Malware Analysis Report – Zeus Trojan(ZBot)

Contents: -

Introduction

WHY?

The **Zeus Trojan (ZBot)** is one of the most significant and well-documented malware families in cybersecurity, making it an ideal subject for research and analysis. It was chosen because of its **global impact, technical sophistication, and long-lasting influence** on modern malware. Zeus successfully infected millions of systems worldwide, targeting banks, corporations, and government agencies, resulting in billions of dollars in financial losses. Its design demonstrates advanced techniques such as **form grabbing, man-in-the-browser attacks, and encrypted command-and-control (C2) communication**, which remain relevant in today's threat landscape. Furthermore, the **leak of Zeus's source code in 2011** allowed researchers and cybercriminals alike to create numerous powerful variants, including **Gameover Zeus** and Citadel, extending its legacy. By studying Zeus, one can better understand the evolution of banking Trojans, the challenges of detection, and the development of resilient defense mechanisms. Thus, Zeus provides a comprehensive case study in both malware analysis and cybercrime evolution.

WHAT?

The **Zeus Trojan (ZBot)** is a notorious piece of malware first discovered in **2007**, primarily designed to **steal financial information** such as online banking credentials, credit card data, and login details. Spread mainly through **phishing emails, malicious downloads, and exploit kits**, Zeus silently installs itself on a victim's computer and hides by injecting into legitimate processes like explorer.exe. Once active, it employs techniques such as **keylogging, form grabbing, and man-in-the-browser (MitB) attacks** to capture sensitive data before it is encrypted for secure transmission. A critical feature of Zeus is its **command-and-control (C2) infrastructure**, which allows infected systems to download configuration files, receive instructions, and exfiltrate stolen information. In 2011, Zeus's **source code was leaked**, leading to several powerful variants like **Gameover Zeus, Citadel, and Ice IX**. Despite global takedowns, Zeus remains one of the most influential banking Trojans, shaping the evolution of modern financial malware.

HOW?

The **Zeus Trojan (Zbot)** operates as a sophisticated banking malware designed to steal sensitive information such as online banking credentials, credit card details, and authentication tokens. Its infection usually begins through phishing emails, malicious attachments, or exploit kits, which deliver the initial dropper to the victim's system. Once executed, this dropper installs the core Zeus bot and ensures persistence by modifying registry keys or creating scheduled tasks so that it runs automatically after each reboot.

After installation, Zeus injects itself into legitimate Windows processes such as explorer.exe or web browsers, allowing it to hide and operate silently. The malware

then connects to its **command-and-control (C2) server**, typically over HTTP or HTTPS, using encrypted communication that resembles normal web traffic. One of its first tasks is to download an encrypted configuration file from the C2. This file contains critical instructions, including which banks or financial institutions to target, which websites to modify, and which backup servers to use in case the main C2 is unavailable.

When a victim visits a targeted banking website, Zeus activates its **browser hooks** to perform **form grabbing** and **man-in-the-browser (MitB) attacks**. This enables it to capture login credentials and session data directly from the browser before the information is encrypted and transmitted over SSL/TLS. Zeus can also inject malicious HTML or JavaScript into legitimate webpages, tricking users into entering additional information such as PINs or one-time passwords (OTPs).

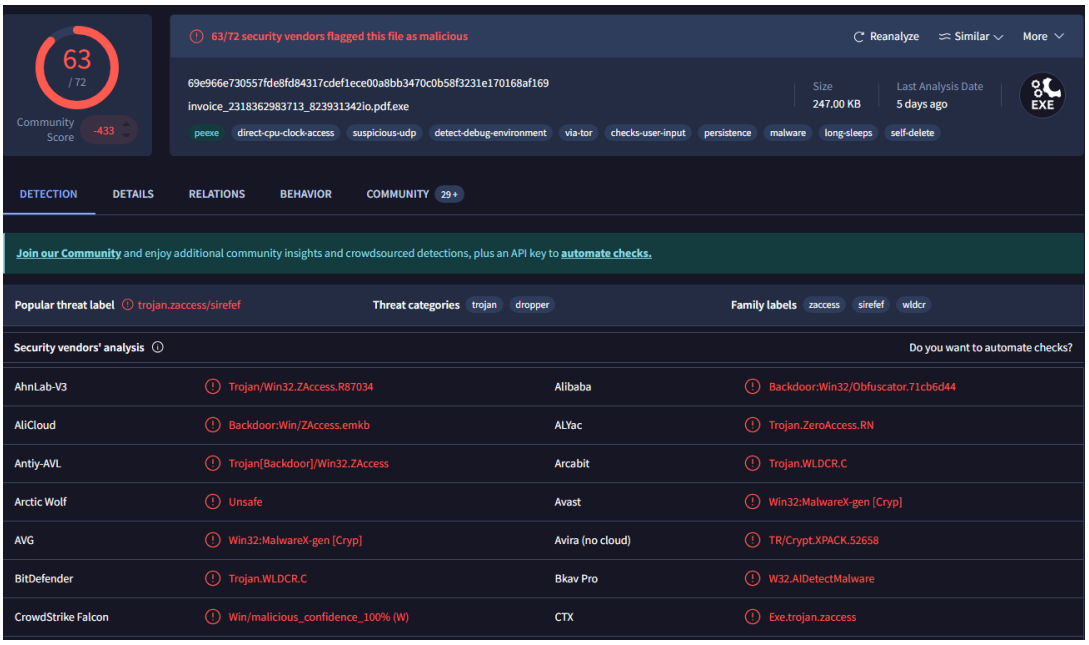
All stolen data is encrypted and sent back to the C2 server, where attackers can either use it for fraudulent transactions or sell it on underground markets. The C2 also allows operators to push updates, download additional malware, or change target lists dynamically. This **modular, stealthy, and resilient design** made Zeus one of the most effective and influential banking Trojans in history.

