Enhancing Malware Detection in Windows: Leveraging Behavioural Analysis and eBPF

Date: 27/10/23

Student Engineer: Max Blackstone

Student ID: n10762710

Supervisor: Gowri Ramachandran

Version	Date	Author	Changes/Comments
1.0	27/10/23	Max Blackstone	

1 Project Summary

This project focuses on the detection of malware in Windows, due to the increasing reliance on the Internet and computers in everyday life. Since cybercrimes are constantly evolving, tools effective at recognising the new behaviours exhibited by malicious programs must be produced in cohesion. It is comprised of two phases; the first phase involves a study of current commercially available antivirus against faux malware. This provides an insight into the current capabilities antivirus provides and identifies potential weak points/strengths. The results of a critical analysis into the data gained by Phase 1 will be used as the foundation for Phase 2. Phase 2 will revolve around the development of an eBPF-based program that uses behavioural methods to detect malware, specifically conducting testing with the same methods and software used for Phase 1. The resulting data should display successful results in identified weak points from antivirus, therefore producing a functional final concept. The new tool will undergo the same testing and use the same attack scenarios designed for phase 1, creating fair comparison between the outcomes of both phases of the project. As a reminder from the proposal, the research questions this project will answer are as stated below:

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 Methodology

The project methodology remained almost identical to the project proposal; the original information on this subject is as follows:

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.

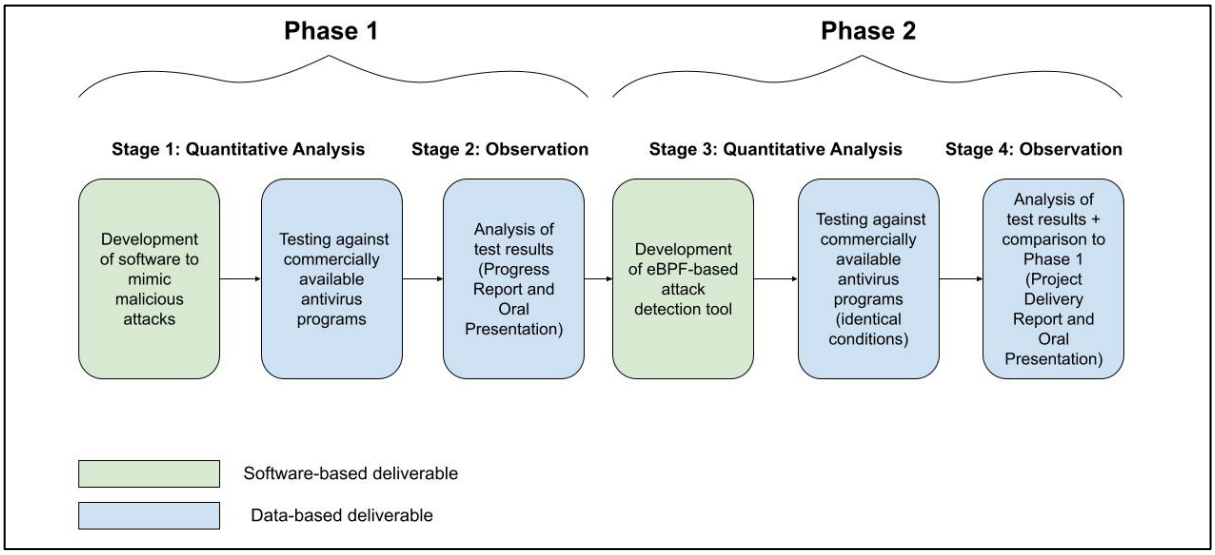

Figure 1. Visual Representation of Project Methodology

The slight changes made since the last document involve using two forms of faux malware rather than three and increasing the number of antivirus software to test against. Development of the software occurred as planned, as did the testing process which is available in Section 3 (Results). In this report, the quantitative analysis stage is primarily focused on, providing depth and understanding to the data gathered by the first two steps of Phase 1. This will provide the foundation for Phase 2, in which an attack detection tool will be developed and tested under identical conditions. Based off this information, a final report and oral presentation will be conducted to present the results and deliver a virtual machine and working tool for use in future avenues of cybersecurity.

3 Results

Description of Malware:

The first piece of testing malware (Malware A) was the Dummy Ransomware PoC (Proof of Concept). This code was developed by Vishal Dogra, a QUT student. The code emulates general ransomware, in which a malicious program disguises itself as legitimate to trick unsuspecting victims into running the executable. Once run, the hacker gains access to personal data on the computer and uses it to lock/encrypt files until the victim caves to the hacker's demands. This PoC targets the main folders of the computer (default target being Documents) and encrypts each available file. The configuration data for encryption is sent to a C2 server, and the exploit is conducted, turning files into indecipherable messes.

