infinitum IT
Power of integrated Security

LokiBot
Technical Analysis Report

infinitumitlabs

CONTENTS

LokiBot and What You Need to Know

What is LokiBot?

LokiBot, also known as Loki PWS or Loki-bot, is a Trojan malware designed for the illicit purpose of stealing sensitive information such as usernames, passwords, cryptocurrency wallets, and other identity-related data. The LokiBot Trojan malware made its initial appearance in 2015 and has since gained significant notoriety among cybercriminal circles as an effective means of establishing a backdoor into compromised Windows systems. This malicious software continues to thrive due to its proficiency in surreptitiously exfiltrating valuable information.

Functioning as a malicious software family, LokiBot monitors both browser and desktop activities, systematically pilfering critical data from victims, including user credentials, bank particulars, and the contents of cryptocurrency wallets. The defining characteristic of LokiBot lies in its ability to capture and record sensitive data, a behavior commonly observed in Trojan horse viruses. It methodically gathers stored login credentials and passwords, primarily within web browsers, and maintains constant surveillance over user activities, such as keystroke logging.

The acquired information is promptly relayed to a remote server under the control of LokiBot's developers. This streamlined transfer of data ensures that captured details are swiftly at the disposal of malicious actors. Notably, malicious cyber agents frequently deploy LokiBot to target systems running Windows and Android operating systems. They propagate this malevolent software through various means, including but not limited to email, phishing websites, text messages, and other personalized communication channels.

In essence, LokiBot epitomizes the art of infiltrating systems with the primary goal of extracting sensitive data. Its intricate techniques and operational methods underline the ongoing challenge posed by such advanced forms of cyber threats, necessitating a comprehensive and multi-faceted approach to cybersecurity.

LokiBot Overview

Kellyzx.exe, a malicious software, has been identified as carrying out a series of harmful actions for malicious purposes. Upon closer examination, it has been determined that the malware engages in loading DLL files to execute harmful processes. Additionally, it establishes persistence within the system by dropping an executable file named **"hjxnj.exe"** and making changes to the Windows Registry. This enables the malware to maintain a foothold within the compromised system, ensuring its presence even after system reboots.

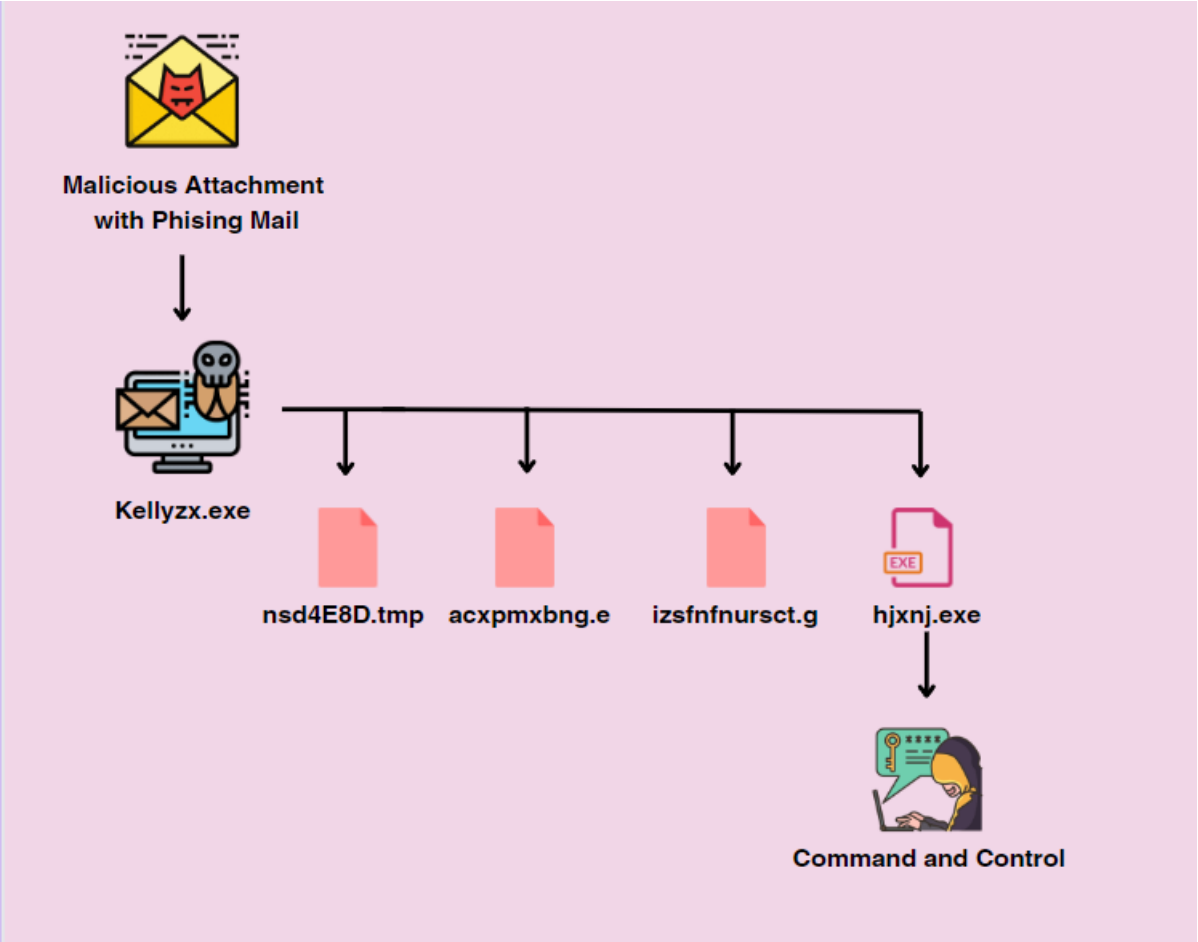
The malware's modus operandi unfolds as follows: After establishing a connection with the IP address **"171[.]22[.]30[.]147"** via TCP, the malware initiates communication with a remote command and control server. This connection serves as a conduit for the malware operators to exert control and issue instructions to the compromised system.

Subsequently, the malware undertakes data collection, which it accomplishes through a POST request to the URL **"/kelly/five/fre[.]php http/1.0 POST."** Through this request, the malware encrypts the harvested sensitive data, thereby concealing its contents from potential interception during transmission. The encrypted data is then sent to the command and control server, enabling the malicious actors to access and exploit the stolen information.

