Assessing Cost-Conscious Antivirus Amidst Malware Threats

Final Research Report submitted for EGH400-2 Semester 2 2024

Student Name: Max BLACKSTONE

Supervisor: Dr. Gowri Ramachandran

Submission Date: 29 May 2024

Abstract

Due to the increasing reliance on the internet, having effective antivirus is a necessity. With the current cost-of-living crisis, many seek out free versions of these programs; as such, the effectiveness of these free versions is called into question. There is a distinct lack of current research detailing the ability of free antivirus programs to detect current malicious attacks, therefore proving the necessity of this work.

To survey the ability of free antivirus to detect malware, a cyclical methodology was used: this involved developing faux malware, analysing with VirusTotal, testing on a virtual machine (VM) and recording the data for further analysis. Due to the volatile nature of malware irrespective of the source, testing was conducted on an isolated VM, with no personal or sensitive information available on it. This created a safe environment. Analysis of the repository/source of malware before use by project leader and supervisor also helped restrict negative consequences, and all developed malicious software was only created for research/educational purposes.

The results from testing displayed an overall lack of effectiveness with all antivirus software used (being Microsoft Defender, AVG, McAfee, Avast, and Kaspersky). Upon initial testing, detection rate of malware was 0/5, 1/5, 3/5, and 0/5 for the Ransomware PoC, File Stealer, Keylogger and File Deletion Tool (FDT) respectively. These results decreased to 0/5 across the board when implementing basic code alterations to the malware (such as renaming the final executable and classes). As such, these free versions of antivirus demonstrated a solely static approach to detection, leading to easy detection avoidance.

The inability for free antivirus to detect malware is an indictment of current antivirus companies, affecting those unable to afford the luxury of safe internet usage despite a societal necessity to do so. This report should be used to demand updated cybersecurity methods for those people, as everyone deserves safety as a basic right. In future, exploration into memory forensics on an individual scale or development of monitoring tools that use methods more lucrative than signature-based searches is paramount. The research within this paper provides the basis of future work based on the societal and economic needs of the world today.

Table of Contents

1 Introduction

The world has entered an era in which cybercrimes are constantly evolving, prompting the necessity of consistently developing tools effective at recognising the behaviours exhibited by malicious software. However, this prompts interest into surveying the success of current antivirus programs against modern attack methods. To do so will involve research into recent literature, as papers dated pre-2020 will be invalid in providing an accurate assessment of modern malicious programs and malware detection. Society is reliant on the internet as the basis for most of their everyday tasks, including work, banking, bills, and more. Therefore, safety online is equally paramount to physical safety in today's climate. With the cost-of-living crisis Australia currently is experiencing, it is also important to consider the economic viability of this research; paid antivirus is not accessible to everyone. This was also deemed societally important, as a lack of money to spend on antivirus should not withhold suitable protection online for the public. As such, testing with free antivirus programs would build the basis of a grander analysis of antivirus as a whole, and provide insight into the safety of the public. The desired outcome of this project was to finalise a critical analysis of free antivirus performance whilst providing potential new avenues to be explored to increase their effectiveness. Therefore, the following research question has been developed:

RQ: How effective are free, commercially available antivirus software at protecting data from current malicious cyber-attacks?

Therefore, the aim of this project was to complete a survey and analysis of free antivirus software performance, supported by the following objectives:

- Creation of malware attacks typically used in a modern context for testing
- Assessment of the detection rate provided by free, commercially available antivirus software
- Analysis of potential reasoning for the results
- Suggestions for future implementations of the data gathered for this project

2 Literature Review

2.1 Modern cyber threats

Through perusal of several relevant research papers, various tactics used by cybercriminals have been outlined, laying the groundwork for the necessary understanding required to develop effective countermeasures (Cyble, 2023; You & Yim, 2010; Jensen, Schwenk, Gruschka & Iacono, 2009). Such tactics include misspelled packages, cloud malware injection, code obfuscation, and many more. In the Python installation process, it is a common mistake to misspell the packages (Cyble, 2023); hackers prey upon this vulnerability and the ever-growing number of packages available on the internet. These malicious packages use techniques like code obfuscation and malicious downloaders to run the malware without being noticed by the victim. Cloud malware injection takes place by ensuring a Cloud system recognises a malicious virtual machine as legitimate and redirecting requests from the user to this instance, thereby running the malicious code. A current effective countermeasure includes the usage of a hash value for comparison, allowing easier identification

between valid versus malicious instances (Jensen, Schwenk, Gruschka, & Iacono, 2009). Finally, code obfuscation consists of several techniques, such as 'register reassignment' (where the registers are switched generation by generation despite outputting the same behaviour) and 'instruction substitution', where a seemingly identical yet malicious instruction replaces the original (You & Yim, 2010). These are just a few of the malicious attacks faced everyday by internet users.

2.2 Detection Methods

The presence of cyber threats has led to the creation of a variety of potentially effective detection methods, including signature-based, behaviour-based, heuristic, model checking, deep learning, cloud-based, mobile-based, and IoT-based (Aslan & Samet, 2020). Signature-based methods use a static approach to detection, doing comparisons against pre-determined patterns to identify malicious programs. These patterns can be identified through strings, recurring features, byte sequences, etc. (Caviglione, et al., 2021). Behaviour based methods employ a more dynamic approach are able to achieve this detection by recognising malicious behaviour as a threat, even against a clean template (Aslan & Samet, 2020). Heuristic analysis is essentially the combined sum of signature and behaviour-based approaches, with the additional input of machine learning. It claims to out-perform common antivirus programs such as Norton and McAfee yet can also be evaded using more complex viruses (Arnold & Tesauro, 2000). The primary issue with this research is the age of the paper; it is unclear whether the data would remain relevant given the 24-year gap in literature. A more feasible current approach being explored is the implementation of deep learning-based detections, using example images of malicious attacks to consistently learn and effectively target unwanted code (Aslan & Samet, 2020). However, reasonably evasive techniques employed by cyber-attacks allow recognition avoidance, providing a known path for hackers to exploit (M. & Sethuraman, A comprehensive survey on deep learning based malware detection techniques, 2023). Out of all discussed methods, the majority are not fully developed nor implemented on a commercially available scale, with the exception of the signature and behaviour-based detections. Primarily, most still use the static approach provided by signature-based recognition (Aslan & Samet, 2020), which is an effective and quick method. However, new malware (which is appearing constantly) can easily evade detection due to there being no established patterns for comparison. This is worsened by the easily accessible knowledge of which patterns are identified, allowing hackers to evolve malware to avoid known signatures. Therefore, behavioural detection is the most promising model, and for effective antivirus, should be the standard method employed.

