Project: Detecting Cyber Attacks in Windows using eBPF

Date: 6/9/23

Student Engineer: Max Blackstone

Student ID: n10762710

Supervisor: Gowri Ramachandran

Version	Date	Author	Changes/Comments
1.0	23/8/23	Max Blackstone	

1 General Objective

The world is becoming increasingly reliant on the internet, as is the necessity for security against cyber threats. With over 5 billion people using the internet (Petrosyan, 2023), methods to steal users' information and data are constantly evolving; hence, cybersecurity tools must evolve simultaneously. Static malware detection strategies are becoming less effective with the sheer number of attacks that all act and present differently to current anti-virus software. Therefore, this project aims to focus on improving detection methods of behavioural cyber-attacks on Windows. Doing so will consist of two phases: a data-gathering phase, and the software development phase. The first phase will involve the creation of software to mimic common malicious cyber-attacks, identified from the literature review. After, a variety of anti-virus software will be installed on a Windows machine, then tested using the attack software. The results of this testing will be analysed and assessed for weak points in this pre-existing software. Phase 2 will involve the development of a new cyber-attack prevention software based off the eBPF tool. Using the same attack scenarios designated for phase 1, the new software will undergo the same testing so the results can be compared. Upon completion of this project and all deliverables, the following two research questions will be answered:

RQ1: How effective are commercially available antivirus software at protecting data from current malicious cyber-attacks?

RQ2: How can the implementation of eBPF as a monitoring tool improve the success of a behavioural malware detection software compared to current commercially available antivirus software?

2 Key Finding from the Literature - Impact on Your Project Design

To ensure this project is appropriately informed, a literature review was conducted. The entirety of this review is available within the Appendix. The key takeaways deemed most relevant were that cyber-attacks are a constantly evolving threat, with new forms of malware and ransomware being developed every day (Aslan & Samet, 2020). Security against these attacks needs to be updated using new countermeasures to protect the millions of internet users from them. Focusing on detection via behaviour yields promising results according to literature, being deemed a more beneficial approach rather than using the more common signature-based methods (Chakravarty, Raj, Paul, & S., 2019). Whilst there are numerous works based on the issue of cloud cybersecurity and modern cyber-attacks, there is a gap in focus on behaviour-based detection and the more efficient tools used to achieve it; more so, there is almost no implementation of eBPF as a monitoring tool, with literature generally excluding it from lists of security observation tools. As such, this project aims to fill the gap in literature with a promising solution to counter modern attack methods.

3 Stakeholders & Resources

For this project, there is a single stakeholder: the supervisor, Gowri Ramachandran. There are no external companies or team members involved in this project. Therefore, Mr Ramachandran is the only person privy to project progress and outcomes. In terms of resources, the project requires a Windows computer, an internet connection, and the eBPF program (which is open-source and easily accessible). Additionally, due to concerns listed in the Risks, Requirements & Constraints section, access to a secure network is of high importance.

4 Project Methodology

The methodology for this project will consist of two primary phases, each consisting of a quantitative analysis and observation stage. Initially, the first step is the preparation of a proposal, outlining the project and providing necessary information for the stakeholder to review. Upon completion and stakeholder satisfaction, the first phase can begin:

The steps for Phase 1 are as follows:

1. Referencing code from GitHub, develop software to mimic three different current malicious attacks.
2. Conduct testing against 3 different forms of commercially available antivirus software (including built-in options such as Windows Defender).
3. Prepare a progress report detailing the outcome of the testing and how it will be implemented to influence phase 2 of the project (presented to the stakeholder by 25th October 2023).

Upon the development of the faux-cyberattacks, a quantitative analysis is conducted via running them against commercially available antivirus software and collecting the data based off their performance. The attacks may mimic code obfuscation techniques, ransomware, and other methods used by cybercriminals. Then, the observation stage is fulfilled by a progress report detailing the results of the testing and analysing it for areas of concern (where improvement is necessary). Phase 2 will then consist of the following steps:

4. Using eBPF, develop an attack detection tool, following a specific timeline outlined in the Timeline table included later in the document.
5. Using the previously developed faux attacks from Phase 1, run a second round of testing using the eBPF-based tool in an identical environment to the previous phase.
6. Prepare a final report analysing the results and comparing them to the results of Phase 1, proving the new software is more effective at detecting researched attacks such as code obfuscation and ransomware (finished by Week 10 Semester 1 2024).
7. Finally, present the findings of this project via an oral presentation.

