

# AI-Driven Malware Detection and Uncovering Anomalous Traffic Patterns with SIEM Integration



By

**Waqas Ahmad**

Registration No: 00000401385

Department of Information Security

Military College of Signals

National University of Sciences and Technology (NUST)

Islamabad, Pakistan

(September 2024)

# AI-Driven Malware Detection and Uncovering Anomalous Traffic Patterns with SIEM Integration



By

Waqas Ahmad

(Registration No: 00000401385)

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

in partial fulfillment of the requirements for the degree of

Masters in

Information Security

Supervisor: Prof. Dr. Muhammad Faisal Amjad

Military College of Signals  
National University of Sciences and Technology (NUST)  
Islamabad, Pakistan  
(2024)


**THESIS ACCEPTANCE CERTIFICATE**

Certified that final copy of MS Thesis written by Mr. Waqas Ahmad, Registration No. 00000401385, 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. Muhammad Faisal Amjad

Date: 24/09/2024

Signature (HOD):  \_\_\_\_\_  
HOD  
Information Security  
Military College of Sigs

Date: 24/09/2024

Signature (Dean/Principal)  \_\_\_\_\_  
Date: 27/4/10/24 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 **NS Waqas Ahmad, MSIS-21 Course** Regn No **00000401385** Titled: **"AI-Driven Malware Detection and Uncovering Anomalous Traffic Patterns with SIEM Integration"** be accepted in partial fulfillment of the requirements for the award of **MS Information Security** degree.

**Examination Committee Members**


1. Name: **Ammar Hassan (MCS-NUST)**

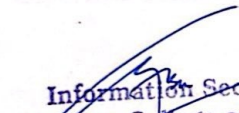
Signature: 

2. Name: **Dr. Faiz ul Islam (MCS-NUST)**

Signature: 

Supervisor's Name: **Dr. Muhammad Faisal Amjad**

Signature: 


  
Ho.  
Information Security  
Military College of Sigs  
Head of Department

Date: \_\_\_\_\_

24/9/2024  
Date

**COUNTERSIGNED**

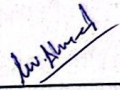
Date: 4/10/24

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

### **CERTIFICATE OF APPROVAL**


This is to certify that the research work presented in this thesis, entitled "AI-Driven Malware Detection and Uncovering Anomalous Traffic Patterns with SIEM Integration" was conducted and defended by Waqas Ahmad on 18<sup>th</sup> September 2024 under the supervision of Dr. Muhammad Faisal Amjad. 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:** Waqas Ahmad

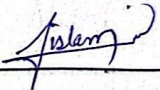
Signature:  \_\_\_\_\_

Examination Committee:

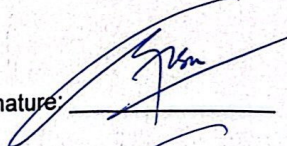
a) External Examiner 1: Ammar Hassan (MCS-NUST)

Signature:  \_\_\_\_\_

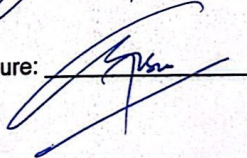
b) External Examiner 2: Dr. Faiz ul Islam (MCS-NUST)

Signature:  \_\_\_\_\_

Name of Supervisor: Dr. Muhammad Faisal Amjad

Signature:  \_\_\_\_\_

Name of Dean/HOD: Dr. Muhammad Faisal Amjad

Signature:  \_\_\_\_\_

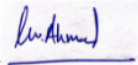
Associate Professor  
**Dr. Muhammad Faisal Amjad**  
Dept of Info Security  
Military College of Signals (NUST)

**PLAGIARISM UNDERTAKING**

I solemnly declare that research work presented in the thesis titled “**AI-Driven Malware Detection and Uncovering Anomalous Traffic Patterns with SIEM Integration**” 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 the 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 the award of the MS degree, the University reserves the right 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: 

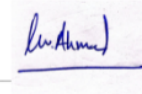
Name: Waqas Ahmad

Date: 18 September, 2024

**AUTHOR'S DECLARATION**

I **Waqas Ahmad** hereby state that my MS thesis titled "**AI-Driven Malware Detection and Uncovering Anomalous Traffic Patterns with SIEM Integration**" is my own work and has not been submitted previously by me for taking any degree from the 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: \_\_\_\_\_

Name: Waqas Ahmad

Date: 18 September, 2024

## DEDICATION

*This thesis is dedicated to my Family, Teachers, and Friends for their unconditional love, endless support, and continuous encouragement.*



## **ACKNOWLEDGEMENTS**

I would like to convey my gratitude to my supervisor, Prof. Dr. Muhammad Faisal Amjad, for his supervision and constant support. His invaluable help, constructive comments, and suggestions throughout the experimental and thesis work are major contributions to the success of this research. I would also like to thank my committee members, Mr. Ammar Hassan and Dr. Faiz ul Islam, for their useful comments and suggestions.

# Contents

<b>ACKNOWLEDGEMENTS</b>	<b>VII</b>
<b>LIST OF TABLES</b>	<b>X</b>
<b>LIST OF FIGURES</b>	<b>XI</b>
<b>LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS</b>	<b>XII</b>
<b>ABSTRACT</b>	<b>XIV</b>
<b>1 INTRODUCTION</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Problem Statement . . . . .	2
1.3 Objective . . . . .	3
1.4 Relevance to National Needs . . . . .	5
1.5 Contributions . . . . .	6
<b>2 LITERATURE REVIEW</b>	<b>8</b>
2.1 Overview . . . . .	8
2.2 Related Work . . . . .	9
2.3 Summary . . . . .	14
<b>3 PROPOSED RESEARCH METHODOLOGY</b>	<b>16</b>
3.1 Dataset . . . . .	16
3.2 Working of Isolation Forest and $K$ -means in Proposed Solution . . . . .	18
3.3 Integration of Anomaly Detection with SIEM . . . . .	20
3.4 Features Extraction from HTTP logs . . . . .	21
3.5 Scaling Strategies for the proposed solution . . . . .	22
3.6 System Design . . . . .	23
3.7 System Design of the HTTP malicious files detection . . . . .	27

<b>4</b>	<b>RESULTS AND FINDINGS</b>	<b>30</b>
4.1	Experimental Setup . . . . .	30
4.2	Visualization and Analysis of Anomalous HTTP Logs . . . . .	33
4.3	Results and findings of HTTP Malicious Files Detection . . . . .	39
<b>5</b>	<b>CONCLUSIONS AND FUTURE RECOMMENDATION</b>	<b>44</b>
	<b>REFERENCES</b>	<b>47</b>

# List of Tables

2.1	Overview of Datasets Used in the Study . . . . .	14
3.1	Key Features and their description . . . . .	18
3.2	Summary of Key Points: Isolation Forest vs. K-means Clustering . . .	20
3.3	Integration of Anomaly Detection with SIEM Solution Using ELK Stack	22
3.4	Extracted Features from Zeek HTTP Logs . . . . .	22
3.5	Apache Kafka's Data Management and Processing Workflow . . . . .	25
3.6	System Design and Data Flow . . . . .	27
3.7	Random Forest Model Details . . . . .	28
3.8	System Overview and Data Flow . . . . .	29
3.9	Random Forest Model Details . . . . .	29
4.1	Features of HTTP Logs in Elasticsearch Index Pattern . . . . .	35
4.2	Key Performance Metrics & Anomaly Detection Techniques . . . . .	37
4.3	Key Performance Metrics for Anomaly Detection Models . . . . .	37
4.4	Overview of Anomaly Detection Techniques . . . . .	38
4.5	Silhouette Score Interpretation . . . . .	38
4.6	Silhouette Score Interpretation . . . . .	39

# List of Figures

3.1	Flowchart of the Solution and the SIEM Network Diagram . . . . .	24
3.2	Flowchart of HTTP malicious files detection . . . . .	27
4.1	HTTP Logs and Anomalous HTTP Logs . . . . .	31
4.2	Cluster Groups of Anomalous HTTP Logs . . . . .	32
4.3	Decision Function of the Proposed Machine Learning Model) . . . . .	32
4.4	HTTP Logs Overview in Elasticsearch . . . . .	35
4.5	Anomalous HTTP Logs Clustering & Analysis . . . . .	36
4.6	Details of HTTP Files Logs and Anomalous HTTP File . . . . .	40
4.7	Heatmap of confusion matrix . . . . .	42

# LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS

<b>SIEM</b>	Security Information and Event Management
<b>SOC</b>	Security Operations Center
<b>HTTP</b>	Hypertext Transfer Protocol
<b>HTTPS</b>	Hypertext Transfer Protocol Secure
<b>IDS</b>	Intrusion Detection System
<b>UNC</b>	Unsupervised Niche Clustering
<b>PCA</b>	Principal Component Analysis
<b>MDE</b>	Maximal Density Estimator
<b>KDD</b>	Knowledge Discovery in Databases
<b>UNIDS</b>	Unsupervised Network Intrusion Detection System
<b>SSC</b>	Sub-Space Clustering
<b>PE Files</b>	Portable Executable Files
<b>ML</b>	Machine Learning
<b>EAC</b>	Electronic Access Control
<b>NIDS</b>	Network Intrusion Detection System
<b>NADS</b>	Network Anomaly Detection System
<b>SVM</b>	Support Vector Machine
<b>DLLs</b>	Dynamic-Link Libraries
<b>API</b>	Application Programming Interface
<b>CNN</b>	Convolutional Neural Network
<b>LightGBM</b>	Light Gradient Boosting Machine
<b>EDR</b>	Endpoint Detection and Response
<b>NDR</b>	Network Detection and Response
<b>SOAR</b>	Security Orchestration, Automation, and Response
<b>TI</b>	Threat Intelligence
<b>WAF</b>	Web Application Firewall

<b>VMS</b>	Virtual Memory System (commonly also Video Management System depending on context)
<b>ETL</b>	Extract, Transform, Load
<b>IPS</b>	Intrusion Prevention System
<b>DPI</b>	Deep Packet Inspection
<b>IP</b>	Internet Protocol
<b>SSL</b>	Secure Sockets Layer
<b>TLS</b>	Transport Layer Security

# ABSTRACT

The amount of network traffic continuously rises, and network services are becoming increasingly more complex and vulnerable. Intrusion detection systems are employed to safeguard these networks. Signature-based intrusion detection cannot detect new attacks, so anomaly detection is necessary. Also, most of the traditional Security Information and Event Management (SIEM) doesn't have any live dataset creation and anomalous data logs detection. In this solution, we propose a framework for detecting anomalies in Hypertext Transfer Protocol (HTTP) logs which would enable fast detection of anomalies, visualization of network traffic, and integration with SIEM solutions. This solution gets HTTP logs from the network and preprocesses them. Then, the solution checks the traffic for anomaly detection by applying the Isolation Forest algorithm using unsupervised learning. Later, we will then explore those anomalies with clustering using the  $K$ -means method. The proposed the solution uses no predefined dataset file. But, it trains on the live data logs, where the user checks for any anomalous data logs within a given timestamp. The logs from this specified timestamp serves as the dataset for the model's training. The solutions are on live network HTTP logs, but in this thesis, we are using this solution on a test network where we have 153 logs, of which 32 are declared anomalous by our proposed solution and after that, the clustering on those anomalous data logs were applied. Finally, the results are sent to the SIEM solution, which visualizes the network and the anomalous traffic. Besides this, the solution is quick to detect intrusion attempts. This method is significant in the sense that it contributes to proactive cybersecurity strategies specifically designed to meet the requirements of Security Operations Center (SOC)



analysts and threat-hunting teams.

# Chapter 1

## INTRODUCTION

### 1.1 Introduction

Cyberattacks have become increasingly varied and large-scale in recent years, resulting in significant losses and adverse impacts. Consequently, the importance of network intrusion detection has grown significantly. There are two primary types of network intrusion detection systems [1, 2].

The most popular method of intrusion detection remains signature based. This signature based method involves searching for invasive behavior using pre-set attack rules. A rule is something that can be applied to the network traffic or some other action, and an alarm will be generated once the rule is matched. Signature-based detection operates a fast and wide range of attacks that predefined rules can detect [3][4][5]. In addition, it usually results in a low false alarm rate. The downside of this approach is that signatures for attacks need to be defined manually. For example, if attackers quickly find and exploit a new vulnerability before appropriate rules are created, unknown vulnerabilities can still be exploited until Intrusion Detection rule (IDS) rule has been improved.