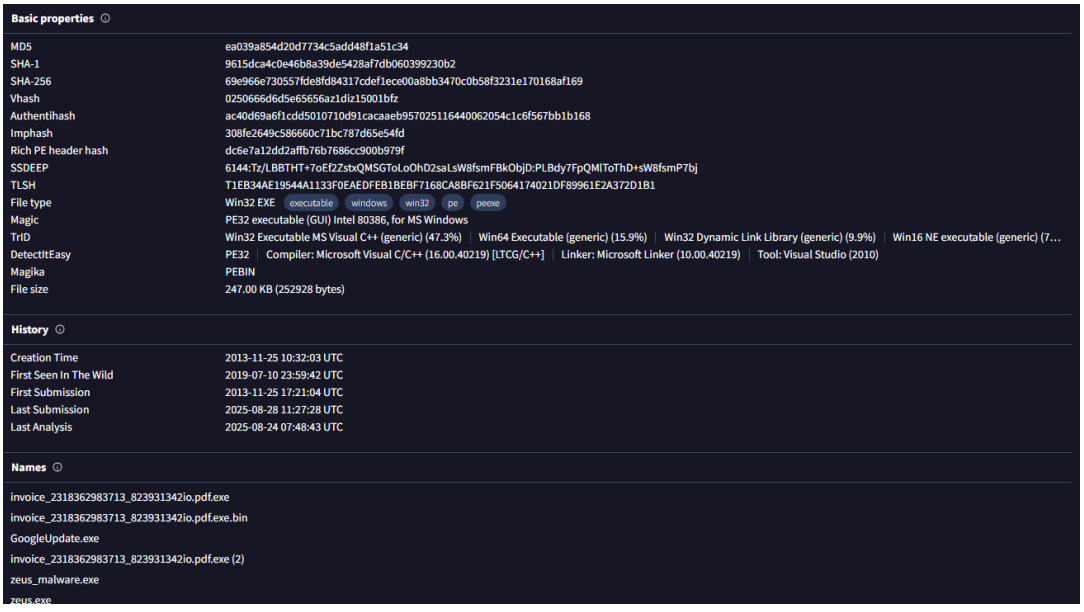
Tools used in the analysis: -

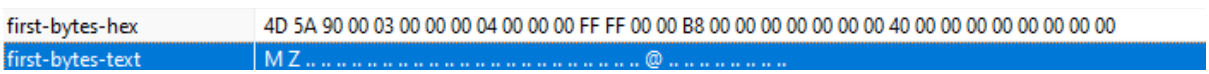
1. Virustotal
2. PEstudio
3. Floss
4. Capa
5. Cutter
6. INetSim
7. Wireshark
8. ProcMon
9. YARA

Static Analysis

1. Fingerprints







2. Basic Static Analysis

Section headers

property	value	value	value	value	value	value
name	.text	.data	.itext	.pdata	.rsrc	.reloc
md5	679FBF23D7317D8207D350B...	73FDAE90C1738941B6AFEC6...	7F89AD170FFEA80A9C7304E...	A8448D1B94E56BC8F80ED85...	B3AF18982AEE2E1B3991523...	37469A130E838CD467FF445...
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	0x00000400	0x0000BA00	0x0001E400	0x0001EE00	0x00036C00	0x0003C600
raw-size (251904 bytes)	0x0000B600 (46592 bytes)	0x00012A00 (76288 bytes)	0x00000A00 (2560 bytes)	0x00017E00 (97792 bytes)	0x00005A00 (23040 bytes)	0x00001600 (5632 bytes)
virtual-address	0x00401000	0x0040D000	0x00420000	0x00421000	0x00439000	0x0043F000
virtual-size (250379 bytes)	0x0000B571 (46449 bytes)	0x000128B1 (75953 bytes)	0x0000084D (2125 bytes)	0x00017CBE (97470 bytes)	0x000058F2 (22770 bytes)	0x000015EC (5612 bytes)
entry-point	**0x0000A3B6**	-	-	-	-	-
characteristics	0x60000020	0xC0000040	0xC0000040	0xE0000020	0x40000040	0x42000040
writable	-	x	x	x	-	-
executable	x	-	-	x	-	-

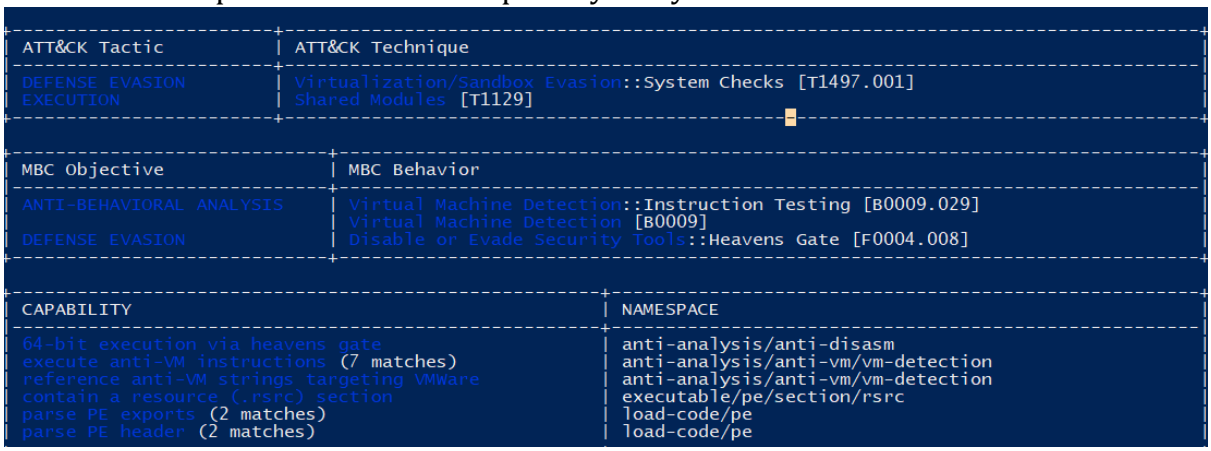
Here '.text' file contains the main code of the malicious software
'.data' contains variables
'.idata' contains imports
'.pdata' and '.rsrc' contains payload data and resources
'.reloc' contains relocatable address

