# Zeus Banking Trojan: An In-Depth Examination

## Introduction

The **Zeus Banking Trojan**, commonly referred to as **Zbot**, is among the most infamous and widely deployed malware designed to extract sensitive financial information. It primarily focuses on stealing online banking credentials, credit card details, and other confidential data by logging keystrokes and injecting malicious scripts into authentic banking websites. First identified in **2007**, Zeus has been linked to numerous financial fraud cases worldwide, infecting millions of devices and causing financial losses in the billions. Over the years, it has undergone multiple transformations, incorporating advanced methods to evade detection and enhance data theft efficiency.

Zeus is mainly spread through phishing emails, drive-by downloads, and deceptive attachments disguised as legitimate software updates. Once a system is compromised, the malware operates discreetly in the background, logging user keystrokes, injecting fake login forms, and transmitting stolen credentials to cybercriminal-controlled servers. Despite continuous efforts by cybersecurity experts to disrupt Zeus operations, newer versions of the malware continue to emerge, maintaining its status as a significant cybersecurity risk.

## Evolution of Zeus Trojan

First discovered in **2007**, Zeus initially infected Windows-based systems by exploiting vulnerabilities to harvest banking details. Cybercriminals were able to acquire and customize the malware through underground forums, tailoring attacks to specific targets. Its modular structure allowed for easy modifications, leading to the development of multiple enhanced variants, including:

- **Gameover Zeus (2011):** A peer-to-peer version that removed centralized control, making it more challenging to dismantle.
- **Citadel (2012):** A refined version featuring improved security evasion mechanisms, better obfuscation techniques, and an intuitive botnet management interface.
- **Panda Banker (2016):** A contemporary variant aimed at financial institutions globally, leveraging web injection attacks and automated transaction fraud.

In **2011**, the original creator of Zeus, known by the alias **"Slavik"**, announced their retirement and released the malware's source code, which led to the proliferation of numerous derivatives. Cybercriminals quickly capitalized on this by developing new strains that remain a persistent threat. Despite law enforcement interventions, Zeus continues to evolve and pose a serious challenge in the cybersecurity landscape.

# Classification of Zeus Malware

Zeus is classified as a **Trojan malware**, specifically a **banking Trojan**, and exhibits the following characteristics:

- **Trojan Horse:** Masquerades as a legitimate file (e.g., invoice.pdf.exe) to deceive users into running it.
- **Keylogger:** Captures keystrokes to extract login credentials.
- **Form Grabbing:** Intercepts data entered into web forms before encryption is applied.
- **Man-in-the-Browser (MITB) Attack:** Modifies banking sessions by injecting malicious scripts to alter transactions.
- **Botnet Capabilities:** Enables attackers to remotely control infected systems.
- **Stealth Mechanisms:** Employs encryption, anti-sandboxing, and anti-debugging techniques to avoid detection.

# Zeus Trojan’s Operational Mechanism

Zeus functions through a multi-stage approach to infect systems and extract sensitive data:

## Step 1: Infection

The malware typically spreads through **phishing emails, malicious downloads, and exploit kits**. Users unknowingly execute the malware, believing it to be a legitimate program or update. Social engineering tactics are often employed to lure victims into clicking harmful links that initiate automatic downloads of the Zeus payload.

## Step 2: System Compromise

Once activated, Zeus:

- Alters **registry settings** to ensure persistence even after a reboot.
- Disables **antivirus programs** and firewalls to remain undetected.
- Embeds itself into system processes like explorer.exe or winlogon.exe.
- Establishes an encrypted communication channel with a **Command and Control (C2) server** for instructions and data transmission.

## Step 3: Credential Theft

- Functions as a **keylogger**, recording keystrokes when users enter login details.
- Utilizes **form grabbing** to capture credentials from banking portals before encryption occurs.
- Executes **MITB attacks**, modifying web pages in real-time to deceive users into revealing security codes or bypassing multi-factor authentication.
- Alters transactions by injecting malicious scripts into legitimate banking sessions.

## Step 4: Data Transmission and Fraud

- Stolen credentials are sent to a **C2 server**, where attackers exploit the information.
- Cybercriminals use these credentials for unauthorized fund transfers, creating fraudulent accounts, or selling them on dark web marketplaces.

- Some Zeus versions automate fraudulent transactions to evade traditional fraud detection mechanisms employed by banks.

# Notable Attacks and Their Impact

Zeus has been involved in several high-profile cyberattacks targeting banks, corporations, and individuals. Some major incidents include:

## 1. 2009: FBI's "Trident Breach" Operation

- Over **74,000 computers** were compromised, leading to financial losses exceeding **$70 million**.
- Several cybercriminals operating the Zeus botnet were arrested as a result of the investigation.

## 2. 2012: Targeting of British Banks

- Zeus launched sophisticated attacks against **UK banking institutions**, redirecting online transactions to fraudulent accounts.
- Exploited authentication system weaknesses to bypass security measures.

## 3. 2014: Gameover Zeus Botnet Takedown

- This advanced **peer-to-peer variant** infected over **500,000 systems** worldwide.
- A collaborative effort by U.S. and European law enforcement successfully disrupted the botnet's activities.

## 4. 2019: Panda Banker Resurgence

- A new Zeus variant, **Panda Banker**, resurfaced, targeting financial organizations in **North America, Europe, and Asia**.
- Used **malicious JavaScript injections** to extract credit card details and login credentials, impacting thousands of users.

## 5. 2021: Zeus-Inspired Attacks on Cryptocurrency Wallets

- Cybercriminals adapted Zeus tactics to compromise cryptocurrency exchanges and digital wallets.

- The malware altered transaction details in real-time, diverting crypto assets to attacker-controlled addresses.
- Enhanced obfuscation techniques made detection and removal significantly more difficult

# Steps to Download

## Step 1: Download the Malware

- Source: Download the malware from GitHub.
- Specific URL: Navigate to **https://github.com/ytisf/theZoo/tree/master/malware/Binaries/ZeusBankingVersion_26Nov2013.**
- File: Download the raw zip file (preferred for analysis).

## Step 2: Browser Recommendation

- Use Microsoft Edge to download the file, as Google Chrome may block it.

## Important Note: Safety Precautions

- Isolate malware samples from any network connection.
- Use a controlled environment (e.g., virtual machine or sandbox) for handling.

## Warning Message

- "Do not download or execute malware samples on your main machine. Always use a virtual environment or sandbox to contain the malware and prevent accidental infection."

## Step 3: Unzip the File

- After downloading, unzip the downloaded zip file.
- Password for unzipping: “infected”.

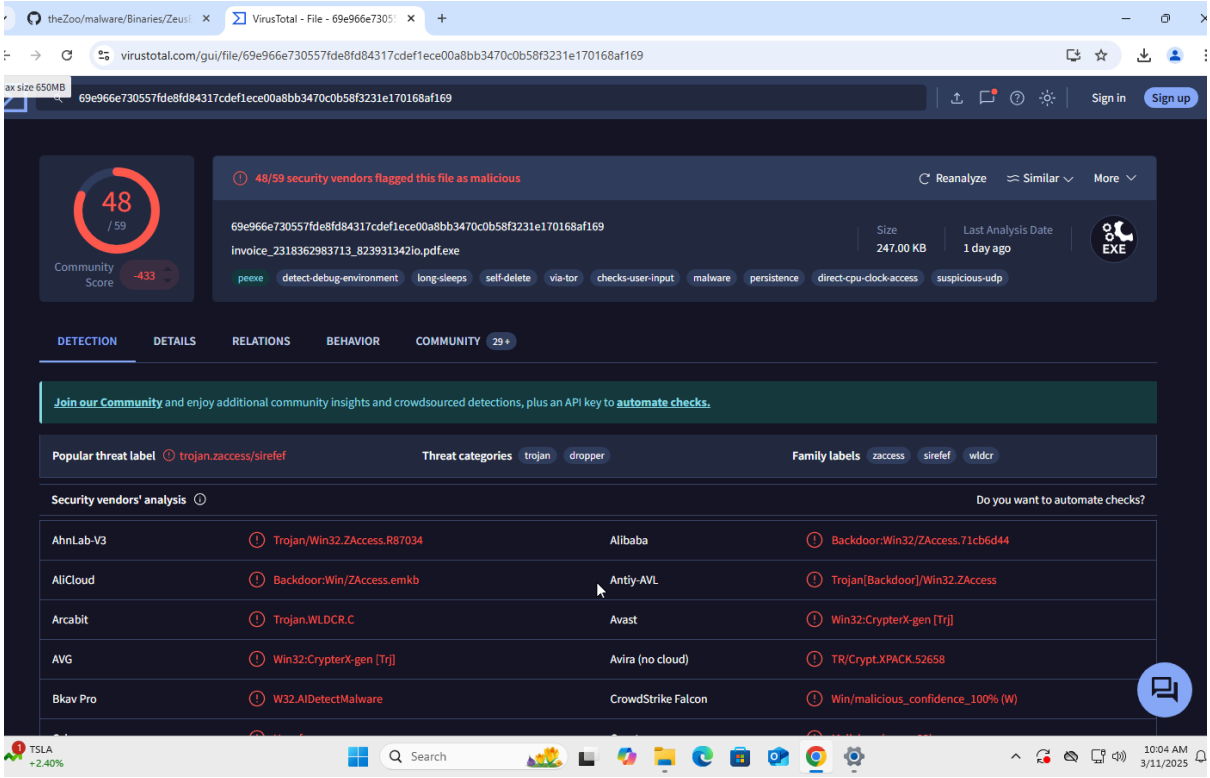
## Step 4: Upload to Sandbox

- Upload the unzipped malware file into a sandbox for analysis.

