Instances: One **User Account:** Local System **Start Time:** At boot time

Description: The **System** process is responsible for most kernel-mode threads. Modules run under **System** are primarily drivers (.sys files), but also include several important DLLs as well as the kernel executable, ntoskrnl.exe.

smss.exe

Image Path: %SystemRoot%\System32\smss.exe

Parent Process: System

Number of Instances: One master instance and another child instance per session. Children exit after creating their session.

User Account: Local System

Start Time: Within seconds of boot time for the master instance

Description: The Session Manager process is responsible for creating new sessions. The first instance creates a child instance for each new session. Once the child instance initializes the new session by starting the Windows subsystem (csrss.exe) and wininit.exe for Session 0 or winlogon.exe for Session 1 and higher, the child instance exits.

wininit.exe

Image Path: %SystemRoot%\System32\wininit.exe

Parent Process: Created by an instance of smss.exe that exits, so tools usually do not provide the parent process name.

Number of Instances: One

User Account: Local System **Start Time:** Within seconds of boot time

Description: Wininit.exe starts key background processes within Session 0. It starts the Service Control Manager (services.exe), the Local Security Authority process (lsass.exe), and lsaiso.exe for systems with Credential Guard enabled. Note that prior to Windows 10, the Local Session Manager process (1sm.exe) was also started by wininit.exe. As of Windows 10, that

functionality has moved to a service DLL (lsm.dll) hosted by sychost.exe.

RuntimeBroker.exe

Image Path: %SystemRoot%\System32\RuntimeBroker.exe

Parent Process: svchost.exe

Number of Instances: One or more

User Account: Typically the logged-on user(s)

Start Time: Start times vary greatly

Description: RuntimeBroker.exe acts as a proxy between the constrained Universal Windows Platform (UWP) apps (formerly called Metro apps) and the full Windows API. UWP apps have limited capability to interface with hardware and the file system. Broker processes such as RuntimeBroker.exe are therefore used to provide the necessary level of access for UWP apps. Generally, there will be one RuntimeBroker.exe for each UWP

app. For example, starting Calculator.exe will cause a corresponding RuntimeBroker.exe process to initiate.

taskhostw.exe

Image Path: %SystemRoot%\System32\taskhostw.exe Parent Process: svchost.exe

Number of Instances: One or more

User Account: Multiple taskhostw.exe processes are normal. One or more may be owned by logged-on users and/or by local service accounts. **Start Time:** Start times vary greatly

Description: The generic host process for Windows Tasks. Upon initialization,

taskhostw.exe runs a continuous loop listening for trigger events. Example trigger events that can initiate a task include a defined schedule, user logon, system startup, idle CPU time, a Windows log event, workstation lock, or workstation unlock.

There are more than 160 tasks preconfigured on a default installation of Windows 10 Enterprise (though many are disabled). All executable files (DLLs & EXEs) used by the default Windows 10 scheduled tasks are signed by Microsoft.