Upon the successful transfer of data, the TCP connection is terminated. This step further obfuscates the malicious activity, complicating efforts to track and mitigate the threat.

In conclusion, **Kellyzx.exe** malicious software employs a multi-step process to infiltrate systems, establish persistence, communicate with a remote server, encrypt stolen data, and evade detection. Countering such threats requires robust cybersecurity measures, including up-to-date security solutions, patch management, and user education to thwart potential attacks and mitigate their impact.

Infection Chain



Static Analysis

Kellyzx.exe Analysis

File Name	kellyzx.exe
SHA-1	E6107CA70A7E8461A0105FAE3F8FE6DE9A65FF17
MD5	3BC68A0764CCC400C9A9F595E9F3ED3E
File Type	PE32/EXE

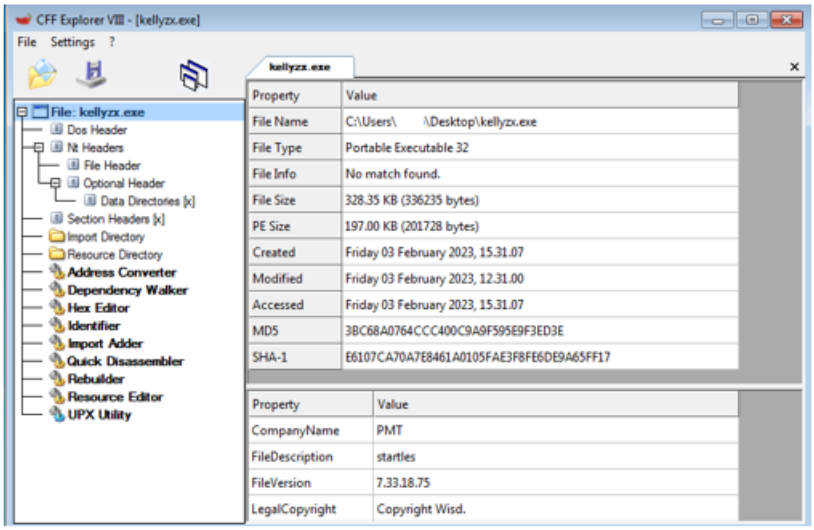


Figure 1- Information about file

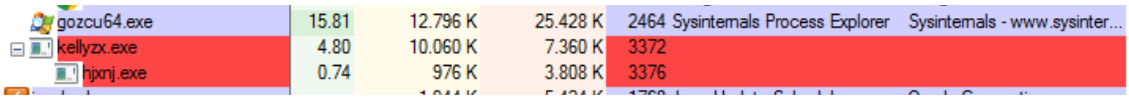


Figure 2- Process monitor logs of kellyzx.exe

Close examination of the malware reveals that it creates a child process by launching the executable **"hjxnj.exe".**

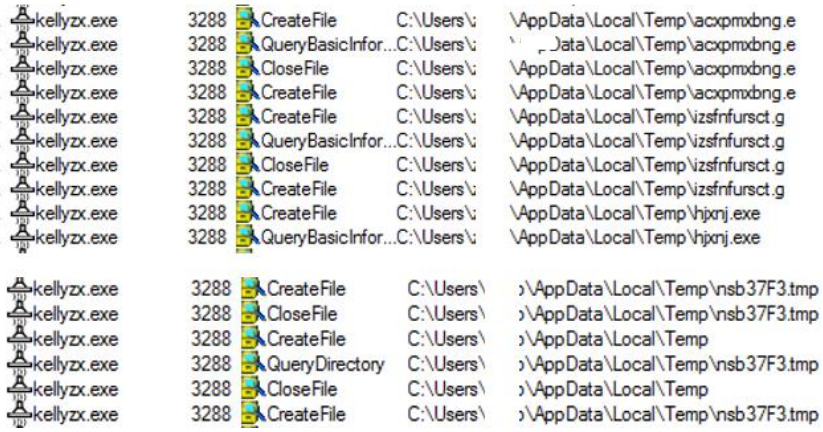


Figure 3- Logs

Upon scrutinizing the log records of the target system, it becomes evident that the malicious file employs the **CreateFile API** to generate files with diverse extensions and randomly assigned names.

Dynamic Analysis

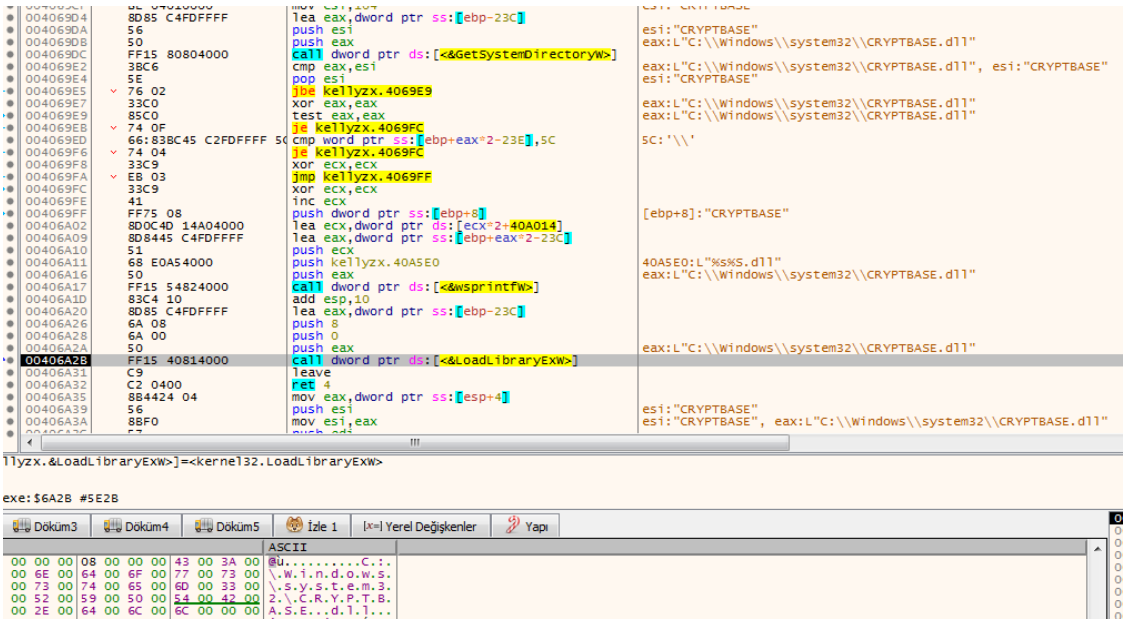


Figure 4- Loads DLL

With the **LoadLibraryExW API**, it loads DLLs to change system settings, access sensitive data or take control of the system, execute malicious drivers or disable security features, and download additional malicious data.

