

Title: **Dissecting Zeus: Malware Analysis Report of the 2013 Zbot Malware**

Investigator 1: Samantha Henry

Investigator 2: Rayyan Baig Mirza

Date: 4/29/2025

1. Summary

The Zeus (Zbot) 2013 Trojan malware is a 32-bit Windows executable that is packed with UPX, Themida, and NSPack. It employs self-modifying code and dynamic API resolution to avoid static detection and reverse engineering. Anti-VM checks for VMware were discovered and suggest that it attempts to circumvent sandbox analysis.

Static and dynamic analysis found Zeus stealing keystrokes (GetAsyncKeyState), clipboard contents (GetClipboardData), writing data to files (WriteFile), anti-debugging techniques (GetTickCount), and running additional commands (WinExec). These actions allow it to steal sensitive data such as online banking credentials and tighten its hold on compromised systems.

Beyond credential theft, Zeus acts like an intruder, adapting to the environment, hiding its tracks, and configuring infected machines for long-term exploitation. Its mixture of stealth, persistence, and flexible attack features made it one of the most successful malware families in financial cybercrime history.

Malware Sample Download:

https://github.com/ytisf/theZoo/tree/master/malware/Binaries/ZeusBankingVersion_26Nov2013

Password: infected

2. Identification

2.1) Filename: ZeusBankingVersion_26Nov2013

File size: 247.00 KB (252928 bytes)

File type: Win32 EXE, PE32 executable (GUI) Intel 80386, for MS Windows

2.2) MAC timestamps

The MAC (Modified, Accessed, Created) timestamps were extracted from the PE header of the sample.

pestudio 9.59 - Malware Initial Assessment - www.winitor.com (read-only)

file settings about

File Analysis (File: invoice_2318367083713)

property value detail

characteristics 0x0102

dynamic-link-library 0x0000 false

32-bit words support 0x0100 true

file can be executed 0x0002 true

system-image 0x0000 false

large-address-aware 0x0000 false

debug-stripped 0x0000 false

line-stripped-from-file 0x0000 false

local-symbols-stripped-from-file 0x0000 false

relocation-stripped 0x0000 false

uniprocessor 0x0000 false

bytes-of-machine-words-reversed-Low 0x0000 false

bytes-of-machine-words-reversed-Hi 0x0000 false

media-run-from-swap 0x0000 false

network-run-from-swap 0x0000 false

general

property	value	detail
compiler-stamp	0x529327223	Mon Nov 25 10:32:03 2013 (UTC)
size-of-optimal-header	0x0050	234 bytes
signature	0x00004550	PE00
machine	0x014C	Intel-386
sections-count	0x0005	6
pointer-symbol-table	0x00000000	0x00000000
number-of-symbols	0x00000000	0x00000000

< >

sha256: 69E966E730557FD18FD4317CDEF1ECE00A8B83470C0656F5331E170160A169

cpu: 32-bit File type: executable Subsystem: GUI

entry-point: 0x000A366

The TimeDateStamp value 0x52932273 was manually decoded to confirm the following:

2013-11-25 13:29:23

2.3) Hashes

MD5: ea039a854d20d7734c5add48f1a51c34

SHA-1: 9615dca4c0e46b8a39de5428af7db060399230b2

SHA-256: 69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169

SSDEEP:

6144:Tz/LBBTHT+7oEf2ZstxQMSGToLoOhD2saLsW8fsmFBkObjD:PLBdy7FpQMIToThD+sW8fsmP7bj
fuzzy)

2.4) Signing Information (Certificates): N/A

2.5) Packer information

UPX (Ultimate Packer for Executables)

Themida

NSPack.

2.6) Aliases

AhnLab-V3 - Trojan/Win32.ZAccess.R87034

AliCloud - Backdoor: Win/ZAccess.emkb

K7AntiVirus - RootKit (004b9ee21)

Microsoft - PWS:Win32/Zbot or Trojan:Win32/Zbot

2.7) Dependencies

The ZeusBankingVersion_26Nov2013 malware sample relies on several Windows libraries (DLLs) to perform its malicious activities. Based on static analysis in PEStudio and IDA Pro, the following dependencies were identified:

GetAsyncKeyState (user32.dll): Capture key presses (keylogging)

GetClipboardData (user32.dll): Steal clipboard data

CallWindowProcW (user32.dll): Interact with GUI windows

CreateFileMappingA (kernel32.dll): Map files into memory

GetEnvironmentVariableA (kernel32.dll): Read system environment info

LocalAlloc / LocalFree (kernel32.dll): Allocate or free memory

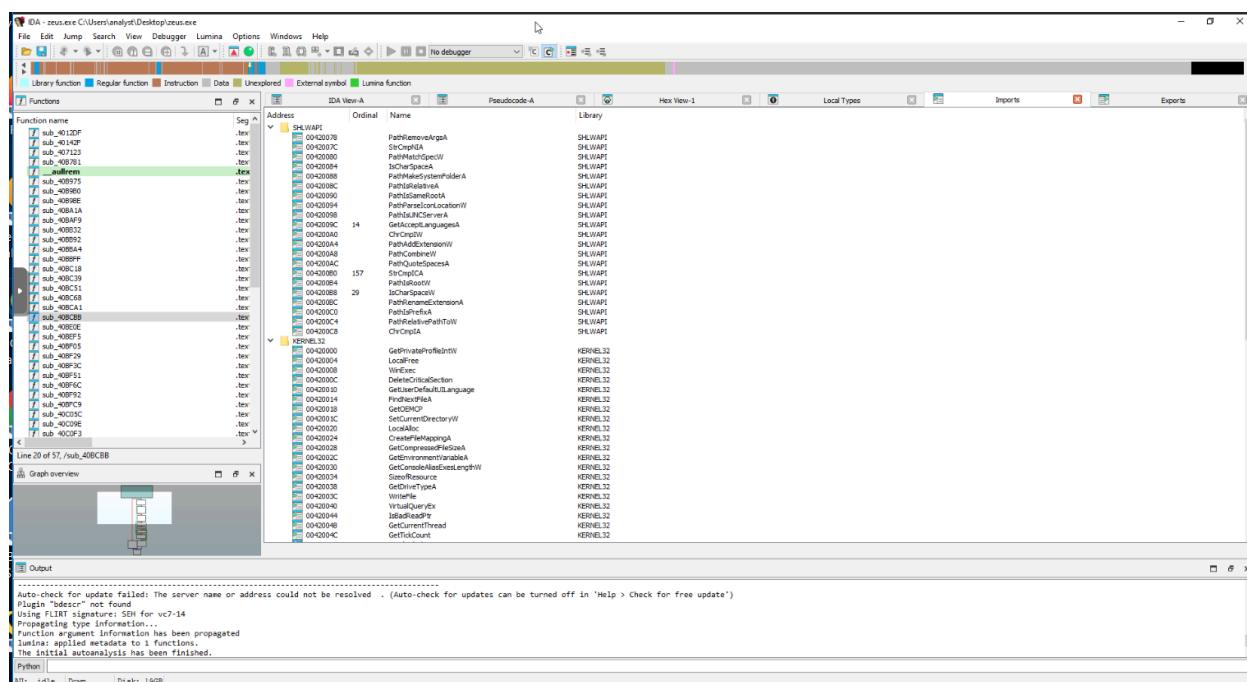
FindNextFileW (kernel32.dll): Enumerate files/folders

GetDriveTypeA (kernel32.dll): Detect connected drives

These dependencies highlight the malware's focus on stealing information through keylogging and clipboard capturing, file system interaction, and system reconnaissance.

library (3)	duplicate (0)	flag (0)	first-thunk-original (INT)	first-thunk (IAT)	type (1)	imports (77)	group (0)	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

File Details							
File Path		File Info		Imports		Exports	
Section	Description	Size	Hash	Count	Count	Count	Count
File	Invoice.exe	1024	sha256: 69E66E730557FDE0FD84317CDEF1CE00A8BB347CC0B58F2321E170168AF169	1	1	0	0
Imports	77	15	0x00020684	1	1	0	0
Export	0	0	0x00020680	1	1	0	0
Manifest	1	0	0x000207C8	1	1	0	0
Resource	11	0	0x00020718	1	1	0	0
Thread Local Storage	0	0	0x00020650	1	1	0	0
NET	0	0	0x00020656	1	1	0	0
Strings	18	0	0x00020670	1	1	0	0
Debug	0	0	0x00020716	1	1	0	0
Manifest	1	0	0x0002078C	1	1	0	0
Resource	0	0	0x000207D4	1	1	0	0
Environment Variable	1	0	0x000207D0	1	1	0	0
Drive Type	4	0	0x00020580	1	1	0	0
Get Tick Count	0	0	0x00020526	1	1	0	0
Environment Variable W	0	0	0x000204D4	1	1	0	0
Logical Drives	0	0	0x00020430	1	1	0	0
Local Free	0	0	0x00020494	1	1	0	0
Local Alloc	0	0	0x00020648	1	1	0	0
Virtual Query Ex	0	0	0x00020508	1	1	0	0
Local Unlock	0	0	0x00020508	1	1	0	0
Heap Free	0	0	0x00020488	1	1	0	0
Swap Mouse Button	0	0	0x00020830	1	1	0	0
VKey Scan A	0	0	0x00020830	1	1	0	0
Get Asynch Key State	0	0	0x0002074A	1	1	0	0
Get Capture	0	0	0x0002077E	1	1	0	0
Copy Accelerator Table A	0	0	0x00020780	1	1	0	0
Path Combine V	0	0	0x00020356	1	1	0	0
Path Quote Space A	0	0	0x00020342	1	1	0	0
Path Rename Extension A	0	0	0x0002031C	1	1	0	0
Find Next File A	0	0	0x000205FA	1	1	0	0
Create File Mapping A	0	0	0x000205B2	1	1	0	0



3. Capabilities & Characteristics

Escalate User Privilege [SO]

Zeus displays a fake Adobe Flash Player UAC prompt to trick users into granting administrative privileges.