# Fingerprint

1 . Upload the file “**invoice_2318362983713_823931342io.pdf. exe**” to virus total .

2 . After getting the result we can see that 48/59 security vendors flagged this file as **malicious** .




3 . Here we can see the hashes of the file .

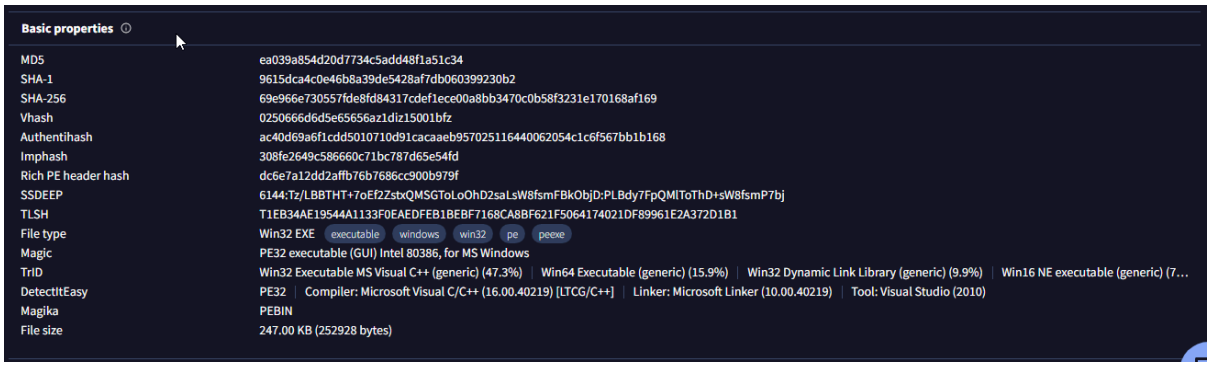
**Hashes :**

**SHA256 :** 69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169

**SHA1 :** 9615dca4c0e46b8a39de5428af7db060399230b2

**MD5 :** ea039a854d20d7734c5add48f1a51c34

**Filename :** invoice_2318362983713_823931342io.pdf.exe



**Basic properties**

| | |
|---|---|
| MD5 | ea039a854d20d7734c5add48f1a51c34 |
| SHA-1 | 9615dca4c0e46b8a39de5428af7db060399230b2 |
| SHA-256 | 69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169 |
| Vhash | 0250666d6d5e65656az1diz15001bfz |
| Authentihash | ac40d69a6f1cdd5010710d91cacaaeb95702511644006205 4c1c6f567bb1b168 |
| Imphash | 308fe2649c586660c71bc787d65e54fd |
| Rich PE header hash | dc6e7a12dd2affb76b7686cc900b979f |
| SSDEEP | 6144:Tz/LBBTHT+7oEf2ZstxQMSGToLoOhD2saLsW8fsmFBkObjD:PLBdy7FpQMlToThD+sW8fsmP7bj |
| TLSH | T1EB34AE19544A1133F0EAEDFEB1BEBF7168CA8BF621F5064174021DF89961E2A372D1B1 |
| File type | Win32 EXE executable windows win32 pe peexe |
| Magic | PE32 executable (GUI) Intel 80386, for MS Windows |
| TrID | Win32 Executable MS Visual C++ (generic) (47.3%) \| Win64 Executable (generic) (15.9%) \| Win32 Dynamic Link Library (generic) (9.9%) \| Win16 NE executable (generic) (7... |
| DetectItEasy | PE32 \| Compiler: Microsoft Visual C/C++ (16.00.40219) [LTCG/C++] \| Linker: Microsoft Linker (10.00.40219) \| Tool: Visual Studio (2010) |
| Magika | PEBIN |
| File size | 247.00 KB (252928 bytes) |

# Static Analysis

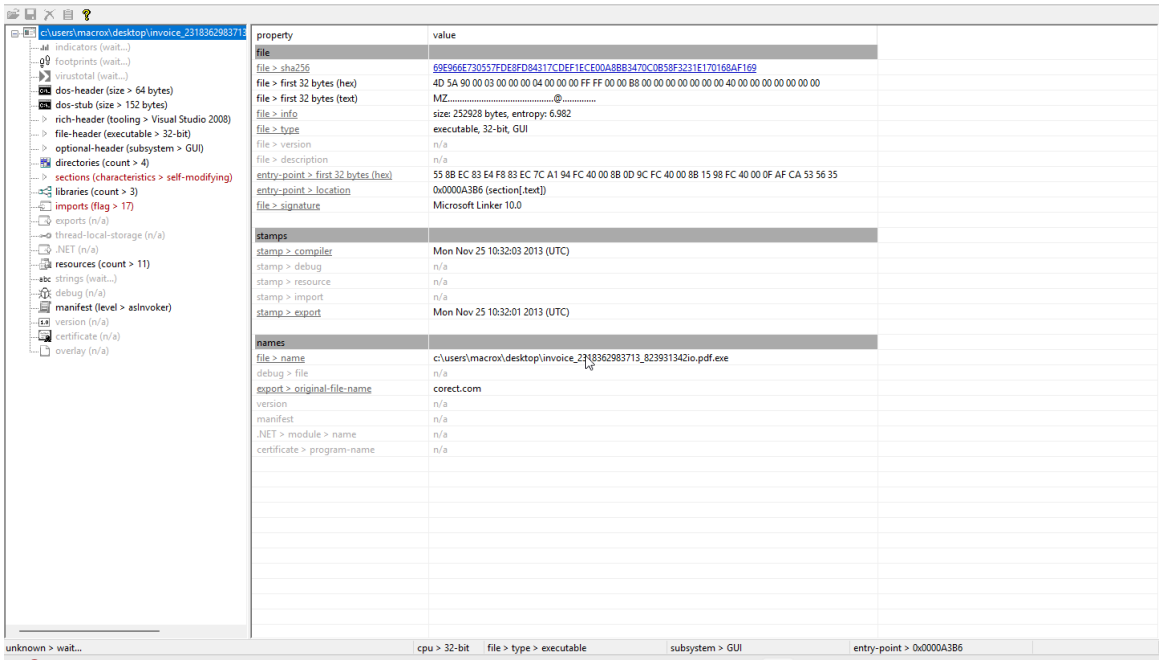
1 . Open the file “**invoice_2318362983713_823931342io.pdf.exe**” in peStudio.

## Overview

- **pestudio** is a static malware analysis tool.
- **Purpose**: Examines Windows executables (PE files) without executing them.

## Key Features :

- Detects suspicious indicators:
  - Malware behavior.
  - Anti-analysis tricks.
- Extracts components of a PE file:
  - Imports.
  - Exports.
  - Resources.
  - Sections.
- Scans the file against VirusTotal to identify known threats.
- Identifies:
  - Strings.
  - Libraries.
  - API calls used within the malware.
- Supports malware triage without requiring execution of the file.



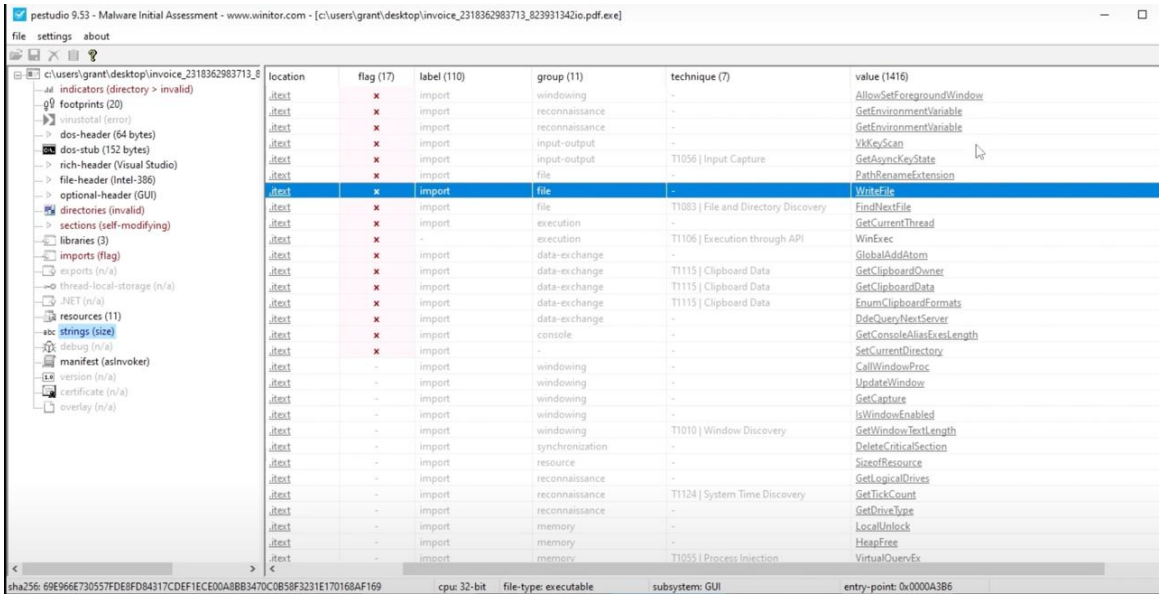

2 . Go to the sections(self-modifying) . Here we can see the size of the file .