C:\\Windows\\system32\\CLBCATQ.dll	C:\\Windows\\system32\\APPHELP.dll
C:\\Windows\\system32\\NTMARTA.dll	C:\\Windows\\system32\\DWMAPI.dll
C:\\Windows\\system32\\UXTHEME.dll	C:\\Windows\\system32\\OLEACC.dll
C:\\Windows\\system32\\USERENV.dll	C:\\Windows\\system32\\CLBCATQ.dll
C:\\Windows\\system32\\SETUPAPI.dll	C:\\Windows\\system32\\SHFOLDER.dll
C:\\Windows\\system32\\NTMARTA.dll	C:\\Windows\\system32\\CRYPTBASE.dll

Table 1- Files with .dll extension loaded by malicious file

Figure 5- Getting the name of the .tmp file

The name of the file **"nsd4E8D.tmp"** is generated in the directory **"C:\Users\Admin\AppData\Local\Temp"** The malicious software creates the name for the **.tmp** extension file with the prefix **"ns"** followed by random characters, as observed. This is often done to obfuscate the file and its purpose, making it harder for security measures to detect it. Using random names and prefixes adds an extra layer of camouflage to the malicious file, increasing its chances of going unnoticed within the system.

Figure 6- Creating the .tmp file

The file created with the GlobalAlloc API is allocated in the **"C:\Users\Admin\AppData\Local\Temp"** directory. After this space, the **"nsd4E8D.tmp"** file with the .tmp extension is created using the CreateFile API. This temporary file is created to store data or execute certain actions as part of the malware's operations.

Figure 7- Creation of acxpmxbng.e file

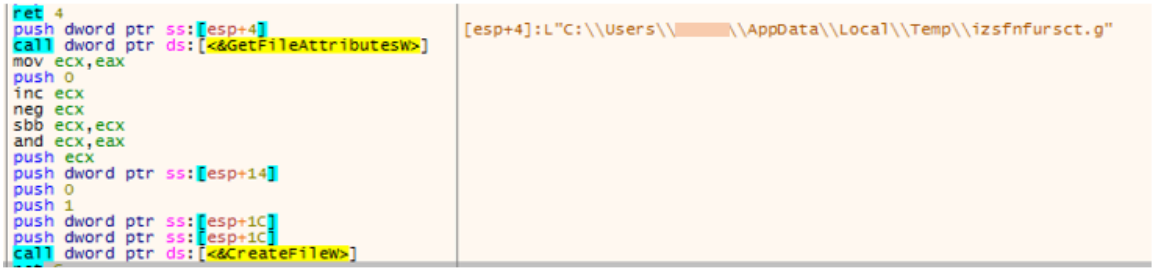


Figure 8- Creating the izsfnfursct.g file

As shown in Figures 7 and 8, files with arbitrary names and **.e and .g** extensions are created in the **"C:\Users\Admin\AppData\Local\Temp"** directory. The malicious file implements an evasion technique by interfering with legitimate files commonly found in temporary directories. By using arbitrary names and extensions, malware aims to avoid detection by security solutions that rely on specific file naming conventions or extensions to identify potentially malicious files. This technique adds to the malware's ability to stay undetected and works secretly within the system.

Figure 9- Creating the file named hjxnj.exe

An executable file named "hjnx.exe" is created in the "C:\Users\Admin\AppData\Local\Temp" directory. This action is executed as a part of the malware's propagation strategy. By generating an executable file within the temporary directory, the malware seeks to establish a foothold on the compromised system and execute its malicious code. The use of the temporary directory aids in concealing the file amidst a myriad of other temporary files, making it harder for users or security solutions to immediately discern its presence.

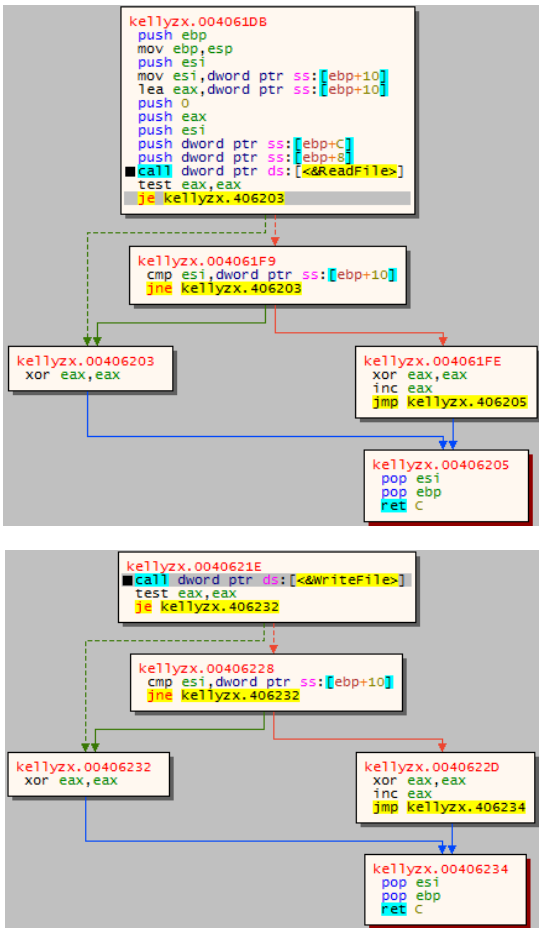


Figure 10- Use of ReadFile and WriteFile APIs

For all files created on the target system, file reading and writing operations are performed using the **ReadFile** and **WriteFile** APIs.

