Icon

Description automatically generated

Practical Malware Analysis & Triage

Malware Analysis Report

Dropper.DownloadFromURL.exe

Feb 2024 | Amna Jasser | v1.0

# Table of Contents

[Table of Contentstr 2](#_Toc159370497)

[Executive Summary 3](#_Toc159370498)

[High-Level Technical Summary 4](#_Toc159370499)

[Malware Composition 6](#_Toc159370500)

[favicon.ico: 6](#_Toc159370501)

[CR433101.dat.exe: 6](#_Toc159370502)

[Basic Static Analysis 7](#_Toc159370503)

[Basic Dynamic Analysis 10](#_Toc159370504)

[Advanced Static Analysis 14](#_Toc159370505)

[Advanced Dynamic Analysis 22](#_Toc159370506)

[Indicators of Compromise 24](#_Toc159370507)

[Network Indicators 24](#_Toc159370508)

[Host-based Indicators 26](#_Toc159370509)

[Appendices 28](#_Toc159370510)

[A. Yara Rules 28](#_Toc159370511)

[B. Callback URLs 28](#_Toc159370512)

# Executive Summary

|  |  |
| --- | --- |
| SHA256 hash | 92730427321A1C4CCFC0D0580834DAEF98121EFA9BB8963DA332BFD6CF1FDA8A |

The file "dropper.downloadfromurl.exe" is a program written in a language called C++. When you run it, it does a few things that might seem a bit tricky. First, it goes to the internet and grabs a picture file called "favicon.ico" from a specific place. Then, it also gets a webpage from another spot on the internet and saves it as a file named "A7102UL2.htm." If it successfully gets the "favicon.ico" picture, it goes on to create a new file called "CR433101.dat.exe," which is like a copy of the picture. After that, it talks to yet another place on the internet using a special function called "InternetOpenUrlW," getting information from a specific webpage (<http://huskyhacks.dev>). Following this, it creates another file, also named "CR433101.dat.exe," but this time in a different location on your computer (C:/Users/Public/Documents/), and this new file is the same as the "favicon.ico" picture it downloaded. In simpler terms, this program seems to be doing some interesting and somewhat sneaky things, like grabbing pictures and web pages from the internet and making copies of them on your computer. The exact reason for doing this isn't clear, but it appears to involve interacting with a specific webpage.

YARA signature rules are attached in Appendix A. Malware sample and hashes have been submitted to VirusTotal for further examination.

# High-Level Technical Summary

dropper.downloadfromurl.exe is a PE file that connect to URLs to get the payloads and execute the dropper.

1. The malware execution begins with the initial PE file, "dropper.downloadfromurl.exe," which is written in C++.
2. The PE establishes a connection to a specific URL (<http://ssl-6582datamanager.helpdeskbros.local/favicon.ico>) with the aim of downloading a file named "favicon.ico." This icon file is retrieved and is stored in the directory "C:/Users/howl/AppData/Local/Microsoft/Windows/Inetchache/IE/N27Z54F7/".
3. Simultaneously, it fetches an HTML page from the URL (<http://huskyhacks.dev>), saving it as a file named "A7102UL2.htm."
4. The creation of the file "CR433101.dat.exe" is conditional upon the successful download of "favicon.ico." If the download is successful, the dropper proceeds to execute "CR433101.dat.exe."
5. "CR433101.dat.exe" replicates the content of the downloaded "favicon.ico." Additionally, it calls the "InternetOpenUrlW" function, a Windows API function that allows developers to open a specified URL (<http://huskyhacks.dev>) and obtain a handle (webpage) to the corresponding internet resource.
6. The obtained handle can be utilized for further operations or interactions with the identified resource. Following this, a new instance of "CR433101.dat.exe" is created in the directory "C:/Users/Public/Documents/." The content of this new file is identical to the previously downloaded "favicon.ico."

# Malware Composition

**dropper.downloadfromurl.exe** consists of the following components:

|  |  |
| --- | --- |
| File Name | SHA256 Hash |
| dropper.downloadfromurl.exe | 92730427321A1C4CCFC0D0580834DAEF98121EFA9BB  8963DA332BFD6CF1FDA8A |
| favicon.ico | C090FAD79BC646B4C8573CB3B49228B96C5B7C93A50F0E3B2BE98  39ED8B2DD8B |
| A7102UL2.htm | E3B0C44298FC1C149AFBF4C8996FB92427AE41E4649B934CA  495991B7852B855 |
| CR433101.dat.exe | C090FAD79BC646B4C8573CB3B49228B96C5B7C93A50F0E3B2BE  9839ED8B2DD8B |

dropper.downloadfromurl.exe

The initial PE file that create CR433101.dat.exe and connect to a URL to download favicon.ico and if it download it successfully then it proceed to execute CR433101.dat.exe ie the dropper.

favicon.ico:

This file is download from this URL (http://ssl-6582datamanager.helpdeskbros.local/favicon.ico) at the beginning of the execution of the unknown file, it is then saved in C:/Users/howl/AppData/Local/Microsoft/Windows/Inetchache/IE/N27Z54F7/

as ico file.

A7102UL2.htm:

It is html page download from (http://huskyhacks.dev).

CR433101.dat.exe:

This file will not be created if favicon.ico not downloaded, if it is downloaded then after it called InternetOpenUrlW which is a function that developers can use to open a URL (http://huskyhacks.dev) and get a handle (webpage) to the corresponding internet resource. This handle can then be used for further operations or interactions with the identified resource, after that this file was created in this directory: C:/Users/Public/Documents/.

This file is exactly the same as favicon.ico.