Black Listed Strings

encoding (2)	size (bytes)	file-offset	blacklist (18)	hint (36)	group (13)	value (785)
ascii	24	0x0001EAC2	x	-	windowing	AllowSetForegroundWindow
ascii	10	0x0001EB80	x	-	windowing	GetCapture
ascii	19	0x0001E9D8	x	-	storage	SetCurrentDirectory
ascii	22	0x0001E832	x	-	reckoning	GetEnvironmentVariable
ascii	22	0x0001E982	x	-	reckoning	GetEnvironmentVariable
ascii	9	0x0001EA64	x	-	keyboard-and-mouse	VkKeyScan
ascii	16	0x0001EB4C	x	-	keyboard-and-mouse	GetAsyncKeyState
ascii	19	0x0001E71E	x	-	file	PathRenameExtension
ascii	9	0x0001E91C	x	-	file	WriteFile
ascii	12	0x0001E9FC	x	-	file	FindNextFile
ascii	16	0x0001E8E6	x	-	execution	GetCurrentThread
ascii	7	0x0001EA40	x	-	execution	WinExec
ascii	13	0x0001E878	x	-	data-exchange	GlobalAddAtom
ascii	17	0x0001EA72	x	-	data-exchange	GetClipboardOwner
ascii	16	0x0001EB18	x	-	data-exchange	GetClipboardData
ascii	20	0x0001EB8E	x	-	data-exchange	EnumClipboardFormats
ascii	18	0x0001EBDC	x	-	data-exchange	DdeQueryNextServer
ascii	25	0x0001E94A	x	-	console	GetConsoleAliasExesLength

The **encoding** column shows data type (ASCII), while **size (bytes)** gives string length. **File-offset** indicates the string's location in the binary. **Blacklist** flags suspicious functions. The **group** column categorizes APIs (e.g., windowing, storage, execution, data-exchange). The **value** column lists specific Windows API calls like WriteFile, GetClipboardData, and WinExec. These APIs suggest interaction with files, processes, clipboard, and system environment, common in malware analysis or reverse engineering of potentially malicious software.

Capa malware.exe -> capability analysis of malware file

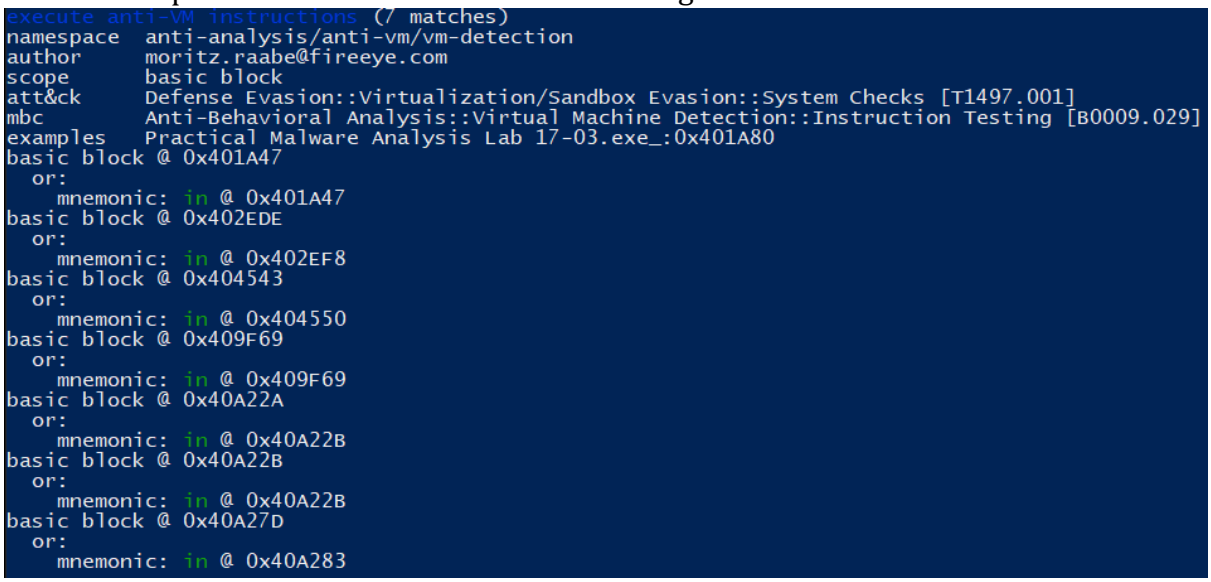


ATT&CK Tactic	ATT&CK Technique
DEFENSE EVASION	Virtualization/Sandbox Evasion::System Checks [T1497.001]
EXECUTION	Shared Modules [T1129]

