|  |  |
| --- | --- |
|  | **MINISTRY OF EDUCATION AND TRAINING** |

|  |
| --- |
| **FPT UNIVERSITY** |
| **Capstone Project Document** |
| **Smart Monitoring and Automated Response for Endpoint using Splunk Machine Learning** |
|  |
| |  |  |  | | --- | --- | --- | | **Group Member** | Nguyen Thanh Hai - Leader  Nguyen Quoc Dai - Member  Huynh Minh Thang - Member  Pham Minh Nghia - Member | SE172059  SE173556  SE172703  SE150935 | | **Supervisor** | Tran Van Ninh | | | **Capstone Project code** | SU25IA05 | | |
|  |
|  |

- Ho Chi Minh, 05/2025 –

# **ABSTRACT**

Cybersecurity threats targeting endpoint devices have become more sophisticated and frequent, posing significant challenges to modern organizations. Traditional monitoring approaches often rely on static rules and manual analysis, which can delay detection and response. This study aims to develop a smart monitoring and automated response system for endpoint security using machine learning techniques implemented through Splunk. The objective is to enhance real-time threat detection and minimize response latency by leveraging behavioral analytics and predictive modeling.

The system collects and analyzes endpoint log data, applying machine learning algorithms to identify anomalous patterns that may indicate malicious activities such as brute-force attacks, data exfiltration, or privilege escalation. Various supervised and unsupervised learning models are evaluated to determine their effectiveness in detecting different types of threats. Upon identifying suspicious behaviors, the system triggers automated response actions, including alert generation, process termination, and isolation of affected endpoints.

The results demonstrate that integrating machine learning into a Splunk-based monitoring framework significantly improves the accuracy and speed of threat detection. Automated response workflows reduce the burden on security analysts and help contain threats before they escalate.

This project highlights the potential of combining machine learning with security monitoring tools to build proactive and intelligent defense mechanisms for endpoint protection, contributing to the advancement of automated cybersecurity operations.

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to all those who supported us throughout the development of this thesis.

First and foremost, we would like to thank our advisor, Tran Van Ninh, for his valuable guidance, insightful feedback, and continuous encouragement during the entire research process. His expertise and dedication were instrumental in shaping the direction and quality of this work.

We also wish to thank the faculty members and staff of the Faculty of Information Assurance, FPT University for providing the academic foundation and resources necessary for completing this project.

Special thanks go to our classmates and friends, whose collaboration and support offered both motivation and helpful perspectives during research and implementation.

We would like to extend our appreciation to Splunk for generously providing a developer license key, which was essential for the successful implementation and testing of the monitoring and automated response system. Their support greatly facilitated the practical aspects of this project.

Finally, we are deeply grateful to our families for their unwavering support, patience, and belief in us throughout this journey. Without their encouragement, this thesis would not have been possible.

**TABLE OF CONTENTS**

[ABSTRACT 2](#_Toc4392)

[ACKNOWLEDGEMENT 3](#_Toc20051)

[LIST OF FIGURES 7](#_Toc6720)

[LIST OF TABLE 9](#_Toc10271)

[ABBREVIATIONS 10](#_Toc30145)

[CHAPTER 1 INTRODUCTION 11](#_Toc650)

[1.1. Background 11](#_Toc10450)

[1.2. Problem Statement 12](#_Toc1799)

[1.3. Research Objectives 13](#_Toc5232)

[1.4. Significance of the Study 14](#_Toc25227)

[1.5. Scope and Limitations 15](#_Toc2538)

[1.6. Thesis Structure 16](#_Toc16557)

[CHAPTER 2 LITERATURE REVIEW 18](#_Toc12274)

[2.1. Review of Previous Studies 18](#_Toc6378)

[2.2. Summary of the Literature Review 20](#_Toc9563)

[2.3. Contribution of Research 20](#_Toc9962)

[CHAPTER 3 22](#_Toc30410)

[METHODOLOGY 22](#_Toc28145)

[3.1. Research Design 22](#_Toc26332)

[3.2. Data Collection Methods 23](#_Toc26855)

[3.2.1. Benchmark Dataset (CICIDS2017) 23](#_Toc17555)

[3.2.2. Real-Time Endpoint Logs (Simulated for Windows and Linux) 23](#_Toc21405)

[3.3. Sampling Data Analysis Techniques: 24](#_Toc10753)

[3.3.1. Model Training Process: 24](#_Toc7994)

[3.3.2. Training Algorithm: 25](#_Toc3476)

[3.3.3. ML Training Techniques: 27](#_Toc29418)

[3.3.4. Log processing and API calls 30](#_Toc8810)

[3.4. Limitations of the Methodology 31](#_Toc25425)

[3.4.1. Dataset Dependency and Generalization Issues 31](#_Toc10389)

[3.4.2. Feature Extraction Limitations (Flow-based Features Only) 31](#_Toc7760)

[3.4.3. Imbalanced Data and Threshold Selection 32](#_Toc5179)

[3.4.4. Model Drift and Evolving Threat Landscape 32](#_Toc30228)

[3.4.5. Limited Contextual Awareness in Automated Response 32](#_Toc27377)

[3.4.6. Splunk Resource Overhead and Scalability Concerns 32](#_Toc19854)

[3.4.7. Lack of Explainability in ML Decisions 33](#_Toc28875)

[CHAPTER 4 34](#_Toc14790)

[EXPERIMENTAL AND RESULTS 34](#_Toc19181)

[4.1. Introduction 34](#_Toc1702)

[4.2. Presentation of Data 34](#_Toc11169)

[4.3. Analysis of Results 37](#_Toc812)

[4.3.1. ML Training Results 37](#_Toc31816)

[4.3.2. Automated Incident Response 38](#_Toc22569)

[4.3.3. Visual Analytics for Security Monitoring through Splunk Dashboards 42](#_Toc23452)

[4.4. Interpretation of Results 43](#_Toc1522)

[4.4.1. Interpretation of Machine Learning Results 43](#_Toc15123)

[4.4.2. Interpretation of the Auto-Response System 44](#_Toc18612)

[4.5. Comparison with Literature 45](#_Toc31857)

[4.6. Implications of the Results 45](#_Toc14531)

[CHAPTER 5: 48](#_Toc24507)

[DISCUSSION 48](#_Toc25691)

[5.1. Restate the Research Problem or Objectives 48](#_Toc2518)

[5.2. Summarize Key Findings 48](#_Toc24453)

[CHAPTER 6: 50](#_Toc11141)

[CONCLUSION AND FUTURE WORK 50](#_Toc16909)

[6.1. Conclusion 50](#_Toc10696)

[6.2. Future Work 50](#_Toc24529)

**LIST OF FIGURES**

[Figure 1 : The Occurrence of Endpoint Attacks 10](#_Toc27073)

[Figure 2 : Processing diagram 12](#_Toc23282)

[Figure 3 : Multi-Layer AI for SOC 13](#_Toc12784)

[Figure 4 : System diagram 14](#_Toc25527)

[Figure 5 : Research Workflow Diagram 21](#_Toc26077)

[Figure 6 : Model Training Process 22](#_Toc6387)

[Figure 7 : Schematic illustration of the XGboost model. 24](#_Toc13068)

[Figure 8 : Boosting algorithm 25](#_Toc4973)

[Figure 9 : Decision tree 25](#_Toc28838)

[Figure 10 : Load & merge data 26](#_Toc7347)

[Figure 11 : Handling null and duplicate values 26](#_Toc14029)

[Figure 12 : Benign vs. Malicious Distribution 26](#_Toc8463)

[Figure 13 : Split & scale 27](#_Toc44)

[Figure 14 : Train with XGBoost 27](#_Toc32183)

[Figure 15 : Service receives, processes and sends logs to Splunk 28](#_Toc2514)

[Figure 16 : Cronjob to send logs 29](#_Toc32580)

[Figure 17 : CICFlowMeter tool 29](#_Toc25920)

[Figure 18 : Distribution in dataset 30](#_Toc9317)

[Figure 19 : A detailed analysis of CICIDS2017 dataset 33](#_Toc28864)

[Figure 20 : Distribution of labels in the CICIDS2017 dataset. 34](#_Toc30933)

[Figure 21 : Listed features of network traffic in CICIDS2017 34](#_Toc10745)

[Figure 22 : ML Training Results 35](#_Toc7306)

[Figure 23 : Update IP blocking rules using Iptable 36](#_Toc3777)

[Figure 24 : Count the number of occurrences when blocked 37](#_Toc19248)

[Figure 25 : Telegram Chatbot Integration 38](#_Toc50)

[Figure 26 : Email Notification to Administrators 39](#_Toc30677)

[Figure 27 : Splunk Dashboards 41](#_Toc23552)

[Figure 28 : Telegram chatbot interaction 44](#_Toc20710)

[Figure 29 : Aided by the Gemini language model 44](#_Toc16393)

[Figure 30 : Difference Between Random Forest vs XGBoost 49](#_Toc16474)

[Figure 31 : Implement periodic retraining or model updates 49](#_Toc11310)

[Figure 32 : SOAR Splunk 50](#_Toc15405)

[Figure 33 : Elastic SIEM 51](#_Toc30306)

**LIST OF TABLE**

[Table 1 : Comparison between Traditional Monitoring Systems and Machine Learning-Based Systems 11](#_Toc2139)

[Table 2 : Review of Previous Studies 17](#_Toc27885)

**ABBREVIATIONS**

**Acronym Meaning**

AI Artificial Intelligence

BA Business Administration

CS Computer Science

CF Computing Fundamental

IA Information Assurance

ITS Information Technology Specialization

Math. Mathematics

MC Multimedia Communications

ML Machine Learning

# **CHAPTER 1 INTRODUCTION**

**1.1. Background**

In recent years, the growing sophistication and frequency of cyberattacks have posed serious challenges to the security of endpoint devices—computers, laptops, and other user-accessed systems that serve as entry points to an organization’s network. Unlike centralized systems, endpoints are distributed, often less protected, and more vulnerable to threats such as malware infections, unauthorized access, credential theft, and insider attacks. As organizations increasingly adopt remote work, cloud services, and BYOD (Bring Your Own Device) policies, securing endpoints has become a top priority in the field of cybersecurity.

Traditional security monitoring systems often rely on rule-based detection methods, which are limited in identifying unknown or evolving threats. These systems may generate a large number of false positives or miss subtle signs of compromise altogether. Moreover, incident response in many environments is still largely manual, resulting in slow reaction times and increased damage when threats occur.

To address these limitations, the integration of machine learning (ML) techniques into Security Information and Event Management (SIEM) platforms has gained significant attention. Machine learning enables systems to learn from historical data, detect anomalies, and make predictions based on behavioral patterns, thus enhancing the ability to identify advanced persistent threats (APTs) and zero-day attacks.

This project aims to apply machine learning within Splunk—a widely used SIEM and data analytics platform—to build a smart monitoring and automated response system specifically designed for endpoint security. By leveraging Splunk’s built-in Machine Learning Toolkit (MLTK), the system can process large volumes of endpoint log data, detect suspicious behavior in real-time, and trigger automated responses to contain or remediate threats. This approach not only improves detection accuracy and response speed but also reduces the operational burden on security analysts, contributing to a more proactive and intelligent defense strategy.

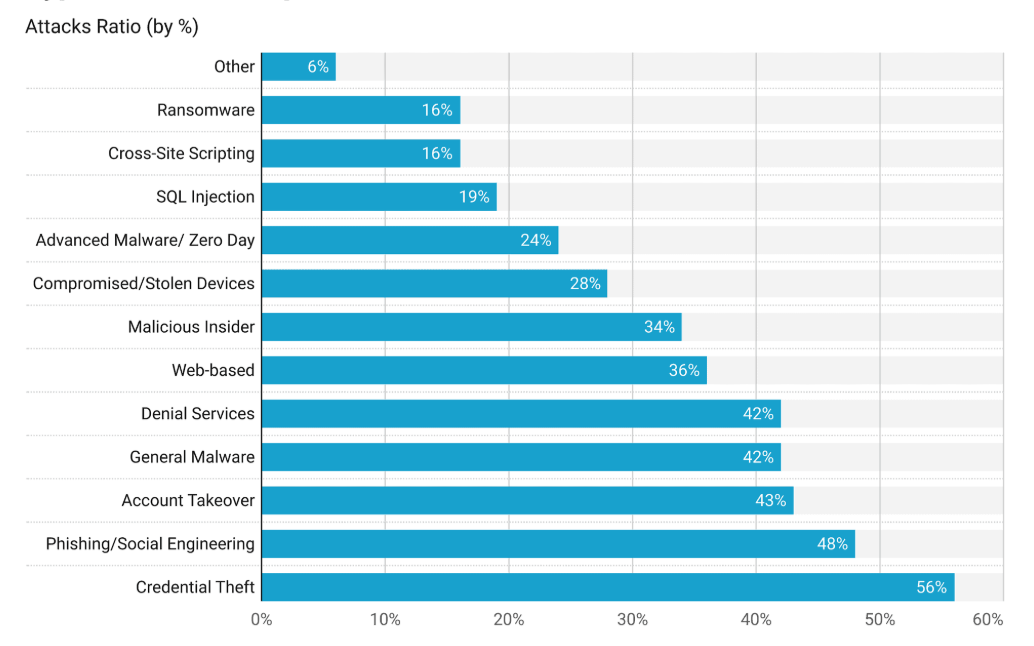


Figure 1: The Occurrence of Endpoint Attacks

**1.2. Problem Statement**

With the increasing number of cyberattacks targeting endpoint devices, organizations face growing challenges in detecting and responding to threats in a timely and accurate manner. Endpoints are often the first point of compromise in attack chains, making them a critical focus in enterprise security strategies. However, many existing security solutions rely heavily on static detection rules, signature-based methods, and manual investigation processes, which are often ineffective against modern, sophisticated attacks.

