Analyzing WhisperGate and BlackCat Malware: Methodology and Threat Perspective

Mathew Nicho[1], Rajesh Yadav[2], Digvijay Singh[3]
Research and Innovation Centre, Rabdan Academy
Abu Dhabi, United Arab Emirates[1]
Department of Computer Science and Engineering, GITAM Univeraity
Visakhapatnam, India[2]
Department of Computer Science and Engineering, BML Munjal University
Gurgaon, India[3]

***Abstract*—The increasing use of powerful evasive ransomware malware in cyber warfare and targeted attacks is a persistent and growing challenge for nations, corporations, and small and medium-sized enterprises. This threat is evidenced by the emergence of the WhisperGate malware in cyber warfare, which targets organizations in Ukraine to render targeted devices inoperable, and the BlackCat malware, which targets large organizations by encrypting files. This paper outlines a practical approach to malware analysis using WhisperGate and BlackCat malware as samples. It subjects them to heuristic-based analysis techniques, including a combination of static, dynamic, hybrid, and memory analysis. Specifically, 12 tools and techniques were selected and deployed to reveal the malware's innovative stealth and evasion capabilities. This methodology shows what techniques can be applied to analyze critical malware and differentiate samples that are variations of known threats. The paper presents currently available tools and their underlying approaches to performing automated dynamic analysis on potentially malicious software. The study thus demonstrates a practical approach to carrying out malware analysis to understand cybercriminals' behavior, techniques, and tactics.**

***Keywords*—Malware analysis; WhisperGate; BlackCat; malware sample; ransomware**

I. Introduction

The geopolitical events in Ukraine at the start of 2022 were preceded by the devastating cyber warfare operation highlighted by WhisperGate malware (A malware that corrupts a system's master boot record, displays a fake ransomware note, and encrypts files based on certain file extensions). WhisperGate is considered dangerous because it can launch cyber-attacks and compromise sensitive information against hardened targets. Since it was deployed in cyber warfare against Ukraine, it could exploit unknown vulnerabilities in a target's security systems and cause significant harm. The destructive capabilities of WhisperGate make it a threat to individual, organizational, and national security. At the end of 2021, a sophisticated malware called BlackCat also known as "AlphaV," emerged, targeting U.S. organizations and their affiliates in Europe, the Philippines, and other locations. While WhisperGate masquerades as ransomware targeting nation-states (in this case, Ukraine), BlackCat has emerged as deadly ransomware targeting U.S. and European retail, construction, and transportation organizations. BlackCat appeared as an innovative ransomware-as-a-service (RaaS) group leveraging the Rust programming language and offering affiliates 80% to 90% of ransom payments [1]. Affiliates included Germany's tank storage and terminal firm Oiltanking and energy firm Mabatnaft, Belgian energy firm Sea-Invest, and Dutch oil and gas firm Evos. These attacks underlined the growing vulnerability of critical infrastructure companies to malicious hackers [2]. Both pieces of malware were challenging to defend against due to their elusive and evasive nature, which intrigued cybersecurity analysts worldwide. Since the top three cyberattacks that organizations are most concerned about are ransomware, social engineering, and malicious insider activities [3], WhisperGate and BlackCat were ideal candidates for our practical malware analysis approach due to their stealth and evasive capabilities and the destructive consequences they can cause.

Multiple malware classes, such as worms, viruses, spyware, Trojan horses, rootkits, ransomware, keyloggers, and adware, are designed with specific functionalities namely data exfiltration, data encryption, and data destruction. Despite the widespread use of antimalware software, the number of malware infections continues to grow. Malware, especially zero-day malware, can evade antimalware solutions and even infect them with its built-in defensive mechanisms. Along with WhisperGate, malware deployed against Ukraine included HermeticWiper, IsaacWiper, HermeticWizard, and CaddyWiper. Once inside the initial network, it leverages that access to compromise user and administrator accounts in the active directory of Windows' server and configures malicious group policy objects through Windows' task scheduler [4].

This paper provides a practical approach to performing malware analysis using integrated tools and techniques to assess WhisperGate and BlackCat. The Microsoft Threat Intelligence Centre disclosed that WhisperGate, categorized as a wiper, targeted several organizations in Ukraine and was tracked as DEV-0586 with a design similar to ransomware but lacking a recovery mechanism [5]. BlackCat, which belongs to a sophisticated ransomware as a service (RaaS) family, extorts money from targeted institutions instead.

The structure of the remainder of this paper is as follows: Section II discusses extant literature on ransomware in general, the identified ransomware, and an evaluation of malware analysis techniques. Section III outlines the methodology, and Section IV focuses on the experimental analysis. Section V discusses the results, and Section VI concludes the research with suggestions on future research.

II. Literature Review

The section provides an overview of ransomware, discusses two specific types of malware (WhisperGate and BlackCat), and then evaluates malware analysis methodologies to focus on the appropriate technique(s).

A. Ransomware

Ransomware is considered one of the most threatening types of malware. Its attacks increased by 151% in 2012, averaging 270 cyberattacks per organization, with each successful breach resulting in a cost of $3.6 million for the affected company [3]. Cyber-attacks predominantly occur through ransomware, social engineering, and malicious insider activity [3]. In particular, ransomware leverages social engineering methods to gain unauthorized access to the victim’s network. Once an infection is spread, the user is extorted and asked for a monetary payment against the locked access [6], but there is no guarantee that they will regain access to their locked files after paying the ransom. Threat actors often receive the payment but still retain the data. These cybercriminals often request payment in cryptocurrency, as it is untraceable and allows them to evade responsibility [7]. Malwares leverage the Trojan by disguising themselves as legitimate software and download the malicious components, which negatively impact the system and tend to infect files and target other systems [8]. While commercial solutions are available, these are not 100% secure, because hackers use more sophisticated techniques to follow the evolution and bypass the protection techniques [9]. WhisperGate is classified as a wiper, i.e., it disguises itself as ransomware but instead aims to cause mass destruction by wiping out hard drives at targeted organizations [10].

Removing the ransomware or restoring the infected devices is ineffective, as the ransomware uses asymmetric cryptography [11], which makes it robust. The encryption makes it so that the victim is unable to access the data without first decrypting it using a key [12]. Threat actors usually ask for a ransom in exchange for the decryption key and target organizations that handle large amounts of sensitive data. The victim is faced with inaccessibility and damage to their data and often pays the ransom demand. Since most of the victims are threatened with their data and sensitive information being exposed [13].

Among the five types of ransomware—locker, crypto leakware, scareware, and pseudo-ransomware [14], WhisperGate comes under the pseudo-ransomware category, while BlackCat comes under leakware category. Also known as doxware, leakware presents a high-risk level because it is well-targeted to institutions such as banks or those that work with confidential and critical data. This ransomware does not destroy the data but threatens to release them into the public domain. Furthermore, since the context can damage the institution’s image, an even greater emphasis can be placed on the quick payment of a ransom. Accordingly, BlackCat operates as a RaaS option that permits earning a percentage of the ransom payment to all the persons who have low technical knowledge about how to create ransomware but are members of this network. It is only necessary that those members spread the ransomware as far as possible while the RaaS vendor can focus on how to make this malicious software cause even more damage.

B. WhisperGate

Unlike traditional ransomware campaigns where the motive is clear, the BlackCat campaign is believed to be pseudo, with its intention being to cause the destruction of infected systems, as evidenced by the Stage 4 wiper that overwrites data on the victim’s system, making decryption impossible [15]. The malware that was explicitly launched against various Ukrainian organizations in geopolitically motivated attacks was first analyzed by the Microsoft Threat Intelligence Center and detected on January 13, 2022 [16]. The BlackCat ransomware campaign targeted Ukraine in 2021 prior to its physical invasion, but it was detected and neutralized before causing any severe damage [17]. Russian cyber operations have targeted Ukraine with destabilization efforts for years through attacks on critical infrastructure, influence operations, website defacement, and attacks against banks and military networks [16]. WhisperGate, while masquerading as ransomware, corrupts a system’s master boot record, displays a fake ransomware note, and then encrypts files based on specific file extensions. While a ransomware message is displayed during the attack, the targeted data is destroyed and is not recoverable even if a ransom is paid [18]. The multi-stage infection chain downloads a payload that wipes the master boot record (MBR). Then, it downloads a malicious Dynamic-link library (DLL) file hosted on a discord server (a platform where people can interact with each other in real time), which drops and executes another wiper payload that destroys files on the infected machines [19]. The malware, which is designed to look like ransomware, is intended to render the targeted devices inoperable rather than to obtain a ransom, as it does not have an inbuilt recovery code. Using social engineering methods in an Advanced Persistent Threat (APT) campaign, the attackers might have used stolen credentials and likely had access to the victim’s network for months before the attack [19]. The malware can also extend to extranet networks. The recommendations from the US Cybersecurity and Infrastructure Security Agency that organizations with ties to Ukraine should carefully consider how to isolate and monitor those connections to protect themselves from potential collateral damage are echoed.

Following the detection of WhisperGate, HermeticWiper, another similar malware masquerading as ransomware used against organizations in Ukraine, was discovered on February 23, 2022. The malware targets Windows devices, manipulating the master boot record and resulting in subsequent boot failure [18]. Since both these pieces of malware (WhisperGate and HermeticWiper) are similar, the WhisperGate malware was selected as an example to study. The diamond model of intrusion analysis (DMIA) illustrates (Fig. 1) the four dimensions of the malware attack: the adversary profile, the affected infrastructure, the deployed capabilities, and the target. Specifically, the adversary deploys a capability over a specific infrastructure against a victim [20].

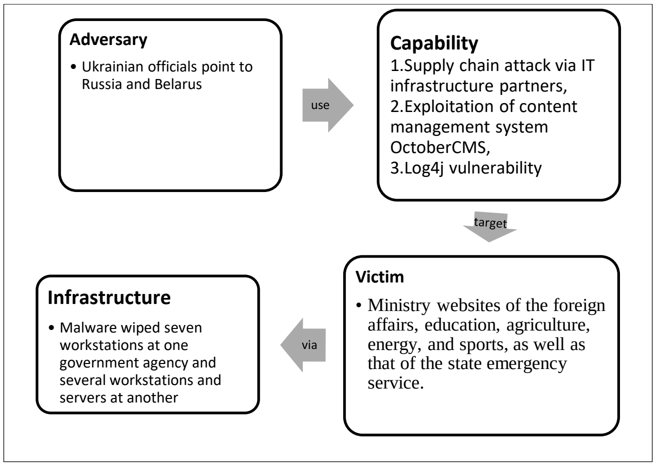


Fig. 1. DMIA model of WhisperGate malware attack.