After the development of the eBPF-based attack detection tool, a quantitative analysis will once again be performed. The data will be gathered in an identical manner and environment to that of Phase 1 to ensure credibility of the results. The final observation stage will involve an analysis of the tool's performance, which will be compared to the outcome of Phase 1 to determine success in the improvement of anti-

malware tools. This combines all quantitative data gathered throughout the project into a synthesized outcome to present the projects findings in full.

To further illustrate the methodology of this project, Figure 1 has been produced. It details the phases, different stages and outcomes intended while colour-coding the deliverables.

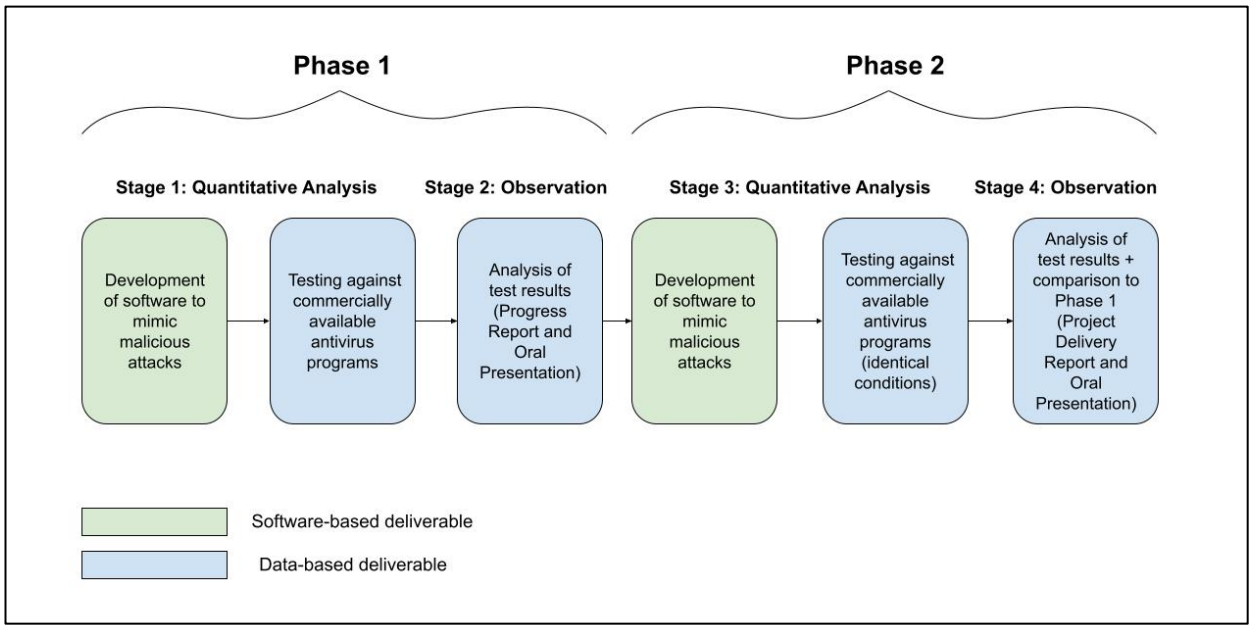

Figure 1. Visual Representation of Project Methodology

5 Deliverables

Throughout this project, several deliverables will be presented to the stakeholder to ensure project progress is accounted for and endorsed. In terms of interim deliverables, weekly meetings are scheduled with the stakeholder updating the progress, and meeting minutes will be produced after each. Additionally, at the end of Semester Two of 2023, a midway status report will be produced detailing the results of Phase 1 of the project, as well as an oral presentation of the project's progression. Weekly meetings will continue in Semester One of 2023, with an additional outline of the software development cycle being presented at the start of the semester to ensure stakeholder satisfaction and that deadlines are being met. By Week 10 of Semester 1 2024, this project will result in a fully developed anti-malware software based off eBPF, and an analytical report comparing the results of Phase 2 to Phase 1. A final oral presentation will be conducted to demonstrate the software and allow for further questions to be answered. Furthermore, in the interest of aiding future research, a virtual machine with a working tool and a video demonstration of the utilisation process will be delivered. This will allow for the project to be built upon by future scholars and contribute to the literature on the topic. Further information on these deliverables and the timeline of their completion will be available in Table 1.