winlogon.exe

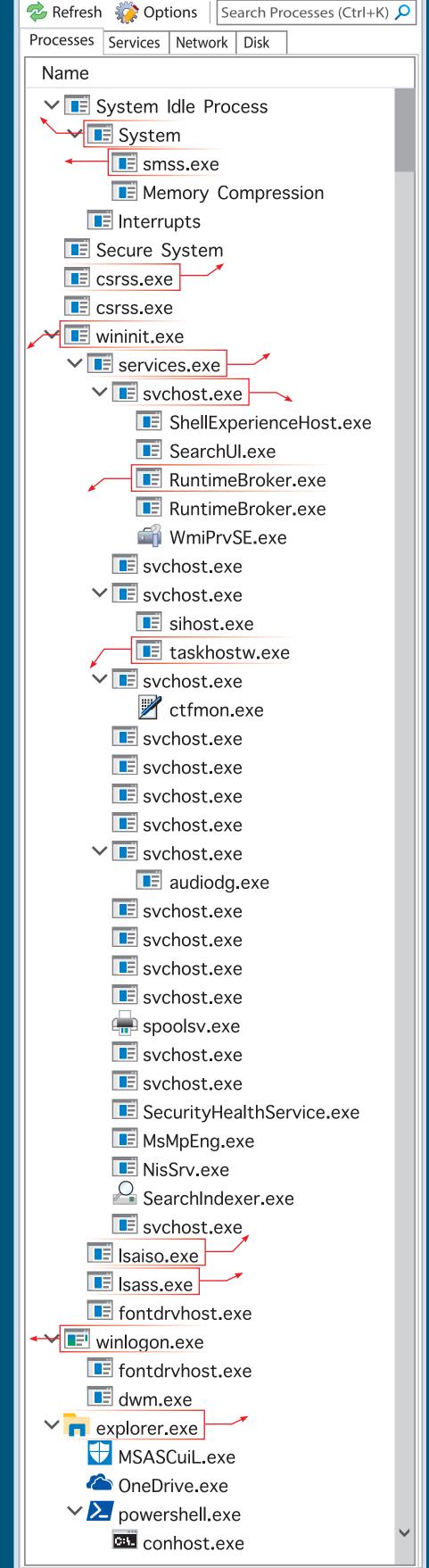
Image Path: %SystemRoot%\System32\winlogon.exe

Parent Process: Created by an instance of smss.exe that exits, so analysis tools usually do not provide the parent process name.

Number of Instances: One or more **User Account:** Local System

Start Time: Within seconds of boot time for the first instance (for Session 1). Start times for additional instances occur as new sessions are created, typically through Remote Desktop or Fast User Switching logons.

Description: Winlogon handles interactive user logons and logoffs. It launches **LogonUI.exe**, which uses a credential provider to gather credentials from the user, and then passes the credentials to **lsass.exe** for validation. Once the user is authenticated, Winlogon loads the user's **NTUSER.DAT** into **HKCU** and starts the user's shell (usually explorer.exe) via userinit.exe.



csrss.exe

Image Path: %SystemRoot%\System32\csrss.exe

Parent Process: Created by an instance of smss.exe that exits, so analysis tools usually do not provide the parent process name.

Number of Instances: Two or more

User Account: Local System

Start Time: Within seconds of boot time for the first two instances (for Session 0 and 1). Start times for additional instances occur as new sessions are created, although often only Sessions 0 and 1 are created.

Description: The Client/Server Run-Time Subsystem is the user-mode process for the Windows subsystem. Its duties include managing processes and threads, importing many of the DLLs that provide the Windows API, and facilitating shutdown of the GUI during system shutdown. An instance of csrss.exe will run for each session. Session 0 is for services and Session 1 for the local console session. Additional sessions are created through the use of Remote Desktop and/or Fast User Switching. Each new session results in a new instance of csrss.exe.

services.exe

Image Path: %SystemRoot%\System32\services.exe

Parent Process: wininit.exe **Number of Instances:** One

User Account: Local System

Start Time: Within seconds of boot time

Description: Implements the Unified Background Process Manager (UBPM), which is responsible for background activities such as services and scheduled tasks. Services.exe also implements the Service Control Manager (SCM), which specifically handles the loading of services and device drivers marked for auto-start. In addition, once a user has successfully logged on interactively, the SCM (services.exe) considers the boot successful and sets the Last Known Good control set (HKLM\SYSTEM\Select\LastKnownGood) to the value of the CurrentControlSet.

svchost.exe

Image Path: %SystemRoot%\system32\svchost.exe

Parent Process: services.exe (most often)

Number of Instances: Many (generally at least 10)

User Account: Varies depending on **svchost** instance, though it typically will be Local System, Network Service, or Local Service accounts. Windows 10 also has some instances running as logged-on users.

Start Time: Typically within seconds of boot time. However, services can be started after boot (e.g., at logon), which

results in new instances of **svchost.exe** after boot time.

