

# Building A Malware Analysis Platform At Home

Jared Stroud (@DLL\_Cool\_J)

---

April 2023

2023



# Disclaimer

---

**The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with or support for the positions, opinions or viewpoints expressed by the author. 2023 The MITRE Corporation. ALL RIGHTS RESERVED**

# Agenda & Goal

This is a DIY Malware Analysis Platform course! You will be building a process to download, extract, and analyze metadata from malware! This two hour course is going to try to expose you to a bunch of different things and leave you with the ability to dive into deeper subjects independently.

- |   |                            |   |                 |
|---|----------------------------|---|-----------------|
| 1 | Identifying Our Goals      | 4 | Data Extraction |
| 2 | Malware & where to find it | 5 | Data Analysis   |
| 3 | Automating Collection      | 6 | Recap & Survey  |

# \$> whoami

---

- Currently:
  - Lead Security Engineer at The MITRE Corporation
  - Adjunct Lecturer at Rochester Institute of Technology
- Previously:
  - Malware Analyst/Security Researcher
- Presented at:
  - ATT&CKCON
  - BSides Roc
  - DEF CON 29 Packet Hacking Village
  - DFRWS-EU
  - ShmooCon Firetalk

# The Lab Environment!



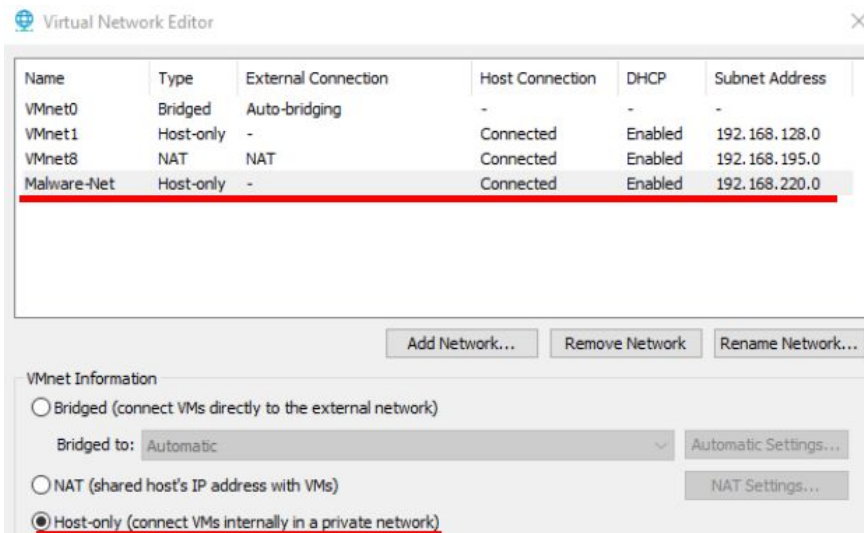
- The entire lab environment has been instrumented via Vagrant!
  - Get it here: <https://developer.hashicorp.com/vagrant/downloads>
- This is to ensure we all have the same environment and tools/tool version when poking at the labs.
  - If you want to use your own distro/VM, that's fine too.
- We are working with **live** malware, **you** are responsible for your machine.
  - Take snapshots/backups/etc...



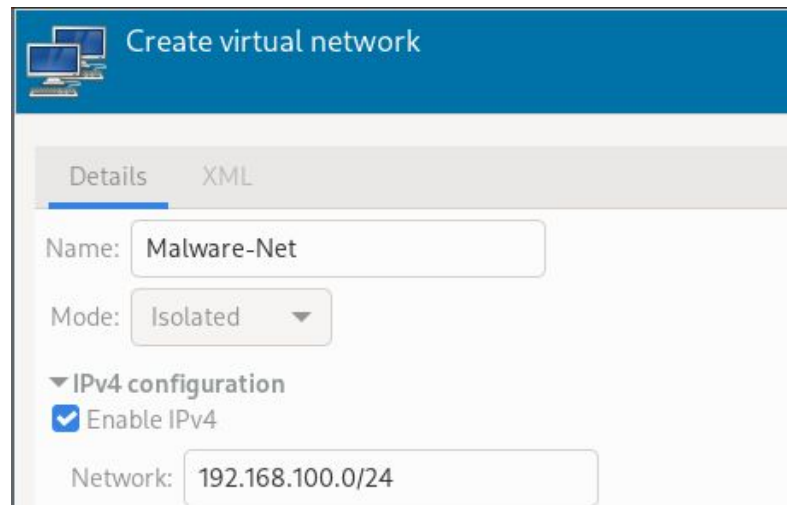
# Configuring Networks for Analysis



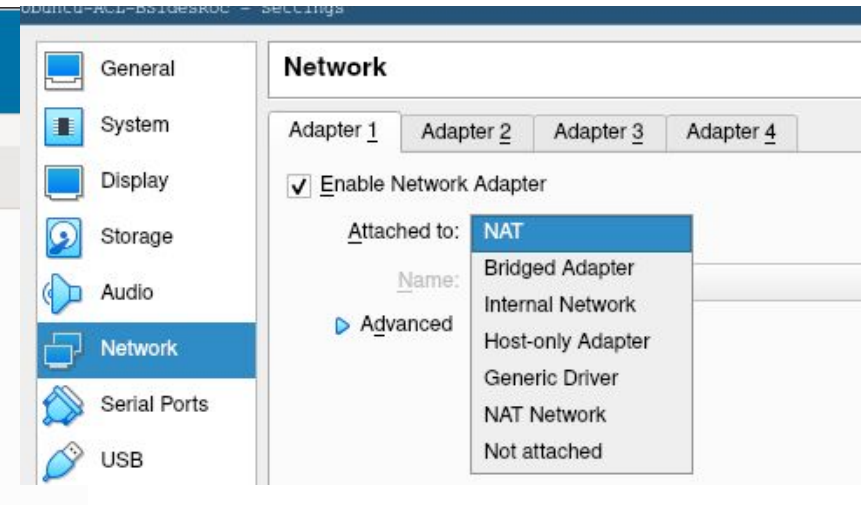
- Dynamic analysis can influence static analysis and vice versa.
  - Capturing traffic in a safe manner is critical.
  - Nothing ruins a day faster than accidentally executing malware in a insecure way.
  - Have a known good state w/ snapshots and do a dry run prior to malware execution.



VMware Desktop



virt-manager



Virtualbox



# Configuring Networks for Analysis – “Smoke Test”



```
(dllcoolj@pwnhost3000)-[~/bots]
$ ping 8.8.8.8
ping: connect: Network is unreachable

(dllcoolj@pwnhost3000)-[~/bots]
$ ping google.com
ping: google.com: Temporary failure in name resolution
```