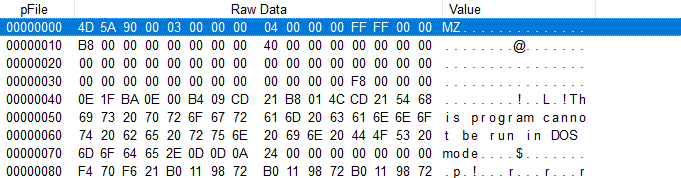
# Basic Static Analysis

{Screenshots and description about basic static artifacts and methods}

**dropper.downloadfromurl.exe.malz**

PESTUDIO and PEVIEW output:

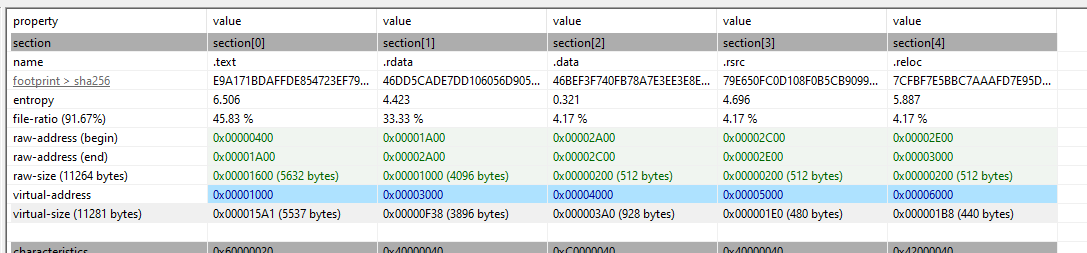
Looking at the first bytes it indicated it is a PE file:



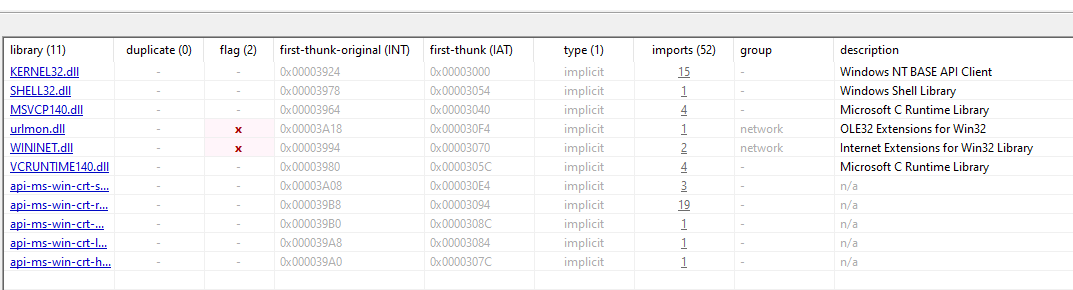
Signature: Microsoft Visual C++

Cpu: 32-bit

Looking at the virtua address and the raw address size there is not significant difference so it is not packed:

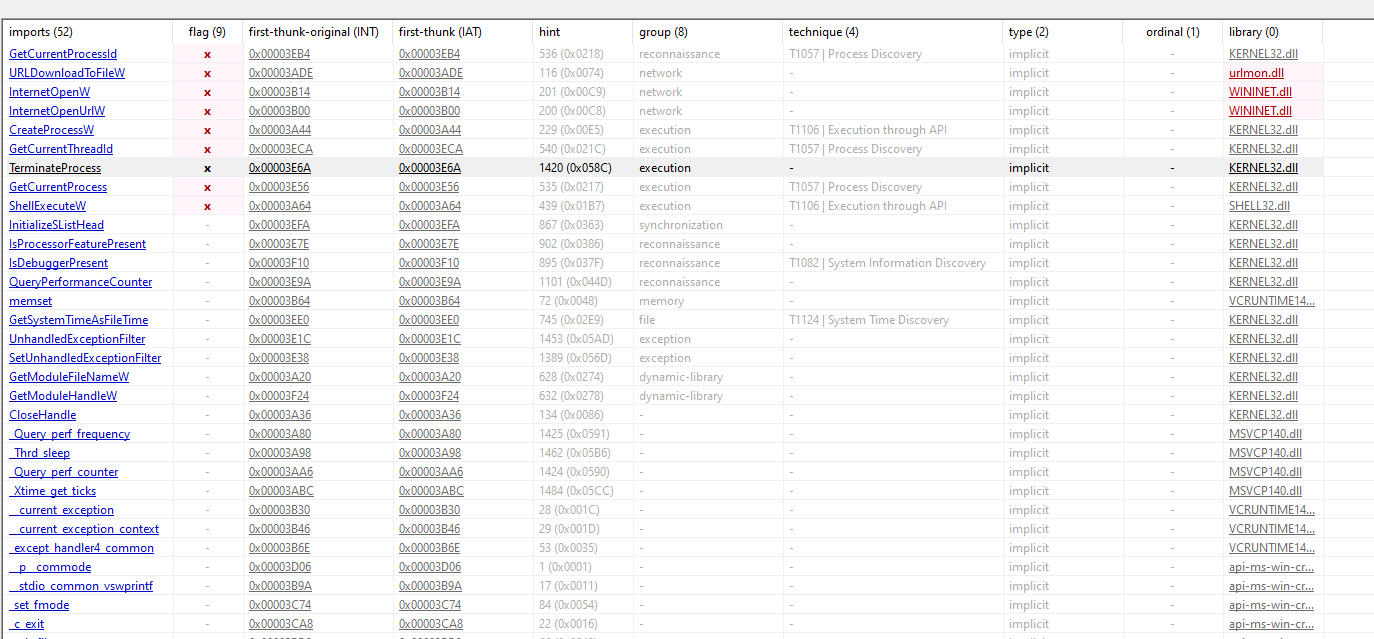


Libraries used:



\*urlmon.dll is a crucial Dynamic Link Library (DLL) in Microsoft Windows responsible for handling Uniform Resource Locators (URLs) and managing internet protocols. It plays a key role in URL parsing, internet communication, security zone determination, and Object Linking and Embedding (OLE) for embedding objects in documents.

