Will Artificial Intelligence solve the challenge of finding solutions to malwares?

London Metropolitan University,
Faculty of Computing School of Computing and Digital Media
London, United Kingdom

School of Computing and Digital Media

Master of Science in Computer Networking and Cyber Security

Summer 2023

Author: **Areef Ismail Rakhangi**

Student No: 22012107

Supervisor: Mrs. Richa Sharma

Summary

This dissertation delves into finding out whether Artificial Intelligence can really provide solutions to malware problems. Malware is an acronym for malicious software or code that is designed to cause theft, destruction, gaining unauthorized access.

Malwares today are targeted with no readymade solutions available and hence understanding malware code immediately and find solution to the issue themselves is very important. Traditional methods are slow and expensive in terms of manhours and knowhow of Malware Analysts.

This study aims to go beyond the traditional technologies of human programmer's interpretation and find out if Artificial Intelligence can solve the challenge of Analysing malware to a large extent and leave it to the experts to find solutions to it.

The study primarily delves into understanding traditional malware analysis and tries to use the latest artificial intelligence technology to investigate computer code thoroughly.

Previous studies into automated malware analysis have faltered and with somewhat limited result even with curated datasets. "Companies are not turning over their cybersecurity programs to AI " (Violino, 2022) and such studies warrant a new type of solution.

Artificial intelligence has revolutionized the process of seeking answers from a knowledge base which is quite helpful in malware analysis. The study examines how such available tools in AI can be used to find answers to understanding code and using it to find solutions which was traditionally not possible.

It also tries to create a new better tool in python to harness the power of Ai with traditional search while examining code with the aim of bettering the intelligence on any code and to develop an easier workflow for use by Malware Analysts.

The tool contains automated malware analysis, Ai based search, list of tools unique to every organisation , AI prompts for better research etc.

It also discusses the creation of a Malware Analysis tool with Artificial intelligence based on curated data and deep learning so that newer malware could easily be deciphered with the Quantum computing technology that are soon going to be available. It list current shortcoming and a future expectations from existing knowledge of AI in 2023.

The paper believes that Artificial Intelligence when applied with a specific richer curated dataset and deep learning can yield the most correct malware analysis and also give solutions.

Acknowledgements

While it has been an arduous journey to learn programming and explore Machine Learning, the effort has been worthwhile as it opened up many avenues of understanding in the Cybersecurity field to me. I came across many new concepts and techniques related to the present technologies which ordinarily would not have happened unless I had not been enrolled in this course. This study also left me enriched with a better understanding of how to create the new IT infrastructure of servers, software and security by ensuring maximum cybersecurity due to the new understanding of Malware Analysis.

Equally important has been my discussion with professors and colleagues of the highest prominence whose experienced words made me realize the hype from reality in the computing world. My experience over all these years concentrated on building IT infrastructure and commissioning IT projects of various sizes. It also impressed me the fact why certain audit requirements are in place and to better appreciate the standards in IT. The project work made all my understanding come together leaving me confident of a better future and not be much worried about Artificial intelligence or employment. I am now rather keen to explore it more confidently and use it for good purposes, which will give solutions to organizations' problems.

Finally, I thank my wife who has taken a leap of faith and assisted me to take a bold decision to leave a healthy job for higher education. Taking a decision in my late 40's with a kid in college was not easy but her unshakable faith in me helped me reach those goals and coveted master's which was one of my lifelong dreams.

I also thank my father who dreamt of sending me for higher education but passed away recently, who instilled in me a love for education and the training that he gave me in all fields. His interaction with engineers at Bhabha atomic centre was relayed back to me which years ago introduced unheard concept of foldable television and many such technologies.

Also, I would like to thank all my family, friends and best wishes who have cheered me on through this difficult phase of my life.

Finally, I thank everyone who contributed indirectly (those bloggers, YouTube channels and websites) which helped me understand more of the subject that I was researching on.

Thanks & Regards

Areef Ismail RAKHANGI

TABLE OF CONTENTS

Contents

Chapter 1. Introduction

Computer programming in recent decades has played a significant role in optimizing various aspects of human life and automating processes worldwide. The diligent efforts of programmers in problem-solving have greatly benefited humanity, particularly in areas such as communication, automation, information access, healthcare, entertainment, business, finance, transportation, education, and scientific research.

However, alongside these positive advancements, there have been individuals with malicious intent who have developed software tools and techniques that pose security threats, economic consequences, privacy invasions, disruptions of services, propagation mechanisms, botnets, Advanced Persistent Threats (APTs), ransomware, and cybersecurity issues for the computer industry.

The software code written by these malicious actors, whether for amusement or causing harm, is commonly referred to as malware. Malware encompasses any malicious software or code designed to engage in activities such as theft, destruction, gaining unauthorized access, or any other harmful actions without the knowledge or authorization of the system owner, user, or relevant authority.

1. Aims and Objectives

The whole aim of the dissertation is about answering the question
"**Will Artificial Intelligence solve the challenge of finding solutions to malwares**?"
The objective of the study are

a) Understand in brief how malwares function.
b) Recommend a method of securing organization by understanding the source code and impact it could potentially have on the organisation.
c) Find out what are the current strategies that are in place to nullify malware attack and find effective strategy from a Chief Information Security Officer (CISO) perspective.
d) Look at the actual code of the malware and if its found decipher it to decide whether it was a specially targeted attack or a general attack.

e) Develop a strategy to help identify network and system weaknesses to mitigate the risk and safeguard key asset.
f) The study may not be just contained to the malware code as such but also get into other spaces of Cybersecurity as a whole.

The study started with the above stated ambitious targets and proceeded to answering those questions.
To meet the object certain concepts need to be revisited to have a wholistic approach to the issue. First and foremost let us understand basically the following.

2. What is software?

An operating system (OS) comprises of various software components that perform a wide range of tasks and the OS acts as an intermediary between a computers hardware and users applications. These software binaries, bundled with the operating system, are designed to serve valuable functions. Each new version either includes newer binary or removes existing binaries with an aim to provide some newer functionality or targeted use with specific types of hardware.

Multiple operating systems are available, including Windows, Linux, macOS, Unix, and various derivatives of these core operating systems. They can be broadly categorized as single-user or multi-user systems, tailored to run on general-purpose hardware or specialized machines with specific capabilities. Their design aligns with the computational requirements of the tasks they handle.
Operating systems can be either closed source, meaning their original code is not publicly released and is closely guarded by the vendor, or open source, where their source code is freely available for public modification and compilation.

Many of these software programs may have vulnerabilities that are discovered accidentally by individuals worldwide or by checking a operating systems components. When a company acknowledges its flaws, it takes steps to rectify them and issues patches to resolve the issues.

For unknown flaws it depends on the public or customers to report it so that an effective patch can be worked out and released in spite of all the quality control in software design.

Every operating system includes a shared library, allowing multiple programs to access it simultaneously, thereby conserving memory space required by the system. Since many programs require access to similar data, it is inefficient for each computer processor service to initiate its own programs to retrieve identical data.

In the case of Windows operating systems, these shared libraries are known as Dynamic Link Libraries (DLLs), while similar functionality is provided by Shared Libraries on the Linux platform and other operating systems. This approach conserves memory and enables multiple programs to utilize the same code and data without replicating the process for each executable in memory While this is resourceful yet it also poses a potential risk, as any process can request such data, whether with malicious intent or not and also can crash the operating system leaving the other application in a hanged state.

Operating systems are always unique to the state of hardware and software development of their time. When a newer hardware is available the relevant software has to be recompiled to ensure its efficiency and include the newer drivers for the new hardware.

Operating systems manage the Abstraction of Hardware, Process Management, Memory Management, File System Services, Device Drivers, Input/Output Services, User and Security Management, Networking Services, Interprocess Communication (IPC), Error Handling and Exception Handling, Resource Allocation, User Interface Services, etc. as every software is not be in a position to handle these complexities on its own. These are basically the functions of an operating system.

Any weakness in any of the component leads to a compromise which is called as vulnerability. These vulnerabilities are either well known or they may not be known but not in the public domain. Although companies do try to mitigate such vulnerabilities yet its impossible to resolve all the issue at one since some vulnerabilities are by design a function or feature of a program that have a legitimate purpose but has been wrongly exploited for a bad use. Malicious actors or

criminals on the other hand are always on the lookout for such weakness to profit from such vulnerabilities and exploit them to their best of advantage or fun.

The CISO of a company is entrusted with protecting an organisations IT structure and its his responsibility to ensure that every aspect of cybersecurity for the organisation be looked into and close the gap where there exists vulnerabilities.

As the field of cybersecurity deals with managing and mitigating these risks and provide solutions to such code vulnerability the CISO must ensure that the code that runs in the business must be properly vetted before putting it to production. He should ensure that properly authorized code is only run in the company environment and nothing is left to chance. Today we see a large implementation of technologies which allow only certain codes or application that can be run by only certain users even if the software is already installed on the system. Large companies like Kaspersky and others give administrators the power to white list application for certain users on specific system to be able to use certain software only.

Increasingly this is the trend in many platforms as the number of attacks are increasing day by day and organisations want smooth functioning instead of cutting edge software implementation. A cautious approach is taken to software both in terms of licensing and usage. Unlicensed or unauthorized applications are not being allowed to be run or even executed. But that is not the case with every individual or company.

There are software code which are dangerous and often intrude the systems and are called Malware.

3. What are Malwares and what are they used for?

Malware is malicious code, and its purpose is to cause harm or disruption to systems. It is often used for ransom attacks, DDoS attacks, data theft, cryptocurrency mining, law enforcement, espionage, and more. Interestingly malwares are classified into various categories based on the type of activity they perform. Below are the kind of malwares and their short definition:

a) Virus – These malicious programs inject their code into other files, spreading it to other hosts on the network or modifying copies attached to hosts or files. They are typically transmitted through email attachments and network files.

b) Trojan – Trojans appear as harmless applications, such as games or images, but carry out malicious activities in the background, stealing sensitive information like credentials. Remote Access Trojans (RATs) may leave a backdoor open for remote access and screen monitoring.
c) Ransomware – Ransomware demands payment in cryptocurrencies to unlock data it has encrypted. Payments provide the decryption key, and cryptocurrencies make tracing difficult.
d) Worm – Worms replicate rapidly, consuming bandwidth, memory, or disk space, and spread to other systems on the network.
e) Rootkit – Rootkits provide a toolkit of tools for controlling infected systems and executing various activities. They can be used to launch attacks on other systems, potentially making it seem like the system owner is responsible.
f) Adware – Adware displays targeted ads to users without their consent and gathers personal information like browsing habits or system activity.
g) Spyware – Spyware collects user data, including browsing habits and personal information entered on the system.
h) Scareware – Scareware tricks users into thinking they have a problem and prompts them to contact a certain number or send an email. Scammers may follow up with voice calls, attempting to procure remote access or extort money from the user.
i) Crypto miners – Crypto miners exploit a computer's processing power to mine cryptocurrencies without the user's knowledge or consent.

The above classification describes software based on the type of action they take. In the IT industry many a times a software needs to be analysed before putting it into production or sometimes its noted that a software is on a system although it was not authorized to install. People who are responsible to analyse software in this way are called analysts and people who specially analyse malicious software are called Malware Analyst.
The above classification of software is based on the typical actions the malware takes. Similar malware often belongs to the same family and are typically derived from the same source code with minor modifications. Its therefore clear that every malware has some set purpose around

which all its actions are based and is programmed in such a way. If a malware is a trojan it would try to connect to some server on the web where it would try to relay the stolen information. If it's a ransomware it would try to change the files on the system and also give information on how to contact the malware writer who would provide the key after payment of money for the locked data by the ransomware. A worm would only be interested in replication of its code and creating traffic on the network and affect the memory, disk space etc. The rootkit would have a programming to remain unseen to the anti malware software or other EDR solutions and infect other system by replicating itself and launch attacks. An adware would show irrelevant ads or targeted advertisement to users on a system by hijacking its browser or show popups etc.
If a system is infected by spyware then it would collect data of user, keylogging and other important information and would send it out to the attacker over the network. Other malwares like scareware try to target people into believing something which may not be the reality and get them to do certain actions like downloading a new solution for payment of money or giving out personal details and control of their machine to scamsters etc. There are also known software that try to run on a users machine and steal the power and CPU cycles by mining crypto currencies. All in all a software can do what it is programmed to do and only the software developer has complete control of what can be programmed.
To understand malware better it is important understand a little history of malwares.

4. History of Malware

Malware existed in some form or another ever since people started writing programming code. Ever since it was discovered that code could be replicated on a network without any human intervention people began writing malwares which were termed as viruses. For more than four decades now the malware industry has evolved alongside the software industry.
Several companies have been researching, developing, and providing fully automated systems to protect systems from such malware. Prominent companies like Symantec, McAfee, Kaspersky, Trend Micro, Bitdefender, Sophos, ESET, Avast, CrowdStrike, Malwarebytes, Palo Alto Networks, Cylance (BlackBerry), and others have created solutions which are used by individuals and organizations.

These solutions that they provided worked effectively until a few years ago in quickly identifying and cataloguing any malware. They achieved this through their global teams who continuously detect malware, name it, develop solutions, and then distribute their products worldwide. The competition between such company is high and there is a constant race to be the first to market a solution. Highly specialized teams and individuals work on such team and are knowledgeable about every aspect of malware. They analyse each new reported malware and these human analysts generate a lot of intelligence on any new code . Each team works on a specific aspect of addressing a specific vulnerability which is known to have been exploited by the malware.
They also work with Subject matter experts (SME) in the IT industry to create effective solutions to prevent infections or develop technical countermeasures.

Most programs from such companies installed on computers receive updates on an hourly, daily, weekly, or monthly basis, or on-demand. These updates include known signature checksums (CRC). Although their effectiveness has diminished in recent years. (James Scott, 2017) yet that is so far the only effective way to quickly recognize existing malwares. Although newer methods of detection are always in the offering which claim to limit the impact of malware yet, not all of the solutions are equal as they have varying degree of research and development.
Although much progress has been made yet the malware criminals are not far behind and have kept themselves abreast of all the counter measures taken by companies.

Several governments have also used malware as tools to effectively gather or nuke enemies or enemy state to protect their own national interest and anti-terrorism activities. While the use may be controversial in many cases yet it's a well-known fact that such uses exist.
Interestingly laws regarding backdoors to operating systems and encryption exist which make it compulsory for software or any other company to abide by it. Export control laws of many countries dictate that the government should have access to technologies when requested or ordered.
It suffices to say that Encryption & decryption laws, backdoors in operating system, disclosure laws, International agreements between countries, users privacy and industry led initiative are part of the mix of the problem and a very lengthy topic to deal with in the first place.

All the more government can call software developers, software companies etc to answer the question related to their product, methodology or any other aspect if it affects the government in some way.

The latest hearing was majorly in the case of Sam Altman to the house of congress for a senate hearing on Artificial Intelligence regulation in (Kang, 2023) where he had to explain to the committee of common politicians the details of AI in openai.com under oath.

Malwares on the other hand have caused havoc in many countries and no government has been able to deal with catching the criminals effective today. The technology used in these attacks allow the criminals to operate remotely and profit as various international issues of jurisdiction, etc. create a roadblock in catching hold of these criminals. Many countries do have agreements between them for extradition but they too are mired up in delay and controversy due to various law, money and politics.

In the last few years when the whole world was affected with corona virus and mobility to office was affected, companies had to allow remote ways of working for people from home in order to provide goods and services. The companies who had the technical capabilities rushed to this opportunity with half baked software and strategy. Although they were successful at allowing people to communicate and use their office computers yet they performed poorly on the cybersecurity aspects.

Hackers took advantage of this situation and hundreds of malicious codes emerged.

Incidents like “Zoom Bombing” occurred when unwanted visitors or hackers disrupted a Zoom meeting by Zoom hacking. (Ovide, 2023) and this can happen even today as the vulnerability still exists. Many such incidents have been reported such as SolarWinds, NHS attacks etc.

Having said that, even with all the attacks we find that in the last few years, there has been a dramatic shift with accessing to better forms of secure communication, cloud-based computing resources, co-operation between countries, technology transfer and assistance etc. To really understand Malware analysis let us look into how traditionally malware analysis is done by a Malware Analyst.

5. *Malware Analyst job*

Malware analysis is an important role of a malware analyst, who analyses malware, gathers information about it, and provides solutions. They are cybersecurity professional responsible for dissecting malware samples, reverse-engineering their code, and offering valuable insights based on the analysis. They traditionally rely on their judgment and experience to uncover the behaviour of malware code.

The process may involve that

1. The analyst may examine the binary file using specialized tools to analyse various components of the file. They may begin the analysis by gathering preliminary intelligence on the file by checking its CRC or other signatures against trusted antimalware company databases to begin with and utilize well known malware engines to ascertain if the file is a known malware. If it is a known malware or belongs to a known family of malware then the research benefits immensely since they get readymade report and in depth findings with their proposed patch or solution.

2. If the file proves to be an unknown malware, the analyst has to proceed with Static or Dynamic Analysis. A static analysis involves a thorough examination of the sampled file without running it. This stage can reveal crucial data, including file properties, file size, file type, file metadata, hash values, file structure, headers, sections, resources, string analysis, code examination, resource analysis, embedded payloads, obfuscation, encryption, and more.

3. In dynamic analysis, a secure, compatible system is set up with pre-existing tools to observe the malware's behaviour upon execution. This includes monitoring system calls, running services, memory operations, disk operations, temporary file creation, event logs, network traffic flow, encryption, software commands, and more.

The analyst utilizes the available tools to compile a detailed report, documenting important elements as discussed above. This report serves as a reference for further analysis or specialist case reviews. In the past, most malware were typically examined statically for its content or

detonated on a separate system and observed . After careful observation, resolutions were attempted and tested in consultation with SME.

4. Traditional Method of Malware Analysis

Malware analysis involves identifying how it operates, its functionality, its potential impact, and the aftermath it leaves on the system. Any Malware analyst typically performs the below steps in the following order.

a. Identifying the file responsible for suspicious activity.
b. Collect a sample for malware analysis.
c. Perform an analysis on the file, either through Static Analysis or Dynamic Analysis.
d. Attempt to identify the code, either from the compiled code or directly if the code is available (Reverse engineering).
e. Make sense of the code and identify any vulnerabilities exploited by the malware.
f. Determine the malware's behaviour, such as its use of persistence mechanisms, payload analysis, or anti-aliasing techniques.
g. Prepare documentation and a report for further analysis by specialists responsible for other parts of the network or machines.
h. Collaborate with other specialists, vendors or subject matter experts as needed.
i. Develop an incident response plan if malware is active within the organization. This plan includes remediation, creating and installing patches, and enhancing defence against similar threats.
j. Establish a protection and prevention plan for updating malware solutions, firewall solutions, IPS or IDS systems, operating system patches, software patches, close ports and any other aspects related to system or the network.

Chapter 2. Literature Review

For this project a number of studies were researched into. Below listed are some of the studies which were looked into

a) “An Overview of Artificial Intelligence Used in Malware” by Lothar Fritsch, Aws Jaber & Anis Yazidi (Lothar Fritsch, Aws Jaber & Anis Yazidi, 2023) discusses Artificial intelligence

(AI) and machine learning (ML) methods which are increasingly adopted in cyberattacks and how it supports the establishment of covert channels, as well as obfuscation of malware. It also proposed that defenders must therefore expect unconventional malware with the new sophisticated and changing features and functions. This study was used to see what developments had been reported in research paper regarding the actual use of AI in malware development which is so rampant today.

b) In 2018 Carlin Domanial (Domhnall, 2018) in his thesis "Dynamic analyses of malware" had concluded "poor and inadequate sampling of malware into datasets underpinning research to date". He concluded that static and dynamic malware analysis had been contributed well by identifying the opcode that could determine a malware up to 99%. He had mentioned that the cost of computation to be a limiting factor in complete identification of malware and placed hope on newer hardware with more cores to really give us a better chance at identifying malwares. He stressed that unsupervised learning techniques along with opcode representation should be used for a better identification. He had expressed hope that future emerging hardware could offer significant efficient embedded security within the fabric of the cloud. While this may have been true, yet human ingenuity is something nobody can predict. Even with all the favourable conditions in place no one has been able to design a system which would catch all the malwares. The most critical problem is identification of the set of actions that the malware will perform after which any solution can be derived by the specialist discipline. This was an important theory that deal with our subject matter and hence it was studied.

c) In 2023 (Amir Djenna, 2023)" Artificial Intelligence-Based Malware Detection, Analysis, and Mitigation" proposed a new systematic approach to identifying modern malwares using dynamic dep learning-based on methods combined with heuristic approaches to classify and detect newer malware families. Though their study they pointed out that anti malware companies are not able to detect all type of malwares and hence there should be some sort of artificial intelligence or another system which will detect the kind of malware that could be produced. They meant that they should analyse the new platforms of mobiles etc. and think of all the possibilities that could be. But this approach is not really possible without the collaboration of expert individuals across countries if such an exercise was to

be ever conducted as today there are not much co-operation in this field and things are done for benefit. The study has gone to length in the field of Artificial Intelligence and discussed the Malware Classifications based on Deep Learning, a Convolutional Neural Network. It is proposing a right approach since an expert in Malware Analysis can analyse code better so should the Ai be trained to mimic the behaviour of thousands of expert professionals. The key to it is the vetting of data by a body and fine tuning it.

d) A recent paper "An Attacker's Dream? Exploring the Capabilities of ChatGPT for Developing Malware "by (Pa, et al., 2023) explored the potential for malwares to be developed using Artificial intelligence on ChatGPT Open AI Playgrounds"text-davince-003" model while BARD API has not been released.

Chapter 3. The Project

1. The story of choice of this project

While doing the master's in computer Networking and Cybersecurity at London Metropolitan University a whole range of new topics were introduced which were coherent and required a deep of learning in many particular aspects of Cybersecurity.

The entire gamut of technologies that go towards protecting organizations are varied, complex, ever evolving and unique to most organizations. This masters program trained students on managing Cyber Security, switched networks, Hardening of network infrastructure, Cybercrime and Cybersecurity, Network routing protocols and network Troubleshooting. This also meant that this could lead to a career as a Chief Information Security Officer (CISO) of a company.

When the opportunity of the final project study arose, I tried to find and choose a topic which encompassed my learning and which I felt passionately about. Having worked in the IT industry for more than two decades I aim to be a CISO and understand that with great power comes great responsibility. In order to challenge myself I chose a particularly difficult topic of after several failed choices. The topic was something that was never studied by me before and it involved getting out of my comfort zone and into unchartered waters for me in areas such as Python

programming, Artificial Intelligence and related subjects which have a high amount of relation to cybersecurity.

I quickly realized that the topic was too large and unless a systematic approach is followed to problem-solving you cannot simply expect good results.

It always intrigued me as to why viruses so many viruses and malware are out there. Why until now no complete solutions have been found?. This led me to investigate into the topic of Malware Analysis. The topic was also deeply intertwined in the cybersecurity subject and many a times I had to refresh my networking knowledge to understanding how malwares worked or find solutions to many issues.

The recommended schedule for project work was a good one and I aspired to be follow it.

Regular Interaction with Supervisors was adhered to and their criticism (professor / supervisor) through consultation was taken with a pinch of salt. The work was refine and further in-depth reading was done on disparate subjects to thoroughly examine the subject matter.

It was challenging in so many ways as I had never in the last decade done such an activity.

After finalization of the topic the journey to create something started. From an unstructured idea up until reaching the subject question I took a long time. Sometime I failed and had to retract to keep focused on the topic. Sometimes not being convinced of your own writing pushes you to again write something better.

2. Research methodology

Since malware analysis is a large topic learning it would only come from doing it.

It was realized that one must do many malwares analysis for a meaningful conclusion. Also the analyst should also have good knowledge of programming languages, systems, network technologies etc.

Without a holistic idea a researcher might end up in issues. He should also have real life experience in order to understand how it would affect the system and what is it that the code is trying to achieve.

Many information that systems give out may appear to be harmless but in the hands of a criminal everything could be exploited. The old adage Prevention is better than cure appeared to be so true in the Malware analysis business.