2.3 Behaviour-based approach

There are a lot of promising applications as result of a focus on behavioural detection methods for malware (Maniriho, Mahmood, & Chowdury, 2022; Chew & Kumar, 2019). As earlier mentioned, behavioural detections feature a dynamic approach to analyse malware, which can be portioned into two primary tactics. These tactics involve analysing the difference between a fixed start to end point, and implementation of monitoring tools to track program behaviour during run-time (Mira, 2019). Using a fixed start and end point allows for easy observation of any unwanted changes occurring during the specified timeframe, displaying malicious activity effectively. Monitoring tools

are especially capable with API calls, yet both allow for comprehension of several malware indicators such as file entropy, file changes, and canary files (Chew & Kumar, 2019). Despite not necessarily indicating malicious behaviour, files with high entropy generally feature encryption, a common trait in malware. Alternatively, after purposeful encryption, files can append to unknown extensions; these extensions indicate that an unwanted program is running on the victim's computer. Canary files are files that appear useless, presenting with filler data, that attract manipulation via malware and therefore notify the user of the issue. Through research, (Chew & Kumar, 2019) found that these methods are subject to false positives (particularly file entropy), promoting the necessity of reading file's magic bytes. This furthers unknwon extension detections, which paired with the other indicators can increase effectiveness of indication the system was compromised. Current literature also displays a wave of new approaches to behavioural malware detection, such as MalDetConV, a behaviour-based framework created by (Maniriho, Mahmood, & Chowdhury, 2022). MalDetConV employs deep learning and neural networks to conduct API monitoring, emphasising the difference between malicious and benign programs whilst maintaining accuracy for both known and unknown malware. Additional further techniques being explored include both specification-based monitoring and anomaly detection (Chakravarty, Raj, Paul, & S., 2019). Combining behavioural and signature-based detection in an almost heuristic approach, specification-based monitoring writes policies that run a specified action in the case of events not following the required standards. An example of this is having a Whitelist that prevents malicious downloads from automatically occurring by only enabling downloads for websites on the list. Alternatively, anomaly detection is similar to the fixed start and end behaviour-based method, storing regular system activity as a reference for uncommon behaviour (Mira, 2019). If a program with no ties to a directory begins writing calls to it, this can be deemed as malicious. However, like the fixed start and end method, this is susceptible to both mimicry and false positives. Specification-based monitoring is considered advantageous over anomaly detection, featuring a lower false positive rate and increased resilience as observed by (Chakravarty, Raj, Paul, & S., 2019). This is made possible through simple policy adjustments that can be made to consistently improve performance. Despite the many options with a behaviour-based approach, there are also several limitations that need to be addressed. The likelihood of false positives is enhanced as there are difficulties identifying anomalous versus normal behaviour; thus, while its possible to detect unknown malware, several challenges arise in practice. Additionally, mimicry attacks are specifically developed to evade behavioural detection, essentially a form of trojan software in which malicious behaviour is obscured by a normal façade (Khan, Foley, & O'Sullivan, 2022). All these factors require consideration when exploring this facet of cybersecurity.

2.4 Monitoring tools and their uses in cybersecurity

Behavioural malware detection can be strengthened through the use of monitoring tools. There are many available via the internet, such as: Zeek, Google Rapid Response (GRR), and REMnux (Talukder & Talukder, 2020). Zeek is a tool that allows users to track network activity, providing a clear view into any anomalies in expected behaviour (also known as an Intrusion Detection System). GRR, on the other hand, is a Python client that takes memory input and reviews it to identify footprints left by malware, allowing for convenient remote analysis to take place. REMnux is a Linux toolkit, using

resources and examples to find vulnerabilities in the browser and the presence of ransomware, essentially reverse-engineering malware. These are all open-source tools based on achieving behavioural malware detection, with slight variations in the approach taken and the operating software (os) designed for use. Due to the popularity of Windows as the chosen os, malware is exponentially created (over 100 million in 2021) aimed at targeting Windows (Drapkin, 2021). Specifically for windows, a further three tools can be explored: Process Hacker, ProcMon (Process Monitor), and RegShot (Forum of Incident Response and Security Teams, Inc., 2023). Process Hacker is a freely available and powerful tool that allows for resource monitoring, software debugging, and malware detection (Winsider Seminars & Solutions, Inc., 2023). ProcMon, available through Microsoft itself, is a tool focused on file and registry access (Russinovich, 2023). It features a real-time file system, analysis of process/thread activity, and registry observations, and is overall relatively similar to Process Hacker. Finally, RegShot is an additional tool specifically analysing registry changes, employing comparison via snapshots. These are a handful of relevant tools, however many more are constantly being released and developed, providing an interesting path in which the cybersecurity field can explore.

2.5 Antivirus Software

Given the current reliance on the Internet to perform necessary everyday tasks, 76% of computers worldwide use antivirus software as protection against the threats that comes associated with a web presence (Aashind, 2023). Furthermore, a large number of people employ free versions of antivirus software, relying on it for the safety of sensitive data. Examples of which include Windows Defender, McAfee Free, AVG, Avast, and Kaspersky. Windows 10 comes with a built-in antivirus option known as Windows Defender, notably providing security against system takeovers (Delaney, 2020). The other four options of antivirus are commercially available programs that can be downloaded from the internet, using a simple user interface to display numerous options for cybersecurity and claims high detection rates on current malware. However, the commercial aspect of these programs can be a crux, with new forms of malware being created at a rate greater than vendors can upgrade protection (Kapersky IT Encyclopedia, 2023). Antivirus must also feature a positive balance between safety and performance, attempting minimal CPU usage while conducting in-depth scans of the entire system for all possible viruses. Unfortunately, due to the nature of the software, only one option can be used at a time. This is required to avoid kernel conflicts and unnecessary vulnerabilities, forcing users choose a single antivirus program to protect all personal information.

2.6 Gap

There is a small quantity of research discussing comparison between free and paid antivirus services, and an evaluation of which is best (Santos, 2021) (Frost & Månsson, 2014). However, these primarily do not aim for the same objective as this paper; neither are restricted to free programs, which given the current socioeconomic climate is an important factor. This is supported by the most similar paper being created in 2014, a vastly different landscape than that of today. As such, the data provided by it is outdated, and requires updated research based off changes in the last ten years. The free vs. paid comparative document is also a poster as opposed to a paper, therefore not being

as analytically involved as a full research paper, as this report intends to be. Therefore, it was identified that this is a significant gap in current literature, making the conducted research viable for use.

3 Methodology

The methodology of this project is fairly cyclical, comprising of two identical phases that build to provide a final overall understanding of the effectiveness of free antivirus programs. Each phase has two stages, the first of which being a quantitative analysis, followed by observations. The project began with a proposal, providing a detail overview of the project to be reviewed by the stakeholder. Due to the positive response, the first phase began:

The steps for Phase 1 are as follows:

1. Referencing code from GitHub, develop faux malware mimicking two cyber-attacks currently employed by hackers.
2. Conduct testing against 5 different forms of free, commercially available antivirus software (including built-in options such as Windows Defender).
3. Present a progress report to the stakeholder by October 25th 2023 displaying the results of Phase 1 and the implications relevant to Phase 2.

As implied in the steps, the faux malware developed will be scanned by the free antivirus software, thereby allowing a quantitative analysis to take place via the results collected from the testing. The observation stage was encapsulated by the progress report, providing a detailed analysis of both VirusTotal and actual results while addressing areas of concern. Phase 2 will then consist of furthering the critical analysis against free antivirus by essentially repeating the steps with additional forms of faux malware:

4. Referencing code from a variety of sources, create further malicious attacks to be tested against the same antivirus under the same conditions previously set.
5. Conduct further testing, including a more extensive view into reports from antivirus in the event of malware detection.
6. Prepare final report detailing the outcome of the success of current free antivirus against malicious code, then providing insight into social effect of the results and potential avenues for a positive outcome regarding the results.