Capture Keyboard Input [TO]

Zeus captures keystrokes using GetAsyncKeyState to monitor user activity and steal confidential information.

Self-Modification [SO]

Zeus modifies its own code at runtime via writable and executable sections to avoid detection.

Obfuscate Imports [TO]

Zeus dynamically resolves APIs (e.g., GetProcAddress) instead of using static imports to obstruct static analysis.

Detect VM Environment [TO]

Zeus checks for virtualized environments such as VMware and avoids execution in sandboxes.

Persist After System Reboot [TO]

Zeus creates a registry key under HKCU\Software\Microsoft\Windows\CurrentVersion\Run to automatically re-execute after reboot.

Capabilities
<ul style="list-style-type: none">— Load-Code Resolve function by parsing PE exports
<ul style="list-style-type: none">— Anti-Analysis Reference anti-VM strings targeting VMWare

pestudio 9.59 - Malware Initial Assessment - www.wintor.com (read-only)

file settings about

File indicators (sections > self-modifying)

sha256: 69E966E730557FDE8FD84317CDEF1ECE00A8BB3470C0B58F3231E170168AF169

cpu: 32-bit file > type: executable subsystem: GUI entry-point: 0x0000A3B6

property	value	value	value	value	value	value
section	section[0]	section[1]	section[2]	section[3]	section[4]	section[5]
name	.text	.data	.text	.pdata	.src	.reloc
footprint > sha256	830985D320B3D392E25AFD5...	510A0F9AF189356CA7819A...	4CDD5D9821CC0790A1D703...	70CC3E025CCED228E4EBB2...	CB1CB914AD7F61C98BF865...	7C2F4C4D1
entropy	6.707	6.130	4.819	6.768	6.143	6.441
file-ratio (99.60%)	18.42 %	30.16 %	1.01 %	38.66 %	9.11 %	2.23 %
raw-address (begin)	0x00000400	0x0000BA00	0x0001E400	0x0001EE00	0x00036C00	0x0003C50
raw-address (end)	0x0000B400	0x0001E400	0x0001EE00	0x00036C00	0x0003C500	0x0003C0
raw-size (251904 bytes)	0x00008600 (46592 bytes)	0x00012400 (76288 bytes)	0x00000A00 (2560 bytes)	0x00017E00 (97792 bytes)	0x00005A00 (23040 bytes)	0x000160
virtual-address	0x00001000	0x0000D000	0x00020000	0x00021000	0x00039000	0x0003F00
virtual-size (250379 bytes)	0x0000B571 (46449 bytes)	0x000128B1 (75953 bytes)	0x0000084D (2125 bytes)	0x00017CBE (97470 bytes)	0x000058F2 (22770 bytes)	0x00015E1
characteristics	0x60000020	0xC0000040	0xC0000040	0xE0000020	0x40000040	0x4200004
write	-	x	x	x	-	-
execute	x	-	-	x	-	-
share	-	-	-	-	-	-
self-modifying	-	-	-	x	-	-
virtual	-	-	-	-	-	-
items						
directory > import	-	-	0x00020140	-	-	-
directory > resource	-	-	-	-	0x00039000	-
directory > relocation	-	-	-	-	-	0x0003F00
directory > import-address	-	-	0x00020000	-	-	-
exports > name (RVA)	-	-	-	0x00033F6	-	-
manifest	-	-	-	-	0x0003C378	-
base_of_code	0x00001000	-	-	-	-	-
base_of_data	-	0x0000D000	-	-	-	-
entry-point	0x0000A3B6	-	-	-	x	-

4. Dependencies

GetAsyncKeyState (user32.dll): Capture key presses (keylogging)

GetClipboardData (user32.dll): Steal clipboard data

CallWindowProcW (user32.dll): Interact with GUI windows

CreateFileMappingA (kernel32.dll): Map files into memory

GetEnvironmentVariableA (kernel32.dll): Read system environment info

LocalAlloc / LocalFree (kernel32.dll): Allocate or free memory

FindNextFileW (kernel32.dll): Enumerate files/folders

GetDriveTypeA (kernel32.dll): Detect connected drives

5. Static Analysis

This screenshot shows the VirusTotal scan results for the Zeus sample. Out of 72 antivirus engines, 64 detected the file as malicious, labeling it with names like Trojan.ZAccess and Trojan.WLDCR. Based on the threat categories listed (Trojan, dropper), it confirms that the sample is likely used for stealing data and possibly dropping malware.

64/72 security vendors flagged this file as malicious

64e966e730557fde8fd84317cdef1ece00aabb3470c0b58f3231e170168af169
invoice_2318362983713_823931342io.pdf.exe

Size 247.00 KB | Last Analysis Date 4 days ago | EXE

Community Score -433

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

Popular threat label trojan.zaccess/sirefef

Threat categories trojan dropper

Family labels zaccess sirefef wlfdc

Security vendor	Analysis	Family	Analysis
AhnLab-V3	Trojan/Win32.ZAccess.R87034	Alibaba	Backdoor:Win32/ZAccess.71cb6d44
AliCloud	Backdoor:Win/ZAccess.emkbt	ALYac	Trojan.ZeroAccess.RN
Anti-AVL	Trojan[Backdoor]/Win32.ZAccess	Arcabit	Trojan.WLDCR.C
Arctic Wolf	Unsafe	Avast	Win32:MalwareX-gen [Cryp]
AVG	Win32:MalwareX-gen [Cryp]	Avira (no cloud)	TR/Crypt.XPACK.52658
BitDefender	Trojan.WLDCR.C	Bkav Pro	W32.AIDetectMalware
CrowdStrike Falcon	Win/malicious_confidence_100% (W)	CTX	Exe.trojan.wldcr

5.1) Top-level components

The sample is a 32-bit Windows PE32 Intel 80386 targeted executable. The main sections, such as .text, .data, .rsrc, .pdata, and .reloc were discovered. The .itext section contains write and execute permissions, which is a sign of self-modifying code behavior. Sections such as .pdata and .rsrc contained high entropy values (~6.7), which is an indication of compression or encryption.

pstudio 9.59 - Malware Initial Assessment - www.winitor.com (read-only)																																																									
file	settings	about																																																							
<ul style="list-style-type: none"> clusers\analyst\desktop\invoice_2318362983713 ↳ indicators (sections > self-modifying) <ul style="list-style-type: none"> file footprints (type > sha256) <ul style="list-style-type: none"> virustotal (status > error) dos-header (size > 64 bytes) dos-dstb (size > 152 bytes) rich-header (tooling > Visual Studio 2010) file-header (executable > 32-bit) optional-header (subsystem > GUI) directories (count > 4) <ul style="list-style-type: none"> ↳ sections (characteristics > self-modifying) <ul style="list-style-type: none"> libraries (count > 3) <ul style="list-style-type: none"> ↳ imports (flag > ?) <ul style="list-style-type: none"> file > n/a file > sha256 file > size file > type file > signature tooling ↳ resources (count > 11) <ul style="list-style-type: none"> ↳ strings (flag > 18) <ul style="list-style-type: none"> debug resource > n/a ↳ resources (count > 11) <ul style="list-style-type: none"> ↳ strings (flag > 18) <ul style="list-style-type: none"> debug resource > n/a ↳ manifest (level > asInvoker) <ul style="list-style-type: none"> version (n/a) certificate (n/a) overlay (n/a) 	<ul style="list-style-type: none"> indicator (24) <ul style="list-style-type: none"> sections > self-modifying <ul style="list-style-type: none"> imports > flag <ul style="list-style-type: none"> string > suspicious string > flag file > entropy file > signature tooling file > sha256 file > size file > type virustotal > status <ul style="list-style-type: none"> ↳ executable, 32-bit, GUI ↳ executable, 32-bit resource > items <ul style="list-style-type: none"> manifest > general debug <ul style="list-style-type: none"> entry-point > address certificate imports > ordinal exports overlay 	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">detail</th> <th style="text-align: right;">level</th> </tr> </thead> <tbody> <tr> <td>name: pdata</td> <td style="text-align: right;">+++++</td> </tr> <tr> <td>count: 2</td> <td style="text-align: right;">+++</td> </tr> <tr> <td>Path\Name\ExtensionsA WinExec FindNextFileA GetEnvironmentVar...</td> <td style="text-align: right;">+++++</td> </tr> <tr> <td>size: 6707 bytes</td> <td style="text-align: right;">++</td> </tr> <tr> <td>size: 6709 bytes</td> <td style="text-align: right;">++</td> </tr> <tr> <td>size: 6710 bytes</td> <td style="text-align: right;">++</td> </tr> <tr> <td>size: 6701 bytes</td> <td style="text-align: right;">++</td> </tr> <tr> <td>size: 6708 bytes</td> <td style="text-align: right;">++</td> </tr> <tr> <td>count: 18</td> <td style="text-align: right;">++</td> </tr> <tr> <td>6.92</td> <td style="text-align: right;">+</td> </tr> <tr> <td>Visual Studio 2008</td> <td style="text-align: right;">+</td> </tr> <tr> <td>09E9606730557F0E8FD84317CDEF1ECE00A8BB347DC0B59F3231E170168A...</td> <td style="text-align: right;">+</td> </tr> <tr> <td>252928 bytes</td> <td style="text-align: right;">+</td> </tr> <tr> <td>executable, 32-bit, GUI</td> <td style="text-align: right;">+</td> </tr> <tr> <td>executable, 32-bit</td> <td style="text-align: right;">+</td> </tr> <tr> <td>The server name or address could not be resolved</td> <td style="text-align: right;">+</td> </tr> <tr> <td>Mon Nov 25 13:32:03 2013</td> <td style="text-align: right;">+</td> </tr> <tr> <td>count: 11, size: 22178 bytes, file-ratio: 8.77%</td> <td style="text-align: right;">+</td> </tr> <tr> <td>name: n/a, description: n/a, level: asInvoker</td> <td style="text-align: right;">+</td> </tr> <tr> <td>n/a</td> <td style="text-align: right;">+</td> </tr> <tr> <td>0x0000A3B6</td> <td style="text-align: right;">+</td> </tr> <tr> <td>n/a</td> <td style="text-align: right;">+</td> </tr> <tr> <td>count: 3</td> <td style="text-align: right;">+</td> </tr> <tr> <td>n/a</td> <td style="text-align: right;">+</td> </tr> <tr> <td>n/a</td> <td style="text-align: right;">+</td> </tr> <tr> <td>n/a</td> <td style="text-align: right;">+</td> </tr> </tbody> </table>	detail	level	name: pdata	+++++	count: 2	+++	Path\Name\ExtensionsA WinExec FindNextFileA GetEnvironmentVar...	+++++	size: 6707 bytes	++	size: 6709 bytes	++	size: 6710 bytes	++	size: 6701 bytes	++	size: 6708 bytes	++	count: 18	++	6.92	+	Visual Studio 2008	+	09E9606730557F0E8FD84317CDEF1ECE00A8BB347DC0B59F3231E170168A...	+	252928 bytes	+	executable, 32-bit, GUI	+	executable, 32-bit	+	The server name or address could not be resolved	+	Mon Nov 25 13:32:03 2013	+	count: 11, size: 22178 bytes, file-ratio: 8.77%	+	name: n/a, description: n/a, level: asInvoker	+	n/a	+	0x0000A3B6	+	n/a	+	count: 3	+	n/a	+	n/a	+	n/a	+	<ul style="list-style-type: none"> entry-point: 0x0000A3B6
detail	level																																																								
name: pdata	+++++																																																								
count: 2	+++																																																								
Path\Name\ExtensionsA WinExec FindNextFileA GetEnvironmentVar...	+++++																																																								
size: 6707 bytes	++																																																								
size: 6709 bytes	++																																																								
size: 6710 bytes	++																																																								
size: 6701 bytes	++																																																								
size: 6708 bytes	++																																																								
count: 18	++																																																								
6.92	+																																																								
Visual Studio 2008	+																																																								
09E9606730557F0E8FD84317CDEF1ECE00A8BB347DC0B59F3231E170168A...	+																																																								
252928 bytes	+																																																								
executable, 32-bit, GUI	+																																																								
executable, 32-bit	+																																																								
The server name or address could not be resolved	+																																																								
Mon Nov 25 13:32:03 2013	+																																																								
count: 11, size: 22178 bytes, file-ratio: 8.77%	+																																																								
name: n/a, description: n/a, level: asInvoker	+																																																								
n/a	+																																																								
0x0000A3B6	+																																																								
n/a	+																																																								
count: 3	+																																																								
n/a	+																																																								
n/a	+																																																								
n/a	+																																																								

