Dhruv Verma

Enhancing Cybersecurity Through Adaptive Anomaly Detection Using Modern AI Techniques

Master's thesis of artificial intelligence

May 22, 2024

University of Jyväskylä

Department of Mathematical Information Technology

Author: Dhruv Verma

Contact information: dhruvverma5923@gmail.com

Supervisors: Prof. Vagan Terziyan

Title: Enhancing Cybersecurity Through Adaptive Anomaly De-tection Using Modern AI Techniques

Työn nimi: Kyberturvallisuuden parantaminen mukautuvan poikkeavuuksien havaitsemisen avulla käyttämällä nykyaikaisia tekoälytekniikoita

Project: Master's thesis

Study line: Information Technology

Page count: 7 + 62

Abstract:

This thesis explores how artificial intelligence (AI) techniques and machine learning (ML) algorithms can enhance adaptive anomaly detection frameworks while aiming to develop effective strategies for identifying and mitigating evolving cyberthreats. The main objective of this study is to create a unified framework that reduces manual intervention, minimizes false positives, and offers a robust and resilient approach to threat mitigation.

To collect datasets for analyzing the behavior of cyberthreats, a number of cyber-attack simulations, including threats such as - malware, data breaches, and SQL injections, were created. Additionally, malicious datasets from Kaggle were utilized to provide a larger amount of data. After normalizing the datasets, several ML algorithms were utilized to train the data and establish a threshold mechanism. This mechanism dynamically adjusts parameters corresponding to specific cyberthreats, ensuring accurate identification and mitigation. This study demonstrates how modern AI techniques can transform anomaly detection, making it more effective, time-efficient, and resource-friendly.

Keywords: Artificial intelligence, adaptive anomaly detection, threat mitigation, machine learning, cyber threat simulations, threshold mechanism

Suomenkielinen tiivistelmä:

Tämä opinnäytetyö tutkii, kuinka tekoälytekniikat (AI) ja koneoppimisalgoritmit voivat parantaa adaptiivisia poikkeamien havaitsemiskehyksiä ja pyrkiä kehittämään tehokkaita strategioita kehittyvien kyberuhkien tunnistamiseen ja lieventämiseen. Tämän tutkimuksen päätavoitteena on luoda yhtenäinen viitekehys, joka vähentää manuaalisia toimenpiteitä, minimoi vääriä positiivisia tuloksia ja tarjoaa vankan ja kestävän lähestymistavan uhkien lieventämiseen.

Tietojen keräämiseksi kyberuhkien käyttäytymisen analysointia varten luotiin useita kyberhyökkäyssimulaatioita, mukaan lukien uhkia, kuten - haittaohjelmat, tietomurrot ja SQL-injektiot. Lisäksi Kagglen haitallisia tietojoukkoja hyödynnettiin suuremman tietomäärän tuottamiseen. Datajoukkojen normalisoinnin jälkeen käytettiin useita ML-algoritmeja tietojen kouluttamiseen ja kynnysmekanismin luomiseen. Tämä mekanismi säätää dynaamisesti tiettyjä kyberuhkia vastaavia parametreja varmistaen tarkan tunnistamisen ja lieventämisen. Tämä tutkimus osoittaa, kuinka nykyaikaiset tekoälytekniikat voivat muuttaa poikkeamien havaitsemista tehden siitä tehokkaamman, aikatehokkaamman ja resursseja säästävämmän.

Avainsanat: Tekoäly, mukautuva poikkeamien havaitseminen, uhkien lieventäminen, koneoppiminen, kyberuhkien simulaatiot, kynnysmekanismi

Preface

This study summarizes my research, which was carried out at the University of Jyväskylä's Department of Mathematical Information Technology on the subject of "Enhancing Cyber-security Through Adaptive Anomaly Detection Using Modern AI Techniques."

I have gained a great deal of knowledge and understanding of various topics during my study and research journey. I would like to express my gratitude to Professor Vagan Terziyan, my supervisor, for his guidance and support, which have enabled me to complete the research and make a significant contribution to the fields of cybersecurity and artificial intelligence.

Jyväskylä, May 22, 2024

Dhruv Verma

Glossary

SVM	Support Vector Machines
SOM	Self-Organizing Maps
GAN	Generative Adversarial Networks
CNN	Convolutional Neural Networks
RNN	Recurrent Neural Networks
DoS	Denial of Service
DDoS	Distributed Denial of Service
MitM	Man-in-the-Middle
SQL	Structured Query Language
DNS	Domain Name System
IDS	Intrusion Detection Systems
SIEM	Security Information and Event Management
IOC	Indicators of Compromise
DBSCAN	Density-Based Spatial Clustering of Applications with Noise

List of Figures

List of Tables

Contents

1 Introduction

Since the beginning of modern networking, the internet has become a necessary component of modern society. It has led to a number of developments in the fields of information distribution, commerce, and communication. However, as the number of activities conducted online increases, so do the issues regarding to cyberthreats. These threats pose a significant risk to people, businesses, and countries. This highlights the need for effective and efficient cybersecurity measures, which are becoming increasingly necessary as a result of the evolution of cyberthreats. These threats target sensitive data, private information, and even organizations and their infrastructures. The aim of this thesis is to create a robust framework that uses ML and AI algorithms to enhance the adaptive anomaly detection in cybersecurity practices. The framework is programmed to be adaptive in nature, which allows for dynamic adjustments of detection methods. This detection methods increases efficiency, effectiveness, and responsiveness in detecting and mitigation several threats such as malware infections, network intrusions, insider threats, data breaches, etc.

1.1 Background Theory

The increasing volume of sensitive data leads to the emergence of high risks cyberthreats. This increased volume requires innovative approaches to safeguard and monitor digital assets. At its core, the cybersecurity challenge revolves around the dynamic interactions between the complicated threats and antivirus defenses, trying to protect their respective digital resources. Traditional cybersecurity measures prove to be effective while detect-ing and mitigating known threats, however they become insufficient when facing the evolving cyberthreats. To counter this problem, the integration of modern AI techniques offers promising opportunities for enhancing cybersecurity defenses, by achieving better security and measures. The integration of ML algorithms allows the cybersecurity frameworks to adapt and evolve by providing dynamic and updating defense mechanisms. To address this challenge, algorithms such as - supervised learning, unsupervised learning, deep learning, and reinforcement learning are utilized. Using these algorithms allows the systems to autonomously detect and respond to anomalous behavior that potentially indicates cyberthreats.

Traditional threat detection includes basic rule-based and signature-based systems, which are used to identify the pre-determined threats. These detection systems have evolved to advanced, context-aware models. These developments enable the systems to identify subtle patterns in large datasets. The increasing availability of data and computing power has led to an increasing need of integrating ML algorithms in traditional cybersecurity systems. Algorithms like supervised learning are useful at identifying patterns from existing labeled data. Other algorithms such as unsupervised learning techniques do not use any previous data to identify abnormal behavior. Additionally, techniques like learning patterns and neural network architectures can also be used to classify the datasets.

Support Vector Machines (SVMs) categorize each data instance into groups like - benign or malicious. Furthermore, Self-Organizing Maps (SOMs) are effective for clustering and representing high-dimensional data in lower-dimension maps. These maps help to visualize and identify abnormal patterns within the complex datasets. Moreover, Generative Adversarial Networks (GANs) are useful to generate synthetic data to address the challenge of imbalanced datasets. This generation of statistical data also provides options for testing and optimizing different algorithms.

In addition to this, the combination of deep learning techniques like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), makes it effective to extract detailed features from large and complicated datasets. This enhances the detection of cyber-threats such as malware and phishing attacks. Methods such as ensemble learning combine multiple anomaly detection models to enhance overall robustness, resulting in superior performance compared to individual methodologies. Finally, explainable AI (XAI) techniques improve the interpretability of the model's decision-making process of models and make them more reliable.

The essential analysis of the theoretical and practical aspects of adaptive anomaly detection techniques using modern AI methods is presented in the thesis. It gives deep insights and analysis on various methods and approaches used in anomaly detection. The initial section covers the theory of cybersecurity and its relation to AI. Then, it involves classification and

demonstration of different ML approaches applied in the field of cybersecurity. Each technique is analyzed to explore its functions and importance and significance related to threat detection defenses. Following this, the practical implementation of the framework is explored. Many real-world scenarios have been created and simulated to analyze the efficiency and effectiveness of these techniques in detecting anomalies from different datasets.

The study also explores different strategies which can be useful to optimize various anomaly detection techniques. Finally, the thesis concludes with the results and findings from the research and practical application process, and offers information and suggestions for future work.

1.2 Research Problems

The fundamental issue in the current threat environment is the growth of dynamic cyber threats. This problem requires new approaches to protect and guard digital information. Common methods use static defense mechanisms like firewalls and antivirus software to protect the systems. Unfortunately, these defenses are ineffective against the complex nature of modern threats (Sontan & Samuel, 2024). Furthermore, relying on manual detection and mitigation reduces response speed which makes environment more susceptible to exploitation (Fadziso et al., 2023). To solve these challenges, there is an urgent need for effective AI-driven system. This system should be capable of preventing real-time threats of data breaches and anomalies (Huyen & Bao, 2024).

Integrating cognitive computing methodologies present a favorable opportunity for enhancing AI-driven cybersecurity systems. By adapting and fine-tuning cognitive computing techniques, such systems can exhibit self-updating and learning capabilities. This ensures the continuous adaptation to emerging threats and vulnerabilities. This exploration produces a lot of important questions, such as - reflecting on the most efficient AI-driven techniques to prevent data attacks and how to provide a streamlined and resilient security solution. Strategies for optimizing real-time anomaly detection algorithms for autonomous identification are also considered. Furthermore, fine-tuning, and self-updating algorithms are also taken

into the account to ensure the continuous adaptation to the emerging threats. Lastly, the innovative approaches to integrate AI techniques with traditional cybersecurity tools are reflected upon.

To address these research problems, the thesis proposes an investigating innovative strategies and methodologies that are useful in threat detection. In addition, it also seeks to contribute to the development of flexible and adaptive cybersecurity solutions that are capable of effectively combating emerging cyberthreats in real-time. This is achieved by analyzing different techniques, algorithms, and methodologies for improving the responsiveness and efficacy of anomaly detection.

1.3 Research Methods

Several important steps were involved in the research methodology to fully analyze the chosen methods via practical applications. A comprehensive review of relevant literature was done initially. This was intended to help in understanding the theory, the advancements, and adaptive nature of anomaly detection in cybersecurity. This literature review also helped in framing the re-search questions and defining the scope of the study. After the review, different ML algorithms were chosen based on their relevance, effectiveness, and potential impact on cybersecurity. Techniques such as SVMs, SOMs, CNNs, and GANs were used for further analysis.

To evaluate the effectiveness of these techniques, a series of experiments and simulations were conducted using real-world cybersecurity datasets. These datasets were carefully selected and represented a wide range of cyberthreats and anomalies, including computer network intrusions, malware infections, and other anomalies. The contents of the selected datasets were thoroughly normalized to ensure that the behaviors within them were widely representational and relevant to current cybersecurity challenges. Ultimately, each adaptive anomaly detection technique was implemented and fine-tuned using appropriate ML algorithms. Parameters associated with selection of features, the optimization methods, and other related aspects were adjusted to maximize the accuracy of the threat detection model and

minimize false results. Different scenarios such as network intrusion detection, endpoint security, and application log detection were considered to assess the adaptability and effectiveness of the techniques in cybersecurity.

Overall, the research methodology adopted in this study aimed to provide a thorough and evidence-based evaluation of adaptive anomaly detection techniques through modern AI methodologies. By combining theoretical insights with practical experimentation, the study aims to contribute to the advancement of cybersecurity practices against emerging cyber-threats.

1.4 Literature Review

The study - *'Adaptive anomaly detection system based on machine learning algorithms in an industrial control environment'* by Vávra, Hromada, Lukáš, & Dworzecki examines adaptive anomaly detection systems using ML in industrial control environments. The study presents a solution by integrating four ML algorithms and preprocessing techniques to defend against cyber-attacks. The findings of the study show susceptibility to overfitting but the results still confirm to be proven applicable in real environments (Vávra et al., 2021). This review is utilized to find techniques useful to mitigate specific threats.

Another study - *'A Review of Anomaly Detection Strategies to Detect Threats to Cyber-Physical Systems'* by Jeffrey, Tan, and Flecha provides a comprehensive review of anomaly detection strategies for Cyber-Physical Systems (CPS). It emphasizes the urgent need of safeguarding critical infrastructure against malicious attacks. The study has identified different challenges based on the analysis of 296 papers. The challenges include problems such as - resource constraints and the inconsistency between academic research and industry adoption. Though the industrial implementation of the AI- behavior-based detection remains limited. This study also underscores the need for collaborative efforts to bridge the gap and enhance CPS cybersecurity (Jeffrey et al., 2023). This review was crucial to understand the theory behind the mechanisms of the infrastructure required to mitigate threats.

Similarly, another research - *'Adaptive generative AI for dynamic cybersecurity threat detection in enterprises'*, by Vemuri, Thaneeru, and Tatikonda (2024) offers a detailed review

of adaptive generative AI applications for dynamic cybersecurity threat detection. By analyzing existing literature and case studies, the research dives into various methodologies, including ML algorithms and real-time data processing. These methodologies are aimed at enhancing threat detection capabilities. The study also highlights the evolving nature of cyberthreats and the necessity for adaptive solutions. Through an examination of strengths and limitations, the research provides valuable information about the effectiveness of adaptive generative AI in mitigating cybersecurity risks. It also spotlights the collaborative importance among humans and AI. (Naveen Vemuri et al., 2024).

The study - *'Effectiveness of Artificial Intelligence Techniques Against Cyber Security Risks in the IT Industry'* by Bilal Alhayani, Husam Jasim Mohammed, and Ibrahim Zeghaiton Chaloob focuses on assessing the impact of AI techniques on cybersecurity. Using a quantitative approach with primary data, the research utilized factor analysis, discriminant validity, and hypothesis testing. The results indicate significant positive impacts of intelligence agents and neural nets on AI's effectiveness in countering cyberthreats. Though the study highlighted the growing importance of cybersecurity amidst increasing internet usage and cybercrimes, but it noted some limitations corresponding sample size and accessibility (Alhayani et al.). The research information provided in the review was significant in understanding the training and collection data mechanisms utilized in this study.

1.5 Structure of Thesis

The thesis is divided into six chapters, covering the theory, practical implementation, research questions and subsequently the conclusion of the research,

Chapter 1 serves as the introduction, setting the stage by defining the motivation behind the research and expresses the main research problem and objectives. It provides an initial exploration into the field of adaptive anomaly detection in cybersecurity, explaining its significance in the digital age.

In chapter 2, the conceptual framework is elaborated. Here, the theoretical foundations and key concepts of anomaly detection methodologies are examined. Additionally, it provides a

comprehensive overview of threat landscape and protocols essential for understanding cybersecurity dynamics.

Chapter 3 is dedicated to the data collection and processing methodologies employed in the research. It outlines the techniques utilized to capture application and network data for subsequent analysis. Moreover, it discusses the criteria employed in selecting modern AI techniques beneficial to adaptive anomaly detection, while also detailing the experimental setup.

Chapter 4 delves into the research questions; their AI techniques and consequent ML algorithms used to explore and explain the ideas and information for the said questions.

Chapter 5 is devoted to the implementation phase of the research. It provides the detailed account of how the framework was constructed, based on the collected and processed data, as discussed in chapter 3.

Chapter 6 focuses on the future work and provides a ground for implementing additional features for adaptive anomaly detection systems.

Lastly, chapter 7 summarizes the conclusion, the key findings, and their implications for cybersecurity practices.

2 Conceptual Layout

This chapter focuses on the conceptual layout of the research, breaking down the key concepts, theoretical foundations of the anomaly detection in cybersecurity. Additionally, it offers insights into the application of modern AI methodologies within the adaptive nature of the anomaly detection framework.

2.1 Anomaly Detection in Cybersecurity

Anomaly detection is one of the most significant components of the defensive strategies aimed at identifying and mitigating various malicious activities and security breaches within different digital systems and networks. It includes continuous monitoring and analyzing data streams to identify deviations from the existing patterns that could signify security threats. These data streams include but are not limited to – network traffic logs, system event records, user activity trails, application logs, and cloud server logs. These datasets serve as the key component to detect deviations by recognizing irregularities and unusual patterns within the data itself (Nicolas Guzman Camacho, 2024). The ability to distinguish between real fluctuations and genuine threats, is assisted by a diverse arrangement of detection techniques. Figure 1 presents the hierarchy that make up the concept of anomaly detection.