MBC Objective	MBC Behavior
ANTI-BEHAVIORAL ANALYSIS	Virtual Machine Detection::Instruction Testing [B0009.029]
	Virtual Machine Detection [B0009]
DEFENSE EVASION	Disable or Evade Security Tools::Heavens Gate [F0004.008]

CAPABILITY	NAMESPACE
64-bit execution via heavens gate	anti-analysis/anti-disasm
execute anti-VM instructions (7 matches)	anti-analysis/anti-vm/vm-detection
reference anti-VM strings targeting VMware	anti-analysis/anti-vm/vm-detection
contain a resource (.rsrc) section	executable/pe/section/rsrc
parse PE exports (2 matches)	load-code/pe
parse PE header (2 matches)	load-code/pe

Capa –vv malware.exe -> Address having anti VM instructions



```
execute anti-VM instructions (7 matches)
namespace  anti-analysis/anti-vm/vm-detection
author     moritz.raabe@fireeye.com
scope      basic block
att&ck     Defense Evasion::Virtualization/Sandbox Evasion::System Checks [T1497.001]
mbc        Anti-Behavioral Analysis::Virtual Machine Detection::Instruction Testing [B0009.029]
examples   Practical Malware Analysis Lab 17-03.exe_:0x401A80
basic block @ 0x401A47
  or:
    mnemonic: in @ 0x401A47
basic block @ 0x402EDE
  or:
    mnemonic: in @ 0x402EF8
basic block @ 0x404543
  or:
    mnemonic: in @ 0x404550
basic block @ 0x409F69
  or:
    mnemonic: in @ 0x409F69
basic block @ 0x40A22A
  or:
    mnemonic: in @ 0x40A22B
basic block @ 0x40A22B
  or:
    mnemonic: in @ 0x40A22B
basic block @ 0x40A27D
  or:
    mnemonic: in @ 0x40A283
```

The binary uses **defense evasion** (e.g., virtualization/sandbox checks, Heavens Gate technique) and **execution via shared modules**. It includes **anti-VM instructions**, VMware-specific detection, and system checks to avoid analysis. Capabilities detected: parsing PE headers, PE exports, resources, and switching to 64-bit code via Heavens Gate. The disassembly highlights instructions (e.g., in opcode) at specific offsets, used to detect virtualized environments. These behaviors indicate **anti-analysis and anti-behavioral techniques**, suggesting the malware actively hides from sandboxes and security researchers.

The Zeus sample contains suspicious strings, particularly function calls related to critical Windows DLLs:

1. **SHLWAPI.dll**: This is a Shell Lightweight Utility Library that provides functions for handling strings, paths, and registry entries.
2. **KERNEL32.dll**: This is a critical part of the Windows kernel and provides core system functions such as memory management, input/output operations, and interrupts.

3. **USER32.dll**: It handles user interface components like windows, buttons, and input from keyboards and mice.
4. **KERNEL32.MulDiv**: A specific function in the KERNEL32.dll that performs combined multiplication and division in a single step for accurate calculations without intermediate overflow. Malware often leverages system-level DLLs like these to interact with the operating system in stealthy ways.

Embedded URL- corect.com: Another suspicious indicator is the embedded **corect.com** URL within the binaries. When I examine this URL using VirusTotal, it shows no malicious flags. Further inspection of the website reveals that it's a Romanian news website.

However, the inclusion of an embedded URL like this could point to **Command and Control (C2)** communication, or could serve as a decoy. While this specific URL is harmless, it's common for malware to hide C2 servers in plain sight, using legitimate-looking URLs as a cover for malicious activity.

```
pt,JXvGcq2uEocMa1e661HGNPt4yfh5V1e0JjXXcPwg5mVN/ErD1/YjC9hKg,Z1fU1LfcsDXF3{4
TYCTXxUhIw0LOApT{ZDAxiL11Y
glWP15xcm6:d6zsRUahNiBqz6EvG0R,m3w1wvEQOdqNfKkz56cJwtRxjngq1H/kxZ3G4N3n/
bdgDEjPucBCDWQ9t7nXxJ7JRu3{yUS3TQ55G744iSKCF2WRKk50E3/
XAzUJUmZ9YxpKg[JNWdRwhvRT2mx6vFbL+RmxYZv:nhB6r6px0odw766rPIubE1jL3l7UzTBL4iW
+j18mQ2Bf8g9O2jV12CTuRWlZ5XXu29DSZErqmQdthvFxLDL8RZQsDa{BFY2/r9X/
yWmUFvCtH7FrplTsVpNXtZMuqcOhm7I9QaBUQt[2u7frj/
MczPbOMF{VZr5r0tLC3KX2hgeRRd[8GeJR4Z9MItIpr2PdjQh2ZU9PYuxdTKe+wqdxhbo3N85q5m
KmvuNp,ACY0XluddlcpYW51sgpk+WcGA[NwFPnH1YVy6itI5Jnj/5HRXnUQEKz[Whfu8:scixH4c
y2Se:Ts08IXx:f[YQXr6SdvkZKqiUWJV4ixFGxDEIV3IXCGQHptQQaDnFug9f+nr4V,DRBtb3Yh2
i1KTGz5GvzQ02NVND8YipUQ59VD8DKHkIqgpYQ,G,pjEwMkDNiklY39+b0HW:e:AVdID8Z8D4N0e
corect.com
?
AidsvowsBootFaysGiveCuesmadslallcarlwot@@YGXACMACUPelfOdasbachSlitfogymug@@U
?
HermArcoludeUmpsjiaoTareOhmsLimetumpdentdellAlifboosmy@@YGEACHUtagLOGFONTA@@
?MayoapoddrekheftExedqueyAlkypap@@YGXACUFlatmisscolyHantOldyspy@@UDecoappsSa
```

If I go into that website, I will see that it is a Romanian News website.

