12-15-2022

Malware Detection and Analysis

Namratha Suraneni
Grand Valley State University

Malware Detection and Analysis

Namratha Suraneni
Advisor: Xinli Wang

Abstract - Malicious software poses a serious threat to the cybersecurity of network infrastructures and is a global pandemic in the form of computer viruses, Trojan horses, and Internet worms. Studies imply that the effects of malware are deteriorating. The main defense against malware is malware detectors. The methods that such a detector employ define its level of quality. Therefore, it is crucial that we research malware detection methods and comprehend their advantages and disadvantages. Attackers are creating malware that is polymorphic and metamorphic and has the capacity to modify their source code as they spread. Furthermore, existing defenses, which often utilize signature-based approaches and are unable to identify the previously undiscovered harmful executables, are significantly undermined by the diversity and volume of their variations. Malware families' variations exhibit common behavioral characteristics that reveal their origin and function. Machine learning techniques may be used to detect and categorize novel viruses into their recognized families utilizing the behavioral patterns discovered via static or dynamic analysis. In this paper, we'll talk about malware, its various forms, malware concealment strategies, and malware attack mechanisms. Additionally, many detection methods and classification models are presented in this study. The method of malware analysis is demonstrated by conducting an analysis of a malware program in a contained environment.

Index: Malware, Malware Analysis Techniques, Malware Detection Techniques, Classification Models, Image analysis.

I. INTRODUCTION

Malware is a short for malicious software, and as its name implies, malwares is designed to hurt computers and their users by stealing data, damaging files, or just engaging in mischievous activities to harm the user [1]. It has been stated that malware is extensively disseminating and that computer security incidents have dramatically increased. Malware prevents networks from developing. The internet-based apps that are the target of malware. The necessity to identify and disable malware as soon as possible has increased since practically every aspect of life now depends on the Internet to enhance its level of service and prevent the bad effects that these malwares might cause.

Each year, there is an increase in the volume and sophistication of cyberattacks, which affect governments, businesses, and individuals equally and result in significant reputational, financial, and societal harm. As an illustration, hostile cyber activities costed the U.S. economy alone up to 109 billion USD in 2016 [2]. Cybercriminals currently carry out a variety of cyberattacks, including as man-in-the-middle attacks, malware, and birthday strikes. Malware assaults in particular have become one of the most difficult problems in the cybersecurity field and the major instrument used by hackers. As a result, several tools and techniques have been developed to identify and stop malware assaults. By assessing whether a particular software has malicious intent or not, antimalware technologies stop malware. Specifically, the majority of anti-malware techniques don't have low enough mistake rates. Additionally, when they encounter unknown viruses, their performance significantly suffers. While 360,000 new malware samples are discovered every day [3]. The competition between malware creators and defenders is intensifying as both malware in the wild and anti-malware software advance. There is still a long way to go in the pursuit of scalable and reliable automated malware detection technologies.

This study provides an overview of the many types of malware, malware analysis methodologies, and malware detection strategies. This study also offers a realistic examination of malware in sandbox environment.

II. OVERVIEW OF MALWARE

As the technologies advances, various malware types are created by the malicious actors to bypass the security features of a system. The malware also uses some vulnerabilities to exploit the system securities. So, it is very important to take into consideration the following information about malware [1].

A. Types of Malwares

Figure 1: Types of Malware

Due to the wide variety of malware, it is crucial that each malware item be clearly identified and differentiated from other harmful software [2]. Therefore, the various malware categories are as follows:

1) Ransomware

Ransomware is software that uses encryption to disable access to the target's data until a ransom is paid. Victim organizations are blocked to access the data until payment is made, but there is no guarantee that payment will generate the required decryption key or that the provided decryption key will work properly [4].

This year, the city of Baltimore was attacked by a form of ransomware called RobbinHood, which halted all city activities including tax collection, property transfers and government emails for weeks. The attacks have cost the city more than $18 million to date, and the costs are still rising. The same type of malware was used against the city of Atlanta in 2018, costing $17 million [3].

2) Adware

Adware tracks the users browsing activities to determine which ads to serve. Adware is similar to spyware, but does not install software on the user's computer or record keystrokes. The danger of adware is that it destroys the user's privacy. The data obtained by the adware is matched with overtly or covertly obtained data about the user's activities elsewhere on the Internet to create a profile of that person, including who they are friends with, what they bought, where they travelled and more [1]. This information may be shared or sold to advertisers without users' consent.

An adware called Fireball infected 250 million computers and devices in 2017, hijacking browsers to change default search engines and track web activity. However, this malware can be more than just an annoyance. Three quarters of them were able to execute code remotely and download malicious files [3].

3) Spyware

Spyware collects information about user activity without the user's knowledge or consent. This may include passwords, PINs, payment information and unstructured messages. The use of spyware is not limited to desktop browsers. It may even work with important apps and mobile phones. Even if the data stolen is small, spyware can often ripple through an organization, slowing performance and hurting productivity [2].

Using the hotel's WIFI to target corporate and government leaders, DarkHotel used several types of malware to gain access to the systems of certain powerful individuals. Once gained, the attackers installed a keylogger to obtain the target's passwords and other sensitive information [3].

4) Trojan horse

Trojans disguise themselves as target code or software. If downloaded by an unsuspecting user, the Trojan can take control of the victim's system for malicious purposes. Trojans can be hidden in games, programs, even software patches, or embedded in phishing email attachments [4].

Emotet is a sophisticated banking Trojan that has been around since 2014. Emotes are hard to fight. This is because it evades signature-based detection, is stable, and includes a player module to aid in propagation. This Trojan is so common that it has been warned by the US Department of Homeland Security [3]. It shows that Emotet will pay state, local, tribal and territorial governments up to $1 million to remediate each incident.