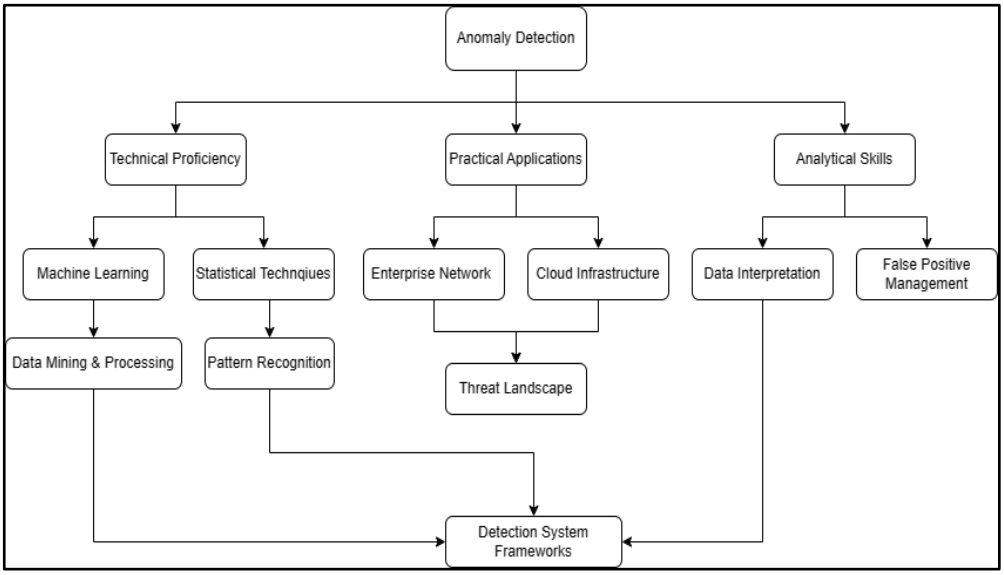

Figure 1. Anomaly Detection - Hierarchy

The main approach for anomaly detection allows for detecting previously unseen threats that do not match the known attack signatures. There are multiple types of anomalies – point anomalies, contextual anomalies, and collective anomalies. By potentially identifying the root of these abnormalities, organizations can respond quickly and mitigate the potential threats before they escalate into full-fledged attacks.

Point anomalies are individual data points which stray significantly from the dataset. As an example, high network traffic spikes; or system log outliers. They are helpful in identifying Denial of Service (DoS) attacks, brute force attacks, exfiltration of data and network intrusions. On the other hand, contextual anomalies depend on context and can only be regarded as malicious under specific conditions established within the system. These require an understanding of the broader context in which the events occur (Hayes & Capretz, 2014). For example, accessing sensitive information outside normal working hours might be seen as a contextual anomaly. This type can assist in recognizing insider threats, phishing attacks, and data breaches. Finally, collective anomalies occur collectively across multiple data points just as the name suggests. Nevertheless, these anomalies are subtle and necessitate analysis of various trends and patterns within the data for their detection. They are helpful when it comes to identifying Distributed Denial of Service (DDoS) attacks, botnet activity, persistent threats as well as financial frauds.

By utilizing contextual and collective anomalies, organizations can uncover more complex and coordinated attacks that may otherwise go undetected with traditional detection methods. Consequently, this enables proactive threat detection and quick response.

2.2 Cyber-Threats Landscape

Numerous cybersecurity threats emerge every year, ranging from malware infections and insider threats to data breaches and cross-scripting attacks. Effectively detecting and mitigating these requires optimal threat detection strategies that include a combination of techniques and technologies. By understanding the complications of such strategies and implementing the corresponding comprehensive security measures, organizations can strengthen

their defenses and overcome any potential risks effectively and efficiently. Given the abundance of cyberthreats, it becomes quite vital to classify them into separate categories. The cybersecurity threats can be classified into three categories – network attacks, malware threats, and application-level threats. Each of these classifications depends on the specific type of data that each threat is engineered to exploit. Some threats exploit the network logs, some attack the system directories, and the others attack the application itself.

2.2.1 Network Attacks

Network attacks primarily aim at an organization's network infrastructure to interrupt services, spy on communications or gain unauthorized entry into network resources. These kinds of attacks can cause many different effects such as downtime and service disruption, changing communication between services, listening in on private/sensitive content, altering data transmission paths or taking advantage of weaknesses within systems and programs. In addition, it may also lead to redirecting services towards malicious servers thereby creating unintended destinations. An example includes DDoS (Distributed Denial of Service) where multiple devices are used at once to overload a targeted server with requests and data traffic until it becomes unreachable; Man-in-the-middle attack (MitM) involves eavesdropping on a two-party communication system while impersonating either endpoint; SQL injection refers to code injection technique that might destroy systems containing databases among other things if not appropriately checked against; DNS spoofing occurs when falsified information about domain name resolution is sent by an attacker so as to redirect users from legitimate sites to fake ones – normally used during phishing campaigns aimed at stealing login credentials from unsuspecting victims.

In order to ensure that the network is safe from attacks, intrusion detection systems (IDS) are used to keep an eye on network traffic and identify any abnormal patterns that may be a sign of an attack. Security information and event management (SIEM) systems connect the dots between non- comparable data sources, creating alerts for suspicious activities throughout the network (Wu et al., 2020). Although these methods work well, they must be kept up-to- date with changes in cybersecurity threats. This is where machine learning (ML) and artificial intelligence (AI) come into play. ML algorithms can sift through large volumes of

information to spot suspicious activities that could indicate harmful intent with little demand for resources. Furthermore, AI- driven threat intelligence platforms have the ability to continuously track global threat feeds in real time so as to recognize known attack signatures or indicators of compromise (IoCs). This enables organizations to detect and respond to network attacks more effectively. For example, anomaly detection algorithms can characterize unusual traffic spiles in the network packets, whether from DDoS attacks or brute force attacks.

Figure 2 shows the number of packets received in a given time. The horizontal axis represents the time in seconds, and the vertical axis represents the number of packets. In this specific case, the expected amount is lower than the received amount, indicating the chances of a DoS or DDoS attack.

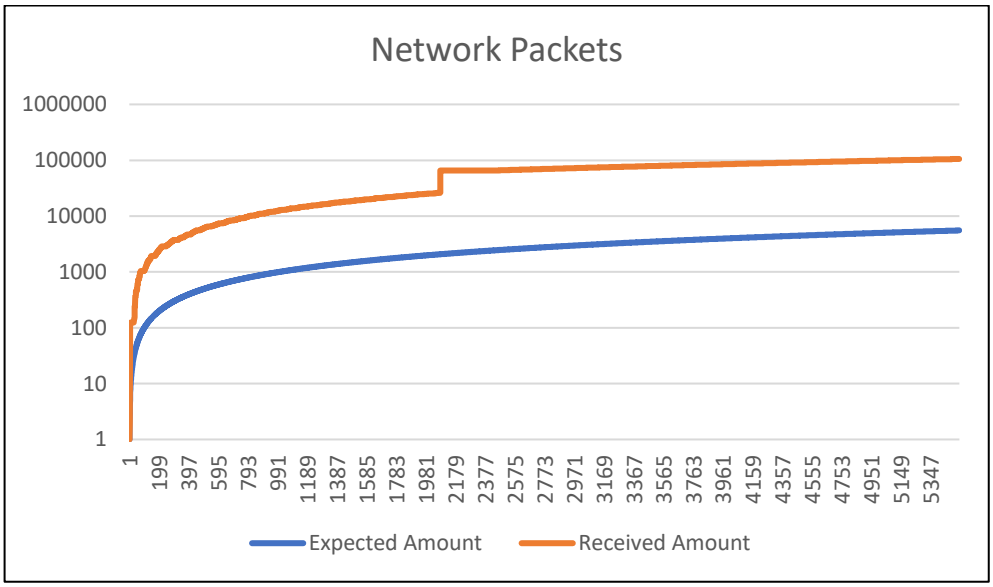

Figure 2. Network Packets

2.2.2 Malware Infections

Malware includes various types of threats such as viruses, worms, trojans, ransomwares, and spywares. These attach themselves to legitimate programs or files and replicate themselves into sensitive directories at the time of their execution. Each malware threat significantly damages the system and it is important to eliminate the threat as soon as possible. The effects

of malware range from performing unauthorized actions and encrypting files to stealing user activity and collecting sensitive information. Similar to network attacks, malware infections can be mitigated by recognizing patterns stored in antivirus databases. This mitigation strategy is known as signature-based detection. On the other hand, behavior-based detection analyzes the program behavior for any malicious activity by isolating it into a sandbox environment. These techniques check for system logs in order to find out the commands and/or logic behind the program. This allows for utilizing the previously stored information in the antivirus application to compare with the logs of the infected program.

However, the traditional techniques come at the cost of resources. In order to improve efficiency, and quick mitigation, anomaly detection ML algorithms can be deployed to carry out pattern recognition and analyze attack vectors (Scott, 2017). This overall can be integrated with AI methodology to self-learn, thus improving accuracy and reducing the risks of infections. Figure 3 shows the log entry (event status, source, detection ID, etc.) of a malware zip file downloaded into a Windows 7 OS.

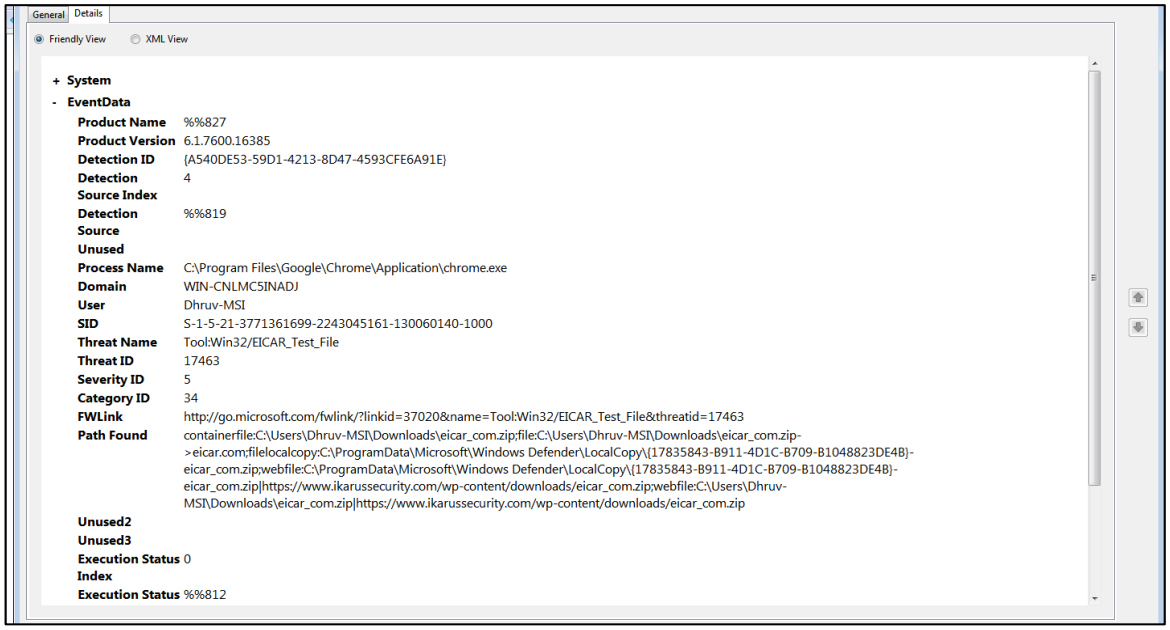

Figure 3. Malware Log Entry

2.2.3 Application-Level Threats

Application-Level threats specifically target the vulnerabilities present in the software applications, web applications, and services. Such threats exploit weaknesses and gain unauthorized access to sensitive data and resources. These threats usually inject malicious scripts inside the application itself leading to session hijacking or data theft. Sometimes, they target applications' database layer to execute malicious Structured Query Language (SQL) commands and gain access to the sensitive information. Similarly, memory buffering, executing arbitrary code, and over-riding the initial intent of the application is not out of bounds for application-level threats (Swamy et al., 2017). Cross-Site Scripting (XSS), SQL injections (SQLi), buffer overflow attacks all fall into this category.

Web Application Firewalls (WAF) filter and monitoring Hypertext Transfer Protocol (HTTP) traffic can be used to detect and block malicious requests, including the SQLi(s) and XSS attacks. Code reviews and vulnerability assessment also prove be of significant assistance to overcome these potential threats.

AI and ML techniques can specifically analyze the application logs much efficiently and identify suspicious behavior and patterns that indicate the malicious activity. These algorithms can learn from historical application data and detect deviations from the normal behavior that may signal an attack. This includes identifying unusual user interactions and code execution patterns. Additionally, AI-powered code analysis can verify security vulnerabilities in software applications and prioritize remediation efforts based on the risk severity. Figure 4 represents the application error log for the Bonjour service. This networking service enables automatic discovery of devices on a local network, and is easily breached.

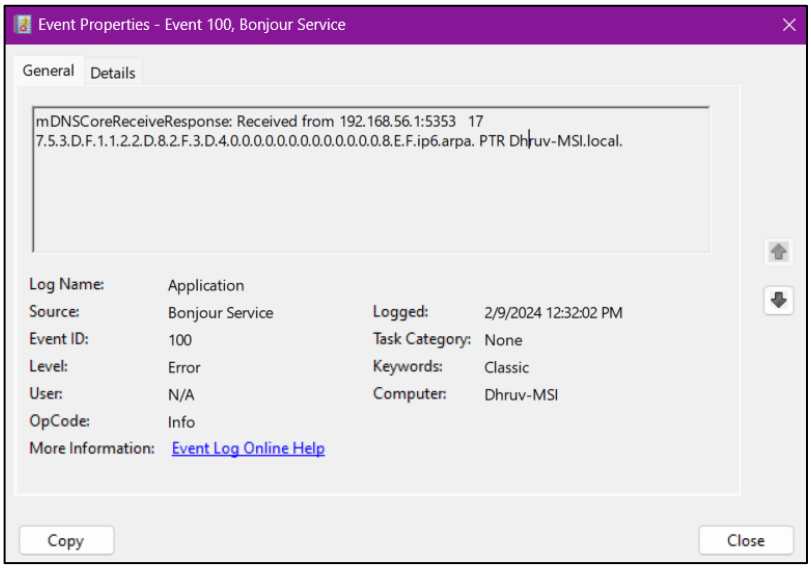

Figure 4. Error Log - Application

2.3 Dynamic and Adaptive Models

In cybersecurity, dynamic and adaptive models apply the transformative nature of AI and ML algorithms to effectively revolutionize anomaly detection methodologies. At their core, these models utilize complex algorithms to adjust to the evolving cyber threat landscape by continuously learning from new data and refining their detection capabilities over time. To achieve this goal, it is essential to apply reinforcement learning algorithms in these models, which allow them to receive a lot of feedback and then recalibrate the decision-making process (Mulugeta, 2023). This dynamic feedback loop ensures that the models remain adjusted to any shift in the cyberthreats by effortlessly adapting to fluctuations in network traffic patterns or the emergence of innovative attack vectors. Figure 5 shows the environment set-up for the dynamic learning with re-training using feedback information and new datasets.