Anomaly-based systems work on a different principle. Whenever new behavior or traffic tips in, it is compared with the normal profile, and all the discrepant behaviors are considered as anomalies. This approach enables the detection of fresh and unknown attack attempts [6][7][8]. Moreover, the network profile can be updated in real-time, thus adjusting it to variations in network traffic. [9].

However, a significant challenge arises when deploying these systems on networks: there is a lack of proper monitoring systems to check the alerts generated by this kind of system. The system state can always be monitored. The implementation of such solutions (e.g., security information and event management (SIEM) solutions) has been challenging due to the prohibitive costs, high computational resource requirements, and continual needs for highly-trained analysts or threat-hunting teams manning most organizations security operations center (SOC)s directly round-the-clock [10].

Also most malware detection solutions use static or dynamic analysis techniques, while some use both techniques [11]. To construct a strong malware detection mechanism, machine learning (ML) techniques employ static analysis methods to derive a feature set. Windows utilizes a type of executable format known as the Portable Executable (PE). Some examples of executable file types are SYS, SCR, DLL, EXE, APP, BAT, etc. [12]. For the Windows OS loader to manage all wrapped executable code, a PE is a data structure that includes every essential piece of information it needs. [13]. A PE file has three headers: the file optional header, the MS-DOS header, and an optional header [12].

## 1.2 Problem Statement

Traditional Security Information and Event Management (SIEM) solutions play a crucial role in security monitoring by aggregating and analyzing log data from various network components to detect anomalies that may indicate malicious activities. Within these systems, Security Operations Center (SOC) analysts are tasked with the crucial responsibility of reviewing daily logs for potential threats. SIEM solutions incorporate various modules designed to detect unusual patterns and generate alerts based on these observations. However, the efficacy of these modules is often constrained by their reliance on static datasets, which are pre-configured to recognize known threats.

One significant limitation of traditional SIEM solutions is their dependence on static datasets. These datasets are designed to identify previously recognized threats and patterns, but as the cyber threat landscape evolves, such datasets can quickly become outdated. As new strains of malware and sophisticated cyber threats emerge, traditional SIEM solutions may lack the necessary information to identify these novel threats effectively. This limitation can lead to an increased risk of undetected malware and emerging attacks, thereby compromising the overall security posture of the organization.

Furthermore, the sheer volume of data that SOC analysts are required to manage poses an additional challenge. With the continuous growth of data generated by network activities, the workload on SOC analysts has intensified. Analysts are tasked with sifting through vast amounts of logs and alerts on a daily basis. This overwhelming volume of information can lead to analyst fatigue, resulting in slower response times and a higher likelihood of missing critical security incidents. The increased workload exacerbates the difficulty of distinguishing genuine threats from false positives, further stressing the need for more efficient and scalable solutions.

In summary, while traditional SIEM solutions offer a foundational level of security monitoring, their reliance on static datasets and the escalating burden on SOC analysts underscore the need for more dynamic and adaptive approaches to threat detection. To address these challenges, there is a pressing need for advanced detection techniques that leverage machine learning and adaptive algorithms to enhance the ability to identify emerging threats and reduce the manual workload on security analysts.

### 1.3 Objective

The following are the objectives of this research

**Develop a Framework for Integration:** The primary goal of this research is to design and develop a comprehensive and robust framework that seamlessly integrates K-means clustering with Zeek HTTP logs and

historical data from a Security Information and Event Management (SIEM) solution. This framework aims to streamline the process of anomaly detection by combining the real-time data captured by Zeek with historical data stored in SIEM systems. By achieving this integration, the framework will enhance the efficiency and accuracy of detecting abnormal traffic patterns and potential security threats.

**Implement Real-Time Anomaly Detection:** A crucial aspect of this research is the application of K-means clustering to Zeek HTTP logs in real time. The objective is to enable the immediate detection of abnormal behaviors as they occur, thereby allowing for swift responses to potential security incidents. This real-time capability is essential for maintaining an effective security posture, as it facilitates the early identification of anomalies that could indicate malicious activities or network intrusions.

**Support Threat Hunting Efforts:** This research seeks to provide a tool that supports and enhances threat-hunting activities by automating a significant portion of the anomaly detection process. By leveraging the K-means clustering approach, the tool will assist security professionals in identifying and investigating unusual patterns in network traffic. This automation will free up valuable time for security analysts, enabling them to concentrate on more complex and strategic aspects of threat identification and mitigation. The ultimate goal is to improve the overall efficiency and effectiveness of threat-hunting efforts.

**Integrate with Existing SOC Tools:** Ensuring that the proposed solution integrates seamlessly with existing Security Operations Center (SOC) tools and workflows is a key objective of this research. The integration aims to minimize the learning curve for SOC analysts by aligning the new anomaly detection system with established SOC practices. By facilitating smooth adoption and integration, the solution will enhance the overall functionality of SOC operations and ensure that analysts can leverage the new tool without significant disruptions to their current workflows.

## 1.4 Relevance to National Needs

Following are the objectives for this research

**National Defense and Intelligence:** National defense and intelligence agencies rely heavily on secure and resilient communication networks to protect sensitive operations and data from adversaries. The detection of anomalies in web traffic is crucial, as such anomalies can serve as indicators of potential cyber threats targeting these critical networks. This research contributes to enhancing the security of national defense and intelligence systems by providing a robust method for identifying anomalous activities in HTTP logs. By implementing the proposed anomaly detection framework, agencies can maintain the integrity and confidentiality of their communication channels, thereby strengthening their overall defensive posture against cyber-attacks.

**Protection of Sensitive Information:** Government entities manage extensive volumes of sensitive and classified information, which are prime targets for unauthorized access and malicious activities. Anomalies observed in HTTP logs may signal early warnings of such security breaches. This research offers a proactive approach to safeguarding sensitive data by enabling the detection of potential threats before they manifest into more severe incidents. By leveraging machine learning techniques to analyze HTTP traffic, this study aids in protecting information from espionage, unauthorized access, and data manipulation, thereby ensuring that critical information remains secure and intact.

**Mitigating Cyber Espionage:** Cyber espionage poses a significant threat to national security, often perpetrated by state-sponsored actors seeking to acquire valuable and confidential information. The detection of anomalies within HTTP logs is essential for identifying patterns indicative of espionage activities. This research addresses this challenge by providing tools and methodologies for uncovering potential spy operations through detailed analysis of web traffic anomalies. By effectively recognizing and addressing these anomalies, the proposed approach contributes to mitigating cyber espionage threats, both domestically and internationally, enhancing

national security efforts against sophisticated adversarial tactics.

**Data Breach Prevention:** Data breaches can have far-reaching consequences across both public and private sectors, impacting organizational integrity, public trust, and national security. The research focuses on preventing such breaches by detecting anomalies in real-time and utilizing historical data to anticipate and counteract potential threats. The framework developed in this study enhances the capability to identify and address data breaches before they escalate into significant incidents. By implementing this anomaly detection approach, organizations can improve their preparedness and response strategies, thus reducing the risk of data breaches and minimizing their potential impact on national and organizational security.

## 1.5 Contributions

This research introduces a method for detecting anomalies in HTTP logs using advanced machine learning techniques, aimed at identifying unusual traffic patterns efficiently. The core contribution of this research lies in its ability to swiftly detect anomalies, which enables rapid responses to potential threats. By integrating this method with Security Information and Event Management (SIEM) solutions, the research enhances the visibility and quick detection of network anomalies, thereby providing a robust defense mechanism against emerging threats.

One of the key advantages of the proposed solution is its speed in anomaly detection. This capability ensures that potential threats can be identified quickly, allowing for timely responses and mitigating risks associated with network anomalies. Unlike traditional approaches that rely heavily on predefined datasets, the proposed solution operates without the need for such datasets. Instead, it utilizes the Isolation Forest algorithm to detect anomalies by training on live data logs collected over a specified period. This approach significantly reduces reliance on historical datasets and improves the system's ability to handle novel and evolving attack vectors, offering a stronger defense against new and emerging threats.

Following the identification of anomalies, the solution employs K-means clustering to categorize the detected anomalies. This clustering process provides a deeper understanding of the nature of the anomalies, offering valuable insights that aid in addressing potential security issues more effectively. By grouping similar anomalies together, the system helps in analyzing patterns and trends, which enhances the overall security posture and response strategies.

Moreover, the integration with SIEM solutions facilitates the visualization of network traffic and other relevant variables. This feature is particularly beneficial for Security Operations Center (SOC) analysts and threat-hunting teams, as it provides them with comprehensive insights during monitoring and investigative processes. The platform's compatibility with SIEM solutions ensures that network traffic is displayed in an accessible manner, aiding in real-time analysis and decision-making.

The solution is designed with scalability and flexibility in mind, making it suitable for various network sizes and configurations. It is specifically tailored for SOC analysts and threat-hunting teams, aiming to improve their efficiency in detecting and responding to multiple intrusion attempts. By enhancing visualization and reporting capabilities, the platform supports these teams in managing and mitigating security threats more effectively.

In addition to the anomaly detection solution, this research also proposes a method for detecting malicious executable files through a thorough analysis of Portable Executable (PE) files using a Random Forest (RF) classifier. This machine learning model demonstrates exceptional accuracy, achieving 99.45% in distinguishing between malware and benign files. The dataset utilized for this classification consists of 70% malware and 30% benign files. Deployed on the network side, this solution detects malicious PE files and forwards the detection details to the SIEM solution. This integration supports SOC analysts and threat-hunting teams by providing precise and timely information regarding potential malware threats, further enhancing the security infrastructure.



# Chapter 2

## LITERATURE REVIEW

### 2.1 Overview

In the field of cybersecurity, detecting anomalies in network traffic is critical for identifying and mitigating threats. Recent research has focused on developing advanced machine learning and clustering techniques to enhance intrusion detection systems (IDS) and network anomaly detection systems (NADS). Various approaches have been explored, including Unsupervised Niche Clustering (UNC), which automatically identifies the number of clusters and employs a fuzzy membership function alongside Principal Component Analysis (PCA) to improve detection accuracy. Additionally, methods like the Unsupervised Network Intrusion Detection System (UNIDS) use a combination of clustering techniques to detect outlying flows in network traffic without relying on labeled data. Other studies have introduced real-time unsupervised NIDS that leverage algorithms like DBSCAN and employ specialized engines to detect botnets and other threats in high-speed networks. These methodologies have been evaluated using datasets such as KDD Cup 1999, DARPA, and ISCX, demonstrating high detection rates and low false positive rates.

Across various studies, machine learning techniques such as Isolation Forests, Random Forests, and hybrid algorithms have been applied to network traffic data, often yielding impressive results. However, challenges remain, particularly regarding the outdated nature of commonly used

datasets like KDD99 and NSL-KDD. Despite these challenges, researchers continue to refine anomaly detection methods, incorporating deep learning approaches, feature extraction techniques, and hybrid models to improve the accuracy and efficiency of intrusion detection systems.

This body of research highlights the ongoing efforts to enhance the detection of anomalies and potential threats in network traffic. By employing a diverse range of machine learning algorithms, clustering methods, and feature selection techniques, these studies contribute to the development of more robust and effective IDS and NADS. Although the reliance on outdated datasets presents a limitation, the continuous evolution of these approaches shows promise in adapting to the ever-changing landscape of cybersecurity threats.

