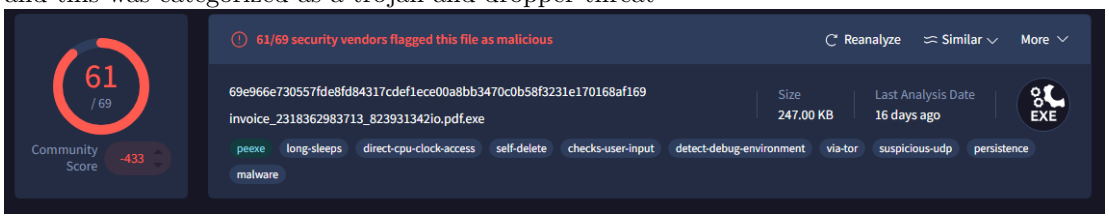
Zeus Trojan Analysis

Jackson McCullough

August 2025

1 Fingerprint

Using **VirusTotal** for fingerprinting, we can see how often this file was flagged, and this was categorized as a trojan and dropper threat



This gives us the hash of the files, size, and scoring

1.1 Hashes

The hashes given by VirusTotal are:
MD5: ea039a854d20d7734c5add48f1a51c34
SHA-1: 9615dca4c0e46b8a39de5428af7db060399230b2
SHA-256: 69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169

When analyzing the file in **pestudio**, the output for the hashes are identical, which means that the file we are analyzing is, in fact, the Zeus Trojan, and we are working with the same exact sample (this is mostly useful when working with a team and the malware is static).

1.2 File ID

Filename: invoice_2318362983713_823931342io.pdf.exe

We can see that it is saying "invoice", possibly trying to impersonate an individual or company, targeting finances, and trying to mask itself as a pdf file by adding the ".pdf" at the end.

2 Static Analysis

Note: Embedded url of “corect.com” in file. Notice misspelling and hard-coded into file.

2.1 Size

The raw size and virtual size are fairly similar(as displayed), indicating that this is not compressed or packed. Therefore, we should be able to see its core functionality and all it has to offer upfront.

raw-address (end)	0x0000BA00
raw-size (251904 bytes)	0x0000B600 (46592 bytes)
virtual-address (begin)	0x00001000
virtual-address (end)	0x0000C571
virtual-size (250379 bytes)	0x0000B571 (46449 bytes)

2.2 Flagged Strings

Type	Len	Offset	Flag	String
ascii	7	0x0001EA40	x	WinExec
ascii	9	0x0001EA64	x	VkKeyScan
ascii	9	0x0001E91C	x	WriteFile
ascii	12	0x0001E9FC	x	FindNextFile
ascii	13	0x0001E878	x	GlobalAddAtom
ascii	14	0x0001E90A	x	VirtualQueryEx
ascii	16	0x0001EB4C	x	GetAsyncKeyState
ascii	16	0x0001EB18	x	GetClipboardData
ascii	16	0x0001E8E6	x	GetCurrentThread
ascii	17	0x0001EA72	x	GetClipboardOwner
ascii	18	0x0001EBDC	x	DdeQueryNextServer
ascii	19	0x0001E71E	x	PathRenameExtension
ascii	20	0x0001EB8E	x	EnumClipboardFormats
ascii	22	0x0001E982	x	GetEnvironmentVariable
ascii	22	0x0001E832	x	GetEnvironmentVariable
ascii	24	0x0001EAC2	x	AllowSetForegroundWindow
ascii	25	0x0001E94A	x	GetConsoleAliasExesLength

2.3 Suspicious Unflagged Strings

Type	Len	Offset	Flag	String
ascii	10	0x000311F6	-	corect.com
ascii	10	0x0001EC42	-	USER32.dll
ascii	10	0x0001EC00	-	DeleteMenu
ascii	10	0x0001EBF2	-	LoadBitmap
ascii	10	0x0001EB80	-	GetCapture
ascii	10	0x0001EADE	-	AppendMenu
ascii	10	0x0001E9CA	-	LocalAlloc
ascii	10	0x0001E736	-	PathIsRoot
ascii	11	0x0003180C	-	Dumpcotsavo
ascii	11	0x0003162E	-	BardHolyawe
ascii	11	0x0001EB72	-	InflateRect
ascii	11	0x0001EAFA	-	GetSysColor
ascii	11	0x0001EAEC	-	GetCaretPos
ascii	11	0x0001E86A	-	FreeLibrary
ascii	11	0x0001E824	-	LocalUnlock
ascii	11	0x0001E816	-	SHLWAPI.dll
ascii	11	0x0001E7D8	-	IsCharSpace
ascii	11	0x0001E758	-	PathCombine
ascii	12	0x0001EAB2	-	UpdateWindow
ascii	12	0x0001EA54	-	KERNEL32.dll
ascii	12	0x0001E928	-	GetDriveType