The Python File Stealer program (Malware B) is an aptly named form of faux malware that subtly runs on computers (unbeknownst to the victim) to steal valuable personal information. This data can be monetised and used by hackers as they gain access to bank account passwords, emails, saved files and much more. The specific GitHub repository referenced was developed by 'Antsbatscats' and used a Discord Webhook to gather the data stolen after running the file. It also blacks out the screen and hides all open applications, blocks keyboard input, and recursively conducts a search through all drives and directories to find suitable information to be sent to the webhook. For this project, due to the inherent risk of running code not developed in-house or from a reputable source, the code was altered to replace some of the functionality. While initially blacking out the screen still, applications are no longer hidden so the terminal can be observed after the exploit has occurred. Finally, rather than using a Discord Webhook a simple print statement was sent to the terminal detailing the success of the infiltration and the file path accessed. This removed the risk of potential hidden malicious code in the repository, while still accomplishing the actions necessary to present as malware.

Testing Process:

For testing, both Malware A and Malware B were compiled into executable files using PyInstaller. This was done to maintain functionality expected after testing in the terminal, while also being a suitable format for VirusTotal. VirusTotal is a website aimed to analyse suspicious files and review detectability rates from antivirus, as well as provide more detailed information on the behaviour and aspects of the file. The data from VirusTotal will provide confirmation of the faux software presenting as malicious and inform expected results from further testing. After this, using a series of commercially available antivirus software, tests will be conducted for each malware on its ability to evade detection. To achieve this, a custom scan will be run on the folders containing the executables to determine if there is any error message or actions from the antivirus.

VirusTotal Results:

Table 1. Analysis of results from VirusTotal after comprehensive scan of both forms of malware

	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

Referring to Table 1, both faux malicious files were analysed. VirusTotal identified the two different forms of malware as 'Trojan' and 'Dropper' for the Ransomware and File Stealer respectively. This is beneficial for the data as it widens the scope from defence against a singular type of threat to two. Each only had a relatively small number of antiviruses estimated to 'flag' the executable; 9 and 8 detections for the Dummy Ransomware PoC and File Stealer respectively. Both Malware A and B have respective detection percentages of 12.5 and 11.3%, implying that just under 90% of common antivirus software does not recognise these malicious behaving files as a threat. Also included in addition to the successful antivirus software list was the detection label for each engine. These will be analysed to discover how to successfully detect faux malware for implementation in Phase 2 of this project. According to the list, out of the three chosen antivirus methods, only Microsoft Defender is expected to have a response to the malware. The other two, being Avast and McAfee, are expected to not be able to detect the threat within the executable. Another indication of potential malicious behaviour is the extremely high entropy that both files have. Entropy doesn't necessarily identify a file as a virus, but entropy is increased with encrypted/obfuscated code and data; hence, nearly all malware has high entropy due to the legitimate presentation made possible by hidden malicious behaviour. An important aspect of the file's behavioural analysis are the behaviour similarity hashes. These are identifiers that cluster common patterns and behaviours of similar code to inform of the file's intentions. Doing so provides a baseline for behaviour commonly presented by malicious code, which benefits future development of a behaviour-based antivirus by utilising the common trends. The VirusTotal report also included a full list of files opened, deleted, and written. This allows in-depth knowledge of the exact process of the search through personal data conducted by the malicious file, which further improves knowledge of the malicious code. In Phase 2, this will potentially be utilised to combat malware in the development of the eBPF-based program. The knowledge gained from the VirusTotal reports will be used in conjunction with results from testing to gain an overall conclusion on the current effectiveness of popular commercially available antivirus software.

Antivirus Software Detection Results:

The antivirus software selected was chosen based on a few different factors. First, only software present on the list provided by VirusTotal was considered for use, to ensure comparison between detection ability in the two mediums. Second, this project aims to have zero economic constraints; therefore, only free-to-use antivirus were selected, with no paid services utilised. This removed a significant portion of the list, especially affecting software that did flag the malware on VirusTotal, as they were primarily paid applications. This was important to the project as economic consequences are aimed to be strictly avoided. The resulting list of (five) tested antivirus software was then tested as per the process previously described.