6 Risks, Requirements & Constraints

This project features very few risks, although there are two primary ones that must be identified. The first major risk is the necessity to verify the attack software used in Phase 1 mimics malicious attacks, instead of using real malware. The second risk identified is to keep testing off networks containing sensitive information. Although the testing software is a non-malicious recreation of a cyber-attack, it could have negative impacts or be wrongfully identified by existing security and cause any number of issues. Additionally, the introduction of real malware, even in a secure network, presents numerous likely safety concerns, such as the accidental participation in an actual cyberthreat leading to potential loss of personal data. Hence, the software will be run on a safe and private network, using faux cyberthreats to ensure no negative impacts result from this project. In terms of existing systems, eBPF is required to be interfaced with, as it is the monitoring tool that will be utilised to develop the attack detection software. This is a necessity to the progression of the project. Currently, there are no limitations to the scope; if issues arise further down the timeline, the scope can be changed/refined based off the strategies in the Management of Project Changes section.

7 Quality & Sustainability

The quality of the outcome will be determined based on the effectiveness of the software produced during Phase 2 compared to the commercially available anti-virus software of Phase 1. This will be detailed in a final project report that will succinctly deliver the analysis and validity of the software and make judgements on future implications of the project. Furthermore, in terms of other deliverables such as progress reports and oral presentations, the quality will be judged by the stakeholder in the interest of producing an informative and detailed project.

Due to this project being a computer/software project, sustainability is not of high relevance in the physical sense. However, since the threat of cyber-attacks are consistently evolving, the software will only have a short-term impact in its final state at the end of this project. To achieve a longer life cycle would involve continuous work and research to appropriately detect the new forms of malware being released. The virtual machine and working tool that will be delivered at the submission of this project can aid in this process.

8 Timeline & Deliverables

Table 1. Project Timeline incl. Deliverables and Due Dates

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

9 Management of Project Changes

Since the only affecting party is the supervisor, management of project changes will be solved by open communication between myself and the supervisor. Any changes (based off existing work on the project) will be discussed at meetings and detailed in the minutes, with a mutual understanding of the direction the project should progress understood. In the event of an agreed shift in project direction, the scope and all documents outlining the project will be updated to reflect this change and keep communication clear and consistent.

10 Sign off

	SIGNATURE	DATE
STUDENT ENGINEER	MB	6/9/2023

11 Appendix: Review of Literature

Introduction

Malicious cyber-attacks are a constantly evolving threat to cloud users and is of major concern due to the necessity of web usage in everyday life today. However, there is little research on the effects of focusing on behavioural detection of attacks. The aim of this project is to explore how a specifically behavioural attack detection model can suppress malicious cyber-attacks better than current anti-virus software commercially available today. For the selection of relevant sources, it was important to primarily restrict papers to within the past three years. Since new forms of malware are constantly found, it is important to remain up to date with current issues. This also led to the inclusion of some blogs as relevant literature; the ability to upload the information without the time-consuming review process allowed more recent attacks to be analysed. However, particularly pertinent papers that featured many citations by recent literature were also included, due to the evidence of it remaining relevant. The review was conducted in a broad-to-specific pattern, approaching the problem in a broader sense before narrowing down the research to increasingly specific issues.

Key themes

Relevant papers have identified numerous threats used by cyber-attackers, producing a foundation for the knowledge required to create countermeasures (Cyble, 2023; You & Yim, 2010; Jensen, Schwenk, Gruschka & Iacono, 2009). These threats include code obfuscation, misspelled packages, cloud malware injection, account and service hijacking, and more. Misspelled packages are a significant risk, posing as legitimate Python packages with a slight difference in the name (Cyble, 2023). Due to the installation process typically used for Python, misspellings are common, and a growing number of these malicious packages leaves no room for error. These packages make use of techniques such as malicious downloaders, and code obfuscation to hide malware that will be executed upon the fake package being downloaded. Code obfuscation can consist of techniques such as ‘instruction substitution’, in which instructions are replaced with equivalent ones for malicious purposes. Alternatively, another technique is ‘register reassignment’, where the malware switches registers generation by generation while presenting as the same code and behaviour (You & Yim, 2010). Cloud malware injection involves tricking a Cloud system to treat a malicious virtual machine instance as valid, then redirecting user requests to the malicious instance, allowing the malware to execute. Promising countermeasures include usage of a hash value for comparison to identify valid instances versus malicious instances (Jensen, Schwenk, Gruschka, & Iacono, 2009). This is just a handful of the potential cyber threats internet users face daily.