## 2.2 Related Work

An innovative approach to identifying outliers using Unsupervised Niche Clustering (UNC) has been presented in [14], which is mainly used for identifying alien presence in computer networks. The UNC is an automatic genetic niching procedure for clustering that automatically identifies the number of clusters. The authors use a fuzzy membership function to characterize each predicted cluster. Additionally, they employ principal component analysis (PCA) to simplify the dataset and improve the performance of the suggested method while refining the Maximal Density Estimator (MDE) to refine the quality of the solution. The knowledge discovery in databases (KDD) Cup 1999 dataset was used to test the model, and the results show a 99.2% detection rate and a 2.2% false alarm rate. [15] presented an Unsupervised Network Intrusion Detection System (UNIDS) have the ability to detect undisclosed network attacks without using any type of signatures, labeled traffic, or training. There are three sequential steps in UNIDS. It first applies a standard change-detection algorithm to three basic and traditional volume metrics (bytes, packets, and flows per time slot) to find an unusual interval for the clustering examination. Second, in order to detect outlying flows,

it employs a strong multi-clustering algorithm that combines Sub-Space Clustering (SSC), Density-based Clustering, and Evidence Accumulation Clustering (EAC) approaches. Ultimately, the anomalies can be identified using a predetermined threshold. They evaluate the system on the dataset which is KDD Cup 1999 where the recognition rate is more than 90%, and the false positive rate is less than 1%. [16] presented a real-time unsupervised NIDS for speedy networks. Two well defined engines are added in this NIDS to enable network intrusion detection. The first engine uses DBSCAN with an automated self-adaptive threshold to continuously monitor network behavior and identify intrusions. The internal botnet is to be found by the second engine. To identify both centralized and decentralized botnet C&C communication, it clusters particular network features. The suggested method was assessed using two datasets: DARPA and ISCX [17]. The accuracy of the suggested model was 98.39%, with a false positive rate of 3.61%. For the unsupervised Network Anomaly Detection Systems (NADSs), [18] provides a linear metric learning approach that uses distance-based clustering or classification techniques. The training and testing stages are included in the unsupervised clustering-based NADS. There are five steps in the training phase: boundary estimation, transformation, filtering, clustering, and metric learning. They perform the boundary estimation using a one-class SVM. Additionally, there are three steps in the testing phase: transformation, division, and classification. Using the Kyoto 2006+ [11] and NSL-KDD [12] datasets, they assess their NADS. The algorithms can perform better with metric learning. To find anomalies and potential attacks, [19] network traffic-based anomaly detection model, isolation forest, was employed. The anomaly score was used to assess the outcomes. Furthermore, the unsupervised machine learning algorithm known as Isolation Forest was trained using the KDD data set. [20] presented the hybrid ML algorithms,  $K$ -means algorithms, and Random Forest algorithms, which produce good results and where the accuracy level is ideal for anomaly detection. This idea is good for fixed data sets, not for real-time data. [21] overcome its tree rules NIDS's visibility problems. Based on a decision tree, the pruning algorithms are

changed. This framework's first benefit is that privacy is preserved by carefully choosing which rules are crucial, and its second benefit is that minor adjustments have an impact on system performance rather than the process selection approach. [22] explains the impacts of autoencoder-based feature learning on network data performance are the subject of this research. Three machine learning techniques, PCA, variational autoencoder, and autoencoder, are applied in this work. It offers high-dimensional data and a decent outcome. However, employing three distinct machine-learning approaches adds to the system's complexity. The work [23] covered outlier detection in unstructured data and illustrated it with a graph. Although this study primarily focuses on graph-based approaches, it also surveys dynamic, static, and machine-learning procedures for anomaly detection. Its drawback is the utilization of several graph scenarios, which adds complexity to the system. A research study [24] has proposed an approach of deep learning for developing an IDS that is dependent on recurrent neural networks that possess strong modeling powers in intrusion detection with high precision rates. The training data used was a popularly utilized data set termed NSL-KDD, which is a modified and polished copy of the earlier KDD99 data set. Besides [25] study, other referenced works used a deep learning-based approach with the aid of autoencoders and random forest algorithm combination for intrusion detection with both KDD99 and NSL-KDD data sets involved in the training process. [26] study built a fast-learning network for intrusion detection based on particle swarm optimization, and this was validated and tested using the KDD99 dataset [26]. [27] study also validated using KDD99 and NSL-KDD datasets. The detection method introduced in [13] involved changing the original Random Forest algorithm and its relationship to Information Gain for the feature vector. After that, the files in the dataset are PE files only, so feature extraction is relatively easy to some extent. This means that in self-assembled cells, the accuracy was 97%, and the end outcome was a 0.03 false-positive (FP) rate. [28] provided potential extraction procedures based on specified PE headers, DLLs (Dynamic-link Libraries), and API (Application Programming Interface) functions. They previously

developed a static malware detection procedure based on information gain and Principal Component Analysis (PCA) with three classifiers: SVM, J48 Decision Tree, and Naive Bayes. In the end, the J48 algorithm got the highest accuracy with 99% accuracy from the researchers. 17% with PE header feature, where 88.72% had only hybrid PE header and API functions feature type and 10.92% with only hybrid PE header and API functions feature. 1% with API functions. [29] introduced different methods for selecting features and explained the significance of n-grams in detecting static malware. These methods aim to minimize the computation time required by extracting characteristics of unknown malicious software using n-gram features. The most effective approach involved combining PCA and SVM; it employed only a few attributes instead of all other classifier models. [30] perform optimal k-means clustering concerning malware size and build the executable. Promising features were classified into different groups and used as training data for five classifiers, which could recognize unknown malware, were expressed. At last, this approach detected unknown malware with 99.11% accuracy.

[31] presented an ML method that can either show if a sample is clean or infected with high precision, and little computation effort was introduced. Their technique produced an integrated feature from the PE header fields raw values and derived values. They chose six different classifiers in their approach. Later, they compared how these two classifiers perform when raw features and the suggested combined one are used. Using 10-fold cross-validation, the random forest classifier gains an accuracy of 98.4%. [32] suggested identifying malware by examining the characteristics of network traffic like port numbers and protocols, from which they seized a collection comprising thirty-five characteristics to represent an average example containing the desired property. They also experimented with various methods such as SVM, Convolutional Neural Networks (CNN), and Multilayer Perceptrons (MLPs), among which some worked better than others. In particular, CNN or RF with CNN or RF having an accuracy rate greater than 85% yielded the best results. [33] proposed yet further solution that attained a very high detection rate, averaging 98.8%, coupled

with an incredibly low false positive rate, in which they utilize both artificial immune systems and deep learning technologies. [34] proposing the employment of LightGBM (Light Gradient-Boosting Machine), one of the gradient-boosting decision tree models in forecasting the Bodmas dataset and examining the findings. LightGBM demonstrated an accuracy level greater than 98% on only top-2 and top-3 PE file malware families, which are already known, whereas, for all other algorithms employed, there was a lower accuracy value compared to what was obtained with LightGBM regarding results. [35] have been involved in designing ML techniques useful for detecting previously unseen malware. Moreover, several models used in this study were cross-validated after performing feature extraction and feature selection procedures with a random forest method. The researchers employed many classifiers, including KNN, SVM, and others, which were known to work well even though the decision tree had higher accuracy among them. [36] presents an approach to identify harmful executables by meticulously examining the PE files. It utilizes the static analysis procedure to acquire features from Portable Executable files. In classifying the malware, various supervised learning algorithms utilize these characteristics. [37] curated the dataset called Ransomary, which comprises 2871 ransom and 4208 benign PE files and is intended to enable researchers to build their algorithms that can detect Ransomary quickly and accurately. The authors then investigated feature extraction and raw data approaches in static analysis using the Ransomary dataset. Effective feature selection in EMBER, DeepDetectNet, and Ransomary datasets through the LightGBM model produced greater than 0.99 AUC. Lastly, it was shown that the finest model achieved an AUC (Area Under the Curve) of 1 in EMBER and DeepDetectNet. [38] aimed at researching the effectiveness of hybrid machine learning algorithms in detecting malware PE Files. The voting classifier and LightGBM, XGBoost (eXtreme Gradient Boosting), and Logistic Regression are the base models of this research. The research demonstrates that these hybrid ML algorithms perform better than the ensemble method LightGBM regarding recall. This algorithm has a recall rate of 99.5026% compared to LightGBM,

with a recall rate of 99.4480%. [39] proposed a new method of classifying malware from large PE files with deep neural decision trees by combining neural nets with decision trees for a hybrid approach where each node is essentially a neural net that has been trained to recognize only one binary output kind as is done in a decision tree. The data set used has 7196 benign and 16698 malicious files being analyzed to classify 14 features taken from the headers of PE files. 0.88 Precision and 0.32 Recall are recorded, the Matthew Coefficient Correlation (MCC) stands at 0.302 While the AUC of Receiving Operating Characteristic (ROC) indicates 0.63 AUC value, and the average precision score reflects 0.69 is thus used for evaluating the classifier. It is evident from the outcome that the binary classifier differentiates between two categories: malware and benign.

The main issues with these datasets include the existence of a huge number of redundant and identical records. Also, many of these datasets were built over a decade ago, meaning they do not include instances of modern network attacks [40]. As a result, the datasets used in the literature review are largely outdated. While using a newer dataset might offer some short-term improvements, it will eventually become outdated and unable to detect new anomalous traffic patterns. Below is a Table 2.1 listing the datasets and their creation dates used in the above literature review:

**Table 2.1.** Overview of Datasets Used in the Study

<b>Dataset</b>	<b>Date</b>
KDD Cup 1999	1999
DARPA	1999
ISCX	2012
Kyoto 2006+	2006
NSL-KDD	1999

## 2.3 Summary

Recent advancements in network anomaly detection and intrusion detection systems (IDS) have increasingly leveraged machine learning and clustering

techniques to enhance threat detection capabilities. The exploration of methods such as Unsupervised Niche Clustering (UNC) and Unsupervised Network Intrusion Detection Systems (UNIDS) has demonstrated notable progress in identifying anomalous activities. Additionally, real-time network intrusion detection systems (NIDS) utilizing algorithms like DBSCAN have been rigorously evaluated using established datasets, including KDD Cup 1999, DARPA, and ISCX. These evaluations have consistently shown high detection rates coupled with low false positive rates, indicating the effectiveness of these approaches in recognizing and responding to network threats.

Despite these promising outcomes, a significant challenge persists within the field. Many studies and systems rely on outdated datasets that may not fully capture the spectrum of contemporary network threats. As cyber threats continue to evolve, there is a growing need for datasets that accurately reflect current attack vectors and tactics.

Moreover, the landscape of network anomaly detection is continually evolving with the development of advanced methodologies. Recent advancements in deep learning techniques and the integration of hybrid models offer new avenues for enhancing detection capabilities. Deep learning approaches, with their ability to learn complex patterns and features from large volumes of data, hold the potential to significantly improve detection accuracy and adaptability. Hybrid models, which combine various machine learning techniques, are also being explored to leverage the strengths of different approaches and address their limitations.

In conclusion, while current methodologies and technologies in network anomaly detection and IDS have shown commendable performance, ongoing research and development are crucial to keep pace with the evolving threat landscape. Future work will likely benefit from incorporating more up-to-date datasets and exploring innovative approaches such as deep learning and hybrid models to further improve detection capabilities and effectively combat emerging network threats.



## Chapter 3

# PROPOSED RESEARCH METHODOLOGY

In this chapter, we explain the proposed solution framework for anomaly detection in HTTP logs. We start with the discussion of the dynamic dataset generation process. Next, we will discuss that how machine learning algorithms are used to recognize and cluster anomalies within the logs. We also figure out the integration of anomaly detection with the existing SIEM solutions, focusing on its role in strengthening the network security. In the end, the section provides an overview of the system design and the strategies implemented to ensure scalability and reliability.

### 3.1 Dataset

A key advantage of the proposed solution is its ability to generate specific datasets in real-time, tailored to user-defined timeframes, rather than relying on static, pre-defined datasets. Traditional anomaly detection systems often depend on static datasets that may not reflect the current network activity, leading to potential gaps in detection accuracy and relevance. In contrast, the proposed solution dynamically gathers HTTP logs based on the exact timestamps specified by the user.

When a user selects a particular timeframe, the solution automatically retrieves the HTTP logs from that period, creating a customized dataset directly relevant to the specified timeframe. This approach ensures that

the dataset used for analysis mirrors the actual network traffic during that period, allowing for training and evaluation that aligns closely with the network conditions and activities of that specific moment.

The real-time dataset generation process enhances the precision and relevance of anomaly detection. By working with the most current and contextually appropriate data, the solution addresses the limitations of conventional methods that may rely on outdated or generic information. This ensures that the machine learning models are continuously exposed to fresh, relevant data, improving their accuracy and effectiveness in detecting anomalies.

The solution leverages advanced machine learning algorithms, specifically Isolation Forest and K-means clustering, to identify anomalies and potential security threats within these customized datasets. Isolation Forest is employed to detect anomalies by isolating observations in the data, while K-means clustering groups similar data points to highlight deviations from typical patterns. By focusing specifically on logs from user-defined timeframes, the solution trains its models on the network traffic occurring during that period, enhancing its ability to recognize unusual patterns, potential intrusions, or other security issues as they emerge.