Looking at the import table:



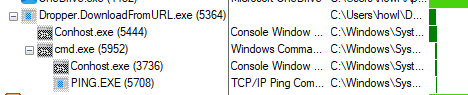
Looking at the strings output we see:

|  |
| --- |
| cmd.exe /C ping 1.1.1.1 -n 1 -w 3000 > Nul & Del /f /q "%s" |
| ping 1.1.1.1 -n 1 -w 3000 > Nul & C:\Users\Public\Documents\CR433101.dat.exe |
| C:\Users\Matt\source\repos\HuskyHacks\PMAT-maldev\src\DownloadFromURL\Release\DownloadFromURL.pdb |
| http://ssl-6582datamanager.helpdeskbros.local/favicon.ico |
| C:\Users\Public\Documents\CR433101.dat.exe |
| http://huskyhacks.dev |

# Basic Dynamic Analysis

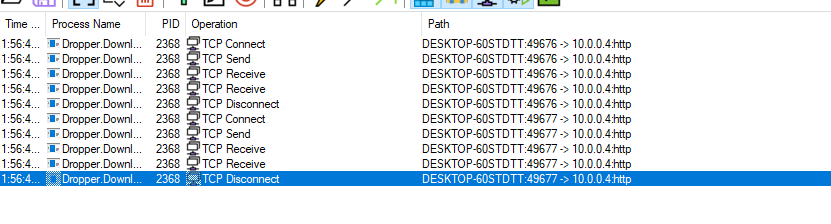
{Screenshots and description about basic dynamic artifacts and methods}

When running the file without internet connection:

1. It shows a black command line for a second and then it deleted itself.
2. 

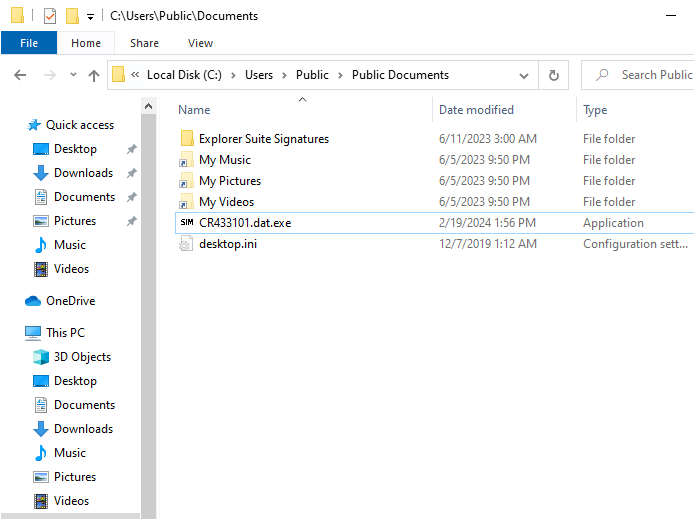
Running this command: Cmd.exe /C ping 1.1.1.1 -n 1 -w 3000 > Nul & Del /f /q "C:\Users\howl\Desktop\Dropper.DownloadFromURL.exe"

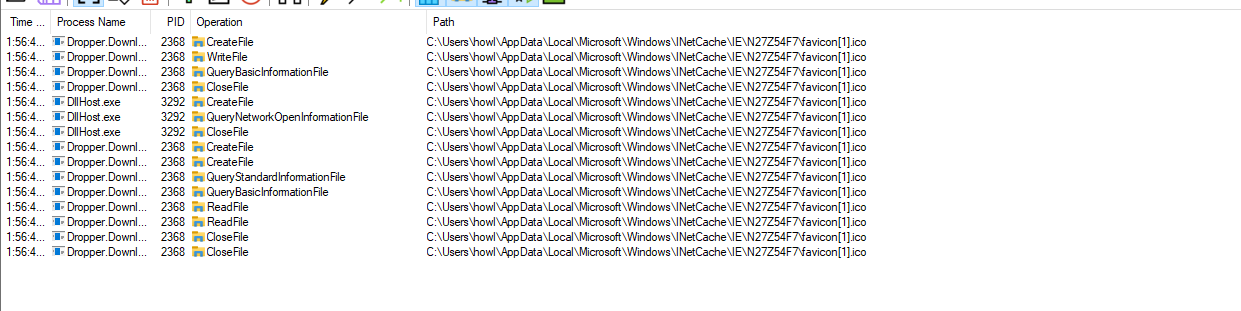
When there is internet connection:

1. There was a black command was showing, and everything looks normal, but the file is not deleted.
2. 

There were TCP traffic going to the internet.