5) Virus

A virus is code that inserts itself into a program and is executed when the program is running. Once inside a network, viruses can be used to steal sensitive data, launch DDoS attacks, or conduct ransomware attacks [2]. Viruses cannot run or reproduce unless an infected program is running. This dependence on a host program differentiates viruses from Trojan horses that users must download and worms that run without a program.

Nimda is a sophisticated virus containing a mass mailing worm component that spreads as an email attachment called README.EXE. This affects users of Windows 95, Windows 98, Windows Me, Windows NT 4, and Windows 2000 [3].

6) Worm

Worms target operating system vulnerabilities to install themselves on networks. It can gain access in a variety of ways, including backdoors built into software, unwanted software vulnerabilities, or flash drives [2]. Once deployed, worms can be used by malicious attackers to launch DDoS attacks, steal sensitive data, or conduct ransomware attacks [1].

Stuxnet was probably developed by American and Israeli intelligence agencies to thwart Iran's nuclear program. It has entered to Iran through a flash drive. Because the environment was so air-gapped, its creators did not expect Stuxnet to escape the target network, but it did [3]. Once spread, Stuxnet spread wildly, but did little damage because its only function was to interfere with the industrial controllers that ran the uranium enrichment process.

7) Root kit

A rootkit is software that allows malicious attackers to remotely control a victim's computer with full administrative privileges [1]. Rootkits can

be injected into applications, kernels, hypervisors, or operating systems. They are spread through phishing, malicious attachments, malicious downloads, and compromised shared drives. Rootkits can also be used to hide other malware such as keyloggers.

Zacinlo infects systems when users download fake VPN apps. After installation, Zacinlo performs a security check for competing malware and tries to remove it. It then opens an invisible browser and interacts with the content like a human by scrolling, highlighting, and clicking. This activity is intended to trick the behaviour analysis software. Zacinlo loading occurs when the malware clicks on an ad in an invisible browser [3]. This scam offers malicious agents a portion of the commission by clicking on the ad.

8) Bot/Botnet

A robot is a software program that performs automated tasks based on commands. These are used for legitimate purposes such as search engine indexing, but when used for malicious purposes, they take the form of self-propagating malware that can connect to central servers. And are often used to create a network of bots, which is used to launch a flood of remote-controlled mass attacks such as DDoS attacks. Botnets can grow very large. For example, the Mirai IoT botnet ranged from 800,000 to 2.5 million computers [3].

Echobot is a variant of the famous Mirai. Echobot attacks a wide range of IoT devices and exploits more than 50 different vulnerabilities, including exploits in Oracle WebLogic Server and VMWare's SD-Wan networking software. Additionally, the malware looks for older, unpatched systems. Echobot can be used by malicious actors to launch DDoS attacks, disrupt supply chains, steal sensitive supply chain information, and sabotage businesses [3].

9) Keylogger

A keylogger is a type of spy software that monitors user activity. Keyloggers have legal uses. Businesses can use them to monitor employee activity, and families can use them to track their children's online behaviour, banking information, and other confidential information. Keyloggers can be injected into user’s system through phishing, social engineering, or malicious downloads [1].

A keylogger called Olympic Vision has been used in business email compromise (BEC) attacks targeting merchants in the United States, the Middle East, and Asia. Olympic Vision uses phishing and social engineering techniques to infect targeted systems, steal sensitive data, and spy on business transactions. Keyloggers are not sophisticated, but are available on the black market for $25, making them very accessible to malicious actors [3].

10) Wiper

A Wiper is a type of malware with one goal which is to erase user data so that it cannot be recovered [2]. The wiper is used to destroy the computer networks of public or private companies in different sectors. Attackers also use wipers to cover the tracks left behind after the break-in, weakening the victim's ability to respond [2].

Malware called WhisperGate has been reported to have been deployed against Ukrainian targets. The incident was widely reported to involve three separate components deployed by the same enemy. This includes malicious bootloaders, Discord-based downloaders, and file cleaners that destroy local disks if detected. This activity occurred at the same time that several websites belonging to the Ukrainian government were defaced [3].

11) Mobile malware

Mobile malware threats are as diverse as those targeting desktops, including Trojans, ransomware, and ad click fraud [2]. They are distributed through phishing and malicious downloads and are particularly problematic on jailbroken phones, which usually lack the default protections that were part of the device's original operating system.

Triada is a rooted Trojan that has entered the supply chain of millions of Android devices with pre-installed malware. Triada accesses sensitive areas of the operating system and installs spam programs. Spam programs display ads and sometimes replace legitimate ads. When a user clicks on one of the unauthorized ads, the revenue from that click goes to Triada developers [1].

12) Fileless malware

Fileless malware doesn't install anything initially, instead making changes to native OS files, such as PowerShell and WMI. Fileless attacks are not caught by antivirus software because the operating system recognizes the edited file as legitimate. Because these attacks are stealthy, they are up to ten times more successful than traditional malware attacks [2].

Astaroth is a Fileless malware campaign that targets users with links to LNK shortcut files. When the user downloaded the file, the WMIC tool was launched along with many other legitimate Windows tools. These tools downloaded redundant code that only ran in memory, leaving no evidence that vulnerability scanners could detect. The attackers then downloaded and executed a Trojan horse that stole the credentials and uploaded them to a remote server [3].

