# 領會威脅情資研究祕法

Threat Intelligence 101

Still Hsu

TEAMT5

Persistent Cyber Threat Hunters

# AGENDA

TEAMT5

## 01 走入 CTI 的第一步

- CTI 到底是啥？
- CTI 產品生命週期

## 02 我的第一篇情資報告

- 報告 SOP
- 找尋敵人以及相關研究
- 研究事件樣本並拼湊出事件發生的由來

## 03 Lab #1: 工欲善其事，必先利其器

- 調查事件總不能每次都空手硬幹或遇到心樣本幹掉重練，學會用點工具吧！

## 04 Lab #2: 從蛛絲馬跡到破案關鍵

- 學會如何透過惡意樣本分析，一步一腳印蒐集各式各樣的資訊，並找出背後的藏鏡人。

## 05 Lab #3: 我的第一個 YARA 規則

- 為了未來而準備，開始寫你的第一支 YARA 規則來進行威脅狩獵吧！

## whoami

TEAMT5

### Still Hsu / 安坂星海

- BEL, English Dep. @ NPTU (屏東大學)
  - Pingtung Hacker TA
- Threat Intelligence Researcher @ TeamT5
- Interested in...
  - Windows internals
  - .NET
  - Occasional VTube streams
  - Anything and everything!

# Disclaimer

This lab session assumes you have…

- Basic reverse engineering skills
- A disassembler/decompiler installed
  - Preferably IDA Pro, though any other ones are fine
- A **CONTAINED ENVIRONMENT** for testing (e.g., VM) that should be **OFFLINE**

**The lab session WILL require interaction with a real malware.**

- If you are not confident enough, don’t risk it.
- Feel free to watch others do it instead.

## Introduction to CTI

TEAMT5

## What is CTI, anyways?

## What is CTI, anyways?

“Threat intelligence is data that is collected, processed, and analyzed to understand a threat actor’s motives, targets, and attack behaviors.”

- (CrowdStrike, 2021)

# What is CTI, anyways?

- Past
- Present
- Future

Records and collects indicators of cyberthreats

# What is CTI, anyways?

- Past
- Present
- Future

Records and collects indicators of cyberthreats

Provides additional insights into these threats

- Tactics
- Operations
- Strategies

# What is CTI, anyways?

- Past
- Present
- Future

Records and collects indicators of cyberthreats

Provides additional insights into these threats

- Tactics
- Operations
- Strategies

- Reduce impact
- Improve IR efficiency
- Gain better knowledge in the cyberthreat landscape

Helps businesses plan for mitigations

TEAMT5

## Why CTI?

A decade ago…

- Typical incident responses
- Process and provide feedback as cases come along
- Number of cases were few and far between
- Most of them were trivial
- Relatively easy to handle

TEAMT5

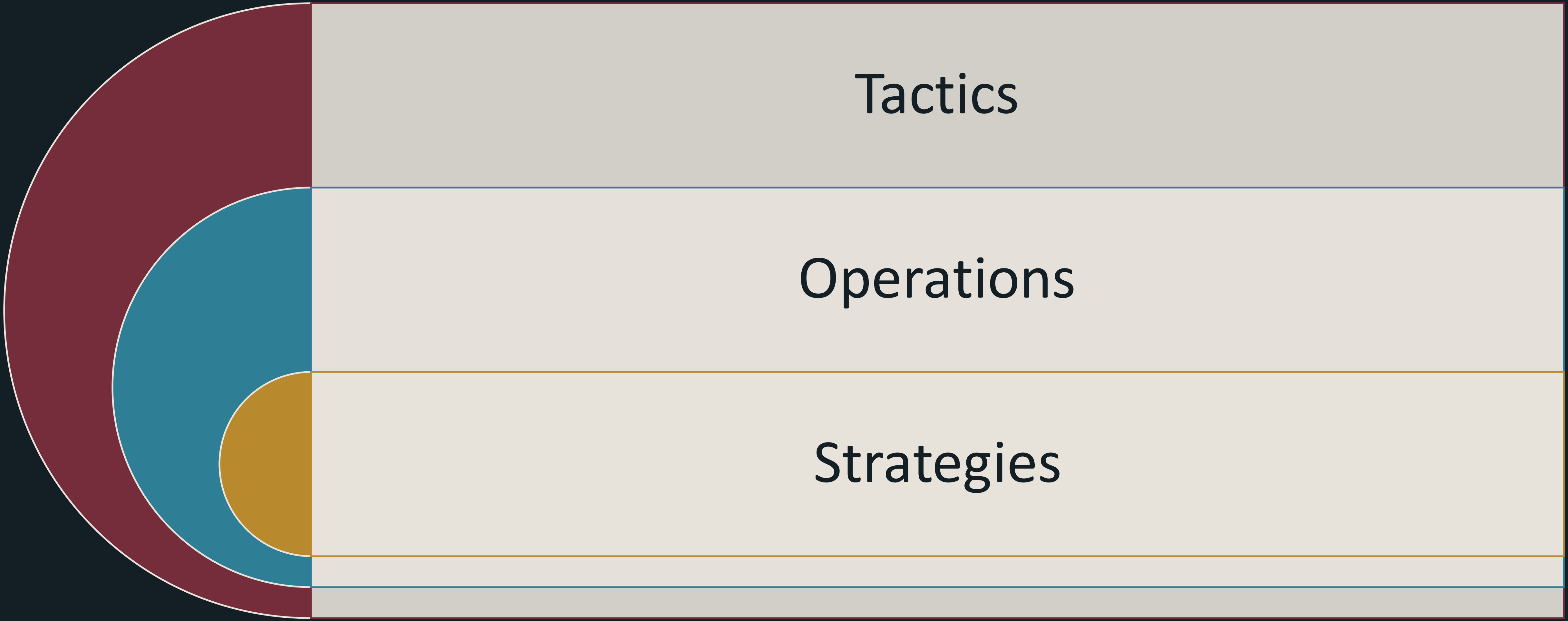
## Why CTI?

| | |
|---|---|
| A decade ago… | Typical incident responses |
| | Process and provide feedback as cases come along |
| | Number of cases were few and far between |
| | Most of them were trivial |
| | Relatively easy to handle |
| Now… | Complex attacks are now being carried out by APTs worldwide |
| | Number of sophisticated social engineering attacks have been increasing yearly |
| | Signature-based defenses are being defeated left and right |

(Jelen, 2019)

TEAMT5

## Who uses CTI?



## Who uses CTI?

| | |
|---|---|
| Tactics | • Valid malware and traffic signatures<br>• Critical vulnerabilities |
| Operations | • Detailed context around alerts and events |
| Strategies | • Better understanding of trends<br>• Better decision-making regarding security budgets, improvements, technological requirements |

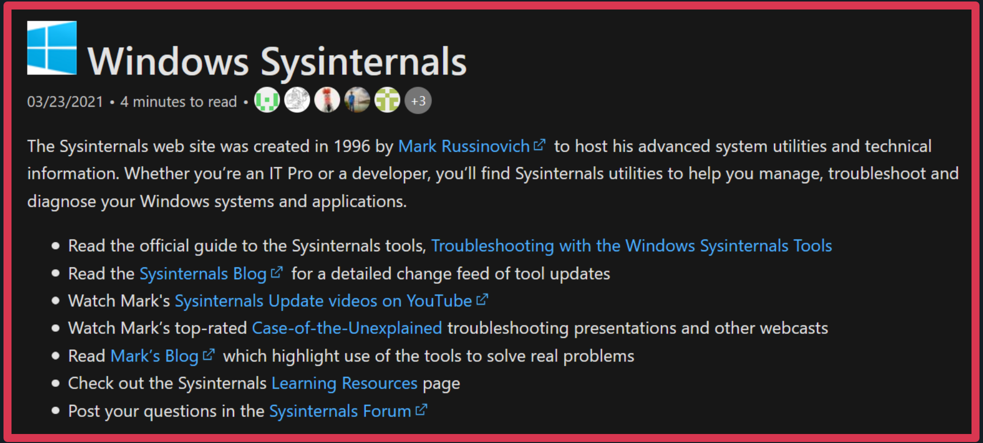
## Sounds scary…

I don’t know anything about threat research.

Neither did I!

## Lab #1: Getting the Hang of Tools

# Sysinternals Suite



**Windows Sysinternals**

03/23/2021 • 4 minutes to read • +3

The Sysinternals web site was created in 1996 by Mark Russinovich to host his advanced system utilities and technical information. Whether you're an IT Pro or a developer, you'll find Sysinternals utilities to help you manage, troubleshoot and diagnose your Windows systems and applications.

- Read the official guide to the Sysinternals tools, Troubleshooting with the Windows Sysinternals Tools
- Read the Sysinternals Blog for a detailed change feed of tool updates
- Watch Mark's Sysinternals Update videos on YouTube
- Watch Mark's top-rated Case-of-the-Unexplained troubleshooting presentations and other webcasts
- Read Mark's Blog which highlight use of the tools to solve real problems
- Check out the Sysinternals Learning Resources page
- Post your questions in the Sysinternals Forum

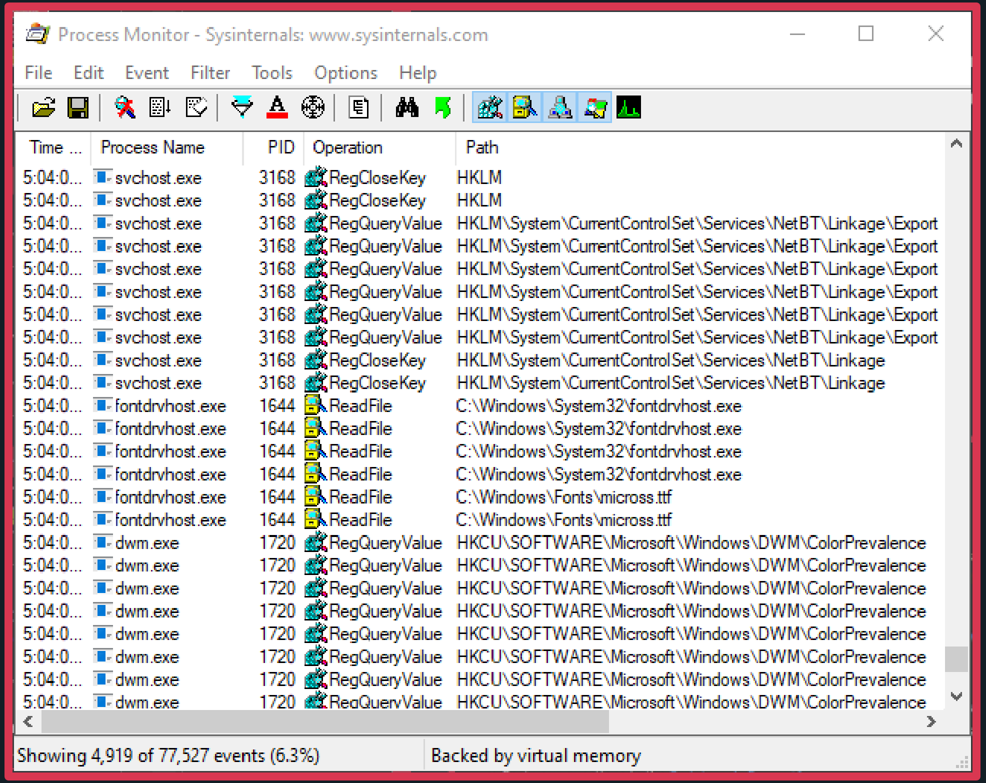
(Microsoft Corp., 2021)

- Description
  - Originally third-party, now acquired by Microsoft
  - Contains a series of tools for system management and Windows debugging

# Sysinternals Suite

TEAMT5

- Procmon
  - *Process Monitor* is an advanced monitoring tool for Windows that shows real-time file system, Registry and process/thread activity.
  - Helps give a quick idea of what the malware will do upon startup.