Description: Generic host process for Windows services. It is used for running service DLLs. Windows will run multiple instances of svchost.exe, each using a unique "-k" parameter for grouping similar services. Typical "-k" parameters include DcomLaunch, RPCSS, LocalServiceNetworkRestricted, LocalServiceNoNetwork, LocalServiceAndNoImpersonation, netsvcs, NetworkService, and more. Malware authors often take advantage of the ubiquitous nature of sychost.exe and use it either to host a malicious DLL as a service, or run a malicious process named svchost.exe or similar spelling. Beginning in Windows 10 version 1703, Microsoft changed the default grouping of similar services if the system has more than 3.5 GB of RAM. In such cases, most services will run under their own instance of sychost.exe. On systems with more than 3.5 GB RAM, expect to see more than 50 instances of svchost.exe (the screenshot in the poster is a Windows 10 VM with 3 GB RAM).

Isaiso.exe

Image Path: %SystemRoot%\System32\lsaiso.exe

Parent Process: wininit.exe

Number of Instances: Zero or one

User Account: Local System

Start Time: Within seconds of boot time

Description: When Credential Guard is enabled, the functionality of **lsass.exe** is split between two processes – itself and <code>lsaiso.exe</code>. Most of the functionality stays within <code>lsass.exe</code>, but the important role of safely storing account credentials moves to <code>lsaiso.exe</code>. It provides safe storage by running in a context that is isolated from other processes through hardware virtualization technology. When remote authentication is required, lsass.exe proxies the requests using an RPC channel with <code>lsaiso.exe</code> in order to authenticate the user to the remote service. Note that if Credential Guard is not enabled, lsaiso.exe should not be running on the system.

Isass.exe

Image Path: %SystemRoot%\System32\lsass.exe Parent Process: wininit.exe

Number of Instances: One

User Account: Local System

Start Time: Within seconds of boot time

Description: The Local Security Authentication Subsystem Service process is responsible for authenticating users by calling an appropriate authentication package specified in HKLM\SYSTEM\CurrentControlSet\Control\Lsa. Typically, this will be Kerberos for domain accounts or MSV1 0 for local accounts. In addition to authenticating users, **1sass.exe** is also responsible for implementing the local security policy (such as password policies and audit policies) and for writing events to the security event log. Only one instance of this process should occur and it should rarely have child processes (EFS is a known exception).



Image Path: %SystemRoot%\explorer.exe

Parent Process: Created by an instance of userinit.exe that exits, so analysis tools usually do not provide the parent process name.

Number of Instances: One or more per interactively logged-on user

User Account: <logged-on user(s)>

Start Time: First instance starts when the owner's interactive logon begins

Description: At its core, Explorer provides users access to files. Functionally, though, it is both a file browser via Windows Explorer (though still explorer.exe) and a user interface providing features such as the user's Desktop, the Start Menu, the Taskbar, the Control Panel, and application launching via file extension associations and shortcut files. **Explorer.exe** is the default user interface specified in the Registry value **HKLM\SOFTWARE** Microsoft\Windows NT\CurrentVersion\Winlogon\Shell, though Windows can alternatively function with another interface such as cmd.exe or powershell.exe. Notice that the legitimate explorer.exe resides in the **SystemRoot** directory rather than **SystemRoot** System32. Multiple instances per user can occur, such as when the option "Launch folder windows in a separate process" is enabled.

Process listing from Windows 10 Enterprise

CPU Usage: 4.50% | Physical Memory: 20.67% | Processes: 125

DFPS_FOR508_v4.10_02-23.indd

Hunt Evil: Lateral Movement

During incident response and threat hunting, it is critical to understand how attackers move around your network. Lateral movement is an inescapable requirement for attackers to stealthily move from system to system and accomplish their objectives. Every adversary, including the most skilled, will use some form of lateral movement technique described here during a breach. Understanding lateral movement tools and techniques allows responders to hunt more efficiently, quickly perform incident response scoping, and better anticipate future attacker activity. Tools and techniques to hunt the artifacts described below are detailed in the SANS DFIR course FOR508: Advanced Digital Forensics, Incident Response, and Threat Hunting