B. Malware Concealment Techniques

Malicious actors have used several malware concealment techniques in order to avoid detection

by anti-malware programs. These techniques are as follows:

1) Encryption

By using this technique, malware is encrypted. and comprises of malicious programs, keys, and encryption and decryption methods. Every time, the attacker creates a brand-new malware version using a fresh encryption technique and key. Since the decryption technique is constant, there is a larger chance of being discovered [2]. This approach aims to prevent static analysis and slowing down the investigation.

2) Packing

Malware executable files are compressed and encrypted using a packing process. Reverse engineering techniques or the proper unpacking algorithm are required to detect malware that uses a packing strategy, which can be challenging at times because it calls for knowledge of the actual packing/compression process [1]. Two types of packing are UPX and Upack.

3) Obfuscation

One of the various ways employed by malware to avoid static analysis techniques and conventional anti-malware solutions that rely on hashes and strings for malware identification and analysis is obfuscation [2]. By using this strategy, the core logic of the code is obscured, preventing unauthorized access to the code. Obfuscated malware's destructive activity is hidden until it is triggered. Inconsequential jumps and using trash instructions are crucial obfuscation techniques [4].

4) Polymorphism

Malware that is polymorphic in nature, is made to seem different every time it is run, yet it keeps all of the original code. A polymorphic virus can use an infinite number of encryption methods, as opposed to encryption techniques, such that a piece of the decryption code is altered in each implementation. The transformation engine is often contained in malware that is encrypted [4]. A random encryption algorithm is generated whenever a mutation takes place to re-encrypt the virus and engine with a new decryption key. Different malicious behaviors may be concealed by encryption techniques. Since the original code is still present, polymorphic malware is quite simple to find. The first polymorphic virus was discovered is 1260 [1].

5) Metamorphism

The malicious code in metamorphism malware is altered during each run to produce a unique version that bears no resemblance to native code but still functions as intended. Metamorphic malwares fall into two kinds. open-source malware, such as the Conficker worm, that mutates by contacting other websites online [1]. Open-world malware, such as the Win32/Apparition infection, may reprogram itself without interacting with the outside world by altering binary code or using pseudocode representation [2].

C. Malware Attack Vectors

Malware also uses various methods to spread beyond the initial attack vector to other computer systems. The definition of a malware attack vectors may include the following:

- Email attachments containing malicious code can be opened and executed by unsuspecting users. If these emails are forwarded, the malware can spread deeper into the organization and further compromise the network.
- Malware can spread quickly when users access and download infected files, such as file servers based on Common Internet File Systems (CIFS) and Network File Systems (NFS).
- File sharing software allows malware to replicate itself on removable media and on computer systems and networks.
- Peer-to-Peer (P2P) file sharing allows malware to enter by sharing seemingly harmless files, such as music or images.
- Remotely exploitable vulnerabilities allow hackers to gain access to systems regardless of geographic location, with little or no intervention from the computer user.

III. MALWARE DETECTION AND CLASSIFICATION

Malware threats are becoming increasingly complicated. Malware is still the most potent danger to the online world, despite advancements in detection & classification methods and models over time [5]. Malware detection and classification are crucial because they determine which family of malware the malicious program belongs to, and on that basis, malware prevention or anti-virus solutions may be developed with a distinctive signature to identify the malware.

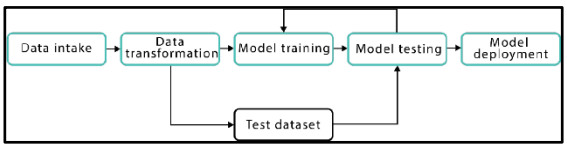


Figure 2: Malware Detection and Classification Process

A. Malware Detection Techniques

Utilizing methods and technologies to detect, prevent, notify, and handle the malware threats is malware detection [6]. Basic malware detection methods can assist in identifying and limiting

known risks. Artificial intelligence and machine learning are used by advanced malware detection solutions to proactively search for and uncover new and undiscovered malware threats [6].

1) Signature-based detection

Signature-based detection uses unique digital footprints called signatures, of software programs running on the protected system. Antivirus programs scan the software, identify its signatures, and compare them to signatures of known malware. Antivirus products use large databases of known malware signatures [5]. This database is usually maintained by a security research team run by the antivirus vendor. This database is updated frequently and the latest version is synchronized with the protected device.

When an antivirus program detects software that matches a known signature, it stops the process and quarantines or deletes it. It's a simple and effective approach to malware detection and an important first line of defense. However, as attackers become more sophisticated, signature-based approaches are unable to detect a wide variety of new threats [7].

2) Checksumming

This method is a form of signature analysis that involves calculating a cyclic redundancy check (CRC) checksum. Checksumming help to make sure that the files are not corrupted. The main drawback of signature-based detection is that it creates a large database and generates false positives. Checksums are designed to handle this issue.

Hackers often use polymorphic malicious advertisements to evade detection by signature-based detection methods [5]. Polymorphic viruses can modify themselves during replication and remove fixed search strings. Hackers usually encode a random set of decryption commands in the form of a non-static key in the virus code. So, when security teams detect a malicious signature, the malware no longer contains code fragments, making it undetectable [8]. Since variable code has no detectable signature, other malicious code detection techniques are required, such as:

- *Statistical analysis* - The frequency of processor commands is examined using statistical analysis to ascertain whether a file is contaminated.
- *Cryptanalysis* - An equation system is used in cryptanalysis to decipher encrypted viruses. The decryption program's algorithm and keys are rebuilt by the cryptanalysis system, which then uses the algorithm to decode segments of the virus's overall body [5].
- *Heuristics* - Malware detection solutions use heuristics to monitor and analyze behavioral data to spot suspicious activities. The group needs to look for dangerous code linked to suspicious conduct. The security team may then sort suspicious instances by priority and carry out more research [5].
- *Reduced masks* - When getting static code, the malware detection solution might get around the necessity for an encryption key by using components inside the encrypted virus body. The static code generated might show the malware's mask or signature.

