DIGITAL FORENSICS LAB

PROJECT

Hassaan Jamil

I21-2774

Introduction:

This is a report about an incident that happened which resulted in damage to a system. The incident is uploaded at https://www.binary-zone.com/2023/03/27/challenge-7-sysinternals-case/ and is a forensics related case.

Expected Outcome:

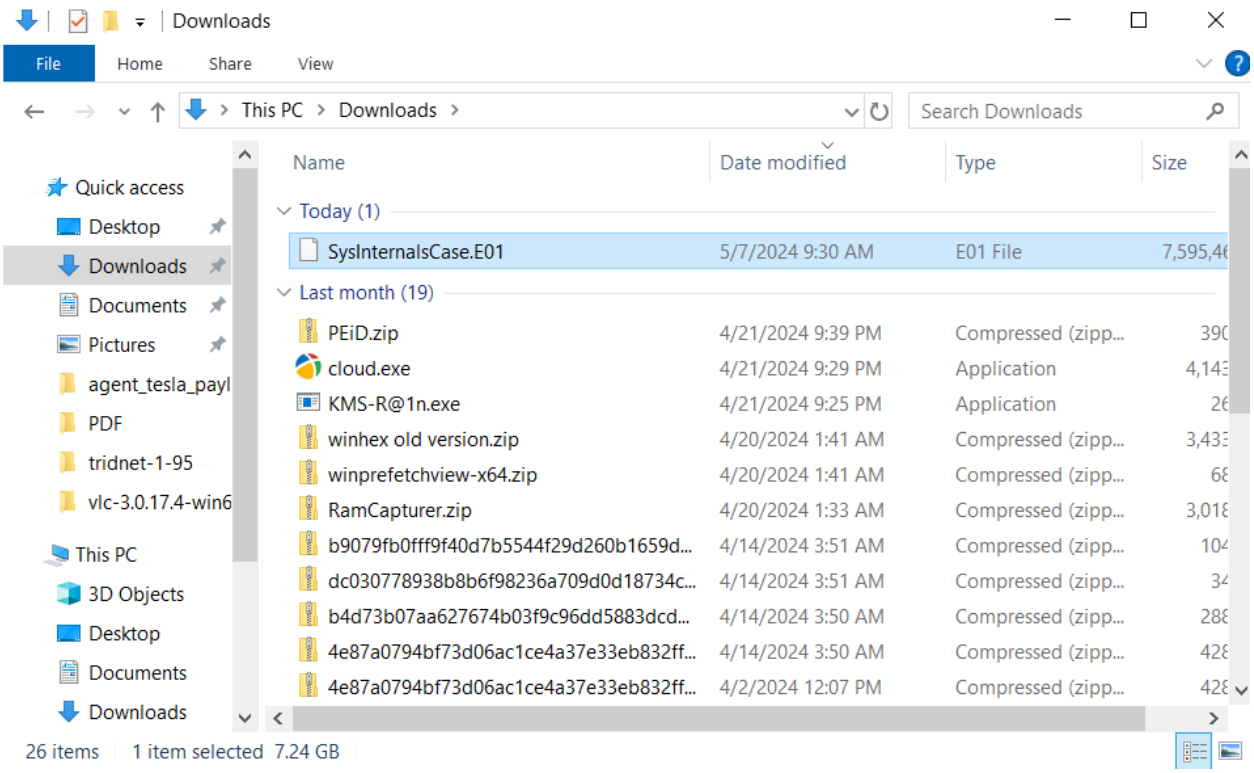
The expected outcome of this analysis is finding out what happened after the user ran the malicious file and when it happened.

Background

The user downloaded what they thought was the SysInternals tool suite, double-clicked it, but the tools did not open and were not accessible. Since that time, the user has noticed that the system has “slowed down” and become less and less responsive.

