**Sample 1**

1. When was this file compiled?

Ans: May 14, 2009 5:12:41 PM. The value is given as 4A0C5109 which is in Hexadecimal format in the file header under the “TimeDateStamp” portion and converting that gives this particular time. 

2. List a few imports or sets of imports and describe how the malware might use them.

a. CreateProcessA (from kernel32.dll): This function creates a new process and its primary thread. The malware might use CreateProcessA to launch other programs or executable payloads as part of its operation, enabling it to execute additional commands or load other malware components.



b. GetComputerNameA (from kernel32.dll): This function retrieves the name of the local computer.The malware could use GetComputerNameA for reconnaissance purposes. This function helps gather system information, which could be sent to a command-and-control (C2) server to customize its behavior based on the host environment.



c. ShellExecuteExA (from shell32.dll): This function performs an operation on a specified file, such as opening, printing, or executing it. With this the malware might attempt to open or execute additional files, scripts, or payloads. This function can be employed to spread the infection by launching scripts or opening URLs in the default web browser.



3. What are a few strings that stick out to you and why?

a. <http://www.ueopen.com/test.html>

This string indicates a potential network indicator or URL that the malware may be using for communication, likely connecting to a remote server as part of its C2 (command-and-control) infrastructure or for data exfiltration. The presence of a URL suggests that the malware has networking capabilities and may attempt to contact this domain to send or receive instructions.



b. cmd.exe exit

This string points to the use of command-line operations, which the malware could use for executing system commands, deleting files, or modifying the system environment. Using cmd.exe allows malware to interact with the operating system and carry out various tasks such as file deletion or script execution. This suggests that the malware may leverage command-line commands to perform tasks such as deleting evidence, modifying the system, or executing other commands as part of its payload.



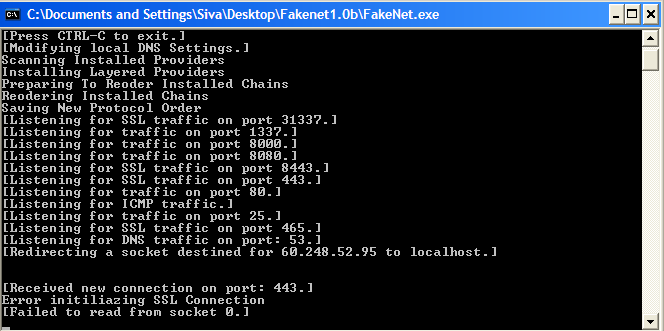
c. Open > nul /c del COMSPEC

This string indicates an attempt to suppress output while executing a del (delete) command targeting the COMSPEC environment variable, which points to the command-line interpreter path (e.g., C:\Windows\System32\cmd.exe). This could be a technique to delete or tamper with the command processor's settings, making system recovery or further analysis difficult.

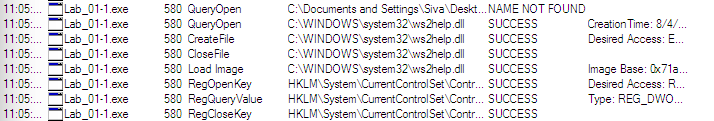


4. What happens when you run this malware? Is it what you expected and why?

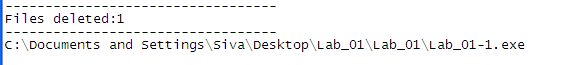
Ans: Quite a number of things happen when the malware file is executed. It tried to establish a connection with the internet via the 443 port which is denoted in the figure below via the fakenet screenshot.



Next it opened the terminal (shown in process explorer) and then the whole malware terminated. And various operations are done by the process among which some notable ones include creating and closing files with the operations named CreateFile, CloseFile, QueryOpen and some registry related operations like RegOpenKey, RegQueryValue, RegCloseKey and etc. The operations are denoted below in the screenshot of Process Monitor.



And through RegShot it was noticed that quite some processes were done with multiple changes in the system along with the malware file deleting itself after a certain point.



5. Name a procmon filter and why you used it.

A procmon filter named “Process Name” would be useful enough to look for the processes done by a certain file having the name as it is denoted. It helps to separate out the process and focus on the desired project easily.

6. Are there any host­based signatures? (Files, registry keys, processes or services, etc).

If so, what are they?

Some of the host-based signatures are:

7. Are there any network based signatures? (URLs, packet contents. etc) If so, what are

they?

8. Is there anything that impeded your analysis? How so? How might you overcome this?

9. What do you think is the purpose of this malware?

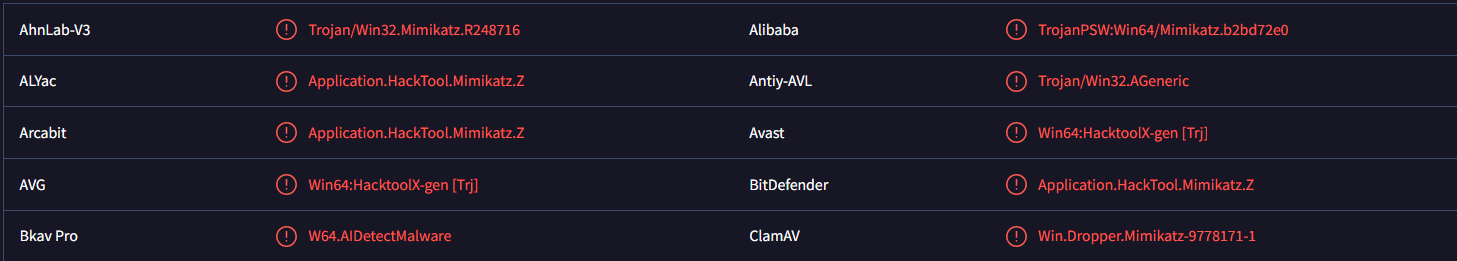
**Sample 2**

1. What is the md5sum? What of interest does VirusTotal Report?

Ans: The md5sum of the malware file is: 02658BC9801F98DFDF167ACCF57F6A36.