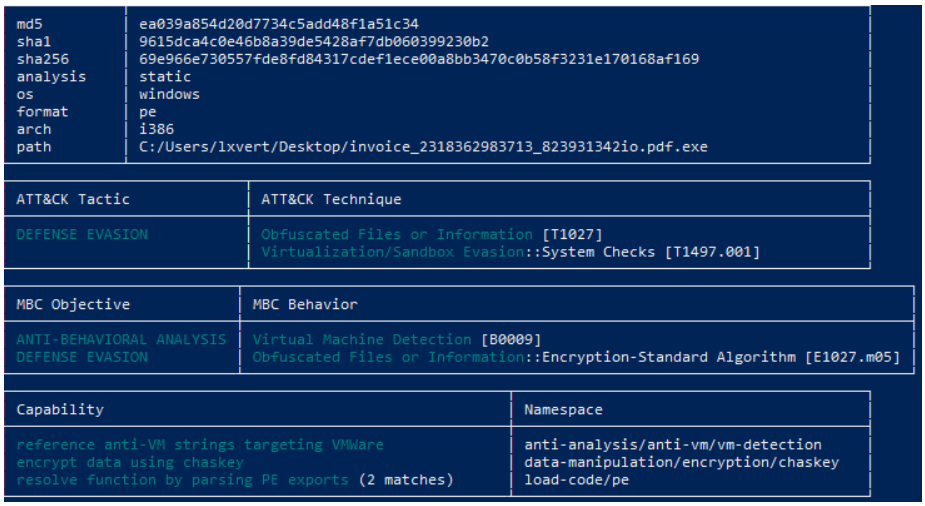
Using **Floss**, no other suspicious strings were noticed, whether it be URL's hardcoded, dll's, etc.

2.4 Imported Libraries

- `KERNEL32.dll`
- `SHLWAPI.dll`
- `USER32.dll`

2.5 ATT&CK/MBC

Using **Capa**, this malware shows that it was mostly working for Evasion & Obfuscation:



md5	ea039a854d20d7734c5add48f1a51c34
sha1	9615dca4c0e46b8a39de5428af7db060399230b2
sha256	69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169
analysis	static
os	windows
format	pe
arch	i386
path	C:/Users/1xvert/Desktop/invoice_2318362983713_823931342io.pdf.exe

ATT&CK Tactic	ATT&CK Technique
DEFENSE EVASION	Obfuscated Files or Information [T1027] Virtualization/Sandbox Evasion::System Checks [T1497.001]

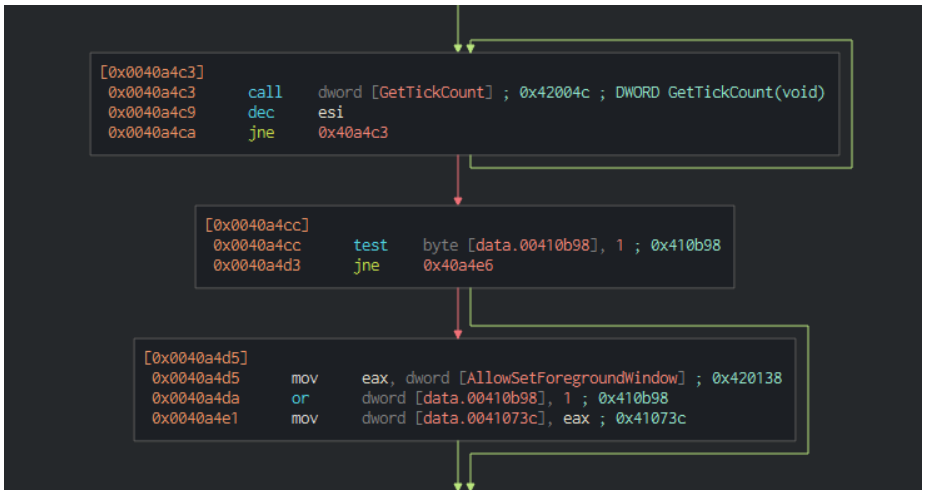
MBC Objective	MBC Behavior
ANTI-BEHAVIORAL ANALYSIS	Virtual Machine Detection [B0009]
DEFENSE EVASION	Obfuscated Files or Information::Encryption-Standard Algorithm [E1027.m05]

Capability	Namespace
reference anti-VM strings targeting VMWare	anti-analysis/anti-vm/vm-detection
encrypt data using chaskey	data-manipulation/encryption/chaskey
resolve function by parsing PE exports (2 matches)	load-code/pe

Update: corect.com yielded no suspicious results. **VirusTotal** showed zero vendors flagged this site, and **Wayback machine** showed it to be a news source during the time of peak infections.