3 . By observing the raw-size and virtual-size , we can that this program is unpacked

| property | value |
|---|---|
| section | section[0] |
| name | .text |
| section > sha256 | 8309B5D320B3D392E25AFD5... |
| entropy | 6.707 |
| file > ratio (99.60%) | 18.42 % |
| raw-address (begin) | 0x00000400 |
| raw-address (end) | 0x0000BA00 |
| raw-size (251904 bytes) | 0x0000B600 (46592 bytes) |
| virtual-address | 0x00001000 |
| virtual-size (250379 bytes) | 0x0000B571 (46449 bytes) |

4 . Now go to strings section .

5 . Here we can see all the strings . Here we can observe some function of the program ,which may be useful and tells us about the program




6 . After scrolling down , here we can see some dlls .

Here we can observe , before the dll their is something randomly is written . May be this might be the function names , which are obfuscated .


7 . Here we can see a magic number MZ , which means it is an windows executable file .

| property | value |
| --- | --- |
| sha256 | 69E966E730557FDE8FD84317CDEF1ECE00A8BB3470C0B58F3231E170168AF169 |
| sha1 | 9615DCA4C0E46B8A39DE5428AF7DB060399230B2 |
| md5 | EA039A854D20D7734C5ADD48F1A51C34 |
| 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 00 00 |
| first-bytes-text | M Z . . . . . @ . . . . . |

8 . Now go to libraries option , here we can see 3 libraries .

| library (3) | duplicate (0) | flag (0) | bound (0) | first-thunk-original (INT) | first-thunk (IAT) | type (1) | imports (77) | group | description |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SHLWAPI.dll | - | - | - | 0x00020208 | 0x00020078 | implicit | 21 | - | Shell Light-weight Utility Library |
| KERNEL32.dll | - | - | - | 0x00020190 | 0x00020000 | implicit | 29 | - | Windows NT BASE API Client |
| USER32.dll | - | - | - | 0x00020260 | 0x000200D0 | implicit | 27 | - | Multi-User Windows USER API Client Library |

9 . Next we will use tool **"Floss"** .

**FLOSS (FireEye Labs Obfuscated String Solver)** is a powerful static analysis tool used to automatically extract and decode obfuscated strings from malware binaries. It helps malware analysts and reverse engineers understand the behavior of malicious programs without executing them.

**Key Features:**

- Extracts plaintext & obfuscated strings (URLs, commands, API calls).
- Identifies custom string decryption functions.
- Automates string analysis to speed up malware research.

10 . Now lets go to **Cmder** , and go to the directory in which the

"**invoice_2318362983713_823931342io.pdf.exe**" file is stored .

11 . Now run the command and lets store the output in strings.txt file :

**floss invoice_2318362983713_823931342io.pdf. exe > strings.txt**

12 . Now lets open the strings.txt file

```
strings.txt - Notepad
File  Edit  Format  View  Help

FLARE FLOSS RESULTS (version v2.3.0-0-g037fc4b)

+------------------------+-------------------------------------------------------------------------------+
| file path              | invoice_2318362983713_823931342io.pdf.exe                                     |
| extracted strings      |                                                                               |
|  static strings        | 791                                                                           |
|  stack strings         | 2                                                                             |
|  tight strings         | 0                                                                             |
|  decoded strings       | 66                                                                            |
+------------------------+-------------------------------------------------------------------------------+

 ____________________
  FLOSS STATIC STRINGS
 ____________________

+-----------------------------------+
| FLOSS STATIC STRINGS: ASCII (790) |
+-----------------------------------+

!This program cannot be run in DOS mode.
Rich
.text
`.data
.itext
.pdata
.rsrc
@.reloc
jrjy
SVW;
9ATt
SVW5
Km}0+
=#NEw1zUTMNNONvJ2NRo+OjJzndo4djQeYoJlUoqiQeJ3gyK8RpqRQ9GyWDDzetolmnR7tLSM7SL
cT6b
```

13 . Now lets look is their anything with **“.com”** , we can find “**corect.com**”

15 . Now open Windows PowerShell .

16 . Now lets use **capa**

capa is a static malware analysis tool developed by MANDIANT that helps identify capabilities in executable files. It is particularly useful for analyzing malware and reverse-engineered binaries.

**Key Features of capa:**

•**Identifies Capabilities:** Detects functionalities like file manipulation, process injection, network communication, and more.

•**Static Analysis:** Analyzes binaries without executing them, making it safer than dynamic analysis.

•**Rule-Based Detection:** Uses YARA-like rules to detect common malware behaviors.

•**Supports Multiple Formats:** Works with PE (Windows executables), ELF (Linux executables), and shellcode.

•**Integrates with IDA Pro, Ghidra, and Radare2**: Helps in reverse engineering workflows.

17 .  Enter this command :

**capa .\invoice_2318362983713_823931342io.pdf.exe**

```
Select Windows PowerShell
PS C:\Users\Grant\Desktop> capa .\invoice_2318362983713_823931342io.pdf.exe
matching: 100%|
+------------------------+------------------------------------------------------------------------------------+
| md5                    | ea039a854d20d7734c5add48f1a51c34                                                   |
| sha1                   | 9615dca4c0e46b8a39de5428af7db060399230b2                                           |
| sha256                 | 69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169                   |
| os                     | windows                                                                            |
| format                 | pe                                                                                 |
| arch                   | i386                                                                               |
| path                   | invoice_2318362983713_823931342io.pdf.exe                                          |
+------------------------+------------------------------------------------------------------------------------+

+------------------------+------------------------------------------------------------------------------------+
| ATT&CK Tactic          | ATT&CK Technique                                                                   |
|------------------------+------------------------------------------------------------------------------------|
| DEFENSE EVASION        | Virtualization/Sandbox Evasion::System Checks T1497.001                            |
+------------------------+------------------------------------------------------------------------------------+

+----------------------------+-----------------------------------------------------------------------------+
| MBC Objective              | MBC Behavior                                                                |
|----------------------------+-----------------------------------------------------------------------------|
| ANTI-BEHAVIORAL ANALYSIS   | Virtual Machine Detection [B0009]                                           |
+----------------------------+-----------------------------------------------------------------------------+

+-------------------------------------------------+-------------------------------------------------+
| CAPABILITY                                      | NAMESPACE                                       |
|-------------------------------------------------+-------------------------------------------------|
| reference anti-VM strings targeting VMWare      | anti-analysis/anti-vm/vm-detection              |
| contain a resource (.rsrc) section              | executable/pe/section/rsrc                      |
| resolve function by parsing PE exports          | load-code/pe                                    |
+-------------------------------------------------+-------------------------------------------------+

PS C:\Users\Grant\Desktop>
```

18 . Now for more information Enter this command :

**capa -v .\invoice_2318362983713_823931342io.pdf.exe**

```
Windows PowerShell
PS C:\Users\Grant\Desktop> capa -v .\invoice_2318362983713_823931342io.pdf.exe
matching: 100%|
md5                     ea039a854d20d7734c5add48f1a51c34
sha1                    9615dca4c0e46b8a39de5428af7db060399230b2
sha256                  69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169
path                    invoice_2318362983713_823931342io.pdf.exe
timestamp               2023-08-09 16:55:01.606232
capa version            5.1.0
os                      windows
format                  pe
arch                    i386
extractor               VivisectFeatureExtractor
base address            0x400000
rules                   C:\Users\Grant\AppData\Local\Temp\_MEI79962\rules
function count          80
library function count  1
total feature count     9498

reference anti-VM strings targeting VMWare
namespace  anti-analysis/anti-vm/vm-detection
scope      file

contain a resource (.rsrc) section
namespace  executable/pe/section/rsrc
scope      file

resolve function by parsing PE exports
namespace  load-code/pe
scope      function
matches    0x40A3B6
```

19 . U can use this command for more verbose information :

**capa -vv .\invoice_2318362983713_823931342io.pdf.exe**

**20 . This are the API Calls we got .**

**API Calls :**

AllowSetForegroundWindow

GetEnvironmentVariable

GetEnvironmentVariable

VkKeyScan

GetAsyncKeyState

PathRenameExtension

WriteFile

FindNextFile

GetCurrentThread

WinExec

GlobalAddAtom

GetClipboardOwner

GetClipboardData

EnumClipboardFormats

DdeQueryNextServer

GetConsoleAliasExesLength

SetCurrentDirectory.