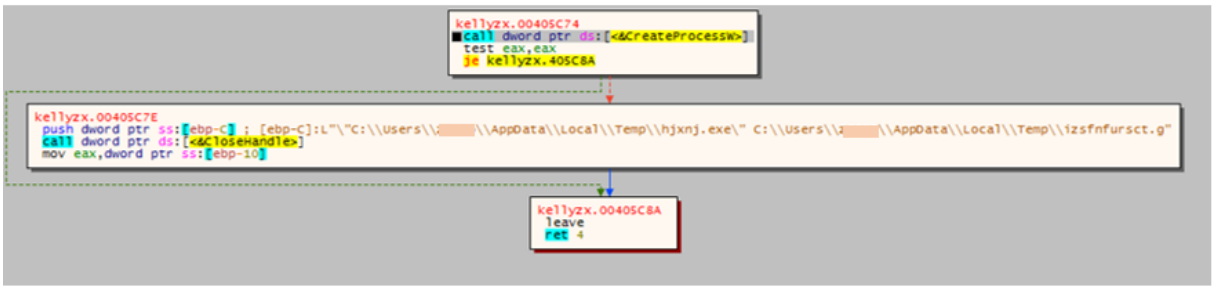


Figure 11- Usage of CreateProcess API

The file named **hjxnj.exe** created by the malware is started as a process on the target system using the **CreateProcess API.**

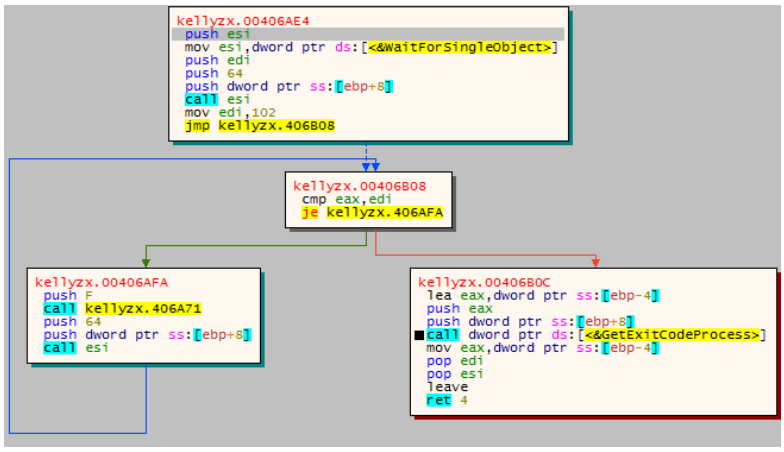


Figure 12- Closing the malicious file

As seen in Figure 12, it gets the termination status of the **kellyzx.exe** process running using the **GetExitCodeProcess API**.

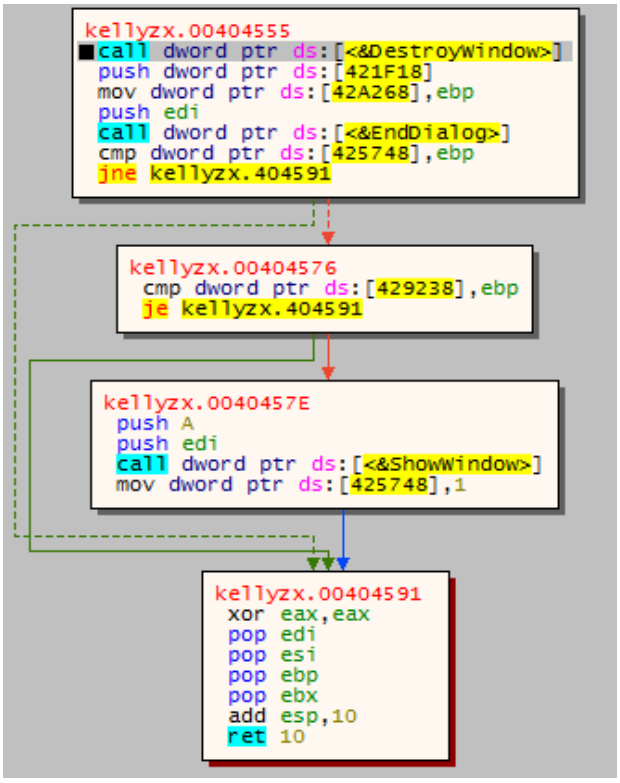


Figure 13- Termination of the kellyzx.exe

The malware first closes the current window with the **DestroyWindow API**. It then closes the dialog with the **EndDialog API**.

The malware continues with the **hjxnj.exe** it created to other processes on the target device.

hjxnj.exe Analysis

File Name	hjxnj.exe
SHA-1	EC95F0F1DC55A51903696E021963EEACC210FF05
MD5	CF86B09B00E89238F9205E6D469BCDD6
File Type	PE32/EXE