The adaptability of this procedure is a significant benefit. As network conditions evolve and new threats emerge, the solution remains responsive and effective by continuously updating its training with fresh data. This ongoing adaptation ensures that the solution remains capable of identifying and mitigating security threats in an ever-changing landscape. The dynamic nature of this approach makes it particularly well-suited for modern, dynamic network environments where conditions and threats can shift rapidly. By providing real-time adaptability and continuous updates, the proposed solution offers a robust and resilient tool for modern network security management.

**Table 3.1.** Key Features and their description

Feature	Description
Dynamic Dataset Generation	Automatically generates datasets based on user-defined timestamps, ensuring real-time, context-specific data analysis.
User-Defined Timeframes	Users specify the timeframe for analysis, allowing for tailored data collection from relevant HTTP logs.
On-the-Fly Data Collection	Collects HTTP logs from the specified period, creating datasets that reflect current network activity.
Precision in Anomaly Detection	By focusing on relevant, up-to-date data, the solution enhances the precision of identifying anomalies.
Machine Learning Algorithms	Utilizes Isolation Forest and K-means clustering to detect anomalies and potential security threats in the logs.
Adaptability	Continuously adapts to changing network conditions and evolving threats by training on fresh data.
Improved Network Scanning	Offers more accurate network scanning by analyzing data that reflects real-time conditions.

### 3.2 Working of Isolation Forest and $K$ -means in Proposed Solution

Anomaly detection in data often uses this method called the Isolation Forest algorithm. It is based on an assumption that outliers are few and very different from other points which make it easier to separate them[41, 42]. The Isolation Forest algorithm operates by creating numerous decision trees from random subsets of data. Each tree is constructed by recursively splitting the data into smaller partitions. The core principle behind this algorithm is that anomalies, which are inherently different from normal data points, require fewer splits to be isolated. In contrast, normal data points, which are more similar to each other, are found deeper within the tree structure and thus require more splits. As a result, anomalies tend to be isolated closer to the root of the tree, leading to shorter path lengths compared to normal points. The algorithm averages the path lengths across all trees for a given data point; a shorter average path length typically indicates that the point is an anomaly.

An important consideration in implementing Isolation Forest is the selection of the number of trees to be created. The accuracy of the anomaly detection can be significantly influenced by this parameter. Too few trees may not capture the data's nuances adequately, while too many trees might lead to diminishing returns in accuracy and increased computational cost.

While Isolation Forest is highly effective in detecting anomalies, it does not inherently group these anomalies into meaningful clusters. This is where the K-means clustering algorithm complements it. K-means clustering divides the data into a predefined number of clusters,  $k$ , by assigning each data point to the cluster whose centroid is closest. This approach ensures that data points within the same cluster are similar to each other, while points in different clusters are more distinct. After the Isolation Forest identifies anomalies in HTTP traffic data, K-means clustering can be employed to group these anomalies based on their similarity. This grouping process facilitates the identification of patterns, such as anomalies that originate from the same IP range or involve similar types of HTTP requests.

The benefit of clustering anomalies for SOC analysts is substantial. Instead of sifting through a long list of individual anomalies, analysts can focus on clusters of related anomalies. This clustering helps in recognizing trends, understanding the nature of outliers, and taking appropriate actions. For instance, if a cluster of anomalies shows a pattern related to a particular type of HTTP request or a specific IP range, it can indicate a potential security threat or an ongoing attack that requires targeted investigation and response.

In summary, the proposed solution leverages the strengths of Isolation Forest for its effective and precise anomaly detection capabilities. Once anomalies are detected, K-means clustering is applied to group these anomalies into clusters, making it easier for SOC analysts to investigate and address potential security threats by focusing on clusters of similar incidents rather than isolated data points. This combined approach enhances the efficiency and accuracy of anomaly detection and response in the context of HTTP traffic data.

**Table 3.2.** Summary of Key Points: Isolation Forest vs. K-means Clustering

Aspect	Isolation Forest	K-means Clustering
Purpose	Detects anomalies in data	Groups detected anomalies into clusters
Underlying Principle	Anomalies are rare and distinct, making them easier to isolate	Data points within the same cluster are similar, and different clusters are distinct
Method	Builds multiple decision trees by randomly splitting the data	Divides data into $k$ predefined clusters based on proximity to centroids
Key Feature	Anomalies are isolated near the root of the tree, resulting in short path lengths	Each data point is assigned to the nearest cluster, ensuring similar points are grouped together
Calculation	Average path length across trees determines anomaly status	Centroids represent the mean value of each cluster
Outcome	Short path lengths indicate anomalies	Clusters of anomalies highlight patterns or shared characteristics
Use Case	Initial detection of anomalies in HTTP traffic	Grouping anomalies for easier analysis by SOC analysts
Advantages	Efficient and accurate in identifying rare and distinct data points	Simplifies analysis by organizing anomalies into meaningful groups
Parameter Selection	Number of trees affects accuracy	Predefined $k$ value determines the number of clusters

### 3.3 Integration of Anomaly Detection with SIEM

The proposed solution in this thesis, anomaly detection, is effective in identifying unusual patterns in network traffic. However, integrating this anomaly detection framework with a SIEM system enhances its functionality and provides a more comprehensive security solution. SIEM systems play a crucial role in aggregating, analyzing, and visualizing security-related data from various sources, offering valuable insights and facilitating rapid incident response.

There are numerous SIEM solutions available on the market, each with

its own features and capabilities. These solutions can be broadly categorized into open-source and commercial options. Open-source SIEM solutions offer flexibility and cost-effectiveness, while commercial solutions often provide advanced features and dedicated support. Notable examples include Wazuh, FortiSIEM, and Splunk SIEM, among others.

For this thesis, we opted to utilize the ELK stack for building our SIEM solution. The ELK stack, an acronym for Elasticsearch, Logstash, and Kibana, is a powerful set of tools designed to handle large volumes of log data and provide real-time analysis and visualization capabilities. Although the ELK stack itself is not a complete SIEM solution, it serves as a foundational component for constructing one.

In our implementation, we used Filebeat to forward HTTP logs and logs of detected anomalous clusters to Elasticsearch. Filebeat is a lightweight shipper for forwarding and centralizing log data, ensuring that logs are transmitted efficiently and reliably. Once the data reaches Elasticsearch, it is indexed and stored, making it readily accessible for search and analysis.

Kibana, the visualization component of the ELK stack, is used to create interactive dashboards and visualizations of the data. By visualizing HTTP logs and anomalous cluster logs in Kibana, we can gain insights into network behavior, identify trends, and detect potential security incidents. This integration not only enhances the anomaly detection framework but also provides a user-friendly interface for monitoring and investigating network anomalies.

Overall, the integration of anomaly detection with the ELK stack SIEM solution significantly enriches the analysis and response capabilities, offering a robust platform for managing and securing network traffic.

### **3.4 Features Extraction from HTTP logs**

We used Zeek to get the HTTP traffic. The Zeek can be used to log all HTTP traffic from your network to the http.log file. This file can then be used for analysis and auditing purposes. when we are getting the HTTP traffic, Zeek gives around twelve features, but we only extract five features

**Table 3.3.** Integration of Anomaly Detection with SIEM Solution Using ELK Stack

Step	Description
Select SIEM Solution	Choose a SIEM solution, such as ELK Stack (Elasticsearch, Logstash, Kibana) for integration.
Use Filebeat to Send Logs	Configure Filebeat to send HTTP logs and anomalous cluster logs to Elasticsearch.
Store Logs in Elasticsearch	Logs are stored in Elasticsearch for indexing and searching.
Visualize Logs in Kibana	Use Kibana to create visualizations and dashboards for the logs and detected anomalies.

for our use, and they are `ts`, `id.resp_p`, `method`, `resp_mime_types`, `request_body_len`. Below is the table that shows the extracted features:

**Table 3.4.** Extracted Features from Zeek HTTP Logs

Feature Name	Description
<code>ts</code>	Timestamp of the HTTP request.
<code>id.resp_p</code>	Response port of the HTTP request.
<code>method</code>	HTTP method used (e.g., GET, POST).
<code>resp_mime_types</code>	MIME types of the response.
<code>request_body_len</code>	Length of the HTTP request body.

### 3.5 Scaling Strategies for the proposed solution

In our proposed solution, as displayed in Fig 3.2, we leverage Apache Kafka’s robust capabilities to efficiently manage increasing network traffic. Apache Kafka excels at handling large volumes of data through its distributed architecture. By partitioning topics, Kafka can spread data across multiple servers, allowing for parallel processing of data streams. This partitioning is key to achieving high throughput and low latency, as it enables producers to push data to different partitions simultaneously while consumers can read from them concurrently.

Kafka’s design ensures that each partition can be replicated across multiple brokers. This replication mechanism is crucial for maintaining data durability and availability. When data is replicated, each piece is stored on several brokers, which protects against potential server failures. If one broker becomes unavailable, the system can continue to function using the replicated data from other brokers, thus enhancing the resilience and robustness of the data processing pipeline.

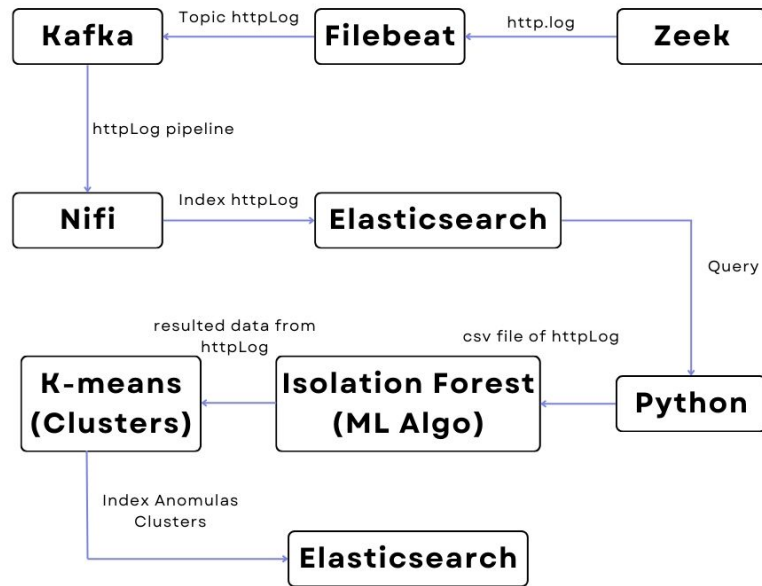
The distributed nature of Kafka not only supports high efficiency and reliability but also provides scalability. As network traffic grows, additional brokers can be added to the Kafka cluster, allowing it to handle even larger volumes of data without a significant impact on performance. This scalable architecture ensures that our solution remains effective and responsive as the data load increases, making it a powerful tool for managing and processing large-scale data streams in real time.