Thus, the two stages are near identical, discovering quantitative data and conducting extensive analysis. This will create a clearer picture of the tools those less fortunate are forced to use to protect themselves and their computers, as well as the effectiveness of said tools against targeted attacks. This report will combine all quantitative data gathered throughout the project into a synthesized outcome to present the projects findings in full. Due to the cyclical nature of the methodology, a graphic has been produced to aid in understanding of the steps taken to gather data. Originally, the methodology was slightly different (the diagram for the original method can be found in the Appendix) however, the updated version is the one displayed below:

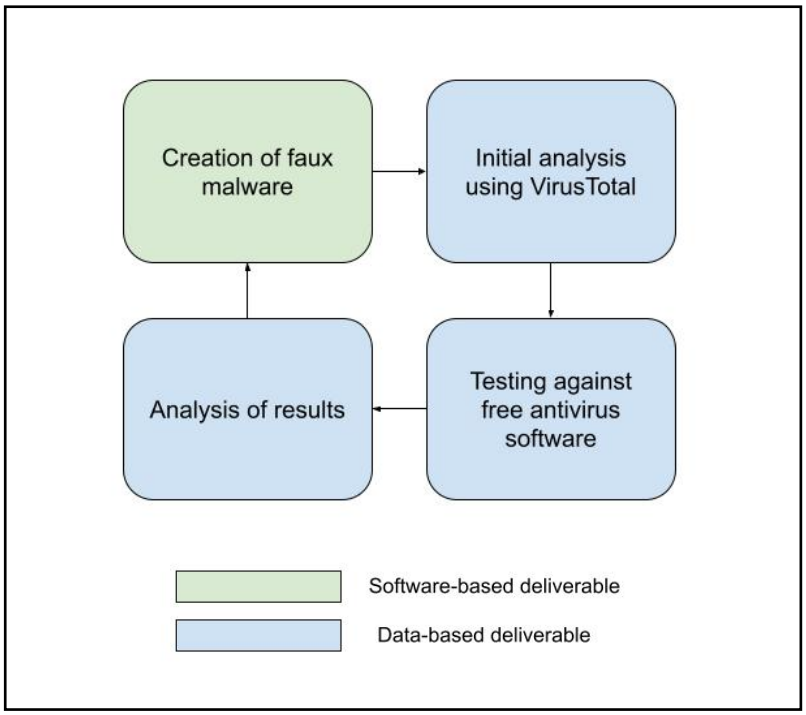

Figure 1. Updated methodology based off cyclical testing associated with new project scope

3.1 Assumptions

There were a number of limitations placed on this project, such as available time and implementation of malicious code on a QUT eResearch VM (which came with a number of security concerns). As such, assumptions had to be made to keep the project viable. This included the five antivirus programs being sufficient. The potential programs were limited to those involved in the VirusTotal analysis, further reduced based on availability and accessibility within Australia.

3.2 Stakeholders

The primary (sole) stakeholder in this project was supervisor Dr. Gowri Ramachandran. To establish consistent communication, weekly meetings were held and attended by both parties. This ensured comprehension of tasks and consistent feedback on updates and difficulties, supplemented by emails where necessary. Dr. Ramachandran's input was invaluable to the project, providing beneficial insights and greatly aiding the redirection of the project after difficulties with the initial scope. Additionally, having an open line to discuss risky objectives and review potential GitHub repositories made a significant impact on the safety of the project. Dr. Ramachandran also created the connection to Vishal Dogra, who supplied the ransomware code proof of concept, which was of great significance to Phase 1 of the project. The stakeholder also greatly contributed to the understanding of risks, as well as practice of ethics and sustainability over the course of all work conducted.

3.3 Risks, Ethics, Sustainability

The implementation of malware, even in a research application, comes with inherent risk. As such, utmost caution was required during testing. The primary defense against unwanted consequences was the use of a virtual machine when running/analysing code. The VM created an isolated environment in which no personal details or sensitive data could be accessed, greatly protecting the project lead. Additionally, the faux malware developed was limited to forms that affected the system specifically, not the network (such as a computer worm). Since the VM ran on the QUT network, incorrect implementation of a seemingly harmless program could compromise all data linked to the university's network. Since the malware was referenced from a number of GitHub repositories and blogs, each source was carefully inspected by the supervisor, Dr. Gowri Ramachandran, and the project leader to ensure the code was created for educational purposes and did not pose a risk during use.

Throughout this project, it has been important to abide by the Code of Ethics presented by Engineers Australia. The four pillars of this code are: demonstrate integrity, practice competently, exercise leadership, and promote sustainability (Engineers Australia, 2022). Each aspect of the ethics code has three primary guidelines, each of which will be explored.

Acting on the basis of a well-informed conscience, honesty and trustworthiness, and respecting people's dignity are the three guidelines necessary when it comes to demonstrating integrity. All actions taken were within the ability of the project lead and are believed to be objectively what's right for the current social climate. Legal, contractual, and employment obligations are important to consider, yet not relevant for this project due to the external companies not being one of the stakeholders for the project. There was a degree of honesty and trustworthiness in all communications between project lead and stakeholders, ensuring criticism was well received and a healthy level of understanding could be maintained. Credit was given where due to all involved parties (such as Vishal Dogra for supplying the Ransomware PoC), and as all malware development was purely for research purposes, no criminal or fraudulent conduct took place. To ensure respect for people's dignity, a safe and respectful environment was cultivated, ensuring a courteous approach to research regardless of age, gender, or any other factors. Therefore, integrity was an aspect of the code dutifully considered.

Competent practice was a necessary aspect of this project, enhanced by the risks associated with the implementation of volatile malicious code. There was a continual development of the subject and relevant knowledge throughout the course of the project, and no attempt was made to work on anything outside the ability of the project lead. This was made aware to the stakeholders through transparency on qualifications and is demonstrated through the numerous reports submitted (proposal, progress report, final paper). Additionally, all work was conducted in accordance to accept engineering standards, thus embodying this particular aspect of the Code of Ethics.

The third principal outlined by the Code of Ethics, practicing leadership, was determined the least relevant in the case of this project, due to it being an individual task. However, there are still elements demonstrated that can be discussed. There was clear and effective communication

between the project lead and stakeholders, emphasised by the consistent meeting minutes as a result of upholding weekly progress updates. Additionally, it was considered important to uphold the reputation of the engineering practice, thereby ensuring the project could be completed in an appropriate and justified manner. Diversity and providing opportunities was rarely necessary, however did occur with the case of Vishal Dogra's contribution to the project, which had a significant impact on the results.

In terms of sustainability, there are three main areas to be considered: environment, social and economic. Environmental/physical sustainability was not considered overly influential in the project due to the project being software-based. Using pre-existing hardware (computers) to work on software causes no further environmental degradation as a result of the project. Overall, it can be confirmed that while the project has no real environmental benefits, there are no environmental-based negatives that can affect any of the three facets of sustainability.