C. BlackCat

BlackCat is a sophisticated and innovative ransomware family that surfaced in mid-November 2021. It operates as a RaaS business model, and it gained notoriety for soliciting affiliates in known cybercrime forums and offering them to leverage the ransomware and keep 80%–90% of the ransom payment [21]. BlackCat made headlines as one of the first ransomware families written in the Rust programming language, which is used to evade detection by conventional security solutions that may struggle to analyze and parse binaries written in Rust [5].

The rise of cybercrimes has been fueled by the anonymity and non-reversibility of cryptocurrencies, particularly Bitcoin, which makes ransomware payments simple for victims and risk-free for ransomware operators. The trend towards using cryptocurrencies such as Monero, which offers improved security, privacy, and anonymity, is growing, as Monero transactions cannot be traced back to a specific user or address, and the transaction history is kept private. Nonetheless, Bitcoin remains the most popular payment method for ransomware [14]. Among the 31 Ransomware listed by Unit 42, BlackCat has only the seventh largest number of victims listed on their leak site. However, while Lockbit 2.0 ransomware has a list of 50 victims over a period of six months, BlackCat has had an impressive record of 12 victims in just one month since its emergence tanner [21], which makes it a suitable candidate for analysis. In some cases, BlackCat operators use triple extortion by threatening to perform a Distributed Denial of Service (DDoS) attack on the victims' infrastructure: if the ransom is not paid, leak the information along with the data encryption [21]. The DMIA model illustrates the attack process (Fig. 2).

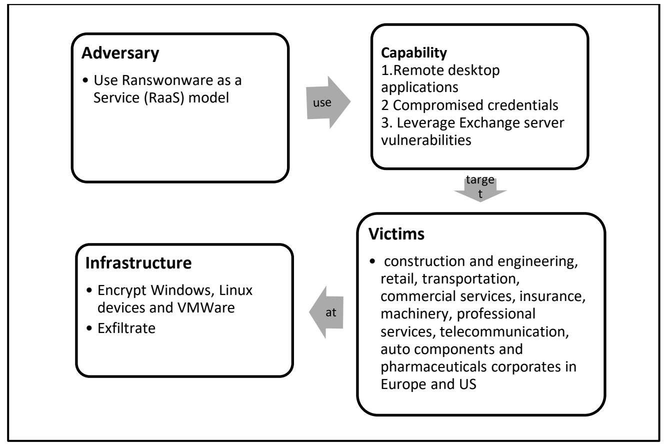


Fig. 2. DMIA model of a BlackCat malware attack.

D. Malware Analysis

Malware analysis applies program analysis and network analysis techniques to understand the behavior and evolution of malicious samples over time [22] and estimate the level of threat and harm a file can cause. Additionally, this kind of analysis helps identify a malicious file's purpose, origin, process execution, file monitoring, and hidden indicators [23].

Malware analysis from a heuristic-based detection (also known as anomaly or behavior based) consists of four types, namely, static, dynamic, hybrid and memory analysis [24]. Being heuristic-based, the proposed research focus on the static, dynamic, hybrid and memory analysis. First, static analysis was used to examine malware samples without the file's execution to extract necessary information from a suspicious file, which assisted us in classifying and identifying its execution. This information is usually gathered using static analysis tools, which examine the sample code more effectively [25]. Static analysis assists in the discovery of the binary code, which contains very useful information about the malicious behavior of a program in the form of op-code sequences, functions, and parameters. However, this method alone may not suffice for a zero-day malware (WhisperGate and BlackCat) before its discovery because new pieces of malware are created daily. The signature-based detection approach followed by the static analysis method requires frequent updates of the virus signature database, which is the method's main disadvantage [26].

Dynamic analysis is deployed, since hackers use various techniques, such as code obfuscation, dynamic code loading, encryption, and packing, to evade static analysis (including signature-based antivirus tools). Furthermore, dynamic analysis can help understand the analyzed file, thus improving detection capabilities [27]. In dynamic malware analysis, the suspicious files are executed and monitored in a controlled environment

[24]. Dynamic analysis includes function call monitoring, network simulation, and registry and file changes. Interactive behavioral properties are observed and analyzed after the simulation of malware. When malware is executed in a dynamic environment, it changes its behaviors. Therefore, static features can be extracted easily and correctly. Hence, the extraction of static features in a dynamic environment detects malware efficiently. The accuracy of dynamic malware analysis alone may instead not be efficient due to the malware's intelligent behaviors [28].

While both static and dynamic analysis techniques are effective on their own, in specific situations, an integrated technique combining the relative merits of each is more efficient. Hybrid analysis that combines both static and dynamic malware analysis is thus generally preferred [29]. Memory analysis, which is used in both malware analysis and malware forensics, involves both the acquisition and analysis phase, thus providing a more comprehensive view of the malware than static and dynamic analyses and an excellent way to analyze memory by preserving a system's contents [29]. Since malware can hide its code in the computer system effectively, it must execute its code in the memory to perform its tasks [24]. Therefore, based on the evaluation of static, dynamic, hybrid, and memory analysis, and having reviewed the efficiency and effectiveness of each of these approaches, the proposed research thus focusses on all four heuristic methods.

E. *Methodologies Deployed in Malware Analysis*

A survey of extant research presented relevant methods used for malware analysis, namely Eureka, disassembled code analyzer for malware (DCAM), malware analysis reverse engineering (MARE), and systematic approach to malware analysis (SAMA). Eureka, a framework allowing a static analysis of malware binaries, highlights the need to produce unpacked code. It provides Windows application programming interface (API) resolution to identify the system calls in the unpacked code [30]. Or-Meir et al. conducted an overview of existing dynamic analysis methods and provided a malware classification based on each category's behavior, mapping layouts, techniques, and flow comprising memory forensics using volatile tools [27]. Almashhadani et al. used the Lock family of crypto-ransomware as their case study for their comprehensive behavioral analysis (BA) of crypto-ransomware [31]. Their work assisted us in the malware analysis of BlackCat, as the latter showed similarities with crypto-ransomware. Ren et al. provided a three-level ransomware detection and prevention mechanism using virtual machines on Petya and NotPetya ransomware [32]. Similar to WhisperGate in terms of its behavior, NotPetya falls under the category of a wiper disguised as ransomware. Hence, its analysis assisted us in the analysis on WhisperGate.

DCAM is a static malware detection technique using code disassembly to recognize malware variants based on a common core signature with promising results on a set of malware [33]. MARE introduced a four-stage approach covering a structured analysis process that focuses on producing an objective outcome to detect malware followed by isolation and extraction phases, as shown by [34], who introduced the malware behavioral technique, malware reverse engineering, and code analysis. The author in [35] proposed an automated analysis framework to analyze executable behaviors through a synergic combination of malware detection techniques, including using a virtual machine over a sandbox to enhance invisibility. SAMA provides detailed information on the working of malware, and its applicability over any type of malware makes it robust. It follows a four-stage approach, namely, an modified version of MARE, as shown by [36]. The authors pointed to the execution order provided by MARE and noted that code analysis must be executed along with behavioral analysis.

SAMA is a complete methodology for performing malware analysis, and malware analysts have used it to analyze the following malware threats: Stuxnet, Dark Comet, Poison Ivy, Locky, Careto, and Sofacy Carberp, including Flame and Red October, as shown by [36]. However, the authors did not explain how they used their tools for each step. Furthermore, through the use of SAMA, the authors have only partially discussed memory analysis. Additionally, the stage of packaging obfuscation is executed after the initial five steps of classification in SAMA. However, it would be more impactful to include obfuscation checking before any other step because the analysis could lead to incorrect results. Finally, a hybrid technique should be included as part of the methodology to perform an in-depth analysis, which is missing in SAMA. Accordingly, hybrid analysis was performed to obtain relevant information and fast results to assess WhisperGate and BlackCat malware. The proposed approach is illustrated against those presented in the literature in Table I. Specifically, the lab setup process and static, dynamic, code, and memory analysis is compared.

TABLE I. COMPARATIVE ANALYSIS OF ANALYSIS METHODOLOGIES

Research Methods	Lab Set-up	Static	Dynamic	Hybrid	Memory
DCAM	No	Yes	No	No	No
MARE	No	No	Partial	No	No
Vidyarthi et al.	Partial	Yes	Yes	No	No
SAMA	Yes	Yes	Yes	No	Partial
Proposed approach	Yes	Yes	Yes	Yes	Yes

III. METHODOLOGY

Since the main objective of the paper is to illustrate a practical approach to carrying out malware analysis, the two sample pieces of malware were analyzed using the following integrated tools and the four analyses. The implemented methodology is shown in Fig. 3 and can be used as a guideline for future comprehensive malware analyses.

A. *Lab Setup*

Flare VM is an open-source collection of software installation scripts for Windows systems to easily setup and maintains a reverse engineering environment on a virtual machine. It was installed on Virtual Box hypervisor to analyze the encrypted malicious file downloaded from Malware Bazaar—a project from abuse.ch). Then, it was downloaded and installed on Windows 10 VM. A system snapshot was taken before each analysis to preserve the integrity of the results and for future-reference analysis. The following sub sections discuss the rationale for choosing the four malware analysis techniques.

B. Applying Static Analysis

Static analysis examines the malicious sample by gathering maximum information without executing it by following a three-step phases approach: de-obfuscation, basic properties analysis (BPA), and advanced static analysis (ASA) (Fig. 4).

The cyber attacker uses obfuscation to intentionally disguise some attributes of the malware specimen by packing it. Hence, before proceeding with the analysis, the initial step should be to perform an obfuscation check and bring the sample to its unpacked state (BPA) In this respect, the file type and signature identification are crucial steps of static analysis to obtain useful information, such as the target operating system (OS) and the architecture of the suspicious file. Among Windows' basic executable files, the presence of portable executables in the form of .exe or .dll provides a glimpse of hexadecimal values and notes that are present in a file.

Malware hashing involves the generation of cryptographic hashes for a malicious file. Hashing algorithms are commonly used to generate hash values of the malicious files are MD5, SHA-256, and SHA-1. This step provides unique values that act as fingerprints for the malware samples. VirusTotal website allows for the flexibility to either upload a file or a URL or simply search the hash value of the sample, and it offers API-based support for detection and recognition by supplying the details of the previous records created by other researchers. String analysis extracts legible letters and words from the malware and focuses on critical information that can be fetched from strings, such as file names, IP addresses, registry keys, and URLs. However, an attacker may include fake strings to divert an analyst from disrupting their task, as strings provide an overview of what malware can do.