3) Application whitelists

Application whitelisting is the opposite of the attack signature approach. Instead of deciding what software to block, the antivirus keeps a list of approved applications and blocks everything else [7]. This is not a perfect solution, but it is very effective, especially in high security environments. Legitimate applications often contain security vulnerabilities or introduce unnecessary features that increase the attack surface. In some cases, the program itself is harmless, but using it can threaten end-user device [8]. For example, in some environments it may be necessary to block web browsing or email. Application whitelisting works best on strictly task-focused devices, such as web servers and Internet of Things (IoT) devices.

4) Machine learning behavior analysis

The Machine learning behaviour analysis is also known as a "static" detection technique because it relies on binary rules to match or not match the processes running in the environment [6]. Static malware detection cannot be learned. the solution is flexible to add more rules or change them over time to expand the coverage. In contrast, new dynamic techniques based on artificial intelligence and machine learning (AI/ML) enable security tools to distinguish between legitimate and malicious files and processes, even when they do not match known patterns or signatures [7]. It does this by observing file behavior, network traffic, process frequency, deployment patterns, and more.

Over time, these algorithms learn what "bad" files look like, allowing them to detect new and unknown malware. AI/ML malware detection is known as “behavioral” detection because it is based on analysing the behaviour of suspicious processes [7]. These algorithms have malicious behavior thresholds, and if a file or process exhibits abnormal behavior above the threshold, it is determined to be malicious. Behavioral analysis is powerful, but it can miss malicious processes or misclassify legitimate processes as malicious. Additionally, attackers can manipulate the AI/ML training process [8].

5) Endpoint Protection Platform (EPP)

EPP is deployed on endpoints such as employee workstations, servers, and cloud-based

resources [8]. They act as the first line of defense that can detect and block threats before they can harm users' sensitive assets. EPP uses several techniques to detect and block malware, which are as follows:

- *Static Analysis* - EPP uses traditional static analysis techniques to identify known malware types and allow/deny applications flagged by administrators.
- *Behavioral Analysis* - EPP adds behavioral analysis to identify unknown threats or known malware that use evasion tactics such as hopping and encryption [5].
- *Sandboxed inspection* - EPP can run suspicious content in a safe box isolated from the main operating system. This allows the technician to explode a file and observe its behavior to see if it is truly malicious.
- *Content Disarmament and Reconstruction (CDR)* - EPP removes malicious elements from legitimate content and allows users to access the content itself [8]. For example, if a Word document contains malicious macros, CDR can remove the macros and allow the user to access the file instead of blocking the file completely.

In addition to these techniques, EPP can proactively protect user environment, such as isolating endpoints from the network when malware is detected.

6) Endpoint Detection and Response (EDR)

EDR solutions complement EPP solutions and enable security teams to detect and respond to attacks on endpoint devices. If EPP fails to contain threats, EDR can:

- *Alert Triage and Investigation* - EDR provides information rich data from endpoints that allows security analysts to identify symptoms of an attack and investigate them to confirm security incidents [6].
- *Threat Hunting* - EDR allows analyst to proactively search for endpoints and examine associated data for signs of compromise. Once analysts spot threats on endpoints, EDR platforms can be used for incident response. For example, analysts can quarantine all malware-affected devices, wipe and reimage infected endpoints, and run automated security plays [8]. Analyst can use security playbooks to coordinate their response to malware threats across multiple security tools, such as firewalls, network segmentation, intrusion prevention systems (IPS), and email security. Many EDR solutions have built-in EPP functionality [5].

B. Malware Classification Models

Malware classification have traditionally depended on pattern matching using signatures taken from particular malware samples. While straightforward and effective, signature scanning is easily thwarted by a variety of widely used evasive techniques [10]. Because of this, statistical and machine learning-based solutions have emerged that are more resistant to code alteration. As a result, malware authors have created sophisticated varieties of malware that change the structural and statistical characteristics of their code, which can lead to the failure of statistical models [9]. To counteracts these types of malware various machine learning models are used for classification. These models are as follows:

1) Multilayer Perceptron

A perceptron may be used to create a classifier based on a threshold since it computes a weighted sum of its components in the form of a hyperplane. In situations when the data itself is not linearly separable, it follows that a perceptron cannot give optimal separation [10]. This is a serious restriction since even something as simple as the XOR function cannot be linearly separated. An artificial neural network that uses many layers in the shape of perceptrons is known as a multilayer perceptron (MLP). MLPs may properly mimic more complicated functions since they are not constrained to linear decision boundaries like single layer perceptrons.

The link between linear support vector machines (SVM) and SVMs based on nonlinear kernel functions is quite similar to that between perceptrons and MLPs. Given that there are hidden layers separating the input from the output and it is unclear how changing the weights in these hidden levels would influence the output or the other hidden layers, training an MLP would seem to be difficult [11]. Today, backpropagation is typically used to train MLPs. An important innovation that made deep learning feasible was the realisation that backpropagation may be utilised for training neural networks [9].

2) Convolutional Neural Network

Fully linked layers are what artificial neural networks often employ. A fully connected layer has the benefit of being able to deal with correlations between any points in training vectors in an efficient manner [10]. However, because to the enormous number of weights that must be learnt, completely linked layers are impractical for large training vectors. A convolutional neural network (CNN), on the other hand, is built to handle local structure. When important information is not local, a convolutional layer cannot be expected to perform effectively [11].