Apart from the above I involved myself early on to go to seminars and trade show to understand what the software trend is and chanced upon meeting some of the leading vendors of these technologies. It expanded my horizon and it proved to be important in writing the dissertation. The below engineering process was followed in the dissertation

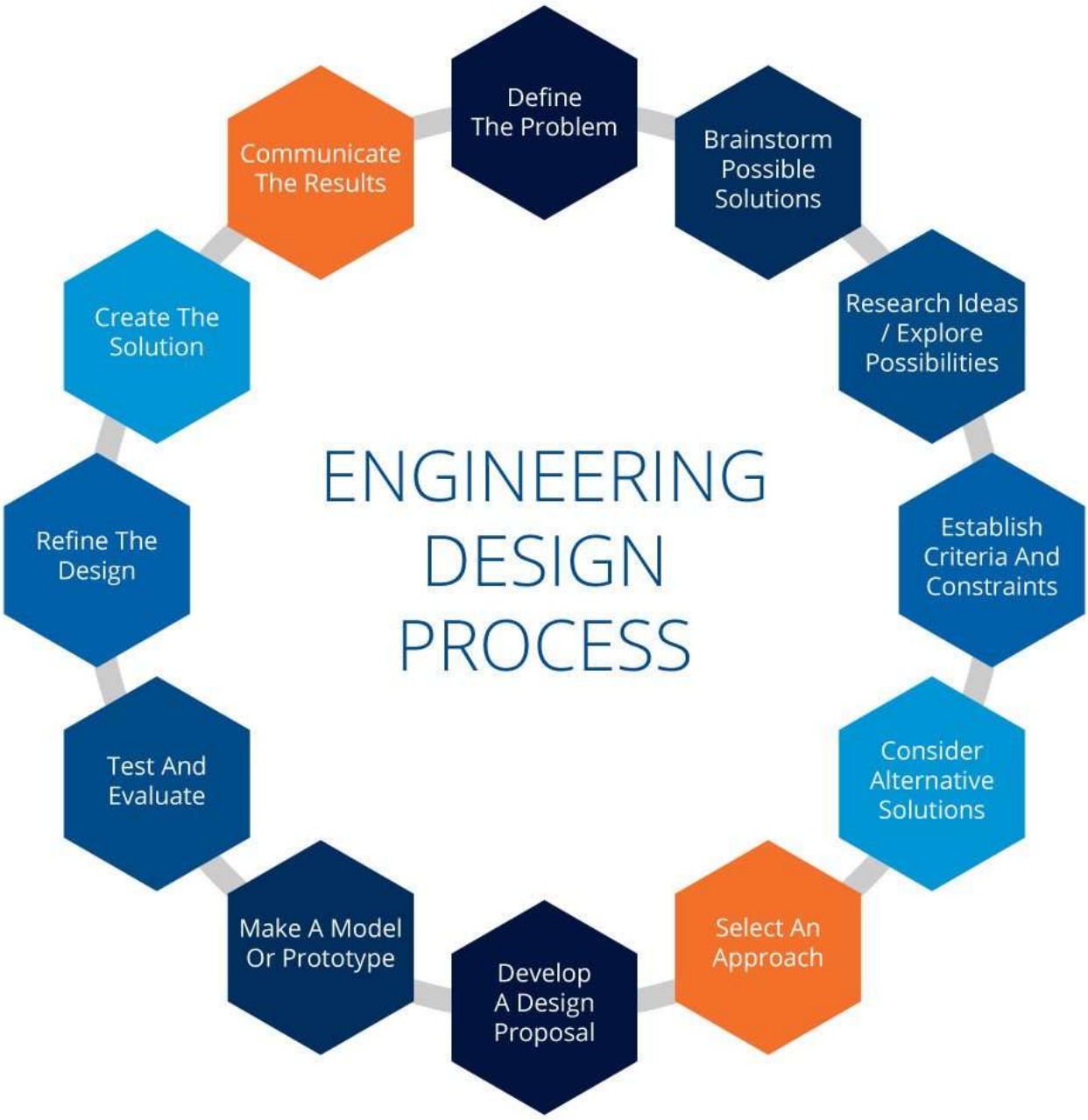

Figure 1 Credit https://www.twi-global.com/technical-knowledge/faqs/engineering-design-process

3. *Project Timeline & Chronology*

The project work started after we were assigned supervisors for the project . Below is a timeline and activities that were done during the project

- Spent the **week 1-3** to find a topic where there exists a problem that has not been solved. Malware Analysis was a topic which not only required understanding of technologies in-depth of cybersecurity but also required quite a bit of experience with tools that are currently in use. The topic was new to me and challenging. I had dealt with this on a very top level and never ventured deep into it. With the existing knowledge from the masters, I determined that it would be a good topic. Deciding on how I would write the topic was determined and I started to write about the topic.
- During **week 4-6** There were sessions with the assigned teacher and the topic proved to be too big in scope. Also, the critique about what I am going to contribute to the project of my own got me in trouble as I expected it to be just a mere research essay sort of topic where I finish with my opinion. At the same time ChatGPT was released and it swept the world by a storm. Just this one application was touted to be the solution to new things. I kept my research on this topic malware analysis on but started using ChatGPT too to understand how it works. Much to my surprise it provided a lot of answers to code during malware analysis where I could not make sense of the code, but ChatGPT could in parts. Thus, I saw a great possibility that this technology of Artificial intelligence with Machine learning could be explored further to carry out malware analysis.
- I spent **week 7-9 testing** by using ChatGPT as is and found the answers to be quite useful. I started relearning programming again in Python and making a rough model of what I was going to do. Laid the framework for the software and worked on coding the software. Also purchased a GBP 20 API license from openai.com if I was to make my own software to produce meaningful results. I made several small programs but saw that all the things that I was trying to achieve had been done in the past, which set me back. But then I realized that if a commercial implementation was to be done it could be done when actually doing a product, but the actual idea should be sufficiently explored to make malware analysis easy and build a tool that would have many future iterations as required. Wrote a good program and documented every versions code on my blog.

- **week 10-11** started to fully implement and test my product and its functionalities and write down the idea behind it.
- And **week 12-13** started finalizing the write up for the report before submission.

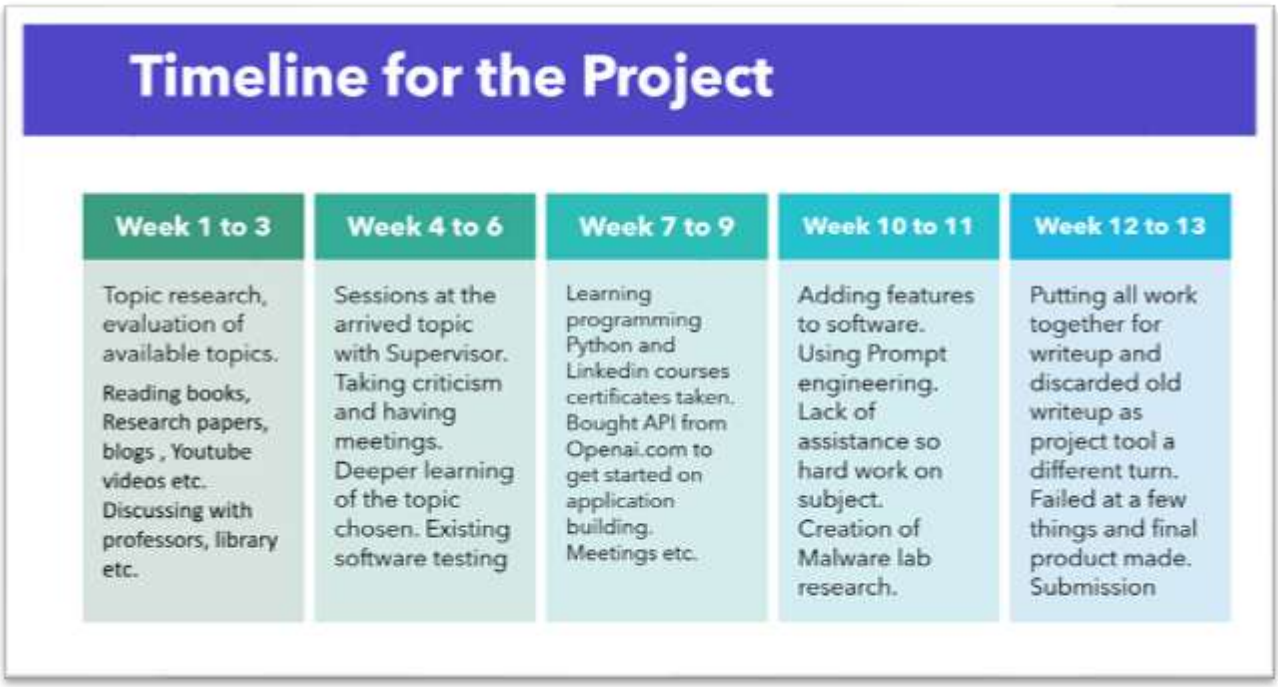

Figure 2 Timeline followed for the project

Chapter 4. The Malware Lab

Malware analysis should be performed in a safe environment which is isolated from your network and detonation of a malware should not cause problems to other computers on the network.
Thus it was important that a Malware Lab was setup.
A malware lab is generally an environment that is either virtual or a real set of non virtualised computers are used. Such a virtualised environment is called a sandbox.
It was decided to have a virtual lab (sandbox) based on VMware as a tool of choice and greater reliability and flexibility. A number of popular sandbox environments were evaluated as below

1. Sandbox environment for testing

a) Falcon Sandbox: Falcon Sandbox is a sophisticated malware analysis environment that performs both static and dynamic malware analysis. It executes the malware in a

sandboxed environment and monitors the spawned processes. The system meticulously documents the changes made to the environment, and its sensors record information about any processes triggered by the malware, as well as any quarantined files. This up-to-date malware solution provides a comprehensive report detailing the malware's characteristics, file requirements, and risk assessment. The report uses the MITRE attack indicators to classify the file as malicious, suspicious, or informational. It includes screenshots of the attack execution and a network activity tab to highlight suspicious network behaviours and connection attempts. Falcon Sandbox aids in identifying Indicators of Compromise (IOCs) on the network and other systems that may have been infected or targeted. The detailed analysis includes extracted strings and memory forensics. The discovered IOCs can be downloaded in various formats for sharing with other security programs and utilities. The intelligence report page displays any other reports related to the malware actor and offers proactive steps to combat similar vulnerabilities. This fully automated analysis process delivers actionable results in minutes, providing organizations with a more complete context for addressing security issues.

b) Sentinel One Singularity: Similar to Falcon Sandbox, Sentinel One Singularity is a sandbox solution that utilizes deep learning to identify patterns in malware behaviour. It employs natural language processing to analyse malware code, identifying keywords and phrases associated with the malware. Sentinel One Singularity continuously learns and adapts to evolving threats and attack patterns, staying ahead of the curve to offer advanced protection against the latest malware.

c) Palo Alto Networks Cortex XDR: Palo Alto Networks Cortex XDR utilizes AI for malware identification and sample investigation in response to malware attacks. The algorithms continually improve, resulting in an ever-increasing detection rate.

d) Cuckoo Sandbox: Cuckoo Sandbox is an open-source sandbox that can be easily configured on a PC with sufficient resources. It collects information about malware behaviour,

heuristics, and any changes made to the system or registry. Its automated analysis feature is valuable for analysts using their own systems.

e) Flare VM: Flare VM is a malware analysis sandbox derived from a commercial company specializing in malware analysis. This sandbox includes a range of freely available and paid tools for observing malware, performing reverse engineering, disassembly, process hacking, and more. It is user-friendly and widely used.

The Flare VM sandbox was chosen as the sandbox of choice as it had several tools which were easy to use and quite useful. Below is a list of some of the tools in FLAREVM

2. Choosing Flare VM and its tools

FLARE VM is a free and open-source security distribution based on Windows. It's specifically designed for reverse engineers, malware analysts, incident responders, and penetration testers. FLARE VM is provided as a pre-configured virtual machine (VM) that includes an extensive suite of security tools. These tools encompass debuggers, disassemblers, decompilers, static and dynamic analysis utilities, network analysis and manipulation tools, web assessment utilities, exploitation tools, vulnerability assessment applications, and more.

FLARE VM was developed by Mandiant, a renowned cybersecurity company. It operates on the Windows 10 operating system and utilizes Chocolatey, a Windows-based package management system, for tool installation and configuration.

FLARE VM is a potent resource that serves multiple purposes, such as malware analysis, security incident investigation, and penetration testing. It holds immense value for security professionals and researchers alike.

Key features of FLARE VM include:
Pre-configured with a comprehensive array of security tools
User-friendly and easy to maintain.

Open-source and community-driven

Supported by a large and active user and developer community.

For a security professional or researcher, FLARE VM is an invaluable of choice . The following are the tools in a Flare VM setup

a) Disassemblers

1. **IDA Pro**/**Ghidra** : This tool has been one of the best tools to reverse Engineer any piece of code and understand it in many perspectives. The tool developed and maintained by ***National Security Agency*** (***NSA***) which is an intelligence agency of the United States Department of Defence and has a lot of credibility and built in utilities which make the analysis of code very quick and easy. Although very costly for individual use, yet for commercial purposes it is the go-to tool. For a free tool that is almost similar in terms of functionality Ghidra is the opensource available tool which can be used but lacks the large existing database support of the NSA but is used for Malware Analysis.
2. **Ida pro** also offers a few tools as below.
    a. BinDiff which compares binary files against a known database of malware to find similarities and differences in the dissembled codes.
    b. efiXplorer that looks at UEFI firmware aspect is called code analysis and reverse-engineering.
    c. Ret Sync which Re-sync is a set of plugins that helps to synchronize a debugging session.
3. **Ghidra** has several tools which help it perform a lot of functions and following are its plugins.
    a. VTGrep: This plugin takes in the functionality offered by VirusTotal web services into GHIDRA's user interface which can help in searching strings, similar code on hex or binary level
    b. Binwalk: This plugin helps bookmark the findings for malware analysts to record and use.
    c. Yara: This Ghidra gets its search capabilities through YARA search which places a PRE_COMMENT at the location of each match to make it easy to work with.

d. Golang Renamer: Function names that are stripped away in the Go language to obscure name etc. are restored with this utility which restores the function names.
e. Daenerys: Under this interop framework you can run IDA Python scripts under Ghidra, with minor or no modification.

b) Debuggers

1. Windbg : is a Multipurpose debugger for Microsoft Windows OS to reverse engineer code of device drivers etc. in the kernel mode itself.
2. x32/x64 Debugger: This tool is a Gui tool which makes it easy to understand and do binary debugging.

c) Hex Editors

This editor allows user to view and edit the hex (hexadecimal code) of a file which is a binary representation of a sequence of hexadecimal digits (Wikipedia, 2023).

d) Hiew

This is an advanced version hex editor which can be run in disassembly mode and particularly useful for editing executable files such as COFF, PE, or ELF executables.

e) Monitors

Apart from the above tools which solely work on saved files the way in which a malware operates or affects the system is important to be monitored at run time. Various monitors are available which perform the below functions.

a) **System monitors**: They collect data about the system's resources, such as CPU usage, memory usage, and network traffic which can be used to identify malware that is consuming excessive resources or that is communicating with a remote server.
b) **Network monitors**: These monitors collect data about network traffic, such as the source and destination IP addresses, the ports used, and the data being transferred. These helps identify malware that is communicating with any remote server which is valuable information in the dissection of a malware.

c) **Process monitors**: Every processes that are running on the system, such as the process name, the process ID, and the memory usage is identified so that a rogue process can be identified when a malware is running on the system.

d) **API monitors**: API (Application programming interface) are programmatic way to connect to a server and send calls through these API from any software but is secure and protected by password so that network sniffing does not occur. These tools monitor and collect data about the calls that are made to the operating system's APIs. This data can be used to identify malware that is using the APIs to perform malicious actions.

e) **File monitors**: They observe and collect data about files that are being created, modified, or deleted. This gives vital clues in the identification malware that is creating or modifying files on the system.

f) PE studio

PEstudio tool can analyse a file without running the file. It is notable that the software combines several utilities in a GUI format for easy analysis.

PE studio is a GUI utility which is easy and can be used to decompile malware. It has the capability to convert the malware's binary code into human-readable code in assembly language. This helped in understanding how the malware worked to a certain extent and for developing defences against it.

It provided important information such as

2.1. name, size, and creation date

2.2. PE header, which contains information about the file's structure and format.

2.3. imports and exports, which list the functions that the file calls and provides.

2.4. file's resources, which include icons, strings, and other data.

2.5. strings contained inside the malware which could be used to identify the malware.

2.6. file's sections, which are divided into different parts for loading and execution.

2.7. information about whether the file is a packed version etc.

2.8. Checking the file with services like VirusTotal to check if the CRC of the file is known.

a) Ghidra

Is a software reverse engineering (SRE) framework developed by National Security Agency (NSA) for their internal use and was released to the public in March 2019. This tool is highly advanced debugging option and has a lot of scripts integration possible in Java, Python and C. It supports a lot of architectures from x86 to Risc to Sparc etc. When the tool was used the binary code could be immediately converted to C equivalent code.

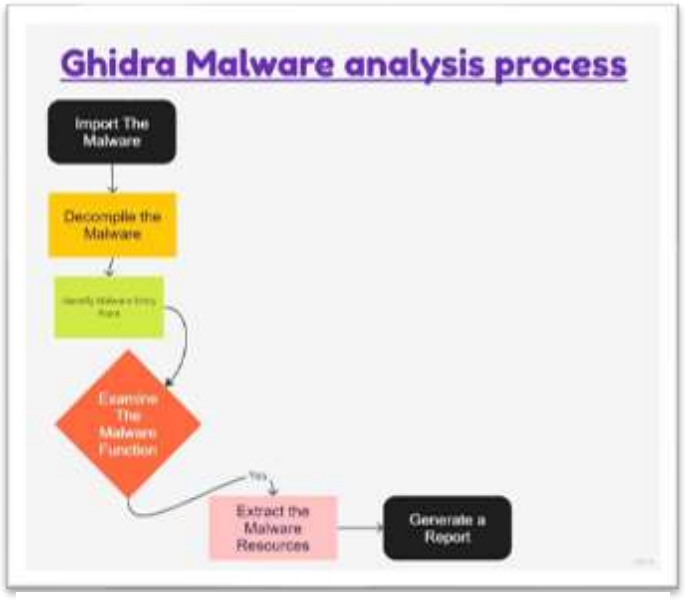

Figure 3 Process of Analysis in Ghidra

Often the code is hard to understand because it is a near guess of the assembly language from which it is derived.

The process of Ghidra is quite simple as explained in Figure 10.

Today the process is a manual one and therefor the Analyst has to be experienced and conversant with the language of the code. He must be aware or able to work out the flow of the code and its implications. Knowing these well will actually help him use all the monitors in Gidhra which monitor various part of memory, processes values etc.

Ghidra has not release a version recently and therefore we can only hope that the next version is much smarter in every sense.

The potential for incorporating new technologies such as AI is immense in these tools.

It will help a large engine break down the code properly and give proper intelligence on parts of the code. Have a large model will also train the analysis of the Ai engine to be more accurate.

Reverse engineering could potentially be expanded to such a level that where required any code could be reverse engineered to almost perfection.

Until recently code written was compiled and the resultant binary were observed documented.

The reverse engineering process tried to figure out resultant set and created their reverse engineering engines or routines.

With the power of AI this laborious and large job could be offloaded to the Ai engine and surprising results could be achieved with the right kind of logic and algorithm. Issue with it could also be tweaked. There is no reason to not believe this because we are already seeing this happen in many industries quite well.

The process of Ghidra is straightforward. Any exe or file can be imported into the software, and it almost accurately reverse engineers the code to assembly language.

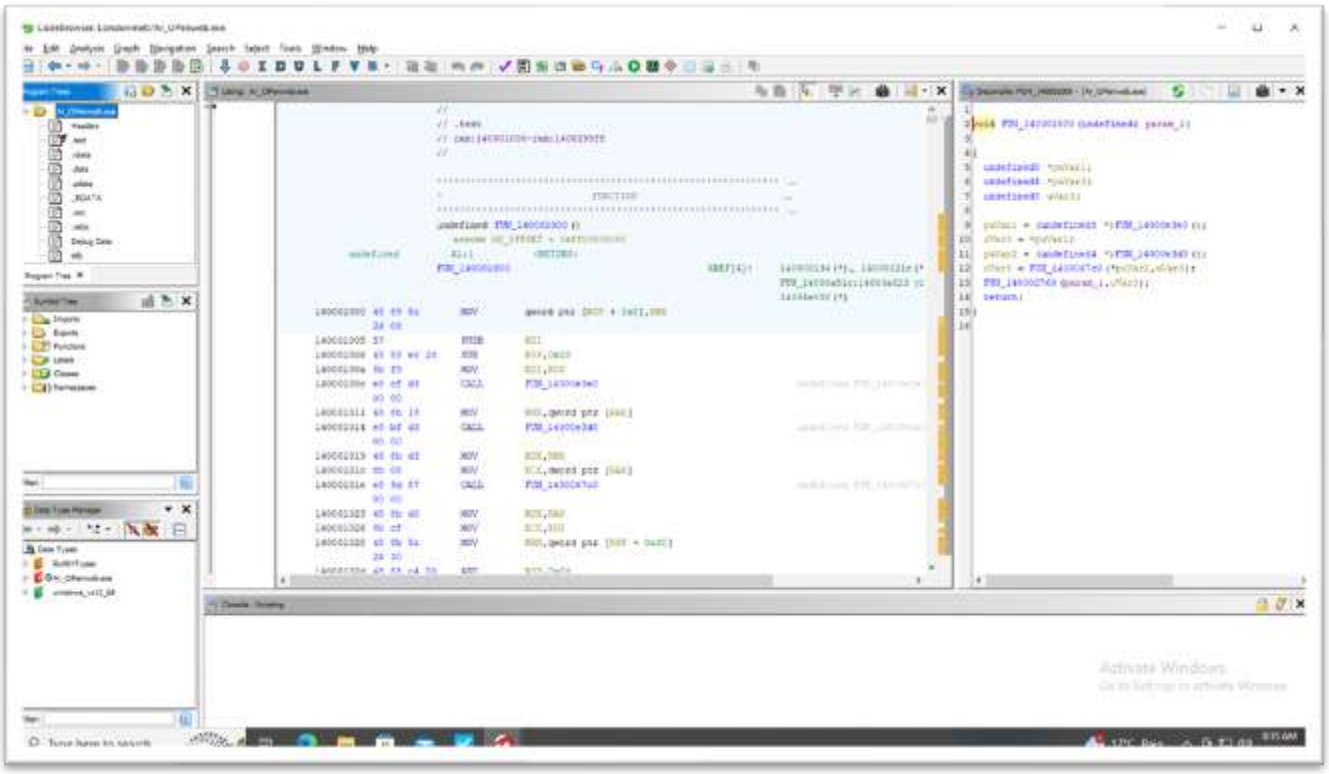

Figure 4 Opening screen of Ghidra where the sections show how a file is converted to Assembly language and its near guess code to C

Once a file has been imported in the software it provides detailed information as seen below which identify the compiler in which the program was compiled along with the count of functions data types etc.

Figure 5 Information screen from Gidhra import

Ghidra on the other hand is a very good tool that not only gives several details about the file but it also extracts a lot of other information from the file. It does the above but also gives

us a near approximate analysis of code which was written into C language.

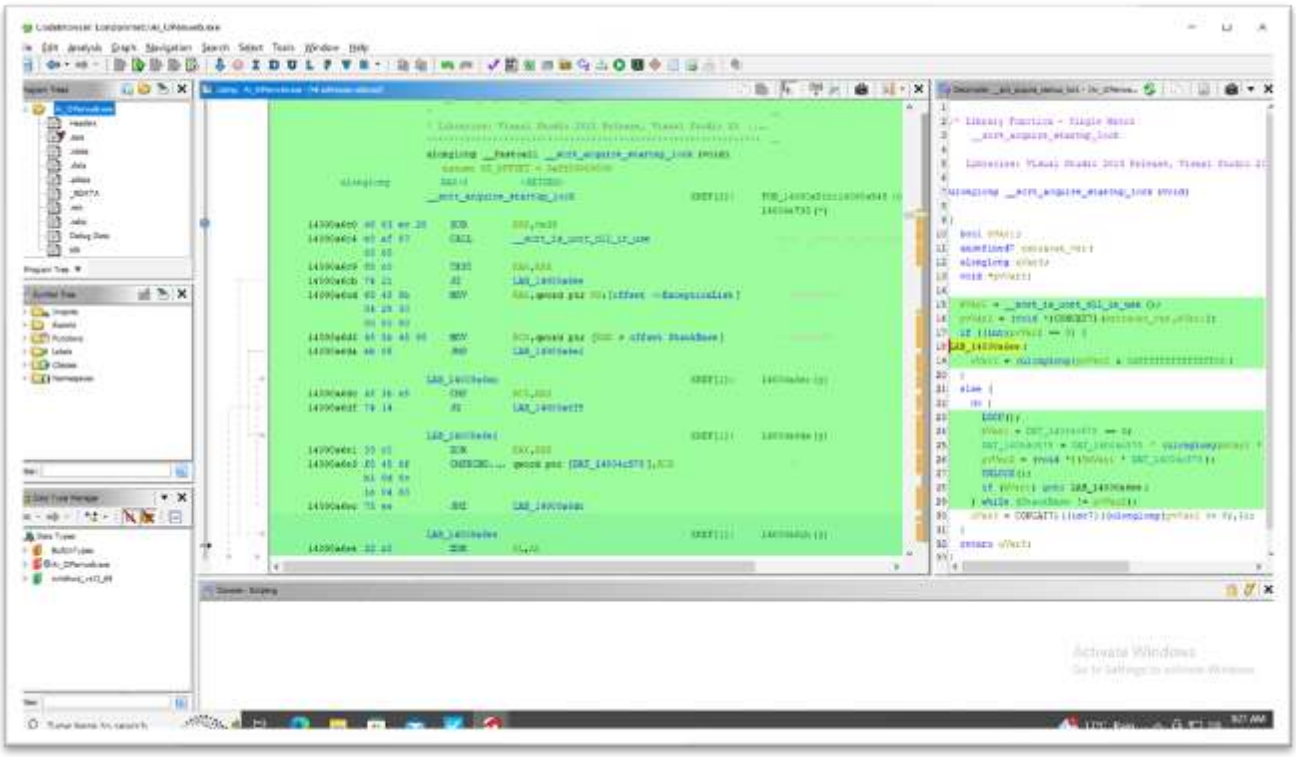

Figure 6 Ghidra when analysing a sample malware

Ghidra is also extensible. It can decompile any executable from machine language to assembly language and c instantly which helps a lot in understanding the file. While it may not be a surefire way of decompiling back to the exact source code, yet it does a good of near guess of the original source code.

It includes disassemblers, debuggers, and a few data visualizers which can be used to identify the malware features and behaviour patterns.

Ghidra also allows extension of its capabilities by allowing users to add their own analysis tools, scripts, programs etc. The customized approach allows it to be catered to specific needs.

3. *Process of installation of FLARE VM*

The installation process began with the installation of the Windows 10 operating system in a VMware Workstation virtual environment. Following the instructions provided by Flare VM, it was ensured that the environment was configured correctly and created a snapshot to revert back

if necessary. A few attempts at installing the flare VM failed due to some packages not being available. The issues were researched, and a final installation was done with most tools. Many other tools that it offered needed to be installed based on other dependency. The purpose of this setup was to get along with the process of malware analysis and understand the process and the tools generally used. It was by no means an expert analysis as this requires quite a large amount of experience at coding and decoding malwares. Not having such an experience was certainly a handicap but the determination to go into unchartered waters was there surely present.

A windows 10 operating system with the latest service pack was applied in the Virtual machine. The configuration was as below.

