Forensics Crash Course

MAYhem 2021-05-15T140000

TLP:WHITE

What’s cooking today?

- Short introduction to DFIR
- Windows artefacts
- Hands on labs
- CTF
- Maybe some memes…

TLP:WHITE

Disclaimer

The views and opinions expressed in the presentation are those of the authors and do not necessarily reflect the official policy or position of Finnish Transport and Communications Agency Traficom.

This course material includes malware and other stuff that may harm your and others’ computers. Repeat tasks with your own responsibility.

TLP:WHITE

Goal: Get more people interested in forensics and get them going on with technical stuff. I am trying to give you just the basics.

TLP:WHITE

forensic store

TLP:WHITE

The forensicstore project can create, access and process forensic artifacts bundled in so called forensicstores (a database for forensic artifacts).

TLP:WHITE

Relatively a new thing

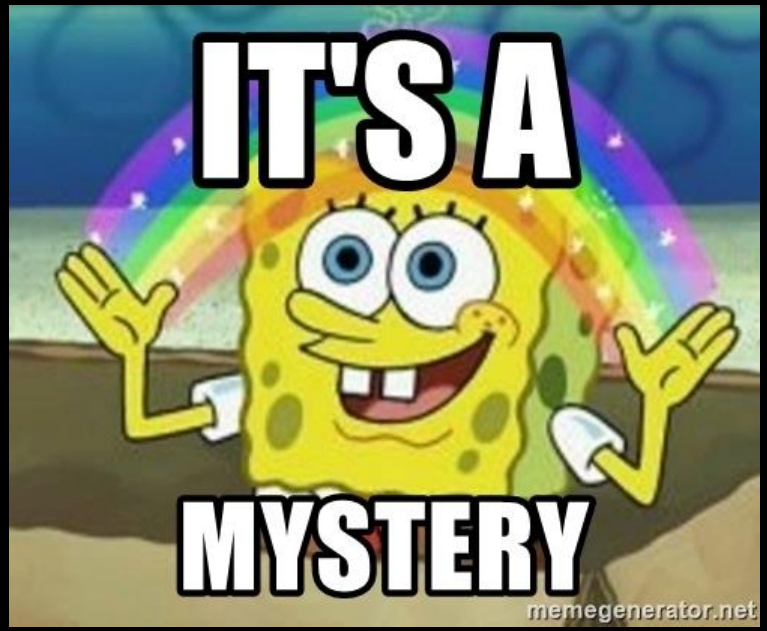
- Collect data to one bundle using
  - https://github.com/forensicanalysis/artifactcollector
- Investigate and extract data
  - https://github.com/forensicanalysis/elementary
  - $ wget https://github.com/forensicanalysis/elementary/releases/download/v0.3.0-rc.13/elementary_0.3.0-rc.13_Linux_amd64.deb
    $ sudo apt-get install -f ./elementary_0.3.0-rc.13_Linux_amd64.deb
  - $ elementary archive unpack store.forensicstore

TLP:WHITE

TLP:WHITE

DFIR

TLP:WHITE



TLP:WHITE

"There are only two types of companies:
those that have been hacked,
and those that will be."

Robert Mueller
FBI Director, 2001-2013

TLP:WHITE

Image Source: http://azterlan.blogspot.com/2017/07/cyber-security-forgotten-item-in_18.html

TLP:WHITE

Image Source: https://twitter.com/wef/status/420078886244814848/photo/1

The anatomy of a cyber attack

Pre-exploitation

Reconnaissance

Scanning
Information gathering
Phishing for information

Exploitation

Intrusion

Malspam
Phishing
Brute-forcing
Supply chain compromise
Trusted relationship
Vulnerabilities
Drive-by compromise
Privilege escalation
Persistence

Post-exploitation

Lateral movement

Exploitation of remote services
Internal spearphishing

Command & control

Maintain access
Fallback channels

Preparations

Gather and prepare information
Enumerate backup systems

Actions on objective

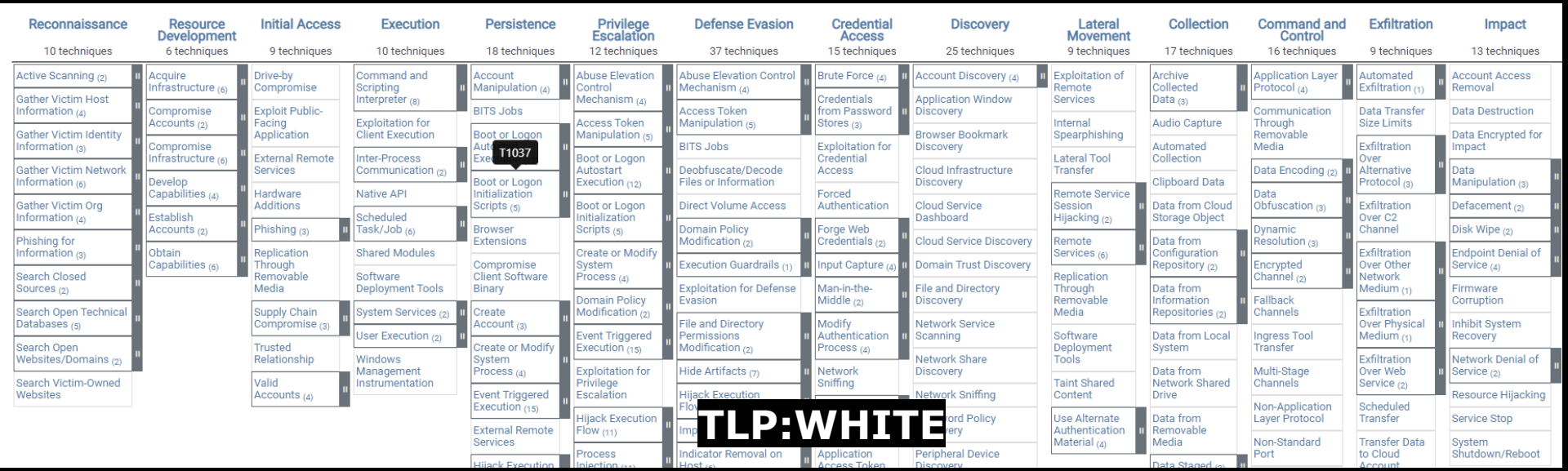
Stealing the crown jewels
Denial-of-Service
Ransomware
FUD

Based on https://attack.mitre.org/

TLP:WHITE

MITRE ATT&CK®

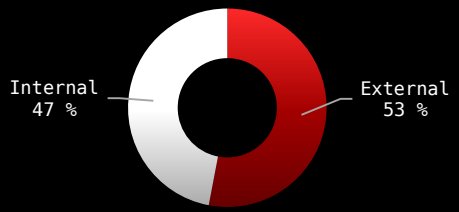
- Adversary tactics and techniques classification
- Understanding the taxonomy helps to understand the ongoing attack
  - Red team: What should we do next? How we report this for our customer?
  - Blue team: How should I categorize this attack? What is the attacker trying to do?



Detecting an intrusion is hard for organizations

- Organizations usually fail to detect intrusions
  - In 2019, over 50% of intrusions were detected by external parties
- The dwell time is long, the time is on attacker’s side
  - Dwell time = number of days between initial compromise and detection

Intrusion detection



Dwell time

TLP:WHITE

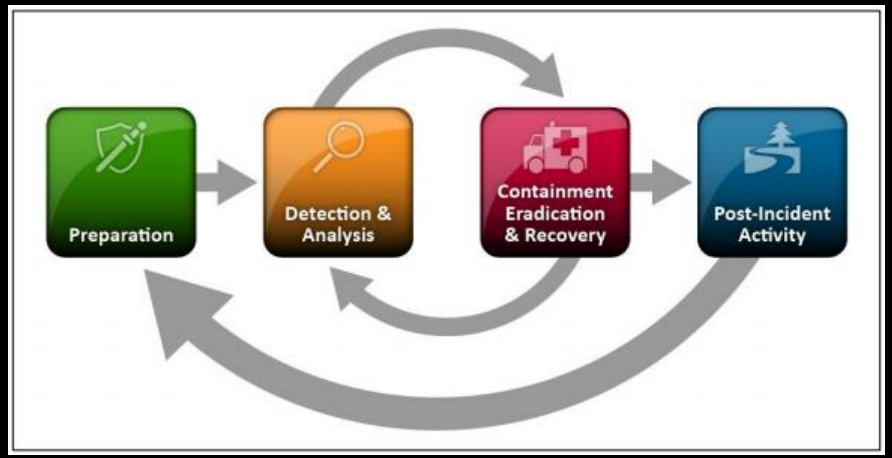
Source: https://content.fireeye.com/m-trends/rpt-m-trends-2020

What is Incident Response (IR) then?

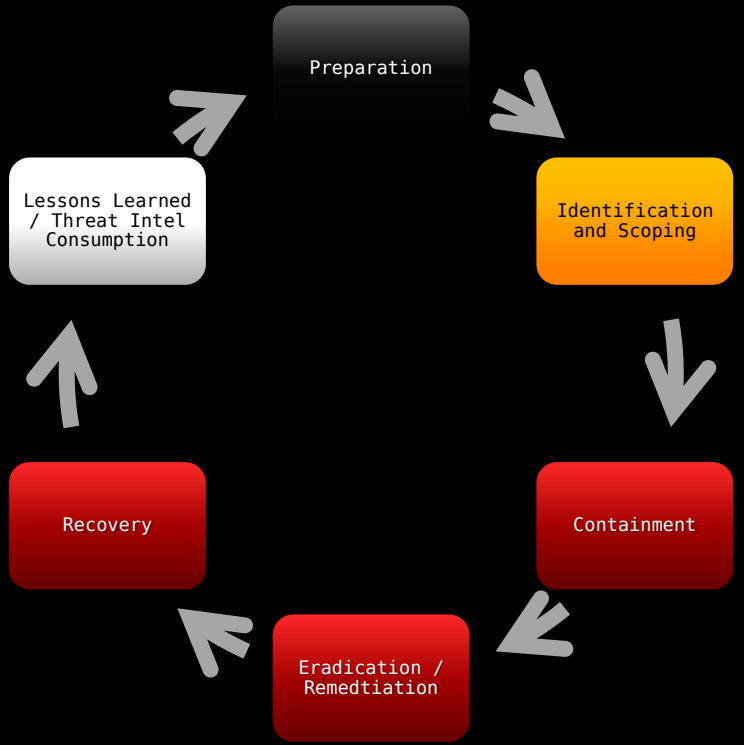
- NIST SP800-61r2:
  - Rapidly detect and verify incidents
  - Minimize the loss and the destruction
  - Mitigate the weakness that let the intrustion to happen
  - Restore IT services
- Make sure the incident is handled properly and ensure business continuity