The project should contribute positively to the social aspect of sustainability, with no negative impacts affecting the general population. The only people negatively affected by testing and development of behaviour-based antivirus software are cybercriminals.

Economically, the project is greatly sustainable in the form being developed for overall submission; it solely uses free development tools (VSCode) and antivirus programs. Therefore, benefits of this project (a safer environment during computer use) outweighs the economic cost.

4 Results & Analysis

4.1 Description of new Malware

The first malware used was the Dummy Ransomware PoC, developed by fellow QUT student Vishal Dogra. The program features a trojan approach, emulating a legitimate application whilst maliciously affecting the victim's computer. Upon receiving the necessary command, the program finds the specified files and encrypts them, required monetary payment in order to unlock them. In this case, the default folder targeted is the Documents; files within this folder are encrypted, with encryption data sent to a C2 server. Upon doing so, the exploit is conducted, resulting in all files being completely unreadable until the unlock command is provided.

The second piece of faux malware is the Python File Stealer, a program designed to steal personal data from the victim's computer. Infromation such as emails and bank account passwords can be acquired, providing great monetary benefit to the hacker. Referenced from a user called 'Antsbatscats' on GitHub, the original code sourced from the repository featured much of the same code, but with the implementation of a Discord Webhook for transferral of data. This feature was removed for testing due to the inherent risk accompanying such usage, providing slight alteration to the original code. However, many of the other features were retained, such as blacking out the screen and locking the user out, as well as recursively searching throughout the entire drive and all directories for files to steal. To replace the Webhook, a print statement was employed to detail the file path and success of the malware. As such, the malicious nature of the program was retained, whilst removing the unnecessary risks associated with direct implementation of the original code.

The two new forms of malware created were a Keylogger and File Deletion Tool (FDT). The keylogger is a type of surveillance malware (also known as spyware), that will track the keystrokes input to the keyboard. Upon doing so, it can be setup for two different forms of output, the first of which being the primary use case for testing. This involves simply saving all the keystrokes over the set timeframe to a file pre-emptively named and dated to the folder of choice. The second possibility is integration of email modules to send these same files to a chosen email address rather than allowing it to remain on the victim's computer. The result of running this keylogger can be devastating to the victim; any sensitive data entered (bank account access, email passwords, etc.) will be recorded and formatted in a manner that is easily readable for the user, thereby gaining them access to any protected areas accessed by the victim. The code for this malware was primarily referenced from an ethical hacking webpage created by Abdeladim Fadheli (Fadheli, 2023).

The second malware is a File Deletion Tool, a malware that is simply destructive to the victim; using the *os* library, the malware is directed to a folder of choice. Upon reaching it, it will analyse all files of the chosen filetype within the folder/subfolders and permanently remove the files from the computer. Rather than recreating another ransomware (for example, saving a copy of the files to blackmail the victim into payment), it was decided to use destructive malware. Destructive malware is a serious threat, especially to heavily data-based operations, as it could ruin a variety of important files such as evidence against cybercriminals or single copy passcode files. As such, trying these additional forms of malware provides a greater overall introspective into how antivirus handles different attacks that are a common possibility.

4.2 VirusTotal Results

Initial analysis via the website VirusTotal was used to predict the results that would occur in practice. The results for the Keylogger and File Deletion Tool are available below, and the results for the Ransomware PoC and File Stealer are located within the Appendix.

Table 1. VirusTotal Analysis for the two new forms of malware

	Keylogger	**File Deleter**
Threat Label	Trojan	Trojan
Effective Antivirus Software	ALYac – Gen:Variant.Mikey.165617 Avast – Multi:KeyLogger-I [Trj] Crowdstrike Falcon – Win/malicious_confidence_70% (W) DeepInstinct – MALICIOUS ESET-NOD32 – Python/Spy.KeyLogger.TZ Malwarebytes – Spyware.KeyLogger.Python.Generic SecureAge – Malicious	ALYac – Gen:Variant.Mikey.165617 Avast – Win64:PWSX-gen [Trj] Bkav Pro – W64.AIDetectMalware SecureAge – Malicious Zillya – Trojan.Disco.Win32.11744 Antiy-AVL – Trojan/Python.Kryptik AVG – Win64:PWSX-gen [Trj] Malwarebytes – Malware.AI.3934082399

	ZoneAlarm by Checkpoint – HEUR:Trojan-Spy.Python.KeyLogger.ae AVG – Multi:KeyLogger-I [Trj] Bkav Pro - W64.AIDetectMalware Elastic – Malicious (moderate Confidence) Fortinet – Python/KeyLogger.UJ!tr Kaspersky - HEUR:Trojan-Spy.Python.KeyLogger.ae SentinelOne – Static AI – Suspicious PE Zillya – Trojan.Disco.Win32.11744 Skyhigh (SWG) - BehavesLike.Win64.FlyAgent.wc Antiy-AVL - Trojan/Python.Kryptik	Skyhigh (SWG) – BehavesLike.Win64.FlyAgent.vc
Entropy	7.997619	7.997502
Similar hashes	CAPA: a3e8b24667cc9b2a6dcf4f78937ea8ce CAPE Sandbox: 7fdbb8069c1a7eae15fcc734a176ceb6 Microsoft Sysinternals: 2191d57db626ba8ecac11a0bb8622574	CAPA: a3e8b24667cc9b2a6dcf4f78937ea8ce CAPE Sandbox: 925c63c772a0ac5610ae09966d892199 Microsoft Sysinternals: b37a1d7cec280c51121a91233084ff22

Regarding the new software, it is important to note that the keylogger (a significantly more known and obvious malware) had a much stronger detection percentage on VirusTotal as opposed to the custom created File Deletion Tool (FDT); 23.94% compared to 13.43%. Of the relevant antivirus software being tested, the FDT should only be detected by AVG and Avast, with the Keylogger found by both of those and Kaspersky. Therefore, Microsoft Defender and McAfee should not be able to detect either of the malware, and Kaspersky should not recognise the FDT. Both viruses were classed as 'trojan' threats, implying they provide a legitimate disguise whilst acting in a malicious manner. No code obfuscation or concealing attempts have been made; however, the argument could be made that the keylogger does not openly seem dangerous, until the implications are understood (causing the specific Keylogger calls that some forms of antivirus make). All of the faux malware had extremely high entropy (with values greater than 7.99). While this doesn't prove the files to be malicious, entropy is increased with obfuscated code and data, making it a feasible point of identification when detecting suspicious files. It was interesting both files had the same behaviour similarity hash from CAPA, indicating they behave or are structured in a similar manner. However, the only similarity that should occur is the malicious nature, which calls back to the original identification of both being trojan threats. With this information, testing could be conducted with reasonable expectations for results provided.

4.3 Antivirus Software Detection Results:

Following the same standard as the progress report, the antivirus software choices remained the same across both iterations of testing. The importance of only using freely available antivirus must be reinforced, due to the social impacts to be explored within the Discussion of this report.