These traditional approaches suffer from several limitations:

Inability to detect zero-day attacks: Because they rely on predefined patterns, they cannot recognize previously unknown threats.

Alert fatigue among security analysts: The high volume of security logs generated by endpoint devices overwhelms analysts, increasing the risk of missing real threats.

Delayed incident response: Manual containment and remediation introduce time gaps, giving attackers more opportunities to exploit vulnerabilities and escalate privileges.

Moreover, the lack of integration between monitoring tools and automated response systems creates a gap between threat detection and mitigation, resulting in slow and inefficient security operations. This not only increases the potential damage of attacks but also puts a significant burden on human resources in the Security Operations Center (SOC).

Therefore, there is a clear need for an intelligent, automated system that can continuously monitor endpoint activities, accurately detect suspicious behavior using advanced analytics, and trigger appropriate response actions without human intervention. This project addresses that need by leveraging Splunk’s machine learning capabilities to build a smart monitoring and automated response framework tailored for endpoint security. The solution aims to bridge the gap between detection and response, improve accuracy, reduce response time, and optimize the overall effectiveness of enterprise cybersecurity operations.

| **Criteria** | **Traditional Monitoring Systems** | **Machine Learning-Based Systems** |
| --- | --- | --- |
| Anomaly Detection | Rule-based, manual | Automated, data-driven |
| Capability to Detect New Attacks | Low | High |
| Response Speed | Slow, human-dependent | Near real-time |
| Accuracy | High rate of false positives | Reduced false positive rate |
| Scalability | Limited | Flexible and scalable |

Table 1: Comparison between Traditional Monitoring Systems and Machine Learning-Based Systems

**1.3. Research Objectives**

The primary objective of this research is to design and implement an intelligent system for monitoring and automatically responding to endpoint threats using machine learning techniques within the Splunk platform. This project aims to enhance endpoint security by combining advanced data analytics with automated decision-making capabilities to detect and mitigate cyber threats in real time. The specific objectives of the study are as follows:

* To analyze common endpoint security threats and understand their behavioral patterns through log data, including attacks such as brute-force attempts, privilege escalation, and suspicious process execution.
* To collect and preprocess endpoint security logs for effective use in machine learning models, ensuring data quality, consistency, and relevance for threat detection tasks.
* To apply and evaluate machine learning algorithms (both supervised and unsupervised) for anomaly detection and classification of security events based on endpoint behavior.
* To integrate the trained models into the Splunk environment, utilizing the Splunk Machine Learning Toolkit (MLTK) for real-time monitoring and threat detection.
* To design and implement automated response workflows using Splunk’s alerting and orchestration features, enabling timely actions such as alert generation, session termination, or endpoint isolation upon detection of threats.
* To assess the system’s performance in terms of detection accuracy, false positive rate, and response time, comparing it with traditional rule-based approaches.
* To propose a scalable and adaptable framework that can be further extended to support different types of endpoints and evolving threat landscapes.

By achieving these objectives, the project aims to contribute a practical and effective solution for improving endpoint security through intelligent monitoring and automated response mechanisms.

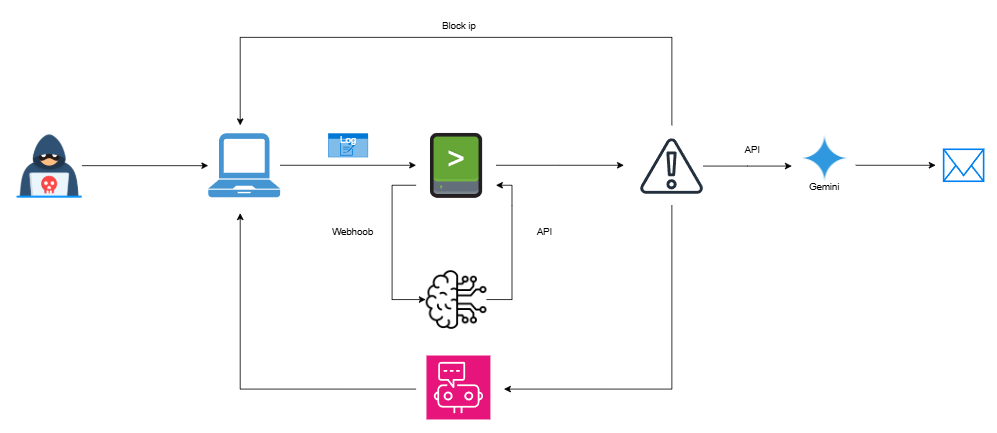


Figure 2: Processing diagram

**1.4. Significance of the Study**

As cyber threats continue to grow in both complexity and frequency, organizations are under increasing pressure to enhance their ability to detect and respond to security incidents, especially at the endpoint level. Endpoint devices are often the initial targets of cyberattacks, serving as potential entry points for malicious activities such as ransomware, credential theft, or unauthorized access. Traditional rule-based security systems are no longer sufficient to cope with these advanced threats, especially in environments with high volumes of data and limited human resources.

This study is significant because it proposes a practical and intelligent approach to strengthening endpoint security through the integration of machine learning techniques into a widely used SIEM platform—Splunk. By leveraging machine learning for behavioral analysis and anomaly detection, the proposed system aims to improve detection accuracy, reduce false positives, and uncover previously unknown threats. Furthermore, the implementation of automated response mechanisms can help security teams act more quickly and efficiently, reducing the potential impact of attacks and minimizing operational downtime.

From an academic perspective, this research contributes to the growing field of intelligent cybersecurity solutions by demonstrating how machine learning can be effectively applied in real-world monitoring environments. It provides a replicable framework that future researchers can build upon, extend, or adapt to other platforms or threat types.

From a practical standpoint, organizations and security professionals can benefit from the proposed system’s ability to automate critical security operations, improve situational awareness, and enhance the overall resilience of their IT infrastructure. This study aligns with modern cybersecurity demands and supports the transition toward proactive, data-driven, and automated security operations in an era of increasing digital risk.

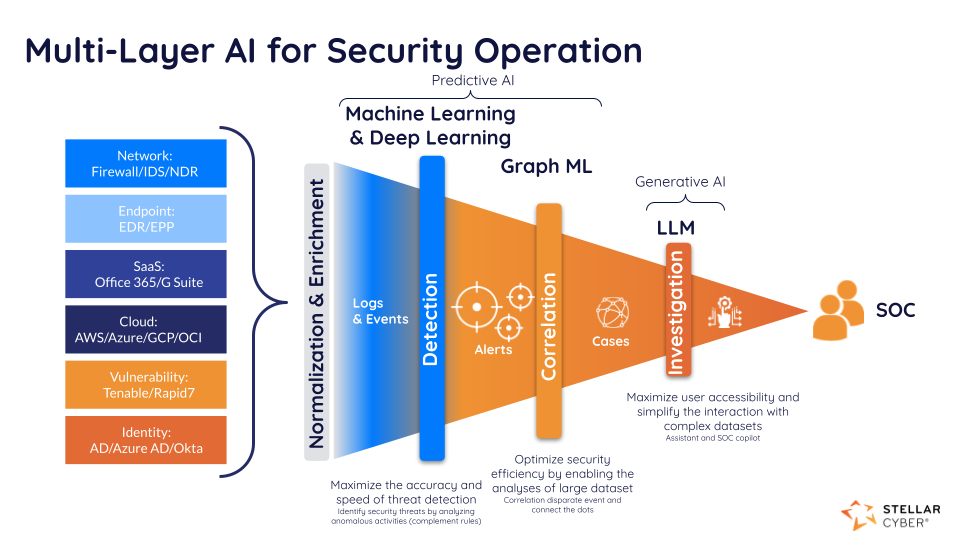


Figure 3: Multi-Layer AI for SOC

**1.5. Scope and Limitations**

This project aims to develop a smart monitoring and automated response system for endpoints, specifically focusing on Linux and Windows-based devices using Splunk and its Machine Learning Toolkit (MLTK). The primary scope includes:

* Collecting and analyzing system logs from both Linux and Windows endpoints.
* Detecting suspicious behaviors such as brute-force attacks, credential dumping and other common endpoint threats.
* Utilizing machine learning techniques within Splunk to classify and prioritize alerts based on severity and risk level.
* Demonstrating automated responses such as alert generation, tagging malicious behavior, or notifying administrators.

Operating in a test environment to simulate realistic attack scenarios and evaluate system performance.

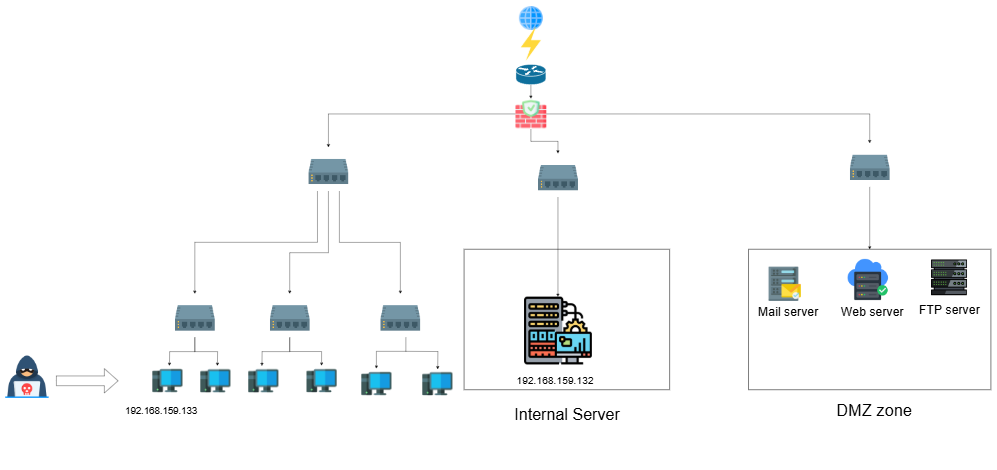


Figure 4: System diagram

Although comprehensive, the project has the following limitations:

* The system is designed exclusively for endpoint devices (Linux and Windows) and does not cover IoT, smartphones, or other non-traditional devices.
* It is implemented in a controlled lab environment, which may not capture the full complexity and diversity of real-world enterprise settings.
* The dataset used for training machine learning models is limited and may not represent all possible attack patterns.
* Automated responses are basic and not deeply integrated with SOAR platforms or external defense mechanisms.

The detection scope is constrained to specific threat categories, not covering the entire spectrum of endpoint-related attacks.

**1.6. Thesis Structure**

This thesis is organized into the following chapters:

Chapter 1 Introduces the research background, objectives, significance, scope, and structure.

Chapter 2 Reviews relevant existing research and demonstrates how your thesis fits into the broader academic conversation.

Chapter 3 Outlines the methodology, including system design, tool integration, data collection, and model training.

Chapter 4 Details the implementation process and presents the experimental setup.

Chapter 5 Discusses the results, evaluation, and analysis of the system’s performance.

Chapter 6 Concludes the study, summarizes findings, and suggests areas for future research.

# **CHAPTER 2 LITERATURE REVIEW**

**2.1. Review of Previous Studies**

In recent years, a significant number of studies have explored the use of machine learning (ML) techniques for enhancing endpoint detection and response (EDR) systems. These studies provide a foundation for the current project by highlighting the advantages, challenges, and research gaps in applying ML within security operations. Several key findings from prior research are summarized below:

* ML-based Threat Detection in Endpoint Environments: Research has shown that ML algorithms such as decision trees, random forests, and support vector machines (SVM) can effectively detect malicious behaviors on endpoints. For instance, demonstrated a high detection rate for brute-force attacks using SVM trained on Windows logs. [1]
* Log-Based Security Monitoring with Splunk: A number of practical studies and whitepapers have explored using Splunk as a log aggregation and analysis platform. Splunk’s Machine Learning Toolkit (MLTK) allows for the integration of anomaly detection models, which has been applied in multiple works to detect credential misuse, lateral movement, and insider threats.
* Hybrid Approaches Combining Rule-Based and ML Techniques: Some researchers have proposed combining rule-based methods (e.g., MITRE ATT&CK logic) with ML to improve alert accuracy and reduce false positives. This hybrid approach has proven effective in constrained environments where training data is limited.