TLP:WHITE

IR process

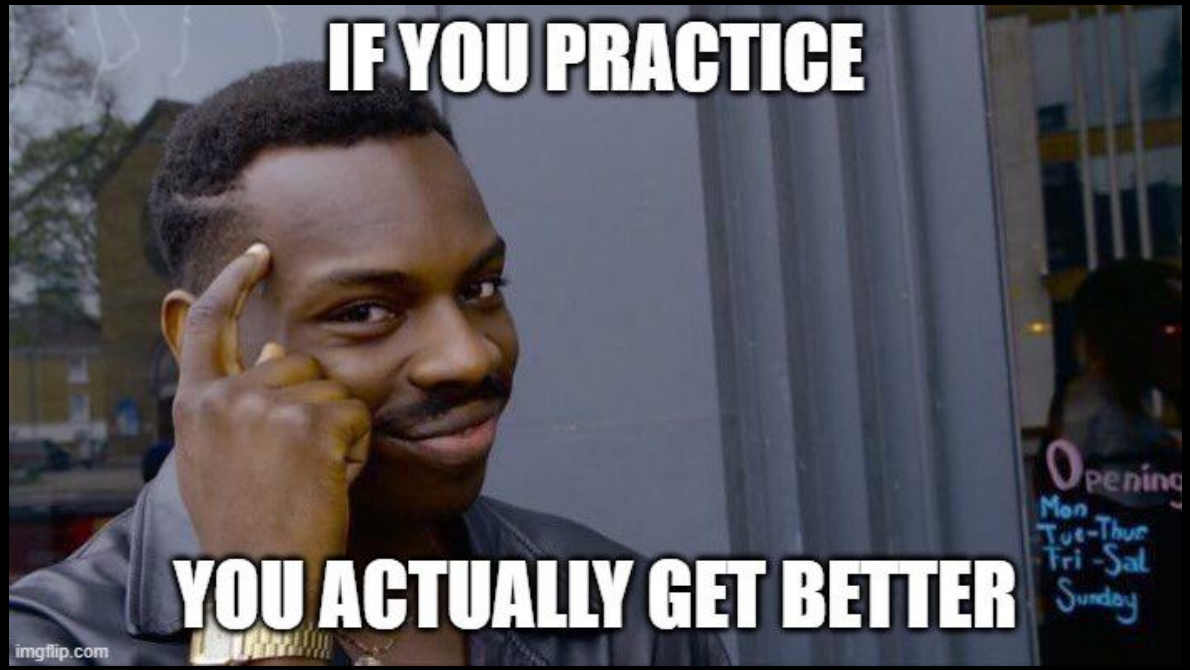


Source: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf

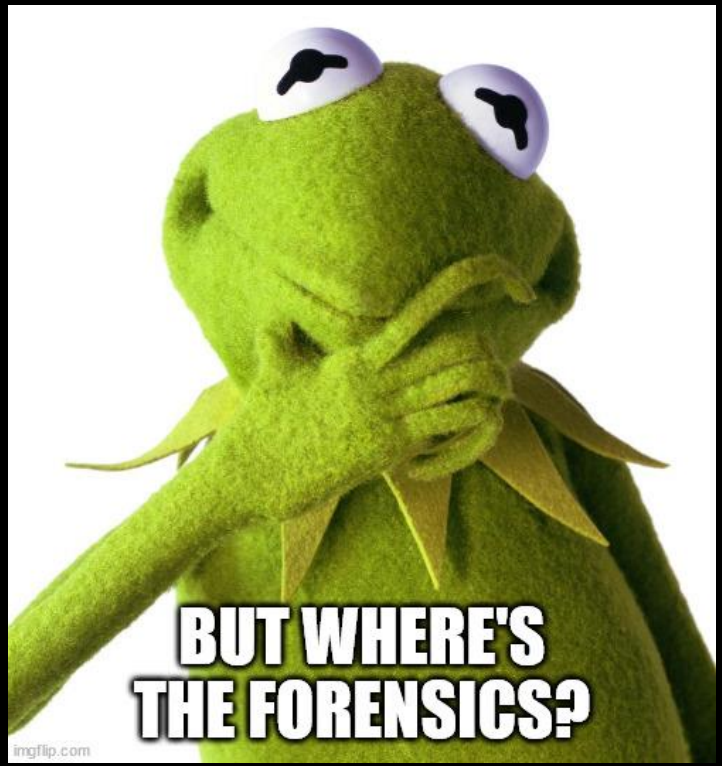


Source: SANS FOR508

TLP:WHITE

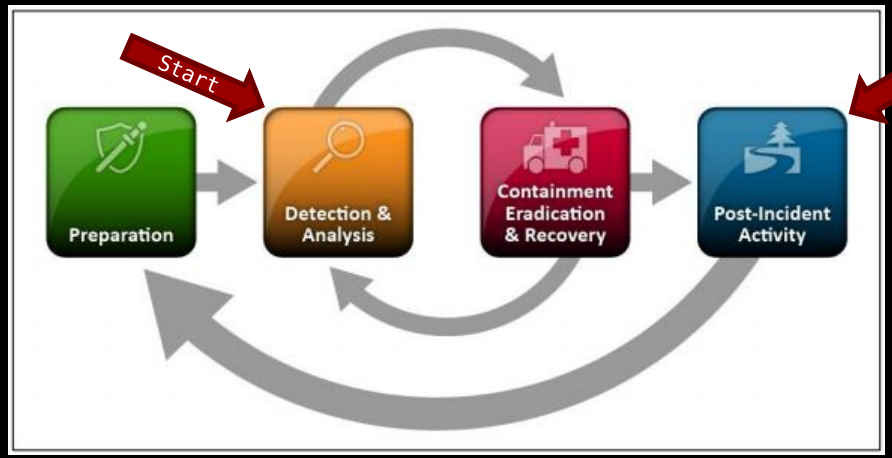

IF YOU PRACTICE
YOU ACTUALLY GET BETTER
imgflip.com
Mon
Tue-Thur
Fri -Sat
Sunday

TLP:WHITE

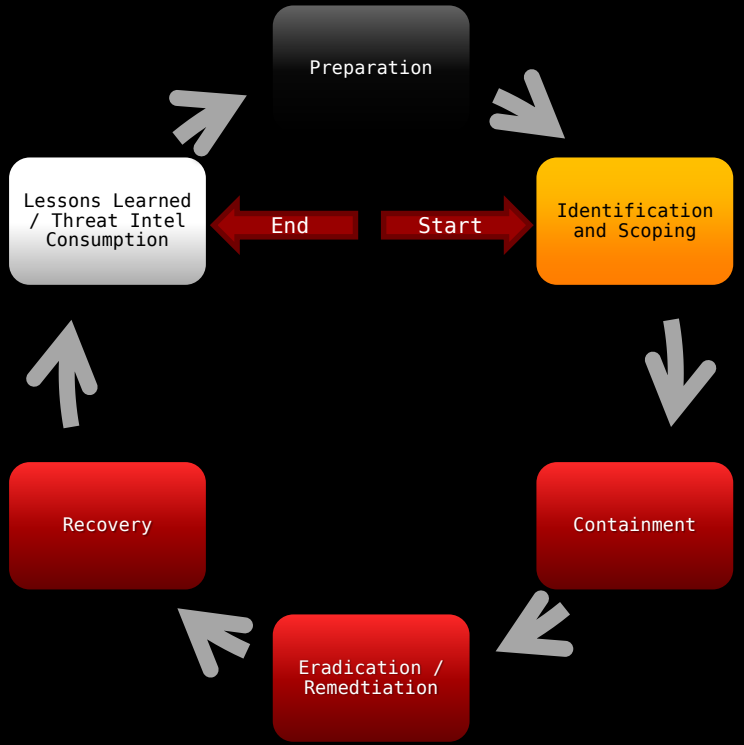


TLP:WHITE

IR process



Source: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf



Source: SANS FOR508

TLP:WHITE

Image source: https://cdn2.hubspot.net/hubfs/3350762/Whac%20a%20Mole.png

TLP:WHITE

Tactics

TLP:WHITE

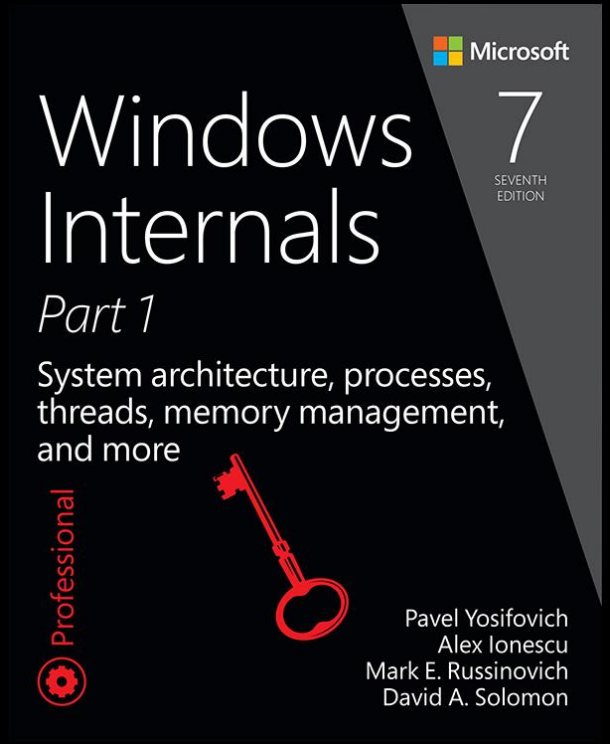
Digital Forensics workflow

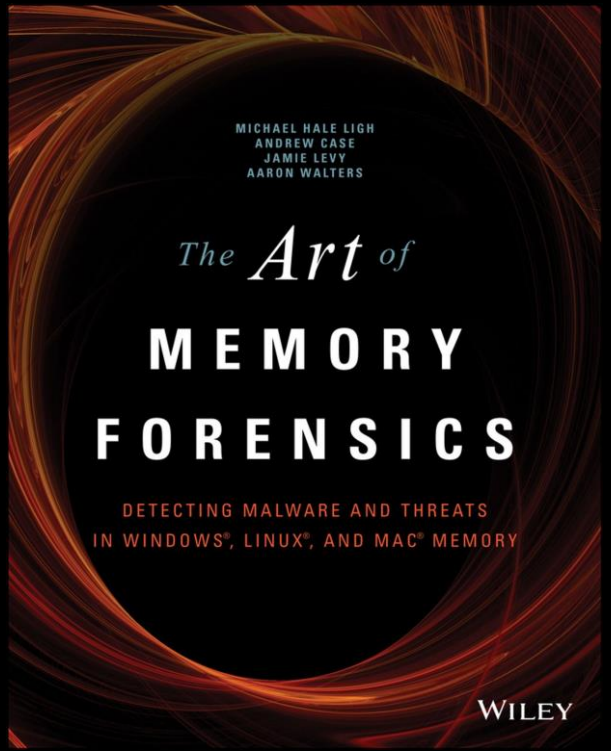
TLP:WHITE

Memory

TLP:WHITE

Book recommendations



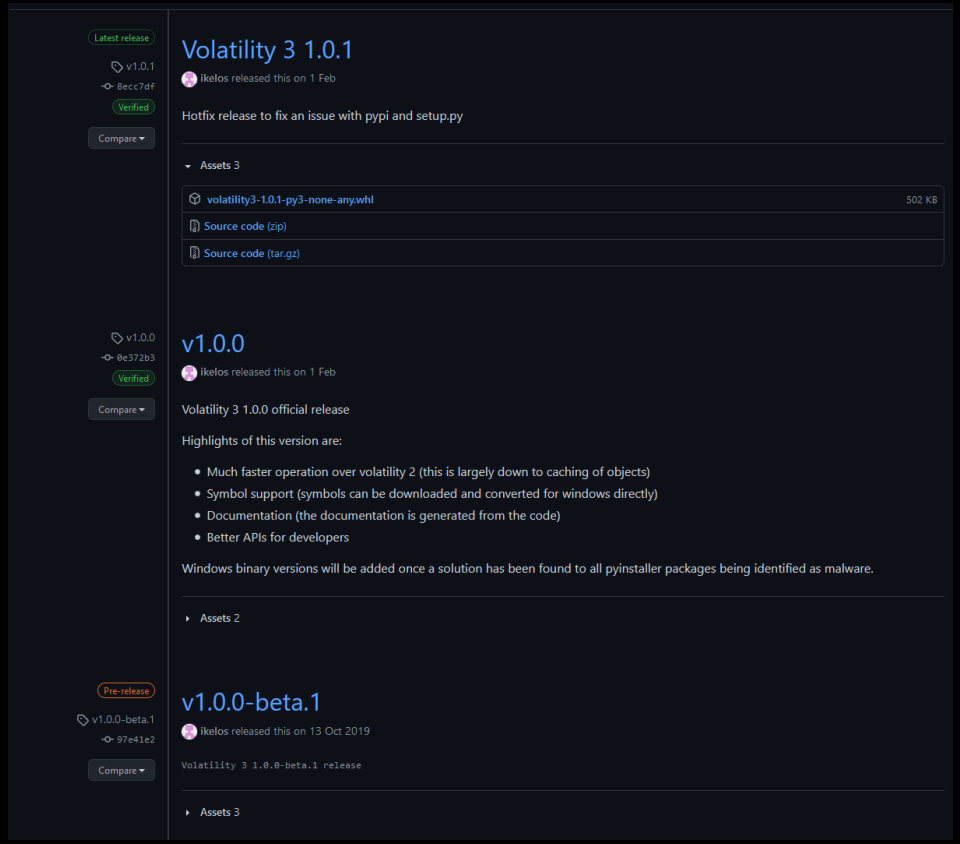


TLP:WHITE

TLP:WHITE

Memory forensics

- Super good and effective [and fun]
- However, the cases where you actually get usable memory dump are rare [from my opinion]
- Volatility is way to go…
  - https://www.volatilityfoundation.org
- …BUT MemProcFS will soon be better than Volatility → go, check and try it out already
  - https://github.com/ufrisk/MemProcFS


Latest release
v1.0.1
8ecc7df
Verified
Compare
Volatility 3 1.0.1
ikelos released this on 1 Feb
Hotfix release to fix an issue with pypi and setup.py
Assets 3
volatility3-1.0.1-py3-none-any.whl
502 KB
Source code (zip)
Source code (tar.gz)
v1.0.0
0e372b3
Verified
Compare
v1.0.0
ikelos released this on 1 Feb
Volatility 3 1.0.0 official release
Highlights of this version are:
Much faster operation over volatility 2 (this is largely down to caching of objects)
Symbol support (symbols can be downloaded and converted for windows directly)
Documentation (the documentation is generated from the code)
Better APIs for developers
Windows binary versions will be added once a solution has been found to all pyinstaller packages being identified as malware.
Assets 2
Pre-release
v1.0.0-beta.1
97e41e2
Compare
v1.0.0-beta.1
ikelos released this on 13 Oct 2019
Volatility 3 1.0.0-beta.1 release
Assets 3

TLP:WHITE

Updated README (#553)

Browse files

```
* Update README.md

* Update README.md
```

master (#553)

joachimmetz committed on 18 Oct 2020 Verified

1 parent f6dd536 commit 55d1925f2df9759a989b35271b4fa48fc54a1c86

Showing 1 changed file with 19 additions and 0 deletions.