Table 2. Results of testing faux malware against selected antivirus software

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

As evident in Table 2 above, both forms of malware successfully evaded all the different antivirus programs. According to the data from Table 1, these results are supportive of VirusTotal's analysis that Avast, McAfee, and Kaspersky do not detect the malicious file. However, Microsoft Defender was specified to flag the file as suspicious; in practice, this did not occur. Therefore, despite the high entropy of the files and the popular threat label assigned to the executables by VirusTotal, Defender, Avast, Kaspersky, and McAfee all lacked the ability to detect the malicious code. In contrast, AVG detected the Python File Stealer when running a scan on the folder; this is visible in Figure 2 below:

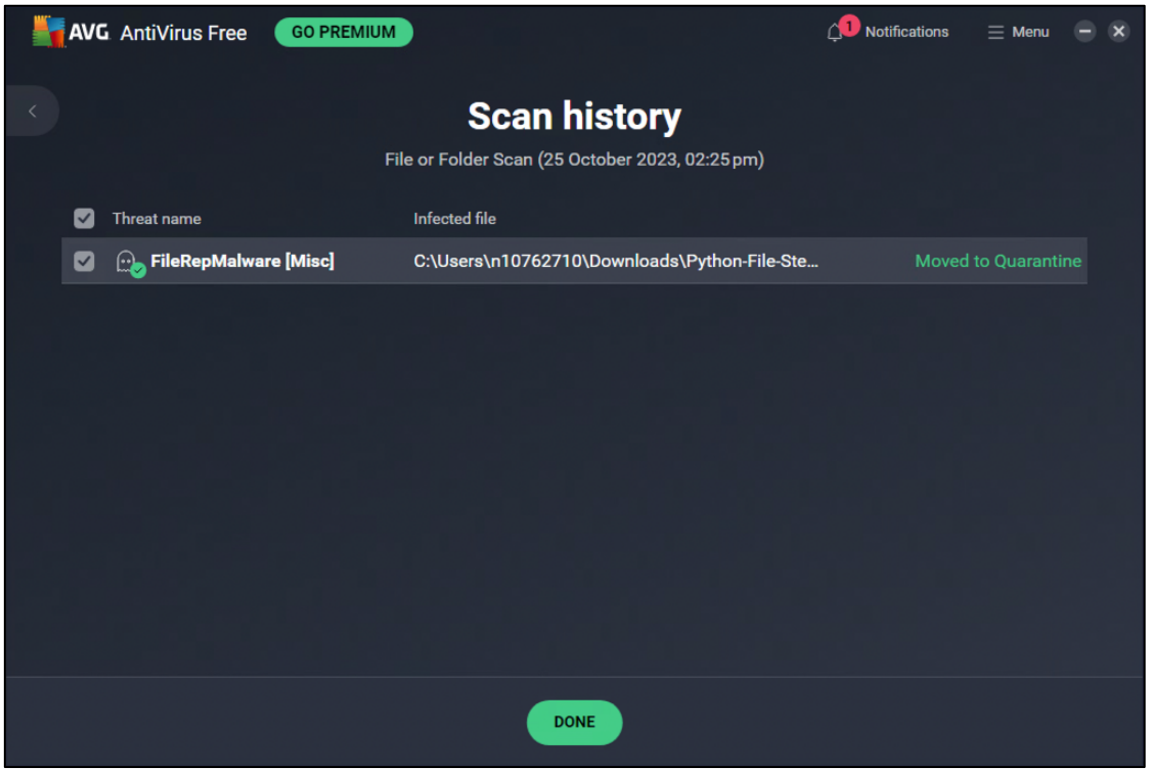

Figure 2. Successful Identification from AVG antivirus

It successfully identified the file as infected, then moved it to quarantine to ensure no further attacks. It displayed the correct file path to the executable and labelled the threat 'FileRepMalware [Misc]'. However, AVG was unable to detect the Ransomware PoC when running an identical scan on the folder.

It is also important to note that the majority of the antivirus software that VirusTotal predicted to detect the malware are not free programs; all require a financial input for usage. Examples include Cynet, DeepInstinct, and SecureAge. These results, in cohesion with the data from the VirusTotal reports, support the idea that antivirus needs consistent evolution to protect against modern cyber security threats. Additionally, there is an evident lack of accessibility to successful software due to financial requirements.

4 Interim Deliverables

The primary deliverables available at this point in the project are fully encompassed in Section 3 (Results). Two forms of faux malware were developed/used: a Ransomware PoC and a dummy File Stealer. The altered code for the File Stealer (referencing the original repository) will be included in the Appendix as an example of the faux malware development stage of the project. The Ransomware PoC, as earlier mentioned, was generously sourced from Vishal Dogra. The creation and testing of these abide by the timeline included in the Appendix. Other deliverables include the literature review and project proposal earlier submitted. Weekly meetings occurred with the stakeholder to ensure the project remained focused and accounted for, with meeting minutes emailed within one business day of every meeting. As such, these are available as evidence of consistent dedication and effort being applied to the project. Additionally, this document and its corresponding oral presentation provide a status report on the progress of the project, providing evidence of the continuation of involved work. At the oral presentation, any further questions not answered by this document will be answered, and criticism given on the current status of the project will be attentively received and influence the work conducted in Phase 2.

