Roll Your Own Incident Response with PowerShell
A Little Background

- Founder and president of Vertigrate
- Digital forensics, incident response, and malware reverse engineering
- Proactively engages with business and security teams of all sizes on blue team engagements.
- SANS Community Instructor
- Volunteers as
  - Arizona chapter’s board of the High Technology Crime Investigation Association (HTCIA),
  - Membership Director of the Phoenix chapter’s Membership Director Information Systems Security Association (ISSA),
  - State Bar of Arizona’s Technology Committee.
- Certification Hound:
  - CISSP, CISM, GREM, GCFE, GCIH, GPEN, CCME
Agenda

- Why I wrote it
- Caveats
- ~350 line PowerShell script
- Section Review
  - What is it collecting
  - Why is it collecting
  - How to use the data
- To Do List
Why?

- To triage networks and prioritize C E R
- Started in 2014
- Been adding / editing ever since
- Couldn’t find / afford suitable solution
- Wanted to customize it for my needs
- Combination of ten years of experience, including FOR408 / SEC504
What Does It Do?

- Queries a live system locally or across the network
- Pulls back system artifacts for offline review
- File sets named by host and/or user/SID
- Native Windows commands
- Allows efficient triage of numerous hosts with light touch and admin effort
What Do We Get?

- Local Admins & Groups
- Areas of Persistence
  - Services, Scheduled Tasks, Startup, etc.
- Installed Programs & Patches
- Listing of Shadow Copies
- Live Statistics
  - Processes, Network Stats, etc.
- Big 3 Event Logs, plus PS and Tasks
- SYSTEM, SOFTWARE & SAM Registry Hives
- All User Registry Hives
- Prefetch Listing & Contents
- List of User Folders, plus contents for
  - Browsers
  - Jumplists
  - Link Files
- Full File Listing if Desired
What Does it NOT Do?

- Linux
- Find EVIL Automagically
  - i.e.; Detect Rootkits
- Create a forensic image
- Not leave artifacts, including in your output
What Does It Need?

- Minimum PowerShell 2.0
- Local Admin Account
- Network Access to the Host(s)
- (Optional) Autorunsc.exe to speed review
Acknowledgements

- **Get-LocalAdmin**

- **Get-LocalGroup**
  - Thank you Boe Prox, https://mcpmag.com/articles/2015/06/18/reporting-on-local-groups.aspx
The Basics

- Local Users & Groups, including admins
- Logs
  - Security
  - Application
  - System
  - Task Scheduler
  - PowerShell
- `wevtutil epl Security c:\temp\sec\logs\${Env:ComputerName}-Security.evtx`
- Etc...
- `wevtutil epl Microsoft-Windows-PowerShell/Operational c:\temp\sec\logs\${Env:ComputerName}-PowerShell.evtx`
Persistence Areas

- Autoruns
  - `C:\temp\SEC\tools\autorunsc64.exe -accepteula -a * -c > C:\temp\sec\${Env:ComputerName}-autoruns.csv`

- Scheduled Tasks
- Services
- Plus Startup and Registry Areas... more on those later
Scheduled Tasks & Running Services

#SCHEDULED TASKS

- schtasks /query /fo list /v | out-file C:\temp\sec\prog\${Env:ComputerName}-ScheduledTasks.txt
- schtasks /query /fo csv /v | out-file C:\temp\sec\prog\${Env:ComputerName}-ScheduledTasks.csv

#RUNNING WINDOWS SERVICES

- Get-Service | Where-Object {$_status -eq "running"} | select-object Name, DisplayName, Status | Export-Csv C:\temp\sec\prog\${Env:ComputerName}-WinSvcs.csv -NoTypeInformation
Running Processes

#TASK LIST COMMANDS

- `tasklist /svc | out-file C:\temp\sec\prog\${Env:ComputerName}-TaskList-Services.txt`
- `tasklist /v | out-file C:\temp\sec\prog\${Env:ComputerName}-TaskList-Verbose.txt`
- `tasklist /m | out-file C:\temp\sec\prog\${Env:ComputerName}-TaskList-Modules.txt`
- `Get-WmiObject -class win32_process | select-object -property processname, ws, parentprocessid, processid, sessionid | Export-csv C:\temp\sec\prog\${Env:ComputerName}-tasklist.txt`
- `get-process | select Name, Description, ID, @{Label="Memory Usage(KB)";Expression={$_.WS / 1KB}}, @{Label="CPU Time (s)";Expression={$_.CPU}} | export-csv C:\temp\sec\prog\${Env:ComputerName}-tasklist.csv -NoTypeInformation`
Network Stats