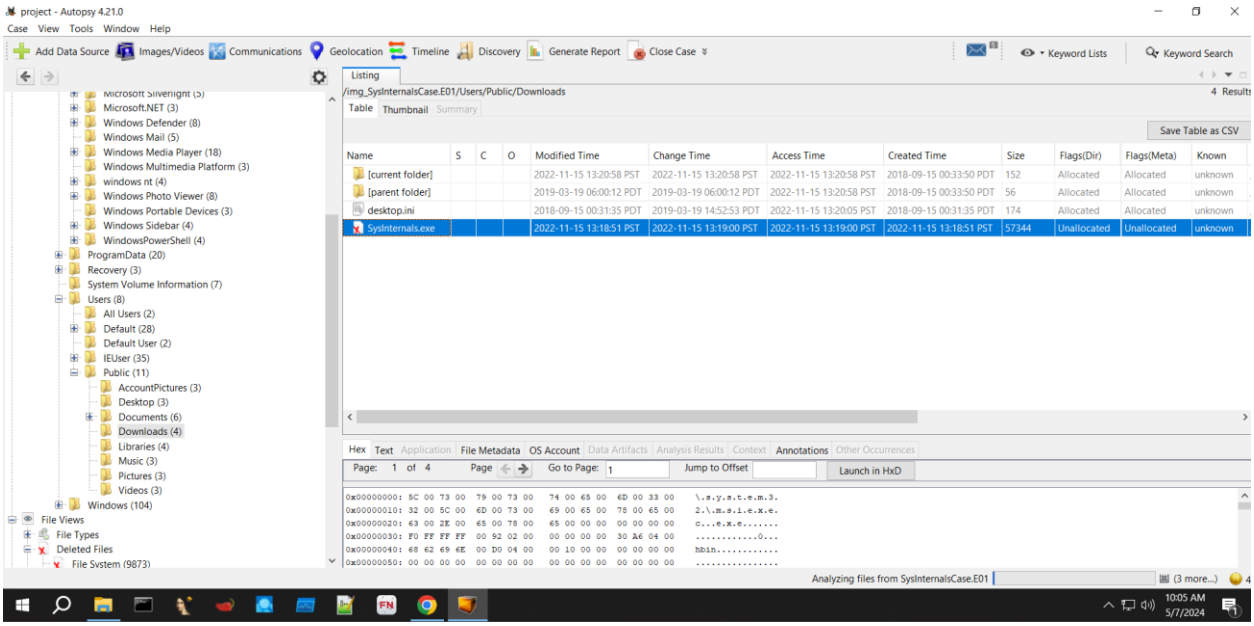
Analysis

The analysis starts by analyzing the disk image provided.