CallWindowProc

UpdateWindow

GetCapture

IsWindowEnabled

GetWindowTextLength

DeleteCriticalSection

SizeofResource

GetLogicalDrives

System Time

GetTickCount

GetDriveType

LocalUnlock

HeapFree

VirtualQueryEx

LocalAlloc LocalFree

CopyAcceleratorTable

SwapMouseButton PathQuoteSpaces

PathCombine

GetCompressedFileSize

CreateFileMapping

GetPrivateProfileInt

FreeLibrary

GetModuleHandle

**Suspected Function Calls :**

AsksmaceaglyBubuPulsKaifTeasMistPeelGhisPrimChaoLyreroeno

KERNEL32.MulDiv

BagsSpicDollBikeAzonPoopHamsPyasmap

KERNEL32.SetCurrentDirectory

BardHolyawe

SHLWAPI.SHFreeShared

BathEftsDawnvilepughThroCymakohloverMitefuzerat

SHLWAPI.PathMakeSystemFolder

BemaCadsPodsWavyCedeRadsbrioOustPerefenom

USER32.SetDIgItemText

BullbonyaweeWaitsnug TierDriblibye

KERNEL32.VirtualQuery

CameValeWauler

USER32. Islonic

CedeSalsshulLimy ThroliraValeDonabox

USER32.CreateCaret

CellrotoCrudUntohighCols

KERNEL32.CreateFile

DenyLubeDunssawsOresvarut

SHLWAPI.PathRemoveFileSpec

DragRoutflusCrowPeatmownNewsyaksSerfmare

USER32.DestroyIcon

Dumpcotsavo

USER32.SetDIgItemInt

DungBadebankBangGelthoboCocaBozotsksWheyVaryShoghoseNipsCadisi

USER32.EndPaint

ExitRollWoodGumsgamaSloerevsWussletssinkYearZitiryesHypout

USER32.GetClassInfo

FociTalcileador

KERNEL32.ConvertDefaultLocale

GeneAilshe

KERNEL32.FindFirstFile

GhisGoodHowlCoonCigscateged

KERNEL32.GetWindowsDirectory

GimpWadsdashHoraYardSeatDeanScanscowRantKeasfib

KERNEL32.LCMapString

Haesourte

USER32.GetKeyNameText

HoggSoonLasstwaeNapeCeilBawlscopdub

KERNEL32.SystemTimeToFileTime

Icontellnoway

SHLWAPI.PathRemoveBlanks

ImidslatJokyCombdrubChefBilkSale

USER32.GetShellWindow

IzararfsFlamWostAirsconsMouefemelallPoretweeSacsOxidMinx

SHLWAPI.PathAddExtension

JabsNaveFateLariManyLeeksecshiesBawlwoo

KERNEL32.CreateloCompletionPort

KatsDoreOmerBetsKoraKeef

KERNEL32.GetShortPathName

KineChamLows

KERNEL32.SetCurrentDirectory

LeerMiff

KERNEL32.LeaveCriticalSection

MaarSectFiscNextMattbamsErasnimstoeaBadshon

USER32.GetClassInfo

MarkMokeOsesShwaSkegpornlimemim

KERNEL32.GetStartupInfo

MeanOrrabirogirtWorkGawpSassPirnVinoLotaPledEidefe

SHLWAPI.SHLockShared

NextLoveOralwanySurfhm

KERNEL32.VerSetConditionMask

NisiBoyolineJiaoveryObiaowedblamHaetMaulweensky,

SHLWAPI.PathCanonicalize

OastcabskamiKartDumbInksSomsMass

KERNEL32.SetCurrentDirectory

PeckQuinFillrillsaw

KERNEL32.GetThreadPriority

RamilimaputtHastJobs

KERNEL32.FindNextFile

RemsSlaySoreAnoaaxalbuffusesemeuMapsvogaHangLoud/

SHLWAPI.PathMakePretty

RidsFineZingMickMomsdue

USER32.GetMonitorInfo

SeminerdsoloseenYaginobox

SHLWAPI.PathIsLFNFileSpec

SiretomsbritGrewIckyNapaLumsBoaren

KERNEL32.0penFileMapping

SlabKitsSlayseptPfftjiffSabsdeskOafsNowtMemsKirnKepiMiffDunt

KERNEL32.0penSemaphore

SoldKartAgueiliaRushWauldhal

SHLWAPI.PathIsUNC

SuitplieGunsMaidBaitFeus/iaotodycol/AlbsLuneToyspe

USER32.GetProp

SungActaKopsMaarposyparefuzedeck

SHLWAPI.PathIsDirectory

ToeaTailecusGeesSoliCadeSpueEndsPlaykaphall

SHLWAPI.PathRemoveArgs

Vavsrubepodsjadebrooli

USER32.GetUpdateRgn

VeerCrawFlateel

SHLWAPI.PathParseIconLocation

WainMeekPinyWonkpooflaudsir

KERNEL32.GetWindowsDirectory

WhopTestrangrapsdebsTzarNipaYins

KERNEL32.DeleteFile

YeukMags

KERNEL32.GlobalHandle

ZetaBeduPirnhipsjailTingSrisTeleAposhuskNameHoerflagemuwo

USER32.LoadIcon

# Dynamic Analysis

1. Run the Process Monitor -**Sysinternals**

**Process Monitor (ProcMon)** is a real-time system monitoring tool from Microsoft that tracks registry, file system, process, and network activity on Windows.

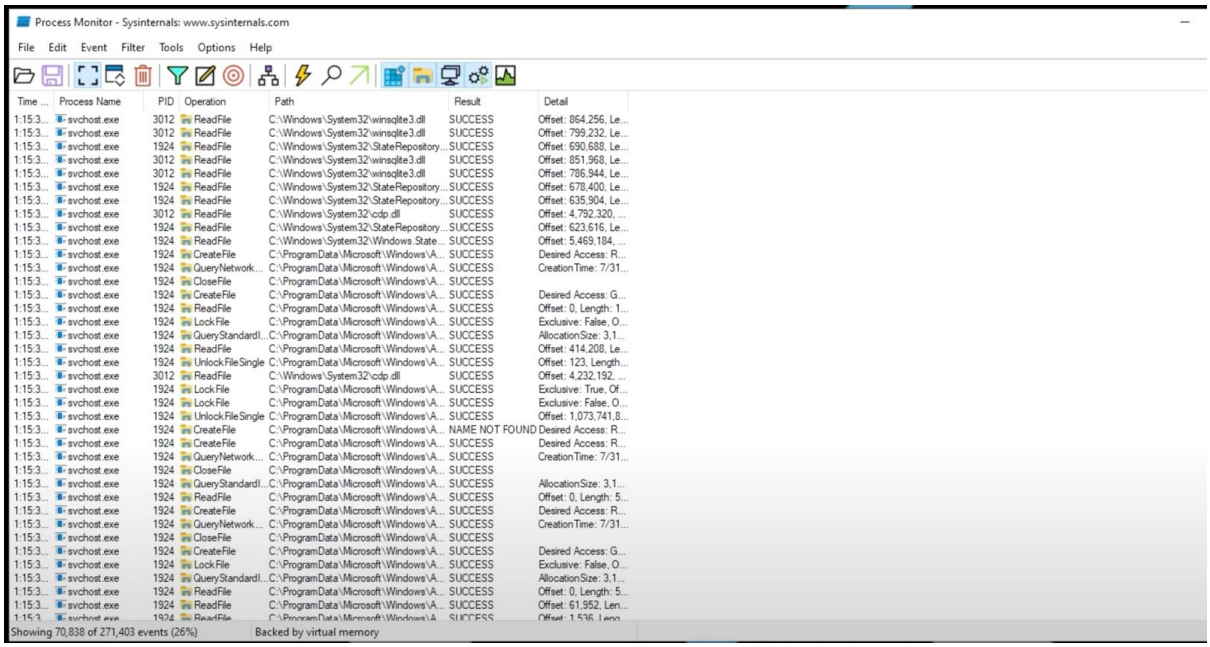
**Key Features:**

Monitors real-time process execution (PID, command-line, paths).

Tracks registry changes, file system activity, and network connections.

Detects malware behaviors (persistence mechanisms, DLL injection, etc.).