# What do YOU want to achieve?

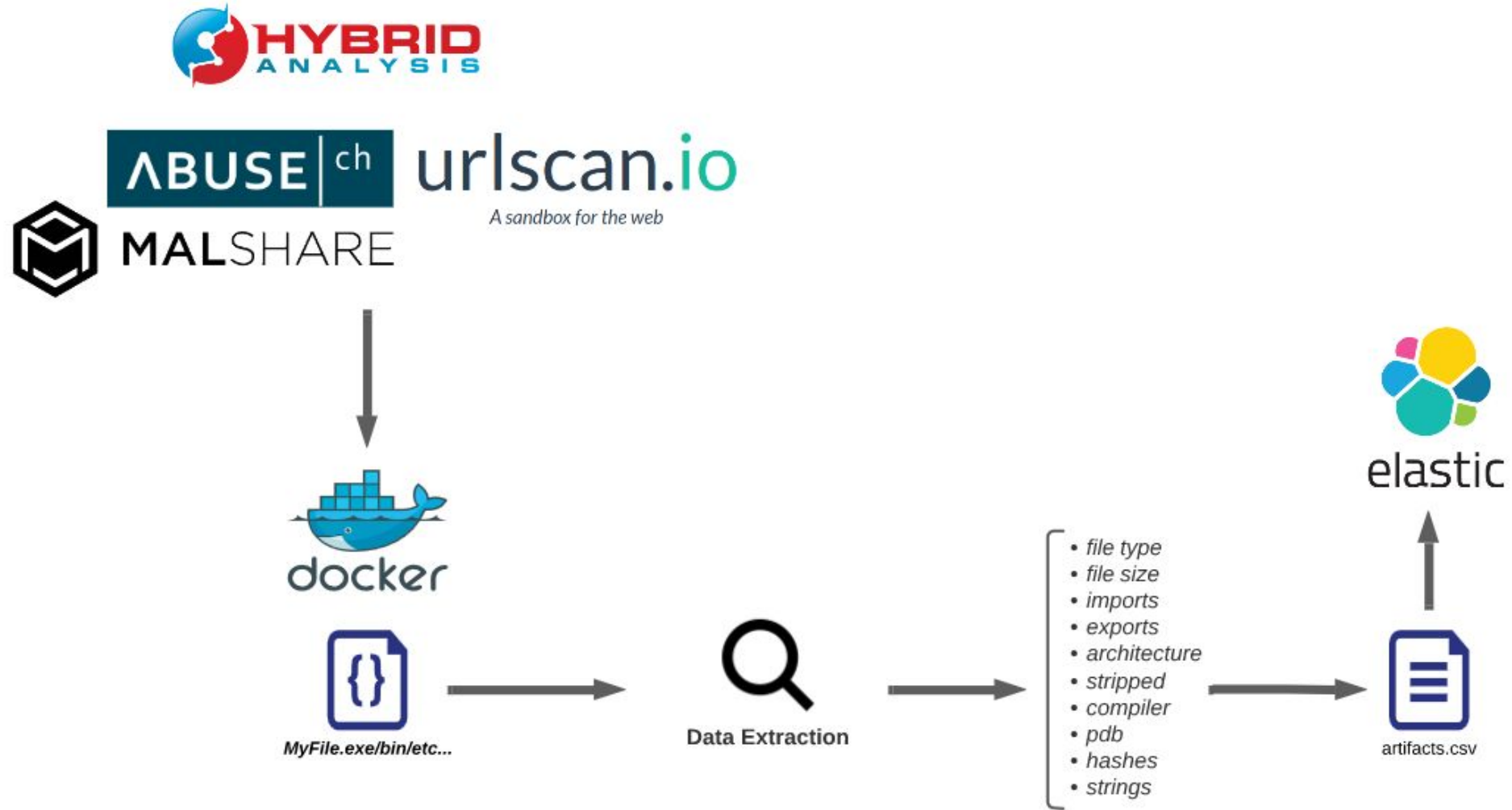


- No one has the same 24 hours in a day.
- How you invest them is critical to what you receive from said investment.  
Malware Analysis/Homelab Malware Platforms help build a variety of skills.
- For example:
  - Analyzing the Malware? - **Reverse Engineering/DFIR** skills
  - Building software to collect and analyze malware? - **Software Engineering**
  - Automating the downloads and passing data to different microservices? - **DevOps**
  - Blogging - **Writing/Presentation/Soft Skills**





# What are we Building?



# Malware and Where to Find It



- Well curated samples
- Historic APT Campaigns
- Source Code
- Papers



- Daily dumps of IoCs
- Search for hashes
- YARA Rule Hunting



GREYNOISE



- IP telemetry
- Domain telemetry

*Note: Not an exhaustive list*



# VX-Underground – Case Study 3CX VoIP



**vx-underground** @vxunderground · 1d

We have malware samples from the recent 3CX VOIP supply chain attack.

- SmoothOperator.7z
- 48.1MB compressed
- Samples from CrowdStrike and SentinelOne reports

You can download the malware samples here: [share.vx-underground.org](https://share.vx-underground.org)

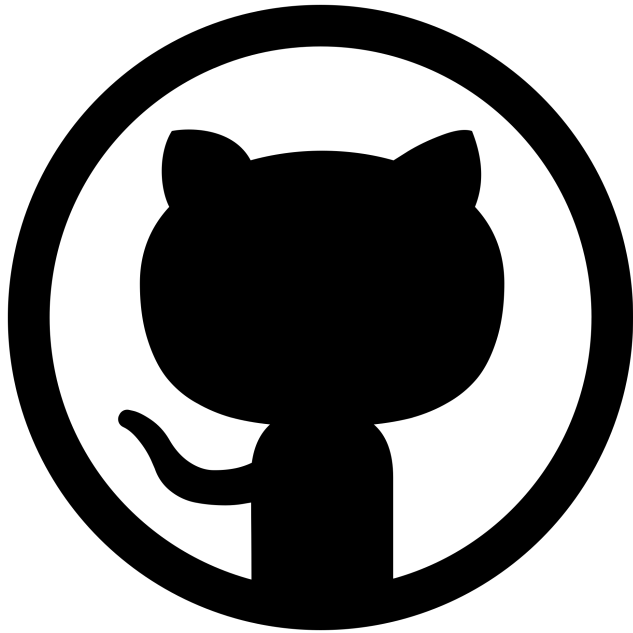


- Alleged DPRK threat actors
- Less than 24 hours samples were available via VX-Underground

*Note: Not an exhaustive list*



# Malware and Where to Find It



- Malware Dumps (theZoo)
- IoT Botnet Source Code
- ...



- Honeypots
- Web Server logs?

*Note: Not an exhaustive list*



# Malware and Where to Find It – Notable Mentions



- Sophos Reversing Labs “SOREL” 20 Million Samples
- Dataset for ML purposes, but also great for RE



# Provisioning Our Malware Machine!



- **Vagrant** is a utility that allows you to quickly spin up Virtual Machines.
  - These Virtual Machines could be enabled via Virtualbox, Libvirt, or VMWare\*
    - VMWare requires a paid plugin
- Provisioning these machines can be achieved via shell commands, Chef, or Ansible.
  - Our demo will have us completing the provisioning with Ansible



# Daily Malware Dumps!



- Everyday there's a new article on a new malware variant/technique/etc... being exploited in the wild.
- Some of these services provide daily dumps of ongoing campaigns, but not all let you download.
  - Malshare/Hybrid-Analysis/Abuse.ch allows you to download samples for **free!**
    - Today, we'll focus on malshare
- Interacting with these services requires an account to get an API key to then download the daily dumps.
- How do we identify *what* these binaries are?
  - YARA!



# YARA Rules



- Why YARA?
  - Industry standard rule format for identifying malicious family of binaries.
  - Open Source (BSD License)
  - Significant amount of examples on the internet to use for scanning
- Combining Daily Dumps of malware w/ YARA Rules can help us identify what a given sample is or what features the malware contains.

```
rule silent_banker : banker
{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true

    strings:
        $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
        $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
        $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"

    condition:
        $a or $b or $c
}
```

*Example from <https://github.com/VirusTotal/yara>*





# YARA Rules - Python Documentation



```
rule silent_banker : banker
{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true

    strings:
        $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
        $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
        $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"

    condition:
        $a or $b or $c
}
```

*Example Python yara Detection*



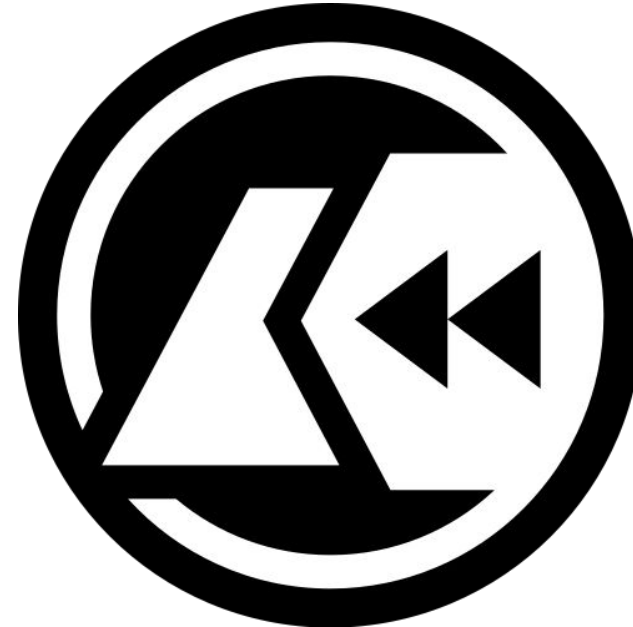
# Some Tools of the Trade



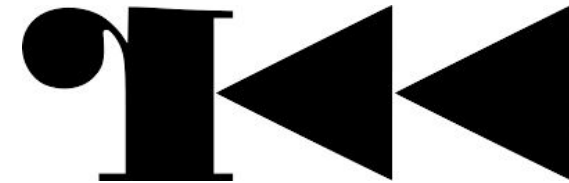
**IDA Pro**  
([hex-rays.com](https://hex-rays.com))