- IP Config
- ARP
- Hosts
- Netstat
- Shares
- Sessions
- Windows Firewall Config (and log if enabled)
Basic IP Stuff

- #NETWORKING RELATED COMMANDS#

  - #IP INFO
  - `ipconfig /all | out-file C:\temp\sec\net\${Env:ComputerName}-IPInfo.txt`
  - `ipconfig /displaydns | out-file -append C:\temp\sec\net\${Env:ComputerName}-IPInfo.txt`

- #ADDRESS RESOLUTION PROTOCOL
  - `arp -a | out-file C:\temp\sec\net\${Env:ComputerName}-ARPTable.txt`

- #COPY THE HOST FILE
  - `copy-item -path C:\Windows\System32\drivers\etc\hosts -destination C:\temp\sec\net\`
#NETSTAT for established and listening network connections.
netstat -naob | out-file C:\temp\sec\net\${Env:ComputerName}-Netstat.txt

#LOCAL COMPUTER SHARES (Hidden included)
get-WmiObject -class Win32_Share | out-file C:\temp\sec\net\${Env:ComputerName}-LocalShares.txt

#GET LIST OF DRIVES
Get-WmiObject -class Win32_LogicalDisk | out-file C:\temp\sec\net\${Env:ComputerName}-LocalDrives.txt
#OUTBOUND SESSIONS (similar to net use)

```powershell
write-output "The following outbound connections are currently in session:
| out-file C:\temp\sec\net\${Env:ComputerName}-Sessions.txt

Get-WmiObject Win32_NetworkConnection | out-file C:\temp\sec\net\${Env:ComputerName}-sessions.txt -append
```
Inbound Sessions

#INBOUND SESSIONS

$computer = "LocalHost"
$namespace = "root\CIMV2"
$userSessions = Get-WmiObject -class Win32_ServerConnection -computername $computer -namespace $namespace

Write-output "The following users are connected to your PC (Inbound sessions): " | out-file -append C:\temp\sec\net\${Env:ComputerName}-Sessions.txt

foreach ($userSession in $userSessions)
{
    $userDetails = [string]::Format("User {0} from machine {1} on share: {2}", $userSession.UserName, $userSession.ComputerName, $userSession.ShareName)

    Write-output $userDetails | out-file C:\temp\sec\net\${Env:ComputerName}-Sessions.txt -append
}

SANS Blue Team Summit 2018 – Mike Lombardi, Vertigrate

4/24/2018
#FIREWALL CONFIGURATION

- `netsh advfirewall show allprofiles | out-file C:\temp\sec\net\${Env:ComputerName}-FirewallConfig.txt`
- `netsh advfirewall firewall show rule name=all type=dynamic | Out-File C:\temp\sec\net\${Env:ComputerName}-FirewallRules.txt`
- `copy-item -path C:\Windows\system32\LogFiles\Firewall\pfirewall.log -Destination C:\temp\sec\net\${Env:ComputerName}-pfirewall.log`
How to Deploy

USB

- Lots of Manual Effort, but no PSEXEC considerations
- \USBLiveResponse.ps1 *&gt;&amp;1 | Tee-Object -File $Env:ComputerName-consolelog.txt -Append

Across the Network

- SCCM: Fastest, but requires an IT Admin who is comfortable
- Batch Script
  - *PowerShell, PSEXEC and CMD all working together*
- PS-Remoting
  - Perhaps...
```powershell
$ErrorActionPreference = 'SilentlyContinue'
$CollectionServer = "CONFIGURE ME"
$CollectionShare = "CONFIGURE ME"
$Machines = Get-Content C:\temp\hosts.txt
ForEach ($machine in $Machines)
{
    mkdir "$machine\c$\temp\sec\tools"
    Copy-Item C:\Temp\autorunsc64.exe "$machine\c$\temp\sec\tools\autorunsc64.exe" -Force -Recurse
    Copy-Item "C:\Temp\LiveResponse.ps1" "$machine\c$\temp\sec\LiveResponse.ps1"
    & cmd /c "C:\Temp\PsExec64.exe -accepteula "$machine" cmd /c echo .^|PowerShell Set-ExecutionPolicy -ExecutionPolicy Unrestricted"
    & cmd /c "C:\Temp\PsExec64.exe -accepteula "$machine" cmd /c echo .^|PowerShell -f "$machine\c$\temp\sec\LiveResponse.ps1"
    & cmd /c "robocopy "$machine\c$\temp\sec" "$CollectionServer\$CollectionShare\triage" $machine /E /r:4 /w:3 /FP /NP /log+:C:\$machine-RobocopyReturnResults.txt"
    Remove-Item -Path "$machine\c$\temp\sec" -Force -Recurse
    & cmd /c "C:\Temp\PsExec64.exe -accepteula "$machine" cmd /c echo .^|PowerShell Set-ExecutionPolicy -ExecutionPolicy Restricted"
}
\Loop.ps1 *>&1 | Tee-Object -File $Env:ComputerName-consolelog.txt -Append
```
TXT or CSV files... that is