### **3.6 System Design**

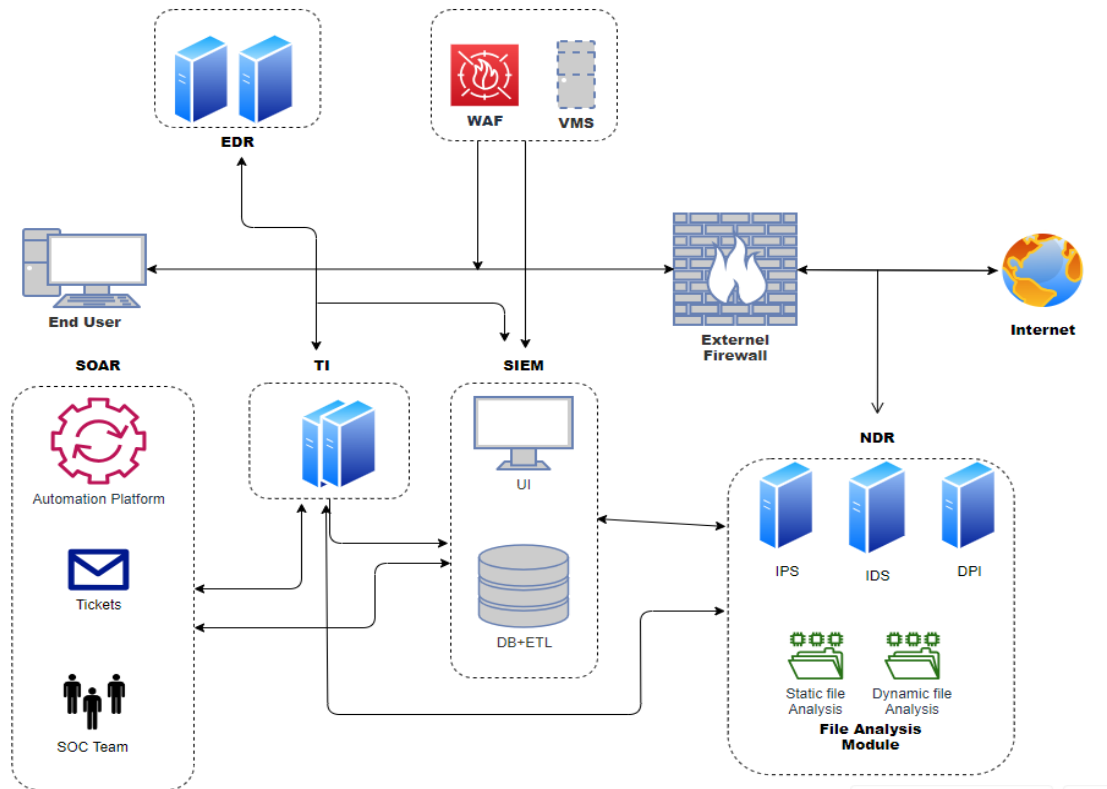
Uses of a security information and event management system (SIEM) include risk scoring, threat detection and response, and security monitoring. SIEM gathers security event data from application, network, and endpoint sources.

The Fig. 3.1(b) shows the overall network of the SIEM, which contains many modules like EDR, NDR, SIEM, SOAR, and TI, but we will only focus on the Network Detection and Response (NDR). Because our solution will be deployed in that module where the whole network traffic is passed, and we need that network traffic, that’s why we deployed the solution in that space. We explain Fig 3.1(b) that how the whole network works. We start with SIEM, and then we discuss each module. Security Information and Event Management System (SIEM) is a solution used for the gathering of security information from various components belonging to the information system and presents it as actionable information through a single interface. SIEM collects data from servers, networks, and devices and then transforms, aggregates and correlates the data for potential threats. After these





((a)) Flowchart of the System



((b)) Security Information and Event Management (SIEM) Network Diagram

Figure 3.1: Flowchart of the Solution and the SIEM Network Diagram

**Table 3.5.** Apache Kafka’s Data Management and Processing Workflow

Step	Description
Data Ingestion	Apache Kafka efficiently manages rising network traffic by handling large volumes of data via a distributed structure.
Produce Data to Topics	Producers send data to different topics within Kafka.
Topics Split into Partitions	Each topic is divided into multiple partitions for parallel processing.
Distribute Partitions Across Servers	Partitions are spread across multiple servers to enable parallel data processing.
Parallel Data Processing	Enables simultaneous processing of data from different partitions to achieve high efficiency.
Data Replication to Multiple Brokers	Data is replicated across several brokers to ensure durability and availability.
Ensure Data Durability and Availability	Replication ensures that data remains accessible and robust even in the event of server failures.
Handle Server Failures	Replication mechanism provides flexibility and reliability against server failures.

processes, the next step is to discover vulnerabilities and generate alerts and Reports. So when gathering the security logs from the different components, such as the network and different devices, the Network side, NDR, the Network Detection and Response solution, is used. The NDR focuses on detecting and responding to threats to the network. Integrating NDR with SIEM can achieve a more holistic and effective threat detection and response approach. NDR provides additional visibility into network traffic, also allowing SIEM solutions to detect threats that may otherwise go unnoticed. So, on the endpoint side, the EDR, which is Endpoint Detection and Response, helps the SIEM by giving detailed endpoint visibility, improving threat detection and response capabilities, and enabling detailed incident analysis. Integrating EDR with SIEM ensures a robust and coordinated defense against a wide range of threads. Also, on the SIEM correlation side,

SOAR strengthens the SIEM in detecting and correlating security events with the orchestration and automation capabilities of SOAR. SOAR provides an effective means of automating and managing all related events. The Threat Intelligence integration with SIEM solutions converts large amounts of unprocessed data into useful intelligence, improving the functionality of security operations and simplifying them while increasing productivity. Lastly, the Web Application Firewall (WAF) and Vulnerability Management System (VMS) have been integrated into the SIEM to provide safety for web applications employing filtering and monitoring HTTP traffic between them and the internet. Also, VMS helps identify, assess, prioritize, and fix vulnerabilities in systems, networks, applications, and software.

Fig 3.2 shows our system's design; means how our system works and that system will be the part of NDR in SIEM.4 we set up the Zeek, which is a traffic analyzer on the network, to get all the traffic. The Zeek can be used to log all HTTP traffic from your network to the http.log file. This file can then be used for analysis and auditing purposes. So, for getting these hypertext transfer protocol (HTTP) logs and sending them to Elasticsearch. Elasticsearch is a decentralized, open-source search as well as analytics system that can quickly save, search, and examine massive volumes of information. So, we used Filebeat, which is the data shipper, to send these HTTP logs to Kafka using a defined topic. We used Kafka for clustering, which is designed to manage massive processing and real-time data feeds. If the network traffic increases, the Kafka clusters we defined can handle this traffic. After receiving the traffic on the Kafka, we then send the traffic to the NiFi to parse the traffic. At that stage, we can also see how much traffic we received, and after parsing the traffic, we then send it to the Elasticsearch Index pattern. At that time, all the HTTP traffic is shown on the Kibana, and then we make a query request to that index pattern on Elasticsearch to get all the required data and save it in a CSV file, and give it to the machine learning model. In that, the Isolation Forest algorithm detects anomalous logs. After the detection of anomalous logs, the *K*-means create clusters from these anomalous logs, and these clusters data will be shown on the new index pattern on Elasticsearch.

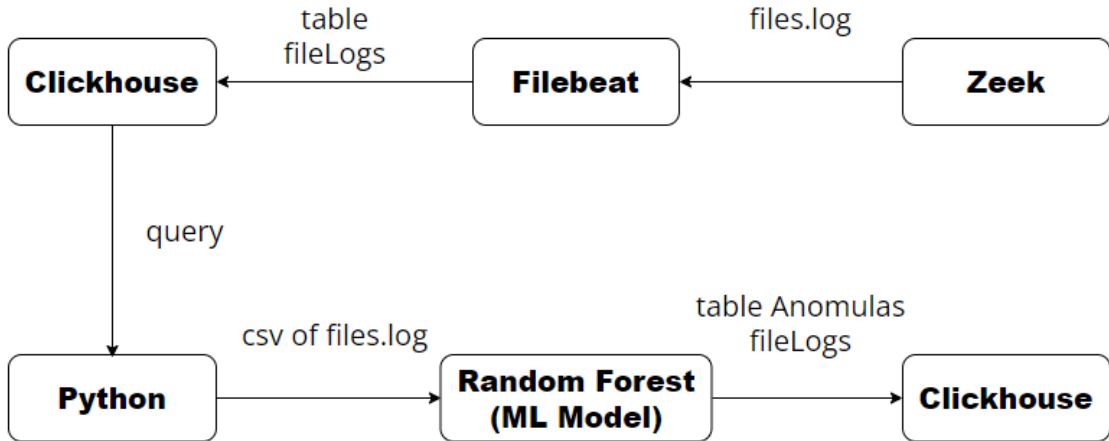


Figure 3.2: Flowchart of HTTP malicious files detection

Component	Description
SIEM	System equipped with EDR, SOAR, NDR, SIEM, and TI components; NDR is the focus where network traffic passes.
Zeek	Traffic analyzer that logs all HTTP files to <code>files.log</code> .
ClickHouse	Open-source column-oriented database for online analytical processing; stores HTTP file logs.
Filebeat	Data shipper used to send HTTP file logs to ClickHouse.
ML Model	Random Forest (RF) algorithm analyzes data for anomalies in PE files.
Output	Anomalies detected by RF model are recorded in the SIEM system with node IP, timestamp, file hash, etc.

Table 3.6. System Design and Data Flow

### 3.7 System Design of the HTTP malicious files detection

The Fig. 3.1(b) displays the entire system of the SIEM, which is equipped with various components such as EDR (Endpoint Detection and Response), SOAR (Security Orchestration, Automation, and Response), NDR (Network Detection and response), SIEM (Security Information and Event Management) and TI (Threat Intelligence), but we will only look into

Aspect	Details
Algorithm	Random Forest (RF)
Dataset Size	150,000 records
Dataset Distribution	70% malicious and 30% benign
Accuracy	99.45%
Output	Classification of files as either ‘Malicious’ or ‘not Malicious’
Log Details	Node IP, timestamp of file download, file hash value, and other relevant fields recorded in the SIEM system.

**Table 3.7.** Random Forest Model Details

the NDR because it is the module in which our system will be implemented where all network traffic passes through and since in this network traffic, we require HTTP (Hypertext Transfer Protocol) files, and therefore, is installed there.

Fig 3.2 shows our system’s design; we set up the Zeek, a traffic analyzer on the network, to get all the traffic. One of the Zeek properties is to log all the HTTP files from the network to the Zeek files.log file. This file can then be used for analysis and auditing purposes. So, to get these HTTP file logs, send them to the Clickhouse. A real-time analytical report on SQL would be generated by users using ClickHouse, an open-source column-oriented database management system made for online analytical processing. So, we used Filebeat, the data shipper, to send these HTTP file logs to the Clickhouse table. At that time, all the HTTP file logs are shown on the Clickhouse user interface, and then we make a query request to that table on Clickhouse to get all the required data with a required timestamp, save it in a CSV file, and give it to the ML model. There, the Random Forest (RF) algorithm detects the anomalous PE files. Additionally, the RF model is trained with the dataset containing 150,000 records of both malicious and benign PE files, with a distribution of 70% malicious and 30% benign records in the dataset. Also, the accuracy we got is 99.45%. When the model analyzes a file, it produces a result of either ‘Malicious’ or ‘not Malicious’. If identified as malicious, the system records such information on the SIEM system. This log contains fields such as node IP (Internet

Protocol) and timestamp of when the file was downloaded, along with the file’s hash value and any other relevant fields.

<b>Component</b>	<b>Description</b>
NDR (Network Detection and Response)	Module where the system is implemented, handling all network traffic including HTTP files.
Zeek	Traffic analyzer logging HTTP files to ‘zeek_files.log’.
Filebeat	Data shipper sending HTTP file logs to Clickhouse.
Clickhouse	Column-oriented database management system for real-time analytics; stores HTTP file logs.
ML Model (Random Forest)	Analyzes HTTP file logs to detect anomalies; trained with 150,000 records.

**Table 3.8.** System Overview and Data Flow

<b>Detail</b>	<b>Value</b>
Training Dataset Size	150,000 records
Malicious Records	70%
Benign Records	30%
Accuracy	99.45%
Outcome	‘Malicious’ or ‘Not Malicious’

**Table 3.9.** Random Forest Model Details

# Chapter 4

## RESULTS AND FINDINGS

### 4.1 Experimental Setup

The experimental setup for the anomaly detection system involves a detailed configuration of both hardware and software components to ensure efficient processing and analysis of HTTP logs.

On the hardware side, the system is equipped with a 40 GB hard disk to accommodate the storage needs of the data and logs. The memory capacity is set at 24 GB, which provides ample space for handling and processing large volumes of data. The computational power is supported by four processor cores, enabling the system to perform complex computations and handle multiple tasks simultaneously.

For the software setup, the system utilizes several key technologies. Elasticsearch and Kibana, both at version 7.16.3, are deployed using Docker containers. Elasticsearch serves as the core search and analytics engine, while Kibana provides the visualization layer, allowing for interactive exploration of the data. Apache Kafka, version 3.6.0, is also configured using Docker to facilitate real-time data streaming. Kafka's role is crucial in managing and processing the incoming data streams efficiently. Additionally, Apache NiFi, version 2.0.0, is employed to automate the flow of data between systems and ensure smooth data ingestion and processing.