**Ghidra**  
([ghidra-sre.org](https://ghidra-sre.org))



**Cutter**  
([cutter.re](https://cutter.re))

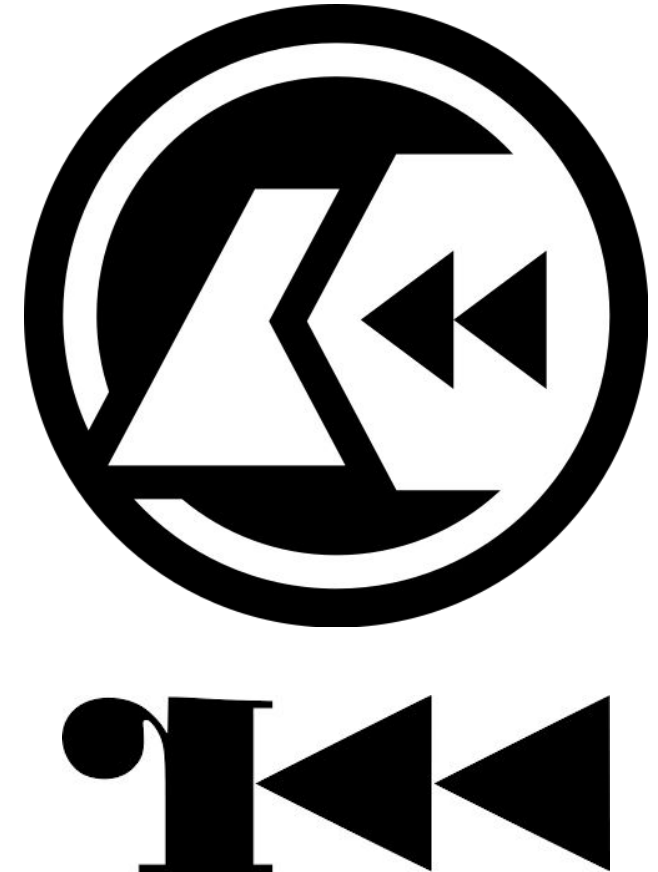


**Radare2**  
([radare2.org](https://radare2.org))

# The labs use radare2, Why?



- Free
- Cross Platform
- Lets you rapidly explore binaries all via the console.
- Allows you to explore and understand file formats.
- Excellent for automation
- GUI components (cutter) integrate some of the best parts of both IDA & Ghidra
  - Ghidra's Decompiler
  - IDA's Graph
- Cons: high learning curve (*think vim, but for RE*)



# What is a PE/ELF?



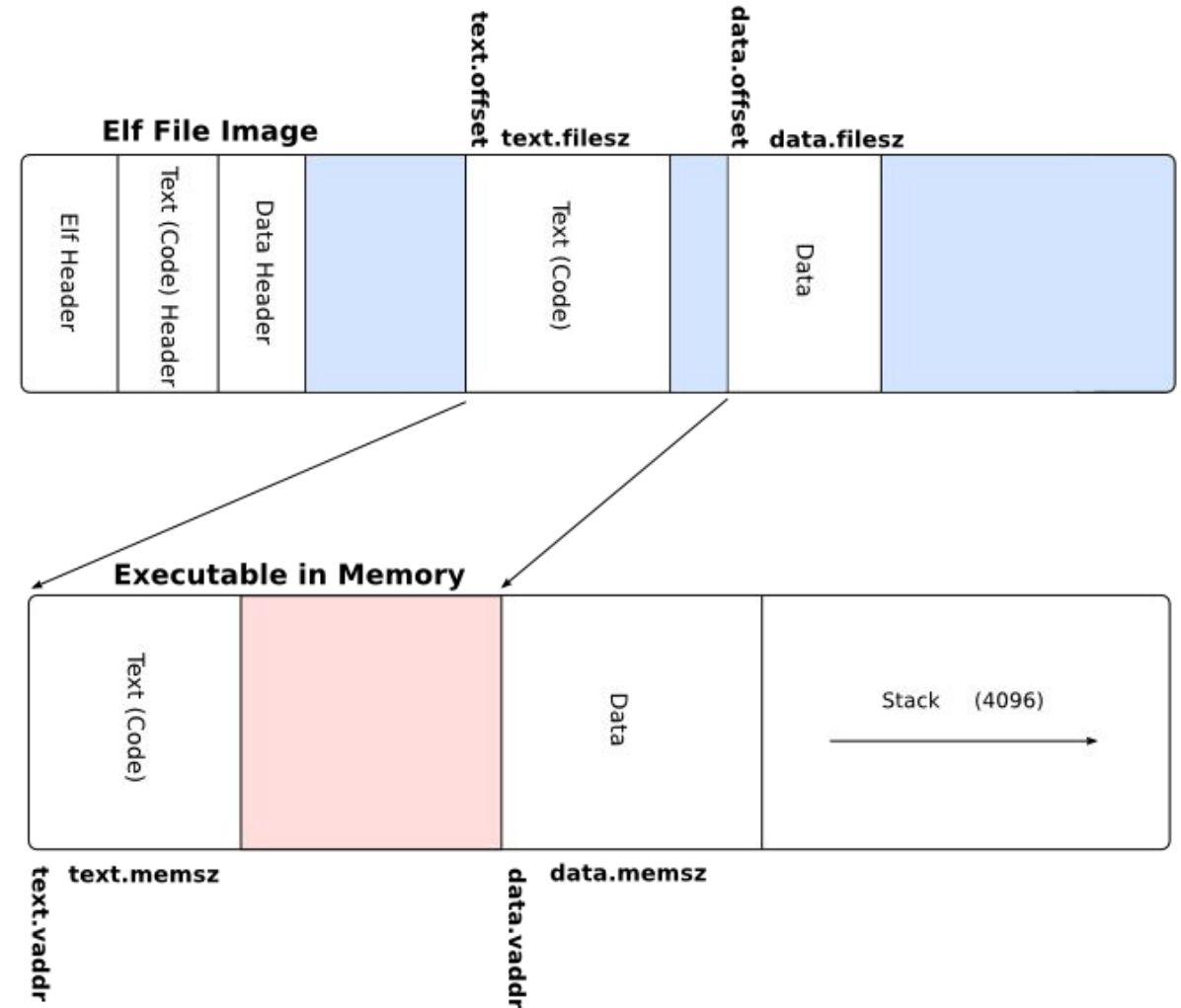
- Executable files contain a structure that the underlying Operating System loader understands how to parse in order to allocate memory, copy sections into memory, load additional files and ultimately begin execution of the process.
- A file is broken down into **headers, sections and segments (program headers)**.
  - Think of **headers** like the table of contents in a book.
    - *It tells you where to look for a given topic.*
  - **Sections:** used by the linker to build an executable
  - **Segments** contain data for runtime.



# Preventing Accidental Execution



- Ensure binary is NOT executable on download.
  - During file creation we can make binaries read only.
- How does the OS execute a binary?
- TL;DR: The OS loader parses the file format (PE/ELF) to then allocate memory regions to copy the given regions of a binary into memory and start execution.
  - **If the header is broken this process stops.**
  - A fake header can prevent accidental execution of binaries in our environment.



Source: <https://wiki.osdev.org/ELF>



# Checking and Breaking Headers for ELFs



```
ELF Header:
Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
Class:                               ELF64
Data:                                   2's complement, little endian
Version:                             1 (current)
OS/ABI:                             UNIX - System V
ABI Version:                         0
Type:                                DYN (Position-Independent Executable file)
Machine:                            Advanced Micro Devices X86-64
Version:                            0x1
Entry point address:                 0x1040
Start of program headers:            64 (bytes into file)
Start of section headers:           13496 (bytes into file)
Flags:                               0x0
Size of this header:                 64 (bytes)
Size of program headers:             56 (bytes)
Number of program headers:           13
Size of section headers:            64 (bytes)
Number of section headers:           30
Section header string table index: 29
```

Machine: 0x3e (62)

```
ELF Header:
Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
Class:                               ELF64
Data:                                   2's complement, little endian
Version:                             1 (current)
OS/ABI:                             UNIX - System V
ABI Version:                         0
Type:                                DYN (Position-Independent Executable file)
Machine:                             None
Version:                             0x1
Entry point address:                 0x1040
Start of program headers:            64 (bytes into file)
Start of section headers:           13496 (bytes into file)
Flags:                               0x0
Size of this header:                 64 (bytes)
Size of program headers:             56 (bytes)
Number of program headers:           13
Size of section headers:            64 (bytes)
Number of section headers:           30
Section header string table index: 29
```

Machine: 0x00

- Note: this **will** change the underlying hash
- `man 5 elf` to see expected header values
- Automating with Radare2:

```
r2 -w -c 'aaa;s 0x11; w0 2;q' -q a.out
```

- `aaa`: analyze
- `s`: seek to offset 0x11 in the binary (*Machine ID in the ELF header*)
- `w0 2`: write 2 zeros at this location
- `q`: quit



# Simple but effective! Case study: SoRel Data Set



- Case example: **Sophos Reversing Labs SoRel** data set.
  - **20 Million** malware samples (Windows) labeled for machine learning/data analysis purposes.
  - Modified *FileHeader.Machine header & FileHeader.OptionalHeader.Subsystem* to prevent accidental execution.
  - **Modifications to the file itself changes the file hash!**
    - SoRel has the file name of a binary as the original file hash.