Process Monitor - Sysinternals: www.sysinternals.com

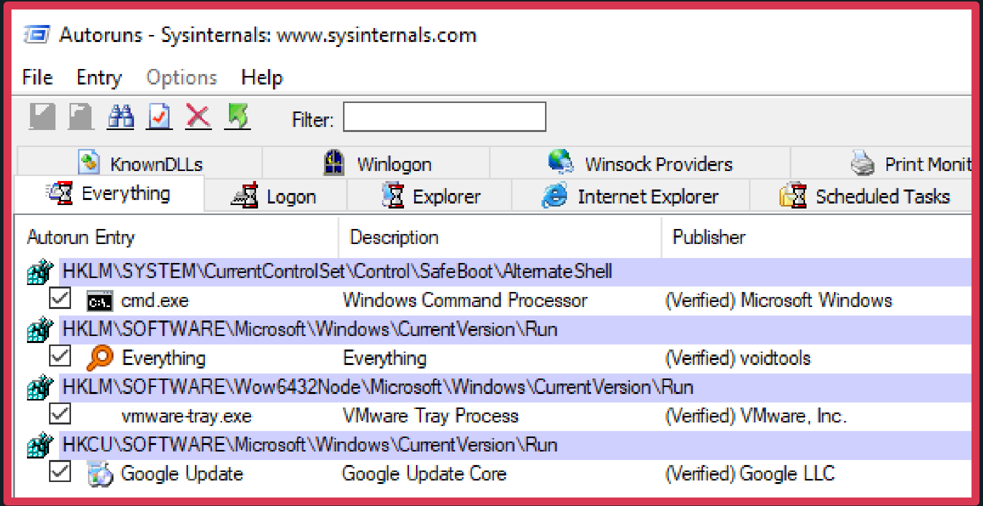
File Edit Event Filter Tools Options Help

```
Time ...   Process Name      PID   Operation       Path
5:04:0...  svchost.exe       3168  RegCloseKey     HKLM
5:04:0...  svchost.exe       3168  RegCloseKey     HKLM
5:04:0...  svchost.exe       3168  RegQueryValue   HKLM\System\CurrentControlSet\Services\NetBT\Linkage\Export
5:04:0...  svchost.exe       3168  RegQueryValue   HKLM\System\CurrentControlSet\Services\NetBT\Linkage\Export
5:04:0...  svchost.exe       3168  RegQueryValue   HKLM\System\CurrentControlSet\Services\NetBT\Linkage\Export
5:04:0...  svchost.exe       3168  RegQueryValue   HKLM\System\CurrentControlSet\Services\NetBT\Linkage\Export
5:04:0...  svchost.exe       3168  RegQueryValue   HKLM\System\CurrentControlSet\Services\NetBT\Linkage\Export
5:04:0...  svchost.exe       3168  RegQueryValue   HKLM\System\CurrentControlSet\Services\NetBT\Linkage\Export
5:04:0...  svchost.exe       3168  RegCloseKey     HKLM\System\CurrentControlSet\Services\NetBT\Linkage
5:04:0...  svchost.exe       3168  RegCloseKey     HKLM\System\CurrentControlSet\Services\NetBT\Linkage
5:04:0...  fontdrvhost.exe   1644  ReadFile        C:\Windows\System32\fontdrvhost.exe
5:04:0...  fontdrvhost.exe   1644  ReadFile        C:\Windows\System32\fontdrvhost.exe
5:04:0...  fontdrvhost.exe   1644  ReadFile        C:\Windows\System32\fontdrvhost.exe
5:04:0...  fontdrvhost.exe   1644  ReadFile        C:\Windows\System32\fontdrvhost.exe
5:04:0...  fontdrvhost.exe   1644  ReadFile        C:\Windows\Fonts\micross.ttf
5:04:0...  fontdrvhost.exe   1644  ReadFile        C:\Windows\Fonts\micross.ttf
5:04:0...  dwm.exe           1720  RegQueryValue   HKCU\SOFTWARE\Microsoft\Windows\DWM\ColorPrevalence
5:04:0...  dwm.exe           1720  RegQueryValue   HKCU\SOFTWARE\Microsoft\Windows\DWM\ColorPrevalence
5:04:0...  dwm.exe           1720  RegQueryValue   HKCU\SOFTWARE\Microsoft\Windows\DWM\ColorPrevalence
5:04:0...  dwm.exe           1720  RegQueryValue   HKCU\SOFTWARE\Microsoft\Windows\DWM\ColorPrevalence
5:04:0...  dwm.exe           1720  RegQueryValue   HKCU\SOFTWARE\Microsoft\Windows\DWM\ColorPrevalence
5:04:0...  dwm.exe           1720  RegQueryValue   HKCU\SOFTWARE\Microsoft\Windows\DWM\ColorPrevalence
5:04:0...  dwm.exe           1720  RegQueryValue   HKCU\SOFTWARE\Microsoft\Windows\DWM\ColorPrevalence
5:04:0...  dwm.exe           1720  RegQueryValue   HKCU\SOFTWARE\Microsoft\Windows\DWM\ColorPrevalence
```

Showing 4,919 of 77,527 events (6.3%)    Backed by virtual memory

# Sysinternals Suite

TEAMT5

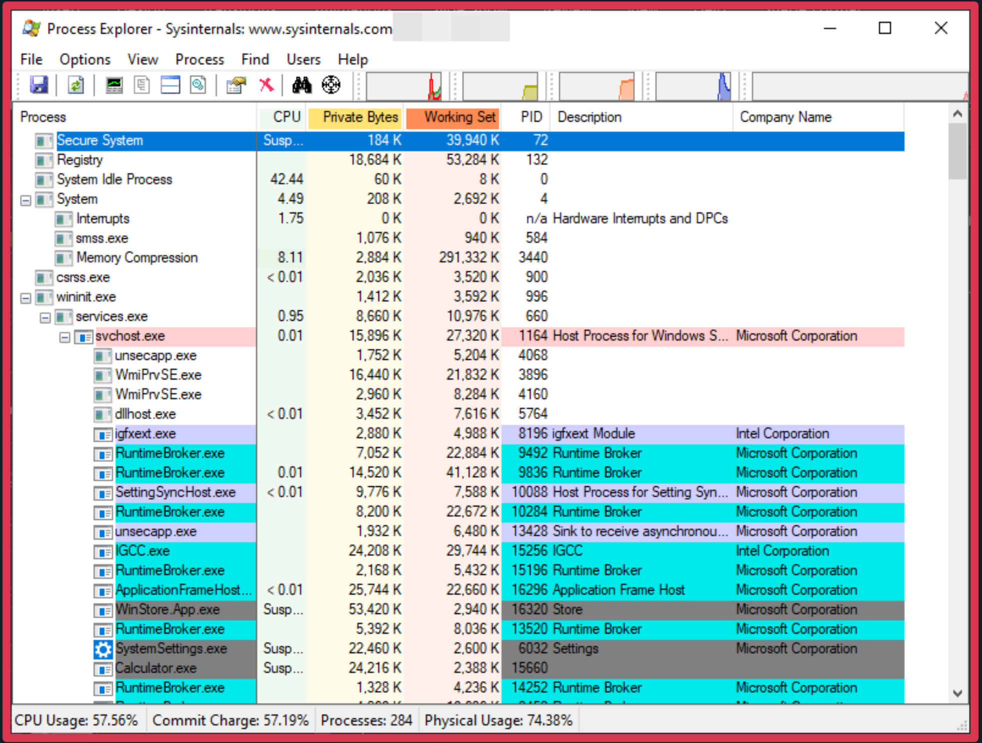

Autoruns - Sysinternals: www.sysinternals.com

| Autorun Entry | Description | Publisher |
|---|---|---|
| HKLM\SYSTEM\CurrentControlSet\Control\SafeBoot\AlternateShell | | |
| ☑ cmd.exe | Windows Command Processor | (Verified) Microsoft Windows |
| HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run | | |
| ☑ Everything | Everything | (Verified) voidtools |
| HKLM\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run | | |
| ☑ vmware-tray.exe | VMware Tray Process | (Verified) VMware, Inc. |
| HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run | | |
| ☑ Google Update | Google Update Core | (Verified) Google LLC |

- AutoRuns
  - *AutoRuns* shows you what programs are configured to run during system bootup or login, and when you start various built-in Windows applications like Internet Explorer, Explorer and media players.
  - Quick overview of the existing persistence entries on the machine.

# Sysinternals Suite

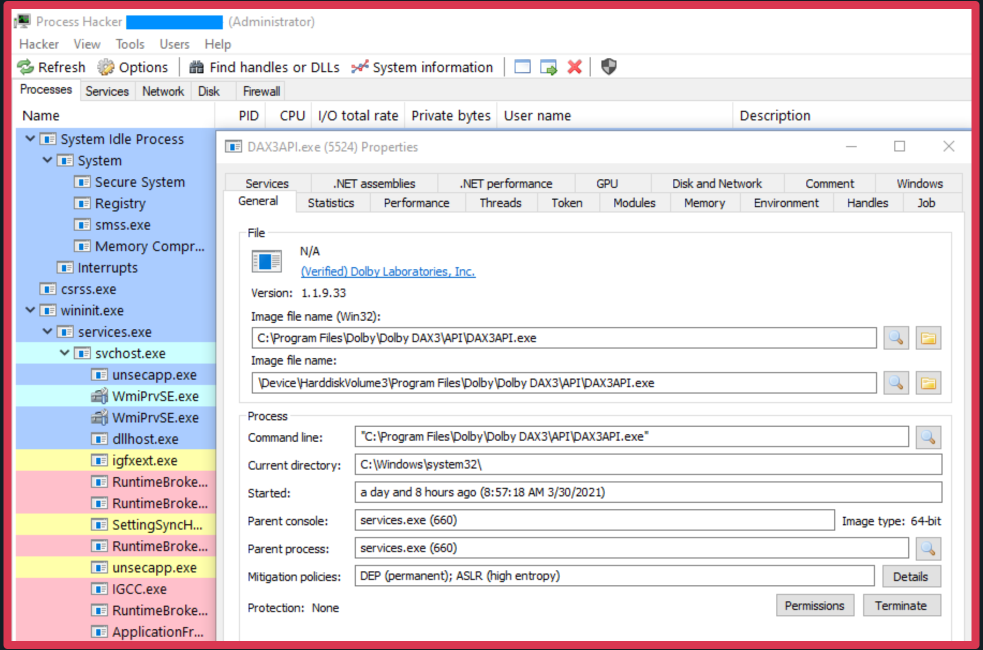
- Process Explorer
  - *Process Explorer* shows you information about which handles, and DLLs processes have opened or loaded.
  - Buffed up Task Manager, useful for dynamic analysis (e.g., memory dump, handle listing, etc.)



