

Machine Learning based Advanced Persistent Threats (APT) detection in Windows



By

Madiha Hassan

(Registration No: 00000400918)

A thesis submitted to the National University of Sciences and Technology, Islamabad,

in partial fulfillment of the requirements for the degree of

Master of Science in
Information Security

Supervisor: Dr. Waseem Iqbal

Military College of Signals (MCS)

National University of Sciences & Technology (NUST)

Islamabad, Pakistan

(2024)

THESIS ACCEPTANCE CERTIFICATE

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS Thesis written by Madiha Hassan, Registration No. 00000400918, of Military College of Signals has been vetted by undersigned, found complete in all respects as per NUST Statutes/Regulations/MS Policy, is free of plagiarism, errors, and mistakes and is accepted as partial fulfillment for award of MS degree. It is further certified that necessary amendments as pointed out by GEC members and local evaluators of the scholar have also been incorporated in the said thesis.

Signature: _____

Name of Supervisor Dr M.M Waseem Iqbal

Date: 5/7/24

Signature (HOD): _____

HoD
Information Security
Military College of Sigs

Date: 5/7/24

Signature (Dean/Principal) _____

Date: 5/7/24

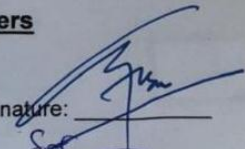
Brig
Dean, MCS (NUST)
(Asif Masood, PhD)

NATIONAL UNIVERSITY OF SCIENCES & TECHNOLOGY
MASTER THESIS WORK

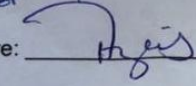
We hereby recommend that the dissertation prepared under our supervision by Madiha Hassan, MSIS-21 Course Regn No 00000400918 Titled: "Machine Learning based Advanced Persistent Threats (APT) detection in Windows." be accepted in partial fulfillment of the requirements for the award of MS Information Security degree.

Examination Committee Members

1. Name: Dr Muhammad Faisal Amjad

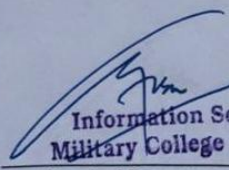
Signature: 

2. Name: Dr Syed Qaiser Ali Shah

Signature: 

Supervisor's Name: Dr M.M Waseem Iqbal

Signature: 

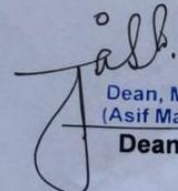

HoD
Information Security
Military College of Sigs
Head of Department

Date: _____

1-7-2024
Date

COUNTERSIGNED

Date: 2/7/24


Brig
Dean, MCS (NUST)
(Asif Masood, Phd)
Dean

CERTIFICATE OF APPROVAL

CERTIFICATE OF APPROVAL

This is to certify that the research work presented in this thesis, entitled "Machine Learning based Advanced Persistent Threats (APT) detection in Windows." was conducted by Madiha Hassan under the supervision of Dr. Mian Muhammad Waseem Iqbal. No part of this thesis has been submitted anywhere else for any other degree. This thesis is submitted to the Military College of Signals, National University of Science & Technology Information Security Department in partial fulfillment of the requirements for the degree of Master of Science in Field of Information Security Department of information security National University of Sciences and Technology, Islamabad.

Student Name: Madiha Hassan

Signature: Madiha Hassan

Examination Committee:

a) External Examiner 1: Dr M. Faisal Amjad (MCS)

Signature: 1-7-2024

b) External Examiner 2: Dr. Syed Qaiser Ali Shah (MCS).

Signature: for

Name of Supervisor: Dr. M.M Waseem Iqbal

Signature: Dr. M.M Waseem Iqbal

Name of Dean/HOD: Dr Muhammad Faisal Amjad

Signature: Dr Muhammad Faisal Amjad

HoD
Information Security
Military College of Sigs

AUTHOR'S DECLARATION

AUTHOR'S DECLARATION

I **Madiha Hassan** hereby state that my MS thesis titled Machine Learning based **Advanced Persistent Threats (APT) detection in Windows** is my own work and has not been submitted previously by me for taking any degree from National University of Sciences and Technology, Islamabad or anywhere else in the country/ world. At any time if my statement is found to be incorrect even after I graduate, the university has the right to withdraw my MS degree.

Student Signature: Madiha Hassan

Name: **Madiha Hassan**

Date: 1-7-2024

PLAGIARISM UNDERTAKING

PLAGIARISM UNDERTAKING

I solemnly declare that research work presented in the thesis titled **Machine Learning based Advanced Persistent Threats (APT) detection in Windows** is solely my research work with no significant contribution from any other person. Small contribution/ help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero-tolerance policy of the HEC and National University of Sciences and Technology (NUST), Islamabad towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of MS degree, the University reserves the rights to withdraw/revoke my MS degree and that HEC and NUST, Islamabad has the right to publish my name on the HEC/University website on which names of students are placed who submitted plagiarized thesis.

Student Signature: Madiha Hassan

Name: Madiha Hassan

Date: 1-7-2024

DEDICATION

To my parents, whose love, encouragement, and sacrifices have paved the way for all my achievements. Your unwavering belief in me has been a constant source of strength and inspiration. This thesis is dedicated to you.

To my brother and sisters, who have been my guiding lights and steadfast supporters throughout my academic journey. Your support has been invaluable.

To my mentor and advisor, Dr. Mian Muhammad Waseem Iqbal, for his profound wisdom, guidance, and faith in my abilities. Your mentorship has been instrumental in shaping my path as a researcher.

To my friends, who have been the pillars of strength and a source of joy throughout this academic endeavor. Your friendship has made this journey not only bearable but also profoundly enjoyable.

ACKNOWLEDGEMENTS

First and foremost, I am profoundly grateful to Almighty Allah, the most gracious and the most merciful, who bestowed upon me health, wisdom, knowledge, and the power of communication. I owe a great deal of gratitude to my supervisor, whose valuable guidance, encouragement, and supervision made it possible for me to undertake this project and complete my training in this area. I am deeply thankful for the support of my colleagues, especially Syed Sohaib Karim, Major Amara Riaz and Rabiya Tariq whose cooperation was instrumental in the completion of my thesis. I am also immensely grateful to my parents, brother and sisters for their motivation, sacrifice, cooperation, love, and affection, which have helped me overcome my difficulties and enabled me to complete this research work.

Table of contents

ACKNOWLEDGEMENTS	I
LIST OF TABLES	IV
LIST OF FIGURES	VI
LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS	VII
ABSTRACT	VIII
CHAPTER 1:	1
INTRODUCTION	1
1.1 Overview	1
1.2 Motivation	2
1.3 Problem Statement	4
1.4 Research Objectives	5
1.5 Relevance to National Needs	5
1.6 Area of Application	6
1.7 Advantages	7
1.8 Thesis Organization	8
CHAPTER 2:	9
LITERATURE REVIEW	9
2.1 Introduction	9
2.2 Related Work	9
2.3 Types Of Advanced Persistent Threat	17
CHAPTER 3	20
PROPOSED METHODOLOGY	20
3.1 Introduction	20
3.2 Overview	20
3.3 Data Collection	21
3.4 Dynamic Malware Analysis Using Cuckoo Sandbox	21
3.5 Preprocessing	22
3.6 Feature Extraction	22
3.7 Model Training	23
3.8 Validation	24
3.9 Justification of Proposed Methodology	25
3.10 Summary:	28
CHAPTER 4	29
IMPLEMENTATION AND EXPERIMENTAL RESULTS	29

4.1	Introduction:	29
4.2.1	Creation of Dataset:	30
4.3	Cuckoo	30
4.3.1	Modular Architecture of Cuckoo	30
4.3.2	Basic Static tools Used by Cuckoo	31
4.3.2.1	File Analysis Tools	31
4.3.2.2	Malware identification tools	31
4.3.2.3	Hashing Devices	31
4.3.2.4	Code Analysis tools	31
4.3.3	Basic Dynamic tools Used by Cuckoo	32
4.3.3.1	Monitoring and instrumentation tool	32
4.3.3.2	Network analysis tool	32
4.3.3.3	Behavioral Analysis Tool	32
4.3.3.4	Memory Analysis Tools	33
4.3.4	Reporting and visualization	33
4.3.5	Virtualization Technology	33
4.3.6	External Communities	33
4.4	Installation of Cuckoo:	34
4.5.1	Mounting Google Drive:	38
4.5.2	Importing necessary Libraries	38
4.5.3	Defining function to read DOCX Document:	38
4.5.4	Defining a Function to Count Features using Regular Expressions:	38
4.5.5	Characterizing a function to Compose Results to a CSV Document:	38
4.5.6	Getting a list of DOCX Records from the directory	38
4.5.7	Processing Each file and Gathering feature Counts	39
4.5.8	Composing the results to a CSV Record	39
4.6	Preprocessing:	41
4.6.1	Data Cleaning	41
4.6.2	Eliminating Missing Data Values	41
4.6.3	Remove duplicate rows	43
4.6.4	Check Data type of each column	44
4.6.5	Encoding	44
4.6.6	Normalizing	45
4.6.7	Model Selection	45
4.6.8	Analysis	64
CHAPTER 5		69
CONCLUSION		69
5.1	Future Work:	70
BIBLIOGRAPHY		71

LIST OF TABLES

Table 1 :Some examples of how LOTL are used by adversaries[2]	14
Table 2 :Comparative Analysis of Previous work	15
Table 3 : Total Samples	37
Table 4 :List of features extracted from Registry Features.....	39
Table 5 :List of features extract from file activity	39
Table 6 :Lists of feature to check if there is activity of task schedule	40
Table 7 :List of other Extracted features.....	40
Table 8 :Accuracy of RF On Label with diff no of tree.....	46
Table 9 : Accuracy of RF On Hot Encoding with diff no of tree.....	47
Table 10 :Accuracy of RF On Frequency with diff no of tree	48
Table 11 :Accuracy of MLP On Label with diff layer size.....	49
Table 12 :Accuracy of MLP On frequency with diff layer size.....	51
Table 13 :Accuracy of MLP On Hot Encoding with diff Layer size	52
Table 14 :Accuracy of LSTM On Label Encoding with diff configuration	54
Table 15 :Accuracy of LSTM On frequency Encoding with diff configuration.....	54
Table 16 :Accuracy of LSTM on hot Encoding with diff configuration	55
Table 17 :Accuracy of NB On Label Encoding with diff configuration.....	57
Table 18 :Accuracy of NB On frequency Encoding with diff configuration.....	58
Table 19 :Accuracy of NB On Hot Encoding with diff configuration.....	59
Table 20 :Accuracy of CNN on Label Encoding with diff Hidden layers.....	61
Table 21 :Accuracy of CNN on frequency Encoding with diff Hidden layers	62

Table 22 :Accuracy of CNN on Hot Encoding with diff Hidden layers.....62

Table 23 :Performance Comparison Machine Learning Algorithms.....64

LIST OF FIGURES

Figure 1 General Workflow of Advanced Persistent Threat.....	11
Figure 2: Analysis of Windows Security System	15
Figure 3: Workflow of APT28/29.....	17
Figure 4:Proposed Methodology.....	24
Figure 5:Architecture Of Cuckoo	36
Figure 6: Web Interface of cuckoo	37
Figure 7: Cuckoo while Running	37
Figure 8:Heatmap of missing values before Handling.....	42
Figure 9:Heatmap of missing values after handling	43
Figure 10:Checking Duplicate Rows	43
Figure 11:Visualization of Accuracy of RF On label Encoding.....	47
Figure 12: Visualization of Accuracy of RF On Hot Encoding.....	48
Figure 13: Visualization of Accuracy of RF On Frquency Encoding.....	49
Figure 14:Visualization of Accuracy of MLP On Label Encoding	51
Figure 15:Visualization of Accuracy of MLP On Frequency Encoding.....	52
Figure 16: Visualization of Accuracy of MLP On Hot Encoding	53
Figure 17:Visualization of Accuracy of LSTM On Label Encoding.....	54
Figure 18:Visualization of Accuracy of LSTM On Frequency Encoding	55
Figure 19:Visualization of Accuracy of LSTM On hot Encoding.....	56
Figure 20:Visualization of Accuracy of NB On Label Encoding.....	58
Figure 21:Visualization of Accuracy of NB On Frequency Encoding	59
Figure 22:Visualization of Accuracy of NB On HotEncoding	60
Figure 23:Visualization of Accuracy of CNN On Label Encoding	62
Figure 24:Visualization of Accuracy of CNN On Frequency Encoding	62
Figure 25:Visualization of Accuracy of CNN On Hot Encoding	63
Figure 26:Accuracy of diff ML Model using diff encoding techniques	66
Figure 27: Comparison of MLP over diff Datasets.....	68

LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS

APT Advanced Persistent Threat

LOTL Living OFF the Land Techniques

ML Machine Learning

MLP Multi Layer Perceptron

RF Random Forest

CNN Convolution Neural Network

Abstract

Advanced Persistent Threats (APTs) turns into significant and ongoing concern of cyber security as the smart threat actors use advance ways to infiltrate and persist within targeted systems for a longer period of time. Every APT attack goes through various stages before its completion making it difficult for conventional signature-based techniques and rule-based intrusion detection systems to recognize these elusive threats. Machine learning (ML) techniques are utilized in recent past for the detection of the APTs in Windows environment using Network based detection. Due to the limited information extracted from network data, less known features are used by ML algorithms for detection therefore evading security systems. Therefore, this research focus on finding sophisticated new features that can aid in identifying the latest APTs in Windows environment. ML models like Random forest (RF), Convolution Neural Network(CNN), Naïve Bayes (NB), Multi-Layer Perceptron (MLP) and Long short-term Memory (LSTM) with different encoding techniques (Frequency, Label, and Hot) are utilized for detection of the Windows APTs. Results show that MLP model using Label encoding, accomplished the highest accuracy i.e., (95.45%) and F1 score (95.267%), highlighting the potential of neural network in APT detection. These result validate the proposed model's efficiency, feature selection, and data pre-processing in building effective Windows based security solutions.

Keywords: Advance Persistent Threat (APT), Machine Learning, Detection, Windows, Classification

Chapter 1:

Introduction

1.1 Overview:

In a time overwhelmed by advanced interconnectedness, the area of cybersecurity is consistently challenged by the emergence of sophisticated and target cyber threats. Among these threats, Advanced Persistent threats (APTs) stand apart as exceptionally covert and persistent adversaries fit for invading organizations, exfiltrating sensitive data, and working undetected for extended period. Traditional security measures often struggle to recognize APTs because of their capacity to adjust and develop their strategies powerfully.

The objective of attack method is to exfiltrate information or dishonor institutes, associations, legislatures, and so on. Practically all key government associations, organizations can become a victim of Advanced persistent threats. The life cycle and characteristics of advanced persistent threat include the stages: Initial Compromise, Establish Foothold, Escalate Privileges, Initial Reconnaissance, Move Laterally, Maintain Presence and Complete Mission. According to statistics, APT attacks often focus on exploiting vulnerabilities and weaknesses of system and then afterward growing into the system. Therefore, to prevent APT attacks, it is necessary to build systems to monitor and detect APTs from the user side [1]. Most secured systems keep strong boundary as of strength between the web and the intranet, in this way attacker focuses on that approach has behind the organization security capabilities (e.g., firewalls, IPS and so

on.). It is difficult for attacker to send attacks against resources that resides in the intranet. [4].

The rising complexity and scale of cyber threats require inventive ways to deal with sustain network and host safeguards. Machine learning, with its capacity to analyze the immense amount of data and identify patterns, holds extraordinary commitment in expanding protection. This proposal centers around utilizing machine learning techniques for the detection of APTs, planning to improve the resilience of organizations against these sophisticated APTs.

Advanced persistent threats (APTs) utilizing Machine learning includes algorithms and models to recognize patterns and anomaly-based activity of these sophisticated and persistent attacks. APTs can evade traditional security measures, operate secretly, and keep a presence within a target network.

In recent years, Machine learning has become a potential tool for APTS detection due to its ability to evaluate massive amount of data and find a tiny pattern that pin to malicious activities. So, by analyzing the different APT campaigns, finding and extracting the most important features for APT detection in windows by use of different machine learning models. Some research utilizes hybrid models for higher efficiency and accuracy.

1.2 Motivation:

Information security has turned into a need for all organizations because of the rising number of new types of malware and attacks. Advanced Persistent Threats (APTs) are a class of threats identify by slow and low development, determined to take information

without getting caught. APT assailants utilize advanced tools to enter the target organization and gradually expand their foothold, trading important data to their command-and-control center. APT attacks are normally performed by very much subsidized attackers and can make irrecoverable damage to the victim organization on the off chance that not identified in time [1].

Advanced persistent threat has key characteristics include persistence, advanced techniques [2] (used of windows legitimate tools like binaries, code injection), long lasting impact on target machine and evade the windows standard security safeguards. Only ten percent of attacks were reported as warning or malicious, Seventy percent out of 100 of attacks on Windows endpoints failed and twenty percent of attacks were recognized and reported [2] . There are some Windows applications, which are extensively abused for the purpose of processing injections.

Detection of APTs in windows by machine learning algorithm can contribute to the advancement in cyber security and lead to more robust, efficient, and adaptable detection that offers better protection against APTs and other advanced threats. Machine learning algorithms succeed at recognizing subtle patterns and abnormalities inside large datasets. By investigating different aspects of Windows event logs, network traffic, and user activities, Machine Learning models can recognize APTs with more prominent accuracy compared with traditional rule-based or signature-based systems.