Unified Split

19 ■■■■■ README.md

```
@@ -1,3 +1,22 @@
+ # Rekall discontinuation
+
+ This project is no longer maintained.
+
+ In December 2011, a new branch within the Volatility project was created to explore how to make the code base more modular, improve performance, and
 increase usability. This branch was later forked to become Rekall. The modularity allowed physical memory analysis functionality to be used in
 [GRR](https://github.com/google/grr) to enable remote live in-memory analysis.
+
```

TLP:WHITE

Updated README (#55

* Update README.md

* Update README.md

master (#553)

joachimmetz committed

59a989b35271b4fa48fc54a1c86

Browse files

Showing 1 changed file with

Unified Split

19 README.md

```
... ... @@ -1,3 +1,22 @@
1  + # Rekall dis
2  +
3  + This project
4  +
5  + In December
   increase usabi
   [GRR](https://
6  +
```

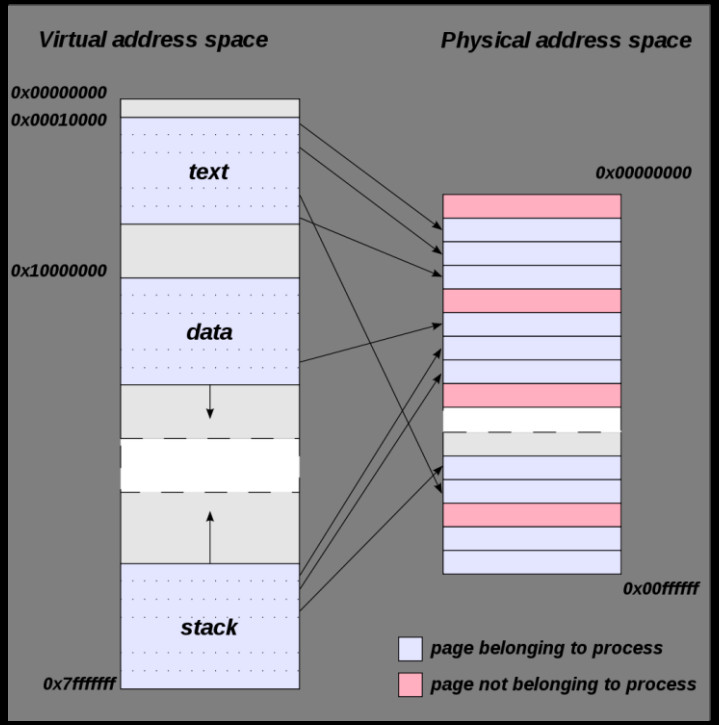
e performance, and
be used in

TLP:WHITE

Why u forensicate memory?

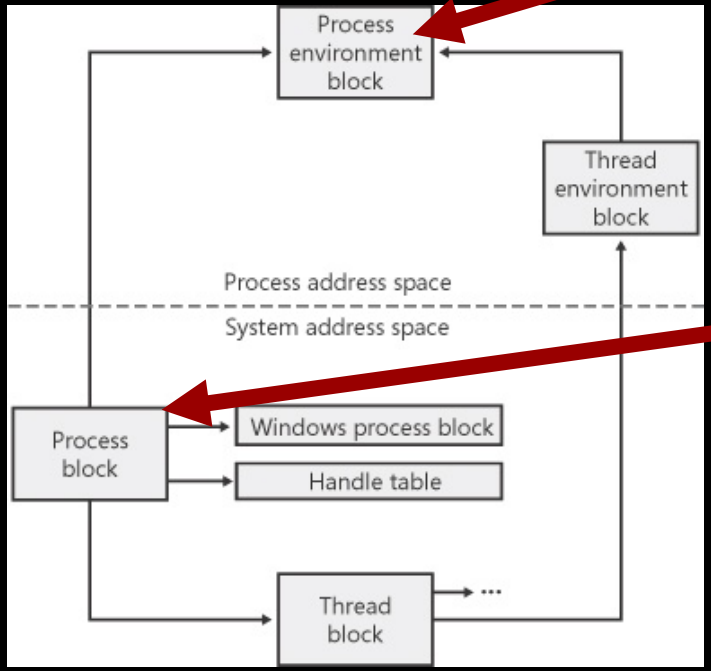
- What can you find from memory?
  - Windows registry and log files
  - Opened files
  - Secrets
  - Configuration files and data
  - All running processes (including malware)
  - Network artifacts

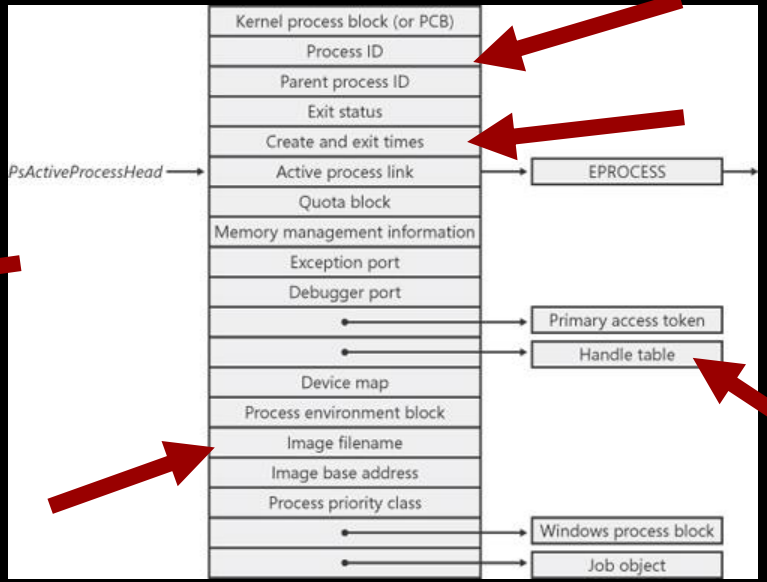
- Memory is the best place in analyzing malicious software activity
- Some evidence can’t be found elsewhere
  - “Fileless malware”
  - Malware can EITHER success in
    - Hiding
    - Executing



Source: Wikipedia, https://en.wikipedia.org/wiki/Virtual_address_space#/media/File:Virtual_address_space_and_physical_address_space_relationship

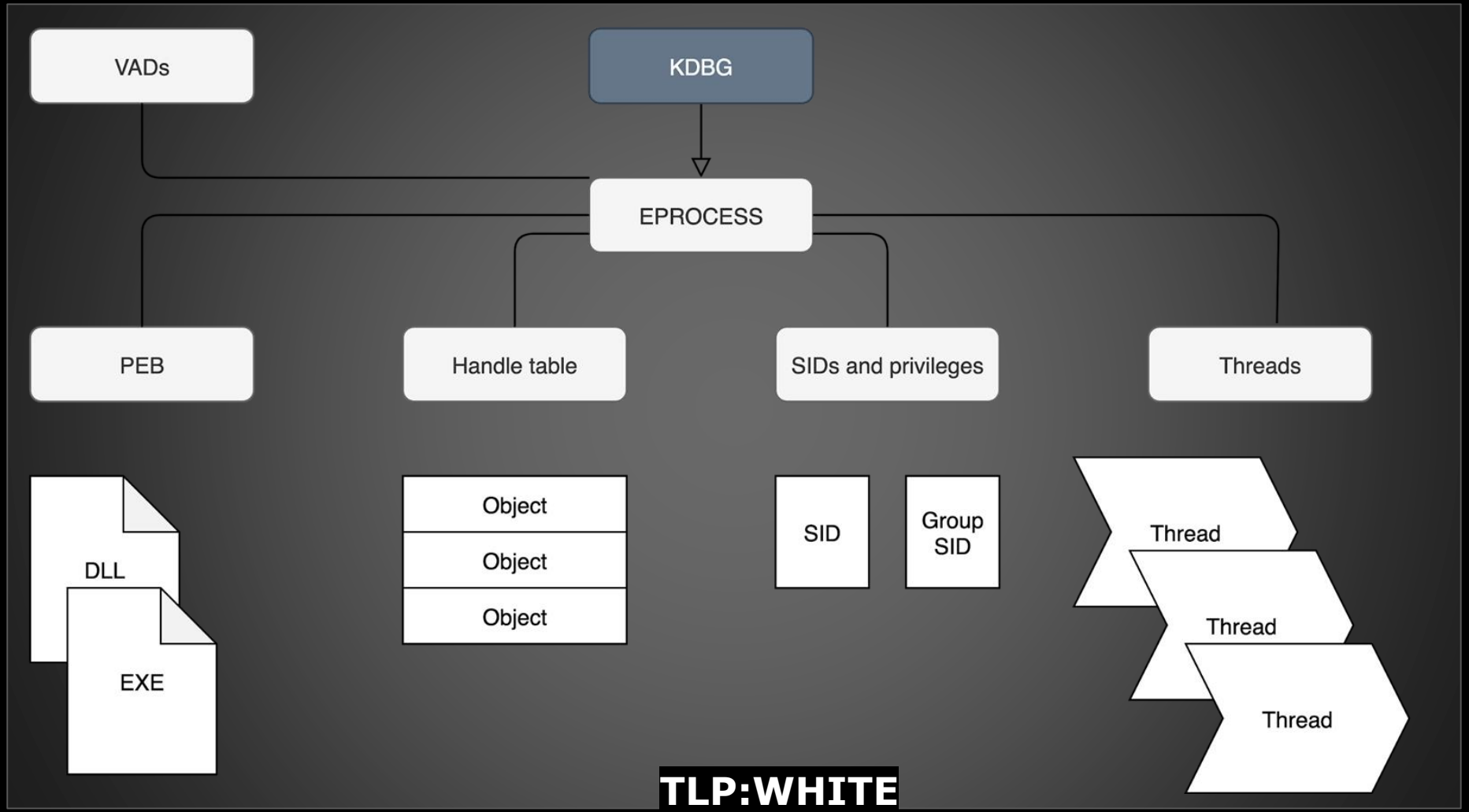
TLP:WHITE





Source: Mark E. Russinovich and David A. Solomon, Windows Internals, 5th Edition, https://www.microsoftpressstore.com/articles/article.aspx?p=2233328

TLP:WHITE


VADs
KDBG
EPROCESS
PEB
Handle table
SIDs and privileges
Threads
DLL
EXE
Object
Object
Object
SID
Group SID
Thread
Thread
Thread
TLP:WHITE

TTP examples

T1059.001
Powershell

T1548.002
Bypass UAC

T1543.003
Persistence
(Win Service)

TLP:WHITE

In-memory attacks (TTP exmp)

T1055
Shellcode
Injection

T1055
Reflective
DLL Injection

T1055.012
Process
hollowing

https://blog.f-secure.com/memory-injection-like-a-boss/

TLP:WHITE

Analysis overview (SANS)

1. Identify Rogue Processes
2. Analyze Process DLLs and Handles
3. Review Network Artifacts
4. Look for Evidence of Code Injection
5. Check for Signs of a Rootkit
6. Extract Processes, Drivers, and Objects

https://digital-forensics.sans.org/media/volatility-memory-forensics-cheat-sheet.pdf

TLP:WHITE

Volatility cheat sheet

Basics

```
$ vol.py –f [image] --
profile=[PROFILE] [plugin]

$ vol.py --help
$ vol.py --info
$ vol.py [plugin] –help
$ vol.py imageinfo
$ vol.py kdbgscan
```

Analyze Process DLLs and Handles

```
Lists all loaded DLL-files per process
$ vol.py dlllist

Shows SIDs (Security Identifier) for
each process
$ vol.py getsids
```

Check for Signs of a Rootkit

```
$ vol.py psxview
$ vol.py modscan
$ vol.py apihooks
$ vol.py ssdt
$ vol.py driverirp
$ vol.py idt
```

Identify Rogue Processes

```
Process list:
$ vol.py pslist

Scan the memory for EPROCESS blocks:
$ vol.py psscan

Process tree:
$ vol.py pstree
(-v for verbose)
```

Review Network Artifacts

```
Connections and sockets (Vista
onwards)
$ vol.py netscan
```

Look for Evidence of Code Injection

```
$ vol.py malfind
$ vol.py hollowfind
```

Extract Processes, Drivers, and Objects

```
Dump the process image
$ vol.py procdump

Dumps all registry files (NTUSER.DAT,
SYSTEM, SAM etc.)
$ vol.py dumpregistry

$ vol.py filescan
$ vol.py dumpfiles
```