1. Then it download/created these files in the system:





A screenshot of a computer

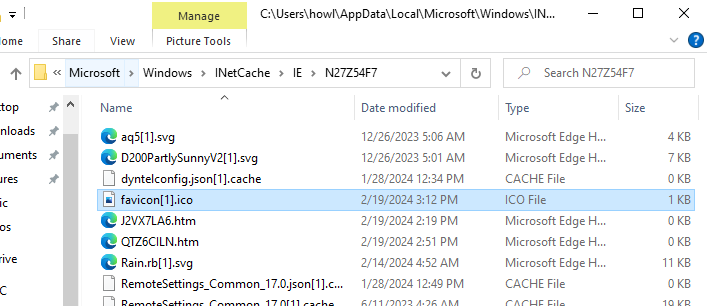
Description automatically generated

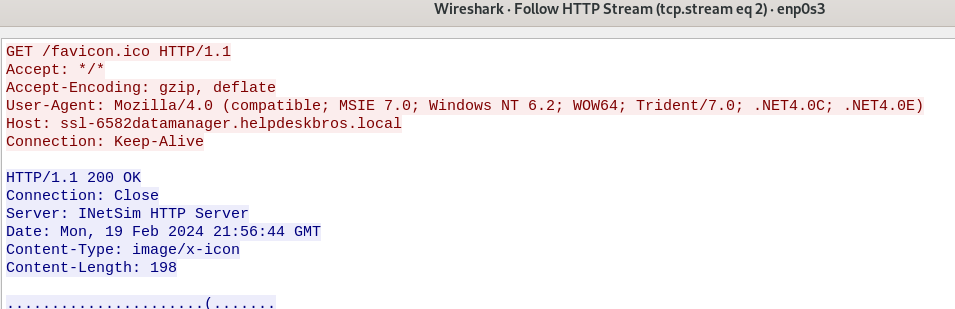
A screenshot of a computer

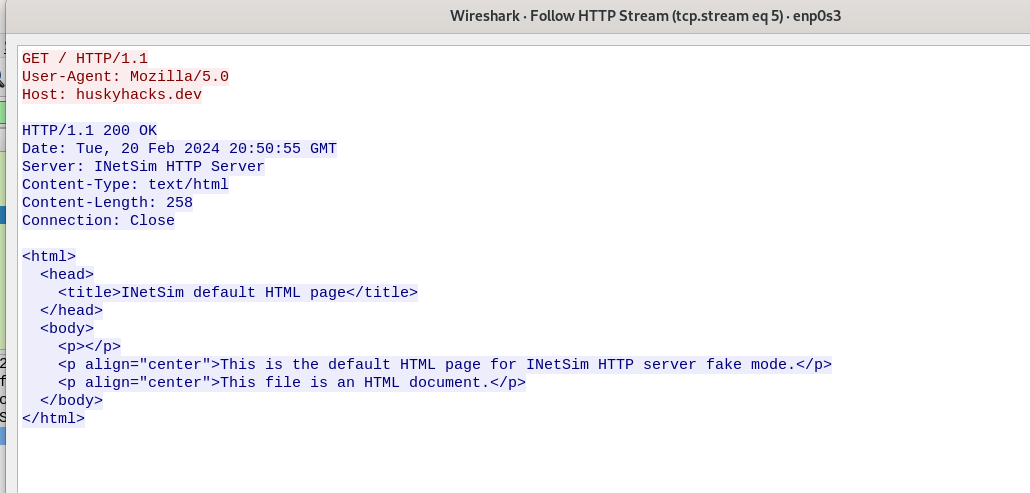
Description automatically generated

A screenshot of a computer

Description automatically generated



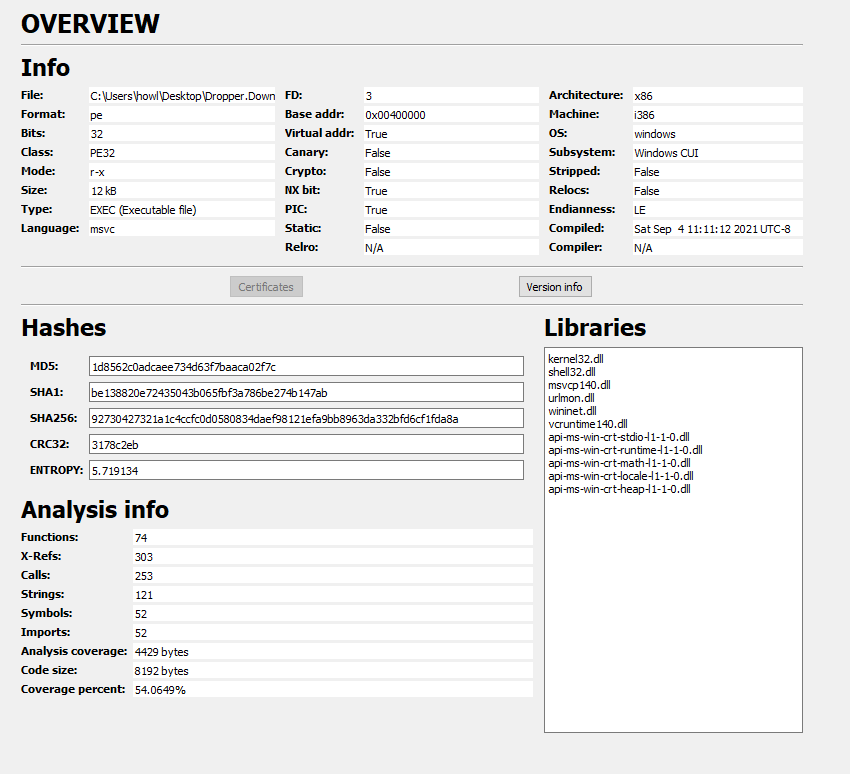




# Advanced Static Analysis

{Screenshots and description about findings during advanced static analysis} Could not

Looking at cutter info about this PE:



We now know it is built with Microsoft Visual C++

Looking at the main function:



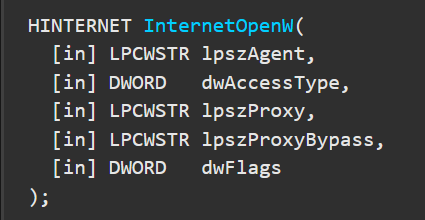
There are two functions calls:

1. InternetOpenW:

The function is used to Initializes an application's use of the WinINet functions.

The function returns an HRESULT value indicating the success or failure of the download operation.