5 Risk Management

Risk management was deemed of utmost importance due to the nature of the project. Therefore, it was essential to consider any adverse effects that could occur as due to sourcing and testing faux malware.

As some of the code used was referenced from GitHub, an investigation into the intention of the code and its suitability for use was paramount to conducting the tests in a safe environment. Only repositories that stated a safe outcome and educational purpose were referenced and used for this project, after being evaluated by both the project lead and the primary stakeholder (Gowri Ramachandran).

Additionally, to avoid the case of the virtual machine being hacked, no personal information was entered at any stage of the process; a new email and GitHub account were created using a name and password with no relevance to sensitive details. Potential personal data leaks were of great concern during this project, therefore validating the necessity of these methods.

As stated in the project proposal, it was important to perform testing away from sensitive networks, as an accidental mistake in the faux malware code could lead to data from every user of the network being stolen, potentially have severe negative effects.

Therefore, there was no negative outcomes from conducting the testing, as there was no personal data to be taken, nor tests done with malicious intentions.

6 Ethical Considerations

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

To demonstrate integrity, three main guidelines were followed: being honest and trustworthy, respecting dignity of other people, and acting from a well-informed conscience. Throughout the project, honesty was of utmost concern; criticism on work from the stakeholder has been accepted and understood by all people involved. Any concerns were communicated in the weekly meetings, with mutual understanding ensured before leaving. It was also deemed crucial to give credit where it's due, as was the case with Vishal Dogra allowing the implementation of his Ransomware PoC in this project. Additionally, in terms of respect for other's dignity, a safe environment was always cultivated, regardless of gender, age, religion, or any other factors. All actions undertaken while working on the project were feasible for the project lead, well-informed, and conducted in a professional manner. This encompasses the behaviour necessary to abide by the aspect of integrity in the ethics code.

Competent practice is an important skill, especially when working with areas of risk such as cybersecurity. The three main aspects of this are: acting with adequate knowledge, maintaining and developing skills, and objective representation of competence. There is an ongoing development of skills throughout the project, proven by the outcome of this report and the literature review completed in the project proposal. Additionally, due to the risky nature of the project, caution and diligence were important qualities to have as per the risk management section. Transparency on qualifications in this area has been maintained throughout meetings with the stakeholder from the start of the project.

Exercising leadership was not considered a primary concern in this project due to it comprising of individual work. However, there are still elements of this aspect of the code that were applied. In terms of upholding the reputation of engineering, ethical practice has been supported throughout the project. However, diversity among group members is not particularly relevant due to the project being conducted in an independent manner; despite this, diversity is promoted and there was an opportunity for another engineering practitioner (Vishal Dogra) to make an impact on this work. Another clearly demonstrated element of leadership was the clear communication with the stakeholder, with issues being revealed in a timely manner.

Finally, the promotion of sustainability is inherent in the project. Engagement with the community/stakeholders in a responsible manner is the first guideline, exercised through constant involvement of the stakeholder and clear communication of possible consequences of the project. Maintaining the health and safety of the community/environment was also integral to the process; hence, the inclusion of the risk management sections in both this report and the project proposal. Additionally, the impacts to future generations were considered, identifying sustainable and positive outcomes overall.

7 Sustainability

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 makes use of an open-source monitoring tool and has no intended costs associated. Therefore, benefits of this project (a safer environment during computer use) outweighs the economic cost.

In the final state of the project, the software will unfortunately have a fairly short-term lifecycle. Longevity is only attainable via constant work due to the evolving nature of cybercrime; absolute prediction of future methods of malicious attacks is an unattainable goal. However, the final virtual machine and working tool will provide a solid foundation for further development and impact, allowing the project to have an impact on future forms of cybersecurity. Hence, the developmental process of the malware detection tool should be documented thoroughly for ease-of-understanding in the future.

8 Information and Resources

A resource drawn on to complete the testing necessary for this report was a Dummy Ransomware PoC developed by Vishal Dogra, a QUT student also working on a thesis project. Due to the risky nature of sourcing faux malware from the internet for testing, using this resource provided a suitable software (developed in-house) that was totally safe and came from a trustworthy source. It also allowed communication in the case of errors, considering it was originally a Linux program converted for use on Windows as per this project's intended use.