APTs are known for their sophisticated and dynamic nature, continually evolving to bypass conventional security measures. Machine learning models can adjust to new attack patterns and methods, providing a proactive defense that develops with the changing threat landscape. Machine Learning models support constant monitoring and

learning, guaranteeing that the detection system remains up-to-date and effective in the face of emerging threats [5].

1.3 Problem Statement:

Detection of advanced persistent threats (APTs) in Windows environment using machine learning revolve around the advancing and sophisticated nature of cyber threats that traditional security measures struggle to address sufficiently is the main challenge. APTs, organized by highly skilled attackers, use advanced techniques to invade, infiltrate and persist inside Windows-based networks, often avoiding detection by traditional security system. The challenge is to develop a successful and adopted solution that can precisely distinguish APTs by analyzing the diverse data, including Windows event logs, network traffic, and user exercises.

Traditional security measures, based on signature rules, are lacking to identify these powerful threats. It might create countless false positives, leading to false alert which cause useless fatigue among security examiners. Windows environments generate vast amounts of data, including event logs as it has some important information and network traffic, but the main challenge is to integrate this information with machine learning models. As advanced persistent threats persist over network over the extended period of time So, it is necessary to continuous monitoring and updating of detection models.

So, there is a need of a solution that can adopt and learn from emerging techniques of advanced persistent threats, performed details behavioral analysis and identify anomalies based activities of advanced persistent threats in windows environment, handle

and analyze huge amount of data to distinguish between normal activities from potential APT related activities, requiring scalable and effective machine learning models. Also, models should be designed that not only reduced false positive but maintain a high level of accuracy in APT detection, can adapt easily to new APT behavior and maintain relevance in the face of new emerging threats. The improvement of an effective APT detection solution must balance accuracy with effectiveness, versatility, and a guarantee to ethical practices in taking care of sensitive information.

1.4 Research Objectives:

The principal fundamentals for this research thesis are summed up in the following broad scope of objectives:

RO1: To study and analyze the behavior of advanced persistent threats and dig into complexity of Windows Operating system like its fundamentals, file system, windows event log, registry and memory analysis.

RO2: To propose and develop an enhanced Machine learning based mechanism to detect Advanced persistent threats in Windows.

RO3: Comprehensive analysis of proposed methodology leverages ML Model, reverse engineer, behavioral analysis, continuous learning with existing solution available.

1.5 Relevance to National Needs:

Machine Learning based Advanced Persistent threats in Windows is profoundly relevant to national need in light of multiple factors:

- APTs are sophisticated and targeted cyber threat frequently associated with nation state actors. Although governmental and defense-related institutes are the main targets

of APTS. The country might strengthen its defenses against cyber threats such as espionage and data breaches by using machine learning to increase the effectiveness of antivirus software and prevent the compromising of vital national security data.

- The energy, transportation, financial, and medical services areas are among the most vulnerable against the serious threats presented by Advanced Persistent threats (APTs). Making ML based detection procedures can help with safeguarding this critical system against cyberattacks that could interfere with the arrangement of key services.
- System based on Machine Learning can give fast incident response on time by identifying threats in real time. This flexibility is fundamental for decreasing the impacts of cyberattacks and halting their escalation.
- While cyber threats are continually changing, a proactive methodology that utilizes machine learning can strengthen a country's protections against APTs and other new cyber security issues.
- By resolving this issue, colleges, legislative associations, and network safety organizations in the private sector are urged to team up on research projects. Cutting edge innovation and arrangements might be created in this cooperative setting.

1.6 Area of Application:

APT detection is adaptable and ranges across sectors where the protection of sensitive information and critical systems is vital. It plays a significant role in strength cyber security measures and tending to the evolving landscape of advanced cyber threats.

- ML based detection strength the enterprise Cybersecurity

- Protecting classified information, military network and defense system using ML enhance the Governmental and Defense security.
- Increase the security postures of bank and other financial sectors.
- In Medical Applications ensuring confidentiality and integrity of sensitive medical information.
- Organizations offering cyber security can use machine learning for detection to give advanced threat detection capacities to their clients.

1.7 Advantages:

Carrying out machine Learning-based advanced persistent threat (APT) detection in Windows offers a few benefits across different areas.

- Machine Learning models excel at investigating huge datasets and recognizing patterns. This enables early detection of APTs, even in situations where traditional signature-based systems might come up short. Early discovery is urgent for keeping APTs from causing extensive harm.
- Machine Learning models can be incorporated with automated response systems to quickly mitigate identified threats. This diminishes the response time, restricting the effect of APTs and keeping them from spreading across the organization.
- ML based APT detection gives constant monitoring of system activities. This proactive methodology permits security groups to recognize and response to threats in real time, reducing the probability of successful APT attack.

- Machine learning models can be prepared to decrease false positive by recognizing normal way of behaving from suspicious activities. This helps security teams focus around real threats, limiting the time and assets spent on investigating false alarms.

1.8 Thesis Organization:

The research work has been organized and distributed in the following chapters listed below. Also Fig. 1.3, represents the layout of this thesis which is described in detail in this section.

Chapter 1: A brief introduction is given in this chapter. Problem statement followed by research objectives, relevance to national need, area of application and its advantages are elaborated.

Chapter 2: This chapter describes related works carried out for detection and define the flow and stages of Advanced Persistent Threat

Chapter 3: This chapter explains the proposed model in detail. It is covering the major research objectives of this research by explaining all phases of proposed model in detail

Chapter 4: This Chapters presents the detail practical implementation, result and analysis of proposed methodology with previous used method

Chapter 5: This Chapter sums up the research with conclusions drawn and discusses the future aspects of this research.

Chapter 2:

Literature Review

2.1 Introduction:

Detection of advanced persistent threats (APTs) in Windows is a challenge in cybersecurity. Using Machine Learning (ML) for APT detection presents a strong and versatile way to deal with identifying sophisticated and stealthy threats. The integration of ML algorithms inside Windows security system improves the capacity to recognize abnormal patterns, ways of behaving, and activities that might connote the presence of APT.

The many existing mitigation and processes/strategies, their applicability, and capabilities, as well as an intensive assessment of what has previously been done and what should be thought of, are all introduced top to bottom. At the determination, a table posting all difficulties and mitigation measures are given.

2.2 Related Work:

Indeed, Machine Learning (ML) has arisen as an important tool in the domain of cybersecurity, particularly in the identification of Advanced Persistent Threats (APTs). APTs address sophisticated and covert cyber threats orchestrated by well-funded and highly skilled attackers, frequently with the target of prolong penetration and information exfiltration. Detection and solution of these threats as quickly as possible is basic to shielding sensitive data and keeping up with integrity of computer systems.

Researchers have taken critical steps in using machine Learning as well as deep learning for most sophisticated APT attacks by focusing on different phases of the APT

lifecycle as it possess the most sophisticated nature and find a new ways to penetrate into target system as it involve multiple stages [5][6] [7] [8]:

Initial Reconnaissance :The stage includes gathering data about the target system and distinguishing potential weaknesses/vulnerabilities or exploiting zero-day vulnerabilities.

Initial Compromise: The initial compromise stage commonly includes spear phishing or other social engineering to get close enough to the target system.

Persistence: When inside the system , attacker lay out perseverance by creating the backdoor or introducing APT to keep up with access.

Lateral Movement: Lateral Movement is the stage where attacker move horizontally inside the network, investigating and compromising additional and extra systems.mostly attackers use dumping credentials to move laterly.

Data Exfiltration: Information exfiltration includes taking sensitive data from the compromised system and moving it to the command and control server.

Clean up: At last, the cleanup stage includes covering tracks and eliminating any proof of the attack to stay away from discovery.

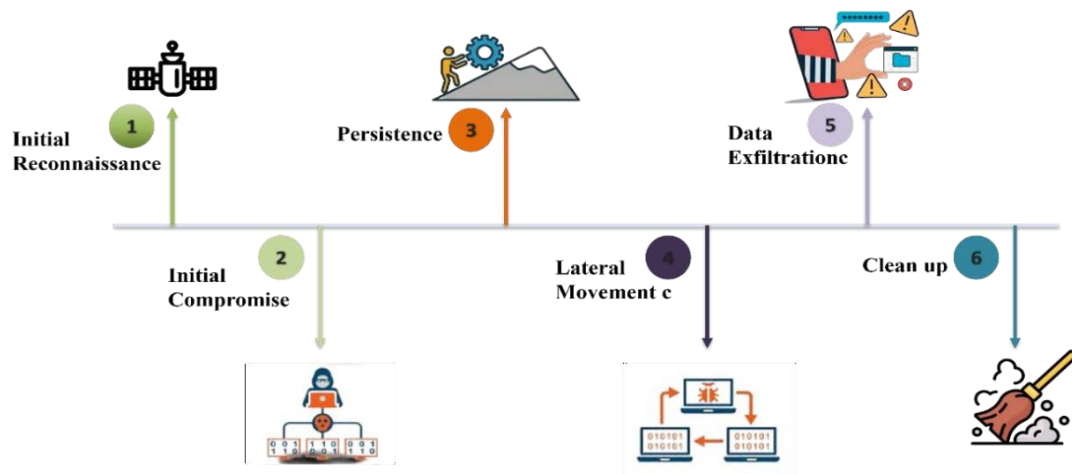


Figure 1 General Workflow of Advanced Persistent Threat

In Windows, Advanced Persistent Threats (APTs) detection is challenging for cybersecurity. Using Machine Learning (ML) for APT detection presents a strong and versatile way to deal with identifying sophisticated and stealthy threats. The integration of ML algorithms inside Windows security system improves the capacity to recognize abnormal patterns, ways of behaving, and activities that might connote the presence of APTs .

Tim Bai proposes a system which detect APT during lateral movement using windows event logs and RDP (Remote Desktop Protocol) session. It reveals that different feature sets can be extracted from Windows RDP event logs to detect lateral movement in cyber-attacks. Some of the possible feature sets include: The unified dataset and comprehensive dataset comprises of five different logs, to be explicit authentication, flow, process, DNS, and red group logs. The red group logs in the far-reaching dataset contains a subset of event from the authentication log, explicitly created from red team activities. These logs give data about account logons, logon failures, logoffs, and other relevant events [13].

Network based detection system is very much improved, however it has a few deficiencies. First and foremost, restricted data extricated from organization and it against the law against the law to investigate information over network. Furthermore, 72% of organization traffic is scrambled by protocol Transport layer security (TLS). Supervised ML with Light Boosting (LB) was utilized for classifying RDP session indicating high accuracy and recalls while doing comparative analysis of models like LR, DT, FNN, GNB, RF [14].

Lateral movement recognition using heterogenous graph embedding capture the structural characteristics of networks and event id, applying algorithms for graph representation for highlighting features use decision tree and algorithm of random forest for order from Mellon University, Los Alamos National Lab (LANL) and Software Engineering Institute Carnegie dataset with accuracy of 95%. It comprises of three modules: path representation, unsupervised path detection of anomaly-based attacks and heterogeneous graph development [15].

One of the most important techniques which is used by malware and advanced persistent threat is used of LOTL techniques (Living-off the land Technique) to maintain persistence. LOTL refers to legitimate binaries like PowerShell, Windows Management Instrumentation (WMI), and JavaScript-based tools or tools that are used by attackers to complete malicious activity without raising doubt as shown in Table 2. LOTL has a wide range of technique, not exact definition, evolving new techniques and detection gap as some antiviruses are failed to detect apt and malware using LOTL techniques to evade detection are the main challenges for detection of APTs [2]

Integration of deep learning and ML models including LSTM, Convolutional Neural Network i-e CNN, Random Forest (RF), and Multi-Layer Perceptron (MLP) to enhance the efficiency of model which is trained on the behavior profile which is built on event IDs of the kernel of the workstation set as label. This label can be malicious, normal, and unknown and analyze using Graph isomorphism network while using other models such as CNN-LSTM, GNN and BiLSTM-GCN for APT attack classification and detection. For classification purpose and balance data, the RF algorithm shows more efficiency than deep learning models. For detecting malware based on file behavior profiles, the GIN model is more useful than DGCNN and GCN models [16].

By using the dynamic system call and windows API function extracted by KNN, RF, XGB and DT shows the typical behavior of APT malware and system is designed to construct for domain knowledge of APT then by using classification contribution of APIs for sorting them, getting the of 99.28 and 98.85 percent of detection and classification of APT malware [17].

Detection of advanced persistent threats by analyzing the network traffic by designed own feature C2Loadfluct and Bad rate and generating the DNS traffic and HTTP/TCP/HTTPS features set. These two sets help for identification and detection of APT by analyzing the traffic by generating the connections with the remote C2 server and leaking data by using the improved PADASYN and to build a three-class classifier, it uses the AdaBoost algorithm [18].

Another study [19] shows detection model for Command-and-control server stage of APT attacks using anonymous dataset generated over two months. Feature extraction is done by isolation forest with other algorithms including LOF and KNN but there is no proper

accuracy is stated. Another research done on detection of LOTL techniques is the use of LOLAL framework. As explained earlier, different legitimate binaries as shown in Table 1 are used by using command line text. So first tokenization is done to break full token into individual, then labeling is done after word embedding that is used to convert words into vector which is then selected by human analyst. A different classifier is used to identify between malicious and benign samples and reach up to accuracy of 96%. But effectiveness may vary depending on the specific attack classes [20].

Table 1: Some examples of how LOTL are used by adversaries[2].

LOTLBIN	Description	Malicious Command Line
Certutil	Decoding a Base64-encoded file into malicious executable file	certutil -decode b64file newFile.exe
Bitsadmin	Download malicious files at temporary location, submit jobs to execute the malicious payload	bitsadmin /create 1 bitsadmin /addfile 1 https://foo.com/a.exe c:\a.exe bitsadmin /resume 1 bitsadmin /complete 1
Msbuidl	Execute and build C# project stored in the target file	msbuild.exe pshell.xml
Regsvr32	Execute the specified remote or local .SCT script	regsvr32.exe /s /u /i:file.sct scrobj.dll
Msiexec	Install and execute malicious code from remote servers	msiexec /q /i http://192.168.83.2/cmd.jpeg

A comprehensive comparison between windows security system reveal that Seventy percent of attacks on Windows endpoints failed, ten percent were reported as warnings or suspicious behavior that was seen but not identified by EDR, and twenty percent were recognized and reported as successful under different attack scenarios like PE and DLL injection ,SC Shell and procdump by exploiting it vulnerabilities like process injection and Dll,permission and lsass [4].

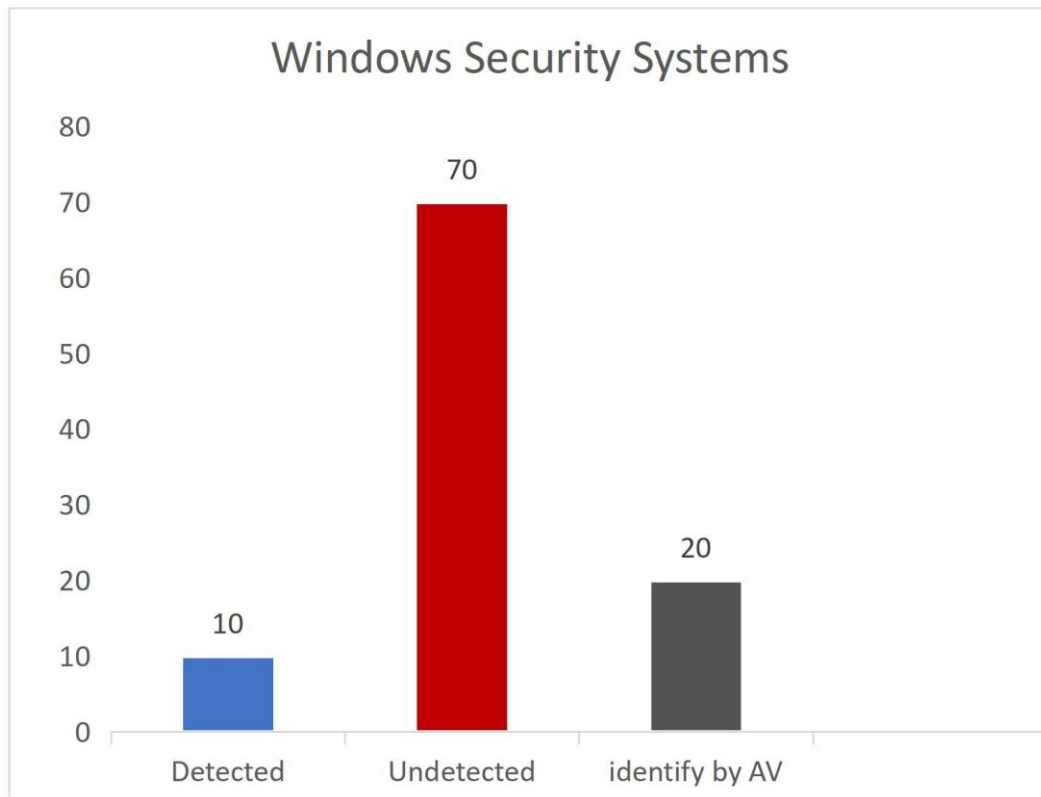


Figure 2: Analysis of Windows Security System

Study [11][12] shows different methods of detection malware in general and particularly APT. Accordingly the main three approaches for APT malware: signature-based method, anomaly-based method and hybrid. A comparative study shows the limitation of each of research work done previously. Also compare the accuracy on dataset used by researchers .