Limitations in Previous Studies:

* Many studies focus solely on Windows environments, leaving a gap in Linux-based monitoring.
* Most research is either simulation-based or lacks real-time implementation and response actions.
* There is often a lack of integration between detection and automated response workflows.

| **Study / Author** | **Environment** | **Detection Method** | **Platform** | **Limitations** |
| --- | --- | --- | --- | --- |
| Michael Hart et al., Splunk Threat Research Team (2025) | Windows | Autoencoder-based anomaly detection (unsupervised deep learning) | Splunk MLTK + Jupyter/Python | Focused only on Windows logs and process behavior; no automated response in lab environment |
| Splunk Threat Research Team – SMB Traffic Spike (2025) | Windows | DensityFunction anomaly detection (MLTK) | Splunk Enterprise Security (ES) | Experimental analytic, untested datasets, flagged for failing validation/integration |
| Splunk Threat Research Team – Unusually Long Command Line (2025) | Windows | Anomaly detection on command-line length using MLTK | Splunk ES | Experimental and unvalidated analytic; limited use case scope |
| QFunction Blog by Ryan Smith (2023–2024) | Mixed endpoints (domain controllers) | Supervised/predictive ML models (e.g. to predict malicious commands or network connections) | Splunk DSDL, Splunk MLTK | Labs and blog experiments; not enterprise‐grade automation; high compute cost |
| Rhode et al. (2019) – **ArXiv** | Endpoint (Windows/Linux implied) | Statistical filtering + ML to detect malware processes, auto kill malicious processes | Custom ML system on endpoints | High false-positive rate (~14%), code-only prototype, separate from Splunk |

Table 2: Review of Previous Studies

**2.2. Summary of the Literature Review**

The review of previous studies highlights the growing interest in applying machine learning to enhance endpoint detection and response (EDR) systems. Various research efforts, particularly from Splunk’s Threat Research Team and other practitioners, have demonstrated the effectiveness of machine learning models—such as anomaly detection, supervised classification, and hybrid approaches—in identifying malicious activities based on endpoint log data.

Most existing works primarily focus on Windows-based environments, utilizing Splunk’s Machine Learning Toolkit (MLTK) for detection of suspicious behaviors such as unusual command-line executions, SMB traffic anomalies, and brute-force attacks. While these approaches show promise, they often remain experimental, with limited scope, minimal automation, and insufficient validation in real-world settings. Moreover, the integration of automated response actions is rarely addressed in a comprehensive manner.

A key gap identified in the literature is the lack of attention to Linux endpoints and the absence of end-to-end systems that cover both detection and response within the same framework. Additionally, many studies suffer from small datasets, simulation-only implementations, or a narrow focus on specific attack techniques

This project aims to address these limitations by designing a practical monitoring and automated response system for both Windows and Linux endpoints, using Splunk and machine learning. It contributes to the field by extending detection coverage across platforms and demonstrating a proof-of-concept system capable of real-time analysis and automated response..

**2.3. Contribution of Research**

This research contributes both theoretically and practically to the field of cybersecurity, particularly in the domain of endpoint detection and response (EDR) using machine learning. The major contributions are as follows:

* The system supports security monitoring for both Windows and Linux endpoints, addressing the limitation in prior studies that typically focus on only one operating system, especially Windows. This ensures broader applicability in diverse enterprise environments.
* The research applies Splunk's Machine Learning Toolkit (MLTK) to analyze system log data in real time, enabling the identification of suspicious behaviors such as brute-force login attempts, credential dumping, and unusual command-line activity. This demonstrates a practical application of machine learning in security operations.
* In addition to detection, the system incorporates automated response mechanisms. Depending on the threat type and severity, the system can generate alerts, tag suspicious users or processes, and initiate predefined remediation actions. This closes the gap between detection and action, which is often missing in previous research.
* A complete prototype was developed and tested in a controlled lab environment, using real log data and simulated attack scenarios. The system is designed to be reproducible and adaptable, serving as a proof-of-concept for future expansion or enterprise deployment.
* The detection logic used in the system maps directly to recognized MITRE ATT&CK techniques (e.g., T1110 – Brute Force, T1003 – Credential Dumping, T1114.003 – Email Forwarding), enhancing the project’s relevance to standardized threat intelligence and security operations center (SOC) workflows.[2]
* By combining academic knowledge in machine learning with the practical capabilities of Splunk, this research bridges the gap between theoretical models and operational requirements, providing valuable insight into real-world deployment challenges and opportunities.

**CHAPTER 3**

**METHODOLOGY**

**3.1. Research Design**

This study employs a quantitative experimental research design to develop and evaluate a smart monitoring and automated response system for Linux and Windows endpoints using Splunk and machine learning. The research follows a structured workflow consisting of multiple phases:

1. Research and Literature Review:  
   An in-depth review of existing intrusion detection systems (IDS), endpoint monitoring tools, and machine learning approaches, especially focusing on real-time log analysis and automated responses.
2. Dataset Selection and Preprocessing:  
   The CICIDS2017 dataset, developed by the University of New Brunswick, is used for model training and testing. The dataset is preprocessed to extract relevant features suitable for endpoint behavior analysis.
3. Environment Setup:  
   A hybrid environment simulating both Windows and Linux endpoints is created. Splunk agents are installed to collect logs and forward them to a centralized Splunk server for analysis.
4. Model Development using XGBoost:  
   The XGBoost algorithm is chosen due to its high accuracy and efficiency. The model is trained on labeled network traffic data from the CICIDS2017 dataset.
5. Integration with Splunk:  
   The trained model is integrated into Splunk using the Machine Learning Toolkit. Real-time data streams from endpoints are analyzed to detect suspicious behaviors.
6. Automated Response Mechanism:  
   Based on detection results, the system triggers automated responses (e.g., alerts, user account disablement, process termination) through Splunk SOAR or custom scripts.
7. Evaluation and Validation:  
   The model's performance is evaluated using standard metrics (accuracy, precision, recall, F1-score), and results are compared with findings from previous research.

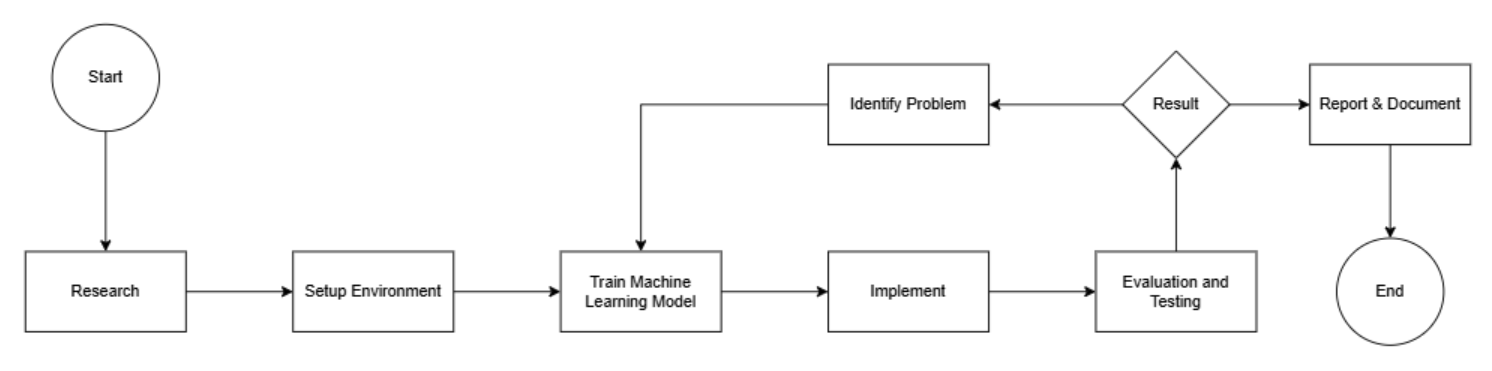
****

Figure 5: Research Workflow Diagram

**3.2. Data Collection Methods**

***3.2.1. Benchmark Dataset (CICIDS2017)***

Their evaluations of the existing eleven datasets since 1998 show that most are out of date and unreliable. Some suffer from a lack of traffic diversity and volume, some do not cover the variety of known attacks, while others anonymize packet payload data, which cannot reflect current trends. Some also lack comprehensive feature sets and metadata.

The CICIDS2017 dataset contains benign traffic and the most up-to-date common attacks, resembling real‐world PCAP data. It also includes the results of network traffic analysis using CICFlowMeter, with flows labeled by time stamp, source and destination IP addresses, source and destination ports, protocols, and attack type (CSV files).

Generating realistic background traffic was their top priority in building this dataset. They used the proposed B-Profile system (Sharafaldin et al., 2016) to profile abstract human interaction behaviors and generate naturalistic benign background traffic. For this dataset, they constructed abstract behaviors for 25 users based on HTTP, HTTPS, FTP, SSH, and email protocols.

The data capture period began at 9 a.m. on Monday, July 3, 2017 and ended at 5 p.m. on Friday, July 7, 2017, spanning five days. Monday served as a normal day, including only benign traffic. The implemented attacks included Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet, and DDoS. These attacks were executed both in the morning and afternoon on Tuesday, Wednesday, Thursday, and Friday.

***3.2.2. Real-Time Endpoint Logs (Simulated for Windows and Linux)***

In this study, network traffic data is collected in real-time directly from the network interface using CICFlowMeter, a flow-based network traffic generator developed by the Canadian Institute for Cybersecurity (CIC). This tool captures raw packets from a specified network interface (e.g., eth0) and converts them into bidirectional flow-based records. Each flow contains statistical features such as duration, total packets, bytes, source/destination ports, and protocol information, which are crucial for intrusion detection.

To integrate the collected data with Splunk, a Splunk Universal Forwarder is deployed on the same machine. This forwarder is configured to monitor the output CSV file and forward it to the central Splunk indexer for further analysis and visualization. The configuration is defined as follows:

This approach ensures that network traffic is collected and streamed in near real-time into Splunk, enabling timely detection and analysis of anomalies and potential threats. It also allows for the application of machine learning models such as XGBoost, which require structured input formats like those provided by CICFlowMeter.

By utilizing CICFlowMeter in real-time and integrating it with Splunk, this research ensures a continuous, automated, and reliable data pipeline from traffic capture to advanced analysis.

**3.3. Sampling Data Analysis Techniques:**

***3.3.1. Model Training Process:***

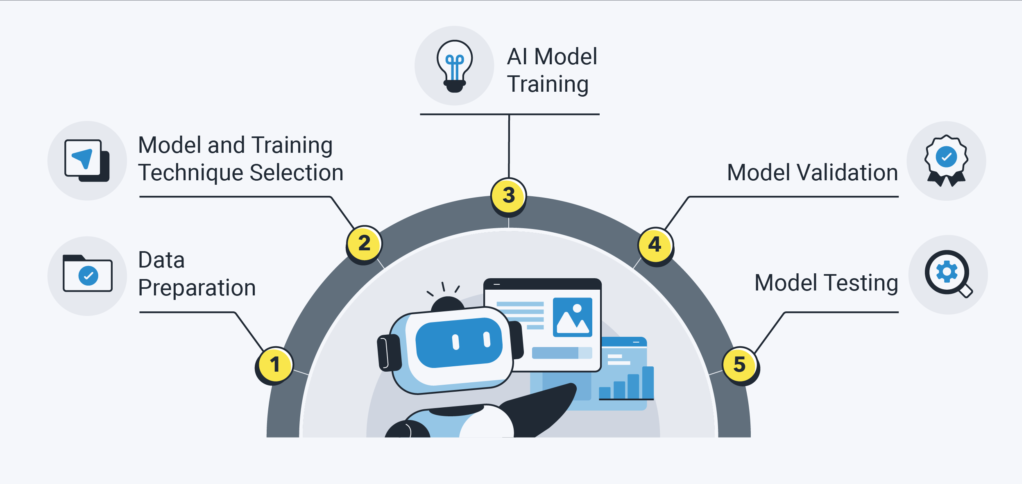


Figure 6: Model Training Process

3.3.1.1. Data Preparation

Before training the model, the data must be carefully prepared, including steps such as data cleaning, normalization, and data splitting. The training data should be representative of the scenarios the model is expected to encounter in real-world applications.

3.3.1.2. Selecting an Appropriate Model

Depending on the specific problem, data scientists select an appropriate model such as linear regression, decision tree, neural network, or support vector machine. Choosing the right model helps optimize the performance of the training process.

3.3.1.3. Model Training

In this step, the training data is fed into the model to enable it to learn patterns and relationships within the data. Machine learning algorithms continuously adjust the model’s parameters to improve prediction accuracy.