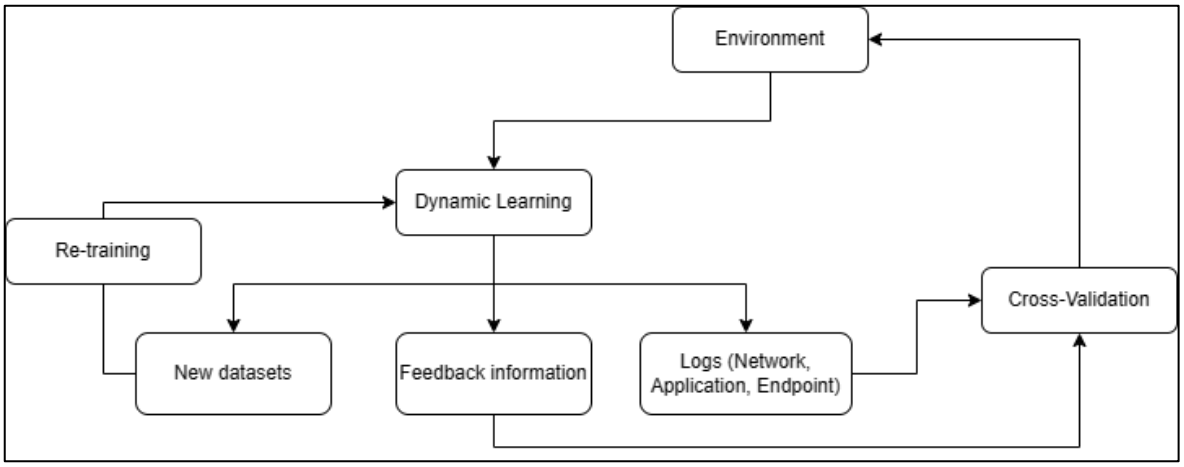

Figure 5. Dynamic Learning

Furthermore, the learning ability of these models is directly correlated to the involvement of neural networks and deep learning architectures. These algorithms server as the key component of the models' data processing, by allowing the models to examine vast amount of data with utmost precision. By extracting and filtering complicated patterns and relationships from the raw datasets, these models can uncover subtle anomalies that evade the traditional detection methods. Fundamentally, the effectiveness of the models in countering unseen threats is underpinned by the deep understanding of the data structures. Using outlier analysis, these models can verify the deviations even in the absence of historical data. This adaptability equips them to proactively mitigate zero-day exploits or any emerging attack vectors.

Moreover, dynamic and adaptive models excel at removing false positives which is one of the most significant challenges in anomaly detection. By iteratively refining the detection algorithms based on the real-time feedback, these models minimize the risk of inaccurately flagging the non-malicious data. This capability is invaluable as the static thresholds set for the models are likely to yield false alarms. By using these advanced methodologies, organizations can enhance their anomaly detection capabilities, effectively mitigate risks, and fortify their defenses against complex nature of cyberthreats. Figure 6 shows the data collection and training cycle for anomalous detection to conclude if the potential threat is either malicious or benign.

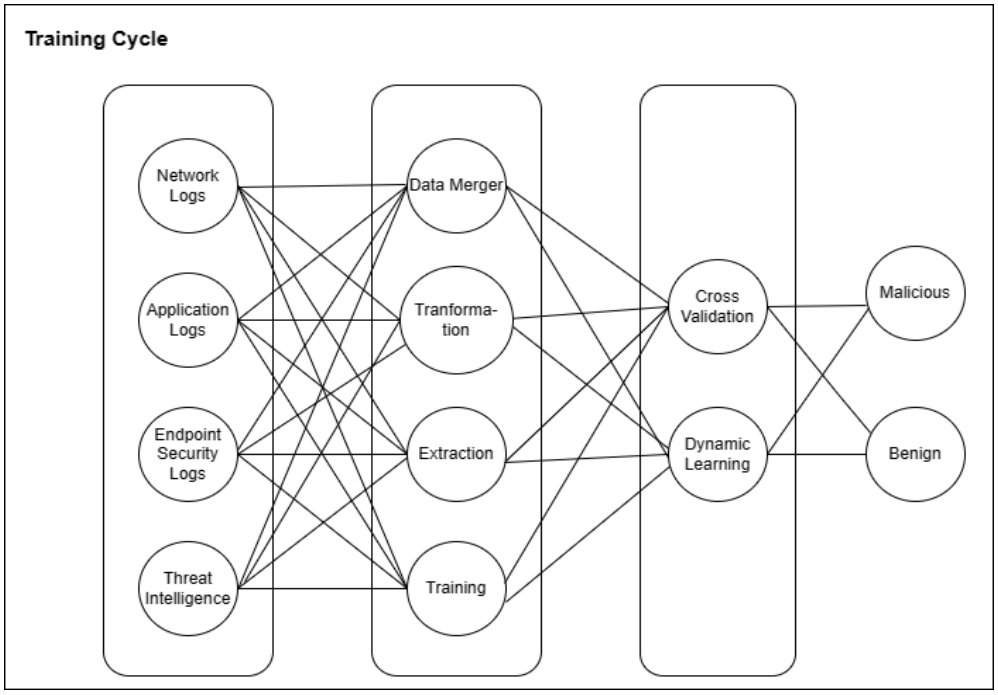

Figure 6. Data Training Cycle

2.4 Integration with Existing Security Infrastructure

One major way to improve anomaly detection is to integrate existing security infrastructure with the dynamic and adaptive cybersecurity models. This integration ensures smooth compatibility and seamless communication between different security systems such as SIEM platforms, endpoint security solutions, and Network Intrusion Detection Systems (NIDS). By implementing standard protocols and APIs, the models can facilitate seamless data exchange and collaboration among diverse security tools.

The main benefit of these systems is the ability of the models to combine the strengths of each component to create a unified defense mechanism. For example, SIEM platforms excel at gathering and analyzing security events from various sources. These events provide a detailed view of security incidents across the entire system. With the integration, the adaptive and dynamic models can identify and respond to the event emerging threats immediately. In addition to this, the NIDS monitors the network traffic for signs of suspicious activities within the logs, such as intrusion attempts or unauthorized access. By using the adaptive

models, the network-based threats can be easily detected and promptly mitigated. Moreover, merging endpoint security with the adaptive models enhances endpoint visibility and protection. The endpoint solutions, such as antivirus software and Endpoint Detection and Response (EDR) platforms, focus on safeguarding individual devices from malicious activities. This reduces the risks of compromise and data breaches. Furthermore, adding the threat intelligence feeds and platforms compliment the integrated security model by providing the information for emerging threats and attack patterns. Utilizing the said information can help in updating the models and organizations can easily anticipate and prepare for such attacks (Schmitt, 2023).

In essence, integrating existing infrastructure with dynamic and adaptive security models is essential for creating a robust, resilient, effective, and efficient cybersecurity defense framework, thereby safeguarding the critical assets and sensitive data.

3 Research Methodology and Data Collection

This chapter focuses on the research methodology and is dedicated to the data collection and processing techniques applied before the implementation. All the methodologies that allow to capture network logs, end-point logs, and application logs are utilized. These captured raw datasets are processed and subsequently analyzed.

3.1 Data Collection

To establish a unified anomaly detection system, it is necessary to collect and process various types of datasets, enabling training within ML and AI frameworks.

3.1.1 Network Traffic Data

Network traffic data provides a comprehensive, yet clear view of all the communications occurring within the networks, including all the interaction between services, devices, applications, and users. This visibility is crucial for detecting anomalous patterns and behaviors that may indicate the potential security threats. In order to capture such packets different approaches like – passive monitoring and packet sniffing are applied. In order to comply with this necessity, a network bot was created.

This network bot followed a Python script, programmed to dispatch various requests to a multiple of servers and websites, including common public domains like Google and Amazon, as well as platforms like eBay, Discord, Craigslist etc. The URLs for these destinations were sourced from open directories on the web, enabling the bot to access a diverse array of sites. To manage the vast amount of data generated by this operation, the websites were segmented into 8 sets, each containing around 100 URLs.

Figure 7 shows the code to capture the network packets for the datasets. This bot captures general packets using the command line function for tshark, which is a subset of Wireshark.

```
urls = [...

current_dir = os.path.dirname(os.path.abspath(__file__))
capture_file = os.path.join(current_dir, "capture1.pcap")

print("Starting Wireshark capture...")
capture_process = subprocess.Popen(["tshark", "-i", "eth", "-w", capture_file], stdout=subprocess.PIPE, stderr=subprocess.PIPE)

for url in urls:
    visit_url(url)
    time.sleep(2)
```

Figure 7. Network Capture Code

Once the bot initiated its visits to the websites, it carefully captured and stored the traffic data logs using the packet capturing tool – Wireshark. This tool proved instrumental in dissecting and recording packets based on the required protocols, including TCP, TLS, AJP13, and ARP. Recorded network logs with different protocols are presented in the figure 8.

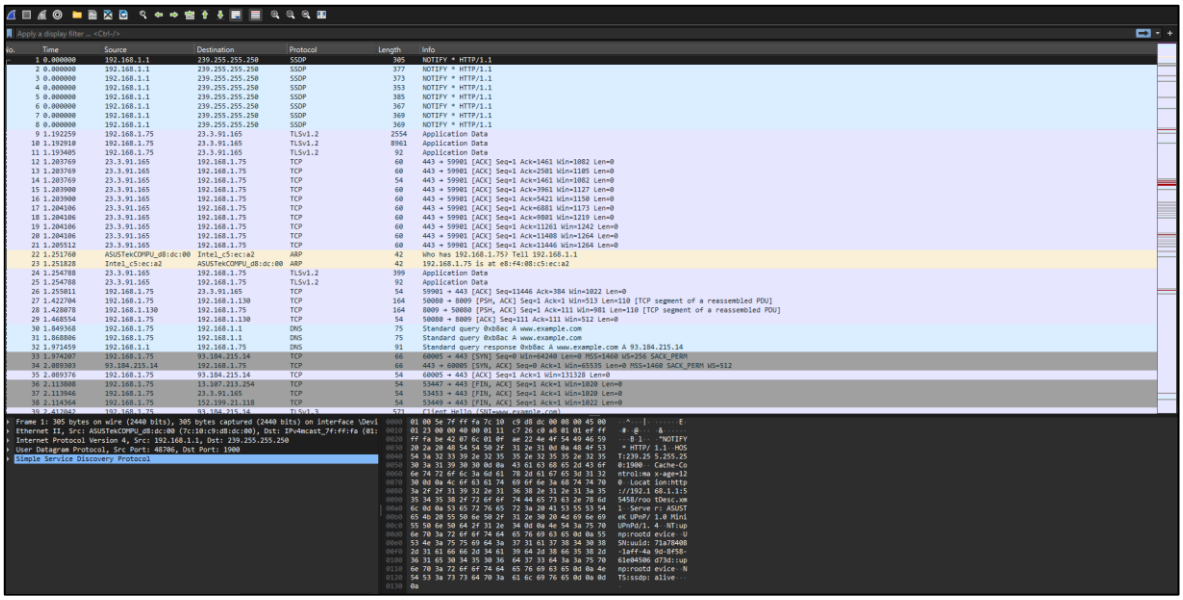

Figure 8. Network Packets - Wireshark

By utilizing the communication between the network bot and the visited websites, Wireshark provided valuable insights into the network interactions. The raw dataset harvested from the network bot's interactions undergoes an important initial phase of processing, characterized by filtration and cleaning methods. These processes are essential for refining the dataset, removing noise and irrelevant information, and enhancing its usability for further analysis.

Following filtration and cleaning, the processed dataset is then employed to generate a normalized indication. This indication serves as a pivotal component within the framework, facilitating the classification of potential threats as either benign or malicious. For the training of malicious data, the network calls are further divided into subsets. Each subset indicates the factors necessary for categorizing different threats, such as DDoS, MitM, botnet activity, exfiltration, and DNS spoofing attacks. For each category, open-source datasets from Kaggle were downloaded. These datasets represent different network calls focusing on protocols like HTTP, ARP, TCP, FTP, and DNS.

Notwithstanding only the utilization of the pre-existing datasets, several simulations were also conducted. Since, each category follows a different set of protocols, every simulation had to be different, as presented in Table 1.

Cyberthreats	**Network Protocols**
Man-in-the-Middle (MitM) Attack	ARP, DNS, TCP
DNS Spoofing	DNS, QUIC, HTTP
DDoS	UDP, TCP, ICMP, SSDP
Botnet Activity	IRC, P2P, SMTP
Data Exfiltration	HTTP, FTP, DNS

Table 1. Cyberthreats and Network Protocols

In the simulated Man-in-the-Middle (MitM) attack scenario, a controlled environment was created using two virtual machines connected to a bridged network, devoid of external connections. Within this isolated network, one virtual machine assumed the role of the target system, while the other acted as the attacker. The attacker's virtual machine executed a script to identify and capture the IP address of the target machine. The primary objective was to

deceive the target operating system into believing that its requests were directed to the legitimate server. In the figure 9 it is clear that the MAC or the physical address for the server with IP address 192.168.137.1 as listed in the target machine changed before (00-50-56-c0-00-08) and during the attack (00-0c-29-3e-b0-60), clearly representing the breach.

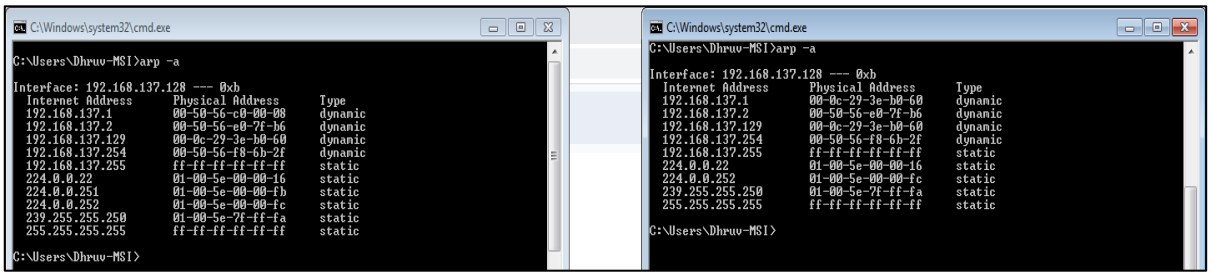

Figure 9. ARP - Mac Address MitM

Through this deception, all data packets transmitted by the target machine were intercepted and monitored within the attacker's operating system, effectively replicating the behavior of a real-world MitM attack. All the logs captured from the simulation were unified and converted to a csv file for further transformation.

In the figure 10, it is clear that the ARP protocol is switched as the attack commences. The attack machine identifies as the server, and the target machine communicates while staying ignorant of the false identification.

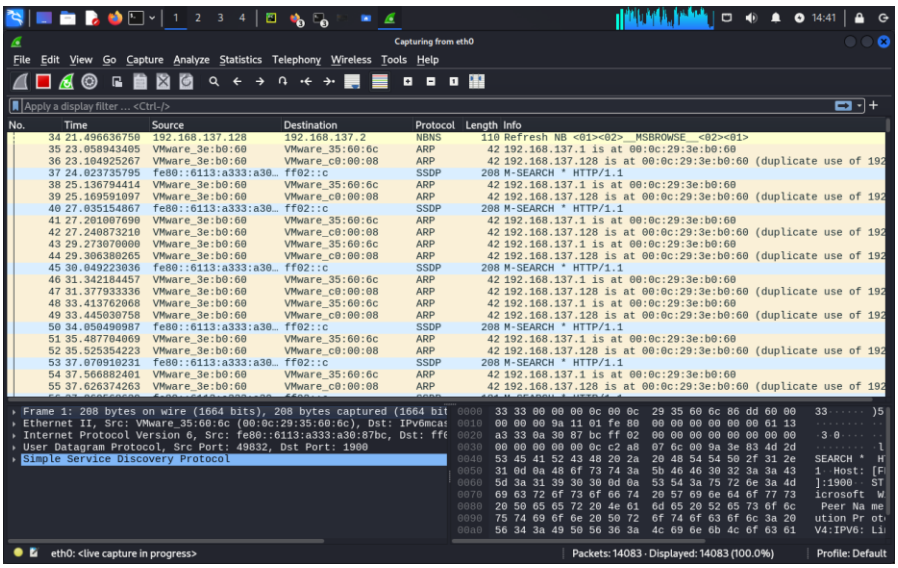

Figure 10. ARP Wireshark - MitM

In order to create DDoS attacks and botnet activity, fake packages were created and sent over the UDP protocol to the targeted host. In this specific scenario, the attacker and target were chosen specifically to be the same in order to recognize patterns in the UDP flow. DDoS attack was carried out by sending more than 100000 requests to the targeted server in order to overload the server until the program crashed, thus denying the services.