| Process | CPU | Private Bytes | Working Set | PID | Description | Company Name |
|---|---|---|---|---|---|---|
| Secure System | Susp... | 184 K | 39,940 K | 72 | | |
| Registry | | 18,684 K | 53,284 K | 132 | | |
| System Idle Process | 42.44 | 60 K | 8 K | 0 | | |
| System | 4.49 | 208 K | 2,692 K | 4 | | |
| Interrupts | 1.75 | 0 K | 0 K | n/a | Hardware Interrupts and DPCs | |
| smss.exe | | 1,076 K | 940 K | 584 | | |
| Memory Compression | 8.11 | 2,884 K | 291,332 K | 3440 | | |
| csrss.exe | < 0.01 | 2,036 K | 3,520 K | 900 | | |
| wininit.exe | | 1,412 K | 3,592 K | 996 | | |
| services.exe | 0.95 | 8,660 K | 10,976 K | 660 | | |
| svchost.exe | 0.01 | 15,896 K | 27,320 K | 1164 | Host Process for Windows S... | Microsoft Corporation |
| unsecapp.exe | | 1,752 K | 5,204 K | 4068 | | |
| WmiPrvSE.exe | | 16,440 K | 21,832 K | 3896 | | |
| WmiPrvSE.exe | | 2,960 K | 8,284 K | 4160 | | |
| dllhost.exe | < 0.01 | 3,452 K | 7,616 K | 5764 | | |
| igfxext.exe | | 2,880 K | 4,988 K | 8196 | igfxext Module | Intel Corporation |
| RuntimeBroker.exe | | 7,052 K | 22,884 K | 9492 | Runtime Broker | Microsoft Corporation |
| RuntimeBroker.exe | 0.01 | 14,520 K | 41,128 K | 9836 | Runtime Broker | Microsoft Corporation |
| SettingSyncHost.exe | < 0.01 | 9,776 K | 7,588 K | 10088 | Host Process for Setting Syn... | Microsoft Corporation |
| RuntimeBroker.exe | | 8,200 K | 22,672 K | 10284 | Runtime Broker | Microsoft Corporation |
| unsecapp.exe | | 1,932 K | 6,480 K | 13428 | Sink to receive asynchronou... | Microsoft Corporation |
| IGCC.exe | | 24,208 K | 29,744 K | 15256 | IGCC | Intel Corporation |
| RuntimeBroker.exe | | 2,168 K | 5,432 K | 15196 | Runtime Broker | Microsoft Corporation |
| ApplicationFrameHost... | < 0.01 | 25,744 K | 22,660 K | 16296 | Application Frame Host | Microsoft Corporation |
| WinStore.App.exe | Susp... | 53,420 K | 2,940 K | 16320 | Store | Microsoft Corporation |
| RuntimeBroker.exe | | 5,392 K | 8,036 K | 13520 | Runtime Broker | Microsoft Corporation |
| SystemSettings.exe | Susp... | 22,460 K | 2,600 K | 6032 | Settings | Microsoft Corporation |
| Calculator.exe | Susp... | 24,216 K | 2,388 K | 15660 | | |
| RuntimeBroker.exe | | 1,328 K | 4,236 K | 14252 | Runtime Broker | Microsoft Corporation |

CPU Usage: 57.56% | Commit Charge: 57.19% | Processes: 284 | Physical Usage: 74.38%

# Other Third-party Tools



Process Hacker [redacted] (Administrator)
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
Processes Services Network Disk Firewall
Name PID CPU I/O total rate Private bytes User name Description

```
System Idle Process
  System
    Secure System
    Registry
    smss.exe
    Memory Compr...
  Interrupts
csrss.exe
wininit.exe
  services.exe
    svchost.exe
      unsecapp.exe
      WmiPrvSE.exe
      WmiPrvSE.exe
      dllhost.exe
      igfxext.exe
      RuntimeBroke...
      RuntimeBroke...
      SettingSyncH...
      RuntimeBroke...
      unsecapp.exe
      IGCC.exe
      RuntimeBroke...
      ApplicationFr...
```

DAX3API.exe (5524) Properties
Services .NET assemblies .NET performance GPU Disk and Network Comment Windows
General Statistics Performance Threads Token Modules Memory Environment Handles Job
Details
Permissions
Terminate

File

N/A

(Verified) Dolby Laboratories, Inc.

Version: 1.1.9.33

Image file name (Win32): C:\Program Files\Dolby\Dolby DAX3\API\DAX3API.exe

Image file name: \Device\HarddiskVolume3\Program Files\Dolby\Dolby DAX3\API\DAX3API.exe

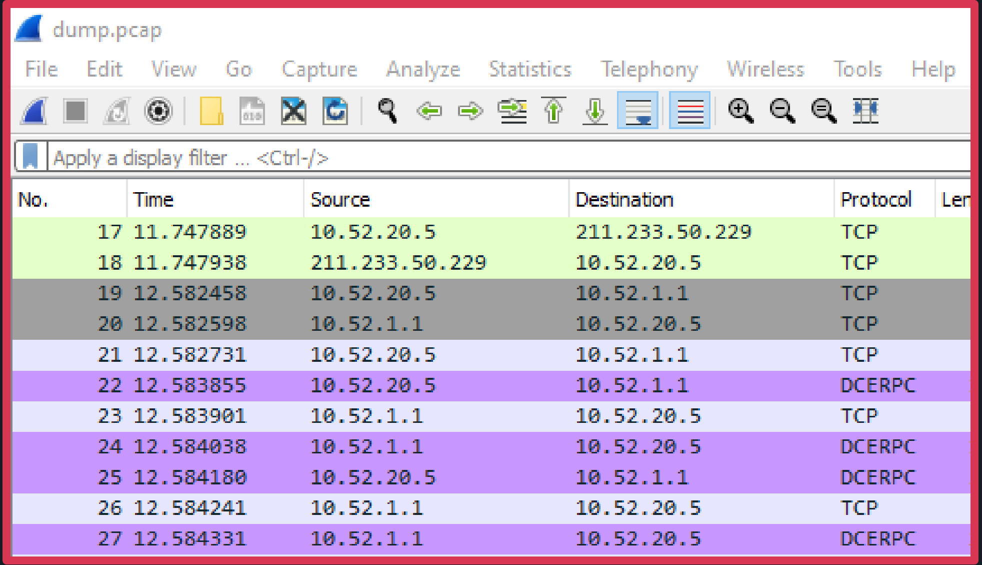
Process

| | |
|---|---|
| Command line: | "C:\Program Files\Dolby\Dolby DAX3\API\DAX3API.exe" |
| Current directory: | C:\Windows\system32\ |
| Started: | a day and 8 hours ago (8:57:18 AM 3/30/2021) |
| Parent console: | services.exe (660) Image type: 64-bit |
| Parent process: | services.exe (660) |
| Mitigation policies: | DEP (permanent); ASLR (high entropy) |
| Protection: | None |

- Process Hacker
  - Community-maintained `procmon` clone; actively maintained (v3 nightly branch)
  - Provides even more details for each processes where possible (e.g., detailed .NET assembly view, service management, looks nicer, etc.)

# Other Third-party Tools

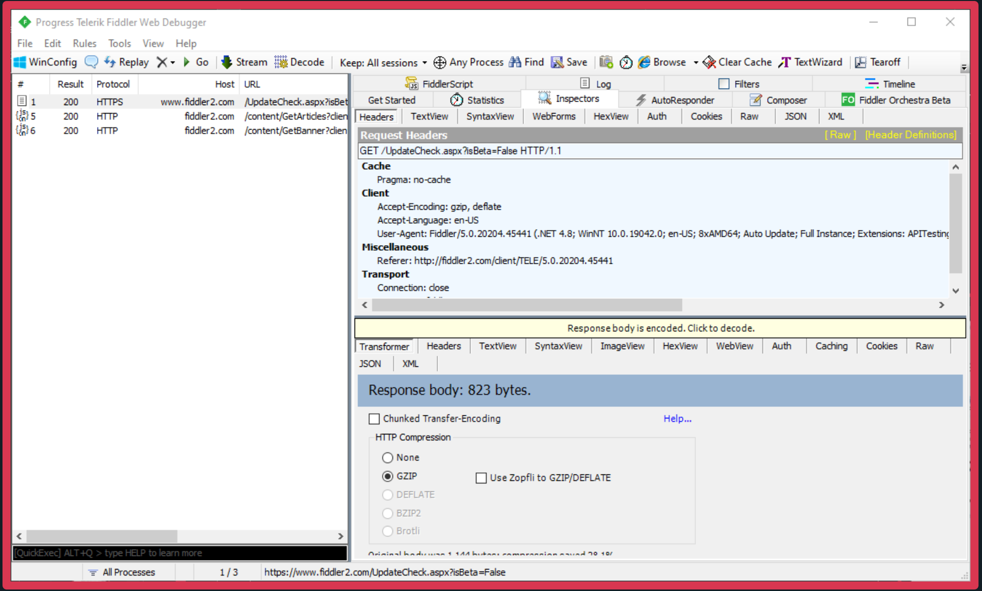
- Wireshark
  - **Wireshark** is the world's foremost and widely-used network protocol analyzer.
  - Provides an overview of the incoming and outgoing packets; useful for traffic analysis.


dump.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

| No. | Time | Source | Destination | Protocol | Len |
|---|---|---|---|---|---|
| 17 | 11.747889 | 10.52.20.5 | 211.233.50.229 | TCP | |
| 18 | 11.747938 | 211.233.50.229 | 10.52.20.5 | TCP | |
| 19 | 12.582458 | 10.52.20.5 | 10.52.1.1 | TCP | |
| 20 | 12.582598 | 10.52.1.1 | 10.52.20.5 | TCP | |
| 21 | 12.582731 | 10.52.20.5 | 10.52.1.1 | TCP | |
| 22 | 12.583855 | 10.52.20.5 | 10.52.1.1 | DCERPC | |
| 23 | 12.583901 | 10.52.1.1 | 10.52.20.5 | TCP | |
| 24 | 12.584038 | 10.52.1.1 | 10.52.20.5 | DCERPC | |
| 25 | 12.584180 | 10.52.20.5 | 10.52.1.1 | DCERPC | |
| 26 | 12.584241 | 10.52.1.1 | 10.52.20.5 | TCP | |
| 27 | 12.584331 | 10.52.1.1 | 10.52.20.5 | DCERPC | |

# Other Third-party Tools

- Fiddler
  - Proxy debugger for HTTP(s)-based traffic
  - Useful for dissecting HTTP(s)-based malware traffic
  - AutoResponder
    - Sends forged responses based on the incoming request


Progress Telerik Fiddler Web Debugger
File Edit Rules Tools View Help
WinConfig Replay Go Stream Decode Keep: All sessions Any Process Find Save Browse Clear Cache TextWizard Tearoff
FiddlerScript Log Filters Timeline
Get Started Statistics Inspectors AutoResponder Composer Fiddler Orchestra Beta
Headers TextView SyntaxView WebForms HexView Auth Cookies Raw JSON XML
Transformer Headers TextView SyntaxView ImageView HexView WebView Auth Caching Cookies Raw JSON XML
[QuickExec] ALT+Q > type HELP to learn more
All Processes 1 / 3

| # | Result | Protocol | Host | URL |
|---|---|---|---|---|
| 1 | 200 | HTTPS | www.fiddler2.com | /UpdateCheck.aspx?isBet |
| 5 | 200 | HTTP | fiddler2.com | /content/GetArticles?clien |
| 6 | 200 | HTTP | fiddler2.com | /content/GetBanner?clien |

```
Request Headers                                    [ Raw ]  [Header Definitions]
GET /UpdateCheck.aspx?isBeta=False HTTP/1.1
Cache
    Pragma: no-cache
Client
    Accept-Encoding: gzip, deflate
    Accept-Language: en-US
    User-Agent: Fiddler/5.0.20204.45441 (.NET 4.8; WinNT 10.0.19042.0; en-US; 8xAMD64; Auto Update; Full Instance; Extensions: APITesting
Miscellaneous
    Referer: http://fiddler2.com/client/TELE/5.0.20204.45441
Transport
    Connection: close
```

Response body is encoded. Click to decode.

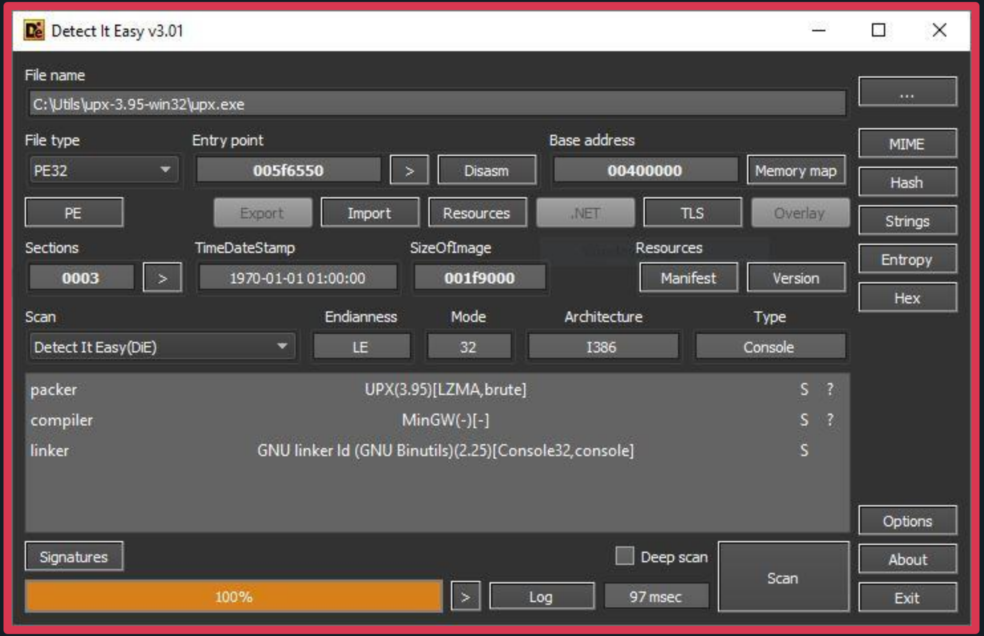
Response body: 823 bytes.

