
Malware analysis environment with the use of Elastic Stack

Master Thesis
Songshuo Wang/20220323

Aalborg University Copenhagen



Robotics
Aalborg University
<http://www.aau.dk>

AALBORG UNIVERSITY

STUDENT REPORT

Title:

Malware analysis environment with the use of Elastic Stack

Theme:

Thesis Project

Project Period:

Spring Semester 2024

Project Group:**Participant(s):**

Songshuo Wang
No. 20220323

Supervisor(s):

Marios Anagnostopoulos

Page Numbers: 56

Date of Completion:

May 30, 2024

Abstract:

As malware evasion and obfuscation techniques become more powerful, sandboxing, the current workhorse for dynamic malware analysis, becomes time-consuming when confronted with specific malware. In this thesis, we discuss an alternative approach to dynamic malware analysis that is different from sandboxing, i.e., using Elastic Stack to perform dynamic analysis. This is done by installing and running Elasticsearch, Kibana and integrations, then running real malwares to collect data, and finally using a graphical interface to perform in-depth analysis of the malware's behaviour. This thesis finally summarises some methods as well as examples for determining malware behaviour and more importantly describes the detailed steps for analysis using Elastic Stack.

The content of this report is freely available, but publication (with reference) may only be pursued due to agreement with the author.

Contents

1	Introduction	1
1.1	Problem formulation	2
1.1.1	Contribution	2
1.1.2	Limitations	3
1.2	Structure of the report	3
2	Background	5
2.1	Introduction to malware	5
2.1.1	What is malware	5
2.1.2	Introduction of CIA	5
2.1.3	Malware Classification	6
2.2	Malware analysis	8
2.2.1	Static malware analysis	8
2.2.2	Dynamic malware analysis	8
2.2.3	Dynamic analysis VS Static analysis	9
2.2.4	Using Elastic Stack instead of sandbox	10
2.3	Introduction to Elastic Stack	11
2.3.1	Elastic Stack and malware analysis	12
3	Literature Review	13
3.1	Dynamic malware analysis techniques	13
3.1.1	Dynamic analysis with machine learning	13
3.1.2	Dynamic analysis with sandboxes	14
3.2	Principles of Malware	14
3.3	Elastic Stack in cyber security	15
4	Methodology	16
4.1	Analysis design	16
4.1.1	Network Topology-based analysis	16
4.1.2	Analysis in the infected machine	18
4.2	Malware selection	18

4.2.1	Worms	19
4.2.2	Ransomware	19
4.2.3	Trojan Horses	20
4.2.4	Decision	20
4.3	Implementation	20
4.3.1	Installation	20
4.3.2	Configuration	21
4.3.3	Integration	22
4.3.4	Fleet server,Elastic agent and policy	23
4.3.5	Rules	24
4.3.6	Snapshot	25
4.3.7	The Alert dashboard	25
5	Analysis and Results	27
5.1	Analysis	27
5.1.1	syt.exe	27
5.1.2	FinalPayload.exe	32
5.1.3	b23.exe	36
5.1.4	torn.exe	42
6	Discussion and Conclusion	46
6.1	Discussion	46
6.2	Conclusion	50
6.2.1	Secure system	50
6.2.2	Detect malware	50
6.3	Summary and future work	53
	Bibliography	54

Chapter 1

Introduction

In an era where technology plays an integral role in facilitating organizational operations, the escalating reliance on digital infrastructure has brought forth a critical concern – the pervasive threat of cyber attacks. Malicious activities, particularly in the form of malware, pose substantial risks to the confidentiality, integrity, and availability of sensitive information. As organizations grapple with the evolving landscape of cyber security threats, it becomes imperative to devise robust mechanisms for the detection and mitigation of these digital menaces.

This MSc thesis embarks on the journey to construct a specialized environment dedicated to monitoring malware activities, utilizing the power of the Elastic Stack. This project focuses on the malware behaviour using the Event analysis feature of Elastic Defend. The subsequent phase of the project involves the meticulous collection and analysis of logs generated during these experiments. This wealth of data will be harnessed to develop advanced methodologies for malware detection. By deciphering the patterns and anomalies within the log files, the research aims to enhance our understanding of malware behaviors, ultimately contributing to the development of effective countermeasures.

As the digital landscape continues to evolve, the significance of proactive cyber security measures cannot be overstated. This thesis seeks to address the pressing need for innovative approaches in monitoring and countering malware activities. Through the fusion of cutting-edge technology and empirical experimentation, this research endeavors to make substantial strides in fortifying organizations against the ever-evolving threat landscape of cyber attacks.

1.1 Problem formulation

This project focuses on two main parts: the Elastic Stack and dynamic malware analysis. Since Elastic is not widely used for dynamic malware analysis, it is necessary to understand its preparation and its advantages. In this project, an attempt has been made to answer the following questions, the first two about Elastic Stack and the last two about dynamic malware analysis.

1. What preparation is needed for dynamic malware analysis with Elastic Search?
2. What are the advantages of using Elastic Stack for dynamic malware analysis?
3. How is the security of the system guaranteed when performing dynamic analyses?
4. What behaviours in dynamic analysis identify a target program as malware?

1.1.1 Contribution

This research project seeks to address a critical gap in the realm of cyber security by focusing on the dynamic analysis of malware activities and leveraging the Elastic Stack for comprehensive monitoring.

By delving into the specifics of dynamic malware analysis and the utilization of Elastic Stack and related tools, this research aims to advance the understanding of malware behaviors in real-world scenarios.

The significance of this research lies in its practical application, aligning with the evolving landscape of cyber security threats. Through a systematic study of existing literature on dynamic malware analysis and hands-on experimentation with Elastic Stack, the project endeavors to offer a nuanced perspective on malware detection. The insights gained from the analysis of logs not only contribute to the academic discourse on cyber security but also hold practical implications for organizations seeking effective countermeasures against malware.

This research project contributes by providing a hands-on exploration of dynamic malware analysis, experimentation with Elastic Stack, and a detailed analysis of logs to enhance our understanding of malware activities. The ultimate goal is to empower organizations with practical insights and recommendations to fortify their defenses against the persistent and evolving threat of malware in the digital landscape.

1.1.2 Limitations

Although this project has tried to cover as many situations as possible, some limitations are unavoidable. Among them, the main challenge is that the research samples and data are not diverse, which will lead to a certain gap between the final results and the real world. Addressing this issue in the future may involve working with companies to obtain real-world samples to further analyze data about malware.

In addition, this project does not consider the impact of human factors, such as user awareness. Malware campaigns frequently leverage psychological tactics, and user behavior plays a crucial role in either fortifying or compromising cyber security defenses. With psychology or human Cooperating with behavioral experts and using a multidisciplinary approach to research may be a solution. In addition, if more people conduct research on this project again in the future, simulating the behavior of real corporate employees is also a possible solution to this problem.

It is also unfortunate that due to laptop disk space issues, this project did not implement the building of a network topology to study the spread of malware. If the machine has sufficient space in future studies, it would be necessary to create a topology to perform a dynamic analysis.

1.2 Structure of the report

In the first chapter of this report, the malware threats faced in today's cyber world are introduced and a targeted technical approach is proposed to address such threats, namely Elastic Stack.

In Chapter 2, this chapter explains some of the meanings of terms related to this project, such as malware and Elastic Stack. This information provides comprehensive background knowledge of what is being analyzed, the analyze methodology, and the needed tools.

Chapter 3 focuses on making a summary of the research literature related to this paper. Similar to Chapter 2, the summarised literature is also divided into three categories, research objectives, research tools and research methods. By reviewing the research methods and results of previous literature, it was aim to provide new inspiration for this project as well as improve the research methodology.

In Chapter 4, this chapter describes how the research methodology was conceptualised and implemented as appropriate to the specifics of this project. It includes comparisons and trade-offs between different research approaches and research objectives, considering their strengths and weaknesses in a comprehensive manner.

Chapter 5 presents the results of this project and the process of analysing these results.

In Chapter 6, it is divided into two parts. The discussion section records the problems encountered in the course of this project, which are intended to assist subsequent research, and the conclusion section summarises the findings of this project.

Chapter 2

Background

2.1 Introduction to malware

2.1.1 What is malware

There are many definitions of malware, for example: From the definition of Cisco: "Malware, short for malicious software, refers to any intrusive software developed by cyber-criminals (often called hackers) to steal data and damage or destroy computers and computer systems" [1] and also, the experts in ENISA mentioned: "The word Malware is derived from the term 'Malicious Software'. Any piece of software that performs undesirable operations such as data theft or some other type of computer compromise can be categorised as Malware"[2].

As can be seen from the above two definitions, how to explain the "malicious behavior" of malware is the key to defining malware. In this project, we define malicious behavior as "behavior that intentionally harms the target CIA."

Therefore, in this project, we follow as the definition of malware: a piece of program code which aim to harm the target CIA.

2.1.2 Introduction of CIA

As Sun Tzu said : "If you know the enemy and know yourself you need not fear the results of a hundred battles." [3] So in order to analysis the malware, we must know what the malware's purpose: breaking the CIA.

C: Confidentiality relates to the privacy of data, which means that unauthorised access needs to be prevented. Confidentiality is achieved when, for example, an email requires a token to read it, and hacker who do not have that token cannot access the data in the email even they hijack the email.

I: Maintaining the integrity of data means that it must be protected from malicious modification. In the previous email example, if the hacker can delete or tamper with the content, the data would lose integrity.

A: Maintaining availability means that the system is capable to do the intended function. Similarly, in the previous email example, the receiver can successfully read the full message means that usability is guaranteed.

2.1.3 Malware Classification

In the survey by O’Or-Meir et al.[4], it was noted that malware is divided into three main classes, each of which has many subclasses. This classification received wide acceptance, so this project partially follows that survey’s classification when presenting the malware classification. These are Type, behaviour and privilege. In this project, a specific type of malware is planned to be selected for study. This is due to the fact that the functionality and purpose of the malware varies greatly between different malware, so there is a need to select a malware that meets the experimental environment and experimental conditions for the study. The specific selection process is described in the methodology chapter 4.

Type

- Virus

Viruses are usually malicious code present in certain files that are capable of spreading between hosts. The term is also commonly used to refer to many types of malware in general. One example is ILOVEYOU, discovered in 2000, which spreads by e-mail, overwriting the victim’s files.

- Worm

Worms can also spread between hosts, and because they are not restricted by the host programs, they spread back faster and are more threatening [5]. For example, the Code Red worm[6], which takes advantage of a buffer overflow vulnerability to spread quickly and perform DDOS attacks.

- Trojan horse

Trojans are malware disguised as ordinary programs that do not propagate or replicate, and their main purpose is to act as a backdoor for attackers to steal private information, etc. An example of this is the Zeus Trojan [7], which injects malicious code into web browsers to record keystrokes, credentials, and other sensitive information.

- Ransomware

Ransomware usually encrypts the victim’s files as a way to demand a ransom. A typical example of ransomware is wannacry, which appeared in 2017. It spreads on Windows machines via a vulnerability called EternalBlue, encrypts the victim’s data and demands Bitcoin to unlock it.

- **Cryptojacking malware**

The main function of cryptojacking malware is to use the victim's device for mining without the victim's knowledge. This can consume a lot of power from the victim's device and is relatively difficult to detect because there is no obvious malicious behaviour. For example, Coinhive[8] is a well-known Cryptojacking malware in recent years, and it is used to mine bits of the Monero cryptocurrency.

Behavior

- **Denying service**

The purpose of this behaviour is to disrupt the availability of the victim's system, such as a DDOS attack, by generating a large number of requests through a botnet to overload the target so that it cannot continue to serve legitimate users [9]. In a DOS attack, the main intent of the malware is to compromise Availability in the CIA. This means preventing the victim from working normally [10].

- **Information stealing**

This behaviour usually includes theft of credentials, personal privacy, confidential data and so on. It also breaks the confidentiality in the victim system's CIA by stealing sensitive data.

- **Spreading**

Self-propagation is common with viruses and worms, which take advantage of system vulnerabilities and network connections to rapidly infect multiple victim machines. The Morris Worm [11], discovered in 1988, did nothing but replicate and infect itself, but that was enough to cause massive network slowdowns.

Privilege

- **User mode(Ring 3)** User mode has the lowest privileges, programs running in it have access to only a limited number of resources. It is not difficult to remove malware running in user mode.
- **Kernel Mode(Ring 0)** The core of the operating system is called the kernel mode, malware that runs in kernel mode has access to all services including the hardware system, this malware is also known as rootkit.
- **Hypervisor(Ring -1)** The Hypervisor has higher privileges and this technique manages the operation of virtual machines. This means that if the malware has such privileges, it can escape detection by analytical tools.
- **Hardware (Ring -3)** Malware that infects hardware will be trickier to detect because it will have more means of evading detection and can attack from outside the CPU.

2.2 Malware analysis

2.2.1 Static malware analysis

Static analysis means analyzing the malware without actually running it.

Static analysis mainly uses PEbear, PEview, IDA and other tools to disassemble malware code and observe strings, libraries or functions for malicious behaviour.

For example, look at the file hash and compare it to malware database (such like virus-total), or use IDA to list the strings of the file and look for comments left by careless authors. And watch for suspicious function calls, such as the keylogger [12].

2.2.2 Dynamic malware analysis

Dynamic analysis refers to the process of analysing a piece of malware by executing it and observing its behaviour; the goal of the analysis is to expose the activities of the malware as it runs in a secure environment [4].

The typical dynamic malware analysis tool are sandboxes. The ability of sandboxes is to provide a secure environment in which to dynamically run malware. The sandboxing system can highlight malicious activities, such as modification entry in registry, deleting, uploading files in a system [13].

One of the famous example sandbox is Cuckoo sandbox [14]. The figure below shows how does the Cuckoo sandbox work. In short, the host machine is responsible for initiating analysis and monitoring malicious behaviour and generating analysis reports, while the client machine acts as a restricted environment to run the malware and then transmits the results of the analysis back to the Cuckoo [15].

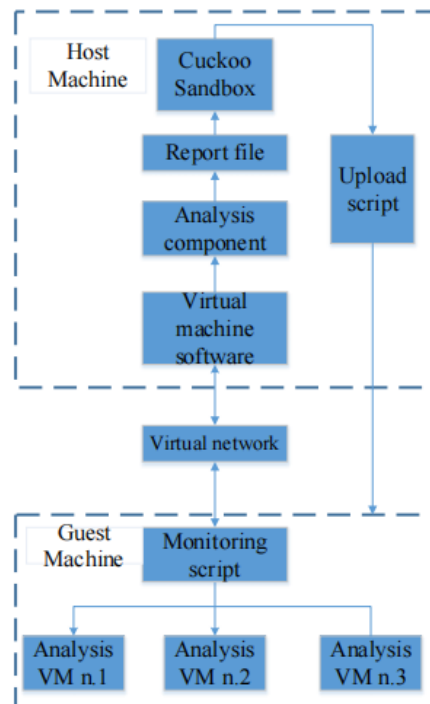


Figure 2.1: How Cuckoo works
[15]

But using sandboxes for dynamic malware analysis is not perfect. Many sandboxes, such as Cuckoo, have a number of shortcomings that cannot be ignored. Aaron Walker et al. show that Cuckoo's malware threat scoring is arbitrary: "The arbitrary nature of the Cuckoo scoring methodology casts confusion upon the threat level of a given malware sample." [16] Moreover, if the authors of the malware have fully researched how the sandbox works, then they may intentionally evade detection. When the malware detects it is running in a simulated environment, it can show non-suspicious behaviour or just stop running [17].