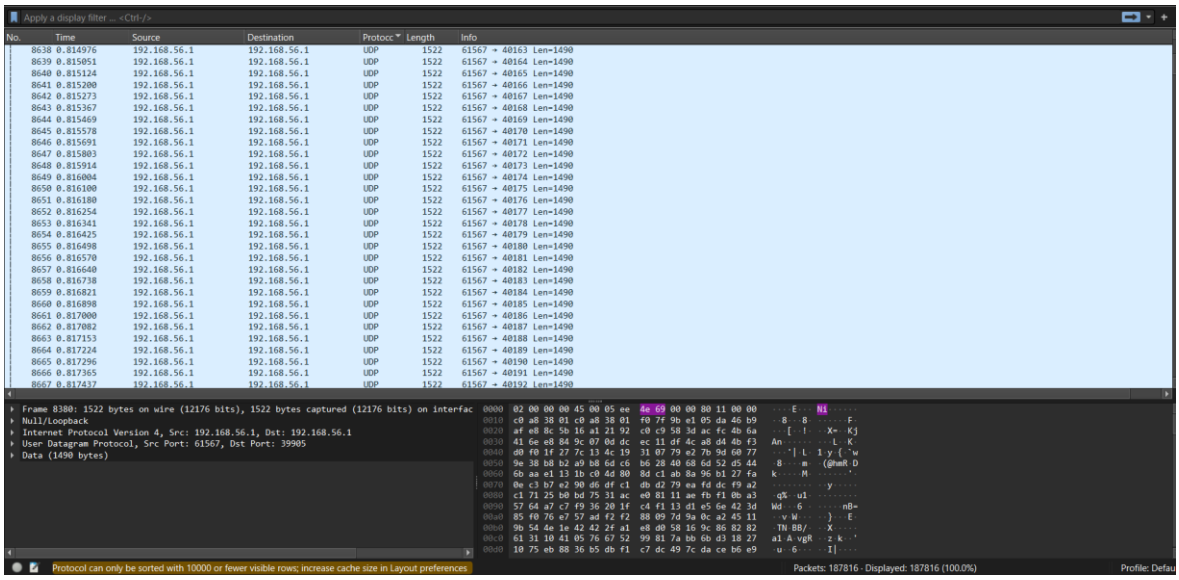

Figure 11. DDoS - Wireshark

The whole network call was captured with Wireshark and the data was stored as a pcap file in order to save all the relevant information regarding the time, source, length, and information from the simulation. Additionally, datasets from threat intelligence feeds and Kaggle were also downloaded to provide more diverse data to further refine and train. Figure 11 shows all the captures for the DDoS attack with the UDP protocol in the feedback loop for the local IP address.

Finally for DNS spoofing, another python script was written. The purpose of the script was to redirect the traffic by manipulating the DNS of the websites and servers from the target machine to the attacker machine. The image shows the network calls accessing the destination IP address but all the traffic was redirected back to the attacker OS. Similar to all the other network datasets, this consisted of the source, timestamps, length, and the protocol.

The figure 12 shows spoofing code written the trick the reached addresses with a temporary spoofed IP address.

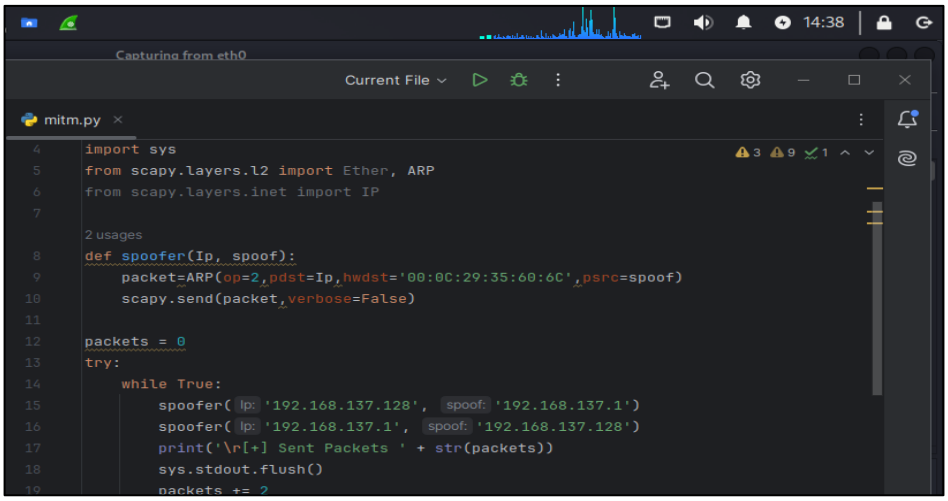

Figure 12. DNS Spoofing - Code

The figure 13 represents the network calls for the spoofed transfers with the QUIC protocol.

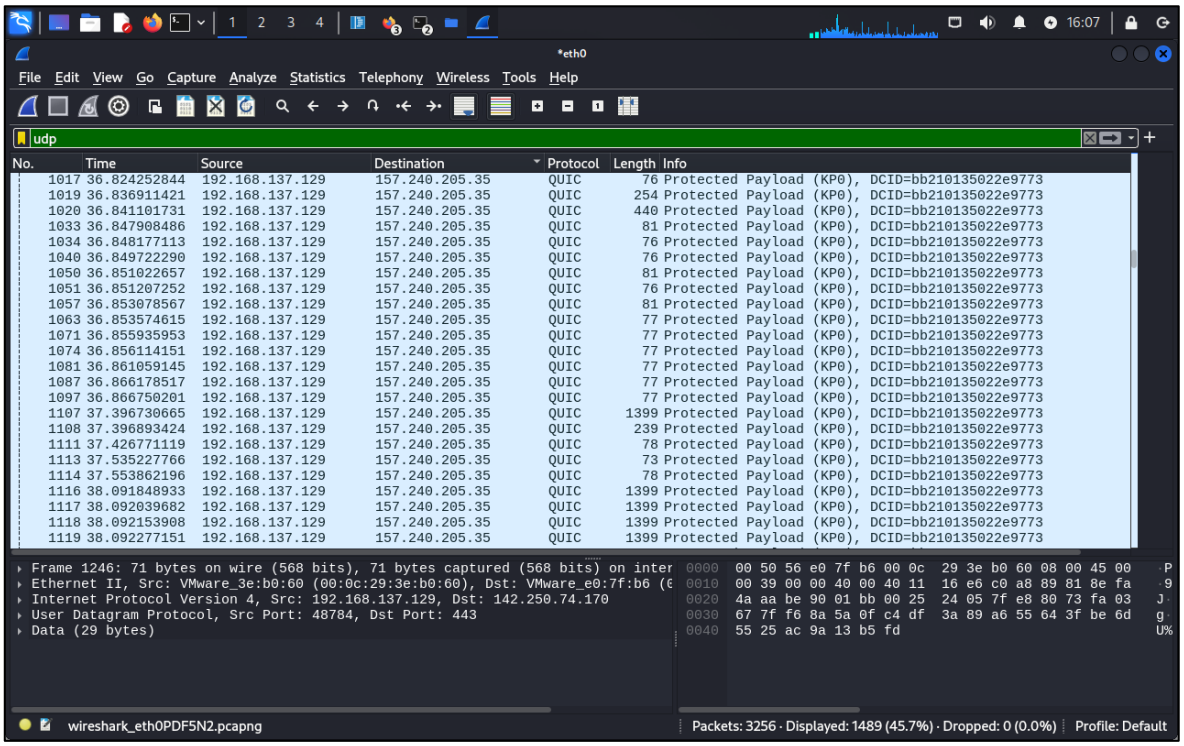

Figure 13. DNS Spoofing - Wireshark Packets

3.1.2 Endpoint Security Logs

Endpoint security logs are records of security-related events and activities that transpire on individual endpoint devices within a network (Hassan et al., 2020). These logs provide very detailed information about different aspects of endpoint security including system events, user activities, and security incidents. These logs are generated by endpoint solutions installed on devices such as laptops, workstations, servers, and mobile devices. To capture the datasets, various logs corresponding to system events, authentication, and security policy enforcement were reviewed. These datasets are significant in recognizing several threats including – malware, data breaches, and unauthorized file system changes.

To generate logs for the training and testing phase, old data stored in the Windows 11 laptop was used. This data comprised logs from Windows Defender, open-source antivirus programs, and certain firewalls. Each log provides the information regarding the event be it benign or a malicious threat. The information contains the product name, detection id, process name, threat id, severity level, path, and category id. Some product names and process names contain specific hashes which are useful in recognizing the pre-existing threats. Additionally, this information is also useful in pattern recognition, threshold adjustments, and consequently quick adaptation of the framework.

Each log is categorized by the event ID which represents the severity of the log – informative or warning. Similar to the network logs, end point security logs are divided into the sets of hundreds for efficient distribution. Figure 14 shows all the logs captured by Windows security, showcasing the security auditing.

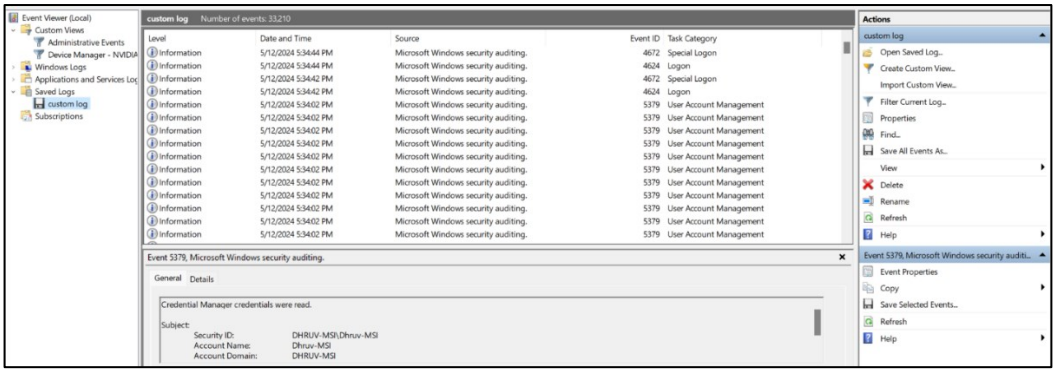

Figure 14. Log Entries – Endpoint Security

The acquisition of malicious datasets involved two distinct methods. Initially, as previously mentioned, endpoint security logs were obtained from open-source cyber threat intelligence feeds and Kaggle. These datasets exhibited similarities to Windows logs and were merged into a cohesive dataset by factoring the severity ID.

For the second approach, a sandbox environment running Windows 7 OS was created on VMWare Workstation. The OS was imbued with up-to-date threat intelligence updates in order to verify current hashes. Several trojans, ransomware, and viruses were deliberately downloaded, each having unique hashes and treated differently based on the target system and their severity level. For unauthorized access attempts resulting in failed authentication, a script was developed to repeatedly submit invalid credentials, triggering logging messages. The recorded information included timestamps, event IDs, user IDs, and the associated processes. Likewise, to simulate high-level security policy violations and potential data breaches, the sandbox OS was linked to the primary network, and a separate computer was employed to attempt breaching firewalls, thereby generating the required logs.

3.1.3 Application Logs and Runtime Behavior

Application logs and runtime behavior data provide valuable insights into the activities and behavior of the application itself. Analyzing these logs can help detect various types of threats related to the application usage such as – insider threats, data leakage, abnormal application behavior (increased resource consumption, unusual network activity, and unexpected errors or crashes), code injections (SQLi and XSS), and privilege escalation (Anderson, 2021).

To get these logs, custom logging code, event tracing and runtime monitoring told were deployed. Each of these mechanisms centralize the logging information, trace the flow of execution, provide real-time visibility into the application activities. The mechanisms followed structured logging techniques to capture logs in a standardized format including key metadata fields such as time stamps, severity level, user ID, process ID, and contextual information. Figure 15 shows the test malware virus download from eicar.com to gather the metadata and intel captured by defender anti-virus.

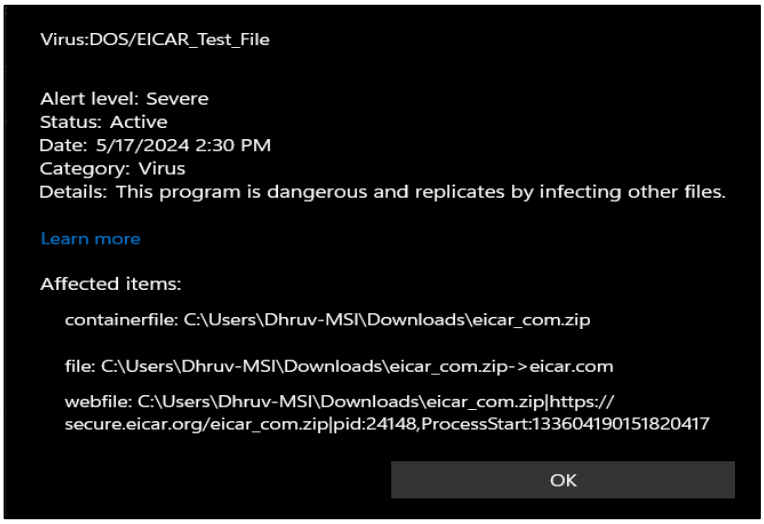

Figure 15. Virus Logs - Windows Defender

For the malicious training datasets, apart from getting information from Kaggle, several simulations were also generated. Firstly, to execute XSS attack, a server with default security and firewall was deployed. This server was linked to a form web application which was susceptible to the user input. Within the entry point, malicious JavaScript code was injected to capture the logs within server calls. The application log files consisted all the sensitive information including timestamps, interactions, and error messages. Similar simulation was generated for SQL injections. But instead of the form data, SQL queries were injected inside the network calls, thereby directly accessing the database. However, the results of the logs were different from the results acquired in the XSS attack simulation.

Finally, in an attempt to get insider threat behavior, various logs with abnormal activities were acquired. Some logs were from pre-existing saved files that captured user interactions with applications and consequently user hours, while other logs were obtained via subtle changes in user application hours by cloning and randomizing the existing data. In the figure, it is clearly captures of the user activity within the system. However, the figure 16 showcases the shutdown of the system, whereas other logs would be useful to capture the specific events within the system.

Startup Time	Last System Event ...	Duration	Shutdown Process	Shutdown Code
4/11/2024 3:52:09 AM	4/11/2024 3:52:21 ...	00:00:11	C:\WINDOWS\system32\winlogon.exe (DHRUV-MSI)	0x000500ff
1/27/2024 6:31:26 PM	1/28/2024 10:26:13...	1 Days + 03:54:47	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
2/14/2024 12:00:29 PM	2/14/2024 12:17:01...	00:16:32	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
2/25/2024 12:46:32 PM	2/25/2024 2:54:18 ...	02:07:46	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
3/4/2024 5:32:10 PM	3/4/2024 6:24:40 PM	00:52:30	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
3/5/2024 12:09:19 PM	3/5/2024 4:26:47 PM	04:17:27	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
3/11/2024 2:56:10 PM	3/11/2024 6:14:17 ...	03:18:06	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
3/13/2024 10:29:45 AM	3/13/2024 12:38:00...	02:08:15	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
3/30/2024 12:41:56 AM	3/30/2024 4:20:45 ...	15:38:49	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
3/30/2024 4:21:21 PM		00:48:08	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
3/31/2024 2:20:12 PM	3/31/2024 8:40:20 ...	06:20:07	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
4/4/2024 3:22:30 PM	4/4/2024 8:05:37 PM	04:43:06	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
4/6/2024 10:41:25 PM	4/6/2024 10:54:00 ...	00:12:35	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
4/11/2024 3:45:17 AM	4/11/2024 3:48:07 ...	00:02:50	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
4/11/2024 3:50:07 AM	4/11/2024 3:51:33 ...	00:01:25	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
4/13/2024 1:14:57 PM	4/13/2024 4:39:35 ...	03:24:37	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
5/10/2024 2:32:07 PM	5/10/2024 4:39:45 ...	02:07:38	C:\Windows\System32\RuntimeBroker.exe (DHRUV-MSI)	0x00000000
2/14/2024 12:17:39 PM		00:00:35	C:\WINDOWS\servicing\TrustedInstaller.exe (DHRUV-MSI)	0x80020003

Figure 16. Runtime Behavior

3.1.4 Identity and Access Management Logs

Identity and access management systems such as Active Directory (AD) and Lightweight Directory Access Protocol (LDAP) are responsible for managing user identities, user authentication, and authorization within an IT infrastructure (Yeh et al., 2002). These management systems verify the identity of users attempting to access resources through various authentication methods like passwords, biometrics, or multi-factored authentication. In addition to this, the management systems also determine the level of access and permissions granted to authenticated users based on predefined policies. Some management logs were gathered from open-source Windows forums, and others were gathered from Kaggle.

3.1.5 Threat Intelligence Feeds

Threat intelligence feeds also play a significant role in enhancing cybersecurity systems by providing important information about emerging threats, attack patterns, and malicious vectors. These feeds enable the systems to stay ahead of evolving cyberthreats (Sun et al., 2023). The two main open-source feeds used to acquire datasets were AlienVault Open Threat Exchange and OpenPhish. Both of these feeds have their own APIs, which were integrated into the application to update the datasets. The reports from these feeds contain several parame-

ters such as IOCs, malware samples, phishing URLs, malicious IP addresses, and other security-related data. Collectively, this information provides a unified methodology for mitigating threats such as data breaches, data leaks, and viruses.