The advantage of CNNs is that because there are less weights, convolutional layers can be learned considerably more quickly than fully connected layers [12]. The majority of the crucial structure in

images—such as edges and gradients—is local. As a result, CNNs are a perfect tool for picture analysis, and they were actually created specifically for image categorization. In contrast, CNNs have excelled in a number of other problem fields [9]. Any issue where local structure predominates is typically a candidate for CNNs.

3) Recurrent Neural Network

MLPs and CNNs are examples of feedforward neural networks, meaning that there is no "memory" of past feature vectors and that data is sent straight through the network. Each input vector in a feedforward network is handled separately from all other input vectors. While feedforward networks are suitable for a variety of issues, dealing with sequential data is not one of them [10]. A feedforward neural network can gain context or memory by using a recurrent neural network (RNN).

Backpropagation via time, a kind of backpropagation, is used to train RNNs (BPTT). The tendency of the gradient computation to become unstable, leading to "vanishing" or "exploding" gradients, is an issue that is particularly significant in BPTT. The number of time steps is restricted in order to solve these issues, but doing so also helps to restrict the usefulness of RNNs [12]. As an alternative, specific RNN designs that allow the gradient to flow across extended timespans are utilised.

4) Long Short-Term Memory

A type of RNN topologies called long short-term memory (LSTM) networks is made to handle long-range dependencies. In other words, LSTM can handle long "gaps" between the time a feature first appears and the time the model really needs it. Due to vanishing gradients, this is often not viable with simple RNNs [11]. An LSTM has an extra conduit for information flow, which is a fundamental distinction between it and a standard, vanilla RNN. This means that in addition to the concealed state, there is a second state known as the cell state that may be utilised to effectively store data from earlier phases. During backpropagation, the cell state is intended to act as a gradient "highway." In this method, there is less likelihood that the gradient will disappear (or expand) along the way and can "flow" considerably further back [10].

5) Gated Recurrent Unit

The LSTM design has seen various variations due to its widespread popularity. The majority of these variations are modest, differing just slightly from a typical LSTM [16]. A gated recurrent unit (GRU), however, differs significantly from an LSTM. There are fewer parameters in a GRU, despite the fact that its internal state is more complicated and less understandable than an LSTM's [9]. As a result, training a GRU is simpler than training an LSTM, necessitating less training data.

6) ResNet152

A residual network (ResNet) includes extra connections that correspond to identity layers, whereas LSTM employs a complicated gating structure to facilitate gradient flow. By efficiently skipping over layers during training thanks to these identity layers, a ResNet model can lower the depth of training and lessen gradient pathologies [10]. ResNet is intuitively able to create larger networks by initially training across a much shallower network, with later training steps serving to detail the intermediary connections.

Pyramidal cells in the brain, which have a similar trait in that they connect "layers" of neurons, served as an inspiration for this method. A special deep ResNet architecture called ResNet152 has been pre-trained using a sizable picture dataset [12]. The output layer of the ResNet152 model is retrained particularly for the malware classification challenge.

7) VGG-19

A dataset with more than 106 pictures was used to pre-train the 19-layer convolutional neural network known as VGG-19. This architecture has excelled in several competitions and has been used to a wide range of image-based issues [9]. One of the two instances of transfer learning for image-based malware classification uses the VGG-19 architecture and a pretrained model [12].

IV. MALWARE ANALYSIS

The process of identifying and minimizing possible dangers in a website, application, or server is known as malware analysis [15]. Understanding a suspicious file or URL's behavior and intended purpose is a critical step in ensuring computer security as well as the safety and security of an organization with relation to sensitive data. Vulnerabilities are addressed through malware analysis before they become major issues [16].

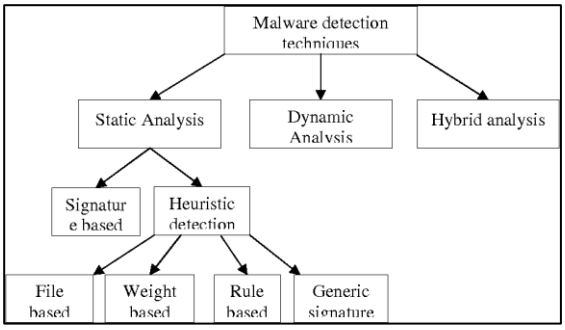


Figure 3: Malware Analysis Tree

Benefits of Malware Analysis

Security analysts and incident responders can greatly benefit from malware research [13]. Here are some of the procedure's main advantages:

- Determining the attack's origin.
- Calculating the harm caused by a security threat.
- Determining the exploitation potential, vulnerability, and necessary patching measures for malware.
- Practically speaking, occurrences are prioritized according to the seriousness of the threat.
- Identifying and blocking any concealed Indicators of Compromise (IOC).
- Enhancing the effectiveness of notifications, alerts, and IOC.
- Adding context to the search for risks.

Types of Malware Analysis

There are three types of malware analysis that can be conducted:

- Static malware analysis.
- Dynamic malware analysis.
- Hybrid malware analysis.

1) Static analysis

Static malware analysis works in a manner akin to that of statistical and signature-based analysis. In reality, it frequently combines elements of the two methods [13]. Most malware exhibits common behavioral and statistical indicators. A security professional or anti-virus software may evaluate the application and determine whether it's probable malware based on these. Additionally, it is not necessary to start the malware software in order to identify it. A malware is identified by looking at its control flow graphs and opcode sequences [14].

The process of extracting strings from the executable file is frequently used to identify questionable programming. Searching and replacing files, establishing connections to external servers, encrypting executables, loading certain libraries and functions, etc. To understand how a piece of code works, it may also be reduced down into assembly language [14].