For log shipping, Filebeat version 7.17.8 is used to forward and centralize log data. Filebeat is lightweight and efficient, making it suitable for shipping



((a)) HTTP Logs of Network



((b)) Anomalous HTTP Logs

Figure 4.1: HTTP Logs and Anomalous HTTP Logs



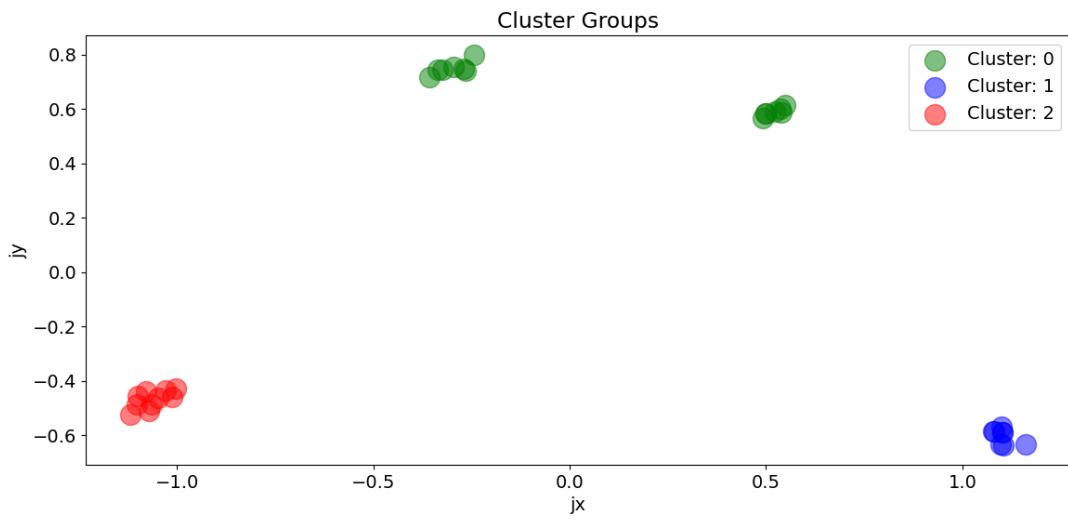


Figure 4.2: Cluster Groups of Anomalous HTTP Logs

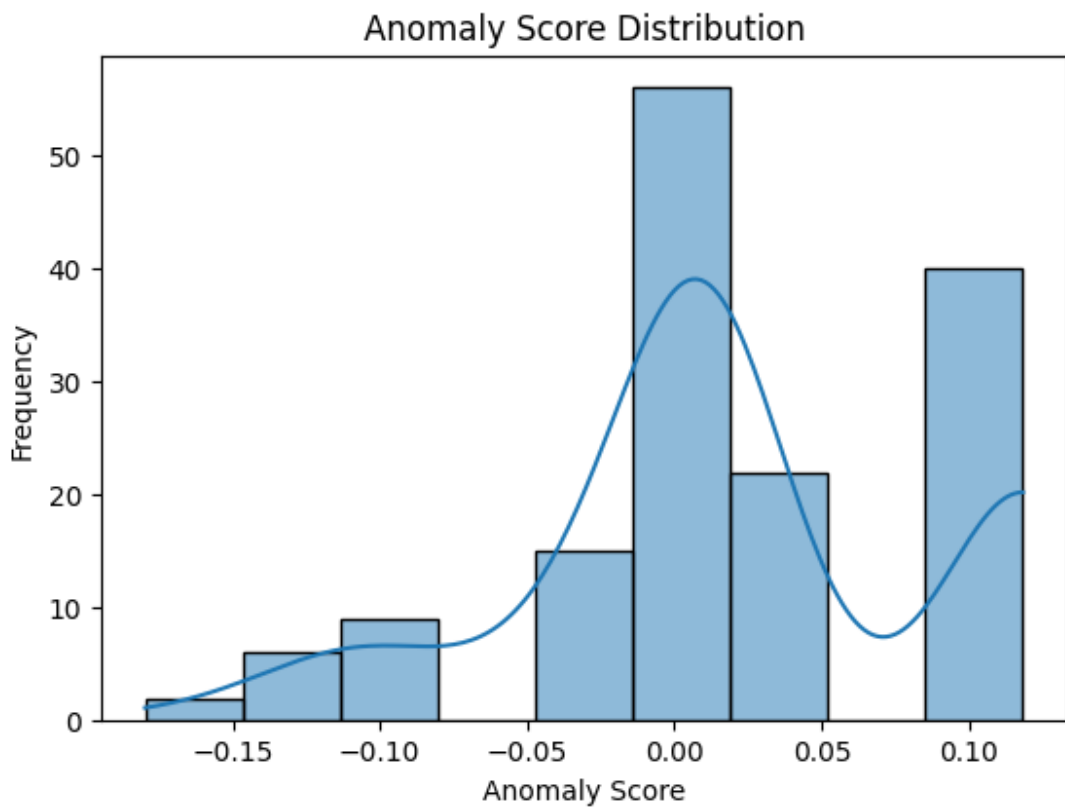


Figure 4.3: Decision Function of the Proposed Machine Learning Model)

logs to the central processing system. On the network traffic analysis front, Zeek version 6.0 is implemented to monitor and analyze network traffic. Zeek’s capabilities in network monitoring and security event detection are leveraged to gain insights into the traffic patterns and identify potential anomalies.

In terms of coding and development, Python 3.10 is the programming language of choice, providing a robust environment for writing and executing the anomaly detection algorithms. The code is developed using Visual Studio Code (VSCoDe) version 1.92, which offers a versatile and user-friendly interface for coding and debugging.

The performance of the system in detecting anomalies and clustering data is notable, with the process typically taking around 2 to 3 seconds. However, when the network traffic volume increases, the system’s capacity is scaled up by configuring three Kafka brokers to handle the increased load efficiently.

Overall, this experimental setup combines powerful hardware and a well-chosen software stack to deliver a robust solution for detecting anomalies and analyzing network traffic.

## **4.2 Visualization and Analysis of Anomalous HTTP Logs**

We begin with Fig. 4.1(a), which shows HTTP logs across the network on the Elasticsearch index pattern. The Index pattern contains benign and anomalous HTTP logs at a certain timestamp. It gives a comprehensive view of all HTTP activities happening around the network. These logs contains many features like timestamp, resource IP, resource port, destination IP, destination port, log file path and many other features. This Index pattern shows all the HTTP logs in a timestamp given by the user.

Next, Fig. 4.1(b) where the index pattern contains only anomalous HTTP logs. These anomalous logs with their cluster numbers and other relevant details are visually depicted in this figure. The cluster also categorize the

anomalous logs for better and deeper understanding of their nature. Also these clusters giving more insights into dealing with possible/imminent security problems. The clustering of anomalous HTTP logs detected using the Isolation Forest algorithm is shown in Fig. 4.2. The data points represent logs that were flagged as being anomalous, which appears in different color-coded according to the cluster it belongs to after applying K-means clustering. The axes  $j_x$  and  $j_y$  correlate with jittered principal components got from PCA, hence making it possible to distinguish overlapping points for clarity. These clusters enable one to identify various kinds of anomalous HTTP logs hence leading to better categorization and assessment. These details help the SOC analyst and threat-hunting team in the detection and investigation of security threats. By isolating and examining these anomalous logs, the threat hunting teams can effectively pinpoint malicious activities and take appropriate action.

The decision function in the Isolation Forest model is, which provides an anomaly score for each of the data samples. As shown in Fig. 4.3, the values of the anomalous score indicate where the individual data points lie with respect to the calculated anomalies. In the Isolation Forest algorithm, higher scores mean that there are actually normal data points, and lower scores (negative in most cases) signify anomalies.

In histogram, frequency refers to the number of data points which goes a long way in informing the extent of the present anomalies.

Also, these 4.3 and 4.4 shows the Key Performance Metrics for Anomaly Detection Models and also as well shows the Overview of Anomaly Detection Techniques. Also, the 4.4 4.5

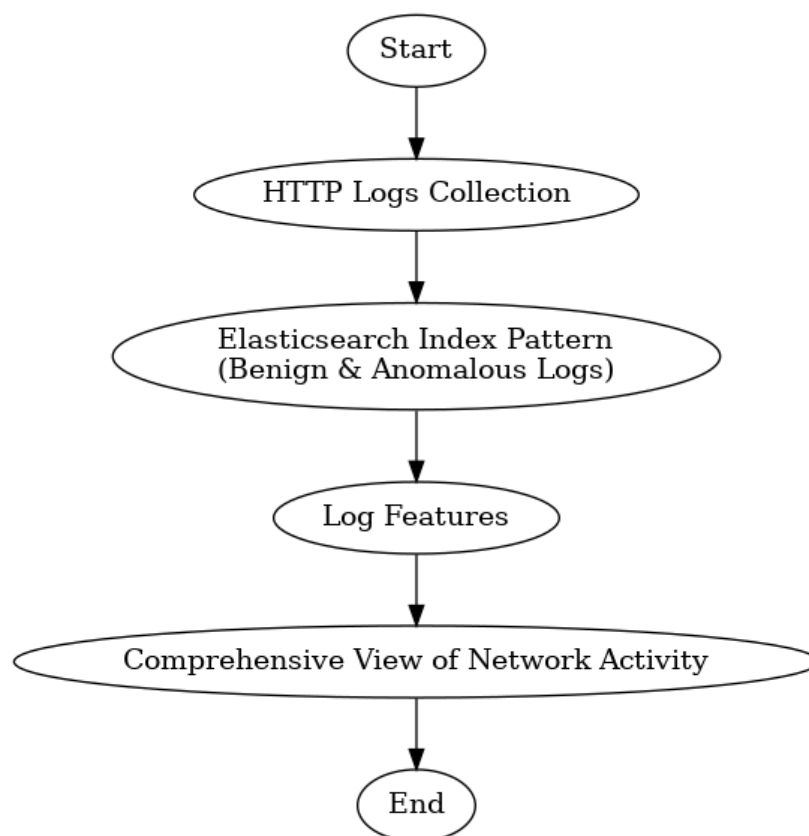


Figure 4.4: HTTP Logs Overview in Elasticsearch

## Tables

**Table 4.1.** Features of HTTP Logs in Elasticsearch Index Pattern

Feature	Description
<b>Timestamp</b>	The time at which the log was recorded.
<b>Resource IP</b>	IP address of the resource generating the log.
<b>Resource Port</b>	The port number used by the resource.
<b>Destination IP</b>	IP address to which the HTTP request is directed.
<b>Destination Port</b>	The port number used at the destination.
<b>Log File Path</b>	Path to the log file in the system.
<b>Other Features</b>	Additional attributes, such as HTTP method, status code, user agent, etc.

**Silhouette Score:** We obtain a 0.3803 silhouette score from our results. It's a metric used to analyze the standard of clustering results. It is used

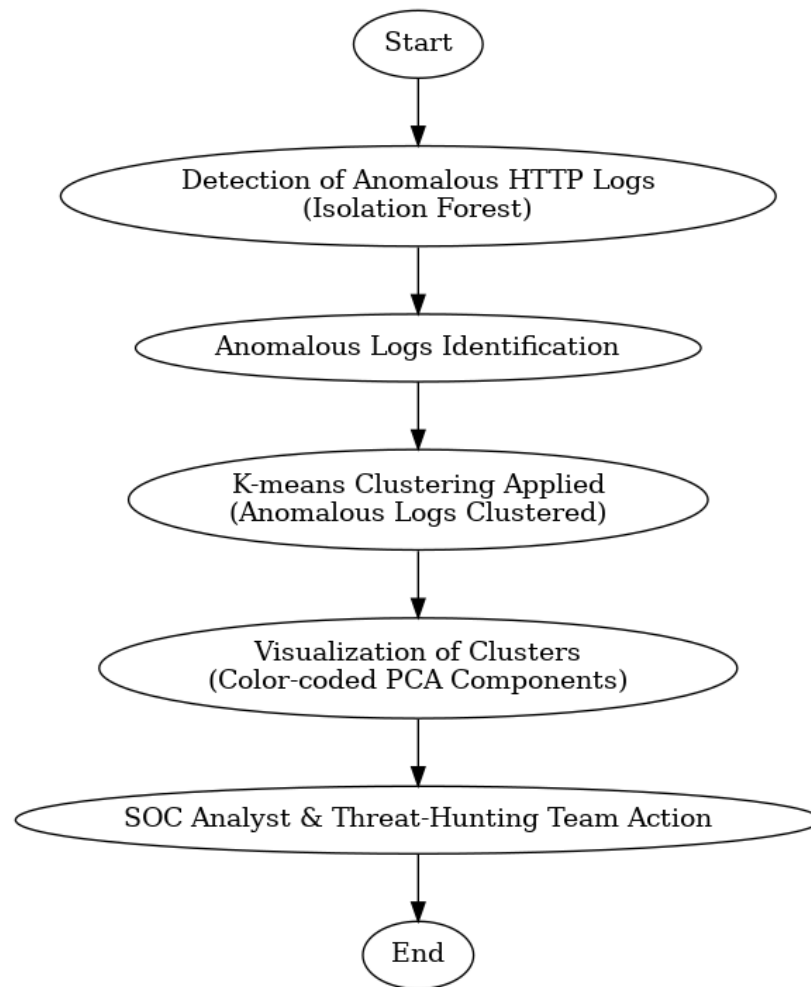


Figure 4.5: Anomalous HTTP Logs Clustering & Analysis

to decide how adjacent an object aligns with its own clusters as compared to other clusters. The silhouette score ranges from -1 to 1, where:

1: They provide information that shows that a data point is fairly balanced with its own cluster but badly matched with adjacent clusters.