3.3.1.4. Model Testing and Evaluation

After the training process is completed, the model is tested using a separate testing dataset to assess its accuracy and generalization capability.

3.3.1.5. Model Optimization

Based on the evaluation results, data scientists may fine-tune the model to enhance its performance and reduce errors. This optimization may involve modifying the model’s structure, adjusting hyperparameters, or experimenting with different machine learning algorithms.

***3.3.2. Training Algorithm:***

XGBoost is a boosting algorithm optimized for speed and performance. It is widely used in supervised learning tasks such as classification and regression due to its ability to handle large datasets, reduce overfitting, and deliver high predictive accuracy.

The XGBoost algorithm works by combining boosting techniques with multiple decision trees to produce the final prediction. This ensemble approach enhances accuracy and improves overall model performance compared to many other machine learning methods.

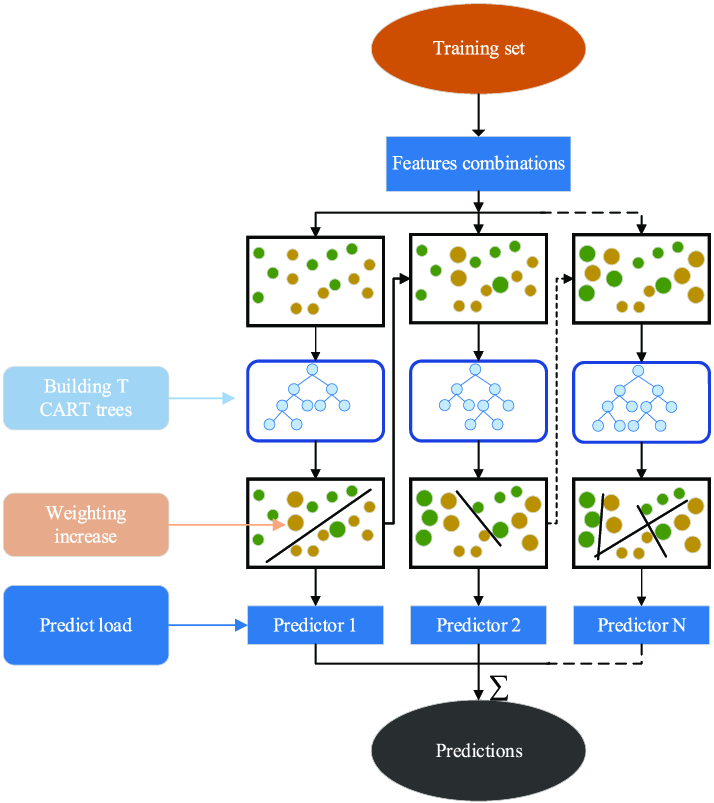


Figure 7: Schematic illustration of the XGboost model.

Boosting is a machine learning technique that incrementally improves weak models by adding more models that focus on correcting the errors made by previous ones. During the boosting process, each new model is trained on samples that were misclassified by earlier models, gradually refining the overall prediction accuracy.



Figure 8: Boosting algorithm

Decision Trees are algorithms that use a tree-like structure to make predictions. The tree is built by recursively splitting the dataset into smaller groups based on thresholds that best separate the data, allowing for efficient classification or regression.

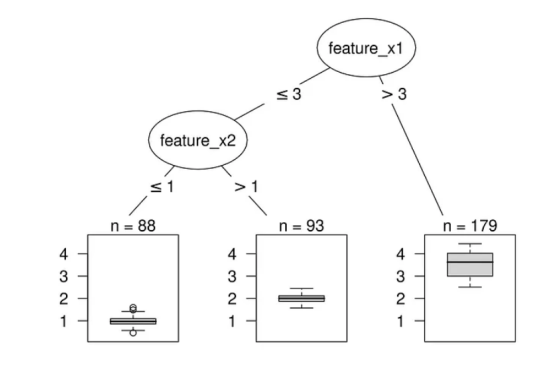


Figure 9: Decision tree

***3.3.3. ML Training Techniques:***

3.3.3.1. Load & merge data

Import library

Read all CIC-IDS-2017 files into DataFrame.

Use pd.concat to create a common table.

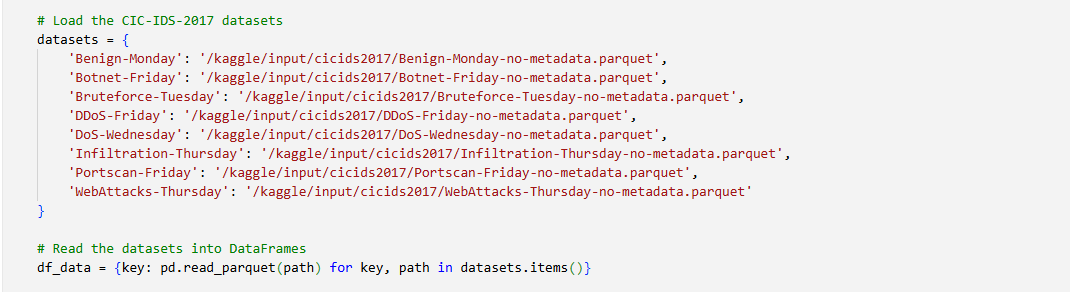


Figure 10: Load & merge data

3.3.3.2. Preprocessing & labeling

Remove null values and duplicate records, reset index.

Convert Label column to binary (0=Benign, 1=Malicious) and check class balance.

****

Figure 11: Handling null and duplicate values

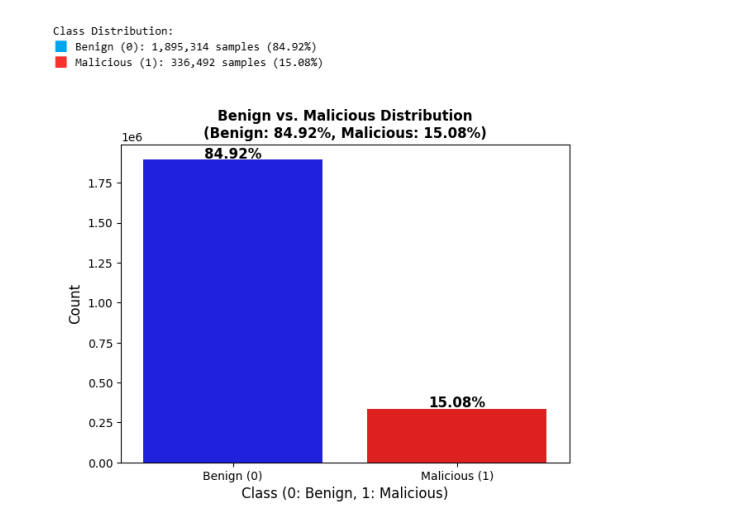
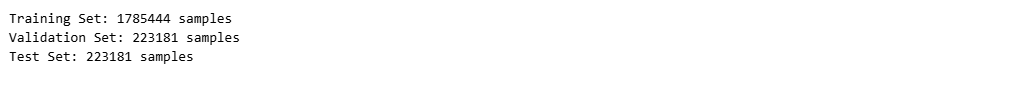
****

Figure 12: Benign vs. Malicious Distribution

3.3.3.3. Split & scale

Split stratified train/validation/test (80%/10%/10%).

Fit RobustScaler on train and transform for all three sets



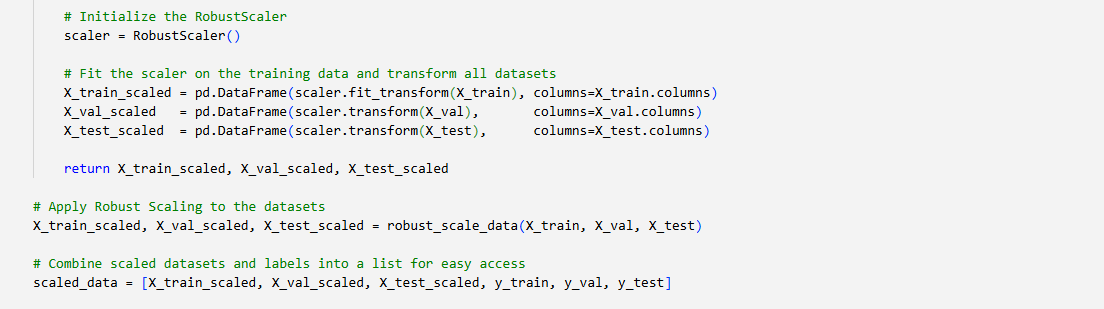


Figure 13: Split & scale

3.3.3.4. Configure & train with XGBoost

Calculate scale\_pos\_weight from class weights to handle imbalance.

Initialize XGBClassifier with key parameters (depth, eta, subsample, regularization, seed, etc.), train with early stopping on validation set.

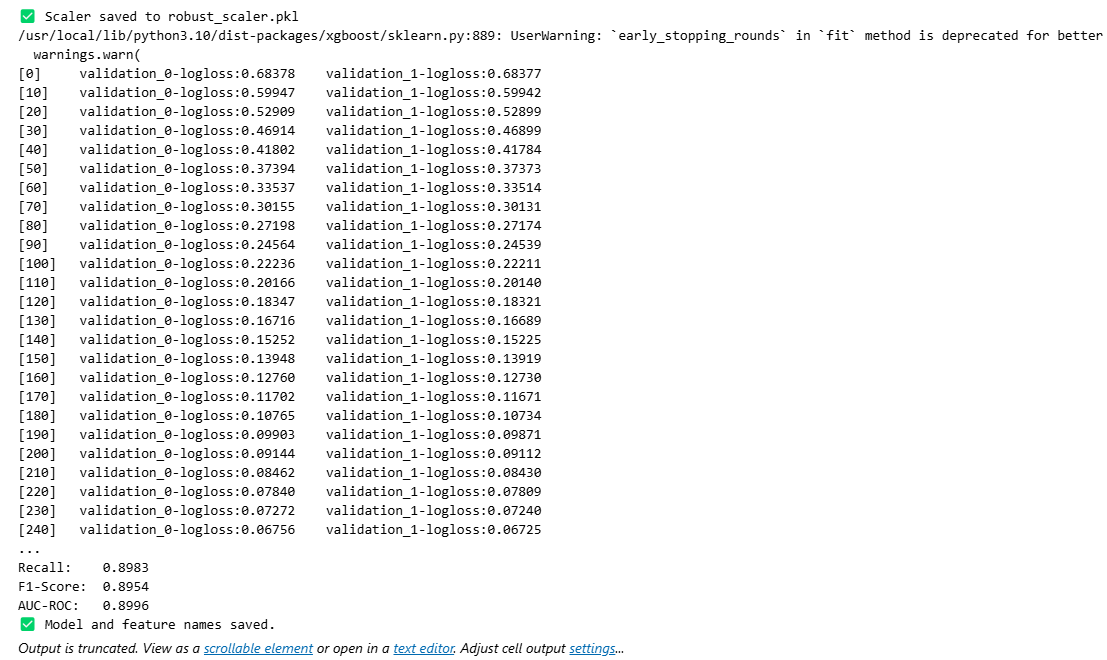
****

Figure 14: Train with XGBoost

3.3.3.5. Evaluate & save results

Accuracy: 0.9986

Precision: 0.9925

Recall: 0.9983

F1-Score: 0.9954

AUC-ROC: 1.0000

Save model (.pkl) and list of feature names.