5.2) Execution points of entry

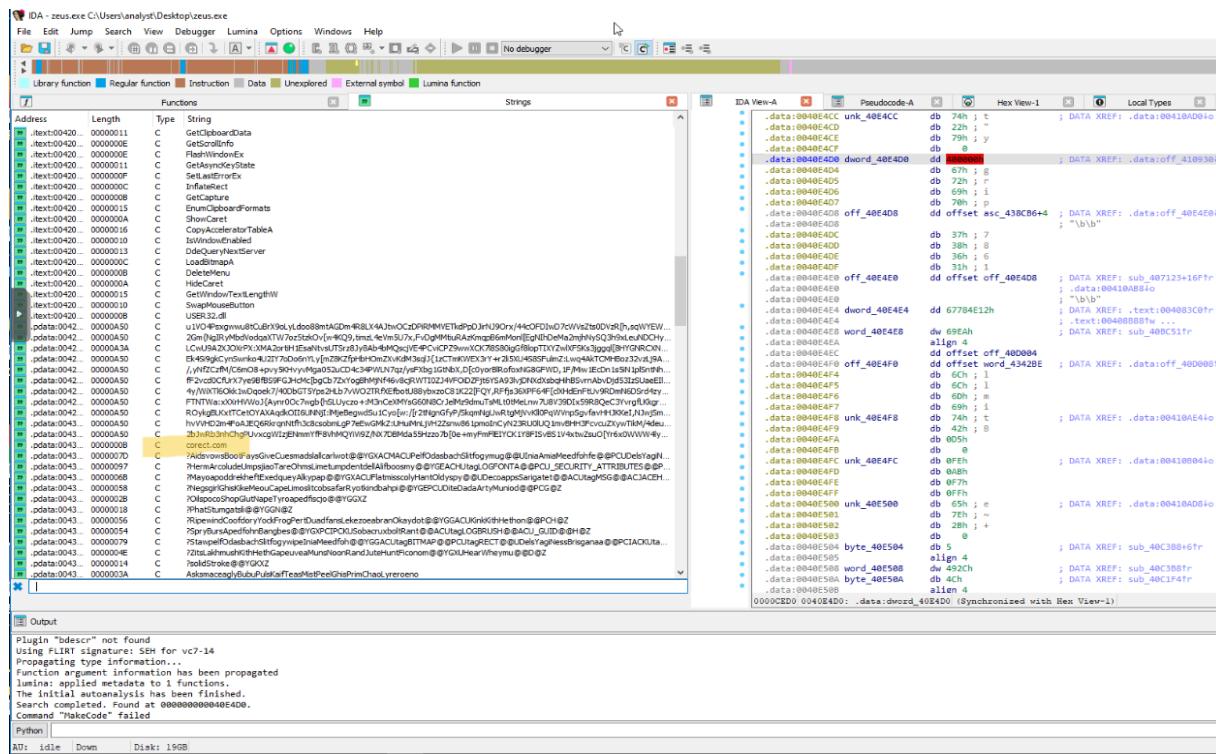
The sample entry point is seen at 0x0000A3B6 in both IDA Pro and PEStudio. There are instructions at the start that push and prepare registers, setting up stack frames, with multiple calls to dynamically resolve API functions, suggesting obfuscation at runtime.

The screenshot shows the Win32 debugger interface with multiple windows open. The main window displays the memory dump of the PE file, highlighting sections like 'optionalHeader.Address' and assembly code for the entry point. The assembly window shows the initial stack setup and calls to kernel32 functions. The registers window shows the CPU register state. The registers window also includes a 'Programmer' view for hex, decimal, octal, and binary representation of the current value (40 A3B6). The status bar at the bottom shows the assembly address (cbp-0019FF80) and the current instruction (0040A3B6 zeus.exe:5A3B6 #9786 <OptionalHeader.AddressOfEntryPoint>).

5.3) Embedded strings

Numerous embedded strings were identified, where a few of them mapped directly to Windows API calls usually employed for malicious purposes. A few of the notable extracted strings included GetClipboardData, GetAsyncKeyState, CreateFileMappingA, GetEnvironmentVariableA, and EnumClipboardFormats, which point towards keylogging, spying on the clipboard, and file functions.

Additionally, the string corect.com was discovered within the sample. This could be a potential Command and Control (C2) domain.



IDA Pro Assembly and Pseudocode

API Function: WriteFile

Description: Writes data to a file or input/output (I/O) device. Commonly used in malware to save stolen data or drop payloads.

IDA - invoice_2318362982712_3239313420.pdf.exe.004 (invoice_2318362982712_3239313420.pdf.exe) C:\Users\analyst\Desktop\invoice_2318362982712_3239313420.pdf.exe:004

File Edit Jump Search View Debugger Lumina Options Windows Help

Library functions Regular function Instruction Data Unexploited External symbol Lumina function

Functions

Function list

0x400511 0x400515 0x400518 0x400519 0x400529 0x40052C 0x40052D 0x40052E 0x40052F 0x400530 0x400531 0x400532 0x400533 0x400534 0x400535 0x400536 0x400537 0x400538 0x400539 0x40053A 0x40053B 0x40053C 0x40053D 0x40053E 0x40053F 0x400540 0x400541 0x400542 0x400543 0x400544 0x400545 0x400546 0x400547 0x400548 0x400549 0x40054A 0x40054B 0x40054C 0x40054D 0x40054E 0x40054F 0x400550 0x400551 0x400552 0x400553 0x400554 0x400555 0x400556 0x400557 0x400558 0x400559 0x40055A 0x40055B 0x40055C 0x40055D 0x40055E 0x40055F 0x400560 0x400561 0x400562 0x400563 0x400564 0x400565 0x400566 0x400567 0x400568 0x400569 0x40056A 0x40056B 0x40056C 0x40056D 0x40056E 0x40056F 0x40056G 0x40056H 0x40056I 0x40056J 0x40056K 0x40056L 0x40056M 0x40056N 0x40056O 0x40056P 0x40056Q 0x40056R 0x40056S 0x40056T 0x40056U 0x40056V 0x40056W 0x40056X 0x40056Y 0x40056Z 0x40056_ 0x40056B_

Line 57 of 57, /nullsub_

000402C7 00402B84: .text 00402B84 (Synchronized with Hex View-1)

IDA View-A Pseudocode-A Hex View-1 Local Types Imports Exports

0x400511 0x400515 0x400518 0x400519 0x400529 0x40052C 0x40052D 0x40052E 0x40052F 0x400530 0x400531 0x400532 0x400533 0x400534 0x400535 0x400536 0x400537 0x400538 0x400539 0x40053A 0x40053B 0x40053C 0x40053D 0x40053E 0x40053F 0x400540 0x400541 0x400542 0x400543 0x400544 0x400545 0x400546 0x400547 0x400548 0x400549 0x40054A 0x40054B 0x40054C 0x40054D 0x40054E 0x40054F 0x400550 0x400551 0x400552 0x400553 0x400554 0x400555 0x400556 0x400557 0x400558 0x400559 0x40055A 0x40055B 0x40055C 0x40055D 0x40055E 0x40055F 0x400560 0x400561 0x400562 0x400563 0x400564 0x400565 0x400566 0x400567 0x400568 0x400569 0x40056A 0x40056B 0x40056C 0x40056D 0x40056E 0x40056F 0x40056G 0x40056H 0x40056I 0x40056J 0x40056K 0x40056L 0x40056M 0x40056N 0x40056O 0x40056P 0x40056Q 0x40056R 0x40056S 0x40056T 0x40056U 0x40056V 0x40056W 0x40056X 0x40056Y 0x40056Z 0x40056_ 0x40056B_

000402C7 00402B84: .text 00402B84 (Synchronized with Hex View-1)

API Function: WinExec

Description: Allows execution of an application or a command. Commonly used in malware to silently launch programs or scripts.