Dynamic Analysis

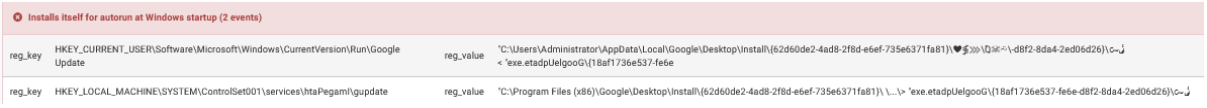
After completing static analysis, I move to **dynamic analysis**, where the malware is executed in a controlled environment to observe its real-time behavior. For this analysis, I also used **Cuckoo Sandbox**, which allows us to monitor system changes, file creation, network connections, and other behavioral indicators.

1. **Attempts to stop active services**

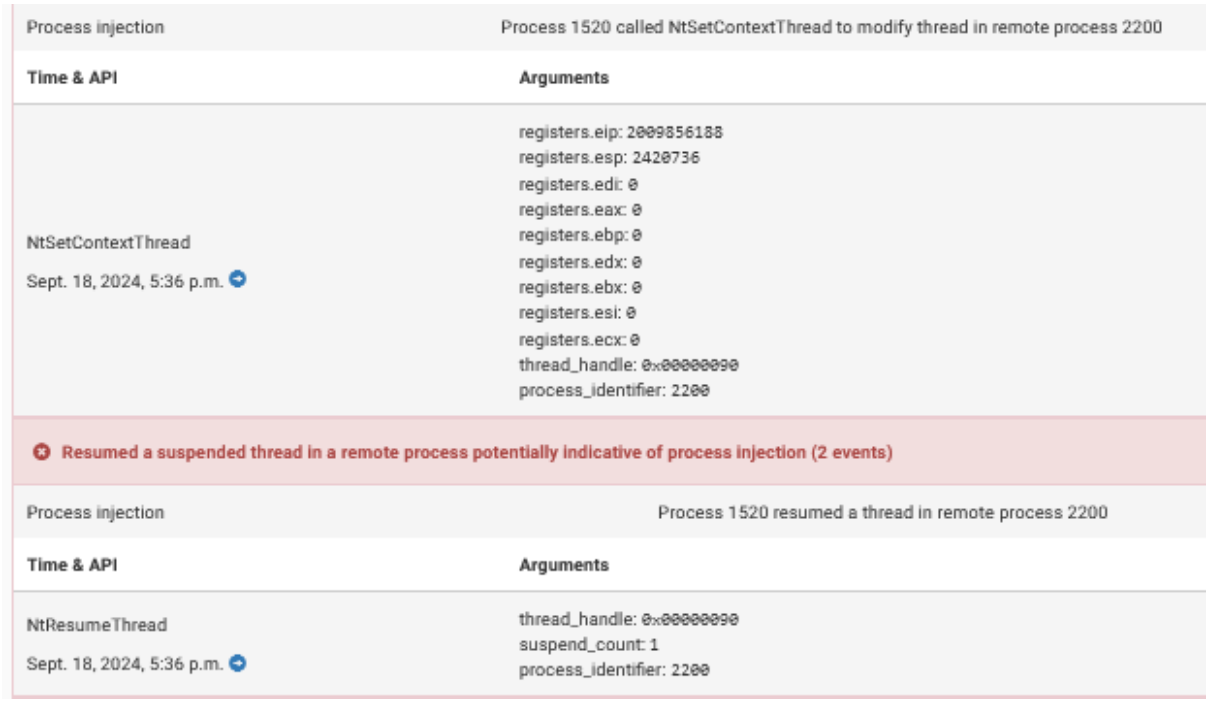
Attempts to stop active services (8 events)

Time & API	Arguments
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce270 service_name: mpssvc control_code: 1
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce388 service_name: SharedAccess control_code: 1
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce270 service_name: RemoteAccess control_code: 1
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce388 service_name: PolicyAgent control_code: 1
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce270 service_name: iphlpsvc control_code: 1
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce388 service_name: wscsvc control_code: 1
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce270 service_name: PcaSvc control_code: 1
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce388 service_name: bfe control_code: 1

2. **Installs itself for AutoRun at Windows startup:** Zeus installs itself to **AutoRun** at system startup, ensuring persistence across reboots. This means that every time the computer is turned on, the malware will execute automatically. This persistence mechanism is crucial for attackers because it guarantees the malware stays active even after system restarts.



1. **Process injections:** Zeus uses **process injection** techniques to embed its malicious code into legitimate system processes, by doing so, it hides its activity under the guise of trusted system processes, making it harder for antivirus software and other monitoring tools to detect it.



1. **Stops Windows services**: One of Zeus's primary actions is to stop certain Windows services that may interfere with its execution. By terminating critical processes, Zeus attempts to weaken the system's defense mechanisms.