# Lab – Automating Collection from Malshare

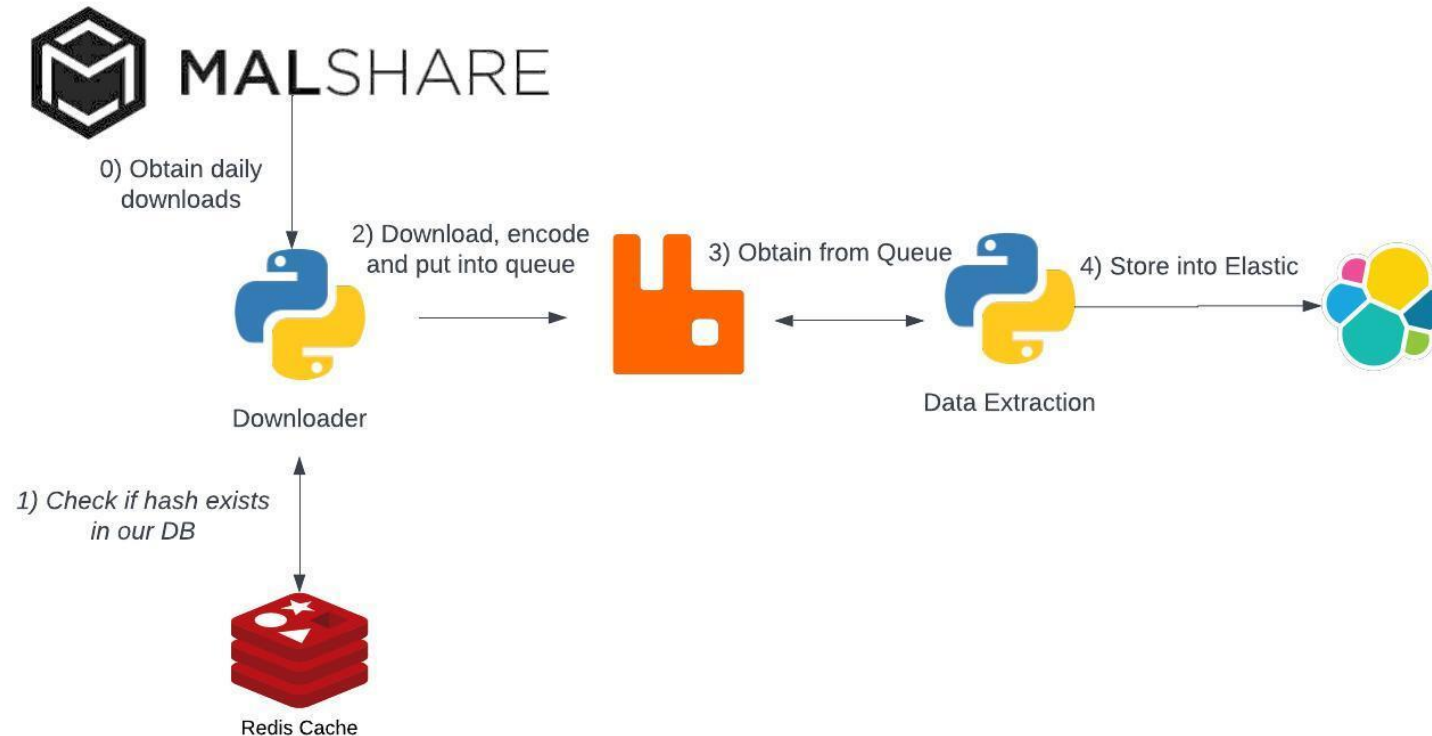


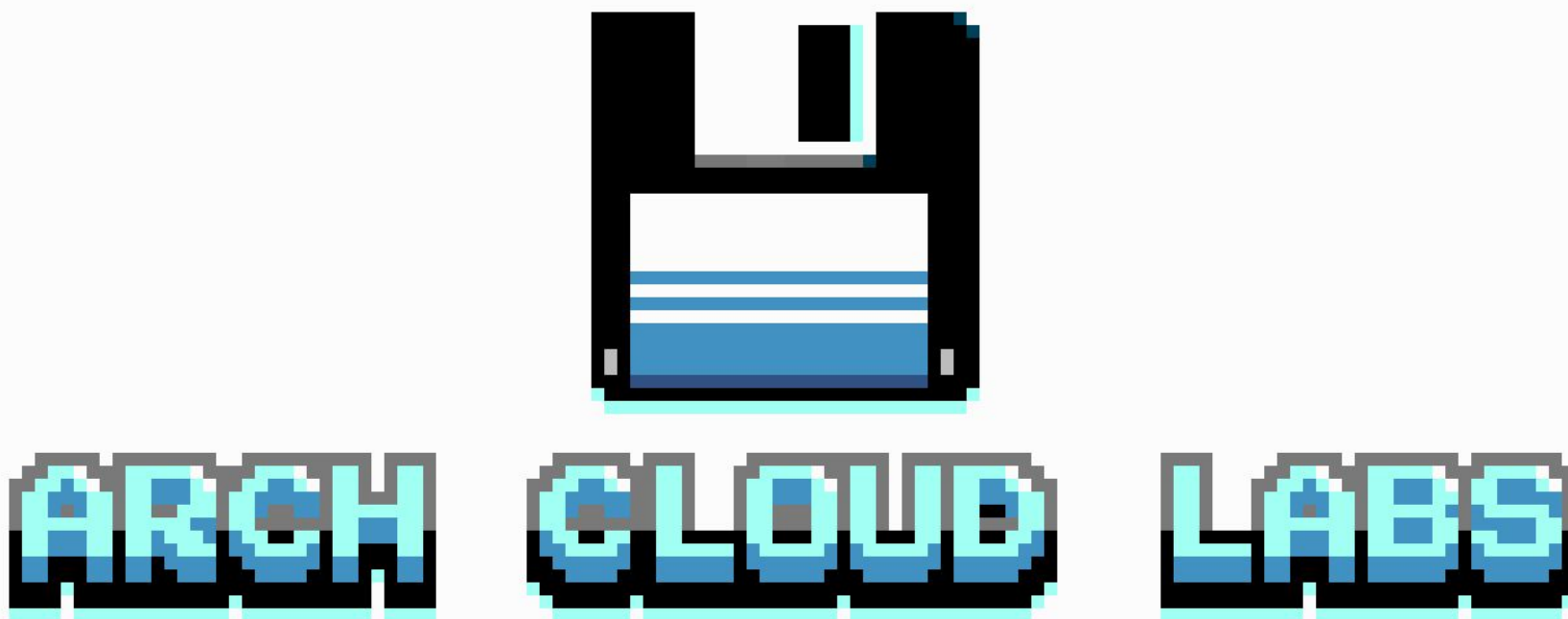


# Beyond The Course – Scaling Malware Collection



- Single points of failure are Bad!
  - If the Python script fails, well there goes the data...
  - If we restart the python script as is, we'll reingest all the previously seen data as well.
  - How do we plan for this?





Break!

# Grouping Related files Together



- **NOTE:** ELF Binary
- **PDB:** Debug string containing path to project on Host machine.
  - *“The data is stored in a separate file from the executable to help limit the size of the executable, saving disk storage space and reducing the time it takes to load the data. This methodology also allows the executable to be distributed without disclosing this significant information which could make the program easier to reverse engineer.” - [MSDN Documentation](#)*
- **IMPHash (Import Hashing):** Hash the ways binaries import libraries
- **Fuzzing Hashing (SSDEEP):** Hashing N-number of bytes together to identify similar files.



# The Art of Recreating Analysis



- Why recreate analysis of other blogs?
  - Better understanding of the DFIR/RE process.
    - Opportunity to **automate** a process
  - Potentially identify things another analyst missed.
  - Start identifying common patterns/mechanisms in Malware
    - Ex: LoadLibrary/RegQueryA/etc...
- Find interesting patterns in your own samples!