The 5 parameters are pushed in the above as seen

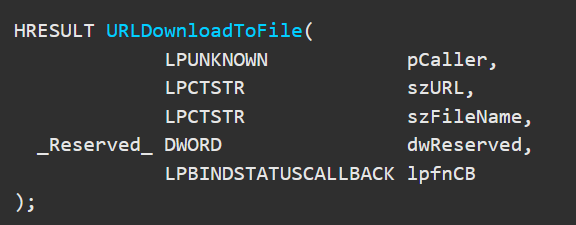


1. URLDownloadToFileW:

It is used to download a file from a specified URL and save it to a local file on the user's system. The "W" in the function name indicates that it is a Unicode (wide character) version of the function, supporting the use of Unicode characters in the URL and file paths.

The function returns an HRESULT value indicating the success or failure of the download operation.

The 5 parameters are pushed in the above as seen.



1. Then based on the last function “URLDownloadToFileW” result, it will branch out.

A red and green rectangles

Description automatically generated

th

If the result of the function result is false “not success connection” then the register that hold the result of the function EAX will be zero, and when the test is checked the ZF is set to1. And then jump if not equal means Jump to 0x401142 if Zero Flag is not set, which is not in this set, at the end the JNE instruction will not result in a jump.

Meaning it will only jump to the right branch if the ZF is not set, if the connection for “URLDownlodFromFileW” resulted in success, if we look at the code smippet:

1. “GetModuleFileName”:

* GetModuleFileName is a Windows API function that retrieves the full path of the executable file of a specified module (usually the current executable).
* Syntax: DWORD GetModuleFileName(HMODULE hModule, LPWSTR lpFilename, DWORD nSize);

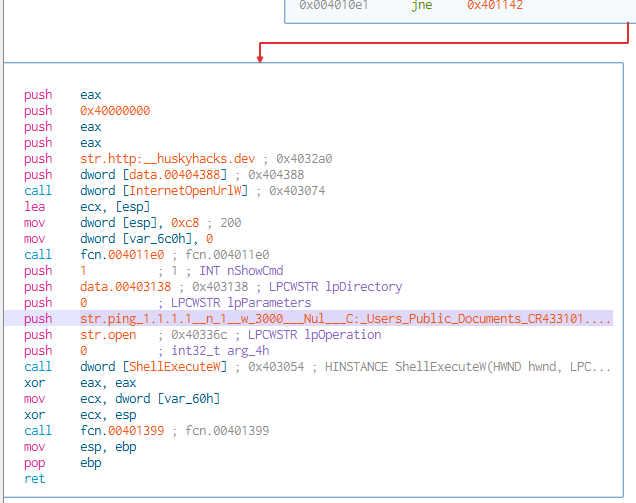
1. Command pushed as a parameter for the next function call, this one looks similar to strings extracted from before analysis: cmd.exe /C ping 1.1.1.1 -n 1 -w 3000 > Nul & Del /f /q "%s":

 ping 1.1.1.1 -n 1 -w 3000 > Nul: This pings the IP address 1.1.1.1 once with a timeout of 3000 milliseconds and discards the output.

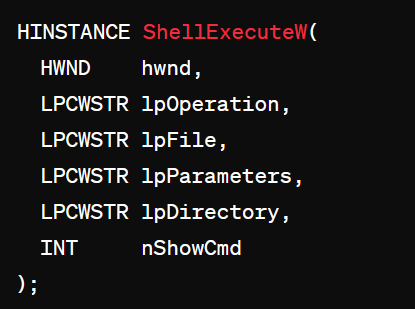
 Del /f /q "%s": Deletes the file specified by the %s placeholder. The /f and /q options force the deletion without prompting and in a quiet mode.

1. CreateProcessW: function that creates a new process and its primary thread.
2. CloseHandle: is a Windows API function used to close an open object handle.

The left branch is taken only if the functions call resulted in a failure and ZF is set,