2.2.3 Dynamic analysis VS Static analysis

One of the most attractive features of static analysis is that it eliminates the need to actually run the malware, which also keeps the system safe. However, the threat of malware is also greatly reduced if it is run in a controlled virtual environment and can be restored to its initial state. In this way, the advantages of static analysis over dynamic analysis are not obvious. In addition, static analysis has many shortcomings.

- More complicated

Compared to dynamic analysis, static analysis requires the researcher to have more

knowledge such as assembly language, hardware fundamentals, and so on. These are not beginner friendly. Dynamic analysis, on the other hand, can use graphical tools, etc., to clearly reflect the behaviour of the malware without having to directly study the underlying code logic.

- Evasion Techniques

Modern malware makes extensive use of techniques such as obfuscating code and self-packaging to circumvent detection. This makes it difficult to perform static analyses. As more and more diverse evasion techniques emerge, static study of the code will also become more time consuming. In contrast, dynamic analysis is able to skip obfuscation behaviours and capture malicious behaviours without the need to study the underlying logic of these obfuscated behaviours. This greatly saves analysis time and increases efficiency.

By considering the above points, this project finally decided to use dynamic analysis to study malware.

2.2.4 Using Elastic Stack instead of sandbox

As mentioned earlier, there are many problems with using sandboxes to study malware. And for these problems, Elastic Stack deals with them well [18].

- Complicated sandbox formation process

The powerful analysis capabilities of many sandboxes are undeniable. But with it comes a complicated configuration process. This can include installing and configuring many interdependent pieces of software, such as Regshot, Wireshark, ProcMon, etc. This also means that there is more uncertainty, and in the event of a problem with one application, it may not be possible to carry out the analysis successfully. And this is something Elastic Stack handles very well. For researching malware, there are only two files to download and a few integrations, making it very beginner friendly.

- Risk of being detected by malware

As previously stated, there is a risk that the sandboxes currently in general use on the web are recognised by malware. Malware authors will purposefully detect these sandbox features, stop execution, and even modify logs. Comparatively speaking, Elastic is a latecomer and is constantly being updated, up to version 8.13 so far. This means that each update to Elastic refines known issues, making malware detection more difficult.

2.3 Introduction to Elastic Stack

The latest Elastic Stack is very simple, requiring only Elastic, Kibana and integrations.

- Elasticsearch

Elasticsearch is a distributed RESTful search and analytics engine for indexing and analysing all kinds of data like registry, system commands[19]. Some of the key features include: open source, easy to extend and integrate, powerful RESTful API, with real-time analytics capabilities[20].

- Kibana

The main function of Kibana is to provide data visualisation. Examples include pie charts, table charts, geographic maps, etc. and uses a browser interface that is flexible and easy to use.

- Integrations

In the latest version, integrations replaces Logstash, filebeat, etc., making the Elastic Stack more streamlined. integrations includes a number of useful tools and applications, including Elastic Defend, which is the focus of this project, and Integrations. Integrations has a very easy to use UI on Kibana, and can often be installed and run with a single click 'add integration'.

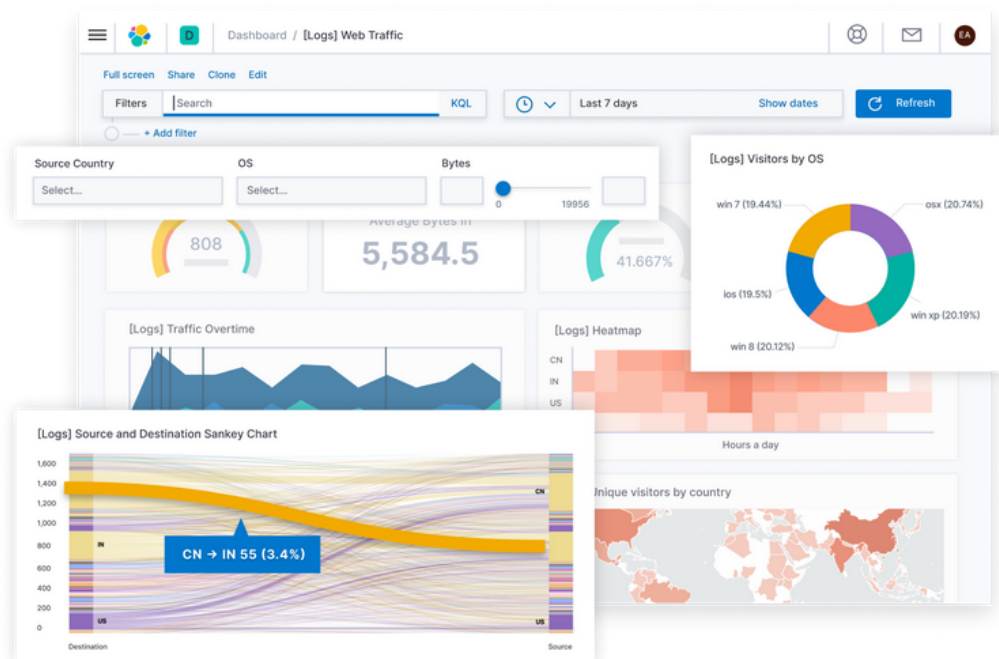


Figure 2.2: Powerful visualisation of Kibana

- Elastic Defend

Elastic Defend belongs to integrations and needs to be installed in order to be used in Elastic. Elastic Defend has a wide range of features such as blocking malware, threat detection, identifying cloud threats and more[21]. In this project, the main use was its ability to detect malware, by installing Rules, Elastic Defend was able to detect and alert on malicious behaviours, as well as perform Event Analyze on each malicious behaviour.

- Fleet and Agent

Elastic Agent is used to monitor the host's data, logs, and so on, and re-query data from the operating system[22], a host can only be configured with one Elastic Agent.

Fleet's role is to provide a web-based UI for managing connected Elastic Agents and their policies, i.e., multiple Elastic Agents can be associated with a single Fleet Server to monitor the data of multiple hosts at the same time.

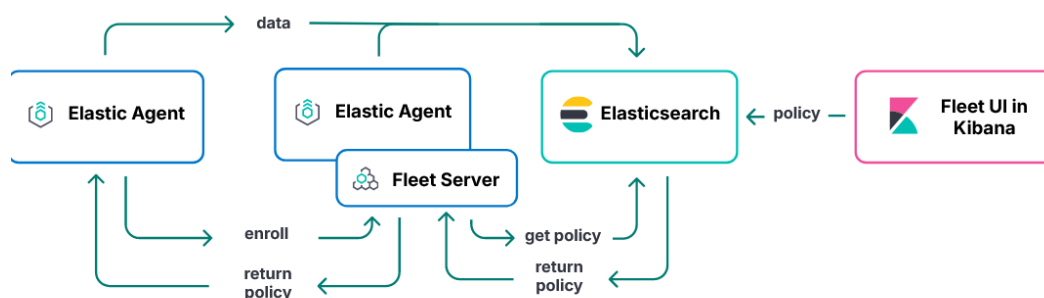


Figure 2.3: Fleet server and Elastic agent

2.3.1 Elastic Stack and malware analysis

Elastic Stack's data analysis capabilities are very powerful, which makes it an ideal tool for analysing malware. Typically, log datasets for malware analysis are relatively large, capable of reaching hundreds of megabytes in large cases, or even in gigabytes. and one of the advantages of using ELK technology is the ability to handle large volumes of data. At the same time, Elasticsearch uses unsupervised machine learning algorithms, which can help researchers efficiently detect anomalies in the system and thus identify malware infections. In addition, visualisation is also an important advantage, because the number of system logs is huge, and there is a lot of useless information, and malware analysis needs to look for possible suspicious points in a large amount of useless data, which makes the visualisation can minimise the pressure on researchers and improve the efficiency of analysis.

Chapter 3

Literature Review

This project is about using Elastic Stack to perform dynamic analysis of malware. So in the literature research, the focus is on what methods the researchers have used to perform dynamic analysis of malware, how Elastic Stack can be specifically applied to the analysis steps and learning the principles of malware propagation based on papers. The aim is to synthesise and compare the research methods of these literatures and get inspiration that can be used to refine the analysis methods of this project.

3.1 Dynamic malware analysis techniques

3.1.1 Dynamic analysis with machine learning

In the field of malware analysis, traditional analysis methods are considered time consuming and complex, so people have started to combine emerging machine learning and AI techniques with malware analysis. Nassiri et al.[23] applied the Random Forest algorithm from machine learning to improve the accuracy of log analysis to 92 percent. Similarly, Si-hwail et al.[24] applied machine learning in their study on dynamic analysis of malware. They used a number of algorithms including vector machine (SVM), the decision tree, Naïve Bayes (NB) to learn and classify malicious behaviours. Although machine learning is not the main focus of this project, the dynamic research methods proposed by these authors are very valuable, e.g. combining behavioural and memory analysis, API function calls.

In addition, researchers have discovered the power of neural networks in the field of malware analysis. In a study by YT Huang et al.[25] the authors developed a neural network that became TagSeq to identify malicious behaviour. This paper analyses in detail the life cycle of Eggnog malware, malicious behaviours, etc., pointing out three very valuable observations, namely: access to system resources (e.g. registry entries), limited information about malware family names and studying the complexity of individual program API calls. These problems are solved by the TagSeq technique proposed by the authors. This

paper is very inspiring for this project as it not only provides more ideas for dynamic analysis of malware, but also provides a solution to the potential difficulties.

3.1.2 Dynamic analysis with sandboxes

Sandbox have long been an important tool for conducting dynamic malware research, such as Cuckoo. in the 2018 paper by Sainadh Jamalpur et al.[13] a detailed process of sandbox analysis is shown, including the study of Process tree, Dropped files, register files. even though this project ultimately does not consider the using sandboxes for dynamic analysis, this paper still provides a great deal of valuable guidance on dynamic analysis. Similarly, Chih-Hung Lin et al.[26] used sandboxes, their approach used VTCSandbox and used a more novel technique, namely virtual time controller for their study. This complex technique includes hypervisor security monitor, the key acceleration component, VTC and sandbox. While the details of this technique are outside the scope of this project's implementation, the techniques demonstrated in the paper to prevent malware from evading detection and to optimise the speed of detection are valuable to this project.

3.2 Principles of Malware

Only by understanding how malware works can researchers know how to perform dynamic analysis and where to find data representing malicious behaviour. In Long Chen et al.[27]'s study, after complex computation of Graph Clustering, Behaviour Analysis, and machine learning of detection algorithms, the authors summarised malware propagation models including Social Network Propagation, Network-Level Propagation and Telecom Network, etc. Although this research focuses on mobile malware propagation models, its value lies in providing the principles of malware propagation, and it is not difficult to apply them to the study of non-mobile malware.

Similarly, in the study by P Eder-Neuhauser et al.[28] the propagation characteristics of three types of malware and countermeasures are explained. They are: pandemic malware, endemic malware and contagion malware. for the first malware, the authors propose that it is characterised by aggressive scanning, automatic initiation of TCP connections and a relatively simple payload. For the second malware, the characteristics also include automatic initiation of TCP connections, list scanning, and a large payload. And in Contagion malware, the characteristics are passive scanning, TCP connections are also legitimate and have larger payload. After complex calculations and derivations, the authors propose that the malware propagation can be detected by restricting VLANs, deploying firewalls, using asymmetric encryption and IDS. This article uses a very complex and scientific approach with a lot of calculations, which is beyond the scope of this project, but nevertheless, the topology construction, malware propagation characteristics and detection methods proposed by the authors are very informative.

Worms are a typical type of malware, and they are also ideal samples for studying

the principles of malware propagation. In the study of worm propagation by Jun Li et al.[29] the basic principles of how to detect worms are explained in detail and a novel worm detector called SWORD is proposed. In the paper it is suggested that worm detection is divided into host and network and specific detection methods are given such as buffer overflow detection, memory errors etc. are used for host detection and observing signatures and detecting traffic are used for network detection. The article also suggests that the traditional detection scheme cannot cope well with the worm's superior means of evading detection, and gives SWORD as a solution: combining two detectors, BDD (Burst Duration Detector) and QPD (Quiescent Period Detector), to detect worms. This means that if a worm wants to avoid detection, it has to ensure both a long activity period and a enough quiescent periods, which is very difficult. In the end, SWORD's detection results were impressive, which is very informative for the worm research in this project.

3.3 Elastic Stack in cyber security

A convenient and powerful tool is very important in conducting malware research. Elastic Stack was chosen for this project, and the experience of some researchers in using Elastic Stack and how to use it were also referred to. For example, in the paper by Alin Puncioiu et al.[30] the authors use Elastic Stack stacks to detect attacks. Specifically, the authors used Winlogbeat to send logs to Logstash for processing, and also integrated virustotal inside Logstash to scan for malware. For Elasticsearch, the authors integrated it with MISP to search for malware activity information. After setting all this up, the authors enabled network monitoring and used Packetbeat to analyse network traffic. In addition, the authors configured machine learning inside Elasticsearch to detect anomalies. In the end, the results section of the article clearly demonstrates how powerful Elastic Stack is, as it successfully detected a large number of malware domains, and data leakage attacks. The topic of this paper fits well with this project, so the tests mentioned in the article as well as the methods of using them are very helpful for the deployment and implementation of this project.

Although this project does not use CTI (Cyber Threat Intelligence) technology, the study by Hamad Almohannadi et al.[31] still provides inspiration and reference for analysis using Elastic Stack. The authors first searched out the attack events and logs using elastic-search and then visualised them using Kibana. Similarly, in N Shah et al.[32] although there is no guidance on how to use Elastic Stack to analyse malware, this paper is still highly informative because malware analysis requires sifting through a large number of text logs, so optimising the efficiency of the work is particularly important. To improve the efficiency of Elastic Stack, the authors put forward three suggestions, namely: an optimised shard size, a standardized structure for data and a specific configuration file.

Chapter 4

Methodology

4.1 Analysis design

In designing the research methodology, two approaches were considered: building a network topology for the research, which means separating the monitoring machine from the infected machine. Or using only one machine and running malware analysis while it is infected.

4.1.1 Network Topology-based analysis

In the early stages of this project, a topology-based research method was the preferred option. The reasons are based on the following:

- Observing how malware spreads

Spreading between networks and replicating itself has always been a signature behaviour of malware. Building a topological network can be easy to study the mechanism of malware propagation.

- Securing the analysing machine

On the analyser, since its task is not to collect data, it is possible to run powerful virus protection programs. This means that even if malware spreads over the network to the analyser machine, firewalls and virus detection programs will isolate the threat in the first instance and protect the normal operation of the analyser machine.

- More data

Malware may have different behaviours on different systems and versions, so building experimental machines for different systems and versions helps to collect more data for a more comprehensive analysis.