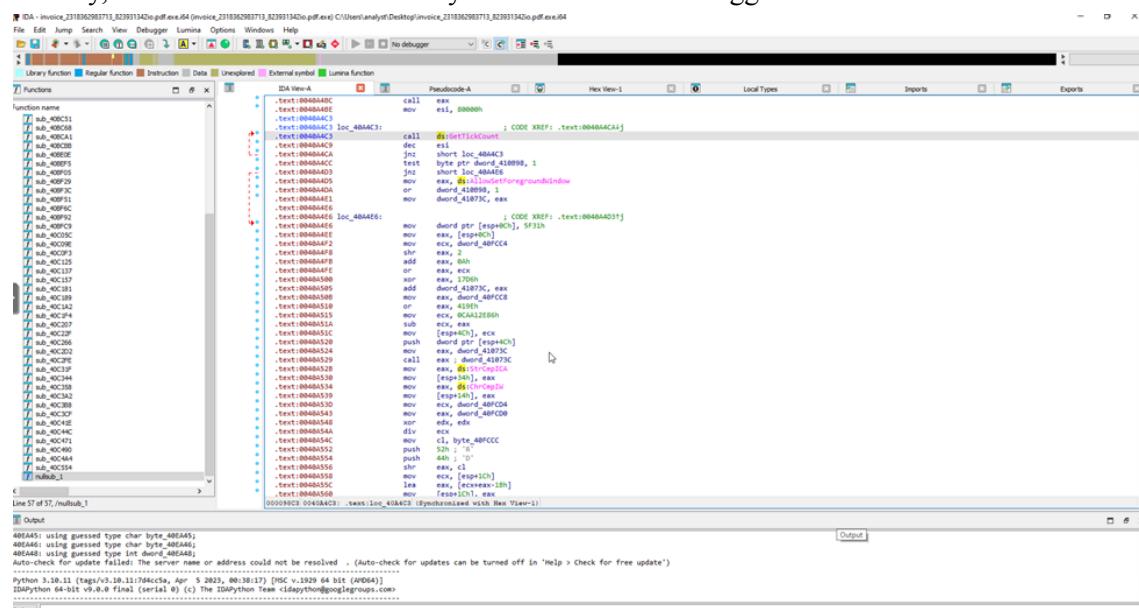
API Function: GetAsyncKeyState

Description: Checks if specific keys are being pressed. Commonly used in malware to capture keyboard input for a keylogger.

```
.text:004040CF8 xor    ax, cx
.text:004040CFE mov    ax, word_40E598
.text:004040D05 cmp    ax, cx
.jz    loc_405484
.text:004040D0E mov    eax, [8]@GetSyncKeyState
    [ebp-4], eax
.text:004040D13 mov    [ebp-24h], esi
.text:004040D16 mov    eax, dword_40F000
.text:004040D19 mov    ecx, dword_40EFFC
.or    eax, ecx
.text:004040D26 mov    cl, byte_40EFF8
.add   eax, 0FFFF1E4F0h
.text:004040D31 shr    eax, cl
.push 3
.text:004040D35 pop    ecx
.text:004040D36 xor    edx, edx
.text:004040D38 sub    eax, 21Bh
.text:004040D3D mov    [ebp-8Ch], eax
.text:004040D40 mov    [ebp-8], ecx
.text:004040D43 mov    dword ptr [ebp-38h], 14C1h
.text:004040D4A mov    dword ptr [ebp-4], 763h
.text:004040D51 mov    eax, dword_40F030
.text:004040D56 mov    esi, [ebp-4]
.text:004040D59 imul  eax, 3930h
.text:004040D5F div    esi
.text:004040D61 imul  edx, 5693h
.text:004040D67 sub    edx, 0F78C54h
.text:004040D6D mov    [ebp-4], edx
.text:004040D78 mov    [ebp-28h], ecx
.text:004040D73 mov    ecx, dword_40F04C
.text:004040D79 mov    eax, dword_40F048
.text:004040D7E sub    eax, ecx
.text:004040D80 mov    ecx, dword_40F044
.text:004040D66 xor    edx, edx
.text:004040D88 div    ecx
.text:004040D8A xor    edx, edx
.text:004040D8C mov    edi, eax
.text:004040D8E mov    eax, [ebp-4]
.text:004040D91 mov    eax, dword_40F034[eax*4]
.text:004040D98 mov    ecx, dword_40F02C
.text:004040D9E add    eax, ecx
.text:004040D9F mov    [ebp-4], eax
.text:004040D9J mov    eax, dword_40F01C
.text:004040D98 mov    ecx, 003h
.text:004040D9D div    ecx
00001000 0000000E: .text:004040D9E (Synchronized With Hex View-1)
```

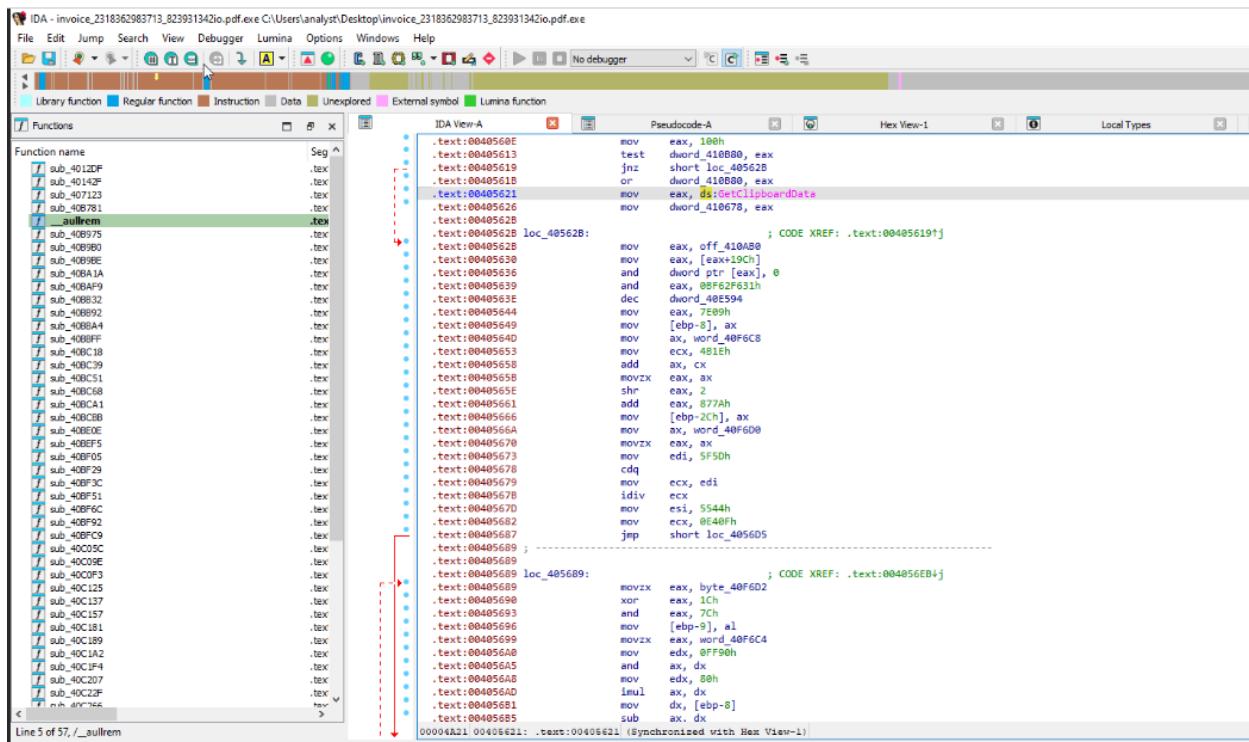
API Function: GetTickCount

Description: Retrieves the number of milliseconds since the system has started. Commonly used for anti-debugging to detect if the malware code is stepping through instructions, if the code is being analyzed slowly, or if there are abnormal delays in sandboxes or debuggers.



API Function: GetClipboardData

Description: Retrieves data from the clipboard. Commonly used in malware to capture sensitive content.