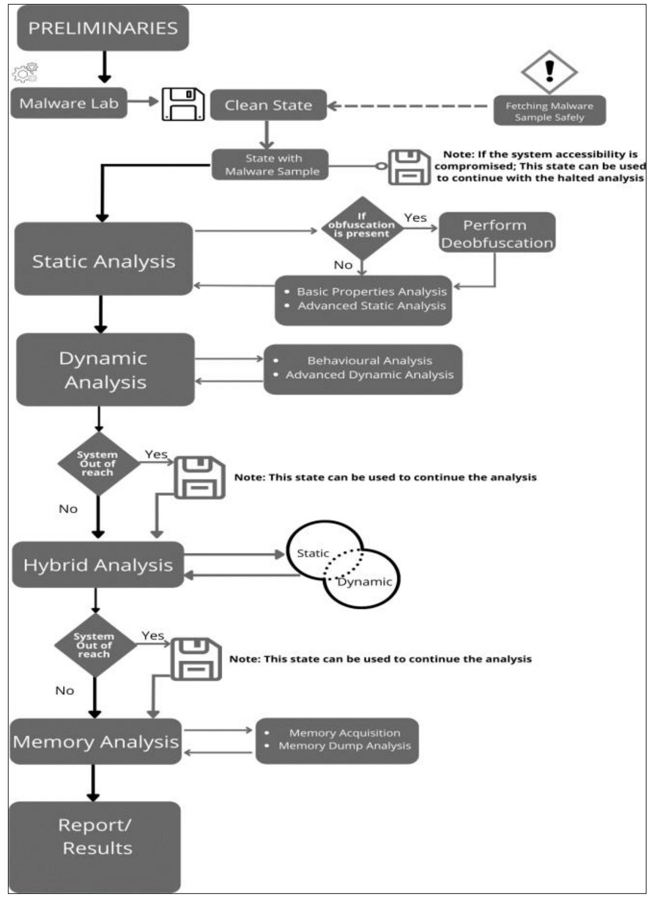


Fig. 3. Malware analysis methodology.

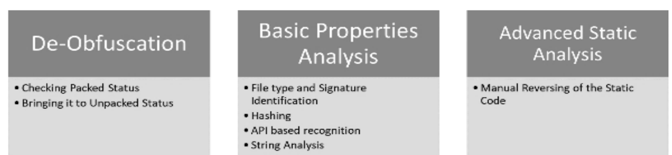


Fig. 4. The three-step phase of the static analysis.

The goal of advanced static analysis of the static code is to understand the malware code's design. This kind of analysis includes the analysis of the machine code by disassembling a file. Performing the static code analysis after BA is appropriate because it requires analyzing the processes and behavior of malware by comparing the two states, namely pre- and post-execution. Conversely, if static code analysis is performed before behavioral analysis, it might reduce the accuracy.

C. Applying Dynamic Analysis

Dynamic analysis implies the execution of the malware sample in a contained and safe environment (sandboxed) to understand the malware's functionality, which includes changes in registry and the files created by it. The objective is to cover two important phases of dynamic analysis, namely behavioral analysis (BA) and advanced dynamic analysis (ADA) (Fig. 5).

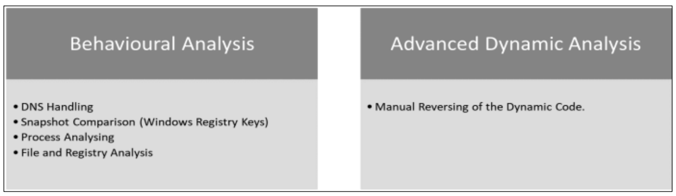


Fig. 5. The two step phase of dynamic malware analysis.

The objective of BA is to understand the suspicious behavior of malware through the interaction with the sample to gather maximum information. BA helps understand the changes in registry, network, and files. The methods in this step include domain name system (DNS) handling, snapshot comparison, process analysis, and registry analysis. DNS handling involves setting up a fake server to generate responses for the requests created by the suspicious file. When the malware is executed, it creates a DNS request so that it can perform the required malicious behavior. These requests are resolved by creating a fake server that fools the malware and generates the response. The snapshot comparison of Window's registry keys focuses on obtaining information about the changes in the number of registry keys and their values before and after the execution of the suspicious file. A registry key is an organizational unit that serves the purpose of an internal database and is used by the computer to store information related to configuration. In the process analysis stage, the malicious application is executed to elicit information regarding its behavior, namely the activities of the application and the details of threads, memory, handles, and the child processes created by it. Analysis of real-time registry, file system, and thread activity of the malicious file involves advanced monitoring of the applications using thread stacks, sessions created, and their activities. It also helps obtain information on the path the processes have traversed in the system, including the changes made. The objective of the ADA stage is to perform advanced analysis on the code by debugging the dynamic code by executing a file.

D. Applying Hybrid Analysis

Since hybrid malware analysis assist to obtain the benefits of both static and dynamic malware analysis [24], this increases their ability to detect malicious software correctly. Furthermore, this analysis technique has all the strengths of static and hybrid analyses while overcoming the shortcoming they have when they are performed independently.

E. Applying Memory Analysis

The main objective of performing this stage is to gain information by monitoring memory changes. An analyst examines the memory dump to gain additional information on process execution and performs a restoration step to make a clean state for further analysis. Memory analysis is integrated with hybrid analysis when an analyst applies basic static analysis to the information gathered during interactive BA, namely the execution of malicious code to generate memory changes followed by dynamic analysis. This phase will be reverted to perform basic static analysis on that memory dump. The table below lists the tools used in malware analysis.

TABLE II. TOOLS AND TECHNIQUES USED IN MALWARE ANALYSIS

No.	Tools	Static	Dynamic	Hybrid	Memory
1	ExinfoPE	Y	X	X	X
2	Hex Editor	Y	X	X	X
3	PeStudio	Y	X	X	X
4	Virustotal	Y	X	X	X
5	Ghidra	Y	X	Y	X
6	ApateDNS	X	Y	X	X
7	Regshot	X	Y	X	X
8	Process Monitor	X	Y	X	X
9	IDA	X	X	Y	X
10	Procmon	X	X	Y	X
11	AccessData FDK	X	X	X	Y
12	Volatility	X	X	X	Y

IV. RESULTS AND ANALYSIS

This section presents a practical approach to analyzing malware using open-source and powerful tools (Table II). The WhisperGate malware is analyzed first, followed by the analysis of BlackCat malware. The preliminary lab setup was meticulously followed to ensure the status of the malware before and after each process (i.e., static, dynamic, hybrid and memory analysis). The following subsections highlight the experimental results obtained through the judicious use of appropriate tools.

A. Experimental Findings - WhisperGate

1) Static analysis: The static analysis followed a three-step approach, namely deobfuscation, BPA, and ASA. Exeinfo PE is a software that can be used to view executable file properties. When using the tool Exeinfo PE to deobfuscate and identify whether the malware was obfuscated (Fig. 6), it was found that the file was unpacked.

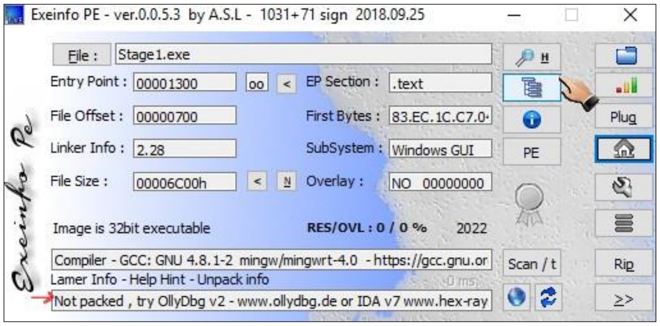


Fig. 6. Deobfuscation of WhisperGate using Exinfo PE.

In the File and Signature Identification step of BPA, the tools Hex Editor (Hxd) were used to obtain detailed information on the signature (Fig. 7 and 8), and PeStudio was used to identify the file type of the malware (Fig. 9). HxD is a tool that can inspect, compare, and verify files, disks, disk images, memory, and log files; patch errors, and repair disk structures. Used for malware detection, PeStudio analyzes the executable files and provides information about the file's properties, characteristics, and potential risks.

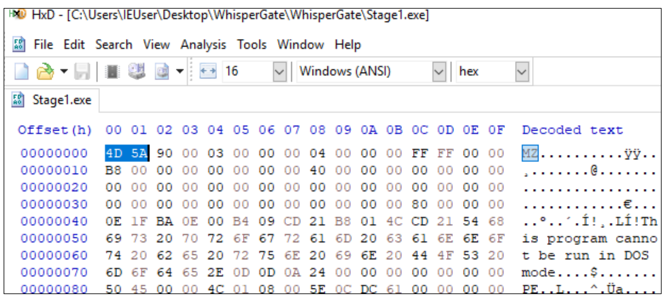


Fig. 7. File and signature identification of WhisperGate using Hxd.

The figure shows that the first two bytes contain 4D 5A, and the decoded text is MZ (which stands for Mark Zbikowski, a leading developer of DOS). Both these values are a crucial factor, which tells us that it is a portable executable. Another file signature that can be observed from the tool is the note that tells that "This program cannot be run in DOS mode." The decoded text states that "Your hard drive has been corrupted… In case you want to recover all hard drives of your organization, You, should pay us $10k Dollars via bitcoin wallet ** and send message via tox ID ** with your organization name. We will contact you to give further instructions" (Fig. 8).

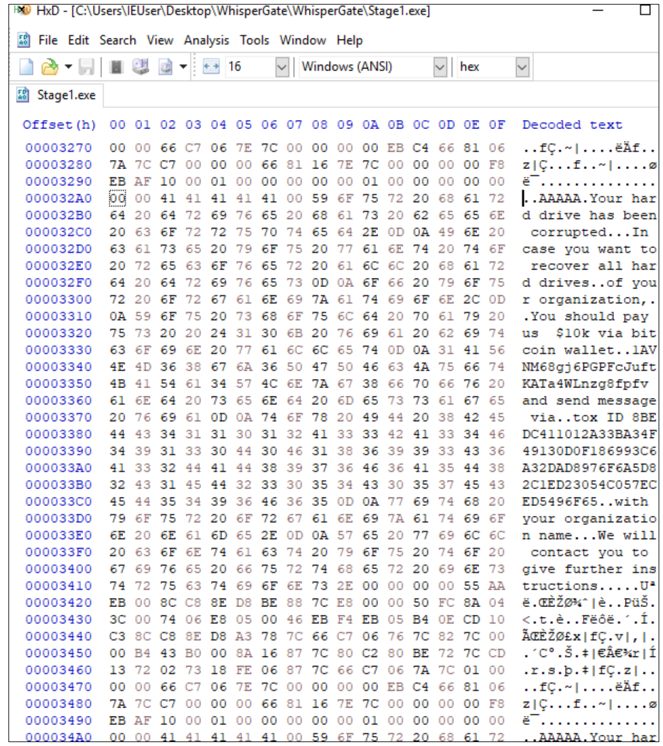


Fig. 8. Decoded text message of WhisperGate using Hxd.