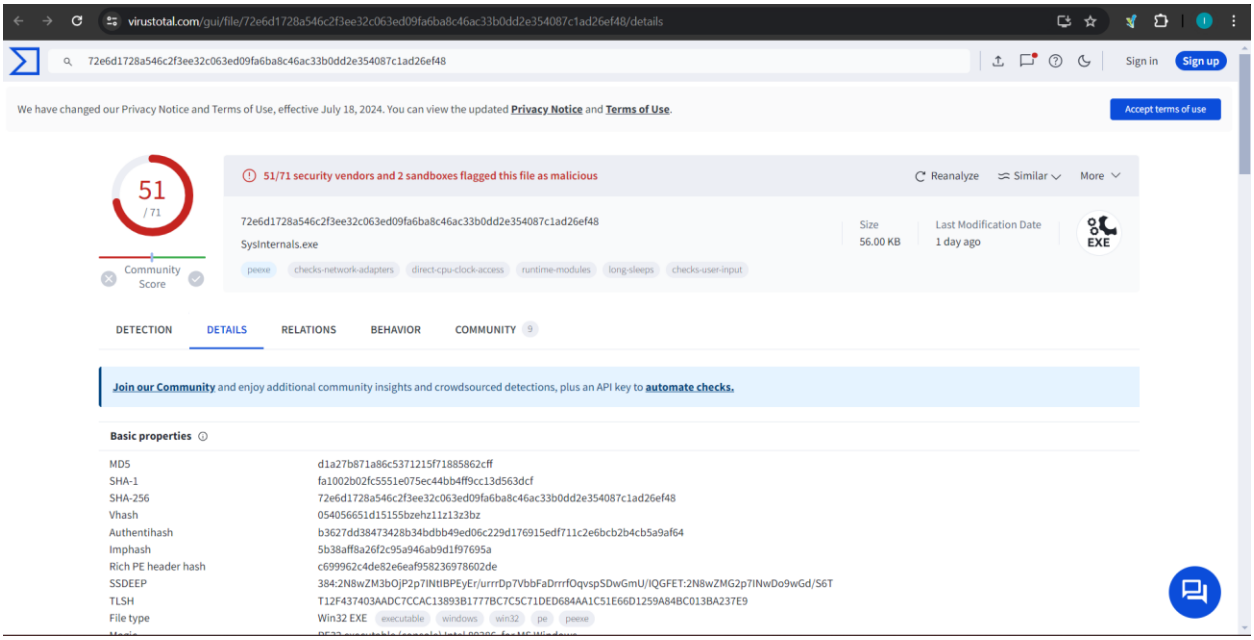


To analyze the disk image, **Autopsy** is used.

After searching the image, the file “Sysinternals.exe” was found in **C:\Users\Public\Downloads**

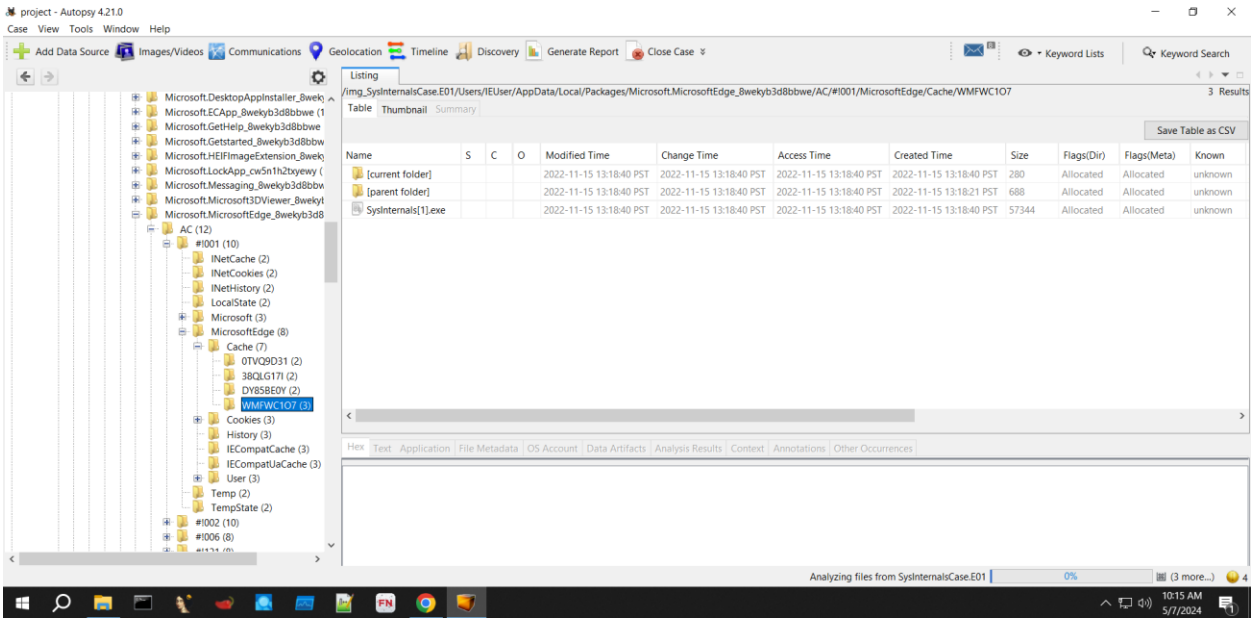


After finding the file**, VirusTotal** was used to extract some basic information of the file. Hash of file was put in the search box and the result was this:



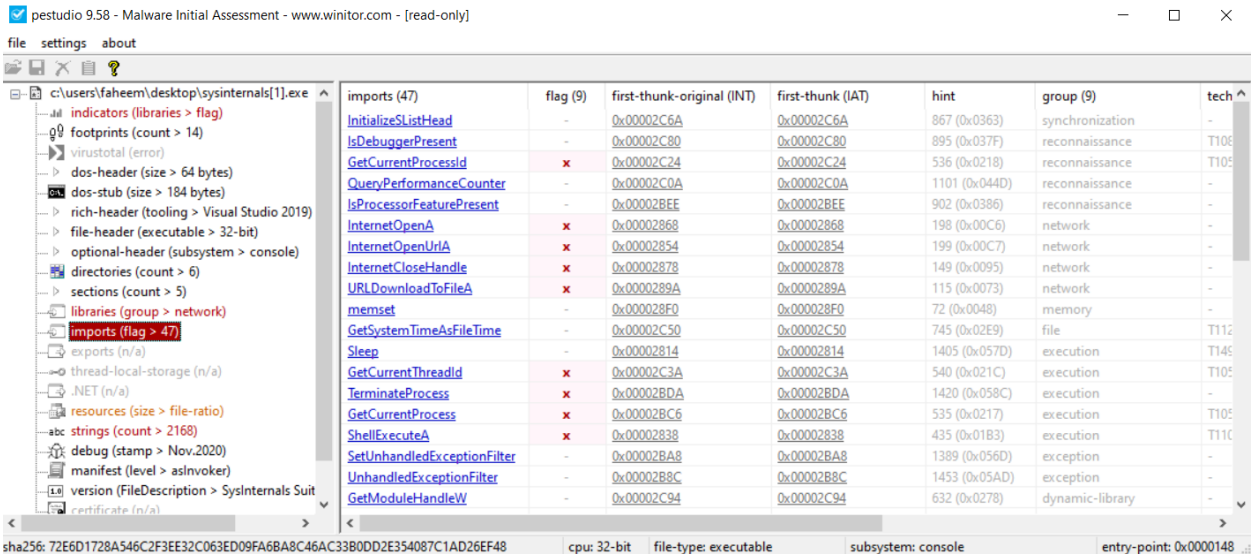
This shows that the file was malicious for sure. Secondly, this helped to find another malicious file that was created after running the malicious file **Sysinternals.exe.** This file was located in:

C:\Users\IEUser\AppData\Local\Packages\Microsoft.MicrosoftEdge_8wekyb3d8bbwe\AC#!001\MicrosoftEdge\Cache\WMFWC1O7\sysinternals[1].exe

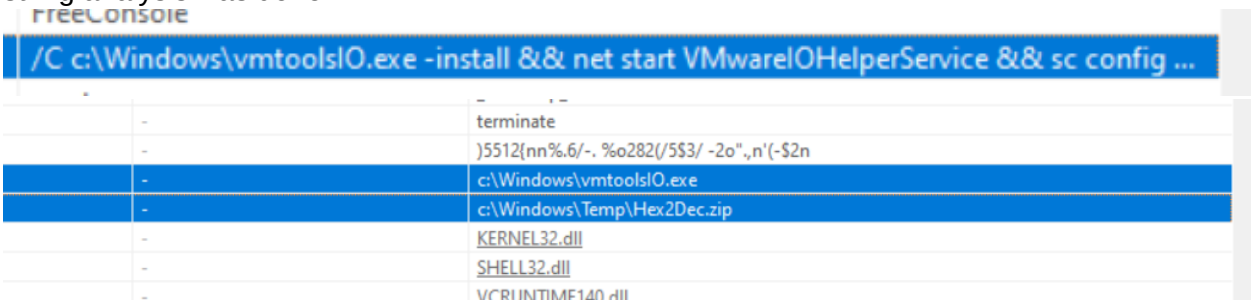


After identifying these files, we begin our analysis on them.