# Recreating Analysis Case Study: Mandiant Blog



- [Definitive Dossier of Devilish Debug Details – Part One: PDB Paths and Malware](#)
  - Steve Miller, Mandiant Blog 2019
- TL;DR linking together threat actors based on shared PDB file paths.
- Radare2 provides an easy way to gather this data from the command line



*Shared PDB Paths Among APT-1 Threat Actors*

# PDBs Can Also Be Very Telling



```
D:\Monthly Task\August 2011\USB Prop\Usb Propagator.09-24\nn\Release\nn.pdb
Y:\Uploader\HTTP\HTTP Babylon 5.1.1\HTTP Babylon 5.1.1\Httpbackup\Release\HttpUploader.pdb
e:\Project\mm\Wininet\Attack\MiniAsp3\Release\MiniAsp.pdb
d:\Projects\WinRAR\SFX\build\sfxrar32\Release\sfxrar.pdb
C:\Documents and Settings\Admin\Desktop\Newuploader\Release\Newuploader.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
C:\Ocean\Project-VS2008\el.4\Release\AutoLoad_DLL.pdb
E:\Datahelp\SCode\BOT\MATRIX_1.3.3\CLIENT\Build\Win32\Release\appinclient.pdb
C:\BNaga\SCode\BOT\MATRIX_1.2.2.0\appinbot_1.2_120308\Build\Win32\Release\deleter.pdb
f:\keylogger\KeyLog\keytest1\keytest\taskmnq.pdb
d:\Projects\WinRAR\SFX\build\sfxrar32\Release\sfxrar.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
C:\BNaga\kaam\kaam\New_FTP_HttpWithLatestfile2_FirstBlood_Released\New_FTP_HttpWithLatestfile2\Release\Ron.pdb
F:\Utility\Release\Utility.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
d:\Projects\WinRAR\SFX\build\sfxrar32\Release\sfxrar.pdb
t:\final project backup\complete task of ad downloader & usb grabber&uploader\New folder\with icon +shortcut link
Over 1.5.3 (Startup)\Release\Http_t.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
T:\final project backup\uploader version backup\fud all av hangover1.5.4\with icon +shortcut link\HangOver 1.5.3 (
up)\Release\Http_t.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
C:\Documents and Settings\Administrator\Desktop\UsbP\Release\UsbP.pdb
D:\Monthly Task\August 2011\USB Prop\Usb Propagator.09-24\nn\Release\nn.pdb
Z:\Uploader\HTTP\ron uplo\RON 2.0.0\Release\Ron.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
D:\Projects\Elance\AppInSecurityGroup\FtpBackup\Release\Backup.pdb
E:\C\moon1.5\Release\MoonService2.pdb
d:\Projects\WinRAR\SFX\build\sfxzip32\Release\sfxzip.pdb
```

*PDBs from VX Underground Malware Samples*





# R2ELK – Radare2-to-ELK



- <https://www.github.com/archcloudlabs/r2elk>
- Automatically extract metadata from Executables and import them into Elasticsearch
- Useful for bulk analysis to then upload into Elasticsearch

↑ imphash	▼ dbg_file	▼ md5
2b32e0511524544aac45d0cfd2fb8cd3	E:\code\moon1.5\Release\MoonClient2.pdb	8e813f38167adb3fd5ae383c2b263db9
d0ff6dfb0f8157c5376a630d8c562a47	pchsvc.pdb	5e5d040913dbf7b7f195ef8f7eb1ed12
2b32e0511524544aac45d0cfd2fb8cd3	D:\M tools\Moon\Release\MoonClient2.pdb	e0b68a555262df2c64dfe14b0ed8fb38
67bd577347c27541634483619f962b3c	E:\XiaoME\AiH\20120410\Attack\MiniAsp3\Release\MiniAsp.pdb	c2e61c98946073040a55924c123419d2
bd413965aa18045a3fea462ca1473b37	E:\XiaoME\SunCloud-Code\Eclipse_A\Release\Eclipse_Client_B.pdb	03b9bbf42c9caf535738a16fb430c95d
4557b502e756a3acfc77d7bd38f2078e	d:\Projects\WinRAR\rar\build\rar32\Release\RAR.pdb	4abeb4095b68bc93a377ec0d32bf5624
5776b1400eb618f9f213ae9dee30ce2f	d:\drizt\projects\auriga\branches\stone_2\server\exe\i386\riodrv32.pdb	b4364a67d9c2349ebf2ebc114d9c677b
bd413965aa18045a3fea462ca1473b37	E:\pjts2008\Eclipse_A\Release\Eclipse_Client_B.pdb	78c5847bc10d683b21485a62bf188ddf
bd413965aa18045a3fea462ca1473b37	E:\XiaoME\SunCloud-Code\Eclipse_A\Release\Eclipse_Client_B.pdb	0ae3fcf5cb5f813d9cc5272b5ef8f96d

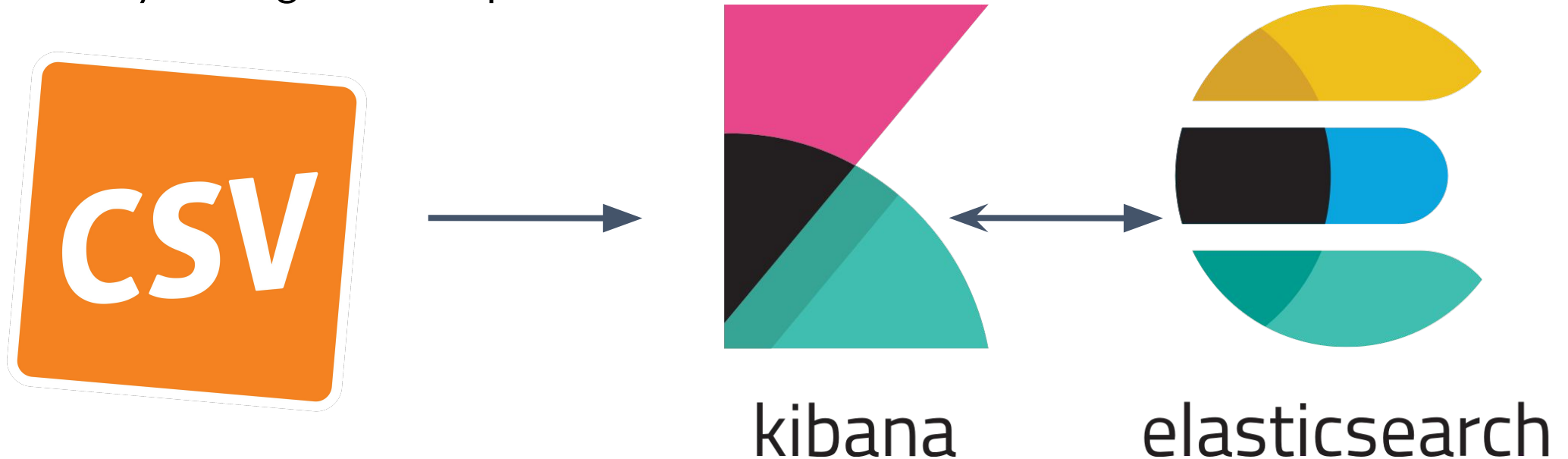
IMPHash & PDB Analysis of APT1 Samplest



# Analyzing our Data in Kibana/Elasticsearch



- **First, execute the following from the home folder:** `$> docker compose up -d`
- **Kibana:** Front end to Elasticsearch.
- **Elasticsearch:** popular Open Source database for logs.
  - Observability/Security/Analytics/etc...
- Our provisioning scripts already have this setup and running.
- Browse to your Vagrant IP on port 5601 in a web browser to access Kibana