Table 2: Comparative Analysis of Previous work

Paper -years	Method	Dataset	Feature	Accuracy
Systematic Analysis of Windows Malware LOTL 2021[2]	Analysis of technique used by malware and APTS	Legitimate tools and binaries	Power shell, java script, WMI	26.4 % of this technique used by APTS

RDP based lateral movement detection-2021 [13]	Supervised ML with LB Model	Comprehensive and unified dataset	Flow, Red team logs Authentication, Process, DNS	99.99%
LM Tracker: LM path detection based on heterogeneous. graph embedding-2022[15]	Heterogeneous graph using Decision tree and random forest algorithm	Software Engineering Institute Carnegie Mellon University and Los Alamos National Lab(LANL)	Event logs and traffic	95%
A novel app for detection using deep graph for EDS[16]	Deep learning using Graph Isomorphism Network (GIN) with some ML Model	Logs,Network traffic	Event Ids, process Ids	86.77 with GIN
APT Mal Insight: Based on System calls and ontology knowledge system, identification and recognition APT malware 2020 [17]	APT Mal Insight framework	Certification Center, China Information Security Evaluation, and Virus Share	Dynamic system call and Api sequence	98.85%
Two statistical traffic features for identification of APT group 2022 [18].	Analysis of DNS behaviour and traffic feature using IF,KNN ,LOF	Pcap traffic packet generated by benign software's and APTS samples	TCP/HTTP/HTTP	DNS dataset: 0.8426 and TCP datasets: 0.8645.
APT Detection using network Traffic Log Features [19]	KNN,LOF and isolation Forest	Anonymized datasets of network Traffic over 2 months	C2 domain name access records	Not specific accuracy
Detection Using Active Learning of LOTL Command [20]	Ensemble boosting classifiers	Binaries used for LOTL	Command Line text	96%

2.3 Types Of Advanced Persistent Threat

APTs vary in their tactics, techniques, and procedures (TTPs), and they may be associated with nation-states, cybercriminal organizations, or hacktivist groups. Here are some common types of APTs.

APT28: The Russian APT group, center around gathering knowledge that would be generally valuable to the Russian government by focusing on public foundation in the USA, Europe and different nations. It steals information after exploitation by sending spear-phishing emails, compromised the website and situate malware to infect website visitors and gained access to organizations by undermining their web-facing servers [9] [10].

APT29: It is known for its profound interest in client malware. This malware incorporates special binaries which are used by tools like PowerShell. APT29 is said to direct its functional tempo to many factors. This approach reflects an exceptionally elevated degree of complexity and makes APT29 more dangerous and powerful in its efforts.

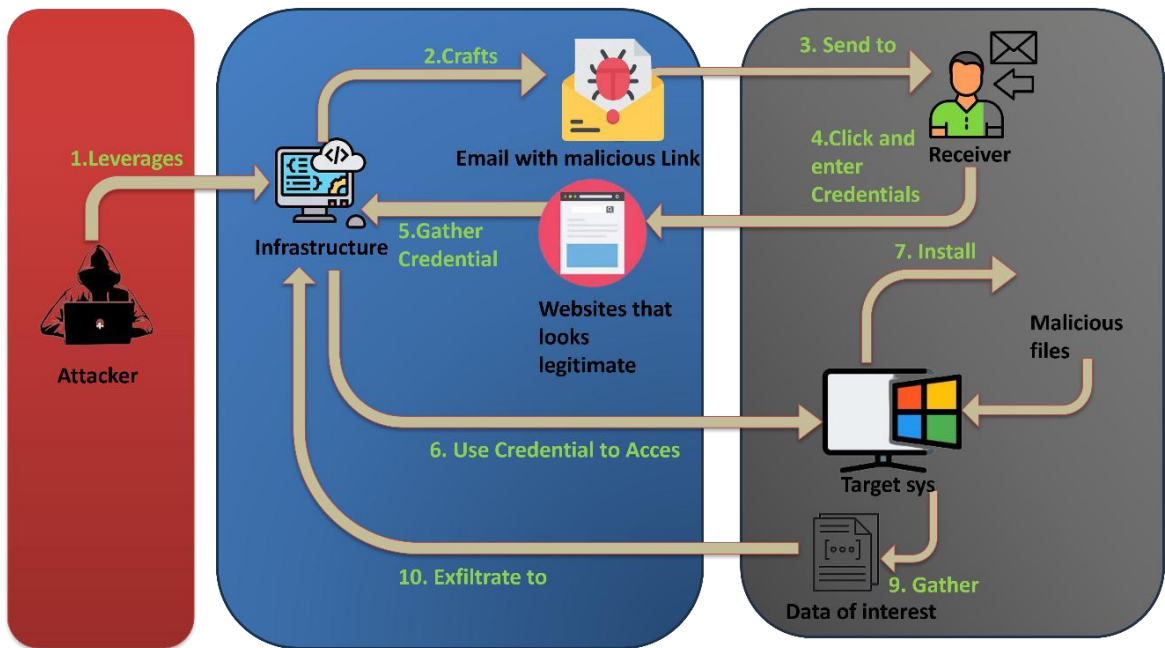


Figure 3: Workflow of APT28/29

APT41: A China based group utilizes an extensive collection of 40+ malware instruments and groups of devices to help their procedures. Spear phishing is frequently utilized with attachments, for example, .chm documents to trick and used for victimization. One single mission by APT41 can use many different malware tools, including rootkits, keyloggers, backdoor, and significantly more. For long term persistence, APT41 has been recognized utilizing rootkits and master boot record boot kit for packing of malware. Boot kits are especially stealthy because the code is executed before the operating system initialization.

APT40: Chinese Advanced Persistent Threat (APT) group APT40 has utilized various strategies and procedures and an library of custom and open-source malware quite a bit of which is shared among different apt group to get access through client and administrator credentials, enable lateral movement e inside the organization, and find high worth resources to exfiltrate information.

APT35: Iranian government-supported cyber reconnaissance group has generally depended on sophisticated tools, including openly accessible web shells and penetration testing devices, recommending a moderately beginning improvement capacity. In any case, the expansiveness and extent of APT35's activities APT35 regularly depends on spear phishing to compromise an organization initially, frequently utilizing draws connected with health/medical care, work postings, resumes, or secret phrase approaches.

APT32: APT32 uses a custom set-up of malware instruments, increased by financially accessible devices, frequently commonly found while "living off the land" inside target networks. They target unfamiliar companies are manufacturing, hospitality (hotels), and

consumer products... These are exceptionally huge business areas inside Vietnam. APT32 danger entertainers have additionally been artfully focusing on network security and technology corporations.

APT33/34: APT33 is an Iranian Threat Group sent off the Shamoon wiper malware on both the Center East and Europe. Shamoon is an exceptionally pernicious and damaging malware planned by APT33 to obliterate all information on infected system.

Chapter 3

Proposed Methodology

3.1 Introduction:

Advanced Persistent Threats (APTs) pose a major challenge to cyber security as they occur stealthily and continuously, often targeting specific facilities to steal information or disrupt operations over an extended period of time. Traditional security measures are usually inadequate to detect and defend against such sophisticated threats. Therefore, this chapter proposes a comprehensive methodology that utilises machine learning (ML) techniques in conjunction with dynamic malware analysis tools such as Cuckoo Sandbox to improve detection capabilities against APTs in Windows environments.

3.2 Overview

To detect Advanced Persistent Threats (APTs) in windows, the proposed methodology is to do dynamic malware analysis through the Cuckoo Sandbox with refined machine learning techniques. The model captures the detailed logs that are necessary for the training of machine learning algorithms by examining the behavior of potential malwares in a secure isolated system. Then, these algorithms are fine-tuned to detect the anomalies and subtle patterns that are the features of APTs. The proposed methodology steps start from the comprehensively collection of data from multiple resources then vigorous pre-processing and feature extraction processes are used to ensure the optimality of data for the training of machine learning algorithms. Next, model is rigorously validated to make sure the reliability and accuracy earlier it is deployed model in operational environment. The combination of Cuckoo Sandbox's dynamic

malware analysis with advanced machine learning techniques doesn't just enhance detection capabilities against sophisticated cyber threats; it also ensures ongoing updates and flexibility to combat new and emerging dangers contributes to strengthens our cybersecurity defenses.

3.3 Data Collection

Data is the fundamental part of any system that is based on ML. For the purpose of APT detection, the relevant data encompasses executable files, scripts, and binaries which may potentially harbor malicious content. These are sourced from:

Publicly available malware databases: These repositories provide a wealth of verified malicious files which can be used to train the detection models.

Honeypots: Deployed within network environments, honeypots attract attackers and capture the malware used.

Industry sharing platforms: Organizations often share indicators of compromise and other relevant data through platforms like VirusTotal or via industry partnerships. Each collected sample is labeled as either benign or malicious based on the consensus from various antivirus engines and other verification mechanisms. This dataset not only serves as the training set but also helps in continuously refining the detection algorithms.

3.4 Dynamic Malware Analysis Using Cuckoo Sandbox

Dynamic malware analysis is pivotal in understanding the behavior of malware by executing it in a controlled, isolated environment to observe its actions without affecting the host system. For that Cuckoo Sandbox is being used because it gives comprehensive logging features and customizability. The features it provides are :

Generation of Behavioral Logs: The different behaviors of malware in execution and log actions like changes in processes, file system , registry and in network communication are monitored by Cuckoo.

Automated Report Generation: The detailed reports can be generated by cuckoo after complete analysis. These reports summarize the behavior of the executables and give information whether it is potential malicious or not.

The gathered data that is generated from Cuckoo Sandbox improves the initial data set by the behavioral patterns and is crucial for the subsequent steps.

3.5 Preprocessing

Preprocessing is used to transform the raw data into refined data set ready for ML processing. It includes data cleaning, normalization, encoding and features selection. In data cleaning, incomplete data values and corrupted files are removed. Normalization do the scaling of the features to a uniform range and make sure the reasonable weighting of ML algorithms. Encoding is used to convert the categorical data into numerical format through ML models. To reduce the model training time and to improve the performance of model, features selection reduce the dimensionality of the data by choosing the most relevant features.

3.6 Feature Extraction:

Feature extraction looks up the behavior of malware and derive attributes that are the indications of malware's malicious intent. These features can have:

File Operations: The operations like creation, deletion or modification of files might indicate that there could be an effort performed by malware to tamper the important files or install malware components.

Network Activity: Through network activity one can know the unusual network requests or attempts at data exfiltration are being made.

Registry Activity: One of the important techniques used by APTs to maintain a persistent is to change the registry.

To differentiate between benign and malicious software, these features are very helpful as they especially designed to capture the behavior of malware.

3.7 Model Training

With the help of prepared data set the machine learning model is trained and include following steps:

Algorithm selection: A suitable ML algorithm is selected depending on the features and characteristics of the dataset. These algorithms are Random Forest, Convolution Neural Network, Long Short-Term Memory naive Bayes or Long Short-Term Memory.

Cross-validation: Cross validate the model to avoid its overfitting and generalize it well to new unseen data.

Parameter tuning: The accuracy and efficiency of the ML model improves by optimizing the parameter of the model.

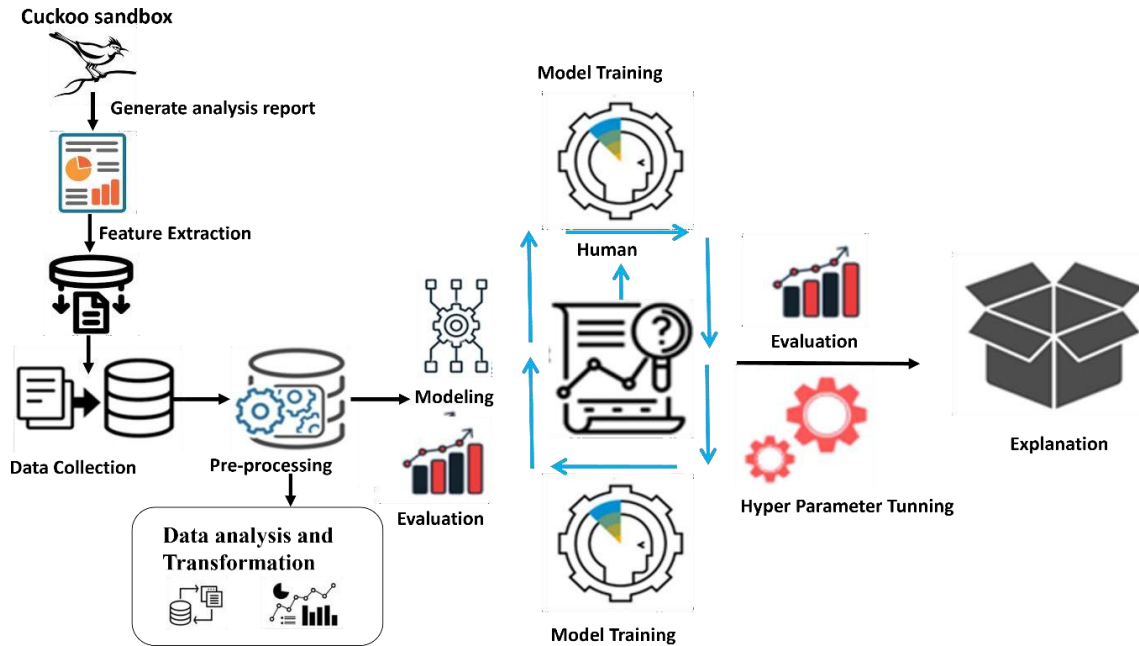


Figure 4: Proposed Methodology

3.8 Validation:

After the model has been trained, it is validated using a separate data set that was not used during the training phase. This helps to evaluate the effectiveness of the model. The most important metrics for validation are:

a) **Accuracy:** The overall correctness of the model in classifying malware.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

In which: TP - True positive: The number of malicious samples classified correctly; False negative: The number of malicious samples classified as normal; TN - True negative: The number of normal samples classified correctly; FP - False positive: The number of normal samples classified

as malicious.

b)**Precision and recall** : Precision measures the accuracy of positive predictions, and recall indicates the ability to find all relevant instances of malware.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

c)**F1 Score**: The harmonic mean of precision and recall, which provides a single metric for evaluating model quality.

$$\text{F1} = \frac{2 \text{ precision recall}}{\text{precision} + \text{recall}}$$

3.9 Justification of Proposed Methodology:

The proposed methodology for detecting Advanced Persistent Threats (APTs) in Windows operating environments provides a robust framework that leverages the strengths of machine learning (ML) and dynamic malware detection

Machine Learning and dynamic malware analysis supported by tools such as Cuckoo Sandbox. This methodology has been carefully developed to address the sophisticated nature of APTs, which are characterized by their stealth, persistence and the significant threats they pose to national security and corporate information systems. The justification for this comprehensive approach is rooted in several key areas: the complexity and stealth of APTs, adaptability to evolving threats, comprehensive data analysis, enhanced detection capabilities, scalability, proactive security posture and cost-effectiveness.

Advanced Persistent Threats are particularly difficult to detect due to their stealthy and slow tactics. Unlike other cyber threats that seek immediate success through quick attacks, APTs establish a long-term presence in host networks to systematically steal valuable data or carry out insidious attacks damage over time. This stealthy nature often allows APTs to evade conventional cybersecurity measures, which are typically designed to detect more overt and immediate threats. Therefore, a more sophisticated approach is required, one that can monitor, analyze, and predict activities based on deep behavioral insights. This is where dynamic malware analysis tools such as Cuckoo Sandbox become essential. By analyzing malware in a controlled environment, Cuckoo Sandbox captures detailed logs of behavior that static analysis would miss, providing rich data that feeds into the subsequent ML processes. Machine learning stands out as an ideal technology to address the complexity of APT detection due to its ability to learn and improve over time. By training ML models on up-to-date, diverse datasets that include the latest manifestations of malware, the proposed system remains adaptable to the continuously evolving threat landscape. Regular updates to the training datasets and retraining of the models ensure that the system not only keeps up with the latest threats but also stays ahead of them, maintaining its relevance and effectiveness.

The integration of machine learning with dynamic malware analysis enables a comprehensive investigation of each malware sample. ML algorithms can process the extensive behavioral data recorded by Cuckoo Sandbox to detect patterns and anomalies that indicate malicious intent. This capability is critical considering the volume and complexity of data required to detect APTs.

By using advanced algorithms, the methodology can efficiently sift through this data and identify threats with greater accuracy and speed than human analysts could. In addition, the proposed method improves the overall detection capabilities of cybersecurity systems. The dual approach of using both dynamic analysis for data collection and machine learning for data processing creates a synergy that significantly increases the sensitivity and specificity of malware detection. Dynamic analysis provides a wealth of data about malware behavior, while machine learning uses this data to learn and make accurate predictions about new and unseen patterns, increasing the detection rate of new or mutated APTs.

Scalability is another crucial aspect of the proposed methodology. As organizations grow and the number of endpoints increases, the amount of data to be processed can become overwhelming. The use of machine learning algorithms allows the system to handle this increase efficiently. In addition, tools like Cuckoo Sandbox automate the analysis of each sample, which scales better than manual analysis and requires less investment in additional resources as system requirements increase.

In the modern cyber threat landscape, where the cost of a security breach can be catastrophic, a proactive security posture is essential. The proposed methodology enables organizations to detect and respond to threats before they escalate into breaches. This proactive approach not only secures data, but also preserves the integrity of company operations against the insidious nature of APTs.

Finally, despite the initial set-up and training costs, the cost-effectiveness of this method becomes clear in the long term. By automating the detection and analysis processes, the need for extensive manual work is significantly reduced, which lowers

operating costs. In addition, the early detection and containment of threats minimizes the economic impact that APTs could have on an organization, making the investment in such a sophisticated detection system economically justifiable.