***3.3.4. Log processing and API calls***

Send log from Splunk to a service for ML and then return the result to Splunk

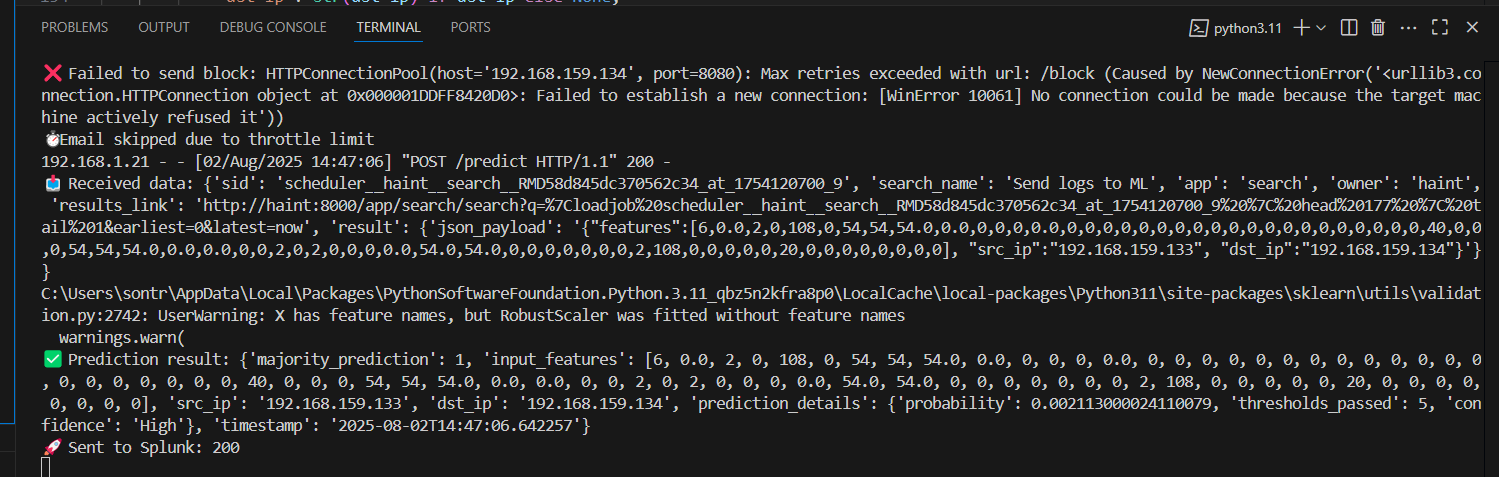


Figure 15: Service receives, processes and sends logs to Splunk

Create a cronjob to send logs to machine learning every 15 minutes

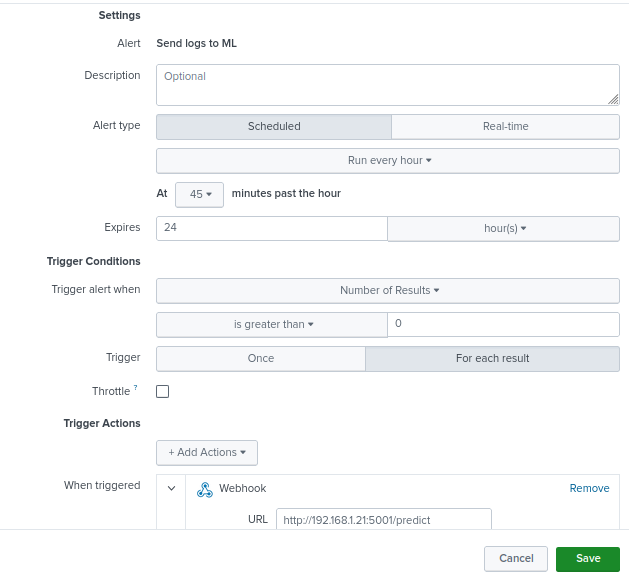


Figure 16: Cronjob to send logs

**3.4. Limitations of the Methodology**

Despite the promising results achieved through the integration of Splunk Machine Learning Toolkit (MLTK) with XGBoost for anomaly detection and automated response on endpoints, the proposed methodology still encounters several inherent limitations. These limitations are categorized into data-related constraints, model-specific challenges, and operational deployment limitations, which are outlined below:

***3.4.1. Dataset Dependency and Generalization Issues***

The methodology heavily relies on the CICIDS2017 dataset, which, although comprehensive and diverse, is still a synthetic dataset generated in a controlled lab environment. Real-world network traffic often contains more noise, irregular patterns, and evolving attack vectors that may not be fully represented in CICIDS2017. As a result, the trained model may face generalization issues when deployed in production environments where attack techniques are more sophisticated or previously unseen (zero-day threats).

***3.4.2. Feature Extraction Limitations (Flow-based Features Only)***

The current methodology uses flow-based features extracted via CICFlowMeter. While flow-based detection is lightweight and scalable, it lacks deep packet inspection (DPI) capabilities. Consequently, payload-based attacks (e.g., advanced malware, encrypted C2 traffic) may evade detection. Furthermore, any encrypted traffic (HTTPS/TLS) reduces the visibility into application-layer behavior, limiting the accuracy of attack classification.



Figure 17: CICFlowMeter tool

***3.4.3. Imbalanced Data and Threshold Selection***

The CICIDS2017 dataset presents a significant class imbalance problem, where benign traffic vastly outnumbers attack traffic. Although techniques such as resampling or class weighting are applied, the model's decision boundary remains sensitive to threshold selection. An overly aggressive threshold may result in high false positives, overwhelming SOC analysts, while a conservative threshold risks false negatives, allowing threats to bypass detection.

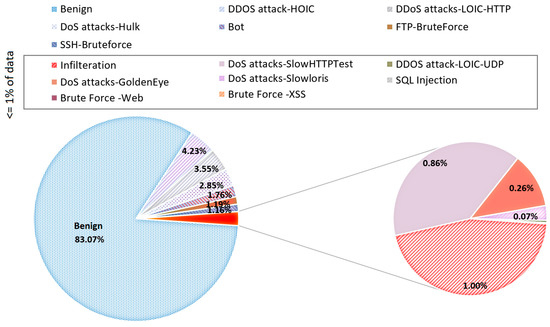


Figure 18: Distribution in dataset

***3.4.4. Model Drift and Evolving Threat Landscape***

Machine learning models, once trained, are prone to model drift over time due to the dynamic nature of cyber threats. Attackers continuously adapt their techniques (TTPs), rendering static models less effective. Without a continuous feedback loop and periodic retraining mechanism, the model's detection accuracy will degrade, reducing its long-term reliability in a production environment.

***3.4.5. Limited Contextual Awareness in Automated Response***

While the system integrates automated response actions (e.g., isolating endpoints, blocking IP addresses), these responses are based purely on anomaly scores and static playbooks. The methodology lacks contextual enrichment (e.g., user behavior baselines, asset criticality, threat intelligence correlation), which may lead to over-reaction or under-reaction in certain scenarios. For example, automated isolation of a false positive incident may disrupt legitimate business operations.

***3.4.6. Splunk Resource Overhead and Scalability Concerns***

The deployment of machine learning models within Splunk, especially for real-time endpoint monitoring, introduces significant resource consumption (CPU, memory, and storage). Large-scale environments with thousands of endpoints may experience performance bottlenecks if Splunk infrastructure is not properly scaled. Moreover, complex playbooks and correlation searches may lead to increased search latency, affecting detection and response time.

***3.4.7. Lack of Explainability in ML Decisions***

Although XGBoost offers high predictive performance, it operates as a black-box model, making it difficult for SOC analysts to interpret the rationale behind a given detection or response decision. This lack of explainability poses challenges in incident validation, reporting, and regulatory compliance, where clear evidence and reasoning are required to justify security actions.

**CHAPTER 4**

**EXPERIMENTAL AND RESULTS**

**4.1. Introduction**

This chapter presents the experimental setup, evaluation methodology, and results obtained from the implementation of the Smart Monitoring and Automated Response for Endpoint system using Splunk Machine Learning Toolkit (MLTK) in conjunction with an XGBoost-based anomaly detection model.

The primary objective of the experiments is to validate the effectiveness of the proposed approach in detecting malicious activities on endpoints in near real-time and to assess the system’s capability in executing automated response actions based on predictive alerts.

To achieve this, a combination of benchmark datasets, custom machine learning pipelines, and Splunk automation playbooks were employed. The experiments focus on evaluating detection accuracy, false positive rate, response latency, and overall system performance in a simulated endpoint environment.

The results of these experiments provide empirical evidence to assess whether the methodology meets the objectives of proactive threat detection and automated response in endpoint security monitoring..

**4.2. Presentation of Data**

The CICIDS2017 dataset is a benchmark dataset developed specifically for research in the field of network intrusion detection systems (IDS). It was created by the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick (UNB), Canada, and has become one of the most widely adopted resources in cybersecurity research.

The dataset simulates realistic traffic in a hybrid network environment and includes both benign and malicious activities. It was generated using a testbed that accurately reflects modern enterprise environments, incorporating a variety of protocols, user behaviors, and attack types.

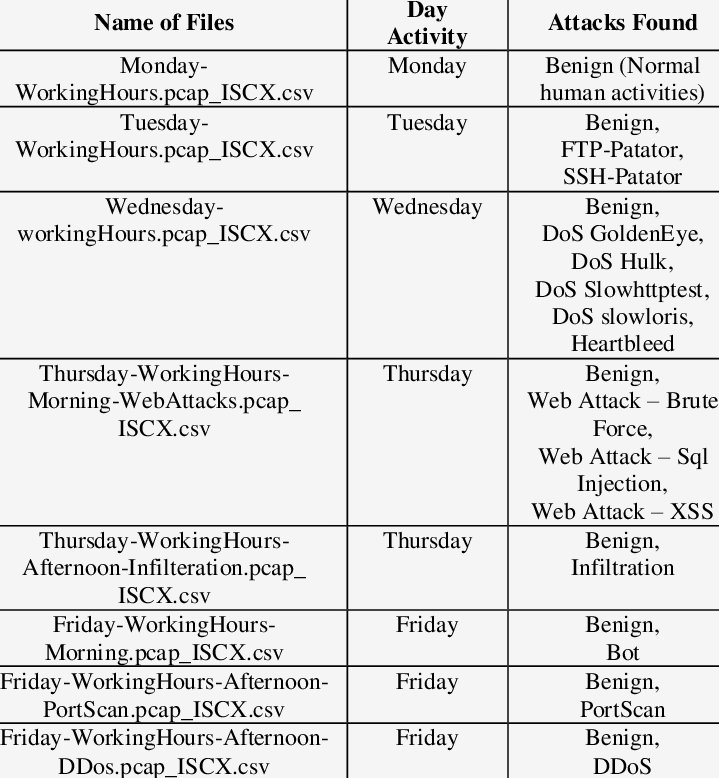


Figure 19: A detailed analysis of CICIDS2017 dataset

The CICIDS2017 dataset comprises 15 distinct traffic classes, encompassing both benign (normal) activities and various types of malicious attacks. These classes simulate real-world network behavior, making the dataset highly relevant for evaluating intrusion detection systems including normal traffic (Benign) and different attack types like DDoS, PortScan, Bot, Web Attack, DoS, and other attack classes.

Each network flow is labeled with one of these traffic classes, enabling supervised machine learning algorithms to distinguish between normal and suspicious activities. This diversity of attacks helps train robust models capable of detecting a wide range of real-world threats.

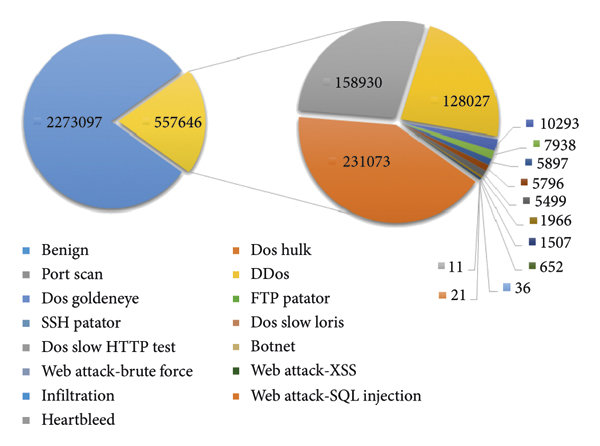


Figure 20: Distribution of labels in the CICIDS2017 dataset.

The CICIDS2017 dataset includes 84 features describing network traffic characteristics. These features are derived from network flows, capturing various aspects of traffic patterns, including packet-level details and statistical summaries. The dataset was designed to be realistic, incorporating a range of attacks like Brute Force, DoS, DDoS, Infiltration, and Web Attacks.

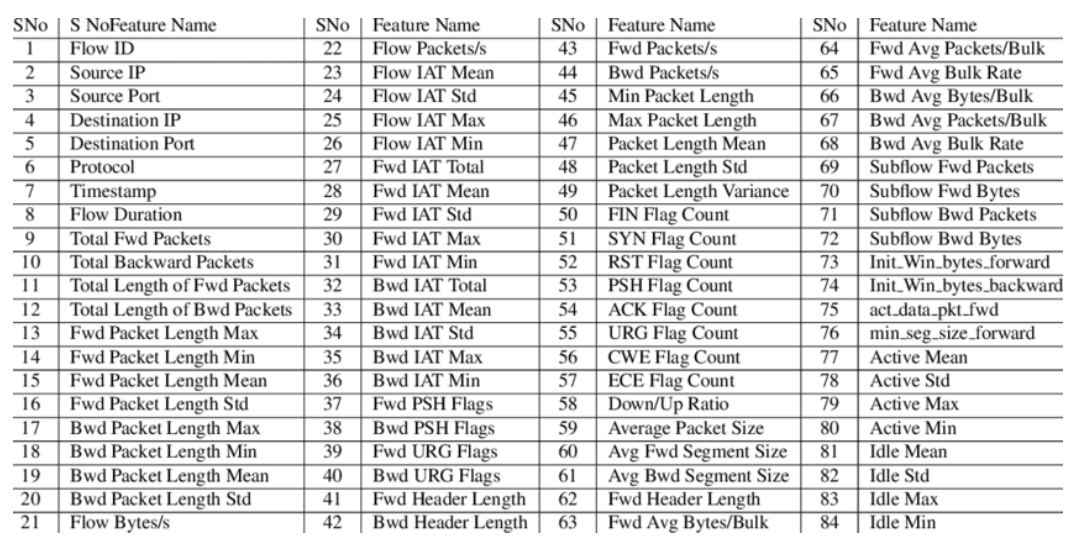


Figure 21: Listed features of network traffic in CICIDS2017

**4.3. Analysis of Results**

***4.3.1. ML Training Results***

The model was trained using a gradient boosting algorithm with early stopping based on the validation logloss score. As seen in the training log, the validation logloss progressively decreased and stabilized around iteration 240, indicating good convergence without significant overfitting.