TLP:WHITE

strings & grep

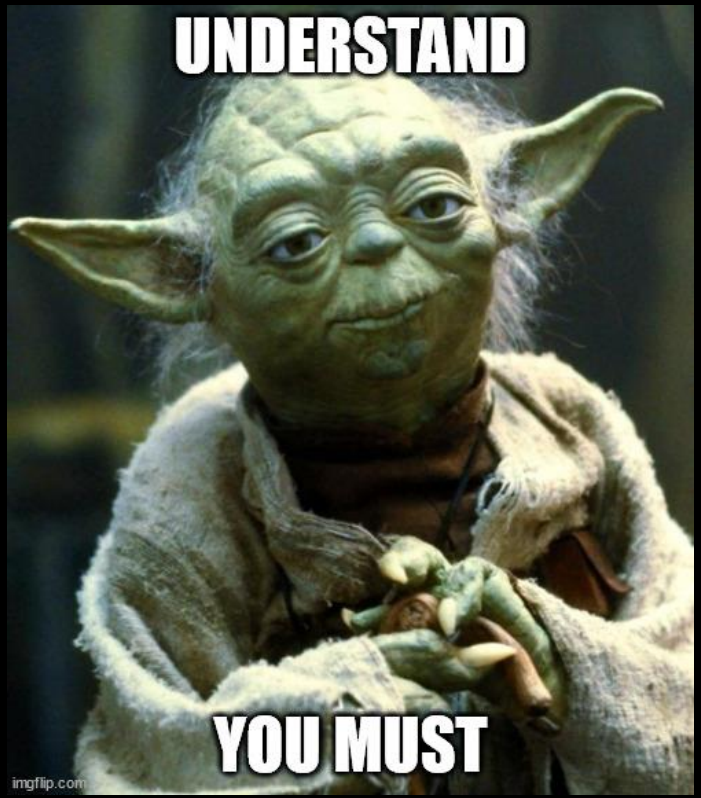
Sooooo good together (+ awk, sed, cut) <3

ASCII strings
```
$ strings -a memory.dmp > strings_a.txt
```

Unicode strings
```
$ strings -e l memory.dmp > strings_u.txt
```

Grep all the things
```
$ grep -ihra keyword
$ grep -Ero '(http|https)://[^/"]+'
```

TLP:WHITE

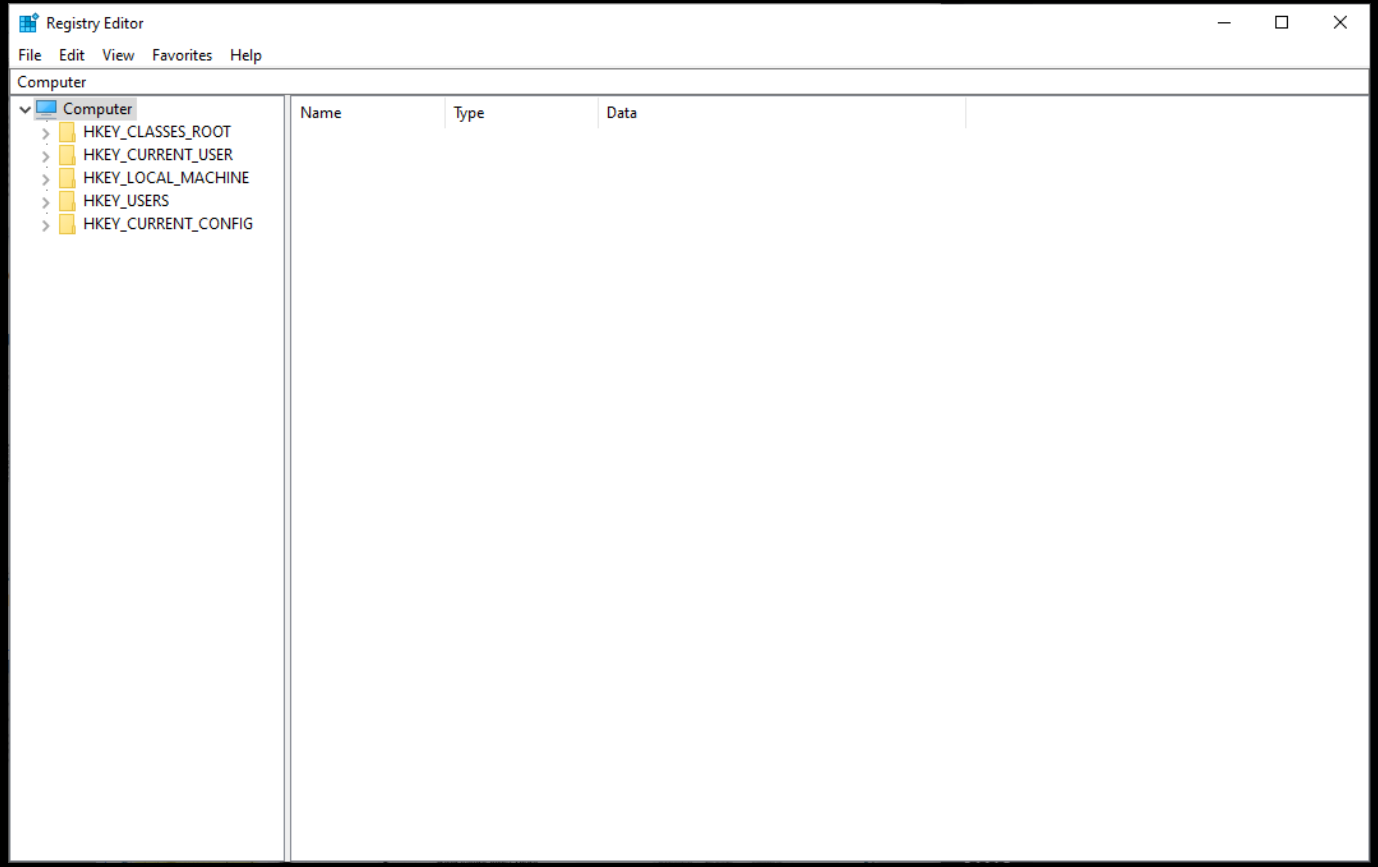


TLP:WHITE

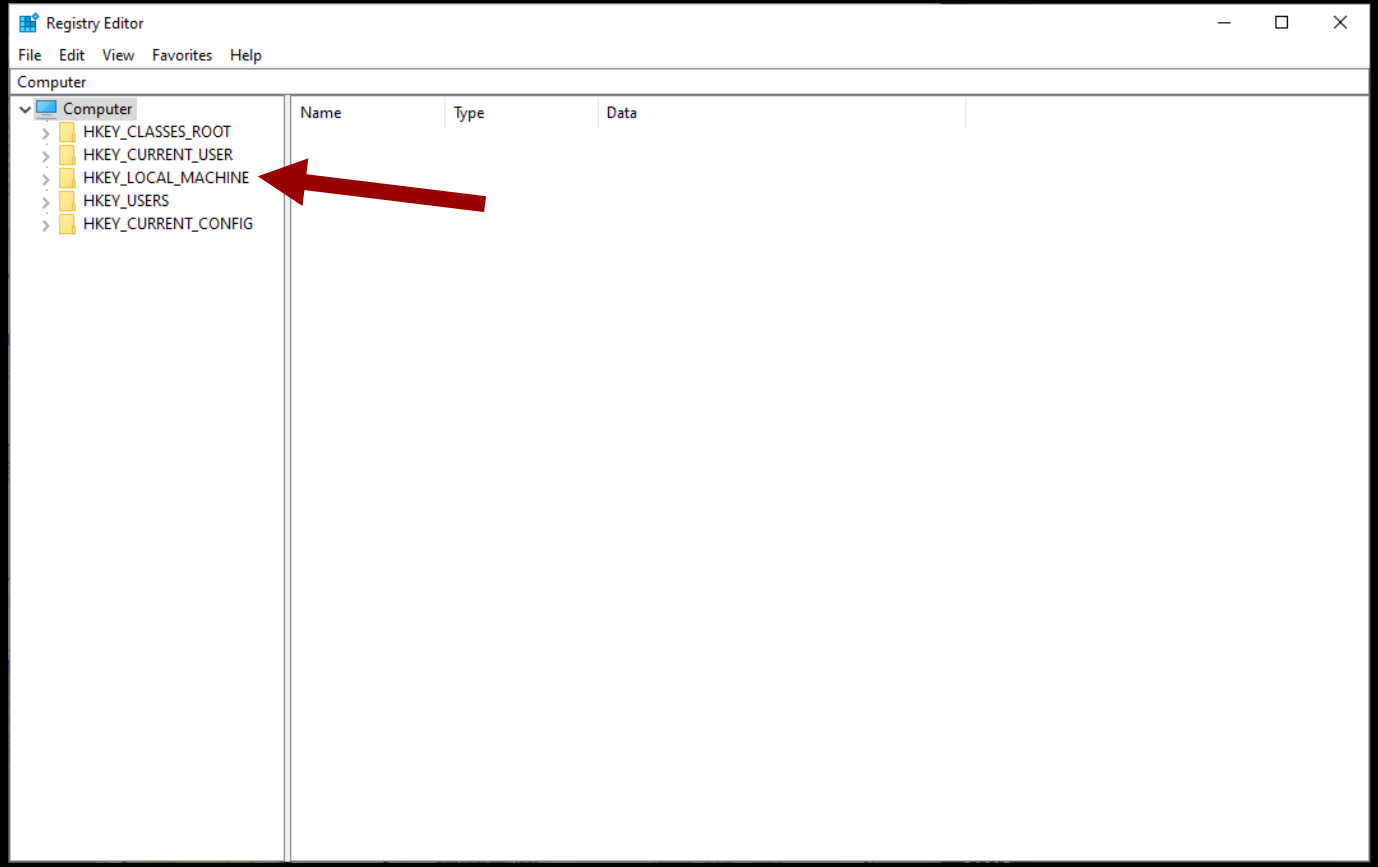
TLP:WHITE

Registry

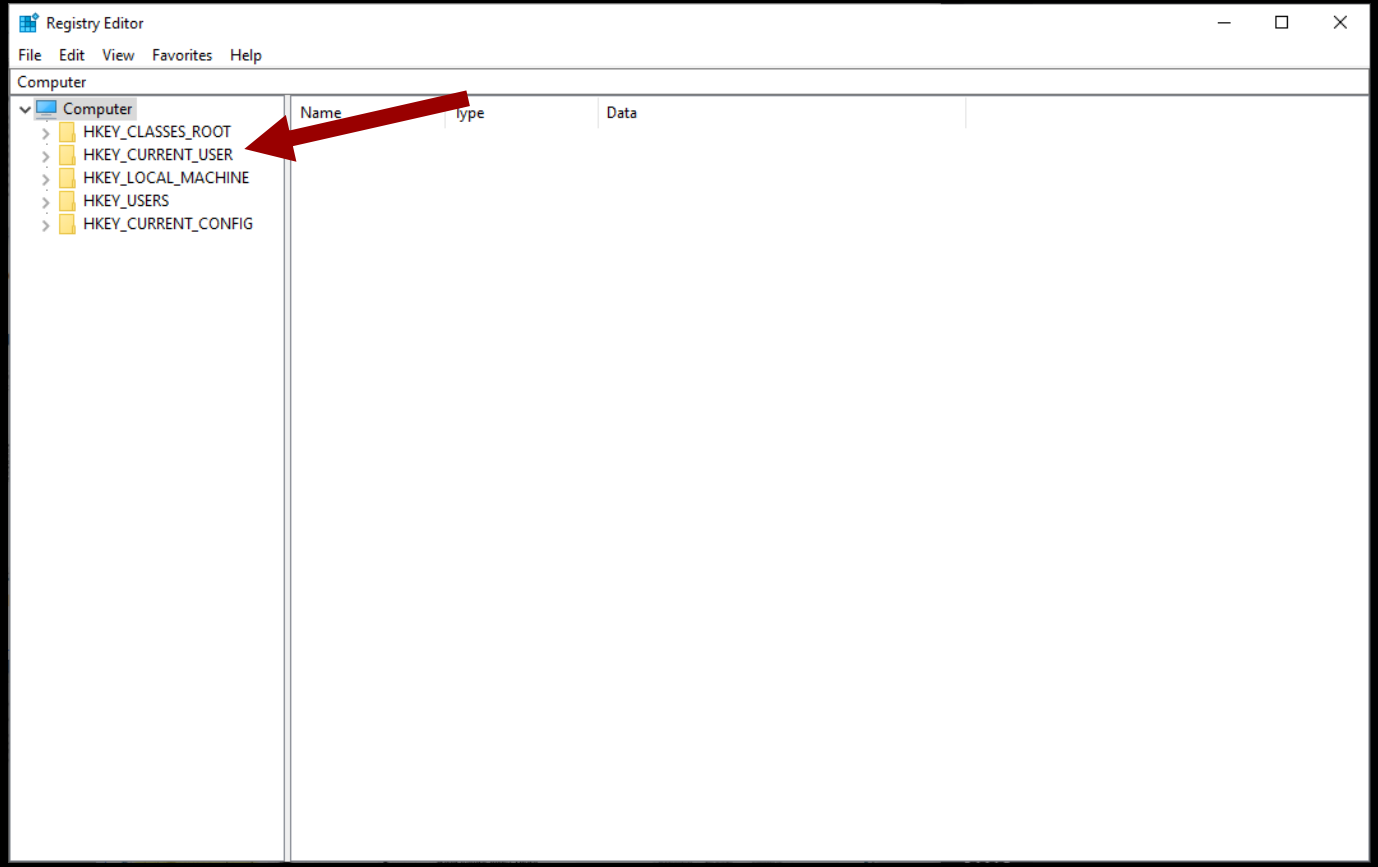
TLP:WHITE


Registry Editor
File Edit View Favorites Help
Computer
Computer
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
HKEY_USERS
HKEY_CURRENT_CONFIG
Name
Type
Data