The screenshot shows the IDA Pro interface with the assembly and pseudocode for the `_aullen` function. The pseudocode view highlights the `GetClipboardData` API call, which is annotated with a red arrow and the text `; CODE XREF: .text:00405619+j`. The assembly code shows the function's logic, including various bitwise operations and memory manipulations.

```
IDA View-A Pseudocode-A Hex View-1 Local Types
Function name: _aullen
Seg: .text
Line 5 of 57, /_aullen
00004A11 00405621: .text:00405621 (Synchronized with Hex View-1)

; CODE XREF: .text:00405619+j
00004A11 00405621: .text:00405621 ; GetClipboardData
00004A12 00405622: .text:00405622
00004A13 00405623: .text:00405623
00004A14 00405624: .text:00405624
00004A15 00405625: .text:00405625
00004A16 00405626: .text:00405626
00004A17 00405627: .text:00405627
00004A18 00405628: .text:00405628
00004A19 00405629: .text:00405629 loc_40562B:
00004A1A 0040562A: .text:0040562A ; CODE XREF: .text:00405619+j
00004A1B 0040562B: .text:0040562B
00004A1C 0040562C: .text:0040562C
00004A1D 0040562D: .text:0040562D
00004A1E 0040562E: .text:0040562E
00004A1F 0040562F: .text:0040562F
00004A20 00405630: .text:00405630
00004A21 00405631: .text:00405631
00004A22 00405632: .text:00405632
00004A23 00405633: .text:00405633
00004A24 00405634: .text:00405634
00004A25 00405635: .text:00405635
00004A26 00405636: .text:00405636
00004A27 00405637: .text:00405637
00004A28 00405638: .text:00405638
00004A29 00405639: .text:00405639
00004A2A 0040563A: .text:0040563A
00004A2B 0040563B: .text:0040563B
00004A2C 0040563C: .text:0040563C
00004A2D 0040563D: .text:0040563D
00004A2E 0040563E: .text:0040563E
00004A2F 0040563F: .text:0040563F
00004A30 00405640: .text:00405640
00004A31 00405641: .text:00405641
00004A32 00405642: .text:00405642
00004A33 00405643: .text:00405643
00004A34 00405644: .text:00405644
00004A35 00405645: .text:00405645
00004A36 00405646: .text:00405646
00004A37 00405647: .text:00405647
00004A38 00405648: .text:00405648
00004A39 00405649: .text:00405649
00004A3A 0040564A: .text:0040564A
00004A3B 0040564B: .text:0040564B
00004A3C 0040564C: .text:0040564C
00004A3D 0040564D: .text:0040564D
00004A3E 0040564E: .text:0040564E
00004A3F 0040564F: .text:0040564F
00004A40 00405650: .text:00405650
00004A41 00405651: .text:00405651
00004A42 00405652: .text:00405652
00004A43 00405653: .text:00405653
00004A44 00405654: .text:00405654
00004A45 00405655: .text:00405655
00004A46 00405656: .text:00405656
00004A47 00405657: .text:00405657
00004A48 00405658: .text:00405658
00004A49 00405659: .text:00405659
00004A4A 0040565A: .text:0040565A
00004A4B 0040565B: .text:0040565B
00004A4C 0040565C: .text:0040565C
00004A4D 0040565D: .text:0040565D
00004A4E 0040565E: .text:0040565E
00004A4F 0040565F: .text:0040565F
00004A50 00405660: .text:00405660
00004A51 00405661: .text:00405661
00004A52 00405662: .text:00405662
00004A53 00405663: .text:00405663
00004A54 00405664: .text:00405664
00004A55 00405665: .text:00405665
00004A56 00405666: .text:00405666
00004A57 00405667: .text:00405667
00004A58 00405668: .text:00405668 loc_405669:
00004A59 00405669: .text:00405669 ; CODE XREF: .text:004056EB+j
00004A5A 0040566A: .text:0040566A
00004A5B 0040566B: .text:0040566B
00004A5C 0040566C: .text:0040566C
00004A5D 0040566D: .text:0040566D
00004A5E 0040566E: .text:0040566E
00004A5F 0040566F: .text:0040566F
00004A60 00405670: .text:00405670
00004A61 00405671: .text:00405671
00004A62 00405672: .text:00405672
00004A63 00405673: .text:00405673
00004A64 00405674: .text:00405674
00004A65 00405675: .text:00405675
00004A66 00405676: .text:00405676
```

IDA Pro Pseudocode

During the static analysis in IDA Pro, no decompiled pseudocode directly referenced the Windows APIs: WriteFile, WinExec, GetTickCount, GetAsyncKeyState, and GetClipboardData. These API functions were mainly visible in the disassembly view or the Imports table. However, the function `_aullen` was observed performing bitwise operations (`>>`, `<<`, `&`) and XORs, which are likely associated with hashing, decryption, or obfuscation. This suggests the malware may be decrypting API names, payloads, strings, configuration data.

5.4) Code-related observations (Reflection, Obfuscation, Encryption, Native code, etc)

Dynamic API Resolution:

Functions are resolved at runtime through techniques like manually parsing PE exports instead of static imports (LoadLibrary, GetProcAddress style obfuscation).

Self-Modifying Code:

The `.iText` section is marked as writable and executable, enabling the malware to alter its code during execution to evade static analysis.

Anti-VM Techniques

The malware attempts to detect VMware environments, checking for specific registry keys and process names

Entropy Analysis

Sections like .pdata and .rsrc have elevated entropy, suggesting packing, encryption, or data compression to obfuscate content

Native Code Execution

Malware does not utilize a managed runtime (such as .NET); instead, it executes native 32-bit Windows instructions.

Command-line tools

The malware sample was analyzed using the capa tool on the FLARE-VM environment to extract behavioral capabilities and tactics quickly. The capa scan confirmed the following:

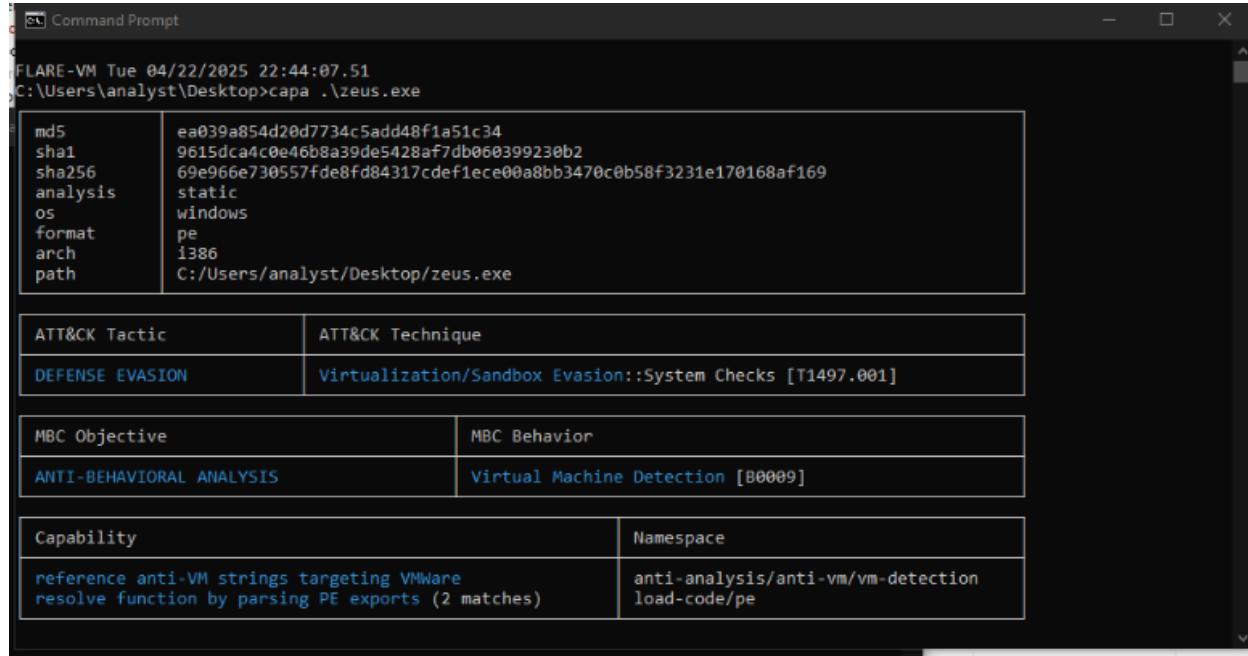
Anti-Analysis Behavior:

The sample contains functionality to detect if it is running inside a virtual machine environment, particularly VMware. This supports defense evasion and anti-behavioral analysis tactics.

Dynamic Code Loading:

Capa detected behavior associated with resolving function addresses at runtime by parsing PE exports. This matches manual findings from static analysis in IDA Pro and supports the malware's obfuscation and anti-static analysis capabilities.

Mapping to MITRE ATT&CK and MBC:



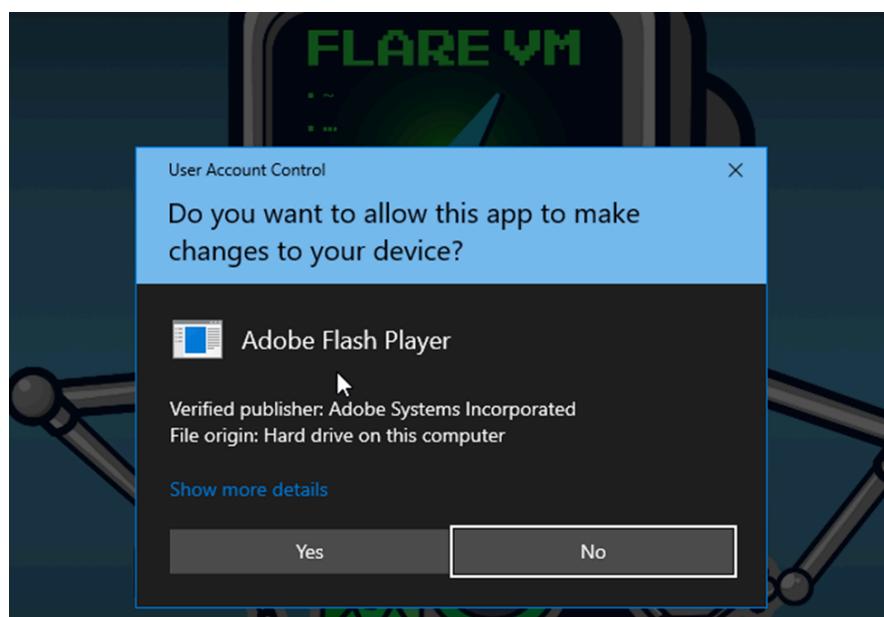
ATT&CK Tactic	ATT&CK Technique
DEFENSE EVASION	Virtualization/Sandbox Evasion::System Checks [T1497.001]
MBC Objective	MBC Behavior
ANTI-BEHAVIORAL ANALYSIS	Virtual Machine Detection [B0009]
Capability	Namespace
reference anti-VM strings targeting VMWare resolve function by parsing PE exports (2 matches)	anti-analysis/anti-vm/vm-detection load-code/pe