2) Dynamic analysis

The suspicious software is run in a secure, sandbox environment virtual during dynamic analysis so that it cannot impact the real systems. This enables the user to see the suspected malware in action and determine if it is malware by looking at how it behaves [13]. Controlling the execution flow and seeing how the sample interacts with the system, such as efforts to create persistence or access sensitive data, also enables malware analysts to gain a deeper knowledge of the virus. In terms of delivering results, dynamic analysis already outperforms static analysis in a number of areas. Run-time actions are far more difficult to obscure or hide than static binary code [15]. Without the analyst needing to examine its internals, the malware is just carrying out its intended function.

3) Hybrid analysis

In order to overcome the weaknesses of each methodology, hybrid analysis uses approaches from both. When unpacking the binary files or reading them in assembly code, some operations that might be masked at run-time might be seen [16]. Similar to this, obfuscated opcode may be exposed after execution when the outcomes or actions are seen in real time.

Malware analysis

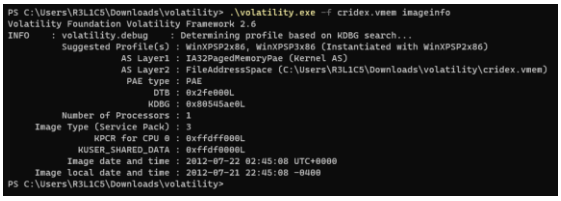
In this section, we are going to demonstrate malware analysis using the tool volatility in the windows environment.

We'll be utilising volatility, one of the most well-liked volatile memory software analysers, in this part. With the aid of this programme, we can examine a volatile memory dump from a machine that could be infected. This software will assist us in retrieving important data such as active processes, recently updated files, or even user browsing history that has been saved in the computer's memory.

In this part, we'll perform a number of volatility commands using the following straightforward case: the Cridex malware

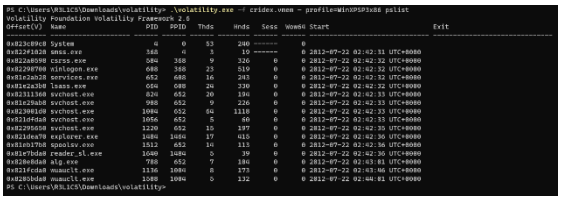
Analyses of dump

To learn more about the memory dump, we need to execute the `imageinfo` command as the first step in the volatility plugin by giving the `imageinfo` and `-f` options for our dump file.

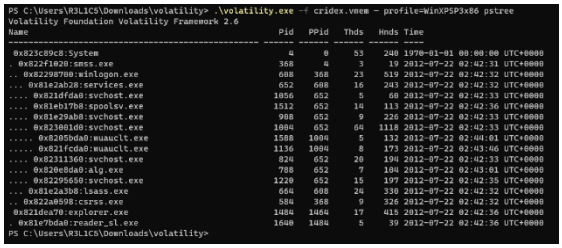


Now that we know the operating system from which this memory dump originates (WinXPSP2x86). We can now start the investigation by giving volatility the OS profile (—profile=WinXPSP2x86) and attempting to determine what took place on the victim's machine.

Let's use the `pslist` plugin to determine what processes were active.



To display the processes and their parent processes instead of the `pslist` plugin, the command `pstree` is used



At first inspection, we can see a strange process with the name ‘reader sl.exe’ and the parent process (PPID) ‘explorer.exe’, which was one of the last processes to run on the computer.

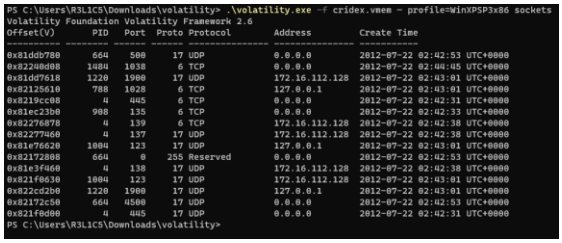
Before digging deeper into these two processes, let's perform one final command. When operating on the computer, processes that attempt to disguise themselves will be listed by `psxview`; this plugin may be quite helpful.

Except for our instance, no processes appear to be hidden; if they are, “False” will be displayed in the first two columns (`pslist` and `psscan`).

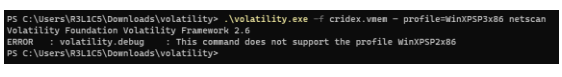
Let's return to this examination; after seeing the processes, it would be wise to look at the computer's active sockets and open connections. These several plugins will be used in this process: sockets, `connscan`, and `netscan`

`Connscan` is a plugin that scans for TCP connections,

Sockets prints a list of open sockets,

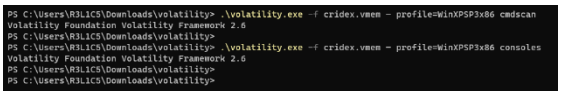


And netscan scans an image for connections and sockets but cannot be used in our scenario owing to the profile used.

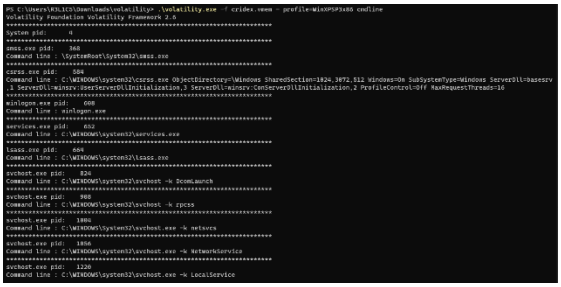


In our example, the process with PID 1484 utilises two TCP connections by looking at our command history outputs we can easily link the PID 1484 to the process explorer.exe. One of these TCP connections, utilising port 1038 and connecting to the target IP address 41.168.5.140, is still active, as can be seen.