1- Used **PE Studio** to analyze the file **SysInternals[1].exe.**



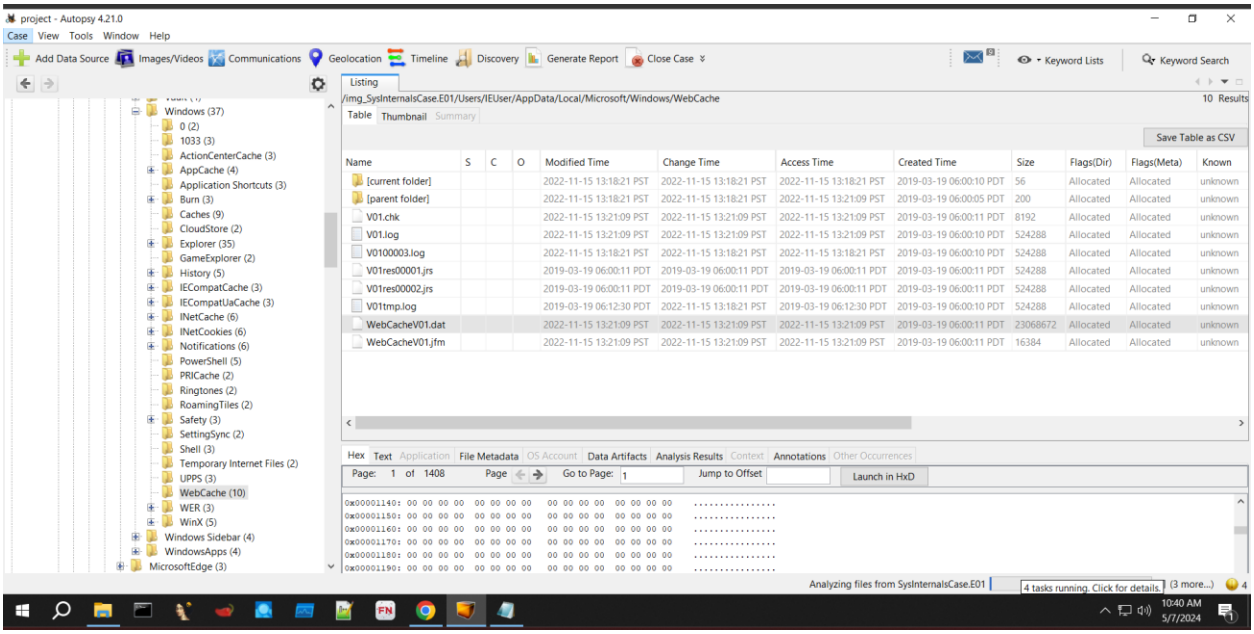
After looking at the imports of this file, imports such as URLDownloadToFileA, InternetOpenUrlA and ShellExecuteA were found that indicate that other files were also downloaded by this executable. To further look into the functionality of this malware, string analysis was done.



These strings were identified as suspicious. Furthermore, evasion techniques and other malware techniques were identified in strings:

ag (9)	label (64)	group (9)	technique (5)	value
x	import	reconnaissance	T1057 | Process Discovery	GetCurrentProcessId
-	import	reconnaissance	T1082 | System Information Discovery	IsDebuggerPresent
x	import	network	-	InternetOpenUrl
x	import	network	-	InternetOpen
x	import	network	-	InternetCloseHandle
x	import	network	-	URLDownloadToFile
-	file	network	-	WININET.dll
-	file	network	-	urlmon.dll
-	-	memory	-	memset
-	import	file	T1124 | System Time Discovery	GetSystemTimeAsFileTime
x	import	execution	T1106 | Execution through API	ShellExecute
x	import	execution	T1057 | Process Discovery	GetCurrentProcess
x	import	execution	-	TerminateProcess
x	import	execution	T1057 | Process Discovery	GetCurrentThreadId
-	-	execution	T1497 | Sandbox Evasion	Sleep
-	import	exception	-	UnhandledExceptionFilter