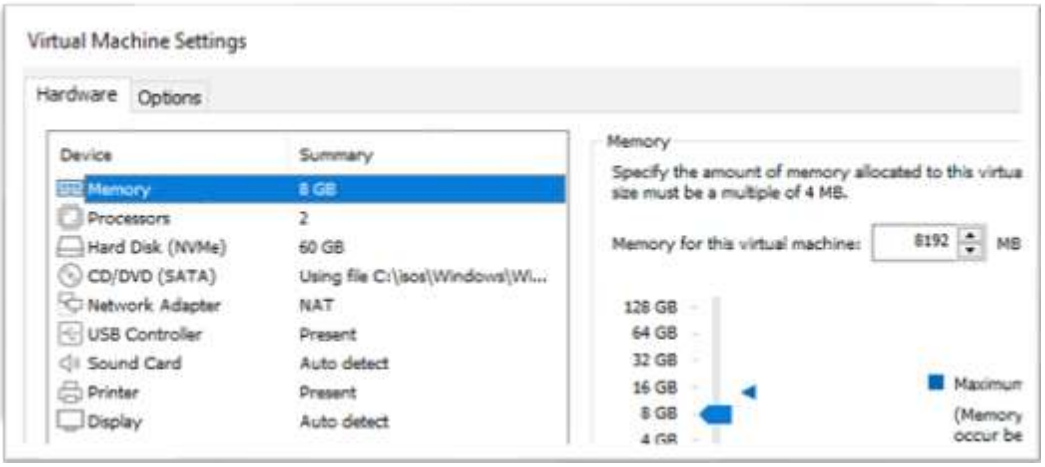

Figure 7 VMware Workstation configuration for Flare VM

Turning off windows defender was tricky and never succeeded with the normal ways but was achievable through scripting.

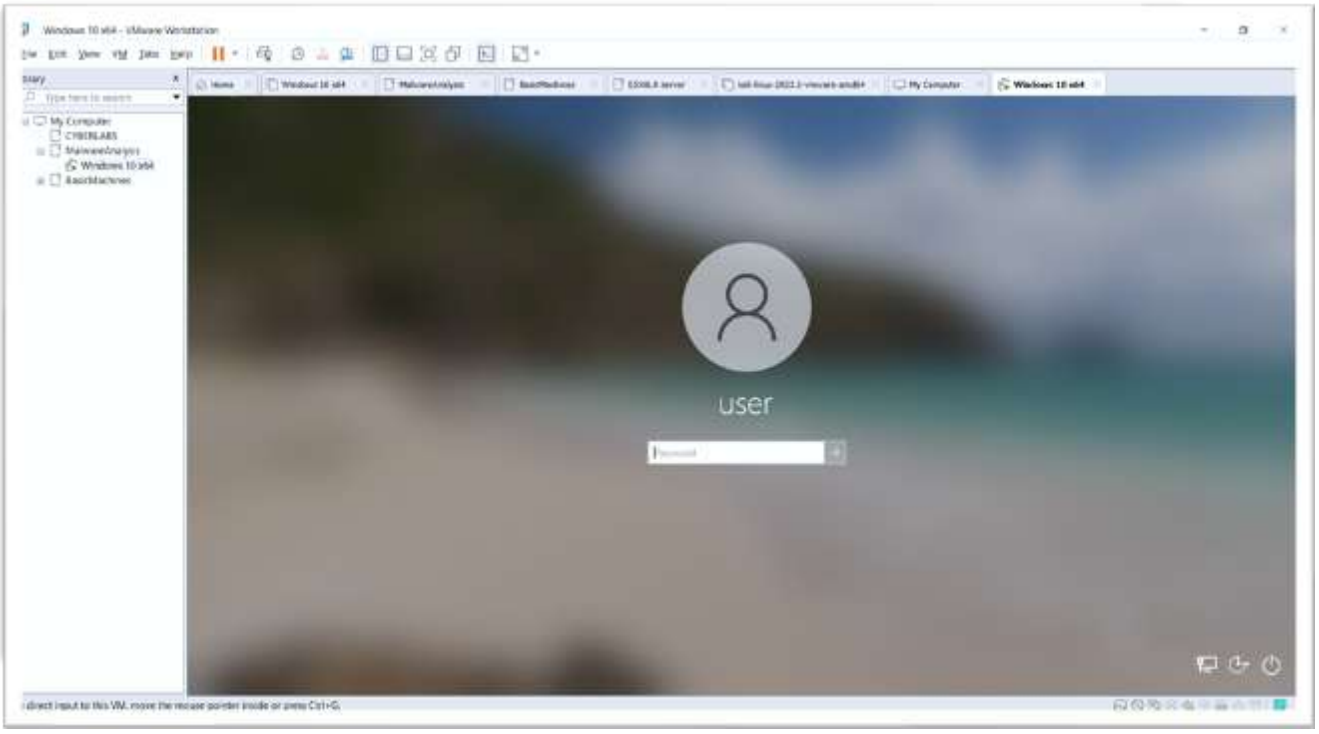

Figure 8 Installed Windows 10 operating system in VMware workstation

Next, the supplied PowerShell script was executed in administrator mode, which systematically installed all the tools mentioned above, creating an environment with all the listed tools.

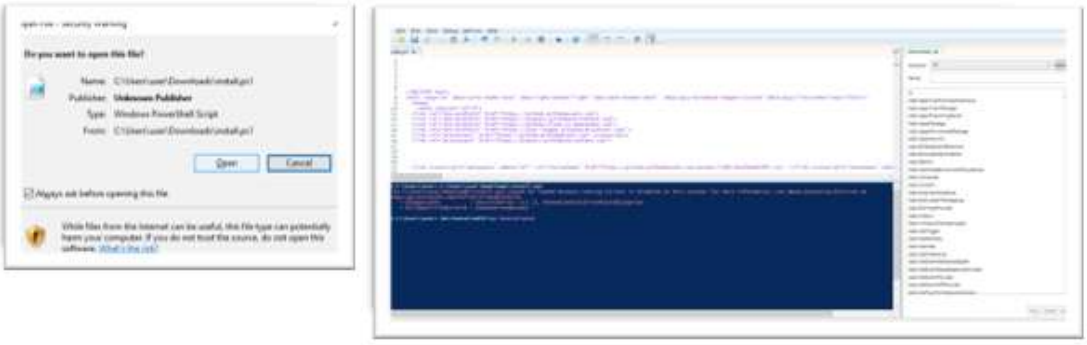

Figure 9 Powershell script run to install from FlareVM repository on GitHub

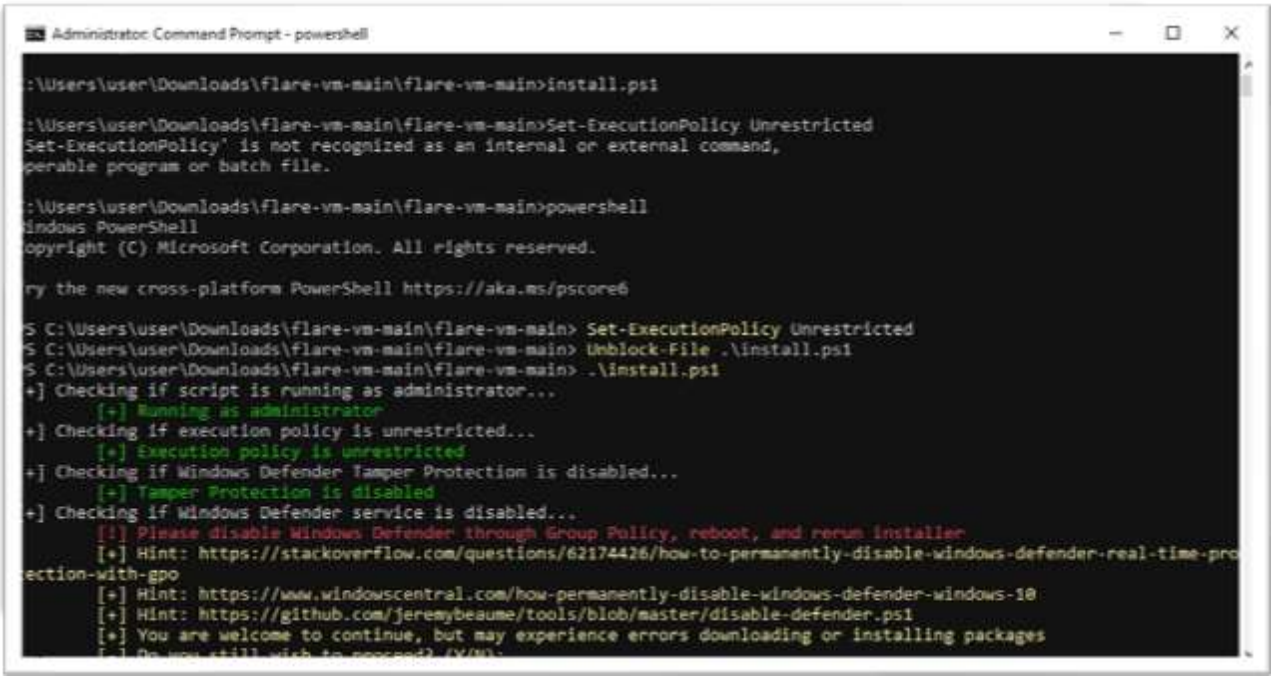

Figure 10 Install routine screen for FlareVm tool from internet

Several tools were installed, and any errors encountered were fixed, which took approximately 3 hours

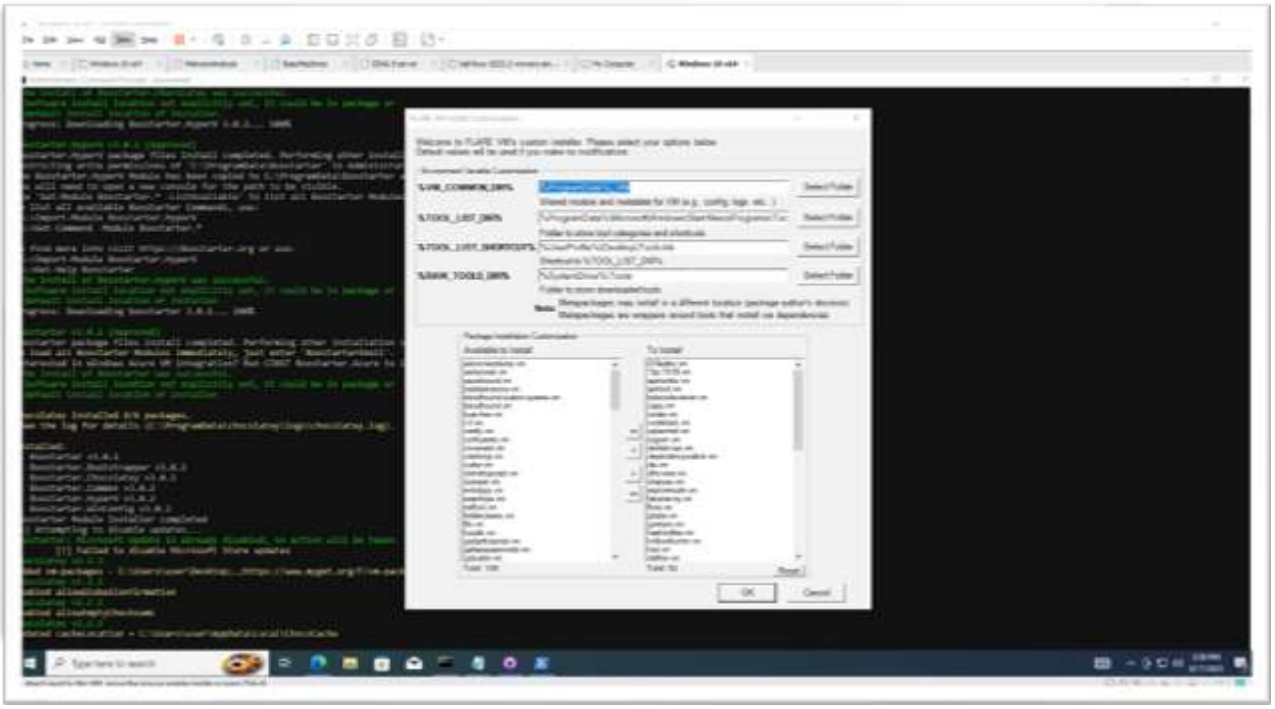

Figure 11 Choice of softwares tools that can be installed

Figure 12 Finalisation of Installation of Flare VM

Once the process was completed, all the necessary tools for malware analysis were successfully installed on the operating system.

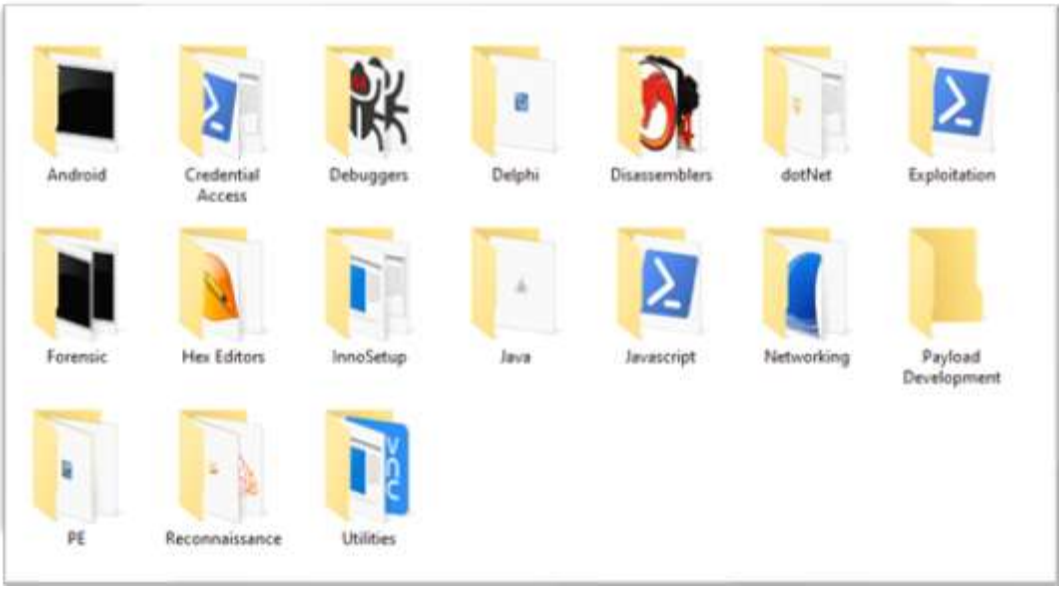

Figure 13 Directory of Tools installed for Malware Analysis

Each tool is designed to be used for a specific purpose. Let us try and understand the important components.

Chapter 5. The different processes of Malware Analysis

The recommended processes to be followed for malware analysis are qualitative and quantitative. These processes are outlined below:

1. Qualitative method of Malware Analysis

This process is a manual method where the malware is sampled, analysed, code recreated from binary through reverse engineering to identify the features and functions of the malware.
The steps involved were broadly as below.

a) Collecting the malware sample: The first step is to collect the malware sample. This can be done by downloading it from a website, opening an infected attachment, or clicking on a malicious link.
b) Analysing the malware sample: Once the malware sample has been collected, it is analysed using static and dynamic analysis techniques.
c) Classifying the malware sample: The malware sample is then classified based on its type, such as virus, worm, trojan horse, or ransomware.
d) Determining the malware's capabilities: The malware's capabilities are determined by observing its behaviour during dynamic analysis.
e) Developing a mitigation strategy: A mitigation strategy is developed to prevent malware from infecting other systems. This may involve removing the malware from the infected system, updating antivirus software, or blocking malicious websites.
f) Malware analysis is a complex and challenging process, but it is essential for protecting against malware attacks. By understanding how malware works, security researchers and analysts can develop effective defences to protect systems and networks.

For the above analysis it was important that a safe environment was created where the accidental execution of a malware would not affect the systems and multiple tools were available for analysis

2. Quantitative method of Malware Analysis

Quantitative methods in malware analysis makes use of numerical data and metrics. Every aspect of the malware which relates to malware behaviour pattern, its impact on the system or network and other characteristics are analysed. The analysis gives security professionals and researchers a complete picture of the attack pattern of the malware in an environment. In the quantitative method a safe environment is either created or an existing system is observed with the right tools and services for its action on detonation of a malware. A range of technique and Analysis are followed such as

a) Signature-based Analysis: This method involves creating unique signatures or patterns based on the characteristics of known malware. Quantitative metrics can be used to measure the similarity between the signature and a file under analysis. For example, antivirus software often uses signature-based analysis to identify and detect malware.

b) Statistical Analysis: Statistical methods are used to analyse patterns and anomalies in data. In malware analysis, statistical techniques can be applied to features extracted from malware samples to identify common patterns, detect outliers, and classify malware into categories.

c) Behavioural Analysis: Behavioural analysis focuses on observing how malware behaves when executed in a controlled environment (such as a sandbox). Quantitative metrics can be used to measure various aspects of malware behaviour, such as system calls, network communication, file system interactions, and registry modifications.

d) Entropy Analysis: Entropy is a measure of randomness in data. Malware often exhibits higher entropy in its code or data sections compared to legitimate software. Entropy analysis quantifies this randomness and can be used to detect potentially malicious files.

e) Code and API Analysis: Quantitative methods can be applied to analyse the code structure of malware and the application programming interfaces (APIs) it interacts with. Metrics can be used to identify suspicious or obfuscated code, as well as unusual API calls.

f) Network Traffic Analysis: When malware communicates with command and control servers or other malicious entities, network traffic analysis can be used to quantify the volume, frequency, and patterns of this communication. Unusual network behaviour can be an indicator of malware activity.

g) Resource Utilization Analysis: Malware often consumes system resources (CPU, memory, disk space) differently from legitimate software. Quantitative analysis of resource utilization can help identify malware based on resource usage patterns.

h) Timeline Analysis: This method involves creating a timeline of events associated with malware infections, such as when files were created or modified, processes were launched, or network connections were established. Analysing these timelines quantitatively can provide insights into the progression of an attack.

i) Machine Learning and Data Mining: Machine learning algorithms can be trained on large datasets of known malware and benign samples to build predictive models for malware detection. These models use quantitative features extracted from samples to make predictions.

j) Economic Impact Assessment: In some cases, organizations may want to quantify the economic impact of a malware infection, considering factors like data loss, downtime, and remediation costs. This can help in risk assessment and resource allocation.

k) Quantitative methods in malware analysis are valuable for automating detection, improving accuracy, and providing a more systematic understanding of the malware

landscape. However, they often complement qualitative analysis techniques, which involve human expertise and judgment to interpret results and make informed decisions.

Let us understand what are the common tools that are used in malware analysis

The tools installed were various and for the purpose of static analysis PE studio was studied with available documentation. Since some of the functionality was not clear videos on YouTube were resorted to and that helped gain good understanding of the utility. PE studio is power and below are the salient features of this utility

a) PE Studio is a free and open-source tool that can be used to analyse executable files (EXEs) on Windows systems. It provides a wealth of information about an EXE file, including its PE headers, sections, imports, exports, strings, and resources. This information can be used to identify malware and understand its behaviour. PE Studio is an amalgamation of various small utilities presented in a graphical format, making it much more accessible than running and knowing every command.
b) This software facilitated an understanding of Portable Executable (PE) files, their internal sections, and file characteristics, such as PE headers, sections, imports, exports, strings, and resources.
c) PE Headers: These headers offer information about the file's size, entry point, and version, aiding in malware identification and behaviour comprehension.
d) Sections: EXE file sections contain code, data, and resources. PE Studio allows for the viewing and analysis of these sections.
e) Imports: Import information lists the functions the file depends on, helping identify malware dependencies and its system interactions.
f) Exports: Export information lists the functions the file exports, revealing the malware's capabilities and potential uses.

g) Strings: Embedded text strings in the EXE file can be viewed and analysed using PE Studio. This data aids in understanding the malware's purpose and potentially tracking down its creators.
h) Resources: Non-code or data resources within an EXE file can also be viewed and analysed with PE Studio, assisting in understanding the malware's purpose and potential origin.

Chapter 6. The actual malware analysis process

As a newbie the first thing done after learning the above information was to actually try and test file. Therefore, to gain hands-on experience, a sample Python application was created, compiled into an EXE.
This file was sent to VirusTotal for an analysis by uploading the file to VirusTotal.

1. File submission to a malware engine

A simple submission of the file to VirusTotal gather a lot of intelligence on the file from such site. It checked the files properties etc. automatically and gave a detailed static analysis report.

Figure 14 Submission of Malware to https://www.virustotal.com/gui/home/upload

This site checks from multiple engines if the submitted file has been observed by any other anti malware product. It gives a detailed report on it. Similar process could also be done with another utility but this is an automated response from VirusTotal. Below screenshot displays the kind of report and details that can be obtained by the VirusTotal site

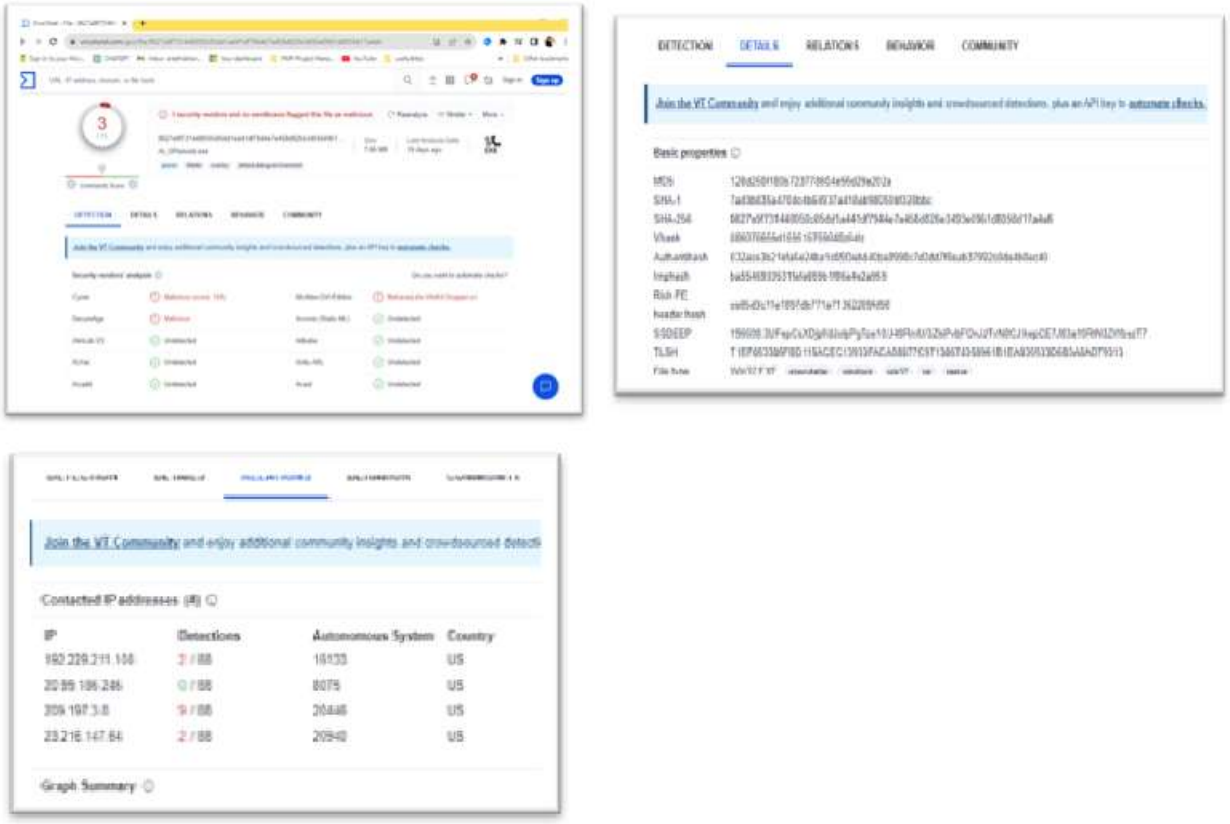

Figure 15 Various information that Virustotal returns after analysis of a file

Files detected as malwares generally have detailed information on relevant sites.

There are other site like hybrid-analysis.com and others where a file could be uploaded and similar responses could be received whether static or dynamic.

If VirusTotal does not detect it as a malicious file then the chances are pretty good that its not a malware as it uses several vendors programs to detect if the file is malicious.

This however is not a guarantee that is may not at all contain a malware. It could also be a false positive. Zero day Threats are vulnerabilities who are very new and anti-malware company may

have no information on it or identified it to be so. Chances are that the file could be using techniques to evade detection.

2. *Malware Download.*

Malwares were downloaded in the LAB from https://bazaar.abuse.ch/browse/

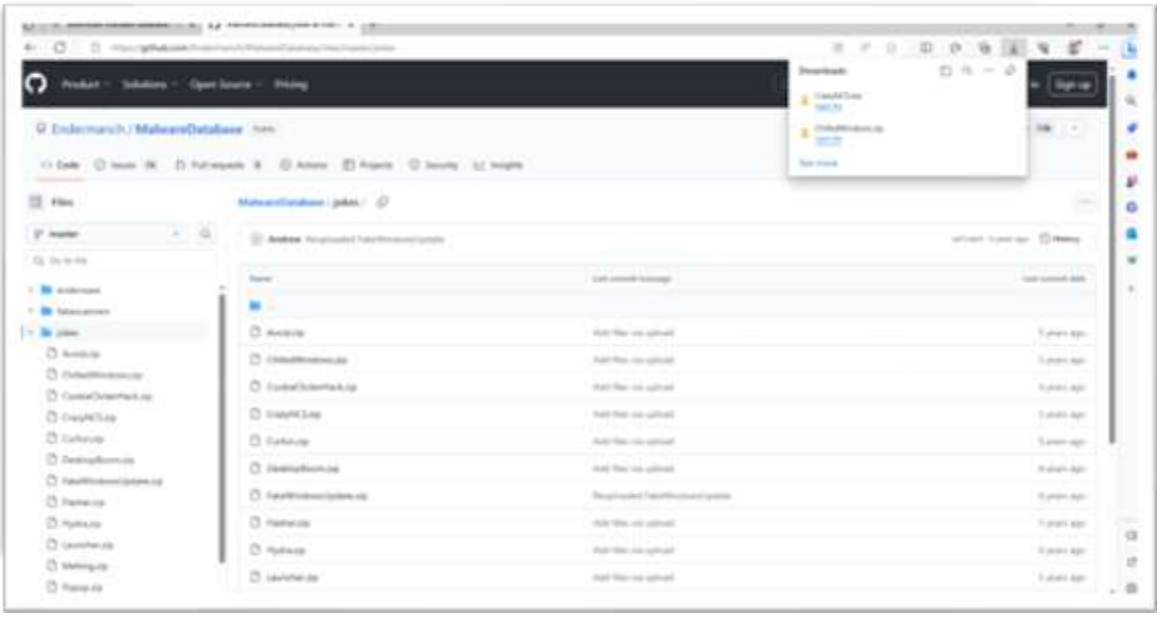

Figure 16 Download of Malware from Malware Bazar

These were downloaded onto a window 10 machine with the latest service pack and defender was purpose closed permanently in the system to carry out the analysis and allow the virus to download on the virtual machine without which windows deletes any malware immediately.