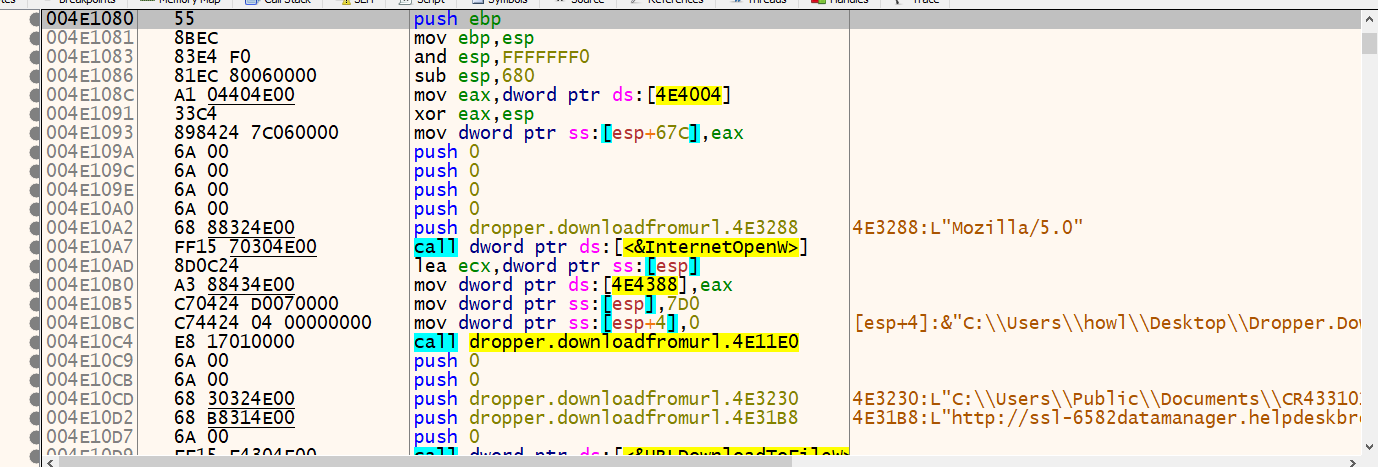


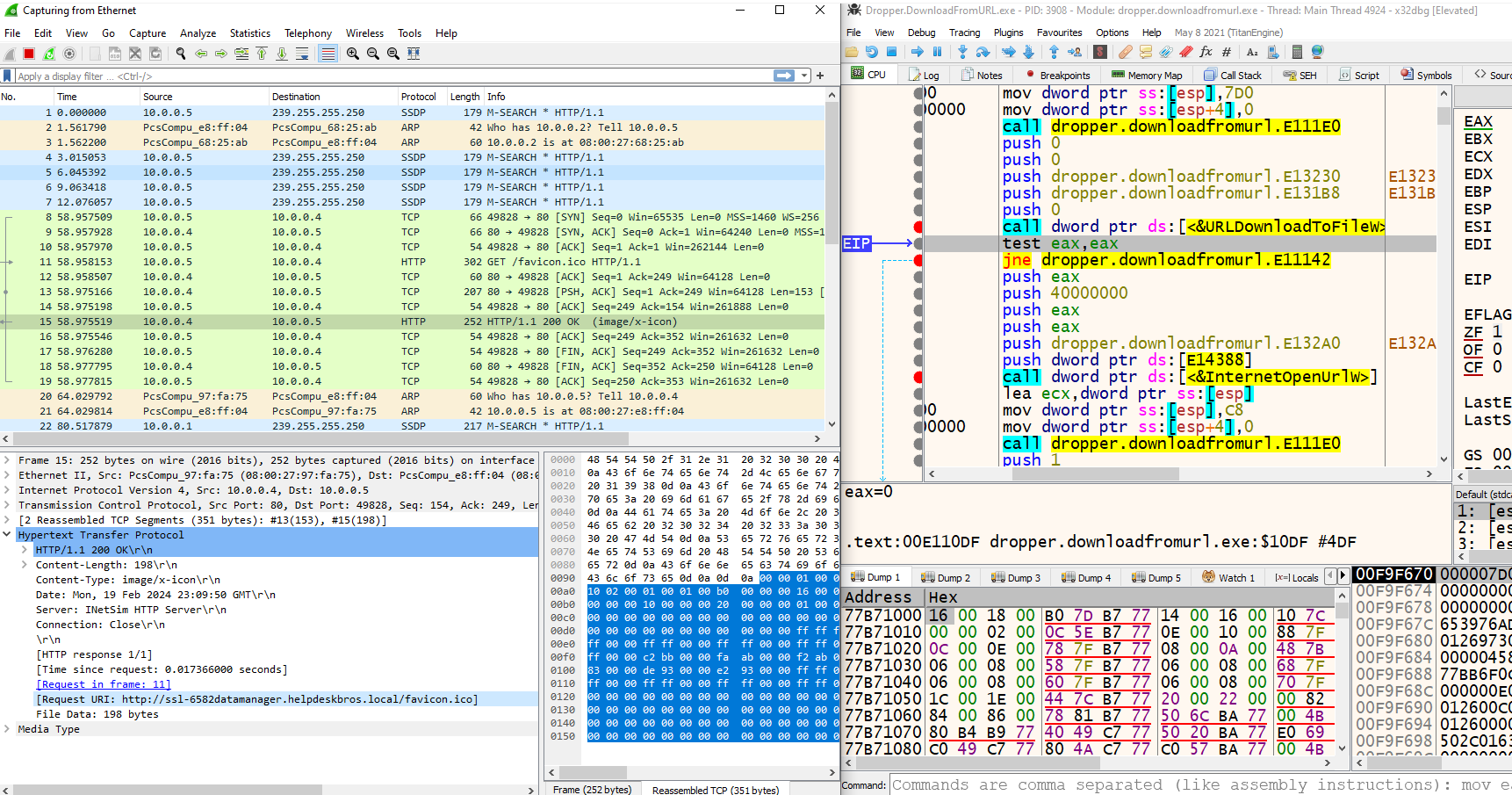
1. InternetOpenUrlW: It is used to open a URL and obtain a handle to the resource identified by the URL.
2. Command pushed as a parameter: ping 1.1.1.1 -n 1 -w 3000 > Nul & C:\Users\Public\Documents\CR433101.dat.exe
3. ShellExcuteW: Performs an operation on a specified file.