☐ Chunked Transfer-Encoding  Help...

HTTP Compression
- ○ None
- ● GZIP  ☐ Use Zopfli to GZIP/DEFLATE
- ○ DEFLATE
- ○ BZIP2
- ○ Brotli

https://www.fiddler2.com/UpdateCheck.aspx?isBeta=False

# Other Third-party Tools

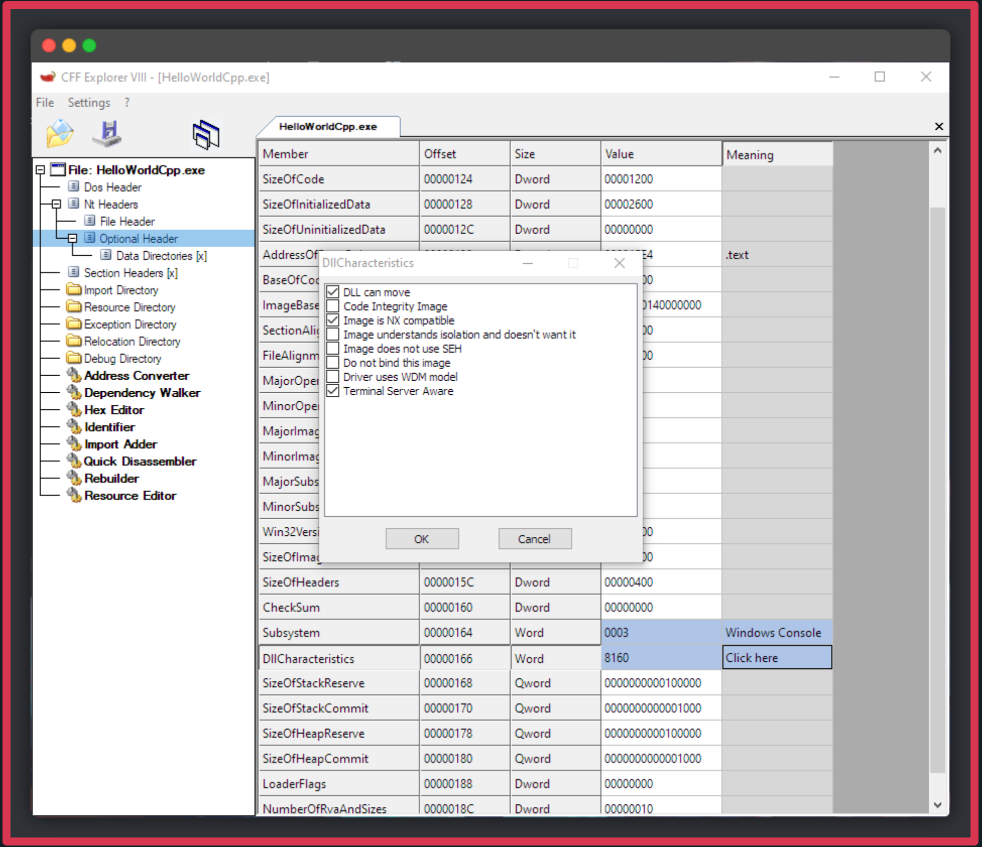


```
packer      UPX(3.95)[LZMA,brute]                                S  ?
compiler    MinGW(-)[-]                                          S  ?
linker      GNU linker ld (GNU Binutils)(2.25)[Console32,console] S
```

- Detect it Easy
  - Swiss-army knife of examining PEs
  - Quick overview of any specified file (incl. compiler, packer, linker, etc.) based on community-submitted signatures.
  - Examine import tables, exports, hashes of the file, strings, and more!

# Other Third-party Tools

- NTCore Explorer Suite - CFF Explorer
  - Another PE viewer
  - Header overview
    - Ability to make quick edits to the header
  - Dependency walker
  - Imports/exports view
  - and more!
  - R/W by default; easy ASLR toggle

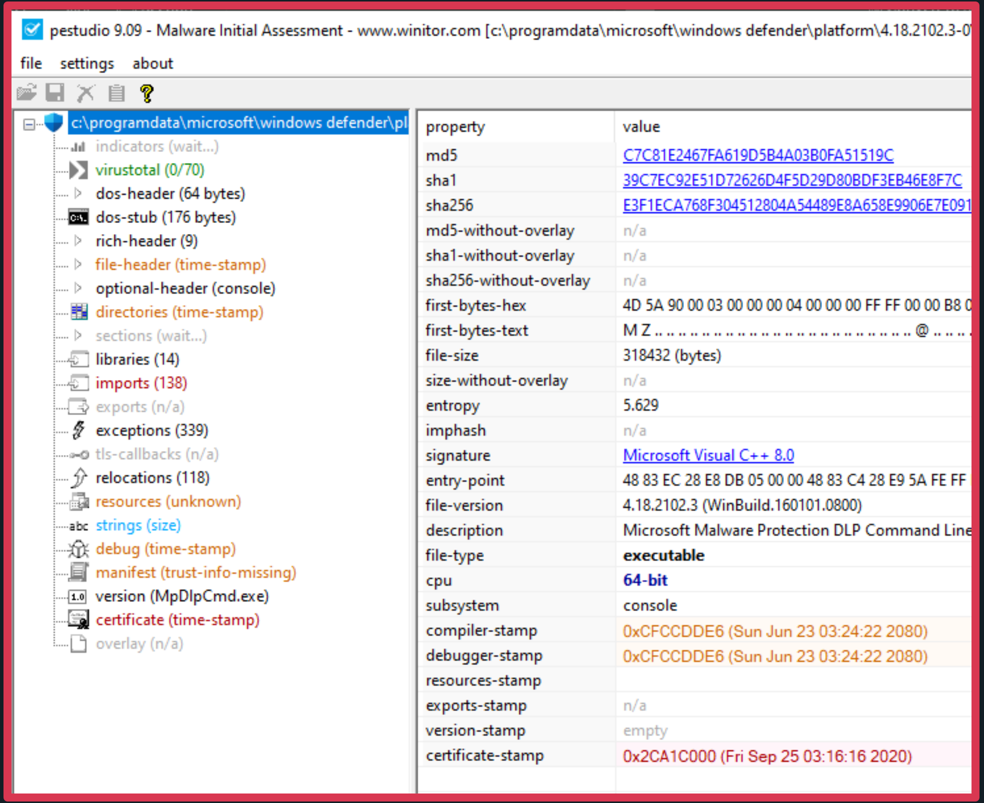


DllCharacteristics

- [✓] DLL can move
- [ ] Code Integrity Image
- [✓] Image is NX compatible
- [ ] Image understands isolation and doesn't want it
- [ ] Image does not use SEH
- [ ] Do not bind this image
- [ ] Driver uses WDM model
- [✓] Terminal Server Aware

| Member | Offset | Size | Value | Meaning |
|---|---|---|---|---|
| SizeOfCode | 00000124 | Dword | 00001200 | |
| SizeOfInitializedData | 00000128 | Dword | 00002600 | |
| SizeOfUninitializedData | 0000012C | Dword | 00000000 | |
| AddressOf | | | E4 | .text |
| BaseOfCo | | | 00 | |
| ImageBase | | | 0140000000 | |
| SectionAli | | | 00 | |
| FileAlignm | | | 00 | |
| MajorOper | | | | |
| MinorOper | | | | |
| MajorImag | | | | |
| MinorImag | | | | |
| MajorSubs | | | | |
| MinorSubs | | | | |
| Win32Versi | | | 00 | |
| SizeOfImag | | | 00 | |
| SizeOfHeaders | 0000015C | Dword | 00000400 | |
| CheckSum | 00000160 | Dword | 00000000 | |
| Subsystem | 00000164 | Word | 0003 | Windows Console |
| DllCharacteristics | 00000166 | Word | 8160 | Click here |
| SizeOfStackReserve | 00000168 | Qword | 0000000000100000 | |
| SizeOfStackCommit | 00000170 | Qword | 0000000000001000 | |
| SizeOfHeapReserve | 00000178 | Qword | 0000000000100000 | |
| SizeOfHeapCommit | 00000180 | Qword | 0000000000001000 | |
| LoaderFlags | 00000188 | Dword | 00000000 | |
| NumberOfRvaAndSizes | 0000018C | Dword | 00000010 | |

## Other Third-party Tools


pestudio 9.09 - Malware Initial Assessment - www.winitor.com [c:\programdata\microsoft\windows defender\platform\4.18.2102.3-0

```
c:\programdata\microsoft\windows defender\pl
  indicators (wait...)
  virustotal (0/70)
  dos-header (64 bytes)
  dos-stub (176 bytes)
  rich-header (9)
  file-header (time-stamp)
  optional-header (console)
  directories (time-stamp)
  sections (wait...)
  libraries (14)
  imports (138)
  exports (n/a)
  exceptions (339)
  tls-callbacks (n/a)
  relocations (118)
  resources (unknown)
  strings (size)
  debug (time-stamp)
  manifest (trust-info-missing)
  version (MpDlpCmd.exe)
  certificate (time-stamp)
  overlay (n/a)
```

| property | value |
|---|---|
| md5 | C7C81E2467FA619D5B4A03B0FA51519C |
| sha1 | 39C7EC92E51D72626D4F5D29D80BDF3EB46E8F7C |
| sha256 | E3F1ECA768F304512804A54489E8A658E9906E7E091 |
| md5-without-overlay | n/a |
| sha1-without-overlay | n/a |
| sha256-without-overlay | n/a |
| first-bytes-hex | 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 0 |
| first-bytes-text | M Z .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. .. @ .. .. .. . |
| file-size | 318432 (bytes) |
| size-without-overlay | n/a |
| entropy | 5.629 |
| imphash | n/a |
| signature | Microsoft Visual C++ 8.0 |
| entry-point | 48 83 EC 28 E8 DB 05 00 00 48 83 C4 28 E9 5A FE FF |
| file-version | 4.18.2102.3 (WinBuild.160101.0800) |
| description | Microsoft Malware Protection DLP Command Line |
| file-type | **executable** |
| cpu | **64-bit** |
| subsystem | console |
| compiler-stamp | 0xCFCCDDE6 (Sun Jun 23 03:24:22 2080) |
| debugger-stamp | 0xCFCCDDE6 (Sun Jun 23 03:24:22 2080) |
| resources-stamp | |
| exports-stamp | n/a |
| version-stamp | empty |
| certificate-stamp | 0x2CA1C000 (Fri Sep 25 03:16:16 2020) |