There are a vast number of potential detection methods being created and used against cyber threats. These include signature-based, behaviour-based, heuristic, model checking, deep learning, cloud-based, mobile-based, and IoT-based (Aslan & Samet, 2020). Signature-based methods involve using a pre-determined pattern to identify attacks; these patterns can be byte sequences, strings, recurring features, and more (Caviglione, et al., 2021). Behaviour-based methods, alternatively, recognise malicious or unwanted behaviour against clean templates. Although code may be changed, it will be registered as behaving similarly, and therefore identified as a threat (Aslan & Samet, 2020). Heuristic involves a combination of the two previous methods; using both rules and machine-learning techniques to identify malware using strategies both through signatures and behaviourally. Whilst highly accurate, it falters on complicated viruses, yet is claimed to out-perform anti-virus software such as McAfee and Norton (Arnold & Tesauro, 2000). However, due to the age of the paper it is unclear whether this could hold up to modern

cyber-attacks. Deep learning-based detection methods are becoming increasingly viable, believed to be powerful and effective by learning from example images (Aslan & Samet, 2020). However, it fails to detect evasion attacks, as crafted inputs can deceive machine-learning based models, leading to a failure to classify the attack as malicious (M. & Sethuraman, 2023). Many of these methods are promising yet not fully implemented, unlike signature- and behaviour-based methods. As stated by (Souri & Hosseini, 2018), the primary method for detecting cyber-attacks is still signature-based methods rather than any of the other number of methods. Despite this, (Aslan & Samet, 2020) point out that whilst signature-based methods are efficient and quick, they are insufficient at detecting unknown threats. This issue is worsened by the fact that as viable signatures are developed, they are generally known by malware and can consequently be easily evaded. Therefore, it would be prudent to develop an improved behavioural-recognition model, that cannot be evaded as easily as signature-based models and could become the new standard for anti-malware protection software.

Focusing on behaviour-based methods for malware and ransomware detection reveals a lot of promising applications (Maniriho, Mahmood, & Chowdury, 2022; Chew & Kumar, 2019). Behaviour-based detection relies on dynamic analysis of the malware; generally, there are two main approaches. First, is analysing the difference between a fixed start and end point after allowing a malware sample to run. The combination between the two states reveals any modifications that occurred to the original system. Alternatively, malicious activities can be monitored during run-time using tools to track the behaviour (Mira, 2019). This works especially well with API calls. There are also a variety of different compromise indicators that can help identify malicious behaviour though, including file changes, file entropy and canary files (Chew & Kumar, 2019). File changes refers to a file appending to an unknown extension after being encrypted; this unknown extension is a sign of malicious activity occurring. File entropy is useful as encryption results in high entropy; therefore, the entropy level can determine if there's a malicious file in the system. Finally, canary files are primarily irrelevant files with filler data, that notify the user in the event of being manipulated by malware. However, (Chew & Kumar, 2019) found that individually these methods (specifically file entropy) can have high occurrences of false positives, so it is recommended that the magic bytes of a file are also read. This allows for the detection of an unknown extension, which when paired with analysis of high entropy greatly reduces false positives. There are also many new approaches based off behavioural analysis to combat malware; for example, (Maniriho, Mahmood, & Chowdhury, 2022) presented a behaviour-based framework called MalDetConV, which uses a combination of neural networks and deep learning to achieve extremely high detection accuracy on both known and unseen malware. It focuses on monitoring the behaviour of API calls, which gives significant understanding of the difference between a benign and malicious executable. Other techniques for behaviour-based malware detection include anomaly detection and specification-based monitoring (Chakravarty, Raj, Paul, & S., 2019). Anomaly detection is similar to the fixed start and end point analysis (Mira, 2019), in which normal behaviour is stored as a reference. Therefore, if calls are written to a directory via a program that has no interaction with that directory, it is deemed as suspicious. Drawbacks to this method involve vulnerability to false alarms (a previously mentioned issue in behavioural detection) and being susceptible to mimicry attacks. Specification-based monitoring combines aspects of behavioural detection with signature-based methods. A policy is written that events must adhere to, or a specified action will run. An example of this is only allowing downloads from websites on the Whitelist to prevent malicious downloads automatically occurring. According to (Chakravarty, Raj, Paul, & S., 2019), this method has advantages over anomaly detection, such as increased resilience and lesser chances of false alarms due to easy adjustment of the policy. While there are many promising paths to malware detection via behaviour, it is also important to acknowledge the limitations of the approach. Behaviour-based systems are looking for deviation from