Additional Event Logs

EVENT LOGS

• 4648 - Logon specifying alternate

Current logged-on User Name

RDPClient%4Operational.evtx

- Destination Host Name/IP

- Alternate User Name

- Process Name

■Microsoft-Windows-

TerminalServices-

credentials - if NLA enabled on

security.evtx

destination

• 1102

Process-tracking events, Sysmon, and similar logging capabilities are not listed here for the sake of brevity. However, this type of enhanced logging can provide significant visibility of an intruder's lateral movement, given that the logs are not overwritten or otherwise deleted.

Additional FileSystem Artifacts

Deep-dive analysis techniques such as file carving, volume shadow analysis, and NTFS log file analysis can be instrumental in recovering many of these artifacts (including the recovery of registry and event log files and records).

Additional References

SANS DFIR FOR508 course: http://sans.org/FOR508 ATT&CK Lateral Movement: http://for508.com/attck-lm JPCERT Lateral Movement: http://for508.com/jpcert-lm

Artifacts in Memory Analysis

Artifacts in memory analysis will allow for additional tracking of potential evidence of execution and command line history. We recommend auditing and dumping the "conhost" processes on the various systems. Example: vol.py -f memory.img --profile=<profile> memdump -n conhost --dump-dir=. strings -t d -e l *.dmp >> conhost.uni

DESTINATION

Perform searches for executable keywords using grep. Also check running processes (mstsc, rdpclip, etc.).

SOURCE

REGISTRY FILE SYSTEM ■ Remote desktop destinations ■ UserAssist - **NTUSER.DAT** ■ Jumplists - C: \Users\<Username>\ AppData\Roaming\Microsoft\Windows\ •mstsc.exe Remote Desktop Client execution Recent\AutomaticDestinations\ • Last Time Executed • {MSTSC-APPID}automaticDestinations-ms Server Client\Servers Number of Times Executed

- Tracks remote desktop connection ■ RecentApps - **NTUSER.DAT** destination and times •mstsc.exe Remote ■ Prefetch - C:\Windows\Prefetch\ Desktop Client execution •mstsc.exe-{hash}.pf Last Time Executed
- Number of Times Executed ■ Bitmap Cache - C:\users\<username>\ • RecentItems subkey tracks AppData\Local\Microsoft\Terminal Server Client\Cache connection destinations and •bcache##.bmc •cache###.bin

FILE SYSTEM

■ Prefetch - C:\Windows\Prefetch\

• Review shortcut files and jumplists for

remote files accessed by attackers, if they

•net.exe-{hash}.pf

■ User Profile Artifacts

•net1.exe-{hash}.pf

had interactive access (RDP)

Remote Desktop

REMOTE ACCESS

Security Event Log security.evtx • **4624** Logon Type 10 - Source IP/Logon User Name • 4778/4779

- IP Address of Source/Source System Name - Logon User Name Microsoft-Windows-RemoteDesktopServices-RdpCoreTS%4Operational.evtx
- **131** Connection Attempts - Source IP • 98 - Successful Connections

Security Event Log -

EVENT LOGS REGISTRY ■ ShimCache - **SYSTEM** ■Microsoft-Windows-Terminal Services-RemoteConnection •rdpclip.exe Manager%4Operational.evtx • tstheme.exe

• 1149 - Source IP/Logon User Name Blank user name may indicate use of Sticky Keys ■Microsoft-Windows-Terminal

Services-LocalSession Manager%4Operational.evtx • 21, 22, 25 - Source IP/Logon User Name

■ AmCache . hve

First Time Executed

REGISTRY

REGISTRY

■ New service creation configured in SYSTEM\

CurrentControlSet\

Services\PSEXESVC

• "-r" option can allow

■ ShimCache – **SYSTEM**

•psexesvc.exe

First Time Executed

•psexesvc.exe

■ AmCache.hve

attacker to rename service

•rdpclip.exe

• tstheme.exe

■ Prefetch - C:\Windows\Prefetch\ •rdpclip.exe-{hash}.pf • tstheme.exe-{hash}.pf

FILE SYSTEM

EVENT LOGS

- Destination Host Name

Destination IP Address

- security.evtx • 4648 – Logon specifying
- Current logged-on User Name - Alternate User Name Destination Host Name/IP Process Name
- Microsoft-Windows-• **31001** – Failed logon to destination