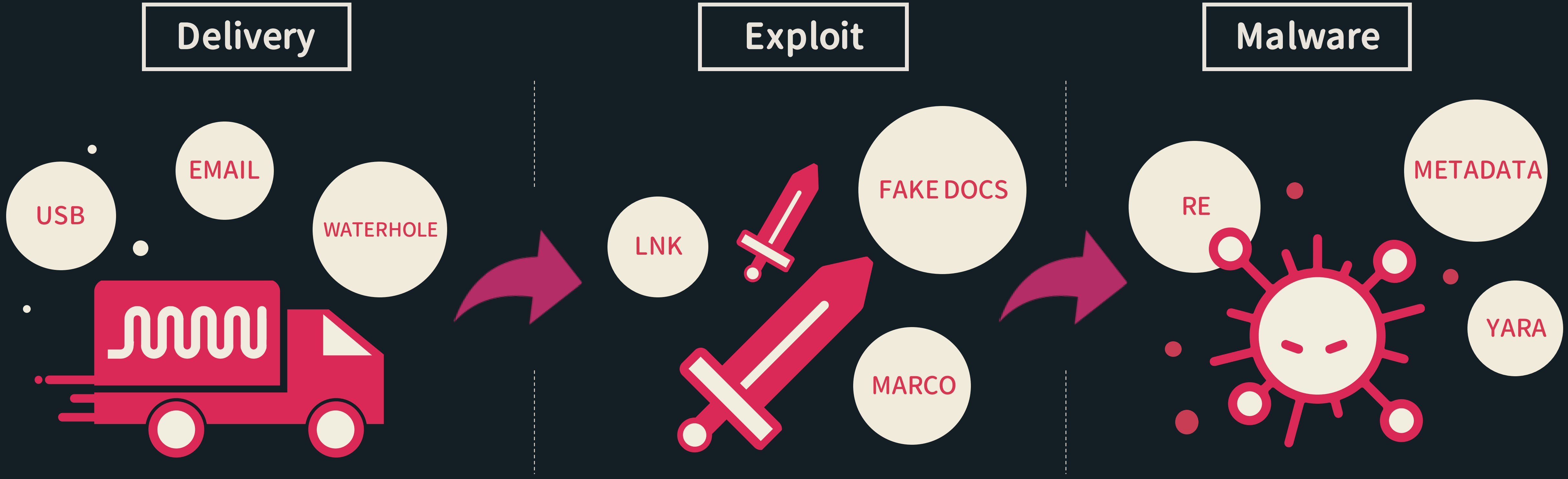
- pestudio
  - Yet another PE viewer
  - Useful for initial malware assessment
  - Provides a quick overview of…
    - File type
    - Target architecture
    - Hashes
    - Compiled date
    - DLL characteristics
    - Strings
    - Imports/exports
    - and more!

## Lab #2: Connecting the Dots

Try using these tools while installing your favorite software and see what happens!

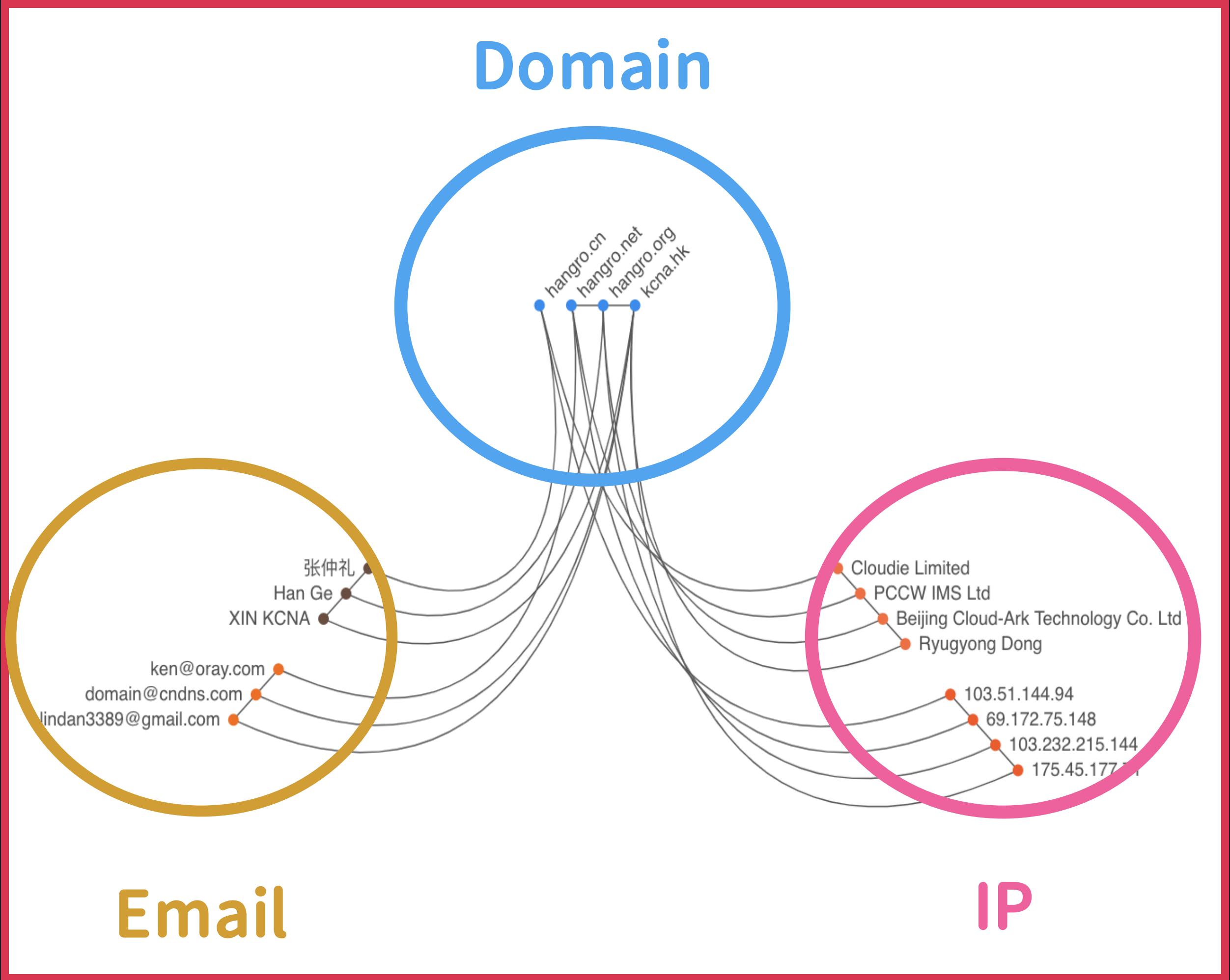
TEAMT5

# Capability Analysis



# Infrastructure Analysis

- Domain
  - WHOIS -> Email
  - Passive DNS -> IP
- IP
  - Passive DNS -> Domain
- Email
  - Reverse WHOIS - > Domain

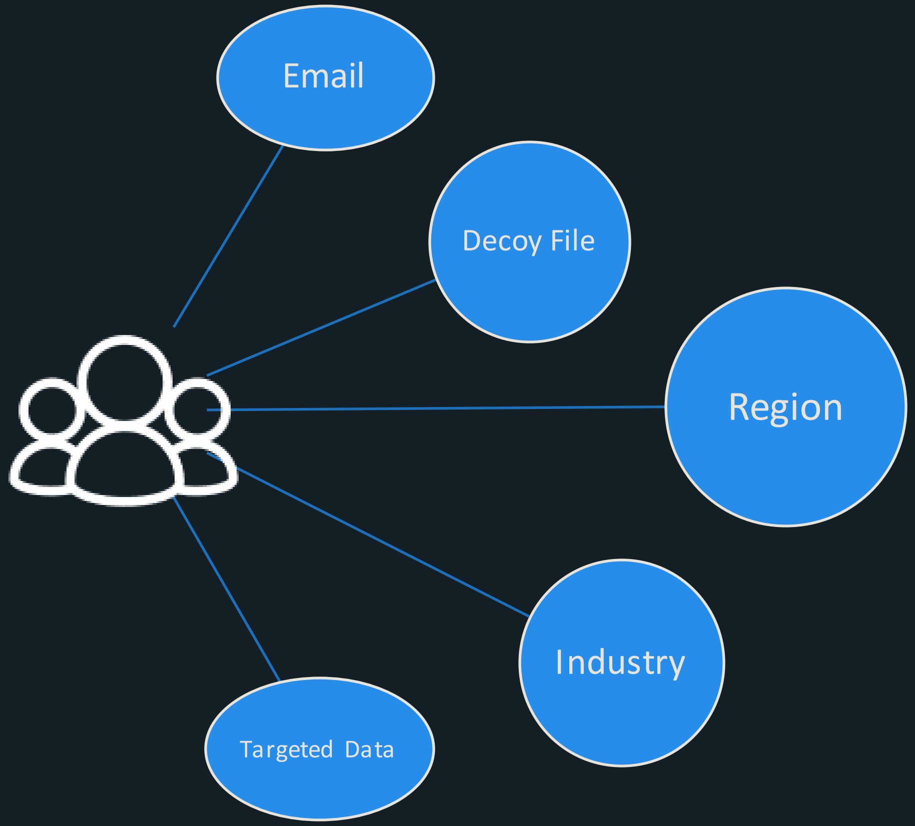


# Adversary Analysis

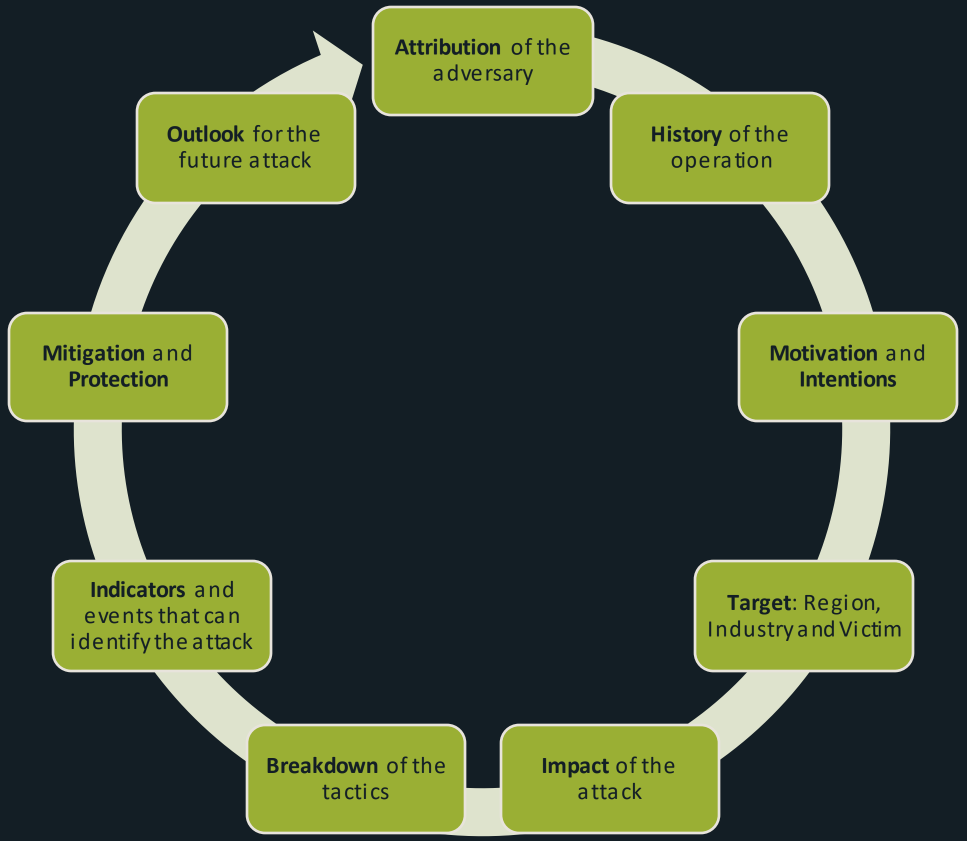
- Actors
  - Language
  - Tools
  - Infrastructure
  - Time zone
- Motivations, intentions
- Cooperation relationship between different groups
  - Shared Tool
  - Shared C2