Advanced filtering and logging for deep forensic analysis.




2 . Now open run **REMnux ,** and configure **INetSim** with command :

**Command : inetsim**

**REMnux** is a Linux distribution designed for malware analysis and reverse engineering. It provides a collection of tools for static, dynamic, and memory analysis of malicious software.

**Key Features**:

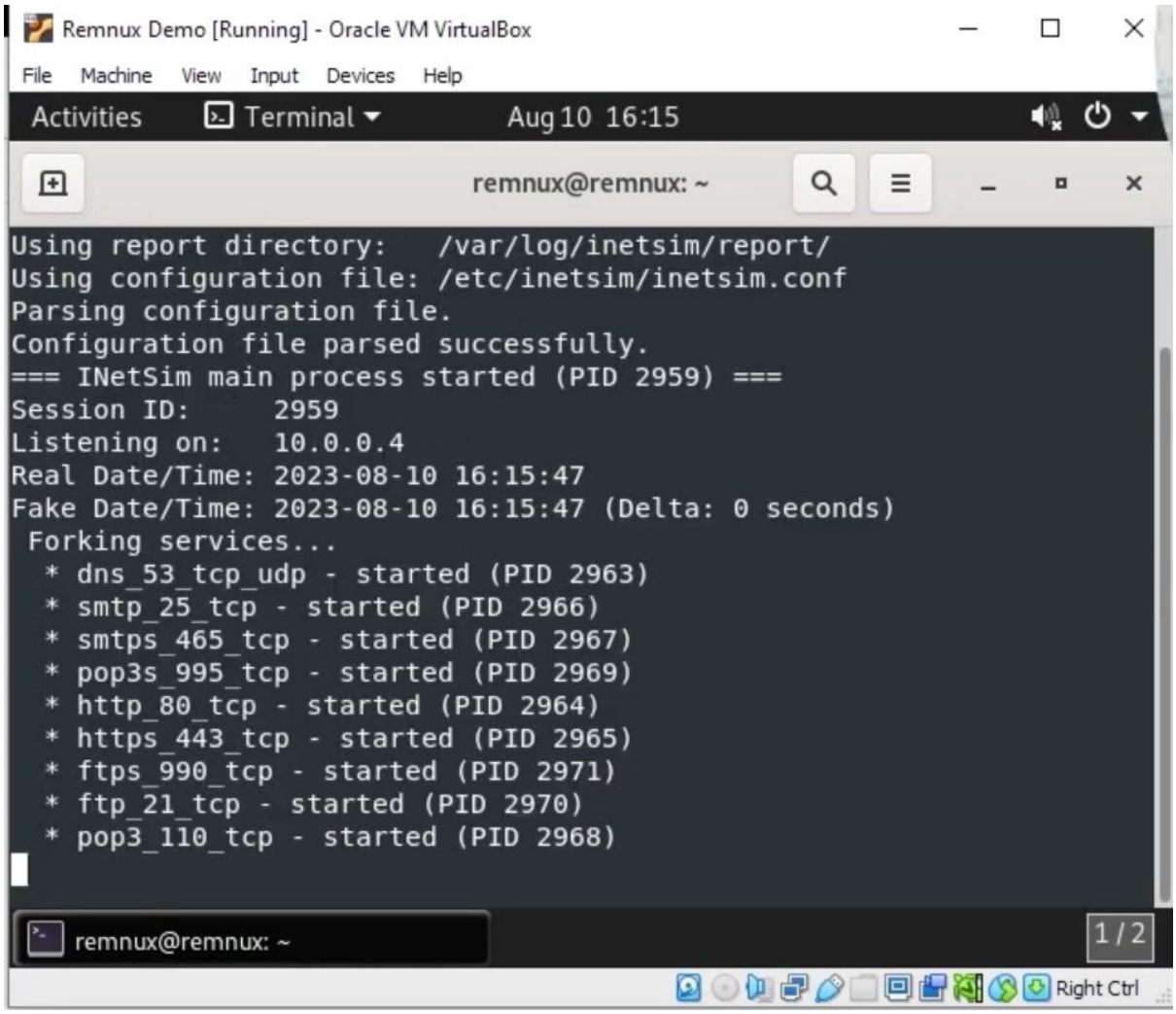
Pre-installed tools for static analysis (capa, FLOSS, pestudio).

Dynamic analysis tools (Cuckoo Sandbox, Process Monitor, Wireshark).

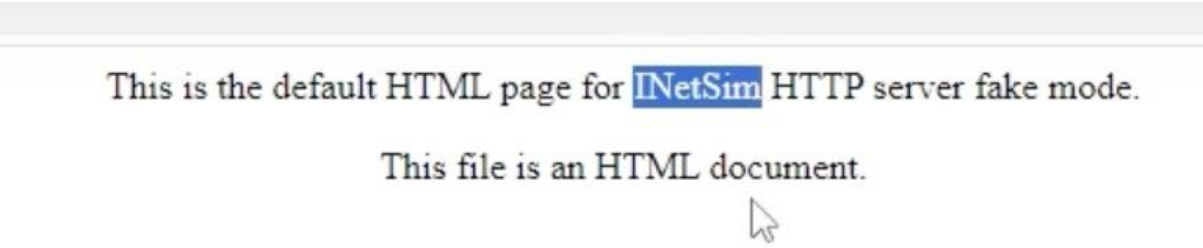
Memory forensics (Volatility, Rekall).

Network analysis (Suricata, Zeek, INetSim for malware sandboxing).

Lightweight and works as a VM or Docker container




3 . To conform that **INetSim** is working , go to browser and search google.com



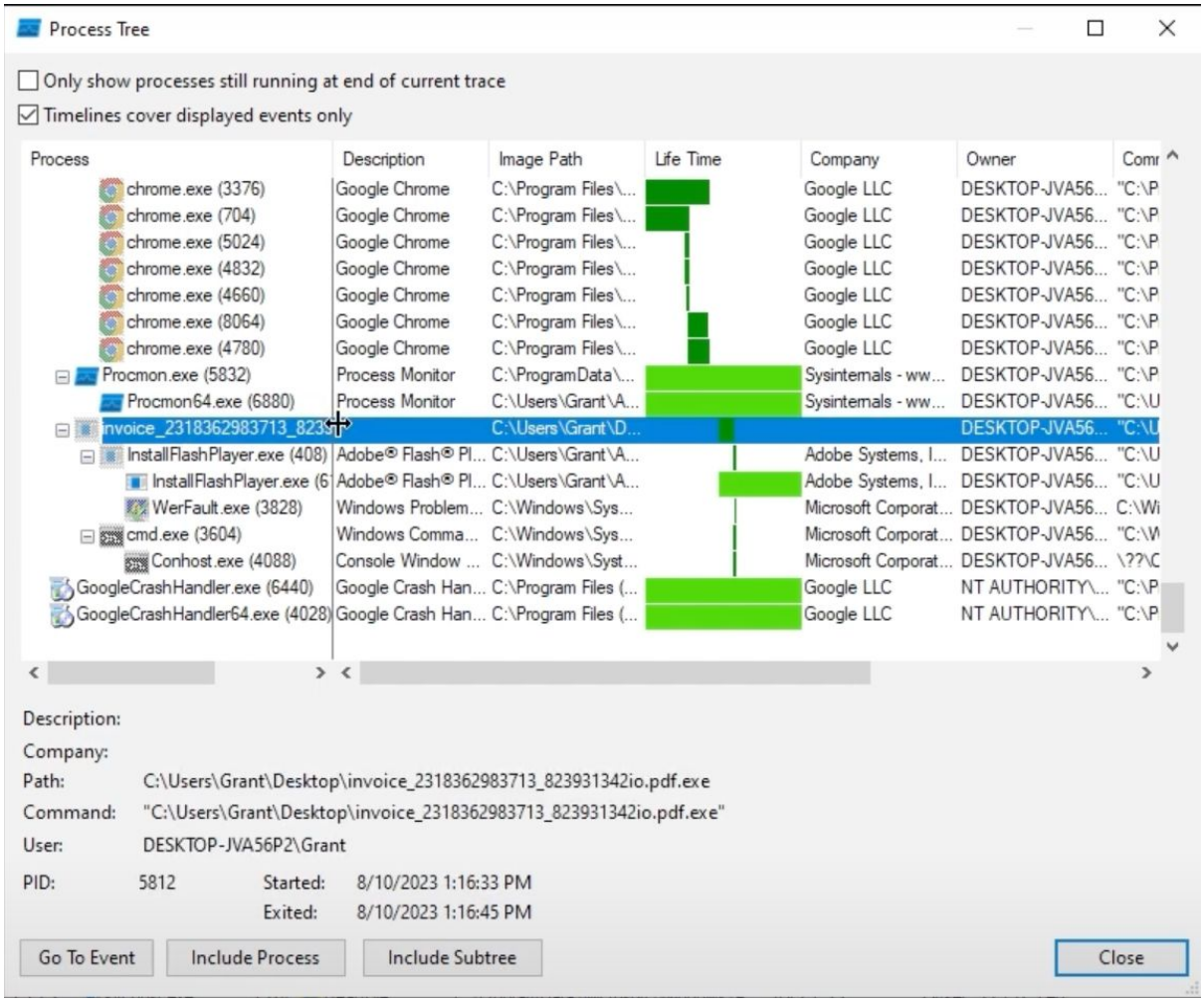

4 . Now double-click on "**invoice_2318362983713_823931342io.pdf.exe**" file , we can see that is say that do you want to install Adobe Flash Player .