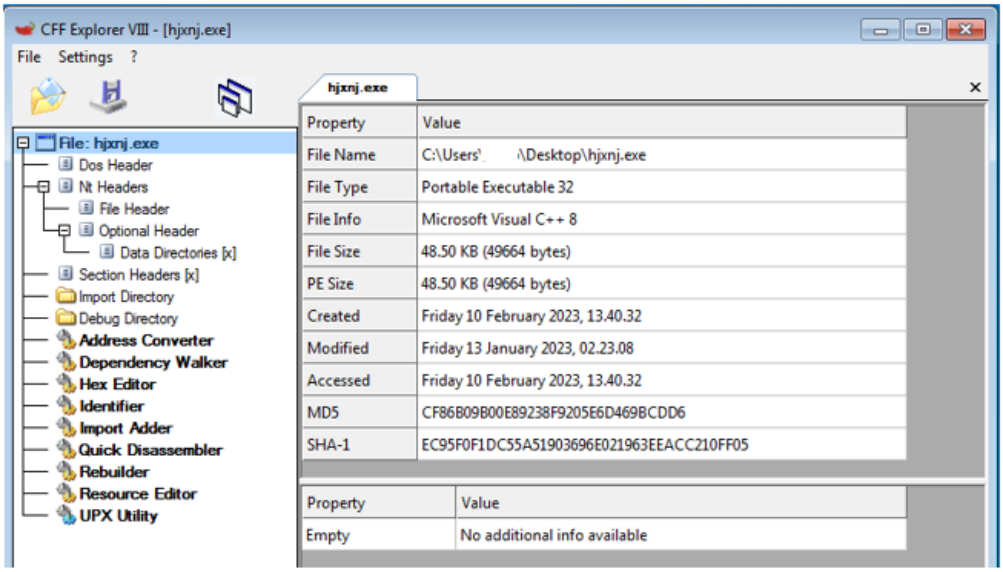


Figure 14- Examination of the Pest in CFF Explorer

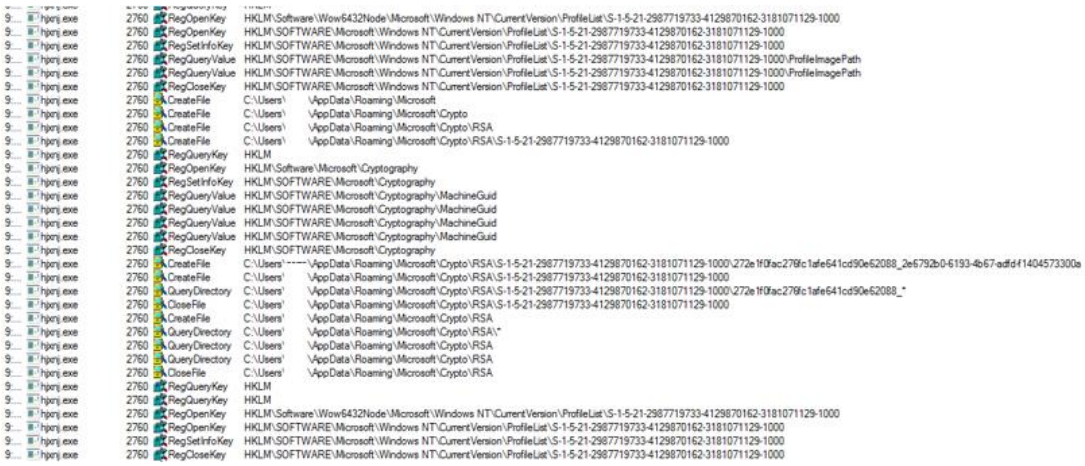


Figure 15- Examination of log records

The log records of the file named **hjxnj** on the target device after the malware starts running are shown in Figure 15.

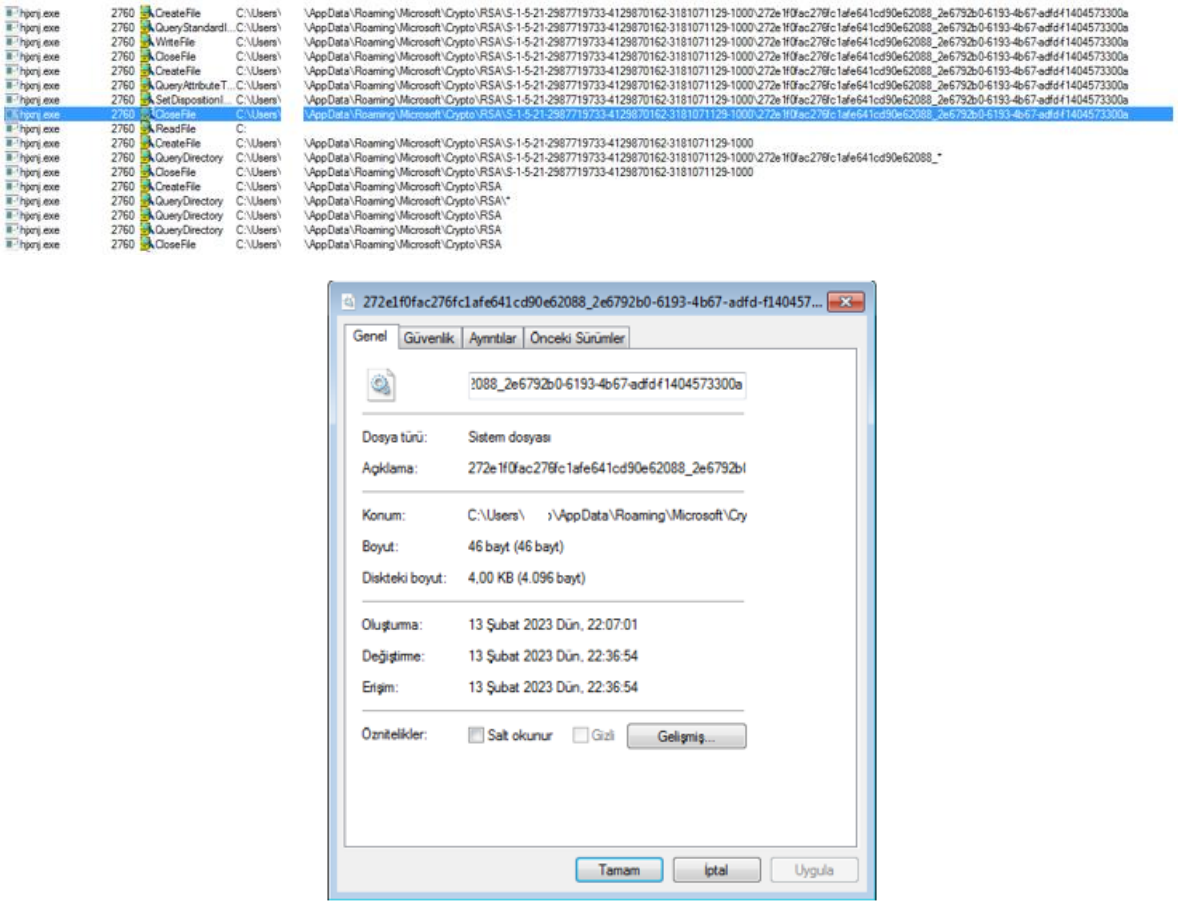


Figure 16- Changes made by the malware in the RSA folder

It modifies certificates stored in the system file named **"272e1f0fac276fc1afe641cd90e62088_2e6792b0-6193-4b67-adfd-f1404573300a,"** found in the directory **"C:\Users\Admin\AppData\Roaming\Microsoft\Crypto\RSA\S-1-5-21-2987719733-4129870162-3181071129-1000"** By leveraging the WriteFile API, it alters the contents of the file, manipulating the certificates and impersonating the user identity to gain access to sensitive data. Through this process, the malicious actor seeks to exploit vulnerabilities, potentially compromising confidential information.