Out-File contains list of file and line where each hit was found

Where $Match1 is our rogue executable
HKLM System and Software Hives

#Dump HKLM\SYSTEM, SOFTWARE & SAM - Registry tools will parse

- `reg save HKLM\SYSTEM c:\temp\sec\registry\${Env:ComputerName}-SYSTEM`
- `reg save HKLM\SOFTWARE c:\temp\sec\registry\${Env:ComputerName}-SOFTWARE`
- `reg save HKLM\SAM c:\temp\sec\registry\${Env:ComputerName}-SAM`
#Dump available User registries separately. Some may error out. Like the blank top line.

#First generate list of potential user registries.
reg query HKU > C:\temp\sec\registry\HKU-list.txt

#Then cleanup the list.
(Get-Content C:\temp\sec\registry\HKU-list.txt) -replace "HKEY_USERS\","" | Out-File C:\temp\sec\registry\HKU-list.txt

#Now iterate through the list and save each key.
$RegistryUsers = Get-Content C:\temp\sec\registry\HKU-list.txt
ForEach ($RegistryUser in $RegistryUsers)
{
    reg save HKU\$RegistryUser c:\temp\sec\registry\${Env:ComputerName}-$RegistryUser-NThive
}
Let’s Look at Programs

- First & Foremost: Prefetch
- Startup Programs
- Installed Programs
- Installed Patches
- Flash version
- Executable & Script File in User Folders
# CONTENTS OF PREFETCH FOLDER

```bash
ls C:\Windows\Prefetch | select-object Name, FullName, CreationTime, LastAccessTime, LastWriteTime, Mode | Export-Csv C:\temp\sec\prog\${Env:ComputerName}-PrefetchFolder.csv -NoTypeInformation
```

# Copy prefetch folder contents

```bash
robocopy c:\windows\prefetch C:\temp\sec\prog\prefetch_grab /copyall /ZB /TS /r:4 /w:3 /FP /NP /log+:C:\temp\sec\prog\${Env:ComputerName}-PrefetchGrabLog.txt
```
<table>
<thead>
<tr>
<th>Filename</th>
<th>Created Time</th>
<th>Modified Time</th>
<th>File Size</th>
<th>Process EXE</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTOSBOOT-B00DFAAAD.pf</td>
<td>10/30/2017 3:44:28 PM</td>
<td>3/21/2018 7:38:24 PM</td>
<td>2,698,564</td>
<td></td>
</tr>
<tr>
<td>MPSISGSTUB.EXE-6CB27A06.pf</td>
<td>10/23/2017 13:54:10 AM</td>
<td>3/22/2018 14:11:46 AM</td>
<td>1,260,672</td>
<td></td>
</tr>
<tr>
<td>CSCRIPT.EXE-D1EF4768.pf</td>
<td>11/7/2017 10:37:57 AM</td>
<td>3/22/2018 15:38:51 AM</td>
<td>1,092,224</td>
<td></td>
</tr>
<tr>
<td>GOOGLEUPDATE.EXE-B95715F5.pf</td>
<td>10/30/2017 6:36:01 AM</td>
<td>3/22/2018 8:01:28 AM</td>
<td>55,560</td>
<td></td>
</tr>
<tr>
<td>MSCORSVW.EXE-C3C515BD.pf</td>
<td>3/21/2018 7:38:30 AM</td>
<td>3/22/2018 7:38:30 AM</td>
<td>82,850</td>
<td></td>
</tr>
<tr>
<td>SSSVC.EXE-B0F8131B.pf</td>
<td>11/6/2017 3:08:41 AM</td>
<td>3/22/2018 8:01:35 AM</td>
<td>91,894</td>
<td></td>
</tr>
<tr>
<td>RVHOST.EXE-34F3AEF8.pf</td>
<td>11/6/2017 1:15:05 PM</td>
<td>3/22/2018 8:01:42 AM</td>
<td>24,396</td>
<td></td>
</tr>
<tr>
<td>POLICYHOST.EXE-517A7B9F.pf</td>
<td>10/30/2017 5:01:03 AM</td>
<td>3/22/2018 8:01:54 AM</td>
<td>55,766</td>
<td></td>
</tr>
</tbody>
</table>