# Analyzing our Data in Kibana/Elasticsearch



	↓ @timestamp 🕒	▼ md5	▼ imphash	▼ yara_rules
				APT1_WEBEC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]
✓ <input type="checkbox"/>	Apr 1, 2023 @ 21:52:23.171689000	f410474099dca541f621b0393c104045	0d72b49ed68430225595cc1efb43ced9	[SEH_Init, network_http, network_dropper, win_mutex, win_registry, win_token, win_files_operation, Str_Win32_Wininet_Library, Str_Win32_Internet_API, Str_Win32_Http_API, maldoc_find_kernel32_base_method_1, APT1_WEBEC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]
✓ <input type="checkbox"/>	Apr 1, 2023 @ 21:52:17.822725000	4f33caa8da60de15e6d99171e29955c6	0d72b49ed68430225595cc1efb43ced9	[SEH_Init, network_http, network_dropper, win_mutex, win_registry, win_token, win_files_operation, Str_Win32_Wininet_Library, Str_Win32_Internet_API, Str_Win32_Http_API, maldoc_find_kernel32_base_method_1, APT1_WEBEC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]
✓ <input type="checkbox"/>	Apr 1, 2023 @ 21:52:04.666829000	8f65f01ecce82700582deddf49c7c1bf	0d72b49ed68430225595cc1efb43ced9	[SEH_Init, network_http, network_dropper, win_mutex, win_registry, win_token, win_files_operation, Str_Win32_Wininet_Library, Str_Win32_Internet_API, Str_Win32_Http_API, maldoc_find_kernel32_base_method_1, APT1_WEBEC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]
✓ <input type="checkbox"/>	Apr 1, 2023 @ 21:51:48.497675000	4f33caa8da60de15e6d99171e29955c6	0d72b49ed68430225595cc1efb43ced9	[SEH_Init, network_http, network_dropper, win_mutex, win_registry, win_token, win_files_operation, Str_Win32_Wininet_Library, Str_Win32_Internet_API, Str_Win32_Http_API, maldoc_find_kernel32_base_method_1, APT1_WEBEC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]



# Analyzing our Data in Kibana/Elasticsearch



↓ @timestamp 🕒	md5	imphash	yara_rules
<pre>rule APT1_WEBC2_GREENCAT {     meta:         author = "AlienVault Labs"         info = "CommentCrew-threat-apt1"      strings:         \$1 = "reader_sl.exe" wide ascii         \$2 = "MS80547.bat" wide ascii         \$3 = "ADR32" wide ascii         \$4 = "ControlService failed!" wide ascii      condition:         3 of them }</pre>		0d72b49ed68430225595cc1efb43ced9	[APT1_WEBC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]
		0d72b49ed68430225595cc1efb43ced9	[SEH_Init, network_http, network_dropper, win_mutex, win_registry, win_token, win_files_operation, Str_Win32_Wininet_Library, Str_Win32_Internet_API, Str_Win32_Http_API, maldoc_find_kernel32_base_method_1, APT1_WEBC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]
		0d72b49ed68430225595cc1efb43ced9	[SEH_Init, network_http, network_dropper, win_mutex, win_registry, win_token, win_files_operation, Str_Win32_Wininet_Library, Str_Win32_Internet_API, Str_Win32_Http_API, maldoc_find_kernel32_base_method_1, APT1_WEBC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]
		0d72b49ed68430225595cc1efb43ced9	[SEH_Init, network_http, network_dropper, win_mutex, win_registry, win_token, win_files_operation, Str_Win32_Wininet_Library, Str_Win32_Internet_API, Str_Win32_Http_API, maldoc_find_kernel32_base_method_1, APT1_WEBC2_GREENCAT, IsPE32, IsWindowsGUI, HasRichSignature]

[https://github.com/Yara-Rules/rules/blob/85cb1fad9a58efedc71f696eb334e0226a166ba0/malware/APT\\_APT1.yar#L950](https://github.com/Yara-Rules/rules/blob/85cb1fad9a58efedc71f696eb334e0226a166ba0/malware/APT_APT1.yar#L950)




# Lab – Data Extraction & Labeling




# Lab – Data Extraction & Labeling




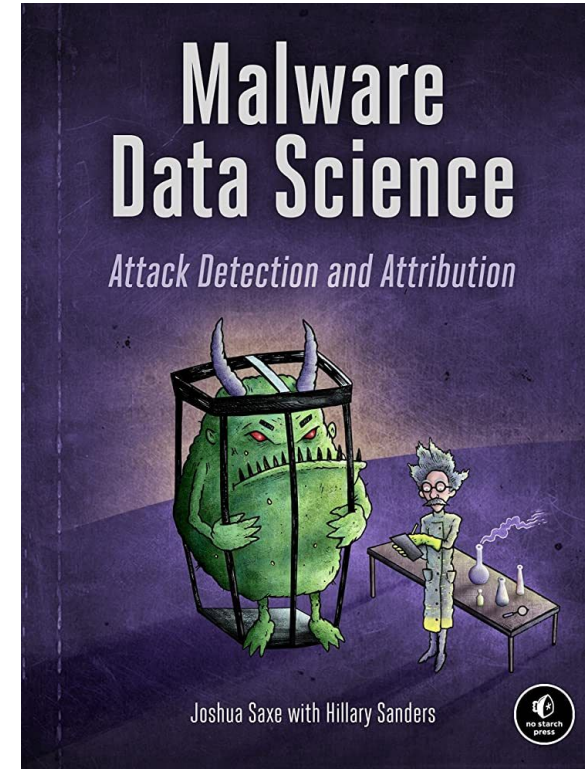
SUNDAY

 **Jared Stroud** (He/Him) • 10:34 PM  
Hello! I'm writing to request permission to cite and use the publicly available Mlaware Data Science dataset in an upcoming free malware workshop. The book helped me on my journey and I'd like to use the malware samples to help others and hopefully have them engage with the book after the con.

 **Joshua Saxe** • 10:34 PM  
Thanks, and sure, go ahead!

MONDAY

 **Jared Stroud** (He/Him) • 7:52 AM  
Awesome thanks!



Data Set: <https://tinyurl.com/yc6j7cmr>

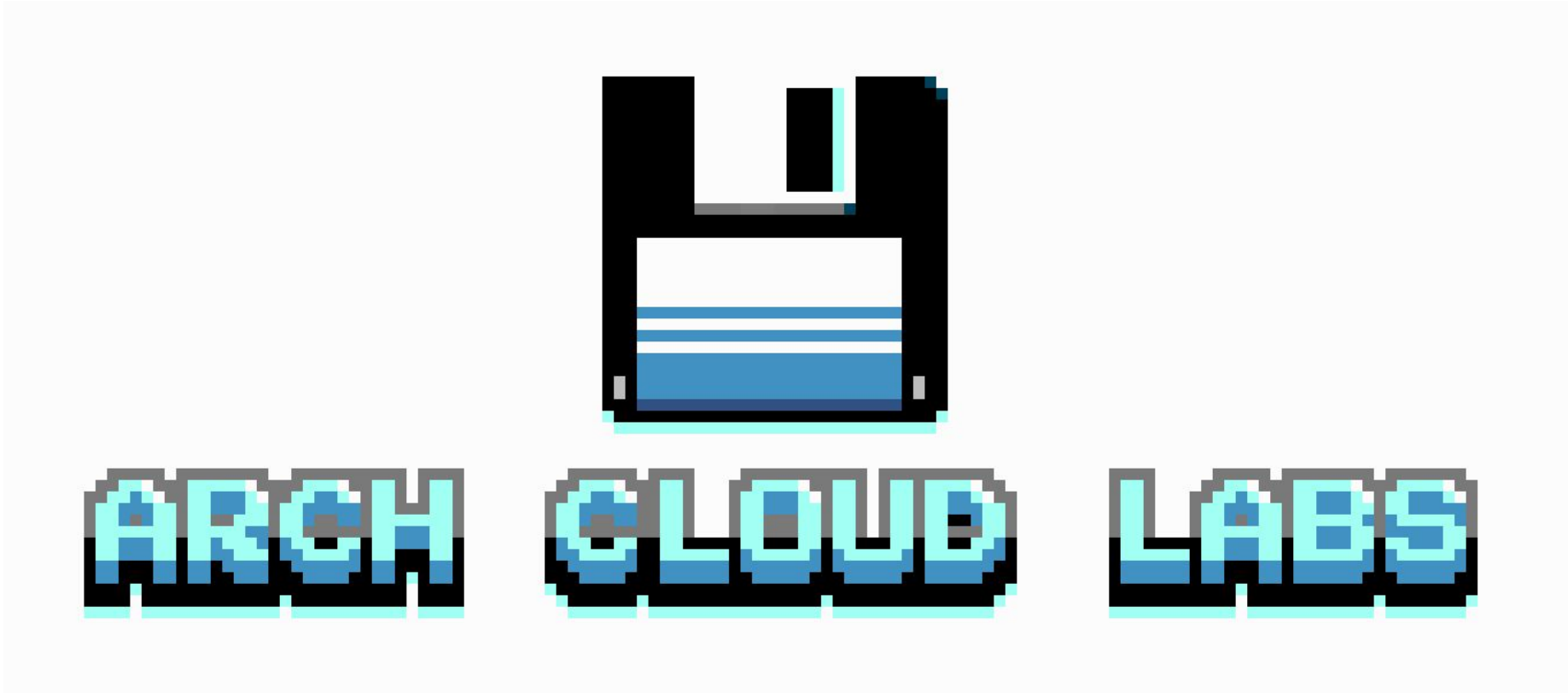


# Beyond The Class – Automating Ingestion



- We've been creating JSON/CSV files and uploading them manually via Kibana.
- What if we just ingested them directly into Elastic with our tool?
- Investigate how to leverage the Python API for Elastic to ingest data directly from our parsing utility into Elasticsearch.