2. **Executes processes and injects codes**

Executed a process and injected code into it, probably while unpacking (6 events)

Time & API	Arguments
NtResumeThread Sept. 18, 2024, 5:36 p.m.	thread_handle: 0x000000cc suspend_count: 1 process_identifier: 1520
NtResumeThread Sept. 18, 2024, 5:36 p.m.	thread_handle: 0x00000104 suspend_count: 1 process_identifier: 1520
CreateProcessInternalW Sept. 18, 2024, 5:36 p.m.	thread_identifier: 328 thread_handle: 0x00000090 process_identifier: 2200 current_directory: filepath: C:\Windows\System32\cmd.exe track: 1 command_line: filepath_r: C:\Windows\system32\cmd.exe stack_pivoted: 0 creation_flags: 134217732 (CREATE_NO_WINDOW|CREATE_SUSPENDED) inherit_handles: 0 process_handle: 0x00000094
NtGetContextThread Sept. 18, 2024, 5:36 p.m.	thread_handle: 0x00000090
NtSetContextThread Sept. 18, 2024, 5:36 p.m.	registers.eip: 2009856188 registers.esp: 2420736 registers.edi: 0 registers.eax: 0 registers.ebp: 0 registers.edx: 0 registers.ebx: 0 registers.esi: 0 registers.ecx: 0 thread_handle: 0x00000090 process_identifier: 2200
NtResumeThread Sept. 18, 2024, 5:36 p.m.	thread_handle: 0x00000090 suspend_count: 1 process_identifier: 2200

1. **Dropped files:** Zeus also drops additional files onto the infected system. These files are often used to help maintain persistence or communicate with external servers. In the case of our analysis, the following files were dropped:
    - **invoice_2318362983713_823931342io.pdf.exe**: This is the primary executable that contains the malicious code.
    - **cmd.exe**: Zeus injects code into the command prompt process to execute system-level commands.

These files are critical components of the Zeus infection process, helping it to evade detection and perform malicious activities, such as stealing credentials or exfiltrating data.

Some additional dropped files:

1. Dropped Files

Dropped Files

Name	**38adea3be60e54ab_@** Download Submit file
Filepath	C:\Users\Administrator\AppData\Local\Google\Desktop\Install\{62d60de2-4ad8-2f8d-e6ef-735e6371fa81}\❤⚡»»\Q✉~\@\{18af1736e537-fe6e-d8f2-8da4-2ed06d26}\c~ن
Size	2.0KB
Processes	1520 (**invoice_2318362983713_823931342io.pdf.exe**)
Type	data
MD5	368cc56b35931a5d4b2b8dfa5b134a63
SHA1	d2d90853a305210c17355598f316460de6c26c53
SHA256	38adea3be60e54ab5506ba9bf46ef71eccf86aec893e2f5a012e234c35307172
CRC32	104009CC
ssdeep	None
Yara	None matched
VirusTotal	Search for analysis

Name	**69e966e730557fde_googleupdate.exe** Download Submit file
Filepath	C:\Program Files (x86)\Google\Desktop\Install\{62d60de2-4ad8-2f8d-e6ef-735e6371fa81}\ \...\exe.etadpUelgooG\{18af1736e537-fe6e-d8f2-8da4-2ed06d26}\c~ن
Size	247.0KB
Processes	1520 (**invoice_2318362983713_823931342io.pdf.exe**)
Type	PE32 executable (GUI) Intel 80386, for MS Windows
MD5	ea039a854d20d7734c5add48f1a51c34
SHA1	9615dca4c0e46b8a39de5428af7db060399230b2
SHA256	69e966e730557fde8fd84317cdef1ece00a8bb3470c0b58f3231e170168af169
CRC32	B6012D5E
ssdeep	None
Yara	• keylogger - Run a keylogger • win_files_operation - Affect private profile
VirusTotal	Search for analysis

2. **Services for** invoice_2318362983713_823931342io.pdf.exe **process tree**

Time & API	Arguments
OpenSCManagerW Sept. 18, 2024, 5:36 p.m.	desired_access: 983103 database_name: machine_name:
OpenServiceW Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce270 service_name: mpssvc service_manager_handle: 0x005ce310 desired_access: 983551
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce270 service_name: mpssvc control_code: 1
DeleteService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce270 service_name: mpssvc
OpenServiceW Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce388 service_name: SharedAccess service_manager_handle: 0x005ce310 desired_access: 983551
ControlService Sept. 18, 2024, 5:36 p.m.	service_handle: 0x005ce388 service_name: SharedAccess control_code: 1
DeleteService	service_handle: 0x005ce388

3. **Processes for** cmd.exe **process tree.**

Time & API	Arguments
NtTerminateProcess Sept. 18, 2024, 4:36 p.m.	status_code: 0xffffffff process_identifier: 0 process_handle: 0x00000000
NtTerminateProcess Sept. 18, 2024, 4:36 p.m.	status_code: 0xffffffff process_identifier: 0 process_handle: 0x00000000
NtTerminateProcess Sept. 18, 2024, 4:36 p.m.	status_code: 0xffffffff process_identifier: 2200 process_handle: 0xffffffff

