Digital Forensics and Incident Response

65 minute read

Introduction

This post is inspired by all the hard working DFIR, and more broadly security professionals, who have put in the hard yards over the years to discuss in depth digital forensics and incident response.

Disclaimer

This page contains a variety of commands and concepts which are known through experience, higher education, tutorials, online blogs, YouTube Videos, professional training, reading the manual, and more. All references to original posts or material will aim to be documented in the 'Special Thanks' section. This is not designed as a manual on how to perform DFIR, and serves only as a quick reference sheet for commands, tools, and common items of interest when performing Incident Response. If you need to undertake Digital Forensics for legal proceedings, seek specialist advice.

Artifact locations

A number of forensic artifacts are known for a number of operating systems.

A large number of these are covered on the Digital Forensics Artifact Repository, and can be ingested both by humans and systems given the standard YAML format.

- ForensicArtifacts (https://github.com/ForensicArtifacts/artifacts/tree/master/data)

Windows Cheat Sheet

Order of Volatility

If performing Evidence Collection rather than IR, respect the order of volatility as defined in: rfc3227

- registers, cache
- routing table, arp cache, process table, kernel statistics, memory
- temporary file systems
- disk
- remote logging and monitoring data that is relevant to the system in question
- physical configuration, network topology
- archival media

Memory Files (Locked by OS during use)

Note: To obtain these files while they're in use you can use a low level file extractor such as RawCopy (https://github.com/jschicht/RawCopy)

hiberfil.sys (RAM stored during machine hibernation)

- %SystemRoot%\hiberfil.sys

pagefile.sys (Virtual memory used by Windows)

- %SystemDrive%\pagefile.sys

swapfile.sys (Virtual memory used by Windows Store Apps)

- %SystemDrive%\swapfile.sys

Binalyze IREC Evidence Collector (https://binalyze.com/products/irec) (GUI or CommandLine)

```
IREC.exe --license AAAA-BBBB-CCDD-DDDD --profile memory
```

Note: Can be used as an all in one collector (License required for full collection, free version available).

Latest documentation (https://irec.readthedocs.io/en/latest/commandline.html)

Belkasoft Live RAM Capturer (https://belkasoft.com/get?product=ram)

```
RamCapture64.exe "output.mem"
```

OR for 32 bit OS

```
RamCapture32.exe "output.mem"
```

Redline

Excellent resource:

> *https://resources.infosecinstitute.com/memory-analysis-using-redline/*

Memoryze

```
MemoryDD.bat --output [LOCATION]
```

Comae DumpIT

```
DumpIt.exe /O [LOCATION]

        - Used for getting a memory crash file (Useful for analysis with both windbg and volatility)

DumpIt.exe /O [LOCATION]\mem.raw /T RAW

        - Used for getting a raw memory dump (Considered a legacy format)
```

These can be bundled with PSEXEC to execute on a remote PC; however, this will copy the file to the remote PC for executing. There's limitations if the tool requires other drivers or files to execute (such as RamCapture). An example command may be:

```
psexec \\remotepcname -c DumpIt.exe
```

Magnet Forensics (Mostly GUI)

- Magnet Forensics Tools (https://www.magnetforensics.com/resources/?cat=Free%20Tool)
- Magnet RAM Capture (https://www.magnetforensics.com/free-tool-magnet-ram-capture)
- Magnet Process Capture (https://www.magnetforensics.com/resources/magnet-process-capture/)

Volexity Surge (https://www.volexity.com/blog/2018/06/12/surge-collect-provides-reliable-memory-acquisition-across-windows-linux-and-macos/)

Imaging Live Machines

FTK Imager (Cmd version, mostly GUI for new versions) (https://accessdata.com/product-download)

```
ftkimager --list-drives
ftkimager \\.\PHYSICALDRIVE0 "[Location]\Case" --e01
ftkimager [source] [destination]
ftkimager \\.\PHYSICALDRIVE0 "[Location]\Case" --e01 --outpass securepasswordinsertedhere
```

DD

```
dd.exe --list
dd.exe if=/dev/<drive> of=Image.img bs=1M
dd.exe if=\\.\<OSDrive>: of=<drive>:\<name>.img bs=1M --size --progress
(LINUX) sudo dd if=/dev/<OSDrive> of=/mnt/<name>.ddimg bs=1M conv=noerror,sync
```

X-Ways Imager (https://www.x-ways.net/order.html)

Encase Forensic (https://www.guidancesoftware.com/encase-forensic)

Tableau Imager (https://www.guidancesoftware.com/tableau/download-center)

Guymager (https://guymager.sourceforge.io/)

Live Windows IR/Triage

CMD and WMIC (Windows Management Instrumentation Command-Line) Note: less information can be gathered by using 'list brief'.

Interact with remote machine

Enable Powershell remoting:

```
wmic /node:[IP] process call create "powershell enable-psremoting -force"
```

Powershell:

```
Enter-PSSession -ComputerName [IP]
```

PSExec:

```
PsExec: psexec \\IP -c cmd.exe
```

https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/enter-pssession?view=powershell-6

System information

```
echo %DATE% %TIME%
date /t
time /t
systeminfo
wmic computersystem list full
wmic /node:localhost product list full /format:csv
wmic softwarefeature get name,version /format:csv
wmic softwareelement get name,version /format:csv
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion" /s
echo %PATH%
SET
wmic bootconfig get /all /format:List
wmic computersystem get name, domain, manufacturer, model,
numberofprocessors,primaryownername,username,roles,totalphysicalmemory /format:list
wmic timezone get Caption, Bias, DaylightBias, DaylightName, StandardName
wmic recoveros get /all /format:List
wmic os get /all /format:list
wmic partition get /all /format:list
wmic logicaldisk get /all /format:list
wmic diskdrive get /all /format:list
fsutil fsinfo drives
```

(psinfo requires sysinternals psinfo.exe):

```
psinfo -accepteula -s -h -d
```

Obtain list of all files on a computer

```
tree C:\ /F > output.txt
dir C:\ /A:H /-C /Q /R /S /X
```

User and admin information

```
whoami
net users
net localgroup administrators
net group /domain [groupname]
net user /domain [username]
wmic sysaccount
wmic useraccount get name,SID
wmic useraccount list
```

Logon information

```
wmic netlogin list /format:List
```

NT Domain/Network Client Information

```
wmic ntdomain get /all /format:List
wmic netclient get /all /format:List
nltest /trusted_domains
```

Firewall Information

```
netsh Firewall show state
netsh advfirewall firewall show rule name=all dir=in type=dynamic
netsh advfirewall firewall show rule name=all dir=out type=dynamic
netsh advfirewall firewall show rule name=all dir=in type=static
netsh advfirewall firewall show rule name=all dir=out type=dynamic
```

Pagefile information

```
wmic pagefile
```

Group and access information

(Accesschk requires accesschk64.exe or accesschk.exe from sysinternals):

```
net localgroup
accesschk64 -a *
```

```
C:\Users\*\AppData\Local\Microsoft\Windows\INetCookies
C:\Users\*\AppData\Roaming\Microsoft\Windows\Cookies
C:\Users\*\AppData\Roaming\Microsoft\Windows\Cookies\Low
```

RecentDocs Information

Special thanks Barnaby Skeggs (https://twitter.com/barnabyskeggs)

*Note: Run with Powershell, get SID and user information with ‘wmic useraccount get name,SID’

```
$SID = "S-1-5-21-1111111111-11111111111-1111111-11111"; $output = @(); Get-Item -Path
"Registry::HKEY_USERS\$SID\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs" | Select-
Object -ExpandProperty property | ForEach-Object {$i = [System.Text.Encoding]::Unicode.GetString((gp
"Registry::HKEY_USERS\$SID\Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs" -Name
$_).$_); $i = $i -replace '[^a-zA-Z0-9 \.\-_\\/()~ ]', '\^'; $output += $i.split('\^')[0]}; $output
| Sort-Object -Unique
```

More information on recent documents may be found:

```
C:\Users\[username]\AppData\Local\Microsoft\Windows\FileHistory\Data
```

Startup process information

```
wmic startup list full
wmic startup list brief
Get-CimInstance Win32_StartupCommand | Select-Object Name, command, Location, User | FL

$Malware="TERMTOSEARCH";
Get-Service -Name "*$Malware*";reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "ImagePath" |
findstr "$Malware";reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "ServiceDLL" | findstr
"$Malware";Get-ScheduledTask | ? {$_.TaskPath -match "$Malware"};gci -path C:\windows\system32\tasks
| Select-String Command | ? {$_.Line -match "$Malware"} | FL Line, Filename;
```

Scheduled task/job information

```
at (For older OS)
schtasks
schtasks /query /fo LIST /v
schtasks /query /fo LIST /v | findstr "Task To Run:"
schtasks /query /fo LIST /v | findstr "appdata"
schtasks /query /fo LIST /v | select-string "Enabled" -CaseSensitive -Context 10,0 | findstr "exe"
schtasks /query /fo LIST /v | select-string "Enabled" -CaseSensitive -Context 10,0 | findstr "Task"
schtasks /query /fo LIST /v | Select-String "exe" -Context 2,27
gci -path C:\windows\system32\tasks -recurse | Select-String Command | ? {$_.Line -match
"MALICIOUSSERVICE"} | FL Line, Filename
wmic job get Name, Owner, DaysOfMonth, DaysOfWeek, ElapsedTime, JobStatus, StartTime, Status
```

Powershell:

```
Get-ScheduledTask
gci -path C:\windows\system32\tasks -recurse | Select-String Command | FL Filename, Line
gci -path C:\windows\system32\tasks -recurse | Select-String Command | ? {$_.Line -match
"MALICIOUSNAME"} | FL Filename, Line
```

Remediate malicious scheduled tasks

```
schtasks /Delete /TN [taskname] /F
```

Powershell:

```
Unregister-ScheduledTask -TaskName [taskname]
Unregister-ScheduledTask -TaskPath [taskname]
```

UAC Bypass Fodhelper

```
reg query HKCU\Software\Classes\ms-settings\shell\open\command
reg query HKU\{SID}\Software\Classes\ms-settings\shell\open\command
```

Quick overview of persistent locations (AutoRuns) (https://docs.microsoft.com/en-us/sysinternals/downloads/autoruns)

```
autorunsc.exe -accepteula -a * -c -h -v -m > autoruns.csv
autorunsc.exe -accepteula -a * -c -h -v -m -z 'E:\Windows' > autoruns.csv
```

Persistence and Automatic Load/Run Reg Keys

Replace: “reg query” with “Get-ItemProperty -Path HK:" in Powershell*

e.g.: Get-Item -Path HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

User Registry (NTUSER.DAT HIVE) - Commonly located at:

```
C:\Users\[username]
```

*Note: These are setup for querying the current users registry only (HKCU), to query others you will need to load them from the relevant NTUSER.DAT file and then query them.

```
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run"
reg query "HKCU\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows" /f run
reg query "HKCU\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows" /f load
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Windows\Scripts"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\RecentDocs"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\RunMRU"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\RunMRU"
reg query "HKCU\SOFTWARE\AcroDC"
reg query "HKCU\SOFTWARE\Itime"
reg query "HKCU\SOFTWARE\info"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\User Shell Folders"
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Applets\RegEdit" /v LastKey
reg query "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSavePidlMRU" /s
reg query "HKCU\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell"
reg query "HKCU\SOFTWARE\Microsoft\Windows\currentversion\run"
reg query "HKCU\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Terminal
Server\Install\Software\Microsoft\Microsoft\Windows\CurrentVersion\Run"
reg query "HKCU\Software\Microsoft\Windows NT\CurrentVersion\Terminal
Server\Install\Software\Microsoft\Microsoft\Windows\CurrentVersion\RunOnce"
reg query "HKCU\SOFTWARE\Microsoft\Office\[officeversion]\[word/excel/access
etc]\Security\AccessVBOM"
        reg query "HKCU\SOFTWARE\Microsoft\Office\15.0\Excel\Security\AccessVBOM
        reg query "HKCU\SOFTWARE\Microsoft\Office\15.0\Word\Security\AccessVBOM
        reg query "HKCU\SOFTWARE\Microsoft\Office\15.0\Powerpoint\Security\AccessVBOM
        reg query "HKCU\SOFTWARE\Microsoft\Office\15.0\Access\Security\AccessVBOM
```

Local Machine (SOFTWARE HIVE)

```
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon"
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce"
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx"
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServicesOnce"
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices"
reg query "HKLM\SOFTWARE\Policies\Microsoft\Windows\System\Scripts"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows" /f AppInit_DLLs
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run"
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Win\Userinit"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options" /s
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit" /s
reg query "HKLM\SOFTWARE\wow6432node\Microsoft\Windows\CurrentVersion\policies\explorer\run"
reg query "HKLM\SOFTWARE\wow6432node\Microsoft\Windows\CurrentVersion\run"
reg query "HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Windows"
reg query "HKLM\SOFTWARE\Microsoft\Office\[officeversion]\[word/excel/access etc]\Security\AccessVBOM"
reg query HKLM\SOFTWARE\Microsoft\Active Setup\Installed Components\[Random]\StubPath" /s
        reg query "HKLM\SOFTWARE\Microsoft\Office\15.0\Excel\Security\AccessVBOM
        reg query "HKLM\SOFTWARE\Microsoft\Office\15.0\Word\Security\AccessVBOM
        reg query "HKLM\SOFTWARE\Microsoft\Office\15.0\Powerpoint\Security\AccessVBOM
        reg query "HKLM\SOFTWARE\Microsoft\Office\15.0\Access\Security\AccessVBOM
```

Don’t be afraid to use “findstr” to find entries of interest, for example file extensions which may also invoke malicious executables when run, or otherwise.

```
reg query "HKLM\SOFTWARE\Classes" | findstr "file"
reg query HKCR\CLSID\{AB8902B4-09CA-4bb6-B78D-A8F59079A8D5} /s
reg query HKCR\AppID\ /s | findstr "exe"
```

Local Machine (SYSTEM HIVE)

Note: This not only contains services, but also malicious drivers which may run at startup (these are in the form of “.sys” files and are generally loaded from here: \SystemRoot\System32\drivers)

```
reg query "HKLM\SYSTEM\CurrentControlSet\Services\[Random_name]\imagePath"
reg query "HKLM\SYSTEM\CurrentControlSet\Services\ /s /f "*.exe"
reg query "HKLM\SYSTEM\CurrentControlSet\Services" /s | findstr "ImagePath" | findstr ".exe"
reg query "HKLM\SYSTEM\CurrentControlSet\Services" /s | findstr "ImagePath" | findstr ".sys"
Get-Service -Name "*MALICIOUSSERVICE*"
gwmi win32_service | ? {$_.PathName -match "MALICIOUSSERVICE"}|FL Name,PathName
Get-ItemProperty "HKLM:\SYSTEM\CurrentControlSet\Services\*" | FL DisplayName,ImagePath,ObjectName
gci -Path C:\Windows\system32\drivers -include *.sys -recurse -ea 0 -force | Get-AuthenticodeSignature
gci -Path C:\Windows\system32\drivers -include *.sys -recurse -ea 0 -force | Get-FileHash
```

Note: Some useful commands to show relevant service information

```
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "ImagePath"
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "ServiceDLL"
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "FailureCommand"
```

T1015 Accessibility Features

```
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\sethc.exe" /v "Debugger"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\utilman.exe" /v "Debugger"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\AtBroker.exe" /v "Debugger"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\Narrator.exe" /v "Debugger"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\Magnify.exe" /v "Debugger"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\DisplaySwitch.exe" /v "Debugger"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\osk.exe" /v "Debugger"
sfc /VERIFYFILE=C:\Windows\System32\sethc.exe
sfc /VERIFYFILE=C:\Windows\System32\utilman.exe
sfc /VERIFYFILE=C:\Windows\System32\AtBroker.exe
sfc /VERIFYFILE=C:\Windows\System32\Narrator.exe
sfc /VERIFYFILE=C:\Windows\System32\Magnify.exe
sfc /VERIFYFILE=C:\Windows\System32\DisplaySwitch.exe
sfc /VERIFYFILE=C:\Windows\System32\osk.exe
```

T1098 Account Manipulation

```
N/A
```

T1182 AppCert DLLs

```
reg query "HKLM\System\CurrentControlSet\Control\Session Manager" /v AppCertDlls
```

T1103 AppInit DLLs

```
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows" /v Appinit_Dlls
reg query "HKLM\Software\Wow6432Node\Microsoft\Windows NT\CurrentVersion\Windows" /v Appinit_Dlls
Get-WinEvent -FilterHashtable @{ LogName='System'; Id='11'} | FL TimeCreated,Message
```

T1138 Application Shimming

```
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\Custom"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\InstalledSDB"
```

T1197 BITS Jobs

```
bitsadmin /list /allusers /verbose
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-Bits-Client/Operational'; Id='59'} | FL TimeCreated,Message
```

T1067 Bootkit

Note: This exists below the OS in the Master Boot Record or Volume Boot Record. The system must be booted through Advanced Startup Options with a Command Prompt, or through a recovery cd.

```
bootrec /FIXMBR
bootrec /FIXBOOT
```

Extra: If your boot configuration data is missing or contains errors the below can fix this.

```
bootrec /REBUILDBCD
```

If you're thinking of a bootkit more as malicious system drivers you can go with the below.

Unsigned Drivers

```
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-CodeIntegrity/Operational'; Id='3001'} | FL TimeCreated,Message
gci -path C:\Windows\System32\drivers -include *.sys -recurse -ea SilentlyContinue | Get-AuthenticodeSignature | where {$_.status -ne 'Valid'}
```

T1176 Browser Extensions

Chrome

```
Get-ChildItem -path "C:\Users\*\AppData\Local\Google\Chrome\User Data\Default\Extensions" -recurse -erroraction SilentlyContinue
reg query "HKLM\Software\Google\Chrome\Extensions" /s
reg query "HKLM\Software\Wow6432Node\Google\Chrome\Extensions" /s
```

Firefox

```
Get-ChildItem -path "C:\Users\*\AppData\Roaming\Mozilla\Firefox\Profiles\*\extensions" -recurse -erroraction SilentlyContinue
Get-ChildItem -path "C:\Program Files\Mozilla Firefox\plugins\" -recurse -erroraction SilentlyContinue
Get-ChildItem -path registry::HKLM\SOFTWARE\Mozilla\*\extensions
```

Edge

```
Get-ChildItem -Path C:\Users\*\AppData\Local\Packages\ -recurse -erroraction SilentlyContinue
```

Internet Explorer

```
Get-ChildItem -path "C:\Program Files\Internet Explorer\Plugins\" -recurse -erroraction SilentlyContinue
reg query 'HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\Browser Helper Objects'
reg query 'HKEY_LOCAL_MACHINE\Software\Microsoft\Internet Explorer\Toolbar'
reg query 'HKEY_LOCAL_MACHINE\Software\Microsoft\Internet Explorer\URLSearchHooks'
reg query 'HKEY_LOCAL_MACHINE\Software\Microsoft\Internet Explorer\Explorer Bars'
reg query 'HKU\{SID}\Software\Microsoft\Internet Explorer\Explorer Bars'
reg query 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Internet Explorer\Extensions'
```

T1109 Component Firmware

Note: This is incredibly rare, and doesn’t have an easy detection/remediation mechanism. Using the Windows CheckDisk utility may assist but isn’t guaranteed.

```
chkdsk /F
```

T1122 Component Object Model Hijacking

Note: This involves replacing legitimate components with malicious ones, and as such the legitimate components will likely no longer function. If you have a detection based on DLLHost.exe with /Processid:{xyz}, you can match xyz with the CLSID mentioned below to check for any malicious EXE or DLL.

```
reg query HKLM\SOFTWARE\Classes\CLSID\ /s /f ".dll"
reg query HKLM\SOFTWARE\Classes\CLSID\ /s /f ".exe"
gci -path REGISTRY::HKLM\SOFTWARE\Classes\*\shell\open\command
reg query HKU\{SID}\SOFTWARE\Classes\CLSID\ /s /f ".dll"
reg query HKU\{SID}\SOFTWARE\Classes\CLSID\ /s /f ".exe"
```

T1136 Create Account

```
net user
net user /domain
```

T1038 DLL Search Order Hijacking

```
reg query "HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\KnownDLLs"
gci -path C:\Windows\* -include *.dll | Get-AuthenticodeSignature | Where-Object Status -NE "Valid"
gci -path C:\Windows\System32\* -include *.dll | Get-AuthenticodeSignature | Where-Object Status -NE "Valid"
```

T1133 External Remote Services

```
N/A
```

T1044 File System Permissions Weakness

```
Get-WmiObject win32_service | FL name,PathName
get-acl "C:\Program Files (x86)\Google\Update\GoogleUpdate.exe" | FL | findstr "FullControl"
```

T1158 Hidden Files and Directories

```
dir /S /A:H
```

T1179 Hooking

```
[Contextis Blog](https://www.contextis.com/en/blog/common-language-runtime-hook-for-persistence)
[GetHooks](https://github.com/jay/gethooks/releases/tag/1.01)
```