5. Click on Yes.

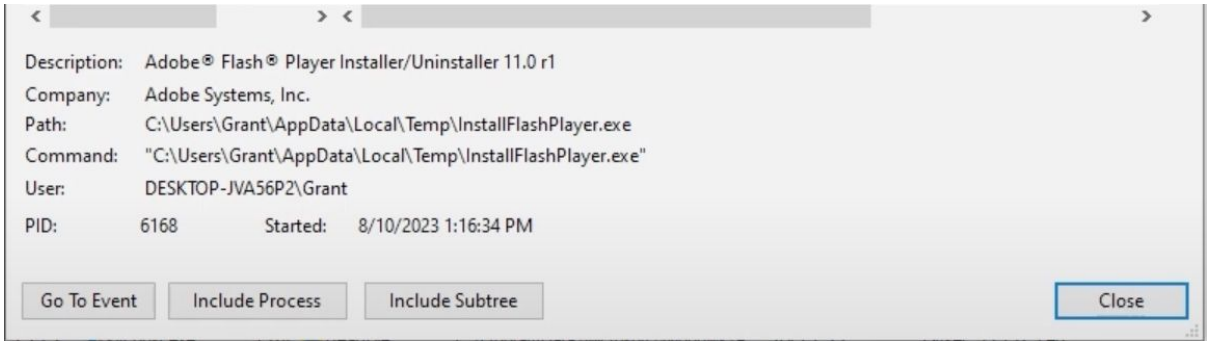
6 . Here in Process Monitor , under Process Tree we can see that

"**invoice_2318362983713_823931342io.pdf.exe**" file is running .

And we can **see InstallFlashPlayer.exe** is also running.

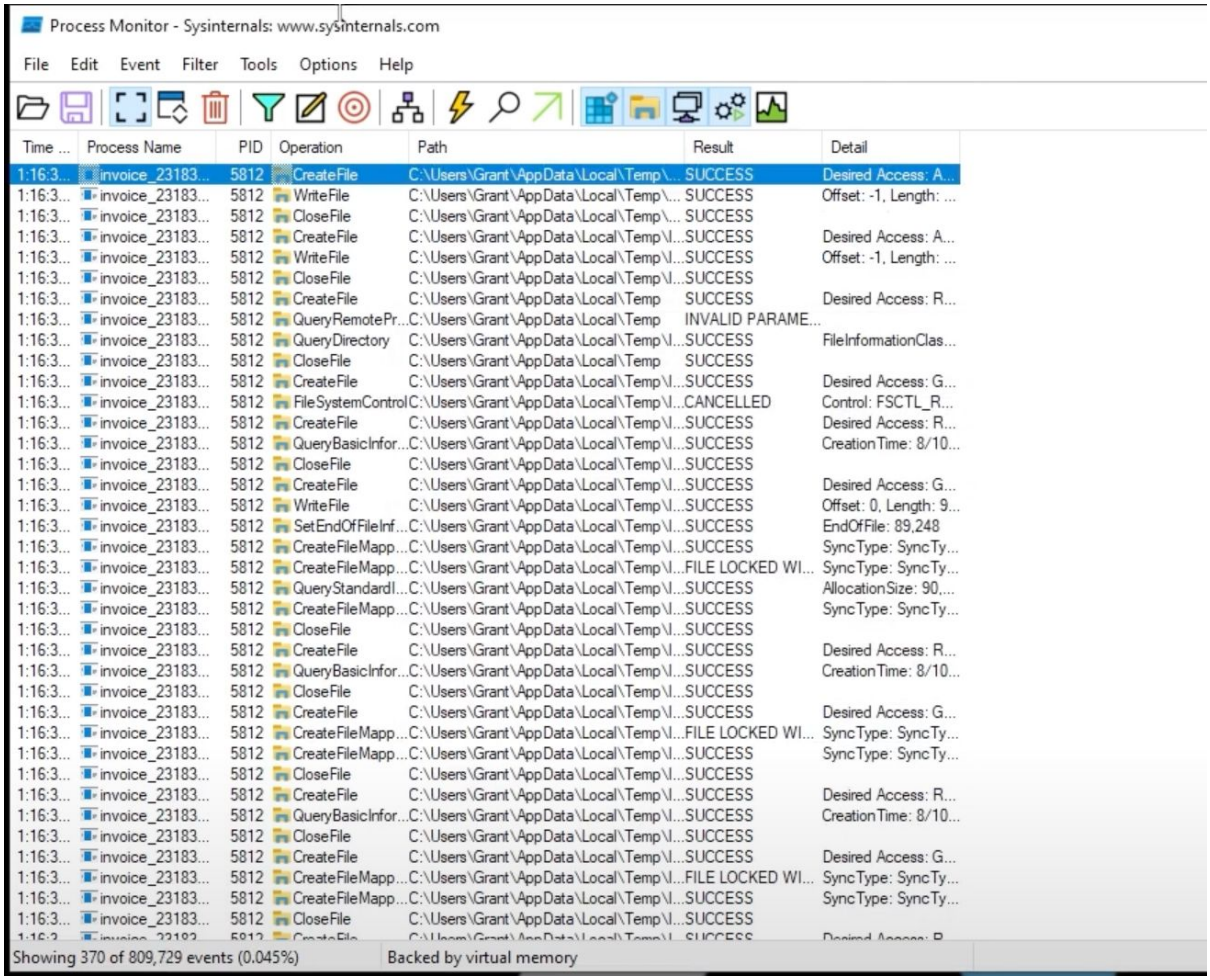



7 . Here we can see InstallFlashPlayer.exe is stored in Temp Folder.

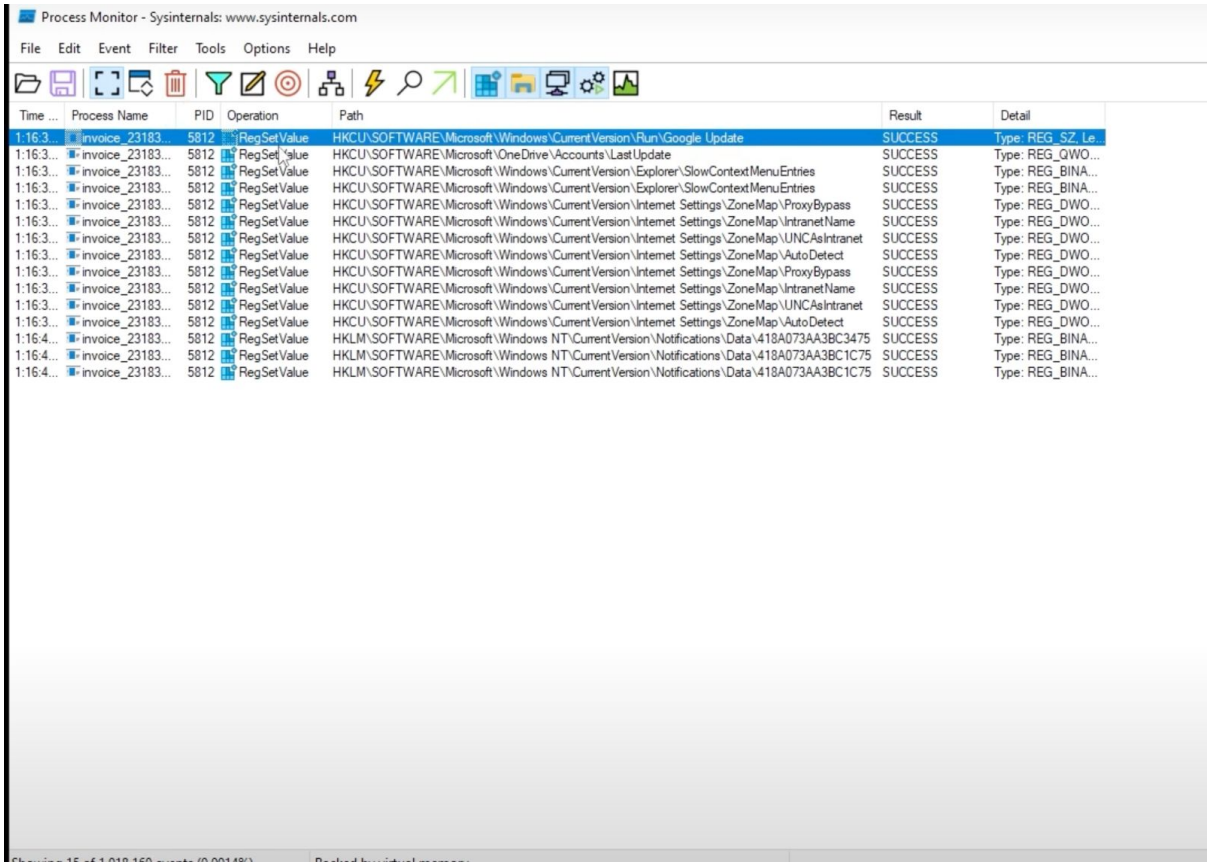



8. Now go to filter and select Process Name , select contains , and type invoice , and Click on Add.

9 . Now go to filter and select Path , select contains , and give the path of Temp folder .