normal behaviour, which means that technically it's possible to identify unseen attacks. However, in practice it is much more difficult to specify this difference between normal and anomalous actions. Additionally, there are threats such as mimicry attacks that are built to evade behavioural detection, presenting normative behaviour yet acting maliciously (Khan, Foley, & O'Sullivan, 2022). These are important considerations to note within this topic.

The behaviour-based approach to malware detection requires the use of a tool for monitoring program behaviour. Currently, there is a variety of malware analysis tools available for use, including REMnux, Zeek, and Google Rapid Response (GRR) (Talukder & Talukder, 2020). REMnux is a Linux toolkit aimed at reverse-engineering malware. It contains several examples and resources to detect browser-based vulnerabilities and ransomware. Zeek is essentially an Intrusion Detection System, allowing users to view network activity and find anomalies. GRR is a Python client, used to review memory and identify malware footprints in a convenient manner for simple remote analysis. All these programs are open source, focusing on slightly different methods yet achieving a similar goal. While some of these tools are applicable for multiple types of operating software, but it is especially important to develop cybersecurity software for Windows. Since Windows is one of the largest OS, there is an exponential increase in the production of malware; over 100 million new threats in the year 2021 (Drapkin, 2021). There are several malware detection tools specifically targeted for Windows, including ProcMon (Process Monitor), Process Hacker, and RegShot (Forum of Incident Response and Security Teams, Inc., 2023). Process Monitor is a software available through Microsoft, used to observe Registry, real-time file system, and process/thread activity (Russinovich, 2023). It has focused use on file and registry access, as well as analysing processes currently running. Process Hacker (also known as System Informer) is a similar tool to ProcMon, which helps users to detect malware, debug software and monitor resources. It is free to use and is highly powerful in its purpose (Winsider Seminars & Solutions, Inc., 2023). RegShot is another open-source tool yet is specifically targeted to changes in the registry through comparison of changes via snapshots. However, these are just some of the tools listed in literature; there are many new tools being released with promising implications for use within the cybersecurity field.

Due to the necessity of internet safety, commercial anti-virus software is widely available and around 76% of computers worldwide have a form of antivirus software installed (Aashind, 2023). These could include McAfee, Avast, AVG and Windows Defender. Windows Defender is a free built-in security program for Windows 10, which notably provides system takeover prevention (an absent feature in most commercial antivirus software) (Delaney, 2020). McAfee, Avast and AVG are all primarily paid-for solutions to virus detection, with a variety of helpful features and high success rates against cyber-attacks. However, there are problems that stem from the commercial aspect of the software (Kapersky IT Encyclopedia, 2023). Due to how quickly new malware spreads, antivirus vendors can struggle to keep up with protection against all of them, leaving vulnerabilities in the protection they offer. Additionally, it is difficult to balance performance and protection; thorough data scanning is necessary to check for all possible viruses yet will increase CPU usage and slow the computer. Finally, as some programs focus on different aspects of protection more, people could want multiple for the safest experience possible. This causes conflicts in the kernels and could cause both to miss malicious events and more, therefore being impossible to do.

Conclusion

In conclusion, the literature states there are ever-increasing forms of malicious cyber-attacks, and a large set of detection methods being employed as countermeasures. This is agreed upon by the majority of literature, all stating the numerous ways cyber-attacks are evading detection, while agreeing upon the current methods used. There appears to be a lacking focus on behavioural detection methods though, specifically missing the use of eBPF as a monitoring tool. Overall, this is a promising avenue for improving malware detection methods, and further research will improve understanding and hopefully aid in creating a safer online environment.

12 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.

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

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

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/

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., & 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.

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

Petrosyan, A. (2023, August 29). *Number of internet and social media users worldwide as of April 2023*. Retrieved from Statista: https://www.statista.com/statistics/617136/digital-population-worldwide/

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

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)* .

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.