To summarize, the proposed method of detecting APTs using machine learning and Cuckoo Sandbox is a sophisticated, effective and sustainable solution to a complex problem. It addresses the unique challenges posed by APTs with a dynamic, adaptive and proactive approach that ensures organizations can protect their critical assets from some of the most dangerous cyber threats in the modern world. This blend of advanced technological tools and strategic foresight makes the methodology a vital component in the arsenal of any organization serious about securing its digital borders.

3.10 Summary:

This chapter describes a method that allows you to leverage the power of machine learning and dynamic malware analysis to effectively detect and analyze Advanced Persistent Threats in Windows environments Windows environments. By implementing this approach, organizations can improve their defenses against sophisticated cyber threats that may be overlooked by traditional security tools. The strategy outlined emphasizes a systematic flow from data collection to deployment to ensure a thorough and effective implementation.

Chapter 4

Implementation and Experimental Results

4.1 Introduction:

This chapter dig into the practical execution of the proposed methodology for distinguishing Advanced Persistent Threat (APTs) in Windows environment through a integrated methodology consolidating AI (ML) and dynamic malware analysis utilizing Cuckoo Sandbox. The main point of this implement is to overcome the gap between theoretical models and real world application, guaranteeing that the system isn't just effective in a controlled setting yet additionally vigorous and versatile under functional circumstances.

The execution cycle is organized to methodically execute the sequence of steps illustrated in the procedure, from setting up the underlying environment to deploying the complete Detection System. Each stage is joined by detailed description of tools, technique and technologies used. This part likewise addresses difficulties experienced during the implementation phase and the solutions devised to overcome them, giving an comprehensive insight into the practical aspects of deploying a sophisticated cyber security defense mechanism.

4.2 Data Collection:

The foundation of any ML based system is Dataset. To APT detection, the relevant data encompasses executable files, scripts, and binaries which may potentially harbor malicious content. So Data collection includes many steps :

4.2.1 Creation of Dataset:

In Machine learning based Advanced Persistent Threat Detection, creation of dataset is very challenging because in case of advanced persistent, analyzing the dynamic behavior of APTs is mandatory due to its advanced tactics, techniques, persistence and stealthy nature .So these files are be collected from various sources, including Virus total, Malware bazar any any.run . Considering the Specific types of APTs (as mentioned) that is well know for LOTL techniques can provide an enhanced and efficient way of improving the detection the Advanced Persistent Threat.

So for Creation of Dataset, it is mandatory to dynamically analyze the malware in isolated environment to see the actual working of malware.

4.3 Cuckoo:

For this, I used Cuckoo Sandbox.it is an open-source, automated malware analysis system intended to analyze and identify malicious software. It gives a dynamic analyses environment where malware can be executed in a controlled environment, allowing experts to notice their way of behaving and comprehend their capacities without risking the integrity of system.

4.3.1 Modular Architecture of Cuckoo:

A cuckoo's modular architecture considers adaptability and extensibility. It comprises of various parts, including the Cuckoo core, analysis modules, and helper modules. Analysts can customize and expand Cuckoo's functionality by adding or changing modules to suit their specific requirements. Cuckoo supports the analysis of malware samples focusing on target different operation system, including Windows, Linux, and macOS. This stage

skeptical approach empowers analysts to analyze a wide range of malware types and better understand their cross-stage capabilities.

4.3.2 Basic Static tools Used by Cuckoo:

4.3.2.1 File Analysis Tools:

File Utility: Determines document type and extract basic information .

PEiD/PEinfo: Distinguishes packers and compilers used in Windows executables.

ExifTool: Extract metadata from different file types, like pictures.

4.3.2.2 Malware identification tools:

YARA: Permits the creation of rules for pattern matching in documents to distinguish known malware.

ClamAV: Performs antivirus scan on files to identify known malware signatures

VirusTotal Programming interface: Incorporates with the VirusTotal services to inquire for file reputation and additional threat intelligence information.

4.3.2.3 Hashing Devices:

MD5, SHA1, SHA256: find hashes of documents for integrity verification and identification of different file types.

4.3.2.4 Code Analysis tools:

PEview: Display data about the PE (portable Executable) file format.

Dependency Walker: List DLL dependencies of Windows executables.

Strings: Extract ASCII and Unicode strings from binary Executables.

4.3.3 Basic Dynamic tools Used by Cuckoo:

4.3.3.1 Monitoring and instrumentation tool:

API Monitor: Monitor Programming interface calls made by the malware during execution.

Process Monitor (Procmon): Catches system events like file system and process activity.

Regshot: Takes before and after snapshots of registry to detect changes made by the malware.

4.3.3.2 Network analysis tool:

Wireshark: Capture and analyze network traffic generated by the malware test to understand its network communication.

Tcpdump: Catches network packet for analysis

Bro Network Security Monitor: Gives network traffic Analysis and Protocol Detection.

4.3.3.3 Behavioral Analysis Tool:

Cuckoo's Built-in Behavioral Analysis: Cuckoo itself monitor different system activities, such as, file system changes, registry Modification, network communication, and process behaviour.

Sysinternals Suite: Incorporates different tools like Process Explorer, Process Monitor and Autoruns for detail system analysis investigation.

4.3.3.4 Memory Analysis Tools:

Volatility: Analyze memory dumps to extract data about running processes, network connection, Loaded DLLs, and other system artifacts.

WinDbg: Debugger for analyzing crash dumps and investigating Windows applications.

4.3.4 Reporting and visualization:

After analyzing a sample, Cuckoo produces detail analysis reports using HTML, PDF Report, D3.js and GraphViz containing data about the sample way of behaving, indicator of compromise (IOCs), and potential security risk. Moreover, Cuckoo gives perception instruments to graphically addressing the connections between different system entities, like process, file and network communication.

4.3.5 Virtualization Technology:

Cuckoo depends on virtualization technology, like VirtualBox, VMware, or KVM/QEMU, to establish isolated environment for executing malware samples. These virtualized environment guarantee that the analysis doesn't influence the host system and consider simple snapshotting and rollback of system changes.

4.3.6 External Communities:

Cuckoo can share analysis report and indicator of compromise (IOCs) with other security tools and organization utilizing the Malware Data Sharing Platform (MISP). While Elasticsearch and Kibana utilized to store and visualize a lot of analysis data. They help in looking, separating, and correlating analysis over time

These tools, combined with Cuckoo's modular architecture and community-driven development, enable analysts to conduct in-depth analysis of malware samples and generate actionable insights to enhance cybersecurity defenses.

These tools, joined with Cuckoo's modular architecture and community-driven, enable analyst to conduct to in depth analysis of malware and create significant insight for betterment of cyber security defense.

4.4 Installation of Cuckoo:

The Cuckoo host components is completely written in Python, therefore it is required to have an appropriate version of Python installed. Cuckoo is only fully supported by Python 2.7. Older version of Python and Python 3 versions are not supported. So for this reason I used Ubuntu 18.04.4 Lts desktop as my second operating system in my Laptop.

Several steps have taken during the installation of Cuckoo:

- Update and upgrade existing system Packages.
- Install necessary dependencies like Python, libffi, and libssl.
- Install VirtualBox and MongoDB.
- Set up PostgreSQL and configure the database.
- establish network Packets catching using tcpdump.

```
sudo chgrp pcap /usr/sbin/tcpdump
```

```
sudo setcap cap_net_raw,cap_net_admin=eip /usr/sbin/tcpdump
```

- Set user authorizations and improve security effort.
- Set up and tailor Cuckoo inside a Python virtual climate.

```
sudo usermod -a -G vboxusers cuckoo
```

- Run a script for virtual environment
- Establish another virtual machine explicitly for Cuckoo and add required dependencies
- Mount a Windows ISO to set up the virtual machine.
- Use VMCloak to install and change settings on virtual machines.
[vmcloak-vboxnet0](#)
- Copy and prepare virtual machines with required applications, like IE11.
- Make snapshots of virtual machines for integration with Cuckoo.
- Start initializing Cuckoo setups and incorporate community script.
- Change VirtualBox settings to configure VMs functional mode.
- Organize network setting and establishing port forwarding.
- Actuate Cuckoo services across different terminal tabs:
- Start the cuckoo router
- Execute the cuckoo main process
- Begin the cuckoo web interface

Overall Cuckoo setup Look like this:

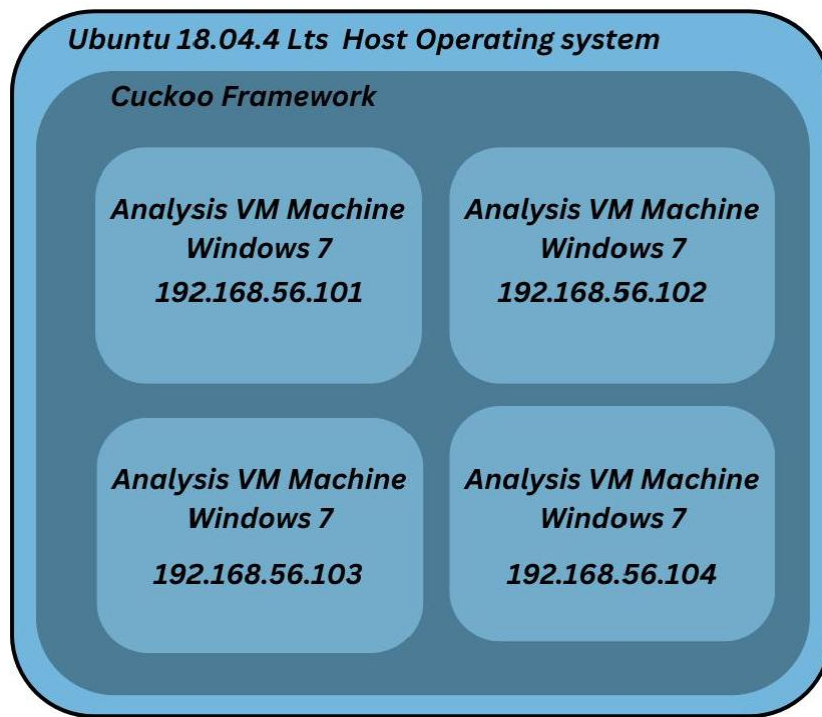


Figure 5: Architecture Of Cuckoo

Installed Cuckoo Version : 2.0.7

So web interface look like this :

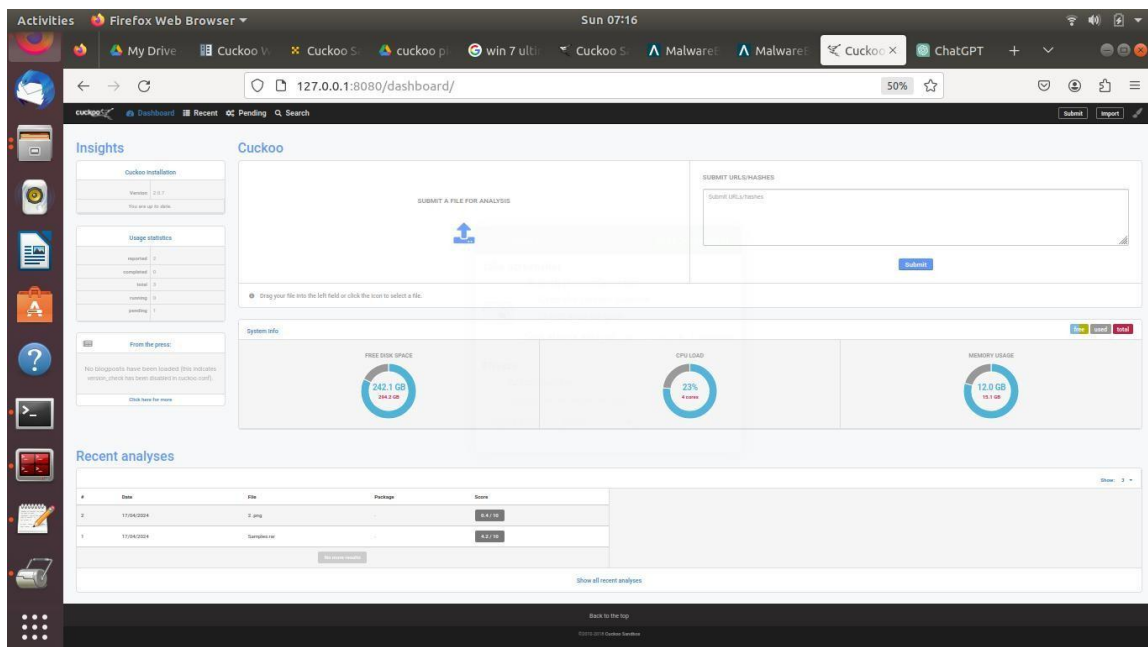


Figure 6: Web Interface of cuckoo

After Uploading a file,cuckoo start doing analysis :

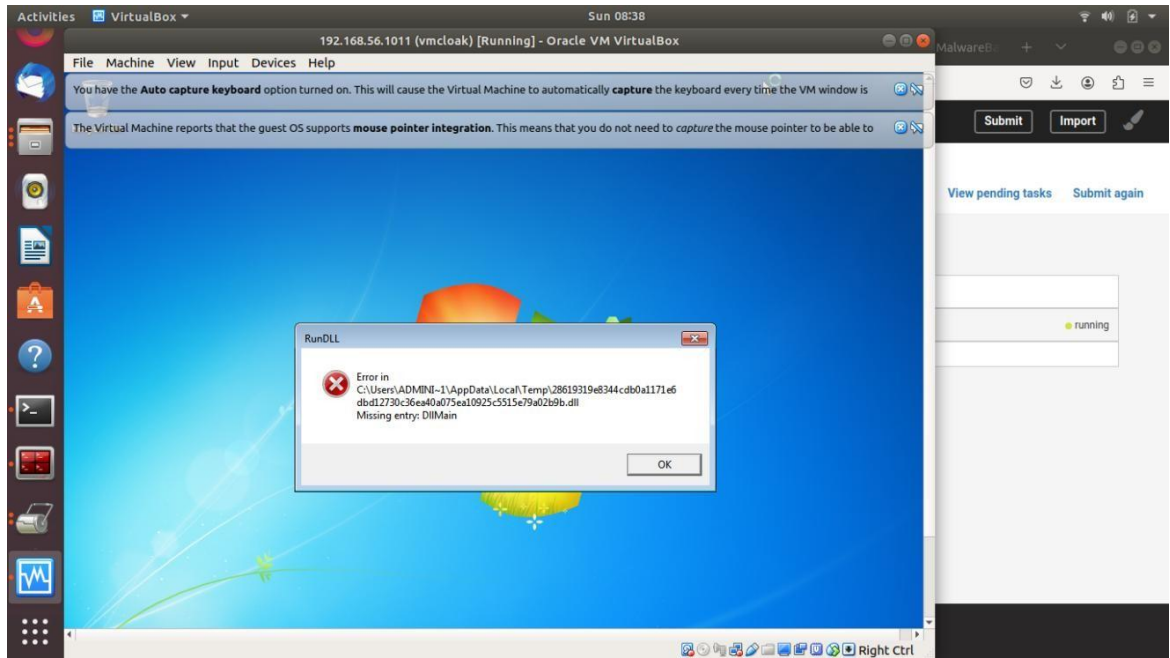


Figure 7: Cuckoo while Running

So after analysis it generate a report in `./cuckoo/storage/analysis/List` of folder containing separate analysis of malware.

Total Sample :90

Table 3: Total Samples

File Format	Total	Malicious	Benign
EXE	75	45	30
DLL	15	9	6

4.5 Feature Extraction:

So after analyzing the malware using cuckoo, report is generated. Next step is to extract the features from report.

Feature extraction is a basic step toward machine learning based detection system , as it includes changing raw information into an suitable format for analysis and model training. With regards to malware detection , feature extraction from different sources like registry activity, process activity, file activity, and network activity give significant insight into the behaviour of potentially malware samples. Using **Google Collab** extract relevant feature using string matching (python) from activities done by malware.

Steps taken during feature extraction

4.5.1 Mounting Google Drive: The code mounts Google Drive to get to documents put away in the specific directory.

4.5.2 Importing necessary Libraries: The necessary libraries, for example, Document from docx, re, csv, and os are imported.

4.5.3 Defining function to read DOCX Document: The read_docx() function read the items in a DOCX file and returns the text.

4.5.4 Defining a Function to Count Features using Regular Expressions: The count_features() function counts the events of predefined features involving regular expression in the text extracted from DOCX documents.

4.5.5 Characterizing a function to Compose Results to a CSV Document: The write_to_csv() function composes the feature counts for each feature to a CSV file.

4.5.6 Getting a list of DOCX Records from the directory : The script list all DOCX documents in the predefined directory.

4.5.7 Processing Each file and Gathering feature Counts: The script repeats through each DOCX file, extract its text, counts the feature of all files and stores the count in dictionary

4.5.8 Composing the results to a CSV Record: At last, the feature counts for all reports are written in CSV document.

Table 4:List of features extracted from Registry Features

Name	Data type	Description
HKEY_LOCAL_MACHINE	Number	No of HKEY_LOCAL_MACHINE
HKEY_CLASSES_ROOT	Number	No of HKEY_CLASSES_ROOT
HKEY_USERS	Number	No of HKEY_USERS
HKEY_CURRENT_CONFIG	Number	No of HKEY_CURRENT_CONFIG
startup_registry	Binary	Use or Not?
service_registry_key	Binary	Use or Not?
dll_injection_registry_key	Binary	Use or Not?
shell_spawn_registry_keys	Binary	Use or Not?
bho_registry_key	Binary	Use or Not?