**Advanced Options**

**Prefetch Folder:**

`C:\Win\2018-03-06\sec\Pref\prefetch_grab`

- **Browse...**
- **OK**
- **Cancel**
# STARTUP PROGRAM LIST
```
get-wmiobject -class "Win32_StartupCommand" | select-object Name, command, User, Location | Export-Csv C:\temp\sec\prog\${Env:ComputerName}-StartupList.csv -NoTypeInformation
```

# GET INSTALLED PROGRAMS
```
WMIC Product List Full /format:csv | Out-File C:\temp\sec\prog\${Env:ComputerName}-InstalledPrograms.txt
```

# Output Program Files Directory Structures
```
Get-ChildItem -Directory $env:ProgramFiles | out-file C:\temp\sec\prog\${Env:ComputerName}-ProgramFilesDirectories.txt
Get-ChildItem -Directory ${env:ProgramFiles(x86)} | out-file C:\temp\sec\prog\${Env:ComputerName}-ProgramFilesx86Directories.txt
```

# GET INSTALLED PATCHES
```
WMIC qfe list | Out-file C:\temp\sec\prog\${Env:ComputerName}-InstalledPatches.txt
```

SANS Blue Team Summit 2018 – Mike Lombardi, Vertigrate
Flash & Interesting User Executables

#Search for Flash

get-childitem $env:SystemRoot\SysWOW64\Macromed\Flash | out-file C:\temp\sec\prog\${Env:ComputerName}-Flash64bitHost.txt
get-childitem $env:SystemRoot\System32\Macromed\Flash | out-file C:\temp\sec\prog\${Env:ComputerName}-Flashx86Location.txt

#List file types from Users directory that may indicate malicious file along with NTUser file metadata.

$extensions="*.cmd","*.bat","*.vbs","*.js","*.com","*.exe","*.wsf","*.swf","*.jar","*.dat"

Get-ChildItem -Recurse c:\Users -Include $extensions | out-file C:\temp\sec\prog\${Env:ComputerName}-FileTypesOfInterest.txt
Let’s Loop Through Users

- #Build list of user folders.
- Get-ChildItem -Directory c:\Users -Name | out-file C:\temp\sec\prog\${Env:ComputerName}-UserFolders.txt

- Web: WebCache, Firefox, Chrome
- Jumplists
#Section designed to iterate User profile folders and then grab each profile's WebCache folder.

#At end of script b/c command shuts down certain services and processes. Best if machine is restarted once the entire collection is finished.

#May only need the `taskkill` for `taskhost.exe` command on certain systems.

```powershell
net stop COMSysApp

Get-ChildItem -Directory c:\Users -Name | out-file C:\temp\sec\prog\${Env:ComputerName}-UserFolders.txt

$userfolders = Get-Content C:\temp\sec\prog\${Env:ComputerName}-UserFolders.txt

ForEach ($userfolder in $userfolders)
{
    robcopy c:\Users\$UserFolder\AppData\Local\Microsoft\Windows\WebCache C:\temp\sec\internet\${Env:ComputerName}\$UserFolder\WebCache /copyall /ZBE /TS /r:4 /w:3 /FP /NP /log+:C:\temp\sec\internet\${Env:ComputerName}\$UserFolder-WebCacheV01GrabLog.txt
}

net start ComSysApp
```
# Grab Firefox SQLite files, if available, from each user profile.

```powershell
$UserFolders = Get-Content C:\temp\sec\prog\${Env:ComputerName}-UserFolders.txt
ForEach ($UserFolder in $UserFolders)
{
    robocopy C:\Users\$UserFolder\AppData\Roaming\Mozilla\Firefox\Profiles C:\temp\sec\internet\${Env:ComputerName}-$UserFolder\Firefox*.sqlite /S /copyall /ZB /TS /r:4 /w:3 /FP /NP /log+:C:\temp\sec\internet\${Env:ComputerName}-$UserFolder-FirefoxGrabLog.txt
}
```