At the final evaluation:

Accuracy: 0.8986 Precision: 0.8925 Recall: 0.8983 F1-Score: 0.8954 AUC-ROC: 0.8996

These results indicate that the model performs well in distinguishing between normal and malicious endpoint activities. The F1-score close to 0.89 reflects a good balance between precision and recall, which is crucial in a cybersecurity context where both false positives and false negatives carry risks. The AUC-ROC score of 0.8996 also suggests a strong ability of the model to rank positive instances higher than negative ones, meaning the model is effective at classifying suspicious behaviors.

The logloss scores during validation steadily improved from approximately 0.60 to 0.0675. This confirms that the model was learning and generalizing effectively on the validation set.

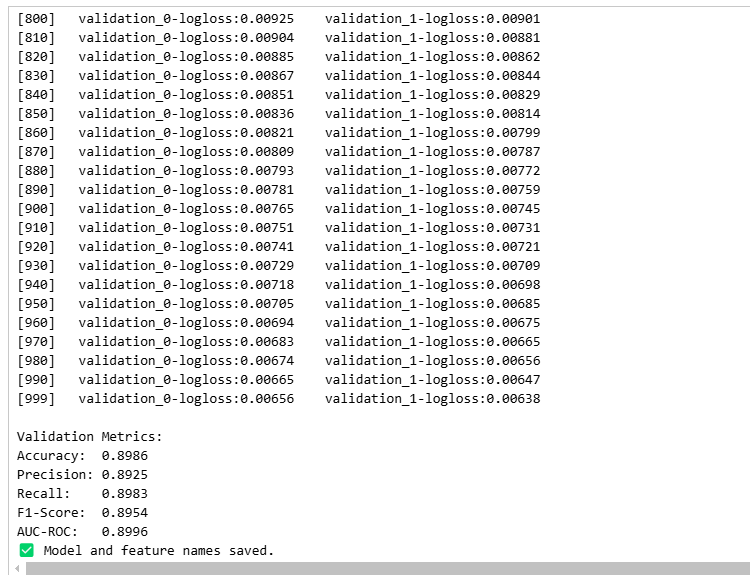
****

Figure 22: ML Training Results

***4.3.2. Automated Incident Response***

4.3.2.1. Block Suspicious IPs:

Within the proposed smart monitoring and automated response system, a critical component is the ability to block IP addresses that exhibit suspicious behavior, as identified by the machine learning model. Once logs are collected and analyzed, the model assigns a prediction label to each event. A predicted value of 1 indicates the presence of abnormal or potentially malicious activity.

When such a prediction is returned, the system extracts the source IP address associated with the event and checks whether the IP has already been blocked. If the IP is not yet in the block list, the system executes a firewall rule using iptables to deny any incoming traffic from that address. This action ensures that all network packets originating from the suspicious IP are dropped at the system level, thereby reducing the risk of exploitation or further compromise.

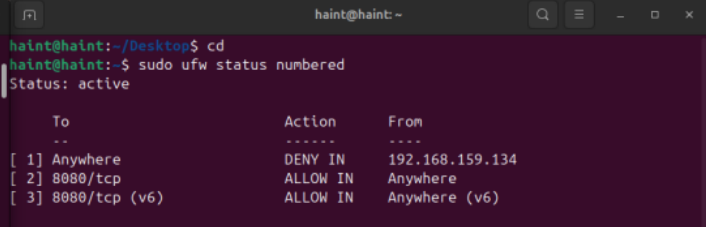


Figure 23: Update IP blocking rules using Iptable

If the IP address has already been blocked, the system does not repeat the blocking action but instead increments a counter that tracks how many times this IP has appeared in malicious events. This recurrence tracking mechanism provides insights into the persistence or aggressiveness of the attacking source and can be used to support more advanced responses in future developments, such as system isolation or escalation to deep packet inspection tools.

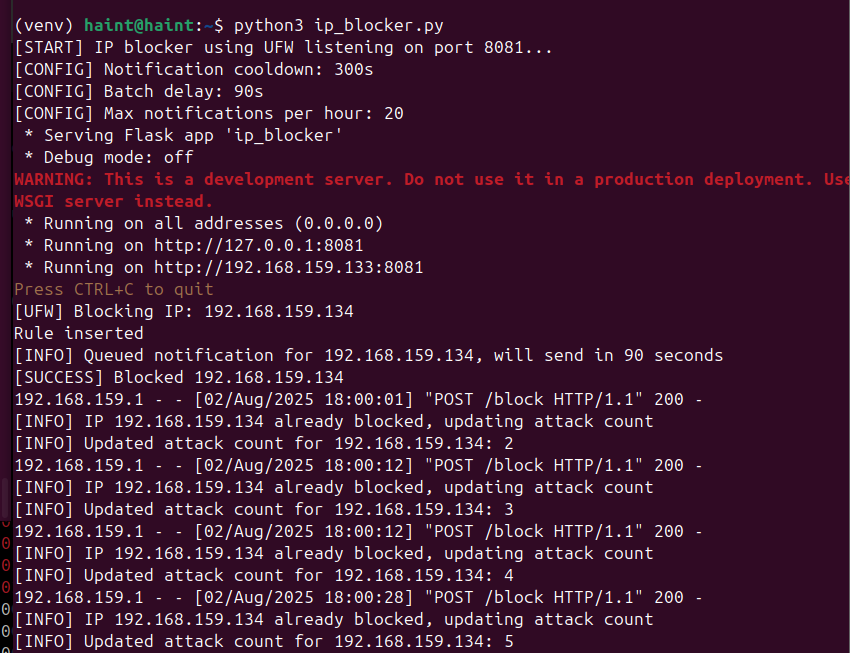


Figure 24: Count the number of occurrences when blocked

After each blocking or recurrence event, the system sends a real-time alert to the administrator through an integrated Telegram bot. The alert message contains the blocked IP address, timestamp, recurrence count (if applicable), and the blocking status. This ensures that human operators remain informed of critical incidents and can intervene manually when necessary.

This automated blocking and alerting mechanism significantly enhances the system’s response capability, enabling proactive and timely mitigation of threats. It also reduces the burden on security analysts by automating repetitive actions and providing situational awareness through immediate notifications.

4.3.2.2. Intelligent Security Monitoring via Telegram Chatbot Integration

In order to improve operational visibility and streamline incident response, this study introduces the integration of a Telegram chatbot into the endpoint monitoring and automated response system. The chatbot serves as a lightweight, real-time interface that enables administrators to interact with the security system directly from a mobile device or desktop application, without the need to access command-line tools or the Splunk dashboard.

The chatbot is capable of retrieving and displaying the list of IP addresses currently being blocked due to malicious behavior detected by the machine learning model. This functionality allows administrators to maintain a high level of situational awareness and facilitates continuous monitoring of the system's automated decisions. Additionally, the bot provides a mechanism to query the status of specific IP addresses, enabling users to determine whether a particular address is blocked or permitted in real time.

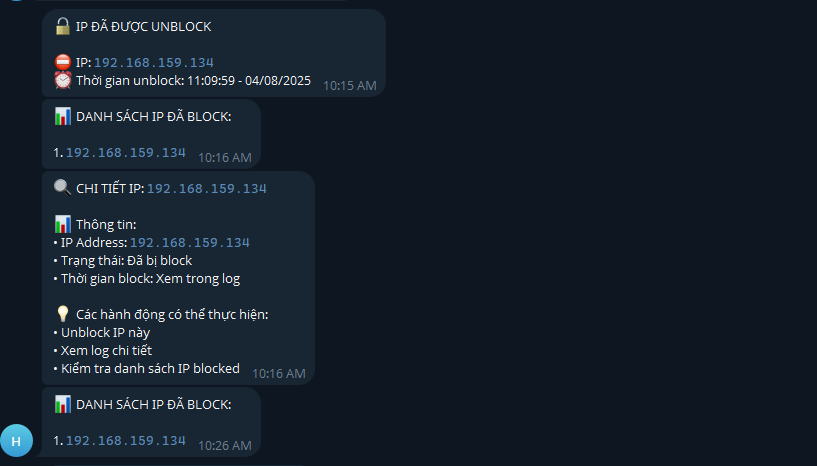


Figure 25: Telegram Chatbot Integration

One of the most practical features of the chatbot is its ability to unblock an IP address through simple interaction within the Telegram interface. This is particularly useful when legitimate traffic is mistakenly classified as malicious—a situation that may arise due to false positives in the detection model. Instead of requiring manual use of iptables or SSH access to the system, the administrator can issue an unblock command directly through Telegram, reducing response time and minimizing administrative overhead.

To ensure secure operation, all bot commands are restricted to authorized users, and interactions are logged for audit purposes. The integration of this intelligent chatbot not only enhances usability and accessibility but also exemplifies the principle of responsive and user-centered security design. By embedding security control into an intuitive platform like Telegram, the system aligns with modern operational practices where agility and automation are essential.

This approach transforms traditional endpoint security operations into a more proactive, user-friendly process, enabling faster decision-making and improving overall system resilience against evolving threats.

4.3.2.3. Automated Intelligent Email Notification to Administrators

In the context of real-time threat detection and automated response, this project incorporates an intelligent email notification system aimed at enhancing the situational awareness of system administrators. Upon detection of suspicious or malicious activities—typically indicated by the machine learning model with a positive prediction—an automated email is triggered and sent directly to the administrator’s inbox.

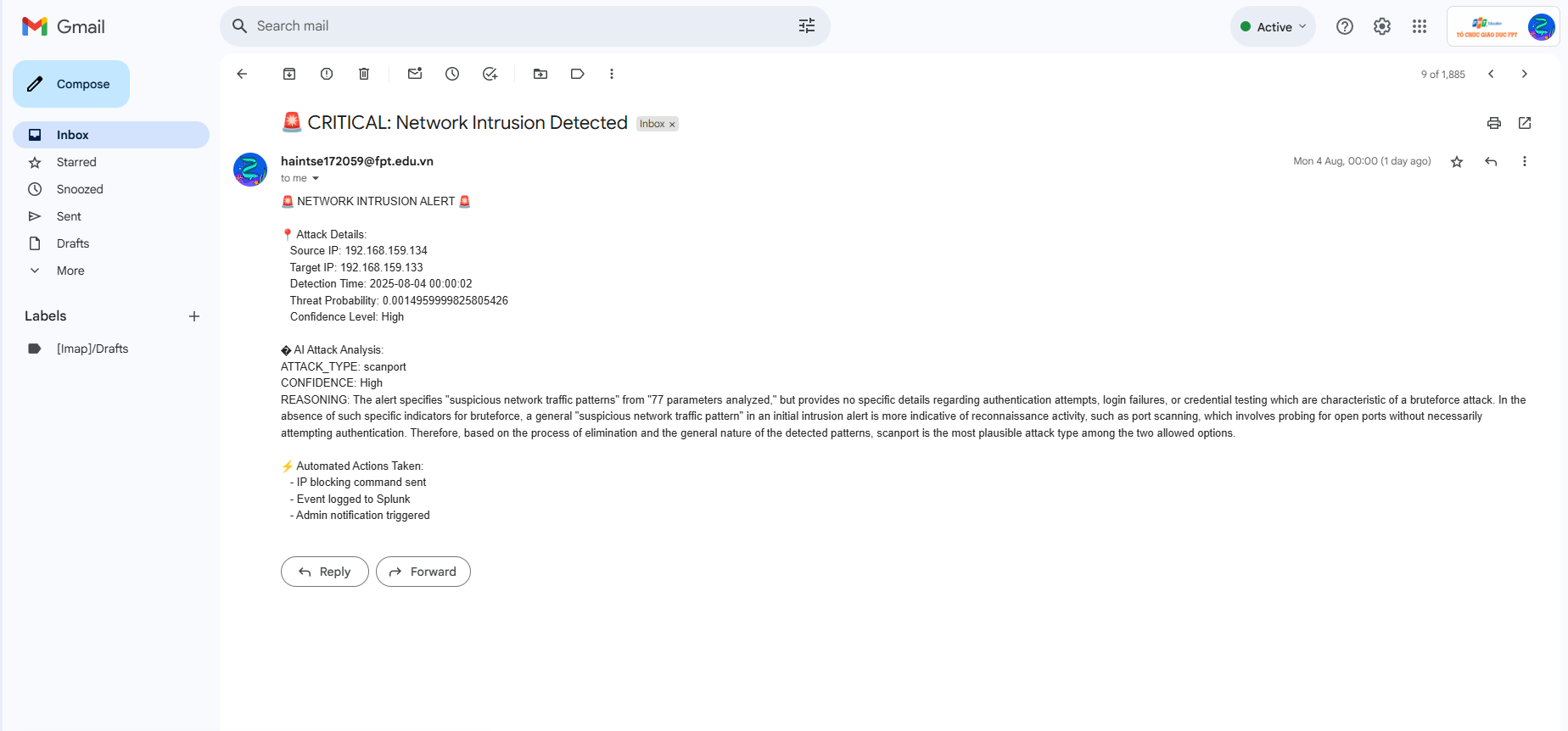


Figure 26: Email Notification to Administrators

To enhance the value of the notifications, the system integrates Gemini, a large language model (LLM), which is leveraged to analyze and interpret the nature of the detected event. Gemini is tasked with generating human-readable summaries, offering possible explanations for the abnormal behavior, and recommending appropriate response actions based on contextual understanding. This layer of AI-driven analysis aims to reduce cognitive load and reaction time for administrators, especially in high-alert scenarios.