Table 5:List of features extract from file activity

Name	Data Type	Description
Cache.dat Present	Binary	Use or not?
EXE Files Deleted	Number	No of EXE Files Deleted
Files Deleted	Number	No of Files Deleted
Files Dropped	Number	No of Files Deleted

Files Opened	Number	No of Files Opened
Files Written	Number	No of Files Written
Text Files Deleted	Number	No of Text Files Deleted
XML Files Deleted	Number	No of XML Files Deleted
Startup Folder Present	Number	No of Startup Folder Present
WER/Temp Files Deleted	Number	No of WER/Temp Files Deleted

Table 6:Lists of feature to check if there is activity of task schedule

Name	Datatype	Description
Task Scheduler Library	Binary	Use or not?
Task Scheduler Configuration Files	Binary	Use or not?
Task Scheduler Database	Binary	Use or not?
Task Scheduler Logs	Binary	Use or not?
Scheduled Task Executables	Binary	Use or not?
Task Scheduler Service Files	Binary	Use or not?

Table 7:List of other Extracted features

Name	Data type	Description
File Extensions Used by Malware	String	List of file ext used
Network Activity Detected	Binary	Detected or not?
Startup Folder Present	Binary	Present or not?

Administrative Rights Required	Binary	Admin right required or not?
user_shell_folders_key	Binary	Present or not?

4.6 Preprocessing:

Preprocessing includes changing and cleaning the extracted feature to improve the quality of dataset and enhance the performance of machine learning algorithms. It includes different methods, for example, data normalization, feature scaling, handling missing values, and encoding categorical variables. The objective of preprocessing is to make the data appropriate for input into ML models and mitigate issues like information skewness, noise, sparsity and overfitting.

This stage involves:

4.6.1 Data Cleaning:

Data cleaning is the most common way of identifying and correcting errors, irregularities, and missing values in the dataset. It includes errands, for example,

4.6.2 Eliminating Missing Data Values:

Dataset contain some missing values Visualize using heat map

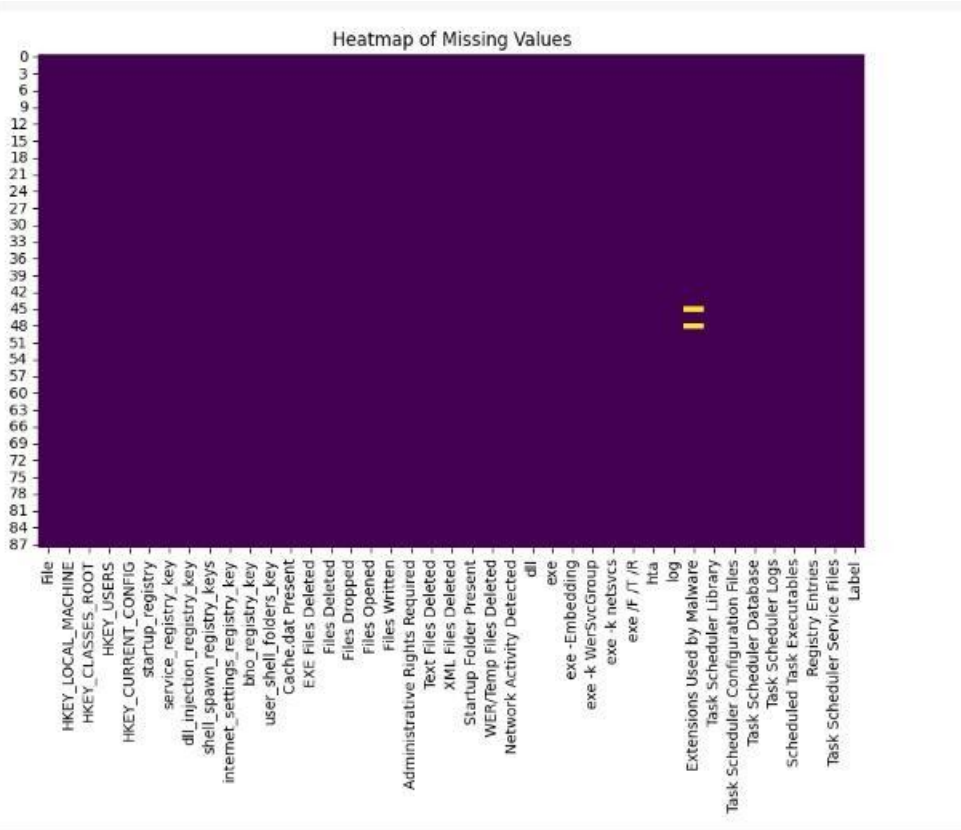


Figure 8: Heatmap of missing values before Handling

After handling missing Values, heatmap show like this

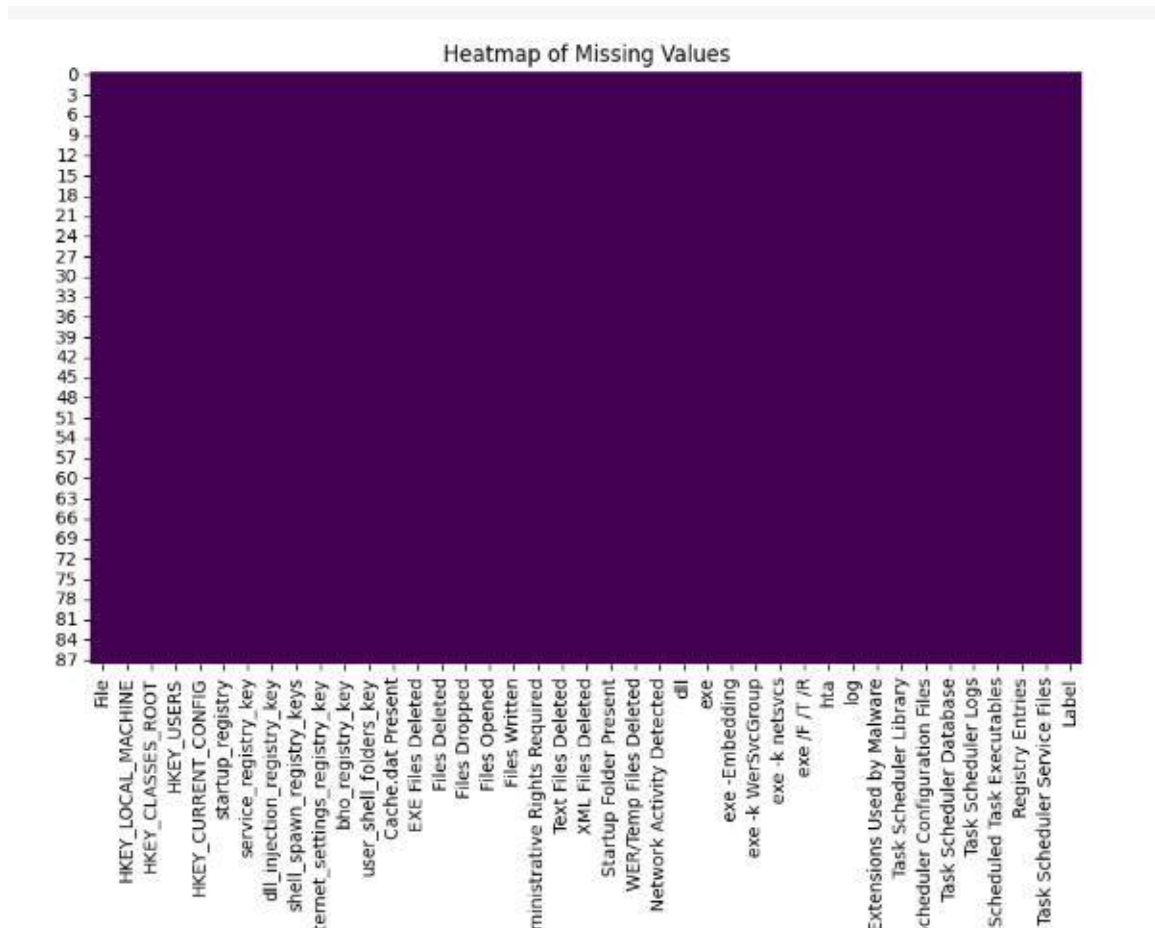


Figure 9: Heatmap of missing values after handling

4.6.3 Remove duplicate rows:

To Check duplicate rows , following function is used

```
duplicate_rows = df[df.groupby(df.columns.tolist()).transform('size') > 1]
```

```
Duplicate Rows:
Empty DataFrame
Columns: [File, HKEY_LOCAL_MACHINE, HKEY_
Index: []

[0 rows x 41 columns]
No duplicate rows found.
```

Figure 10: Checking Duplicate Rows

4.6.4 Check Data type of each column:

Data type of column consist of Object,int and binary.

4.6.5 Encoding :

Encoding includes changing categorical data into numerical form for processing by ML models. Normal encoding methods include:

4.6.5.1 Frequency Encoding:

- Assign every classification a numerical value relating to its frequency in the dataset.
- Helps ML models learn from event patterns of various categories.

4.6.5.2 Label Encoding:

- Convert categorical data into ordinal value by assigning integer.
- Appropriate for factors with an intrinsic request or progressive system.
- Should be utilized cautiously to avoid unintended ordinal connections between classifications.

4.6.5.3 One-Hot Encoding:

- Makes binary vectors for every class, addressing its presence or nonappearance.
- Ideal for nominal variable without inherent order.
- Keeps the model from accepting ordinal connections however may increment dimensionality and present multicollinearity.

4.6.6 Normalizing:

In the preprocessing phase of my proposal, **standard scaling** arose as an essential method to guarantee the uniformity and similarity of numeric features. By using standard scaling, changed the distribution of numerical data, bringing each feature to a mean of 0 and a standard deviation of 1. This standardization process was critical for mitigating biases in the Machine Learn models and working with fair weightage among features during training. Implementing standard scaling inside the preprocessing pipeline considered better model execution and faster convergences during training cycles.

4.6.7 Model Selection:

In the model choice period of thesis, I painstakingly considered a different range of ML algorithms to create a powerful detection system for malware and other security threats. The chosen algorithm encompasses various methodologies and approaches to address various parts of the detection task.

4.6.7.1 Random Forest (RF):

- RF was chosen for its effectiveness and capacity to deal with high-dimensional data.
- Ensembles of decision tree catches complex connections between features.
- Gives versatility to noise and anomalies while keeping up with high predictive accuracy.

Different estimator (decision tree) is used to evaluate the model and tried on three dataset obtained from techniques used for encoding as mentioned earlier. Random shuffling of data before splitting is set to 42.

Key metrics for validation include:

Table 8: Accuracy of RF On Label with diff no of tree

n_estimators	Accuracy	Precision	Recall	F1-Score
10	0.863636	0.857323	0.863636	0.858009
50	0.909091	0.91866	0.909091	0.900253
100	0.909091	0.91866	0.909091	0.900253
150	0.909091	0.91866	0.909091	0.900253
200	0.909091	0.91866	0.909091	0.900253

Table 8 shows the result accuracy of classifying process with different numbers of trees. With 50 and 100, RF model has high accuracy with no change. However, with number of trees 10 these is discrepancy in accuracy, precision recall and F1-score. Although change of number trees changes from 50 to 200, there is no change in accuracy. This analysis demonstrates that the Random Forest algorithm offers robust and consistent classification performance across different numbers of trees. However, in practical applications, achieving the optimal balance between model performance and computational efficiency requires careful consideration and selection of appropriate parameters to attain the desired outcomes.

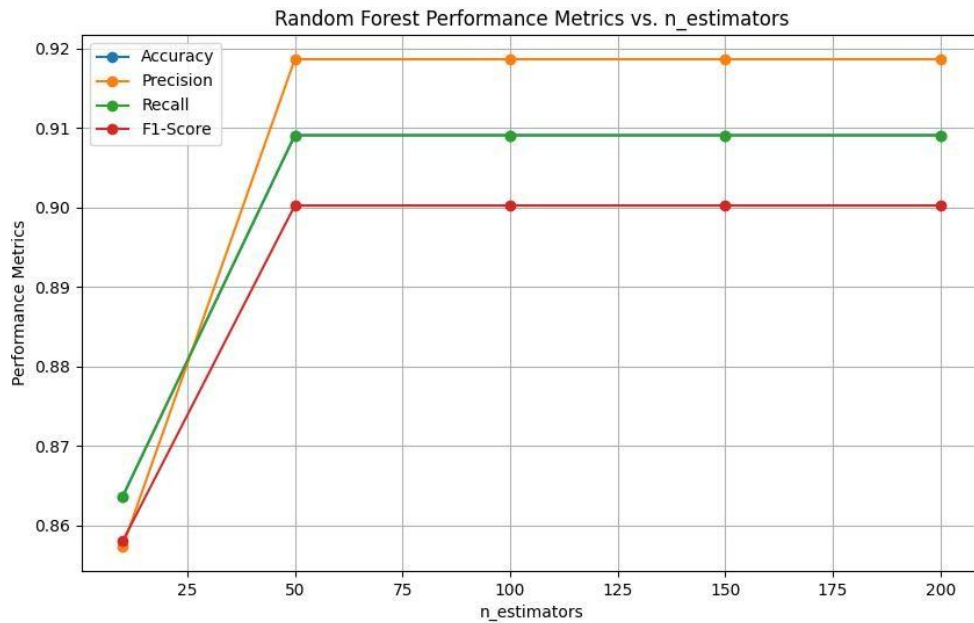


Figure 11: Visualization of Accuracy of RF On label Encoding

Table 9: Accuracy of RF On Hot Encoding with diff no of tree

n_estimators	Accuracy	Precision	Recall	F1-Score
10	0.727273	0.727273	0.727273	0.727273
50	0.818182	0.802233	0.818182	0.800505
100	0.863636	0.857323	0.863636	0.858009
150	0.863636	0.857323	0.863636	0.858009
200	0.863636	0.857323	0.863636	0.858009

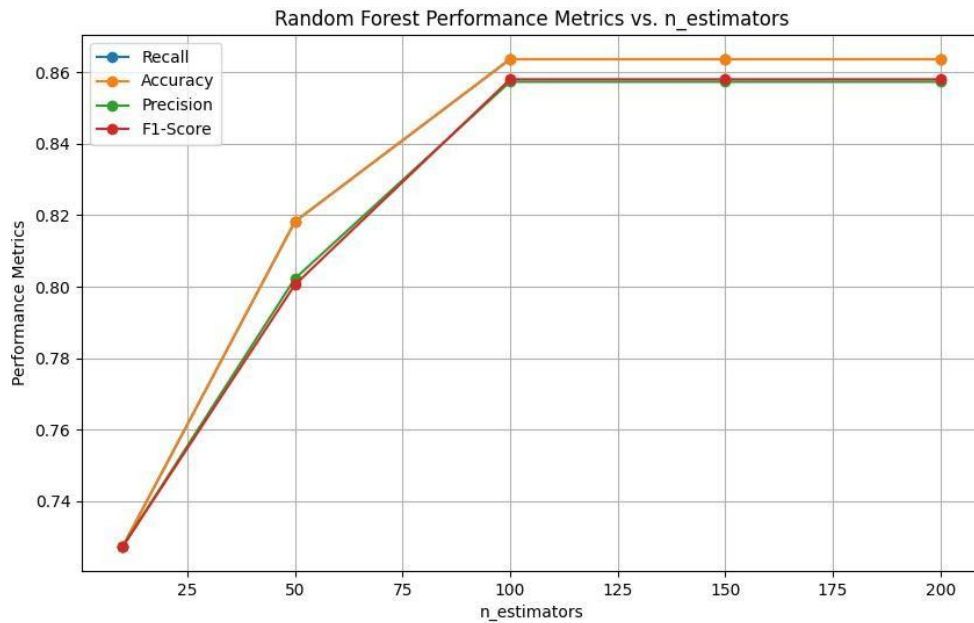


Figure 12: Visualization of Accuracy of RF On Hot Encoding

Table 10: Accuracy of RF On Frequency with diff no of tree

n_estimators	Accuracy	Precision	Recall	F1-Score
10	0.818182	0.818182	0.818182	0.818182
50	0.818182	0.818182	0.818182	0.818182
100	0.772727	0.789773	0.772727	0.779614
150	0.772727	0.789773	0.772727	0.779614
200	0.772727	0.789773	0.772727	0.779614

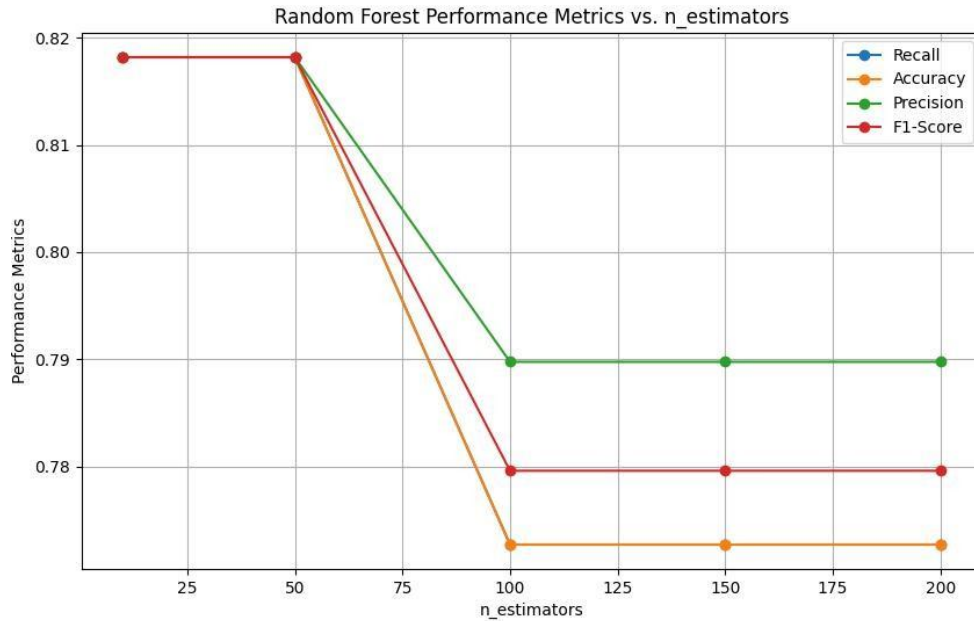


Figure 13: Visualization of Accuracy of RF On Frquency Encoding

4.6.7.2 Multi-Layer Perceptron (MLP)

- Multi-layer Perceptron offers adaptability and versatility for modeling non linear relationships.
- Capable for learning complex Pattern and representation inside diverse feature set.
- Gives versatility to varying data distribution and feature spaces.