T1062 Hypervisor

```
N/A
```

T1183 Image File Execution Options Injection

```
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\SilentProcessExit" /s       | findstr "MonitorProcess"
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File Execution Options" /s    | findstr "Debugger"
```

T1037 Logon Scripts

```
reg query "HKU\{SID}\Environment\UserInitMprLogonScript"
```

T1177 LSASS Driver

```
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4614';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='3033';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='3063';} | FL TimeCreated,Message
```

T1031 Modify Existing Service

```
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "ImagePath"
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "ServiceDLL"
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "FailureCommand"
```

T1128 Netsh Helper DLL

```
reg query HKLM\SOFTWARE\Microsoft\Netsh
```

T1050 New Service

```
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "ImagePath"
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "ServiceDLL"
reg query HKLM\SYSTEM\CurrentControlSet\Services /s /v "FailureCommand"
Get-WmiObject win32_service | FL Name, DisplayName, PathName, State
Get-WinEvent -FilterHashtable @{ LogName='System'; Id='7045';} | FL TimeCreated,Message
```

T1137 Office Application Startup

```
reg query "HKU\{SID}\Software\Microsoft\Office test\Special\Perf" /s
reg query "HKLM\Software\Microsoft\Office test\Special\Perf" /s
Get-ChildItem -path registry::HKLM\SOFTWARE\Microsoft\Office\*\Addins\*
Get-ChildItem -path registry::HKLM\SOFTWARE\Wow6432node\Microsoft\Office\*\Addins\*
Get-ChildItem -path registry::HKLM\SOFTWARE\Wow6432node\Microsoft\Office\*\Addins\*
Get-ChildItem -path "C:\Users\*\AppData\Roaming\Microsoft\Templates\*" -erroraction SilentlyContinue
Get-ChildItem -path "C:\Users\*\AppData\Roaming\Microsoft\Excel\XLSTART\*" -erroraction
SilentlyContinue
Get-ChildItem -path C:\ -recurse -include Startup -ea 0
ls 'C:\Program Files\Microsoft Office\root\Office16\XLSTART'
ls 'C:\Program Files\Microsoft Office\root\Office16\STARTUP'
reg query HKCU\Software\Microsoft\Office\<Outlook Version>\Outlook\WebView\Inbox
reg query HKCU\Software\Microsoft\Office\<Outlook Version>\Outlook\Security
reg query HKCU\Software\Microsoft\Office\<Outlook Version>\Outlook\Today\UserDefinedUrl
reg query HKCU\Software\Microsoft\Office\<Outlook Version>\Outlook\WebView\Calendar\URL
Get-WinEvent -FilterHashtable @{ LogName='Microsoft Office Alerts'; Id='300';} | FL
TimeCreated,Message
```

T1034 Path Interception

```
N/A
```

T1013 Port Monitors

```
reg query "HKLM\SYSTEM\CurrentControlSet\Control\Print\Monitors" /s /v "Driver"
```

T1504 PowerShell Profile

```
ls C:\Windows\System32\WindowsPowerShell\v1.0\Profile.ps1
ls C:\Windows\System32\WindowsPowerShell\v1.0\Microsoft.*Profile.ps1
ls C:\Windows\System32\WindowsPowerShell\v1.0\Microsoft.*Profile.ps1
gci -path "C:\Users\*\My Documents\PowerShell\Profile.ps1"
gci -path "C:\Users\*\My Documents\PowerShell\Microsoft.*Profile.ps1"
```

T1108 Redundant Access

```
N/A
```

T1060 Registry Run Keys / Startup Folder

```
reg query "HKU\{SID}\Software\Microsoft\Windows\CurrentVersion\Run"
reg query "HKU\{SID}\Software\Microsoft\Windows\CurrentVersion\RunOnce"
reg query "HKU\{SID}\Software\Microsoft\Windows\CurrentVersion\RunOnceEx"
reg query "HKLM\Software\Microsoft\Windows\CurrentVersion\Run"
reg query "HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce"
reg query "HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnceEx"
reg query "HKU\{SID}\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders"
reg query "HKU\{SID}\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders"
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders"
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders"
reg query "HKLM\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce"
reg query "HKU\{SID}\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce"
reg query "HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices"
reg query "HKU\{SID}\Software\Microsoft\Windows\CurrentVersion\RunServices"
reg query "HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run"
reg query "HKU\{SID}\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\Run"
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Userinit"
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell"
reg query "HKU\{SID}\Software\Microsoft\Windows NT\CurrentVersion\Windows"
reg query "HKLM\System\CurrentControlSet\Control\Session Manager"
gci -path "C:\Users\*\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\*" -include *.lnk,*.url
gci -path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp\*" -include *.lnk,*.url
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-Shell-Core/Operational'; Id='9707'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-Shell-Core/Operational'; Id='9708'} | FL TimeCreated,Message
```

T1053 Scheduled Task

```
schtasks /query /fo LIST /v | select-string "Enabled" -CaseSensitive -Context 10,0 | findstr "Task"
gci -path C:\windows\system32\tasks |Select-String Command|FT Line, Filename
Get-ChildItem -path 'registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\'
ls 'C:\Windows\System32\WptsExtensions.dll'
        Note: thanks to [Markus Piéton]
(https://twitter.com/markus_pieton/status/1189559716453801991) for this one.
```

T1180 Screensaver

```
reg query "HKU\{SID}\Control Panel\Desktop" /s /v "ScreenSaveActive"
reg query "HKU\{SID}\Control Panel\Desktop" /s /v "SCRNSAVE.exe"
reg query "HKU\{SID}\Control Panel\Desktop" /s /v "ScreenSaverIsSecure"
```

T1101 Security Support Provider

```
reg query "HKLM\SYSTEM\CurrentControlSet\Control\Lsa\OSConfig" /v "Security Packages"
reg query "HKLM\SYSTEM\CurrentControlSet\Control\Lsa" /v "Security Packages"
```

T1505 Server Software Component

```
N/A
```

T1058 Service Registry Permissions Weakness

```
get-acl REGISTRY::HKLM\SYSTEM\CurrentControlSet\Services\* |FL
get-acl REGISTRY::HKLM\SYSTEM\CurrentControlSet\Services\servicename |FL
```

T1023 Shortcut Modification

```
Select-String -Path "C:\Users\*\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\*.lnk"
-Pattern "exe"
Select-String -Path "C:\Users\*\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\*.lnk"
-Pattern "dll"
Select-String -Path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp\*" -Pattern "dll"
Select-String -Path "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp\*" -Pattern "exe"
gci -path "C:\Users\" -recurse -include *.lnk -ea SilentlyContinue | Select-String -Pattern "exe" |
FL
gci -path "C:\Users\" -recurse -include *.lnk -ea SilentlyContinue | Select-String -Pattern "dll" |
FL
```

T1198 SIP and Trust Provider Hijacking

```
reg query "HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CryptSIPDllGetSignedDataMsg" /s
/v "Dll"
reg query "HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CryptSIPDllVerifyIndirectData" /s
/v "Dll"
reg query "HKLM\SOFTWARE\Microsoft\Cryptography\Providers\Trust\FinalPolicy" /s /v "`$DLL"
reg query "HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\OID\EncodingType
0\CryptSIPDllGetSignedDataMsg" /s /v "Dll"
reg query "HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\OID\EncodingType
0\CryptSIPDllVerifyIndirectData" /s /v "Dll"
reg query "HKLM\SOFTWARE\WOW6432Node\Microsoft\Cryptography\Providers\Trust\FinalPolicy" /s /v
"`$DLL"
```

T1019 System Firmware

```
reg query HKLM\HARDWARE\DESCRIPTION\System\BIOS
Confirm-SecureBootUEFI
Get-WmiObject win32_bios
```

T1209 Time Providers

```
reg query "HKLM\System\CurrentControlSet\Services\W32Time\TimeProviders" /s | findstr "Dll"
```

T1078 Valid Accounts

```
N/A
```

T1100 Web Shell

```
gci -path "C:\inetpub\wwwroot" -recurse -File -ea SilentlyContinue | Select-String -Pattern "runat" | FL
gci -path "C:\inetpub\wwwroot" -recurse -File -ea SilentlyContinue | Select-String -Pattern "eval" | FL
```

T1084 Windows Management Instrumentation Event Subscription

Get WMI Namespaces

```
Function Get-WmiNamespace ($Path = 'root')
{
        foreach ($Namespace in (Get-WmiObject -Namespace $Path -Class __Namespace))
        {
                $FullPath = $Path + "/" + $Namespace.Name
                Write-Output $FullPath
                Get-WmiNamespace -Path $FullPath
        }
}
Get-WMINamespace -Recurse
```

Query WMI Persistence

```
Get-WmiObject -Class __FilterToConsumerBinding -Namespace root\subscription
Get-WmiObject -Class __EventFilter -Namespace root\subscription
Get-WmiObject -Class __EventConsumer -Namespace root\subscription
```

T1004 Winlogon Helper DLL

```
reg query "HKU\{SID}\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify" /s
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify" /s
reg query "HKU\{SID}\Software\Microsoft\Windows NT\CurrentVersion\Winlogon" /v "Userinit"
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon" /v "Userinit"
reg query "HKU\{SID}\Software\Microsoft\Windows NT\CurrentVersion\Winlogon" /v "Shell"
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon" /v "Shell"
reg query "HKLM\Software\WOW6432Node\Microsoft\Windows NT\CurrentVersion\Winlogon" /s
```

Other - Winsock Persistence

```
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-Winsock-WS2HELP/Operational'; Id='1'} |
FL TimeCreated,Message
```

Check disabled task manager (often from malware)

```
reg query HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System /v DisableTaskMgr
```

Locate all user registry keys

```
$UserProfiles = Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList\*"
| Where {$_.PSChildName -match "S-1-5-21-(\d+-?){4}$" } | Select-Object @{Name="SID"; Expression=
{$_.PSChildName}}, @{Name="UserHive";Expression={"$($_.ProfileImagePath)\ntuser.dat"}}
```

Load all users registry keys from their ntuser.dat file (perform above first)

```
Foreach ($UserProfile in $UserProfiles) {If (($ProfileWasLoaded = Test-Path
Registry::HKEY_USERS\$($UserProfile.SID)) -eq $false) {reg load HKU\$($UserProfile.SID)
$($UserProfile.UserHive) | echo "Successfully loaded: $($UserProfile.UserHive)"}}
```

Query all users run key

```
Foreach ($UserProfile in $UserProfiles) {reg query
HKU\$($UserProfile.SID)\SOFTWARE\Microsoft\Windows\CurrentVersion\Run}
```

Unload all users registry keys

```
[gc]::Collect()
Foreach ($UserProfile in $UserProfiles) {reg unload HKU\$($UserProfile.SID)}
```

Locate, load, query all users/local machine run keys, and then unload

```
$UserProfiles = Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList\*"
| Where {$_.PSChildName -match "S-1-5-21-(\d+-?){4}$" } | Select-Object @{Name="SID"; Expression=
{$_.PSChildName}}, @{Name="UserHive";Expression={"$($_.ProfileImagePath)\ntuser.dat"}};Foreach
($UserProfile in $UserProfiles) {If (($ProfileWasLoaded = Test-Path
Registry::HKEY_USERS\$($UserProfile.SID)) -eq $false) {reg load HKU\$($UserProfile.SID)
$($UserProfile.UserHive) | echo "Successfully loaded: $($UserProfile.UserHive)"}};reg query
HKLM\Software\Microsoft\Windows\CurrentVersion\Run;Foreach ($UserProfile in $UserProfiles) {reg
query HKU\$($UserProfile.SID)\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce;reg query
HKU\$($UserProfile.SID)\SOFTWARE\Microsoft\Windows\CurrentVersion\Run;reg query
"HKU\$($UserProfile.SID)\Environment\UserInitMprLogonScript"};[gc]::Collect();Foreach ($UserProfile
in $UserProfiles) {reg unload HKU\$($UserProfile.SID)};
```

Remediate Automatic Load/Run Reg Keys

```
reg delete [keyname] /v [ValueName] /f
reg delete [keyname] /f
Foreach ($UserProfile in $UserProfiles) {reg delete
HKU\$($UserProfile.SID)\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce /f}
Foreach ($UserProfile in $UserProfiles) {reg delete
HKU\$($UserProfile.SID)\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /f}
```

Powershell:

```
Remove-ItemProperty -Path "[Path]" -Name "[name]"
```

Scan Registry keys for specified text

```
Get-ChildItem -path HKLM:\ -Recurse -ea SilentlyContinue | where {$_.Name -match 'notepad' -or
$_.Name -match 'sql'}
Get-ChildItem -path HKLM:\ -Recurse -ea SilentlyContinue | get-itemproperty | where {$_ -match
'notepad' -or $_ -match 'sql'}
```

Persistent file locations of interest

```
%localappdata%\<random>\<random>.<4-9 file ext>
%localappdata%\<random>\<random>.lnk
%localappdata%\<random>\<random>.bat
%appdata%\<random>\<random>.<4-9 file ext>
%appdata%\<random>\<random>.lnk
%appdata%\<random>\<random>.bat
%appdata%\<random>\<random>.bat
%SystemRoot%\<random 4 chars starting with digit>
%appdata%\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\*
"C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp\*"
%SystemRoot%\System32\<randomnumber>\
%SystemRoot%\System32\tasks\<randomname>
%SystemRoot%\\<randomname>
C:\Users\[user]\appdata\roaming\[random]
C:\Users\Public\*
```

You can scan these directories for items of interest e.g. unusual exe, dll, bat, lnk etc files with:

```
dir /s /b %localappdata%\*.exe | findstr /e .exe
dir /s /b %appdata%\*.exe | findstr /e .exe
dir /s /b %localappdata%\*.dll | findstr /e .dll
dir /s /b %appdata%\*.dll | findstr /e .dll
dir /s /b %localappdata%\*.bat | findstr /e .bat
dir /s /b "%appdata%\Microsoft\Windows\Start Menu\Programs\Startup\" | findstr /e .lnk
dir /s /b "C:\Users\Public\" | findstr /e .exe
dir /s /b "C:\Users\Public\" | findstr /e .lnk
dir /s /b "C:\Users\Public\" | findstr /e .dll
dir /s /b "C:\Users\Public\" | findstr /e .bat
ls "C:\Users\[User]\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup" | findstr /e .lnk
```

Locate LNK Files with a particular string (Special thanks to the notorious)

```
Select-String -Path 'C:\Users\[User]\AppData\Roaming\Microsoft\Windows\Start
Menu\Programs\Startup\*.lnk' -Pattern "powershell" | Select Path
```

Determine Timestomping

Within the Master File Table (Located at the Win root) there are 2 elements, $STANDARD_INFORMATION and $FILE_NAME, both of which have values for a file being created, modified, accessed and written.

These are known as MACB times (Modified, Accessed, Changed, Birth). The $STANDARD_INFORMATION element can be modified from a malicious process, but the $FILE_NAME element is left intact and cannot without some extra trickery.

These discrepancies generally indicate Timestomping with the $FILE_NAME entry being the source of truth. This can be determined by obtaining the MFT (e.g. using a tool such as Rawcopy), and comparing timestamps on the file (e.g. using a tool such as MFTExplorer).

Rawcopy (https://github.com/jschicht/RawCopy)

```
RawCopy.exe /FileNamePath:C:0 /OutputPath:C:\Audit /OutputName:MFT_C.bin
```

MFTExplorer (https://ericzimmerman.github.io/#!index.md)

Remove BITSAdmin Persistence

```
bitsadmin /reset /allusers
import-module bitstransfer
Get-BitsTransfer -AllUsers | Remove-BitsTransfer
```

Locate Possible Trickbot

```
gci -path C:\Users\*\AppData\Roaming\*\Data -recurse -ea SilentlyContinue
gci -path C:\Users\*\AppData\Roaming\*\Modules -recurse -ea SilentlyContinue
gci -path C:\Users\*\AppData\Local\*\Data -recurse -ea SilentlyContinue
gci -path C:\Users\*\AppData\Local\*\Modules -recurse -ea SilentlyContinue
schtasks /query /fo LIST /v | findstr "appdata"
```