Due to the above attractive advantages, the following topology network was attempted to be built using GNS3 at the beginning of the implementation of this project:

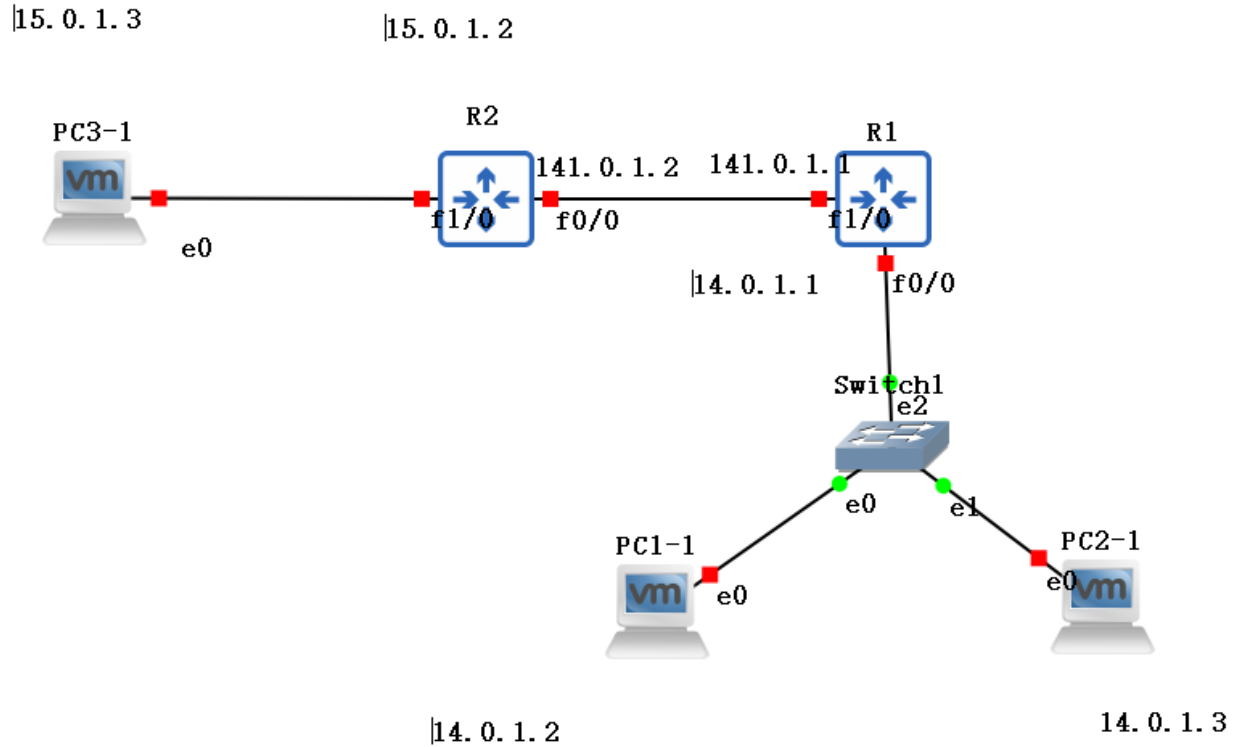


Figure 4.1: Ideal case topology

The topology plan uses PC3 as the analysis machine, and PC1, PC2 as the infected data collection machines.

This method relies on the previously mentioned Elastic agent and Fleet server. The core implementation idea is to configure Fleet server in the research machine, and then enroll the elastic agent of the infected machine into the Fleet server of the research machine to obtain the data of the infected machine. This means that all infected machines and research machines need to run Elasticsearch.

However, after a period of trial and error, several major flaws in this research methodology were exposed.

Since the implementation of this project is restricted to a personal laptop, memory footprint becomes a huge issue for the implementation of this topology. If too little running memory is allocated to the VMs, for example 2GB, then the individual VMs will be extremely slow when running Elastic Stack. As an example, loading the Elastic pre-built rules (a rule base with 1088 inspection rules) took the VMs up to an hour or so to load. To make matters worse, there is a high probability that the VM will simply crash and have to

be restarted while loading.

However, allocating more memory to each VM would cause the host to become overwhelmed. To be clear, to start the GNS3 topology, the official GNS3 VM will also be started automatically. this further compresses the available space. Eventually the topology network research method had to be cancelled due to the difficult to balance memory issues.

4.1.2 Analysis in the infected machine

The modified solution is to use both Fleet server and Elastic agent on one machine, and when infected by malware, Elastic Defend is an integration that collects alerts and analysis of malware behaviours. This solution proved to be a huge space saving solution and ensured that the virtual machine was up and running.

However there are drawbacks to this method. This approach does not provide the 'detection of malware propagation' and 'more data' previously mentioned in the topological network approach.

It is also important to mention that since the machine needs to run both malware and detection, it is important to make sure that the malware used is not so powerful that it can modify Elasticsearch logs, as well as other malicious behaviours that affect the operation of the software. One example of this is that when trying to run the Tsuchigumo.bat malware[33], Elasticsearch is unable to collect alerts and data, the possible reason for this is that this software shuts down windows explorer.

While this method does have the problems mentioned above, the impact of these drawbacks can be reduced to a minimum. Firstly, there is the problem of not being able to detect malware propagation, even though malware propagation and self-replication is a feature of viruses, but not all of them. Excluding this, some very obvious malicious behaviours can be detected, such as modifying the registry and invoking ping commands. So in this approach, although it is not possible to detect if the malware is spreading in the network, it is still possible to dynamically analysis other malicious behaviours. The collection of 'more data' is essentially to ensure the accuracy of the analysis results, for which multiple runs of malware on the same machine can achieve similar results. Lastly, regarding the issue of some malware being too powerful, since this project is not focused on countering powerful malware, low or medium threat malware is a more ideal sample to analysis.

4.2 Malware selection

In this section different malware to be selected will be compared and finally one will be selected for analysis. Viruses will be omitted because they can be considered as an alias or generic term for a type of malware. And for cryptojacking malware, which is also not on the research candidate list for this project because of its specialised use.

4.2.1 Worms

Worms are often considered a subclass of viruses. Unlike viruses, worms do not usually require a file, or user action, to propagate[34]. The advantages and disadvantages regarding the use of worms are listed below:

Advantages

- Relatively simple Since the main purpose of a worm is to spread over the Internet, most worms do not seriously affect the normal operation of the operating system itself.
- Clear purpose The main purpose of the worm is clear: to propagate. So the goal is relatively clear and the behaviour is easier to understand during the research process.

Disadvantages

- Evading detection Worms have been iterated over a long period of time, so it is also very likely that the detection means to make detection to avoid detection.
- Threat to the host machine Some powerful worms are even able to spread to host machines via virtual machines, which is a very dangerous behaviour.

4.2.2 Ransomware

Advantages

- Very obvious infectious behaviour Ransomware is designed to block the victim's files, so it is very easy to detect the malicious behaviour of this software. This may facilitate data collection.

Disadvantages

- Dangerous behaviour Ransomware's malicious behaviour is often designed to be dangerous or even irreversible in order to be able to effectively blackmail its victims. This is not an advantage for the research of this project.
- Functional complexity In order not to be easily disarmed by the victims, the authors of ransomware usually design it to be extremely complex with many functions. This makes it difficult to be detected and analysed.

4.2.3 Trojan Horses

Advantages

- Relatively safe Trojans are generally designed as a backdoor, so there is no obvious dangerous behaviour.
- Persistent Once enabled, Trojans are likely to stay on continuously, which means the research window is also longer and more conducive to data collection.

Disadvantages

- Difficult to detect While it has the advantage of being more secure, this is also a disadvantage. The lack of obvious malicious activity means that even if the data collection window is long, it is not easy to obtain information.
- Complication Modern Trojans can be so complex and multifunctional that it may be difficult to find their primary intent in a dynamic study.

4.2.4 Decision

After reflecting on and comparing the advantages and disadvantages of using these malware for research, the project ended up using a worm for dynamic analysis. This was because the risk was too high for ransomware, which could directly affect system operation and block files, so it was discarded. It was a little difficult to choose between worms and Trojan horses, but worms were chosen because the project initially planned to run the analysis in a network, so more data would be collected using worms. The decision to use worms was kept even though the plan later proved to be unachievable.

4.3 Implementation

4.3.1 Installation

In the latest version (8.13) of Elastic Stack, there is no need to download as many installation packages as in previous versions, It only needs to download and install Elasticsearch and Kibana from the official website. There are two deployment options in the latest version, local installation of Elastic Stack or using Elastic Cloud, the latter has the advantages of faster deployment, less space, and comes with AI assistants, but it requires a monthly rent, while the former can be obtained for free. Since this project is not complex, it is natural to choose locally deployed Elastic Stack.

4.3.2 Configuration

The first thing to do is to note down the initial user elastic and its password. After launching `elasticsearch.bat` for the first time, the programme will automatically generate the elastic super account and corresponding password, which will be printed to the screen.

```
Authentication and authorization are enabled.
TLS for the transport and HTTP layers is enabled and configured.

The generated password for the elastic built-in superuser is : lIdBRwGUa-JpZXzKkz_+

If this node should join an existing cluster, you can reconfigure this with
'/usr/share/elasticsearch/bin/elasticsearch-reconfigure-node --enrollment-token <token-here>'
after creating an enrollment token on your existing cluster.

You can complete the following actions at any time:

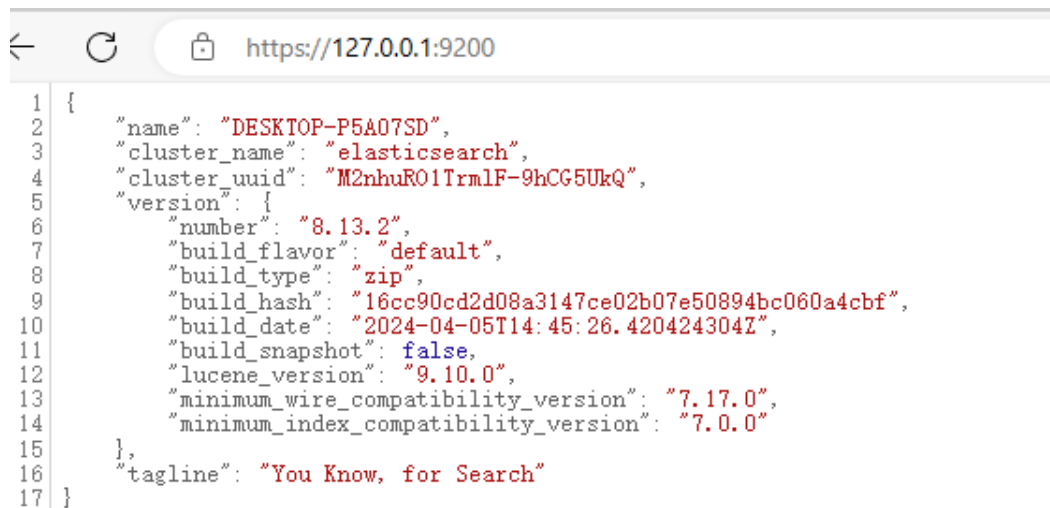
Reset the password of the elastic built-in superuser with
'/usr/share/elasticsearch/bin/elasticsearch-reset-password -u elastic'.

Generate an enrollment token for Kibana instances with
'/usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s kibana'.

Generate an enrollment token for Elasticsearch nodes with
'/usr/share/elasticsearch/bin/elasticsearch-create-enrollment-token -s node'.
```

Figure 4.2: Initial user

After that, it is necessary to configure the `elasticsearch.yml` file, such as configuring 'network.host' as the IP address of the machine and 'path.logs' as the appropriate path. Once the configuration is complete, the default server address can be accessed by typing `https://localhost:9200` into browser with elastic user and password.



```
1 {
2   "name": "DESKTOP-P5A07SD",
3   "cluster_name": "elasticsearch",
4   "cluster_uuid": "M2nhuR01TrmlF-9hCG5UkQ",
5   "version": {
6     "number": "8.13.2",
7     "build_flavor": "default",
8     "build_type": "zip",
9     "build_hash": "16cc90cd2d08a3147ce02b07e50894bc060a4cbf",
10    "build_date": "2024-04-05T14:45:26.420424304Z",
11    "build_snapshot": false,
12    "lucene_version": "9.10.0",
13    "minimum_wire_compatibility_version": "7.17.0",
14    "minimum_index_compatibility_version": "7.0.0"
15  },
16   "tagline": "You Know, for Search"
17 }
```

Figure 4.3: Elasticsearch web page

Once successfully logged in to the Elasticsearch web page, it is important to configure Kibana for a more intuitive and user-friendly graphical interface. Kibana requires an enrollment token to pair with Elasticsearch, which is printed in the output of the initial run of `elasticsearch.bat`, and also by running `elasticsearch-create-enrollment-token -s kibana`. Kibana runs on port 5601 by default. Enter `localhost:5601` in a browser to access the Kibana interface.

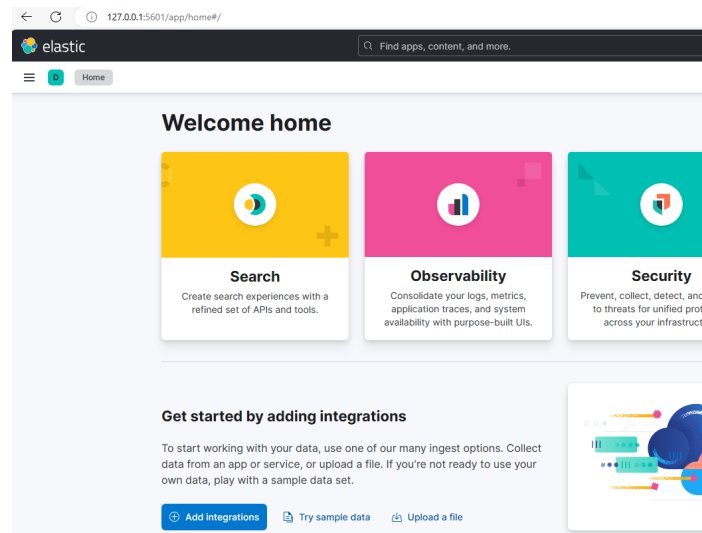


Figure 4.4: Kibana web page

4.3.3 Integration

After configuring Elasticsearch and Kibana, the integrations need to be added. For this project, a total of five integrations have been added, namely Fleet Server, Elastic Agent, Elastic Defend, System, where Fleet Server and Elastic Agent are required to enable Elastic Defend, which is used to dynamically analyze malware. System is mainly used to help collect log files generated by the system.