Another resource used for testing was the Python File Stealer code from GitHub (Antsbatscats, 2023). It underwent a severe investigation of safety and was deemed appropriate for use. However, the original code used a Discord Webhook to extract information, which was determined to be too risky. Hence, the code was altered to remove this data transferral and instead print a message to terminal listing success/failure for each file path checked. Additionally, the file would hide windows, essentially making the computer useless unless restarted. This hindered the ability to check if the program was running successfully, as the terminal in which the messages were printed was inaccessible. As such, this feature was also removed in the interest of successful testing.

9 Stakeholder Impact

Any input received on this report has been supportive and constructive, allowing the development of a constantly improving final product. The supervisor, Gowri Ramachandran, has been of great aid in shaping the project, ensuring a feasible and involved outcome for each deliverable. Additionally, the weekly meetings have allowed for comprehension of required activities for tasks in a timely fashion. Other areas of impact were the reviewing of potential GitHub repositories considered for use and the connection to Vishal, who provided the Ransomware PoC used in testing.

10 Sign off

	SIGNATURE	DATE
STUDENT ENGINEER	MB	25/10/2023

11 Appendix

Table 3. 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 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	

Python File Stealer: (Antsbatscats, 2023)

```
import os
import requests
import time
import threading
import ctypes
from win32 import win32gui
# from win32 import win32con
import win32.lib.win32con as win32con

ctypes.windll.user32.BlockInput(True)
ctypes.windll.user32.SendMessageW(0xFFFF, 0x112, 0xF170, 2)

def enum_windows(hwnd, window_list):
    window_list.append(hwnd)

windows = []
win32gui.EnumWindows(enum_windows, windows)

# Directories to ignore during file search
BLACKLISTED_DIRS = ['C:\\Windows\\', 'C:\\Program Files\\', 'C:\\Program Files (x86)\\',
'C:\\$Recycle.Bin\\','C:\\AMD\\']
MAX_FILE_SIZE_MB = 8
def check_file(file_path):
    allowed_extensions = ['.txt', '.pdf', '.png', '.jpg', '.jpeg', '.gif','.mp4','.mp3','.py','.js','.mkv','.docx','.xls']
    max_size_mb = 8
    if os.path.splitext(file_path)[1].lower() not in allowed_extensions:
        print(f"Skipping file {file_path} - invalid file type")
        return False
    elif os.path.getsize(file_path) > max_size_mb * 1024 * 1024:
        print(f"Skipping file {file_path} - file size too large")
        return False
    elif os.path.isfile(file_path) and not os.access(file_path, os.R_OK):
        print(f"Skipping file {file_path} - file requires admin privileges")
```

```
        return False
    elif any(blacklisted_dir in file_path for blacklisted_dir in BLACKLISTED_DIRS):
        print(f"Skipping file {file_path} - in blacklisted directory")
        return False
    else:
        return True
def upload_file(file_path):
    try:
        with open(file_path, "rb") as f:
            files = {"file": f}
            print(f"Successfully Hacked - {file_path}")
    except Exception as e:
        print(f"Failed to hack {file_path} - {str(e)}")
def search_files(root_dir):
    for root, dirs, files in os.walk(root_dir):
        if any(blacklisted_dir in root for blacklisted_dir in BLACKLISTED_DIRS):
            # Skip blacklisted directories
            continue
        for file in files:
            file_path = os.path.join(root, file)
            if check_file(file_path):
                upload_file(file_path)
def thread_files(root_dirs):
    for root_dir in root_dirs:
        search_files(root_dir)
# Get a list of all available drives
drives = ["%s:\\" % d for d in "ABCDEFGHIJKLMNOPQRSTUVWXYZ" if os.path.exists("%s:" % d)]
# Split the drives into groups of 4
drive_groups = [drives[i:i+4] for i in range(0, len(drives), 4)]
# Search for files in each group in parallel
for group in drive_groups:
    threads = []
    for drive in group:
        thread = threading.Thread(target=search_files, args=(drive,))
        threads.append(thread)
```

```
        thread.start()
    # Wait for all threads to finish before moving on to the next group
    for thread in threads:
        thread.join()


ctypes.windll.user32.BlockInput(False)
ctypes.windll.user32.SendMessageW(0xFFFF, 0x112, 0xF170, -1)
```

12 Bibliography

Antsbatscats. (2023, May 8). *Python-File-Stealer*. Retrieved from GitHub: https://github.com/Antsbatscats/Python-File-Stealer/tree/main

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