Locate Possible Emotet Service Binaries (Excludes Win10 Common Executables)

```
gi -path C:\Windows\System32\*.exe -exclude
agentactivationruntimestarter.exe,aitstatic.exe,alg.exe,AppHostRegistrationVerifier.exe,appidcertsto
recheck.exe,appidpolicyconverter.exe,appidtel.exe,ApplicationFrameHost.exe,ApplyTrustOffline.exe,App
roveChildRequest.exe,appverif.exe,ARP.EXE,at.exe,AtBroker.exe,attrib.exe,audiodg.exe,auditpol.exe,Au
thHost.exe,autochk.exe,autoconv.exe,autofmt.exe,AxInstUI.exe,backgroundTaskHost.exe,BackgroundTransf
erHost.exe,bash.exe,bcdboot.exe,bcdedit.exe,BdeUISrv.exe,bdeunlock.exe,BioIso.exe,BitLockerDeviceEnc
ryption.exe,BitLockerWizardElev.exe,bitsadmin.exe,bootcfg.exe,bootim.exe,bootsect.exe,bridgeunattend
.exe,browserexport.exe,browser_broker.exe,bthudtask.exe,ByteCodeGenerator.exe,cacls.exe,calc.exe,Cam
eraSettingsUIHost.exe,CastSrv.exe,CertEnrollCtrl.exe,certreq.exe,certutil.exe,changepk.exe,charmap.e
xe,CheckNetIsolation.exe,chkdsk.exe,chkntfs.exe,choice.exe,CIDiag.exe,cipher.exe,cleanmgr.exe,clicon
fg.exe,clip.exe,ClipRenew.exe,ClipUp.exe,CloudExperienceHostBroker.exe,CloudNotifications.exe,cmd.ex
e,cmdkey.exe,cmdl32.exe,cmmon32.exe,cmstp.exe,cofire.exe,colorcpl.exe,comp.exe,compact.exe,CompatTel
Runner.exe,CompMgmtLauncher.exe,CompPkgSrv.exe,ComputerDefaults.exe,conhost.exe,consent.exe,control.
exe,convert.exe,convertvhd.exe,coredpussvr.exe,CredentialEnrollmentManager.exe,CredentialUIBroker.ex
e,credwiz.exe,cscript.exe,csrss.exe,ctfmon.exe,cttune.exe,cttunesvr.exe,curl.exe,CustomInstallExec.e
xe,dasHost.exe,DataExchangeHost.exe,DataStoreCacheDumpTool.exe,DataUsageLiveTileTask.exe,dccw.exe,dc
omcnfg.exe,ddodiag.exe,Defrag.exe,deploymentcsphelper.exe,desktopimgdownldr.exe,DeviceCensus.exe,Dev
iceCredentialDeployment.exe,DeviceEject.exe,DeviceEnroller.exe,DevicePairingWizard.exe,DevicePropert
ies.exe,DFDWiz.exe,dfrgui.exe,dialer.exe,directxdatabaseupdater.exe,diskpart.exe,diskperf.exe,diskra
id.exe,DiskSnapshot.exe,Dism.exe,dispdiag.exe,DisplaySwitch.exe,djoin.exe,dllhost.exe,dllhst3g.exe,d
mcertinst.exe,dmcfghost.exe,dmclient.exe,DmNotificationBroker.exe,DmOmaCpMo.exe,dnscacheugc.exe,dosk
ey.exe,dpapimig.exe,DpiScaling.exe,dpnsvr.exe,driverquery.exe,drvcfg.exe,drvinst.exe,DsmUserTask.exe
,dsregcmd.exe,dstokenclean.exe,DTUHandler.exe,dusmtask.exe,dvdplay.exe,dwm.exe,DWWIN.EXE,DXCap.exe,D
XCpl.exe,dxdiag.exe,dxgiadaptercache.exe,Dxpserver.exe,Eap3Host.exe,EaseOfAccessDialog.exe,easinvoke
r.exe,EASPolicyManagerBrokerHost.exe,EDPCleanup.exe,edpnotify.exe,EduPrintProv.exe,efsui.exe,EhStorA
uthn.exe,escsvc64.exe,esentutl.exe,eudcedit.exe,eventcreate.exe,eventvwr.exe,expand.exe,extrac32.exe
,fc.exe,fhmanagew.exe,FileHistory.exe,find.exe,findstr.exe,finger.exe,fixmapi.exe,fltMC.exe,fodhelpe
r.exe,Fondue.exe,fontdrvhost.exe,fontview.exe,forfiles.exe,fsavailux.exe,FsIso.exe,fsquirt.exe,fsuti
l.exe,ftp.exe,fvenotify.exe,FXSCOVER.exe,FXSSVC.exe,FXSUNATD.exe,GameBarPresenceWriter.exe,GamePanel
.exe,GenValObj.exe,getmac.exe,gpresult.exe,gpupdate.exe,grpconv.exe,hdwwiz.exe,help.exe,HOSTNAME.EXE
,hvax64.exe,hvix64.exe,icacls.exe,IcsEntitlementHost.exe,icsunattend.exe,ie4uinit.exe,ie4ushowIE.exe
,ieUnatt.exe,iexpress.exe,immersivetpmvscmgrsvr.exe,InfDefaultInstall.exe,InputSwitchToastHandler.ex
e,ipconfig.exe,iscsicli.exe,iscsicpl.exe,isoburn.exe,klist.exe,ksetup.exe,ktmutil.exe,label.exe,Lang
uageComponentsInstallerComHandler.exe,LaunchTM.exe,LaunchWinApp.exe,LegacyNetUXHost.exe,LicenseManag
erShellext.exe,licensingdiag.exe,LicensingUI.exe,LocationNotificationWindows.exe,Locator.exe,LockApp
Host.exe,LockScreenContentServer.exe,lodctr.exe,logagent.exe,logman.exe,LogonUI.exe,lpkinstall.exe,l
pksetup.exe,lpremove.exe,LsaIso.exe,lsass.exe,Magnify.exe,makecab.exe,manage-
bde.exe,MbaeParserTask.exe,mblctr.exe,MBR2GPT.EXE,mcbuilder.exe,MDEServer.exe,MDMAgent.exe,MDMAppIns
taller.exe,MdmDiagnosticsTool.exe,MdRes.exe,MdSched.exe,mfpmp.exe,microsoft.windows.softwarelogo.sho
wdesktop.exe,MicrosoftEdgeBCHost.exe,MicrosoftEdgeCP.exe,MicrosoftEdgeDevTools.exe,MicrosoftEdgeSH.e
xe,mitigationscanner.exe,mmc.exe,mmgaserver.exe,mobsync.exe,mountvol.exe,mousocoreworker.exe,mpnotif
y.exe,MpSigStub.exe,MRINFO.EXE,MRT-
KB890830.exe,MRT.exe,MSchedExe.exe,msconfig.exe,msdt.exe,msdtc.exe,msfeedssync.exe,mshta.exe,msiexec
.exe,msinfo32.exe,mspaint.exe,msra.exe,MsSpellCheckingHost.exe,mstsc.exe,mtstocom.exe,MuiUnattend.ex
e,MultiDigiMon.exe,MusNotification.exe,MusNotificationUx.exe,MusNotifyIcon.exe,Narrator.exe,nbtstat.
exe,ndadmin.exe,NDKPing.exe,net.exe,net1.exe,netbtugc.exe,netcfg.exe,NetCfgNotifyObjectHost.exe,NetE
vtFwdr.exe,NetHost.exe,netiougc.exe,Netplwiz.exe,netsh.exe,NETSTAT.EXE,newdev.exe,NgcIso.exe,nltest.
exe,nmbind.exe,nmscrub.exe,notepad.exe,nslookup.exe,ntoskrnl.exe,ntprint.exe,nvspinfo.exe,odbcad32.e
xe,odbcconf.exe,ofdeploy.exe,omadmclient.exe,omadmprc.exe,openfiles.exe,OpenWith.exe,OptionalFeature
```

```
s.exe,osk.exe,pacjsworker.exe,PackagedCWALauncher.exe,PasswordOnWakeSettingFlyout.exe,PATHPING.EXE,p
calua.exe,pcaui.exe,pcwrun.exe,perfmon.exe,phoneactivate.exe,PickerHost.exe,PinEnrollmentBroker.exe,
PING.EXE,PkgMgr.exe,PktMon.exe,plasrv.exe,PnPUnattend.exe,pnputil.exe,poqexec.exe,pospaymentsworker.
exe,powercfg.exe,PresentationHost.exe,prevhost.exe,print.exe,printfilterpipelinesvc.exe,PrintIsolati
onHost.exe,printui.exe,proquota.exe,provlaunch.exe,provtool.exe,ProximityUxHost.exe,prproc.exe,psr.e
xe,pwlauncher.exe,quickassist.exe,rasautou.exe,rasdial.exe,raserver.exe,rasphone.exe,rdpclip.exe,rdp
input.exe,RdpSa.exe,RdpSaProxy.exe,RdpSaUacHelper.exe,rdrleakdiag.exe,ReAgentc.exe,recdisc.exe,recov
er.exe,RecoveryDrive.exe,refsutil.exe,reg.exe,regedt32.exe,regini.exe,Register-
CimProvider.exe,regsvr32.exe,rekeywiz.exe,relog.exe,RelPost.exe,RemotePosWorker.exe,repair-
bde.exe,replace.exe,ResetEngine.exe,resmon.exe,RMActivate.exe,RMActivate_isv.exe,RMActivate_ssp.exe,
RMActivate_ssp_isv.exe,RmClient.exe,rmttpmvscmgrsvr.exe,Robocopy.exe,ROUTE.EXE,RpcPing.exe,rrinstall
er.exe,rstrui.exe,runas.exe,rundll32.exe,runexehelper.exe,RunLegacyCPLElevated.exe,runonce.exe,Runti
meBroker.exe,sc.exe,schtasks.exe,sdbinst.exe,sdchange.exe,sdclt.exe,sdiagnhost.exe,SearchFilterHost.
exe,SearchIndexer.exe,SearchProtocolHost.exe,SecEdit.exe,secinit.exe,securekernel.exe,SecurityHealth
Host.exe,SecurityHealthService.exe,SecurityHealthSystray.exe,SensorDataService.exe,services.exe,sess
ionmsg.exe,sethc.exe,setspn.exe,SettingSyncHost.exe,setupcl.exe,setupugc.exe,setx.exe,sfc.exe,SgrmBr
oker.exe,SgrmLpac.exe,shrpubw.exe,shutdown.exe,sigverif.exe,SIHClient.exe,sihost.exe,SlideToShutDown
.exe,slui.exe,smartscreen.exe,smss.exe,SndVol.exe,SnippingTool.exe,snmptrap.exe,sort.exe,SpaceAgent.
exe,spaceman.exe,SpatialAudioLicenseSrv.exe,Spectrum.exe,spoolsv.exe,SppExtComObj.Exe,sppsvc.exe,srd
elayed.exe,SrTasks.exe,stordiag.exe,subst.exe,svchost.exe,sxstrace.exe,SyncHost.exe,SysResetErr.exe,
systeminfo.exe,SystemPropertiesAdvanced.exe,SystemPropertiesComputerName.exe,SystemPropertiesDataExe
cutionPrevention.exe,SystemPropertiesHardware.exe,SystemPropertiesPerformance.exe,SystemPropertiesPr
otection.exe,SystemPropertiesRemote.exe,systemreset.exe,SystemSettingsAdminFlows.exe,SystemSettingsB
roker.exe,SystemSettingsRemoveDevice.exe,SystemUWPLauncher.exe,systray.exe,tabcal.exe,takeown.exe,Ta
piUnattend.exe,tar.exe,taskhostw.exe,taskkill.exe,tasklist.exe,Taskmgr.exe,tcblaunch.exe,tcmsetup.ex
e,TCPSVCS.EXE,ThumbnailExtractionHost.exe,TieringEngineService.exe,timeout.exe,TokenBrokerCookies.ex
e,TpmInit.exe,tpmvscmgr.exe,tpmvscmgrsvr.exe,tracerpt.exe,TRACERT.EXE,TSTheme.exe,TSWbPrxy.exe,TsWpf
Wrp.exe,ttdinject.exe,tttracer.exe,typeperf.exe,tzsync.exe,tzutil.exe,ucsvc.exe,UIMgrBroker.exe,unlo
dctr.exe,unregmp2.exe,upfc.exe,UpgradeResultsUI.exe,upnpcont.exe,UserAccountBroker.exe,UserAccountCo
ntrolSettings.exe,userinit.exe,UsoClient.exe,usocoreworker.exe,Utilman.exe,VaultCmd.exe,vds.exe,vdsl
dr.exe,verclsid.exe,verifier.exe,verifiergui.exe,VsGraphicsDesktopEngine.exe,VsGraphicsRemoteEngine.
exe,vssadmin.exe,VSSVC.exe,vulkaninfo-1-1-0-65-1.exe,vulkaninfo-1-999-0-0-
0.exe,vulkaninfo.exe,w32tm.exe,WaaSMedicAgent.exe,waitfor.exe,WallpaperHost.exe,wbadmin.exe,wbengine
.exe,wecutil.exe,WerFault.exe,WerFaultSecure.exe,wermgr.exe,wevtutil.exe,wextract.exe,WFS.exe,where.
exe,whoami.exe,wiaacmgr.exe,wiawow64.exe,wifitask.exe,wimserv.exe,WinBioDataModelOOBE.exe,Windows.Me
dia.BackgroundPlayback.exe,Windows.WARP.JITService.exe,WindowsActionDialog.exe,WindowsUpdateElevated
Installer.exe,wininit.exe,winload.exe,winlogon.exe,winresume.exe,winrs.exe,winrshost.exe,WinRTNetMUA
HostServer.exe,WinSAT.exe,winver.exe,wkspbroker.exe,wksprt.exe,wlanext.exe,wlrmdr.exe,WMPDMC.exe,Wor
kFolders.exe,wowreg32.exe,WpcMon.exe,WpcTok.exe,WPDShextAutoplay.exe,wpnpinst.exe,wpr.exe,write.exe,
wscadminui.exe,WSCollect.exe,wscript.exe,wsl.exe,wslconfig.exe,WSManHTTPConfig.exe,wsmprovhost.exe,w
sqmcons.exe,WSReset.exe,wuapihost.exe,wuauclt.exe,WUDFCompanionHost.exe,WUDFHost.exe,wusa.exe,WWAHos
t.exe,XblGameSaveTask.exe,xcopy.exe,xwizard.exe
gi -path C:\Windows\*.exe -exclude
bfsvc.exe,explorer.exe,HelpPane.exe,hh.exe,notepad.exe,regedit.exe,RtkBtManServ.exe,splwow64.exe,win
hlp32.exe,write.exe
gi -path C:\Windows\SysWOW64\*.exe -exclude
appidtel.exe,appverif.exe,ARP.EXE,at.exe,AtBroker.exe,attrib.exe,auditpol.exe,autochk.exe,autoconv.e
xe,autofmt.exe,backgroundTaskHost.exe,BackgroundTransferHost.exe,bitsadmin.exe,bootcfg.exe,bthudtask
.exe,ByteCodeGenerator.exe,cacls.exe,calc.exe,CameraSettingsUIHost.exe,CertEnrollCtrl.exe,certreq.ex
```

```
e,certutil.exe,charmap.exe,CheckNetIsolation.exe,chkdsk.exe,chkntfs.exe,choice.exe,cipher.exe,cleanm
gr.exe,cliconfg.exe,clip.exe,CloudNotifications.exe,cmd.exe,cmdkey.exe,cmdl32.exe,cmmon32.exe,cmstp.
exe,colorcpl.exe,comp.exe,compact.exe,ComputerDefaults.exe,control.exe,convert.exe,CredentialUIBroke
r.exe,credwiz.exe,cscript.exe,ctfmon.exe,cttune.exe,cttunesvr.exe,curl.exe,dccw.exe,dcomcnfg.exe,ddo
diag.exe,DevicePairingWizard.exe,dfrgui.exe,dialer.exe,diskpart.exe,diskperf.exe,Dism.exe,dllhost.ex
e,dllhst3g.exe,doskey.exe,dpapimig.exe,DpiScaling.exe,dplaysvr.exe,dpnsvr.exe,driverquery.exe,dtdump
.exe,dvdplay.exe,DWWIN.EXE,DXCap.exe,DXCpl.exe,dxdiag.exe,EaseOfAccessDialog.exe,edpnotify.exe,efsui
.exe,EhStorAuthn.exe,esentutl.exe,eudcedit.exe,eventcreate.exe,eventvwr.exe,expand.exe,explorer.exe,
extrac32.exe,fc.exe,find.exe,findstr.exe,finger.exe,fixmapi.exe,FlashPlayerApp.exe,fltMC.exe,Fondue.
exe,fontdrvhost.exe,fontview.exe,forfiles.exe,fsquirt.exe,fsutil.exe,ftp.exe,GameBarPresenceWriter.e
xe,GamePanel.exe,getmac.exe,gpresult.exe,gpupdate.exe,grpconv.exe,hdwwiz.exe,help.exe,hh.exe,HOSTNAM
E.EXE,icacls.exe,icsunattend.exe,ieUnatt.exe,iexpress.exe,InfDefaultInstall.exe,InputSwitchToastHand
ler.exe,instnm.exe,ipconfig.exe,iscsicli.exe,iscsicpl.exe,isoburn.exe,ktmutil.exe,label.exe,LaunchTM
.exe,LaunchWinApp.exe,lodctr.exe,logagent.exe,logman.exe,Magnify.exe,makecab.exe,MASetupCleaner.exe,
mcbuilder.exe,mfpmp.exe,mmc.exe,mmgaserver.exe,mobsync.exe,mountvol.exe,MRINFO.EXE,msdt.exe,msfeedss
ync.exe,mshta.exe,msiexec.exe,msinfo32.exe,mspaint.exe,msra.exe,mstsc.exe,mtstocom.exe,MuiUnattend.e
xe,ndadmin.exe,net.exe,net1.exe,netbtugc.exe,NetCfgNotifyObjectHost.exe,netiougc.exe,Netplwiz.exe,ne
tsh.exe,NETSTAT.EXE,newdev.exe,notepad.exe,nslookup.exe,ntprint.exe,odbcad32.exe,odbcconf.exe,OneDri
veSetup.exe,openfiles.exe,OpenWith.exe,OposHost.exe,PackagedCWALauncher.exe,PasswordOnWakeSettingFly
out.exe,PATHPING.EXE,pcaui.exe,perfhost.exe,perfmon.exe,PickerHost.exe,PING.EXE,PkgMgr.exe,poqexec.e
xe,powercfg.exe,PresentationHost.exe,prevhost.exe,print.exe,printui.exe,proquota.exe,provlaunch.exe,
psr.exe,quickassist.exe,rasautou.exe,rasdial.exe,raserver.exe,rasphone.exe,RdpSa.exe,RdpSaProxy.exe,
RdpSaUacHelper.exe,rdrleakdiag.exe,ReAgentc.exe,recover.exe,reg.exe,regedit.exe,regedt32.exe,regini.
exe,Register-
CimProvider.exe,regsvr32.exe,rekeywiz.exe,relog.exe,replace.exe,resmon.exe,RMActivate.exe,RMActivate
_isv.exe,RMActivate_ssp.exe,RMActivate_ssp_isv.exe,RmClient.exe,Robocopy.exe,ROUTE.EXE,RpcPing.exe,r
rinstaller.exe,runas.exe,rundll32.exe,RunLegacyCPLElevated.exe,runonce.exe,sc.exe,schtasks.exe,sdbin
st.exe,sdchange.exe,sdiagnhost.exe,SearchFilterHost.exe,SearchIndexer.exe,SearchProtocolHost.exe,Sec
Edit.exe,secinit.exe,sethc.exe,SettingSyncHost.exe,setup16.exe,setupugc.exe,setx.exe,sfc.exe,shrpubw
.exe,shutdown.exe,SndVol.exe,sort.exe,SpatialAudioLicenseSrv.exe,srdelayed.exe,stordiag.exe,subst.ex
e,svchost.exe,sxstrace.exe,SyncHost.exe,systeminfo.exe,SystemPropertiesAdvanced.exe,SystemProperties
ComputerName.exe,SystemPropertiesDataExecutionPrevention.exe,SystemPropertiesHardware.exe,SystemProp
ertiesPerformance.exe,SystemPropertiesProtection.exe,SystemPropertiesRemote.exe,SystemUWPLauncher.ex
e,systray.exe,takeown.exe,TapiUnattend.exe,tar.exe,taskkill.exe,tasklist.exe,Taskmgr.exe,tcmsetup.ex
e,TCPSVCS.EXE,ThumbnailExtractionHost.exe,timeout.exe,TokenBrokerCookies.exe,TpmInit.exe,tracerpt.ex
e,TRACERT.EXE,TSTheme.exe,TsWpfWrp.exe,ttdinject.exe,tttracer.exe,typeperf.exe,tzutil.exe,UIUMPSrv.e
xe,unlodctr.exe,unregmp2.exe,upnpcont.exe,user.exe,UserAccountBroker.exe,UserAccountControlSettings.
exe,userinit.exe,Utilman.exe,verclsid.exe,verifiergui.exe,VsGraphicsDesktopEngine.exe,VsGraphicsRemo
teEngine.exe,vulkaninfo-1-1-0-65-1.exe,vulkaninfo-1-999-0-0-
0.exe,vulkaninfo.exe,w32tm.exe,waitfor.exe,wecutil.exe,WerFault.exe,WerFaultSecure.exe,wermgr.exe,we
vtutil.exe,wextract.exe,where.exe,whoami.exe,wiaacmgr.exe,Windows.Media.BackgroundPlayback.exe,Windo
ws.WARP.JITService.exe,winrs.exe,winrshost.exe,WinRTNetMUAHostServer.exe,winver.exe,wlanext.exe,wowr
eg32.exe,WPDShextAutoplay.exe,write.exe,wscadminui.exe,wscript.exe,WSManHTTPConfig.exe,wsmprovhost.e
xe,wusa.exe,WWAHost.exe,xcopy.exe,xwizard.exe
```