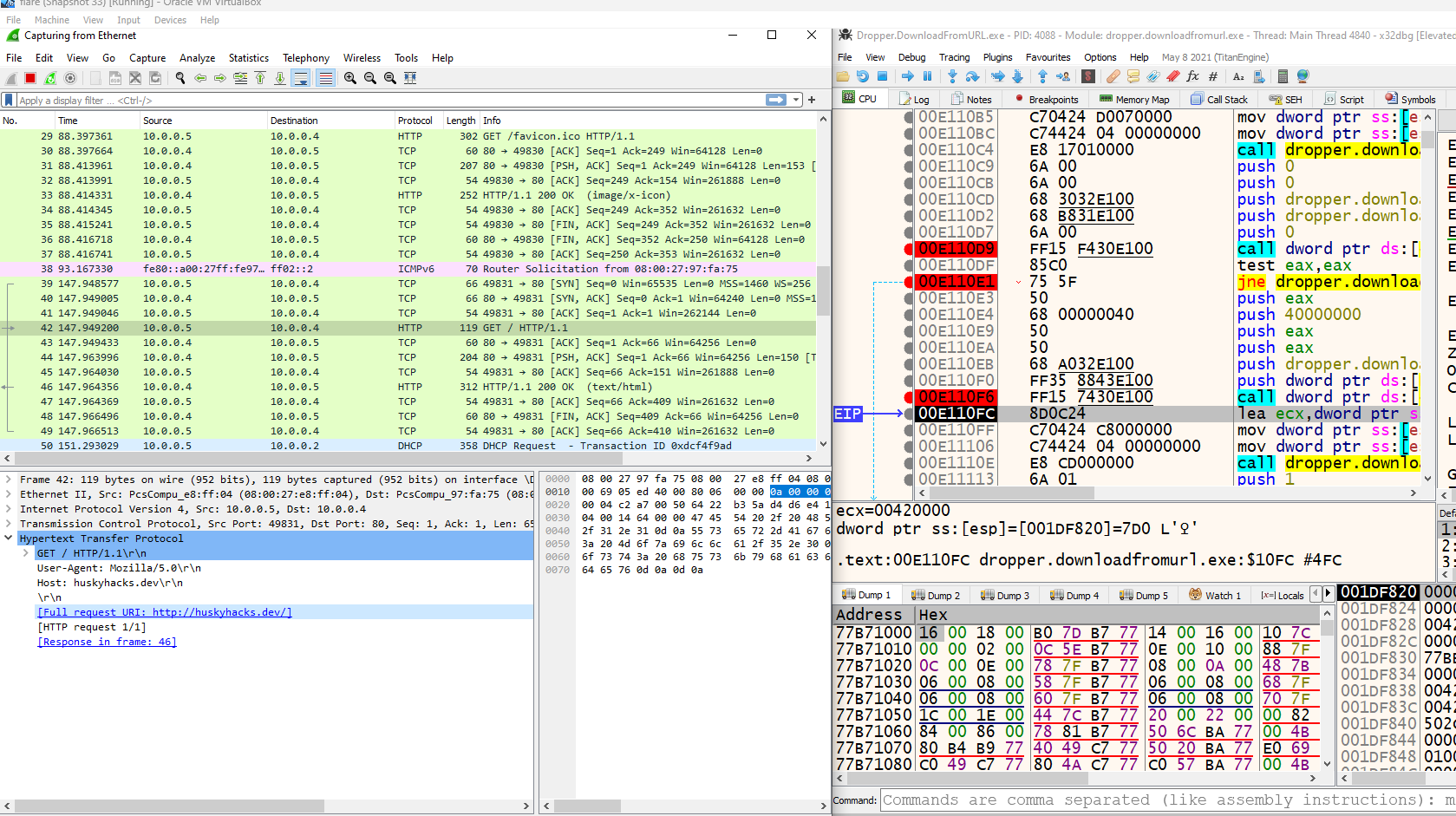
1. Then exited the same as the right branch.

# Advanced Dynamic Analysis





URLDownloadToFileW: the URL was 



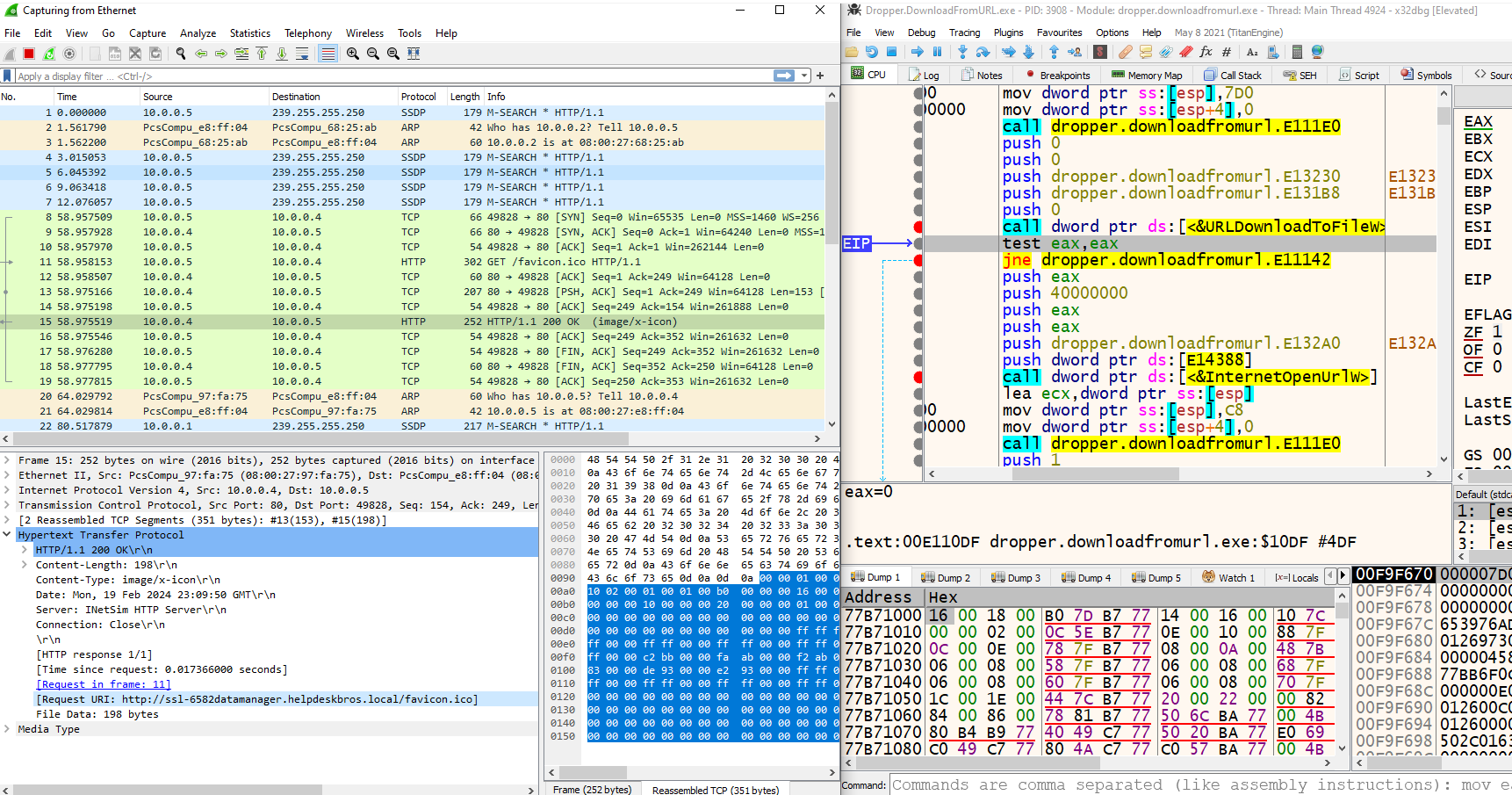
InternetOpenUrlW: <http://huskyhacks[.]dev>