3.2 Data Processing

After acquiring all the datasets, these still need to be cleaned, normalized, filtered, and then finally integrated into their own cohesive datasets. Each dataset was categorized into different classifications based on the extracted features.

3.2.1 Data Cleaning and Normalization

Data cleaning and normalization are the first, yet quite essential steps in preparing the datasets for analysis, ensuring that the data is accurate, consistent, and suitable to use in ML models and analytical processes. The cleaning process begins after merging all the datasets in their respective category – network calls, endpoint security logs, application logs, etc. The basic ideology for all the categories was the same i.e. removing all the duplicate information. This was done to ensure to maximize efficiency and to avoid skewing analysis.

For network traffic data, outliers in the traffic volume, packet size, and transmission rates were identified and moderated to certify correct analysis. Additionally, non-essential traffic such as feedback loops, background noise, and routine system communication was excluded. Afterwards, uniform timestamps across different protocols such as – ARP, DNS, UDP were ensured to facilitate the chronological analysis. However, the main goal for these normalization steps was to standardize the packet size to a common scale across the traffic patterns and to enable comparison across different network segments by modifying the transmission rates and bandwidth usage.

Similarly, endpoint security logs and application logs were checked for privilege escalation activities, high file system changes, and the respective access attempts. All the non-security-related logs and correct authentication events were ignored and the datasets were clean of any maintenance tasks and background processes. After standardizing the uniform

timestamps for all the events, file system changes were processed. File sizes, ownership, modification time, user activity within those systems (HTTP queries or database queries) all were processed and normalized to enable pattern recognition and anomaly detection – either by application behavior or user behavior. Lastly, for IAM logs, resource access events were captured and standardized with uniform time stamps. Each resource was given its own privilege access level depending on the significance to facilitate the detection of un-authorization attempts or abnormal resource consumption.

3.2.2 Feature Extraction

After the cleaning and normalization of data, certain features needed to be extracted from the datasets before proceeding with training of the said datasets with ML algorithms. Majority of the information was already present in the network call datasets in the format of pcap files. For the other logs, extraction methods were applied to retrieve the process names, process IDs, file paths, user access activities, hash values and signatures, resource IDs, SQL strings etc. All the required features corresponding to their datasets are listed in the table 2.

Dataset Types	**Extracted Features**
Network Datasets	Network Calls, Date & Time Stamps, Length, Log Information, Source, Protocol, and Destination.
Endpoint Logs	Level, Data & Time, Source, Resource ID, Event ID, Task Category, and Severity
Application Logs	Duration, Last Time Access, Process, Process Codes, Hashes, File Paths
Identity and Access Management Logs	Access Identities, User Signatures, Date & Time, Resource Accessed, and Level

Table 2. Extracted Features

3.3 Data Usage

After feature extraction and categorization, each dataset is ready to be trained with ML algorithms and techniques.

3.3.1 Data Training

Different ML algorithms and AI techniques were applied to train each dataset category. The techniques ranged from supervised learning and unsupervised learning to sequence modelling and rule-based reinforcement learning. All the datasets were trained on their respective algorithms to finalize the threshold framework. This framework provided a way to dynamically adjust itself based on the receiving data and policy thresholds, thereby focusing on mitigating threats effectively and efficiently.

The table 3 shows what techniques were utilized for which category of the datasets. For instance, SOMs excel in unsupervised learning scenarios, allowing for the clustering and representation of network call datasets efficiently. In contrast, Graph Neural Networks (GNNs), RNNs and Long Short-Term Memory (LSTM) networks are well suited for temporal analysis, thereby capturing dependencies over time and modelling complex patterns present in application and endpoint security logs. For behavior pattern analysis, Logistic Regression and SVMs were used for classification tasks as the output data is categorial. Similarly for hash patterns and threshold functionality, techniques like dynamic learning and continuation adaption were implemented.

Datasets	**Threats Detected & Mitigated**	**ML Algorithms**
Network Calls	DDoS, MitM, Data Breach, DNS Spoofing, Botnet	SOMs and GNNs
Application Logs	SQLi, XSS, Privilege Escalation	LSTM, Attention Mechanisms, and GNNs

Endpoint Security Logs	Malware, Unauthorized Access	RNNs & LSTM
IAM Logs	Behavior pattern, Administrative Policies	Logistic Regression, SVM, and Isolation Forest
Intelligence Feeds	Current Malware Hashes, Network Data	Dynamic Learning & Continuous Adaptation

Table 3. ML Algorithms - Datasets

3.3.2 Data Testing and Validation

After training, in this phase, the datasets undergo a critical procedure of testing to evaluate the performance accurately. The datasets were split into two subsets – one for training of the model as mentioned before, and the other was reserved for the testing phase, ensuring that the model's effectiveness can be rigorously assessed on the unseen data. Additionally, cross-validation techniques such as time series validation, k-fold cross-validation, and leave-one-out cross-validation, were employed to further enhance the robustness of the evaluation process. Each testing and training phase had more than 1000 samples for every subset within the categorial datasets.

3.3.3 Continuous Monitoring System

In the continuous monitoring approach, the focus shifts towards the operation deployment of the real-time anomaly detection system that could potentially monitor the incoming data streams for abnormal patterns. The streams were set up using Apache Flink to process different sources including network traffic, endpoint logs, and application activity. This monitoring approach also helped in adjust the thresholds required in the framework, based on the observed characteristics of the incoming streams. By utilizing the reinforcement learning algorithms, the detection system was formulated to dynamically allocate the resources and its overall impact. While monitoring the streams, a feedback mechanism was also employed to manual detect any false positives, and the threshold system was improved upon in order to refine the detection system.

4 Research Questions

Based on the data gathered from various sources, as stated in chapter 3, the study identifies four important research problems in the field of adaptive anomaly detection with modern AI methodologies. The research is based on these questions and are addressed below, exploring all the factors of identifying and mitigating respective cyberthreats using behavior analysis, anomalous detection, fine-tuning existing techniques, optimizing real-time strategies, and merging traditional cybersecurity with AI techniques.

4.1 AI-Driven Effective Techniques

The first research questions is - *What are the most effective AI-driven techniques and frameworks for real-time prevention of data leaks and breaches, independent of supplementary software layers like firewalls and antivirus, to deliver a streamlined and resilient security solution?*

The system designed specifically to prevent real-time data leaks and breaches requires a combination of multiple ML models. While firewalls, and antivirus programs have decent mitigation strategies based on the severity level, they still compromise either effectiveness or efficiency. When being effective, several data streams are not even reachable by and to the user, thus reducing the efficiency and vice-versa. Different techniques such as behavior analytics, security solutions, and web application firewalls can be implemented to prevent these threats. Data breaches and leaks are commonly associated with insider threats, data exfiltration, MitM attacks, malwares (trojans & spywares), and SQL injections.

4.1.1 Behavior Analysis

Since all of the threats mentioned above follow specific patterns of network calls or hashes, behavior analytics is the best technique to deploy to scan and identify the abnormal patterns in network calls, user activity, and application logs. Based on the chronological gathered data and research, techniques like self-organizing maps, k-means clustering, and DBSAN

have proven to be fundamentally crucial to identify any behavior patterns. This technique is especially useful in detecting insider threats.

- **K-means** clustering algorithm works by partitioning the datasets into pre-determined clusters based on the similarity. This algorithm is effective for grouping similar network traffic patterns together. This helps in identifying communication behaviors such as - regular data transfers, protocol violations, or unusual spikes in activity. For user activity logs, similar behavior users can be grouped together which leads in detecting common usage patterns. The usage patterns include login times, access frequency, and application usage trends. To monitor unexpected user interactions and frequent errors, the application logs can be segmented based on the system interactions or usage patterns which can ultimately detect the abnormalities in the data.

- **DBSCAN** is a density-based clustering algorithm that can identify the clusters of various shapes, making it useful for capturing the complex and irregular nature of any abnormal behavior (Zhang et al., 2017). These identified clusters can help in detecting patterns related to port scanning, which may exhibit unusual density. Similarly, this algorithm can detect outliers and sparse regions in the user activity and application logs to recognize the privilege escalation, repeated access attempts, abnormal resource usage, and unauthorized attempts.

- **Self-organizing maps** are another way to detect anomalies. There are quite useful as they preserve the topological properties of the input data and visualize the data in a low-dimensional map. Therefore, the SOMs are quite effective in highlighting the clusters of anomalous behavior. Similarly, user data and application logs can be presented in the two-dimensional map to identify unusual characteristics and potential security incidents.

The figure 17 shows the cycle of behavior analysis using the above-mentioned methodologies and algorithms to get specific data factors such as regular transfers, access fre-

quencies, login times, privilege escalation, resource usage, system interactions, and application patterns. All these combined lead to understanding the user or application's behavior which can be used for further analysis in a cohesive framework.

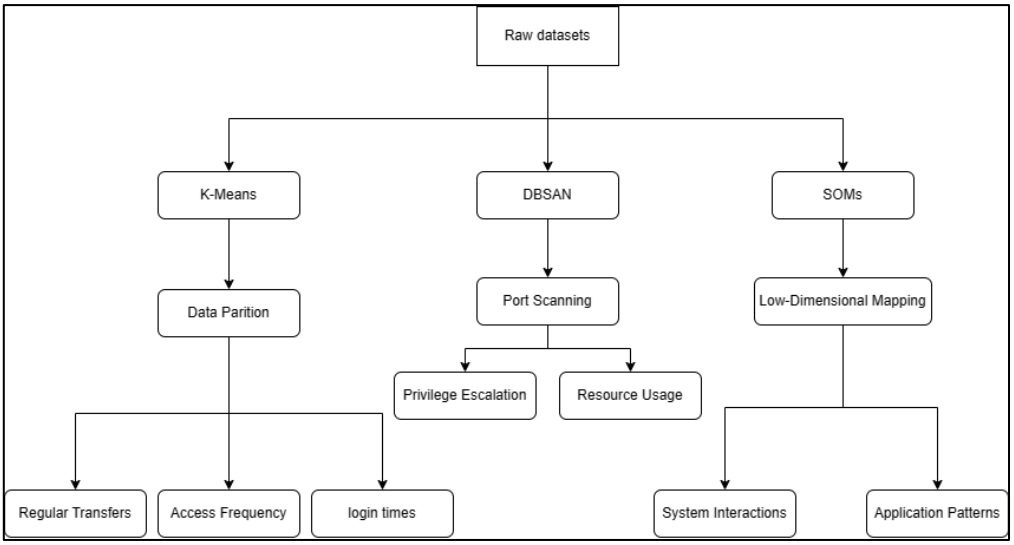

Figure 17. Behavior Analysis Cycle

4.1.2 Anomalous Detection

ML algorithms such as – CNNs, RNNs, and autoencoders are effective in the context of analyzing sequential and spatial data (Canizo et al., 2019). These algorithms offer valuable insights for identifying abnormal patterns and potential security threats.

- **RNNs** are well-suited for sequential analysis and detect anomalies in streamlined data for network calls and activity logs. RNNs can capture dependencies and temporal relationships. This allows the algorithm to identify unusual sequences of events that deviate from expected patterns of login attempts, high network packets, etc.

- **CNNs** excel at analyzing spatial data making the algorithm effective in detecting anomalies where the spatial relationships between data points are important. CNNs can extract relevant features from data representations, such as packet payloads or image snapshots of application behavior, to identify deviations.

- Like the CNNs and RNNs, **Autoencoders** offer a versatile approach to detect anomalies by learning compressed representations of input data and reconstructing it with minimal error. Deviations between the original data and reconstructed data indicate the potential anomalies like identifying unusual patterns in packet payloads or in application logs by detecting unexpected sequences of user actions or system events.

The figure 18 shows the cycle of anomalous detection with RNNs, CNNs, and Autoencoders to capture login sequences, network usage, temporal dependencies, application behavior, packet payloads, user actions, and system events.

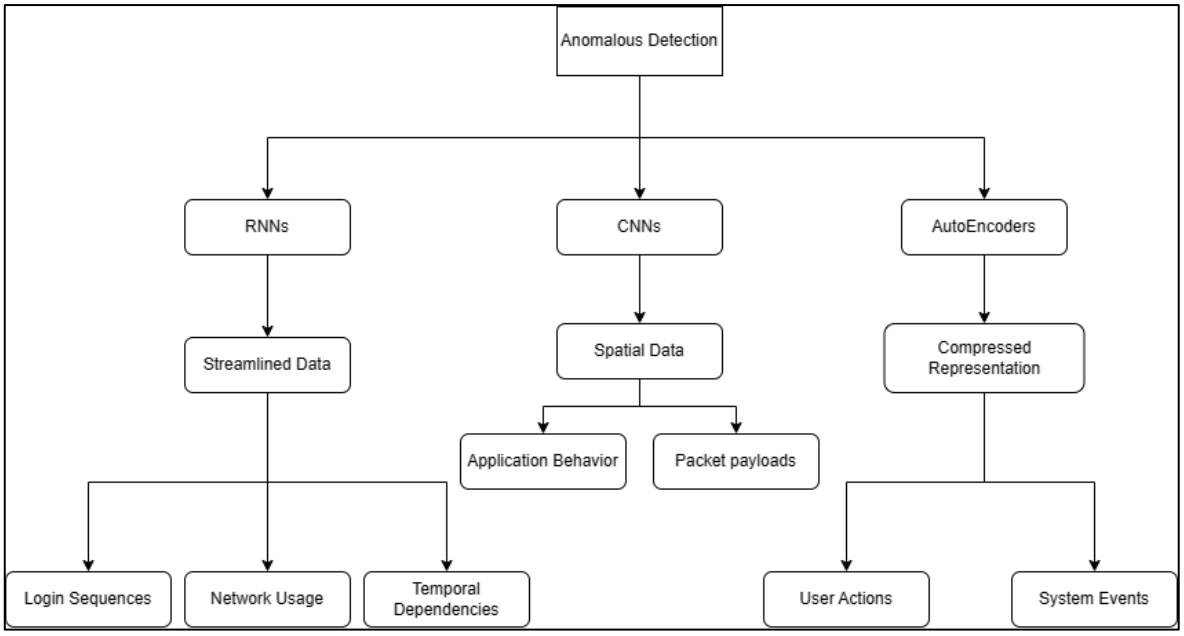

Figure 18. Anomalous Detection Cycle

4.1.3 Endpoint Security Solutions

These solutions utilize algorithms such as SVMs and decision trees employed in conjunction with ensemble methods to safeguard devices and networks by identifying threats including malware infections, insider threats, and unauthorized access attempts and mitigate them accordingly.

- **SVMs** excel at separating different classes of data by finding optimal hyperplane than maximizes the margin between them. When applied to network traffic data, SVMs can distinguish between normal and anomalous traffic by learning underlying patterns in the data. Similarly, this algorithm can classify user and application activities as either normal or malicious based on the features extracted from the logs such as login times, IP addresses, API calls, request parameters, response codes, and accessed resources and consequently flag abnormal activities.

- **Decision trees**, on the other hand, provide a transform and interpretable approach by recursively partition the data based on the network features and resources. (Vu et al., 2019) These algorithms can create simple decision rules based on the attributes and can contribute to anomaly detection by identifying sequences of malicious behavior, such as privilege escalation, unexpected error rates, suspicious API usage, and unusual data access patterns. Figure 19 represents in the anomalous decision trees cycle that could be employed in endpoints to get the respective information.