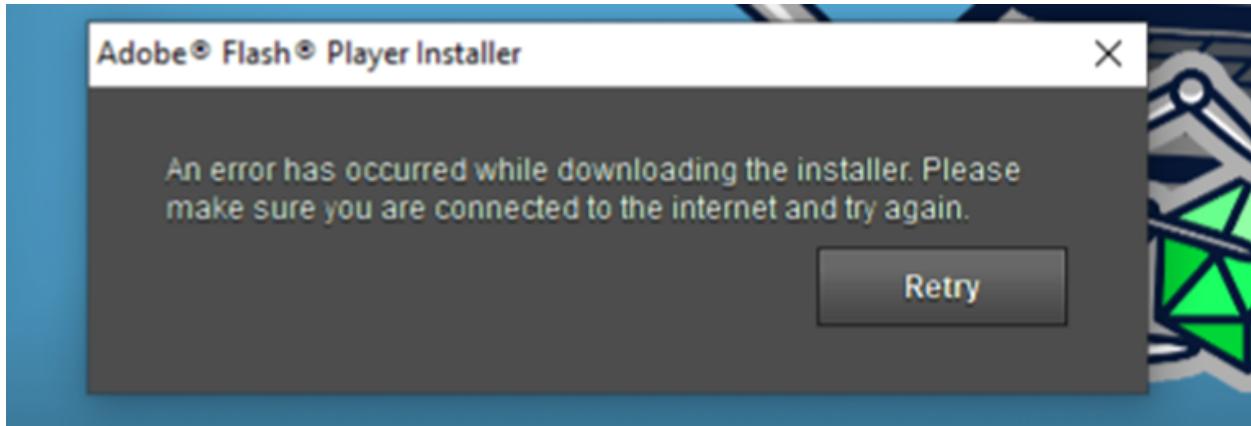
6. Dynamic Analysis

Before the execution of the Zeus Banking Trojan, InetSim was established on the REMnux virtual environment. In the Flare virtual environment, the Remnux IP address, ‘10.10.37.101,’ was entered into the Firefox browser to retrieve the INetSim default HTML page. Upon execution of the malware, a User Account Control (UAC) window for Adobe Flash Player appears on the screen. The malware prevents the analyst from selecting “No” or the exit button on the top right corner of the window, as the UAC prompt reappears. This suggests that the malware manipulated the UAC prompt to force the user to select the “Yes” button. Once the “Yes” button has been chosen, the malware is executed, and immediately an Adobe Flash Player Installer window appears with an error message stating, “An error has occurred while downloading the installer. Please make sure you are connected to the internet and try again.” There is a Retry button, but clicking it makes the prompt reappear. However, despite exiting the error message, the malware still executes in the background and deletes itself from the desktop.

Activities Terminal Apr 9 19:02 remnux@remnux: ~

```
remnux@remnux:~/Downloads$ cd ~
remnux@remnux:~$ inetsim
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg
Using log directory: /var/log/inetsim/
Using data directory: /var/lib/inetsim/
Using report directory: /var/log/inetsim/report/
Using configuration file: /etc/inetsim/inetsim.conf
Parsing configuration file.
Configuration file parsed successfully.
== INetSim main process started (PID 193283) ==
Session ID: 193283
Listening on: 10.10.37.101
Real Date/Time: 2025-04-09 19:02:01
Fake Date/Time: 2025-04-09 19:02:01 (Delta: 0 seconds)
Forking services...
* smtp_25_tcp - started (PID 193289)
* pop3_110_tcp - started (PID 193291)
* pop3s_995_tcp - started (PID 193292)
* https_443_tcp - started (PID 193288)
* http_80_tcp - started (PID 193287)
* smtps_465_tcp - started (PID 193290)
* ftps_990_tcp - started (PID 193294)
* ftp_21_tcp - started (PID 193293)
done.
Simulation running.
```





6.1) Network Traffic Analysis

```
Capturing from Ethernet 2
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter... - Ctrl+Shift+F
No. Time Source Destination Protocol Length Info
1 0.000000 10.10.37.100 10.10.37.100 DNS 73 Standard query 0x3333 A 1.maxmind.com
2 4.033771 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
3 4.033814 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
4 4.033820 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
5 4.038020 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
6 4.038146 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
7 4.038256 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
8 4.038273 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
9 4.192254 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
10 7.550615 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
11 7.550675 10.10.37.100 85.114.128.127 DNS 62 Dynamic update 0x350b!Malformed Packet
12 7.673461 10.10.37.100 10.10.37.250 DNS 85 Standard query 0x7f39 A fpdfdownload.macromedia.com
13 7.673810 10.10.37.250 10.10.37.100 DNS 85 Standard query response 0x7f39 A fpdfdownload.macromedia.com
14 12.809465 ProxoxyServe_48:ca1a ProxoxyServe_48:ca1a ARP 48 10.10.37.100 15. at bcc:24:11:c9:79:b3
15 12.809465 ProxoxyServe_48:ca1a ProxoxyServe_48:ca1a ARP 48 10.10.37.100 15. at bcc:24:11:c9:79:b3
16 21.782782 10.10.37.100 10.10.37.250 DNS 85 Standard query 0x15c4 A fpdfdownload.macromedia.com
17 21.782782 10.10.37.100 10.10.37.250 DNS 85 Standard query response 0x15c4 Refused A fpdfdownload.macromedia.com
18 26.392890 ProxoxyServe_48:ca1a ProxoxyServe_48:ca1a ARP 48 10.10.37.100 15. at bcc:24:11:c9:79:b3
19 26.392890 ProxoxyServe_48:ca1a ProxoxyServe_48:ca1a ARP 48 10.10.37.100 15. at bcc:24:11:c9:79:b3
20 26.709578 10.10.37.100 10.10.37.250 DNS 85 Standard query 0x0435 A fpdfdownload.macromedia.com
21 26.709578 10.10.37.100 10.10.37.250 DNS 85 Standard query response 0x0435 Refused A fpdfdownload.macromedia.com
22 26.711302 10.10.37.100 10.10.37.250 DNS 86 Standard query 0xb276 Wpad cyberrange.page.local
23 26.711537 10.10.37.250 10.10.37.100 DNS 86 Standard query response 0xb276 Refused A wpad.cyberrange.page.local

> Frame 1: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface \Device\NPF_{A56F09CD-8876-4D0B-BB84-D94DE5AB20F8}, h 0000 bc 24 11 48 ca 1a bc 24 11 c9 f9 b3 08 45 00 . $..H.. $... E
  Ethernet II Src: ProxoxyServe_48:c9:f9:b3 (bc:24:11:c9:f9:b3) Dst: ProxoxyServe_48:ca1a (bc:24:11:48:ca1a)
  Internet Protocol Version 4, Src: 10.10.37.100, Dst: 10.10.37.250
  User Datagram Protocol Src Port: 60492, Dst Port: 53
  > Domain Name System (query)
```

6.1.1) DNS Queries

To determine if the malware communicates with command-and-control (C2) servers or if there is any suspicious network activity, the network traffic is examined in Wireshark. The captured traffic shows that there were a few DNS requests where the malware attempted to resolve to the domain `fpdownload.macromedia.com`. Through further research, it has been discovered that this domain is associated with legitimate Adobe Flash Player updates. This could have been the malware's attempt to test if the system has internet access.

6.1.2) HTTP Conversations

No HTTP conversations were established or captured during the malware's execution.

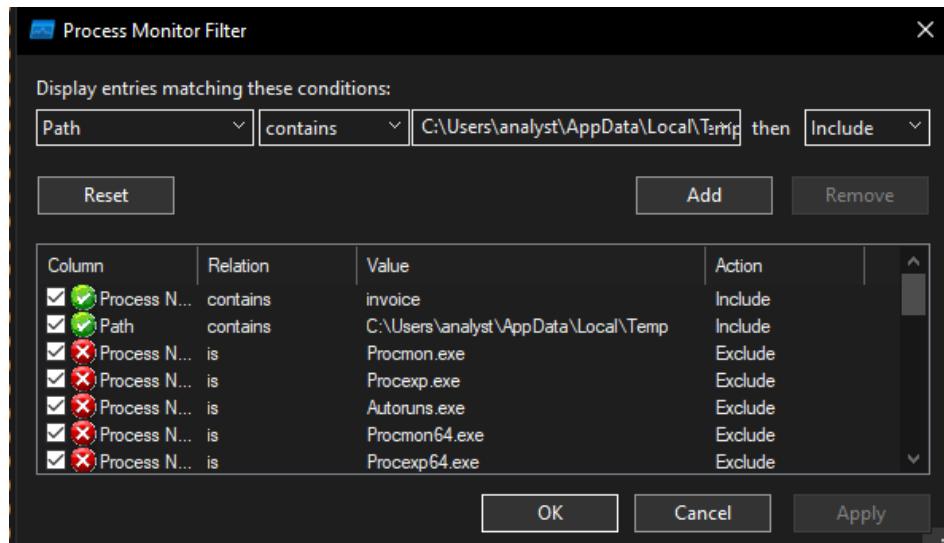
6.1.3) TCP/UDP Communication

There is no presence of TCP or UDP communication initiated by the malware. However, as expected, the DNS queries were transmitted over UDP port 53.