After the signature identification, PeStudio was used to identify the correct type of file (Fig. 9). Based on the results, the file is built on a 32-bit CPU architecture with a file size of 27648 bytes.

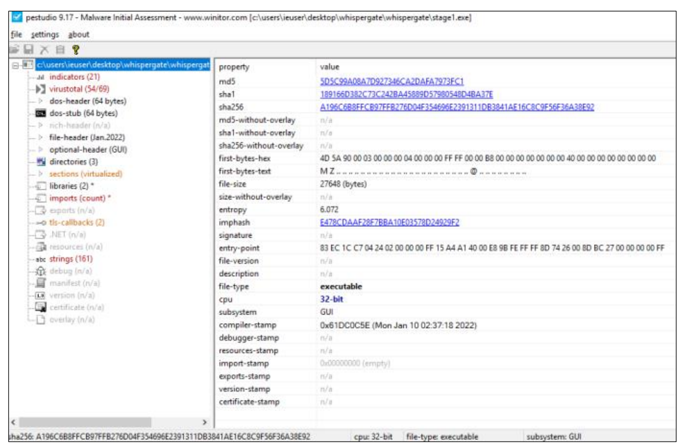


Fig. 9. File type identification of WhisperGate using PeStudio.

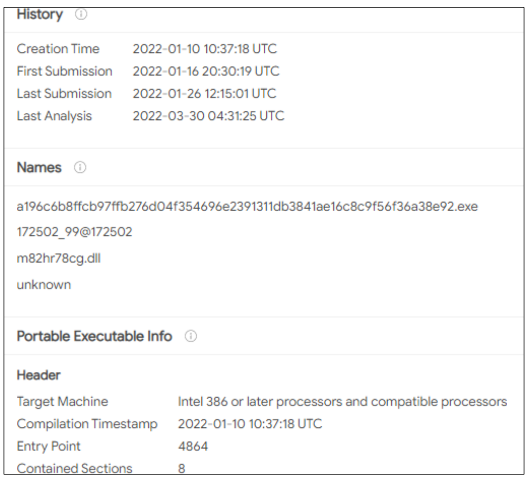
Upon hashing, PeStudio generated the hash values:

MD5 : 5D5C99A08A7D927346CA2DAFA7973FC1

SHA-1:
189166D382C73C242BA45889D57980548D4BA37E

SHA-256:
A196C6B8FFCB97FFB276D04F354696E2391311DB3841AE16C8C9F56F36A38E92

The API-based identification stage involves using the web-based tool VirusTotal. VirusTotal is an online portal where users can upload suspicious files. It uses antivirus engines and website scanners to detect various types of malware and malicious content. The identification and classification of the malware show a community score of 54 on a scale of 69, signifying malware detection by 54 security vendors out of 69. Details such as the history of the application, the compilation stamp, and the information of the target identified through the API are shown below in Fig. 10.


History

Creation Time	2022-01-10 10:37:18 UTC
First Submission	2022-01-16 20:30:19 UTC
Last Submission	2022-01-26 12:15:01 UTC
Last Analysis	2022-03-30 04:31:25 UTC

Names

a196c6b8ffcb97ffb276d04f354696e2391311db3841ae16c8c9f56f36a38e92.exe
172502_99@172502
m82hr78cg.dll
unknown

Portable Executable Info

Header

Target Machine	Intel 386 or later processors and compatible processors
Compilation Timestamp	2022-01-10 10:37:18 UTC
Entry Point	4864
Contained Sections	8

Fig. 10. API identification of WhisperGate from VirusTotal.

In the last step of BPA, namely string analysis, PeStudio was used to analyze the strings and retrieve useful information, as shown in Fig. 11.

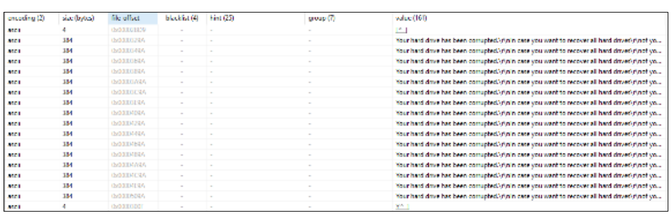


Fig. 11. String identification of WhisperGate using PeStudio.

A total of 161 strings were identified, four of which were blacklisted, and 16 carried the note "Your hard drive is corrupted," indicating that malicious activity could be carried out using the sample. Advanced static analysis involves the single task of manually reversing the static code. The tool Ghidra (a free and open source reverse engineering tool) was used to disassemble the code and further examine the functions, which provided relevant information regarding the nature of the file, as shown in Fig. 12.

```
*(undefined4 *)(&param_1 + iVar1) = 0;
*(undefined4 *)(&stack0x00000000 + iVar1) = 0;
*(undefined4 *)((int)&p_Stack4 + iVar1) = 3;
*(undefined4 *)(&stack0xfffffff8 + iVar1) = 0;
*(undefined4 *)(&stack0xfffffff4 + iVar1) = 3;
*(undefined4 *)(&stack0xfffffff0 + iVar1) = 0x10000000;
*(wchar_t **)(&stack0xffffffec + iVar1) = L"\\\\.\\PhysicalDrive0";
*(undefined4 *)((int)&uStackY24 + iVar1) = 0x403bcf;
pvVar3 = CreateFileW(*(LPCWSTR *)(&stack0xffffffec + iVar1),*(DWORD *)(&stack0xfffffff0 + iVar1),
                     *(DWORD *)(&stack0xfffffff4 + iVar1),
                     *(LPSECURITY_ATTRIBUTES *)(&stack0xfffffff8 + iVar1),
                     *(DWORD *)((int)&p_Stack4 + iVar1),*(DWORD *)(&stack0x00000000 + iVar1),
                     *(HANDLE *)(&param_1 + iVar1));
*(HANDLE *)(&stack0xffffffec + iVar1) = pvVar3;
*(undefined4 *)((int)&p_Stack4 + iVar1) = 0;
*(undefined4 *)(&stack0xfffffff8 + iVar1) = 0;
*(undefined4 *)(&stack0xfffffff4 + iVar1) = 0x200;
*(undefined4 **)(&stack0xfffffff0 + iVar1) = local_2020;
*(undefined4 *)((int)&uStackY24 + iVar1) = 0x403bfe;
WriteFile(*(HANDLE *)(&stack0xffffffec + iVar1),*(LPCVOID *)(&stack0xfffffff0 + iVar1),
          *(DWORD *)(&stack0xfffffff4 + iVar1),*(LPDWORD *)(&stack0xfffffff8 + iVar1),
          *(LPOVERLAPPED *)((int)&p_Stack4 + iVar1));
*(HANDLE *)(&stack0xffffffec + iVar1) = pvVar3;
*(undefined4 *)((int)&uStackY24 + iVar1) = 0x403c09;
BVar4 = CloseHandle(*(HANDLE *)(&stack0xffffffec + iVar1));
*(BOOL *)(&stack0xffffffec + iVar1) = BVar4;
return 0;
```

Fig. 12. Code reverse of WhisperGate using ghidra.

The images taken from Microsoft documentation (Fig. 13 and 14) shows that the functions, CreateFileA (which opens a physical disk drive or a volume) and WriteFile (which writes data to the specified file or input/output (I/O) device) fall under the category of Data Access and Storage. Hence, it is possible to relate the general syntax with the disassembled code provided by Ghidra (Fig. 12), which contains the same two functions as the one from Microsoft documentation.


CreateFileA function (fileapi.h)

Article • 02/16/2022 • 29 minutes to read

Creates or opens a file or I/O device. The most commonly used I/O devices are as follows: file, file stream, directory, physical disk, volume, console buffer, tape drive, communications resource, mailslot, and pipe. The function returns a handle that can be used to access the file or device for various types of I/O depending on the file or device and the flags and attributes specified.

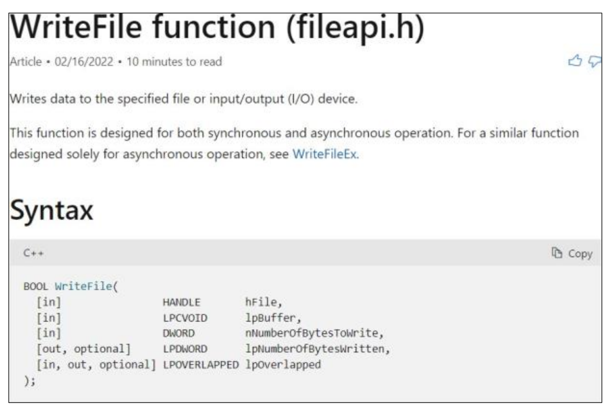
To perform this operation as a transacted operation, which results in a handle that can be used for transacted I/O, use the CreateFileTransacted function.

Syntax

C++ Copy

```
HANDLE CreateFileA(
  [in]           LPCSTR                lpFileName,
  [in]           DWORD                 dwDesiredAccess,
  [in]           DWORD                 dwShareMode,
  [in, optional] LPSECURITY_ATTRIBUTES lpSecurityAttributes,
  [in]           DWORD                 dwCreationDisposition,
  [in]           DWORD                 dwFlagsAndAttributes,
  [in, optional] HANDLE                hTemplateFile
);
```

Fig. 13. The CreateFileA function that relates to the result in Fig. 12.


WriteFile function (fileapi.h)

Article • 02/16/2022 • 10 minutes to read

Writes data to the specified file or input/output (I/O) device.

This function is designed for both synchronous and asynchronous operation. For a similar function designed solely for asynchronous operation, see WriteFileEx.

Syntax

C++ Copy

```
BOOL WriteFile(
  [in]                HANDLE       hFile,
  [in]                LPCVOID      lpBuffer,
  [in]                DWORD        nNumberOfBytesToWrite,
  [out, optional]     LPDWORD      lpNumberOfBytesWritten,
  [in, out, optional] LPOVERLAPPED lpOverlapped
);
```

Fig. 14. The WriteFile function that relates to the result in Fig. 12.