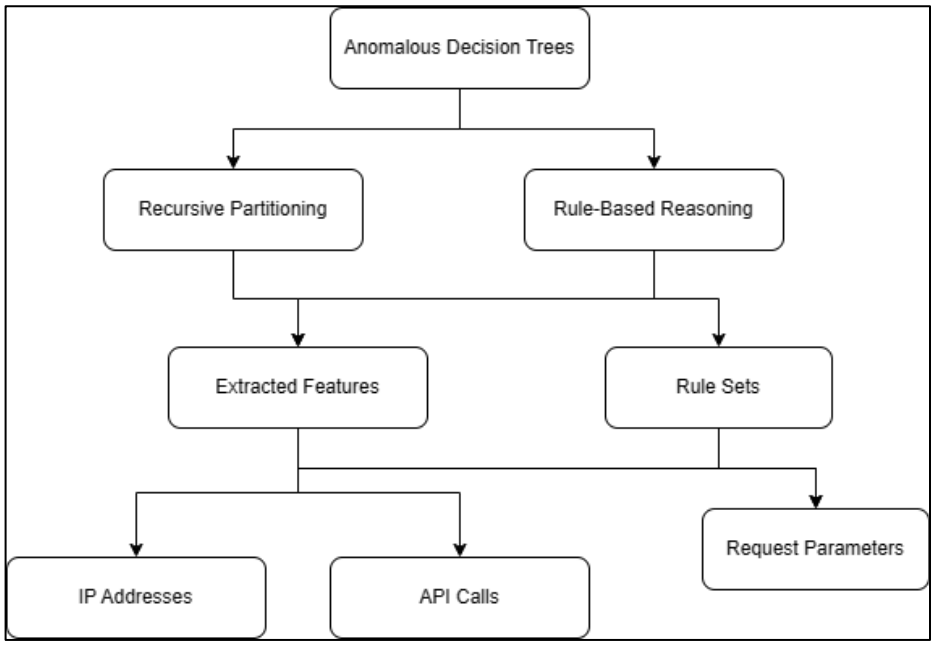

Figure 19. Anomalous Decision trees

4.1.4 Web Applications Firewalls

Web application firewalls play a critical role in safeguarding web applications against a variety of threats including SQL injections, MitM attacks, and vulnerability targets. Integrating algorithms such as LSTM networks and transformer architectures can vastly enhance the detection and mitigating capabilities of the application.

- **Transformer architectures** process input sequences in parallel formations, making them highly efficient for capturing long range dependencies. The architecture consists of encoders and decoder layers. Each layer has multi-head self-attention mechanisms and feedforward neural networks. This allows the algorithm to weigh the importance of different factors within the input data stream, thus attending to the relevant parts and identifying subtle pattern shifts in logs and network calls.

- **LSTM** networks are a type of RNN designed to overcome the gradient problem, allowing the algorithm to retain the sequential information. Consisting of three gates, LSTM regulates the flow of information into, out of and within the memory cell. Utilizing the functionality, the algorithm can capture long-range dependencies and subtle changes by analyzing the patterns overtime. This is quite effective for identifying anomalies such as DNS spoofing and insider threats.

4.2 Real-Time Anomaly Detection Optimization Strategies

The second research question is based on optimizing the real-time detection for efficiency and effectiveness - *What strategies can be employed to optimize real-time anomaly detection algorithms for autonomous identification and mitigation of cyberthreats, thereby reducing reliance on manual intervention and elevating the efficacy of cybersecurity measures?*

To optimize the real-time defense mechanisms, many approaches could be applied to adapt to real-time threats and then mitigate them accordingly. There are several factors that contribute to the autonomous adaptability of the strategies. These factors can reduce the reliance on manual intervention overall. The techniques that can be within the framework to efficiently optimize the detection algorithms are given in the subpoints of 4.2.

4.2.1 Contextual Information Integration

This approach involves using advanced data merging and fusion techniques to combine diverse data sources. The data sources include all the behavior patterns, topological maps, and system configuration logs. This integration is achieved through feature engineering, where relevant attributes from the data streams are extracted, transformed, and merged to create a unified representation of system behavior (Com & Hinton, 2008). In the case of optimizing anomalous detection, techniques such as Principal Component Analysis (PCA) and t-distributed Stochastic Neighbor Embedding (t-SNE) can be utilized to reduce dimensionality of access patterns, network flows, application logs. This also preserves important contextual information. Contextual information by itself can include user feedback, additional factor integration, etc.

4.2.2 Dynamic Threshold Mechanisms

Dynamic thresholds use statical methodologies to adapt and adjust the detection thresholds based on the real-time data and the contextual information. Techniques such as Exponentially Weighted Moving Averages (EMWA) and Autoregressive Integrated Moving Average (ARIMA) models can help in forecasting expected values and predict anomalous deviations in the real-time (Raza et al., 2015). This leads to analyzing patterns and monitoring key performance indicators which help in dynamically adjusting the factors and thresholds to accommodate the changing data streams. For instance, if the model detects changes in ARP protocol in network calls, then the thresholds would be adjusted to factor in the DNS spoofing or insider threats. This optimization approach is compatible with algorithms like Gaussian Mixture Models (GMMs) and Hidden Markov Models (HMMs). Because these models observe and learn from data distributions within the logs, calls, and events.

4.2.3 Continuous Learning and Adaptation

Continuous learning mechanisms enable the anomaly detection algorithms to evolve over time by updating the models based on new data and feedback, as suggested in dynamic

thresholds and contextual integration. Learning algorithms such as Stochastic Gradient Descent (SGD) and adaptive learning methods update the model parameters in response to the incoming data stream (Ziyin et al., 2021). This is additionally backed up with reinforcement learning algorithms such as Q-learning and deep Q-networks. To optimize the adaptive nature of anomalous detection, these reinforcement learning algorithms can prove to be beneficial by adjusting and refining the detection policies based on the observed outcomes and malicious threats.

4.2.4 Automated Response Actions

Following the continuous learning, automated response actions enable the response workflows to counter detected anomalies in real-time by removing or mitigating them. These actions encompass predefined mitigation measures and are triggered by rule-based decision engines or ML classifiers, thereby reducing the manual effort to counter the threats (Paschke & Kozlenkov, 2009). By defining and processing the indicators of compromise, methods like rule-based expert systems, decision trees, and security orchestrated response solutions can facilitate the integration of the automated responses countering any threat while processing data streams in real-time, thereby optimizing the solutions to the detected threats.

4.3 Fine-Tuning Self Updating and Learning AI Capabilities

The third question is related to the self-adaptation techniques and mechanisms within AI-drive cybersecurity systems - *How can computing methodologies, be adapted and fine-tuned to enable self-updating and learning capabilities within AI-driven cybersecurity systems, ensuring continuous adaptation to emerging threats and vulnerabilities?*

Creating self-updating and learning capabilities in any system involves many factors. This is particularly true for adaptive anomaly detection frameworks, where repeated learning and transformation are complex processes. One of the most critical aspects of fine-tuning these AI-driven systems is gathering, processing, and monitoring data. These systems increase their ability to detect and neutralize new emerging threats by analyzing the gathered data on

a constant basis. This technique allows the systems to keep ahead of possible vulnerabilities, thus making them more effective.

4.3.1 Natural Language Processing

Natural Language Processing (NLP) is an important aspect to ensure the continuous adaptation to emerging threats. This algorithm extracts and evaluate the unstructured textual information from a variety of data sources. These data sources include security reports, threat advisories, research papers, and intelligence feeds. NLP systems can grasp the meaning and the context behind the information and can achieve the deep understanding of security related content. This allows the systems to automatically update their internal knowledge bases with most recent threat intelligence and crucial insights. (Mireshghallah et al., 2022).

4.3.2 Continuous Monitoring and Feedback

Continuous monitoring and feedback loops are critical in maintaining the adaptability of the cyber systems. After processing the latest information, real-time monitoring systems can be deployed. These monitoring systems evaluate the most recent data and instantly discover any anomalies or variations which allows for quick preventive responses. Feedback loops are also significant since they collect useful information from security events and other automated systems. This constant flow of information aids the system in refinement of algorithms. This refinement is based on new data and feedback information which allows he system to adapt dynamically.

4.3.3 Dynamic Updating and Re-training

Dynamic updating and re-training are essential for AI systems. ML models are able to adjust to incoming data by drawing on previously collected feedback. Techniques such as – online learning and incremental learning enable new information to be seamlessly integrated into existing models. This shortens the time between threat detection and respective response. However, if the model becomes significantly old or erroneous, re-training methods with updated datasets can be applied to restore the overall effectiveness.

4.3.4 Adaptive Decision-Making Processes

Adaptive decision-making systems can play an important role by responding to the threats based on real-time analysis. These systems can make timely and well-informed decisions based on the threat severity and impact assessments. The decision support systems take many factors into consideration before formulating effective strategies. The factors include organizational priorities, impact level, threat target and resource availability. These strategies lower the need for manual response calls and makes the system more responsiveness. These processes can also be automated so that the systems can quickly respond to the emerging threats.

Figure 20 shows the fine-tuning techniques and adaptive process cycle.

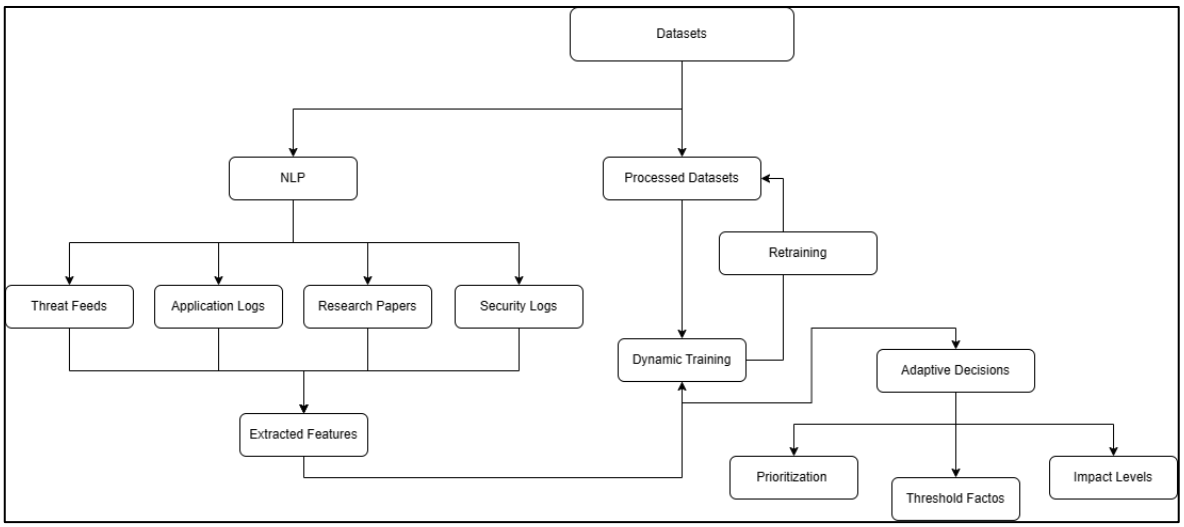

Figure 20. Fine-Tuning & Adaptive Process

4.4 Merging Traditional Cybersecurity with AI-Driven Techniques

The last research question focuses on the approaches to merge the traditional cybersecurity tools and processes with AI-driven techniques and ML algorithms - *What innovative approaches can be employed to integrate AI-driven anomaly detection techniques with traditional cybersecurity tools and processes, fostering synergy and enhancing overall threat detection and response capabilities in dynamic and heterogeneous computing environments?*

The main advantage of merging the traditional cybersecurity measures with AI and ML is their combined strength. Traditional methods provide a solid foundation for defending against known threats and ensure compliance with the privacy and industry regulations. Meanwhile, AI-driven techniques enhance the overall capabilities by offering additional addons. These addons include predictive analytics, real-time threat detection, automated response functions, false positives reduction, and provision of actionable insights for security analysis.

The main approach to achieve this integration is the development of a hybrid detection system. These systems combine the signature-based, rule-based, and ML based anomaly detection techniques. This enables the systems to minimize false positives and benign warnings, generate fresh dynamic information for emerging threats, and provide a full picture of all the hashes and logs linked to current threats. Another strategy involves the integration of Security Information and Event Management (SIEM) systems (Ban et al., 2023). These platforms can uncover the security events, detect anomalies and generate alerts more effectively. This consequently helps the system to identify subtle as well as complex attack patterns that evade the traditional detection methods.

In the similar fashion, AI can help Endpoint Detection and Response (EDR) systems by monitoring the endpoint activity. The AI extension can help to detect suspicious behavior and respond in real-time. Endpoints are often the source of cyberthreats; therefore, it is necessary to enhance security to protect critical assets and sensitive data. Furthermore, threat detection capabilities can be greatly improved by AI-driven Network Intrusion Detection Systems (NIDS) and Intrusion Prevention Systems (IPS) (Dahiya et al., 2021). ML algorithms can vastly boost efficiency, efficacy, and accuracy in traditional systems by evaluating different patterns, datasets, and attack vectors. This can overall result in proactive threat mitigation and lowers the risks of successful attacks.

5 Practical Implementation

In order to develop a framework that recognizes and respectively identifies the information corresponding to multiple cyberthreats, all the trained data (from chapter 3), was used to create a threshold mechanism. The threshold mechanism works on different factors. These factors consider network calls, hashes, patterns, and adjusts its parameters to identify the possible matches from a multitude of cyberthreats.

5.1 Network Threats

Since network threats conceptually operate on similar principles, the thresholds used to identify these threats were consistent across various types. These thresholds included User and Entity Behavior Analytics (UEBA), data transfer volume, anomalous traffic patterns, alert events, and endpoint security events. The parameters were adjusted and prioritized based on the specific type of threat detected. The datasets were broken down into multiple sets, due to the amount of records present. The threat detected in graphical representation is highlighted by the red color (Sample data used in figure 21).

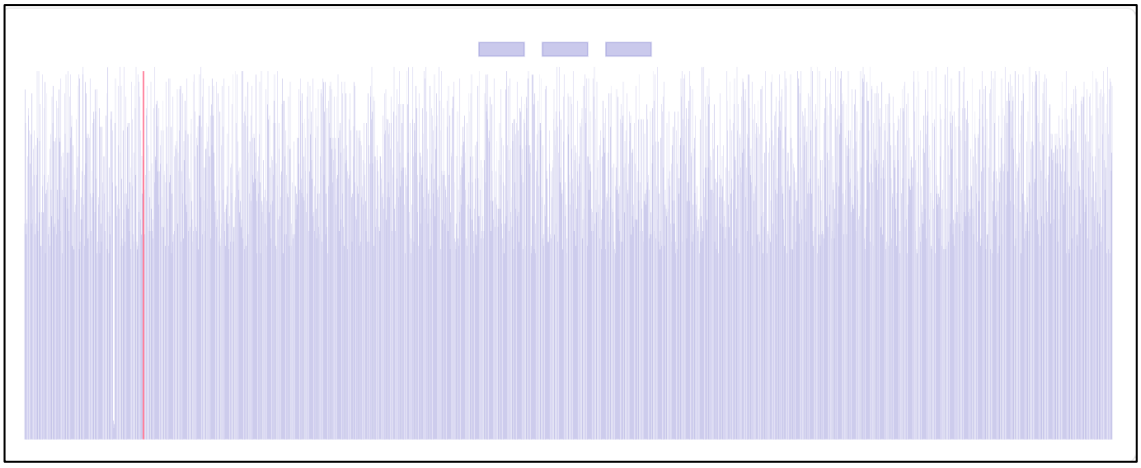

Figure 21. Sample Data – Threat Indication Graph

Threats such as DDoS, MitM, data exfiltration, data breaches, and insider threats share common characteristics. They operate on specific protocols, originate from identifiable sources, target particular destinations, and exhibit distinct patterns. Due to these shared attributes, it

becomes practical to analyze these threats and implement a unified model to identify each type of threat from streamlined data. By utilizing this grouping of these commonalities, security systems can more effectively detect and mitigate diverse network threats through a cohesive and comprehensive approach.

For threats such as phishing, insider threats, and data breaches, the prioritization shifts towards UEBA, security events, data transfer volume, alert events, and finally traffic patterns. UEBA is prioritized because it helps analyze compromised user credentials, providing insight into security event patterns. Next, data transfer volume is considered to narrow down the type of threat. This progression continues with alert events and traffic pattern analysis, ultimately determining if the threat is phishing, an insider attack, or a data breach.

Table 4 presents the prioritization levels of set thresholds for identifying and consequently mitigating network threats. These are based on the techniques discussed in chapter 4 which relate to analyzing, optimizing, and fine-tuning algorithms.

Cyberthreats	**Priority Thresholds**
Phishing, Insider Threats, Data Breaches	1. UEBA 2. Endpoint Security Events 3. Data Transfer Volume 4. Alert Events 5. Traffic Patterns
DDoS, DOS	1. Data Transfer Volume 2. Traffic Patterns 3. Alert Events 4. UEBA 5. Endpoint Security Events
MitM, Data Leaks	1. Alert Events 2. UEBA 3. Traffic Patterns 4. Endpoint Security Events

	5. Data Transfer Volume

Table 4. Network Priority Thresholds

The initial stage involved deploying the framework onto a server, serving as a platform for uploading files, preferably in CSV format. These files typically contain structured data packets sourced from various channels such as network calls, application logs, runtime environment details, and endpoint security logs. The framework processes and analyzes this data to derive insights and execute diverse operations. During the initial deployment, a CSV file containing DDoS attack logs was uploaded to the server, triggering the real-time generation of a graphical representation of network calls. The resulting snapshot, depicted in Figure 22, illustrates the graph generated from the CSV file. Upon detection of any threat in the logs, it undergoes scrutiny based on several factors, which are employed in the threshold mechanism and further examined within the logs.