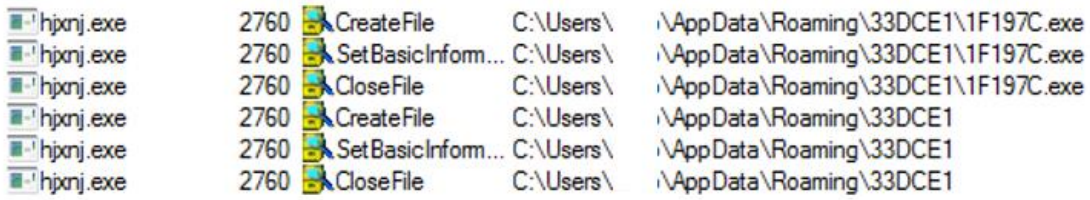


Figure 17- The malware creates an executable file

It creates an executable file named **1F197C.exe** in the **"C:\Users\Admin\AppData\Roaming\33DCE1"** directory and then closes this file.

```
.text:00404656
.text:00404656 loc_404656:
.text:00404656 push    esi
.text:00404657 lea     eax, [ebp+SystemTimeAsFileTime] ; Load Effective Address
.text:0040465A push    eax             ; lpSystemTimeAsFileTime
.text:0040465B call    ds:GetSystemTimeAsFileTime ; Indirect Call Near Procedure
.text:00404661 mov     esi, [ebp+SystemTimeAsFileTime.dwHighDateTime]
.text:00404664 xor     esi, [ebp+SystemTimeAsFileTime.dwLowDateTime] ; Logical Exclusive OR
.text:00404667 call    ds:GetCurrentProcessId ; Indirect Call Near Procedure
.text:0040466D xor     esi, eax        ; Logical Exclusive OR
.text:0040466F call    ds:GetCurrentThreadId ; Indirect Call Near Procedure
.text:00404675 xor     esi, eax        ; Logical Exclusive OR
.text:00404677 call    ds:GetTickCount ; Indirect Call Near Procedure
.text:0040467D xor     esi, eax        ; Logical Exclusive OR
.text:0040467F lea     eax, [ebp+PerformanceCount] ; Load Effective Address
.text:00404682 push    eax             ; lpPerformanceCount
.text:00404683 call    ds:QueryPerformanceCounter ; Indirect Call Near Procedure
.text:00404689 mov     eax, dword ptr [ebp+PerformanceCount+4]
.text:0040468C xor     eax, dword ptr [ebp+PerformanceCount] ; Logical Exclusive OR
.text:0040468F xor     esi, eax        ; Logical Exclusive OR
.text:00404691 cmp     esi, edi        ; Compare Two Operands
.text:00404693 jnz     short loc_40469C ; Jump if Not Zero (ZF=0)
```

Figure 18- Obtaining system information

The malware uses the APIs shown in Figure 18 to get the information of the system time, the current process and thread id.

```
mov edi,edi
push edi
push hjxnj.409EC4                          409EC4:L"KERNEL32.DLL"
call dword ptr ds:[<&GetModuleHandleW>]
mov edi,eax
test edi,edi
jne hjxnj.4044C6
call hjxnj.4041F6
xor eax,eax
pop edi
ret
push esi
mov esi,dword ptr ds:[<&GetProcAddress>]
push hjxnj.409F00                          409F00:"FlsAlloc"
push edi
call esi
push hjxnj.409EF4                          409EF4:"FlsGetValue"
push edi
mov dword ptr ds:[<&FlsAlloc>],eax
call esi
push hjxnj.409EE8                          409EE8:"FlsSetValue"
push edi
mov dword ptr ds:[<&FlsGetValue>],eax
call esi
push hjxnj.409EE0                          409EE0:"FlsFree"
push edi
mov dword ptr ds:[<&FlsSetValue>],eax
call esi
cmp dword ptr ds:[<&FlsAlloc>],0
mov esi,dword ptr ds:[<&TlsSetValue>]
mov dword ptr ds:[<&FlsFree>],eax
je hjxnj.404526
cmp dword ptr ds:[<&FlsGetValue>],0
je hjxnj.404526
cmp dword ptr ds:[<&FlsSetValue>],0
je hjxnj.404526
test eax,eax
jne hjxnj.40454A
mov eax,dword ptr ds:[<&TlsGetValue>]
```

Figure 19- APIs used by the malicious file

The malicious file analyzes in dynamic time and uses the necessary APIs.

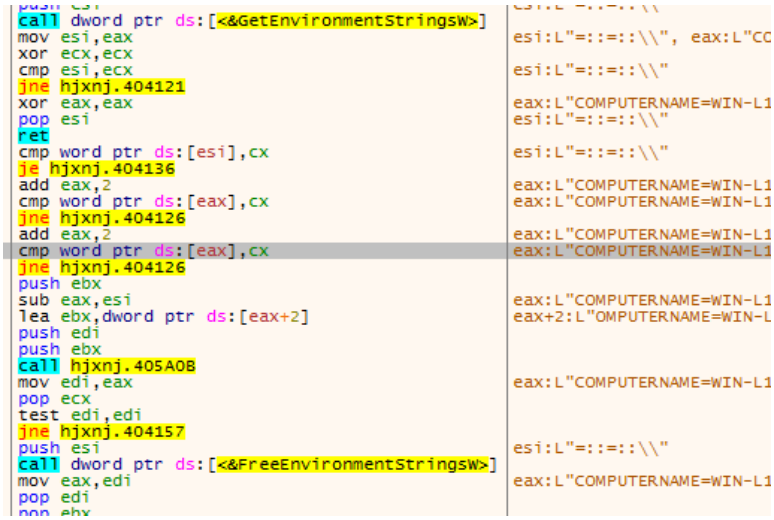


Figure 20- The malware collects information

It appears that the malware obtains this information from files containing important information about the current user's profile, system configuration and environment in order to store sensitive information on the target device, insert malicious code into critical system files or perform various malicious functions.