From the disassembled code, useful information was extracted from the two functions. Their synchronization hints that a file is being opened and overwritten to execute a task. CreateFile accepts the parameter "Physical Drive," which is the name of the file being opened. The access mask used is "0xfffffff0." WriteFile provides important details through the handle buffer. The handle returned by CreateFile is used by this function, while the buffer pvVar3 presents the variable Local 2020.

2) Dynamic analysis: The dynamic analysis followed two phases, namely BA and ADA. In BA, the malware was executed to interact with the sample to determine its behavior and intended purpose (see Fig. 5 for the four steps). In the DNS processing step, the ApateDNS tool (that aid analysts in DNS identification) was used to spoof DNS responses to DNA requests generated by the malware, as shown in Fig. 15.

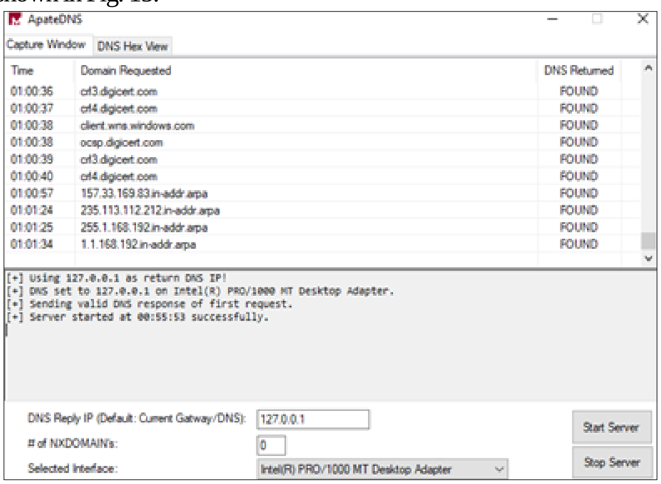


Fig. 15. DNS spoofing of WhisperGate using the tool ApateDNS

During the execution of the sample, the tool successfully captured a list of DNS requests along with the timestamp, indicating that the malware was attempting to connect to different IP addresses for malicious purposes. In the snapshot comparison step, the Regshot tool was used to take sequential snapshots for the pre- and post-execution states to monitor the changes to the registry and files. RegShot that is a tool for controlling changes in the Windows registry can compare the state of registry entries "before" and "after" system changes. The pre- and post-execution snapshots are shown in Fig. 16. The images indicate information regarding the keys, values, and related attributes in the snapshot.

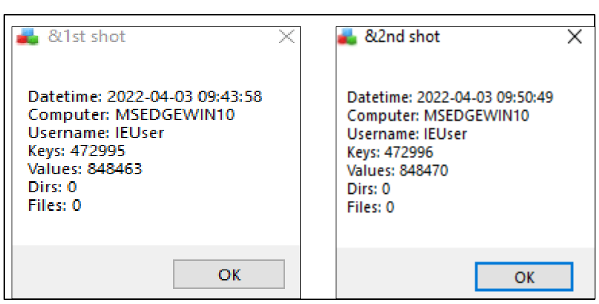


Fig. 16. The pre- and post-execution snapshots using regshot.

The Process Monitor tool was used in the Process Analysis step. Upon invoking the process name filter, the tool generated a list of sub-processes (Fig. 17). The Process Monitor is a troubleshooting and malware hunting monitoring tool for Windows that shows real-time file system, Registry and process/thread activity.

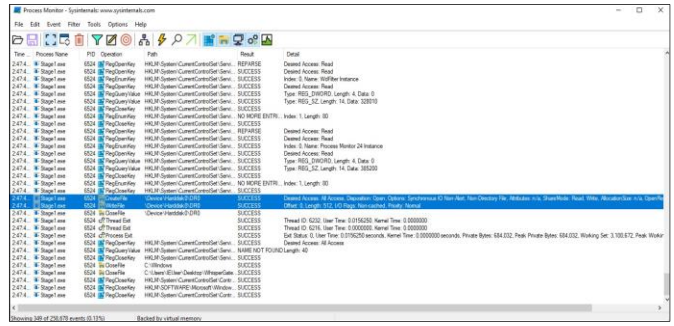


Fig. 17. Process monitoring of WhisperGate using process monitor.

The tool successfully displayed 258,678 events triggered by the malware sample. The main processes highlighted are, again, CreateFile and WriteFile (Fig. 17), and they provide useful information regarding the execution of malware. In particular, CreateFile offers the desired access for the malware to open a file, and WriteFile allows it to overwrite.

In the Registry and File Analysis step, the output generated by Regshot was used to compare the changes made in the registry values and the modifications in the system files, as shown in Fig. 18.

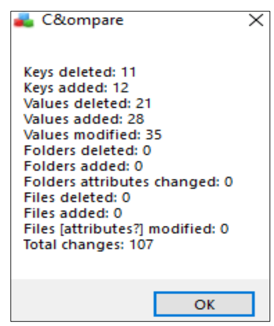


Fig. 18. Registry and file analysis of WhisperGate using regshot.

A total of 107 changes were observed, and the output file displays useful information, such as the addition and deletion of files and registry changes.

In the final ADA step, the IDA tool was used to obtain a debug view of the sample. The IDA tool creates maps of software's execution to display the binary instructions that are actually executed by the processor. The real-time import was used to obtain useful information, as shown in Fig. 19.

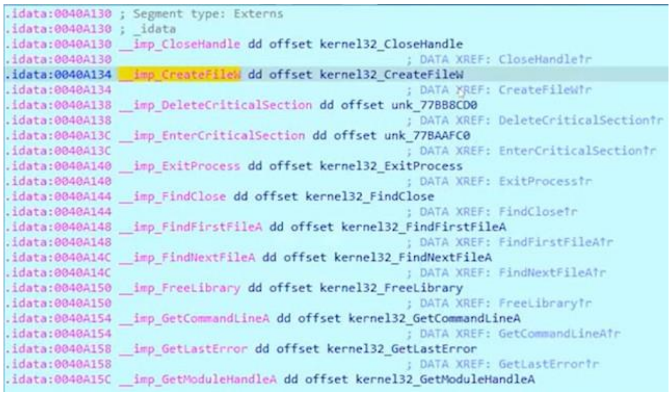


Fig. 19. The debug view of WhisperGate using the IDA tool.

Three vital imported files, CreateFile, CommandLine, and WriteFile, were observed. Out of the 53 imports, four imported an msvcrt library, and the remaining imported a KER-NEL32 library. After creating a breakout for the WriteFile, the file resumed at the GetCommandLine parser. The “stepinto” feature of the tool provides a view of the kernel-based library (Fig. 19).

3) Hybrid analysis: Hybrid malware analysis overcomes the shortcomings of individual malware analysis types, as relying on a single malware analysis method does not provide a comprehensive malware analysis report. In this stage, the information generated using the tools Ghidra and Procmon were integrated. Procmon is a utility for Microsoft Windows OS that captures and displays system and network activity. The static analysis provided an overview of the static code along with the basic properties, while dynamic analysis was applied simultaneously to strengthen the information gathered during the static analysis phase. As soon as the file was run, its behavior could be monitored and compare with the relevant information extracted from the static code analysis. This process highlights the importance of synchronization of both static and dynamic analyses (Fig. 20).

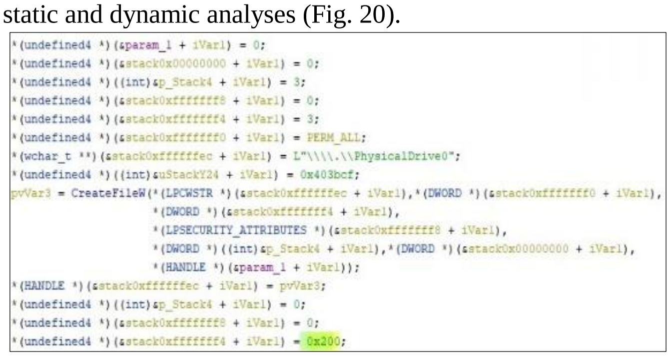


Fig. 20. Result of ghidra illustrating the buffer size.

The buffer size in Hexadecimal code 0x200 (last line in Fig. 20), is revealed as 512 in decimal. The first 512 bytes are equal to the exact size of the MBR buffer. The buffer contains the string “Your hard drive has been corrupted.” It is possible that the sample made an effort to corrupt the MBR, but this hypothesis could not be confirmed without performing a hybrid analysis.

To add value to the information, the thread activity overwriting the device's hard disk with a length of 512 bytes was analyzed. The images shown below (Fig. 21 and 22) provide details such as operation, path, offset value, and the result status of the operation.

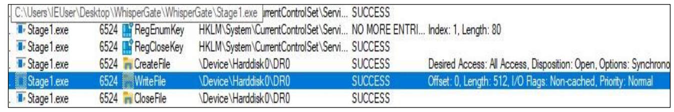


Fig. 21. The thread activity showing the overwriting of the hard disk.

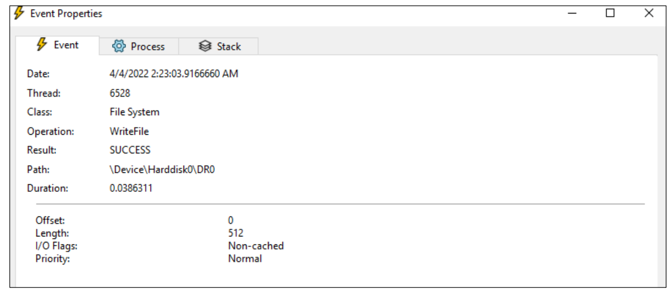


Fig. 22. Event view of the result of the thread activity (see Fig. 21).

The thread confirms the successful execution of the operation WriteFile in overwriting 512 bytes of memory in the hard disk. By synchronizing the use of static code obtained under advanced static analysis and the behavioral characteristic observed under the dynamic analysis, critical information about the malware were successfully gathered. Specifically, the nature of the malware is to write the 512 bytes of the hard disk and corrupt the MBR.

4) Memory analysis: This phase involves a two-step approach: memory acquisition and memory dump analysis. First, a memory dump of the infected state was obtained, and then the analysis was completed by analyzing this memory dump. In the memory acquisition step, the tool AccessData FTK Imager was used to capture the memory dump of the infected state, as shown in Fig. 23. AccessData FTK Imager is a computer forensics software that can create copies, or forensic images of computer data without making changes to the original evidence.