Now let's examine the most recent commands executed using `cmdscan`, consoles, and `cmdline` plugins.



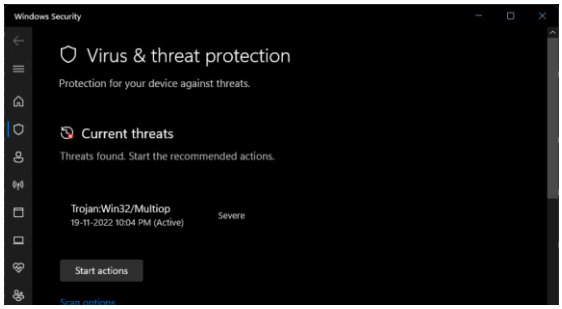
The first two plugins, consoles and `cmdscan`, did not have any data in their buffers. Consoles extracts command history by looking for _CONSOLE INFORMATION, while `cmdscan` extracts command history by looking for _COMMAND HISTORY.



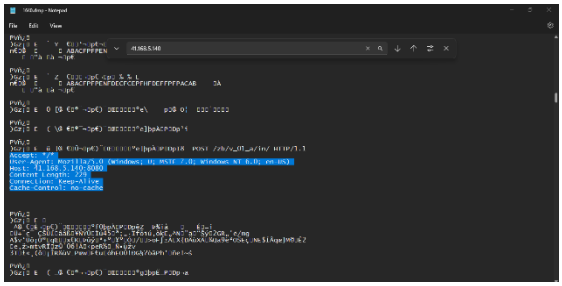
However, the `cmdline` plugin that shows command-line parameters for processes did provide us with useful data. The processes started with PID 1484 and 1640 have, in fact, completed their whole route at this point. More and more questions are being raised about the ‘Reader sl.exe’ process.

As of right now, we know that this process was started by the Explorer process and that it is a traditional Adobe Reader programme. However, we saw that this exact same process was maintaining a connection to an external IP. But we should not leap to conclusions too early. Instead, examine the relevant executable and its analysis using `procdump` and `memdump`, respectively, by supplying the -p 1640, which is its PID, and —dump-dir, which is the directory where these dumps are extracted.

Reader sl.exe is restored in the first file, ‘executable.1640.exe,’ and the process's accessible memory is depicted in the dump file, ‘1640.dmp’, which was extracted. Windows instantly quarantined the file when the.exe file was produced, declaring it to be a Trojan.

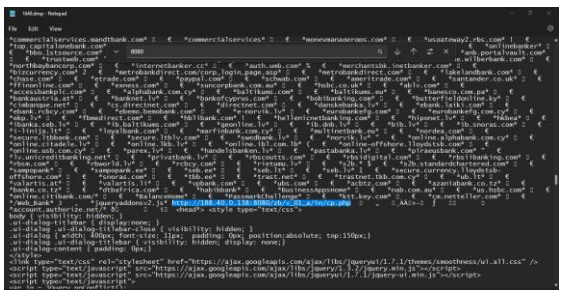


In order to perform a quick examination of these files, we need to open the file in notepad.exe; we have to search for the information because most dumps include a tonne of information. In our instance, we're seeking for a relationship between the data previously taken from the dump—specifically, the tcp connection established with the IP address 41.168.5.140 and with this 1640 process.



We now have additional context for the material obtained thanks to the usage of the grep tool along with the -C *NUMBER option. Here, it is easy to observe that the executable ‘Reader sl.exe’ is sending POST requests to the target IP 41.168.5.140, perhaps stealing data from the victim's computer system.

These intriguing domains can be discovered, by using the strings command to examine the extraction dump

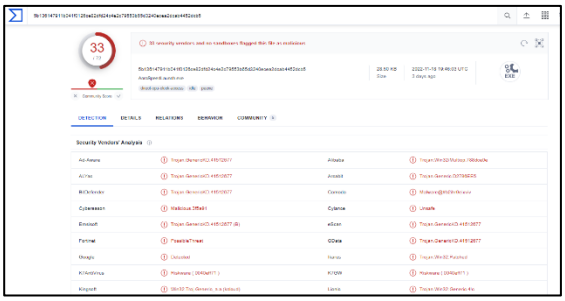


The above output shows us a list of sites with the Bank domain. Looking at the process memory has given us further grounds to be wary about ‘Reader sl.exe’. One may scroll down further to view a fully working web application with a card payment gateway.

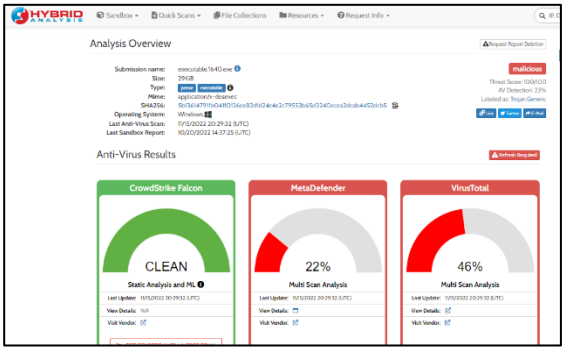
Now let's check to determine if the executable is malicious.

At this point, we have two options on how to proceed. First, either to perform a static analysis and reverse the executable to determine specifically what commands it uses and its purpose, or try a dynamic analysis by utilising a sandbox or online tools that evaluate possibly harmful executables as I'm not comfortable with reverse analysis at the moment.

But let's choose the second option and scan the files with VirusTotal and HybridAnalysis, arguably the most well-known website for suspicious file and website analysis.



Snapshot from VirusTotal showing the result of the PID 1640



Snapshot from HybridAnalysis showing the result of the PID 1640.