TLP:WHITE


Registry Editor
File Edit View Favorites Help
Computer
Computer
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
HKEY_USERS
HKEY_CURRENT_CONFIG
Name
Type
Data

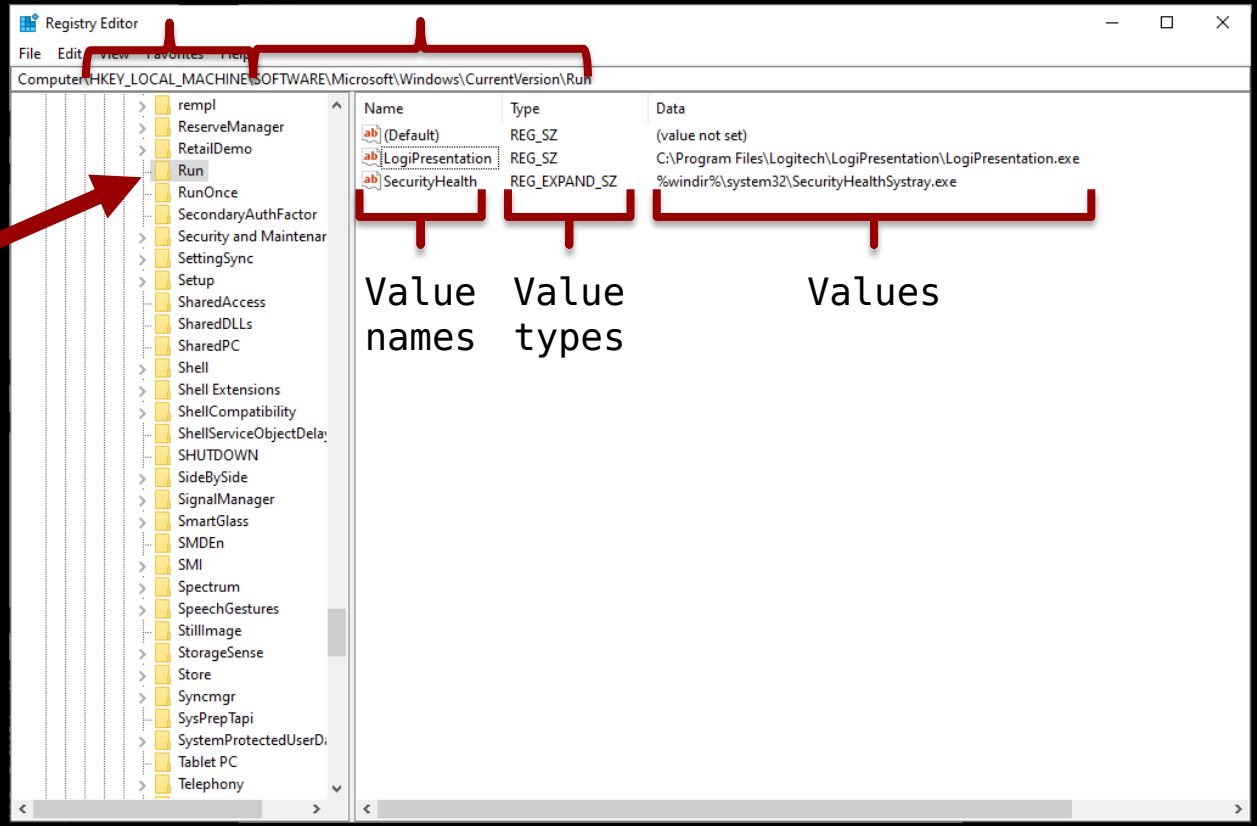
TLP:WHITE


Registry Editor
File Edit View Favorites Help
Computer
Computer
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
HKEY_USERS
HKEY_CURRENT_CONFIG
Name
Type
Data

TLP:WHITE

Hive

Keys/subkeys



"RunKey"

Value names

Value types

Values

TLP:WHITE

Where do I find these?

- %SystemRoot%\System32\Config\
  - Sam – HKEY_LOCAL_MACHINE\SAM
  - Security – HKEY_LOCAL_MACHINE\SECURITY
  - Software – HKEY_LOCAL_MACHINE\SOFTWARE
  - System – HKEY_LOCAL_MACHINE\SYSTEM
- %USERPROFILE%\Ntuser.dat – HKEY_USERS\SID
- %USERPROFILE%\AppData\Local\Microsoft\Windows\Usrclass.dat – HKEY_USERS\<User SID>_Classes (KEY_CURRENT_USER\Software\Classes)

Transaction logs (.LOG)

- Multiple files in the same directory as the registry hive files
  - .LOG1, .LOG2
- LOG file is a journal where the registry changes are being written before writing to hive files
- You can easily rebuild the logs by using tool called regipy
  - `$ pip3 install regipy`
  - `$ registry-transaction-logs NTUSER.DAT -p ntuser.dat.log1 -s ntuser.dat.log2 -o recovered_NTUSER.dat`

TLP:WHITE

Available tools

- RegRipper (rip.pl)
  - https://github.com/keydet89/RegRipper3.0
  - Pretty nice command line tool for fetching values from registry. Has a GUI as well.
- Registry Explorer/RECmd
  - https://ericzimmerman.github.io/#!index.md
  - Pretty nice command line tool for fetching values from registry. Has a GUI as well.

RegRipper cheat sheet

Basics

```
$ rip.pl -r [hive] -p [plugin]

List all plugins
$ rip.pl -c -l

Show only plugins for NTUSER.DAT
$ rip.pl -c -l |sort -n|grep -i ntuser
```

System profiling

```
$ rip.pl -r SYSTEM -p compname
$ rip.pl -r SYSTEM -p nic2

$ rip.pl -r SOFTWARE -p winver
$ rip.pl -r SOFTWARE -p uninstall

$ rip.pl -r NTUSER.DAT -p uninstall
$ rip.pl -r NTUSER.DAT -p pslogging
```

Autoruns

```
$ rip.pl -r SOFTWARE -p run
$ rip.pl -r SOFTWARE -p runonceex
$ rip.pl -r NTUSER.DAT -p run
$ rip.pl -r NTUSER.DAT -p cmdproc
```

Fileless malware

```
$ rip.pl -r [hive] -p fileless
```

User profiling

```
$ rip.pl -r SOFTWARE -p profilelist

$ rip.pl -r NTUSER.DAT -p profiler
$ rip.pl -r NTUSER.DAT -p runmru
$ rip.pl -r NTUSER.DAT -p
wordwheelquery
$ rip.pl -r NTUSER.DAT -p recentdocs
$ rip.pl -r NTUSER.DAT -p userassist
```

TLP:WHITE

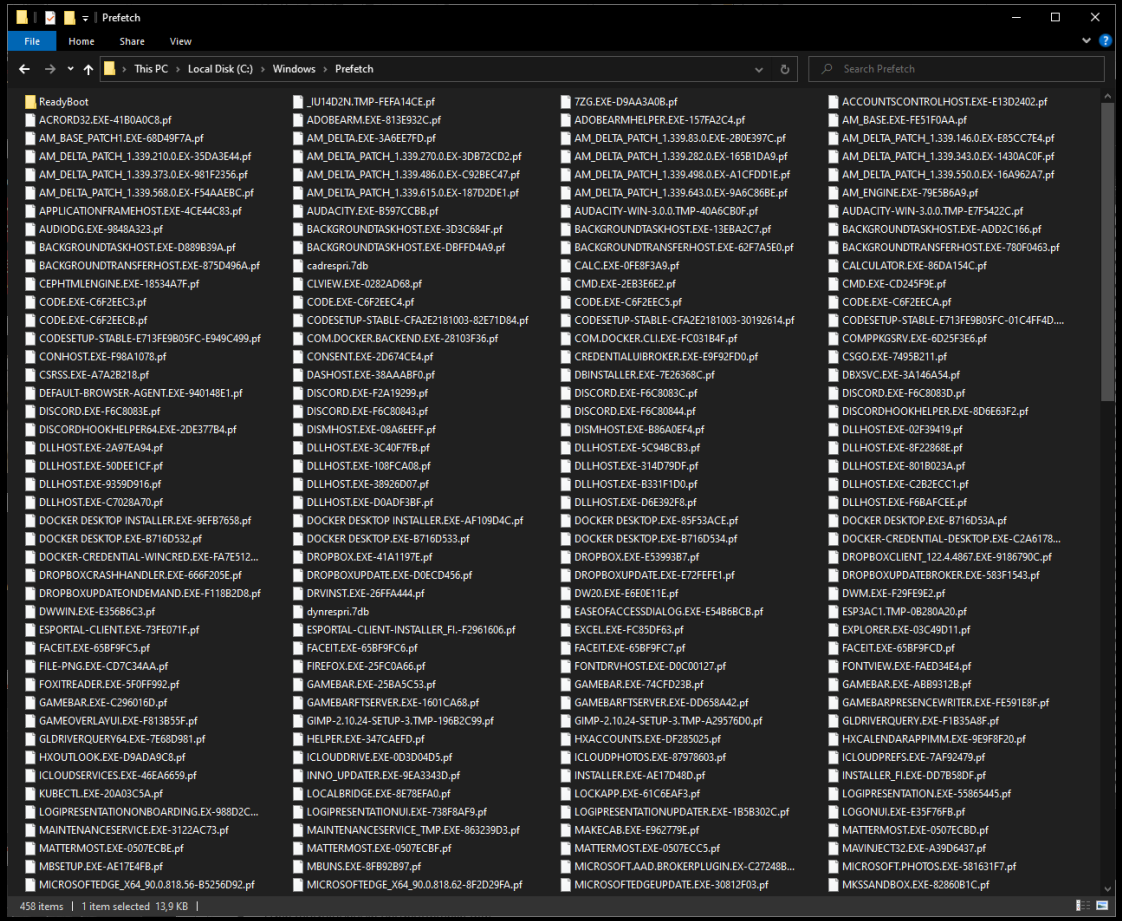
TLP:WHITE

Prefetch

TLP:WHITE

What?

- Windows component, purpose is to make applications load faster
- Basically a cache file
  - Stores everything the software has touched
  - F.E. Word has docx open when it’s shutdown, prefetch should see the docx file as a referenced file
- Can reveal execution timestamps and how many times the software has been executed
- %SystemRoot%\Prefetch