Many of the malwares were downloaded to actually test how the process of malware analysis is.

With the above Sandbox environment and installation of tools the first step was to detect whether a file is a virus or not.

3. *PE Studio*

When a file was analysed by PE studio the below detailed information about the file was received. With this utility a lot of things can be achieved in relation to a file which has been already explained in Chapter 4 in the PE studio section. The applications commercial version gives the full capability of the software while a free version is limited to what is shown below.

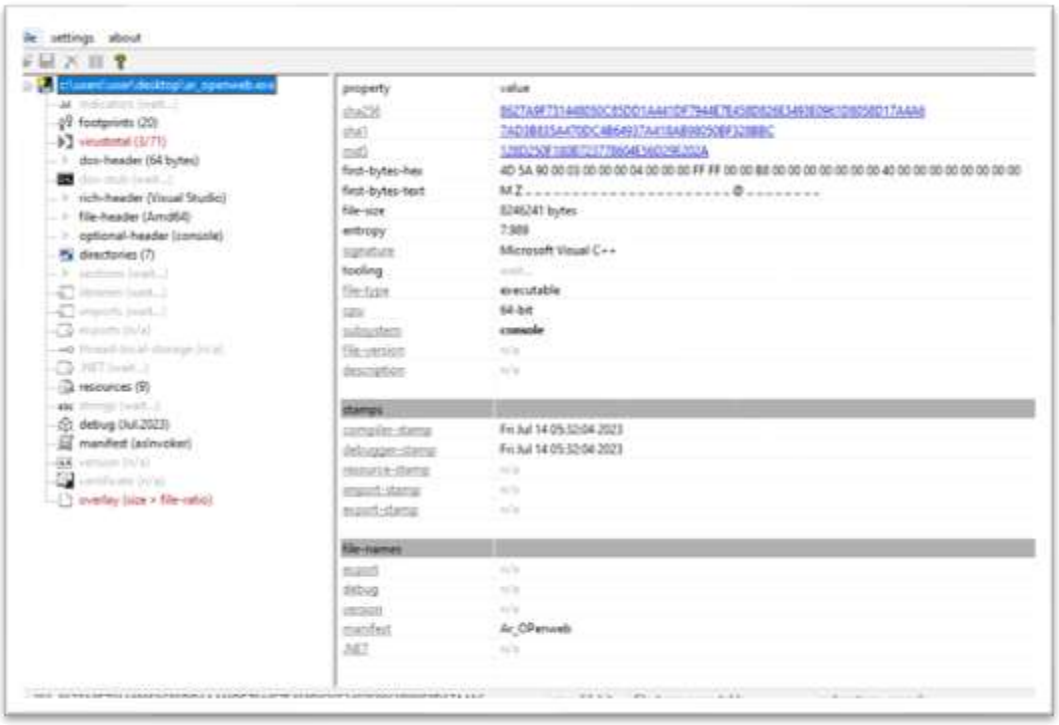

Figure 17 Initial screen of PEStudio with a file for analysis

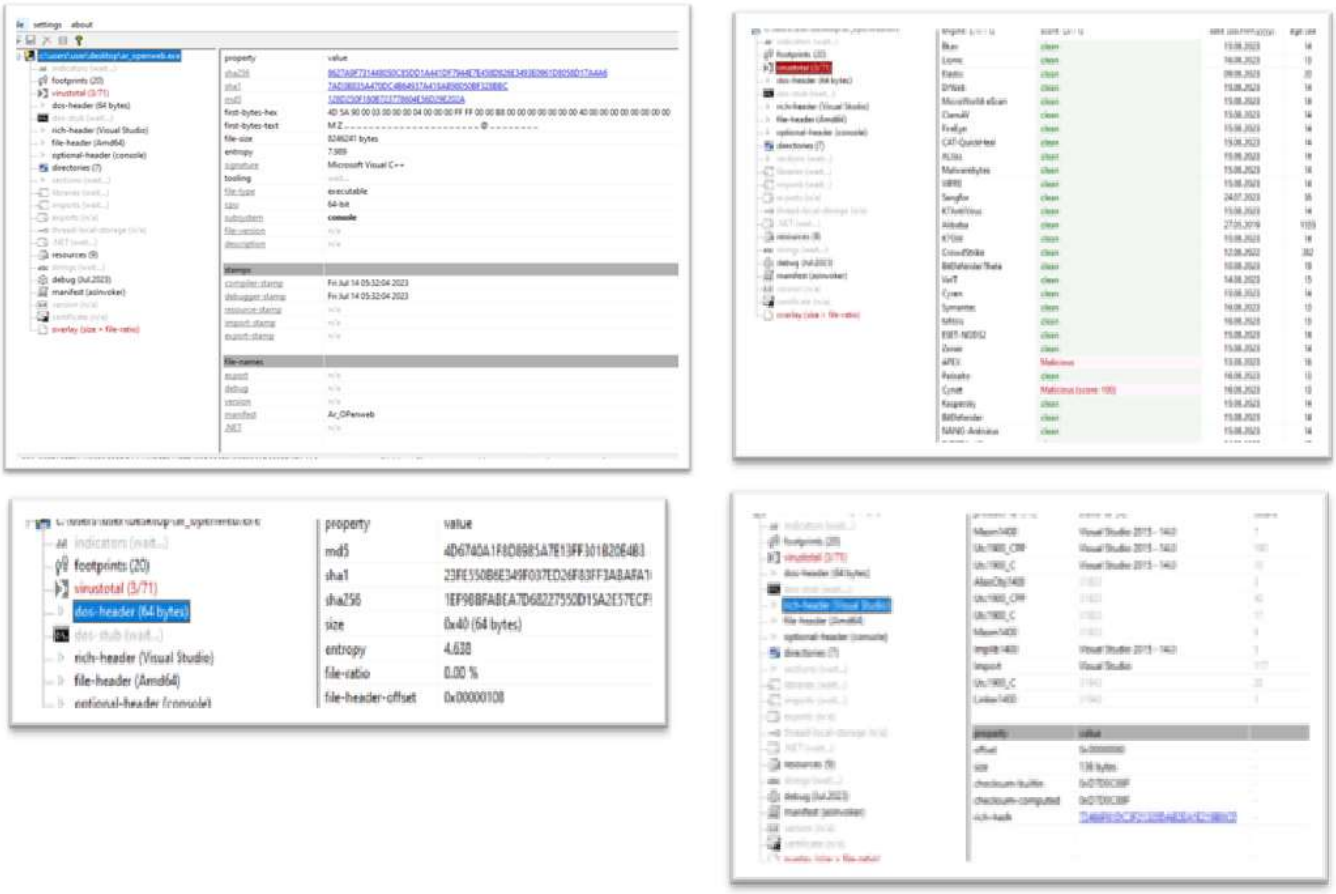

Figure 18 Various details that PEStudio provides for a file Analysis

4. *Using Ghidra*

Ghidra is a extensive tool and has lots of details which pe studio does not have. It not only analyses the binary but converts it to assembly and offers a near C equivalent.

When such tools were used to analyse a malware, it not only reveals a lot of aspect of computer programming and but also forces you to think how code actually runs. Computer code is just a set of instructions that is written in a programming language and read by the CPU to execute it one instruction at a time. The low-level code is just written in Assembly language and hard to understand or make sense of while the higher languages are an abstraction layer for a language in which it is written which is easy for humans to understand e.g., C, Java, Python etc.

The biggest problem therefore is the knowledge level of the human analyst with the language in question. If he is efficient at understanding the complete code, then he can

make quick progress. In situations where the programmer is even efficient yet to really get a summary of what the code does is really difficult as the human will have to completely read the code, understand it and decipher the code. Also, humans have a problem remembering everything they read.

The malware was then loaded into Ghidra, and its binary was analysed. Ghidra decompiles a code and produces a near equivalent code of it in C language and also in the assembly language.

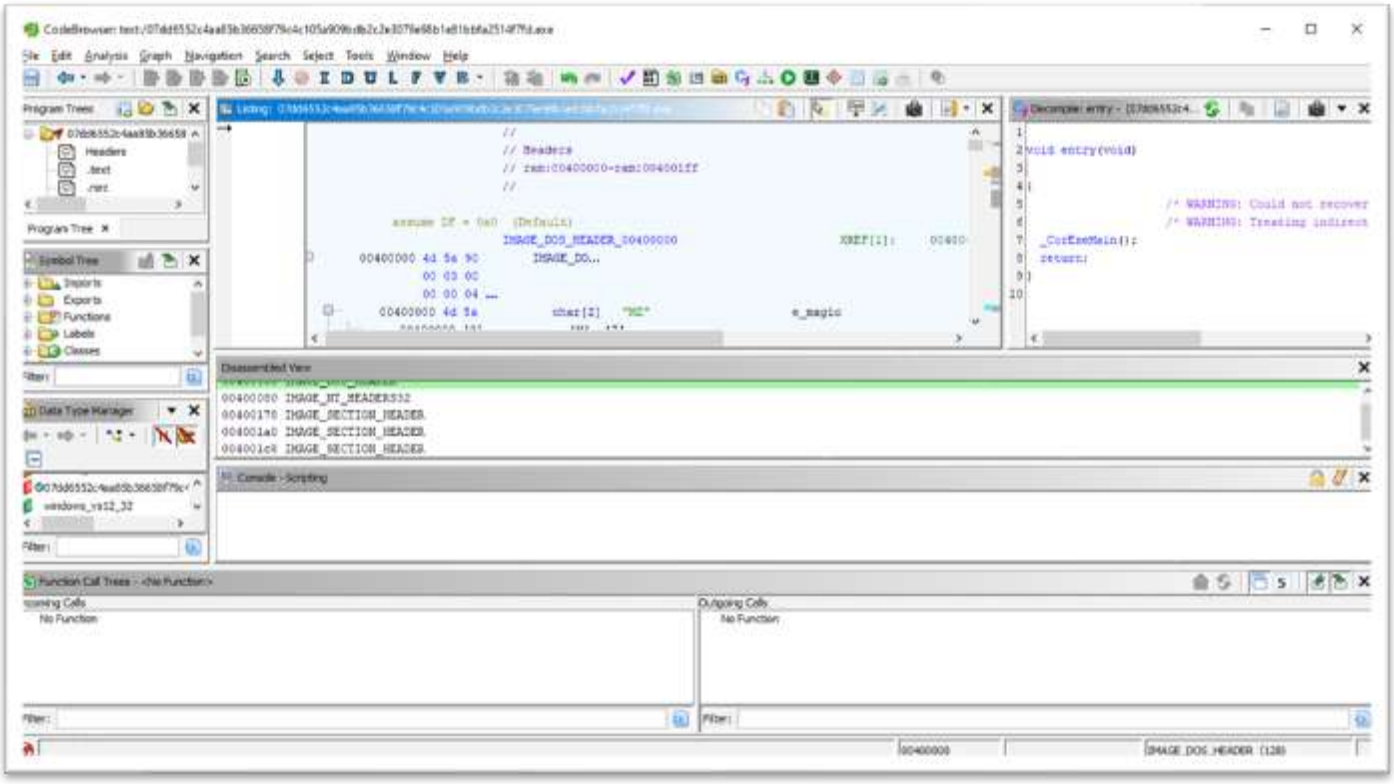

Figure 19 Ghidra initial screen when a file is examined in IT

It has several analysers built into it which help understand programming and the structure followed by the program. Understanding each of them helps you understand the different parts of the program flow.

The below is a very short explanation of what they are.

1. Dataflow Analyser: It tracks data flow throughout the program, identifies which data dependencies exist and what is the control flow of the data in the program.

2. Flowgraph Analyser: IT creates a diagrammatic representation of program control flow which helps in understanding the program's structure.
3. Instruction Analyser: It provides information on every instruction in the program. opcode, operands, and mnemonic are important sources of identifying malicious code (Domhnall, 2018)
4. Memory Analyser: The information on memory usage in the program, including the data segments, heap and stack can be obtained from this part of the program.
5. String Analyser: Important strings like IP address or a particular code their length, encoding and character length can all be obtained through this string analyser.
6. Type Analyser: The Analysis of Data types used such as integers, floats, and strings in the program is available here.
7. Variable Analyser: Every higher-level program stores data in variables and this module provides the information on their scope, its type, and its value.

and analysed in Ghidra in the isolated environment.

5. Issues with Malware Analysis

Learning Malware Analysis was challenging but it was also riddled with various problems.

1. It eased understanding the binary code of the file by converting the Binary code to assembly language and C, yet it did not break it down sufficiently enough to python etc. Specialist knowledge of C programming was required and many a times searches on the internet has to be made to find out the meaning of certain codes. Many of them could not be understood since they were only numbers. Understanding the code and studying the flow of the program requires a lot of effort even for a small program. Time has to be spent reading the code in totality and repeated guesses have to be made as to what the code means.
2. Difficulties in understanding the C code and Obfuscation of many of the codes were particularly problematic. Since Ghidra does not convert c code to python automatically and is ambiguous was not possible due to the way in which software is coded it slowly dawned that there are several things which one should know about malware analysis the kind of code one is dealing.

3. After multiple analysis it seems that Ghidra was dependable as it could at least convert binary code into C very dependable tool as it has several analysers that help in deciphering the code.
4. Ghidra is a software reverse engineering tool that was developed and released by the United States National Security Agency (NSA) to the public in March 2019 and updated in September 2021.
5. There were difficulties faced during this analysis as a few times by mistakes the malware was triggered and it brought down the system or started executing its payload.
6. The whole VM setup had to be setup up from the beginning which lost considerable time, but this was also a moment to learn what happens. Later learning the techniques of avoiding that helped create a process by which these eventualities were covered. Some of the malwares did not work in virtual atmosphere as it was probably programmed not to run in a virtual atmosphere.
7. It was also noticed that since Ghidra operated on the computer's maximum resources available Ghidra had not included bigger plugins and analysers. Whenever a file was imported it took several seconds or minutes based on the size of the program to analyse the code with the modules available.

With all the above noted issues it was only logical to venture into programming as a lot of searches on the net indicated that people use a lot of programming routines to analyse the code and each one is unique. Ghidra supported add-in scripts through the script manager which allowed python or c programs to run inside Ghidra and take benefit of the Ghidra API. It made more sense to have a software inside Ghidra than to have it running as a separate tool. But the task was easier said than done.

The important take away from analysing these two tools is that you need to be an expert programmer with experience at Malware analysis to be really be able to do some meaningful analysis and find solutions.

The solutions will be as good as the through evaluation. Solutions are pretty standard to get as specialist in every IT field can figure out based on their specialty but assessing the malware is of utmost importance without which discussion of any solution is meaningless.

Chapter 7.

1. Exploring Artificial Intelligence Tools

To do meaningful malware analysis through Artificial Intelligence the use of ChatGPT and Bard was repeatedly made which is free to use currently. It did not have any restrictions, but they actually failed to give any meaningful feedback for any malware analysis except a few. They could analyse code to a certain extent only and were very general in answering question related to a piece of code.

While ChatGPT gave better results Bard was not far behind. On repeated use of Chatbot it became clear that organizations would not waste time chatting with these Chatbot but would be interest in processes which could use the AI tools and give out clearcut answers.

A little more research into AI revealed that since these chatbots were based on all the data answers to questions were more generic and it was a hit or miss situation in many cases. It required the proper kind of prompts to be used. Most answers were general in nature.

So, unless a specific chatbot was custom made to specifically deal with Malware Analysis all the desired functionalities would not exist as the current chatbots are general purpose one.

That meant learning programming in a language that would satisfy the requirement would need to be used. In the start at an attempt was made to build something with current knowledge of Visual basic but then it turned out that the language was quite old and not fit for purpose. Trying to build out an application in HTML using CSS and other technology proved to be limit in use but was good in the interface department.

Ultimately after due research finalized that C would be the language of choice. Having studied C previously means that it needed to be fully brushed up. A few attempts at C proved that the project could be built in C but then it becomes too technical and getting help on it was a little difficult.

It was required that some simple language should be used which will both have the power of C and ease ness of basic. Python then emerged the language of choice.

Started learning python online and also from the college lectures that were available at that time. Made quick progress on the same while doing other things simultaneously.

In the short span of time, it was difficult to exhaust all the options available on the internet yet a few of them were tried.

2. Building a new tool process and considerations

The need for a new tool arose from the fact that no existing tools were able to meet the requirements of the project. A literature review found that many bloggers suggested using artificial intelligence (AI) tools, but these tools were often difficult to understand and use.

In the meantime, Microsoft released the GPT-4 model, which generated a lot of excitement in the AI community. This prompted me to take a closer look at what was possible with AI tools. I began watching YouTube videos and reading articles about how to use these tools, and I also started learning Python, a programming language that is often used for AI.

Having not used Python before and for the last several years only been scripting in visual basic and PowerShell and I needed to brush up on my skills. I spent several months learning Python from scratch, using online courses and tutorials. I also took advantage of LinkedIn Learning, which offered a variety of courses on Python and AI.

After many sleepless nights, I completed several courses, seminars and received certificates for some of them.

Figure 20 Webinar on Generative AI

Figure 21 Webinar on Quantum Computing

Saw lots of videos on how Artificial Intelligence current developments and where it was heading. Key interview of SAM Altman and Satya Nadella were a precursor to what was possible. How the future could be affected by this technology and how it was providing solutions in several fields.

3. Conceptualizing A Tool to help in Malware Analysis

After understanding Malware Analysis to a certain extent an attempt was made to try and overcome the problems faced during malware Analysis.
Further it was decided to use Artificial Intelligence extensively to find more about the binary code and see how it could be deciphered further. Examine the ways and means of how the tools or techniques could better the existing technologies.
The tools like Ghidra presented the binary code in an assembly language and c equivalent.
But it did not give any explanation in human understandable language of what the code does or attempts to do even when the code was clear cut python script.

4. Using Artificial Intelligence tools

Using Gidhra as a primary tool required a solid understanding of code primarily in C or assembly language. Most of codes only revealed codes which were anyway non legible to a human.
The use of Artificial intelligence tools like BARD from google and ChatGPT from Microsoft & OpenAI surprisingly provided good answers to questions posed to it in human language
Many a times when the code was small it correctly summarized the code in an easy to understand language. Very rarely was it totally amiss. In those rare occasions the problem seemed to be obfuscation of the code.
Another major problem that was observed was that sometimes Bard would refuse to answer the question claiming that it was just a Language model and not capable of doing some things.
Engines like ChatGPT either lost connectivity or became very slow.
When large questions were they simply refused to give an answer.
Both bard and ChatGPT are trained on a large data set. They tend to have some biases and utilize certain libraries by default to answer questions. If they are very specifically asked to change the code libraries they do change them.
Also large complex programs did not give any results because the token system is in place for both these services.
With all these problems in place and more it was decided to build a tool that would make searching malware easier and allow results to be role oriented.

Chapter 8 Building the tool for Artificial Intelligence

The first prototype was built after working with BARD and learning about code capability through some generic searches. They however proved to be inefficient and sometimes it was un responsive. Faced several roadblocks trying to search for complete program evaluation by ChatGPT and Bard.

Figure 22 Use of Bard in Malware analysis code

Changed the approach several times evaluating between building a application in HTML and then under Python or Access. Access or VB did not provide much headway as the plan was to integrate database component of office Access with it.
Then after learning partially some python started to develop a software skeleton which would be able to query OPenai.com.

Initially a search page was developed which would take the code and search it on google.com. Sometimes google may not be the best search engine and the query had to be tweaked to get a decent response. Also google severely restricts getting the first result directly and hence tried other methods.

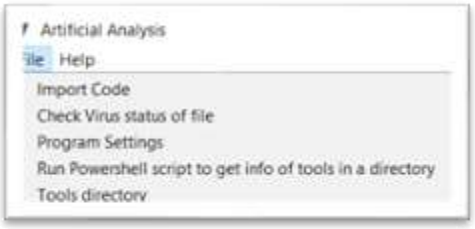

Figure 23 Menu for import of malware code, Submitting file for malware analysis, Programs settings, Running PowerShell for finding tools available for tool directory

1. Mult search Engine

To make searching quick a multi search webpage was created with quick search on multiple search engines such as Google, Bing, yahoo, GitHub, you.com, DuckDuckGo, YouTube, Yandex, Twitter etc. All at once saving user time and give variety of resources to look at for information.

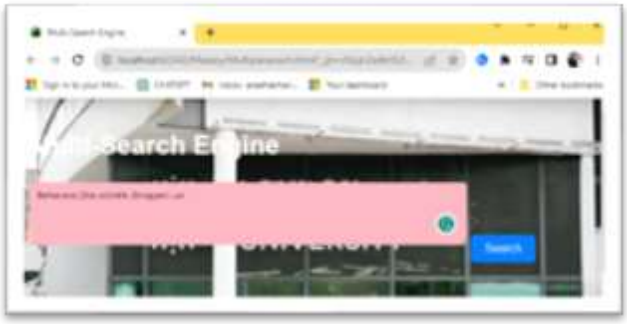

Figure 24 A Mult search Engine which allows to input lines of code and search in several sites at once

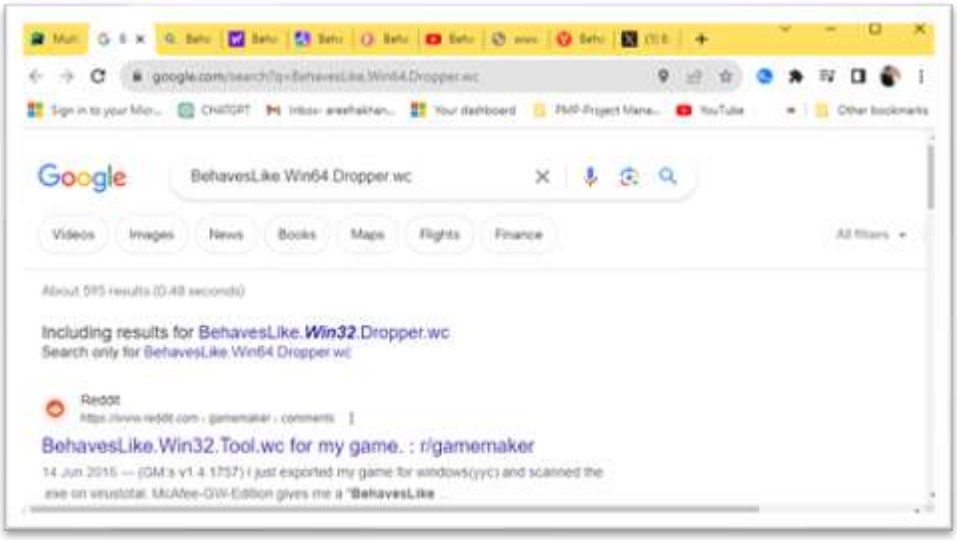

Figure 25 Tabs with requested information search

2. Prototype application

The Basic prototype application in python was completed.

Created the first version which was able to query an AI engine of ChatGPT through its API and receive a proper result based on GPT model 3.5 . Experimented with different question about the code and received helpful response as shown below.