Locate Possible Emotet Service Binaries (Excludes 32Bit Win7 Common Executables)

```
gi -path C:\Windows\System32\*.exe -exclude
AdapterTroubleshooter.exe,aitagent.exe,aitstatic.exe,alg.exe,append.exe,appidcertstorecheck.exe,appi
dpolicyconverter.exe,ARP.EXE,at.exe,AtBroker.exe,attrib.exe,audiodg.exe,auditpol.exe,autochk.exe,aut
oconv.exe,autofmt.exe,AxInstUI.exe,baaupdate.exe,bcdboot.exe,bcdedit.exe,BdeHdCfg.exe,BdeUISrv.exe,B
deUnlockWizard.exe,BitLockerWizard.exe,BitLockerWizardElev.exe,bitsadmin.exe,bootcfg.exe,bridgeunatt
end.exe,bthudtask.exe,cacls.exe,calc.exe,CertEnrollCtrl.exe,certreq.exe,certutil.exe,change.exe,char
map.exe,chglogon.exe,chgport.exe,chgusr.exe,chkdsk.exe,chkntfs.exe,choice.exe,cipher.exe,cleanmgr.ex
e,cliconfg.exe,clip.exe,cmd.exe,cmdkey.exe,cmdl32.exe,cmmon32.exe,cmstp.exe,cofire.exe,colorcpl.exe,
comp.exe,compact.exe,CompatTelRunner.exe,CompMgmtLauncher.exe,ComputerDefaults.exe,conhost.exe,conse
nt.exe,control.exe,convert.exe,credwiz.exe,cscript.exe,csrss.exe,csrstub.exe,ctfmon.exe,cttune.exe,c
ttunesvr.exe,dccw.exe,dcomcnfg.exe,ddodiag.exe,debug.exe,Defrag.exe,DeviceDisplayObjectProvider.exe,
DeviceEject.exe,DevicePairingWizard.exe,DeviceProperties.exe,DFDWiz.exe,dfrgui.exe,dialer.exe,diantz
.exe,dinotify.exe,diskpart.exe,diskperf.exe,diskraid.exe,Dism.exe,dispdiag.exe,DisplaySwitch.exe,djo
in.exe,dllhost.exe,dllhst3g.exe,dnscacheugc.exe,doskey.exe,dosx.exe,dpapimig.exe,DpiScaling.exe,dpla
ysvr.exe,dpnsvr.exe,driverquery.exe,drvinst.exe,DRWATSON.EXE,dvdplay.exe,dvdupgrd.exe,dwm.exe,DWWIN.
EXE,dxdiag.exe,Dxpserver.exe,Eap3Host.exe,edlin.exe,efsui.exe,EhStorAuthn.exe,esentutl.exe,eudcedit.
exe,eventcreate.exe,eventvwr.exe,exe2bin.exe,expand.exe,extrac32.exe,fastopen.exe,fc.exe,find.exe,fi
ndstr.exe,finger.exe,fixmapi.exe,fltMC.exe,fontview.exe,forfiles.exe,fsquirt.exe,fsutil.exe,ftp.exe,
fvenotify.exe,fveprompt.exe,FXSCOVER.exe,FXSSVC.exe,FXSUNATD.exe,GDI.EXE,getmac.exe,GettingStarted.e
xe,gpresult.exe,gpscript.exe,gpupdate.exe,grpconv.exe,hdwwiz.exe,help.exe,HOSTNAME.EXE,hwrcomp.exe,h
wrreg.exe,icacls.exe,icardagt.exe,icsunattend.exe,ie4uinit.exe,ieUnatt.exe,iexpress.exe,InfDefaultIn
stall.exe,ipconfig.exe,irftp.exe,iscsicli.exe,iscsicpl.exe,isoburn.exe,klist.exe,krnl386.exe,ksetup.
exe,ktmutil.exe,label.exe,LocationNotifications.exe,Locator.exe,lodctr.exe,logagent.exe,logman.exe,l
ogoff.exe,LogonUI.exe,lpksetup.exe,lpremove.exe,lsass.exe,lsm.exe,Magnify.exe,makecab.exe,manage-
bde.exe,mblctr.exe,mcbuilder.exe,mctadmin.exe,MdRes.exe,MdSched.exe,mem.exe,mfpmp.exe,MigAutoPlay.ex
e,mmc.exe,mobsync.exe,mountvol.exe,mpnotify.exe,MpSigStub.exe,MRINFO.EXE,MRT.exe,mscdexnt.exe,msconf
ig.exe,msdt.exe,msdtc.exe,msfeedssync.exe,msg.exe,mshta.exe,msiexec.exe,msinfo32.exe,mspaint.exe,msr
a.exe,mstsc.exe,mtstocom.exe,MuiUnattend.exe,MultiDigiMon.exe,NAPSTAT.EXE,Narrator.exe,nbtstat.exe,n
dadmin.exe,net.exe,net1.exe,netbtugc.exe,netcfg.exe,netiougc.exe,Netplwiz.exe,NetProj.exe,netsh.exe,
NETSTAT.EXE,newdev.exe,nlsfunc.exe,nltest.exe,notepad.exe,nslookup.exe,ntkrnlpa.exe,ntoskrnl.exe,ntp
rint.exe,ntvdm.exe,ocsetup.exe,odbcad32.exe,odbcconf.exe,openfiles.exe,OptionalFeatures.exe,osk.exe,
OxpsConverter.exe,p2phost.exe,PATHPING.EXE,pcalua.exe,pcaui.exe,pcawrk.exe,pcwrun.exe,perfmon.exe,PI
NG.EXE,PkgMgr.exe,plasrv.exe,PnPUnattend.exe,PnPutil.exe,poqexec.exe,powercfg.exe,PresentationHost.e
xe,PresentationSettings.exe,prevhost.exe,print.exe,PrintBrmUi.exe,printfilterpipelinesvc.exe,PrintIs
olationHost.exe,printui.exe,proquota.exe,psr.exe,PushPrinterConnections.exe,qappsrv.exe,qprocess.exe
,query.exe,quser.exe,qwinsta.exe,rasautou.exe,rasdial.exe,raserver.exe,rasphone.exe,rdpclip.exe,rdpi
nit.exe,rdpshell.exe,rdpsign.exe,rdrleakdiag.exe,rdrmemptylst.exe,RDVGHelper.exe,ReAgentc.exe,recdis
c.exe,recover.exe,redir.exe,reg.exe,regedt32.exe,regini.exe,RegisterIEPKEYs.exe,regsvr32.exe,rekeywi
z.exe,relog.exe,RelPost.exe,repair-
bde.exe,replace.exe,reset.exe,resmon.exe,RMActivate.exe,RMActivate_isv.exe,RMActivate_ssp.exe,RMActi
vate_ssp_isv.exe,RmClient.exe,Robocopy.exe,ROUTE.EXE,RpcPing.exe,rrinstaller.exe,rstrui.exe,runas.ex
e,rundll32.exe,RunLegacyCPLElevated.exe,runonce.exe,rwinsta.exe,sbunattend.exe,sc.exe,schtasks.exe,s
dbinst.exe,sdchange.exe,sdclt.exe,sdiagnhost.exe,SearchFilterHost.exe,SearchIndexer.exe,SearchProtoc
olHost.exe,SecEdit.exe,secinit.exe,services.exe,sethc.exe,SetIEInstalledDate.exe,setspn.exe,setupcl.
exe,setupSNK.exe,setupugc.exe,setver.exe,setx.exe,sfc.exe,shadow.exe,share.exe,shrpubw.exe,shutdown.
exe,sigverif.exe,slui.exe,smss.exe,SndVol.exe,SnippingTool.exe,snmptrap.exe,sort.exe,SoundRecorder.e
xe,spinstall.exe,spoolsv.exe,sppsvc.exe,spreview.exe,srdelayed.exe,StikyNot.exe,subst.exe,svchost.ex
e,sxstrace.exe,SyncHost.exe,sysedit.exe,syskey.exe,systeminfo.exe,SystemPropertiesAdvanced.exe,Syste
mPropertiesComputerName.exe,SystemPropertiesDataExecutionPrevention.exe,SystemPropertiesHardware.exe
```

```
,SystemPropertiesPerformance.exe,SystemPropertiesProtection.exe,SystemPropertiesRemote.exe,systray.e
xe,tabcal.exe,takeown.exe,TapiUnattend.exe,taskeng.exe,taskhost.exe,taskkill.exe,tasklist.exe,taskmg
r.exe,tcmsetup.exe,TCPSVCS.EXE,timeout.exe,TpmInit.exe,tracerpt.exe,TRACERT.EXE,tscon.exe,tsdiscon.e
xe,tskill.exe,TSTheme.exe,TsUsbRedirectionGroupPolicyControl.exe,TSWbPrxy.exe,TsWpfWrp.exe,typeperf.
exe,tzutil.exe,ucsvc.exe,UI0Detect.exe,unlodctr.exe,unregmp2.exe,upnpcont.exe,USER.EXE,UserAccountCo
ntrolSettings.exe,userinit.exe,Utilman.exe,VaultCmd.exe,VaultSysUi.exe,VBoxControl.exe,VBoxService.e
xe,VBoxTray.exe,vds.exe,vdsldr.exe,verclsid.exe,verifier.exe,vmicsvc.exe,vssadmin.exe,VSSVC.exe,w32t
m.exe,waitfor.exe,wbadmin.exe,wbengine.exe,wecutil.exe,WerFault.exe,WerFaultSecure.exe,wermgr.exe,we
vtutil.exe,wextract.exe,WFS.exe,where.exe,whoami.exe,wiaacmgr.exe,wimserv.exe,wininit.exe,winload.ex
e,winlogon.exe,winresume.exe,winrs.exe,winrshost.exe,WinSAT.exe,WINSPOOL.EXE,winver.exe,wisptis.exe,
wksprt.exe,wlanext.exe,wlrmdr.exe,WOWDEB.EXE,WOWEXEC.EXE,WPDShextAutoplay.exe,wpnpinst.exe,write.exe
,wscript.exe,WSManHTTPConfig.exe,wsmprovhost.exe,wsqmcons.exe,wuapp.exe,wuauclt.exe,WUDFHost.exe,wus
a.exe,xcopy.exe,xpsrchvw.exe,xwizard.exe
gi -path C:\Windows\*.exe -exclude
bfsvc.exe,explorer.exe,fveupdate.exe,HelpPane.exe,hh.exe,notepad.exe,regedit.exe,Sysmon.exe,twunk_16
.exe,twunk_32.exe,winhelp.exe,winhlp32.exe,write.exe
```

Determine if user Trusted a doc and ran a macro

Note: Don’t forget to load in user hive.

```
reg query 'HKU\[SID]\Software\Microsoft\Office\[versionnumber]\Word\Security\Trusted
Documents\TrustRecords'
```

Note: This will show the file name/location and metadata in Hex. If the last lot of hex is FFFFFF7F then the user enabled the macro.

Locate Possible DLL Search Order Hijacking (https://attack.mitre.org/techniques/T1038/)

Note: A legitimate clean executable can be used to run malicious DLLs based on how the software searches for them.

More information on Microsoft Docs (https://docs.microsoft.com/en-us/windows/win32/dlls/dynamic-link-library-search-order)

```
reg query "HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\KnownDLLs"
reg query "HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\SafeDllSearchMode"
```

Search order for desktop applications:

If SafeDllSearchMode is enabled (is by default), the search order is as follows:

- The same directory from which the executable is run.
- The System Directory (Usually C:\Windows\System32).
- The 16-bit System Directory.
- The Windows Directory (Usually C:\Windows).
- The Current Directory (From the process which executed the executable).
- The directories that are listed in the PATH environment variable.

If SafeDllSearchMode is disabled (SafeDllSearchMode has a reg value of 0), the search order is as follows:

- The same directory from which the executable is run.
- The Current Directory (From the process which executed the executable).
- The System Directory (Usually C:\Windows\System32).
- The 16-bit System Directory.
- The Windows Directory (Usually C:\Windows).
- The directories that are listed in the PATH environment variable.

Locate Possible Dll Side Loading (https://attack.mitre.org/techniques/T1073/)

Note: A legitimate clean executable can be used to run malicious DLLs based on issues with a manifest file used by the application to load DLLs.

```
reg query "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\SideBySide\Winners"
```

By placing a malicious DLL in the below locations legitimate binaries may have been used to sideload these malicious DLLs.

- C:\Windows\WinSxS
- C:\Windows\SXS

Unique Sideload DLL hashes (may take some time)

```
(gci -path C:\Windows\WinSxS -recurse -include *.dll|gi -ea SilentlyContinue|filehash).hash|sort -u
```

Unsigned or Invalid Sideload DLLs (there will be a lot)

```
gci -path C:\Windows\WinSxS -recurse -include *.dll | Get-AuthenticodeSignature | Where-Object
Status -NE "Valid"
```

Unsigned Sideload DLLs (Less noise)

```
gci -path C:\Windows\WinSxS -recurse -include *.dll | Get-AuthenticodeSignature | Where-Object Status -E "NotSigned"
gci -path C:\Windows\WinSxS -recurse -include *.ocx | Get-AuthenticodeSignature | Where-Object Status -NE "Valid"
```

Hash of Unsigned Sideload DLLs

```
gci -path C:\Windows\WinSxS -recurse -include *.dll | Get-AuthenticodeSignature | Where-Object Status -E "NotSigned" | Select Path | gi -ea SilentlyContinue | filehash | sort -u
gci -path C:\Windows\WinSxS -recurse -include *.ocx | Get-AuthenticodeSignature | Where-Object Status -NE "Valid" | Select Path | gi -ea SilentlyContinue | filehash | sort -u
```

Find files without extensions

```
Get-ChildItem -Path C:\Users\[user]\AppData -Recurse -Exclude *.* -File -Force -ea SilentlyContinue
```

Remediate malicious files

```
rmdir %localappdata%\maliciousdirectory\ /s
del /F %localappdata%\maliciousdirectory\malware.exe
```

Powershell:

```
Remove-Item [C:\Users\Public\*.exe]
Remove-Item -Path [C:\Users\Public\malware.exe] -Force
Get-ChildItem * -Include *.exe -Recurse | Remove-Item
```

Detect Persistent WMI Subscriptions

```
Get-WmiObject -Class __FilterToConsumerBinding -Namespace root\subscription
Get-WmiObject -Class __EventFilter -Namespace root\subscription
Get-WmiObject -Class __EventConsumer -Namespace root\subscription
```

Remediate Persistent WMI Subscriptions

```
Get-WMIObject -Namespace root\subscription -Class __EventFilter -Filter "Name='[Name]'" | Remove-WmiObject
Get-WMIObject -Namespace root\subscription -Class CommandLineEventConsumer -Filter "Name='[Name]'" | Remove-WmiObject
Get-WMIObject -Namespace root\subscription -Class __FilterToConsumerBinding -Filter "__Path like '%[Name]%'" | Remove-WmiObject
```

Enumerate WMI Namespaces (https://learn-powershell.net/2014/05/09/quick-hits-list-all-available-wmi-namespaces-using-powershell/)

```
Function Get-WmiNamespace ($Path = 'root')
{
        foreach ($Namespace in (Get-WmiObject -Namespace $Path -Class __Namespace))
        {
                $FullPath = $Path + "/" + $Namespace.Name
                Write-Output $FullPath
                Get-WmiNamespace -Path $FullPath
        }
}
Get-WMINamespace -Recurse
```

Mimikatz Detection

The below represent registry keys which make it more difficult for Mimikatz to work. Modification of these keys may indicate an attacker trying to execute Mimikatz within an environment if they were set to their more secure state. Always test prior to changing registry keys such as these in a production environment to ensure nothing breaks.

```
HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest
        - “UseLogonCredential” should be 0 to prevent the password in LSASS
HKLM\SYSTEM\CurrentControlSet\Control\Lsa
        - “RunAsPPL” should be set to dword:00000001 to enable LSA Protection which prevents non-
protected processes from interacting with LSASS.
        - Mimikatz can remove these flags using a custom driver called mimidriver.
                - This uses the command **!+** and then **!processprotect /remove
/process:lsass.exe** by default so tampering of this registry key can be indicative of Mimikatz
activity.
```

The Mimikatz Yara rule (https://github.com/gentilkiwi/mimikatz/blob/master/kiwi_passwords.yar) may also prove useful.

EventLog Tampering Detection

```
reg query HKLM\SYSTEM\CurrentControlSet\Services\EventLog\ /s /v File
reg query HKLM\SYSTEM\CurrentControlSet\Services\EventLog\ /s /v MaxSize
reg query HKLM\SYSTEM\CurrentControlSet\Services\EventLog\ /s /v Retention
sc.exe query eventlog
```

DanderSpritz eventlogedit (https://github.com/fox-it/danderspritz-evtx)

Putty Detection

```
reg query HKCU\Software\SimonTatham\PuTTY\Sessions /s
```

Installed Updates

(WMI Quick Fix Engineering)

```
wmic qfe
```

Installed Software/Packages

```
reg query HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\ /s | findstr
"DisplayName"
reg query HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall\ /s | findstr "DisplayName"
wmic product get name,version /format:csv
wmic product get /ALL
dism /online /get-packages
```

Powershell: Full List for all users using uninstall keys in registry

```
$(Get-ItemProperty HKLM:\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\*; Get-
ItemProperty HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\*;New-PSDrive -Name HKU -
PSProvider Registry -Root Registry::HKEY_USERS| Out-Null;$UserInstalls += gci -Path HKU: | where
{$_.Name -match 'S-\d-\d+-(\d+-){1,14}\d+$'} | foreach {$_.PSChildName };$(foreach ($User in
$UserInstalls){Get-ItemProperty
HKU:\$User\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\*});$UserInstalls = $null;try{Remove-
PSDrive -Name HKU}catch{};)|where {($_.DisplayName -ne $null) -and ($_.Publisher -ne $null)} |
Select DisplayName,DisplayVersion,Publisher,InstallDate,UninstallString |FT
```

Process information

(pslist requires sysinternals pslist.exe):

```
wmic process list full /format:csv
wmic process get name,parentprocessid,processid /format:csv
wmic process get ExecutablePath,processid /format:csv
wmic process get name,ExecutablePath,processid,parentprocessid /format:csv | findstr /I "appdata"
wmic process where processid=[PID] get parentprocessid
wmic process where processid=[PID] get commandline
wmic process where "commandline is not null and commandline!=''" get name,commandline /format:csv
Get-WmiObject win32_process -Filter "name like '%powershell.exe'" | select processId,commandline|FL
pslist
```

Scan for malware with Windows Defender

```
"%ProgramFiles%\Windows Defender\MpCmdRun.exe" -Scan -ScanType 1
"%ProgramFiles%\Windows Defender\MpCmdRun.exe" -Scan -ScanType 2
"%ProgramFiles%\Windows Defender\MpCmdRun.exe" -Scan -ScanType 3 -File C:\Users\
[username]\AppData\Local\Temp
```

Note: Types are as follows

- 1: Quick scan
- 2: Full system scan
- 3: File and directory custom scan

Check Windows Defender for excluded files

```
reg query "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Defender\Exclusions" /s
Get-ChildItem 'HKLM:\SOFTWARE\Microsoft\Windows Defender\Exclusions'
```

Delete Windows Defender excluded files

```
reg delete "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Defender\Exclusions\Paths" /v "
[RegkeyValue]"
reg delete "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows Defender\Exclusions\Paths"
Remove-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\Windows Defender\Exclusions\Paths' -Name "Paths"
```

Check and Set Access Control Lists

```
Get-Acl -Path 'HKLM:\SOFTWARE\Microsoft\Windows Defender\Exclusions\Paths'|FL
Get-Acl -Path [FileWithRequiredAccess] | Set-Acl -Path 'HKLM:\SOFTWARE\Microsoft\Windows
Defender\Exclusions\Paths'
```

Check Security Descriptor Definition Language (SDDL) and Access Control Entries (ACE) for services

```
sc sdshow <servicename>
$A=get-service;foreach ($service in $A){$service;sc.exe sdshow $service.Name}
$A=get-service;foreach ($service in $A){$service;sc.exe sdshow $service.Name|Select-String "A;*DC"}
$A=get-service;foreach ($service in $A){$service;sc.exe sdshow $service.Name|Select-String "A;*WD"}
$A=get-service;foreach ($service in $A){$service;sc.exe sdshow $service.Name|Select-String "A;*WO"}
```

More information on ACE Strings (https://docs.microsoft.com/en-us/windows/win32/secauthz/ace-strings) and the level of access they can provide.

Obtain hash for all running executables

Issues with spaces in names but supports CMD.exe

```
FOR /F %i IN ('wmic process where "ExecutablePath is not null" get ExecutablePath') DO certutil -hashfile %i SHA256 | findstr -v : >> output.txt
```

Powershell (Special thanks Lee Holmes)

```
(gps|gi -ea SilentlyContinue|filehash).hash|sort -u
```

My less efficient powershell

```
foreach ($process in Get-WmiObject win32_process | where {$_.ExecutablePath -notlike ""}) {Get-FileHash $process.ExecutablePath | Format-List}

foreach ($process in Get-WmiObject win32_process | where {$_.ExecutablePath -notlike ""}) {Get-FileHash $process.ExecutablePath | select Hash -ExpandProperty Hash}

$A = $( foreach ($process in Get-WmiObject win32_process | where {$_.ExecutablePath -notlike ""}) {Get-FileHash $process.ExecutablePath | select Hash -ExpandProperty Hash}) |Sort-Object| Get-Unique;$A
```

Obtain hash and network connections for running executables

```
Get-NetTCPConnection -State Established|? RemoteAddress -NotLike "127.*"| Select RemoteAddress, RemotePort, OwningProcess, @{n="Path";e={(gps -Id $_.OwningProcess).Path}},@{n="Hash";e={(gps -Id $_.OwningProcess|gi|filehash).hash}}, @{n="User";e={(gps -Id $_.OwningProcess -IncludeUserName).UserName}}|sort|gu -AS|FT
```

Obtain hash of DLLs currently loaded by processes

```
$A = $(foreach ($dll in gps|select -ExpandProperty modules -ea SilentlyContinue|? FileName -NotLike "C:\Windows\SYSTEM32\*"){Get-FileHash $dll.FileName| select Hash -ExpandProperty Hash})|Sort-Object| Get-Unique;$A

$A = $(foreach ($dll in gps|select -ExpandProperty modules -ea SilentlyContinue){Get-FileHash $dll.FileName| select Hash -ExpandProperty Hash})|Sort-Object| Get-Unique;$A
```

Obtain processes running which are running a DLL

```
$A=(gps|select -ExpandProperty modules -ea SilentlyContinue | where {$_.ModuleName -Like
'sechost.dll' -or $_.ModuleName -Like 'ntdll.dll'} | sort -u);if($A[0].Size -ge -1) {foreach
($Module in $A){tasklist /m $Module.ModuleName}};
```

Obtain hash of unsigned or invalid DLLs currently loaded by processes

```
 $A=$(foreach ($dll in gps|select -ExpandProperty modules -ea SilentlyContinue){Get-
AuthenticodeSignature $dll.FileName |Where-Object Status -NE "Valid"|Select Path});$B=$(foreach
($dll in $A){Get-FileHash $dll.Path| select Hash -ExpandProperty Hash})|Sort-Object| Get-Unique;$B
```

Obtain DLL information ListDLLs (https://docs.microsoft.com/en-us/sysinternals/downloads/listdlls)

```
listdlls [-r] [-v | -u] [processname|pid]
listdlls [-r] [-v] [-d dllname]
```

Obtain unsigned DLL information loaded by processes

```
listdlls -u
```

Obtain DLLs in use by processes

```
listdlls -v processname -accepteula
listdlls -v -d dllname.dll -accepteula
listdlls -v PID -accepteula
```

Determine handles on a file

```
handle [[-a] [-u] | [-c <handle> [-l] [-y]] | [-s]] [-p <processname>|<pid>> [name]
handle -a -u -s -p exp
handle windows\system
```

Verify EternalBlue Patch (MS17-010) is installed - Microsoft (https://support.microsoft.com/en-us/help/4023262/how-to-verify-that-ms17-010-is-installed)

Note: This impacts the SMB 1.0 Server Driver, if you don’t have the below, then it’s not installed. If you do you can use the above to determine patch level.

```
get-item C:\Windows\system32\drivers\srv.sys | FL VersionInfo
get-hotfix -id KB<111111>
```

Obtain TXT records from recently resolved domains

```
foreach ($domains in Get-DnsClientCache){Resolve-DnsName $domains.Entry -Type "TXT"|Select Strings|?
Strings -NotLike ""};
```