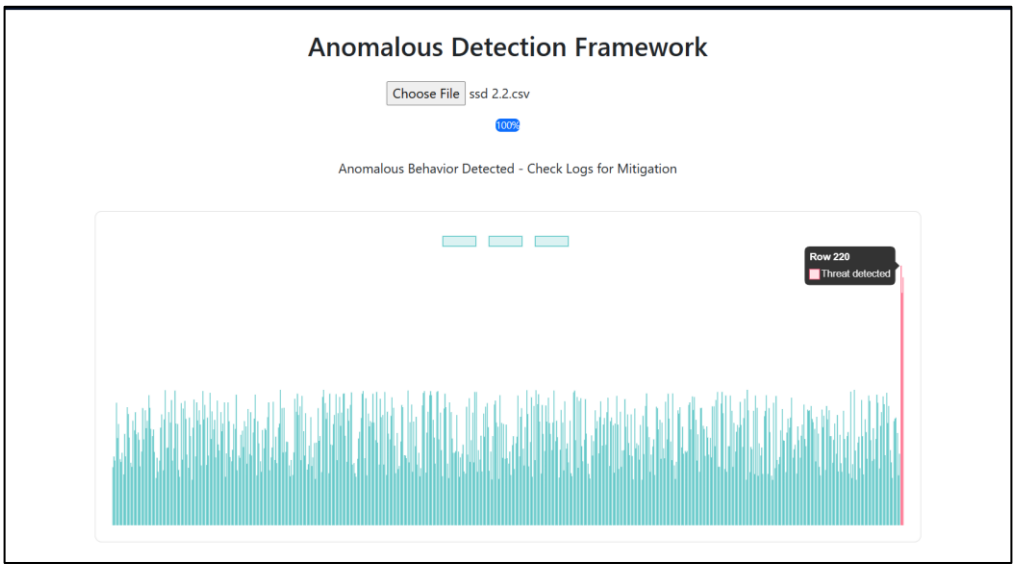

Figure 22. Anomalous Detection Framework Graph

The logs capture information specific to the type of threat detected. In the case of a DDoS attack, which manifests as a network call, key factors such as source, destination, protocol,

and length are logged before the detection process stops. Figure 23 illustrates the call recorded during 200 seconds, with the source and destination – both originating from local environment.

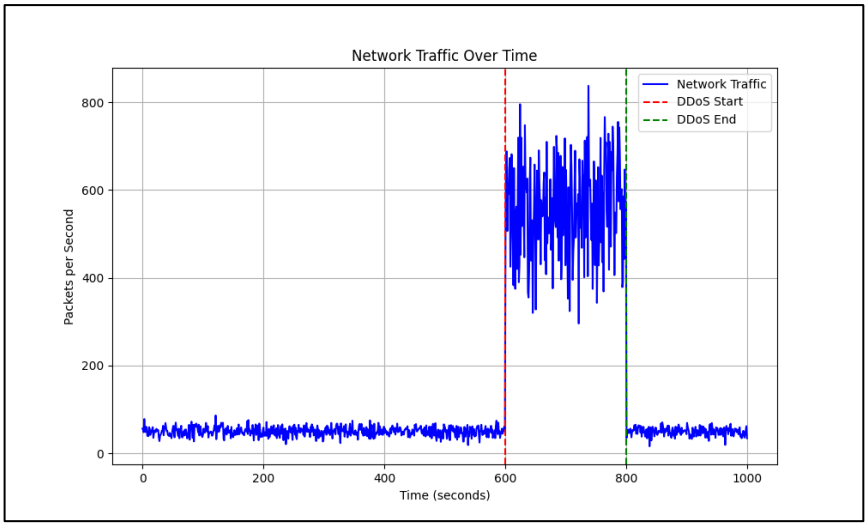

Figure 23. Network Traffic Data Graph

Figure 24 presents the log of the detected network attack. The framework automatically generates a report including multiple parameters for each log. In this case, the threat ID is automatically generated by the framework and stored for future association with similar risks, aiding in optimization.

Parameter	Value
Threat ID	chsFD
Severity Level	3
Attack Type	DoS/DDoS
Source	Origin
Protocol	TCP
Event ID	5149
Packets / Second	380-820
Packets Dropped	472
Destination	localhost
Duration	200

Figure 24. Framework Threat Log – DDoS

The severity level, indicating the threat's intensity, is also recorded and stored in the framework database. The attack type is determined through several key steps, narrowing down possible outcomes, and is then associated with the threat ID. Additional key parameters related to the DDoS attack, such as packet rate, source, destination, protocol, and duration, are also collected. The framework utilizes a relational database created with MySQL. This database facilitates information retrieval and the generation of additional network datasets for model retraining. This process enhances the model's resilience and effectiveness, while also optimizing its efficiency.

5.2 Application-Level Threats

Similar to network call threats, application-level threats also follow some common principles. These principles are effective to identify all the application level threats such as SQLi, XSS, and privilege escalation. These thresholds act as a trigger for the defensive actions taken by the framework to analyze the threats and then mitigate them. All these threats can be associated with unusual input patterns, suspicious user actions, abnormal script behavior, and unexpected outputs.

For instance, to identify SQLi threats, certain thresholds such as unusual input patterns are prioritized over the others (access patterns). Since SQLi attacks often involve injecting malicious SQL code into input fields, therefore monitoring for unusual input patterns helps detect attempts to exploit vulnerabilities in input validation. Following this, abnormal script behavior is considered to be the next threshold which defines the query behavior such as unusual syntax or excessive use of wildcard characters. This is continued by other factors like unexpected output or access patterns, until the threat is detected and properly identified.

Table 5 presents the prioritization levels of set thresholds for identifying and consequently mitigating application-level threats. The priority thresholds were gathered from the research questions discussed in chapter 4.

Cyberthreats	Priority Thresholds
SQL Injections	1. Unusual Inputs 2. Abnormal Script Behavior 3. Unexpected Output 4. Suspicious Application Activity 5. Access Patterns
Cross-Site Scripting	1. Unexpected Output 2. Unusual Inputs 3. Suspicious Application Activity 4. Abnormal Script Behavior 5. Access Patterns
Privilege Escalation	1. Suspicious Application Activity 2. Access Patterns 3. Unusual Inputs 4. Abnormal Script Behavior 5. Unexpected Output

Table 5. Application-Level Priority Thresholds

Similar to the network stage, the file containing the irregular application logs, specifically code injections, was uploaded to the detection framework, and subsequently the logs were generated. The file itself was recorded and contained key information like the timestamps, user IP, request method, requested URL, request body, and user agent. The resulting factors were accessed by the framework which adjusted its threshold parameters and narrowed down the search by understanding the request body parameter, and in this case, the attack was identified as SQLi. Figure 25 shows the threat detected at row 58 – which is the application data log containing the code injection.

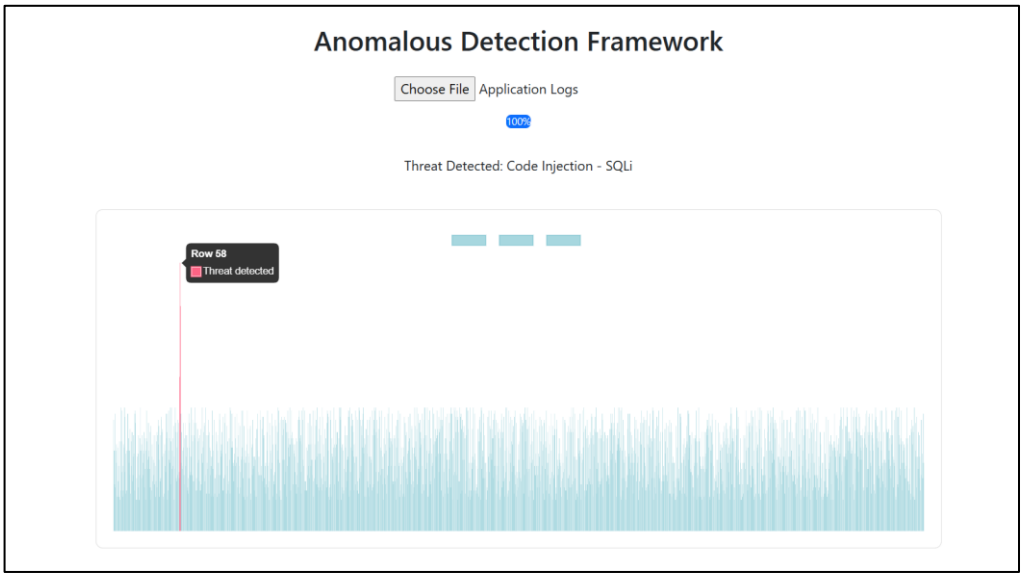

Figure 25. Graph - SQLi Threat Detection

Figure 26 illustrates the distribution of various factors related to SQL query, including the length, frequency of special characters, and error rate during the execution. Each line represents one of the factors across multiple samples of the SQL injection logs. The query files are executed on the server, and when the thresholds are adjusted, the outcome is presented in a JSON format, which then is used to create the factor graph.

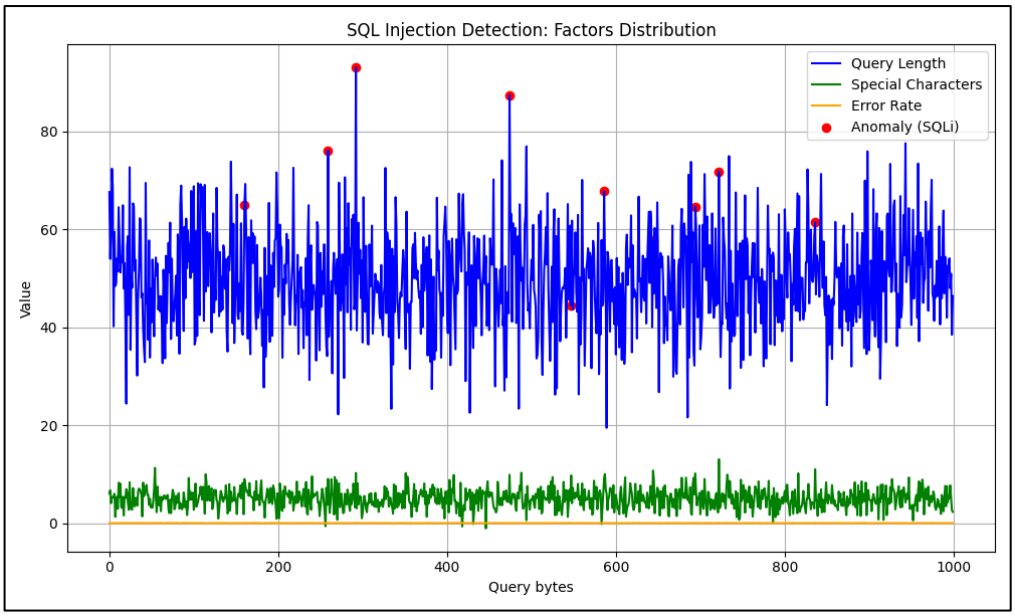

Figure 26. SQL Injection - Factor Distribution

Figure 27 presents the log of the detected SQLi attack. The generated parameters include the threat ID, severity level, attack type, source IP, destination address, database information, SQL query, and response code for the query.

Parameter	Value
Threat ID	GOGZr
Severity Level	4
Attack Type	SQL Injection
Source IP	192.168.1.56
Destination Address	mysql://192.168.1.56:8888/users
Database	users
SQL Query	SELECT * FROM users WHERE username = 'admin' AND password = 'admin' OR 1=1;
Response Code	200

Figure 27. SQLi - Threat Log

5.3 Endpoint Security Threats

Finally, following the same categorizing principles, endpoint security threats are also classified with similar threshold factors. All these factors and the respective combinations facilitate identifying threats like malwares, endpoint breaches, and unauthorized accesses. Corresponding to these threats, associated thresholds are as follows – unauthorized process execution, unusual registry modification, user account permission change, and access to sensitive files or directories.

Malware infections served as the primary focus for implementing endpoint security thresholds within the framework. Since most malware threats involve the execution of malicious code, monitoring unauthorized process executions proceeds as the utmost priority. Subsequently, monitoring access to sensitive directories aids in promptly identifying these threats and responding with appropriate mitigation measures. This priority sequence finalizes over verifying user account permissions, which hold a lower priority level in the context of malware infections.

As described in chapter 4, endpoint security solutions can have multitude of factors. Table 6 presents the prioritization levels of set thresholds.

Cyberthreats	Priority Thresholds
Malware Threats (Virus, Trojan, Spyware, Bot, Adware, etc.)	1. Unauthorized Process Execution 2. Access to Sensitive Directory 3. Unusual Registry Modification 4. User Account Permission Change
Endpoint Breach	1. Access to Sensitive Directory 2. Unauthorized Process Execution 3. Unusual Registry Modification 4. User Account Permission Change
Unauthorized Access	1. User Account Permission Change 2. Access to Sensitive Directory 3. Unauthorized Process Execution 4. Unusual Registry Modification

Table 6. Endpoint Threats - Priority Thresholds

Similar to the previous stages, the file containing the malicious endpoint logs, containing malware hashes, was uploaded to the detection framework. The generated logs were saved to the database for reinforced learning using LSTM algorithm. Significant information factors such as source, destination, core, and event ID were recorded. After adjusting the thresholds and the respective hashes, the framework narrowed the type of threat down to malware – in this particular case, a virus. Figure 28 shows the threat detected at row 104 from 'Test.csv' file containing the hashes for the downloaded virus application.

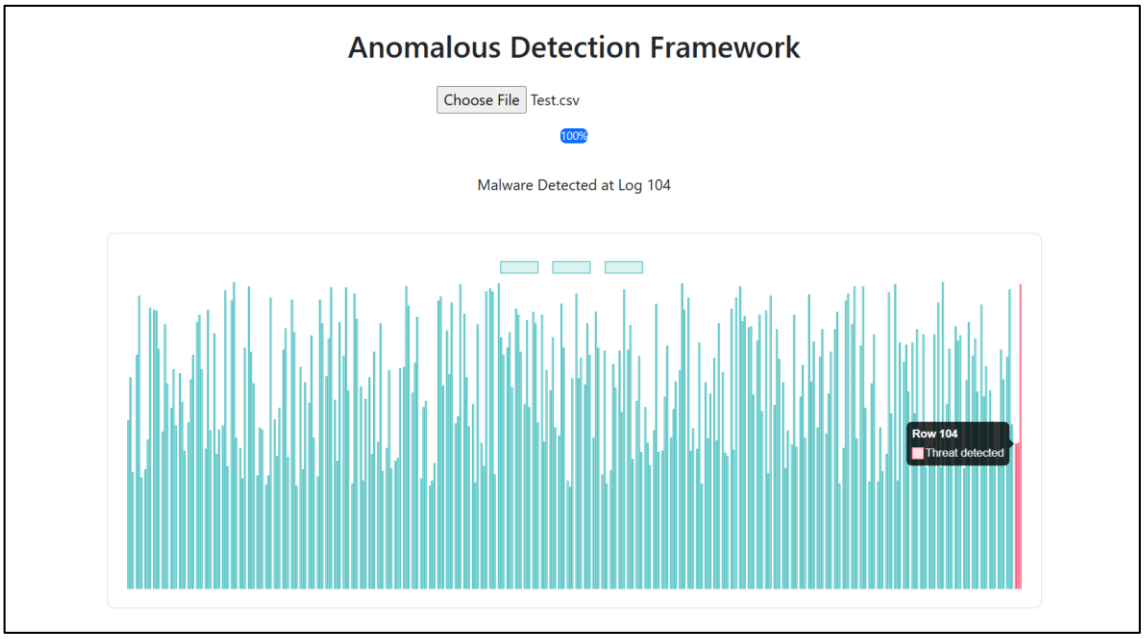

Figure 28. Malware - Threat Detection Framework

Figure 29 shows the hashes detected over the period of 4 seconds while the data file was being read by the server. Like the SQL query, the hashes are analyzed and then recognized by the server to produce the correct logs.