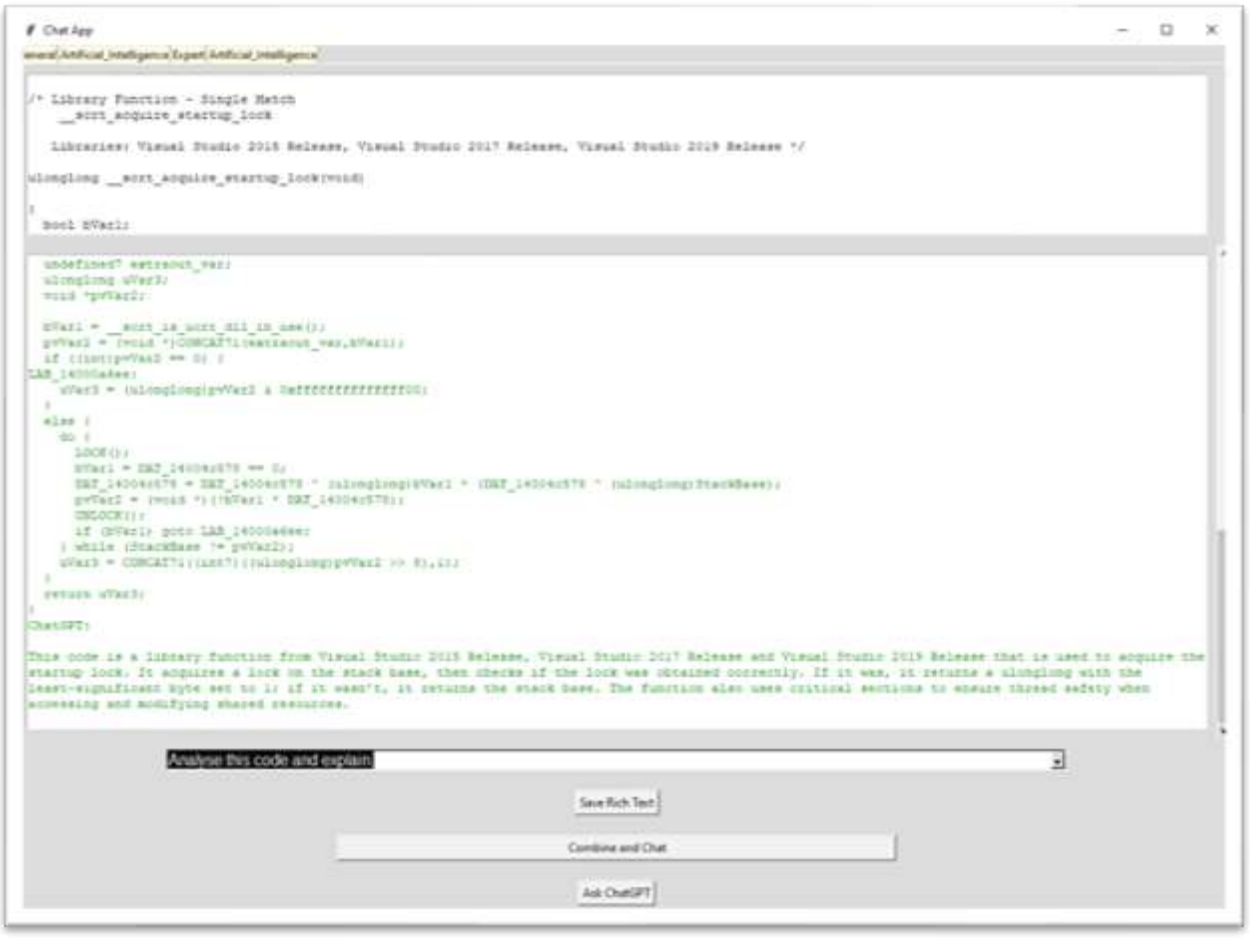

Figure 26 First look of the Malany's prototype Application

After the first version created multiple questions that were important for analysis of any code. Although the web based model was far superior in terms of storing chat, taking on a role and code formatting yet for a malware analyst it would be troublesome because he would have to cut copy paste and move back and forth from the application etc.

3. Developing a real Application for Malware Analysis

Purchased a $20 subscription per month to the API from openai.com which allowed use of API to query the ChatGPT engine. Without the API it's impossible to use it in your own program. Googles API was not released and hence did not incorporate it in the program.
Hypothesized ideas which would make application effective and improved its workflow over time.
Incorporated a lot of features which were possible.
A step by step approach was taken

4. A multi site submission for Automated Malware Analysis

Developed a python routine to browse a file and upload it to VirusTotal and receive a response automatically if the file is malicious or not.

Figure 27 Routine to submit file for Malware Analysis

Once a reply is positive then a detailed report of the analysis is available on VirusTotal is available which can be checked for further information as shown in chapter 6 point 4.
Submission to hybrid analysis did not work but there exists a possibility to do that.

5. Most Frequently asked question bank

The process that is generally followed by Analysts was attempted to be recreated. A question bank was developed which would be used specifically related for the analysis which query ChatGPT to give useful results. It was implemented with a drop down and had the option to be able to add questions to it.
Questions like
Analyse this code and explain.
Does this code appear to be a malware?

In which Language has the code been written?

Identify the codes purpose.

What are its capabilities?

How does the malware spread?

What are the weaknesses of this code?

How can this malware be detected and prevented?

What is the malware C&C (command and control) server?

What is the malware payload?

What are the malware evasion techniques?

What are the malware anti-analysis techniques?

Extracts features from the malware binary.

Analyses the malware binary for malicious patterns.

Thus It becomes a special chatbot for a Malware analyst. Even roles can be added by which the AI answers appropriately as per the audience. Presently

Google Bard has indicated that there exists a possibility in the future that they would allow a system where they would Train Bard on a large dataset of malware samples. This would allow Bard to learn the features and behaviours of malware.

Give Bard access to a variety of malware analysis tools. This would allow Bard to perform static and dynamic analysis of malware samples.

Develop algorithms that allow Bard to identify and classify malware samples. This would allow Bard to automate the process of malware analysis.

Integrate Bard with a sandbox environment. This would allow Bard to execute malware samples in a controlled environment and observe their behaviour

Google in the near future will come up with a product that will be a code analyser.

Query box

This GUI program displays two areas which are the input and output area. Any code is pasted in the input area and there are two ways to search on the information. Firstly with a right click and secondary by choosing the question from the dropdown. Results of the question are available in the lower windows of the program.

6. Tools of the Trade

A section of the program is dedicated to recording all the software and their locations to use as a handy toolbox to deal with the analysis. These are useful in the sense that there are only a few tools people generally use and want a handy access to. The can be opened by double clicking or pressing run

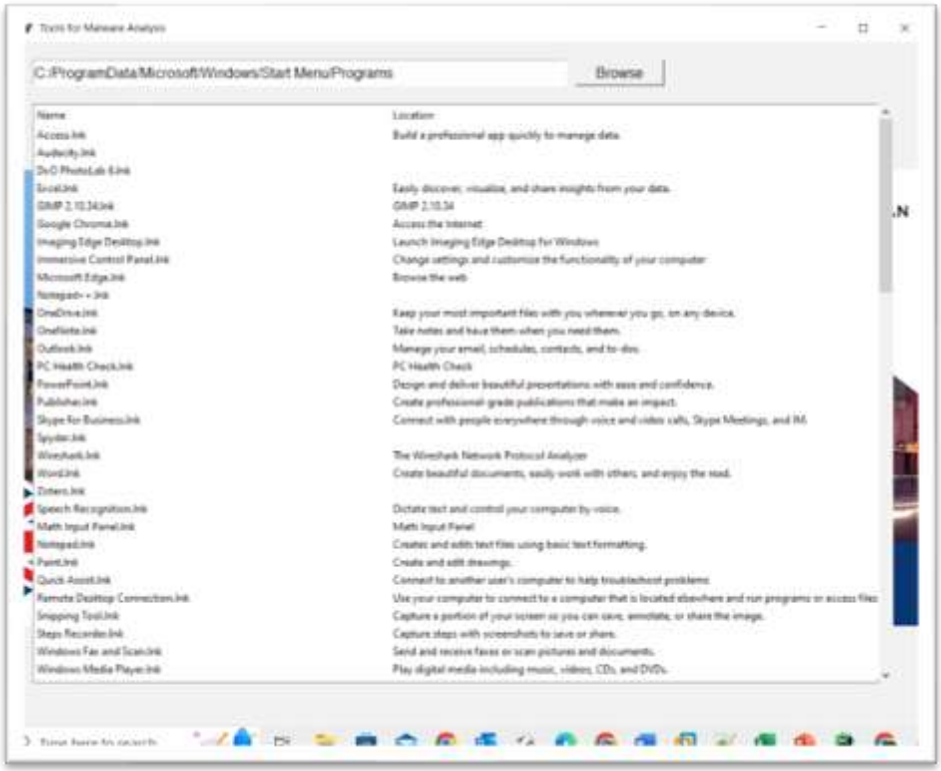

Figure 28 Tool set

Figure 29 Location of any custom program

The above module has potential to keep links to any company database, software that is used internally for the company and that may be required to be entered to gather data from Malware analysis. The user therefore has one place to enter all the software, connects to any internal

database of a company through the right module. This allows to use existing company solutions for recording other program flows.

Below is a summarizes look at the functions of the important function of the software.

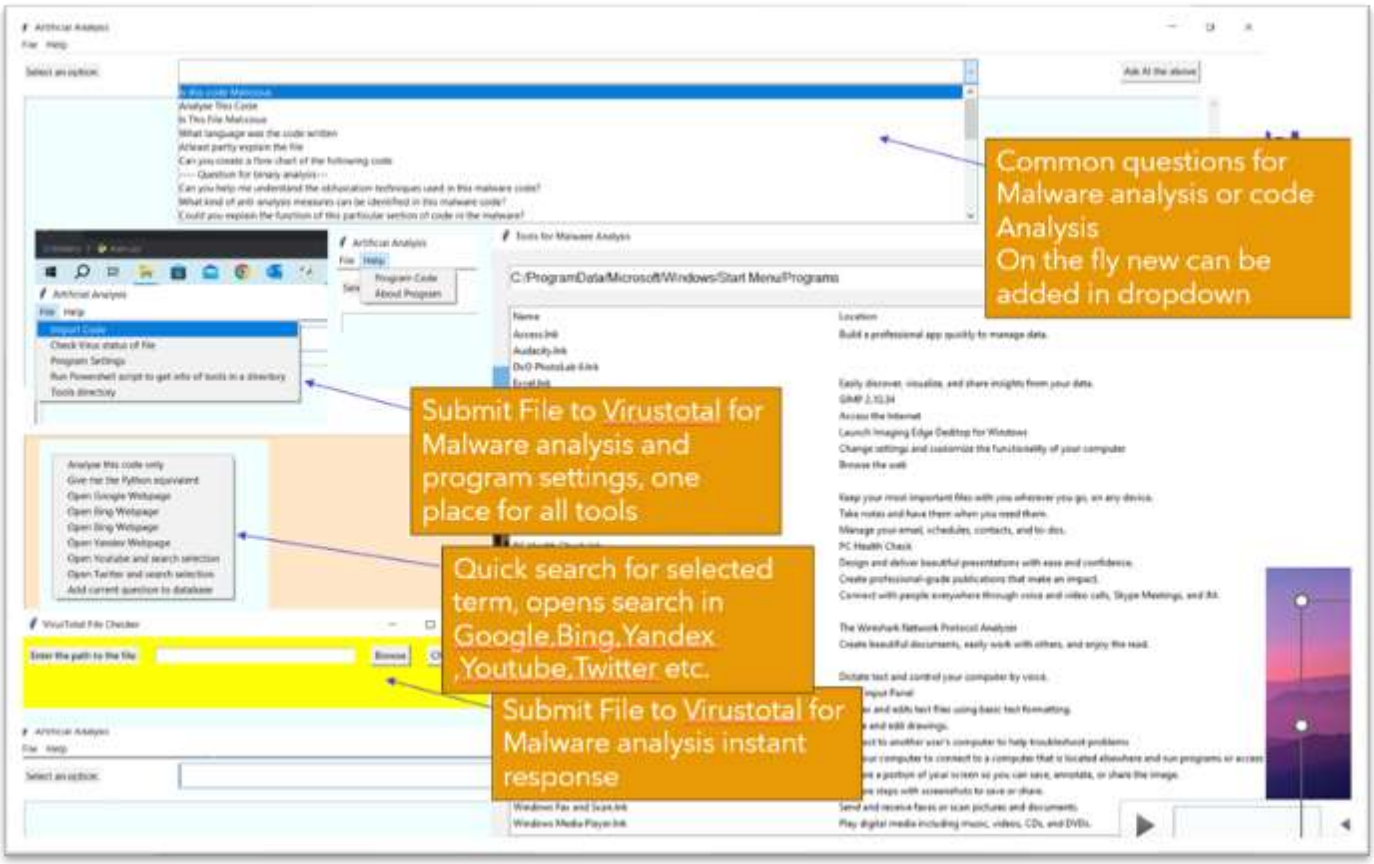

Figure 30 General explanation of program parts

During the coding and development, a number of programs were made and destroyed so that only programs of value for malware analysis are retained which delivered true value to the program. As the software is in Python its code could be easily and can therefore be easily worked on. Supporting datafiles etc.

Presently as the Ai field is evolving rapidly a number of products similar to this idea are being planned by corporates which gives hope that this process is right and very soon we may receive an application that can just do malware analysis based on trained data.

Chapter 10 Solutions to Malware

Cybersecurity is today a vital component of any business or individual big or small. Most computers, devices, instruments etc. today are connected to the internet in some way or the other. It is through this connectivity that programs and services are obtained. In fact, all patches to operating system and any software today are obtained through the internet only. In such a connected world, malicious code getting delivered to a system could jeopardize a system, network or service.

The execution of such malware code can have disastrous consequences leading to loss of data, money, life and several other implications.

It is therefore vital to stop the malicious code getting into the system by ensuring that current recommendation should be adhered to when it comes to defending an organization from Malware. The steps that should be followed are below.

A framework of repute which is suitable to the organization such as NIST Cybersecurity Framework (CSF), ISO/EIC27001, COBIT 5, CIS Control, etc. should be followed in the first place because it provides a structured approach to cybersecurity with years of experience in the framework that can identify and assess risks with appropriate controls to mitigate the risks. This IT security framework contains a series of documented processes which have defined policies and procedures. The implementation and ongoing management of information security controls are based around these policies and procedures. These frameworks act as a blueprint for managing risk to the organization and also reducing vulnerabilities. A proper implementation and regular audit that the procedures are validated will ensure that due diligence in IT has been done by the IT and also ensure smooth work process. Often the first place for a CISO of any company is to observe whether the standards are followed or not. It is also important that such standard be implemented with the risk approach of the organisation in line with the managements decision well documented. The framework mentioned here is an example and every country follows a certain framework which is suited to them in the purview of their law. It is therefore advisable to use the best standard for the particular business and one which in line with their law.

Firewalls and IDS/IPS are essential security measures that must be installed and kept updated to protect organizations from cyberattacks. They provide protection against known threats and can also be configured to detect and block new threats. Certain firewalls are better than other by their pedigree and so those should be endeavoured to be deployed. It is essential that they are configured properly and if not some specialist or specialist company in that firewall business should be contracted to give a optimum solution. Whenever known threat are informed to the firewall owner they should check the rules of the ingress and egress traffic and make amendment with documentation and authorization. Routers should be updated with the latest recommendation from the OEM company as many of the vulnerabilities in routers are necessary to be updated.

a) Anti-malware or antivirus software should be deployed on all systems because it helps to protect against malware infections. Relatively speaking any anti malware or antivirus software should be regularly updated diligently. This offers great protection to the system than no protection at all.
b) Physical security measures should be deployed in corporate networks because they can help to protect against a variety of threats, including Physical Threat, natural Disasters, Environmental Hazards and human errors. Without physical security any damage can occur which could be man made or natural.
c) Regular Audits must be done to help identify and address security risks, ensure compliance with regulations, and improve efficiency. The regular audit also lets administrator or other people stay abreast of security on their system and their security posture better fully knowing well that an audit will occur.
d) Constant review and good security posture should be maintained by IT because the cyberthreat landscape is constantly evolving. New threats are emerging all the time, and existing threats are becoming more sophisticated. By constantly reviewing their security posture, organizations can stay ahead of the curve and protect themselves from these threats.

Even after implementation of the above sometimes malicious code infect systems or gets deployed to user machine inside the network on account of users downloading a malware or virus

unintentionally through various tricks that criminals use such as social engineering, Phishing, Malvertising, zero-day attacks, drive-by downloads, usb drive attacks, watering hole attacks etc. Once a malware is delivered it needs to be executed and this is done through tricking the user into running the software or allowing it to be executed on the system through scheduled task or some other process. Once a malware is deployed it tries to replicate itself, disguising itself as a legitimate process in order to fool the operating system or to the anti-malware software. It then executes the full payload if it already exists inside the malware binary or downloads the payload from the internet from malicious sites or compromised sites.

Many modern malwares are aware of techniques used by anti-malwares companies and therefore they try to use all the techniques that help them in avoiding detection by anti-malware software. These techniques are.

a) Obfuscation – which makes it hard for humans or even machines to understand any complex code logically. Techniques such as compression, complex code construct and encryption are used.
b) Rootkits – which gives it full control over the system that is infected and also hide from any antivirus software or any security measures.
c) Polymorphism – which allows the code of the malware to change as time passes so as to thwart off antivirus software which always gets a new signature of the malware to deal with.
d) Zero-day Attacks- exploits the vulnerabilities that are present and can be utilized on the day the software is released and no patch has been available to fix it.

With the above techniques malware is able to successfully execute on their targeted system and achieve infiltration or fulfil the ulterior motive of the malware.

Malware today tries to avoid leaving a trace in the system log and completely try to operate in memory if possible because most antimalware programs can detect the traces left behind by malware programs and get alerted to destroy or halt the malware. Most malware leave behind some fingerprints as they are used from a file created on the system by the malware. The latest kind of Fileless malwares on the other hand are even more difficult to detect and remove.

1. *What are its current issues?*

Today malware is rampant and in the last few years it has reached alarming proportions. This unethical form of coding has put to loss nations, people, property and life. Malware writers have solely been responsible for thousands of financial losses, Data losses, identity theft, Cyberbullying, Denial of service attacks, sabotage and espionage in various countries. They have also been responsible for electricity blackouts, causing damage to nuclear stations, ships, aero plane, suicide of young people, distributing political and pornographic materials, anti-social behaviours and the list is endless.

While many of the malwares are targeted at causing computer damage yet many are targeted towards changing governments, affecting social behaviour through propaganda etc. Today a tweet could land anyone in jail and many malware rootkits have used unsuspecting people's computers and planted evidence from mutiny to terrorism to land them in jails for no fault of theirs. Applications like Pegasus and others have shaken the confidence in IT of the common man and today most politicians are worried of when they would be under attack without ever knowing. The problem is not only technological but even a social problem to which the IT industry should try and find permanent solutions.

While working with organizations over 2 decades many a times these ethical and moral issues have come up where IT were often asked to investigate primarily or provide evidence when events break out. Whether there is an information leak, a speed issue in the network or processing, a computer crash, a recovery of data, data theft or any other such issues IT is expected to know it better than anyone else.

This leads to a question “Does IT really know the root causes of the issue and how to fix it?” More often than not IT either boils down the issue to a Malware etc. or is unable to point to anything beyond this.

The buck stops here.

Even with firewall, EDR solutions in place the solutions are not precise in actually pinpointing where the issue came from. It is akin to finding a needle in a haystack etc. An average network or system administrator is unable to go beyond this point in determining the root cause of the event or what are the activities that malware is capable of. Where does the malware get its code from? and what locations on the system or network does it affect? This is where Malware Analyst begins.

2. *Project Implementation*

Many malwares utilize these shared resources and carry out attacks using the common libraries available.

On the other hands hackers always look at exploiting the vulnerabilities of a system and try to write malicious code that could harm or disrupt the functionality of a computer.

Thus, any such piece of computer code that performs such unintended action without the explicit permission by the computer system is termed as malware.

Users generally execute their common work-related program on their computers but sometimes they may receive a link or a file from a known or unknown recipient or could get drives etc. which could contain the file. The user is tricked to unintentionally execute the malware on the system and thus the malware gets distributed through various nefarious means. These could include use of social engineering, Phishing, Malvertising, zero-day attacks, drive-by downloads, usb drive attacks, watering hole attacks etc. types of attacks.

Today malware, however, is highly sophisticated and well aware about current anti malware systems. They try to limit their sizes and Hijack available DLLS or shared libraries and also derive their code from cryptic websites that work as a Command-and-control centre. This makes it harder for any antimalware to detect the presence of malware on the system.

Several services and underground players perform reconnaissance attacks first to find out the vulnerabilities of a system or an organization. These vulnerabilities are then worked upon to deliver targeted malware to the organization which most of the time works and the organization has practically very little chance to defend itself. If the infiltration is detected through Intrusion detection systems (IDSs) and intrusion prevention systems (IPSs) then preventative or corrective action can be taken.

If a malware intrusion is suspected or noticed through the above systems, the IT team and management will need answers to the specifics of the malware, which can only be obtained through a thorough analysis of the malware.

With the above background it is clear that malware analysis is a process every company may have to undertake itself even if there are suspected attacks on the system to better equip themselves with appropriate solutions. Let's delve into how a malware analysis is performed.

Chapter 11 Conclusion

1. Artificial Intelligence is the answer.

The recent advancement in artificial intelligence by companies like openai.com and its successful applications like ChatGPT based on the GPT-3.5 and GPT 4 model has completely revolutionized search and given humans the power to get the exact data they are looking for. In fact all of human knowledge was tried to be put into these technologies and over years the dataset on which it has been based has been curated. OpenAI on its official site states that "it can occasionally produce incorrect answers. It has limited knowledge of world and events after 2021 and may also occasionally produce harmful instructions or biased content." (Natalie, 2023) and yet it is today the most widely used program across the world in several languages.

Companies now are trying to harness the power of GPT 4 for every business and application that can really make use of it. AI reduces the manhours required and give results which are astonishingly correct and which learn from human preferences. Using the model of Reinforcement Learning from Human Feedback (RLHF) ChatGPT has been able to train its initial Model on Machine learning and then uses a traditional supervised learning model to present the data to human evaluator for their input to decide if the answer is correct or not. This evaluation by hundred of subject matter experts is tabulated and the most popular answers are spurted out to the user whenever a similar question is received. The user also becomes a evaluator as he accepts an answer or rejects it which is tallied into the final count.

This self propelling system works quite well and hence most answers are mostly accurate and slightly based on the way the question is asked.

The system is based on the rank of the model the curated data is further passed through a reinforcement model which optimizes the expected reward using certain algorithms such as Proximal Policy Optimization (PPO) or Trust Region Policy Optimization (TRPO). On repeated use of the same system the optimization is achieved overall, and the answers most likely become the ones which are correct but its not guaranteed but works well.

Critics of this system state in one paper by Corneil university by (Stephen Casper, 2023) that there has been little finetuning in the method of RHLF and hence there are serious limitations which must be resolved with a multi-faced approach to developing safer system.

Today Malware Analysis is done by human evaluators but the consorted effort to have all the data in one system does not exist. Since the problem is very specific in nature and works in conjunction with rules defined for softwares I believe it's a good area to investigate further and iron out the issues related to identifying the actions that a malware code does.

There are serious challenges to this work as the code is not in human readable form and hence the input cannot come in large quantity from humans. There are obfuscation techniques that hamper the code evaluation. There are algorithms and encryption techniques due to which breaking an encryption for the purpose of one malware evaluation may take years but then there is a possibility clearly which is emerging. In a conference I attended by Digital Trust World Boston 2023 - Webinar -The Next 20 Years: AI and the Quantum Revolution by scientist Michio Kaku the futurist he revealed that the Quantum computing will be able to resolve most of todays encryption methods in a matter of minutes or seconds due to the way its setup.

Hence we can be rest assured that quantum computing can aid the current issues highlighted above and probably go much beyond todays computing power which is again similar to computing power of a laptop today to the supercomputer of yesteryears.

In simple words , the next evolution of computing is going to be 158 million times faster than the most sophisticated supercomputer that we have in the world and that it would be powerful enough to do in four minutes what would take a traditional supercomputer 10,000 years to accomplish developers should be able to easily leverage it to their use in Artificial Intelligence. (Smith, 2022). So anything is possible and its only a matter of time.

2. Malware Analysis with Quantum computing

As explained above the bottlenecks that exist today are due to the kind of programs, computing algorithms, dataset , computing resources and the number of permutations and combinations that are to do with identifying what action a malware would take when it would execute i.e. Malware analysis.

If humans can learn malware analysis in several years and become good which is just text and computer based action then it remains to be seen how this can be codified and used with AI.
The idea is simple. Record all the methods and techniques used for malware analysis.
Record the most likely if then what scenarios that a human would take when he sees code.
Have a curated unique dataset of Malware currently know with options to add new one.
Have effective solutions ready which are known to solve a particular cybersecurity issue.
Combine them through Machine Learning and let Artificial intelligence utilize its techniques and deliver results to humans who will verify it. This in itself will save millions of manhours.
Use the RHLF method to propel this self fulfilling system with regular tweaks and controls.
This will help develop an automated intelligent system evaluate a code and suggest appropriate solutions within minutes.
If its logical and well thought of its possible is my belief.
The best example that exist today is Object Character Recognition. When it started out it seemed to be a problematic area which could never be resolved. But today we see that most of the documents can be easily recognized by the best OCR software. There are still occasional issues with many documents that cannot be done but then most of it can be done and hence it has been a great boon.
If we talk in terms of strategy the chess programs and Go programs are testament to what cannot be achieved if there is efforts fixated to solve a particular issue.
Currently advances in Deep learning (Sharma, 2022) are achieving great results as they are techniques that humans deploy to solve different type of problems.
I propose that

1. Artificial intelligence should be trained on data of known malware types and benign data types. It should learn by itself using deep learning techniques from the possible actions that can be taken and work towards the next best step in solving or understand the code. When these results are achieved it will be clear, what works and what doesn't.
2. The statistical evidence can then be verified by RHLF technique with specialist in the field of malware analyst and determine the best step to take on a particular type of evidence.

3. Hundred of machines should compete against each to figure the above to ensure that all the options are exhausted and the steps quantified for resolving the problem. This will prepare a dataset which will keep evolving until it matures will lead the way to deciphering malwares instantly as soon as they are presented to the specific AI engine.
4. This is similar to how AlphaGo project figured out how to win a game of Go. (Hassabis, 2015) and previously did the same with a different approach to chess. The number of moves in chess game are limited and the outcome can be only a few of them. The below diagram shows the visualization of such a technique.