ALLUSERSPROFILE=C:\\ProgramData	windows_tracing_logfile=C:\\BVTBin\\Tests\\ installpackage\\csilogfile.log
LOCALAPPDATA=C:\\Users\\Admin\\AppData\\ Local	PROCESSOR_ARCHITECTURE=x86
LOGONSERVER=\\ComputerName	PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE; .WSF;.WSH;.MSC
NUMBER_OF_PROCESSORS=1	COMPUTERNAME= WIN-L1KDN79P80J
SESSIONNAME=Console	SystemDrive=C:"
"TEMP=C:\\Users\\Admin\\AppData\\Local\\Temp "	"USERNAME=admin"

Table 2- Some information received

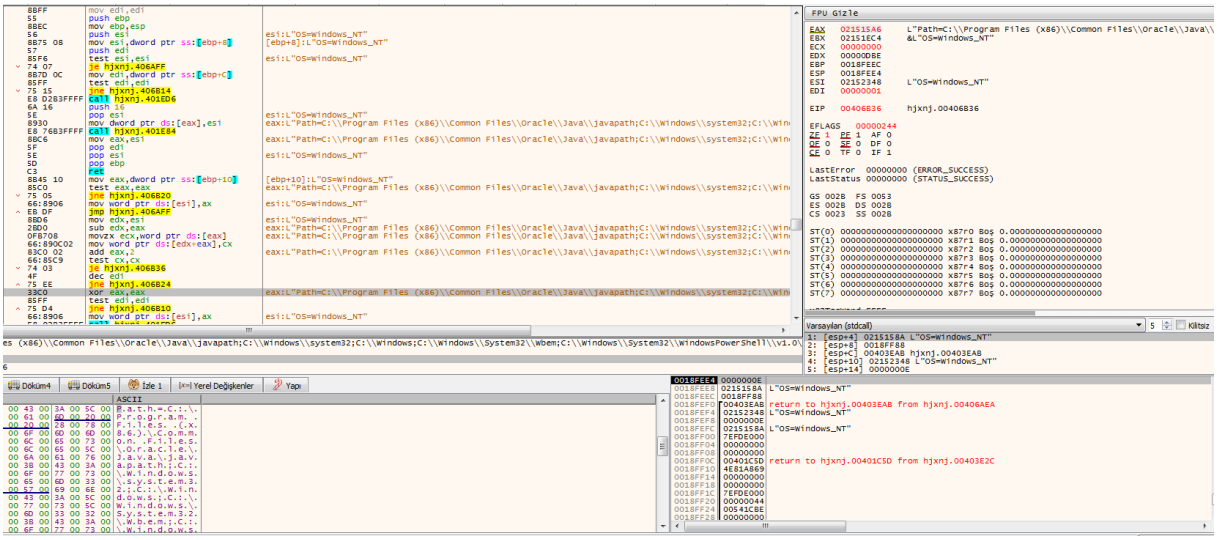


Figure 21- Detection of changes made

It is determined that the pest has added to the path from Environment variables.

These paths are:

C:\\Program Files (x86)\\Common Files\\Oracle\\Java\\javapath

C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\

C:\\Users\\Admin\\AppData\\Local\\Programs\\Python\\Python37\\Scripts\\

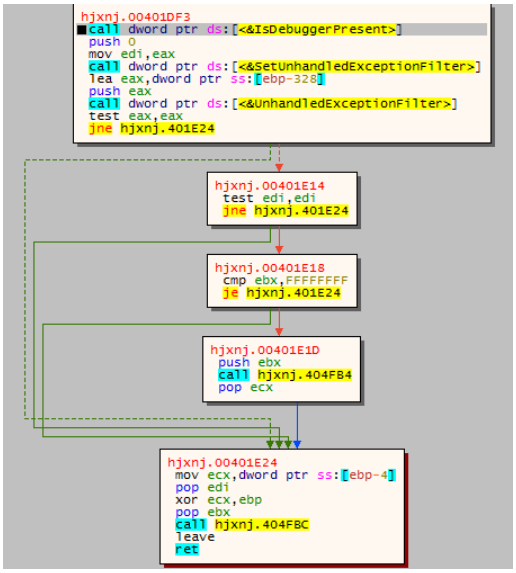


Figure 22- Anti-debug detection

It used **antidebug technique** to detect the malware analysis environment, change the malware behavior or terminate itself. This makes it difficult for analysts to understand the inner workings of the malware.

Network Analysis

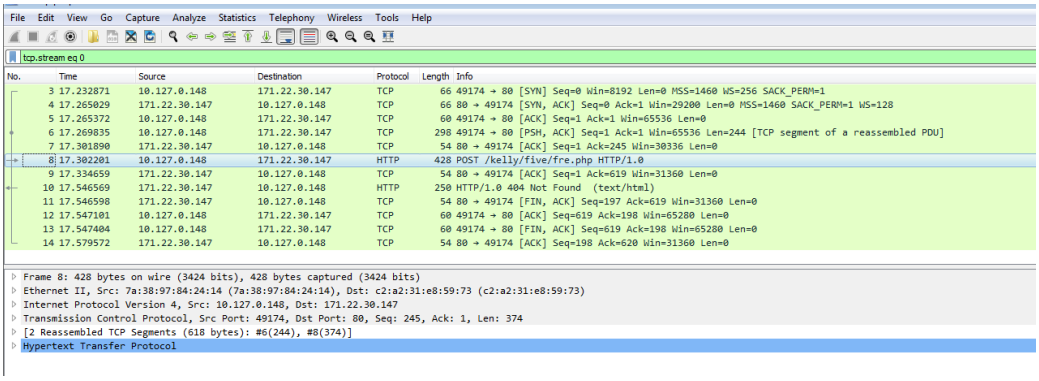
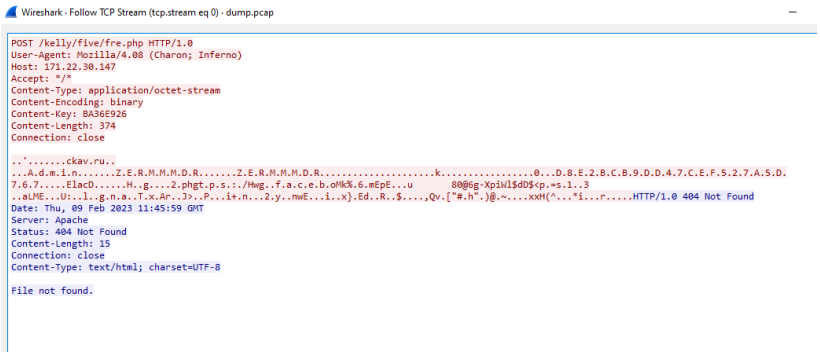