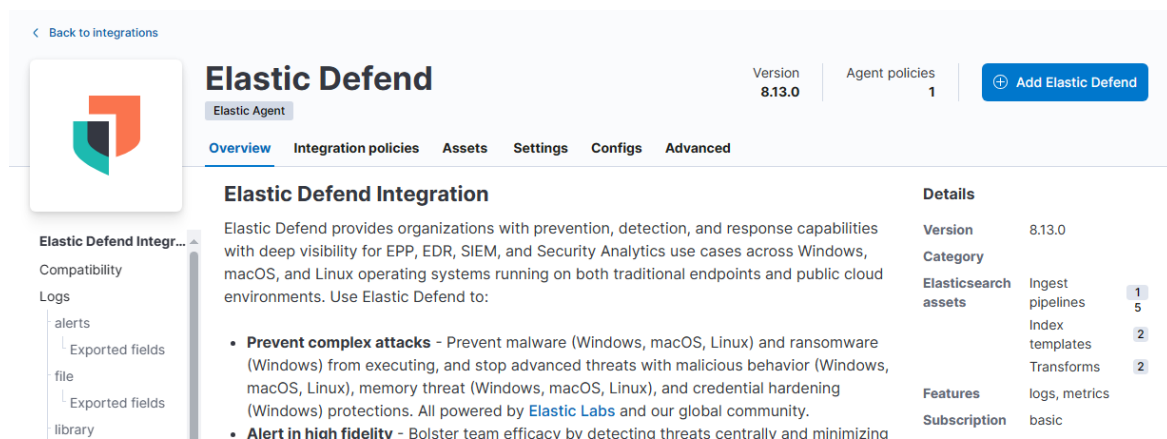


Figure 4.5: Elastic Defend

4.3.4 Fleet server, Elastic agent and policy

In this step, the Fleet server is set up first, then the agent, and finally the policy is added to the agent. None of these steps are complicated and the official quick guide helps one to complete the configuration quickly:

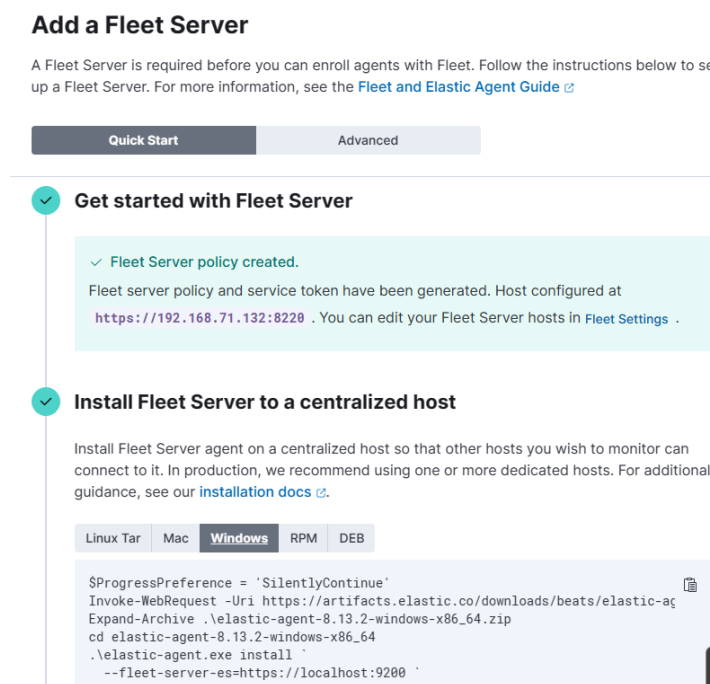


Figure 4.6: Fleet Server setup

After configuring the Fleet server and agent, the next step is to create a policy and add

the required integrations to the policy.

Name ↑	Integration	Namespace	Actions
PC1Defend	Elastic Defend v8.13.0	default ⓘ	...
fleet_server-1	Fleet Server v1.5.0	default ⓘ	...
system-1	System v1.56.0	default ⓘ	...

Figure 4.7: Policy

After everything is configured, a functioning Fleet server should show up as healthy.

Status	Host	Agent policy	CPU ⓘ	Memory ⓘ	Last activity	Version	Actions
Healthy	desktop-p5a07sd	PC1Policy rev. 5	4.99 %	258 MB	20 seconds ago	8.13.2	...

Figure 4.8: Healthy server

The final step is to set Elastic Defend's detection rule to detect instead of prevent, otherwise it will prevent the malware from running.

4.3.5 Rules

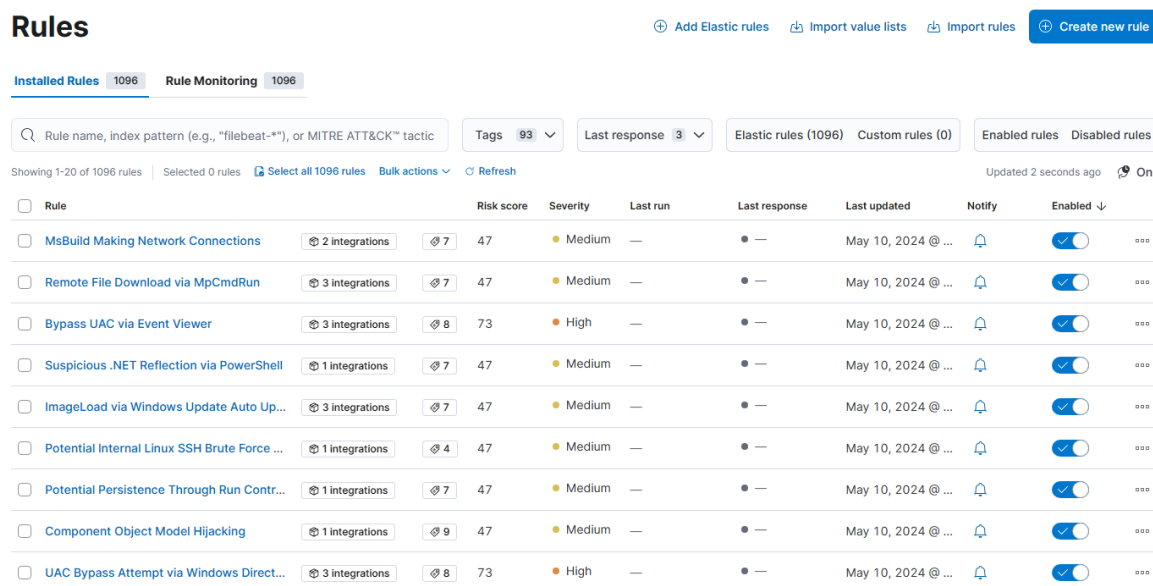
Elastic Defend requires the addition of detection rules to properly detect malicious behaviour. To add detection rules, the API integration key must be configured in Kibana by running `kibana-encryption-keys generate` and copying the three lines of text generated to the bottom of `kibana.yml`. In this project, the three lines of text are:

```
xpack.encryptedSavedObjects.encryptionKey: 8ecc89df163bcfb5b85c85ee399ed88c
xpack.reporting.encryptionKey: 6415fccafc8bb8e0c2bee7f63101dd76
xpack.security.encryptionKey: 91e5966510f7c5df409823cd6b15dfd2
```

After configuring the API integration key, the next step is to install elastic with the built rules. Once installed, use the bulk action to select Enable All.

4.3.6 Snapshot

After this is all set up, Elastic is ready to detect the malware. However, before the malware can actually run, the Internet connection must be severed and the host machine needs to ensure that virus protection software such as Windows Defender, McAfee, etc. is turned on to prevent the malware from exploiting a vulnerability to escape from the virtual machine and compromise the host machine's system security. More importantly, snapshots must be taken of the initial system state before running the malware, with the aim of restoring it to an uninfected state for the next malware after researching one, and taking snapshots also ensures that the virtual machine can be rolled back to a backup state and continue to run after an irreversible event.



Rules ⊕ Add Elastic rules 📄 Import value lists 📄 Import rules ➕ Create new rule

Installed Rules 1096 **Rule Monitoring** 1096

🔍 Rule name, index pattern (e.g., "filebeat-*"), or MITRE ATT&CK™ tactic Tags 93 Last response 3 Elastic rules (1096) Custom rules (0) Enabled rules Disabled rules

Showing 1-20 of 1096 rules | Selected 0 rules 📌 Select all 1096 rules ⚡ Bulk actions 🔄 Refresh Updated 2 seconds ago 🔔 On

<input type="checkbox"/> Rule		Risk score	Severity	Last run	Last response	Last updated	Notify	Enabled ↓	
<input type="checkbox"/> MsBuild Making Network Connections	🔗 2 integrations	🔗 7	47	Medium	—	May 10, 2024 @ ...	🔔	🔵	...
<input type="checkbox"/> Remote File Download via MpCmdRun	🔗 3 integrations	🔗 7	47	Medium	—	May 10, 2024 @ ...	🔔	🔵	...
<input type="checkbox"/> Bypass UAC via Event Viewer	🔗 3 integrations	🔗 8	73	High	—	May 10, 2024 @ ...	🔔	🔵	...
<input type="checkbox"/> Suspicious .NET Reflection via PowerShell	🔗 1 integrations	🔗 7	47	Medium	—	May 10, 2024 @ ...	🔔	🔵	...
<input type="checkbox"/> ImageLoad via Windows Update Auto Up...	🔗 3 integrations	🔗 7	47	Medium	—	May 10, 2024 @ ...	🔔	🔵	...
<input type="checkbox"/> Potential Internal Linux SSH Brute Force ...	🔗 1 integrations	🔗 4	47	Medium	—	May 10, 2024 @ ...	🔔	🔵	...
<input type="checkbox"/> Potential Persistence Through Run Contr...	🔗 1 integrations	🔗 7	47	Medium	—	May 10, 2024 @ ...	🔔	🔵	...
<input type="checkbox"/> Component Object Model Hijacking	🔗 1 integrations	🔗 9	47	Medium	—	May 10, 2024 @ ...	🔔	🔵	...
<input type="checkbox"/> UAC Bypass Attempt via Windows Direct...	🔗 3 integrations	🔗 8	73	High	—	May 10, 2024 @ ...	🔔	🔵	...

Figure 4.9: Rules

4.3.7 The Alert dashboard

With all rules enabled, Elastic has been able to detect malware running. However, during the course of this project, an issue arose: the Alert interface did not show new alerts immediately after running malware. So after each malware run, Elasticsearch and Kibana had to be restarted to refresh the page and get new alerts.

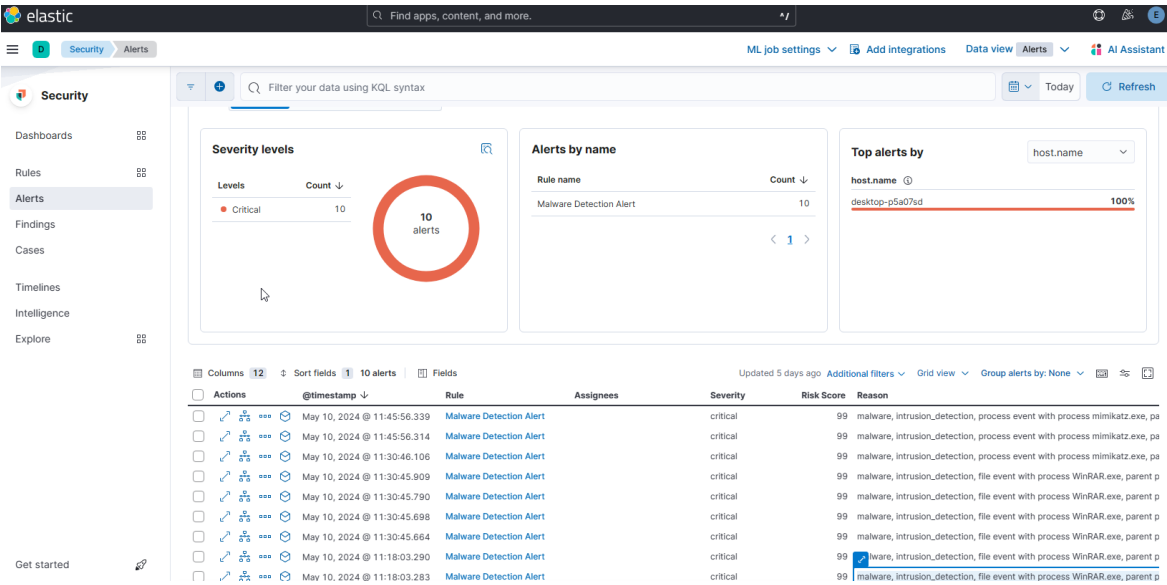


Figure 4.10: Alert Dashboard

Chapter 5

Analysis and Results

This chapter includes all malware analysis process and results. A total of four malware were studied in this project, and the all malware samples were obtained from the Malware Bazaar database[35].

5.1 Analysis

5.1.1 syt.exe

Base information of syt.exe malware:
























SHA256 hash:	 4a589b62115bd62e62995e68e7f92da4011203ce52ab0f31092985a8136637bc
SHA3-384 hash:	 3e65de8c21d545cf6acae1e52a68f7425b1b07f35c8420bb8f993dd5686795d5bb422c690424ac6b41e00dc696d606f2
SHA1 hash:	 73ecd0d486da8daf16ec4c89b298577ce2862dea
MD5 hash:	 d48ff0ecf3271389a65d5e718070f14c
humanhash:	 happy-harry-bluebird-whiskey
File name:	syt.exe
Download:	 download sample
Signature 	 Sytro  Alert 
File size:	51'431 bytes
First seen:	2022-03-02 18:52:04 UTC
Last seen:	2022-04-19 20:32:19 UTC
File type:	 exe
MIME type:	application/x-dosexec
imphash 	 d7b2934b89bc50c5c343ad84032de88e (1 x Sytro)
ssdeep 	 768:filPp7JeTe5MLjH4B5NCPd7m+Z7hl6XmPA+S3y4fBhg6msatoZ:fileK5SYB5s1Zb6XDC4HteO
Threatray 	46 similar samples on MalwareBazaar
TLSH 	 T18D3302017A219BE5C8FB7B718D065515A1E8ECB14EFC8342AA379101FEF81AC88F4D72
Reporter 	 adm1n_usa32
Tags:	  

Figure 5.1: Information of syt.exe

1. Overview

An overview of the syt.exe malware's behaviour is clearly depicted in Elastic's Analysis Event interface. First, the malware calls winlogon.exe, modifies two registry contents, then runs userinit.exe, followed by explorer.exe, and does a number of actions. Finally, the main program syt.exe was run and 50 files were modified.

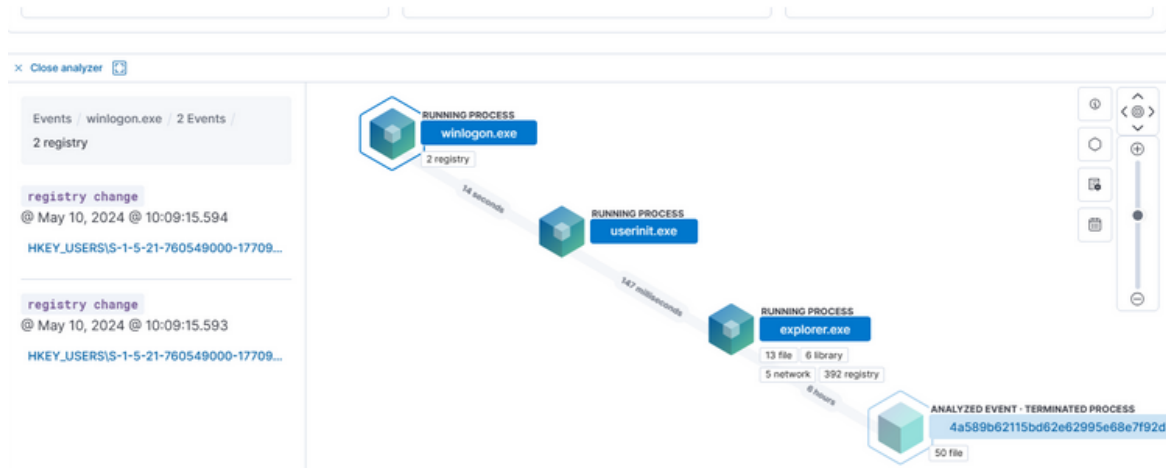


Figure 5.2: syt.exe overview

2. Winlogon.exe

The main responsibilities of Winlogon.exe[36] include loading user configuration files, SAS identification, assigning initial processes to users, etc. The target of the malware to call winlogon.exe is not clear, usually winlogon.exe is a legitimate system process, and in the process of running the program, no abnormality has been detected, so it is assumed that the malware is running winlogon.exe in order to cover up its malicious behaviours.

- Registry change:

HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\Winlogon>PasswordExpiryNotification\NotShownTime

The registry path refers to the specific user and the user's SUID, and the windows registry settings under that user, respectively. What's really interesting is that the malware modifies the **NotShownTime** under **PasswordExpiryNotification** with the following operation: **registry.data.strings 10::09::15, 2024/05/10**

This modification means that from 10 May 2024, the password expiry notification for this particular user has been delayed by 10 days, 9 hours and 15 minutes.