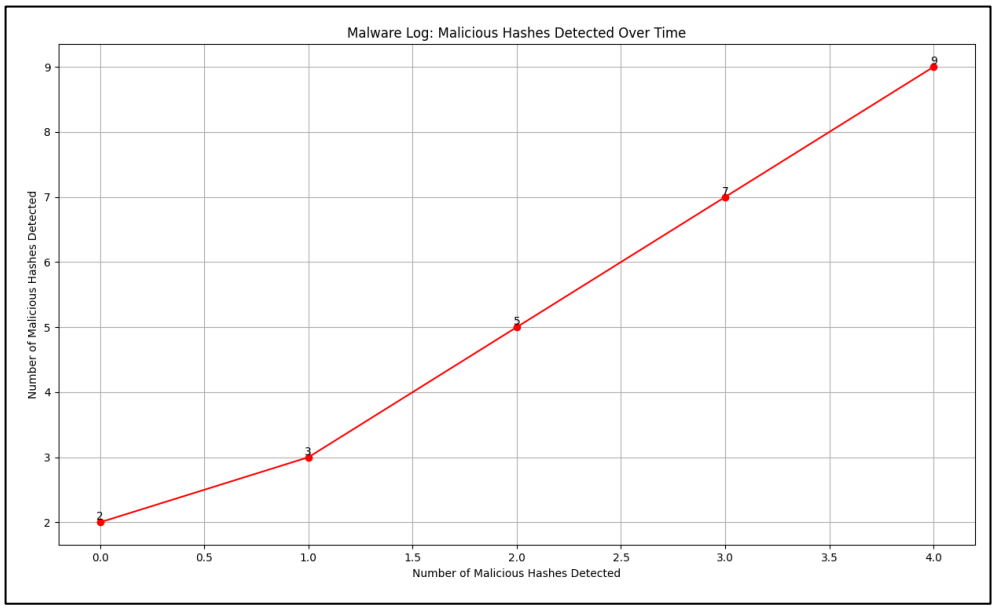

Figure 29. Malware Threats Detected Over Time

Finally, figure 30 presents the log of the detected malware attack. The recorded parameters include the threat ID, severity level, attack type, source of the file, destination file system, malware core information, and the event id.

Parameter	Value
Threat ID	EycdM
Severity Level	5
Attack Type	Malware (Virus)
Source	Win32.Anthrax_Nov2008
Destination	Local System
Core	Assembler Source File
Event ID	1025

Figure 30. Malware Threat Logs

For the final stage of implementation, a Python-based firewall library was developed to integrate the adaptive detection framework into a cohesive network security solution. This library not only allowed the detection of anomalies and threats but also actively controlled network traffic, offering a proactive defense mechanism while enhancing accessibility and versatility.

One of the primary advantages of this firewall library is its robust implementation of various security protocols. It included encryption libraries to safeguard sensitive data in transit, authentication mechanism to verify the identities of users and devices, and secure communication channels, to prevent eavesdropping attacks.

This library not only utilized the core detection capabilities but also extended them to add a more proactive network defense, depending on the original operating system, thus making it a comprehensive solution for modern cybersecurity challenges.

6 Future Work

This chapter focuses on the possible future implementations and different directions in the study of adaptive anomalous detection with AI with potential to focus on fraud detection in real-time, privacy preservation, explainable decision making, and network statistical intelligence.

6.1 NLP Fraud Detection

The use of NLP algorithm to identify information and behaviors in different data sources can prove to be highly important for fraud detection. These sources consist of social media posts, emails, and transaction descriptions. The initial step is to pre-process and analyze the text. This step involves cleaning the raw data and normalizing it. Usually, the data is split into subsets and unnecessary details (special characters) are removed. Afterwards, the entity extraction locates important information like names, addresses, times, and transaction details. This is achieved by establishing a thorough analysis of patterns that can point to potential fraud. There are multiple techniques that can be applied while spotting fraud. Sentiment analysis uses an understanding of emotional tone or sentiments in emails, social media posts, and customer reviews. This can lead to identify phishing attempts and fake reviews. However, other variables such as - grammatical errors, inconsistent writing styles, and anomalies in language patterns are all detected using anomaly detection.

Additionally, topic modelling technique like Latent Dirichlet Allocation (LDA) can be used to identify underlying subjects in huge text datasets, which further helps to prioritize information and allocate resources effectively (Blei et al., 2003). When paired with other ML models, Named Entity Recognition (NER) can be applied to detect and categorize important entities in texts. This can help in building predictive systems that can classify the text inputs as fraudulent or genuine, using algorithms like SVMs or RNNs.

In summary, NLP plays an important role in fraud detection and this could be applied to the cybersecurity framework, which can further help in analyzing key information from unstructured threats. Overall, this enhances the effectiveness to detect and prevent fraudulent activities.

6.2 Privacy Preservation

Privacy preservation is an important aspect in the modern internet age. With the amounts of data available throughout websites and archives, it is important to ensure that the raw data never leaves the local environment of each participating entity. Federated learning presents a transformative approach that enables multiple entities to collaborate and train ML models without the need to share sensitive data directly. Instead, only the model updates (derived from the local environment), are shared and combined to form a global model. In this manner, organizations can benefit from this collective knowledge and diverse data sets of wide range of entities, thus improving the accuracy and robustness of the detection model.

Federated learning makes it possible for different organizations (banks, hospitals, and government agencies) to work together and enhance their cybersecurity measures without compromising the sensitive information (Mothukuri et al., 2021). As each organization trains the model on its local data, the global model, as a whole, becomes more adept at recognizing and mitigating a wide variety of cyberthreats. This removes the issue of data breaches, ethical challenges, and regulatory violations.

6.3 Explainable AI for Interpretability

Traditional AI models usually function as black boxes. This means that they provide little to no information about their decision-making process. Cybersecurity experts often find it difficult to deal with this lack of transparency since they need to understand why specific anomalies are reported in order to respond appropriately. To solve this problem XAI models can offer reasons for their predictions. This prediction procedure includes the precise patterns and elements that contributed to a given threat detection assessment.

Moreover, professionals can modify their respective models and parameters to better meet the security demands of the organizations when they comprehend the logic underlying these processes. In addition to reducing the possibility of bypassing real threats or false positives, this repeated procedure increases threat detection accuracy. It also makes it possible to work in a collaborative setting where AI capabilities and human expertise are complementary.

6.4 Network Statistical Intelligence

One of the major drawbacks of a large network is the inability to monitor all transfer calls and network configurations across all endpoint devices. Although organizations can detect and analyze a unified set of network data, they cannot keep up with all the information at individual network nodes. This can potentially lead to data breaches and leaks. However, employing a network statistical intelligence framework can address this issue.

For all endpoints, a common framework can be employed to recognize pattern shifts within each node. This monitoring enables secure information exchange, better resource allocation, and the removal of dead IP addresses and nodes. While dead nodes may not communicate and share data within the local environment, they are still present and available on the internet. Various network bots, archiving websites, and spyware can exploit this inconsistency. To address the problem, the network intelligence framework calculates the average data consumption of each individual node and performs analysis. This analysis can identify nodes receiving high or irregular amounts of data, leading to the detection of potential breaches within the system.

6.5 Benchmark Analysis

The fundamental idea behind a benchmark analysis concept is to establish a centralized platform to benchmark and analyze different combinations of ML algorithms that work well in particular circumstances in cybersecurity. By systematically evaluating different algorithms under various conditions, the platform can uncover the strengths and weaknesses of each method, and provide insights into their applicability and performance in real-world settings. This analysis not only enhances the selection process for the most suitable algorithms but

also drives continuous improvement in cybersecurity strategies. The best practices can be identified in cybersecurity measures and lessons learned from comprehensive evaluations can be utilized to create a benchmark system which can detail out the perfect combination for a specific setting of any environment.

7 Conclusion

The thesis explored the modern AI and ML methodologies useful in enhancing adaptive anomaly detection frameworks. The main aim of the study was to reduce manual interventions, false alarms, and make the frameworks more robust and resilient. The study concluded with a unified system that can adapt to changing datasets and threats. The combination of cyber threat analysis, adaptive anomaly detection, and AI technologies has resulted in a substantial shift in understanding the dynamic interactions in cybersecurity.

The research took both theoretical and practical approaches to tackle the idea of adaptive anomaly detection. After reviewing modern cybersecurity measures, several cyber-attacks were simulated. The study accumulated a significant amount of data as a result of these simulations. The data included network calls, application logs, endpoint security logs, and access management logs. All this information was then used to create the adaptive model for threat detection.

The first and most crucial aspect to be taken into account when categorizing and identifying threats is their complex nature. The important step, taken by this study, is to break down this nature into multiple steps. This approach is possible only by automatically identifying and mitigating threats from the incoming data streams in real-time. The resulted in an increase in efficiency, effectiveness, and overall resilience of the framework. AI techniques and ML algorithms have highlighted the significance of a customized framework for a range of circumstances.

Among these algorithms are SOMs which are used to cluster data patterns to perform behavior analysis. To focus on sequence identification, decision trees can be utilized to enable the rule-based reasoning. Another algorithm like LSTM allows to retain the information. This transitions to the use of feedback loops which gather the data and memorize the information. This is used with LSTM to improve the response time, enhances detection capabilities, and reduces resource consumption.

In the study, traditional cybersecurity techniques and their significance are also examined. In fact, by incorporating conventional methods, the model can promptly address threats with pre-existing signatures and hashes. As for using the historical data and information, it can be utilized to create more precise and efficient models. This boosts detection and mitigation capabilities of the framework. It is clear that this approach enables the model to continuously learn and adapt, hence reducing expenses and minimizing human errors.

Another aim of this study was to create a usable framework that could be adapted to any situation regarding cyberthreats. This was achieved by creating a server in which the entire framework was implemented. However, this can easily be transformed into a firewall or even a detection security system. Network devices can install this system or firewall library for increased accessibility. Because of its adaptability, the framework can also be customized by organizations to meet their unique security requirements, offering a scalable approach.

There is a lot of promise for future work with the anomaly detection system that combines the modern AI techniques. The data from multiple literature reviews and the findings from this study highlight how AI and ML algorithms have the potential to revolutionize the threat detection industry. This study has addressed other possible areas of research, including explainable logs, fraud detection, privacy protection, and benchmark systems, have been discussed in this study.

In summary, by combining modern AI techniques and ML algorithms with traditional anomaly detection techniques, a coherent and unified adaptive model can be programmed. This results in improved productivity, efficacy, and resilience. Additionally, it also lowers the need for manual intervention, resource usage, time commitment, and overall costs.

Bibliography

Alhayani, Bilal, et al. “Effectiveness of Artificial Intelligence Techniques against Cyber Security Risks Apply of IT Industry.” Materials Today: Proceedings, 10 Mar. 2021, www.sciencedirect.com/science/article/pii/S2214785321016722, https://doi.org/10.1016/j.matpr.2021.02.531.

Anderson, N. (2021). SQL Injection & Web Application Security: A Python-Based Network Traffic Detection Model. Cybersecurity Undergraduate Research. https://digitalcommons.odu.edu/covacci-undergraduateresearch/2023spring/projects/8/

Ban, T., Takahashi, T., Ndichu, S., & Inoue, D. (2023). Breaking Alert Fatigue: AI-Assisted SIEM Framework for Effective Incident Response. 13(11), 6610–6610. https://doi.org/10.3390/app13116610

Blei, D., Edu, B., Ng, A., Jordan, M., & Edu, J. (2003). Latent Dirichlet Allocation. Journal of Machine Learning Research, 3, 993–1022. https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf?ref=http://githubhelp.com

Canizo, M., Triguero, I., Conde, A., & Onieva, E. (2019). Multi-head CNN–RNN for multi-time series anomaly detection: An industrial case study. Neurocomputing, 363, 246–260. https://doi.org/10.1016/j.neucom.2019.07.034

Com, L., & Hinton, G. (2008). Visualizing Data using t-SNE Laurens van der Maaten. Journal of Machine Learning Research, 9, 2579–2605. https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf?fbcl

Dahiya, S., Vikas Siwach, & Harkesh Sehrawat. (2021). Review of AI Techniques in development of Network Intrusion Detection System in SDN Framework. 2021 International Conference on Computational Performance Evaluation (ComPE). https://doi.org/10.1109/compe53109.2021.9752430

Das, R., & Sandhane, R. (2021). Artificial Intelligence in Cyber Security. Journal of Physics: Conference Series, 1964(4), 042072. https://doi.org/10.1088/1742-6596/1964/4/042072

Hayes, M. A., & Capretz, M. A. M. (2014, June 1). Contextual Anomaly Detection in Big Sensor Data. IEEE Xplore. https://doi.org/10.1109/BigData.Congress.2014.19

Informational Cascades and Software Adoption on the Internet: An Empirical Investigation. (n.d.). Misq.umn.edu. Retrieved April 27, 2024, from https://misq.umn.edu/informational-cascades-and-software-adoption-on-the-internet-an-empirical-investigation.html

Mothukuri, V., Parizi, R. M., Pouriyeh, S., Huang, Y., Dehghantanha, A., & Srivastava, G. (2021). A survey on security and privacy of federated learning. Future Generation Computer Systems, 115, 619–640. https://doi.org/10.1016/j.future.2020.10.007

Mulugeta, H. (2023). A Dynamic and Adaptive Cybersecurity Governance Framework. A Dynamic and Adaptive Cybersecurity Governance Framework, 3(3), 327–350. https://doi.org/10.3390/jcp3030017

Nicolas Guzman Camacho. (2024). The Role of AI in Cybersecurity: Addressing Threats in the Digital Age. Journal of Artificial Intelligence General Science (JAIGS) ISSN 3006-4023, 3(1), 143–154. https://doi.org/10.60087/jaigs.v3i1.75

Paschke, A., & Kozlenkov, A. (2009). Rule-Based Event Processing and Reaction Rules. Lecture Notes in Computer Science, 53–66. https://doi.org/10.1007/978-3-642-04985-9_8

Raza, H., Prasad, G., & Li, Y. (2015). EWMA model based shift-detection methods for detecting covariate shifts in non-stationary environments. Pattern Recognition, 48(3), 659–669. https://doi.org/10.1016/j.patcog.2014.07.028

Schmitt, M. (2023). Securing the Digital World: Protecting smart infrastructures and digital industries with Artificial Intelligence (AI)-enabled malware and intrusion detection. *Journal of Industrial Information Integration*, *36*, 100520–100520. https://doi.org/10.1016/j.jii.2023.100520

Scott, J. (2017). Signature Based Malware Detection is Dead. https://informationsecurity.report/Resources/Whitepapers/920fbb41-8dc9-4053-bd01-72f961db24d9_ICIT-Analysis-Signature-Based-Malware-Detection-is-Dead.pdf

Sun, N., Ding, M., Jiang, J., Xu, W., Mo, X., Tai, Y., & Zhang, J. (2023). Cyber Threat Intelligence Mining for Proactive Cybersecurity Defense: A Survey and New Perspectives. IEEE Communications Surveys & Tutorials, 1–1. https://doi.org/10.1109/comst.2023.3273282

Swamy, S. N., Jadhav, D., & Kulkarni, N. (2017). Security threats in the application layer in IOT applications. 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC). https://doi.org/10.1109/i-smac.2017.8058395

Vu, Q. H., Ruta, D., & Cen, L. (2019). Gradient boosting decision trees for cyber security threats detection based on network events logs. 2019 IEEE International Conference on Big Data (Big Data). https://doi.org/10.1109/bigdata47090.2019.9006061

Wu, Y., Wei, D., & Feng, J. (2020). Network Attacks Detection Methods Based on Deep Learning Techniques: A Survey. Security & Communication Networks, 1–17. https://doi.org/10.1155/2020/8872923

Yeh, Y.-S., Lai, W.-S., & Cheng, C.-J. (2002). Applying lightweight directory access protocol service on session certification authority. Computer Networks, 38(5), 675–692. https://doi.org/10.1016/s1389-1286(01)00282-1

Zhang, L., Deng, S., & Li, S. (2017). Analysis of power consumer behavior based on the complementation of K-means and DBSCAN. https://doi.org/10.1109/ei2.2017.8245490

Ziyin, L., Li, B., Simon, J. B., & Ueda, M. (2021). SGD with a Constant Large Learning Rate Can Converge to Local Maxima. ArXiv (Cornell University). https://doi.org/10.48550/arxiv.2107.11774