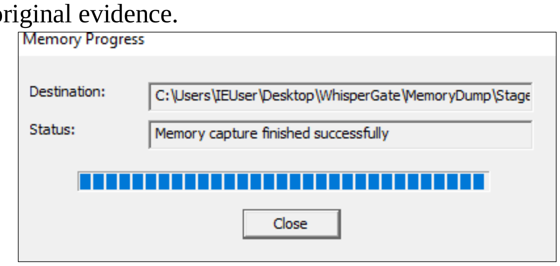


Fig. 23. Acquisition of the memory using the tool AccessData FTK imager.

Subsequently, the tool Volatility was used in the memory dump analysis step to analyze the memory dump of the infected state, which provided valuable information on the running processes (Fig. 24) and the mapping of physical offsets to virtual addresses (Fig. 25). Volatility is a command line memory analysis and forensics tool for extracting and analyzing the volatile data that is temporarily stored in random access memory from memory dumps.

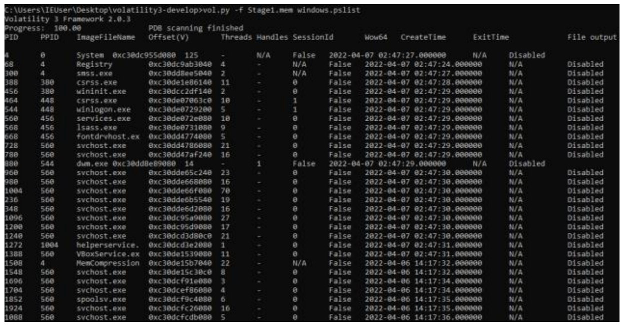


Fig. 24. View of the running processes using the tool volatility.

Fig. 25. View of the mapping of physical offsets to virtual addresses using the tool volatility.

While the images show a list of processes running, the Stage1.exe file is not visible. This indicates that the file has been executed with immediate effect to remove its traces. The virtual mapping provides an overview of the start and close offsets of different regions, such as BootLoaded DriverImages.

B. Experimental Findings - BlackCat

BlackCat malware can be analyzed in a similar manner. However, the analysis is not restricted to the tools which were used to analyze WhisperGate; rather, an analyst can use alternate tools as per the situation, but it should be ensured that the necessary parameters are evaluated according to the target. This section focuses on the findings resulting from the analysis of the BlackCat malware sample.

1) Static analysis: The static analysis followed a three-step approach: deobfuscation, BPA, and ASA. Deobfuscation using the tool ExeinfoPE revealed that the file was not packed (Fig. 26).

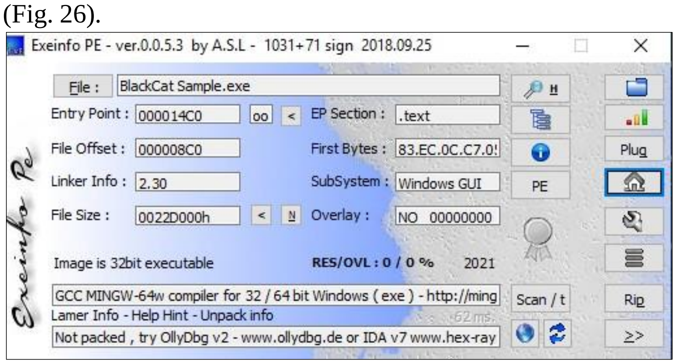


Fig. 26. Deobfuscation of BlackCat using exinfo PE.

The tools HxD and Pestudnio were used in the BPA for file and signature analysis. The Hex view of the sample is provided in Fig. 27.

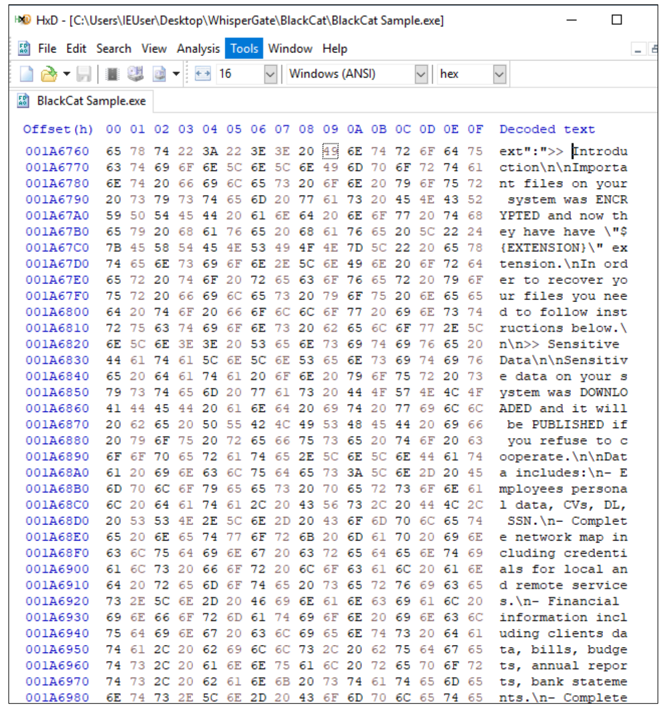


Fig. 27. Decoded text message of BlackCat using Hxd.

The findings revealed the malware to be a PE file with the following decoded text, as shown in Fig. 27: “Important files on your system were ENCRYPTED, and now they have, have. In order to recover your files, you need to follow the instructions below. Sensitive data on your system were downloaded, and they will be published if you refuse to cooperate. Data include: employees’ personal data, CVs, DL, SSN…Caution: do not modify files yourself. Do not use third party software to restore your data. You may damage your files; it will result in permanent data loss. Your data are strongly encrypted; you cannot decrypt it without a cipher key.”

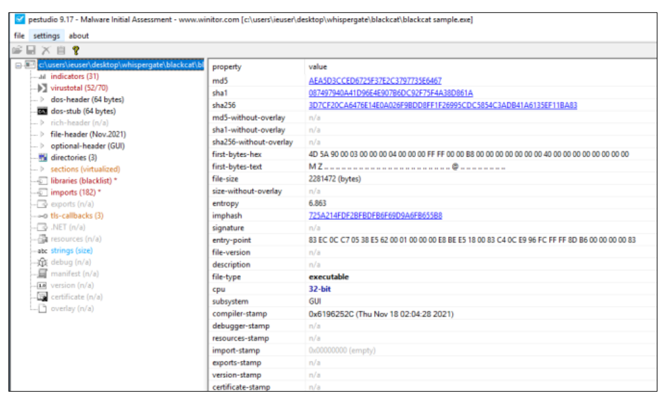


Fig. 28. PeStudio tool identifying the type of file.

After the signature identification, the tool PeStudio identified the type of file (Fig. 28). Based on the results, the file is built on a 32-bit CPU architecture with a file size of

2281472 bytes. In terms of hashing, the following values were obtained:

MD5 : AEA5D3CCED6725F37E2C3797735E6467

SHA-256: 087497940A41D96E4E907B6DC92F75F4 A38D 861°

SHA-1 : 3D7CF20CA6476E14E0A026F9BDD8FF1F26995C DC5854C3A

DB41A6135EF11BA83

The file was identified as a ransomware using Virustotal in the APU-based identification step, with a community score of 52/70.

String analysis through the tool PeStudio (Fig. 29) identified 13454 strings with 73 blacklisted. It was observed that the File-offset 0x0022C514 has a string value WriteFile. This is a critical finding that can assist in the advanced static analysis process.

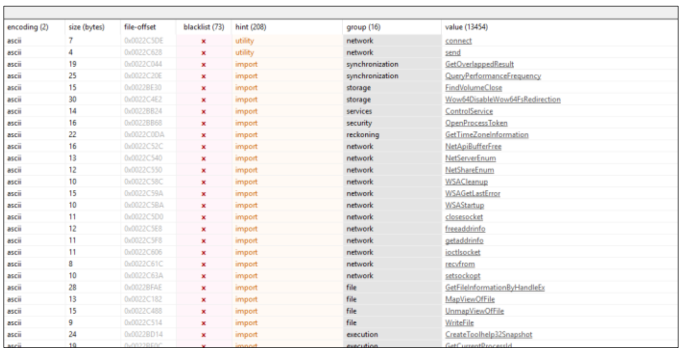


Fig. 29. String analysis of BlackCat using PeStudio.

In the advanced static analysis step, the tool IDA revealed the presence of GetCommanLineW, indicating the intended behavior of the sample when it utilized the command line for a specific task (Fig. 30).

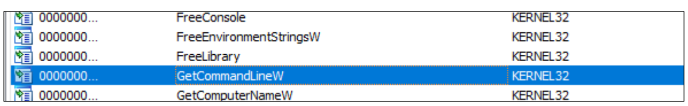


Fig. 30. Advanced static analysis step results using the tool IDA.

2) Dynamic analysis: To analyze the malware behavior, the tool ApateDNS was used to monitor the DNS requests generated by the malware. However, no legitimate responses were identified (Fig. 31).

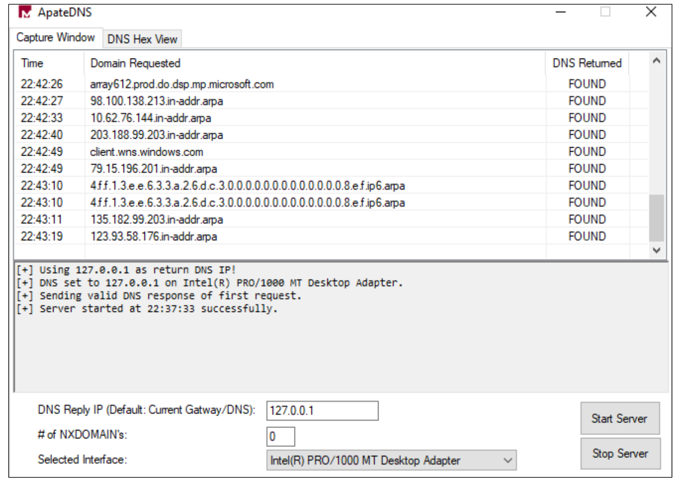


Fig. 31. DNS spoofing of BlackCat using the tool ApateDNS.

The snapshot comparison tool Regshot was used to compare the snapshot of the registry before and after executing the executable (Fig. 32). The snapshots indicate the changes in the keys and values.

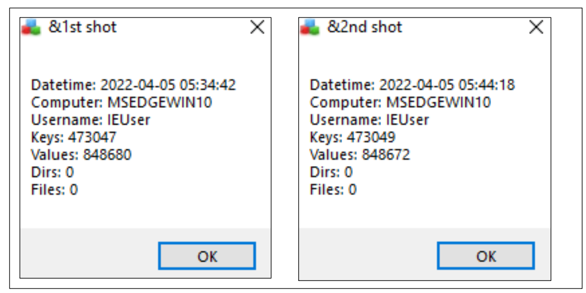


Fig. 32. The pre- and post-execution snapshots using regshot.