- Registry change:

HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\Winlogon\PasswordExpiryNotification\NotShownReason

Under this registry, the malware modification is **registry.data.strings Password-NeverExpires**. this means that the reason users are not notified of password changes is that the password never expires.

- Purpose One possibility for the malware’s modification of the registry regarding password expiration prompts is that the malware wants to prevent users from changing their passwords, making it easier for hackers to subsequently brute-force the credentials.

3. Userinit.exe

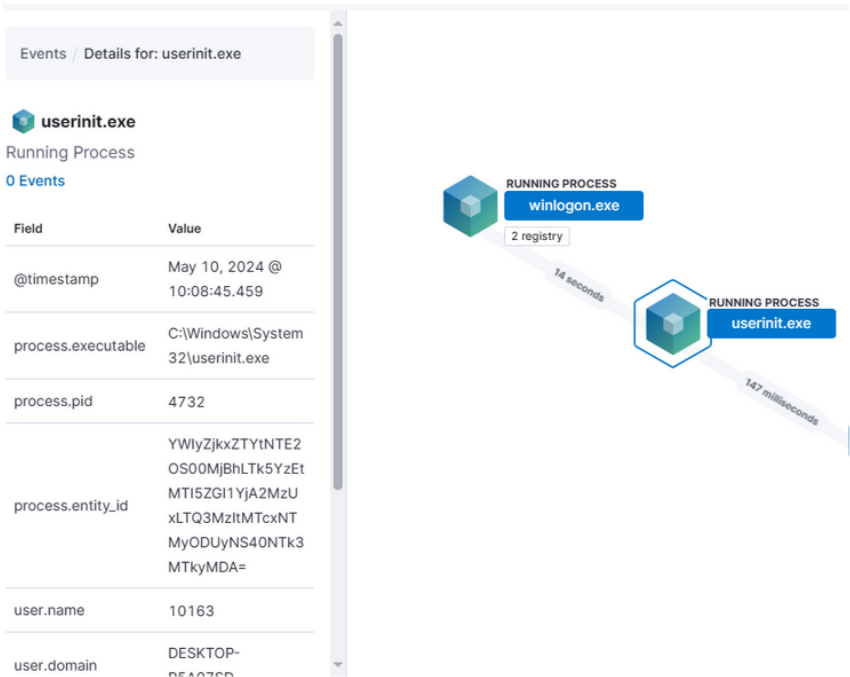


Figure 5.3: syt.exe userinit.exe

Userinit.exe is usually started by winlogon.exe as part of the login process. It ensures that the user’s environment is set up correctly and ready to use. Again this is part of the legitimate processes of the system and no malicious behaviour is detected. Again the assumption is that this is a malware strategy to avoid detection.

4. explorer.exe

explorer.exe has performed a lot of operations. It contains 13 files, 6 libraries, 5 network operations and 392 registry operations. After research, no malicious behaviour was detected.

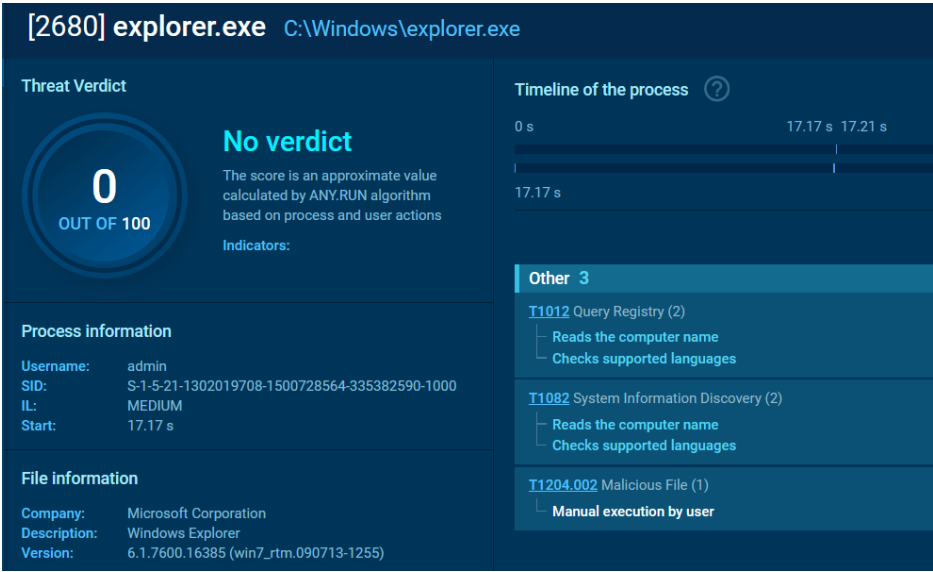


Figure 5.4: syt.exe explorer from any.run [37]

5. syt.exe

The main malware programme changed 50 files, and the evidence can be seen in the target folder :

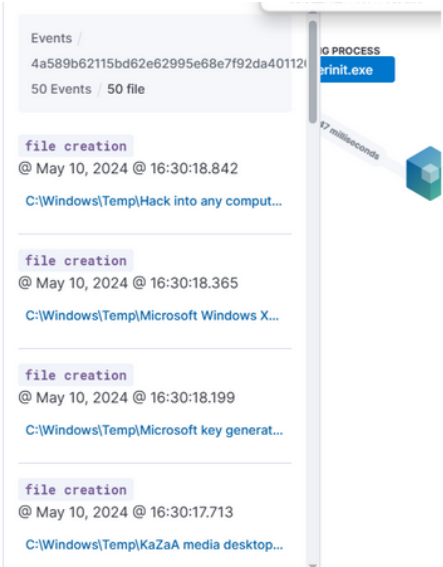


Figure 5.5: syt.exe file changes detection

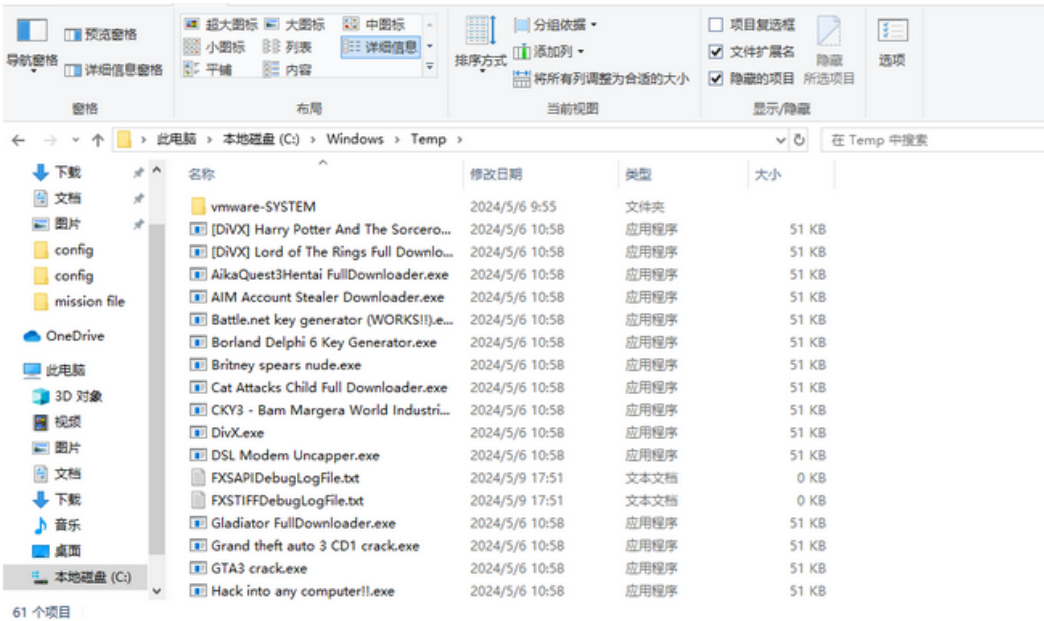


Figure 5.6: syt.exe file Evidence

And then, these files generated by the main malware program were also analysed, and the result for one of them is as follows:



Figure 5.7: One of the exe files

This means that this executable also performs file operations. And this file is also marked as malicious when it is placed in databases such as Virustotal.

6. Result

After detailed analysis of the teardown of each step of the malware’s operation, a reliable hypothesis is that the malware is designed to prevent users from changing their passwords, generating more malware in the target folder (which is consistent with the characteristics of a worm), while running legitimate processes to disguise themselves.

5.1.2 FinalPayload.exe

Base information of FinalPayload.exe:

SHA256 hash:	fbd5ed2986f6dcdfe32fb0a659cb3363c5bb914ab523da589cad645418dc42f3
SHA3-384 hash:	e6cbdded890e3055613ab3bd0189d4b0b3f21eca5c2da253ff4700c236b7d170ab29b62ae778aacfc82cae737fbb1cbae
SHA1 hash:	bddbb63208fc2b777a97b4707321fbfbc096c2cc
MD5 hash:	02b942ad766d717f2d90f6d1c6b69646
humanhash:	network-blue-pluto-lactose
File name:	FinalPayload.exe
Download:	download sample
File size:	287'232 bytes
First seen:	2023-05-11 20:28:34 UTC
Last seen:	Never
File type:	exe
MIME type:	application/x-dosexec
imphash [Ⓢ]	26ea1b45766fe26f9c40750d36f4cf7b (7 x DiskWriter)
ssdeep [Ⓢ]	3072:ZJVD1zx0HrJvtuKm8wVKQ3+nBgBDIQID3RjfpKYHiTWNlkk24Pmqsxu5YUJCU8W:nWFvSH5IQ+hjf8YHuWJjkCU855og9
Threatray [Ⓢ]	19 similar samples on MalwareBazaar
TLSH [Ⓢ]	T1E654E68272A1CD75D1E376BCCB49B320532F3E501B60F15A26B4FD1BBA2558B4E187CA
TrID [Ⓢ]	35.1% (.EXE) Win32 Executable Delphi generic (14182/79/4) 26.1% (.EXE) Win64 Executable (generic) (10523/12/4) 11.1% (.EXE) Win32 Executable (generic) (4505/5/1) 7.4% (.MZIP) WinArchiver Mountable compressed Archive (3000/1) 5.1% (.EXE) Win16/32 Executable Delphi generic (2072/23)
dhash icon [Ⓢ]	cce6b2ccd4e871b2
Reporter [Ⓢ]	Anonymous
Tags:	.exe dropper email worm exe loveware payload worm

Figure 5.8: Information of FinalPayload.exe

1. Overview

FinalPayload.exe seems to run the main program twice (the hash here actually refers to FinalPayload.exe, which can be thought of as a file alias.) Changed four files and the registry six times. Finally ran schtasks.exe and called a library.

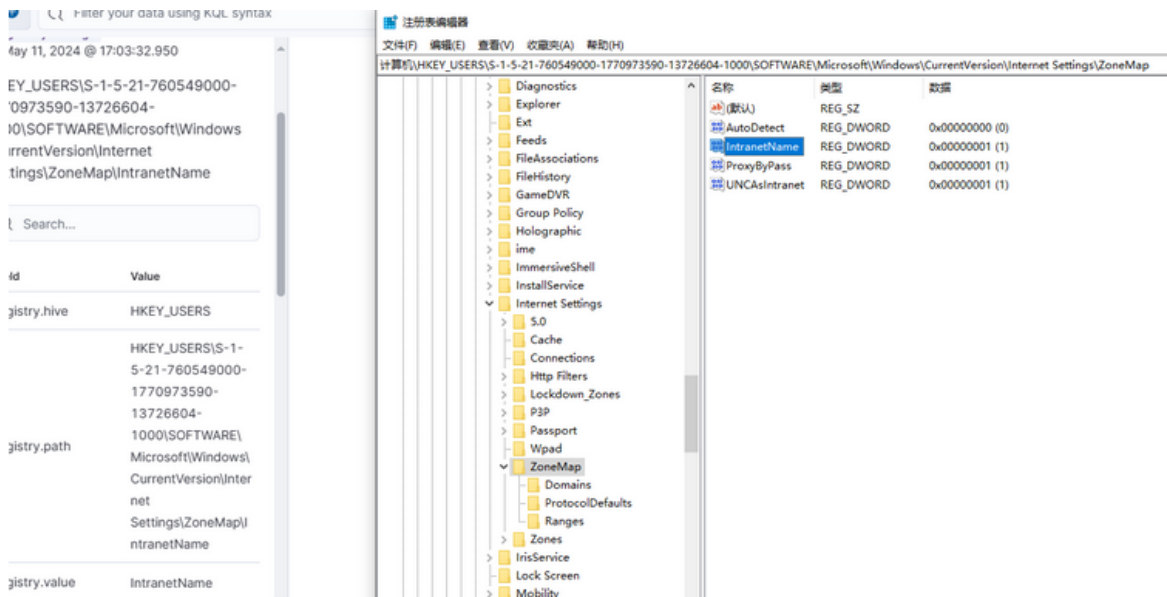


Figure 5.9: Modified registry by malware

2. FinalPayload.exe 1st time

- Registry change:


```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet
```

In this registry path, ZoneMap is used to classify website trust based on trust level. And the next value of UNCAsIntranet represents whether or not Universal Naming Convention (UNC) paths are considered part of the Intranet area in Internet Explorer.

If the value of UNCAsIntranet is set to 1, websites accessed using UNC paths are treated with the same level of security as websites in the Intranet area. This means that security checks are less stringent.

If the value of UNCAsIntranet is set to 0, UNC paths are not considered part of the Intranet area. These paths may have higher security restrictions. This explains why malware was detected that changed this value to 1.
- Registry change:


```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect
```

Auto-detect is a feature in Internet Explorer that automatically determines the safe areas of visited websites. In this registry, **registry.data.strings** was modified by malware to 0. And this may also be to match the UNCAsIntranet value set earlier.

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\
Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName
```

Sites included in the IntranetName list are usually treated with a lower level of security than sites in the Internet zone. This can allow for more relaxed security measures, such as automatic logins or access to internal resources. This registry value is usually a string, and it is not yet clear why malware set it to 1.

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\
Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass
```

ProxyBypass specifies a list of sites or domains that should bypass any configured proxy server. Traffic directed to these bypassed sites will connect directly to the Internet without going through a proxy. The value of ProxyBypass usually consists of a string containing a list of websites or domains. The purpose of changing it to 1 is unclear, it could be to force bypassing all proxies or some kind of vulnerability exploit.

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\
Microsoft\Windows\CurrentVersion\Run\wininit
```

This registry configures applications to start automatically at system startup. Any executables listed here will be launched when the user logs on. So the malware aims to set it to boot up automatically.

```
C:\Users\10163\Desktop\TestMale\Test
Male\fd5ed2986f
6dcdfe32fb0a659c
b3363c5bb914ab5
registry.data.strings 23da589cad64541
8dc42f3\fd5ed29
86f6dcdfe32fb0a6
59cb3363c5bb914
ab523da589cad64
5418dc42f3.exe
```

Figure 5.10: FinalPayload.exe Auto run

3. FinalPayload.exe 2nd time

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\
Microsoft\Windows\CurrentVersion\Run\wininit
```

Same as the previous one ,it may used to make sure that registry is modified.

- File change

Files:

```
\Device\HarddiskVolume1\EFI\Microsoft\Boot\memtest.efi
\Device\HarddiskVolume1\EFI\Microsoft\Boot\bootmgr.efi
\Device\HarddiskVolume1\EFI\Microsoft\Boot\bootmgfw.efi
\Device\HarddiskVolume1\EFI\Boot\bootx64.efi
```

bootmgr.efi: This is the primary boot loader file responsible for launching the Windows operating system on UEFI systems.

bootmgfw.efi: This file might be used for booting in special firmware modes like booting from a recovery drive or for troubleshooting purposes.

memtest.efi: This file is likely a diagnostic tool used to test the system's memory (RAM) for errors.

bootx64.efi: This file might be related to booting 64-bit versions of the Windows operating system on UEFI systems.