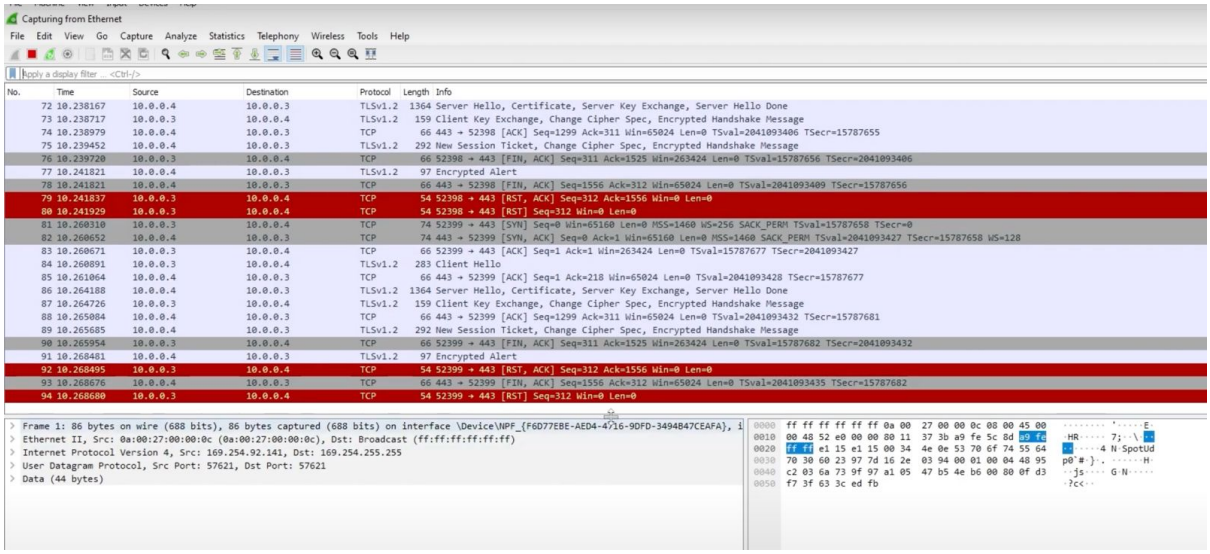

10 . Now go to filter and select Operation , select contains , type RegSetValues and Click on Add . And remove Path filter.



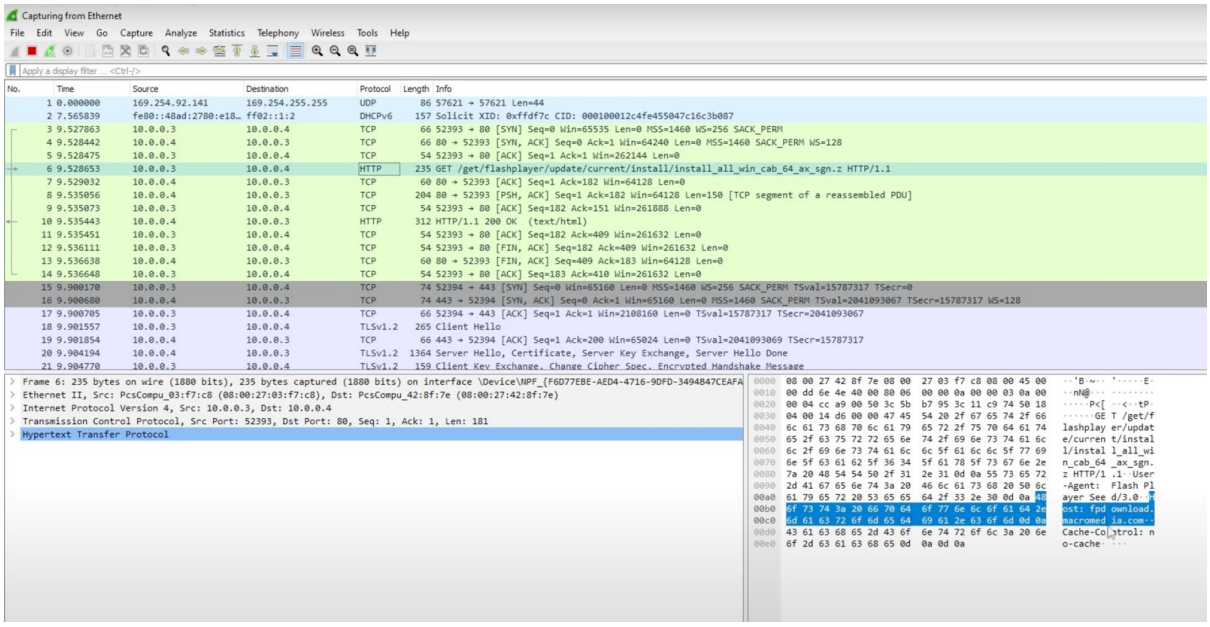

11 . Now lets see their is any Network Based Indicators .

12 . Open Wireshark , now lets collect packets on Ethernet .

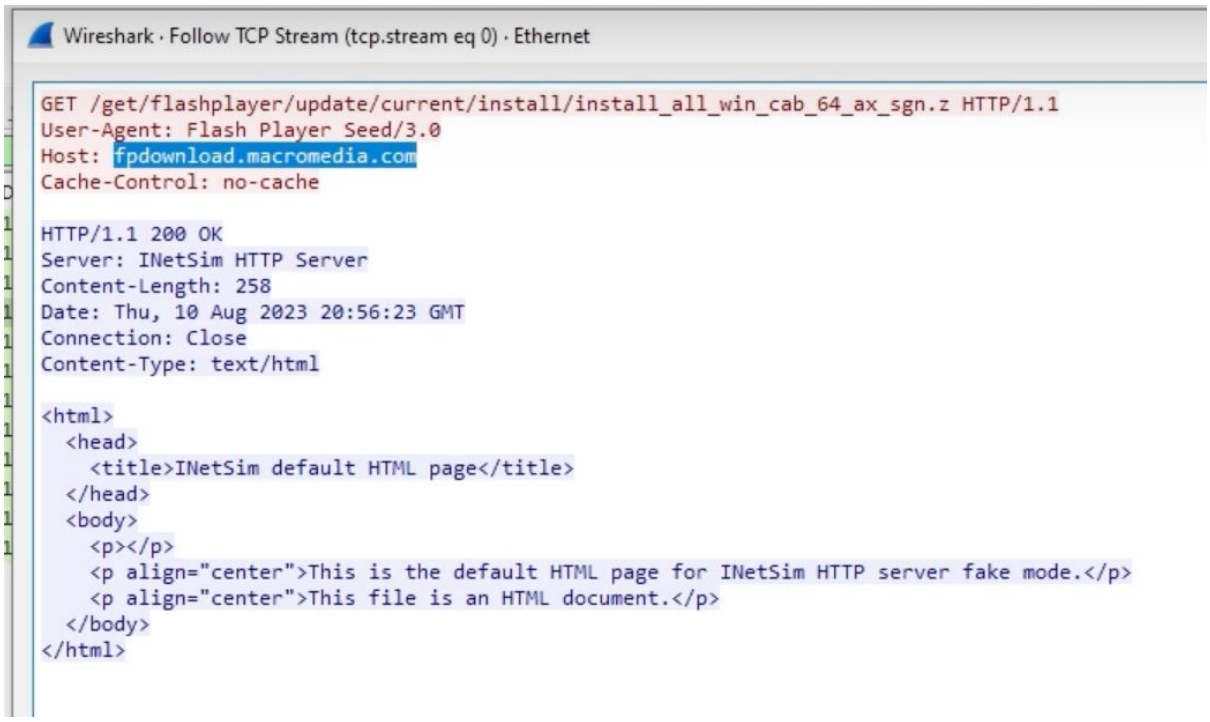
13 . Now start the Wireshark tool and execute "**invoice_2318362983713_823931342io.pdf.exe**" file .