0: This suggests that the data point lies on or very near to the line between two adjacent clusters.

**Negative values:** This implies that the specific content of the data point may have been assigned to the incorrect cluster.

### Why do we need Silhouette Score

**Table 4.2.** Key Performance Metrics & Anomaly Detection Techniques

Metric/Technique	Description
<b>Anomaly Score (Isolation Forest)</b>	A numerical score indicating the likelihood of a log being anomalous.
<b>Frequency in Histogram</b>	The count of data points that fall within certain anomaly score ranges, indicating the extent of anomalies.
<b>Cluster Categorization</b>	The division of anomalous logs into clusters based on K-means, aiding in the understanding of anomaly types.
<b>Principal Component Analysis (PCA)</b>	A dimensionality reduction technique used to differentiate overlapping data points in visualizations.
<b>SOC Analyst &amp; Threat Hunting</b>	The process of analyzing and investigating the anomalies to detect and mitigate potential security threats.

Metric	Definition	Importance	Calculation
<b>Precision</b>	Proportion of true positives among all positive detections	Accuracy in identifying threats	$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$
<b>Recall</b>	Proportion of true positives among all actual positives	Ability to detect all threats	$\frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$
<b>F1-Score</b>	Harmonic mean of precision and recall	Balance between precision and recall	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$
<b>FP Rate</b>	Proportion of false positives among all negative cases	Likelihood of false alarms	$\frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$
<b>Det Time</b>	Time taken to identify and respond to a threat	Crucial for real-time mitigation	Time of Detection – Time of Threat Occurrence

**Table 4.3.** Key Performance Metrics for Anomaly Detection Models

**Evaluate Cluster Quality:** A higher value of the Silhouette Score means data points are close to their cluster center and far from the other cluster center. It is nice in the case of anomaly detection since it tells you that there is a clear separation between 'normal' data and 'anomalies.'

**Validation of the Clustering Algorithm:** The Silhouette Score

Technique	Description	Advantages	Disadvantages
Signature-Based	Matches patterns against a database of known signatures to identify threats	High accuracy for known threats	Ineffective against zero-day attacks
Anomaly-Based	Detects deviations from normal behavior to identify potential threats	Can detect unknown threats	High false positive rate
ML Based	Uses ML models to classify behavior as normal or anomalous	Adaptive and scalable	Requires large amounts of labeled data
Heuristic-Based	Applies rule-based systems to identify suspicious activities	Flexible and customizable	Can miss sophisticated attacks

**Table 4.4.** Overview of Anomaly Detection Techniques

measures how well the clustering algorithm ( $K$ -means here) is performing in segmenting the data points with good signals.

Silhouette Score	Interpretation
1	Data point is well aligned with its own cluster and poorly aligned with other clusters.
0	Data point lies on or near the boundary between two neighboring clusters.
-1	Data point is poorly aligned with its own cluster and may be closer to another cluster.

**Table 4.5.** Silhouette Score Interpretation

Silhouette Score	Interpretation
1	Data point is well aligned with its own cluster and poorly aligned with other clusters.
0	Data point lies on or near the boundary between two neighboring clusters.
-1	Data point is poorly aligned with its own cluster and may be closer to another cluster.

**Table 4.6.** Silhouette Score Interpretation

### 4.3 Results and findings of HTTP Malicious Files Detection

The results and findings section presents all the outcomes from our analysis. This solution is deployed in the NDR module because Zeek directly gets all the benign and malicious HTTP files. After getting these files, our solution decides whether they are malicious and shows the results on Clickhouse. We start from Fig. 4.6(a), which shows all the logs of those HTTP files downloaded in the given timestamp. These logs show the overall details of each HTTP file that has been downloaded. The details contain a timestamp, fuid, uid, id origin host, id origin port, and all other relevant details.

The next Fig. 4.6(b) shows the detail of a malicious file that is detected and that shows on the separate table of clickhouse. The table containing malicious file details helps the SOC analyst and threat-hunting teams detect and investigate security threats.

The model was trained on a dataset of 150,000 records encompassing malicious and benign PE files, with a distribution of 70% malicious and 30% benign records. The model achieved an accuracy of 99.45%. To create a confusion matrix, we have to understand the confusion matrix and the classification results in terms of True Positives, True Negatives, False Positives, and False Negatives.

#### **Confusion Matrix:**

A confusion matrix can depict how effective a classification model is. True



ts	fuid	uid	id_orig_h	id_orig_p	id_resp_h	id_resp_p
2024-06-27 00:19:01	FIRGeFTlUtFu10wRa	CX503k3K59N3EW2eH9	192.168.10.205	37964	196.216.167.196	80
2024-06-27 00:19:01	FDz1cISj7g3Mz2W89	CplEdB3L2oo4GFex64	192.168.10.205	37970	196.216.167.196	80
2024-06-27 00:19:14	F3yakQdAU13tQkrNa	CQ1X1p191sddo6lB77	192.168.10.205	54144	196.216.167.196	80

((a)) HTTP Files of both benign and malicious Logs

Run (Ctrl/Cmd+Enter) ✓	
ts	2024-06-27 00:19:14
fuid	F3yakQdAU13tQkrNa
uid	CQ1X1p191sddo6lB77
id_orig_h	192.168.10.205
id_orig_p	54144
id_resp_h	196.216.167.196
id_resp_p	80
source	HTTP
depth	0
analyzers	["MD5", "SHA1", "EXTRACT", "PE"]
mime_type	application/x-dosexec
duration	4.37533
local_orig	0
is_orig	0
seen_bytes	2228534
total_bytes	2228534
missing_bytes	0
overflow_bytes	0
timedout	0
md5	d59f24b86431eeb25281bce7817783f1
sha1	e2eb75ff817281dd8e21a3933ec0ebab9d0cd712
extracted	HTTP-F3yakQdAU13tQkrNa.exe
extracted_cutoff	0

((b)) Details of Anomalous HTTP File Log

Figure 4.6: Details of HTTP Files Logs and Anomalous HTTP File

Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN) are displayed to give an in-depth analysis of what is happening. Each element of the confusion matrix corresponds to the actual versus predicted classifications made by the model.

**Elements of the Confusion Matrix:**

**True Positive:** The model correctly predicts a certain amount of positive instances. Here, it refers to the quantity of malevolent files accurately identified by the model as harmful.

**True Negative:** This represents the amount of true negative instances. Referring to the benign files that the model correctly classified as non-malicious.

**False Positive:** This refers to the number of false positives in prediction where benign files are wrongly characterized as malicious as per the model’s judgment. It is also referred to as a false alarm in some statistical contexts. Also known as Type I error.

**False Negative:** This indicates several incorrectly predicted negative instances. These are malicious files that were incorrectly recognized as not malicious by the model. Also known as Type II error.

		Predicted Labels	
		Malicious	Benign
Actual Labels	Malicious	103,950	1,050
	Benign	450	44,550

**True Positives:** 103,950

**True Negatives:** 44,550

**False Positives:** 450

**False Negatives:** 1,050

The Fig. 4.7 shows a confusion matrix represented as a heatmap. Four items – True Positives, True Negatives, False Positives, and False Negatives

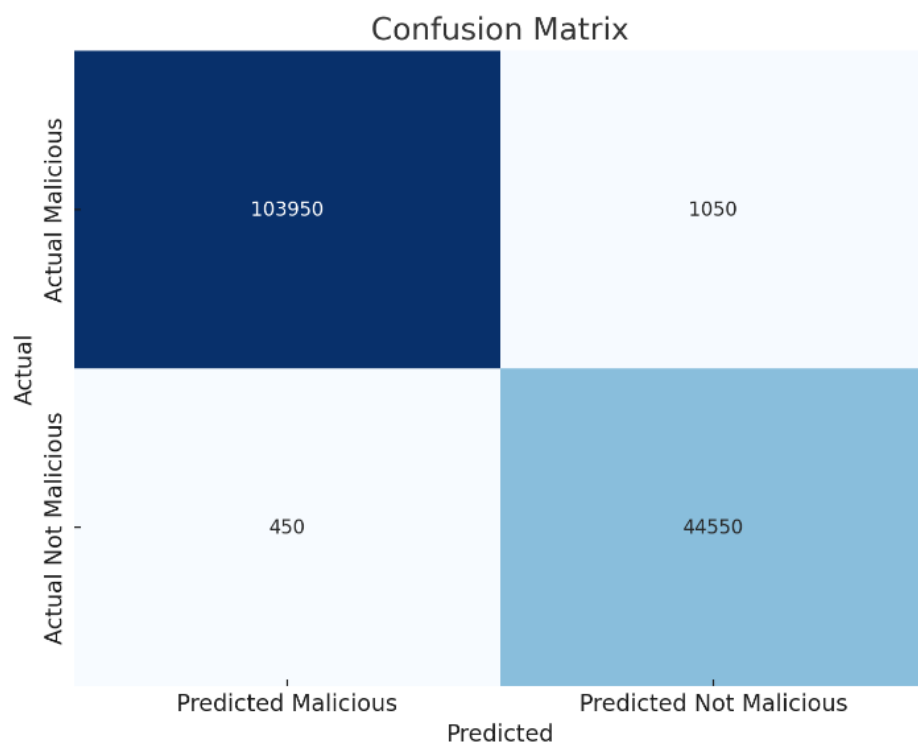


Figure 4.7: Heatmap of confusion matrix

– reveal how the Random Forest model has been performing on the dataset visually.

- **Axes**

**X-axis (Predicted):** This axis represents the predictions made by the model.

- Predicted Malicious: refers to how often a model determines malicious.
- Predicted Not Malicious: refers to the number of instances that the model predicted as not malicious.

**Y-axis (Actual):** This axis represents the actual classifications of the instances.

- Actual Malicious: refers to the number of truly malicious instances.
- Actual Not Malicious: refers to the number of truly benign instances.

- **Color Intensity**

The intensity of color in a cell relates to instances. Dark colors usually mean it has a higher count, therefore enabling one to tell areas of model success or failure easily:

- **Darker Blue Cells:** Indicate higher counts, this is where the model makes most accurate predictions (TP and TN).
- **Lighter Blue Cells:** Indicate lower counts; this is where the model has fewer incorrect predictions (FP and FN).

## Chapter 5

# CONCLUSIONS AND FUTURE RECOMMENDATION

In this thesis, we presented a comprehensive solution aimed at enhancing anomaly detection in network traffic, with a particular focus on analyzing HTTP logs. This work leverages advanced machine learning techniques, specifically Isolation Forest and K-means clustering, to tackle the complex challenge of identifying real intrusion attempts within large-scale, real-world network environments. Both algorithms are utilized in an unsupervised learning context, meaning they do not rely on predefined labels or patterns, which is crucial in scenarios where new or unknown threats may emerge. The success of these algorithms in our solution demonstrates their efficacy in distinguishing between normal and abnormal network behaviors, which is essential for detecting sophisticated cyber threats.

Once anomalies are identified, the information is fed into a Security Information and Event Management (SIEM) system. The integration with SIEM is a pivotal aspect of our solution, as it facilitates the visualization of network traffic patterns and anomalies, enabling security teams to monitor, analyze, and respond to potential threats in real-time. The visualization capabilities provided by SIEM are invaluable in enhancing the situational awareness of security operations centers (SOCs), allowing analysts to quickly identify and investigate suspicious activities. This integration not only streamlines the detection process but also bolsters the overall efficiency of cybersecurity operations, reducing the time and effort required to manage

and mitigate threats.