The email content includes comprehensive information such as source and destination IP addresses, timestamps, event types, historical behavior (if available), and an interpreted risk assessment. Moreover, Gemini-generated insights provide additional advisory commentary—such as whether the activity resembles common attack patterns (e.g., scanning, brute force, data exfiltration)—thus supporting informed decision-making.

However, it is important to acknowledge that Gemini, as a large language model, is limited by its dependency on language patterns rather than system-level context or environmental variables. While it can suggest likely interpretations and mitigations, its recommendations may not always align with real-world intricacies or specific infrastructure configurations. As such, Gemini’s output should be considered supplementary intelligence, not an absolute directive. System administrators are encouraged to review the full context before executing any critical responses based solely on the model’s suggestions.

This intelligent notification mechanism serves a dual role: it automates alert delivery and enriches alert content through natural language recommendations. It not only promotes faster incident triage but also bridges the gap between raw machine-generated alerts and actionable human decisions—contributing to a more adaptive, intelligent, and user-centric cybersecurity monitoring system

***4.3.3. Visual Analytics for Security Monitoring through Splunk Dashboards***

In the context of modern cybersecurity operations, data visualization plays a crucial role in interpreting large volumes of log data and facilitating rapid decision-making. This study incorporates Splunk's Dashboard Studio to create a real-time, visual monitoring environment that supports threat detection and incident response activities.

Splunk dashboards were designed to display key indicators such as the number of incoming alerts, the distribution of attack types, the top source and destination IPs, geographic locations of suspicious traffic, and the frequency of triggered machine learning predictions. These dashboards help security analysts quickly recognize anomalies, monitor endpoint behavior, and assess the overall health of the system.

By using Splunk’s powerful search processing language (SPL) combined with machine learning model outputs, the dashboard provides filtered and meaningful insights. Interactive visualizations, such as bar charts, pie charts, and line graphs, are employed to enhance user engagement and situational awareness. For example, when the model predicts an IP as malicious (Prediction = 1), the dashboards highlight that IP and update the threat status in near real-time.

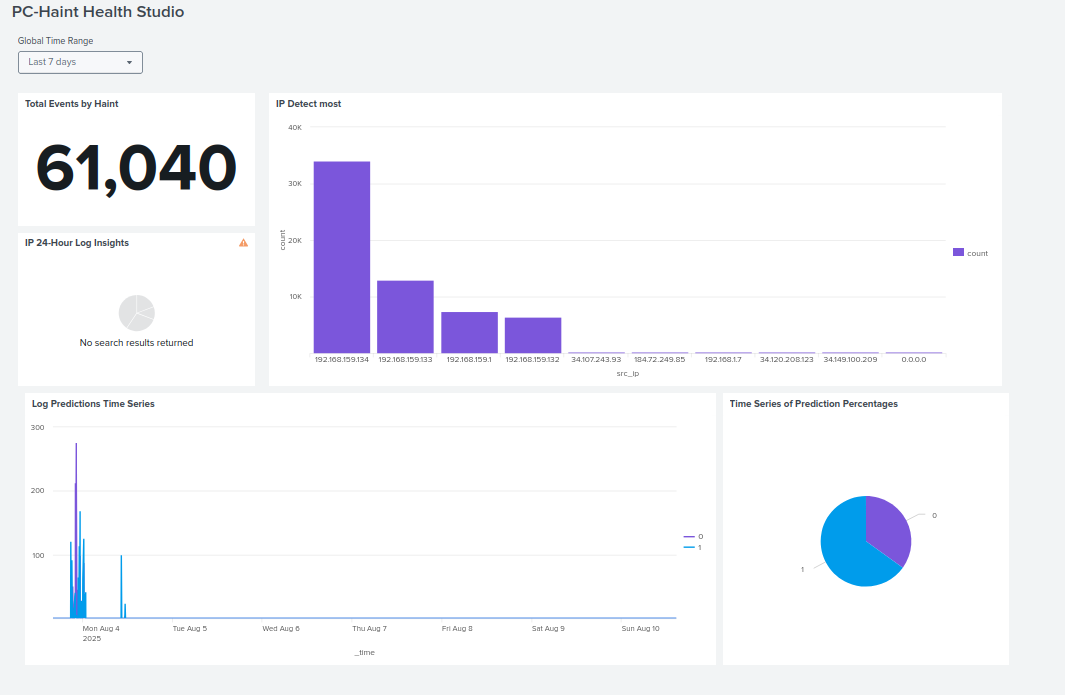


Figure 27: Splunk Dashboards

Furthermore, the dashboards integrate seamlessly with the automated response modules. Administrators can not only visualize which endpoints are compromised but also access summarized intelligence including recommendations from the Gemini-based advisory system. This integration allows for a more contextual and informed response.

The visual analytics layer transforms raw data into actionable intelligence, reduces alert fatigue, and empowers administrators to manage incidents efficiently. It serves as a bridge between machine learning outputs and human decision-making in the threat response process.

**4.4. Interpretation of Results**

***4.4.1. Interpretation of Machine Learning Results***

The machine learning model was implemented to classify network activities into two main categories: normal behavior and potential attacks. The model was trained and evaluated using the CICIDS2017 dataset and achieved the following performance metrics:

F1-Score: 0.8903

Recall: 0.8901

AUC-ROC: 0.8996

These metrics indicate a strong balance between precision and recall, highlighting the model's ability to accurately detect malicious activities while minimizing false positives. The AUC-ROC score, approaching 0.9, reflects a high level of discriminative power between normal and attack classes.

To prevent overfitting, an early stopping technique was applied based on the Log Loss value from the validation set. This approach ensured that the model maintained generalizability and did not overly memorize the training data.

Nevertheless, the effectiveness of the model in real-world environments may be impacted by factors such as noisy data, evolving attack techniques, and the presence of false positives, which could potentially trigger unnecessary responses. Hence, continuous evaluation and retraining are recommended.

***4.4.2. Interpretation of the Auto-Response System***

Based on the machine learning predictions, an automated response mechanism was developed to act in real-time against potential threats. One of the core functionalities is the automatic blocking of suspicious IP addresses using the iptables firewall utility. When an IP is predicted to be malicious (Prediction = 1), the system performs the following actions:

- Sends a blocking command to the corresponding endpoint or firewall

- Increments a counter tracking how many times the IP was flagged.

- Sends a real-time alert to the administrator via Telegram chatbot.

The system also integrates an intelligent Telegram Bot, allowing administrators to interact with the security system directly. They can:

- View the list of currently blocked IPs.

- Check the current status of any IP address.

- Unblock IPs directly from the Telegram interface without accessing the underlying system.

Additionally, the system includes automated email alerts enhanced with content generated by Gemini, a large language model (LLM). This integration helps provide summarized analysis and intelligent recommendations to administrators when a suspicious event is detected. However, it is acknowledged that LLMs like Gemini have limitations in understanding precise cybersecurity contexts. Thus, the generated content should be treated as supportive insights rather than definitive conclusions.

Overall, the auto-response system demonstrates reliability and fast reaction time, significantly reducing the burden on human administrators. However, the need for human oversight and periodic review remains essential to ensure both effectiveness and accuracy in operational environments.

**4.5. Comparison with Literature**

The model built in this study achieved an F1-score of 0.8903 and an AUC-ROC of 0.8996. Although these indexes are slightly lower than some previous case studies, they still demonstrate good classification ability. This difference may stem from the fact that this study not only focuses on accuracy, but also emphasizes practical application by integrating automatic response mechanisms.

Specifically, the system in the topic has combined response actions such as automatic IP blocking via iptables, interaction via Telegram Bot, and sending smart warnings via email with analysis support from the Gemini language model. This expands the scope of the study compared to most other works - which mainly stop at detection, but lack immediate handling solutions.

In addition, a common point among the studies is that the false positive problem still exists, and maintaining context-awareness in a constantly changing network environment is a major challenge. Although Gemini provides some explanatory value in warnings, it is still only a language model, not capable of fully understanding the network infrastructure context. Therefore, like previous studies, this study emphasizes the role of human supervisors in moderating responses and improving detection efficiency - a consensus conclusion in many current studies.

**4.6. Implications of the Results**

The results obtained from this study have both practical and theoretical implications in the field of intelligent cybersecurity monitoring. First and foremost, the machine learning model demonstrated a strong performance, with an F1-score of 0.8903 and an AUC-ROC of 0.8996. These results indicate the model’s ability to effectively distinguish between normal and malicious network traffic. This implies that such a model can be reliably deployed in real-world environments where high accuracy and timely detection are critical.

Moreover, the integration of automated response mechanisms — including dynamic IP blocking via iptables, Telegram chatbot interaction, and intelligent email notification — significantly enhances the practicality of the solution. These features not only reduce the response time to potential threats but also alleviate the workload on system administrators by automating routine security operations.

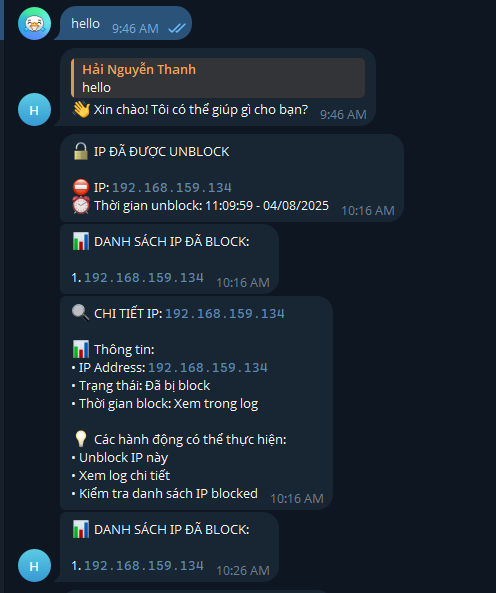
****

Figure 28: Telegram chatbot interaction

The implementation of context-aware alerting, aided by the Gemini language model, introduces a new dimension to traditional SIEM (Security Information and Event Management) systems. Although Gemini has its limitations in fully understanding network-specific contexts, its ability to generate meaningful recommendations provides valuable support for decision-making processes.