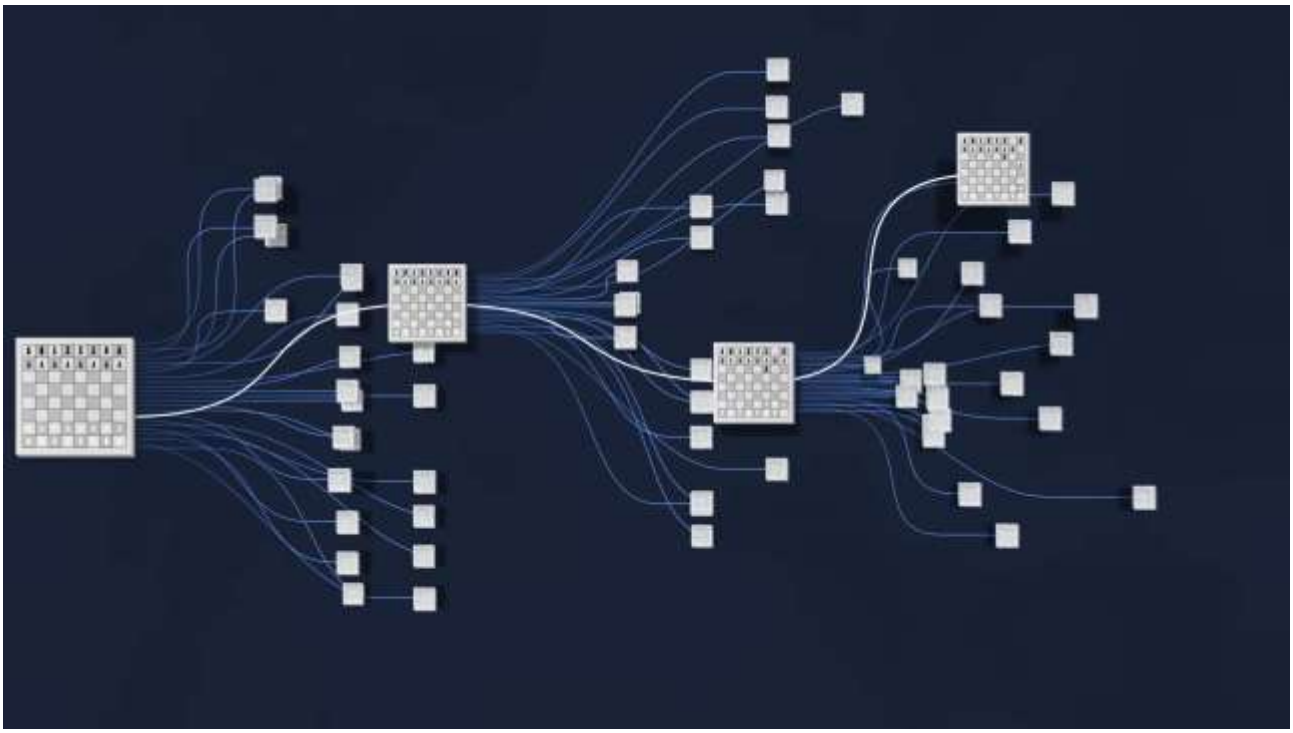

Figure 31 Possible number of moves in chess Source YouTube

5. A similar approach to Malware Analysis should be taken as the input is fixed and only the possibilities need to be figured out.
6. When this is figured the solutions the solutions from the cybersecurity perspective can be tabulated against the identified vulnerability that the malware is attempting to exploit. For every malware the better analysis will ensure a better solution from the SME (Subject Matter Experts).

7. This will help much more in case of targeted attacks as a company would have an engine to submit their file and get an exact set of result as to what plan of action the company must do for the particular malware which therefore makes it highly efficient.
8. The CISO of the company would then would be left to concentrate in working with the implementation of best practices and not worried about a business going down on account of a malware.

Chapter 11. The Actual code and supported files of the program

All the files which are important for the program are mentioned below. Create a folder on the C:\Malany and create the below mentioned files there.

Replace the API keys to OpenAI with your own key and a subscription is required . Increase token for the query in case it does not run well or returns error or use another engine other than `OpenAI GPT-3.5`

`All the actual files are attached in a zip folder. And therefore there is no need to particularly build these file. Just unzip the folder to C:\Malany and the program should be working fine.`

`The code is also available on https://github.com/areefrakhangi/Malany and if any clarification is required kindly contact me on areefrakhangi@gmail.com`

File name	main.py	Location	C:\Malany
Purpose	This is the main file which builds the program. Note all libraries that are not present in your environment must be imported if found missing. There is also a module for that but this is the general guideline An API key needs to be purchased for the project from OpenAI which is stored in API.txt file		

Code	<pre>import tkinter as tk
	from tkinter import ttk, filedialog, messagebox
	import openai
	from fpdf import FPDF
	import pyperclip # Import Pyperclip Module
	import webbrowser # Import Webbrowser Module
	import pyautogui
	import time</pre>		

```
import csv
import subprocess


# Configure your OpenAI API key
API_KEY = open("API_KEY", "r").read()
openai.api_key = API_KEY

class ArtificialAnalysisApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Artificial Analysis")
        self.root.state('zoomed')  # Maximize the window
        self.load_button_data()
        self.create_widgets()
        self.create_menu()
        self.create_context_menu()

    def load_button_data(self):
        self.button_options = []
        try:
            with open("Buttonslist.csv", "r") as file:
                for line in file:
                    self.button_options.append(line.strip())
        except FileNotFoundError:
            pass

    def create_widgets(self):
        # Load your image (replace the file path with your image
file)
        canvas_bg_image =
tk.PhotoImage(file="C:\\Malany\\background_icon.png")

        canvas = tk.Canvas(root, bg="white",
width=canvas_bg_image.width(), height=canvas_bg_image.height())
        canvas.create_image(0, 0, image=canvas_bg_image,
anchor="nw")
        canvas.pack(fill=tk.BOTH, expand=True)




        self.dropdown_label = tk.Label(canvas, text="Select an
option:")
        self.dropdown_label.grid(row=0, column=0, padx=10, pady=10,
sticky="w")

        self.dropdown_var = tk.StringVar()
        self.dropdown = ttk.Combobox(canvas,
textvariable=self.dropdown_var, values=self.button_options,
background="Orange", width=80,
                                     font=("Arial Bold ", 16))
        self.dropdown.grid(row=0, column=1, padx=10, pady=10,
sticky="w")

        self.ask_button = tk.Button(canvas, text="Ask AI the
```

```
above", command=self.ask_ai)
        self.ask_button.grid(row=0, column=2, padx=10, pady=10,
sticky="e")

        self.text_input = tk.Text(canvas, wrap="word",
font=("Courier", 12), fg="blue",background="azure1",
                                  width=150,height=20)  # Adjust
the height here
        self.text_input.grid(row=1, column=0, columnspan=3,
padx=10, pady=10, sticky="nsew")
        self.text_scrollbar = tk.Scrollbar(canvas,
command=self.text_input.yview)

self.text_input.config(yscrollcommand=self.text_scrollbar.set)
        self.text_scrollbar.grid(row=1, column=3, sticky="ns",
pady=10)

        self.send_button = tk.Button(canvas, text="Send to AI",
command=self.send_to_ai)
        self.send_button.grid(row=2, column=0, columnspan=3,
padx=10, pady=10)

        self.output_answer = tk.Text(canvas, wrap="word",
font=("Arial", 12),fg="red", background="bisque1",
                                     width=150,height=20)  # Adjust
the height here
        self.output_answer.grid(row=3, column=0, columnspan=3,
padx=10, pady=10, sticky="nsew")
        self.output_scrollbar = tk.Scrollbar(canvas,
command=self.output_answer.yview)

self.output_answer.config(yscrollcommand=self.output_scrollbar.set)
        self.output_scrollbar.grid(row=3, column=3, sticky="ns",
pady=10)

        self.transfer_button = tk.Button(canvas, text="Transfer
Analysis to File", command=self.transfer_to_pdf)
        self.transfer_button.grid(row=4, column=0, columnspan=3,
padx=10, pady=10)

    def create_menu(self):
        menu_bar = tk.Menu(self.root)
        self.root.config(menu=menu_bar)

        file_menu = tk.Menu(menu_bar, tearoff=0)
        menu_bar.add_cascade(label="File", menu=file_menu)
        file_menu.add_command(label="Import Code",
command=self.import_code)
        file_menu.add_command(label="Check Virus status of file",
command=self.CheckViruscode)
        file_menu.add_command(label="Program Settings",
command=self.Programsettings)
        file_menu.add_command(label="Run Powershell script to get
info of tools in a directory", command=self.Powershellsettings)
        file_menu.add_command(label="Tools directory",
command=self.ToolsDirectory)
```

```
        file_menu.add_command(label="Install Module if not
present", command=self.InstallModules)
        help_menu = tk.Menu(menu_bar, tearoff=0)
        menu_bar.add_cascade(label="Help", menu=help_menu)
        help_menu.add_command(label="Program Code",
command=self.show_program_code)
        help_menu.add_command(label="About Program",
command=self.show_about_program)

    def create_context_menu(self):
        self.input_context_menu = tk.Menu(self.text_input,
tearoff=0)
        self.input_context_menu.add_command(label="Analyse this
code only", command=self.analyse_code)
        self.input_context_menu.add_command(label="Give me the
Python equivalent", command=self.python_equivalent)
        self.input_context_menu.add_command(label="Open Google
Webpage", command=self.open_webpage)
        self.input_context_menu.add_command(label="Open Bing
Webpage", command=self.open_Bing)
        self.input_context_menu.add_command(label="Open Github
Search", command=self.open_Github)
        self.input_context_menu.add_command(label="Open Yandex
Webpage", command=self.open_Yandex)
        self.input_context_menu.add_command(label="Open Youtube and
search selection", command=self.open_Youtube)
        self.input_context_menu.add_command(label="Open Twitter and
search selection", command=self.open_Twitter)
        self.input_context_menu.add_command(label="Add current
question to database", command=self.Add_question_to_database)
        self.text_input.bind("<Button-3>", self.show_context_menu)

    def show_context_menu(self, event):
        self.input_context_menu.tk_popup(event.x_root,
event.y_root)

    def analyse_code(self):
        selected_text = self.text_input.get("sel.first",
"sel.last")
        prompt = "Analyse this code\n" + selected_text
        response = self.chatgpt_request(prompt)
        self.output_answer.insert("end", chars="Question : "+
selected_text )
        self.output_answer.insert("end", response, "brown")
        self.output_answer.insert("end",chars="\n
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~" + " \n ")
        self.output_answer.insert("end", "\n\n")




    def python_equivalent(self):
        selected_text = self.text_input.get("sel.first",
"sel.last")
        response = self.chatgpt_request(selected_text)
        self.output_answer.insert("end", response, "green")
```

```
        self.output_answer.insert("end", "\n\n")

    def open_webpage(self):
        # Save the contents of the clipboard to a string
        selected_text = self.text_input.get("sel.first",
"sel.last")
        # Open the web browser and search for it
        webbrowser.open('http://www.google.com/search?q=' +
selected_text)

    def open_Bard(self):
        # Save the contents of the clipboard to a string
        selected_text = self.text_input.get("sel.first",
"sel.last")
        # Open the web browser and search for it
        webbrowser.open('https://bard.google.com')
        # Send a string of text (e.g., "Hello, World!")
    def open_Youtube(self):
        # Save the contents of the clipboard to a string
        selected_text = self.text_input.get("sel.first",
"sel.last")
        # Open the web browser and search for it

webbrowser.open('https://www.youtube.com/results?search_query=' +
selected_text)

    def open_Bing(self):
        # Save the contents of the clipboard to a string
        selected_text = self.text_input.get("sel.first",
"sel.last")
        # Open the web browser and search for it
        webbrowser.open('https://www.bing.com/search?q=' +
selected_text)

    def open_Github(self):
        # Save the contents of the clipboard to a string
        selected_text = self.text_input.get("sel.first",
"sel.last")
        # Open the web browser and search for it
        webbrowser.open('https://github.com/search?q=' +
selected_text)

    def open_Twitter(self):
        # Save the contents of the clipboard to a string
        selected_text = self.text_input.get("sel.first",
"sel.last")
        # Open the web browser and search for it
        webbrowser.open('https://twitter.com/search?q=' +
selected_text)


    def open_Yandex(self):
        # Save the contents of the clipboard to a string
        selected_text = self.text_input.get("sel.first",
"sel.last")
        # Open the web browser and search for it
```

```
        webbrowser.open('https://yandex.com/search?text=' +
selected_text)


    def import_code(self):
        file_path = filedialog.askopenfilename(filetypes=[("Text
files", "*.txt")])
        if file_path:
            with open(file_path, "r") as file:
                code = file.read()
                self.text_input.delete("1.0", "end")
                self.text_input.insert("end", code)
    def CheckViruscode(self):
        # Replace 'Check_Virus_Status.py' with the actual path to
your script
        script_path = 'Check_Virus_Status.py'

        # Call the external script using subprocess
        try:
            subprocess.run(['python', script_path], check=True)
        except subprocess.CalledProcessError as e:
            print(f"Error running the script: {e}")

    def Programsettings(self):
        # Replace 'Programsettings.py' with the actual path to your
script
        script_path = 'Settings.py'

        # Call the external script using subprocess
        try:
            subprocess.run(['python', script_path], check=True)
        except subprocess.CalledProcessError as e:
            print(f"Error running the script: {e}")

    def Powershellsettings(self):
        # Define the path to the PowerShell script
        ps_script_path = "C:\\Malany\\Scandirectory.ps1"

        # Create a PowerShell command to open the script in the
PowerShell editor with admin rights
        powershell_cmd = f"Start-Process -Wait -Verb RunAs
powershell_ise -ArgumentList '-File \"{ps_script_path}\"'"

        # Run the PowerShell command
        try:
            subprocess.run(["powershell.exe", "-Command",
powershell_cmd], check=True)
        except subprocess.CalledProcessError as e:
            print(f"Error: {e}")
    def ToolsDirectory(self):
        script_path= 'ToolsDirectory.py'
        try:
            subprocess.run(['python', script_path], check=True)
        except subprocess.CalledProcessError as e:
            print(f"Error running the script: {e}")
        subprocess(ToolsDirectory.py)
```

```
    def InstallModules(self):
        script_path = 'InstallModulesifmissing.py'
        try:
            subprocess.run(['python', script_path], check=True)
        except subprocess.CalledProcessError as e:
            print(f"Error running the script: {e}")


    def send_to_ai(self):
        input_text = self.text_input.get("1.0", "end-1c")
        response = self.chatgpt_request(input_text)
        self.output_answer.insert("end", response, "red")
        self.output_answer.insert("end", "\n\n")

    def ask_ai(self):
        selected_option = self.dropdown_var.get()
        input_text = self.text_input.get("1.0", "end-1c")
        combined_text = f"{selected_option}\n{input_text}"

        # This line is particularly stopped for sometime
        response = self.chatgpt_request(combined_text)
        # self.output_answer.insert("end", combined_text, "yellow")
        # self.output_answer.insert("end", "\n\n")
        self.output_answer.insert("end", response, "red")
        self.output_answer.insert("end", "\n\n")

    def chatgpt_request(self, prompt):
        response = openai.Completion.create(
            engine="text-davinci-003",
            prompt=prompt,
            max_tokens=250
        )
        return response.choices[0].text.strip()

    def show_program_code(self):
        with open(__file__, "r") as file:
            code = file.read()
            self.text_input.delete("1.0", "end")
            self.text_input.insert("end", code)

    def show_about_program(self):
        about_text = "Artificial Analysis Application\nVersion
1.0\n\nDeveloped by Your Name\nCopyright © 2023"
        messagebox.showinfo("About Program", about_text)

    def transfer_to_pdf(self):
        analysis_text = self.output_answer.get("1.0", "end-1c")

        pdf = FPDF()
        pdf.add_page()
        pdf.set_font("Arial", size=12)
        pdf.multi_cell(0, 10, analysis_text)
        pdf_file_path =
filedialog.asksaveasfilename(defaultextension=".pdf",
filetypes=[("PDF files", "*.pdf")])
```

```
        if pdf_file_path:
            pdf.output(pdf_file_path)

    def Add_question_to_database(self):
        # Define the text you want to append
        text_to_append = self.dropdown.get()

        # Specify the CSV file name
        csv_file_name = "Buttonslist.csv"

        # Open the CSV file in append mode and write the text
        with open(csv_file_name, mode='a', newline='') as file:
            writer = csv.writer(file)
            writer.writerow([text_to_append])

        print(f'Text "{text_to_append}" appended to
{csv_file_name}')


if __name__ == "__main__":
    root = tk.Tk()
    app = ArtificialAnalysisApp(root)
    root.mainloop()
```

File name	Settings.py	Location	C:\Malany
Purpose	The programs detected and configured are inside the Programlocation.csv. This file just calls a routine and imports those files.		

Code

```
import tkinter as tk
from tkinter import ttk
import csv
import subprocess


def run_program():
    selected_item = tree.selection()
    if selected_item:
        item = tree.item(selected_item)
        location = item["values"][2]
        subprocess.Popen([location], shell=True)


def load_csv_data():
    try:
```

```
        with open('C:/Malany/Programlocation.csv', 'r') as file:
            csv_reader = csv.reader(file)
            headers = next(csv_reader)
            tree["columns"] = headers
            tree.heading("#1", text=headers[0])
            tree.heading("#2", text=headers[1])
            tree.heading("#3", text=headers[2])

            for row in csv_reader:
                tree.insert("", "end", values=row)
    except FileNotFoundError:
        print("CSV file not found.")

# Create the main window
root = tk.Tk()
root.title("Program Launcher")

# Create a Treeview widget to display CSV data
tree = ttk.Treeview(root, columns=("icon", "name", "location"),
show="headings")
tree.pack(padx=10, pady=10)

# Add Run button to launch programs
run_button = tk.Button(root, text="Run", command=run_program)
run_button.pack(pady=10)

# Load CSV data into the Treeview widget
load_csv_data()

# Start the GUI main loop
root.mainloop()
```

File name	ToolsDirectory.py	Location	C:\Malany
Purpose			

Code

```
import tkinter as tk
from tkinter import ttk
from tkinter import filedialog
import csv
from tkinter import PhotoImage

# Function to open the selected folder
def open_folder():
    folder_path.set(filedialog.askdirectory())

# Function to launch the selected program
def launch_program(event):
    selected_item = programs_table.selection()[0]
    location = programs_table.item(selected_item, "values")[1]
    # You can implement the code to launch the selected program
here
    print(f"Launching program at location: {location}")

# Create the main application window
app = tk.Tk()
app.title("Tools for Malware Analysis")
app.geometry("1024x800")

# Set background image
background_image = PhotoImage(file="C:/Malany/background_icon.png")
background_label = tk.Label(app, image=background_image)
background_label.place(relwidth=1, relheight=1)

# Create and set initial folder path entry field
folder_path = tk.StringVar()
folder_path.set('C:/ProgramData/Microsoft/Windows/Start
Menu/Programs')
folder_entry = tk.Entry(app, textvariable=folder_path,
font=("Arial", 12))
folder_entry.place(x=20, y=20, width=600, height=30)

# Create and configure the 'Browse' button
```

```
browse_button = tk.Button(app, text="Browse", font=("Arial", 12),
command=open_folder)
browse_button.place(x=630, y=20, width=100, height=30)

# Create and configure the table
columns = ("Name", "Location")
programs_table = ttk.Treeview(app, columns=columns,
show="headings", selectmode="browse")
programs_table.heading("Name", text="Name", anchor=tk.W)
programs_table.heading("Location", text="Location", anchor=tk.W)
programs_table.column("Name", width=400)
programs_table.column("Location", width=600)
programs_table.place(x=20, y=70, width=950, height=650)

# Create a scrollbar for the table
table_scrollbar = ttk.Scrollbar(app, orient="vertical",
command=programs_table.yview)
table_scrollbar.place(x=970, y=70, height=650)

# Configure the table to use the scrollbar
programs_table.configure(yscrollcommand=table_scrollbar.set)

# Populate the table from the CSV file
try:
    with open('C:/Malany/Programlocation.csv', 'r') as csv_file:
        csv_reader = csv.reader(csv_file)
        next(csv_reader)  # Skip the header row
        for row in csv_reader:
            programs_table.insert("", "end", values=row)
except FileNotFoundError:
    print("CSV file not found!")

# Bind double-click event to launch_program function
programs_table.bind("<Double-1>", launch_program)

# Run the application
app.mainloop()
```


File name	Parsethiscode.py	Location	C:\Malany
Purpose			
Code	

```
import ast
import openai
import tkinter as tk
from tkinter import filedialog, messagebox

# Set your OpenAI API key
api_key =  open("API_KEY", "r").read()
openai.api_key = api_key

# Function to parse Python code using ast
def parse_python_code(file_path):
    try:
        with open(file_path, 'r') as file:
            python_code = file.read()

        # Parse the code into an Abstract Syntax Tree (AST)
        parsed_code = ast.parse(python_code)
        return parsed_code

    except FileNotFoundError:
        return None
    except SyntaxError as e:
        return None
    except Exception as e:
        return None

# Function to handle the "Browse" button click event
def browse_file():
    file_path = filedialog.askopenfilename(filetypes=[("Python
files", "*.py")])
    if file_path:
        parsed_code = parse_python_code(file_path)
        if parsed_code:
            ai_response = ask_chatgpt_about_code(parsed_code)
            if ai_response:
                output_text.delete("1.0", tk.END)
                output_text.insert(tk.END, ai_response)
            else:
                output_text.delete("1.0", tk.END)
                output_text.insert(tk.END, "ChatGPT did not provide
a valid response.")
        else:
            output_text.delete("1.0", tk.END)
            output_text.insert(tk.END, "Parsing failed or the file
was not found.")
```

```
# Create the main GUI window
root = tk.Tk()
root.title("Python Code Analyser")

# Create a "Browse" button to select a Python file
browse_button = tk.Button(root, text="Browse", command=browse_file)
browse_button.pack(pady=10)

# Create a text widget to display the ChatGPT response
output_text = tk.Text(root, wrap=tk.WORD, width=80, height=20)
output_text.pack(padx=10, pady=10)

# Run the GUI main loop
root.mainloop()
```

File name	newprojectapp.py	Location	C:\Malany
Purpose			

Code

```
import tkinter as tk
from tkinter import Menu
from tkinter import scrolledtext
import openai
from tkhtmlview import HTMLLabel
from tkinter import filedialog

# Replace 'YOUR_OPENAI_API_KEY' with your actual API key from
OpenAI.
API_KEY = open("API_KEY", "r").read()
openai.api_key = API_KEY

chat_log = []
class ChatGPTApp:
    def __init__(self, root):
        self.root = root
        self.root.title("ChatGPT Application")
        self.root.geometry("1000x1024")
        self.create_menus()
        self.create_input_output_fields()

    def create_menus(self):
        menubar = Menu(self.root)
        self.root.config(menu=menubar)

        file_menu = Menu(menubar, tearoff=0)
        file_menu.add_command(label="Open",
command=self.browse_a_file)
        file_menu.add_command(label="Exit", command=self.root.quit)
        menubar.add_cascade(label="File", menu=file_menu)

        edit_menu = Menu(menubar, tearoff=0)
        # Add any specific edit menu items here if needed.
        menubar.add_cascade(label="Edit", menu=edit_menu)
```

```
        functions_menu = Menu(menubar, tearoff=0)
        # Add any specific functions menu items here if needed.
        menubar.add_cascade(label="Functions", menu=functions_menu)

        help_menu = Menu(menubar, tearoff=0)
        # Add any specific help menu items here if needed.
        menubar.add_cascade(label="Help", menu=help_menu)

    def browse_a_file(self):
        file_path = filedialog.askopenfilename(filetypes=[("Text
Files", "*.txt"), ("All Files", "*.*")])
        if file_path:
            with open(file_path, "r", encoding="utf-8") as file:
                file_content = file.read()
                self.input_text.delete("1.0", tk.END)  # Clear
previous content
                self.input_text.insert(tk.END, file_content)

    def create_input_output_fields(self):
        self.input_text = scrolledtext.ScrolledText(self.root,
wrap=tk.WORD, width=150, height=5)
        self.input_text.grid(row=0, column=0, columnspan=2,
padx=10, pady=10)

        self.output_html = HTMLLabel(self.root, width=120,
height=10, background="white")
        self.output_html.grid(row=1, column=0, columnspan=2,
padx=10, pady=10)
        width = self.output_html.winfo_width()
        #This sets the width to whatever width of the control
        max_lines = width
        print(self.output_html.winfo_screenwidth())
        send_button = tk.Button(self.root, text="Send",
command=self.send_message)
        send_button.grid(row=2, column=0, columnspan=2, padx=10,
pady=10)

    def send_message(self):
        user_input = self.input_text.get("1.0", tk.END).strip()
        if user_input:
            response = self.get_chatgpt_response(user_input)
            self.display_response(response)

    def get_chatgpt_response(self, user_input):
        # Call OpenAI GPT-3.5 API to get the response
        response = openai.Completion.create(
            engine="text-davinci-002",  # You can also use "gpt-
3.5-turbo" here.
            prompt=user_input,
            max_tokens=150,
        )
        return response.choices[0].text.strip()

    def display_response(self, response):
        # Display response in the HTML output field and print it to
```

```
the console.
        self.output_html.set_html(response)
        print("ChatGPT Response:", response)
        chat_log.append({"role": "user", "content": "Analyse this
code \n" + self.input_text.get("1.0", "end")})
        print(chat_log)




if __name__ == "__main__":
    root = tk.Tk()
    app = ChatGPTApp(root)
    root.mainloop()
```

File name	InstallModulesifmissing.py	Location	C:\Malany
Purpose			

Code

```
import importlib
import subprocess


def check_and_install_missing_modules(project_modules):
    missing_modules = []

    for module_name in project_modules:
        try:
            # Attempt to import the module
            importlib.import_module(module_name)
        except ImportError:
            # If ImportError is raised, the module is missing
            missing_modules.append(module_name)

    if missing_modules:
        print("The following modules are missing and will be
installed:")
        for module_name in missing_modules:
            print(module_name)
            # Use pip to install the missing module
            subprocess.run(["pip", "install", module_name])

    print("All required modules are installed.")


if __name__ == "__main__":
    # List the modules your project depends on
    project_modules = [
        "tkinter",
        "openai",
        "fpdf",
        "pyperclip",
        "pyperclip",
```

```
        "webbrowser",
        "pyautogui",
        "time",
        "csv",
        "subprocess"
        "pandas",
        "requests",
        "nonexistent_module",
    ]

    check_and_install_missing_modules(project_modules)
```