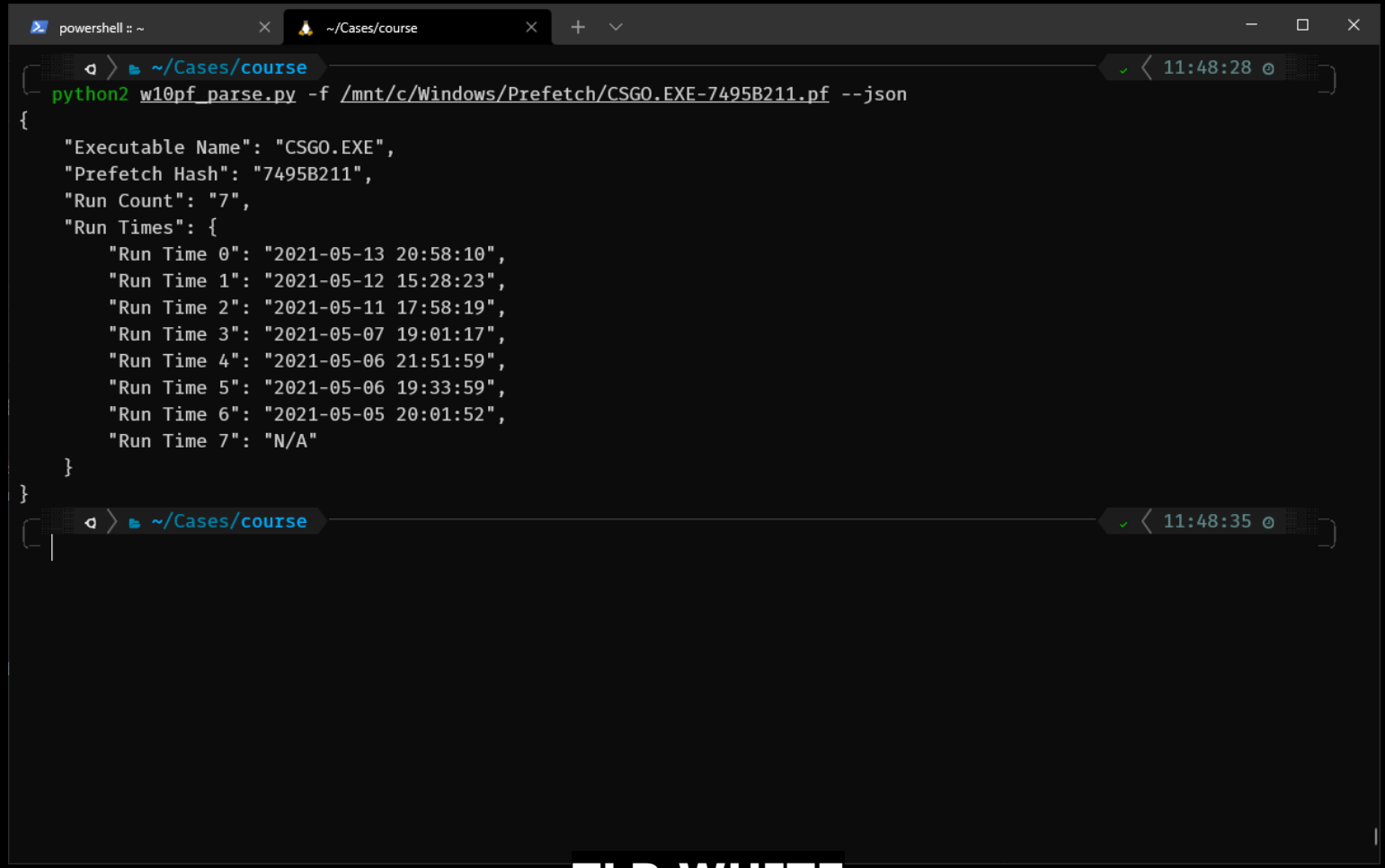

Prefetch
File
Home
Share
View
This PC
Local Disk (C:)
Windows
Prefetch
Search Prefetch
ReadyBoot
ACRORD32.EXE-41B0A0C8.pf
AM_BASE_PATCH1.EXE-68D49F7A.pf
AM_DELTA_PATCH_1.339.210.0.EX-35DA3E44.pf
AM_DELTA_PATCH_1.339.373.0.EX-981F2356.pf
AM_DELTA_PATCH_1.339.568.0.EX-F54AAEBC.pf
APPLICATIONFRAMEHOST.EXE-4CE44C83.pf
AUDIODG.EXE-9848A323.pf
BACKGROUNDTASKHOST.EXE-D889B39A.pf
BACKGROUNDTRANSFERHOST.EXE-875D496A.pf
CEPHTMLENGINE.EXE-18534A7F.pf
CODE.EXE-C6F2EEC3.pf
CODE.EXE-C6F2EECB.pf
CODESETUP-STABLE-E713FE9B05FC-E949C499.pf
CONHOST.EXE-F98A1078.pf
CSRSS.EXE-A7A2B218.pf
DEFAULT-BROWSER-AGENT.EXE-940148E1.pf
DISCORD.EXE-F6C8083E.pf
DISCORDHOOKHELPER64.EXE-2DE377B4.pf
DLLHOST.EXE-2A97EA94.pf
DLLHOST.EXE-50DEE1CF.pf
DLLHOST.EXE-9359D916.pf
DLLHOST.EXE-C7028A70.pf
DOCKER DESKTOP INSTALLER.EXE-9EFB7658.pf
DOCKER DESKTOP.EXE-B716D532.pf
DOCKER-CREDENTIAL-WINCRED.EXE-FA7E512...
DROPBOXCRASHHANDLER.EXE-666F205E.pf
DROPBOXUPDATEONDEMAND.EXE-F118B2D8.pf
DWWIN.EXE-E356B6C3.pf
ESPORTAL-CLIENT.EXE-73FE071F.pf
FACEIT.EXE-65BF9FC5.pf
FILE-PNG.EXE-CD7C34AA.pf
FOXITREADER.EXE-5F0FF992.pf
GAMEBAR.EXE-C296016D.pf
GAMEOVERLAYUI.EXE-F813B55F.pf
GLDRIVERQUERY64.EXE-7E68D981.pf
HXOUTLOOK.EXE-D9ADA9C8.pf
ICLOUDSERVICES.EXE-46EA6659.pf
KUBECTL.EXE-20A03C5A.pf
LOGIPRESENTATIONONBOARDING.EX-988D2C...
MAINTENANCESERVICE.EXE-3122AC73.pf
MATTERMOST.EXE-0507ECBE.pf
MBSETUP.EXE-AE17E4FB.pf
MICROSOFTEDGE_X64_90.0.818.56-B5256D92.pf
_IU14D2N.TMP-FEFA14CE.pf
ADOBEARM.EXE-813E932C.pf
AM_DELTA.EXE-3A6EE7FD.pf
AM_DELTA_PATCH_1.339.270.0.EX-3DB72CD2.pf
AM_DELTA_PATCH_1.339.486.0.EX-C92BEC47.pf
AM_DELTA_PATCH_1.339.615.0.EX-187D2DE1.pf
AUDACITY.EXE-B597CCBB.pf
BACKGROUNDTASKHOST.EXE-3D3C684F.pf
BACKGROUNDTASKHOST.EXE-DBFFD4A9.pf
cadrespri.7db
CLVIEW.EXE-0282AD68.pf
CODE.EXE-C6F2EEC4.pf
CODESETUP-STABLE-CFA2E2181003-82E71D84.pf
COM.DOCKER.BACKEND.EXE-28103F36.pf
CONSENT.EXE-2D674CE4.pf
DASHOST.EXE-38AAABF0.pf
DISCORD.EXE-F2A19299.pf
DISCORD.EXE-F6C80843.pf
DISMHOST.EXE-08A6EEFF.pf
DLLHOST.EXE-3C40F7FB.pf
DLLHOST.EXE-108FCA08.pf
DLLHOST.EXE-38926D07.pf
DLLHOST.EXE-D0ADF3BF.pf
DOCKER DESKTOP INSTALLER.EXE-AF109D4C.pf
DOCKER DESKTOP.EXE-B716D533.pf
DROPBOX.EXE-41A1197E.pf
DROPBOXUPDATE.EXE-D0ECD456.pf
DRVINST.EXE-26FFA444.pf
dynrespri.7db
ESPORTAL-CLIENT-INSTALLER_FI.-F2961606.pf
FACEIT.EXE-65BF9FC6.pf
FIREFOX.EXE-25FC0A66.pf
GAMEBAR.EXE-25BA5C53.pf
GAMEBARFTSERVER.EXE-1601CA68.pf
GIMP-2.10.24-SETUP-3.TMP-196B2C99.pf
HELPER.EXE-347CAEFD.pf
ICLOUDDRIVE.EXE-0D3D04D5.pf
INNO_UPDATER.EXE-9EA3343D.pf
LOCALBRIDGE.EXE-8E78EFA0.pf
LOGIPRESENTATIONUI.EXE-738F8AF9.pf
MAINTENANCESERVICE_TMP.EXE-863239D3.pf
MATTERMOST.EXE-0507ECBF.pf
MBUNS.EXE-8FB92B97.pf
MICROSOFTEDGE_X64_90.0.818.62-8F2D29FA.pf
7ZG.EXE-D9AA3A0B.pf
ADOBEARMHELPER.EXE-157FA2C4.pf
AM_DELTA_PATCH_1.339.83.0.EXE-2B0E397C.pf
AM_DELTA_PATCH_1.339.282.0.EX-165B1DA9.pf
AM_DELTA_PATCH_1.339.498.0.EX-A1CFDD1E.pf
AM_DELTA_PATCH_1.339.643.0.EX-9A6C86BE.pf
AUDACITY-WIN-3.0.0.TMP-40A6CB0F.pf
BACKGROUNDTASKHOST.EXE-13EBA2C7.pf
BACKGROUNDTRANSFERHOST.EXE-62F7A5E0.pf
CALC.EXE-0FE8F3A9.pf
CMD.EXE-2EB3E6E2.pf
CODE.EXE-C6F2EEC5.pf
CODESETUP-STABLE-CFA2E2181003-30192614.pf
COM.DOCKER.CLI.EXE-FC031B4F.pf
CREDENTIALUIBROKER.EXE-E9F92FD0.pf
DBINSTALLER.EXE-7E26368C.pf
DISCORD.EXE-F6C8083C.pf
DISCORD.EXE-F6C80844.pf
DISMHOST.EXE-B86A0EF4.pf
DLLHOST.EXE-5C94BCB3.pf
DLLHOST.EXE-314D79DF.pf
DLLHOST.EXE-B331F1D0.pf
DLLHOST.EXE-D6E392F8.pf
DOCKER DESKTOP.EXE-85F53ACE.pf
DOCKER DESKTOP.EXE-B716D534.pf
DROPBOX.EXE-E53993B7.pf
DROPBOXUPDATE.EXE-E72FEFE1.pf
DW20.EXE-E6E0E11E.pf
EASEOFACCESSDIALOG.EXE-E54B6BCB.pf
EXCEL.EXE-FC85DF63.pf
FACEIT.EXE-65BF9FC7.pf
FONTDRVHOST.EXE-D0C00127.pf
GAMEBAR.EXE-74CFD23B.pf
GAMEBARFTSERVER.EXE-DD658A42.pf
GIMP-2.10.24-SETUP-3.TMP-A29576D0.pf
HXACCOUNTS.EXE-DF285025.pf
ICLOUDPHOTOS.EXE-87978603.pf
INSTALLER.EXE-AE17D48D.pf
LOCKAPP.EXE-61C6EAF3.pf
LOGIPRESENTATIONUPDATER.EXE-1B5B302C.pf
MAKECAB.EXE-E962779E.pf
MATTERMOST.EXE-0507ECC5.pf
MICROSOFT.AAD.BROKERPLUGIN.EX-C27248B...
MICROSOFTEDGEUPDATE.EXE-30812F03.pf
ACCOUNTSCONTROLHOST.EXE-E13D2402.pf
AM_BASE.EXE-FE51F0AA.pf
AM_DELTA_PATCH_1.339.146.0.EX-E85CC7E4.pf
AM_DELTA_PATCH_1.339.343.0.EX-1430AC0F.pf
AM_DELTA_PATCH_1.339.550.0.EX-16A962A7.pf
AM_ENGINE.EXE-79E5B6A9.pf
AUDACITY-WIN-3.0.0.TMP-E7F5422C.pf
BACKGROUNDTASKHOST.EXE-ADD2C166.pf
BACKGROUNDTRANSFERHOST.EXE-780F0463.pf
CALCULATOR.EXE-86DA154C.pf
CMD.EXE-CD245F9E.pf
CODE.EXE-C6F2EECA.pf
CODESETUP-STABLE-E713FE9B05FC-01C4FF4D....
COMPPKGSRV.EXE-6D25F3E6.pf
CSGO.EXE-7495B211.pf
DBXSVC.EXE-3A146A54.pf
DISCORD.EXE-F6C8083D.pf
DISCORDHOOKHELPER.EXE-8D6E63F2.pf
DLLHOST.EXE-02F39419.pf
DLLHOST.EXE-8F22868E.pf
DLLHOST.EXE-801B023A.pf
DLLHOST.EXE-C2B2ECC1.pf
DLLHOST.EXE-F6BAFCEE.pf
DOCKER DESKTOP.EXE-B716D53A.pf
DOCKER-CREDENTIAL-DESKTOP.EXE-C2A6178...
DROPBOXCLIENT_122.4.4867.EXE-9186790C.pf
DROPBOXUPDATEBROKER.EXE-583F1543.pf
DWM.EXE-F29FE9E2.pf
ESP3AC1.TMP-0B280A20.pf
EXPLORER.EXE-03C49D11.pf
FACEIT.EXE-65BF9FCD.pf
FONTVIEW.EXE-FAED34E4.pf
GAMEBAR.EXE-ABB9312B.pf
GAMEBARPRESENCEWRITER.EXE-FE591E8F.pf
GLDRIVERQUERY.EXE-F1B35A8F.pf
HXCALENDARAPPIMM.EXE-9E9F8F20.pf
ICLOUDPREFS.EXE-7AF92479.pf
INSTALLER_FI.EXE-DD7B58DF.pf
LOGIPRESENTATION.EXE-55865445.pf
LOGONUI.EXE-E35F76FB.pf
MATTERMOST.EXE-0507ECBD.pf
MAVINJECT32.EXE-A39D6437.pf
MICROSOFT.PHOTOS.EXE-581631F7.pf
MKSSANDBOX.EXE-82860B1C.pf
458 items | 1 item selected 13,9 KB