Clearly, these two sandboxing websites with strong detection scores have identified the .exe as dangerous. It's time to summarise the many inquiries we've performed and our conclusions using the studied dump:

- Reader sl.exe, PID 1640 which is a suspicious process of Explorer with a ParentPID 1484.
- An open connection of PID 1484 is to the IP address 41.168.5.140:8080
- 41.168.5.140 and bank domains were discovered in the process 1640 dump.
- By sandboxing websites, 1640 executables were identified as dangerous Trojans.

If a user's computer produced these IOCs then we may infer that the machine is trojan-infected. We should stop sending any communication to the specified IPs, note the IPs from which the traffic originated, and isolate those systems. The trojan should then be quarantined and removed from the compromised machine.

V. CONCLUSION

It can be concluded that the importance of malware analysis and detection is very important in today's situation. Without them the malware can pose serious threats to the data and information systems of many organizations. The malware that has the power to spread is the most harmful since there is no centralized control, making it difficult to guard against them. According to studies, one of the biggest threats to computer security is malware. Malware authors frequently come up with innovative concepts. They create malware in such a manner that it alters itself periodically in order to avoid being detected.

Malware authors always strive to create programs that are difficult to detect, and with time, they have effectively improved the tactics they employ to conceal or morph the dangerous code. These concepts begin with straightforward encryption before moving on to oligomorphic, polymorphic, and metamorphic viruses. Antivirus scanners are one method of detecting some of the malware programs, however with advancements in malware creation techniques, malware detectors utilize a variety of strategies to mitigate the negative impacts of this software. Due to the shortcomings of the currently available malware detection methods, machine learning and data mining techniques are coupled with existing detection methods to increase the efficiency of the detection process. In spite of their effectiveness in detecting known malware, signature-based detection techniques are unable to identify polymorphic and unknown malware since these threats can alter their signatures. Because new malware signatures have not yet been generated, signature-based detection cannot detect them. Heuristic-based detection techniques may find new, well-known, and undiscovered viruses, but they have a high percentage of false positives and false negatives, which motivates us to create more precise detection techniques. Heuristic-based detection approaches are paired with machine learning methods to boost malware detection efficiency and accuracy as a result of the exponential growth of polymorphic malware.

ACKNOWLEDGMENT

I would like to thank my school and my professors for all the support they have provided me all throughout my journey with GVSU. Also, I that my family members and friends for having trust on me and supported for pursuing my dreams.

REFERENCES

[1] A. Qamar, A. Karim, and V. Chang. Mobile malware attacks: Review, taxonomy & future directions. Future Generation Computer Systems, 97, pp.887-909. 2019.

[2] A. Al-Sabaawi, K. Al-Dulaimi, E. Foo and M. Alazab. Addressing malware attacks on connected and autonomous vehicles: recent techniques and challenges. In Malware Analysis Using Artificial Intelligence and Deep Learning (pp. 97-119). Springer, Cham. 2021.

[3] L. Chen, S. Hou, Y. Ye and S. Xu. Droideye: Fortifying security of learning-based classifier against adversarial android malware attacks. In 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 782-789). IEEE. 2018, August.

[4] A. Lanz, D. Rogers, and T.L. Alford. An epidemic model of malware virus with quarantine. Journal of Advances in Mathematics and Computer Science, 33(4), pp.1-10. 2019.

[5] O. Or-Meir, N. Nissim, Y. Elovici and L. Rokach. Dynamic malware analysis in the modern era—A state of the art survey. ACM Computing Surveys (CSUR), 52(5), pp.1-48. 2019.

[6] R. Sihwail, K. Omar and K.Z. Ariffin. A survey on malware analysis techniques: Static, dynamic, hybrid and memory analysis. Int. J. Adv. Sci. Eng. Inf. Technol, 8(4-2), pp.1662-1671. 2018.

[7] J. Singh and J. Singh. Challenge of malware analysis: malware obfuscation techniques. International Journal of Information Security Science, 7(3), pp.100-110. 2018.

[8] A. Shalaginov, S. Banin, A. Dehghantanha and K. Franke. Machine learning aided static malware analysis: A survey and tutorial. Cyber threat intelligence, pp.7-45. 2018.

[9] E.J. Alqahtani, R. Zagrouba and A. Almuhaideb. A Survey on Android Malware Detection Techniques Using Machine Learning Algorithms. In 2019 Sixth International Conference on Software Defined Systems (SDS) (pp. 110-117). IEEE. 2019 June.

[10] H. El Merabet and A. Hajraoui. A survey of malware detection techniques based on machine learning. International Journal of Advanced Computer Science and Applications, 10(1). 2019.

[11] R. Tahir. A study on malware and malware detection techniques. International Journal of

Education and Management Engineering, 8(2), p.20. 2018.
[12] J. Kang and Y. Won. A study on variant malware detection techniques using static and dynamic features. Journal of Information Processing Systems, 16(4), pp.882-895. 2020.
[13] A. Abusitta, M.Q. Li and B.C. Fung. Malware classification and composition analysis: A survey of recent developments. Journal of Information Security and Applications, 59, p.102828. 2021.
[14] W.W. Lo, X. Yang and Y. Wang. An xception convolutional neural network for malware classification with transfer learning. In 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS) (pp. 1-5). IEEE. 2019, June.
[15] Ö. Aslan and A.A. Yilmaz. A new malware classification framework based on deep learning algorithms. Ieee Access, 9, pp.87936-87951. 2021.
[16] B. Cakir and E. Dogdu. Malware classification using deep learning methods. In Proceedings of the ACMSE 2018 Conference (pp. 1-5). 2018, March.