Reason code for failed

password)

destination logon (e.g. bad

- alternate credentials
- SmbClient%4Security.evtx Destination Host Name User Name for failed logon
- **REGISTRY** ■ MountPoints2 - Remotely mapped shares
- NTUSER\Software\Microsoft\Windows\ CurrentVersion\Explorer\MountPoints2 ■ Shellbags - USRCLASS.DAT
- Remote folders accessed inside an interactive session via Explorer by attackers
- ShimCache SYSTEM •net.exe •net1.exe

are tracked per-user

■ ShimCache – **SYSTEM**

Desktop Client

Desktop Client

Time Executed

Executed

•mstsc.exe

•mstsc.exe Remote

•mstsc.exe Remote

■ BAM/DAM - **SYSTEM** - Last

■ AmCache.hve - First Time

• NTUSER\Software\

Microsoft\Terminal

- BAM/DAM NTUSER.DAT Last Time Executed •net.exe •net1.exe
- AmCache.hve First Time Executed •net1.exe

net use z: \\host\c\$ /user:domain\username <password>

Map Network Shares (net.exe) to C\$ or Admin\$



and ADMINS

default shares such as C\$ to Local System

EVENT LOGS

• 41

- security.evtx • **4624** Logon Type 3 Name Source IP/Logon User Name
- 4672 Logon User Name Logon by user with administrative rights Requirement for accessing
- 4776 NTLM if authenticating Source Host Name/Logon **User Name**

security.evtx

Connecting User Name

- Logon User Name

• 5140 – Share Access

system.evtx

Service Install

7045

- Source IP/Logon User Name

- ADMIN\$ share used by PsExec

• 4648 Logon specifying alternate credentials

Logon by a user with administrative rights

• **4768** – TGT Granted - Source Host Name/Logon User

 Available only on domain controller • 4769 – Service Ticket Granted if

- Logon User Name

authenticating to Domain Controller - Destination Host Name/Logon User Name - Source IP

EVENT LOGS

4624 Logon Type 3 (and Type 2 if "-u" Alternate Credentials are used)

Requirement for access default shares such as C\$ and ADMIN\$

EVENT LOGS

- Available only on domain controller • 5140 Share Access

• 5145 - Auditing of shared files - NOISY!

FILE SYSTEM ■ File Creation

- Attacker's files (malware) copied to destination system • Look for Modified Time before Creation Time
- Creation Time is time of file copy

FILE SYSTEM

■ Prefetch - C:\Windows\Prefetch\

•psexesvc.exe-{hash}.pf

User profile directory structure

created unless "-e" option used

• psexesvc.exe will be placed in

ADMIN\$ (\Windows) by default, as

well as other executables (evil.exe)

• evil.exe-{hash}.pf

pushed by PsExec

■ File Creation

SOURCE

REGISTRY

REGISTRY

REGISTRY

■ AmCache.hve

schtasks.exe

at.exe

First Time Executed

REMOTE EXECUTION

DESTINATION

EVENT LOGS security.evtx

- 4648 Logon specifying
- Alternate User Name Destination Host Name/IP

Process Name

security.evtx

credentials

alternate credentials Current logged-on User Name

EVENT LOGS

• 4648 – Logon specifying alternate

- Current logged-on User Name

Destination Host Name/IP

- Alternate User Name

Process Name

- REGISTRY • Software\SysInternals\PsExec\EulaAccepted
- ShimCache **SYSTEM** • psexec.exe