It is unclear yet why the malware changed these files.

4. schtasks.exe

Schtasks.exe is a legitimate command line tool for Windows. It is used to schedule tasks on your computer. For example, running programmes or scripts at specific times or intervals.

From the detection result, the malware used this command

```
schtasks.exe /Create /TN wininit /ru SYSTEM /SC ONSTART /TR
'C:\Users\admin\AppData\Local\Temp\FinalPayload.exe(filehash)'
```

/Create: creates a new scheduled task.

/TN wininit: Names the task 'wininit' to disguise it as a legitimate system process.

/ru SYSTEM: runs the task with SYSTEM privileges (the highest access level in Windows).

/SC ONSTART: Schedules the task to run at system startup, ensuring that it is executed every time the computer starts.

/TR

```
'C:\Users\admin\AppData\Local\Temp\FinalPayload.exe'
```

This command is very dangerous, it means that the malware is running with the highest privileges every time the system is booted while masquerading as wininit.exe. this is clear privilege elevation behaviour.

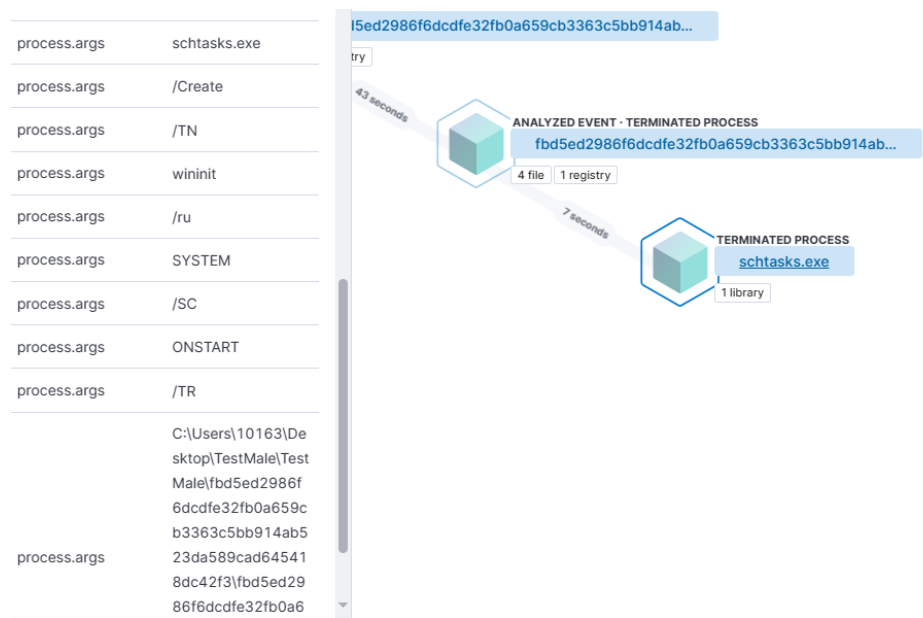


Figure 5.11: The cmd malware used

5. Result

Taken together, each step in the running of the malware has a high likelihood of being aimed at: modifying network security settings so that the machine is able to trust malicious websites that it shouldn't be trusted, and running with the highest privileges every time the machine is switched on.

5.1.3 b23.exe

Base information of b23.exe:

SHA256 hash:	b23183f345d29270a5673d5139fe1fe42a34488e70291746eab567a4468d2e4
SHA3-384 hash:	da9f0c92f426028c80939660616d4b29ee7239cef60c1afcdcaf57c6cdb9f6b27ee31ccd59f0b25da1ab511c79c1d9fa
SHA1 hash:	8f7203463b52baedbbe8eca41a1337746a4f2939
MD5 hash:	4b5f47bd7cbdff09297065b9cfd65d2f
humanhash:	massachusetts-montana-spaghetti-earth
File name:	b23.exe
Download:	download sample
File size:	59'392 bytes
First seen:	2022-04-20 05:35:50 UTC
Last seen:	2022-04-20 06:37:05 UTC
File type:	exe
MIME type:	application/x-dosexec
imphash [Ⓢ]	ebc638f1f6382a3fa539e03cd55aa5
ssdeep [Ⓢ]	768:u2gpFmbvXimSBIWRVJqYOF6dXm3jl3pnBN34X9K0nbcuyD7Uv2Oo+:YKiYAF65m3jIWFNouy8eO
Threatray [Ⓢ]	7'431 similar samples on MalwareBazaar
TLSH [Ⓢ]	T104436E4BBD347A0C57543F00C7B655BDB21B705C32086EB9398E91A6F922D5CE3C1A9
TrID [Ⓢ]	24.8% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13) 21.5% (.EXE) UPX compressed Win32 Executable (27066/9/6) 21.1% (.EXE) Win32 EXE Yoda's Crypter (26569/9/4) 13.1% (.EXE) Microsoft Visual C++ compiled executable (generic) (16529/12/5) 5.2% (.DLL) Win32 Dynamic Link Library (generic) (6578/25/2)
Reporter [Ⓢ]	adm1n_usa32
Tags:	exe worm

Figure 5.12: Information of b23.exe

1. Overview It starts by running the main process, modifying three files and four registries, then runs cmd.exe and modifies eight registries. Afterwards three processes are run by cmd.exe, namely wscript.exe, conhost.exe and ping.exe.

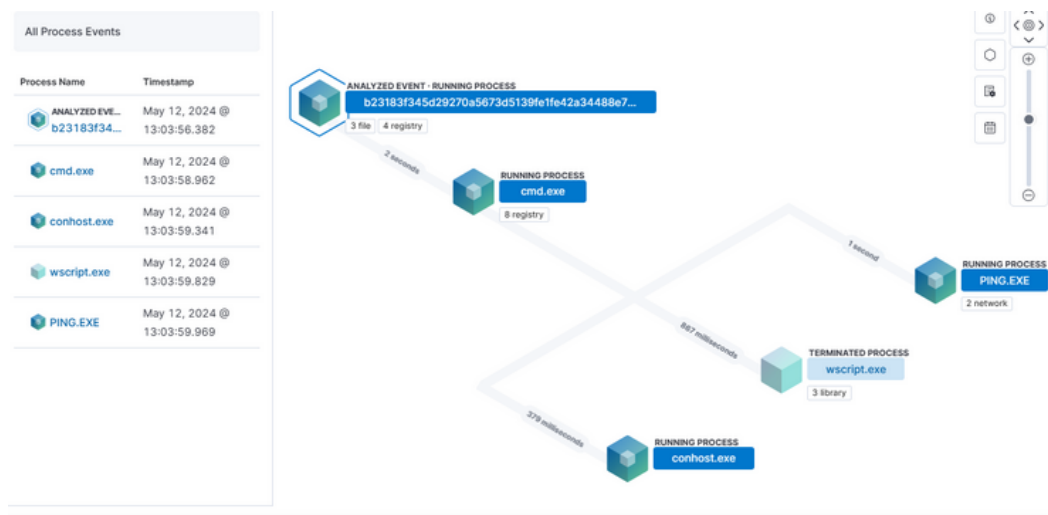


Figure 5.13: Overview of b23.exe

2. b23(filehash).exe

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE
\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\
Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\
Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\
Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet
```

These four registry modifications behave exactly same as FinalPayload.exe.

- File change

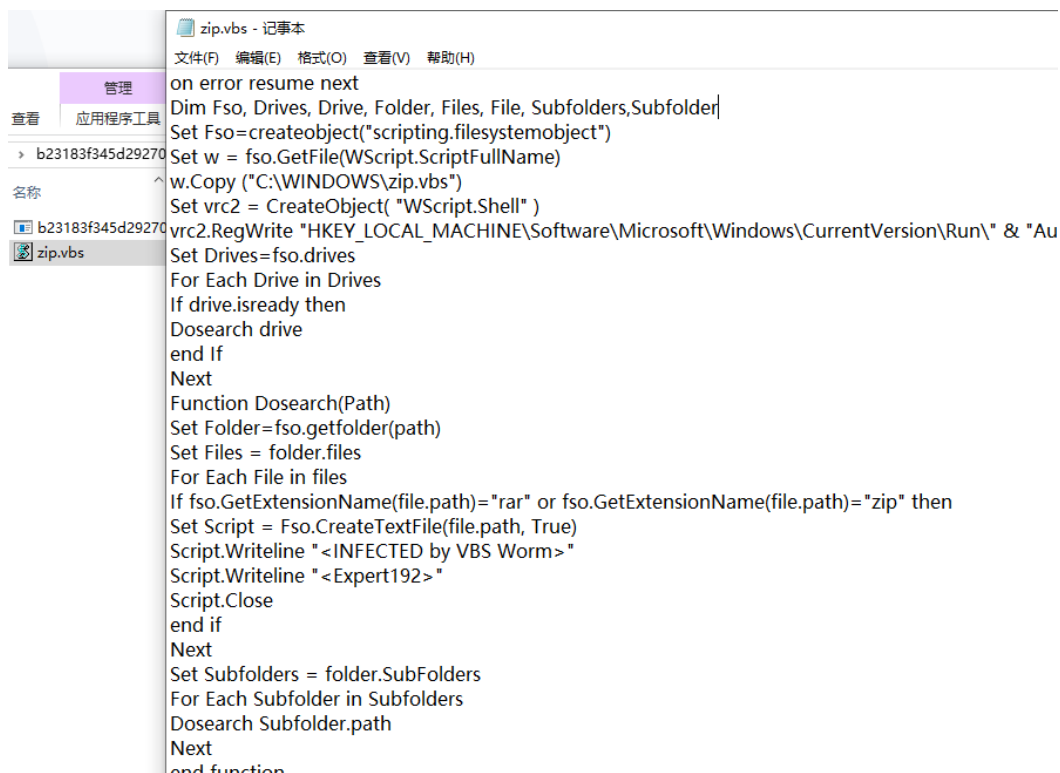
Zip.vbs

Figure 5.14: Zip.vbs

The script checks the extension of each file, e.g. '.rar' or '.zip'. If a match is found, the script creates a new text file in that archive using the same filename. The script then writes malicious text ('<INFECTED by VBS Worm>' and '<Expert192>') to the newly created text file within the archive.

- File change

zipper.bat

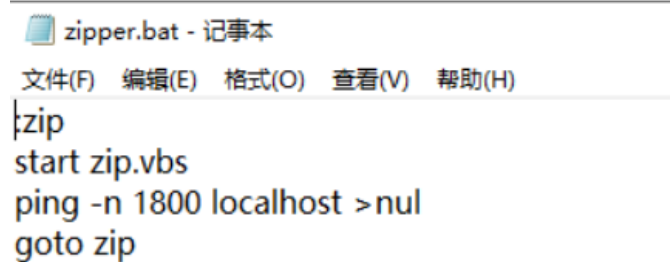


Figure 5.15: zipper.bat

The purpose of this script is to repeat the execution of the 'zip.vbs' VBScript file every 30 minutes.

- File change
Created 96CA...folder which contains the zipper.bat.

3. cmd.exe

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE
\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE
\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\ProxyBypass
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE
\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE
\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet
```

These four changes are same as the previous.

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE
\Microsoft\Windows\CurrentVersion\Explorer\FileExts\.vbs\OpenWithProgids\
VBSFile
```

```
OpenWithProgids\VBSFile
```

This registry entry stores the program that Windows uses by default to open .vbs files. The malware modifies the value of this registry to '00000000', which means that Windows will not know which application is used to open vbs files by default.

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000_Classes\
Local Settings\Software\Microsoft\Windows\Shell
\MuiCache\C:\Windows\System32\WScript.exe.FriendlyAppName
```

```
C:\Windows\System32\WScript.exe.FriendlyAppName
```

:This registry entry stores the user-friendly name displayed for the WScript.exe file located in the Windows system directory. It is not clear why the malware sets this value to '**Microsoft ® Windows Based Script Host**'.

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000_Classes\
Local Settings\Software\Microsoft\Windows\Shell\MuiCache\C:\Windows\System32\
WScript.exe.ApplicationCompany
```

This registry entry stores the 'ApplicationCompany' data associated with the C:\Windows\System32\WScript.exe

The malware appears to use the default value of '**Microsoft Corporation**', the legitimate developer of the WScript.exe file. It is not clear why the malware performs this step, but it may be an attempt to disguise itself.

- Registry change:

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE
\Microsoft\Windows\CurrentVersion
\ApplicationAssociationToasts\VBSFile_.vbs
```

The ApplicationAssociationToasts registry means the path to the Toast notification configuration that appears in Windows when a file type is associated with a program. The malware changes the value of

VBSFile_.vbs
to 0, which disables notifications for .vbs file types.

4. wscript.exe

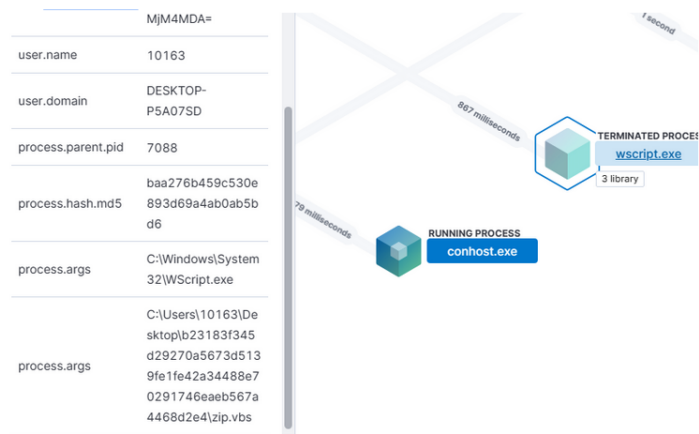


Figure 5.16: Wscript.exe

From the detection result, the malware runs wscript with the command
'C:\Windows\System32\WScript.exe' 'C:\Users\admin\AppData\Local\Temp\zip.vbs'.

This means that the malware is using wscript to run zip.vbs, and the three libraries called during this period are all required for wscript.exe to run. The function of this program is to provide an environment[38] in which the user can execute various scripting languages.

5. conhost.exe

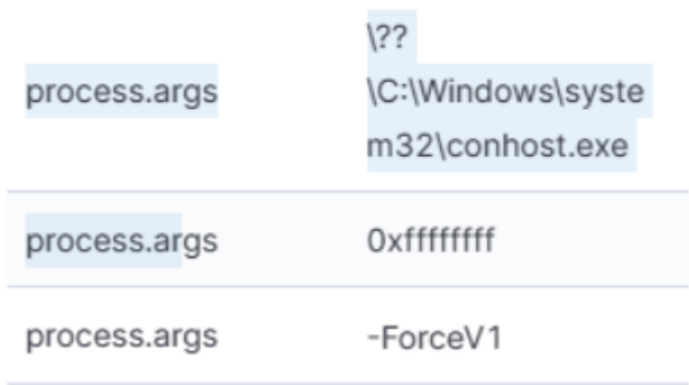


Figure 5.17: conhost.exe

According to the detection result, the malware uses the command 'conhost.exe 0xffffffff -ForceV1'. conhost.exe is a windows core and legitimate program that is responsible for managing console windows. It is unclear what the parameters used by the malware mean.