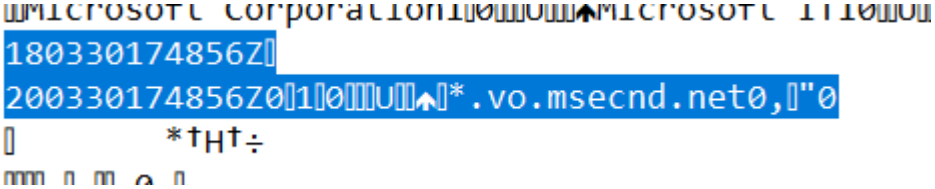
2- To further explore how the malware file got downloaded, analysis of web cache was done.



After opening the .dat file, I found a URL pointing to an official Microsoft sysinternals website.

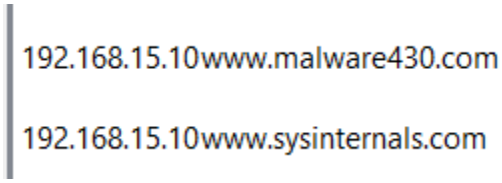
The date of the download was **March 30, 2018, 17:48:56 (UTC)**

Timestamp:



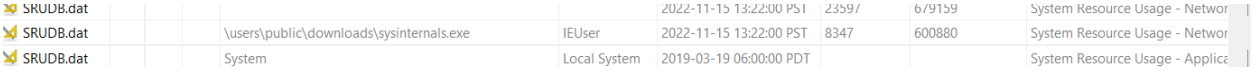
Even though the website is a legitimate Microsoft website, the user was redirected to a malicious website by modifying the **DNS** records, following is the proof:

hosts:



This justifies the download of the malware file as user was fooled by a **redirection attack**.

3- **Execution of the file:**



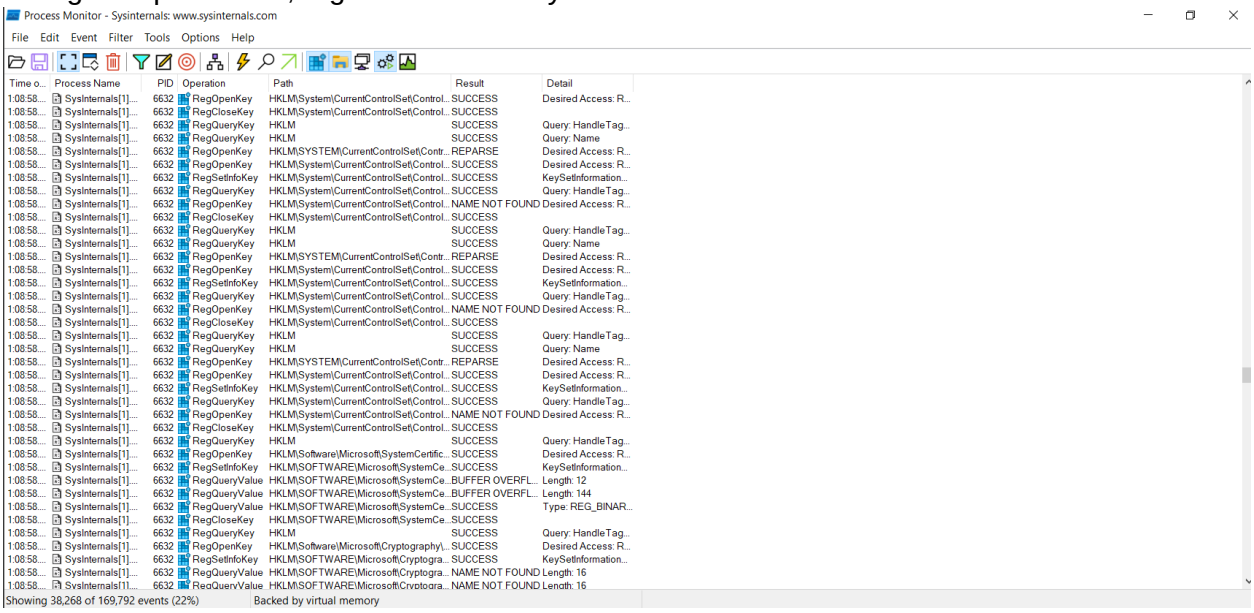
The file was executed by IEUser on 15th Novermber, 2022.