# Target Analysis

## Victim Analysis



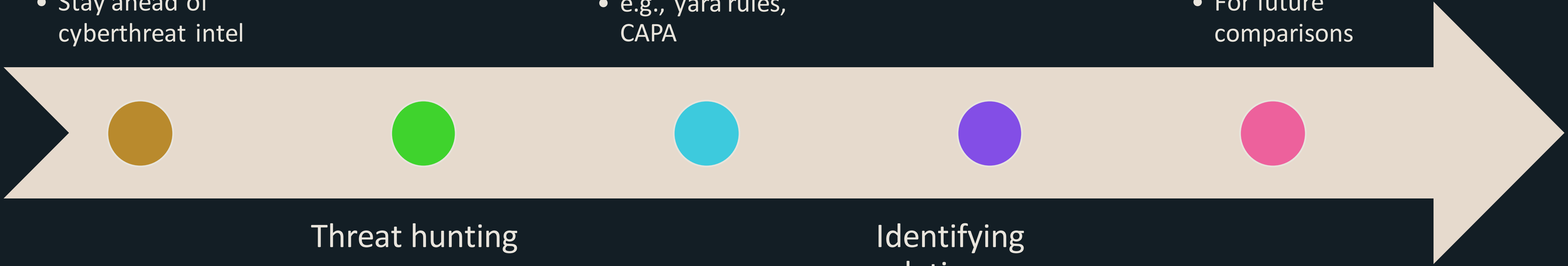
## Threat Analysis Report


**Attribution** of the adversary
**History** of the operation
**Motivation** and **Intentions**
**Target**: Region, Industry and Victim
**Impact** of the attack
**Breakdown** of the tactics
**Indicators** and events that can identify the attack
**Mitigation** and **Protection**
**Outlook** for the future attack

## Writing a CTI Report

TEAMT5

## Standard Operating Procedures



### Intel hunting

- Stay ahead of cyberthreat intel

### Threat hunting

- Collect valuable samples
- e.g., unseen C2 stations, zero-day, new backdoors

### Sample analysis

- Analyze behavior and if signatures exist
- e.g., yara rules, CAPA

### Identifying relations

- Compare with existing or known reports and identify whether a connection exists

### Report

- For future comparisons

## Content of a Report

TEAMT5

**How** did the incident occur?

- Delivery method(s)
- Phishing method(s)/theme(s)
- Exploitation method(s)

## Content of a Report

TEAMT5

**What** did it cause?

- Summary of the malicious behaviors
- IOC (Indicator of Compromise)

**How** did the incident occur?

- Delivery method(s)
- Phishing method(s)/theme(s)
- Exploitation method(s)

# Content of a Report

TEAMT5

**How** did the incident occur?

- Delivery method(s)
- Phishing method(s)/theme(s)
- Exploitation method(s)

**What** did it cause?

- Summary of the malicious behaviors
- IOC (Indicator of Compromise)

**Who** did it?

- Source infrastructure analysis
- Piece everything together with existing reports

## Exploit Methods

### Fake documents

- Executables or shortcuts (LNK) with document icons

### Malicious documents

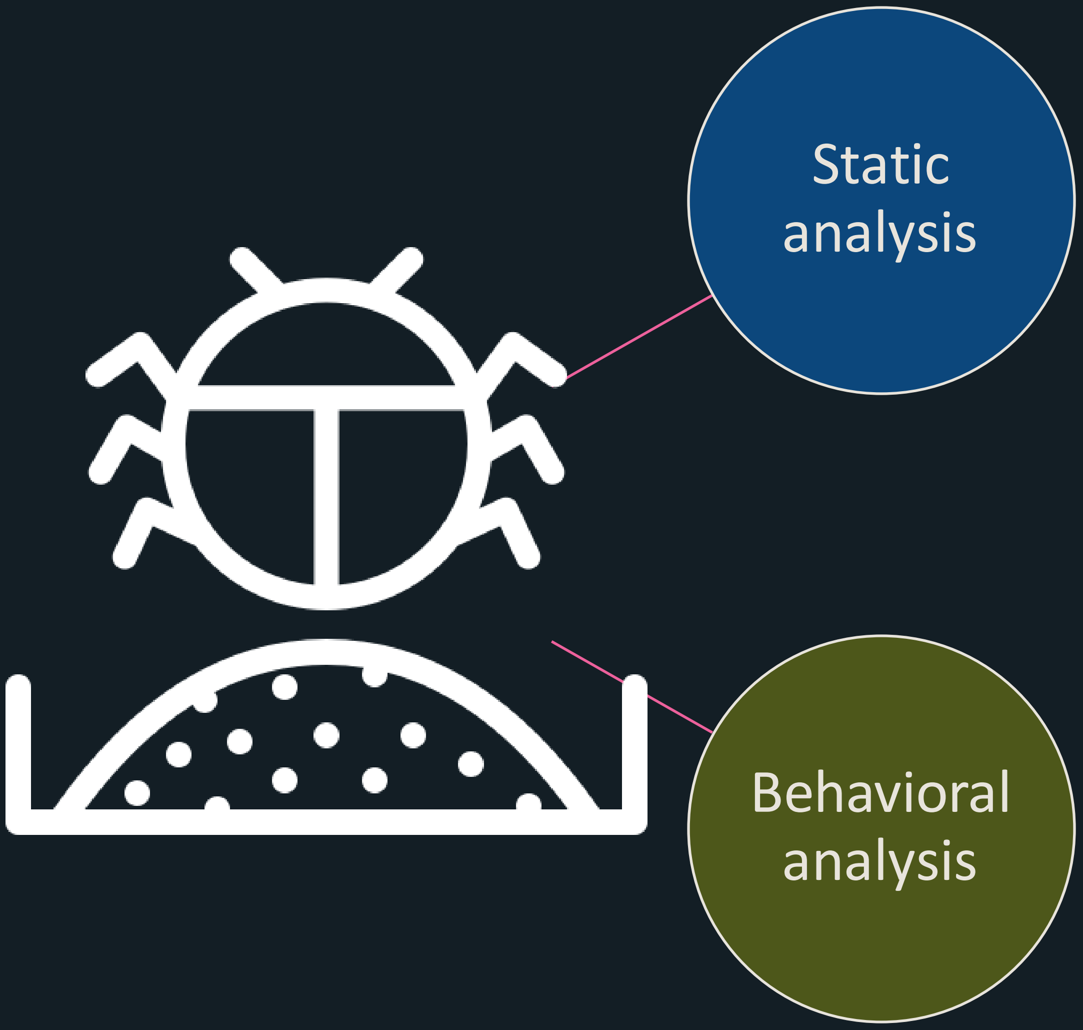
- Macro
- Object Linking and Embedding (OLE)
- Unpatched RCE CVEs (CVE-2018-0798)

### Software vulnerabilities

- CVE-2018-20250 (WinRAR ACE)
- CVE-2018-15982 (Flash Player use-after-free)
- Other CVEs or zero-days

## **What** did it cause?

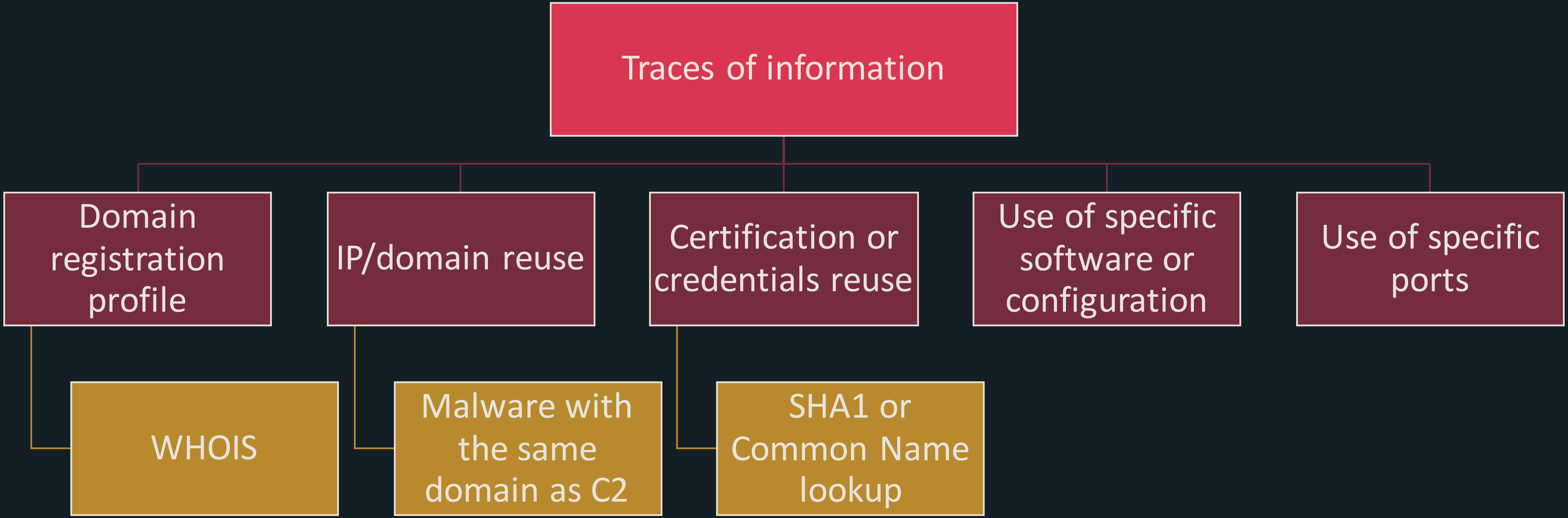
# Malware Analysis



- Disassembler
- Digital signature
  - Stolen cert
- PE header

- Artifacts
- Persistence
- Exfiltrated data
- Network traffic
- Encryption algorithm
- Backdoor

# C2 Relation


Traces of information
Domain registration profile
IP/domain reuse
Certification or credentials reuse
Use of specific software or configuration
Use of specific ports
WHOIS
Malware with the same domain as C2
SHA1 or Common Name lookup

# Compare Findings

- Collect OSINT resources
    - Other analysists’ view or thoughts
        - Twitter, Medium, blogs, etc.
    - Existing reports on the sample published by another security firm or researcher
        - FireEye, Kaspersky, CrowdStrike, Malwarebytes, etc.
- Personal or internal documents
    - Look for past records in the archive, if any
    - Cross-compare C2 used, behaviors exhibited, peculiar strings, etc.

## Lab #2: Connecting the Dots

TEAMT5

# Putting it Together: Example

- You stumbled upon a zipped sample `86950b81df2003d08ae4a7869ecf88fe` on an online sandbox platform.

https://bazaar.abuse.ch/sample/3c5d9ac0741850b5e6bf3af8c807b7ccfdb1bfc702cd75d8897a27b1387031c7/

## Lab #2

What behavior does the sample exhibit?

Is there any embedded data?

Tip: Try not to rely on sandbox reports; they can often be misleading or do not provide a bigger picture!

Don’t skip to the next page until you’ve found something!

TEAMT5

## Putting it Together: Example

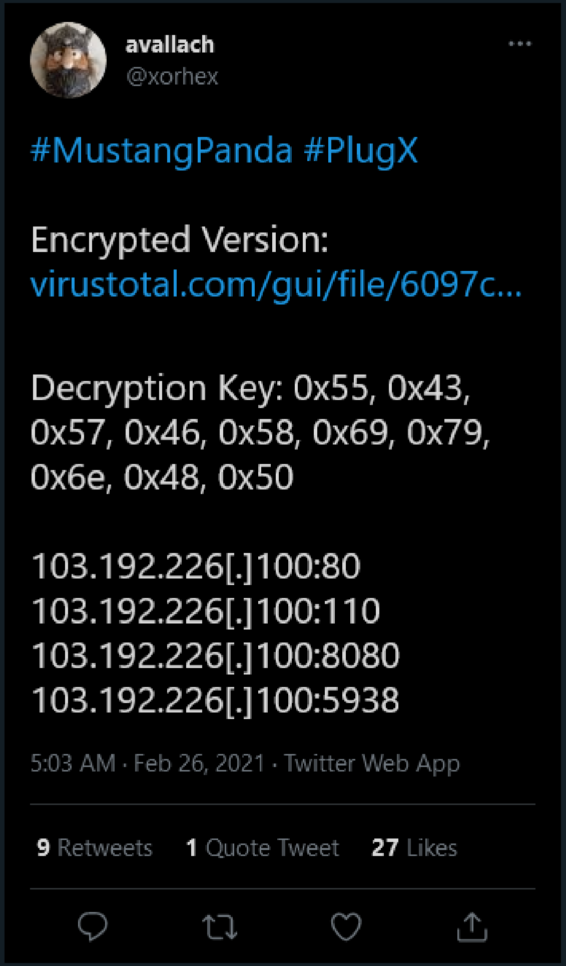
- After an extensive research, you've concluded the following characteristics from the sample,
  - Contacts `103.192.226.100`
  - Loads AvastAuth.dat and decodes it using XOR key "`DFtokTybRE`"
  - The decoded file is a PE file that was compiled on 2020-02-15 20:35:46
  - Contains an encoded configuration file using XOR key "`123456789`"
  - The config has a hardcoded name of "`AvastSvcyHA`"

## Lab #2

Given the clues thus far, what’s the next logical step?