File name	Hybrid-Analysis.py	Location	C:\Malany
Purpose			

Code

```
import tkinter as tk
from tkinter import filedialog
import requests
import json

# Define the path to the API key file
API_KEY_FILE = 'C:\\Malany\\API-Hybrid-Analysis.txt'

# Hybrid Analysis API endpoint
API_ENDPOINT = 'https://www.hybrid-analysis.com/api/v2/submit/file'


def read_api_key():
    try:
        with open(API_KEY_FILE, 'r') as file:
            api_key = file.read().strip()
        return api_key
    except FileNotFoundError:
        return None


def browse_file():
    file_path = filedialog.askopenfilename()
    if file_path:
        analyse_file(file_path)


def analyse_file(file_path):
    api_key = read_api_key()
    if api_key:
        headers = {'api-key': api_key}
        files = {'file': open(file_path, 'rb')}

        response = requests.post(API_ENDPOINT, headers=headers,
files=files)

        if response.status_code == 200:
            analysis_data = response.json()
```

```
            analysis_id = analysis_data['data']['sha256']
            result_label.config(text=f"Analysis ID: {analysis_id}")

        else:
            result_label.config(text="Error sending file for
analysis")
            response = requests.post(API_ENDPOINT, headers=headers,
files=files)
            print(response.status_code)
            print(response.text)

    else:
        result_label.config(text="API key not found. Please check
the API-Hybrid-Analysis.txt file.")


# Create the main window
root = tk.Tk()
root.title("Hybrid Analysis File Uploader")

# Create a label to display the result
result_label = tk.Label(root, text="", padx=10, pady=10)
result_label.pack()

# Create a "Browse" button
browse_button = tk.Button(root, text="Browse File",
command=browse_file)
browse_button.pack()

# Run the GUI application
root.mainloop()
```

File name	Check_Virus_Status.py	Location	C:\Malany
Purpose	This file submits any file to VirusTotal. Depending on the file size the sleeptime in this module should be increased as sometime the enginer of VirusTotal may be busy on internet connectivity slow. Try twice if required.		
Code			

```
import tkinter as tk
from tkinter import filedialog
import requests
import time

# Function to read the VirusTotal API key from a file
def read_api_key(file_path):
    try:
        with open(file_path, 'r') as file:
            api_key = file.read().strip()  # Read the content and
remove leading/trailing whitespace
```

```
        return api_key
    except FileNotFoundError:
        return None
    except Exception as e:
        return None

# Function to submit a file to VirusTotal and check its status
def check_file():
    file_path = file_path_entry.get()
    if not file_path:
        result_label.config(text="Please select a file.")
        return

    api_key = read_api_key(api_key_file_path)
    if not api_key:
        result_label.config(text="API key not found.")
        return

    # URL for submitting files to VirusTotal
    UPLOAD_URL = 'https://www.VirusTotal.com/vtapi/v2/file/scan'

    # URL for checking the scan report
    REPORT_URL = 'https://www.VirusTotal.com/vtapi/v2/file/report'

    # Step 1: Upload the file to VirusTotal
    try:
        with open(file_path, 'rb') as file:
            files = {'file': (file.name, file)}
            params = {'apikey': api_key}
            response = requests.post(UPLOAD_URL, files=files,
params=params)
            response_json = response.json()
            if 'response_code' in response_json and
response_json['response_code'] == 1:
                resource_id = response_json['resource']

                # Display a message while waiting
                result_label.config(text="File uploaded. Waiting
for results...")

                # Wait for 15 seconds
                time.sleep(15)

                # Step 2: Check the scan report
                while True:
                    params = {'apikey': api_key, 'resource':
resource_id}
                    response = requests.get(REPORT_URL,
params=params)
                    response_json = response.json()
                    if 'response_code' in response_json:
                        if response_json['response_code'] == 0:
                            time.sleep(10)  # Wait for a while and
check again
                        else:
                            break
```

```
                    else:
                        result_label.config(text="Error checking
file status.")
                        break

                if 'positives' in response_json:
                    positives = response_json['positives']
                    total = response_json['total']
                    scan_date = response_json['scan_date']
                    if positives == 0:
                        result_label.config(text=f'The file is not
malicious (Scan Date: {scan_date})')
                    else:
                        result_label.config(text=f'The file is
malicious (Scan Date: {scan_date}) - Detected by
{positives}/{total} scanners')
                else:
                    result_label.config(text='Error checking file
status.')
            else:
                result_label.config(text='Error submitting file to
VirusTotal.')
    except Exception as e:
        result_label.config(text=f'Error: {e}')

# Create the main window
window = tk.Tk()
window.title("VirusTotal File Checker")
window.configure(bg="yellow")  # Set the background color to yellow

# Create and configure widgets
file_path_label = tk.Label(window, text="Enter the path to the
file:")
file_path_label.grid(row=0, column=0, padx=10, pady=10)

file_path_entry = tk.Entry(window, width=40)
file_path_entry.grid(row=0, column=1, padx=10, pady=10)

file_path_button = tk.Button(window, text="Browse", command=lambda:
file_path_entry.insert(0, filedialog.askopenfilename()))
file_path_button.grid(row=0, column=2, padx=10, pady=10)

check_button = tk.Button(window, text="Check File",
command=check_file)
check_button.grid(row=0, column=3, padx=10, pady=10)

result_label = tk.Label(window, text="", bg="yellow")
result_label.grid(row=1, column=0, columnspan=4, padx=10, pady=10)

# Replace 'C:\Malany\VirusTotal_APIKEY.txt' with the path to your
API key file
api_key_file_path = 'C:/Malany/VirusTotal_APIKEY.txt'

# Start the GUI main loop
window.mainloop()
```


File name	Multipleserach.html	Location	C:\Malany
Purpose	It's a html file for pasting in multiple line of code and queries multiple engines at the same time. Remember to keep you pop up blocker off when running such a search		

Code	<!DOCTYPE html>
	<html lang="en">
	<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width, initial-scale=1.0">
	<title>Multi-Search Engine</title>
	<style>
	body {
	margin: 0;
	padding: 0;
	background-image: url('https://www.londonmet.ac.uk/media/london-metropolitan-university/london-met-photos/london-met-buildingsfacilities/2023/Tower-Reception.jpg');
	background-size: cover;
	background-repeat: no-repeat;
	background-attachment: fixed;
	font-family: Arial, sans-serif;
	color: #fff;
	text-align: left; /* Align text to the left */
	}
	
	.container {
	display: flex;
	flex-direction: column;
	align-items: flex-start; /* Align content to the top left */
	justify-content: flex-start; /* Align content to the top left */
	height: 100vh;
	padding: 20px; /* Add padding for spacing */
	}
	
	h1 {
	font-size: 36px;
	margin-bottom: 20px;
	}		

```
    textarea {
      padding: 10px;
      width: 80ch;
      height: 5em;
      margin: 10px 0; /* Adjust vertical margin */
      border: none;
      border-radius: 5px;
      background-color: lightpink; /* Set background color to light pink */
    }

    button {
      padding: 10px 20px;
      background-color: #007BFF;
      border: none;
      border-radius: 5px;
      color: #fff;
      font-size: 16px;
      cursor: pointer;
    }

    button:hover {
      background-color: #0056b3;
    }
  </style>
</head>
<body>
  <div class="container">
    <h1>Multi-Search Engine</h1>
    <form>
      <textarea id="searchQuery" placeholder="Enter your search
query"></textarea>
      <button type="button" onclick="searchOnEngines()">Search</button>
    </form>
  </div>

  <script>
    function searchOnEngines() {
      const query = document.getElementById('searchQuery').value;

      // Define search URLs for Google, Bing, Yahoo, You.com,
Codesector.com, DuckDuckGo, YouTube, DailyMotion, Yandex, and Twitter
      const searchEngines = [
        { name: 'Google', url: `https://www.google.com/search?q=${query}` },
        { name: 'Bing', url: `https://www.bing.com/search?q=${query}` },
```

```
            { name: 'Yahoo', url: `https://search.yahoo.com/search?p=${query}` },
            { name: 'Github', url: `https://github.com/search?q=${query}` },
            { name: 'You.com', url: `https://you.com/search?q=${query}` }, // Replace "you.com" with your desired URL
            { name: 'DuckDuckGo', url: `https://duckduckgo.com/?q=${query}` }, // DuckDuckGo search
            { name: 'YouTube', url: `https://www.youtube.com/results?search_query=${query}` }, // YouTube search
            { name: 'Yandex', url: `https://yandex.com/search/?text=${query}` }, // Yandex search
            { name: 'Twitter', url: `https://twitter.com/search?q=${query}` } // Twitter search
            // Add more search engines as needed
        ];

        // Open a new tab for each search engine
        searchEngines.forEach(engine => {
            window.open(engine.url, engine.name);
        });
    }
  </script>
</body>
</html>
```

File name	"C:\Malany\Buttonslist.csv"	Location	C:\Malany
Purpose	This stores the questions in a csv format		
Code	Is this code Malicious Analyse This Code Is This File Malicious What language was the code written Atleast partly explain the file Can you create a flow chart of the following code ---- Question for binary analysis--- STRINGS() Can you help me understand the obfuscation techniques used in this malware code? What kind of anti-analysis measures can be identified in this malware code? Could you explain the function of this particular section of code in the malware? What are the potential payloads that this malware is designed to deliver? What system APIs or libraries does this malware code interact with?		

	Can you identify any network communication methods used by this malware? How does this malware achieve persistence on the infected system? What encryption or decryption techniques are employed by this malware code? What evasion techniques does this malware use to avoid detection by security software? Could you help me analyse the code responsible for spreading the malware? What indicators of compromise (IOCs) can be derived from this malware code? Are there any specific code patterns that suggest this malware is part of a known malware family? Can you explain the process by which this malware injects itself into another process's memory? How does this malware use DLL injection to achieve its goals? Can you identify any callback functions or hooks used by this malware? What is the purpose of this code? How does this code work? What risks does this code pose to a computer system? How is this code used to infect systems? How does this code spread within a computer system? Are there any known exploits related to this code? Can this code be reversed engineered? Are there any long term implications of this code running on a system? Is this code considered malicious or benign? What types of actions does this code attempt to execute? This is the last question This is the last question select the functions in the following code and explain them

File name	C:\Malany\icon_data.csv	Location	C:\Malany
Purpose			
Code	"FileName","IconFileName" "Access.lnk","C:\Malany\icons\icon_Access.ico" "Audacity.lnk","C:\Malany\icons\icon_Audacity.ico" "DxO PhotoLab 6.lnk","C:\Malany\icons\icon_DxO PhotoLab 6.ico" "Excel.lnk","C:\Malany\icons\icon_Excel.ico" "GIMP 2.10.34.lnk","C:\Malany\icons\icon_GIMP 2.10.34.ico" "Google Chrome.lnk","C:\Malany\icons\icon_Google Chrome.ico" "Imaging Edge Desktop.lnk","C:\Malany\icons\icon_Imaging Edge Desktop.ico" "Immersive Control Panel.lnk","C:\Malany\icons\icon_Immersive Control Panel.ico" "Microsoft Edge.lnk","C:\Malany\icons\icon_Microsoft Edge.ico"		

	"Notepad++.lnk","C:\Malany\icons\icon_Notepad++.ico" "OneDrive.lnk","C:\Malany\icons\icon_OneDrive.ico" "OneNote.lnk","C:\Malany\icons\icon_OneNote.ico" "Outlook.lnk","C:\Malany\icons\icon_Outlook.ico" "PC Health Check.lnk","C:\Malany\icons\icon_PC Health Check.ico" "PowerPoint.lnk","C:\Malany\icons\icon_PowerPoint.ico" "Publisher.lnk","C:\Malany\icons\icon_Publisher.ico" "Skype for Business.lnk","C:\Malany\icons\icon_Skype for Business.ico" "Spyder.lnk","C:\Malany\icons\icon_Spyder.ico" "Wireshark.lnk","C:\Malany\icons\icon_Wireshark.ico" "Word.lnk","C:\Malany\icons\icon_Word.ico" "Zotero.lnk","C:\Malany\icons\icon_Zotero.ico"

File name	C:\Malany\information.csv	Location	C:\Malany
Purpose	Location of filenames that can be used if this module is used.		
Code	"Name","Description","Target" "Access.lnk","Build a professional app quickly to manage data.","C:\Program Files (x86)\Microsoft Office\root\Office16\MSACCESS.EXE" "Audacity.lnk","","C:\Program Files\Audacity\Audacity.exe" "DxO PhotoLab 6.lnk","","C:\Program Files (x86)\DxO\DxO PhotoLab 6\DxO.PhotoLab.exe" "Excel.lnk","Easily discover, visualize, and share insights from your data.","C:\Program Files (x86)\Microsoft Office\root\Office16\EXCEL.EXE" "GIMP 2.10.34.lnk","GIMP 2.10.34","C:\Program Files\GIMP 2\bin\gimp-2.10.exe" "Google Chrome.lnk","Access the Internet","C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" "Imaging Edge Desktop.lnk","Launch Imaging Edge Desktop for Windows","C:\Program Files (x86)\Sony\Imaging Edge Desktop\ied.exe" "Immersive Control Panel.lnk","Change settings and customize the functionality of your computer","C:\Windows\System32\Control.exe" "Microsoft Edge.lnk","Browse the web","C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" "Notepad++.lnk","","C:\Program Files\Notepad++\notepad++.exe" "OneDrive.lnk","Keep your most important files with you wherever you go, on any device.","C:\Program Files (x86)\Microsoft OneDrive\OneDrive.exe" "OneNote.lnk","Take notes and have them when you need them.","C:\Program Files (x86)\Microsoft Office\root\Office16\ONENOTE.EXE" "Outlook.lnk","Manage your email, schedules, contacts, and to-dos.","C:\Program Files (x86)\Microsoft Office\root\Office16\OUTLOOK.EXE" "PC Health Check.lnk","PC Health Check","C:\Program Files (x86)\PCHealthCheck\PCHealthCheck.exe"		

	"PowerPoint.lnk","Design and deliver beautiful presentations with ease and confidence.","C:\Program Files (x86)\Microsoft Office\root\Office16\POWERPNT.EXE" "Publisher.lnk","Create professional-grade publications that make an impact.","C:\Program Files (x86)\Microsoft Office\root\Office16\MSPUB.EXE" "Skype for Business.lnk","Connect with people everywhere through voice and video calls, Skype Meetings, and IM.","C:\Program Files (x86)\Microsoft Office\root\Office16\lync.exe" "Spyder.lnk","","C:\Program Files\Spyder\Python\pythonw.exe" "Wireshark.lnk","The Wireshark Network Protocol Analyser","C:\Program Files\Wireshark\Wireshark.exe" "Word.lnk","Create beautiful documents, easily work with others, and enjoy the read.","C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE" "Zotero.lnk","","C:\Program Files (x86)\Zotero\zotero.exe" "Speech Recognition.lnk","Dictate text and control your computer by voice.","C:\Windows\Speech\Common\sapisvr.exe" "Math Input Panel.lnk",,"C:\Program Files (x86)\Common Files\Microsoft Shared\Ink\mip.exe" "Notepad.lnk","Creates and edits text files using basic text formatting.","C:\Windows\system32\notepad.exe" "Paint.lnk","Create and edit drawings.","C:\Windows\system32\mspaint.exe" "Quick Assist.lnk","Connect to another user's computer to help troubleshoot problems","C:\Windows\system32\quickassist.exe" "Remote Desktop Connection.lnk","Use your computer to connect to a computer that is located elsewhere and run programs or access files.","C:\Windows\system32\mstsc.exe" "Snipping Tool.lnk","Capture a portion of your screen so you can save, annotate, or share the image.","C:\Windows\system32\SnippingTool.exe" "Steps Recorder.lnk","Capture steps with screenshots to save or share.","C:\Windows\system32\psr.exe" "Windows Fax and Scan.lnk","Send and receive faxes or scan pictures and documents.","C:\Windows\system32\WFS.exe" "Windows Media Player.lnk","Play digital media including music, videos, CDs, and DVDs.","C:\Program Files (x86)\Windows Media Player\wmplayer.exe" "Wordpad.lnk","Creates and edits text documents with complex formatting.","C:\Program Files (x86)\Windows NT\Accessories\wordpad.exe" "Character Map.lnk","Selects special characters and copies them to your document.","C:\Windows\system32\charmap.exe" "Component Services.lnk","Manage COM+ applications, COM and DCOM system configuration, and the Distributed Transaction Coordinator.","C:\Windows\system32\comexp.msc" "Computer Management.lnk","Manages disks and provides access to other tools to manage local and remote computers.","C:\Windows\system32\compmgmt.msc" "dfrgui.lnk","Optimizes files and fragments on your volumes so that your computer runs faster and more efficiently.","C:\Windows\system32\dfrgui.exe"

	"Disk Cleanup.lnk","Enables you to clear your disk of unnecessary files.","C:\Windows\system32\cleanmgr.exe" "Event Viewer.lnk","View monitoring and troubleshooting messages from windows and other programs.","C:\Windows\system32\eventvwr.msc" "iSCSI Initiator.lnk","Connect to remote iSCSI targets and configure connection settings.","C:\Windows\system32\iscsicpl.exe" "Memory Diagnostics Tool.lnk","Check your computer for memory problems.","C:\Windows\system32\MdSched.exe" "ODBC Data Sources (32-bit).lnk","Maintains ODBC data sources and drivers.","C:\Windows\syswow64\odbcad32.exe" "ODBC Data Sources (64-bit).lnk","Maintains ODBC data sources and drivers.","C:\Windows\system32\odbcad32.exe" "Performance Monitor.lnk","Diagnose performance issues and collect performance data.","C:\Windows\system32\perfmon.msc" "Print Management.lnk","Manages local printers and remote print servers.","C:\Windows\system32\printmanagement.msc" "RecoveryDrive.lnk","Create a recovery drive","C:\Windows\system32\RecoveryDrive.exe" "Registry Editor.lnk","Registry Editor","C:\Windows\regedit.exe" "Resource Monitor.lnk","Monitor the usage and performance of the following resources in real time: CPU, Disk, Network and Memory.","C:\Windows\system32\perfmon.exe" "Security Configuration Management.lnk","View and modify local security policy, such as user rights and audit policies.","C:\Windows\system32\secpol.msc" "services.lnk","Starts, stops, and configures Windows services.","C:\Windows\system32\services.msc" "System Configuration.lnk","Perform advanced troubleshooting and system configuration","C:\Windows\system32\msconfig.exe" "System Information.lnk","Display detailed information about your computer.","C:\Windows\system32\msinfo32.exe" "Task Scheduler.lnk","Schedule computer tasks to run automatically.","C:\Windows\system32\taskschd.msc" "Windows Defender Firewall with Advanced Security.lnk","Configure policies that provide enhanced network security for Windows computers.","C:\Windows\system32\WF.msc" "Cisco Packet Tracer Help.lnk","","C:\Program Files\Cisco Packet Tracer 8.2.1\help\default\index.htm" "Cisco Packet Tracer.lnk","","C:\Program Files\Cisco Packet Tracer 8.2.1\bin\PacketTracer.exe" "Qt Linguist.lnk","","C:\Program Files\Cisco Packet Tracer 8.2.1\bin\linguist.exe" "Saves.lnk","","C:\Program Files\Cisco Packet Tracer 8.2.1\saves" "Uninstall Cisco Packet Tracer.lnk","","C:\Program Files (x86)\Cisco Packet Tracer 8.2.1\unins000.exe" "WebCopy Samples.lnk","Sample WebCopy projects","C:\Users\Home\AppData\Roaming\Cyotek\WebCopy\samples"

	"WebCopy.lnk","Create local copies of websites","C:\Program Files\Cyotek\WebCopy\cyowcopy.exe" "Dell OS Recovery Tool.lnk","","C:\Program Files (x86)\Dell\OS Recovery Tool\DellOSRecoveryTool.exe" "Dynamips Hypervisor.lnk","","C:\Program Files\GNS3\dynamips-start.cmd" "GNS3.lnk","","C:\Program Files\GNS3\gns3.exe" "Loopback Manager.lnk","","C:\Program Files\GNS3\loopback-manager.cmd" "Network device list.lnk","","C:\Program Files\GNS3\network-device-list.cmd" "Uninstall.lnk","","C:\Program Files\GNS3\Uninstall.exe" "VMnet Manager.lnk","","C:\Program Files\GNS3\vmnet-manager.cmd" "VPCS.lnk","","C:\Program Files\GNS3\vpcs-start.cmd" "Website.lnk","","C:\Program Files\GNS3\GNS3.url" "Edit.lnk","Launch Edit for Windows","C:\Program Files (x86)\Sony\Imaging Edge\Edit.exe" "Remote.lnk","Launch Remote for Windows","C:\Program Files (x86)\Sony\Imaging Edge\Remote.exe" "Viewer.lnk","Launch Viewer for Windows","C:\Program Files (x86)\Sony\Imaging Edge\Viewer.exe" "PyCharm 2023.2.lnk","","C:\Program Files\JetBrains\PyCharm 2023.2\bin\pycharm64.exe" "System Update.lnk","","C:\Program Files (x86)\Lenovo\System Update\tvsu.exe" "Database Compare.lnk","Compare versions of an Access database.","C:\Program Files (x86)\Microsoft Office\root\Client\AppVLP.exe" "Office Language Preferences.lnk","Change the language preferences for Office applications.","C:\Program Files (x86)\Microsoft Office\root\Office16\SETLANG.EXE" "Skype for Business Recording Manager.lnk","Manage all your Skype for Business recordings in one place.","C:\Program Files (x86)\Microsoft Office\root\Office16\OcPubMgr.exe" "Spreadsheet Compare.lnk","Compare versions of an Excel workbook.","C:\Program Files (x86)\Microsoft Office\root\Client\AppVLP.exe" "Telemetry Log for Office.lnk","View critical errors, compatibility issues and workaround information for your Office solutions by using Office Telemetry Log.","C:\Program Files (x86)\Microsoft Office\root\Office16\msoev.exe" "qBittorrent.lnk","","C:\Program Files\qBittorrent\qbittorrent.exe" "Uninstall qBittorrent.lnk","","C:\Program Files (x86)\qBittorrent\uninst.exe" "Imaging Edge Desktop.lnk","","C:\Program Files (x86)\Sony\Imaging Edge Desktop\ied.exe" "Task Manager.lnk","Manage running apps and view system performance","C:\Windows\system32\taskmgr.exe" "Thunderbolt™ Software.lnk","","C:\Program Files (x86)\Intel\Thunderbolt Software\Thunderbolt.exe" "Documentation.lnk","","C:\Program Files\VideoLAN\VLC\Documentation.url" "Release Notes.lnk","","C:\Program Files\VideoLAN\VLC\NEWS.txt"

	"VideoLAN Website.lnk","","C:\Program Files\VideoLAN\VLC\VideoLAN Website.url" "VLC media player - reset preferences and cache files.lnk","","C:\Program Files\VideoLAN\VLC\vlc.exe" "VLC media player skinned.lnk","","C:\Program Files\VideoLAN\VLC\vlc.exe" "VLC media player.lnk","","C:\Program Files\VideoLAN\VLC\vlc.exe" "Jupyter Lab (Visual Python).lnk","Run Jupyter Lab on Visual Python environment","C:\Users\Home\visualpython\bin\run_vp_lab.bat" "Jupyter Notebook (Visual Python).lnk","Run Jupyter Notebook on Visual Python environment","C:\Users\Home\visualpython\bin\run_vp_notebook.bat" "Uninstall Visual Python-2.4.5.1.lnk","","C:\Users\Home\visualpython\unins000.exe" "Visual Python Prompt.lnk","Run Prompt on Visual Python environment","C:\Users\Home\visualpython\bin\run_vp_prompt.bat" "Command Prompt for vctl.lnk","","C:\Windows\System32\cmd.exe" "Virtual Network Editor.lnk","","C:\Program Files (x86)\VMware\VMware Workstation\vmnetcfg.exe" "VMware Workstation 17 Player.lnk","","C:\Program Files (x86)\VMware\VMware Workstation\vmplayer.exe" "VMware Workstation Pro.lnk","","C:\Program Files (x86)\VMware\VMware Workstation\vmware.exe" "Windows PowerShell ISE (x86).lnk","Windows PowerShell Integrated Scripting Environment. Performs object-based (command-line) functions","C:\Windows\syswow64\WindowsPowerShell\v1.0\PowerShell_ISE.exe" "Windows PowerShell ISE.lnk","Windows PowerShell Integrated Scripting Environment. Performs object-based (command-line) functions","C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe" "Console RAR manual.lnk","Process RAR, ZIP and other archive formats","C:\Program Files (x86)\WinRAR\Rar.txt" "What is new in the latest version.lnk","Process RAR, ZIP and other archive formats","C:\Program Files (x86)\WinRAR\WhatsNew.txt" "WinRAR help.lnk","Process RAR, ZIP and other archive formats","C:\Program Files (x86)\WinRAR\WinRAR.chm" "WinRAR.lnk","Process RAR, ZIP and other archive formats","C:\Program Files (x86)\WinRAR\WinRAR.exe"

File name	C:\Malany\malwareinfo.csv	Location	C:\Malany
Purpose			
Code	1,Case,Information 2,Case,Complete information		

File name	C:\Malany\options.csv	Location	C:\Malany
Purpose	Important questions prepopulated in the file		

Code	Analyse this code and explain. Does this code appear to be a malware? In which Language has the code been written? Identify the codes purpose. What are its capabilities? How does the malware spread? What are the weaknesses of this code? How can this malware be detected and prevented? What is the malware C&C (command and control) server? What is the malware payload? What are the malware evasion techniques? What are the malware anti-analysis techniques? Extracts features from the malware binary. Analyses the malware binary for malicious patterns. Generates a report on the malware.