Check all Appdata files for unsigned or invalid executables

```
Get-ChildItem -Recurse $env:APPDATA\..\*.exe -ea SilentlyContinue| ForEach-object {Get-
AuthenticodeSignature $_ -ea SilentlyContinue} | Where-Object {$_.status -ine "Valid"}|Select
Status,Path
```

Investigate WMI Usage

Note: Requires Strings (https://docs.microsoft.com/en-us/sysinternals/downloads/strings)

```
strings -q C:\windows\system32\wbem\repository\objects.data
```

Find executables and scripts in Path directories ($env:Path)

```
Get-Command * -Type Application | FT -AutoSize
Get-Command -Name * | FL FileVersionInfo
```

Find files created/written based on date date

```
Get-ChildItem C:\ -recurse -ea SilentlyContinue -force | where-object { $_.CreationTime.Date -match
"12/25/2014"}
Get-ChildItem C:\ -recurse -ea SilentlyContinue -force | where-object { $_.LastWriteTime -match
"12/25/2014"}
Get-ChildItem C:\ -recurse -ea SilentlyContinue -force | where-object { $_.CreationTime.Hour -gt 2 -
and $_.CreationTime.Hour -lt 15}
```

Check running executables for malware via VirusTotal

Note: VT Has a rate limit for the Public API so this won’t work if you are using the Public API. All 1 liners require VTAPIKey to be set as your VirusTotal API key

```
foreach ($process in Get-WmiObject win32_process | where {$_.ExecutablePath -notlike ""}) {Invoke-
RestMethod -Method 'POST' -Uri 'https://www.virustotal.com/vtapi/v2/file/report' -Body @{ resource =
(Get-FileHash $process.ExecutablePath | select Hash -ExpandProperty Hash); apikey = "[VTAPIKey]"}}
```

This query uses a 15 second timeout to ensure only 4 queries are submitted a minute

```
foreach ($process in Get-WmiObject win32_process | where {$_.ExecutablePath -notlike ""}) {Invoke-
RestMethod -Method 'POST' -Uri 'https://www.virustotal.com/vtapi/v2/file/report' -Body @{ resource =
(Get-FileHash $process.ExecutablePath | select Hash -ExpandProperty Hash); apikey = "
[VTAPIKey]"};Start-Sleep -Seconds 15;}
```

This query uses a 15 second timeout to ensure only 4 queries are submitted a minute and only unique hashes are queried

```
$A = $( foreach ($process in Get-WmiObject win32_process | where {$_.ExecutablePath -notlike ""})
{Get-FileHash $process.ExecutablePath | select Hash -ExpandProperty Hash}) |Sort-Object| Get-Unique
-AsString; foreach ($process in $A) {Invoke-RestMethod -Method 'POST' -Uri
'https://www.virustotal.com/vtapi/v2/file/report' -Body @{ resource =($process); apikey = "
[VTAPIKey]"};Start-Sleep -Seconds 15;}
```

Scan systems for IOA/IOC (Yara)

Loki Scanner (https://github.com/Neo23x0/Loki)

```
loki-upgrader.exe
loki.exe -p [Directory]
```

Crowdresponse Scanner (https://www.crowdstrike.com/resources/community-tools/crowdresponse/)

```
CrowdResponse -v -i config.txt -o out.xml
```

IREC Tactical (https://binalyze.com/download/irec/)

```
IREC.exe --triage-memory
IREC.exe -ad "\\MACHINE\IREC-DIR" --triage-ruleset MyYaraRules --triage-memory
```

Yara (https://github.com/virustotal/yara/releases/latest)

```
yara32.exe -d filename=[file defined in ruleset.yar] [ruleset.yar] [file to scan]
yara32.exe -d filename=[svchost.exe] [ruleset.yar] -r [directory to scan]
```

Yara Linux

```
yara rule.yara malware.exe -s
```

Kill malicious process

```
wmic process where name="malware.exe" call terminate
wmic process where processid=[PID] delete
taskkill /IM malware.exe
taskkill /PID [PID] /T
```

Note: Call terminate allows you to specify an exit status in terms of a signed integer or a quoted negative value. Both methods essentially function the same by calling TerminateProcess.

Dump full process memory

(procdump requires systinternals procdump.exe)

```
procdump -ma [processID]
```

Network connections

(tcpvcon requires sysintenals tcpvcon.exe):

```
ipconfig /all
netstat –anob
netstat -ano
Tcpvcon -a
```

Routing table and ARP cache

```
route print
arp -a
```

Contents of DNS revolver

(useful for recent web history)

```
ipconfig /displaydns
```

Latest system activities

(requires Nirsoft’s LastActivityView)

```
LastActivityView.exe /shtml "LastActivityView.html"
```

Driver information

```
wmic sysdriver list brief /format:csv
driverquery
driverquery /FO list /v
driverquery /si
wmic sysdriver list full
```

Process and extra information

```
tasklist /m
tasklist /m /fi "pid eq [PID]"
tasklist /svc
wmic process where processid=[PID] get commandline
tasklist /v
```

Hosts file and service>port mapping

```
type %SystemRoot%\System32\drivers\etc\hosts
type %SystemRoot%\System32\drivers\etc\services
```

Recycle Bin Forensics

- Named as $I = Metadata of file (Info)
- Named as $R = The file contents itself (Recovery)
- Located at %SystemRoot%\..\$Recycle.Bin in win vista and later commonly (C:$Recycle.Bin)
- Use dir /a via cmd to show recycle bin SID folders and files

DCOM Information

```
wmic dcomapp get /all /format:List
```

Service Information

(psservice requires sysinternals psservice.exe):

```
wmic service list full
net start
sc query
wmic loadorder
psservice
```

Stop and disable/delete malicious service

```
net stop [servicename]
sc config [servicename] start= disabled
sc delete [servicename]
```

cmd history

```
doskey /history
```

Linux Subsystem for Windows 10 may have history in a location such as:

```
C:\Users\
[User]\AppData\Local\Packages\CanonicalGroupLimited.UbuntuonWindows_79rhkp1fndgsc\LocalState\rootfs\
home\[user]
```

Files greater than a 10mb

```
FOR /R C:\ %i in (*) do @if %~zi gtr 10000000 echo %i %~zi
```

Temp files greater than 10mb

```
FOR /R C:\Users\[User]\AppData %i in (*) do @if %~zi gtr 10000000 echo %i %~zi
```

Locate process handles (e.g. files open by process)

Note: Requires handles/handles64.exe from sysinternals

```
handle64.exe -p [PID/name] -nobanner
handle64.exe -a -p [PID/name] -nobanner
handle64.exe -a -l -p [PID/name] -nobanner
handle64.exe -a -l -u -p keepass -nobanner
```

Close process handles (e.g. files open by process)

Note: Requires handles/handles64.exe from sysinternals

```
handle64.exe -c [hexhandleref] -p [PID] -nobanner
handle64.exe -c [hexhandleref] -y -p [PID] -nobanner
```

Event logs between a timeframe

This tool is useful for gathering all windows events within a given timeframe: Event Finder2 (https://github.com/BeanBagKing/EventFinder2)

Check audit policies

```
auditpol /get /category:*
```

Set logging on all success/failure events

(WARNING THIS WILL PRODUCE A LOT OF NOISE, TAILOR TO YOUR NEEDS)

```
auditpol /set /category:* /success:enable /failure:enable
```

Check for Windows Security Logging Bypass

Special thanks to Grzegorz Tworek - 0gtweet (https://twitter.com/0gtweet/status/1182516740955226112)

```
reg query HKLM\System\CurrentControlSet\Control\MiniNt
```

Check group policies

```
gpresult /Z /SCOPE COMPUTER
gpresult /Z /SCOPE USER
gpresult /R /SCOPE COMPUTER
gpresult /R /SCOPE USER
gpresult /r /z
ls C:\Users\[username]\AppData\Local\GroupPolicy\DataStore
ls C:\Windows\system32\GroupPolicy\DataStore
```

Obtain mode settings for ports

```
mode
```

Event Logs for offline analysis

Event logs can be found: %SystemRoot%\System32\winevt\Logs

```
wevtutil epl System [Location]\System.evtx
wevtutil epl Security [Location]\Security.evtx
wevtutil epl Application [Location]\Application.evtx
wevtutil epl "Windows PowerShell" [Location]\Powershell.evtx
```

Powershell, export to CSV (Note: has issues):

```
wevtutil el | ForEach-Object { Get-EventLog -Log "$_" | Export-Csv -Path [Location]\EventExport.csv
-Append}
```

Copy all event logs:

```
XCOPY C:\Windows\System32\winevt\Logs [Location] /i
```

Timeline Windows Event Logs.

An easy way to explore Windows event logs is to dump them into a normalized csv format using EvtxExplorer.

EvtxExplorer: (https://ericzimmerman.github.io/#!index.md)

```
EvtxECmd.exe -d "C:\Windows\System32\winevt\Logs" --csv C:\ --csvf AllEvtx.csv
```

From here you can analyse the CSV using Timeline explorer to view relevant information and group by MAPs.

TimelineExplorer: (https://ericzimmerman.github.io/#!index.md)

Common IIS logs can often be found in the below locations:

- %SystemDrive%\inetpub\logs\LogFiles
- %SystemRoot%\System32\LogFiles\W3SVC1
- %SystemDrive%\inetpub\logs\LogFiles\W3SVC1
  - Note: replace 1 with the number for your IIS website ID
- %SystemDrive%\Windows\System32\LogFiles\HTTPERR

Common Apache logs can often be found in the below locations:

- /var/log
- /var/log/httpd/access.log
- /var/log/apache/access.log
- /var/log/apache2/access.log
- /var/log/httpd-access.log

Other logs can be found in the below, often using the Event Trace Log (ETL) format:

- C:\Windows\System32\LogFiles
- C:\Windows\Panther

ETL format can be parsed using tracerpt which is included in Windows, some examples below.

```
tracerpt C:\Windows\System32\LogFiles\WMI\Terminal-Services-RPC-Client.etl
tracerpt logfile1.etl logfile2.etl -o logdump.xml -of XML
tracerpt logfile.etl -o logdmp.xml -of XML -lr -summary logdmp.txt -report logrpt.xml
tracerpt logfile1.etl logfile2.etl -o -report
tracerpt logfile.etl counterfile.blg -report logrpt.xml -df schema.xml
tracerpt -rt "NT Kernel Logger" -o logfile.csv -of CSV
```

Software specific logs are often stored in readable formats at any of the following locations.

```
%AppData%\[softwarename] (e.g. C:\Users\[username]\AppData\Roaming\[softwarename]\)
%LocalAppData%\[softwarename] (e.g. C:\Users\[username]\AppData\Local\[softwarename]\)
%programfiles%\[softwarename] (e.g. C:\Program Files\[softwarename]\)
%programfiles(x86)%\[softwarename] (e.g. C:\Program Files (x86)\[softwarename]\)
```

You may also find useful memory crashdumps at the below:

```
C:\Users\[username]\AppData\Local\CrashDumps
C:\Users\[username]\AppData\Local\Microsoft\Windows\WER\
```

Security log information

Note: Logs and their event codes have changed over time. Most of the references here are for Windows Vista and Server 2008 onwards rather than Windows 2000,XP,Server 2003. More information on them may be added in the future if required.

(psloglist requires psloglist.exe from systinternals):

```
wevtutil qe security /f:text
eventquery.vbs /L security
wevtutil qe security /f:text | Select-String -Pattern "Event ID: [EventCode]" -Context 2,20
wevtutil qe security /f:text | Select-String -Pattern "Event ID: [EventCode]" -Context 2,20 |
findstr "Account Name:"
psloglist -s -x security
```

Note: Some suspicious events - “Event log service was stopped”, “Windows File Protection is not active on this system”, “The MS Telnet Service has started successfully”

- Security: 4720 (Account created)
- Security: 4722 (Account enabled)
- Security: 4724 (Password reset)
- Security: 4723 (User changed password)
- Security: 4736 (Account deleted)
- Security: 4781 (Account renamed)
- Security: 4738 (User account change)
- Security: 4688 (A new process has been created)
- Security: 4732 (Account added to a group)
- Security: 4733 (Account removed from a group)
- Security: 1102 (Audit log cleared)
- Security: 4614 (Security System Extension)
- Security: 4672 (Special privileges assigned to new logon)
- Security: 4624 (Account successfully logged on)
- Security: 4698 (Scheduled Task Creation)
- Security: 4702 (Scheduled Task Modified)
- Security: 4699 (Scheduled Task Deleted)
- Security: 4701 (Scheduled Task Disabled)
- Security: 4700 (Scheduled Task Enabled)
- Security: 4697 (Service Installation)
- Security: 4625 (Account failed to log on)
- Security: 4776 (The domain controller attempted to validate credentials for an account)
- Security: 4634 (Account successfully logged off)
- Security: 4740 (A user account was locked out)
- Security: 4767 (A user account was unlocked)
- Security: 4778 (Remote Desktop session reconnected)
- Security: 4779 (Remote desktop session disconnected)
- Security: 4625 (A user account failed to log on)
- Security: 4648 (A logon was attempted using explicit credentials)

- Security: 4768 (A Kerberos authentication ticket (TGT) was requested)
    - 0x6 (The username doesn’t exist) - Bad username or not yet replicated to DC
    - 0xC (Start time is later than end time - Restricted workstation)
    - 0x12 (Account locked out, disabled, expired, restricted, or revoked etc)
- Security: 4771 (Kerberos pre-authentication failed)
    - 0x10 - Smart card logon is being attempted and the proper certificate cannot be located.
    - 0x17 - The user’s password has expired.
    - 0x18 - The wrong password was provided.
- Security: Greater than 4720 Eand less than 4764 (Account/group modifications)

Logon type information

- Type: 0 (Used only by System account authentications)
- Type: 2 (Interactive Logon)
- Type: 3 (Network Authentication/SMB Auth Logon)
- Type: 4 (Batch Logon)
- Type: 5 (Service Logon)
- Type: 7 (Unlock Logon)
- Type: 8 (Network Cleartext Logon)
- Type: 9 (New Credentials Logon)
- Type: 10 (Terminal/RDP Logon Type)
- Type: 11 (Cached Interactive)
- Type: 12 (Cached Remote Interactive)
    - Same as RemoteInteractive. This is used for internal auditing.
- Type: 13 (Cached Unlock Logon)
    - Same as Unlock Logon.

Special logon information (4672)

Privilege Name	Description	Notes
SeAssignPrimaryTokenPrivilege	Replace a process-level token	Required to assign the primary token of a process. With this privilege, the user can initiate a process to replace the default token associated with a started subprocess.
SeAuditPrivilege	Generate security audits	With this privilege, the user can add entries to the security log.
SeBackupPrivilege	Back up files and directories	Required to perform backup operations. With this privilege, the user can bypass file and directory, registry, and other persistent object permissions for the purposes of backing up the system. This privilege causes the system to grant all read access control to any file, regardless of the access control list (ACL) specified for the file. Any access request other than read is still evaluated with the ACL.
SeCreateTokenPrivilege	Create a token object	Allows a process to create a token which it can then use to get access to any local resources when the process uses NtCreateToken() or other token-creation APIs. When a process requires this privilege, we recommend using the LocalSystem account (which already includes the privilege), rather than creating a separate user account and assigning this privilege to it.
SeDebugPrivilege	Debug programs	Required to debug and adjust the memory of a process owned by another account.With this privilege, the user can attach a debugger to any process or to the kernel. We recommend that SeDebugPrivilege always be granted to Administrators, and only to Administrators. Developers who are debugging their own applications do not need this user right. Developers who are debugging new system components need this user right. This user right provides complete access to sensitive and critical operating system components.
SeEnableDelegationPrivilege	Enable computer and user accounts to be trusted for delegation	With this privilege, the user can set the Trusted for Delegation setting on a user or computer object.The user or object that is granted this privilege must have write access to the account control flags on the user or computer object.
SeImpersonatePrivilege	Impersonate a client after authentication	With this privilege, the user can impersonate other accounts.
SeLoadDriverPrivilege	Load and unload device drivers	Required to load or unload a device driver.With this privilege, the user can dynamically load and unload device drivers or other code in to kernel mode. This user right does not apply to Plug and Play device drivers.

Privilege Name	Description	Notes
SeRestorePrivilege	Restore files and directories	Required to perform restore operations. This privilege causes the system to grant all write access control to any file, regardless of the ACL specified for the file. Any access request other than write is still evaluated with the ACL. Additionally, this privilege enables you to set any valid user or group SID as the owner of a file. With this privilege, the user can bypass file, directory, registry, and other persistent objects permissions when restoring backed up files and directories and determines which users can set any valid security principal as the owner of an object.
SeSecurityPrivilege	Manage auditing and security log	Required to perform a number of security-related functions, such as controlling and viewing audit events in security event log. With this privilege, the user can specify object access auditing options for individual resources, such as files, Active Directory objects, and registry keys.A user with this privilege can also view and clear the security log.
SeSystemEnvironmentPrivilege	Modify firmware environment values	Required to modify the nonvolatile RAM of systems that use this type of memory to store configuration information.
SeTakeOwnershipPrivilege	Take ownership of files or other objects	Required to take ownership of an object without being granted discretionary access. This privilege allows the owner value to be set only to those values that the holder may legitimately assign as the owner of an object. With this privilege, the user can take ownership of any securable object in the system, including Active Directory objects, files and folders, printers, registry keys, processes, and threads.
SeTcbPrivilege	Act as part of the operating system	This privilege identifies its holder as part of the trusted computer base.This user right allows a process to impersonate any user without authentication. The process can therefore gain access to the same local resources as that user.

System log information:

```
wevtutil qe system /f:text
eventquery.vbs /L system
```

Note: Some useful events -

- System: 7030 (Basic Service Operations)
- System: 7040 (The start type of a service was changed from disabled to auto start)
- System: 7045 (Service Was Installed)
- System: 1056 (DHCP Server Oddities)
- System: 10000 (COM Functionality)
- System: 20001 (Device Driver Installation)
- System: 20002 (Remote Access)
- System: 20003 (Service Installation)

Sysmon log information

When installed and running the event log is located at: “Applications and Services Logs/Microsoft/Windows/Sysmon/Operational”

Note: A WMI consumer is a management application or script that interacts with the WMI infrastructure. https://docs.microsoft.com/en-us/windows/desktop/WmiSdk/wmi-architecture

- Sysmon: 1 (Process create)
- Sysmon: 2 (File creation time)
- Sysmon: 3 (Network connection detected)
- Sysmon: 4 (Sysmon service state changed)
- Sysmon: 5 (Process terminated)
- Sysmon: 6 (Driver loaded)
- Sysmon: 9 (Image loaded)
- Sysmon: 10 (Process accessed)
- Sysmon: 11 (File created)
- Sysmon: 12 (Registry object added or deleted)
- Sysmon: 13 (Registry value set)
- Sysmon: 14 (Registry object renamed)
- Sysmon: 15 (File stream created)
- Sysmon: 16 (Sysmon configuration changed)
- Sysmon: 17 (Named pipe created)
- Sysmon: 18 (Named pipe connected)
- Sysmon: 19 (WMI filter)
- Sysmon: 20 (WMI consumer)
- Sysmon: 21 (WMI consumer filter)
- Sysmon: 22 (DNS Query)

Active Directory Investigation

Note: Live information can be found using DSQuery (https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/cc732952(v=ws.11)) or Netdom (https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/cc772217(v=ws.11)).

```
dsquery computer
dsquery user
dsquery contact
dsquery domainroot -inactive 4
dsquery group
dsquery ou
dsquery site
dsquery server
dsquery quota
dsquery *
        - dsquery * -limit 999999999
netdom query fsmo
netdom query trust
netdom query pdc
netdom query DC
netdom query server
netdom query workstation
netdom query OU
```

NT Directory Services Directory Information Tree File (ntds.dit)

Active Directory Database file containing all schema, domain, configuration information (e.g. users, IP, computers, domain trusts etc)

- %SystemRoot%\NTDS\ntds.dit
- %SystemRoot%\System32\ntds.dit
    - File created only when promoting certain OS to a DC, and seldom used.

Edb.log

10MB transaction log used to store temporary data before it is sent to the ntds.dit database.

- %SystemRoot%\NTDS\Edb.log

Edbxxxxx.log

Additional transaction log files if the main edb.log file gets larger than 10MB without being flushed to ntds.dit.

- %SystemRoot%\NTDS\edbxxxxx.log

Edb.chk

Checkpoint file used to determine how much of the transaction logs have been sent to the ntdis.dit database.

- %SystemRoot%\NTDS\edb.chk

Resx.log/Resx.jrs

Reserved log files in case the hard drive fills up, at which point these files will be used (ideally they should never be used).

- %SystemRoot%\NTDS\res1.log
- %SystemRoot%\NTDS\res2.log

Temp.edb

Temporary file to store information during in progress transactions.

- %SystemRoot%\NTDS\temp.edb

Schema.ini

Initialises the ntds.dit file when the domain controller is created, and is then never used again.