TLP:WHITE

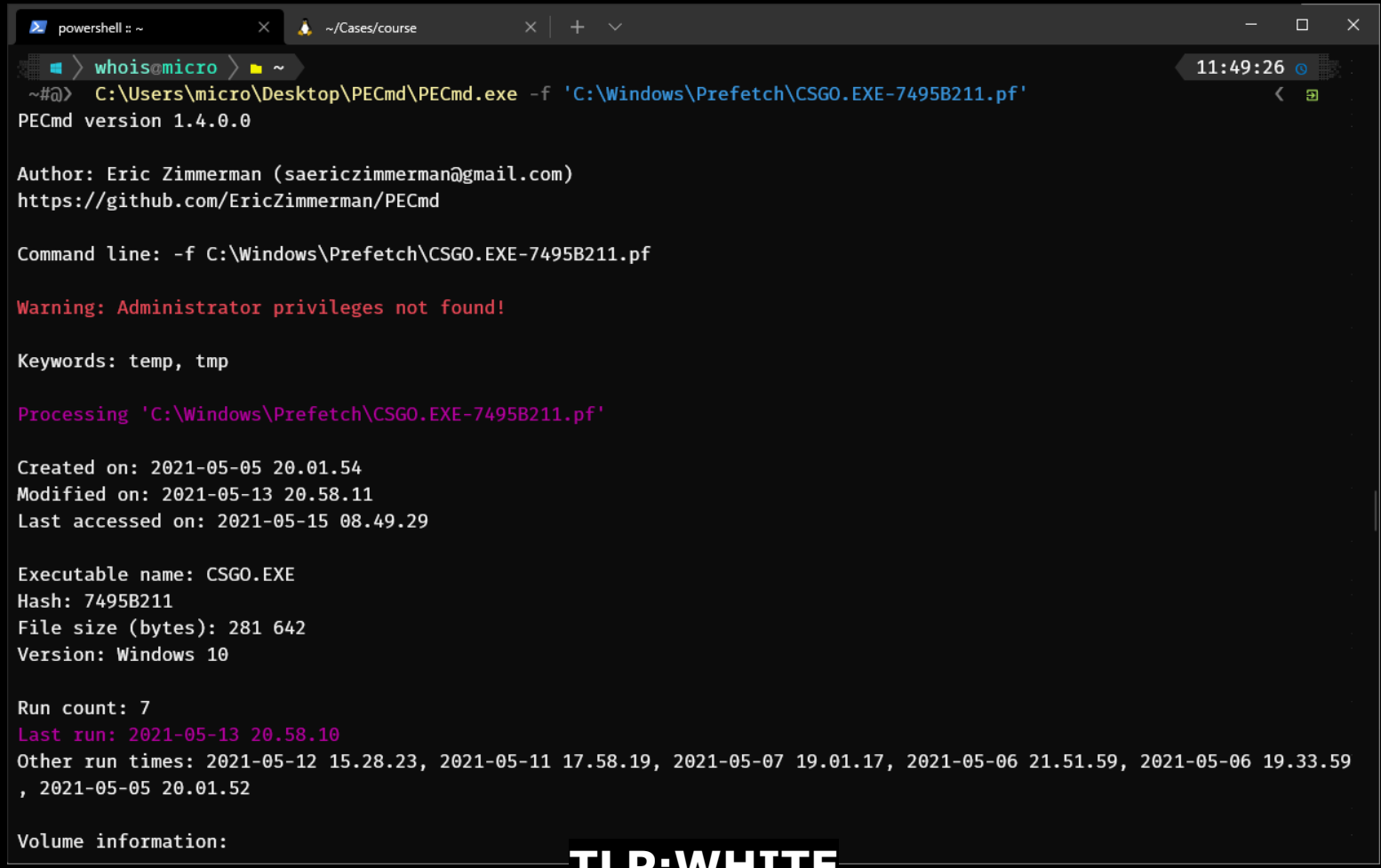
Available tools

- Windows 10 Prefetch Parser (w10pf_parse.py)
  - https://github.com/bromiley/tools/tree/master/win10_prefetch
  - Script that works on Linux, limited output
  - pip2 install libscca-python && wget https://raw.githubusercontent.com/bromiley/tools/master/win10_prefetch/w10pf_parse.py
- Prefetch parser (PECmd)
  - https://ericzimmerman.github.io/#!index.md
  - Windows only

TLP:WHITE


powershell :: ~
~/Cases/course
~/Cases/course
11:48:28
python2 w10pf_parse.py -f /mnt/c/Windows/Prefetch/CSGO.EXE-7495B211.pf --json
{
"Executable Name": "CSGO.EXE",
"Prefetch Hash": "7495B211",
"Run Count": "7",
"Run Times": {
"Run Time 0": "2021-05-13 20:58:10",
"Run Time 1": "2021-05-12 15:28:23",
"Run Time 2": "2021-05-11 17:58:19",
"Run Time 3": "2021-05-07 19:01:17",
"Run Time 4": "2021-05-06 21:51:59",
"Run Time 5": "2021-05-06 19:33:59",
"Run Time 6": "2021-05-05 20:01:52",
"Run Time 7": "N/A"
}
}
~/Cases/course
11:48:35

TLP:WHITE

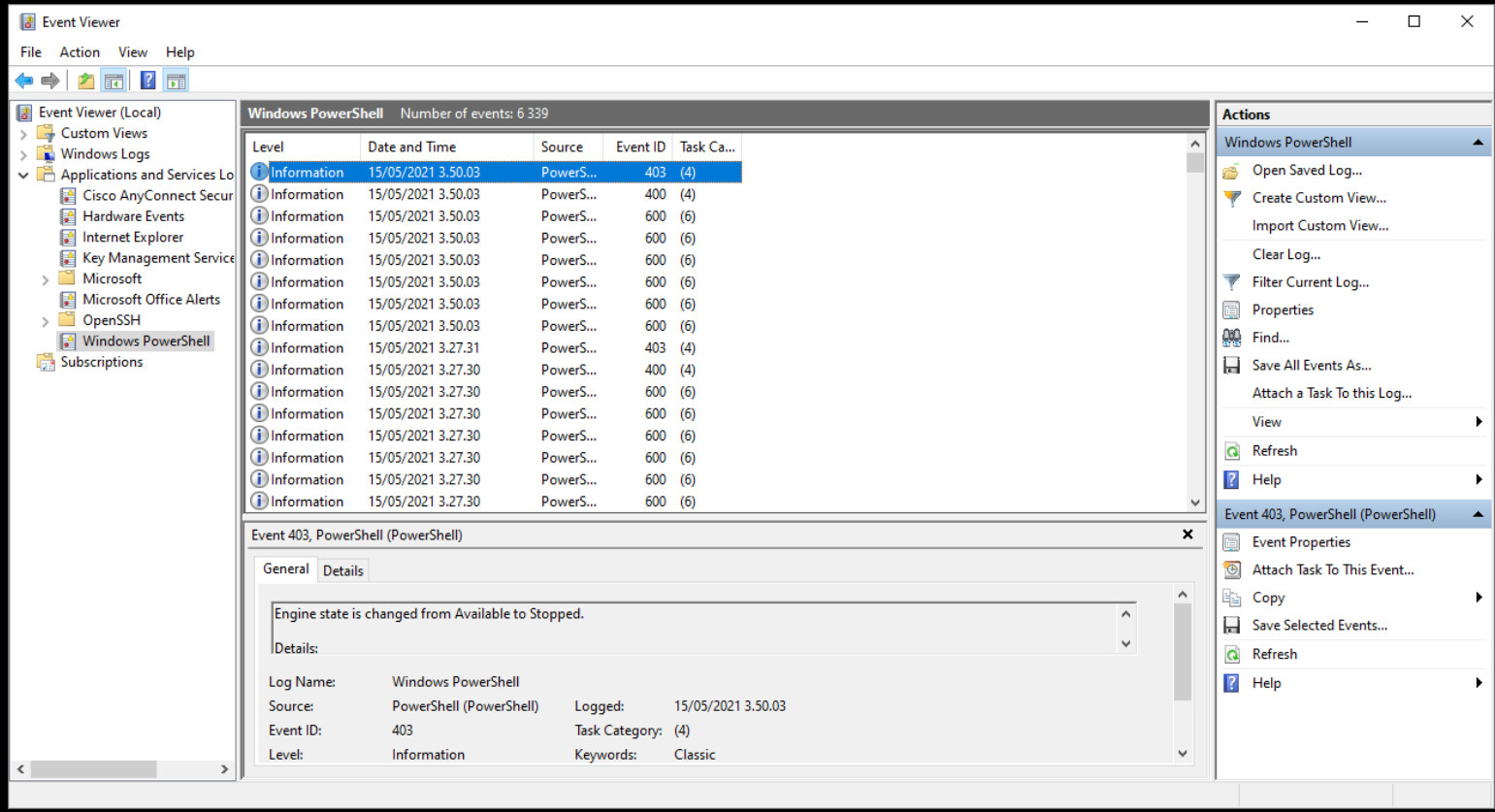

powershell :: ~
~/Cases/course
whois@micro ~
11:49:26
~#@> C:\Users\micro\Desktop\PECmd\PECmd.exe -f 'C:\Windows\Prefetch\CSGO.EXE-7495B211.pf'
PECmd version 1.4.0.0
Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/PECmd
Command line: -f C:\Windows\Prefetch\CSGO.EXE-7495B211.pf
Warning: Administrator privileges not found!
Keywords: temp, tmp
Processing 'C:\Windows\Prefetch\CSGO.EXE-7495B211.pf'
Created on: 2021-05-05 20.01.54
Modified on: 2021-05-13 20.58.11
Last accessed on: 2021-05-15 08.49.29
Executable name: CSGO.EXE
Hash: 7495B211
File size (bytes): 281 642
Version: Windows 10
Run count: 7
Last run: 2021-05-13 20.58.10
Other run times: 2021-05-12 15.28.23, 2021-05-11 17.58.19, 2021-05-07 19.01.17, 2021-05-06 21.51.59, 2021-05-06 19.33.59
, 2021-05-05 20.01.52
Volume information:

TLP:WHITE

TLP:WHITE

winevtx

TLP:WHITE


Event Viewer
File Action View Help
Event Viewer (Local)
Custom Views
Windows Logs
Applications and Services Lo
Cisco AnyConnect Secur
Hardware Events
Internet Explorer
Key Management Service
Microsoft
Microsoft Office Alerts
OpenSSH
Windows PowerShell
Subscriptions
Windows PowerShell Number of events: 6 339
Level Date and Time Source Event ID Task Ca...
Information 15/05/2021 3.50.03 PowerS... 403 (4)
Information 15/05/2021 3.50.03 PowerS... 400 (4)
Information 15/05/2021 3.50.03 PowerS... 600 (6)
Information 15/05/2021 3.50.03 PowerS... 600 (6)
Information 15/05/2021 3.50.03 PowerS... 600 (6)
Information 15/05/2021 3.50.03 PowerS... 600 (6)
Information 15/05/2021 3.50.03 PowerS... 600 (6)
Information 15/05/2021 3.50.03 PowerS... 600 (6)
Information 15/05/2021 3.27.31 PowerS... 403 (4)
Information 15/05/2021 3.27.30 PowerS... 400 (4)
Information 15/05/2021 3.27.30 PowerS... 600 (6)
Information 15/05/2021 3.27.30 PowerS... 600 (6)
Information 15/05/2021 3.27.30 PowerS... 600 (6)
Information 15/05/2021 3.27.30 PowerS... 600 (6)
Information 15/05/2021 3.27.30 PowerS... 600 (6)
Information 15/05/2021 3.27.30 PowerS... 600 (6)
Information 15/05/2021 3.27.30 PowerS... 600 (6)
Event 403, PowerShell (PowerShell)
General Details
Engine state is changed from Available to Stopped.
Details:
Log Name: Windows PowerShell
Source: PowerShell (PowerShell) Logged: 15/05/2021 3.50.03
Event ID: 403 Task Category: (4)
Level: Information Keywords: Classic
Actions
Windows PowerShell
Open Saved Log...
Create Custom View...
Import Custom View...
Clear Log...
Filter Current Log...
Properties
Find...
Save All Events As...
Attach a Task To this Log...
View
Refresh
Help
Event 403, PowerShell (PowerShell)
Event Properties
Attach Task To This Event...
Copy
Save Selected Events...
Refresh
Help

TLP:WHITE

What and where?