The tool Process Monitor was used for the process analysis, which revealed that 389633 processes were triggered on malware execution (Fig. 33). The CreateFile process was highlighted, but no such evidence of WriteFile was produced.

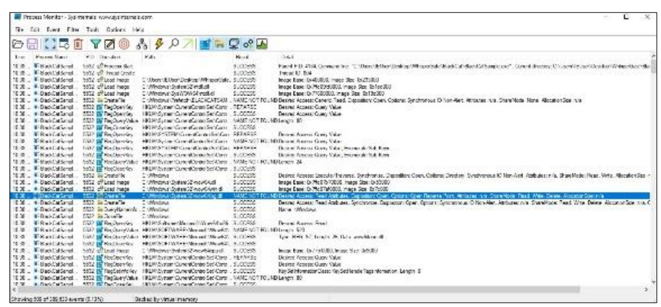


Fig. 33. The result from the tool process monitor with CreateFile highlighted.

In the subsequent step, the tool Regshot was used for registry and file analysis, which highlighted 138 changes (Fig. 34)

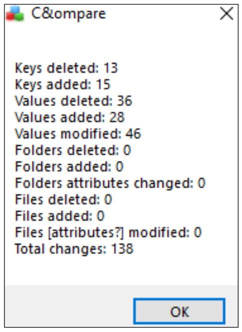


Fig. 34. Registry and file analysis using the tool regshot.

Using the ADA tool IDA, the snapshot revealed that the GetCommandLineW process imported a kernel-based library (Fig. 35).

Fig. 35. Results of the snapshot using the tool IDA.

3) Hybrid analysis: In static analysis, information was gathered about the GetCommandLineW call. Through dynamic code analysis, the complete code was debugged to extract some useful keys: “h,” “p,” “e,” “-,” and “l.” By using the command prompt feature running ProcMon, it was surmised that the keys could be the instructions used in the command prompt, which could be executed to further examine the intended purpose (Fig. 36). Here, the command prompt executes the sample and passes a log file to a particular directory.

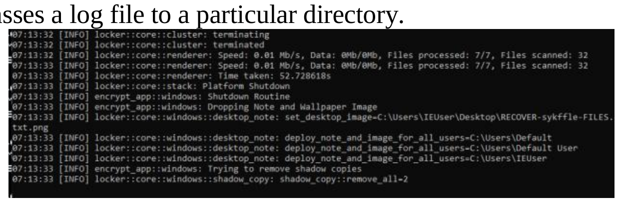


Fig. 36. Results from the command prompt running ProcMon.exe.

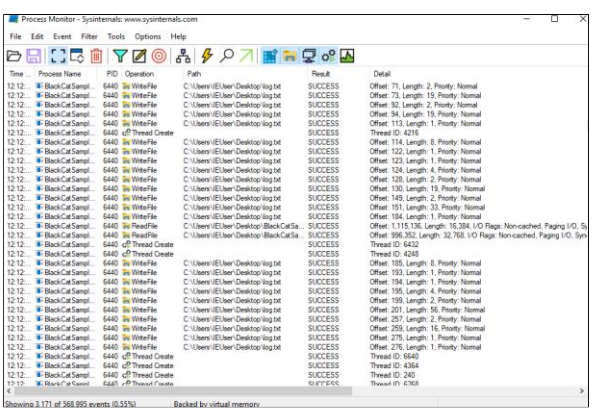


Fig. 37. Results from the tool process monitor.

When using the Process Monitor tool (Fig. 37), it was observed that the malware triggered almost 1.5 times (568,995) the number of processes executed by the same malware when compared to dynamic analysis (389,633). This time, the WriteFile operation was evident and confirmed based on the process executed through the command prompt. The directory set during the execution was successfully injected with the log file, thus corrupting the services.

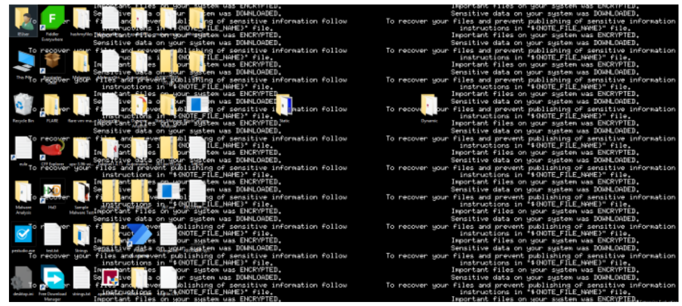


Fig. 38. The wallpaper image dropped through the execution of the malware using cmd.

Once the file was executed through the cmd, it took 53 seconds to corrupt the services, along with dropping a note and wallpaper image (Fig. 38). This demonstrated the speed and potency of the malware in infecting the system.

4) Memory analysis: The tool AccessData FTK was used to capture the memory dump of the infected state (Fig. 39) at the initial memory acquisition step.

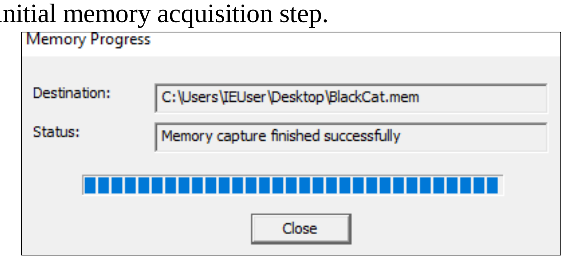


Fig. 39. Capturing the memory dump using the tool AccessData FTK.

Next, in the memory dump analysis step, the Processlist successfully showed the execution of the BlackCat sample in the infected memory dump (Fig. 40).

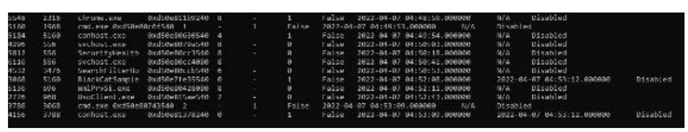


Fig. 40. Image showing the ProcessList upon execution.

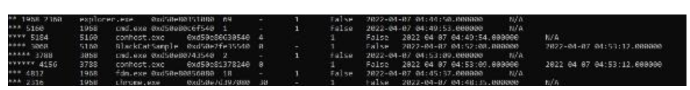


Fig. 41. Image showing the ProcessList details for the malware sample.

The Processtree provides details such as the execution time and the offset value for the malware sample (Fig. 41). The command line operation shows the request to memory accessibility at a particular offset value (Process ID 3788 in Fig. 42).

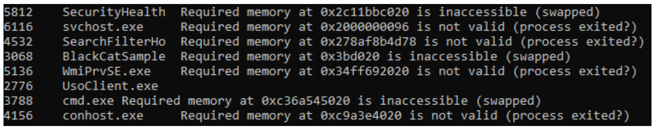


Fig. 42. Image showing the result of the memory accessibility (PID 3788)

V. Discussion

The proposed methodology is applicable for analyzing different malicious files. The case study provides a demonstration of malware analysis on WhisperGate and BlackCat. It is advisable for an analyst to prepare a summary report based on the experimental results of the sample. This section presents a report of the results relative to the two candidate pieces of malware used in the case study.

C. WhisperGate

The analysis of WhisperGate shows a deobsfuscated .exe file with 32-bit CPU architecture carrying threatening information in a static approach. While performing dynamic analysis through the registry modification, the impact of malware was also noticeable. Its nature was identified through hybrid analysis that used both static and dynamic processes in which the malware overwrote the MBR. Changes in the disk led by the malware sample were observed through the offset mapping to the bootloader and driver images (Table III).

TABLE III. Summary Report of WhisperGate Malware

Static	Dynamic	Hybrid	Memory
An unpacked .exe file with a CPU architecture of 32 bits and threatening strings (message) was identified. The API-generated score of 54/69 indicates the presence of characteristics typical of ransomware and wipers. This finding allowed for classifying the malware as a suspicious file with a monetary purpose.	Post-execution impact was visible in the registry modification. Along with a trigger of 2,58,678 events, the WriteFile event brought attention to the modification/overwriting of the file.	Static analysis indicated a buffer 0x200 equal to 512 in decimal, which is indeed the size of MBR. Dynamic analysis generated a thread to overwrite 512 bytes of memory in the hard disk. This finding confirmed the nature of malware corrupting the MBR through overwriting	Virtual mapping of the offsets related to the BootLoaded and DriverImages brought attention to the possible changes in the disk.

Fig. 43 indicates the impact of running the malware sample WhisperGate showing the output "Your hard drive has been corrupted". The sample overwrites the MBR and displays a ransom note demanding $10k via cryptocurrency ("You should pay us $10k via bitcoin wallet"), thus validating the experimental analysis.

Fig. 43. Image showing the impact of executing WhisperGate.

D. BlackCat

Like WhisperGate, the static analysis of BlackCat showed a deobfuscated .exe file with a 32-bit CPU architecture carrying a threatening note. The API score indicates that the sample is ransomware with a monetary purpose. Through registry modification performed in a dynamic approach, the use of the command prompt by the sample was observed. During the hybrid approach, both key identifications used for executing the command prompt were performed by the malware along with a threatening message, indicating the malicious nature of the sample as ransomware. In memory analysis, the same result was revealed through the command line request for memory access (Table IV).

TABLE IV. Summary Report of BlackCat Malware

Static	**Dynamic**	**Hybrid**	**Memory**
An unpacked .exe file with a CPU architecture of 32 bits and threatening Strings (message) was identified. The API-generated score of 52/70 indicated the presence of characteristics typical of ransomware, and a Trojan classified the latter as a suspicious file with a monetary purpose.	Post-execution impact was visible in the registry modification. Along with a trigger of 3,89,633 events, a kernel-based library for the GetCommandLine function indicated the usage of the command prompt by the sample.	Static analysis helped identify keys that were then used for executing via the command prompt in the dynamic analysis. An injected log file encrypting files and a background image showing a threatening message confirmed the malicious nature of the ransomware.	Command line operation requesting memory accessibility indicated the suspicious nature of the sample.

Fig. 44 indicates the impact of running the malware sample BlackCat. The sample corrupted the directory and displayed a background image with a threatening note of the ransom, thus validating the results of the experimental analysis performed using static, dynamic, hybrid, and memory analysis.