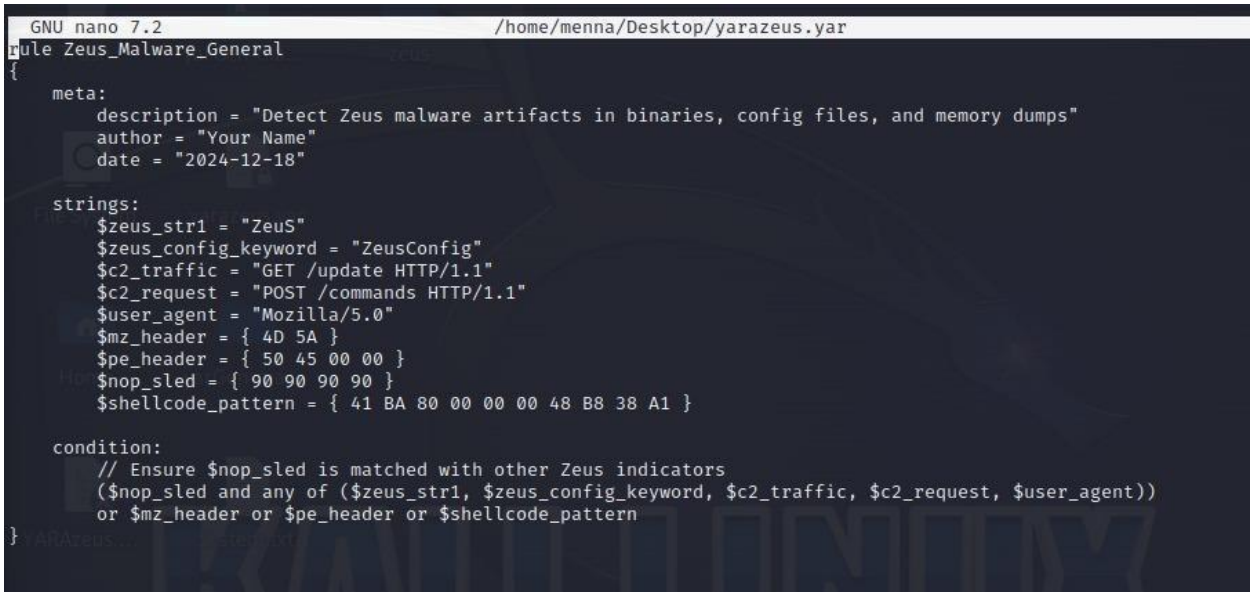
1. **Network analysis:** Zeus uses a mix of **HTTP**, **HTTPS**, and **custom protocols** for communicating with its C2 infrastructure. These communications are usually encrypted or obfuscated to avoid detection by network monitoring tools. Cuckoo Sandbox revealed that Zeus attempted to communicate with several IP addresses and domains that were associated with known malicious activity.

Suricata Alerts

Flow	SID	Signature	Category
UDP 192.168.168.204:49582 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49583 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49584 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49585 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49586 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49587 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49588 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49589 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49590 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49591 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected
UDP 192.168.168.204:49592 -> 85.114.128.127:53	2015474	ET MALWARE ZeroAccess udp traffic detected	A Network Trojan was detected

YARA Rules

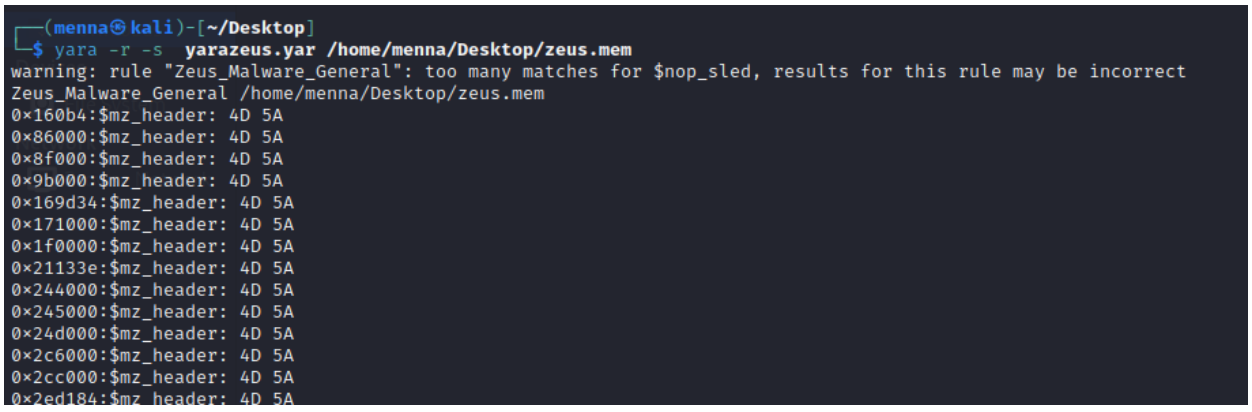
This rule is designed to detect Zeus malware artifacts in binaries, configuration files, and memory dumps. It uses a combination of static strings and byte patterns that are commonly associated with Zeus malware.



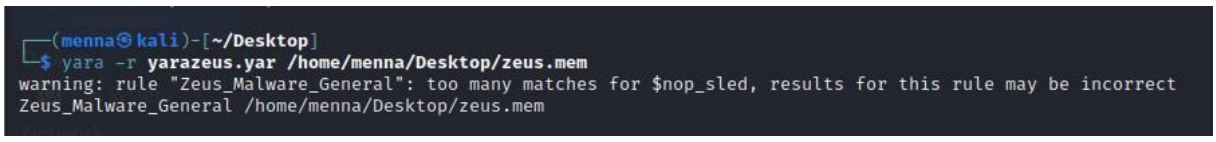
```
  GNU nano 7.2                          /home/menna/Desktop/yarazeus.yar
rule Zeus_Malware_General
{
    meta:
        description = "Detect Zeus malware artifacts in binaries, config files, and memory dumps"
        author = "Your Name"
        date = "2024-12-18"

    strings:
        $zeus_str1 = "ZeuS"
        $zeus_config_keyword = "ZeusConfig"
        $c2_traffic = "GET /update HTTP/1.1"
        $c2_request = "POST /commands HTTP/1.1"
        $user_agent = "Mozilla/5.0"
        $mz_header = { 4D 5A }
        $pe_header = { 50 45 00 00 }
        $nop_sled = { 90 90 90 90 }
        $shellcode_pattern = { 41 BA 80 00 00 00 48 B8 38 A1 }

    condition:
        // Ensure $nop_sled is matched with other Zeus indicators
        ($nop_sled and any of ($zeus_str1, $zeus_config_keyword, $c2_traffic, $c2_request, $user_agent))
        or $mz_header or $pe_header or $shellcode_pattern
}
```