6.2) File operations (files read and written)

Using Process Monitor (Procmon), a set of filters is applied to narrow down the results:

- **Process Name contains 'invoice' then Include** - This ensures that operations performed by invoice_2318362983713_823931342io.pdf.exe are only logged.
- **Path contains C:\Users\analyst\AppData\Local\Temp then Include** - This captures if any operations were performed in the specified file path.



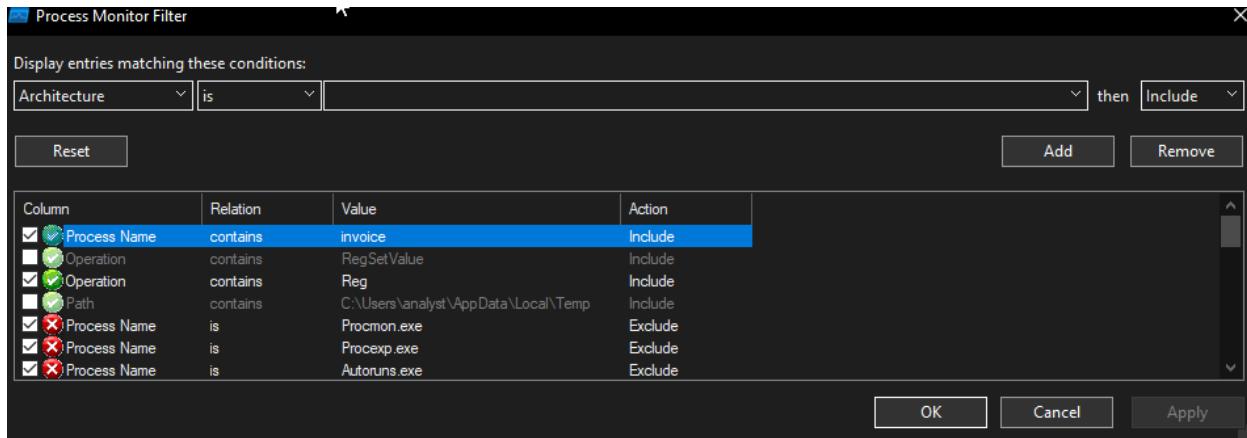
It was observed that the malware had successfully created and written to a DLL file named msimg32.dll in the file path C:\Users\analyst\AppData\Local\Temp. Three operations were conducted in this file path: CreateFile, WriteFile, and CloseFile. The malware may have dropped a suspicious payload in the Temp directory, as it is common for applications to store temporary files, making it difficult to distinguish between malicious and legitimate files.

In the Detail column of the CreateFile operation, it says: Desired Access: Append Data/Add Subdirectory/Create Pipe Instance, Write Attributes, Write DAC, Write Owner, Synchronize, Disposition: OverwriteIf, Options: . It is suspected that the malware intended to create the DLL file to append data, add a subdirectory, and set file attributes and permissions. If the file already existed, it should be overwritten.

Time ..	Process Name	PID	Operation	Path	Result	Detail
7:58.2	invoice_2318362.. 11880	11880	CreateFile	C:\Users\analyst\AppData\Local\Temp\msimg32.dll	SUCCESS	Desired Access: Append Data/Add Subdirectory/Create Pipe Instance, Write Attributes, Write DAC, Write Owner, Synchronize, Disposition: OverwriteIf, Options: .
7:58.2	invoice_2318362.. 11880	11880	WriteFile	C:\Users\analyst\AppData\Local\Temp\msimg32.dll	SUCCESS	Offset: -1, Length: 252,920, Priority: Normal
7:58.2	invoice_2318362.. 11880	11880	CloseFile	C:\Users\analyst\AppData\Local\Temp\msimg32.dll	SUCCESS	

An additional filter was configured to inspect registry activity:

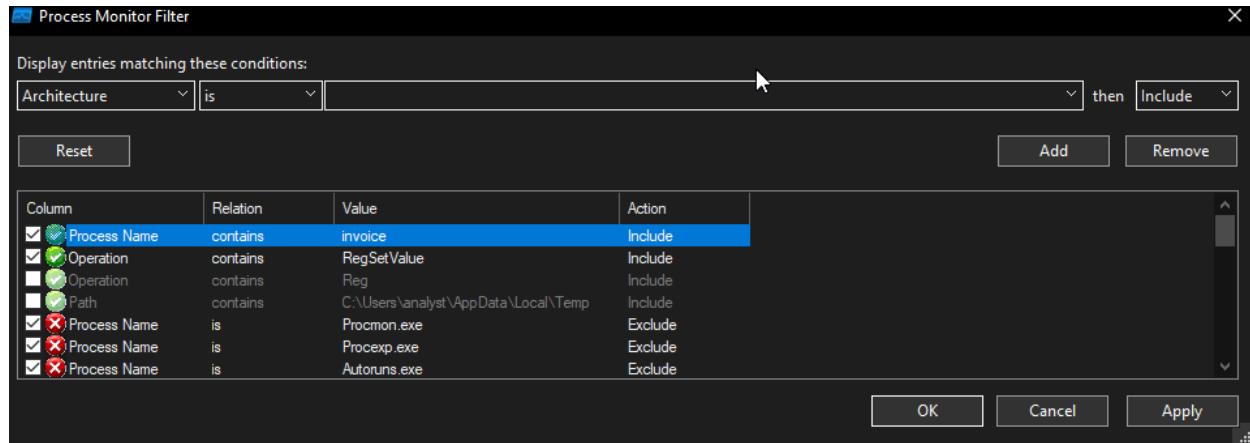
Operation contains Reg then Include - Provide the events that contain the word “Reg”.



Procmon captured the registry-related events with operations RegOpenKey, RegSetValue, RegQueryValue, RegSetInfoKey, and RegCloseKey. The malware interacted with multiple registry keys, primarily under the HKLM\SYSTEM and the HKCU\SOFTWARE hives.

An additional filter was applied to narrow the search even further:

Operation contains RegSetValue, then Include - Show events of any registry activity, including creating or changing a new registry value.



The output from the applied filter shows that the malware wrote specific values to several keys in the Registry. An event that was of interest was the malware setting a value under HKCU\Software\Microsoft\Windows\CurrentVersion\Run\Google Update. This suggests that the malware is intended to create a persistence mechanism, such that if a value is set under Run, then the malware will automatically launch the malicious payload every time the user logs into the system. This malicious payload could be disguised as a Google update.

Time...	Process Name	Op...	Path	Result	Detail	
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Run\Google Update	SUCCESS	Type: REG_SZ, Length: 330, Data: "C:\Users\yanlant\AppData\Local\Google\Desktop\Install\0eb42544fa7-1896-03e\eadbb02a37a\(\进程名)\G...
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ZoneMapInternet	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ZoneMapUAC\UnkHost	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoProtect	SUCCESS	Type: REG_DWORD, Length: 4, Data: 0
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ZoneMapWeb	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ZoneMapUAC\UnkHost	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoProtect	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_U\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ZoneMapWeb	SUCCESS	Type: REG_DWORD, Length: 4, Data: 1
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_M\Software\Microsoft\Windows\NT\CurrentVersion\Notifications\{Data\180A73A3\83C1C75}	SUCCESS	Type: REG_BINARY, Length: 334, Data: FE 16 00 00 00 00 00 04 00 04 01 02 07 00
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_M\Software\Microsoft\Windows\NT\CurrentVersion\Notifications\{Data\418A73A3\83C1C75}	SUCCESS	Type: REG_BINARY, Length: 334, Data: B8 15 00 00 00 00 00 04 00 04 01 01 00 00
7:58...	【进程】 2118362...	11880	RegSetValue	HKEY_M\Software\Microsoft\Windows\NT\CurrentVersion\Notifications\{Data\418A73A3\83C1C75}	SUCCESS	Type: REG_BINARY, Length: 358, Data: B9 15 00 00 00 00 04 00 04 01 01 00 00