Different hidden Layers and other parameters like activation functions (such as logistic and relu), solver algorithms (sgd and adam) and learning_rate strategies (constant and adaptive) were used to evaluate the model and tried on three dataset obtained from techniques used for encoding as mentioned earlier.

Table 11:Accuracy of MLP On Label with diff layer size

Hidden Layer Size	Recall	Accuracy	Precision	F1-Score
(50,)	0.863636	0.863636	0.875947	0.867769
(100,)	0.909091	0.909091	0.909091	0.909091
(50, 50)	0.863636	0.863636	0.857323	0.858009
(80, 80)	0.909091	0.909091	0.909091	0.909091
(100, 100)	0.954545	0.954545	0.957071	0.95267
(128, 256)	0.863636	0.863636	0.857323	0.858009
(80, 80, 80, 256)	0.863636	0.863636	0.875947	0.867769
(256, 512, 80, 80)	0.863636	0.863636	0.857323	0.858009
(80, 80, 80, 80, 80)	0.863636	0.863636	0.857323	0.858009
(128, 80, 512, 80, 80, 80)	0.863636	0.863636	0.857323	0.858009

From experimental results presented in Table 11 it is shown that model architecture has changed more complicated as results shown no significant change for (128, 256), (80, 80, 80, 256), (256, 512, 80, 80), (80, 80, 80, 80, 80) and (128, 80, 512, 80, 80, 80), layers increased, the MLP model gave the results in an increase or decrease that does not follow this rule. Specifically model with hidden layers (100,100) brought the highest results with accuracy 0.954545, precision 0.954545, recall 0.957071 and F1-score 0.95267.

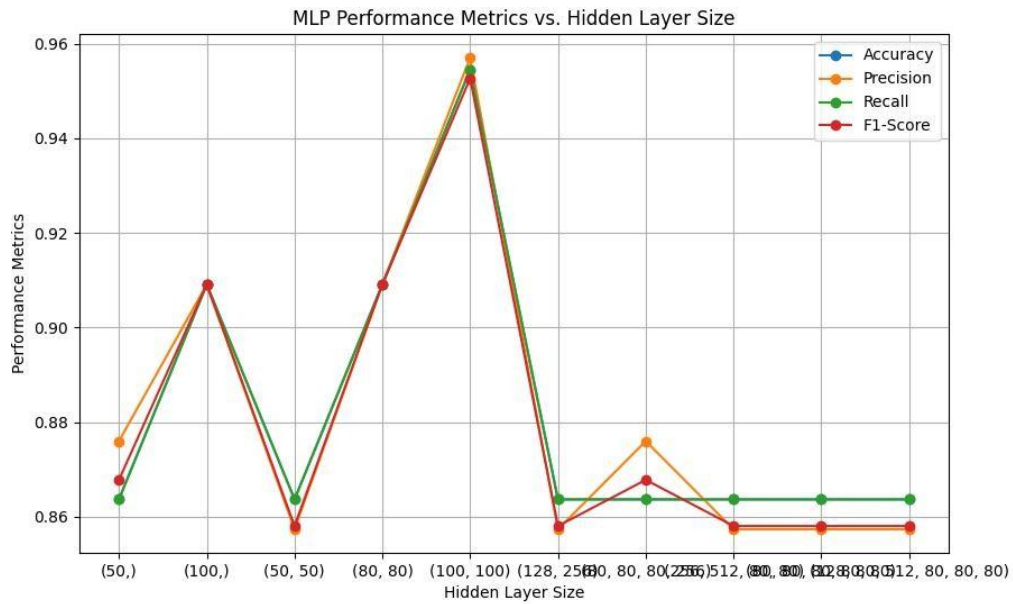


Figure 14: Visualization of Accuracy of MLP On Label Encoding

Table 12: Accuracy of MLP On frequency with diff layer size

Hidden Layer Size	Recall	Accuracy	Precision	F1-Score
(50,)	0.863636	0.863636	0.875947	0.867769
(100,)	0.863636	0.863636	0.875947	0.867769
(50, 50)	0.863636	0.863636	0.875947	0.867769
(80, 80)	0.863636	0.863636	0.875947	0.867769
(100, 100)	0.863636	0.863636	0.875947	0.867769
(128, 256)	0.818182	0.818182	0.851082	0.827652
(80, 80, 80, 256)	0.818182	0.818182	0.851082	0.827652
(256, 512, 80, 80)	0.818182	0.818182	0.851082	0.827652
(80, 80, 80, 80, 80)	0.818182	0.818182	0.851082	0.827652

(128, 80, 512, 80, 0.772727 0.772727 0.831169 0.787954
80, 80)

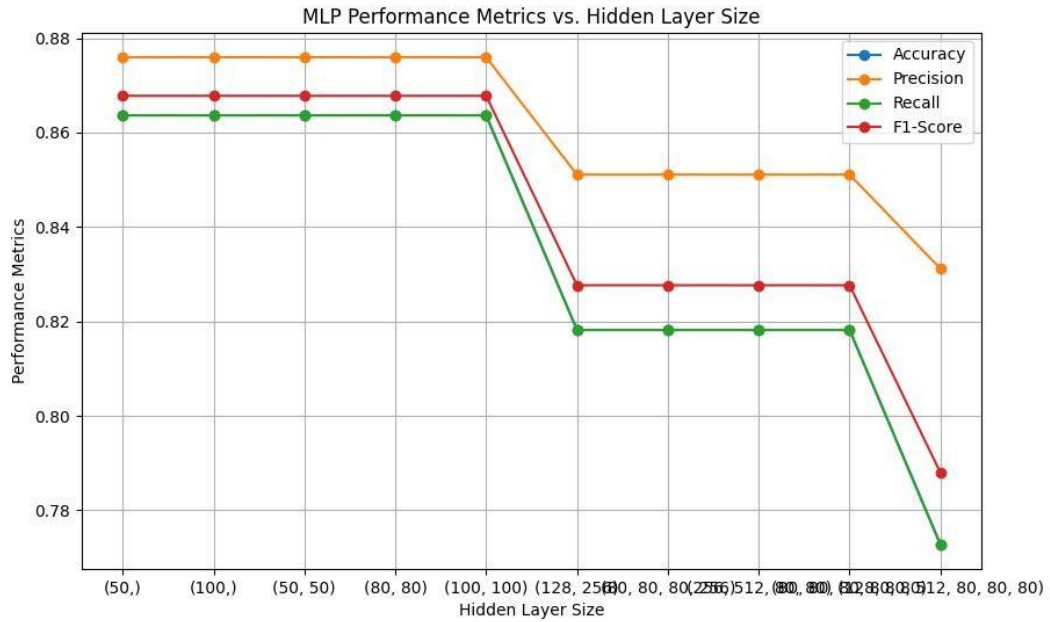


Figure 15: Visualization of Accuracy of MLP On Frequency Encoding

Table 13: Accuracy of MLP On Hot Encoding with diff Layer size

Hidden Layer Size	Recall	Accuracy	Precision	F1-Score
(50)	0.863636	0.863636	0.875947	0.867769
(100)	0.727273	0.727273	0.814297	0.748052
(50, 50)	0.863636	0.863636	0.875947	0.867769
(80, 80)	0.863636	0.863636	0.875947	0.867769
(100, 100)	0.863636	0.863636	0.875947	0.867769

(128, 256)	0.863636	0.863636	0.875947	0.867769
(80, 80, 80, 256)	0.818182	0.818182	0.851082	0.827652
(256, 512, 80, 80)	0.863636	0.863636	0.875947	0.867769
(80, 80, 80, 80, 80)	0.818182	0.818182	0.851082	0.827652
(128, 80, 512, 80, 80, 80)	0.772727	0.772727	0.831169	0.787954

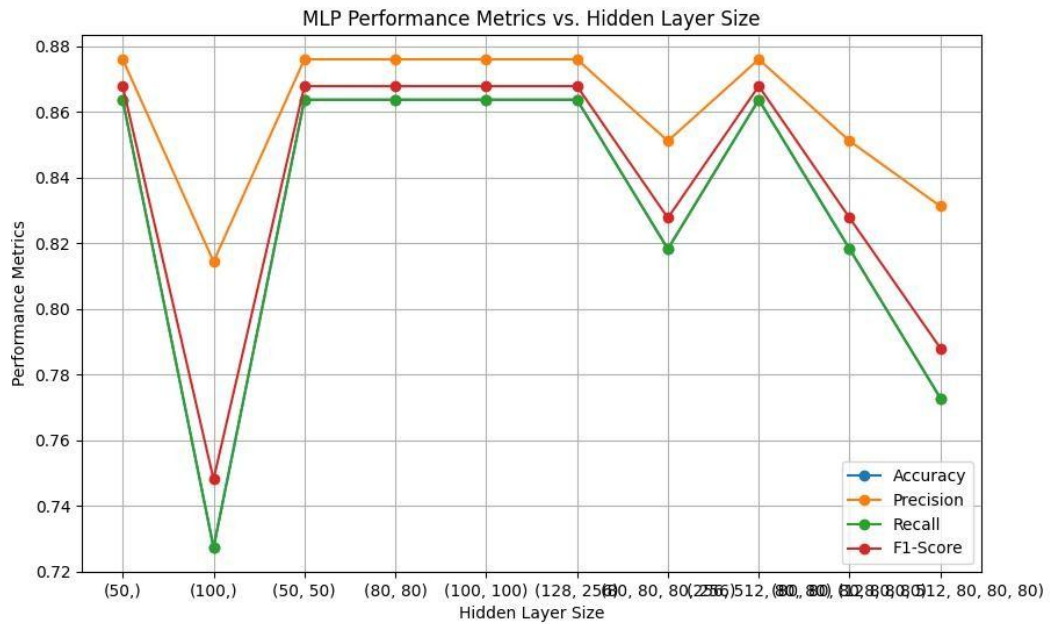


Figure 16: Visualization of Accuracy of MLP On Hot Encoding

4.6.7.3 Long Short-term Memory (LSTM)

- Long Short-Term Memory networks succeed in modeling sequential data and catching temporal dependencies.
- Memory cells and gates effective learning and maintenance of data over long sequence.
- Appropriate for dynamic environment, for example, network traffic analysis.

Different Parameter are used to evaluate the model and tried on three dataset obtained from techniques used for encoding as mentioned earlier.

Table 14: Accuracy of LSTM On Label Encoding with diff configuration

Configuration	Loss	Accuracy	Precision	Recall	F1 Score
32-64	0.603673	0.888889	0.875	1	0.933333
64-128	0.543119	0.777778	0.857143	0.857143	0.857143
128-128	0.41781	0.888889	0.928571	0.928571	0.928571
128-256	0.411782	0.833333	0.866667	0.928571	0.896552

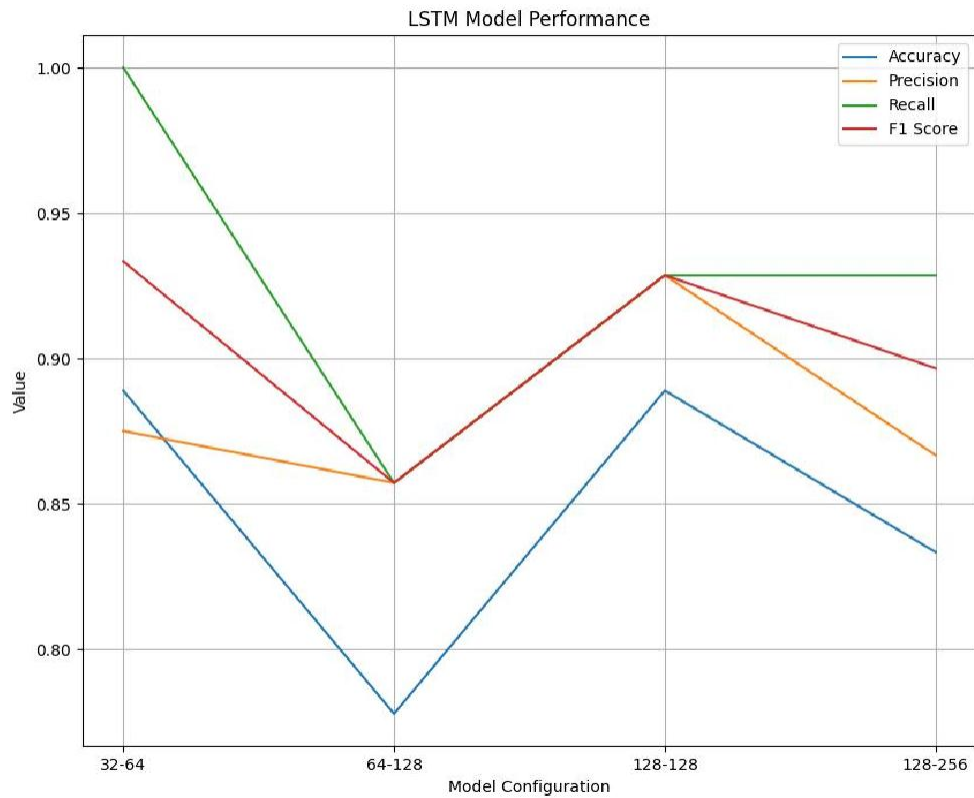


Figure 17: Visualization of Accuracy of LSTM On Label Encoding

Table 15: Accuracy of LSTM On frequency Encoding with diff configuration

Configuration	Loss	Accuracy	Precision	Recall	F1 Score
32-64	0.607631	0.833333	0.866667	0.928571	0.896552
64-128	0.556138	0.777778	0.857143	0.857143	0.857143
128-128	0.492566	0.722222	0.846154	0.785714	0.814815
128-256	0.519816	0.611111	0.818182	0.642857	0.72

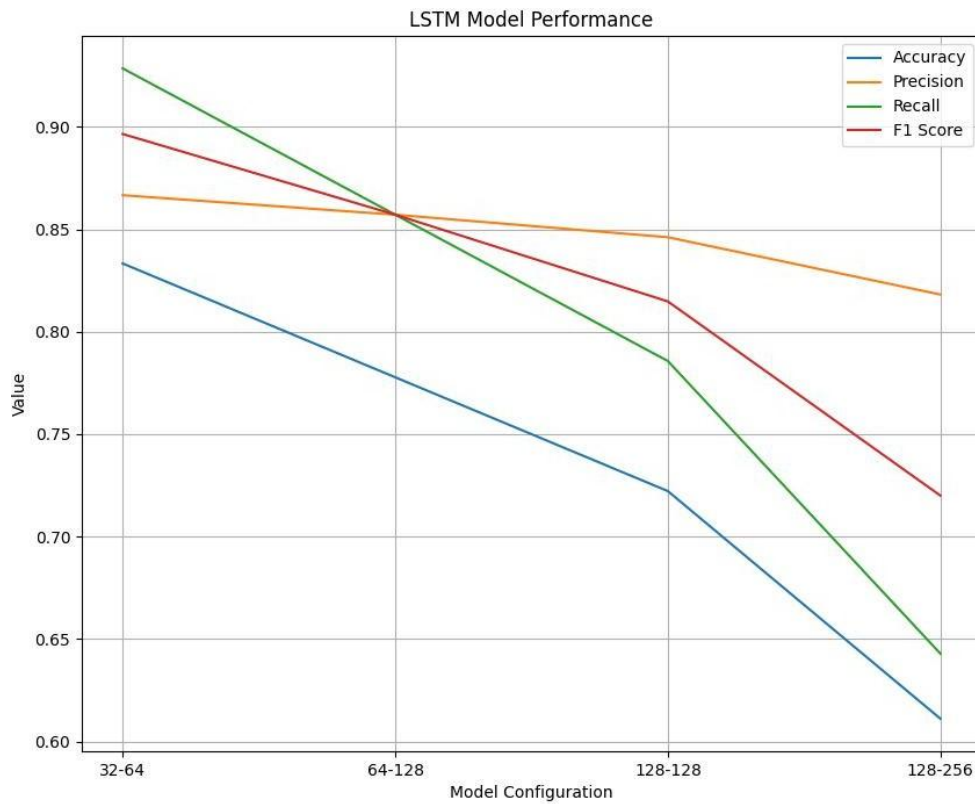


Figure 18: Visualization of Accuracy of LSTM On Frequency Encoding

Table 16: Accuracy of LSTM on hot Encoding with diff configuration

Configuration	Loss	Accuracy	Precision	Recall	F1 Score
32-64	0.581103	1	0	0	0

64-128	0.508483	1	0	0	0
128-128	0.488983	0.888889	0	0	0
128-256	0.397963	0.888889	0	0	0

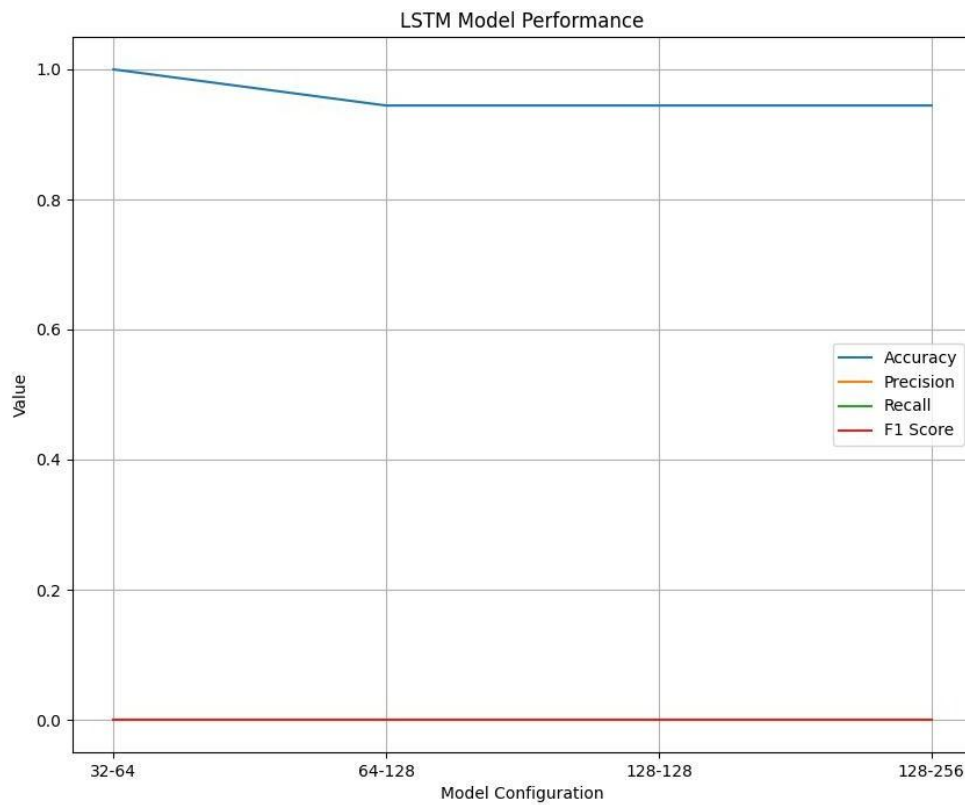


Figure 19: Visualization of Accuracy of LSTM On hot Encoding

4.6.7.4 Naives Bayes:

- Naives Bayes serves in as a basic yet effective probabilistic classifier
- In light of Bayes' theorem , it handles categorical Feature effectively.