• psexec.exe

■ ShimCache - **SYSTEM**

■ BAM/DAM - **SYSTEM** - Last

schtasks.exe

schtasks.exe

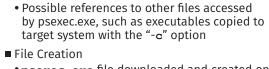
Time Executed

•at.exe

■ NTUSER. DAT

■ BAM/DAM - **SYSTEM** - Last Time Executed • psexec.exe ■ AmCache.hve - First Time Executed

FILE SYSTEM ■ Prefetch - C:\Windows\Prefetch\ •psexec.exe-{hash}.pf



• psexec.exe file downloaded and created on local host as the file is not native to Windows

FILE SYSTEM

■ Prefetch - C:\Windows\Prefetch\

• schtasks.exe-{hash}.pf

•at.exe-{hash}.pf

psexec.exe psexesvc.exe

psexec.exe \\host -accepteula -d -c c:\temp\evil.exe

Scheduled Tasks

at \\host 13:00 "c:\temp\evil.exe" schtasks /CREATE /TN taskname /TR c:\temp\evil.exe /SC once /RU "SYSTEM" /ST 13:00 /S host /U username

security.evtx 4624 Logon Type 3 - Source IP/Logon User Name 4672

- Logon User Name - Logon by a user with administrative rights - Requirement for accessing default shares such as C\$ and

• 4699 - Scheduled task deleted • 4700/4701 - Scheduled task enabled/disabled

■Microsoft-Windows-Task • 200/201 - Scheduled task

executed/completed

• 4698 - Scheduled task created ■ SOFTWARE • 4702 - Scheduled task updated •Microsoft\Windows

Scheduler%4Operational.evtx ■ ShimCache - **SYSTEM** • **106** – Scheduled task created •evil.exe • **140** – Scheduled task updated • 141 - Scheduled task deleted ■ AmCache.hve -First Time Executed

REGISTRY FILE SYSTEM ■ File Creation

- •evil.exe NT\CurrentVersion\ Job files created in Schedule\TaskCache\Tasks C:\Windows\Tasks •Microsoft\Windows XML task files created in NT\CurrentVersion\ Schedule\TaskCache\Tree\
 - C:\Windows\System32\Tasks - Author tag under "RegistrationInfo" can identify: Source system name
 - Creator username ■ Prefetch - C:\Windows\Prefetch\ •evil.exe-{hash}.pf

FILE SYSTEM

• evil.exe or evil.dll malicious

FILE SYSTEM

• evil.mof - .mof files can be used

■ Prefetch - C:\Windows\Prefetch\

to manage the WMI Repository

•scrcons.exe-{hash}.pf

•mofcomp.exe-{hash}.pf

•evil.exe-{hash}.pf

■ Unauthorized changes to the

• wmiprvse.exe-{hash}.pf

WMI Repository in C:\Windows\

System32\wbem\Repository

FILE SYSTEM

EVENT LOGS

- ShimCache **SYSTEM** •sc.exe
- BAM/DAM **SYSTEM** Last Time Executed

■ BAM/DAM - **SYSTEM** - Last Time Executed

■ AmCache.hve - First Time Executed •sc.exe

■ ShimCache - **SYSTEM**

•wmic.exe

■ Prefetch - C:\Windows\Prefetch\ •sc.exe-{hash}.pf

FILE SYSTEM

FILE SYSTEM

■ Prefetch - C:\Windows\Prefetch\

• wmic.exe-{hash}.pf



sc \\host create servicename binpath= "c:\temp\evil.exe" sc \\host start servicename



WMI/WMIC

wmiprvse.exe

• **4624** Logon Type 3 Source IP/Logon User Name 4697 Security records service install, if enabled Enabling non-default Security events such as ID 4697 are particularly useful if only the

security.evtx

• 4672

• **4624** Logon Type 3

Logon User Name

- Logon by an a user with

administrative rights

Source IP/Logon User Name

Security logs are forwarded to a

centralized log server

security.evtx

EVENT LOGS system.evtx • **7034** – Service crashed unexpectedly

• **7035** – Service sent a Start/Stop control

EVENT LOGS

• 5857

• 7036 - Service started or stopped • 7040 – Start type changed (Boot | On Request | Disabled) • **7045** – A service was installed on the system