6. Ping.exe

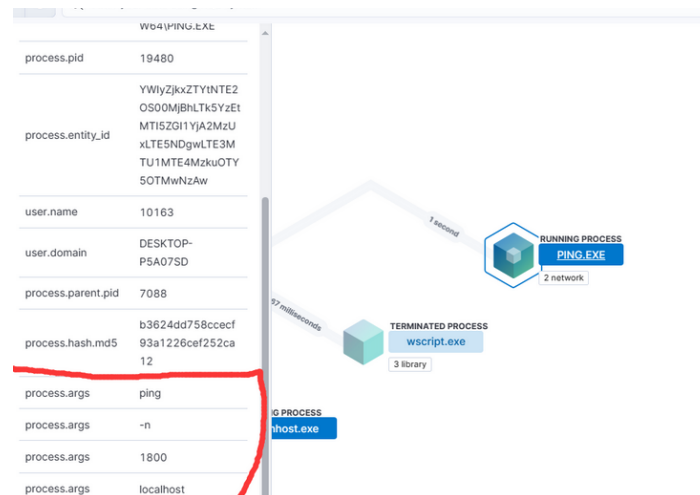


Figure 5.18: Ping.exe

According to the test results, the complete command is 'ping -n 1800 localhost'. This command means to continuously ping the localhost every 1800 seconds (i.e. 30 minutes).

7. Result

The malware has a clear purpose, to modify the registry to ensure that vbs scripts can be run. The malicious vbs script works by overwriting the target file and writing '<INFECTED by VBS Worm>' and '<Expert192>' and modifying the registry to run itself when machine is started.

5.1.4 torn.exe

Base information of torn.exe:

SHA256 hash:	0ea7cb3aeeed6eae9bdf39e4203b288a1e10a48a7cbbbd12fd4fb87c493d5c2e1
SHA3-384 hash:	e96048af828760a801d4e9291292669063fa447b5f313992734d2d4c472e4c1e2814967c07f40ad06f15ebc314244cba
SHA1 hash:	145a7f57de3f6029a81f3e24661bf7e8060cc75e
MD5 hash:	72f69312548871fd3c88c8d306a4d899
humanhash:	ink-artist-mike-arkansas
File name:	torn.exe
Download:	download sample
Signature	Worm.Soulclose Alert
File size:	6'290'507 bytes
First seen:	2022-02-12 07:35:18 UTC
Last seen:	Never
File type:	exe
MIME type:	application/x-dosexec
imphash	5f116d8e20f7d894b4b4ecbad1704009 (2 x SpyEye, 1 x DarkComet, 1 x Worm.Soulclose)
ssdeep	98304:mtRaMMMMM2MMMMMPPMak7497tRaMMMMM2MMMMMPPMak7497tRaMMMMM2MMMMMPPMak3:374O74O74SqnHC
TLSH	T13A566D12B6A04431EC5DAD305A252338F565AE2F7D2CB257BE64BE1C28375F17E38227
File icon (PE):	
dhash icon	e94a6e71e932f033 (4 x Worm.Soulclose)
Reporter	adm1n_usa32
Taqs:	exe soulclose worm Worm.Soulclose

Figure 5.19: Information of torn.exe

1. Overview

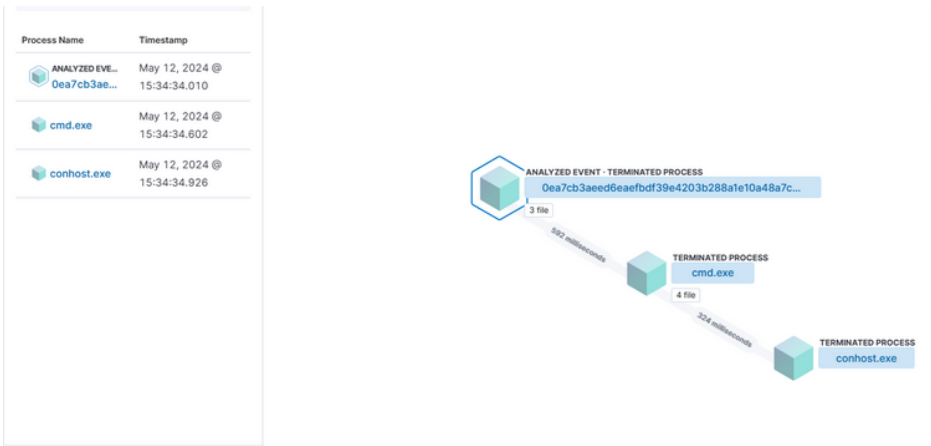


Figure 5.20: Overview of torn.exe

The malware runs a total of three processes, torn.exe, cmd.exe and conhost.exe . In total, seven files were modified.

2. torover.exe

3 files are created

C:\Users\10163\AppData\Local\Temp\~DF44B26E4301CDB2DC.TMP

C:\Users\10163\AppData\Local\Temp\~DF44B26E4301CDB2DC.TMP

C:\Users\10163\AppData\Local\Temp\tornkill.bat

The purpose of this bat file is to delete the first two files.

3. cmd.exe

	d3348ac2130c7e7
process.hash.md5	54754a6e9cb053b09
process.args	C:\Windows\system32\cmd.exe
process.args	/c
process.args	C:\Users\10163\AppData\Local\Temp\0ea7cb3aead6eaeafbdf39e4203b288a1e10a48a7cbbbd12fd4fb87c493d5c2e1kill.bat

Figure 5.21: cmd.exe

According to the detection result, the command it runs is **cmd.exe /c tornkill.bat**This line of command is used to run the bat file and close the cmdline when it is finished.
4 files are deleted

C:\Users\10163\AppData\Local\Temp\tornkill.bat

C:\Users\10163\AppData\Local\Temp\tornkill.bat

C:\Users\10163\Desktop\torn\torn.exe

C:\Users\10163\Desktop\torn\torn.exe

After using the previous command,these 4 files are deleted.

4. conhost.exe

The malware used '**conhost.exe 0xffffffff -ForceV1**', same as the previous one.It is not yet clear why it ran this line of command.

5. Result

According to the analysis, this software does not exhibit serious malicious operations. The detection system gave the reason as manipulation of cmd.exe in order to execute commands and posted a warning about file creation behaviour.

Chapter 6

Discussion and Conclusion

The overall structure of this project is divided into two parts: the first part is to install and run Elastic Stack, and the second part is to dynamically analyse the malware samples. This chapter will summarise each of these two parts.

6.1 Discussion

This section focuses on the issues experienced during the process of installing Elastic Stack and running analysis for this project and the cautions for using Elastic Stack. This section aim to provide solutions to the some problems for researchers using Elastic Stack in the future.

1. Elastic cluster health status is yellow

The issue first appeared early in the process of this project, after launching `Elasticsearch.bat` and displaying `'current.health="YELLOW" message="Cluster health status changed from [RED] to [YELLOW] (reason: [shards started [[.apm-source-map][0]])." previous.health="RED" reason="shards started [[.apm-source-map][0]]'`

According to the official Elastic documentation[39], a yellow or red health status means that there are shards unallocated. This means there is a risk of losing data.

After reviewing multiple documents and forums for solutions, the final reason identified was because of insufficient storage space for the VMs used for this project. The solution was to use a single machine for malware analysis and allocate a sufficiently large amount of disk space to that machine, rather than reallocating the already limited disk space to multiple virtual machines.

2. Fleet server is not healthy

At one time during the process of this project, this issue was the biggest obstacle. The causes of this problem are numerous and are detailed below

- Overview

After opening Fleet on the left side of Kibana, if Status shows unhealthy or offline, it proves that the problem occurs.

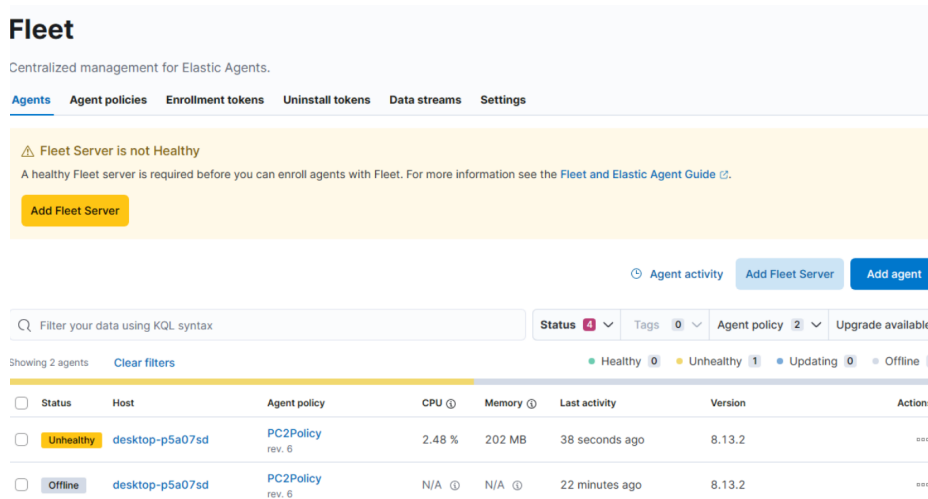


Figure 6.1: Fleet server is not healthy

- Check the logs - unable to connect

The first thing to do when this type of problem occurs is to check the logs. Click on the **Host name** and then click **logs** to go to the logs screen.

The first problem that occurred in this project was that fleet was unable to connect to elasticsearch. there are two settings that need to be checked in order to solve this problem: the address of the **fleet server** and the **privilege** to run elasticsearch.bat. Firstly, make sure that the fleet server is using the correct ip address, something like 192.168.71.32, not 127.0.0.1. Even if the fleet server is running locally, it should not be configured as 127.0.0.1 or localhost. secondly, make sure that elasticsearch and kibana bat files start with administrator, i.e. right click on the bat file and click run as administrator.

Timestamp	event.dataset	Message
11:45:31.656	elastic_agent	ty check with elasticsearch: elastic fail 401: security_exception: error attempting to authenticate request: cluster_block_exception: blocked by: [SERVICE_UNAVAILABLE/1/state not recovered / initialized];
11:45:34.030	elastic_agent.fleet_server	[elastic_agent][error] Unit state changed fleet-server-default (STARTING->FAILED): Error - failed version compatibility check with elasticsearch: elastic fail 401: security_exception: error attempting to authenticate request: cluster_block_exception: blocked by: [SERVICE_UNAVAILABLE/1/state not recovered / initialized];
11:45:34.304	elastic_agent.fleet_server	[elastic_agent.fleet_server][error] failed to fetch elasticsearch version
11:45:34.305	elastic_agent	[elastic_agent.fleet_server][error] Fleet Server failed
11:45:34.305	elastic_agent	[elastic_agent][error] Unit state changed fleet-server-default-fleet-server-fleet_server-930b7b7a-1fea-43e2-89fe-004196af7437 (STARTING->FAILED): Error - failed version compatibility check with elasticsearch: elastic fail 401: security_exception: error attempting to authenticate request: cluster_block_exception: blocked by: [SERVICE_UNAVAILABLE/1/state not recovered / initialized];
11:45:36.571	elastic_agent.fleet_server	[elastic_agent.fleet_server][error] failed to fetch elasticsearch version
11:45:36.620	elastic_agent.fleet_server	[elastic_agent.fleet_server][error] Fleet Server failed
11:45:36.640	elastic_agent	[elastic_agent][error] Unit state changed fleet-server-default-fleet-server-fleet_server-930b7b7a-1fea-43e2-89fe-004196af7437 (STARTING->FAILED): Error - failed version compatibility check with elasticsearch: elastic fail 401: security_exception: error attempting to authenticate request: cluster_block_exception: blocked by: [SERVICE_UNAVAILABLE/1/state not recovered / initialized];

Figure 6.2: Unable to connect

- Check the logs - SSL certificate problem

When this problem occurs, the error entry in logs complains: **ssl remote key was not ok.**

Timestamp	event.dataset	Message
11:45:31.656	elastic_agent	ty check with elasticsearch: elastic fail 401: security_exception: error attempting to authenticate request: cluster_block_exception: blocked by: [SERVICE_UNAVAILABLE/1/state not recovered / initialized];
11:45:34.030	elastic_agent.fleet_server	[elastic_agent][error] Unit state changed fleet-server-default (STARTING->FAILED): Error - failed version compatibility check with elasticsearch: elastic fail 401: security_exception: error attempting to authenticate request: cluster_block_exception: blocked by: [SERVICE_UNAVAILABLE/1/state not recovered / initialized];
11:45:34.304	elastic_agent.fleet_server	[elastic_agent.fleet_server][error] failed to fetch elasticsearch version
11:45:34.305	elastic_agent	[elastic_agent.fleet_server][error] Fleet Server failed
11:45:34.305	elastic_agent	[elastic_agent][error] Unit state changed fleet-server-default-fleet-server-fleet_server-930b7b7a-1fea-43e2-89fe-004196af7437 (STARTING->FAILED): Error - failed version compatibility check with elasticsearch: elastic fail 401: security_exception: error attempting to authenticate request: cluster_block_exception: blocked by: [SERVICE_UNAVAILABLE/1/state not recovered / initialized];
11:45:36.571	elastic_agent.fleet_server	[elastic_agent.fleet_server][error] failed to fetch elasticsearch version
11:45:36.620	elastic_agent.fleet_server	[elastic_agent.fleet_server][error] Fleet Server failed
11:45:36.640	elastic_agent	[elastic_agent][error] Unit state changed fleet-server-default-fleet-server-fleet_server-930b7b7a-1fea-43e2-89fe-004196af7437 (STARTING->FAILED): Error - failed version compatibility check with elasticsearch: elastic fail 401: security_exception: error attempting to authenticate request: cluster_block_exception: blocked by: [SERVICE_UNAVAILABLE/1/state not recovered / initialized];

Figure 6.3: SSL problem

There are multiple reasons that could cause this problem, so first, by using the command: **elastic-endpoint.exe test output** the system will output a test result to help narrow down the problem.

```

PS C:\Program Files\Elastic\Endpoint> .\elastic-endpoint.exe test output
Testing output connections using config file: [C:\Program Files\Elastic\Endpoint\elastic-endpoint.yaml]

Using proxy:

Elasticsearch server: https://192.168.71.130:9200
Status: HTTP code 401: Unauthorized

Global artifact server: https://artifacts.security.elastic.co
Status: Couldn't resolve host name [Could not resolve host: artifacts.security.elastic.co]

Fleet server: https://localhost:8221
Status: SSL peer certificate or SSH remote key was not OK [SSL certificate problem: unable to get local issuer certificate]
Help: Host needs to trust server cert or server cert needs to be added to Elasticsearch/Fleet config
PS C:\Program Files\Elastic\Endpoint>

```

Figure 6.4: Test output

In this example, the problem is clearly shown: the server cannot connect because host cannot trust the server's certificate, i.e., 401. Solving this problem requires adding trust to elasticsearch's certificate. By default, elasticsearch uses a certificate named

http_cert.ca

which is located in the

\config\certs

folder in the elasticsearch directory. To install the certificate, right-click on the certificate, click Install, and install the certificate to a **'Trusted Root Certificate Authority'**.

- Endpoint can not connect to the output server

After configuring the certificate, if the log shows this issue, then it may be necessary to configure the certificate fingerprint in the outputs. Click on fleet, then go to settings, click on the pencil symbol to the right of outputs and paste the certificate fingerprint.