Don’t skip to the next page until you’ve found something!

# Putting it Together: Example


avallach
@xorhex

#MustangPanda #PlugX

Encrypted Version:
virustotal.com/gui/file/6097c...

Decryption Key: 0x55, 0x43, 0x57, 0x46, 0x58, 0x69, 0x79, 0x6e, 0x48, 0x50

103.192.226[.]100:80
103.192.226[.]100:110
103.192.226[.]100:8080
103.192.226[.]100:5938

5:03 AM · Feb 26, 2021 · Twitter Web App

9 Retweets 1 Quote Tweet 27 Likes

*(Avallach (@xorhex) / Twitter, 2021)*

- By looking up the features of the sample, you've discovered that...
  - The IP address points to HK.
  - The IP address was recently documented on Twitter.
  - They referenced a group called `MustangPanda` and something called `PlugX`.

## Lab #2

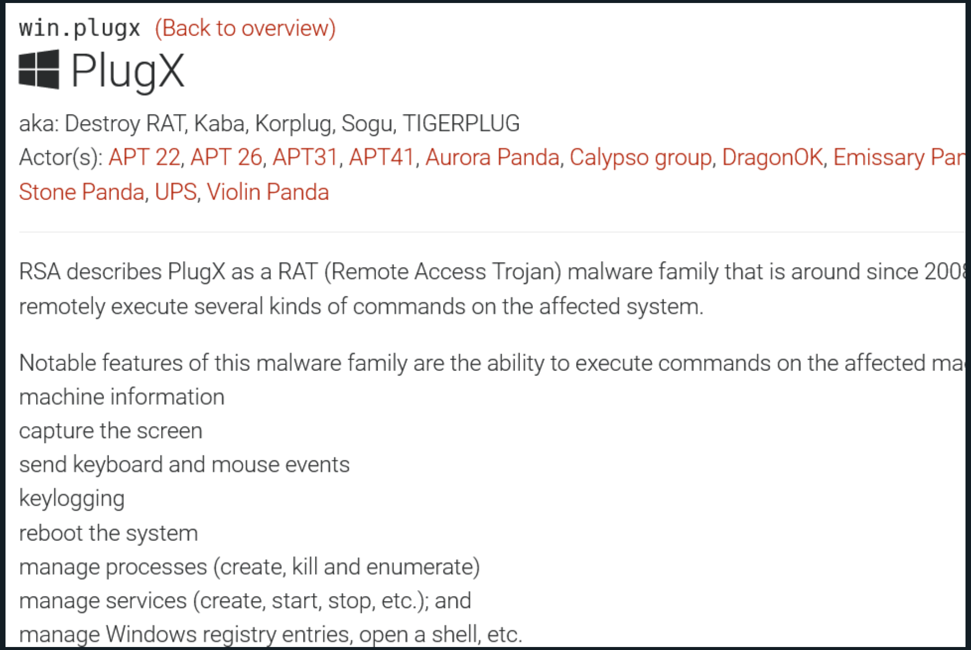
Who are MustangPanda and what is PlugX?

Don’t skip to the next page until you’ve found something!

TEAMT5

## Putting it Together: Example

- By looking up these two mysterious terms, you've discovered...
  - Malpedia is a malware/APT encyclopedia.
  - PlugX is a malware family, specifically, it is used as a RAT backdoor.
  - MustangPanda is a China-based APT group that targets Mongolians.



win.plugx (Back to overview)

PlugX

aka: Destroy RAT, Kaba, Korplug, Sogu, TIGERPLUG

Actor(s): APT 22, APT 26, APT31, APT41, Aurora Panda, Calypso group, DragonOK, Emissary Par
Stone Panda, UPS, Violin Panda

RSA describes PlugX as a RAT (Remote Access Trojan) malware family that is around since 200
remotely execute several kinds of commands on the affected system.

Notable features of this malware family are the ability to execute commands on the affected ma
machine information
capture the screen
send keyboard and mouse events
keylogging
reboot the system
manage processes (create, kill and enumerate)
manage services (create, start, stop, etc.); and
manage Windows registry entries, open a shell, etc.

## Putting it Together: Example

- Through nothing but **FREE** resources, you've learned that…
  - There is an APT group called MustangPanda in China.
    - Malpedia
  - Mongolians may be a target of interest for China.
    - Malpedia
  - PlugX is a RAT, and now you've learned what it may look like internally.
    - Through disassemblers and extensive debugging
  - PlugX may disguise itself as an anti-virus component.
    - Twitter

## Putting it Together: Next Step?

TEAMT5

- Write a YARA rule to threat hunt
  - VirusTotal (Paid)
  - Hybrid Analysis (Free)
  - Abuse.ch MalwareBazaar (Free)
- Publish your finding to help other researchers
  - …and that might help you land a job if you don’t have one already.
- Continue digging down the rabbit hole for other findings

## Intel Research

TEAMT5

## Threat/Intel Hunting Resources

TEAMT5

- Twitter
  - #APT
  - @cyberwar_15
  - @Timele9527
  - @blackorbird
  - @Rmy_Reserve
  - @_re_fox
- Curated Resources
  - https://start.me/p/rxRbpo/ti

## Threat/Intel Hunting Resources

TEAMT5

- Yara rules
  - Yara-Rules/rules @ GitHub
  - InQuest/awesome-yara @ GitHub
  - Neo23x0/signature-base @ GitHub
- CAPA
  - FireEye/CAPA @ GitHub

- Manual analysis
  - Behavior analysis via sandboxes
    - e.g., cuckoo, CAPEv2, etc.
  - Static analysis via disassemblers
    - e.g., IDA Pro, Ghidra, etc.
  - Dynamic analysis via contained environments
    - e.g., virtual machines, physical bare-bones

# Threat/Intel Hunting Resources

TEAMT5

- Open Sandbox Platforms
  - Any.Run
    - Requires registration
  - VirusTotal
    - Requires enterprise license to download sample
  - CAPEv2
  - Hybrid-Analysis
    - Requires approval by filling out the vetting form
- MITRE ATT&CK
- CTI news outlets/blogs
  - FireEye Threat Research Blog
  - JPCERT Blog
  - Kaspersky Lab Resource Center
  - Check Point Software Blog
  - …many more.

# Lab #3: Your First YARA Rule

(hopefully)

TEAMT5

# Lab #3: Your First YARA Rule

TEAMT5

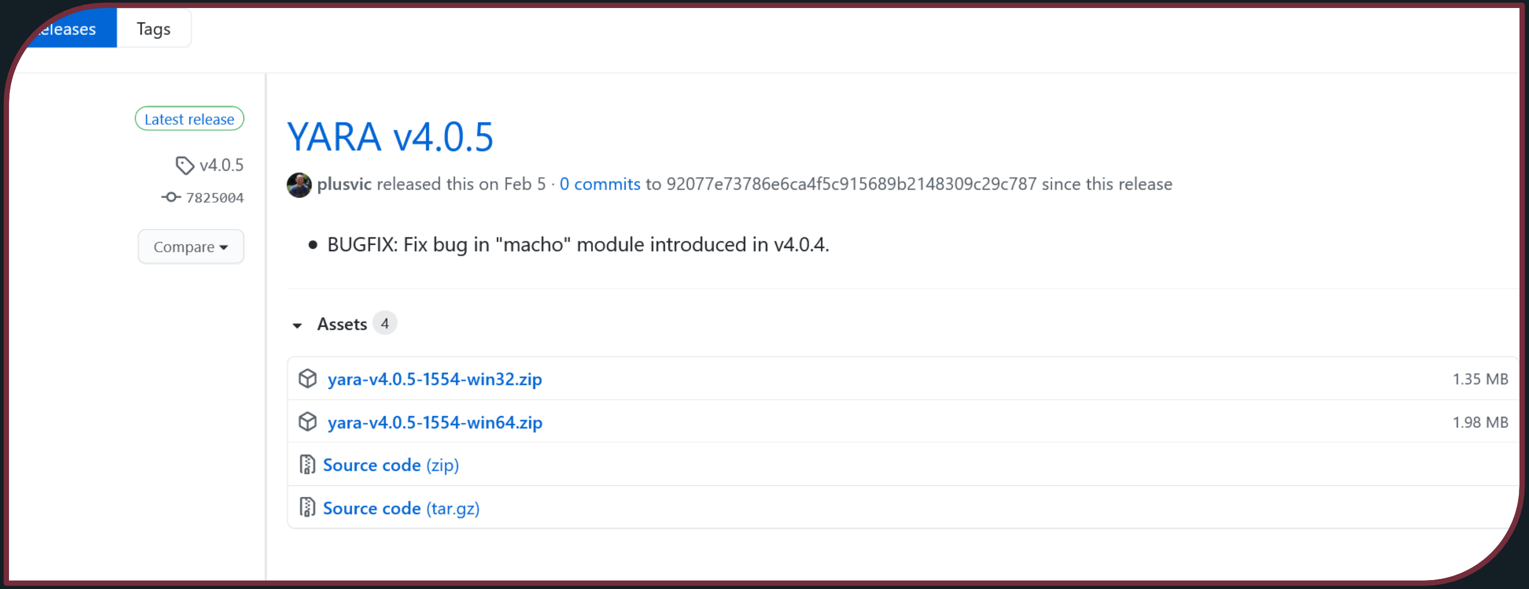
"YARA is a tool aimed at (but not limited to) helping malware researchers to identify and classify malware samples."

- (*VirusTotal/Yara*, 2012/2021)

```
rule REDLEAVES_DroppedFile_ObfuscatedShellcodeAndRAT_handkerchief
{
    meta:
        description = "Detect obfuscated .dat file containing shellcode and core
 REDLEAVES RAT"
        author = "USG"
        true_positive = "fb0c714cd2ebdcc6f33817abe7813c36" // handkerchief.dat
        reference = "https://www.us-cert.gov/ncas/alerts/TA17-117A"
    strings:
        $RedleavesStringObfu = {73 64 65 5e 60 74 75 74 6c 6f 60 6d 5e 6d 64 60
77 64 72 5e 65 6d 6d 6c 60 68 6f 2f 65 6d 6d} // This is 'red_autumnal_leaves_dl
lmain.dll' XOR'd with 0x01
    condition:
        any of them
}
```

## Lab #3: Your First YARA Rule

Install the latest YARA standalone scanner via `VirusTotal/Yara` @ GitHub


eleases Tags

Latest release
v4.0.5
7825004
Compare

YARA v4.0.5

plusvic released this on Feb 5 · 0 commits to 92077e73786e6ca4f5c915689b2148309c29c787 since this release

- BUGFIX: Fix bug in "macho" module introduced in v4.0.4.