# Grab all Google Chrome data, if available, from each user profile. CAREFUL!! Can be large. May skip some files like Current Tabs if they're in use.

```powershell
$UserFolders = Get-Content C:\temp\sec\prog\${Env:ComputerName}-UserFolders.txt
ForEach ($UserFolder in $UserFolders)
{
    robocopy "C:\\Users\$UserFolder\AppData\Local\Google\Chrome\User Data" C:\temp\sec\internet\${Env:ComputerName}-$UserFolder\Chrome /E /copyall /ZB /TS /r:4 /w:3 /FP /NP /log+:C:\temp\sec\internet\${Env:ComputerName}-$UserFolder-ChromeGrabLog.txt
}
```
<table>
<thead>
<tr>
<th>URL</th>
<th>Title</th>
<th>Visited On</th>
<th>Visit Count</th>
<th>Typed Count</th>
<th>Referrer</th>
<th>Visit ID</th>
<th>Profile</th>
<th>URL Length</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://www.google.com/search?q=intel+warns+spectre+patch&amp;rlz=1C1">https://www.google.com/search?q=intel+warns+spectre+patch&amp;rlz=1C1</a>...</td>
<td>intel warns spectre patch - Google Search</td>
<td>2/5/2018 2:54:23 PM</td>
<td>1</td>
<td>0</td>
<td></td>
<td>4795</td>
<td>Default</td>
<td>163</td>
</tr>
</tbody>
</table>
#Grab Jump Lists for each user found on the system. Automatic and pinned (custom).

```powershell
$UserFolders = Get-Content C:\temp\sec\prog\${Env:ComputerName}-UserFolders.txt
ForEach ($UserFolder in $UserFolders)
{
    robocopy "C:\Users\$UserFolder\AppData\Roaming\Microsoft\Windows\Recent\AutomaticDestinations" C:\temp\sec\jumplists\${Env:ComputerName}-$UserFolder /E /copyall /ZB /TS /r:4 /w:3 /FP /NP /log+:C\temp\sec\jumplists\${Env:ComputerName}-$UserFolder-jumplists.txt

    robocopy "C:\Users\$UserFolder\AppData\Roaming\Microsoft\Windows\Recent\CustomDestinations" C:\temp\sec\jumplists\${Env:ComputerName}-$UserFolder /E /copyall /ZB /TS /r:4 /w:3 /FP /NP /log+:C\temp\sec\jumplists\${Env:ComputerName}-$UserFolder-jumplists.txt
}
```
<table>
<thead>
<tr>
<th>Source File Name</th>
<th>Jump List Type</th>
<th>App ID</th>
<th>App ID Description</th>
<th>Link File Count</th>
<th>File Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>…</td>
<td>Automatic</td>
<td>2b466277c90c16e2</td>
<td>Windows Explorer Pinned and Recent</td>
<td>47</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td>Automatic</td>
<td>2b466277c90c16e2</td>
<td>Remote Desktop</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td>Custom</td>
<td>5f7566c36735e0ae</td>
<td>Unknown AppId</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td>Custom</td>
<td>5f7566c36735e0ae</td>
<td>Unknown AppId</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td>Automatic</td>
<td>8d86ec614623b1c3</td>
<td>Chrome 6.0, 807.84</td>
<td>12.0, 742.100</td>
<td>13.0, 785.316</td>
</tr>
</tbody>
</table>

**Properties**
- AppId: 8d86ec614623b1c3
- AppId Description: Windows Explorer Pinned and Recent
- Pinned Count: 5
- Exit Count: 47
- Last Used Entry #: 57
- Version: 1

---

SANS Blue Team Summit 2018 – Mike Lombardi, Vertigrate

4/24/2018
#LIST SHADOW COPIES

C:\windows\system32\vssadmin list shadows | Out-File C:\temp\sec\prog\${Env:ComputerName}-ShadowCopyList.txt
More Stuff Just in Case…

- Full C: drive file listing (time-consuming)
- Program Files Directory structures
To Do List

- Multi-thread
- $MFT, $USNJrnl, $LogFile
- Add and Colorize Some Console Output
- Command switches for some variables
- Account for the occasional username with spaces
- Grab the .LOG files associated with Registry Hives
- Grab a copy of any suspicious files found in user directories
- ZIP the results
What Do We Get?

- Local Admins & Groups
- Areas of Persistence
  - Services, Scheduled Tasks, Startup, etc.
- Installed Programs & Patches
- Listing of Shadow Copies
- Live Statistics
  - Processes, Network Stats, etc.
- Big 3 Event Logs, plus PS and Tasks
- SYSTEM, SOFTWARE & SAM Registry Hives
- All User Registry Hives
- Prefetch Listing & Contents
- List of User Folders, plus contents for
  - Browsers
  - Jumplists
  - Link Files
- Full File Listing if Desired
Questions?

Where to Find Me:

- LinkedIn
- Twitter @vertigrate
- www.vertigrate.com
- Supporting HTCIA and ISSA Meetings
- SANS Events