Table 2. Full Results for all malware tested and all antivirus

Antivirus Software	Ransomware PoC	Python File Stealer	Keylogger	File Deletion Tool
Microsoft Defender	Evaded	Evaded	Evaded	Evaded
Avast Antivirus	Evaded	Evaded	Detected	Evaded
McAfee Antivirus (Free Version)	Evaded	Evaded	Evaded	Evaded
AVG	Evaded	Detected	Detected	Evaded
Kaspersky	Evaded	Evaded	Detected	Evaded

Testing displayed an overall lack of detections on the faux malware. The ransomware PoC completely evaded all antivirus programs, as did the FDT. This is despite no attempt at concealment of malicious code/behaviour. The File Stealer was only detected by AVG, whereas the Keylogger was detected by Kaspersky, AVG, and Avast. The singular detection by AVG for the File Stealer was interesting as VirusTotal results from the Keylogger indicated that AVG and Avast use the same engine for malware detection. However, the File Stealer results refute this. Overall, these results provide a reasonable answer to the research question, indicating that current free antivirus programs are ineffective at dealing with modern cyber-attacks.

To further enhance understanding, a visual depiction of the results has been created. There are a few missing columns, indicating zero detections for the given form of analysis specified in the legend.

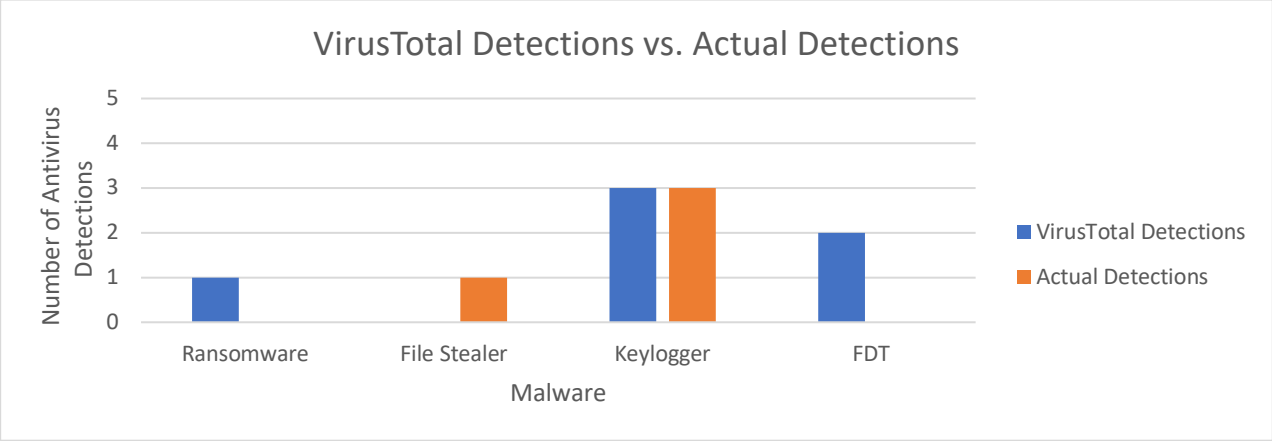

Figure 2. Comparison between malware detection of VirusTotal Analysis compared to Practical Experimentation (missing bars in graph refer to zero detections for the malware)

Evidently, testing primarily did not match the expectations set by VirusTotal, with the exception of the Keylogger. In general, VirusTotal predicted a higher detection rate than the results proved, depicting a discrepancy between methods believed to be used and methods actually used.

4.4 Validation of Results

The validity of the results is contingent upon the assumptions made previously in the report, as well as courses-of-action required due to the inherent risks during testing. The antivirus's response to threats were analysed on a system level; due to the use of the QUT network, nothing that utilised it nor required transferral of the malware via internet were used. As such, testing was conducted either solely on the computer or via a USB transfer to determine if the antivirus would recognise it. It was considered to use email implementation to see if the malware would be detected during download, but this was deemed unsafe by the project leader and supervisor and therefore not explored. Despite this, there is confidence in the validity of the software being malicious, due to the two tested entry points and the focus of scope not extending to network connection.

5 Discussion

5.1 Results Analysis

The results for all four forms of commonly used malware depict a scathing indictment on the effectiveness of current freely available antivirus software. The file deleting tool did not feature detections by any of the five popular antivirus packages, despite its targeting of critical folders (and their subfolders) and use of the operating system library to destroy specified files. Whilst not being obfuscated nor any attempt made to shield its malicious actions, antivirus still did not consider it a threat, with no quarantines nor removals made. This was unexpected due to VirusTotal's analysis expecting both AVG and Avast to detect the malware. However, it is evident in practice that the antivirus does not always behave as expected. Adversely, the keylogger had a significantly higher detection rate than any faux malware tested, with 3/5 forms of antivirus successfully detecting and removing the file immediately. This was appropriate based off the VirusTotal data, which identified all three of these forms of antivirus to have successful recognition. For these practical detections, copies of the resulting information were taken, the most informative of which being the Kaspersky

log. Kaspersky's reaction to the malware was displayed below, with the additional antivirus detections placed in the Appendix for further context:

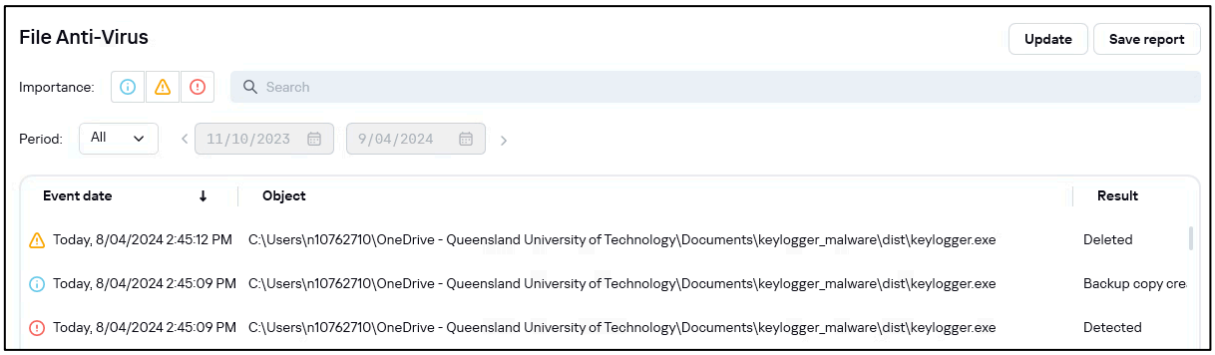

Figure 3. Timeline of detection to removal by Kaspersky for Keylogger malware

Kaspersky - Keylogger Removal Log

Event: Malicious object detected
User: QUTAD\n10762710
User type: Initiator
Application name: explorer.exe
Application path: C:\Windows
Component: File Anti-Virus
Result description: Detected
Type: Trojan
Name: HEUR:Trojan-Spy.Python.Keylogger.ae
Precision: Heuristic Analysis
Threat level: High
Object type: File
Object name: keylogger.exe
Object path: C:\Users\n10762710\OneDrive - Queensland University of Technology\Documents\keylogger_malware\dist
MD5 of an object: E4458942C17F3925C22D65BC19525D63
Reason: Expert analysis
Databases release date: Today, 8/04/2024 12:31:00 PM

Figure 4. Log of threat detection by Kaspersky on Keylogger malware

As Figures 3 and 4 display, Kaspersky successfully used an apparent Heuristic analysis to identify the high threat level of the program. It was even able to successfully detect it as specifically a Python Keylogger, which triggered an automatic quarantine and deletion of the file upon the scan. It also claims the type of be Trojan; this was unexpected, as there was no attempt to obfuscate or hide the code, yet given the successful detection this is not deemed as an issue.