Hyperparameter tuning was directed to enhance the Gullible Bayes classifier's presentation for malware identification. Utilizing grid search with cross-validation, different combination of hyperparameters were efficiently investigated, including the smoothing boundary alpha and the choice to learn class earlier probabilities (fit_prior). By training and assessing the classifier across numerous folds of the training data, strong estimates of model execution were obtained while mitigating the risk of overfitting.

Table 17: Accuracy of NB On Label Encoding with diff configuration

alpha	fit_prior	val_accuracy	test_accuracy	precision_true	recall_true	f1_score_true
0.1	TRUE	0.777778	0.666667	1	0.571429	0.727273
0.1	FALSE	0.777778	0.666667	1	0.571429	0.727273
0.5	TRUE	0.666667	0.666667	1	0.571429	0.727273
0.5	FALSE	0.666667	0.666667	1	0.571429	0.727273
1	TRUE	0.666667	0.666667	1	0.571429	0.727273
1	FALSE	0.666667	0.666667	1	0.571429	0.727273

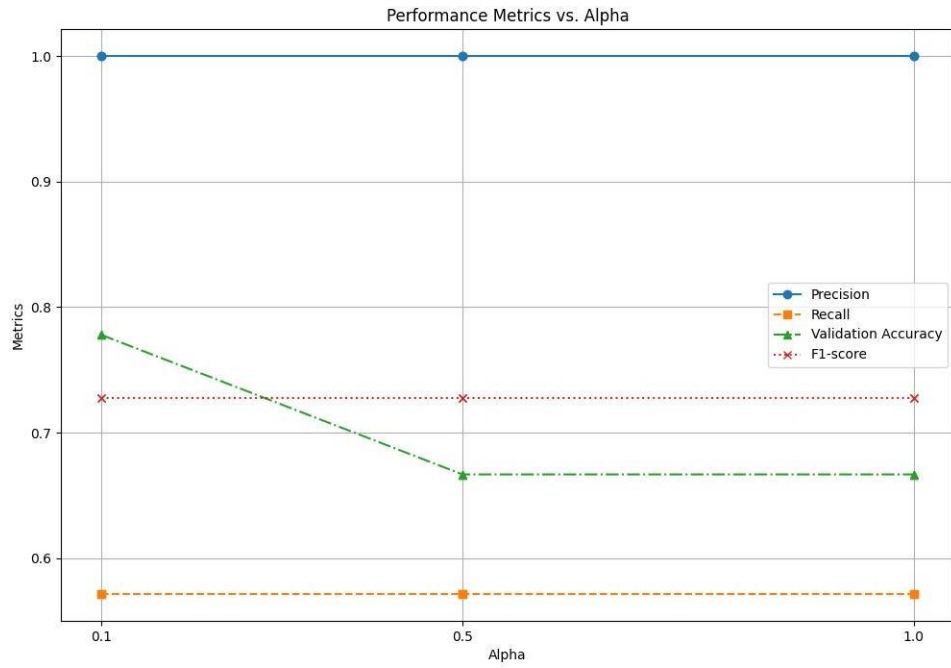


Figure 20: Visualization of Accuracy of NB On Label Encoding

Table 18: Accuracy of NB On frequency Encoding with diff configuration

alpha	fit_prior	val_accur acy	test_accu racy	precision _true	recall_tru e	f1_score_ true
0.1	TRUE	0.777778	0.666667	0.833333	0.714286	0.769231
0.1	FALSE	0.777778	0.666667	0.833333	0.714286	0.769231
0.5	TRUE	0.666667	0.666667	0.833333	0.714286	0.769231
0.5	FALSE	0.666667	0.666667	0.833333	0.714286	0.769231
1	TRUE	0.666667	0.666667	0.833333	0.714286	0.769231
1	FALSE	0.666667	0.666667	0.833333	0.714286	0.769231

Surprisingly, Table IX shows that manipulating alpha (the smoothing parameter) and fit_prior (the class prior learning setting) does not significantly alter the model's effectiveness. This stability across different parameter configurations suggests robust behavior of the Naive Bayes classifier within this context. Despite the consistent performance levels observed in terms of precision (~83.33%), recall (~71.43%), and F1-score (~76.92%), the overall accuracy of approximately 66.67% indicates scope for enhancement. These findings raise important research considerations regarding the inherent limitations of Naive Bayes for this dataset and classification task.

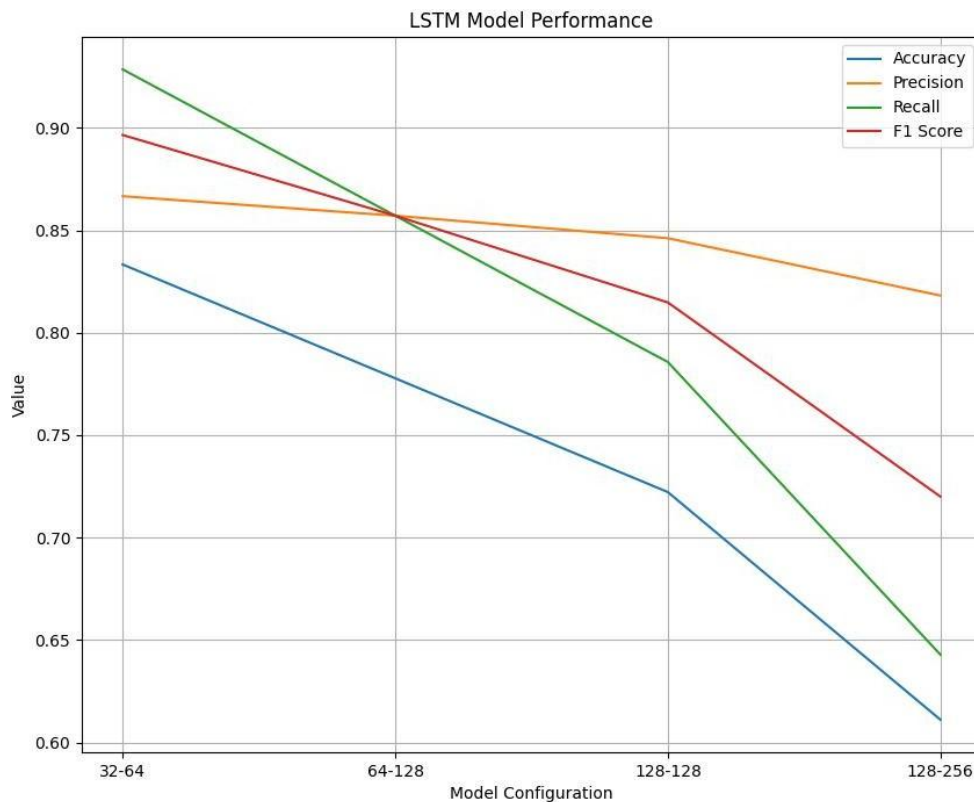


Figure 21: Visualization of Accuracy of NB On Frequency Encoding

Table 19: Accuracy of NB On Hot Encoding with diff configuration

alpha	fit_prio	val_accura	test_accur	precision	recall_tru	f1_score_
	r	cy	acy	_true	e	true
0.1	TRUE	0.777778	0.666667	0.833333	0.714286	0.769231
0.1	FALSE	0.777778	0.666667	0.833333	0.714286	0.769231
0.5	TRUE	0.666667	0.666667	0.833333	0.714286	0.769231
0.5	FALSE	0.666667	0.666667	0.833333	0.714286	0.769231
1	TRUE	0.666667	0.666667	0.833333	0.714286	0.769231
1	FALSE	0.666667	0.666667	0.833333	0.714286	0.769231

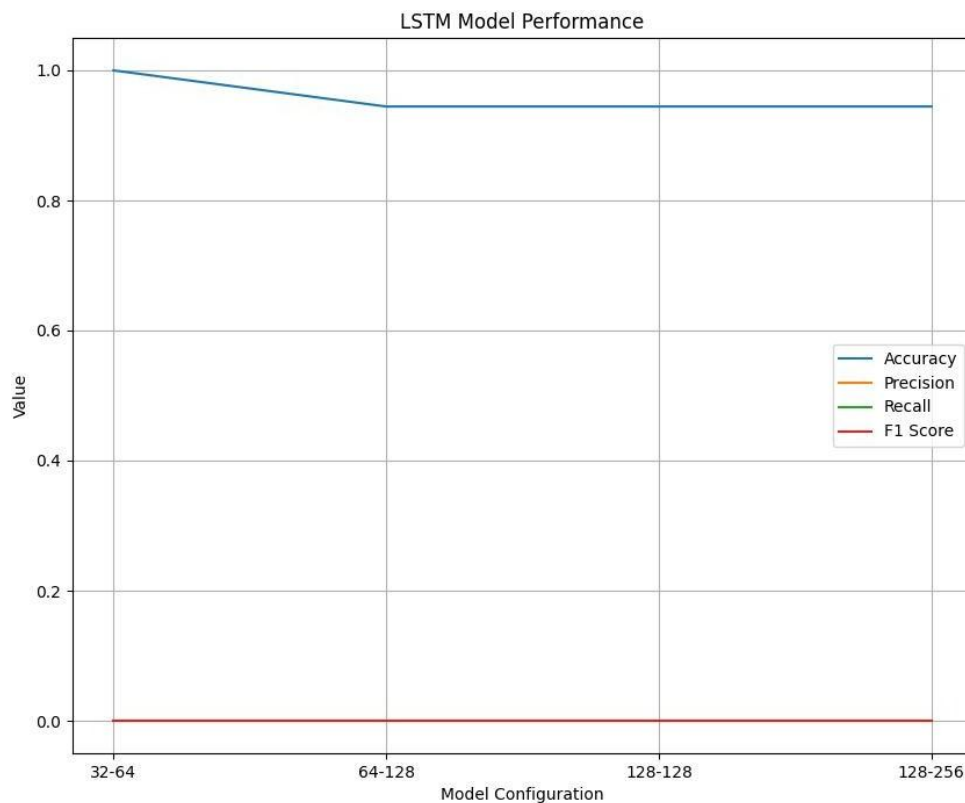


Figure 22: Visualization of Accuracy of NB On HotEncoding

4.6.7.5 Convolutional Neural Network (CNN)

Convolutional Neural Networks are adept at hierarchical feature learning and spatial invariance.

Different Hidden Layer are used to evaluate the model and tried on three dataset obtained from techniques used for encoding as mentioned earlier.

Table 20: Accuracy of CNN on Label Encoding with diff Hidden layers

Hidden Layers	Accuracy	Precision	Recall	F1
[128]	0.944444	0.933333	1	0.965517
[256, 512]	0.944444	0.933333	1	0.965517

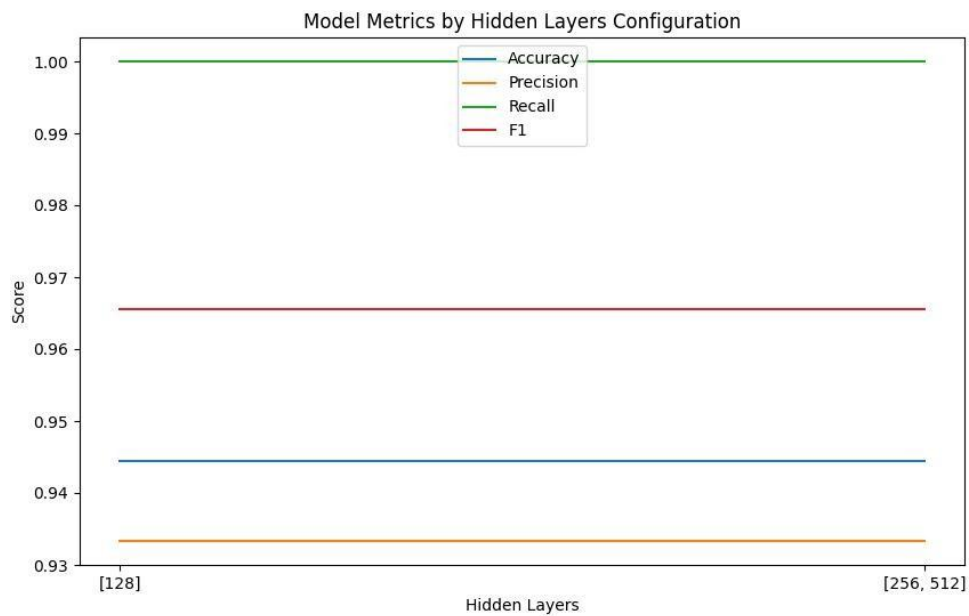


Figure 23: Visualization of Accuracy of CNN On Label Encoding

Table 21: Accuracy of CNN on frequency Encoding with diff Hidden layers

Hidden Layers	Accuracy	Precision	Recall	F1
[128]	0.555556	0.8	0.571429	0.666667
[256, 512, 128]	0.833333	0.866667	0.928571	0.896552

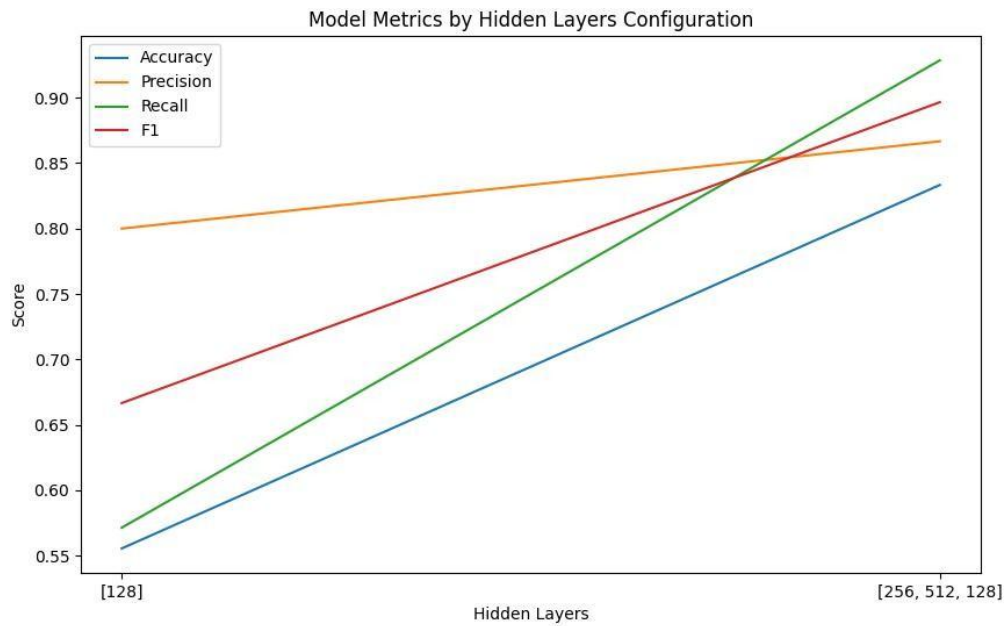


Figure 24: Visualization of Accuracy of CNN On Frequency Encoding

Table 22: Accuracy of CNN on Hot Encoding with diff Hidden layers

Hidden Layers	Accuracy	Precision	Recall	F1
[128]	0.888889	0.875	1	0.933333
[256, 512]	0.944444	0.933333	1	0.965517

The configuration with a single hidden layer of 128 units, the model achieved an accuracy of 88.89%, precision of 87.5%, recall of 100%, and F1-score of 93.33%. This suggests that a simpler architecture with fewer hidden layers can still yield strong performance, particularly in terms of recall where it achieved perfect classification of positive instances. In contrast, the model with two hidden layers (256 units followed by 512 units) demonstrated even higher overall performance. It achieved an accuracy of 94.44%, precision of 93.33%, recall of 100%, and F1-score of 96.55%. The deeper architecture seems to have improved the model's ability to generalize and capture complex patterns in the data, leading to higher precision and F1-score.