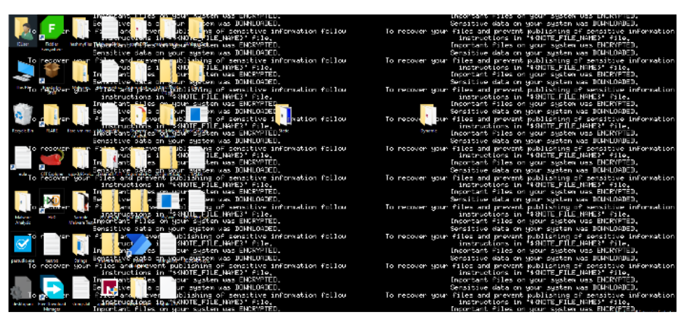


Fig. 44. Image showing the impact of executing BlackCat.

VI. Conclusion and Future Work

The paper offers a comprehensive and practical approach to performing in-depth analyses of pseudo ransomware by illustrating two pieces of malware, namely WhisperGate (used in cyber warfare) and BlackCat (used to target critical organizations of target states). Twelve tools/techniques were selected, and a detailed description of the steps involved in the application, information extraction, analysis of results, and digital forensics is provided. The malware analysis was successfully executed through the use of static, dynamic, hybrid, and memory analysis and then validated. The detailed malware analysis using twelve tools revealed the embedded information and values in the malicious code for greater visibility and subsequent actions for information technology security personnel and forensic analysts. As malware attacks have rapidly risen with the appearance of innovative malware, the research demonstrated a successful methodology for analyzing potent malware through a comprehensive step-by-step approach. The work overcomes the limitations of relying on a single malware analysis technique thus providing a comprehensive approach to malware analysis.

WhisperGate came into the limelight at the beginning of 2022, when it was used to target multiple government and private organizations in Ukraine. The ransomware malware BlackCat was selected as a sample because it was reported to target European affiliations and U.S. organizations in late 2021. Out of the four malware analysis mentioned in the paper, hybrid analysis provided maximum information critical for the malware analyst to understand the extent of damage.

Three limitations have been observed in this study that can lead to further research. First, since only two pieces of malware (i.e., ransomware and pseudo ransomware) were observed, the experimented malware analysis methodology can be extended to diverse malware samples to validate the methodology. Secondly, since the study was limited to traditional malware analysis, appropriate machine learning methodologies can be deployed in future research to compare the findings with those obtained from traditional malware analysis. Thirdly, in this research, open-source tools were deployed for malware analysis that is already known to malicious hackers for circumventing the analysis process. Hence, future research can compare the results of open source tools with subscription based commercial tools.

References

[1] J. Greig. "BlackCat ransomware targeting US European retail, construction and transportation orgs." ZD Net. https://www.zdnet.com/article/blackcat-ransomware-targeting-us-european-retail-construction-and-transportation-orgs/ (accessed February, 2023).

[2] O. Analytica, "Cyberattacks on European energy cement new dynamic," *Emerald Expert Briefings,* no. oxan-db, 2022.

[3] World Economic Forum. "Global Cybersecurity Outlook 2022." World Economic Forum. https://www3.weforum.org/docs/WEF_Global_Cybersecurity_Outlook_2022.pdf (accessed January, 2023).

[4] Centre for Internet Security. "Breaking Down the BlackCat Ransomware Operation." Center for Internet Security. (accessed February, 2023).

[5] Microsoft Security. "The Many Lives of BlackCat Ransomware." Microsoft Inc. https://www.microsoft.com/en-us/security/blog/2022/06/13/the-many-lives-of-blackcat-ransomware/ (accessed February, 2023).

[6] E. Berrueta, D. Morato, E. Magaña, and M. Izal, "A survey on detection techniques for cryptographic ransomware," *IEEE Access,* vol. 7, pp. 144925-144944, 2019.

[7] A. Kapoor, A. Gupta, R. Gupta, S. Tanwar, G. Sharma, and I. E. Davidson, "Ransomware detection, avoidance, and mitigation scheme: a review and future directions," *Sustainability,* vol. 14, no. 1, p. 8, 2021.

[8] M. Sikorski and A. Honig, *Practical malware analysis: the hands-on guide to dissecting malicious software*. William Pollock, 2012.

[9] H. Madani, N. Ouerdi, and A. Azizi, "Ransomware: Analysis of Encrypted Files,"*) International Journal of Advanced Computer Science and Applications,* vol. 14, no. 1, pp. 213-217, 2023.

[10] S. Alelyani and H. Kumar, "Overview of cyberattack on saudi organizations," *Journal of Information Security And Cybercrimes Research,* vol. 1, no. 1, pp. 42-50, 2018.

[11] F. Mercaldo, V. Nardone, A. Santone, and C. A. Visaggio, "Ransomware steals your phone. formal methods rescue it," in *Formal Techniques for Distributed Objects, Components, and Systems: 36th IFIP WG 6.1 International Conference, FORTE 2016, Held as Part of the 11th International Federated Conference on Distributed Computing Techniques, DisCoTec 2016, Heraklion, Crete, Greece, June 6-9, 2016, Proceedings 36*, 2016: Springer, pp. 212-221.

[12] U. Urooj, B. A. S. Al-rimy, A. Zainal, F. A. Ghaleb, and M. A. Rassam, "Ransomware detection using the dynamic analysis and machine learning: A survey and research directions," *Applied Sciences,* vol. 12, no. 1, p. 172, 2022.

[13] P. Zavarsky and D. Lindskog, "Experimental analysis of ransomware on windows and android platforms: Evolution and characterization," *Procedia Computer Science,* vol. 94, pp. 465-472, 2016.

[14] M. Horduna, S.-M. Lăzărescu, and E. Simion, "A note on machine learning applied in ransomware detection," *Cryptology ePrint Archive,* 2023.

[15] Trellix, M. Kersten, and R. Samani. "Return of Pseudo Ransomware." Trellix. (accessed February, 2023).

[16] O. S. Carlos, "Using cyber threat intelligence to support adversary understanding applied to the Russia-Ukraine conflict," *arXiv preprint arXiv:2205.03469,* 2022.

[17] N. Kostyuk and E. Gartzke, "Why Cyber Dogs Have Yet to Bark Loudly in Russia's Invasion of Ukraine (Summer 2022)," *Texas National Security Review,* 2022.

[18] Cybersecurity and Infrastructure Security Agency: uscert. "Update: Destructive Malware Targeting Organizations in Ukraine." Department of Homeland Security. https://www.cisa.gov/uscert/ncas/alerts/aa22-057a (accessed February, 2023).

[19] N. Biasini *et al.* "Ukraine Campaign Delivers Defacement and Wipers, in Continued Escalation." Cisco Talos. https://blog.talosintelligence.com/ukraine-campaign-delivers-defacement/ (accessed February, 2023).

[20] S. Caltagirone, A. Pendergast, and C. Betz, "The diamond model of intrusion analysis," Center For Cyber Intelligence Analysis and Threat Research Hanover Md, 2013.

[21] A. Tanner, A. Hinchliffe, and D. Santos. "Threat assessment: Blackcat ransomware." Palo Alto. https://unit42.paloaltonetworks.com/blackcat-ransomware/ (accessed January, 2023).

[22] D. Ucci, L. Aniello, and R. Baldoni, "Survey of machine learning techniques for malware analysis," *Computers & Security,* vol. 81, pp. 123-147, 2019.

[23] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A Survey on Automated Dynamic Malware Analysis Techniques and Tools ACM Computing Surveys," 2012.

[24] R. Sihwail, K. Omar, and K. Z. Ariffin, "A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis," *Int. J. Adv. Sci. Eng. Inf. Technol,* vol. 8, no. 4-2, pp. 1662-1671, 2018.

[25] E. Gandotra, D. Bansal, and S. Sofat, "Malware analysis and classification: A survey," *Journal of Information Security,* vol. 2014, 2014.

[26] P. Shijo and A. Salim, "Integrated static and dynamic analysis for malware detection," *Procedia Computer Science,* vol. 46, pp. 804-811,

2015.

[27] O. Or-Meir, N. Nissim, Y. Elovici, and L. Rokach, "Dynamic malware analysis in the modern era—A state of the art survey," *ACM Computing Surveys (CSUR),* vol. 52, no. 5, pp. 1-48, 2019.

[28] M. Ijaz, M. H. Durad, and M. Ismail, "Static and dynamic malware analysis using machine learning," in *2019 16th International bhurban conference on applied sciences and technology (IBCAST)*, 2019: IEEE, pp. 687-691.

[29] R. Sihwail, K. Omar, K. A. Zainol Ariffin, and S. Al Afghani, "Malware detection approach based on artifacts in memory image and dynamic analysis," *Applied Sciences,* vol. 9, no. 18, p. 3680, 2019.

[30] M. I. Sharif, V. Yegneswaran, H. Saidi, P. A. Porras, and W. Lee, "Eureka: A Framework for Enabling Static Malware Analysis," in *ESORICS*, 2008, vol. 8: Springer, pp. 481-500.

[31] A. O. Almashhadani, M. Kaiiali, S. Sezer, and P. O'Kane, "A multi-classifier network-based crypto ransomware detection system: A case study of locky ransomware," *IEEE access,* vol. 7, pp. 47053-47067, 2019.

[32] A. Ren, C. Liang, I. Hyug, S. Broh, and N. Jhanjhi, "A three-level ransomware detection and prevention mechanism," *EAI Endorsed Transactions on Energy Web,* vol. 7, no. 26, 2020.

[33] A. Sulaiman, K. Ramamoorthy, S. Mukkamala, and A. H. Sung, "Disassembled code analyzer for malware (DCAM)," in *IRI-2005 IEEE International Conference on Information Reuse and Integration, Conf, 2005.*, 2005: IEEE, pp. 398-403.

[34] C. Q. Nguyen and J. E. Goldman, "Malware analysis reverse engineering (MARE) methodology & malware defense (MD) timeline," in *2010 Information Security Curriculum Development Conference*, 2010, pp. 8-14.

[35] D. Vidyarthi, S. Choudhary, S. Rakshit, and C. S. Kumar, "Malware detection by static checking and dynamic analysis of executables," *International Journal of Information Security and Privacy (IJISP),* vol. 11, no. 3, pp. 29-41, 2017.

[36] J. Bermejo Higuera, C. Abad Aramburu, J.-R. Bermejo Higuera, M. A. Sicilia Urban, and J. A. Sicilia Montalvo, "Systematic approach to malware analysis (SAMA)," *Applied Sciences,* vol. 10, no. 4, p. 1360, 2020