The other successful detections, both AVG and Avast, unfortunately provided a less detailed log of threat detection. Whilst also identifying the program as a Keylogger and quarantining the file, information such as that provided in Figure 4 were not included, and requests for further information from the antivirus were futile. VirusTotal indicated the use of the same engine for

detections, which is further complemented by the almost identical messages displayed, visible in Figure 5.

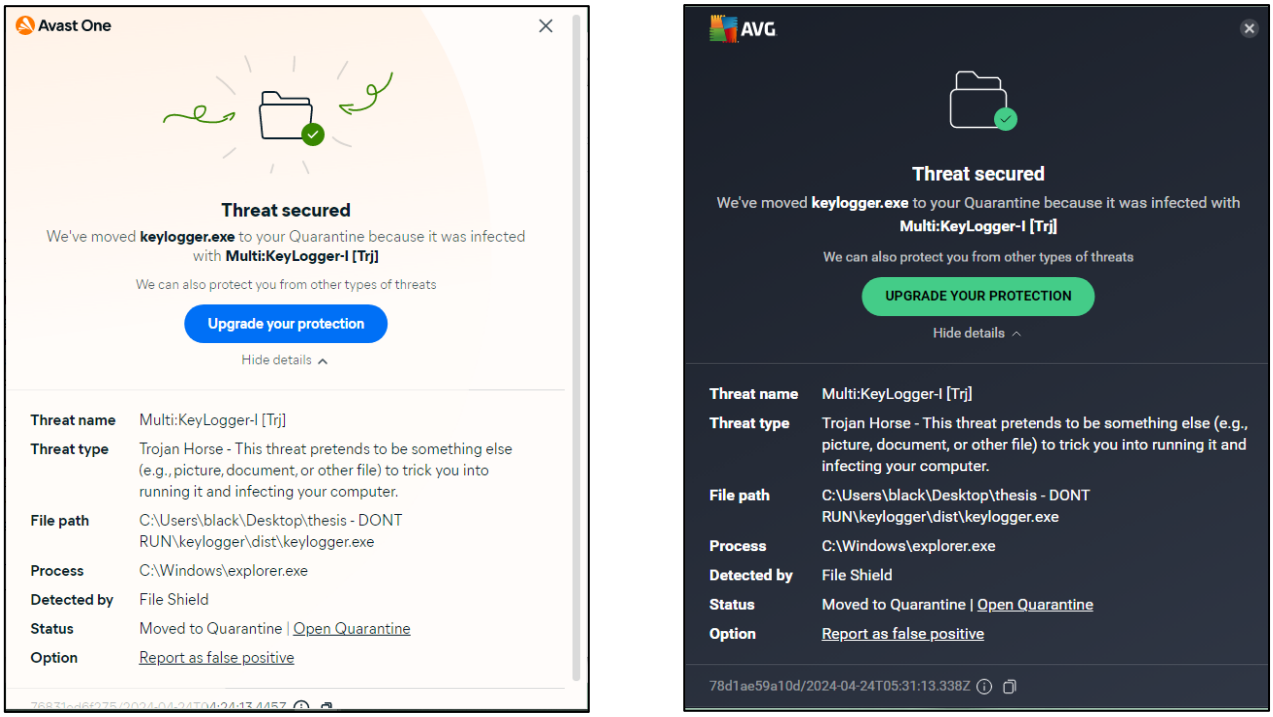

Figure 5. Detection messages displayed by Avast and AVG after custom scan of the folder containing the keylogger virus

5.2 Effects of code alteration

Due to the general inability for the antivirus to recognise malware, it was deemed necessary to conduct further testing on what allows malware to evade detection. Using the keylogger (the most identified malware with a rate of 60% detection), minimal changes were tested upon the hopes of proving a simplistic bypass of the software. It was discovered through trial-and-error that there were only two changes required to bypass detection by both AVG and Avast; the file had to be compiled under an alternate name, and the class (formerly named ‘Keylogger’) named differently. Despite no change to the functionality of the code, nor changes to the comments describing the function of the code (mentioning it’s a keylogger), this was sufficient to have these programs no longer class the malware as malicious. Based off this finding, the File Stealer was also retested against AVG (the only successful antivirus against it) with the same small change of renaming the file and compiling under the new name. Upon doing so, the file was scanned by AVG and determined to be ‘safe’, despite no functionality changes. The result of all testing with the necessary changes can be surmised by the following figure depicting AVG’s message after running custom scans on the altered code:

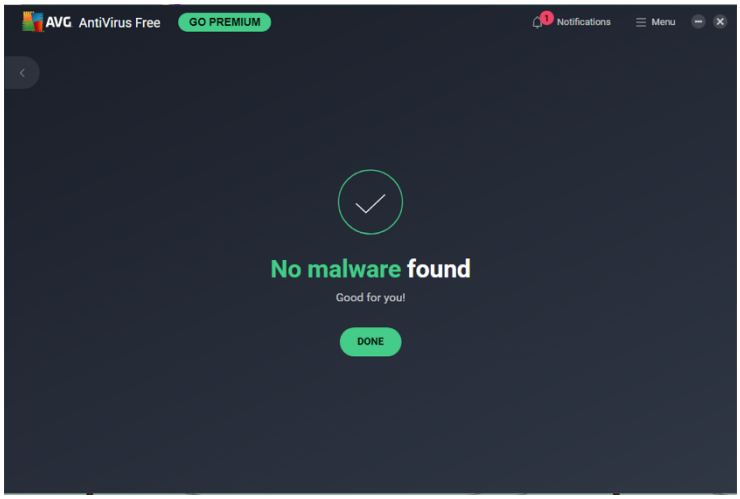

Figure 6. AVG response to altered code

As such, it can be identified that these programs are only conducting static tests based off certain keywords within either the class or file names. This provides an explanation for the lack of effectiveness indicated during this project.

5.3 Potential implementation of eBPF

In the original proposal and progress report for this project, it was expected that there would be a focus on the implementation of eBPF into antivirus software; finding malicious behaviour within the kernels of the computer. However, the novelty of this idea was its use on Windows, a newer version of eBPF differing to its Linux origins. However, eBPF for Windows (Microsoft, 2024) proved to be ultimately too tricky to work with, encountering numerous errors within the setup and starting processes that led to the conclusion the time would be better spent furthering the analysis of malware detection. Hence, an adjustment was made to the project scope, instead refocusing on extended testing to gain a more critical depiction of their performance and the areas in which they struggle.

5.4 The possibilities of volatility3

An interesting possibility to be explored is the wider implementation of memory forensics on a household scale. Specifically, some research and experimentation were conducted using volatility3 (The Volatility Foundation, 2024), a framework for memory extraction. It features numerous commands that, given a memory dump input, can fully analyse all processes called, applications run, and more. This would be especially helpful with consistent input of memory, allowing malicious software currently running to be easily spotted at any time. Unfortunately, volatility3 was deemed too convoluted and difficult to be focused on within the short timeframe of this project, however it is a potential path that could be used for future research built off this paper.