File name	C:\Malany\Programlocation.csv	Location	C:\Malany
Purpose	All the location of programs on system. This can be cleaned and paths as dictated bythe header of the file can be replaced.		
Code	"Name","Description","Location" "Access.lnk","Build a professional app quickly to manage data.","C:\Program Files (x86)\Microsoft Office\root\Office16\MSACCESS.EXE" "Audacity.lnk","","C:\Program Files\Audacity\Audacity.exe" "DxO PhotoLab 6.lnk","","C:\Program Files (x86)\DxO\DxO PhotoLab 6\DxO.PhotoLab.exe" "Excel.lnk","Easily discover, visualize, and share insights from your data.","C:\Program Files (x86)\Microsoft Office\root\Office16\EXCEL.EXE" "GIMP 2.10.34.lnk","GIMP 2.10.34","C:\Program Files\GIMP 2\bin\gimp-2.10.exe" "Google Chrome.lnk","Access the Internet","C:\Program Files (x86)\Google\Chrome\Application\chrome.exe" "Imaging Edge Desktop.lnk","Launch Imaging Edge Desktop for Windows","C:\Program Files (x86)\Sony\Imaging Edge Desktop\ied.exe" "Immersive Control Panel.lnk","Change settings and customize the functionality of your computer","C:\Windows\System32\Control.exe" "Microsoft Edge.lnk","Browse the web","C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" "Notepad++.lnk","","C:\Program Files\Notepad++\notepad++.exe" "OneDrive.lnk","Keep your most important files with you wherever you go, on any device.","C:\Program Files (x86)\Microsoft OneDrive\OneDrive.exe" "OneNote.lnk","Take notes and have them when you need them.","C:\Program Files (x86)\Microsoft Office\root\Office16\ONENOTE.EXE"		

	"Outlook.lnk","Manage your email, schedules, contacts, and to-dos.","C:\Program Files (x86)\Microsoft Office\root\Office16\OUTLOOK.EXE" "PC Health Check.lnk","PC Health Check","C:\Program Files (x86)\PCHealthCheck\PCHealthCheck.exe" "PowerPoint.lnk","Design and deliver beautiful presentations with ease and confidence.","C:\Program Files (x86)\Microsoft Office\root\Office16\POWERPNT.EXE" "Publisher.lnk","Create professional-grade publications that make an impact.","C:\Program Files (x86)\Microsoft Office\root\Office16\MSPUB.EXE" "Skype for Business.lnk","Connect with people everywhere through voice and video calls, Skype Meetings, and IM.","C:\Program Files (x86)\Microsoft Office\root\Office16\lync.exe" "Spyder.lnk","","C:\Program Files\Spyder\Python\pythonw.exe" "Wireshark.lnk","The Wireshark Network Protocol Analyser","C:\Program Files\Wireshark\Wireshark.exe" "Word.lnk","Create beautiful documents, easily work with others, and enjoy the read.","C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE" "Zotero.lnk","","C:\Program Files (x86)\Zotero\zotero.exe" "Speech Recognition.lnk","Dictate text and control your computer by voice.","C:\Windows\Speech\Common\sapisvr.exe" "Math Input Panel.lnk","Math Input Panel","C:\Program Files (x86)\Common Files\Microsoft Shared\Ink\mip.exe" "Notepad.lnk","Creates and edits text files using basic text formatting.","C:\Windows\system32\notepad.exe" "Paint.lnk","Create and edit drawings.","C:\Windows\system32\mspaint.exe" "Quick Assist.lnk","Connect to another user's computer to help troubleshoot problems","C:\Windows\system32\quickassist.exe" "Remote Desktop Connection.lnk","Use your computer to connect to a computer that is located elsewhere and run programs or access files.","C:\Windows\system32\mstsc.exe" "Snipping Tool.lnk","Capture a portion of your screen so you can save, annotate, or share the image.","C:\Windows\system32\SnippingTool.exe" "Steps Recorder.lnk","Capture steps with screenshots to save or share.","C:\Windows\system32\psr.exe" "Windows Fax and Scan.lnk","Send and receive faxes or scan pictures and documents.","C:\Windows\system32\WFS.exe" "Windows Media Player.lnk","Play digital media including music, videos, CDs, and DVDs.","C:\Program Files (x86)\Windows Media Player\wmplayer.exe" "Wordpad.lnk","Creates and edits text documents with complex formatting.","C:\Program Files (x86)\Windows NT\Accessories\wordpad.exe" "Character Map.lnk","Selects special characters and copies them to your document.","C:\Windows\system32\charmap.exe" "Component Services.lnk","Manage COM+ applications, COM and DCOM system configuration, and the Distributed Transaction Coordinator.","C:\Windows\system32\comexp.msc"

	"Computer Management.lnk","Manages disks and provides access to other tools to manage local and remote computers.","C:\Windows\system32\compmgmt.msc" "dfrgui.lnk","Optimizes files and fragments on your volumes so that your computer runs faster and more efficiently.","C:\Windows\system32\dfrgui.exe" "Disk Cleanup.lnk","Enables you to clear your disk of unnecessary files.","C:\Windows\system32\cleanmgr.exe" "Event Viewer.lnk","View monitoring and troubleshooting messages from windows and other programs.","C:\Windows\system32\eventvwr.msc" "iSCSI Initiator.lnk","Connect to remote iSCSI targets and configure connection settings.","C:\Windows\system32\iscsicpl.exe" "Memory Diagnostics Tool.lnk","Check your computer for memory problems.","C:\Windows\system32\MdSched.exe" "ODBC Data Sources (32-bit).lnk","Maintains ODBC data sources and drivers.","C:\Windows\syswow64\odbcad32.exe" "ODBC Data Sources (64-bit).lnk","Maintains ODBC data sources and drivers.","C:\Windows\system32\odbcad32.exe" "Performance Monitor.lnk","Diagnose performance issues and collect performance data.","C:\Windows\system32\perfmon.msc" "Print Management.lnk","Manages local printers and remote print servers.","C:\Windows\system32\printmanagement.msc" "RecoveryDrive.lnk","Create a recovery drive","C:\Windows\system32\RecoveryDrive.exe" "Registry Editor.lnk","Registry Editor","C:\Windows\regedit.exe" "Resource Monitor.lnk","Monitor the usage and performance of the following resources in real time: CPU, Disk, Network and Memory.","C:\Windows\system32\perfmon.exe" "Security Configuration Management.lnk","View and modify local security policy, such as user rights and audit policies.","C:\Windows\system32\secpol.msc" "services.lnk","Starts, stops, and configures Windows services.","C:\Windows\system32\services.msc" "System Configuration.lnk","Perform advanced troubleshooting and system configuration","C:\Windows\system32\msconfig.exe" "System Information.lnk","Display detailed information about your computer.","C:\Windows\system32\msinfo32.exe" "Task Scheduler.lnk","Schedule computer tasks to run automatically.","C:\Windows\system32\taskschd.msc" "Windows Defender Firewall with Advanced Security.lnk","Configure policies that provide enhanced network security for Windows computers.","C:\Windows\system32\WF.msc" "Cisco Packet Tracer Help.lnk","","C:\Program Files\Cisco Packet Tracer 8.2.1\help\default\index.htm" "Cisco Packet Tracer.lnk","","C:\Program Files\Cisco Packet Tracer 8.2.1\bin\PacketTracer.exe" "Qt Linguist.lnk","","C:\Program Files\Cisco Packet Tracer 8.2.1\bin\linguist.exe" "Saves.lnk","","C:\Program Files\Cisco Packet Tracer 8.2.1\saves"

	"Uninstall Cisco Packet Tracer.lnk","","C:\Program Files (x86)\Cisco Packet Tracer 8.2.1\unins000.exe" "WebCopy Samples.lnk","Sample WebCopy projects","C:\Users\Home\AppData\Roaming\Cyotek\WebCopy\samples" "WebCopy.lnk","Create local copies of websites","C:\Program Files\Cyotek\WebCopy\cyowcopy.exe" "Dell OS Recovery Tool.lnk","","C:\Program Files (x86)\Dell\OS Recovery Tool\DellOSRecoveryTool.exe" "Dynamips Hypervisor.lnk","","C:\Program Files\GNS3\dynamips-start.cmd" "GNS3.lnk","","C:\Program Files\GNS3\gns3.exe" "Loopback Manager.lnk","","C:\Program Files\GNS3\loopback-manager.cmd" "Network device list.lnk","","C:\Program Files\GNS3\network-device-list.cmd" "Uninstall.lnk","","C:\Program Files\GNS3\Uninstall.exe" "VMnet Manager.lnk","","C:\Program Files\GNS3\vmnet-manager.cmd" "VPCS.lnk","","C:\Program Files\GNS3\vpcs-start.cmd" "Website.lnk","","C:\Program Files\GNS3\GNS3.url" "Edit.lnk","Launch Edit for Windows","C:\Program Files (x86)\Sony\Imaging Edge\Edit.exe" "Remote.lnk","Launch Remote for Windows","C:\Program Files (x86)\Sony\Imaging Edge\Remote.exe" "Viewer.lnk","Launch Viewer for Windows","C:\Program Files (x86)\Sony\Imaging Edge\Viewer.exe" "PyCharm 2023.2.lnk","","C:\Program Files\JetBrains\PyCharm 2023.2\bin\pycharm64.exe" "System Update.lnk","","C:\Program Files (x86)\Lenovo\System Update\tvsu.exe" "Database Compare.lnk","Compare versions of an Access database.","C:\Program Files (x86)\Microsoft Office\root\Client\AppVLP.exe" "Office Language Preferences.lnk","Change the language preferences for Office applications.","C:\Program Files (x86)\Microsoft Office\root\Office16\SETLANG.EXE" "Skype for Business Recording Manager.lnk","Manage all your Skype for Business recordings in one place.","C:\Program Files (x86)\Microsoft Office\root\Office16\OcPubMgr.exe" "Spreadsheet Compare.lnk","Compare versions of an Excel workbook.","C:\Program Files (x86)\Microsoft Office\root\Client\AppVLP.exe" "Telemetry Log for Office.lnk","View critical errors, compatibility issues and workaround information for your Office solutions by using Office Telemetry Log.","C:\Program Files (x86)\Microsoft Office\root\Office16\msoev.exe" "qBittorrent.lnk","","C:\Program Files\qBittorrent\qbittorrent.exe" "Uninstall qBittorrent.lnk","","C:\Program Files (x86)\qBittorrent\uninst.exe" "Imaging Edge Desktop.lnk","","C:\Program Files (x86)\Sony\Imaging Edge Desktop\ied.exe" "Task Manager.lnk","Manage running apps and view system performance","C:\Windows\system32\taskmgr.exe"

	"Thunderbolt™ Software.lnk","","C:\Program Files (x86)\Intel\Thunderbolt Software\Thunderbolt.exe" "Documentation.lnk","","C:\Program Files\VideoLAN\VLC\Documentation.url" "Release Notes.lnk","","C:\Program Files\VideoLAN\VLC\NEWS.txt" "VideoLAN Website.lnk","","C:\Program Files\VideoLAN\VLC\VideoLAN Website.url" "VLC media player - reset preferences and cache files.lnk","","C:\Program Files\VideoLAN\VLC\vlc.exe" "VLC media player skinned.lnk","","C:\Program Files\VideoLAN\VLC\vlc.exe" "VLC media player.lnk","","C:\Program Files\VideoLAN\VLC\vlc.exe" "Jupyter Lab (Visual Python).lnk","Run Jupyter Lab on Visual Python environment","C:\Users\Home\visualpython\bin\run_vp_lab.bat" "Jupyter Notebook (Visual Python).lnk","Run Jupyter Notebook on Visual Python environment","C:\Users\Home\visualpython\bin\run_vp_notebook.bat" "Uninstall Visual Python-2.4.5.1.lnk","","C:\Users\Home\visualpython\unins000.exe" "Visual Python Prompt.lnk","Run Prompt on Visual Python environment","C:\Users\Home\visualpython\bin\run_vp_prompt.bat" "Command Prompt for vctl.lnk","","C:\Windows\System32\cmd.exe" "Virtual Network Editor.lnk","","C:\Program Files (x86)\VMware\VMware Workstation\vmnetcfg.exe" "VMware Workstation 17 Player.lnk","","C:\Program Files (x86)\VMware\VMware Workstation\vmplayer.exe" "VMware Workstation Pro.lnk","","C:\Program Files (x86)\VMware\VMware Workstation\vmware.exe" "Windows PowerShell ISE (x86).lnk","Windows PowerShell Integrated Scripting Environment. Performs object-based (command-line) functions","C:\Windows\syswow64\WindowsPowerShell\v1.0\PowerShell_ISE.exe" "Windows PowerShell ISE.lnk","Windows PowerShell Integrated Scripting Environment. Performs object-based (command-line) functions","C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell_ISE.exe" "Console RAR manual.lnk","Process RAR, ZIP and other archive formats","C:\Program Files (x86)\WinRAR\Rar.txt" "What is new in the latest version.lnk","Process RAR, ZIP and other archive formats","C:\Program Files (x86)\WinRAR\WhatsNew.txt" "WinRAR help.lnk","Process RAR, ZIP and other archive formats","C:\Program Files (x86)\WinRAR\WinRAR.chm" "WinRAR.lnk","Process RAR, ZIP and other archive formats","C:\Program Files (x86)\WinRAR\WinRAR.exe"

File name	C:\Malany\actual_API_KEY.txt	Location	C:\Malany
Purpose	Confidential key to be only used for academic purpose which will be revoked one month after the project is finished. Replace with your own API key		

Code	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

File name	C:\Malany\AllChatGpt.txt	Location	C:\Malany
Purpose	Chat sample . create the file and delete the content		
Code	Role: user Content: What is the world time today? Role: ChatGPT Content: There is no code provided. Content: Which countries are part of russia Role: ChatGPT Content: The countries that are part of Russia are Belarus, Kazakhstan, Kyrgyzstan, and Tajikistan.		

File name	C:\Malany\API-Hybrid-Analysis.txt	Location	C:\Malany
Purpose	The Api key that is to be used for Hybrid Analysis. Failed to get this module working properly but there is room for someone to change it properly		
Code	5tflh3oj1cedf0313v8sy06r869082e7l80ffmep1699e29el06ns2ox470369e7		

File name	C:\Malany\installed_packages.txt	Location	C:\Malany
Purpose	A routine to record all the programs version. Not important for main functionality		
Code	aiofiles==23.1.0 aiohttp==3.8.4 aiosignal==1.3.1 altair==5.0.1 annotated-types==0.5.0 anyio==3.7.1 argon2-cffi==21.3.0 argon2-cffi-bindings==21.2.0 arrow==1.2.3 asgiref==3.7.2 asttokens==2.2.1 async-lru==2.0.3 async-timeout==4.0.2 attrs==23.1.0 Babel==2.12.1 backcall==0.2.0		

```
backports.zoneinfo==0.2.1
beautifulsoup4==4.12.2
bleach==6.0.0
blinker==1.6.2
bottle==0.12.25
certifi==2023.5.7
cffi==1.15.1
charset-normalizer==3.2.0
click==8.1.6
clr-loader==0.2.5
colorama==0.4.6
comm==0.1.3
contourpy==1.1.0
cycler==0.11.0
darkdetect==0.8.0
debugpy==1.6.7
decorator==5.1.1
defusedxml==0.7.1
Django==4.2.3
django-embed-video==1.4.9
easy-thumbnails==2.8.5
et-xmlfile==1.1.0
exceptiongroup==1.1.2
executing==1.2.0
fastapi==0.100.0
fastjsonschema==2.18.0
ffmpy==0.3.1
filelock==3.12.2
Flask==2.3.2
fonttools==4.41.0
fqdn==1.5.1
frozenlist==1.4.0
fsspec==2023.6.0
gradio==3.38.0
gradio_client==0.2.10
h11==0.14.0
httpcore==0.17.3
httpx==0.24.1
huggingface-hub==0.16.4
idna==3.4
importlib-metadata==6.8.0
importlib-resources==6.0.0
ipykernel==6.24.0
ipython==8.12.2
isoduration==20.11.0
```

```
itsdangerous==2.1.2
jedi==0.18.2
Jinja2==3.1.2
json5==0.9.14
jsonpointer==2.4
jsonschema==4.18.4
jsonschema-specifications==2023.7.1
jupyter-events==0.6.3
jupyter-lsp==2.2.0
jupyter_client==8.3.0
jupyter_core==5.3.1
jupyter_server==2.7.0
jupyter_server_terminals==0.4.4
jupyterlab==4.0.3
jupyterlab-pygments==0.2.2
jupyterlab_server==2.23.0
kiwisolver==1.4.4
linkify-it-py==2.0.2
Markdown==3.4.3
markdown-it-py==2.2.0
MarkupSafe==2.1.3
matplotlib==3.7.2
matplotlib-inline==0.1.6
mdit-py-plugins==0.3.3
mdurl==0.1.2
messagebox==0.1.0
mistune==3.0.1
multidict==6.0.4
nbclient==0.8.0
nbconvert==7.7.2
nbformat==5.9.1
nest-asyncio==1.5.6
notebook==7.0.0
notebook_shim==0.2.3
numpy==1.24.4
openai==0.27.8
openpyxl==3.1.2
orjson==3.9.2
outcome==1.2.0
overrides==7.3.1
packaging==23.1
pandas==2.0.3
pandocfilters==1.5.0
paperclip==2.7.2
parso==0.8.3
```

```
pickleshare==0.7.5
Pillow==9.5.0
pkgutil_resolve_name==1.3.10
platformdirs==3.9.1
prometheus-client==0.17.1
prompt-toolkit==3.0.39
proxy-tools==0.1.0
psutil==5.9.5
pure-eval==0.2.2
pycparser==2.21
pydantic==2.0.3
pydantic_core==2.3.0
pydub==0.25.1
Pygments==2.15.1
pyparsing==3.0.9
pyperclip==1.8.2
PyQt5==5.15.9
PyQt5-Qt5==5.15.2
PyQt5-sip==12.12.1
PyQt5-stubs==5.15.6.0
PyQtWebEngine==5.15.6
PyQtWebEngine-Qt5==5.15.2
PySocks==1.7.1
python-dateutil==2.8.2
python-json-logger==2.0.7
python-magic==0.4.27
python-multipart==0.0.6
pythonnet==3.0.1
pytz==2023.3
pywebview==4.2.2
pywin32==306
pywinpty==2.0.11
PyYAML==6.0.1
pyzmq==25.1.0
referencing==0.30.0
requests==2.31.0
rfc3339-validator==0.1.4
rfc3986-validator==0.1.1
rpds-py==0.9.2
selenium==4.10.0
semantic-version==2.10.0
Send2Trash==1.8.2
simple_webbrowser==0.0.5
six==1.16.0
sniffio==1.3.0
```

	sortedcontainers==2.4.0 soupsieve==2.4.1 sqlparse==0.4.4 stack-data==0.6.2 starlette==0.27.0 terminado==0.17.1 tinycss2==1.2.1 tkhtmlview==0.2.0 tkinterhtml==0.7 tomli==2.0.1 toolz==0.12.0 tornado==6.3.2 tqdm==4.65.0 traitlets==5.9.0 trio==0.22.2 trio-websocket==0.10.3 typing_extensions==4.7.1 tzdata==2023.3 uc-micro-py==1.0.2 uri-template==1.3.0 urllib3==2.0.3 uvicorn==0.23.1 wcwidth==0.2.6 webcolors==1.13 webencodings==0.5.1 websocket-client==1.6.1 websockets==11.0.3 Werkzeug==2.3.6 wsproto==1.2.0 yarl==1.9.2 zipp==3.16.2

File name	C:\Malany\Malanysettings.txt	Location	C:\Malany
Purpose	Make a blank file		
Code			

File name	C:\Malany\VirusTotal_APIKEY.txt	Location	C:\Malany
Purpose	This is the API key made on VirusTotal website. Please make your own on the website and paste the API key here. Note the key will be revoked after a month of submission of program		
Code	b535e82f339d8521529af1239b489c024b7b4143cab901339b849adb1fc03a22		

File name	C:\Malany\Scandirectory.ps1	Location	C:\Malany
Purpose	Powershell script. If needs administrative access to run on your system. Make sure you have them. In case not then try to run it from powershell directory with administrative privileage from the c:\malany folder		

Code	<pre># Define the directory to search for .lnk files
	$directory = "C:\ProgramData\Microsoft\Windows\Start Menu\Programs"
	
	# Create an empty array to store the extracted data
	$lnkData = @()
	
	# Recursively search for .lnk files and extract information
	Get-ChildItem -Path $directory -File -Recurse -Filter "*.lnk" | ForEach-Object {
	  $lnkFile = $_
	
	  # Use Windows Shell COM object to access .lnk file properties
	  $shell = New-Object -ComObject WScript.Shell
	  $lnk = $shell.CreateShortcut($lnkFile.FullName)
	
	  # Extract properties
	  $lnkInfo = [PSCustomObject]@{
	  "Description" = $lnk.Description
	  "Target" = $lnk.TargetPath
	  "Name"=''
	  }
	
	  # Add the extracted data to the array
	  $lnkData += $lnkInfo
	
	  # Release COM object
	  [System.Runtime.Interopservices.Marshal]::ReleaseComObject($lnk)
	  [System.GC]::Collect()
	}
	
	# Export the extracted data to a CSV file
	$lnkData | Export-Csv -Path "C:\Malany\Programlocation.csv" -
	NoTypeInformation
	
	# Display a message indicating the completion of the operation
	Write-Host "Information extracted and saved to C:\Malany\Programlocation.csv"</pre>		

References

Ahmed, S., 2015. *Real time detection of malicious webpages using machine learning techniques.* [Online]
Available at: https://repository.londonmet.ac.uk/710/1/Shafi%20Ahmed%20-%20PhD%20Full%20Thesis.pdf
[Accessed 1 6 2023].

Amir Djenna, A. B. S. R. ,. I. ,. M., 2023. *Artificial Intelligence-Based Malware Detection, Analysis, and Mitigation.* [Online]
Available at: https://www.mdpi.com/2073-8994/15/3/677#:~:text=This%20work%20proposes%20a%20new,%2C%20SMS%20malware%2C%20and%20ransomware.
[Accessed 1 9 2023].

Domhnall, C., 2018. *Dynamic Analyses of Malware.* [Online]
Available at: https://pureadmin.qub.ac.uk/ws/portalfiles/portal/240295001/main_thesis_corrected.pdf
[Accessed 1 9 2023].

Dudipalla Vardhan Krishna,Goditi Shiva Kumar, Relli Likhith Kumar, Kommineni Manikanta,Enireddy Vamsidhar, Mohammed Moulana, 2023. *Malware Detection using Machine Learning.* [Online]
Available at: https://0-ieeexplore-ieee-org.emu.londonmet.ac.uk/stamp/stamp.jsp?tp=&arnumber=10141501&isnumber=10140203
[Accessed 1 8 2023].

Hassabis, D., 2015. *Alphago.* [Online]
Available at: https://www.deepmind.com/research/highlighted-research/alphago
[Accessed 1 9 2023].

Isabella Harford - TechTarget, 2022. *Does AI-powered malware exist in the wild? Not yet.* [Online]
Available at: https://www.techtarget.com/searchsecurity/tip/Does-AI-powered-malware-exist-in-the-wild-Not-yet
[Accessed 29 8 2023].

James Scott, S. F. I., 2017. *Signature Based Malware is Dead.* [Online]
Available at: https://informationsecurity.report/Resources/Whitepapers/920fbb41-8dc9-4053-bd01-72f961db24d9_ICIT-Analysis-Signature-Based-Malware-Detection-is-Dead.pdf
[Accessed 1 9 2023].
Kang, C., 2023. *OpenAI's Sam Altman Urges A.I. Regulation in Senate Hearing.* [Online]
Available at: https://www.nytimes.com/2023/05/16/technology/openai-altman-artificial-intelligence-regulation.html
[Accessed 1 9 2023].
Lothar Fritsch, Aws Jaber & Anis Yazidi, 2023. *An Overciew of Artificial Intgelligence used in Malware.* [Online]
Available at: https://link.springer.com/chapter/10.1007/978-3-031-17030-0_4
[Accessed 1 9 2023].
Natalie, 2023. *What is ChatGPT?.* [Online]
Available at: https://help.openai.com/en/articles/6783457-what-is-chatgpt
[Accessed 6 9 2023].
Ovide, S., 2023. *Remember Zoom-bombing? This is how Zoom tamed meeting intrusions..* [Online]
Available at: https://www.washingtonpost.com/technology/2023/01/24/zoom-bombing-prevention-tips/
[Accessed 1 9 2023].
Pa, Y. M. P., Tanizaki, S. & Tetsui Kou, M. v. E. Y. M., 2023. *An Attacker's Dream? Exploring the Capabilities of ChatGPT for Developing Malware.* [Online]
Available at: https://dl.acm.org/doi/pdf/10.1145/3607505.3607513
[Accessed 6 9 2023].
Sharma, P., 2022. *Top 10 Techniques for Deep Learning that you Must Know!.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2022/01/top-10-techniques-for-deep-learning-that-you-must-know/#:~:text=Generative%20Adversarial%20Networks,generated%20by%20the%20Generator%20Network.
[Accessed 4 9 2023].

Smith, M., 2022. *Quantum computing: Definition, facts & uses.* [Online]
Available at: https://www.livescience.com/quantum-computing#:~:text=Quantum%20computing%20is%20a%20new%20generation%20of%20technology%20that%20involves,supercomputer%2010%2C000%20years%20to%20accomplish.
[Accessed 1 9 2023].
Stephen Casper, X. D. C. S. T. K. G. J. S. J. R. R. F. T. K. D. L. P. F. T. W. S. M. C.-R. S. M. C. A. P. P. C., 2023. *Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback.* [Online]
Available at: Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback
[Accessed 6 9 2023].
Taylor, A., 2000. *Software: Running Programs.* [Online]
Available at: https://web.stanford.edu/class/cs101/software-1.html#:~:text=Operating%20system%20starts%20and%20stops%20programs%20Each%20program,system%2C%20programs%20can%20read%20and%20write%20data%20here
[Accessed 6 9 2023].
Violino, B., 2022. *Artificial intelligence is playing a bigger role in cybersecurity, but the bad guys may benefit the most.* [Online]
Available at: https://www.cnbc.com/2022/09/13/ai-has-bigger-role-in-cybersecurity-but-hackers-may-benefit-the-most.html
[Accessed 9 2 2023].
Wikipedia, 2023. *HIEW.* [Online]
Available at: https://en.wikipedia.org/wiki/HIEW
[Accessed 23 8 2023].
Xiong, Y., 2022. *Can you remember everything all at once?.* [Online]
Available at: https://afurrybear.com/assets/pdf/EssayRotation_Yirong.pdf
[Accessed 2 8 2023].

Table of Figures