Assets 4

| | |
|---|---|
| yara-v4.0.5-1554-win32.zip | 1.35 MB |
| yara-v4.0.5-1554-win64.zip | 1.98 MB |
| Source code (zip) | |
| Source code (tar.gz) | |

# Lab #3: Your First YARA Rule

- Let’s start with the syntax:
  - Similar to YAML
  - Similar to Python naming conventions
    `snake_case` for variables
  - Each rule begins with…
    `rule RuleName`
  - Each rule block requires at least one…
    `condition block`

```
rule My_First_Rule
{
    strings:
    condition:
}
```

# Lab #3: Your First YARA Rule

TEAMT5

- `strings` block
  - Each string is declared with the `$` prefix.
  - Case sensitive by default.
  - A simple string can be declared using a set of quotation marks.
    - e.g., `$my_variable = "asdf"`

```
rule My_First_Rule
{
    strings:
        $vegetal = "vegetal"
    condition:
}
```

# Lab #3: Your First YARA Rule

- strings block
  - A block of bytes can be declared using a set of braces.
    - e.g., $dead_beef = {DE AD BE EF}
  - Unknown bytes can be replaced with ??.
  - A known range of bytes can be replaced with [i] or [i-j].

```
rule My_First_Rule
{
    strings:
        $vegetal = "vegetal"
        $dead_beef = {DE A? ?? EF}
        $face_booc = {FA CE B0 0C}
        $dead_babe = {DE AD [1-9] BA BE}
    condition:
}
```

## Lab #3: Your First YARA Rule

TEAMT5

- strings block
  - A string can have additional modifiers:

    ascii (match ASCII chars; used with wide)

    fullword

    wide (UTF-16 chars)

    xor (search for strings with byte XOR applied)

    base64

    base64wide

    private (never match)

    nocase (case insensitive)

```
rule My_First_Rule
{
    strings:
        $vegetal = "vegetal"
        $utf16_beef = "beef" wide
        $cheese = "cheesecake" xor(0x01-0x05)
        $dead_beef = {DE A? ?? EF}
        $face_booc = {FA CE B0 0C}
        $dead_babe = {DE AD [1-9] BA BE}
    condition:
}
```

## Lab #3: Your First YARA Rule

TEAMT5

- `strings` block
  - Regex can also be used.
    - Perl-like syntax
  - e.g., `/hello{1,3}world/` matches "helloworld", "hellooworld", "hellooooworld".

```
rule My_First_Rule
{
    strings:
        $vegetal = "vegetal"
        $utf16_beef = "beef" wide
        $cheese = "cheesecake" xor(0x01-0x05)
        $dead_beef = {DE A? ?? EF}
        $face_booc = {FA CE B0 0C}
        $dead_babe = {DE AD [1-9] BA BE}
        $pastry = /slice of (cake|pie|bread)/ nocase
    condition:
}
```

## Lab #3: Your First YARA Rule

- condition block
  - Defines when the scanner should mark the target file as positive.
  - All defined strings MUST be referenced in this block.
  - Loosest condition is any of them, which returns true on any string match.

```
rule My_First_Rule
{
    strings:
        $vegetal = "vegetal"
        $utf16_beef = "beef" wide
        $cheese = "cheesecake" xor(0x01-0x05)
        $dead_beef = {DE A? ?? EF}
        $face_booc = {FA CE B0 0C}
        $dead_babe = {DE AD [1-9] BA BE}
        $pastry = /slice of (cake|pie|bread)/ nocase
    condition:
        any of them
}
```

## Lab #3: Your First YARA Rule

TEAMT5

- `condition` block
  - Conditions can be chained using `or`.
  - Conditions can be limited using `and`.

```
rule My_First_Rule
{
    strings:
        $vegetal = "vegetal"
        $utf16_beef = "beef" wide
        $cheese = "cheesecake" xor(0x01-0x05)
        $dead_beef = {DE A? ?? EF}
        $face_booc = {FA CE B0 0C}
        $dead_babe = {DE AD [1-9] BA BE}
        $pastry = /slice of (cake|pie|bread)/ nocase
    condition:
        ($dead_beef and $face_booc) or
            any of them
}
```

# Lab #3: Your First YARA Rule

- condition block
  - any can be substituted with any number of integer.
  - A set of strings with a common variable name can be referenced using wildcard with parentheses around the variable.
    - e.g., 3 of ($bad_*)

```
rule My_First_Rule
{
    strings:
        $evil_vegetal = "vegetal"
        $evil_pastry = /slice of (cake|pie|bread)/ nocase
        $bad_utf16_beef = "beef" wide
        $bad_cheese = "cheesecake" xor(0x01-0x05)
        $bad_dead_beef = {DE A? ?? EF}
        $face_booc = {FA CE B0 0C}
        $dead_babe = {DE AD [1-9] BA BE}
    condition:
        2 of ($bad_*) or
        any of ($evil_*) or
        ($face_booc and $dead_babe)
}
```

# Lab #3: Your First YARA Rule

- `condition` block
  - Many more conditions can be defined.
  - See YARA docs for a list of valid syntaxes.

```
rule Complex_Yara
{
    strings:
        $a = "Aaa"
        $b = "BbBb"
        $c = "ccc"
    condition:
        for any of ($a,$b,$c) : ( $ at pe.entry_point ) or
        for any section in pe.sections : ( section.name == ".text" )
}
```

Isn't this just the `strings` command with extra steps?

Yesn't

## Lab #2: Your First YARA Rule

- `strings` block
  - A block of bytes can be declared using a set of braces.
    - e.g., `$dead_beef = {DE AD BE EF}`
  - Unknown bytes can be replaced with `??`.
  - A known range of bytes can be replaced with `[i]` or `[i-j]`.

```
rule My_First_Rule
{
    strings:
        $vegetal = "vegetal"
        $dead_beef = {DE A? ?? EF}
        $face_booc = {FA CE B0 0C}
        $dead_babe = {DE AD [1-9] BA BE}
    condition:
}
```

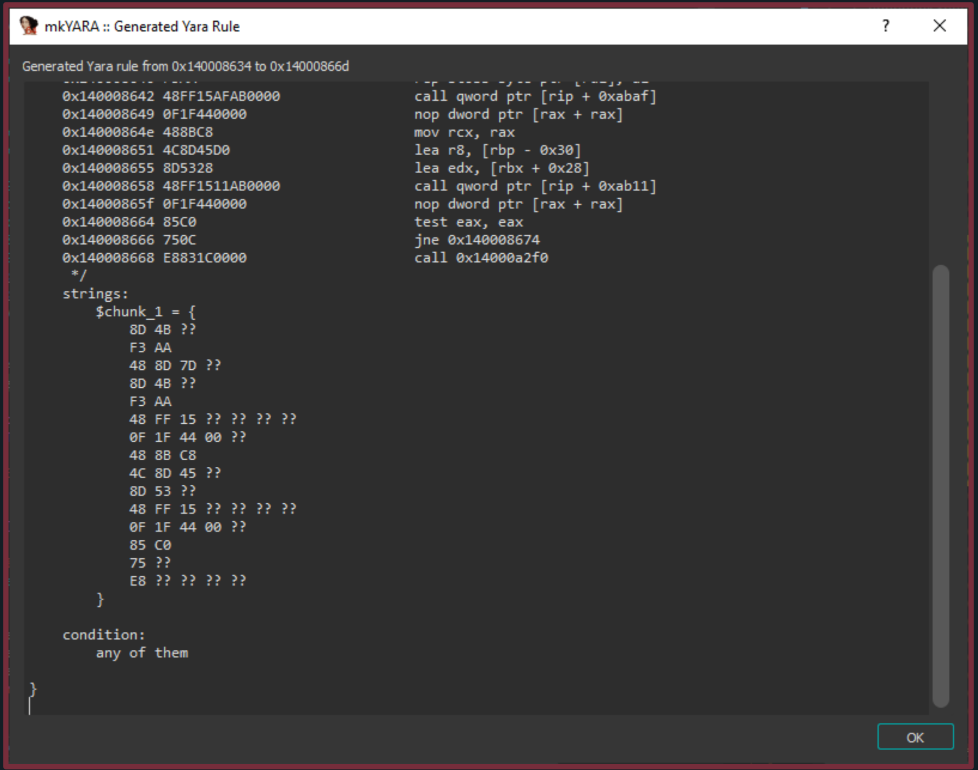
```
strings:
    $vegetal = "vegetal"
    $dead_beef = {DE A? ?? EF}
    $face_booc = {FA CE B0 0C}
    $dead_babe = {DE AD [1-9] BA BE}
```

```
000093d0  55                      push ebp
000093d1  8bec                    mov ebp, esp
000093d3  6aff                    push -1
000093d5  a1f8e50210              mov eax, dword ptr [0x1002e5f8]
000093da  50                      push eax
000093db  e8d0c4ffff              call 0x58b0
000093e0  85c0                    test eax, eax
000093e2  7407                    je 0x93eb
000093e4  b84f050000              mov eax, 0x54f
000093e9  eb21                    jmp 0x940c
000093eb  833df8e5021000          cmp dword ptr [0x1002e5f8], 0
000093f2  7416                    je 0x940a
000093f4  8b0df8e50210            mov ecx, dword ptr [0x1002e5f8]
000093fa  51                      push ecx
000093fb  e80070ffff              call 0x400
00009400  c705f8e5021000000000    mov dword ptr [0x1002e5f8], 0
0000940a  33c0                    xor eax, eax
0000940c  5d                      pop ebp
0000940d  c3                      ret
```

```
rule My_First_Rule
{
    strings:
        $vegetal = "vegetal"
        $dead_beef = {DE A? ?? EF}
        $face_booc = {FA CE B0 0C}
        $dead_babe = {DE AD [1-9] BA BE}
    condition:
}
```

# Lab #3: Your First YARA Rule

- Of course, you can write YARA rule to match for specific sets of instructions!
  - Instructions in a binary are just a series of bytes.
- IDA Pro plugins for writing YARA rules
  - `hyuunnn/Hyara` @ GitHub
  - `fox-it/mkYara` @ GitHub



Generated Yara rule from 0x140008634 to 0x14000866d

```
    0x140008642 48FF15AFAB0000                    call qword ptr [rip + 0xabaf]
    0x140008649 0F1F440000                        nop dword ptr [rax + rax]
    0x14000864e 488BC8                            mov rcx, rax
    0x140008651 4C8D45D0                          lea r8, [rbp - 0x30]
    0x140008655 8D5328                            lea edx, [rbx + 0x28]
    0x140008658 48FF1511AB0000                    call qword ptr [rip + 0xab11]
    0x14000865f 0F1F440000                        nop dword ptr [rax + rax]
    0x140008664 85C0                              test eax, eax
    0x140008666 750C                              jne 0x140008674
    0x140008668 E8831C0000                        call 0x14000a2f0
     */
    strings:
        $chunk_1 = {
            8D 4B ??
            F3 AA
            48 8D 7D ??
            8D 4B ??
            F3 AA
            48 FF 15 ?? ?? ?? ??
            0F 1F 44 00 ??
            48 8B C8
            4C 8D 45 ??
            8D 53 ??
            48 FF 15 ?? ?? ?? ??
            0F 1F 44 00 ??
            85 C0
            75 ??
            E8 ?? ?? ?? ??
        }

    condition:
        any of them

}
```

# Lab #3: Your First YARA Rule

## DO

- ✓ Target unique characteristics common in the same malware family
  - e.g., certain PDB paths or project folder names
- ✓ Compare code from the same malware family

## DO NOT

- × Rely on imports as an indicator
- × Match for common strings
- × Match for instructions that may be part of a library
  - e.g., OpenSSL, json-parser, etc.
- × Write YARA rules for .NET modules without using the .NET YARA module
  - Difficult & high false positive

## Lab #3: Your First YARA Rule

Now give it a try!

Try writing a YARA rule for FD866F6E1B997C31BDB6BA24361663E5.

TEAMT5

TEAMT5
杜浦數位安全

為您量身訂製　專屬勒索防護

立即前往 主題攤位 L04 量身

品牌專頁
QR Code

# THANK YOU!

@AzakaSekai_

still@teamt5.org

TEAMT5

Persistent Cyber Threat Hunters