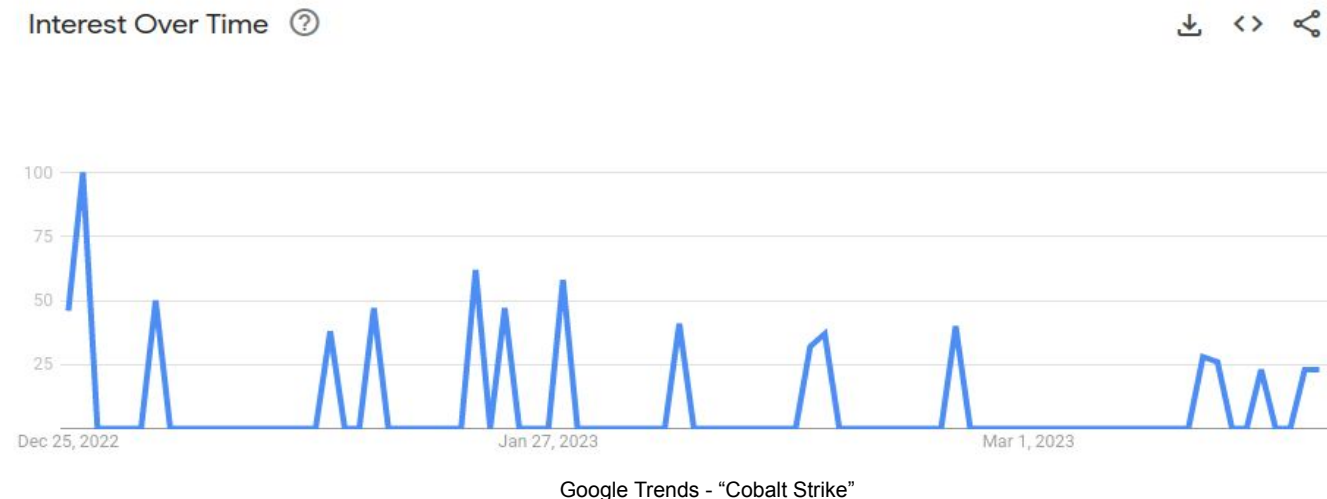
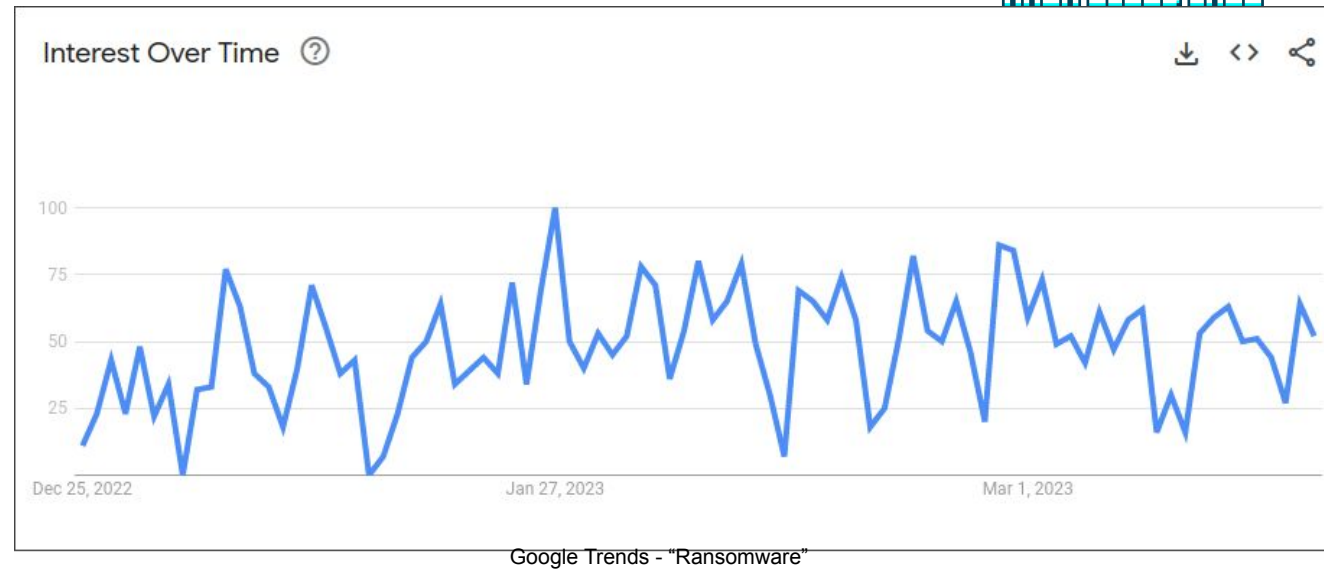
Break !



# Identifying Interesting Samples to Reverse



- Statically analyzing any single sample takes time.
- When choosing what to analyze consider what you're looking to get out of it:
  - Just to have fun
  - More knowledge about a given TTP
  - Getting better at a tool
  - Demonstrating to an employer a specific skill set



# Standing on The Shoulders of Giants: Public Feeds





# Why integrate with public feeds?



- These services are widely used across industry.
- This is a data enrichment/software development activity that can help you analyze Malware analysis trends.
  - Malware Analysis/Threat Intel++
    - See the malware trends as reported by other organizations.
  - Software Development++
    - How do we build services to go forth and fetch this data?
  - DevOps++
    - How do we automate, update and deploy these services?



# Hybrid Analysis Public Freed













































- <https://www.hybrid-analysis.com/feed?json>
- Public feed of JSON results from sandbox execution
- Data includes substantial artifacts from execution:
  - Registry keys
  - “maliciousness score”
  - IP addresses/Domains
  - File hashes
  - Process spawned
  - Files extracted
  - File size

```
20:
  md5: "a9f1d139268416c799550a0741aeaddb"
  sha1: "a2c3f3ff9443023114b94add4f7a30c165edd50c"
  sha256: "0cad2cfc06b38439809d62e1...42dbdb4eaa32e68c2914b55"
  tags: [...]
  isinteresting: false
  analysis_start_time: "2023-03-26 18:49:58"
  threat_score: 100
  threat_level: 2
  threat_level_human: "malicious"
  avdetect: 55
  isunknown: false
  vxfamily: "Trojan.Generic"
  submitname: "0cad2cfc06b38439809d62e1...db4eaa32e68c2914b55.exe"
  isurlanalysis: false
  size: 702464
  type: "PE32 executable (GUI) Intel 80386, for MS Windows"
  hosts: [...]
  hosts_geo: [...]
  compromised_hosts: [...]
  environmentId: "160"
  environmentDescription: "Windows 10 64 bit"
```



# Joe Sandbox – Community Edition



Result	Threat	Antivirus	Icon	Time & Date	Name	Info	Class	Graph	Actions
 <b>MALICIOUS</b> <small>YARA</small>	Amadey, RedLine	69%		2023-03-26 20:51:07 +02:00	NDQi2BkSYE.exe				
 <b>MALICIOUS</b> <small>YARA</small>	Clipboard Hijac...	59%		2023-03-26 20:35:10 +02:00	YTap6Znu4S.exe				
 <b>MALICIOUS</b> <small>YARA SHORT</small>	RedLine	62%		2023-03-26 20:34:09 +02:00	tizYKuC9WG.exe				
 <b>MALICIOUS</b> <small>YARA SHORT</small>	Amadey, Djvu, ...	88%		2023-03-26 20:21:08 +02:00	YyBbnfRqcG.exe				
 <b>MALICIOUS</b> <small>YARA</small>	AgentTesla, zg...	38%		2023-03-26 20:20:18 +02:00	wJ3c87wdKO.exe				
 <b>MALICIOUS</b> <small>YARA</small>	AgentTesla	38%		2023-03-26 20:20:16 +02:00	S9nn90N73S.exe				
 <b>MALICIOUS</b> <small>YARA SHORT</small>	Snake Keylogger	38%		2023-03-26 20:20:16 +02:00	fRxEBH9JSp.exe				