14. Here we can see a HTTP GET request . Now click on it .

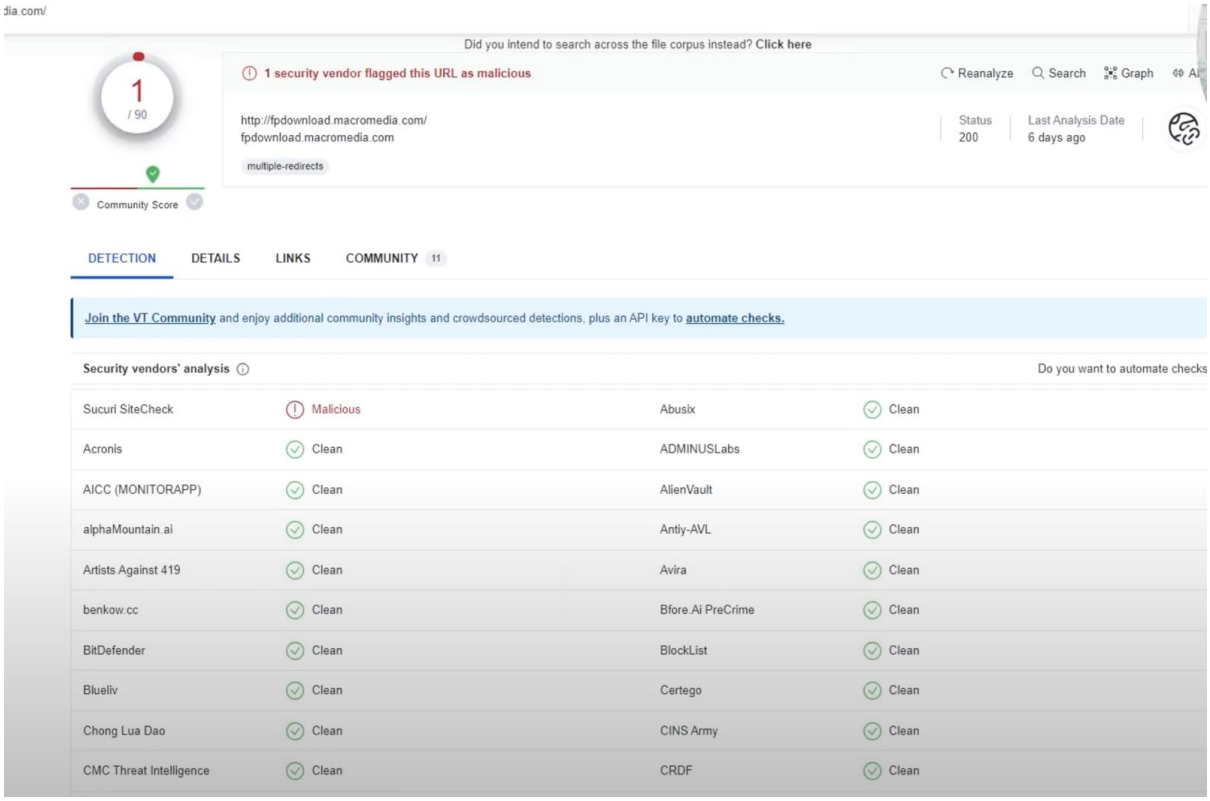



15 . Now right click on it and click on follow , then TCP Stream

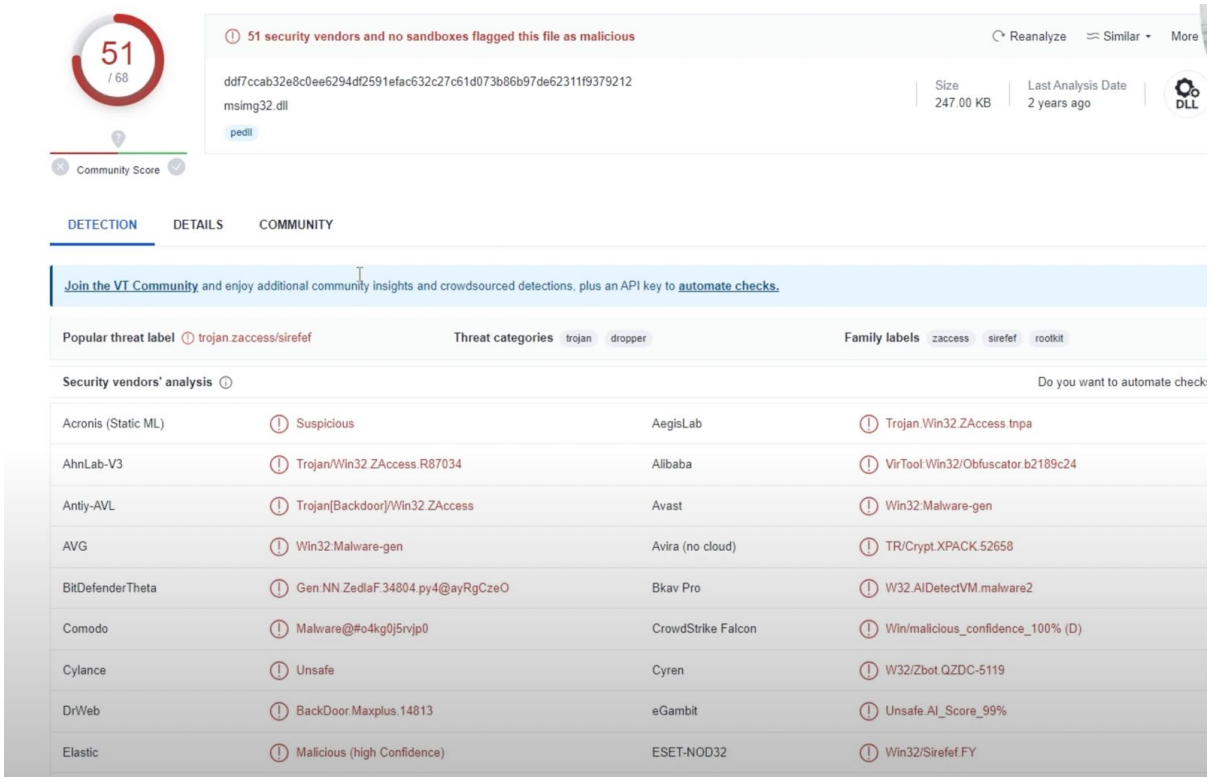



16 . Go to virus total , go to URL and enter the Host name we got .

Host : **fpdownload.macromedia.com**




17. Go to virus total and upload msimg32.dll file .




18 . We can see this is something malicious

# Measures to Prevent Zeus Attacks

To protect systems and networks from Zeus and similar banking Trojans, the following preventive measures should be implemented:

- **Use Multi-Factor Authentication (MFA):** Prevents unauthorized access even if credentials are stolen.
- **Avoid Clicking on Suspicious Links:** Educate users about phishing emails and fake websites.
- **Update Security Software:** Regularly update antivirus, firewalls, and intrusion detection systems.
- **Enable Network Segmentation:** Restrict access between systems to prevent malware propagation.
- **Monitor Network Traffic:** Use tools like Wireshark to detect suspicious communication with C2 servers.
- **Implement Endpoint Detection & Response (EDR):** Helps identify and respond to threats in real-time.

# Existing Solutions for Zeus Mitigation

Several cybersecurity solutions have been developed to detect and mitigate Zeus Trojan infections:

- **Microsoft Defender & Windows Security Tools:** Detects known Zeus variants and blocks execution.
- **Banking Security Software:** Some financial institutions use fraud detection mechanisms to identify unusual transactions.
- **Threat Intelligence Services:** Companies like FireEye, Palo Alto Networks, and CrowdStrike offer solutions to track and prevent botnet activity.
- **Behavioral Analysis Tools:** AI-driven solutions like Darktrace can analyze unusual user behaviors and alert security teams.
- **YARA Rules & Snort Signatures:** Used by cybersecurity professionals to detect and remove Zeus-infected files.

# Conclusion

The Zeus Banking Trojan remains one of the most **sophisticated financial cyber threats** ever created. Despite efforts to dismantle its botnets, its **evolving variants continue to target financial institutions** globally. With increased cybersecurity awareness and the use of advanced detection tools, organizations can reduce the risk of infection and minimize financial losses. The continuous adaptation of cybercriminals necessitates proactive measures, constant monitoring, and improved defense mechanisms to stay ahead of evolving threats.

# References


1. "Zeus Trojan Analysis - A Deep Dive," FireEye Threat Intelligence Report, 2023.
2. Palo Alto Networks, "Understanding and Mitigating Banking Trojans," Cybersecurity Whitepaper, 2022.
3. Microsoft Security Blog, "Zeus and its Variants: Detection and Prevention Strategies," 2021.
4. Darktrace AI Research, "Machine Learning in Cyber Threat Detection: Banking Trojan Case Study," 2023.
5. VirusTotal, "Zeus Malware Samples and Detection Rates," Public Report Database, 2023.
6.