The result of executing the YARA Rules I created on the memory dump file

For more detailed output, I added the **-s** argument



```
┌──(menna㉿kali)-[~/Desktop]
└─$ yara -r -s  yarazeus.yar /home/menna/Desktop/zeus.mem
warning: rule "Zeus_Malware_General": too many matches for $nop_sled, results for this rule may be incorrect
Zeus_Malware_General /home/menna/Desktop/zeus.mem
0×160b4:$mz_header: 4D 5A
0×86000:$mz_header: 4D 5A
0×8f000:$mz_header: 4D 5A
0×9b000:$mz_header: 4D 5A
0×169d34:$mz_header: 4D 5A
0×171000:$mz_header: 4D 5A
0×1f0000:$mz_header: 4D 5A
0×21133e:$mz_header: 4D 5A
0×244000:$mz_header: 4D 5A
0×245000:$mz_header: 4D 5A
0×24d000:$mz_header: 4D 5A
0×2c6000:$mz_header: 4D 5A
0×2cc000:$mz_header: 4D 5A
0×2ed184:$mz_header: 4D 5A
```



```
┌──(menna㉿kali)-[~/Desktop]
└─$ yara -r yarazeus.yar /home/menna/Desktop/zeus.mem
warning: rule "Zeus_Malware_General": too many matches for $nop_sled, results for this rule may be incorrect
Zeus_Malware_General /home/menna/Desktop/zeus.mem
```

Conclusion

Impact on End-User Machines and Financial Institutions

Zeus's primary target is **financial institutions**, but its impact extends to personal devices, corporate networks, and government systems. The malware's ability to steal login credentials, particularly banking details, has made it a preferred tool for cybercriminals seeking financial gain.

Impact on End-User Machines:

Once installed on a personal device, Zeus can perform a range of malicious actions:

1. **Credential Theft**: The malware is designed to capture sensitive login credentials, including usernames, passwords, and bank account information. It achieves this through **man-in-the-browser (MitB)** attacks, where it modifies the web browser to steal information as the user enters it.
2. **System Degradation**: Infected machines often experience significant performance slowdowns. This is due to Zeus's constant monitoring and injection into various system processes, which increases CPU and memory usage.

Impact on Financial Institutions:

Zeus poses a massive threat to financial institutions. Its ability to target banking systems and online transaction platforms means that banks, payment processors, and even e-commerce platforms are at risk.

1. **Credential Harvesting**: Zeus's MitB technique allows it to capture not only personal banking credentials but also **two-factor authentication (2FA)** codes, which are used to secure banking transactions.
2. **Unauthorized Transactions**: After capturing banking credentials, Zeus can initiate unauthorized transfers, draining victim accounts without their knowledge. This has resulted in **millions of dollars in losses** for both individuals and banks.

Preventive Measures Against Zeus Trojan and Its Variants

As Zeus continues to inspire new variants, the threat landscape for financial institutions and end-users evolves. However, several preventive measures can be employed to reduce the risk of infection.

1. User Education

Since phishing and social engineering remain the primary infection methods, educating users is the first line of defense. Employees of financial institutions and end-users should be trained to recognize phishing emails and avoid clicking on suspicious links or downloading untrusted attachments.

2. Email Filtering and Anti-Phishing Solutions

Implementing robust **email filtering** solutions that block phishing emails can help mitigate the risk of infection. Additionally, many anti-phishing tools can detect and block suspicious emails or malicious links before they reach the user's inbox.

3. Network Segmentation

To prevent the spread of malware within an organization, network segmentation should be implemented. By isolating different parts of the network, an infection in one segment will not easily propagate to the rest of the network, reducing the overall impact.

4. Use of Sandboxing Solutions

Security teams should use sandboxing solutions like **Cuckoo Sandbox** to analyze suspicious files before they are allowed to enter the corporate network. By quarantining and analyzing these files, organizations can detect and block malware before it causes damage.

5. Regular Patching and Updates

Outdated software is often exploited by malware like Zeus to gain access to systems. Regularly updating software and applying security patches is critical to prevent Zeus from exploiting known vulnerabilities.

6. Advanced Threat Detection Solutions

Financial institutions should invest in **advanced threat detection** solutions that monitor for suspicious activities like process injections, auto-run modifications, and unusual network communications. By detecting these behaviors early, organizations can block malware like Zeus before it causes significant damage.

Final Prospect

The **Zeus Trojan** is one of the most dangerous and financially devastating pieces of malware ever created. Its evolution into numerous variants, such as **Gameover Zeus**, has made it a formidable adversary for both cybersecurity professionals and financial institutions. Through both **static** and **dynamic analysis**, we've explored how Zeus operates, its exfiltration methods, and its impact on global cybercrime.

As malware continues to evolve, it's vital to employ robust security practices, such as using sandbox environments, educating users, and leveraging advanced threat detection tools to stay ahead of these threats. The fight against malware like Zeus is ongoing, and continuous vigilance is necessary to protect systems and financial data from these sophisticated attacks.

Subject- Reverse Engineering and Malware Analysis

Faculty – Mr. Ashish Revar

Anurag Tripathi

220031101611061

BTech_Sem7