Figure 23- Wireshark Log Record



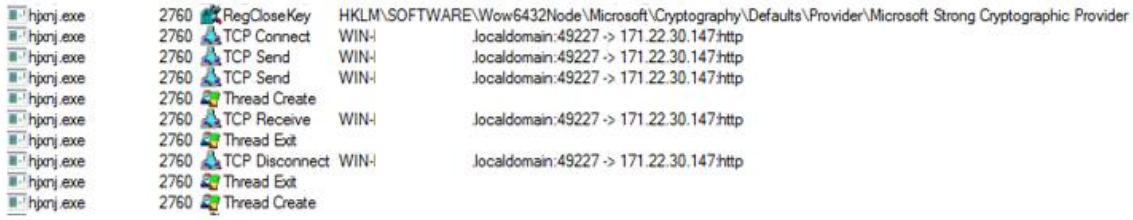


Figure 24- Process Monitor Log Record

It is observed that the malware first establishes a TCP connection with the IP address **"171[.]22[.]30[.]147".** Then it sends the **"/kelly/five/fre[.]php http/1.0 POST"** request by encrypting the data it collects on the target device. Then the TCP connection is closed.

IOCs

IPs :

IOC Type	IOC
IPv4	171[.]22[.]30[.]147
IPv4	188[.]114[.]96[.]13

DOMAINs:

IOC Type	IOC
Domain	/kelly/five/fre[.]php

HASHs:

IOC Type	IOC
MD5	6ac95d0ff18baaa2fa5bbfa1cbe4ff6e
MD5	b4ede3be28a02d4ad6033d5e8021e2f4
MD5	ea1faf5523c76af6706f84401b9809c0
SHA1	c557fbe7b5d90e06a8620f8ecf13c0a91dfc213c
SHA256	1b574a66c84924886daec4841e1b107258e019aaf6f336329ae8fae7cbd52a34
SHA256	4edd01345f58b9cc04a88ca15d6b82895f44f5b9cb51ad63b809de09029670ac
SHA256	8a5a024272361bb1ae12860c033bb52685d7b0ea3bce5fac46439f3f3ad36a84

Kellyzx.exe Yara Rule

```
import "hash"

rule kellyzx

{

meta:

        author = "Kerime Gencay"

        date = "13/02/2023"

        description = "LokiBot YARA Rule"

        file_name = "kellyzx.exe"

        hash1 = "3BC68A0764CCC400C9A9F595E9F3ED3E"


strings:

        $string1 = "7.33.18.75" wide

        $string2 = "RichEdit20W" wide

        $string3 = "%s%S.dll" wide


        $opc1 = {FF 15 38 81 40 00 B9 68 CE 40 00 8B F0 E8 F3 38 00 00 8D 85 D4 FD
        FF FF 68 00 78 43 00 50 E8 D9 2E 00 00 53 68 00 01 00 04 6A 02 53 53 8D 85
        D4 FD FF FF 68 00 00 00 C0 50 FF 15 F4 80 40 00} //allocate, deobfuscate and
        createfile

        $opc2  = {FF 15 04 81 40 00 8B C8 6A 00 41 F7 D9 1B C9 23 C8 51 FF 74 24 14
        6A 00 6A 01 FF 74 24 1C FF 74 24 1C FF 15 F4 80 40 00}


condition:

        uint16(0) == 0x5A4D and

         (any of ($string*) or

            any of ($opc*))

}
```

hjxnj.exe Yara Rule

```
import "hash"

rule hjxnj

{

    meta:

            author = "Kerime Gencay"

            date = "13/02/2023"

            description = "LokiBot YARA Rule"

            file_name = "hjxnj.exe"

            hash1 = "CF86B09B00E89238F9205E6D469BCDD6"

    strings:


          $debug_artifact=  "C:\\xampp\\htdocs\\c5892ccff2804af39d7bac9f5f6d95bb\\Load
          er\\Release\\Loader.pdb"



          $opc1 = {FF 15 B4 90 40 00 8B F0 33 C9 3B F1 75 04 33 C0 5E C3 66 39 0E 74
          10    83 C0 02 66 39 08 75 F8 83 C0 02 66 39 08 75 F0 53 2B C6 8D 58 02 57 53
          E8 C8 18 00 00}


    condition:


          uint16(0) == 0x5A4D and

          (any of ($opc*) or

           1 of ($debug_artifact*))


}
```

MITRE ATT&CK TABLE

Discovery	Command and Control	Defense Evasion	Persistence	Collection
T1012 Query Registry	T1071 Web Protocols	T1222 File and Directory Permissions	T1047 Create or Modify Systems	T1055 Data From Local System
T1082 Information Discovery		T1036 Creates Files inside the user directory		

MITIGATIONS

- Employ comprehensive cyber security solutions that include real-time threat detection, behavior monitoring, and malware removal capabilities. This can help identify and neutralize the presence of malicious files such as **"kellyzx.exe"** and **"hjxnj.exe."**
- Ensure that all operating systems, applications, and software are up to date with the latest security patches. Regular updates can help address vulnerabilities that malware often exploits.
- Deploy reputable antivirus and anti-malware solutions that can proactively scan and remove malicious files. These tools can aid in detecting and eliminating threats like **kellyzx.exe** before they cause harm.
- Implement application whitelisting to restrict the execution of unauthorized applications. This can prevent the launch of malicious executables like "Kellyzx.exe" and "hjxnj.exe."
- Provide training to users to recognize phishing emails, suspicious links, and attachments. User awareness can significantly reduce the likelihood of malware infiltration through social engineering tactics.
- Continuously monitor network traffic for unusual patterns and connections to known malicious IP addresses. This can help in early detection and mitigation of malicious activity.
- If a system is compromised, isolate it from the network to prevent the malware from spreading to other devices or servers.
- Maintain regular backups of critical data. In case of an attack, having backups can mitigate the impact of data loss.

infinitum IT
Power of integrated Security

LokiBot

Technical Analysis Report

infinitumitlabs