- %SystemRoot%\NTDS\schema.ini

Investigation of ntds.dit

Obtaining this file can be done using any of the following and also requires the SYSTEM hive to decrypt (note: ntdsutil may not work on older AD servers).

(Output will be under C:\Audit)

ntdsutil

```
ntdsutil "activate instance ntds" ifm "create full C:\Audit" quit quit
```

vssadmin

```
vssadmin create shadow /for=C:
mkdir C:\Audit
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[Number]\Windows\ntds\ntds.dit C:\Audit\ntds.dit
copy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy[Number]\Windows\System32\config\SYSTEM C:\Audit\SYSTEM
vssadmin delete shadows /shadow=[ShadowCopyID]
```

Other ‘less legitimate’ replication methods can be found detailed on the AD Security Blog by Sean Metcalf (https://adsecurity.org/?p=2398#MimikatzDCSync)

- Or by using Invoke-NinjaCopy (https://github.com/clymb3r/PowerShell/blob/master/Invoke-NinjaCopy/Invoke-NinjaCopy.ps1)

Repair the file if required:

```
esentutl /p /o C:\Audit\ntds.dit
```

Analysing this file offline can be done with tactics such as:

- Ropnop - Extract Hashes and Domain Info (https://blog.ropnop.com/extracting-hashes-and-domain-info-from-ntds-dit/)

Origami-PDF (Malicious PDF Analysis)

Github Download (https://github.com/gdelugre/origami)

```
pdfextract malware.pdf
```

More Malicious PDF/Doc Analysis

```
pdfid.py malware.pdf
pdfparser.py malware.pdf
pdfparser.py malware.pdf --object [number] --filter --raw --dump file.[extension]
oledump.py file.[extension]
oledump.py file.[extension] --select [number] --vbadecompress
```

Exiftool (Image Analysis)

```
exiftool malware.jpeg
```

RDP Cache images

This can be used to display some fragments of images which a user could see when operating on a server using the Windows RDP. The cache files are located: %USERPROFILE%\AppData\Local\Microsoft\Terminal Server Client\Cache\

These can be parsed using BMC-Tools (https://github.com/ANSSI-FR/bmc-tools)

```
bmc-tools.py -s ./ -d ./output
bmc-tools.py -s ./ -d ./output -o -b
```

RDP Activity

```
reg query 'HKU\SID\Software\Microsoft\Terminal Server Client' /s
```

Host Firewall information:

```
netsh firewall show config
advfirewall firewall show rule name=all verbose
```

Model of motherboard and hardware information:

```
wmic baseboard get product,manufacturer
wmic desktopmonitor get /all /format:list
wmic baseboard get /all /format:list
wmic bios get /all /format:list
wmic cpu get /all /format:list
```

Monitoring of open files:

```
openfiles /local on
```

Check Bitlocker Encryption

```
manage-bde -status
```

OR Powershell:

```
Get-BitLockerVolume
```

List open files

(this needs to have been enabled first and the PC rebooted, psfiles requires sysinternals psfile.exe)

```
openfiles /query
psfile
```

Display proxy information

```
netsh winhttp show proxy
```

Disconnect open files based on username:

```
openfiles /disconnect /a username
```

Powershell (some with WMI). Note: Namespace is a group of classes belonging to the same management environment. Most important is the CIMV2 child which is the most common.

Powershell Commands

```
help get-wmiobject
```

Service information

```
Get-WmiObject win32_service | select Name, DisplayName, State, PathName
Get-Service
```

Process WMI objects

```
get-wmiobject -list | where {$_.name -like "*process*"}
```

Process information

```
Get-WmiObject win32_process|select processname,ProcessId,CommandLine
Get-WmiObject win32_process -Filter "name like '%powershell.exe'" | select processId,commandline|FL
Get-Process
```

Baseline processes and services

(Used to compare new process/services)

```
Get-Process | Export-Clixml -Path C:\Users\User\Desktop\process.xml
Get-Service | Export-Clixml -Path C:\Users\User\Desktop\service.xml
$edproc = Import-Clixml -Path C:\Users\User\Desktop\process.xml
$edproc1 = Import-Clixml -Path C:\Users\User\Desktop\process1.xml
$edservice = Import-Clixml -Path C:\Users\User\Desktop\service.xml
$edservice1 = Import-Clixml -Path C:\Users\User\Desktop\service1.xml
Compare-Object $edproc $edproc1 -Property processname
Compare-Object $edservice $edservice1 -Property servicename
```

View and interact with shadow copies (MUST BE RUN FROM ELEVATED CMD.exe)

```
vssadmin list shadows | findstr "VolumeShadowCopy"
mklink /d shadow \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy2\
dir shadow
rmdir shadow
```

With a linked shadow copy we can copy pagefile.sys using the below.

```
mkdir C:\Audit
robocopy shadow C:\Audit pagefile.sys
attrib -s -h C:\Audit\pagefile.sys
```

Create Shadow Copy for C: drive

```
vssadmin create shadow /for=C:
```

Other Shadow Copy Techniques

In Windows 7 or certain other OS you may not have access to use 'vssadmin create'. As such some trickery may be required. In Windows 7 we can create a scheduled task (to execute with System privileges) and use it to create a Shadow Copy with Microsoft DLLs, this simulates the activity of creating a 'System Restore Point'. This can also be done with psexec if you wish to install the psexec service.

```
schtasks /ru "SYSTEM" /Create /SC DAILY /ST "00:00" /TN "\Microsoft\Windows\SystemRestore\SR" /TR
"%windir%\system32\rundll32.exe /d srrstr.dll,ExecuteScheduledSPPCreation" /f
schtasks /run /TN \Microsoft\Windows\SystemRestore\SR
vssadmin list shadows
```

If you want to remove the scheduled task so it doesn't run daily, use:

```
schtasks /delete /TN \Microsoft\Windows\SystemRestore\SR /f
```

You can also back it up using wbadmin (https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/wbadmin-start-backup), but it's a bit more intricate. The below example should backup C drive to E drive.

```
wbadmin start backup -backupTarget:E: -include:c:
```

TCP Connections

```
Get-NetTCPConnection –State Established
```

List of IPV4 addresses who have connected (RDP)

```
Get-WinEvent -Log 'Microsoft-Windows-TerminalServices-LocalSessionManager/Operational' | select -exp Properties | where {$_.Value -like '*.*.*.*' } | sort Value -u
```

Powershell logs

```
Get-WinEvent -LogName "Windows Powershell"
```

Event logs available

```
Get-EventLog -list
Get-WinEvent -Listlog * | Select RecordCount,LogName
wmic nteventlog list brief
```

Live Event Log Filtering

```
$Before = Get-Date 01/07/19;
$After = Get-Date 31/05/19;

Get-WinEvent -FilterHashtable @{ LogName='Security'; StartTime=$After; EndTime=$Before; Id='4624';
Data='127.0.0.1'} | Select -ExpandProperty Message

Get-WinEvent -FilterHashtable @{ LogName='Security'; StartTime=$After; EndTime=$Before; Id='4624';
Data='127.0.0.1'} | Select TimeCreated,Message | Select-String -Pattern "0x621EFDC", "0x825225F"

Get-WinEvent -FilterHashtable @{ LogName='Security'; StartTime=$After; EndTime=$Before; Id='4624';
Data='127.0.0.1'} | Select -ExpandProperty Message > [location]\log.txt;
cat [location]\log.txt | Select-String -Pattern "Subject:", "New Logon:", "Process
information","Network Information:" -Context 0,4;

Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-SmbClient/Connectivity';} | Select
Timecreated,LogName,Message | where {$_.message -like "*Failed to establish a network connection*"}
|FL

Get-WinEvent -FilterHashtable @{ LogName='*SMB*'; Data="[IP/HostName]"} | Select
Timecreated,LogName,Message |FL

Get-WinEvent -FilterHashtable @{ LogName='*SMB*';} | Select Timecreated,LogName,Message | where
{$_.message -like "*[IP/Hostname]*"} |FL

Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | where {$_.message -match
'0x1F260F3E' } | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | where
{$_.TimeCreated.ToString() -match ('28/10/2019')}|FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='2'} | where
{$_.TimeCreated.ToString() -match ('28/10/2019 11:22')}
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='127.0.0.1'} | where
{$_.TimeCreated.ToString() -match ('28/10/2019') -and $_.Message -match 'user' } | FL
TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='127.0.0.1'} | where
{$_.TimeCreated -ge (get-date).addDays(-3) -and $_.TimeCreated.ToString() -match ('11:04') -and
$_.Message -match 'user' } | FL TimeCreated,Message
```

Extract useful fields from Legacy Logs

```
$A=Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='127.0.0.1'} | where
{$_.TimeCreated -ge (get-date).addDays(-3) -and $_.Message -match 'INSERT DESIRED INFO HERE' };
ForEach ($Event in $A){$Event.TimeCreated;$Event.Message|findstr /i /C:"Logon
Type:";$Event.Message|findstr /i /C:"Security ID:";$Event.Message|findstr /i /C:"Account
Name:";$Event.Message|findstr /i /C:"Account Domain:";$Event.Message|findstr /i /C:"Process
ID:";$Event.Message|findstr /i /C:"Process Name:";$Event.Message|findstr /i /C:"Workstation
Name:";$Event.Message|findstr /i /C:"Source Network Address:";$Event.Message|findstr /i /C:"Source
Port:";echo "`n";};
```

Note: You can modify the second string to carve out wanted information, some examples below.

Find User Authenticating

```
ForEach ($Event in $A){$Event.TimeCreated;$Event.Message|findstr /i /C:"Account
Name:";$Event.Message|findstr /i /C:"Account Domain:";echo "`n";};
```

Find IP/Port Authenticating

```
ForEach ($Event in $A){$Event.TimeCreated;$Event.Message|(findstr /i /C:"Source Network
Address:";$Event.Message|findstr /i /C:"Source Port:";)|findstr -v "-";echo "`n";};
```

** Note: In the following section filter based on time for reduction of noise Get-Date (https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.utility/get-date).

e.g. get something like the below and add them to the FilterHashTable: StartTime=$After; EndTime=$Before;

```
$Date = (Get-Date).AddDays(-2)
$Before = Get-Date 01/07/19;
$After = Get-Date 31/05/19;
```

Remote Desktop Lateral Movement Detection (Destinations)

```
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='10'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4778';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4779';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-RemoteDesktopServices-RdpCoreTS/Operational'; Id='98';} | FL Message,ProcessId,TimeCreated
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-RemoteDesktopServices-RdpCoreTS/Operational'; Id='131';} | FL Message,ProcessId,TimeCreated
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TerminalServices-LocalSessionManager/Operational'; Id='21';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TerminalServices-LocalSessionManager/Operational'; Id='22';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TerminalServices-LocalSessionManager/Operational'; Id='25';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TerminalServices-LocalSessionManager/Operational'; Id='41';} | FL TimeCreated,Message
ls C:\Windows\Prefetch\rdpclip.exe*.pf
ls C:\Windows\Prefetch\tstheme.exe*.pf
```

Map Network Shares Lateral Movement Detection (Destinations)

```
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4672';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4776';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4768';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4769';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='5140';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='5145';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='5140';} | FL TimeCreated,Message
```

PsExec Lateral Movement Detection (Destinations)

```
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='2'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4672';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='5140'; Data='\\*\ADMIN$'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='System'; Id='7045';} | FL TimeCreated,Message
reg query HKLM\SYSTEM\CurrentControlSet\Services\PSEXESVC
reg query HKLM\SYSTEM\CurrentControlSet\Services\
ls C:\Windows\Prefetch\psexesvc.exe*.pf
```

Scheduled Tasks Lateral Movement Detection (Destinations)

```
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4672';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4698';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4702';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4699';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4700';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4701';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TaskScheduler/Maintenance'; Id='106';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TaskScheduler/Maintenance'; Id='140';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TaskScheduler/Maintenance'; Id='141';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TaskScheduler/Maintenance'; Id='200';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-TaskScheduler/Maintenance'; Id='201';} | FL TimeCreated,Message
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks" /s
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks" /s /v Actions
Get-ChildItem -path 'registry::HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tasks\' | Get-ItemProperty | FL Path, Actions
reg query "HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree"
gci -path C:\Windows\System32\Tasks\ -recurse -File
```

Services Lateral Movement Detection (Destinations)

```
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4697';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='System'; Id='7034';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='System'; Id='7035';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='System'; Id='7036';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='System'; Id='7040';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='System'; Id='7045';} | FL TimeCreated,Message
reg query 'HKLM\SYSTEM\CurrentControlSet\Services\'
```

WMI/WMIC Lateral Movement Detection (Destinations)

```
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4672';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-WMI-Activity/Operational'; Id='5857';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-WMI-Activity/Operational'; Id='5860';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-WMI-Activity/Operational'; Id='5861';} | FL TimeCreated,Message
C:\Windows\System32\wbem\Repository
ls C:\Windows\Prefetch\wmiprvse.exe*.pf
ls C:\Windows\Prefetch\mofcomp.exe*.pf
```

PowerShell Lateral Movement Detection (Destinations)

```
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4624'; Data='3'} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Security'; Id='4672';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-PowerShell/Operational'; Id='4103';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-PowerShell/Operational'; Id='4104';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-PowerShell/Operational'; Id='53504';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Windows PowerShell'; Id='400';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Windows PowerShell'; Id='403';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-WinRM/Operational'; Id='91';} | FL TimeCreated,Message
Get-WinEvent -FilterHashtable @{ LogName='Microsoft-Windows-WinRM/Operational'; Id='168';} | FL TimeCreated,Message
ls C:\Windows\Prefetch\wsmprovhost.exe*.pf
```

Extra Information

**AmCache

- C:\Windows\AppCompat\Programs\Amcache.hve
    - Amcache.hve\Root\File{Volume GUID}#######

**ShimCache

- C:\Windows\System32\config\SYSTEM
    - HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache

*_Prefetch - ls C:\Windows\Prefetch\evil.exe_.pf

User accounts and logon information

```
Get-WmiObject Win32_UserProfile
```

Share information

```
Get-WmiObject Win32_Share
net share
wmic share list brief
wmic netuse get Caption, DisplayType, LocalName, Name, ProviderName, Status
```

List Alternate Data Streams in current Dir and view them

```
gi * -s *
gc [FILENAME] -s [ADSNAME]
```

List Alternate Data Streams in text files within AppData

```
Get-ChildItem -Recurse -Path $env:APPDATA\..\ -include *.txt -ea SilentlyContinuenore|gi -s *|Select Stream -ea SilentlyContinue| Where-Object {$_.Stream -ine ":`$DATA"}
```

PowerForensics

https://powerforensics.readthedocs.io/en/latest/

General Notes

Under **%SystemRoot%\System32\config** the below registry hives are some of the most important to obtain. Additionally taking these files from within the RegBack directory also assists in comprehensive analysis should any anti-forensics activities have modified these registries.

- DEFAULT
- SAM
- SECURITY
- SOFTWARE
- SYSTEM

Under **\Users\name** there is also a **NTUSER.DAT** file which becomes HKEY_CURRENT_USER into the Registry when a user logs on, and this is very important to obtain. There's also a UsrClass.dat file which can be found: **%USERPROFILE%\AppData\Local\Microsoft\Windows\UsrClass.dat**

Gather artifacts

```
reg save HKLM\SAM [LOCATION]\SAM
reg save HKLM\SYSTEM [LOCATION]\SYSTEM
reg save HKLM\SECURITY [LOCATION]\SECURITY
reg save HKLM\SOFTWARE [LOCATION]\SOFTWARE
```

Powershell execution log

- Located at: C:\Users[name]\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline

Analyse document for macros

Using olevba (https://github.com/decalage2/oletools/wiki/olevba)

```
olevba [Document]
```

Capture powershell memdump and analyse

Using Procdump from sysinternals:

```
procdump -ma [PowershellPID]
```

Using powerdump (https://github.com/chrisjd20/power_dump)

```
python power_dump.py
1
ld powershell.exe_mem_dump
2
3
4
```

Recent execution of programs

- Prefetch Located at : %SystemRoot%\Prefetch\
- RecentFileCache.bcf Located at : %SystemRoot%\AppCompat\Programs\
- Amcache.hve (reg hive) Located at : %SystemRoot%\AppCompat\Programs\

USN Journal (any changes to NTFS volume)

```
fsutil usn readjournal C: > USN.txt
```

Link File Analysis

- LNK Files Located at:

  C:\Users*\AppData\Roaming\Microsoft\Windows\Recent

Jump Lists Analysis

- Jump List Files Located at:

  C:\Users*\AppData\Roaming\Microsoft\Windows\Recent\AutomaticDestinations

A rough PowerShell 1-liner to gather information on previous opened directories and files is below.

```
$Files=$(cat C:\Users\*\AppData\Roaming\Microsoft\Windows\Recent\*Destinations\*.*Destinations-ms);$Files.Split("``")|Select-String "Storage" | findstr -v "1SPSU"|findstr -v "?"
```

SRUM Analysis

- System Resource Usage Monitor Located at: %systemroot%\System32\sru\SRUDB.dat

Great tool to parse to csv: SRUM-Dump (https://github.com/MarkBaggett/srum-dump)

Windows 10 Mail App Forensics

```
%LocalAppData%\Comms\Unistore\data\0 - Windows phone data
%LocalAppData%\Comms\Unistore\data\2 - Contact lists
%LocalAppData%\Comms\Unistore\data\3 - Contents/body of email
%LocalAppData%\Comms\Unistore\data\5 - Calendar invitations
%LocalAppData%\Comms\Unistore\data\7 - Email attachments
```

Capture packets with netsh

Note: You will need to use something like the Microsoft Message Analyser to convert these captures to a cap file for analysis with Wireshark Download (https://www.microsoft.com/en-us/download/details.aspx?id=44226)

```
netsh trace start persistent=yes capture=yes tracefile=c:\temp\packetcapture.etl
netsh trace stop
```

NTUSER.DAT Important Registry entries:

Recent execution of programs (GUI)

HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer

- \RecentDocs *(Notes recent files run, most commonly .lnk files)*
- \UserAssist *(Notes files run and number of times run. Values are ROT13 encoded)*
- \TypedPaths *(Notes file locations visited using Windows Explorer address bar)*
- \RunMRU *(Notes recent commands executed through the 'run' program)*
- \ComDlg32 *(Last file path visited)*
    - \LastVistedPidlMRU *(Last PID which was 'Most Recently Used', e.g. the binaries used to open a file)*
    - \OpenSavePidlMRU *(Last Saved PID file which was 'Most Recently Used', location of a file opened by a binary)*

Execution of Sysinternals Tool

```
reg query HKCU\Software\Sysinternals\ /s /v EulaAccepted
reg query HKU\SID\Software\Sysinternals\ /s /v EulaAccepted
```

Recent Internet Explorer History

```
reg query "HKCU\Software\Microsoft\Internet Explorer\TypedURLs"
'C:\Users\username\AppData\Local\Microsoft\Windows\History\Low\History.IE5\'
'C:\Users\username\AppData\Local\Microsoft\Windows\History\'
'C:\Users\User\AppData\Roaming\Microsoft\Internet Explorer\UserData\Low'
```

Recent Chrome History

```
'C:\Users\Username\AppData\Local\Google\Chrome\User Data\Default\history'
```

Recent Firefox History

More Information (https://support.mozilla.org/en-US/kb/profiles-where-firefox-stores-user-data#w_what-information-is-stored-in-my-profile)

```
C:\Users\username\AppData\Roaming\Mozilla\Firefox\Profiles\*\
```

Shellbags

Shellbags can be used to verify the previous existance of files which have been deleted. This is used by the OS to store information about a file location's customisation e.g. look, feel, size, sorting files method, colour etc and resides after files have been deleted. Shellbags Explorer can be used to parse this information.

HKCU\SOFTWARE\Microsoft\Windows\Shell

- \BagMRU
- \Bags

UsrClass.dat Shellbags

Additional shellbags files can be found in UsrClass.dat

HKCU\SOFTWARE\Classes

- %USERPROFILE%\AppData\Local\Microsoft\Windows\UsrClass.dat

USB Information

Using the VolumeGUID found in SYSTEM\MountedDevices, you can find the user that actually mounted the USB device: NTUSER.DAT\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Mountpoints2 USB Times:

- First time device is connected
- Last time device is connected
- Removal time

SOFTWARE Hive Registry Entries

Common startup locations

- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Runonce
- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunonceEx

USB Information

- HKLM\SOFTWARE\Microsoft\Windows Portable Devices\Devices Note: Find Serial # and then look for FriendlyName to obtain the Volume Name of the USB device

- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\EMDMgmt

    - Key will ONLY be present if system drive is NOT SSD
    - Traditionally used for ReadyBoost
    - Find Serial # to obtain the Volume Serial Numberof the USB deviceoThe Volume Serial Number will be in decimal - convert to hex
    - You can find complete history of Volume Serial Numbers here, even if the device has been formatted multiple times. The USB device's Serial # will appear multiple times, each with a different Volume Serial Number generated on each format.