In VirusTotal report, the md5sum is termed as a Trojan and a Hacktool.



2. List a few imports or sets of imports and describe how the malware might use them.

a. CreateProcessA (from kernel32.dll)

This function creates a new process and its primary thread, which can execute a specified program. The malware might use CreateProcessA to launch other executables, either legitimate ones to mask its behavior or additional malicious payloads to extend its functionality or establish persistence.

b. HttpSendRequestA (from WinInet32.dll)

This function sends an HTTP request to a specified server and is often used to initiate communication with remote servers. In the context of malware, HttpSendRequestA could be part of a routine to send data to or receive commands from a command-and-control (C2) server. It indicates the potential for data exfiltration or remote-control capabilities, which is typical for malware that needs to communicate over the internet.

c. fopen (from MSVCRT.dll)

This function opens a file with specified access permissions (e.g., read, write). With the help of this the malware might read from or write to files on the system. This could be used for logging data, storing configuration files, or accessing files to modify or steal information. File manipulation functions like fopen are essential for understanding how the malware interacts with the file system and handles data.

3. What are a few strings that stick out to you and why?

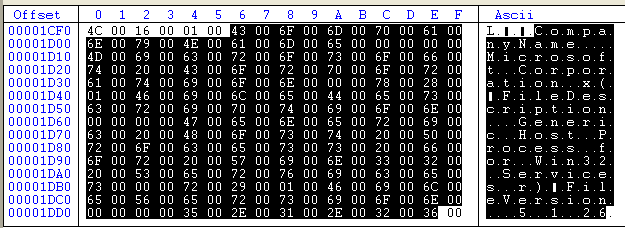
a. “HttpSendRequestA” : These functions from WININET.dll indicate that the program might establish an HTTP connection. Such functions are commonly used by malware for communicating with a Command and Control (C2) server, downloading additional payloads, or exfiltrating data.



b. “GetWindowsDirectoryA”: This function is often used to identify the Windows directory, which may be part of a strategy to locate system-specific files or to store data where it blends in with legitimate system files.



c. “CompanyName: Microsoft Corporation, FileDescription: Generic Host Process for Win32 Services, OriginalFilename: svchost.exe”: The inclusion of these specific strings appears to be an attempt to make the binary look legitimate by mimicking svchost.exe, a core Windows process. Malware frequently uses names, descriptions, and metadata of trusted executables to evade detection.



4. What happens when you run this malware? Is it what you expected and why?

5. Name a procmon filter and why you used it.

6. Are there any host­based signatures? (Files, registry keys, processes or services, etc).

If so, what are they?

7. Are there any network based signatures? (URLs, packet contents. etc) If so, what are

they?

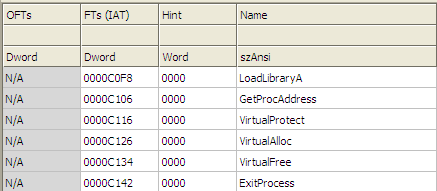
8. Is there anything that impeded your analysis? How so? How might you overcome this?

9. What do you think is the purpose of this malware?

**Sample 3**

1. Are there any indications that this malware is packed? What are they? What is it

packed with?

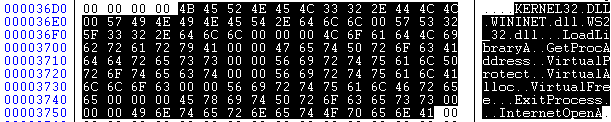


LoadLibraryA and GetProcAddress: These functions are often used by packers to load additional libraries and resolve function addresses dynamically, avoiding the need for a comprehensive import table. This indicates that the malware is packed.

2. Are you able to unpack it? Why or why not?

Ans: No, I'm not able to unpack it with certainty based on the information provided. The presence of minimal imports like LoadLibraryA and GetProcAddress, along with the .rsrc section possibly holding packed code, suggests that this malware might be using a custom packer or obfuscation method. Without running it in a debugger to observe the unpacking process or dealing with potential anti-debugging techniques, it's challenging to unpack it statically with just tools like CFF Explorer.

3. What are a few strings that stick out to you and why?



a. VirtualProtect: This function changes the protection on a region of memory (e.g., from read-only to executable). Malware frequently uses VirtualProtect to make regions of memory executable after unpacking or decrypting payloads, which allows it to run code that was initially hidden or obfuscated. This can be a sign of code injection or unpacking behavior.

b. InternetOpenA: This function initializes an internet connection and is part of the WININET.dll library. Its presence indicates that the malware may try to establish network communication. Combined with WININET.dll, this suggests that the malware could attempt to connect to remote servers, possibly for data exfiltration, receiving commands, or downloading additional payloads.

c. LoadLibraryA and GetProcAddress: These functions allow dynamic loading of libraries and functions at runtime. This is commonly seen in packed or obfuscated malware, as they allow the program to load additional functions without having them explicitly imported. This behaviour helps evade static analysis and can indicate the use of dynamic or evasive techniques.

4. What happens when you run this malware? Is it what you expected and why?

5. Are there any host­based signatures? (Files, registry keys, processes or services, etc).

If so, what are they?

6. Are there any network based signatures? (URLs, packet contents. etc) If so, what are

they?

7. Is there anything that impeded your analysis? How so? How might you overcome this?

8. What do you think is the purpose of this malware?