• New service creation ■ ShimCache – **SYSTEM** •evil.exe ShimCache records existence of malicious service executable, unless implemented as a service DLI ■ AmCache.hve -

First Time Executed

■ ShimCache - **SYSTEM**

•scrcons.exe

•mofcomp.exe

•wmiprvse.exe

First Time Executed

scrcons.exe

• mofcomp.exe

• wmiprvse.exe

• evil.exe

•evil.exe

■ AmCache.hve -

REGISTRY

•evil.exe

REGISTRY

• \CurrentControlSet\

•evil.exe

Services\

■ SYSTEM

service executable or service DLL ■ Prefetch - C:\Windows\Prefetch\ •evil.exe-{hash}.pf

■ File Creation

•evil.exe

■ File Creation

•evil.exe

■ File Creation

EVENT LOGS security.evtx

- 4648 Logon specifying alternate credentials - Current logged-on User Name
- Alternate User Name
- Destination Host Name/IP Process Name
- AmCache.hve First Time Executed •wmic.exe wmic /node:host process call create "C:\temp\evil.exe"
- **EVENT LOGS** • 8, 15, 16, 33 - WSMan Session security.evtx • 4648 – Logon specifying alternate credentials
- Destination Host Name/IP Process Name Microsoft-Windows-WinRM%4Operational.evtx • 6 - WSMan Session initialize
- deinitialization Closing of WSMan session - Current logged-on User Name Current logged-on User Name - Alternate User Name ■ Microsoft-Windows-PowerShell%4Operational.evtx • 40961, 40962

NTUSER.DAT HIVE

associated user account Session created Session created Destination Host Name or IP • 8197 - Connect Current logged-on User Name Session closed

FILE SYSTEM

tracked in powershell.exe prefetch file

Invoke-Command -ComputerName host -ScriptBlock {Start-Process c:\temp\evil.exe}

■ ShimCache - **SYSTEM** ■ Prefetch - C:\Windows\Prefetch\ •powershell.exe •powershell.exe-{hash}.pf ■ BAM/DAM - SYSTEM - PowerShell scripts (.ps1 files) that run within 10 seconds of powershell.exe launching will be Last Time Executed

Invoke-WmiMethod -Computer host -Class Win32 Process -Name create -Argument "c:\temp\evil.exe"

REGISTRY

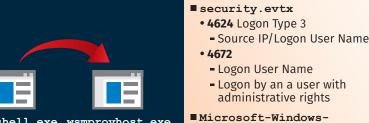
•powershell.exe

Enter-PSSession -ComputerName host

■ Command history ■ AmCache.hve - First C:\USERS\<USERNAME>\AppData\Roaming\ Time Executed Microsoft\Windows\PowerShell\ •powershell.exe PSReadline\ConsoleHost history.txt • With PS v5+, a history file with previous 4096



PowerShell Remoting



PowerShell%4Operational.evtx • **4103**, **4104** – Script Block logging Logs suspicious scripts by default in PS v5 Logs all scripts if configured • 53504 Records the authenticating

EVENT LOGS

■Windows PowerShell.evtx • 400/403 "ServerRemoteHost" indicates start/end of Remoting session • 800 Includes partial script code

Description:

■ Microsoft-Windows-WMI-

provider DLLs

• 5860, 5861

Activity%4Operational.evtx

Indicates time of wmiprvse execution

and path to provider DLL – attackers

- Registration of Temporary (5860) and

Permanent (5861) Event Consumers.