Network Information

- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\NetworkList
    - \Signatures
        - \Unmanaged
            - (record DefaultGatewayMac, DnsSuffix, FirstNetwork(SSID), ProfileGUID)
        - \Managed
    - \Nla\Cache
    - Profiles
- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\HomeGroup
- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\NetworkList\Profiles{GUID}
    - 0x06 = Wired
    - 0x17 = Broadband
    - 0x47 = Wireless

Useful Wireshark filters

All traffic to or from an IP

```
ip.addr == [IP]
```

All TCP traffic on a port

```
tcp.port eq [port]
```

All traffic from an IP

```
ip.dst==[IP]
```

Client>DC traffic filtering noise

```
smb || nbns || dcerpc || nbss || dns
```

SYSTEM Hive Registry Entries

USB Mount Information

- HKLM\SYSTEM\MountedDevices
  - Find Serial # to obtain the Drive Letter of the USB device
  - Find Serial # to obtain the Volume GUID of the USB device

Live System

- HKLM\SYSTEM\CurrentControlSet\Enum\USBSTOR (Class ID/Serial Number)
- HKLM\SYSTEM\CurrentControlSet\Enum\USB (VID/PID)

Forensic Image (Determine Control Set Number from HKLM\SYSTEM\Select\ -> Current Value)

- HKLM\SYSTEM\ControlSet00x\Enum\USBSTOR (Class ID/Serial Number)
- HKLM\SYSTEM\ControlSet00x\Enum\USB (VID/PID)

Note: VID/PID information can be found online. Subdirectories under USB and USBSTOR provide unique USB identifiers (if the & is near the end), if it is near the start they do not conform to MS standards and it is unique to the given PC only.

- HKLM\SYSTEM\CurrentControlSet\Enum\USBSTOR\Ven_Prod_Version\USB iSerial#\Properties{GUID}####
  - 0064 = First Install
  - 0066 = Last Connected
  - 0067 = Last Removal

More Information (https://github.com/woanware/usbdeviceforensics)

OS Information

- HKLM\SYSTEM\CurrentControlSet\Control\TimeZoneInformation
- HKLM\SYSTEM\CurrentControlSet\Control\ComputerName\ComputerName
- HKLM\SYSTEM\CurrentControlSet\services\LanmanServer\Shares
- HKLM\SYSTEM\CurrentControlSet\FileSystem
  - NtfsDisableLastAccessUpdate set at 0x1 means that access time stamps are turned OFF by default

Network Information

```
wmic nic get /all /format:list
wmic nicconfig get /all /format:list
```

- HKLM\SYSTEM\CurrentControlSet\services\Tcpip\Parameters\Interfaces
  - Display interfaces and their IP address configuration (using interface GUID)

Prefetch Information

- HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\PrefetchParameters
  - 0=Disabled
  - 1=Application prefetching enabled
  - 3=Application and Boot prefetching enabled (default)

PowerShell Host Based Investigation and Containment Techniques

Enable PS Remoting using PsExec

```
psexec.exe \\TARGET -s powershell Enable-PSRemoting -Force
```

Thanks Barnaby Skeggs (https://b2dfir.blogspot.com/2018/11/windows-powershell-remoting-host-based.html)

Setup logging for IR

Note: If you enter a PSSession, the logging won't persist, so you will need to enable it on the remote host and pull the file back afterwards

```
Start-Transcript -Path "C:\[location]\investigation-1.log" -NoClobber
```

Establish Remote Session

```
$s1 = New-PSsession -ComputerName remotehost -SessionOption (New-PSSessionOption -NoMachineProfile)
-ErrorAction Stop
```

Enter or exit remote session

```
Enter-PSSession -Session $s1
Exit-PSSEssion
```

Issuing remote command/shell

```
Invoke-Command -ScriptBlock {whoami} -Session $s1
Invoke-Command -file file.ps1 -Session $s1
```

Retrieving/downloading files

```
Copy-Item -Path "[RemoteHostFilePath]" -Destination "[LocalDestination]" -FromSession $s1
```

Checking for running processes

```
Invoke-Command -ScriptBlock {Get-Process} -Session $s1
```

Query Registry Keys

```
Invoke-Command -ScriptBlock {Get-ItemProperty -Path
HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run} -Session $s1
```

PCAP collection

*Note: Script and pcap should be located under: C:\Windows\System32 or your user directory.

```
Invoke-Command -ScriptBlock {ipconfig} -Session $s1

Invoke-Command -ScriptBlock {
$url = "https://raw.githubusercontent.com/nospaceships/raw-socket-sniffer/master/raw-socket-
sniffer.ps1"
Invoke-WebRequest -Uri $url `
        -OutFile "raw-socket-sniffer.ps1"
PowerShell.exe -ExecutionPolicy bypass .\raw-socket-sniffer.ps1 `
        -InterfaceIp "[RemoteIPv4Address]
        -CaptureFile "capture.cap"
        } -Session $s1
```

Blocking a domain

```
Invoke-Command -ScriptBlock { Add-Content C:\Windows\System32\drivers\etc\hosts "`n127.0.0.1
bad.com"} -Session $s1
```

Blocking an IP

```
Invoke-Command -ScriptBlock {New-NetFirewallRule -DisplayName "Block_Malicious_IP" -Direction
Outbound –LocalPort Any -Protocol TCP -Action Block -RemoteAddress 173.182.192.43}  -Session $s1
```

Unblocking an IP

```
Invoke-Command -ScriptBlock {Remove-NetFirewallRule -DisplayName "Block_Malicious_IP"} -Session $s1
```

Quarantining a host

```
Invoke-Command -ScriptBlock {New-NetFirewallRule -DisplayName InfoSec_Quarantine -Direction Outbound
-Enabled True -LocalPort Any -RemoteAddress Any -Action Block} -Session $s1
```

Remove a quarantined host

```
Invoke-Command -ScriptBlock {Remove-NetFirewallRule -DisplayName InfoSec_Quarantine} -Session $s1
```

Credentials and Exposure

When investigating a compromised asset, it’s important to know what remote triage methods leave your credentials on the infected endpoint, and what ones don’t.

Connection Method	Logon Type	Reusable credentials on destination	Notes
Logon via console	Interactive	Y	Includes hardware remote access/network KVM/lights-out cards
RUNAS	Interactive	Y	Nil
RUNAS/NETWORK	NewCredentials	Y	Clones LSA session, but uses new creds when connecting to network resources.
Remote Desktop	RemoteInteractive	Y	Nil
Remote Desktop Failure	RemoteInteractive	N	Only stored briefly
Net Use * \SERVER	Network	N	Nil
Net Use * \ SERVER /user	Network	N	Nil
MMC snap-ins to remote computer	Network	N	Nil
PowerShell WinRM	Network	N	e.g. Enter-PSSession SERVER
PowerShell WinRM with CredSSP	NetworkClearText	Y	e.g. New-PSSession SERVER -Authentication Credssp -Credential PWD
PsExec without explicit creds	Network	N	e.g. PsExec \SERVER cmd
PsExec with explicit creds	Network&Interactive	Y	PsExec \SERVER -u USER -p PWD cmd
Remote Registry	Network	N	Nil
Remote Desktop Gateway	Network	N	Authenticating to Remote Desktop Gateway
Scheduled Task	Batch	Y	Also saved as LSA secret on disk
Tools as Service	Service	Y	Also saved as LSA secret on disk
Vuln Scanners	Network	N	Most use Network logons; however, those that don’t have the risk of creds on destination.
IIS “Basic Authentication”	NetworkCleartext	Y	Nil
IIS “Integrated Windows Authentication“	Network	N	NTLM/Kerberos Providers

Windows Memory Forensics

Volatility 2.x Basics

(Note: Depending on what version of volatility you are using and where you may need to substitute volatility with vol.py)

Note: Version 3 of Volatility (https://github.com/volatilityfoundation/volatility3/) was released in November 2019 which will change some of this. More information on V3 of Volatility can be found on ReadTheDocs (https://volatility3.readthedocs.io/en/latest/basics.html).

Find out what profiles you have available

```
volatility --info
```

Find out the originating OS profile to be used from the memory dump.

```
volatility -f memorydump.mem imageinfo
volatility -f memorydump.mem kdbgscan
```

Determine what plugins are available for use.

```
volatility -f memorydump.mem --profile=<profilename> -h
```

Check what processes were running.

(Note: Any entires for svchost.exe should always have services.exe as a parent process and parameters such as /k should always be present)

```
volatility -f memorydump.mem --profile=<profilename> pslist
volatility -f memorydump.mem --profile=<profilename> psscan
volatility -f memorydump.mem --profile=<profilename> tree
```

Check what commands have been run and their output.

```
volatility -f memorydump.mem --profile=<profilename> cmdscan
volatility -f memorydump.mem --profile=<profilename> consoles
```

Dump process files which were running from memory.

```
volatility -f memorydump.mem --profile=<profilename> procdump -p <processid> --dump-dir=./
```

Dump the memory associated with a process file.

```
volatility -f memorydump.mem --profile=<profilename> memdump -p <processid> --dump-dir=./
```

Dump all cached files from memory.

```
volatility -f memorydump.mem --profile=<profilename> dumpfiles --dump-dir=./
```

Check what drivers or kernal modules were unloaded or hidden.

```
volatility -f memorydump.mem --profile=<profilename> modscan
```

Check what network connectivity has occurred.

```
volatility -f memorydump.mem --profile=<profilename> netscan
```

Check what network connectivity has occurred (Windows XP/Server 2003).

```
volatility -f memorydump.mem --profile=<profilename> connections
volatility -f memorydump.mem --profile=<profilename> conscan
volatility -f memorydump.mem --profile=<profilename> sockets
volatility -f memorydump.mem --profile=<profilename> sockscan
```

Check what information exists within registry from memory.

```
volatility -f memorydump.mem --profile=<profilename> hivelist
volatility -f memorydump.mem --profile=<profilename> hivescan
volatility -f memorydump.mem --profile=<profilename> hivedump --dump-dir=./
volatility -f memorydump.mem --profile=<profilename> userassist
volatility -f memorydump.mem --profile=<profilename> shellbags
volatility -f memorydump.mem --profile=<profilename> shimcache
volatility -f memorydump.mem --profile=<profilename> shimcachemem
```

Scan memory with Yara Rule

```
volatility -f memorydump.mem --profile=<profilename> yarascan -y rule.yara
```

Duplicate image space out as a raw DD file (e.g. dump files such as hiberfil.sys memory from memory).

```
volatility -f memorydump.mem --profile=<profilename> imagecopy
```

Dump timelined artifacts from memory.

```
volatility -f memorydump.mem --profile=<profilename> timeliner
```

Detect persistence mechanisms using Winesap

- Research Paper (http://www.dfrws.org/sites/default/files/session-files/characteristics_and_detectability_of_windows_auto-start_extensibility_points_in_memory_forensics.pdf)
- Volatility Plugin - Winesap (https://gitlab.unizar.es/rrodrigu/winesap)

```
volatility -f memdump.mem --profile=[profile] autoruns
volatility --plugins=./winesap/plugin -f memdump.mem --profile=[profile] autoruns
volatility --plugins=./winesap/plugin -f memdump.mem --profile=[profile] autoruns --match
```

Compare memory dump to known good memory dump.

- csababarta plugins (https://github.com/csababarta/volatility_plugins)

  volatility -f infected.mem –profile= processbl -B clean.mem -U 2>/dev/null volatility -f infected.mem --profile= servicebl -B clean.mem -U 2>/dev/null volatility -f infected.mem --profile= driverbl -B clean.mem -U 2>/dev/null

Output visual .dot file to view process tree

```
volatility -f memorydump.mem --profile=<profilename> psscan --output=dot --output-file=psscan.dot
volatility -f memorydump.mem --profile=<profilename> tree --output=dot --output-file=pstree.dot
dot -Tpng pstree.dot -o pstree.png
dot -Tpng pstree.dot -o psscan.png
```

Rekall Basics

Important Rekall Modules:

```
rekal -f memorydump.mem imageinfo
rekal -f memorydump.mem netstat
rekal -f memorydump.mem pstree
rekal -f memorydump.mem pslist
rekal -f memorydump.mem dlllist
rekal -f memorydump.mem netscan
rekal -f memorydump.mem pedump (fix these)
rekal -f memorydump.mem modules
```

Miscellaneous Tools and Notes

Eric Zimmerman has excellent widely used libraries and tools (https://ericzimmerman.github.io/#!index.md)

RegRipper (https://github.com/keydet89/RegRipper2.8)

```
rip.pl -r NTUSER.DAT -f ntuser | less.
rip.pl -r SAM -f sam | less
rip.exe -l
rip.exe -r C:\Users\User\ntuser.dat -p userassist
```

Kape (https://learn.duffandphelps.com/kape)

(http://savefrom.net/?url=https%3A%2F%2Fwww.youtube.com%

2F
wa
tc
h
%
3F
v
%
3D
L9
H1
uj
2H
Sb
8
&
ut
m
_s
ou
rc
e=
us
erj
s-
ch
ro
m
e
&
ut
m
_
m
ed
iu
m
=e
xt
en
si
on
s&
ut

m
_c
a
m
pa
ig
n=
lin
k_
m
od
ifi
er)

*Note: Video Tutorial (https://www.youtube.com/watch?v=L9H1uj2HSb8)

```
kape.exe --tsource C --target RegistryHives --tdest "[location]"
kape.exe --tsource \\server\directory --target !ALL --tdest "[location]" --vhdx LOCALHOST
```

ShimCaheParser (https://github.com/mandiant/ShimCacheParser)

```
ShimCacheParser.py -h
ShimCacheParser.py -i SYSTEM --BOM
```

AppCompatCacheParser (https://ericzimmerman.github.io/#!index.md)

```
AppCompatCacheParser.exe --csv .\ -t
```

AmCacheParser (https://ericzimmerman.github.io/#!index.md)

```
AmcacheParser.exe --csv .\ -f .\Amcache.hve
```

Windows 10 Timeline Database Parser (https://ericzimmerman.github.io/#!index.md)

```
WxTCmd.exe -f "C:\Users\[username]\AppData\Local\ConnectedDevicesPlatform\L.
[username]\ActivitiesCache.db" --csv .
```

Bulk Extractor (http://downloads.digitalcorpora.org/downloads/bulk_extractor/)

```
bulk_extractor64.exe -o [outputdir] memdump.mem
```

ForensicDots (https://www.forensicdots.de/)

Note: Can be used to determine the Machine Identification Code (https://en.wikipedia.org/wiki/Machine_Identification_Code) of a Printer.

Cyber Chef (https://gchq.github.io/CyberChef/)

The Cyber Swiss Army Knife - a web app for encryption, encoding, compression and data analysis. Note: This was created by an analyst at the GCHQ which is part of the UKs National Cyber Security Centre. The source is actively maintained on Github (https://github.com/gchq/CyberChef)

URLScan (https://urlscan.io/)

OSQuery (https://www.osquery.io/)

Velociraptor (https://www.velocidex.com/docs/)

ViperMonkey (https://github.com/kirk-sayre-work/ViperMonkey)

Parse and interpret VBA macros

```
vmonkey phishing.docm
```

Faster output

```
pypy vmonkey.py -s phishing.docm
```

Less verbose output

```
vmonkey -l warning phishing.docm
```

Google Rapid Response

This comes in the form of a Server > Client architecture but is very flexible.

- GRR Docs (https://grr-doc.readthedocs.io/en/latest/index.html)
- GRR Github (https://github.com/google/grr)

Kansa PowerShell IR Framework

This is a modular PowerShell IR Framework which can be used across multiple hosts in parallel.

- Kansa Github (https://github.com/davehull/Kansa)

Mounting image files in linux

```
mkdir /mnt/windows
imageMounter.py
ImageMounter.py -s <imagefile> /mnt/windows
cd /mnt/windows
```

OR

```
mkdir /mnt/windows
sudo apt install libguestfs-tools
sudo virt-list-filesystems <vhdx file>
sudo guestmount -a <vhdx file> -m /dev/<filesystemabove> -r /mnt/windows -o allow_other
```

Mounting image files in Windows

- Arsenal Image Mounter (https://arsenalrecon.com/downloads/)
- FTK Imager (https://accessdata.com/product-download/ftk-imager-version-4-2-1)
- Autopsy (https://www.sleuthkit.org/autopsy/download.php)

Unpack binary packed with UPX

```
upx -d PackedProgram.exe
```

Scan exchange for phishing emails

Disclaimer: Always test before running against live systems. For those running Office365 this documentation (https://docs.microsoft.com/en-us/office365/securitycompliance/search-for-and-delete-messages-in-your-organization) may be more useful.

```
# This is used to authenticate yourself and connect to the exchange server
$UserCredential = Get-Credential
$Session = New-PSSession -ConfigurationName Microsoft.Exchange -ConnectionUri
http://EXCHANGEHOSTFQDN/PowerShell/ -Credential $UserCredential
Import-PSSession $Session -DisableNameChecking

# This is used to confirm the mailboxes accessible and modules available
Get-Mailbox
Get-Module

# This is used to remove emails from a mailbox and move them to an administrator mailbox as a backup
Search-Mailbox -Identity "NAME" | Search-Mailbox -SearchQuery 'Subject:"SUBJECT LINE"' -
TargetMailbox "ADMINBACKUPMAILBOX" -TargetFolder "BACKUPFOLDER" -DeleteContent

# This is used to run a report on anyone who received an email with a malicious attachment and log
this information in an administrator mailbox
Get-Mailbox -ResultSize unlimited | Search-Mailbox -SearchQuery attachment:trojan* -TargetMailbox
"ADMINBACKUPMAILBOX" -TargetFolder "BACKUPFOLDER" -LogOnly -LogLevel Full

# This is used to disconnect from the established powershell session
Remove-PSSession $Session
```

Common DLL Information

DLL	Description
Kernel32.dll	(Windows Kernel) This is a very common DLL that contains core functionality, such as access and manipulation of memory, files, and hardware.
Advapi32.dll	(Advanced API) This DLL provides access to advanced core Windows components such as the Service Manager and Registry.
Ntdll.dll	(NT Layer) This DLL is the interface to the Windows kernel. Executables rarely import this file directly, although it is always imported indirectly by Kernel32.dll. If an executable deliberately imports this, it means that the author wanted to use functionality not normally available to Windows programs. Some tasks, such as hiding functionality or manipulating processes, will use this interface.
User32.dll	(Windows User) This DLL contains all the user interface components, such as buttons, scroll bars, and components for controlling and responding to user actions.
Wininet.dll	(Windows Internet API) This DLL contains high level networking functions. These implement protocols such as FTP, HTTP, and NTP.
Gdi32.dll	(Graphics Device Interface) This DLL contains functions used for displaying and manipulating graphics.
WSock32.dll and Ws2_32.dll	(Windows Sockets API) These are networking DLLs. A program that accesses either of these will likely connect to a network or perform network related tasks.

- When analysing a binary, small amount of strings present and minimal imported functions help confirm that it is a packed binary.