2.6 Assembly

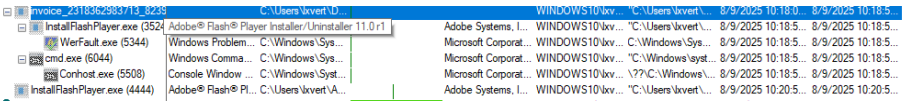
The assembly code (analyzed using the **cutter**) shows that some of the API calls are either looping back to themselves or jumping to other code blocks, continuously looping and calling the tick count while simultaneously decrementing ESI. Then it tests a flag to see if the foreground window permission has already been granted. If it is, it jumps that block and proceeds; if not, it goes through that block that allows it.



3 Dynamic Analysis

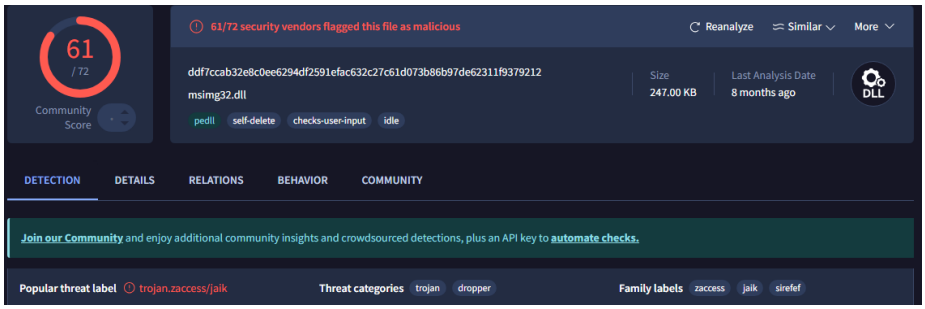
3.1 Processes

Using **Process Monitor**, we could see multiple programs and files were installed after executing the program. The program also deleted itself from the desktop once it was executed.



Invoice also had main priority of creating keys and general key interaction, while the cmd and conhost executables were mainly interacting with files and queries. Also installed an adobe flash player executable along with a new dll.

Looking into the dll and flash player, it seems that flash player was a legitimate executable (checked with Sec trails and VirusTotal, using both URL and IP gotten using **nslookup** command), possibly downloaded from and HTTP request that malware sent when executed. The dll (via **VirusTotal**) did note that the dll downloaded was flagged as malicious, and declared a trojan/dropper, assuming it is tied in some way with the actual Zeus trojan involved. It does seem, however, that this trojan has established its persistence within the google updates. It has infiltrated the binary so that whenever google chrome updates, the invoice file will be executed again.



3.2 Network Analysis

Inetsim was set up along with wireshark to analyze the packets sent on the network. Most seemed normal besides two HTTP requests (as stated before). The requests were the host and a URL called fpdownload.macromedia.com, which was a legitimate site with no malicious files which took you to a help site for adobe, assuming that this URL interacted in some way with the adobe installation. Does not seem to have any real C2 server communication involved.

```
GET /get/flashplayer/update/current/install/install_all_win_cab_64_ax_sgn.z HTTP/1.1
User-Agent: Flash Player Seed/3.0
Host: fpdownload.macromedia.com
Cache-Control: no-cache


HTTP/1.1 200 OK
Server: INetSim HTTP Server
Date: Sat, 09 Aug 2025 14:46:43 GMT
Content-Type: text/html
Connection: Close
Content-Length: 258

<html>
  <head>
    <title>INetSim default HTML page</title>
  </head>
  <body>
    <p></p>
    <p align="center">This is the default HTML page for INetSim HTTP server fake mode.</p
>
    <p align="center">This file is an HTML document.</p>
  </body>
</html>
```

This is the TCP stream of the HTTP request taken from Wireshark.

4 YARA file

YARA file will be attached with document