5.5 Societal impact of results in current cost-of-living crisis and its effect on the future

This project was limited to free antivirus software, given that it was intended to be completed with no expense, and provide an insight into the safety of people who cannot afford the high cost of a paid service. Testing repeatedly demonstrated an inability to provide suitable coverage for users, even being unable to recognise malicious programs with simple misdirection such as renamed

classes and filenames. Due to the inflated cost of living in the current societal climate, it is all too realistic that people must cut back on certain luxuries. With the steep price of annually renewing antivirus software, it is feasible that antivirus (in which a free, effective alternative is advertised) is an area people would choose to save their money. These costs have been compiled for the antivirus tested in Table 3 below:

Table 3. Pricing costs of tested Antivirus Software

	Microsoft Defender – included with Microsoft 365 (Microsoft, 2024)	**Avast (Avast, 2024)**	**McAfee (McAfee, 2024)**	**AVG (AVG, 2024)**	**Kaspersky (Kaspersky, 2024)**
Price (annual) - Standard	$132	$99.99	$159.95	$119.99	$88.95
Price (annual) - Premium	$132	$229.99	$279.95	$169.99	$113.95

Whilst some of these may seem cheap considering it's an annual cost, those living on a week-to-week basis will struggle to be able to afford the cost. This makes it viable for the user to turn to the advertised free versions rather than paying generally upwards of $100 for the product.

However, the results prove the negative consequences of this choice; software that struggles to recognise the most basic malware, worsened by simple code changes due to a seemingly static approach to code detection. Additionally, inability to afford antivirus also suggest inability to pay for services such as streaming sites (e.g. Apple TV+, Stan) meaning the only option is pirating movies. This provides a far too risky environment in which those unable to afford luxury services are subjected to using vulnerable websites with ineffective antivirus software.

It is in the best interests of the project to also consider how well the paid services perform in comparison. It can be assumed that a paid, full antivirus would provide much better protection than the free alternative. However, it is feasible that the paid service also largely underperforms, meaning cybersecurity for the public is insufficient irrespective of their financial situation. Learning this would provide valuable data to this project, potentially supporting the need of cheaper/free appropriate antivirus or necessitating a full revolution of commercial antivirus to reduce cybercrime against the population.

6 Conclusion

The key findings of this report demonstrate the ineffectiveness of freely available commercial antivirus software. As evidenced, the current detection rate of malware in free antivirus is extremely low, only worsening with the slightest of cosmetic changes to the code. As such, enforcing better, more dynamic antivirus would be a great societal benefit, allowing people to use their computer necessitated by the current climate in a safe manner. Further work into both the analysis of paid antivirus and exploration of volatility3 on a grander scale would be of huge benefit, expanding upon the basis laid by this project. In conclusion, the revolution of commercial free antivirus must advance with the never-ending onslaught of new attacks, and usher the world into a new age of cyber safety.

7 Bibliography

Aashind, A. (2023, March 30). *Antivirus Usage Statistics [2023]*. Retrieved from CyberCrew: https://cybercrew.uk/blog/antivirus-usage-statistics/

Arnold, W., & Tesauro, G. (2000). *Automatically generated Win32 heuristic virus detection.* Virus Bulletin.

Aslan, Ö., & Samet, R. (2020). *A Comprehensive Review on Malware Detection Approaches.* IEEE.

Avast. (2024). *Store*. Retrieved from Avast: https://www.avast.com/en-au/store#windows

AVG. (2024). *Discover all AVG products*. Retrieved from AVG: https://www.avg.com/en-au/store#pc

Caviglione, L., Choraś, M., Corona, I., Janicki, A., Mazurczyk, W., Pawlicki, M., & Wasielewska, K. (2021). *Tight Arms Race: Overview of Current Malware Threats and Trends in Their Detection.* IEEE.

Chakravarty, A. K., Raj, A., Paul, S., & S., A. (2019). *A study of signature-based and behaviour-based malware detection approaches.* International Journal of Advance Research, Ideas and Innovations in Technology.

Chew, C. J., & Kumar, V. (2019). Behaviour Based Ransomware Detection. *EPiC Series in Computing*.

Cyble. (2023, June 9). *Misspelled Packages Preying on Unwary Victims*. Retrieved from Cyble: https://cyble.com/blog/over-45-thousand-users-fell-victim-to-malicious-pypi-packages/

Delaney, J. (2020). *The Effectiveness of Antivirus Software.* Utica College ProQuest Dissertations Publishing.

Drapkin, A. (2021, November 26). *Over 100 Million Pieces of Malware Were Made for Windows Users in 2021*. Retrieved from Tech.Co: https://tech.co/news/windows-users-malware

Engineers Australia. (2022, August). *Engineers Australia | Code of Ethics and Guidelines on Professional Conduct1 Code of Ethics and Guidelines on Professional Conduct.* Retrieved from Engineers Australia: https://www.engineersaustralia.org.au/sites/default/files/2022-08/code-ethics-guidelines-professional-conduct-2022.pdf

Fadheli, A. (2023, May). *How to Make a Keylogger in Python.* Retrieved from The Python Code: https://thepythoncode.com/article/write-a-keylogger-python

Forum of Incident Response and Security Teams, Inc. (2023). *Behavioral Analysis*. Retrieved from FIRST: https://www.first.org/global/sigs/malware/ma-framework/behavioralanalysis

Frost, C., & Månsson, P. (2014). *Anti-Virus Programs Evaluation.* Retrieved from Digitala Vetenskapliga Arkivet: https://www.diva-portal.org/smash/get/diva2:696092/FULLTEXT02.pdf

Jensen, M., Schwenk, J., Gruschka, N., & Iacono, L. L. (2009). *On Technical Security Issues in Cloud Computing.* IEEE.

Kapersky IT Encyclopedia. (2023). *Antivirus programs: their quality and issues*. Retrieved from Kaspersky IT Encyclopedia: https://encyclopedia.kaspersky.com/knowledge/antivirus-programs-their-quality-and-issues/

Kaspersky. (2024). *Complete Security Plans For You & Your Family*. Retrieved from Kaspersky: https://www.kaspersky.com.au/home-security

Khan, M. I., Foley, S. N., & O'Sullivan, B. (2022). *Database Intrusion Detection Systems (DIDs): Insider Threat Detection via Behaviour-Based Anomaly Detection Systems - A Brief Survey of Concepts and Approaches.* CCIS.

M., G., & (Ph.D.), S. C. (2023). A comprehensive survey on deep learning based malware detection techniques. *Computer Science Review, Vol. 47*.

M., G., & Sethuraman, S. C. (2023). A comprehensive survey on deep learning based malware detection techniques. *Computer Science Review, Vol. 47*.

Maniriho, P., Mahmood, A. N., & Chowdhury, M. J. (2022). *MalDetConv: Automated Behaviour-based Malware Detection Framework Based on Natural Language Processing and Deep Learning Techniques.* La Trobe University.