- Windows logs different stuff to different places
  - System logs – System events
  - Security logs – For example authentication events
  - Powershell logs – Powershell execution
  - Sysmon – Can log basically anything, can be very verbose, installed and configured rarely
- Logging level depends on the system configuration
- The Windows XML EventLog (EVTX) format
  - Stored at %SystemRoot%\System32\Winevt\Logs

Available tools

- Windows event viewer
  - Ships with your Windows installation
- ELK / Splunk /etc.
- Evtx-dump.py / evtx_dump
  - Converts EVTX to XML
  - Ships with SIFT / REMnux
  - Rust version can be installed with cargo

TLP:WHITE

TLP:WHITE

Check these

TLP:WHITE

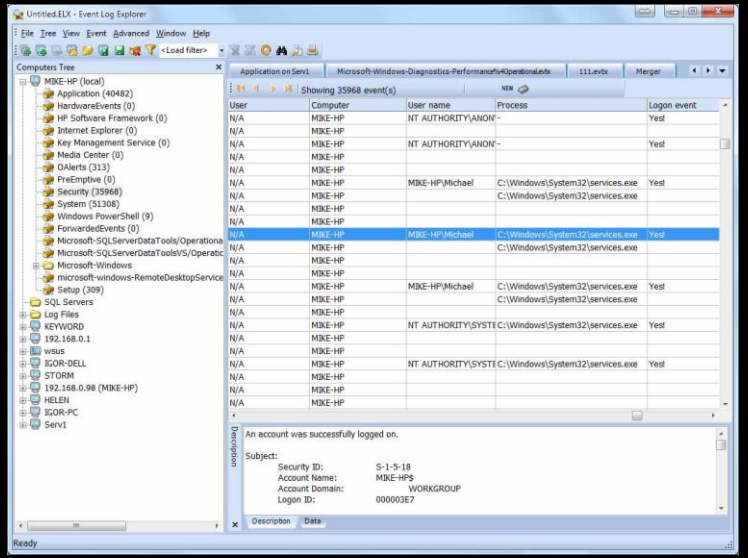
Zimmerman tools

- https://ericzimmerman.github.io/#!index.md
- Great set of FOSS tools
- My personal favorites:
  - Registry Explorer/RECmd
  - RBCmd
  - PECmd
  - JumpList Explorer
  - LECmd
  - Timeline Explorer
  - KAPE [***]

TLP:WHITE

Event Log Explorer™

- https://eventlogxp.com/
- 199 USD for Standard Edition
- Bang for the bucks
- Make custom searches and views for Eventlogs



https://eventlogxp.com/sshots/customcolumns.jpg

TLP:WHITE

Autopsy

- OpenSource tool for disk forensics
- Parses lot of different artefacts and indexes them
- https://www.autopsy.com/

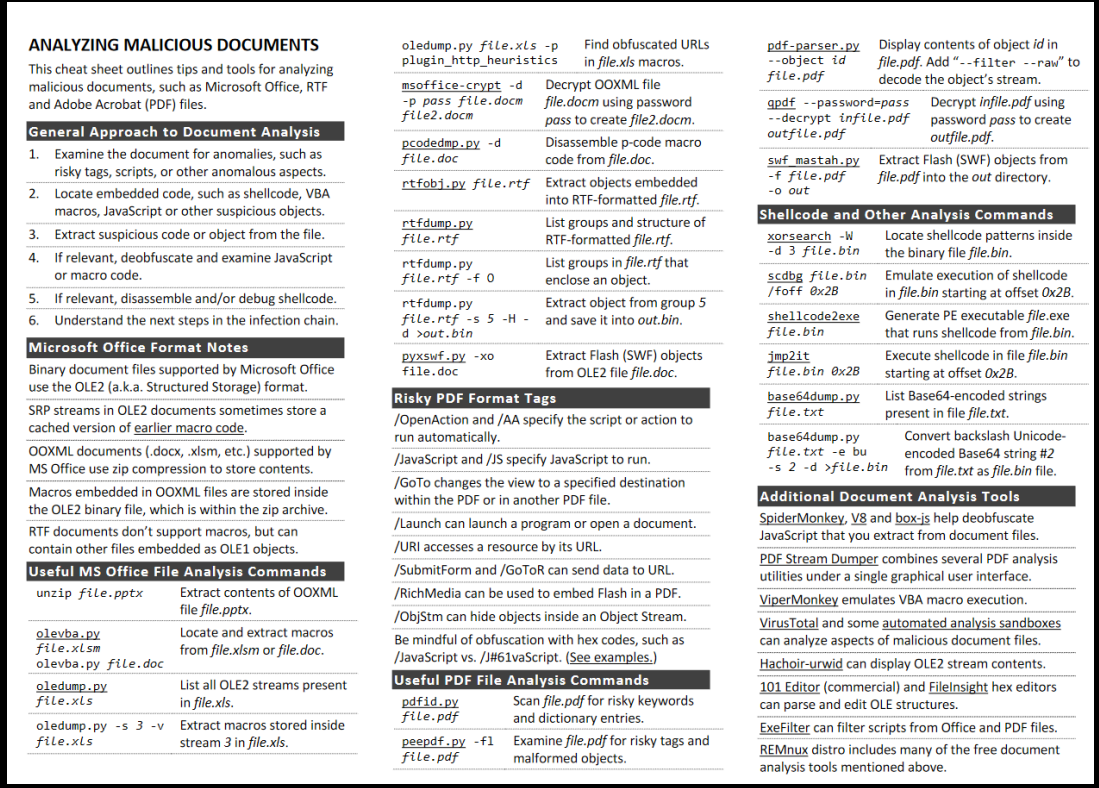
TLP:WHITE

Plaso everything

- In my experience, you should always run plaso/log2timeline against everything
  - Eases up setting up incident timeline
  - Might help you “find evil”
- Even though you would not investigate the timeline immediately, you might need it later and plaso takes some time…
- https://github.com/log2timeline/plaso
- https://github.com/google/timesketch

TLP:WHITE

Analyze malicious documents



ANALYZING MALICIOUS DOCUMENTS

This cheat sheet outlines tips and tools for analyzing malicious documents, such as Microsoft Office, RTF and Adobe Acrobat (PDF) files.

General Approach to Document Analysis

1. Examine the document for anomalies, such as risky tags, scripts, or other anomalous aspects.
2. Locate embedded code, such as shellcode, VBA macros, JavaScript or other suspicious objects.
3. Extract suspicious code or object from the file.
4. If relevant, deobfuscate and examine JavaScript or macro code.
5. If relevant, disassemble and/or debug shellcode.
6. Understand the next steps in the infection chain.

Microsoft Office Format Notes

Binary document files supported by Microsoft Office use the OLE2 (a.k.a. Structured Storage) format.

SRP streams in OLE2 documents sometimes store a cached version of earlier macro code.

OOXML documents (.docx, .xlsm, etc.) supported by MS Office use zip compression to store contents.

Macros embedded in OOXML files are stored inside the OLE2 binary file, which is within the zip archive.

RTF documents don't support macros, but can contain other files embedded as OLE1 objects.

Useful MS Office File Analysis Commands

`unzip file.pptx`	Extract contents of OOXML file *file.pptx*.
`olevba.py file.xlsm` `olevba.py file.doc`	Locate and extract macros from *file.xlsm* or *file.doc*.
`oledump.py file.xls`	List all OLE2 streams present in *file.xls*.
`oledump.py -s 3 -v file.xls`	Extract macros stored inside stream *3* in *file.xls*.
`oledump.py file.xls -p plugin_http_heuristics`	Find obfuscated URLs in *file.xls* macros.
`msoffice-crypt -d -p pass file.docm file2.docm`	Decrypt OOXML file *file.docm* using password *pass* to create *file2.docm*.
`pcodedmp.py -d file.doc`	Disassemble p-code macro code from *file.doc*.
`rtfobj.py file.rtf`	Extract objects embedded into RTF-formatted *file.rtf*.
`rtfdump.py file.rtf`	List groups and structure of RTF-formatted *file.rtf*.
`rtfdump.py file.rtf -f 0`	List groups in *file.rtf* that enclose an object.
`rtfdump.py file.rtf -s 5 -H -d >out.bin`	Extract object from group *5* and save it into *out.bin*.
`pyxswf.py -xo file.doc`	Extract Flash (SWF) objects from OLE2 file *file.doc*.

Risky PDF Format Tags

/OpenAction and /AA specify the script or action to run automatically.

/JavaScript and /JS specify JavaScript to run.

/GoTo changes the view to a specified destination within the PDF or in another PDF file.

/Launch can launch a program or open a document.

/URI accesses a resource by its URL.

/SubmitForm and /GoToR can send data to URL.

/RichMedia can be used to embed Flash in a PDF.

/ObjStm can hide objects inside an Object Stream.

Be mindful of obfuscation with hex codes, such as /JavaScript vs. /J#61vaScript. (See examples.)

Useful PDF File Analysis Commands

`pdfid.py file.pdf`	Scan *file.pdf* for risky keywords and dictionary entries.
`peepdf.py -fl file.pdf`	Examine *file.pdf* for risky tags and malformed objects.
`pdf-parser.py --object id file.pdf`	Display contents of object *id* in *file.pdf*. Add "`--filter --raw`" to decode the object's stream.
`qpdf --password=pass --decrypt infile.pdf outfile.pdf`	Decrypt *infile.pdf* using password *pass* to create *outfile.pdf*.
`swf_mastah.py -f file.pdf -o out`	Extract Flash (SWF) objects from *file.pdf* into the *out* directory.

Shellcode and Other Analysis Commands

`xorsearch -W -d 3 file.bin`	Locate shellcode patterns inside the binary file *file.bin*.
`scdbg file.bin /foff 0x2B`	Emulate execution of shellcode in *file.bin* starting at offset *0x2B*.
`shellcode2exe file.bin`	Generate PE executable *file.exe* that runs shellcode from *file.bin*.
`jmp2it file.bin 0x2B`	Execute shellcode in file *file.bin* starting at offset *0x2B*.
`base64dump.py file.txt`	List Base64-encoded strings present in file *file.txt*.
`base64dump.py file.txt -e bu -s 2 -d >file.bin`	Convert backslash Unicode-encoded Base64 string *#2* from *file.txt* as *file.bin* file.

Additional Document Analysis Tools

SpiderMonkey, V8 and box-js help deobfuscate JavaScript that you extract from document files.

PDF Stream Dumper combines several PDF analysis utilities under a single graphical user interface.

ViperMonkey emulates VBA macro execution.

VirusTotal and some automated analysis sandboxes can analyze aspects of malicious document files.

Hachoir-urwid can display OLE2 stream contents.

101 Editor (commercial) and FileInsight hex editors can parse and edit OLE structures.

ExeFilter can filter scripts from Office and PDF files.

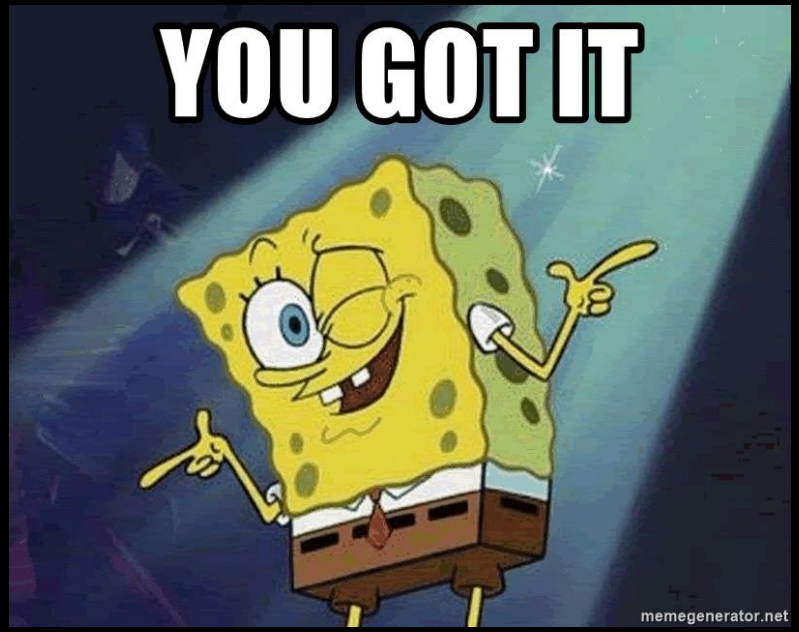
REMnux distro includes many of the free document analysis tools mentioned above.

https://zeltser.com/analyzing-malicious-documents/

TLP:WHITE

Final words

TLP:WHITE



TLP:WHITE

CTF time

- Open 24h from this moment
- COLLABORATE !!! Share knowledge !
- Do not hack the CTFd platform plz
- The best (TOP3) players will be awarded with HelSec SWAG
  - Winners will be announced on my Twitter account: @JuhoJauhiainen
- https://mayhem.dfir.fi
- Password for evidence:

  <WILL BE PRESENTED IN THE STREAM>

TLP:WHITE

T. Hanks!

Questions?

@JuhoJauhiainen

whois [at] helsec.fi

juho.jauhiainen [at] traficom.fi

TLP:WHITE