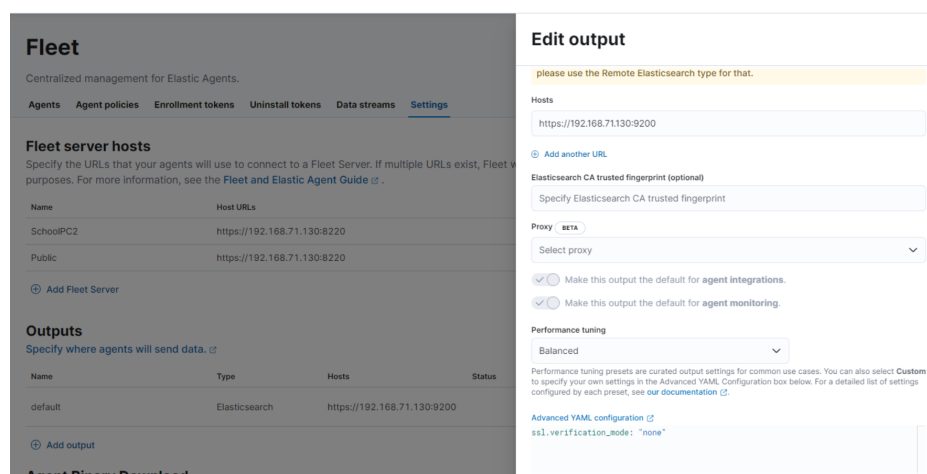


Figure 6.5: Fingerprint

6.2 Conclusion

This section focuses on answering the two questions posed in Introduction1:how to secure the system when running malware, and how to determine if a program is malicious.

6.2.1 Secure system

Two methods are used in this project to ensure security: firstly by disconnecting the network, and then by ensuring that after each run of malware, the snapshot feature is used to restore it to its previous clean state.

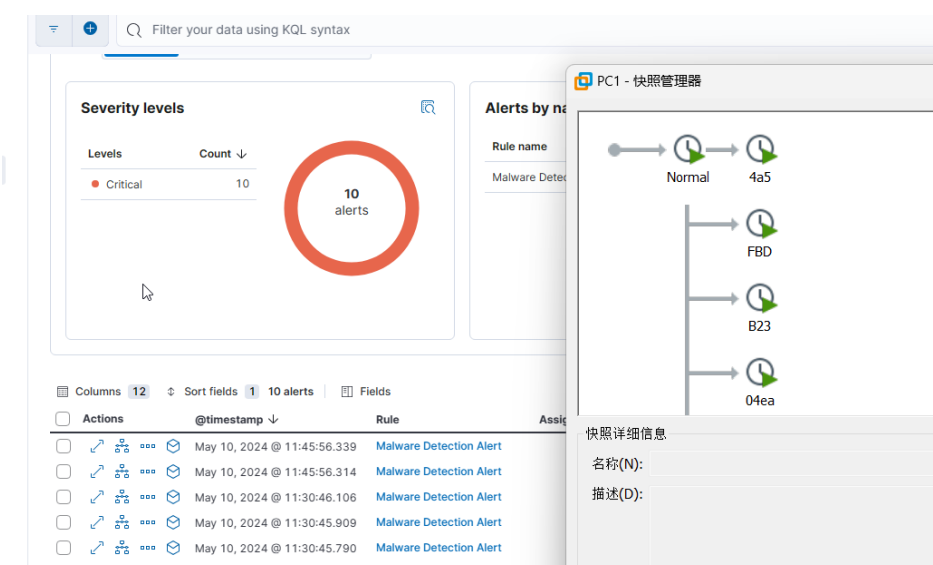


Figure 6.6: Snapshot

6.2.2 Detect malware

Based on the results of previous chapter 5, this project’s conclusions on how to identify malware is divided into the following sections:

1. Registry

In the four malware samples used in this project, some of the registry modification behaviours that are clearly malicious are listed below:

- Password related

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\Winlogon>PasswordExpiryNotification\NotShownTime
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\Winlogon>PasswordExpiryNotification\NotShownReason
```

Once a process is detected trying to modify these registries, it may mean that it wants to change the relevant settings regarding password notifications. The settings related to passwords or credentials are considered sensitive, so this behaviour is considered malicious.

- Network related

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\UNCAsIntranet
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\AutoDetect
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMap\IntranetName
```

Once a process is detected trying to modify these registries, it may mean that it wants to change the relevant settings regarding network trust. This may allow the machine to trust malicious websites and download more malware from them. That's why these settings about modifying the network are also considered malicious behaviour.

- Auto run

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\wininit
```

If the boot self-start is set, the program will run without the user being aware of it. Self-booting without the administrator's consent is considered malicious behaviour.

- Files related

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\FileExts\.vbs\OpenWithProgids\VBSFile
```

```
HKEY_USERS\S-1-5-21-760549000-1770973590-13726604-1000\SOFTWARE\Microsoft\Windows\CurrentVersion\ApplicationAssociationToasts\VBSFile_.vbs
```

Usually when these file configurations are modified, some malicious scripts are trailing behind. So when a process is seen modifying to start apps with some specific suffixes or disabling Toast notifications, etc., these behaviours are considered to be malicious.

2. Files

File manipulation may not be all that malicious, as in the case of the torn.exe sample. However, it is also a form of suspicious behaviour, and the focus is on identifying whether the specific file being operated is suspicious or not. For example, the script file generated in b23.exe is very typical of malicious programs. In this project, two main means are used to identify whether the generated files are suspicious or not: firstly, manual analysis. Firstly, it can be analysed manually. It can be opened by decompiler software or notepad to check the contents of the program. Secondly, the file's hash can be searched directly in Virustotal and other databases to see whether it is marked as malicious.

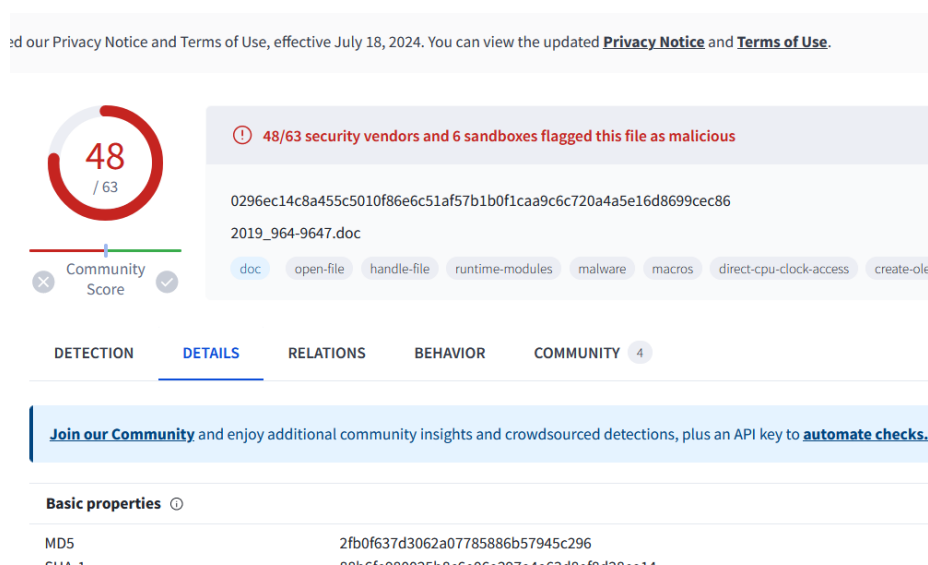


Figure 6.7: Virustotal

3. System commands

Operating system commands are likewise not necessarily malicious. However it is important to focus on the specific commands used by the program, the parameters and details of which can be found in Elastic defend. In the previous example, the command

```
'PING -N 1800 LOCALHOST'
```

```
conhost.exe 0xffffffff -ForceV1
```

may not be suspicious. However, the command

```
schtasks.exe /Create /TN wininit /ru SYSTEM /SC ONSTART /TR
```



```
'C:\Users\admin\AppData\Local\Temp\FinalPayload.exe(filehash)'
```

is obviously a malicious behaviour. This is because the parameters of this command indicate that it intends to operate with SYSTEM highest privileges, which is very much a very sensitive command and can be immediately concluded as malicious behaviour.

It is important to determine if the instructions executed by a process are malicious after Elastic Defend reports them. This can be done with the help of nowadays developed AI tools like ChatGPT, Gemini, etc. to reveal the meaning of the instructions' parameters.

6.3 Summary and future work

By documenting in detail the installation and running process of Elastic Stack and analysing four pieces of malware, this thesis provides a novel and convenient approach to dynamic malware analysis, aiming to improve the efficiency and accuracy for malware researchers in order to combat today's increasing number of cybercrimes.

This project still has many shortcomings, one of the most unfortunate being the failure to implement the topology first conceived due to disk space constraints, as well as the study of malware propagation through the network. In future research, if new experimental machines can be used, then the primary goal is to realise the plan. In addition, too much time was spent on solving the operational problems of the Elastic Stack, resulting in a low number of malware being analyzed. In future research, with the experience of this project, more malware samples will be selected for study.

Bibliography

- [1] Cisco. *What is malware*. Nov. 2023. URL: <https://www.cisco.com/site/us/en/learn/topics/security/what-is-malware.html>.
- [2] ENISA. *Malware*. Feb. 2024. URL: <https://www.enisa.europa.eu/topics/incident-response/glossary/malware>.
- [3] Sun Tzu. *The Art of War*. 1. ed. Capstone Publishing, 2010.
- [4] Ori Or-Meir et al. "Dynamic malware analysis in the modern era—A state of the art survey". In: *ACM Computing Surveys (CSUR)* 52.5 (2019), pp. 1–48.
- [5] Fang Min. *What Is a Worm Virus (Computer Worm)?* Jan. 2024. URL: <https://info.support.huawei.com/info-finder/encyclopedia/en/Worm+Virus.html>.
- [6] Rob. Lemos. *Virulent worm calls into doubt our ability to protect the Net*. June 2011. URL: <http://news.cnet.com/2009-1001-270471.html>.
- [7] Lawrence Abrams. *CryptoLocker Ransomware Information Guide and FAQ*. Oct. 2013. URL: <http://www.bleepingcomputer.com/virus-removal/cryptolocker-ransomware-information>.
- [8] Brian Krebs. *Who and What Is Coinhive?* Mar. 2018. URL: <https://krebsonsecurity.com/2018/03/who-and-what-is-coinhive/>.
- [9] Georgios Kambourakis et al. *Botnets: Architectures, countermeasures, and challenges*. CRC Press, 2019.
- [10] Marios Anagnostopoulos. "Amplification DoS Attacks". In: *Encyclopedia of Cryptography, Security and Privacy*. Springer, 2020, pp. 1–3.
- [11] reasonlabs. *What is Self-propagation?* Sept. 2023. URL: <https://cyberpedia.reasonlabs.com/EN/self-propagation.html>.
- [12] Saurabh. "Advance malware analysis using static and dynamic methodology". In: *2018 International Conference on Advanced Computation and Telecommunication (ICA-CAT)*. IEEE. 2018, pp. 1–5.
- [13] Sainadh Jamalpur et al. "Dynamic malware analysis using cuckoo sandbox". In: *2018 Second international conference on inventive communication and computational technologies (ICICCT)*. IEEE. 2018, pp. 1056–1060.

- [14] Peyman Pahlevani et al. "The Impact of Network Configuration on Malware Behaviour". In: *The International Conference on Cybersecurity, Situational Awareness and Social Media*. Springer. 2023, pp. 91–104.
- [15] Lele Wang et al. "Cuckoo-based malware dynamic analysis". In: *International Journal of Performability Engineering* 15.3 (2019), p. 772.
- [16] Aaron Walker, Muhammad Faisal Amjad, and Shamik Sengupta. "Cuckoo's malware threat scoring and classification: Friend or foe?" In: *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE. 2019, pp. 0678–0684.
- [17] Avira. *Cuckoo Sandbox vs. Reality*. Nov. 2014. URL: <https://www.avira.com/en/blog/cuckoo-sandbox-vs-reality-2>.
- [18] Rasmi-Vlad Mahmoud et al. "Redefining Malware Sandboxing: Enhancing Analysis through Sysmon and ELK Integration". In: *IEEE Access* (2024).
- [19] Elastic Stack. *Functions of elasticsearch*. 2024. URL: <https://www.elastic.co/cn/elasticsearch/features>.
- [20] Saurabh Chhajed. *Learning ELK stack*. Packt Publishing Ltd, 2015.
- [21] elastic. *Elastic Defend*. 2024. URL: <https://docs.elastic.co/en/integrations/endpoint>.
- [22] elastic. *Fleet and Elastic Agent overview*. 2024. URL: <https://www.elastic.co/guide/en/fleet/current/fleet-overview.html>.
- [23] Mohammad Nassiri et al. "Malware elimination impact on dynamic analysis: An experimental machine learning approach". In: *Handbook of Big Data Privacy* (2020), pp. 359–370.
- [24] Rami Sihwail et al. "Malware detection approach based on artifacts in memory image and dynamic analysis". In: *Applied Sciences* 9.18 (2019), p. 3680.
- [25] Yi-Ting Huang, Yeali S Sun, and Meng Chang Chen. "TagSeq: Malicious behavior discovery using dynamic analysis". In: *Plos one* 17.5 (2022), e0263644.
- [26] Chih-Hung Lin, Hsing-Kuo Pao, and Jian-Wei Liao. "Efficient dynamic malware analysis using virtual time control mechanics". In: *Computers & Security* 73 (2018), pp. 359–373.
- [27] Long Chen et al. "Detection, traceability, and propagation of mobile malware threats". In: *IEEE Access* 9 (2021), pp. 14576–14598.
- [28] Peter Eder-Neuhauser, Tanja Zseby, and Joachim Fabini. "Malware propagation in smart grid networks: metrics, simulation and comparison of three malware types". In: *Journal of Computer Virology and Hacking Techniques* 15 (2019), pp. 109–125.

- [29] Jun Li, Devkishen Sisodia, and Shad Stafford. "On the detection of smart, self-propagating internet worms". In: *IEEE Transactions on Dependable and Secure Computing* (2022).
- [30] Răzvan Stoleriu, Alin Puncioiu, and Ion Bica. "Cyber attacks detection using open source ELK stack". In: *2021 13th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*. IEEE. 2021, pp. 1–6.
- [31] Hamad Almohannadi et al. "Cyber threat intelligence from honeypot data using elasticsearch". In: *2018 IEEE 32nd International Conference on Advanced Information Networking and Applications (AINA)*. IEEE. 2018, pp. 900–906.
- [32] Neel Shah, Darryl Willick, and Vijay Mago. "A framework for social media data analytics using Elasticsearch and Kibana". In: *Wireless networks* 28.3 (2022), pp. 1179–1187.
- [33] MalwareBazaar. *Tsuchigumo.bat*. Aug. 2023. URL: <https://bazaar.abuse.ch/sample/8d1d36a7ad23626341f658815bfd21a6274f703aca2126bddfad63fa749041be/>.
- [34] Yini Wang et al. "Modeling the propagation of worms in networks: A survey". In: *IEEE Communications Surveys & Tutorials* 16.2 (2013), pp. 942–960.
- [35] MalwareBazaar. *MalwareBazaar*. 2024. URL: <https://bazaar.abuse.ch/>.
- [36] Microsoft. *Responsibilities of Winlogon*. July 2021. URL: <https://learn.microsoft.com/en-us/windows/win32/secauthn/responsibilities-of-winlogon>.
- [37] any.run. *Analysis syt.exe*. 2022. URL: <https://app.any.run/tasks/b5ae58be-345c-4291-bc4d-fdd85fa6f3e1/>.
- [38] Microsoft. *WScript*. 2023. URL: <https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/wscript>.
- [39] elastic. *Red or yellow cluster status*. 2024. URL: <https://www.elastic.co/guide/en/elasticsearch/reference/current/red-yellow-cluster-status.html>.