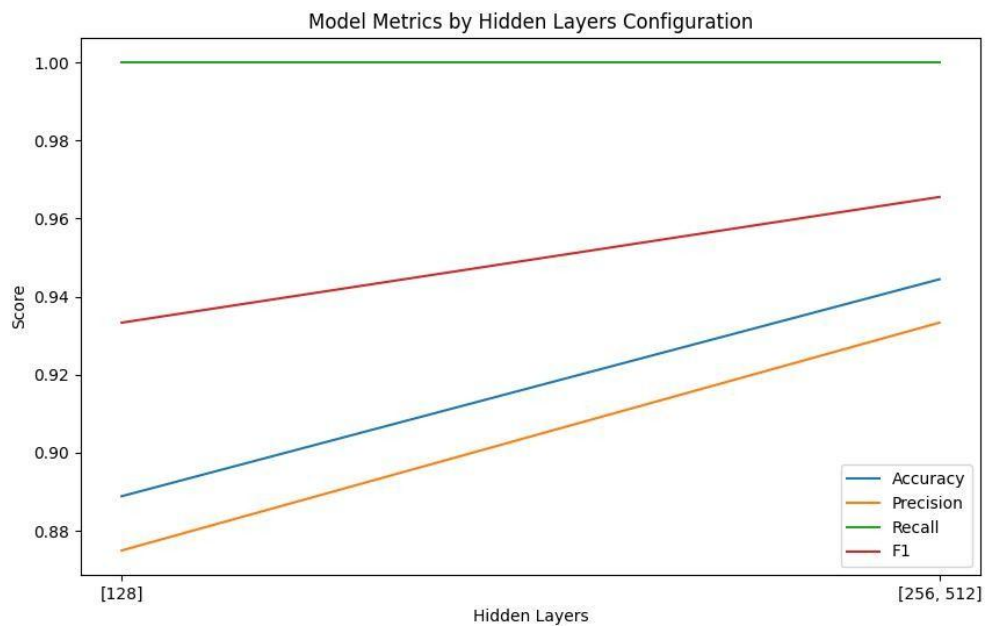


Figure 25: Visualization of Accuracy of CNN On Hot Encoding

Through complete experimentation and assessment of these models, the aim is to identify the best methodology for identifying and mitigating arising security threats in real world environment.

4.6.8 Analysis:

The system's effectiveness is exhibited through its modular architecture, thorough toolset, and use of virtualization technology. Cuckoo Sandbox offers a controlled environment for dynamic malware analysis, which is essential for noticing malware behavior without risking system integrity. The integration of ML enhance setup by giving providing capacities and versatility to new Threats, subsequently working on the effectiveness of the threat detection process.

The experimental result shows that using different Machine learning algorithms and using different dynamic features for detection of Advanced persistent threat can provide an effective and efficient detection system. Mostly features are extracted from registry activity, network activity, file activity and process activity as mostly APT did registry changes to maintain persistence.

From this preliminary analysis, it seems the MLP model using label encoding shows the highest overall metrics, especially with 0.9545 accuracy and 0.952670 F1 score.

Table 23: Performance Comparison Machine Learning Algorithms

Algorithm	Encoding	Configuration	Accuracy	Precision	Recall	F1-Score
Random Forest	Label Encoding	200	90.9091%	91.866%	90.9091%	90.0253%
Random Forest	Hot-Encoding	200	86.3636%	85.7323%	86.3636%	85.8009%
Random Forest	Frequency Encoding	10	0.818182%	0.818182%	0.818182%	0.818182%

MLP	Label Encoding	(100, 100)	95.4545%	95.7071%	95.4545%	95.2670%
MLP	Frequency Encoding	(50, 50)	86.3636%	87.5947%	86.3636%	86.7769%
MLP	Hot- Encoding	(256, 512, 80, 80)	86.3636%	0.863636	0.875947	0.867769
LSTM	Label Encoding	128-128	88.8889%	92.8571%	92.8571%	92.8571%
LSTM	Frequency Encoding	32-64	83.3333%	86.6667%	92.8571%	89.6552%
LSTM	Hot- Encoding	128-256	88.8889%	0	0	0
Naive Bayes	Label Encoding	0.1/0.5/1	77.7778%	83.3333%	71.4286%	76.9231%
Naive Bayes	Frequency Encoding	0.1/0.5/1	77.7778%	83.3333%	71.4286%	76.9231%
Naive Bayes	Hot- Encoding	0.1/0.5/1	77.7778%	83.3333%	71.4286%	76.9231%
CNN	Label Encoding	[256, 512]	94.4444%	93.3333%	100%	96.5517%
CNN	Frequency Encoding	[256, 512, 128]	83.3333%	86.6667%	92.8571%	89.6552%
CNN	Hot- Encoding	[256, 512]	94.4444%	93.3333%	100%	96.5517%

	Encoding					
--	----------	--	--	--	--	--

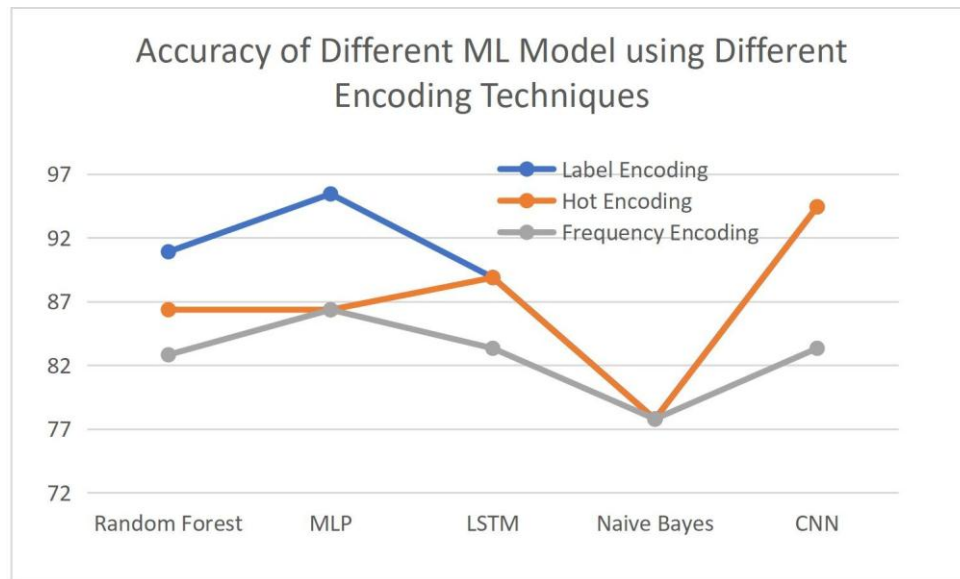


Figure 26: Accuracy of diff ML Model using diff encoding techniques

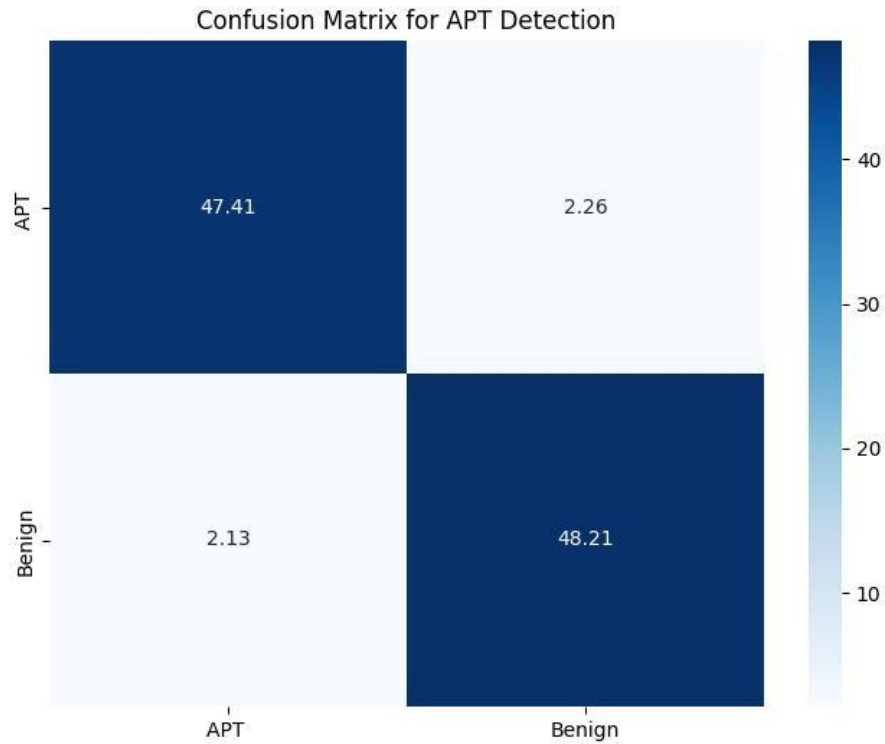


Figure 27: Confusion matrix of the MLP model with parameters (100-100), y-axis=true label, x-axis=predict label

So using different features extracted from dataset and different ML Algorithm with encoding techniques, accuracy 0.9545 achieved by utilizing MLP with Label Encoding technique over parameter (100,100) while referenced from previous research it is concluded that after changing some features as dataset created enhanced the efficiency and detection of most sophisticated cyber threat. Comparison with previous paper[3] it is cleared that using MLP on created dataset will give better accuracy .

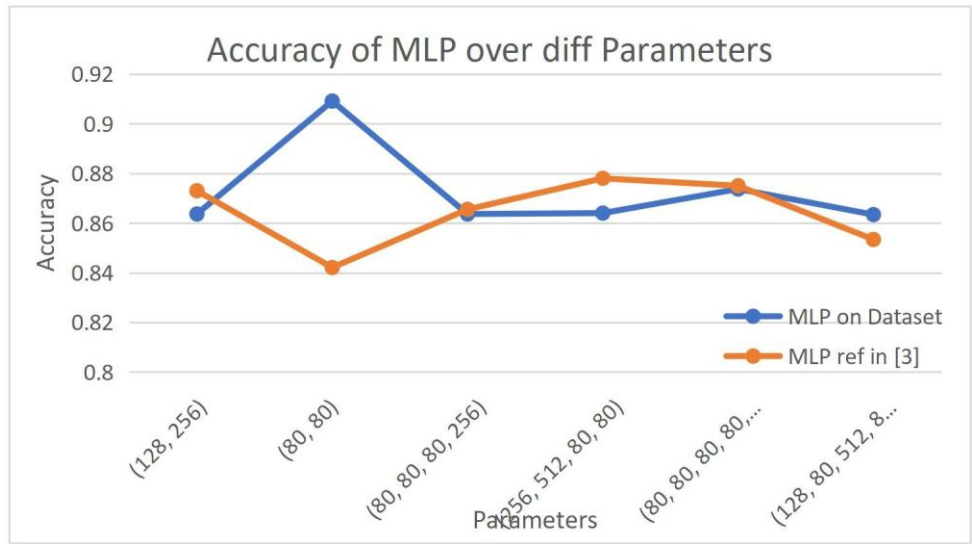


Figure 28: Comparison of MLP over diff Datasets

Chapter 5

Conclusion

This thesis has efficiently explored the use of Machine Learning techniques to enhance the detection capability of Advanced Persistent Threat (APTs) inside Windows operating system . Through a detail examine of lifecycle, behaviors and detection challenges related with APTs, this research has proposed an ML-based system that essentially progresses the field of cyber security.

The essential Contribution of this study lies in the development of robust detection system that integrates the dynamic malware analysis tools, like Cuckoo Sandbox, with Machine Learning Algorithms. This methodology considered the detailed examination of malware dynamic Behavior in an isolated environment identifying the anomalies characteristics of APTs. The system effectiveness and efficiency were additionally improved by using the different machine learning algorithm, including Random Forest, Multi-Layers Perceptron (MLP), and Long short-term Memory (LSTM), each tried and tuned to enhance execution across various features and encoding strategies.

The research discoveries showed that the MLP model, using Label encoding, accomplished the highest accuracy (95.45%) and F1 score (95.267%), highlighting the potential of neural network in cybersecurity. These result validate the proposed model's efficiency as well as feature selection and data preprocessing in building effective security solutions.

5.1 Future Work:

The dataset used is extensive as it contains list of features yet doesn't completely catch the developing nature of APTs, known for their adaptability. As attacker continually developing new techniques, the dataset can quickly become outdated, which may reduce the accuracy against new model.

To proactively address this challenge, future work should focus on enhancing the dataset while getting new APT samples, particularly unpublished zero-day exploit, pose significant challenges by malware sample is publicly not available. So, government organizations, and academic institute could prove invaluable. These organizations could work to get a broad range of arising threats under controlled and secure circumstances, considering the constant development of the dataset.

Bibliography

- [1] A. C. M. D. H. Adel Alshamrani, "A Survey on Advanced Persistent Threats: Techniques, Solutions, Challenges, and Research Opportunities," *IEEE COMMUNICATIONS SURVEYS & TUTORIALS*, vol. 21, Second Quarter 2019.
- [2] X. U.-P. M. G. S. M. Frederick Barr-Smith, "Survivalism: Systematic Analysis of Windows Malware Living-Off-The-Land," *2021 IEEE Symposium on Security and Privacy (SP)*, 2021.
- [3] C. H. D. Do Xuan, "A new approach for APT malware detection based on deep graph network for endpoint systems," *Applied Intelligence*, vol. 52, pp. 14005-14024, 2022/09/01.
- [4] U. S. A. Hira Shahzadi Sikandar, "An Adversarial Approach: Comparing Windows and Linux Security Hardness Using Mitre ATT&CK Framework for Offensive Security," *IEEE 19th International Conference on Smart Communities*, 2022.
- [5] D. McWhorter, "APT1: Exposing One of China's Cyber Espionage units", Mandiant, Alexandria 2022.
- [6] R. S. Ross, "Managing Information Security Risk: Organization, Mission, and Information System View", *Nat. Inst. Stand. Technol*, pp. 800-39, 2011.
- [7] R. Kissel, "Glossary of key information security terms", Nist Interagency/Internal

Rep Jun. 2013.

- [8] L. D. a. C. H. P. Chen, "A study on advanced persistent threats", Proc. IFIP Int. Conf. Commun. Multimedia Security 2014.
- [9] D. G. • M. Theocharidou, "Critical Infrastructure Security and Resilience," springer, pp. 222-230, 2019.
- [10] J. C. M. M. F. Antoine Lemay a, "Survey of publicly available reports on advanced persistent threat actors," computers & security 72 , p. 26–59, (2018)
- [11] Alshamrani A, Chowdhary A, Myneni S, Huang D (2019) A survey on advanced persistent threats: techniques, solutions, challenges, and research opportunities. IEEE Comm Surv Tutor 21(2):1851–1877.
- [12] Quintero-Bonilla S, Rey Á (2020) A new proposal on the advanced persistent threat: a survey. Appl Sci 10:38–74
- [13] H. B. M. A. S. A. A. D. N. L. Tim Bai, "RDP-based Lateral Movement Detection using Machine Learning," Computer Communications, pp. 9-19, january 2021.
- [14] N. L. N. E.-S. a. O. M. C. Sara Ayoubi, "Machine Learning for Cognitive Network Management," IEEE Communications Magazine, pp. 158-165, January 2018.
- [15] C. W. Z. F. C. H. Yong Fanga, " LMTracker: Lateral movement path detection based on heterogeneous graph embedding," Neurocomputing 474 , pp. 37-47, 2022.
- [16] C. H. D. Do Xuan, "A new approach for APT malware detection based on deep graph

network for endpoint systems”, Applied Intelligence, vol. 52, pp. 14005-14024 ,2022/09/01

- [17] J. X. Y. W. F. Z. X. G. Weijie Hana, "APTMallInsight: Identify and cognize APT malware based on system call information and ontology knowledge framework," information Sciences 546 , pp. 633-664, 2021.
- [18] Y. L. J. L. W. S. J. C. R. Z. X. H. Jianyi Liua, "Twostatistical traffic features for certain APT group identification," Journal of Information Security and Applications 67, vol. 103207, 2022.
- [19] B. S. R. Z. C. Z. Xingjie Huang, "APT Attack Detection Method Based on Traffic Log features," IEEE 2nd International Conference on Computer Systems, 2022 .
- [20] J. W. S. S. Talha Ongun, "Living-Off-The-Land Command Detection Using Active Learning," ACM ISBN, November 2021.
- [21] Interactive Online Sandbox: <https://www.virustotal.com/gui/home/upload>
<https://any.run/>
- [22] Samples Available: <https://bazaar.abuse.ch/browse.php?search=tag%3A>
- [23] Apts Hashes(MD5): [https://github.com/RedDrip7/APT_Digital_Weapon/](https://github.com/RedDrip7/APT_Digital_Weapon/blob/master/APT40/APT40_hash.md)
[blob/master/APT40/APT40_hash.md](https://github.com/RedDrip7/APT_Digital_Weapon/blob/master/APT40/APT40_hash.md)