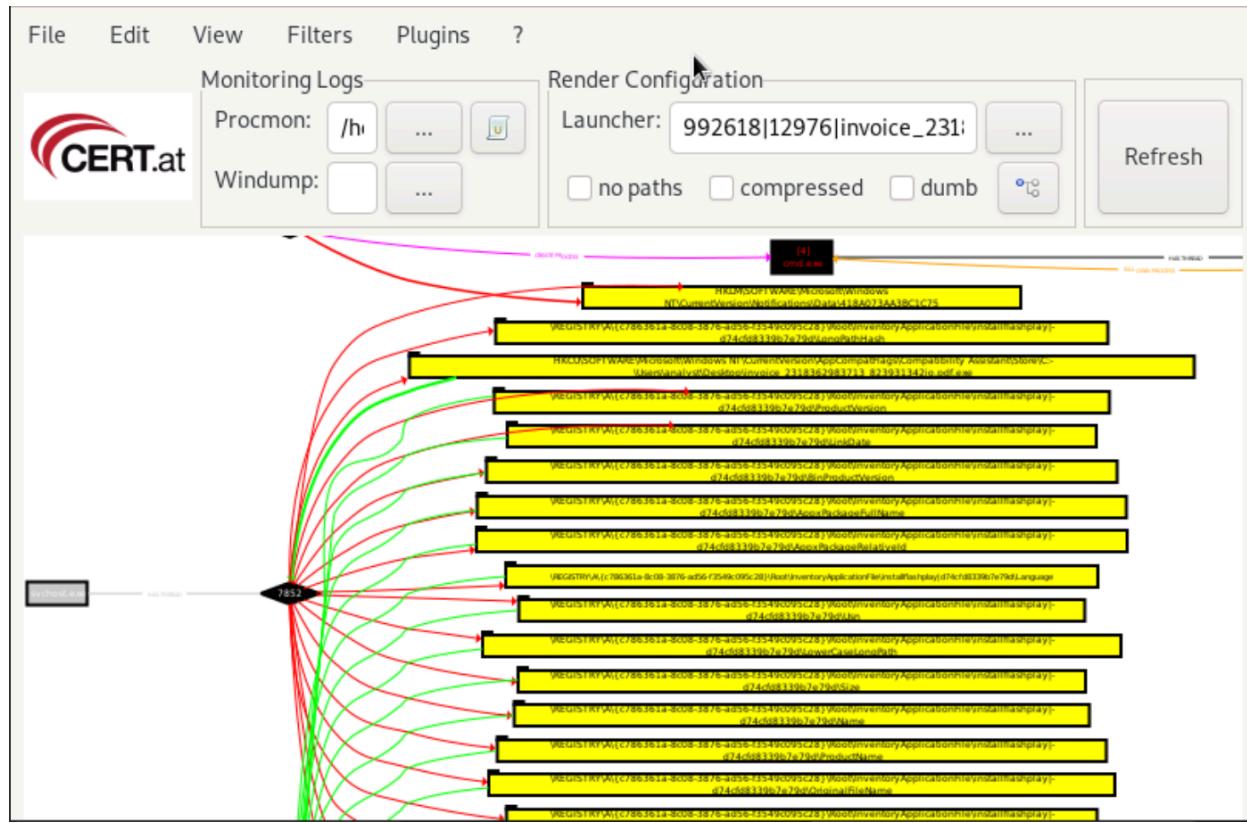
6.3) Services/Processes started

By analyzing Procmon's Process Tree feature, it was seen that the malware had executed the payload `InstallFlashPlayer.exe`, which subsequently launched `cmd.exe` and `Conhost.exe`. No command prompt window appeared on the screen, suggesting that hidden command-line instructions or operations were being performed in the background.

	<p>Adobe® Flash® Player Installer/Uninstaller 11.0 r1 Windows Command Processor Console Window Host Windows Command Processor Console Window Host Notepad The Interactive Disassembler</p>
<p>Description: Console Window Host Company: Microsoft Corporation Path: C:\Windows\System32\Conhost.exe Command: \??\C:\Windows\system32\conhost.exe 0xffffffff -ForceV1 User: DESKTOP-Q7PFT5Q\analyst PID: 5900 Started: 4/26/2025 7:59:40 PM</p>	

The Procmon log data was saved as a CSV file and was uploaded to procdot on the Remnux virtual environment. The procdot graph shows the initial malware execution of invoice_2318362983713_823931342io.pdf.exe and the malware spawning multiple child processes, which initiated various registry activities. InstallFlashPlayer.exe was launched, which potentially acted as a fake Flash installer to deceive users while malicious activity was being performed in the background. The cmd.exe process was also launched, as the window was not displayed, which suggests that command-line activity was potentially being performed discreetly. The registry key HKCU\Software\Microsoft\Windows\CurrentVersion\Run\Google Update is also identified in the graph, where the malware attempted to maintain persistence by relaunching itself when the user logged in to their system.



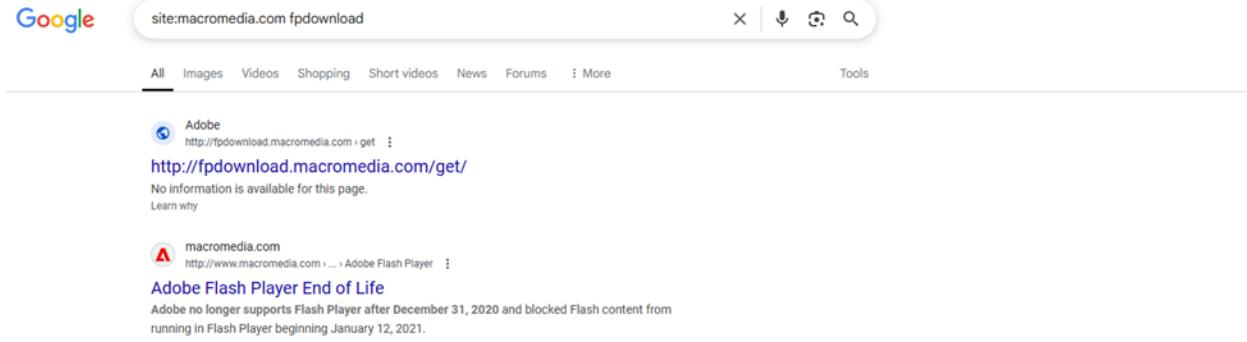


7. Supporting Data

In the file path C:\Users\analyst\AppData\Local\Temp\, the InstallFlashPlayer.exe executable is present and greyed out.

C:\Users\analyst\AppData\Local\Temp				
File	Home	Share	View	
C:\Users\analyst\AppData\Local\Temp				
Quick access	Name	Date modified	Type	Size
Desktop	0mwm2ntq.agi	4/11/2025 6:05 PM	File folder	
Downloads	03vnpe24.duf	4/26/2025 9:41 AM	File folder	
Documents	clink	4/15/2025 3:34 PM	File folder	
Pictures	ConEmu	4/15/2025 3:43 PM	File folder	
hello world	eiopoxx4.osh	4/22/2025 6:05 PM	File folder	
Music	eo1lt3xy.uex	4/10/2025 6:05 PM	File folder	
my_oh_my	ffwxnhzs.tcs	4/8/2025 6:05 PM	File folder	
pub	fl5iwqwk.wux	4/11/2025 6:05 PM	File folder	
This PC	h0jouka3.hsl	4/19/2025 6:05 PM	File folder	
3D Objects	hrmve3oy.orm	4/7/2025 6:05 PM	File folder	
Desktop	hsprefdata_analyst	4/26/2025 9:36 AM	File folder	
Documents	NuGetScratch	3/31/2025 7:06 PM	File folder	
Downloads	ukykyche3.q5t	4/9/2025 6:05 PM	File folder	
Music	vt_cache	4/12/2025 12:07 AM	File folder	
Pictures	4DC6.tmp	4/26/2025 7:58 PM	TMP File	0 KB
Videos	dd_BackgroundDownload_202504260940...	4/26/2025 9:44 AM	Text Document	1,567 KB
Local Disk (C:)	dd_BackgroundDownload_202504260940...	4/26/2025 9:44 AM	Text Document	1,437 KB
Network	dd_BackgroundDownload_202504260940...	4/26/2025 9:44 AM	Text Document	5 KB
	dd_BackgroundDownload_202504260940...	4/26/2025 9:44 AM	Text Document	4 KB
	dd_BackgroundDownload_202504260940...	4/26/2025 9:44 AM	Text Document	0 KB
	dd_BackgroundDownload_202504260940...	4/26/2025 9:44 AM	Text Document	0 KB
	dd_BackgroundDownload_202504261037...	4/26/2025 10:37 AM	Text Document	0 KB
	dd_BackgroundDownload_202504261100...	4/26/2025 11:00 AM	Text Document	0 KB
	dd_BackgroundDownload_202504261136...	4/26/2025 11:36 AM	Text Document	0 KB
	dd_BackgroundDownload_202504261917...	4/26/2025 7:17 PM	Text Document	0 KB
	InstallFlashPlayer.exe	4/26/2025 7:58 PM	Application	88 KB
	Procmon64.exe	4/26/2025 7:57 PM	Application	2,093 KB
	StructuredQuery.log	4/26/2025 10:32 AM	Text Document	220 KB

VirusTotal analysis of the domain fpdownload.macromedia.com shows it has been flagged suspicious for the engine Sucuri SiteCheck, while the other 93 vendors consider it clean. This was an old, legitimate domain that was created 32 years ago and is tied to Adobe. However, its appearance in the malware may indicate an attempt to blend in with the network traffic to check internet connectivity.



8. Indicators of Compromise

These indicators of compromise were extracted from the analysis of the Zeus Banking Trojan malware:

- invoice_2318362983713_823931342io.pdf.exe
- msimg32.dll
- C:\Users\analyst\AppData\Local\Temp\msimg32.dll
- HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\Google Update
- HKLM\System\CurrentControlSet\Control\Session Manager
- fpdownload.macromedia.com
- GetTickCount()
- WriteFile()
- WinExec()
- 85.114.128.127
- 10.10.37.250
- MD5: ea039a854d20d7734c5add48f1a51c34
- SHA-1: 9615dca4c0e46b8a39de5428af7db060399230b2
- SHA-256: 69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169
- SSDEEP:
6144:Tz/LBBTHT+7oEf2ZstxQMSGToLoOhD2saLsW8fsmFBkObjD:PLBdy7FpQMIToThD+sW8fs
mP7bj fuzzy)

9. Recommendations

It is recommended that if a system is suspected or confirmed to be infected with the Zeus banking trojan, then users, administrators, and organizations should consider implementing these best practices to strengthen future defenses.

- Isolate affected systems from the network to prevent lateral movement, C2 communication, and credential theft.
- Scan for additional payloads and persistence mechanisms by employing endpoint detection tools.
- Assume all user and domain credentials are compromised and enforce immediate password changes and a strong password policy.
- Reimage the system from a trusted baseline.
- Update antivirus and EDR solutions to ensure definitions and signatures are current and enabled.
- Keep operating system patches up-to-date.
- Enable multi-factor authentication (MFA) on sensitive systems used for online banking.
- Apply the principle of least privilege and restrict administrative privileges.
- Update firewalls, apply DNS filters, and configure proxies to block known IOCs: IPs, hashes, and domains.
- Deploy EDR and SIEM to detect suspicious process creation and autorun registry keys.
- Conduct user awareness training to educate employees on phishing, malicious attachments, and fake update scams that could be disguised as a legitimate update.
- Segment the network using VLANs and firewall rules to separate workstations and servers from financial systems.
- Apply application whitelisting to allow only trusted applications to run.
- Conduct regular backups for critical systems.