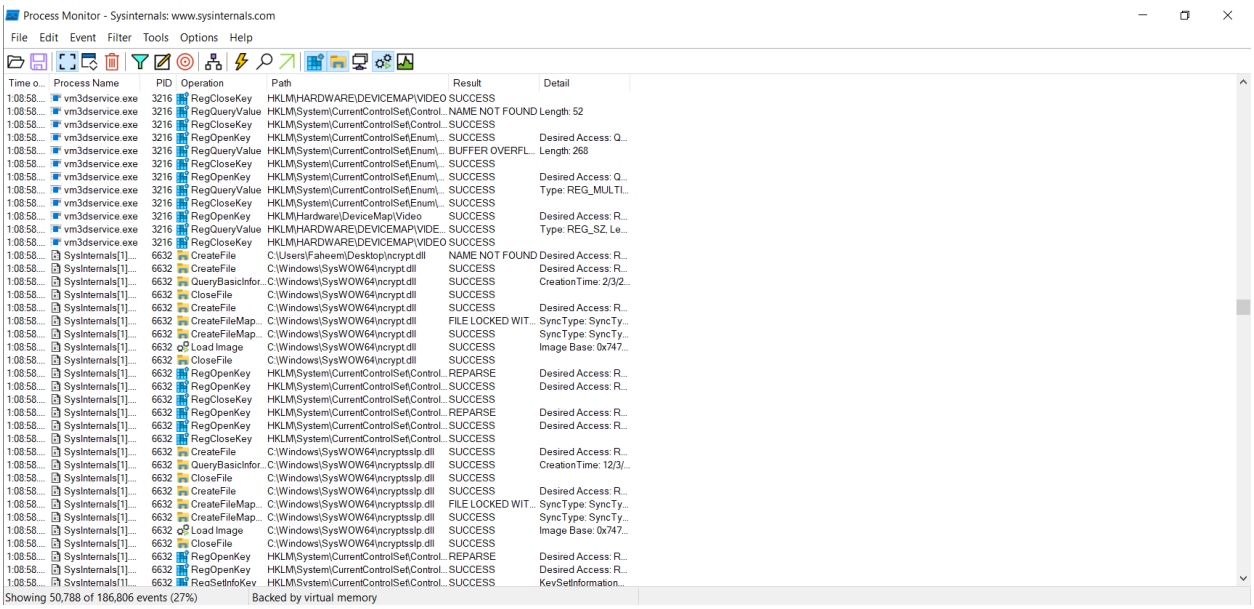
4- **SRUM**

This record makes sure that this executable did communicate with the systems and sent/received bytes to achieve its functionality.

5- **Dynamic Analysis**

After the static analysis, its time to run the file in a monitoring environment to see how the application affects the system in real time. For this, I used **Procmon** to monitor the changes in processes, registries and file system





One of the important functionality it did was execute a script in cmd.exe

```
C:\Windows\SysWOW64\cmd.exe

"C:\Windows\System32\cmd.exe" /C c:\Windows\vmtoolsIO.exe -i
nstall && net start VMwareIOHelperService && sc config VMwar
eIOHelperService start= auto
```

The executable also deleted Prefetch files and stopped the user from creating more Prefetch files, which explains why the system was getting slower.

This picture is taken from VirusTotal.

Conclusion:

To summarize the flow of events, I will conclude this analysis with the steps:

1- Downloaded malware from a redirected DNS record
2- Execution
3- The .exe downloaded multiple files to execute
4- Evaded the Antivirus protection by sleeping
5- Executed
6- Scripting
7- Deleting Prefetch files
8- Changes in Registry
9- Shutdown of system