McAfee. (2024). *Antivirus*. Retrieved from McAfee: https://www.mcafee.com/en-au/antivirus/mcafee-total-protection.html

Microsoft. (2024). *ebpf-for-windows.* Retrieved from GitHub: https://github.com/microsoft/ebpf-for-windows

Microsoft. (2024). *Microsoft Defender for Individuals*. Retrieved from Microsoft: https://www.microsoft.com/en-au/microsoft-365/microsoft-defender-for-individuals#footnotes3

Mira, F. (2019). *A Review Paper of Malware Detection Using API Call Sequences.* IEEE.

Russinovich, M. (2023, March 9). *Process Monitor v3.95*. Retrieved from Microsoft: https://learn.microsoft.com/en-us/sysinternals/downloads/procmon

Santos, D. (2021). *Comparison of Paid Subscription vs Freeware Software on Antivirus program.* Retrieved from University of Hawaii: https://dspace.lib.hawaii.edu/server/api/core/bitstreams/a461115f-4b99-4644-8f48-4a1e9d8b4c1e/content

Souri, A., & Hosseini, R. (2018). A state-of-the-art survey of malware detection approaches using data mining techniques. *Human-centric Computing and Information Sciences* .

Talukder, S., & Talukder, Z. (2020). A SURVEY ON MALWARE DETECTION AND ANALYSIS TOOLS. *International Journal of Network Security & Its Applications (IJNSA)* .

The Volatility Foundation. (2024). *The Volatility Framework.* Retrieved from Volatility: https://volatilityfoundation.org/the-volatility-framework/

Winsider Seminars & Solutions, Inc. (2023). *System Informer*. Retrieved from System Informer: https://systeminformer.sourceforge.io/

You, I., & Yim, K. (2010). *Malware Obfuscation Techniques: A Brief Survey.* IEEE.

8 Appendix 1 – Research Project Timeline

Table 4. Project Timeline incl. Deliverables and Due Dates

Number	Focus	Deliverable	Dependency	Deadlines	Fulfilment
1	Literature Review	Project Proposal		6/9/2023	YES
2	Development of software to mimic malicious attacks		1	25/9/2023	YES
3	Testing against commercially available anti-virus software		2	7/9/2023	YES
4	Analysis of testing	Project Progress Report	3	12/10/2023	YES
5	Draft of Progress Report	Progress Report Draft	4	13/10/2023	YES
6	Project Progress Report & Oral Presentation		5	25/10/2023	YES
7	Outline of Phase 2		6	6/3/2024	YES
8	Development of further software		7	20/3/2024	YES
9	Testing of development software		8	24/4/2024	YES
10	Analysis of final project outcome		9	8/5/2024	YES
11	Draft of Project Delivery Report	Project Delivery Draft	10	15/5/2024	YES
12	Project Delivery Report & Oral Presentation	Project Delivery Report	11	29/5/2024	YES

9 Appendix 2 – VirusTotal Results

Table 5. VirusTotal Results for Ransomware PoC and File Stealer

	Ransomware PoC	Python File Stealer
Threat Label	Trojan	Dropper
Effective Antivirus Software	Cynet – Malicious (score: 100) Gridinsoft (no cloud) – Ransom.Win64.Wacatac.oa!s1 Microsoft Skyhigh (SWG) – BehavesLike.Win64.Backdoor.tc Zillya – Dropper.Agent.Script.405 DeepInstinct – MALICIOUS Jiangmin – Trojan.Generic.hrfzm SecureAge – Malicious Trellix (FireEye) – Generic.mg.7d8c918d65bc6113	Bkav Pro – W64.AIDetectMalware Gridinsoft (no cloud) – Ransom.Win64.Wacatac.oa!s1 ESET-NOD32 – Python/Spy.Agent.UP Skyhigh (SWG) – BehavesLike.Win64.Dropper.rc Cynet – Malicious (score: 100) Jiangmin – Trojan.Generic.hrfzm SecureAge – Malicious TrendMicro-HouseCall – TROJ_GEN.R002V01JI23
Entropy	7.998144	7.997728
Similar hashes	C2AE: fe66e54118bb12b06dcbabb6c2d17206 CAPA: 268a296490a4f42cf0d09b71f94546a2 Microsoft Sysinternals: d988c373a43ba74199d07493174d4797 VirusTotal Jujubox: 38ea0f51e04c4fa138ebd3e125e990c1 VirusTotal Observer: d91b8482a5b9b2355cdea11491de3c8c Zenbox: 6d6c0aac9c414341bf40d2c83027eaf7	C2AE: fe66e54118bb12b06dcbabb6c2d17206 CAPA: 268a296490a4f42cf0d09b71f94546a2 CAPE Sandbox: c46e374ced8a97f6bd6a3d1141ec33c5 Microsoft Sysinternals: bd543ee2dcb52201da7a585473f4fd3c VirusTotal Jujubox: 38ea0f51e04c4fa138ebd3e125e990c1 VirusTotal Observer: ffb9f882ed752ca2390568bfddf06a69 Zenbox: 6d250d70610f748091ea46b82c5bae4f

10 Appendix 3 – Original Methodology

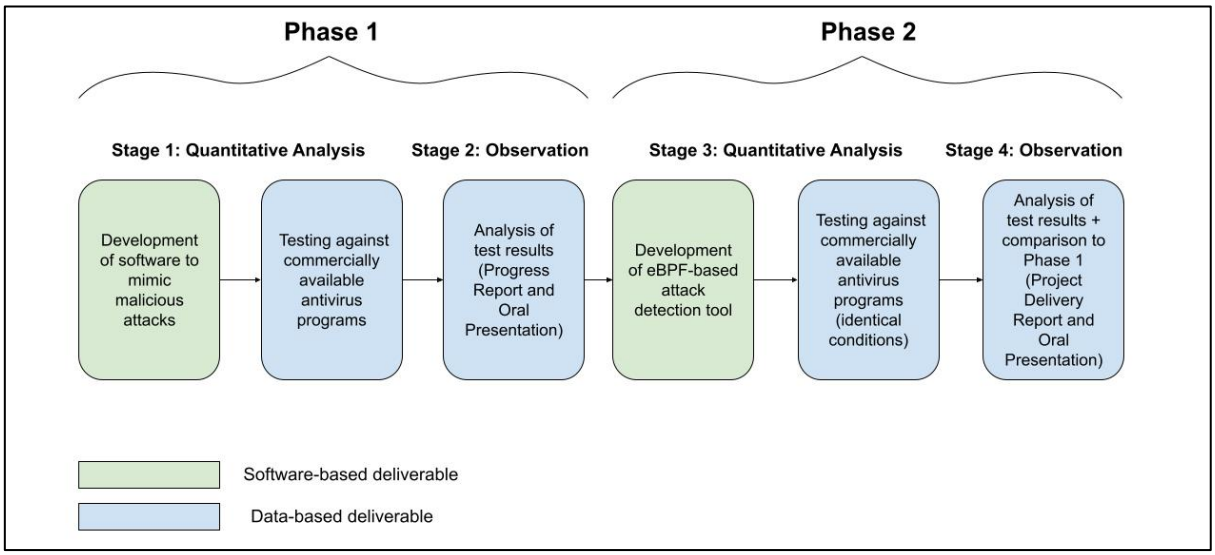

Figure 7. Original methodology developed in project proposal/progress report