# VirusTotal – Graphs



Latest Graphs

Top commented Graphs

Top viewed Graphs



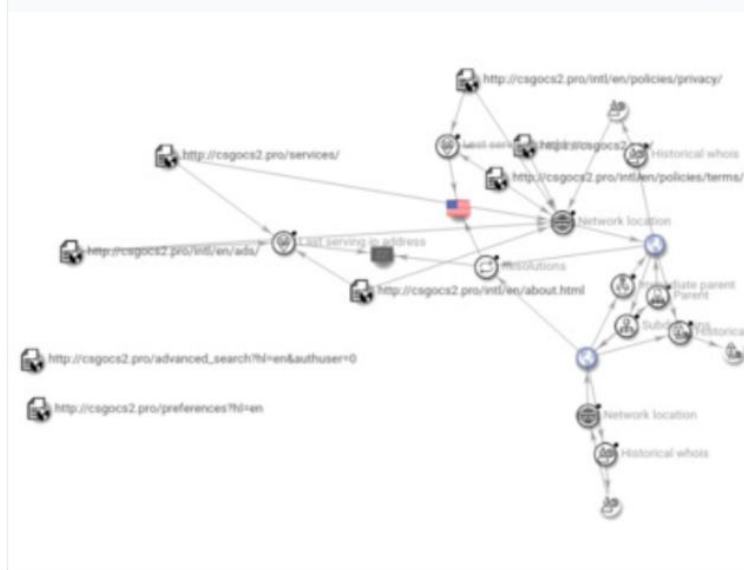
bot-zero-proxy-lion-project-umbrella///a104-...  
2023-03-26 18:01:59



Untitled graph6  
2023-03-26 15:23:20



CS2 QR Scam  
2023-03-26 15:05:00



q.au  
2023-03-26 14:31:14



q.org  
2023-03-26 14:20:24



https://forum.pasja-informatyki.pl/sitemap.xml  
2023-03-26 14:07:41



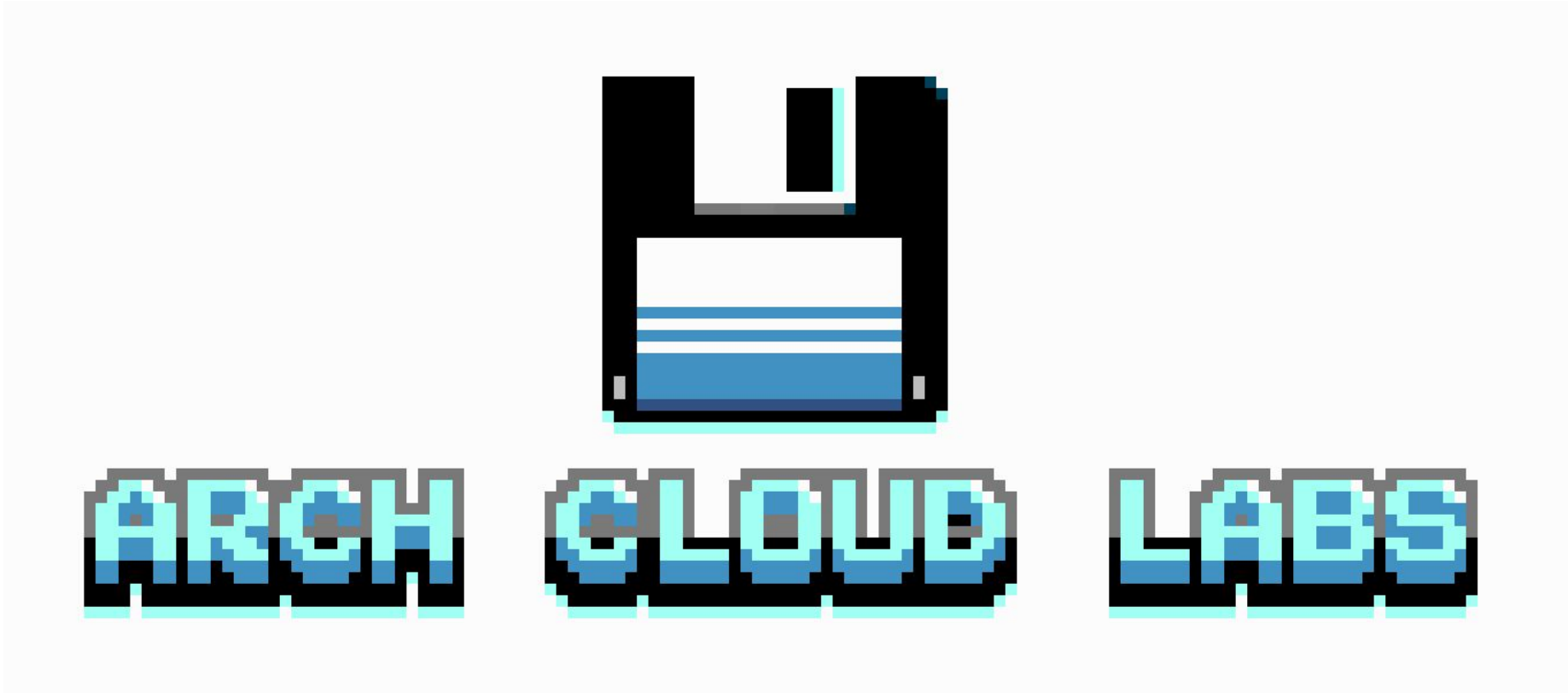
# VirusTotal – Graphs



- Community edition accounts get to use Graph.
  - These can be created for free.
  - API access to further enrich data.
- Browse other's graphs for inspiration
- However, your graphs are made **public**

<https://support.virustotal.com/hc/en-us/articles/360002138677-Does-VT-Graph-consume-quota-How-is-it-measured->





Break !

# Lab – Obtaining Data from Public Feeds

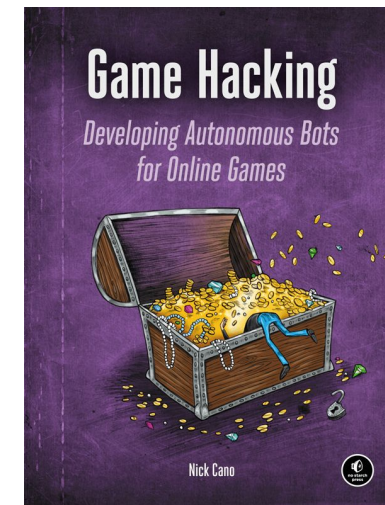
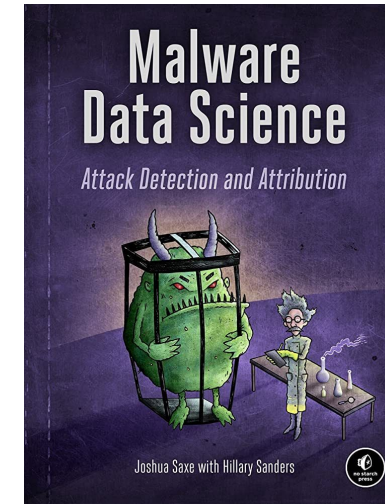




# Would you like to know more?



- Free workshops/classes
  - <https://malwareunicorn.org/#/>
  - <https://p.ost2.fyi/courses>
- Looking for network forensic challenges?
  - <https://www.malware-traffic-analysis.net/>
    - LIVE malware traffic, be careful!
  - <https://www.netresec.com/?page=PcapFiles>
    - ISTS /CCDC Competition



Finished!  
Survey: [tinyurl.com/43htcbst](https://tinyurl.com/43htcbst)  
**Thank You**



...  
**ARCH CLOUD LABS**



[archcloudlabs@gmail.com](mailto:archcloudlabs@gmail.com)



[www.archcloudlabs.com](http://www.archcloudlabs.com)

# Bonus – Building a “Pew Pew” Map



- What is IPInfo?
  - API to give Geolocation based on IP Address
  - `$> curl ipinfo.io/<IPv4_HERE>?token=<TOKEN_HERE>`
  - <https://ipinfo.io/products/free-ip-database>
- What is Cobalt Strike?
  - Prevalent adversary emulation tool.
    - Very configurable via MalleableC2
    - Beacon Object Files
  - Widely used by threat actors and red teams alike.