Typically used for persistence, but

can be used for remote execution.

sometimes install malicious WMI

■ Microsoft-Windows-WinRM%4Operational.evtx • 91 Session creation • 168 Records the authenticating

REGISTRY

■ ShimCache - **SYSTEM** •wsmprovhost.exe evil.exe SOFTWARE

■ AmCache.hve -

• evil.exe

First Time Executed

• wsmprovhost.exe

•Microsoft\PowerShell\1 \ShellIds\Microsoft. PowerShell\ ExecutionPolicy Attacker may change execution policy to a less restrictive setting, such as

• With Enter-PSSession, a user profile directory may be created ■ Prefetch - C:\Windows\Prefetch\ •evil.exe-{hash].pf •wsmprovhost.exe-{hash].pf

Evidence of Program **E**xecution

UserAssist

<mark>II-based programs launched from</mark> the desktop are tracked in the ncher on a Windows System.

NTUSER.DAT\Software\Microsoft\Windows\Currentversion\

xplorer\UserAssist\{GUID}\Count nterpretation: ll values are ROT-13 Encoded GUID for Win7/8/10 **CEBFF5CD** Executable File Execution

F4E57C4B Shortcut File Execution

Records the local initiation

of powershell.exe and

Description: lows Background Activity Moderator (BAM)

em and last execution date/time

BAM/DAM

SYSTEM\CurrentControlSet\Services\bam\UserSettings\{SID} SYSTEM\CurrentControlSet\Services\dam\UserSettings\{SID} des full path of the executable file that was run on the

RecentApps

rogram execution launched on the Win10 ystem is tracked in the RecentApps key

> ach GUID key points to a recent application. AppID = Name of Application LastAccessTime = Last execution time in UTC aunchCount = Number of times executed

NTUSER.DAT\Software\Microsoft\Windows\

urrent Version\Search\RecentApps

ShimCache

Description: Windows Application Compatibility Database is used by Windows to identify possible application compatibility challenges with executables. Tracks the executables' file name, file size, last modified time

Location:

Win7/8/10

SYSTEM\CurrentControlSet\Control\Session Manager\ AppCompatCache Interpretation: Any executable run on the Windows system could be found in this key. You can use this key to identify systems that specific malware was executed on. In addition, based on the

Windows 7/8/10 contains at most 1,024 entries

nterpretation of the time-based data you might be able to

determine the last time of execution or activity on the system

LastUpdateTime does not exist on Win7/8/10 systems

Jump Lists

The Windows 7-10 task bar (Jump List) is engineered to allow users to "jump" or access items they have requently or recently used quickly and easily. This unctionality cannot only include recent media files; t must also include recent tasks. The data stored in the AutomaticDestinations folder will each have a unique file prepended with the AppID of the associated application.

Location: Win7/8/10 :\%USERPROFILE%\AppData\Roaming\Microsoft\

Interpretation: First time of execution of application. Creation Time = First time item added to the

Last time of execution of application with file open. Modification Time = Last time item added to the ist of Jump List IDs ->

Prefetch

ncreases performance of a system by pre-loading code pages of commonly used applications. Cache Manager nonitors all files and directories referenced for each application or process and maps them into a .pf file. Utilized to know an application was executed on a system. Limited to 128 files on Win7

Limited to 1024 files on Win8-10 (exename)-(hash).pf **Location:**

Win7/8/10

C:\Windows\Prefetch Interpretation: Each .pf will include last time of execution, number of times run, and device and file handles used by the

Embedded last execution time of .pf file Last modification date of .pf file (-10 seconds) ww.forensicswiki.org/wiki/List_of_Jump_List_IDs Win8-10 will contain last 8 times of execution

Amcache.hve Description:

ogramDataUpdater (a task associated with the plication Experience Service) uses the registry file mcache.hve to store data during process creation Win7/8/10 C:\Windows\AppCompat\Programs\Amcache.hve (Windows 7/8/10)

Interpretation: Amcache.hve - Keys =

Amcache.hve\Root\File\{Volume GUID}\###### Entry for every executable run, full path information, File's StandardInfo Last Modification Time, and Disk volume the executable was run from

Date/Time file by that name and path was first executed Creation Date of .pf file (-10 seconds) Date/Time file by that name and path was last executed

First Run Time = Last Modification Time of Key SHA1 hash of executable also contained in the key

DFPS_FOR508_v4.10_02-23.indd 2