# Indicators of Compromise

The full list of IOCs can be found in the Appendices.

## Network Indicators

{Description of network indicators}



A screenshot of a computer

Description automatically generated

A screenshot of a computer

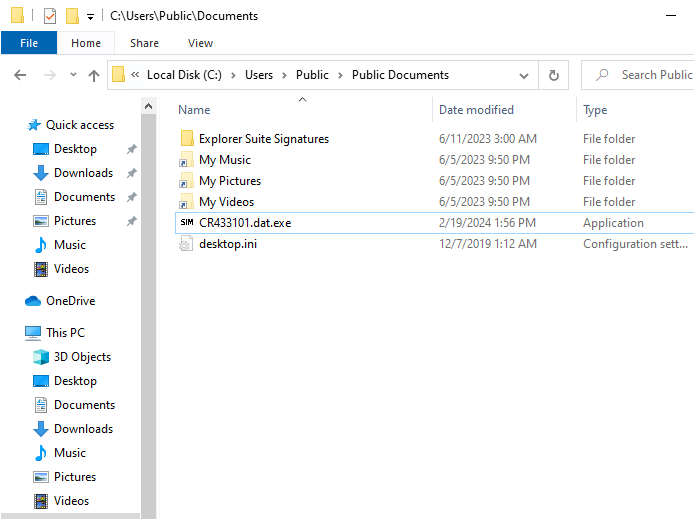
Description automatically generated

A screenshot of a computer

Description automatically generated

## Host-based Indicators

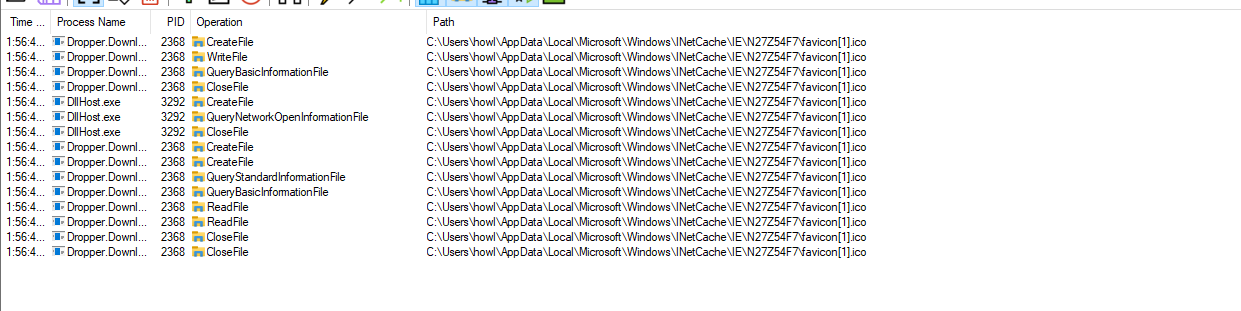
CR533101.dat.exe, this file is executed with the command shell as seen before





Favicon.ico

This is downloaded from URLDowloadFromFileW, the first call to check the condition if its able to download this file it will continour to drop the malicious file, if not able then it clean the file and exit out the program flow, and it is a simple icon file.





A screenshot of a computer

Description automatically generated



# Appendices

## Yara Rules

rule YARA\_example {

    meta:

        description = "dropper.downloadfromurl.exe"

        sha256 = "92730427321A1C4CCFC0D0580834DAEF98121EFA9BB8963DA332BFD6CF1FDA8A"

    strings:

        $string1="cmd.exe /C ping 1.1.1.1 -n 1 -w 3000 > Nul & Del /f /q '%s'" ascii

        $string2 = "ping 1.1.1.1 -n 1 -w 3000 > Nul & C:\\Users\\Public\\Documents\\CR433101.dat.exe" ascii

        $string3="favicon.ico" ascii

        $string4="CR433101.dat.exe" ascii

        $URL1="http://ssl-6582datamanager.helpdeskbros.local" ascii

        $URL2="http://huskyhacks.dev" ascii

        $IS\_PE\_FILE="MZ"ascii

        $Hex={75 00 73 00 6B 00 79 00 68 00 61 00 63 00 6B 00}

    condition:

        $IS\_PE\_FILE at 0  and

        ($string1 and $string2 and $string3 and $string4 ) or ($URL1 or $URL2)

        or $Hex

}

## Callback URLs

|  |  |
| --- | --- |
| Domain | Port |
| http://huskyhacks.dev | 80 |
| http://ssl-6582datamanager.helpdeskbros.local | 80 |

*https://www.virustotal.com/gui/file/92730427321a1c4ccfc0d0580834daef98121efa9bb8963da332bfd6cf1fda8a/detection*