Our solution places a strong emphasis on HTTP traffic analysis, with a specific focus on the examination of files downloaded over HTTP. To achieve this, we employed Zeek, a powerful and widely used network monitoring and analysis tool. Zeek's ability to parse and analyze network protocols, including HTTP, allows us to delve deeper into the content of network traffic, providing a granular level of insight into potential threats. This focus on downloaded files is particularly important, as it addresses a common vector for malware delivery and other malicious activities. By scrutinizing the content of these files, our solution can detect anomalies that might otherwise go unnoticed, thereby enhancing the detection of threats that exploit HTTP traffic.

The approach we have developed is highly beneficial for SOC analysts and threat-hunting teams. One of the key advantages is the ability to identify intrusions without requiring extensive manual intervention or prolonged analysis. The use of unsupervised machine learning algorithms means that the system can autonomously detect deviations from normal behavior, flagging potential threats for further investigation. This automation not only reduces the workload on security teams but also allows them to focus on more strategic tasks, such as threat hunting and incident response. By providing a means to quickly and accurately identify intrusions, our solution enhances the overall effectiveness of cybersecurity operations.

Looking ahead, we see significant potential in further enhancing our solution by integrating an SSL (Secure Sockets Layer) offloader. SSL offloading involves terminating SSL or Transport Layer Security (TLS) connections at a dedicated device or service, effectively offloading the burden of encryption and decryption from other security components. This is particularly important in modern network environments, where a significant portion of traffic is encrypted using HTTPS. Without SSL offloading, security systems may struggle to inspect encrypted traffic, potentially allowing threats to bypass detection. By incorporating SSL offloading into our solution, we can ensure that HTTPS traffic is thoroughly examined, eliminating this blind spot and enhancing the overall security posture of

the network.

The integration of SSL offloading is a forward-looking enhancement that aligns with the evolving landscape of network security threats. As encryption becomes more prevalent, the ability to inspect encrypted traffic without compromising performance becomes increasingly important. SSL offloading not only improves the efficiency of traffic inspection but also ensures that our solution remains effective in the face of emerging threats. By addressing the challenges posed by encrypted traffic, we elevate our solution to a new level of comprehensiveness, making it a robust tool capable of protecting against a wide range of cyber threats.

In conclusion, this thesis presents a multifaceted approach to anomaly detection in network traffic, combining the strengths of machine learning, network analysis, and SIEM integration to create a powerful tool for cybersecurity. The successful implementation of Isolation Forest and K-means clustering, coupled with the detailed analysis of HTTP traffic using Zeek, demonstrates the effectiveness of our approach in real-world scenarios. The proposed integration of SSL offloading further enhances the solution, ensuring that it can adapt to the changing landscape of network security. Overall, our work contributes to the ongoing efforts to protect network environments from cyber threats, providing a scalable and adaptable solution that meets the needs of modern security operations.

# Bibliography

- [1] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection,” in *2010 IEEE Symposium on Security and Privacy*, 2010, pp. 305–316.
- [2] N. Moustafa, J. Hu, and J. Slay, “A holistic review of network anomaly detection systems: A comprehensive survey,” *Journal of Network and Computer Applications*, vol. 128, pp. 33–55, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1084804518303886>
- [3] S. Jin, J.-G. Chung, and Y. Xu, “Signature-based intrusion detection system (ids) for in-vehicle can bus network,” in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2021, pp. 1–5.
- [4] A. H. Almutairi and N. T. Abdelmajeed, “Innovative signature based intrusion detection system: Parallel processing and minimized database,” in *2017 International Conference on the Frontiers and Advances in Data Science (FADS)*, 2017, pp. 114–119.
- [5] F. Massicotte and Y. Labiche, “On the verification and validation of signature-based, network intrusion detection systems,” in *2012 IEEE 23rd International Symposium on Software Reliability Engineering*, 2012, pp. 61–70.
- [6] D. A. M. Villalba, D. F. M. Varon, F. G. Pórtela, and O. A. D. Triana, “Intrusion detection system (ids) with anomaly-based detection and deep learning application,” in *2022 V Congreso Internacional en*



- Inteligencia Ambiental, Ingeniería de Software y Salud Electrónica y Móvil (AmITIC)*, 2022, pp. 1–4.
- [7] S. Goel, K. Guleria, and S. N. Panda, “Anomaly based intrusion detection model using supervised machine learning techniques,” in *2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2022, pp. 1–5.
- [8] U. Kumari and U. Soni, “A review of intrusion detection using anomaly based detection,” in *2017 2nd International Conference on Communication and Electronics Systems (ICCES)*, 2017, pp. 824–826.
- [9] A. Juvonen and T. Hamalainen, “An efficient network log anomaly detection system using random projection dimensionality reduction,” in *2014 6th International Conference on New Technologies, Mobility and Security (NTMS)*, 2014, pp. 1–5.
- [10] A. Khudoyarova, M. Burlakov, and M. Kupriyashin, “Using machine learning to analyze network traffic anomalies,” in *2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus)*, 2021, pp. 2344–2348.
- [11] S. Saad, W. Briguglio, and H. Elmiligi, “The curious case of machine learning in malware detection,” in *International Conference on Information Systems Security and Privacy*, 2019. [Online]. Available: <https://api.semanticscholar.org/CorpusID:88467864>
- [12] M. Pietrek, “Peering inside the pe: a tour of the win32 (r) portable executable file format,” *Microsoft Systems Journal-US Edition*, vol. 9, no. 3, pp. 15–38, 1994.
- [13] P. Singhal and N. Raul, “Malware detection module using machine learning algorithms to assist in centralized security in enterprise networks,” *International Journal of Network Security Its Applications*, vol. 4, 02 2012.

- [14] E. Leon, O. Nasraoui, and J. Gomez, “Anomaly detection based on unsupervised niche clustering with application to network intrusion detection,” in *Proceedings of the 2004 congress on evolutionary computation (IEEE Cat. No. 04TH8753)*, vol. 1. IEEE, 2004, pp. 502–508.
- [15] P. Casas, J. Mazel, and P. Owezarski, “Unsupervised network intrusion detection systems: Detecting the unknown without knowledge,” *Computer Communications*, vol. 35, no. 7, pp. 772–783, 2012.
- [16] P. V. Amoli, “Unsupervised network intrusion detection systems for zero-day fast-spreading network attacks and botnets,” 2015. [Online]. Available: <https://api.semanticscholar.org/CorpusID:37499143>
- [17] A. Shiravi, H. Shiravi, M. Tavallaei, and A. A. Ghorbani, “Toward developing a systematic approach to generate benchmark datasets for intrusion detection,” *Computers Security*, vol. 31, no. 3, pp. 357–374, 2012. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167404811001672>
- [18] R. Aliakbarisani, A. Ghasemi, and S. Felix Wu, “A data-driven metric learning-based scheme for unsupervised network anomaly detection,” *Computers Electrical Engineering*, vol. 73, pp. 71–83, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0045790617332664>
- [19] A. Vikram and Mohana, “Anomaly detection in network traffic using unsupervised machine learning approach,” in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, 2020, pp. 476–479.
- [20] Y. Y. Aung and M. M. Min, “An analysis of random forest algorithm based network intrusion detection system,” *Advances in Science, Technology and Engineering Systems Journal*, vol. 3, no. 1, pp. 496–501, 2018.

- [21] Y. J. Chew, S. Y. Ooi, K.-S. Wong, and Y. H. Pang, “Decision tree with sensitive pruning in network-based intrusion detection system,” in *Computational Science and Technology*, R. Alfred, Y. Lim, H. Haviluddin, and C. K. On, Eds. Singapore: Springer Singapore, 2020, pp. 1–10.
- [22] D. Pérez, S. Alonso, A. Morán, M. A. Prada, J. J. Fuertes, and M. Domínguez, “Evaluation of feature learning for anomaly detection in network traffic,” *Evolving Systems*, vol. 12, no. 1, pp. 79–90, 2021.
- [23] D. J. D’Souza and K. Uday Kumar Reddy, “Anomaly detection for big data using efficient techniques: A review,” in *International Conference on Artificial Intelligence and Data Engineering*. Springer, 2019, pp. 1067–1080.
- [24] C. Yin, Y. Zhu, J. Fei, and X. He, “A deep learning approach for intrusion detection using recurrent neural networks,” *IEEE Access*, vol. 5, pp. 21 954–21 961, 2017.
- [25] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.
- [26] M. Ali, B. Almohammed, A. Ismail, and M. Zolkipli, “A new intrusion detection system based on fast learning network and particle swarm optimization,” *IEEE Access*, vol. PP, pp. 1–1, 03 2018.
- [27] Y. Jia, M. Wang, and Y. Wang, “Network intrusion detection algorithm based on deep neural network,” *IET Information Security*, vol. 13, no. 1, pp. 48–53, 2019. [Online]. Available: <https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-ifs.2018.5258>
- [28] U. Baldangombo, N. Jambaljav, and S.-J. Horng, “A static malware detection system using data mining methods,” 2013.

- [29] B. Mohammed, A. Monemi, S. Joseph, I. Ismail, S. Nor, and M. N. Marsono, "Feature selection and machine learning classification for malware detection," *Jurnal Teknologi*, vol. 77, 10 2015.
- [30] S. K. Sahay and A. Sharma, "Grouping the executables to detect malwares with high accuracy," *Procedia Computer Science*, vol. 78, pp. 667–674, 2016, 1st International Conference on Information Security Privacy 2015. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1877050916001174>
- [31] A. Kumar, K. Kuppusamy, and G. Aghila, "A learning model to detect maliciousness of portable executable using integrated feature set," *Journal of King Saud University - Computer and Information Sciences*, vol. 31, no. 2, pp. 252–265, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1319157817300149>
- [32] M. Yeo, Y. Koo, Y. Yoon, T. Hwang, J. Ryu, J. Song, and C. Park, "Flow-based malware detection using convolutional neural network," in *2018 International Conference on Information Networking (ICOIN)*, 2018, pp. 910–913.
- [33] V. T. Nguyen, L. Dung, and T. Le, "A combination of artificial immune system and deep learning for virus detection," *International Journal of Applied Engineering Research*, vol. 13, no. 22, pp. 15 622–15 628, 2018.
- [34] L. Yang, A. Ciptadi, I. Laziuk, A. Ahmadzadeh, and G. Wang, "Bodmas: An open dataset for learning based temporal analysis of pe malware," in *2021 IEEE Security and Privacy Workshops (SPW)*, 2021, pp. 78–84.
- [35] I. Shhadat, B. Al-bataineh, A. Hayajneh, and Z. Al-Sharif, "The use of machine learning techniques to advance the detection and classification of unknown malware," *Procedia Computer Science*, vol. 170, pp. 917–922, 01 2020.

- [36] S. Tyagi, A. Baghela, K. M. Dar, A. Patel, S. Kothari, and S. Bhosale, “Malware detection in pe files using machine learning,” in *2022 OPJU International Technology Conference on Emerging Technologies for Sustainable Development (OTCON)*, 2023, pp. 1–6.
- [37] O. Barut, T. Zhang, Y. Luo, and P. Li, “A comprehensive study on efficient and accurate machine learning-based malicious pe detection,” in *2023 IEEE 20th Consumer Communications Networking Conference (CCNC)*, 2023, pp. 632–635.
- [38] F. H. Ramadhan, V. Suryani, and S. Mandala, “Analysis study of malware classification portable executable using hybrid machine learning,” in *2021 International Conference on Intelligent Cybernetics Technology Applications (ICICyTA)*, 2021, pp. 86–91.
- [39] R. S. Santos and E. D. Festijo, “Classifying portable executable malware using deep neural decision tree,” in *2022 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom)*, 2022, pp. 162–167.
- [40] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, “Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study,” *Journal of Information Security and Applications*, vol. 50, p. 102419, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2214212619305046>
- [41] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” in *2008 Eighth IEEE International Conference on Data Mining*, 2008, pp. 413–422.
- [42] F. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation-based anomaly detection,” *ACM Trans. Knowl. Discov. Data*, vol. 6, no. 1, mar 2012. [Online]. Available: <https://doi.org/10.1145/2133360.2133363>