Windows Memory Analysis (Example Process with Volatility)

1. Identify memory OS information

```
volatility -f memorydump.mem imageinfo
```

2. Identify suspicious running processes

```
volatility -f memorydump.mem --profile=<profilename> pstree
```

3. Show suspicious running processes based on names.

```
volatility -f memorydump.mem --profile=<profilename> pstree | egrep 'winlogon|lsass|services'
volatility -f memorydump.mem --profile=<profilename> psscan
```

4. Show any malicious or suspicious processes requiring investigation

```
volatility -f memorydump.mem --profile=<profilename> malfind
```

5. Show any Process Hollowing (Hollow Process Injection)

```
volatility -f memorydump.mem --profile=<profilename> hollowfind
```

6. Dump suspicious process executables from memory

```
volatility -f memorydump.mem --profile=<profilename> procdump -p <processid> --dump-dir=./
```

7. Parse the Master File Table

```
volatility -f <memoryDump> mftparser -C --output-file=output.txt
```

8. Reassemble raw hex of file under $DATA back into original file from dump.raw file.

```
xxd -r dump.raw > <filename.originalextension>
```

9. Compare hashes with known detections e.g. VirusTotal.

```
sha256 <filename>
https://www.virustotal.com
```

10. Create a timeline of events.

```
volatility -f memorydump.mem --profile=<profilename> timeliner
volatility -f memorydump.mem --profile=<profilename> timeliner --hive=SECURITY
volatility -f memorydump.mem --profile=<profilename> timeliner --type=Registry
```

Windows Memory Analysis using Windbg

Using Comaeio SwishDbgExt (https://github.com/comaeio/SwishDbgExt) you are able to better analyse Windows Crash (DMP) files using Windbg. To do this, download the latest release, run windbg, load the correct dll and then run a command. At the time of writing there are:

```
!load X:\FullPath\SwishDbgExt.dll

!help              - Displays information on available extension commands
!ms_callbacks      - Display callback functions
!ms_checkcodecave - Look for used code cave
!ms_consoles       - Display console command's history
!ms_credentials    - Display user's credentials (based on gentilwiki's mimikatz)
!ms_drivers        - Display list of drivers
!ms_dump           - Dump memory space on disk
!ms_exqueue        - Display Ex queued workers
!ms_fixit          - Reset segmentation in WinDbg (Fix "16.kd>")
!ms_gdt            - Display GDT
!ms_hivelist       - Display list of registry hives
!ms_idt            - Display IDT
!ms_lxss           - Display lsxx entries
!ms_malscore       - Analyze a memory space and returns a Malware Score Index (MSI) - (based on Frank Boldewin's work)
!ms_mbr            - Scan Master Boot Record (MBR)
!ms_netstat        - Display network information (sockets, connections, ...)
!ms_object         - Display list of object
!ms_process        - Display list of processes
!ms_readkcb        - Read key control block
!ms_readknode      - Read key node
!ms_readkvalue     - Read key value
!ms_regcheck       - Scan for suspicious registry entries
!ms_scanndishook   - Scan and display suspicious NDIS hooks
!ms_services       - Display list of services
!ms_ssdt           - Display service descriptor table (SDT) functions
!ms_store          - Display information related to the Store Manager (ReadyBoost)
!ms_timers         - Display list of KTIMER
!ms_vacbs          - Display list of cached VACBs
!ms_verbose        - Turn verbose mode on/off
!ms_yarascan       - Scan process memory using yara rules
```

Normal Process Relationship Hierarchy (Geneology)

Excellent SANS Reference (https://digital-forensics.sans.org/media/DFPS_FOR508_v4.6_4-19.pdf)

Old:

System

- smss.exe
    - winlogon.exe (upon smss.exe exiting)
        - userinit.exe
            - explorer.exe (upon userinit.exe exiting)
    - wininit.exe (upon smss.exe exiting)
        - lsass.exe
        - services.exe
            - svchost.exe
            - taskhost.exe
    - crss.exe

Windows 10:

System

- smss.exe
    - winlogon.exe (upon smss.exe exiting)
        - userinit.exe
            - explorer.exe (upon userinit.exe exiting)
    - wininit.exe (upon smss.exe exiting)
        - lsass.exe
        - lsaiso.exe (credential guard only)
        - services.exe
            - svchost.exe
                - taskhostw.exe
                - runtimebroker.exe
    - crss.exe

Extra notes

Be mindful of the below:

- svchost.exe should always have services.exe pid as ppid
- there should never be more than 1 lsass.exe process.
- lsass.exe should always have a parent of winlogon.exe (WinXP and older) or Wininit.exe (Vista or newer).
- pslist and pstree follow a ‘Double Linked List’ which malware can ‘unlink’ itself from thus hiding the process.
- psscan looks instead for ‘EPROCESS blocks’ which is memory associated with a windows process.
- Discrepencies between these 2 areas can indicate the process hollowing has occurred.
  - VAD = Virtual Address Descriptor which lives in kernel memory.
  - PEB = Process Environment Block which lives in process memory.
- PAGE_EXECUTE_READWRITE protection indicates memory marked as executable, which may indicate potential shellcode.
- Process hollowing essentially pauses and duplicates a legitimate process, replaces the executable memory with something malicious, and then resumes the process. Process Injection on the other hand injects malicious code into an already running process which causes that process to execute the code.

Linux Cheat Sheet

Dumping Memory

```
dd if=/dev/kmem of=/root/kmem
dd if=/dev/mem of=/root/mem
```

Taking Image

```
fdisk -l
dd if=/dev/sda1 of=/[outputlocation]
```

Misc Useful Tools

FastIR (https://github.com/SekoiaLab/Fastir_Collector_Linux)

```
python ./fastIR_collector_linux.py
```

LinEnum (https://github.com/rebootuser/LinEnum/blob/master/LinEnum.sh)

```
./linenum.sh
./linenum.sh -t
```

Live Triage

System Information

```
date
uname –a
hostname
cat /proc/version
lsmod
```

Account Information

```
cat /etc/passwd
cat /etc/shadow
cat /etc/sudoers
cat /etc/sudoers.d/*
cut -d: -f1 /etc/passwd
getent passwd | cut -d: -f1
compgen -u
```

Current user

```
whoami
who
```

Last logged on users

```
last
lastb
```

Initialisation Files

```
cat /etc/bash.bashrc
cat ~/.bash_profile
cat ~/.bashrc
```

Environment and Startup Programs

```
cat /etc/profile
ls /etc/profile.d/
cat /etc/profile.d/*
```

Scheduled Tasks

```
ls /etc/cron.*
ls /etc/cron.*/*
cat /etc/cron.*/*
cat /etc/crontab
```

Configuration Information

```
ls /etc/*.d
cat /etc/*.d/*
```

Network Connections / Socket Stats

```
netstat
netstat -apetul
netstat -plan
netstat -plant
ss
ss -l
ss -ta
ss -tp
```

Network Configuration

```
ifconfig -a
```

Browser Plugin Information

```
ls -la ~/.mozilla/plugins
ls -la /usr/lib/mozilla/plugins
ls -la /usr/lib64/mozilla/plugins
ls -la ~/.config/google-chrome/Default/Extensions/
```

Kernel Modules and Extensions/

```
ls -la /lib/modules/*/kernel/*
```

Process Information

ps <simple list output threads misc all>'

```
ps -s
ps -l
ps -o
ps -t
ps -m
ps -a
top
```

Search files recursively in directory for keyword

```
grep -H -i -r "password" /
```

Process Tree

```
ps -auxwf
```

Open Files and space usage

```
lsof
du
```

Pluggable Authentication Modules (PAM)

```
cat /etc/pam.d/sudo
cat /etc/pam.conf
ls /etc/pam.d/
```

Disk / Partition Information

```
fdisk -l
```

System Calls / Network Traffic (https://bytefreaks.net/gnulinux/how-to-capture-all-network-traffic-of-a-single-process)

```
strace -f -e trace=network -s 10000 <PROCESS WITH ARGUMENTS>;
strace -f -e trace=network -s 10000 -p <PID>;
```

Note: Below material with thanks to Craig Rowland - Sandfly Security (https://blog.apnic.net/2019/10/14/how-to-basic-linux-malware-process-forensics-for-incident-responders/)

Detailed Process Information

```
ls -al /proc/[PID]
```

Note:

- CWD = Current Working Directory of Malware
- EXE = Binary location and whether it has been deleted
- Most Common Timestamp = When process was created

Recover deleted binary which is currently running

```
cp /proc/[PID]/exe /[destination]/[binaryname]
```

Capture Binary Data for Review

```
cp /proc/[PID]/ /[destination]/[PID]/
```

Binary hash information

```
sha1sum /[destination]/[binaryname]
md5sum /[destination]/[binaryname]
```

Process Command Line Information

```
cat /proc/[PID]/cmdline
cat /proc/[PID]/comm
```

Note:

- Significant differences in the above 2 outputs and the specified binary name under /proc/[PID]/exe can be indicative of malicious software attempting to remain undetected.

Process Environment Variables (incl user who ran binary)

```
strings /proc/[PID]/environ
cat /proc/[PID]/environ
```

Process file descriptors/maps (what the process is ‘accessing’ or using)

```
ls -al /proc/[PID]/fd
cat /proc/[PID]/maps
```

Process stack/status information (may reveal useful elements)

```
cat /proc/[PID]/stack
cat /proc/[PID]/status
```

Deleted binaries which are still running

```
ls -alr /proc/*/exe 2> /dev/null |  grep deleted
```

Process Working Directories (including common targeted directories)

```
ls -alr /proc/*/cwd
ls -alr /proc/*/cwd 2> /dev/null | grep tmp
ls -alr /proc/*/cwd 2> /dev/null | grep dev
ls -alr /proc/*/cwd 2> /dev/null | grep var
ls -alr /proc/*/cwd 2> /dev/null | grep home
```

Hidden Directories and Files

```
find / -type d -name ".*"
```

Immutable Files and Directories (Often Suspicious)

```
lsattr / -R 2> /dev/null | grep "\----i"
```

SUID/SGID and Sticky Bit Special Permissions

```
find / -type f \( -perm -04000 -o -perm -02000 \) -exec ls -lg {} \;
```

File and Directories with no user/group name

```
find / \( -nouser -o -nogroup \) -exec ls -lg  {} \;
```

File types in current directory

```
file * -p
```

Executables on file system

```
find / -type f -exec file -p '{}' \; |  grep ELF
```

Hidden Executables on file system

```
find / -name ".*" -exec file -p '{}' \; | grep ELF
```

Files modified within the past day

```
find / -mtime -1
```

Persistent Areas of Interest

```
/etc/rc.local
/etc/initd
/etc/rc*.d
/etc/modules
/etc/cron*
/var/spool/cron/*
```

Audit Logs

```
ls -al /var/log/*
ls -al /var/log/*tmp
utmpdump /var/log/btmp
utmpdump /var/run/utmp
utmpdump /var/log/wtmp
```

MacOS Cheat Sheet

IMPORTANT NOTE: This section is still in its early stages of documentation and testing. I strongly suggest checking out Sarah Edwards, who is an industry leader in this space, as she has many excellent resources and this section for the most part is reiterating the hard work she has put in. Other excellent resources include the Mac OS X Forensics Wikis and shared spreadsheet containing Forensics Artifacts.

- Sarah Edwards (https://twitter.com/iamevltwin)
- Mac4n6 (https://www.mac4n6.com/)
- SANS FOR518 Reference Sheet (https://digital-forensics.sans.org/media/FOR518-Reference-Sheet.pdf)
- Mac OS X 10.9 Forensics Wiki (https://forensicswiki.org/wiki/Mac_OS_X_10.9_-_Artifacts_Location)
- Mac OS X 10.11 Forensics Wiki (https://forensicswiki.org/wiki/Mac_OS_X_10.11_(ElCapitan)_-_Artifacts_Location)
- Mac OS X Forensics Artifacts Spreadsheet (https://docs.google.com/spreadsheets/d/1X2Hu0NE2ptdRj023OVWIGp5dqZOw-CfxHLOW_GNGpX8/edit#gid=1317205466)

Live Mac IR / Triage

System Information

```
date
sw_vers
uname -a
hostname
```

Network Connections

```
netstat -an
netstat -anf
lsof -i
```

Routing Table

```
netstat -rn
```

Network Information

```
arp -an
ndp -an
ifconfig
```

Open Files

```
lsof
```

Bash History

```
cat ~/.bash_history
history
```

User Logins

```
who -a
w
last
```

Running Processes

```
ps aux
```

System Profiler

```
system_profiler -xml -detaillevel full > systemprofiler.spx
```

Persistent Locations

Quick Overview (KnockKnock) (https://www.objective-see.com/products/knockknock.html)

```
./KnockKnock.app/Contents/MacOS/KnockKnock -whosthere > /path/to/some/file.json
```

XPC Services

```
ls Applications/<application>.app/Contents/XPCServices/
cat Applications/<application>.app/Contents/XPCServices/*.xpc/Contents/Info.plist
ls ~/System/Library/XPCServices/
```

Launch Agents & Launch Daemons

```
ls /Library/LaunchAgents/
ls /System/Library/LaunchAgents/
ls /System/Library/LaunchDaemons/
ls /Library/LaunchDaemons/
```

LoginItems

```
cat ~/Library/Preferences/com.apple.loginitems.plist
ls <application>.app/Contents/Library/LoginItems/
```

Disable Persistent Launch Daemon

```
sudo launchctl unload -w /Library/LaunchDaemons/<name>.plist
sudo launchctl stop /Library/LaunchDaemons/<name>.plist
```

Web Browsing Preferences

```
cat ~/Library/Preferences/com.apple.Safari.plist
ls ~/Library/Application Support/Google/Chrome/Default/Preferences
ls ~/Library/Application Support/Firefox/Profiles/********.default/prefs.js
```

Safari Internet History

```
cat ~/Library/Safari/Downloads.plist
cat ~/Library/Safari/History.plist
cat ~/Library/Safari/LastSession.plist
ls ~/Library/Caches/com.apple.Safari/Webpage Previews/
sqlite3 ~/Library/Caches/com.apple.Safari/Cache.db
```

Chrome Internet History

```
ls ~/Library/Application Support/Google/Chrome/Default/History
ls ~/Library/Caches/Google/Chrome/Default/Cache/
ls ~/Library/Caches/Google/Chrome/Default/Media Cache/
```

Firefox Internet History

```
sqlite3 ~/Library/Application Support/Firefox/Profiles/********.default/places.sqlite
sqlite3 ~/Library/Application Support/Firefox/Profiles/********.default/downloads.sqlite
sqlite3 ~/Library/Application Support/Firefox/Profiles/********.default/formhistory.sqlite
ls ~/Library/Caches/Firefox/Profiles/********.default/Cache
```

Apple Email

```
cat ~/Library/Mail/V2/MailData/Accounts.plist
ls ~/Library/Mail/V2/
ls ~/Library/Mail Downloads/
ls ~/Downloads
cat ~/Library/Mail/V2/MailData/OpenAttachments.plist
```

Temporary / Cached

```
ls /tmp
ls /var/tmp
ls /Users/<user>/Library/Caches/Java/tmp
ls /Users/<user>/Library/Caches/Java/cache
        /Applications/Utilities/Java Preferences.app
```

System and Audit Logs

```
ls /private/var/log/asl/
ls /private/var/audit/
cat /private/var/log/appfirewall.log
ls ~/Library/Logs
ls /Library/Application Support/<app>
ls /Applications/
ls /Library/Logs/
```

Specific Log Analysis

```
bzcat system.log.1.bz2
system.log.0.bz2 >> system_all.log
cat system.log >> system_all.log
syslog -f <file>
syslog -T utc -F raw -d /asl
syslog -d /asl
praudit -xn /var/audit/*
sudo log collect
log show
log stream
```

Files Quarantined

```
ls ~/Library/Preferences/com.apple.LaunchServices.QuarantineEvents.V2
ls ~/Library/Preferences/com.apple.LaunchServices.QuarantineEvents
```

User Accounts / Password Shadows

```
ls /private/var/db/dslocal/nodes/Default/users/
ls /private/var/db/shadow/<User GUID>
```

Pluggable Authentication Modules (PAM)

```
cat /etc/pam.d/sudo
cat /etc/pam.conf
ls /etc/pam.d/
```

File Fingerprinting/Reversing

```
file <filename>
xxd <filename>
nm -arch x86_64 <filename>
otool -L <filename>
sudo vmmap <pid>
sudo lsof -p <pid>
xattr –xl <file>
```

Connected Disks and Partitions

```
diskutil list
diskutil info <disk>
diskutil cs
ap list
gpt –r show
gpt -r show -l
```

Disk File Image Information

```
hdiutil imageinfo *.dmg
```

User Keychain Information

```
security list-keychains
security dump-keychains -d <keychain>
```

Spotlight Metadata

```
mdimport –X | -A
mdls <file>
```

SANS FOR518 Reference (https://digital-forensics.sans.org/media/FOR518-Reference-Sheet.pdf)

Bonus Valuable Links

- MITRE ATT&CK™ (https://attack.mitre.org/)
- MITRE Cyber Analytics Repository (https://car.mitre.org/)
- Atomic Red Team (https://redcanary.com/atomic-red-team/)
- Awesome Incident Response (https://github.com/meirwah/awesome-incident-response)
- Awesome Forensics (https://github.com/Cugu/awesome-forensics)
- Mac OSX Forensics (https://github.com/n0fate/mac-osx-forensics)
- Unofficial Mac 4n6 Resources (https://github.com/pstirparo/mac4n6)
- Apple macOS command line (OS X bash) (https://ss64.com/osx/)
- Mac4n6 (https://www.mac4n6.com/)

Special Thanks:

- 13Cubed (https://www.youtube.com/13cubed) (http://savefrom.net/?url=https%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3DHcUMXxyYsn snw&utm_source=userjs-chrome&utm_medium=extensions&utm_campaign=link_modifier)

- John Strand Windows Live Forensics (https://www.youtube.com/watch?v=HcUMXxyYsnw)

- Blue Team Field Manual - Alan White & Ben Clark
- DFIR Training Windows Registry (https://www.dfir.training/resources/downloads/windows-registry)
- Commandlinekungfu (http://blog.commandlinekungfu.com/)
- Microsoft Audit Logon Events (https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2003/cc787567(v=ws.10))
- Windows Dev Win32_LogonSession (https://docs.microsoft.com/en-us/windows/desktop/CIMWin32Prov/win32-logonsession)

- Black Hills Information Security Windows Memory Forensics (https://www.youtube.com/watch?

(http://savefrom.net/?url=https%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3DcYphLiySA

	C4 & ut m _s ou rc e= us erj s- ch ro m e & ut m _ m ed iu m =e xt en si on s& ut m _c a m pa ig n= lin k_ m od ifi
v=cYphLiySAC4)	er)

- Forensics Wiki (https://forensicswiki.org/wiki/Main_Page)
- Wireshark Wiki (https://wiki.wireshark.org/DisplayFilters)
- Microsoft Sysmon (https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon)
- ACSC Github (https://github.com/AustralianCyberSecurityCentre/windows_event_logging)
- Windows Defender Docs (https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-defender-antivirus/command-line-arguments-windows-defender-antivirus)
- Mikefrobbins (https://mikefrobbins.com/2013/12/19/using-powershell-to-remove-phishing-emails-from-user-mailboxes-on-an-exchange-server/)
- Microsoft Office365 (https://docs.microsoft.com/en-us/office365/securitycompliance/search-for-and-delete-messagesadmin-help)
- Microsoft Exchange (https://docs.microsoft.com/en-us/powershell/module/exchange/mailboxes/search-mailbox)
- Microsoft Threat Protection (https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4768)
- ADsecurity (https://adsecurity.org/?page_id=1821)
- Windows cmd fu (https://twitter.com/wincmdfu)
- Crowdstrike (https://www.crowdstrike.com/blog/kovter-killer-how-to-remediate-the-apt-of-clickjacking/)
- Technet Blog (https://blogs.technet.microsoft.com/tip_of_the_day/2015/03/28/tip-of-the-day-reading-the-usn-journal/)
- IETF RFC3227 (https://tools.ietf.org/html/rfc3227#section-2.1)
- Cybereason Adobe Worm (https://www.cybereason.com/blog/adobe-worm-faker-uses-lolbins-and-dynamic-techniques-to-deliver-customized-payloads)
- Melanijan93 Windows 10 mail forensics (https://medium.com/@melanijan93/windows-10-mail-app-forensics-39025f5418d2)
- Cybereason Trickbot (https://www.cybereason.com/blog/triple-threat-emotet-deploys-trickbot-to-steal-data-spread-ryuk-ransomware)
- Bryan Ambrose (https://www.datadigitally.com/2019/06/retrieving-memory-image-remotely.html)
- Lee Holmes (https://twitter.com/Lee_Holmes)
- Florian Roth (https://www.bsk-consulting.de/2014/08/28/scan-system-files-manipulations-yara-inverse-matching-22/)
- Matt Graeber (https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent%20Asynchronous-And-Fileless-Backdoor-wp.pdf)
- Vasily Gusev (https://social.technet.microsoft.com/wiki/contents/articles/4242.wmi-discovery-using-powershell.aspx)
- MS Docs 4672 (https://docs.microsoft.com/en-us/windows/security/threat-protection/auditing/event-4672)

- Kris (https://www.pdq.com/blog/modifying-the-registry-users-powershell/)
- Barnaby Skeggs (https://twitter.com/barnabyskeggs)
- Sarah Edwards (https://twitter.com/iamevltwin)
- Fahim Hossain (https://medium.com/@fahimhossain_16989/adding-startup-scripts-to-launch-daemon-on-mac-os-x-sierra-10-12-6-7e0318c74de1)
- Surendra Anne (https://www.linuxnix.com/linux-directory-structure-explainedetc-folder/)
- SANS Hunt Evil (https://digital-forensics.sans.org/media/SANS_Poster_2018_Hunt_Evil_FINAL.pdf)
- Craig Rowland - Sandfly Security (https://blog.apnic.net/2019/10/14/how-to-basic-linux-malware-process-forensics-for-incident-responders/)
- Habibar Rahmen - MSDN Blog (https://blogs.msdn.microsoft.com/servergeeks/2014/10/14/active-directory-files-and-their-functions/)
- Mari DeGrazia (https://az4n6.blogspot.com/2014/10/timestomp-mft-shenanigans.html)
- Markus Piéton (https://www.a12d404.net/windows/2019/10/30/schedsvc-persist-without-task.html)
- Samir (https://github.com/sbousseaden/Slides/blob/master/Windows%20DFIR%20Events.pdf)
- FireEye (https://www.fireeye.com/blog/threat-research/2019/12/breaking-the-rules-tough-outlook-for-home-page-attacks.html)
- Microsoft-Mitigating Pass-the-Hash Attacks (https://www.microsoft.com/en-us/download/details.aspx?id=36036)

Tags: Cheatsheet DFIR Forensics Incident IR Response

Categories: Cheatsheet DFIR

Updated: January 13, 2020