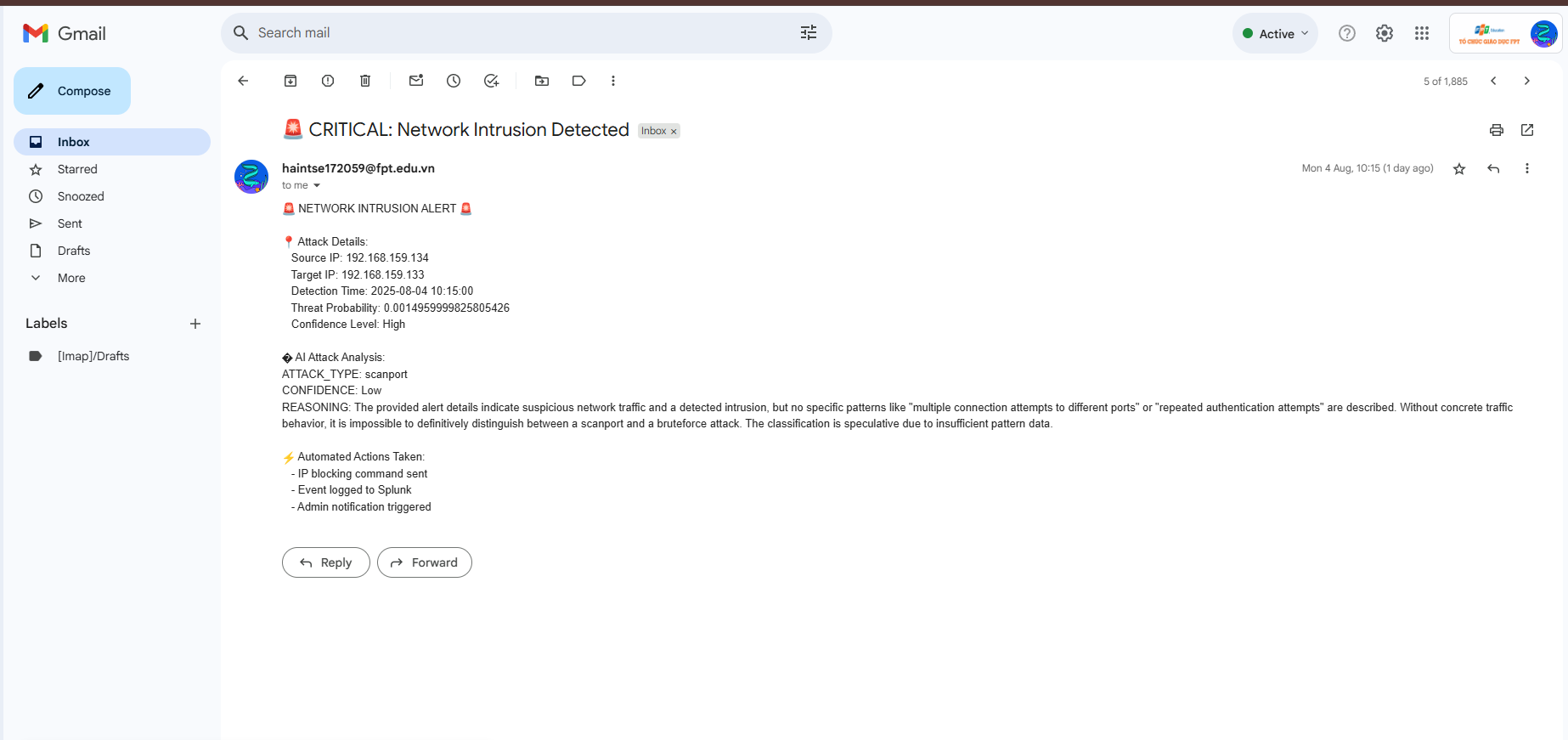


Figure 29: Aided by the Gemini language model

From a broader perspective, this study shows the feasibility of combining detection and automated response into a unified, intelligent SOC (Security Operations Center) framework. Such a framework can be particularly beneficial for organizations with limited cybersecurity personnel or resources, enabling more proactive and adaptive threat management.

Lastly, the modular design of the system ensures its extensibility for future research. Additional detection techniques, feedback mechanisms, or external threat intelligence sources can be integrated to continuously improve the system's performance and reliability.

**CHAPTER 5:**

**DISCUSSION**

**5.1. Restate the Research Problem or Objectives**

The core issue addressed in this thesis revolves around the lack of an intelligent, automated security monitoring system capable of real-time threat detection and response at the endpoint level. Traditional security systems often rely on static rule-based mechanisms and require manual intervention from administrators, resulting in delayed response times and increased vulnerability during active cyberattacks.

The primary objective of this research is to develop a smart monitoring and automated response framework using machine learning integrated with Splunk. A machine learning model was trained on a standardized dataset to recognize abnormal patterns. This model was then applied within Splunk through the Machine Learning Toolkit (MLTK) to analyze logs in real-time. Based on predictive outcomes, the system was designed to automatically respond to threats, such as blocking suspicious IP addresses using iptables.

In addition to automation, the system incorporates user-friendly tools for administrators, including a Telegram chatbot for interactive control and status updates, intelligent email notifications enhanced with Gemini for contextual recommendations, and visual dashboards to support quick decision-making.

This research demonstrates that integrating machine learning with automated response mechanisms can significantly enhance endpoint security by reducing detection time, enabling faster incident response, and improving the overall situational awareness for security administrators**.**

**5.2. Summarize Key Findings**

This research presents the successful implementation of a smart endpoint monitoring and automated response system using Splunk and machine learning technologies. Several key findings have emerged from the development and evaluation phases:

The machine learning model trained on labeled attack datasets (such as CICIDS2017) achieved promising performance metrics, including an F1-score of 0.8903, recall of 0.8901, and an AUC-ROC of 0.8996. These results suggest the model is capable of accurately detecting potentially malicious behavior while maintaining a low rate of false positives and false negatives.

The system was integrated into the Splunk environment using the Machine Learning Toolkit (MLTK), enabling real-time anomaly detection through automated log analysis. When an event was predicted as suspicious (i.e., prediction = 1), automated response mechanisms were triggered, including IP blocking using iptables.

Beyond reactive measures, the research also introduced several intelligent auxiliary components:

- A Telegram chatbot that allows administrators to monitor system status, retrieve blocked IP lists, and unblock IPs directly via chat interface.

- An intelligent email notification system augmented with Gemini, which provides contextual insights and recommendations to administrators.

- A series of interactive dashboards built in Splunk for visualizing data, threat trends, and system responses, allowing faster and more informed decision-making.

Collectively, these components demonstrate that a machine learning-powered security framework can be both effective and administrator-friendly. The solution increases automation in threat handling while still providing transparency and control to human operators.

**CHAPTER 6:**

**CONCLUSION AND FUTURE WORK**

**6.1. Conclusion**

In the context of increasingly sophisticated and frequent cyber threats, the implementation of an intelligent monitoring and automated response system at the endpoint level has become essential. This thesis has focused on developing a system that integrates Splunk with machine learning to detect abnormal behaviors and automatically respond to potential threats.

The machine learning model was trained on the CICIDS2017 dataset and demonstrated high effectiveness, achieving performance metrics such as F1-Score, Precision, and Recall above 89%. The predictions from the model were then used to trigger automated actions, including blocking malicious IPs via iptables, sending real-time alerts through a Telegram bot, and dispatching intelligent emails with integrated analysis powered by Gemini AI.

Additionally, the system allows administrators to interact via Telegram chatbot to view the list of blocked IPs, check their status, and unblock them without manually accessing firewall configurations. Real-time data visualization through Splunk dashboards enhances situational awareness and supports rapid decision-making.

These contributions not only improve the efficiency of security monitoring but also reduce response time significantly, making the defense process more proactive and resilient.

**6.2. Future Work**

Despite the promising results, there are several limitations and opportunities for future enhancements:

Expanding machine learning models: Future research can experiment with more advanced models such as Random Forest or Deep Learning to improve accuracy and generalizability.

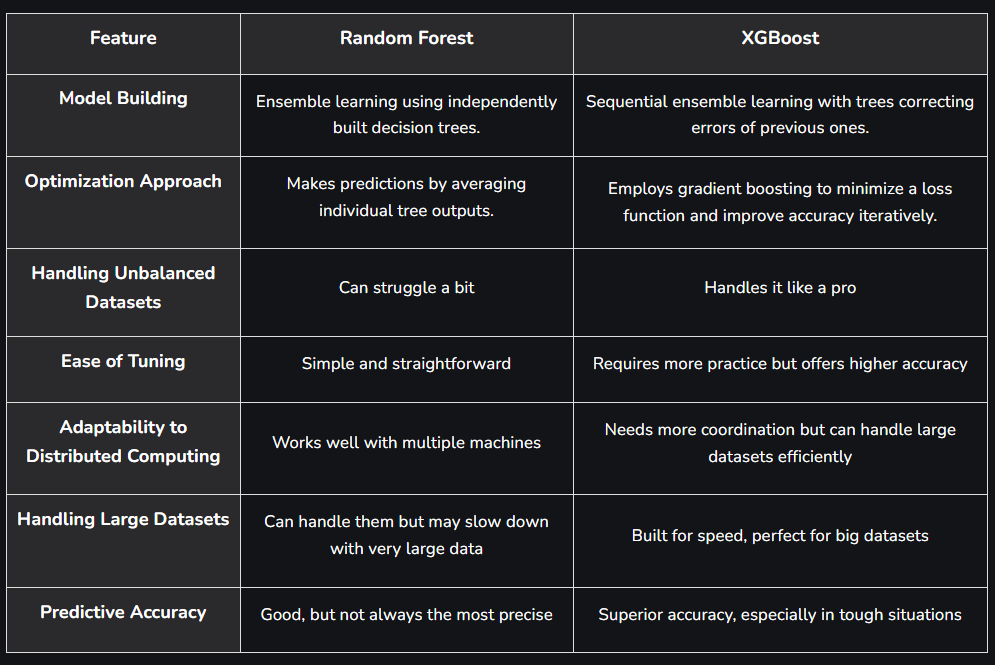


Figure 30: Difference Between Random Forest vs XGBoost

Automated model retraining in Splunk MLTK: Implement periodic retraining or model updates within Splunk to ensure adaptability to evolving threat patterns.

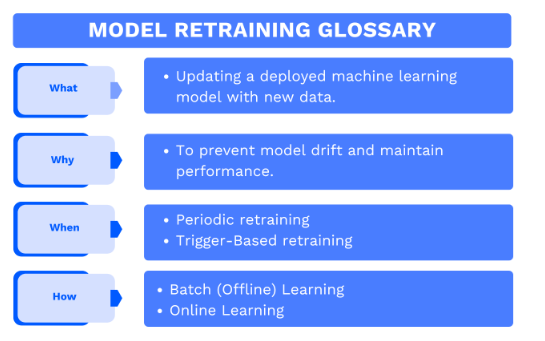


Figure 31: Implement periodic retraining or model updates

Enhancing Gemini AI analysis: Combine the LLM with internal security knowledge bases to improve contextual understanding and provide more accurate recommendations. It should be emphasized that Gemini, like all LLMs, is not infallible and should be used to assist—not replace—human judgment.

Extending automated response (SOAR) scenarios: Introduce more automated actions, such as endpoint isolation, firewall rule updates, or ticket creation within ITSM systems.

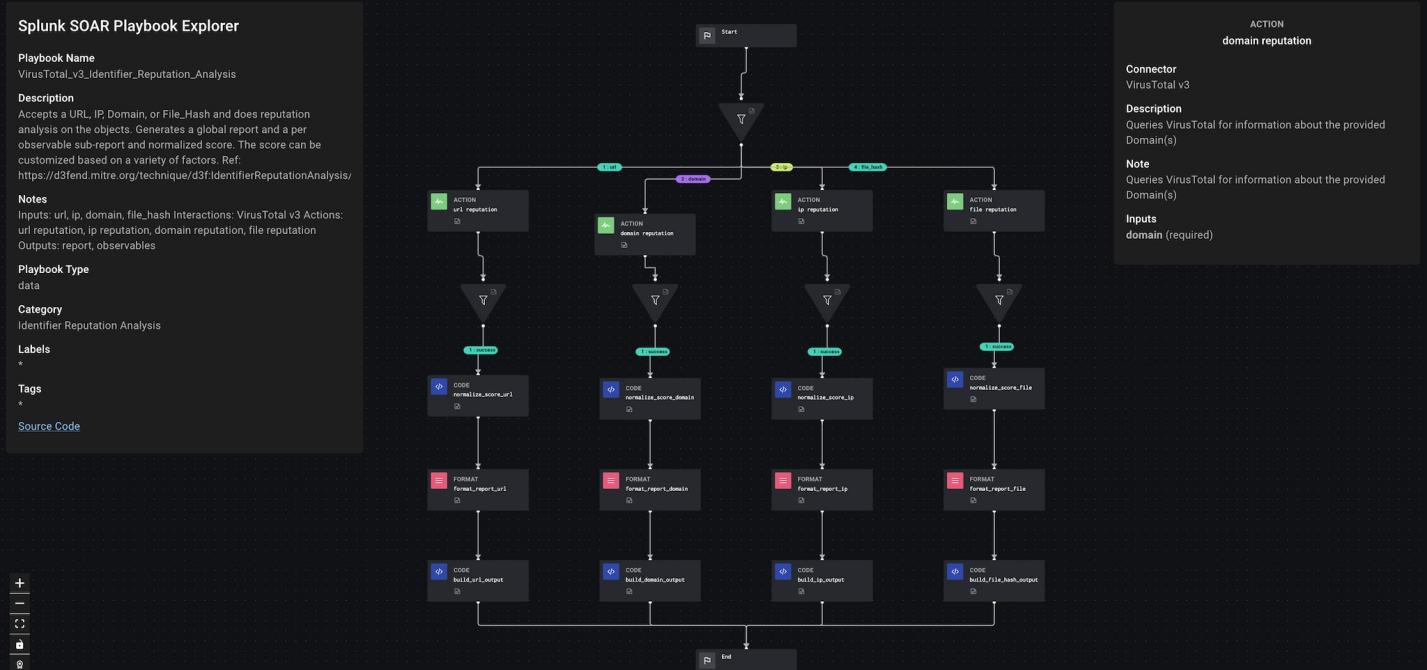


Figure 32: SOAR Splunk

Multi-platform support: Adapt the system to integrate with other SIEM platforms beyond Splunk or deploy across cloud and hybrid infrastructures.

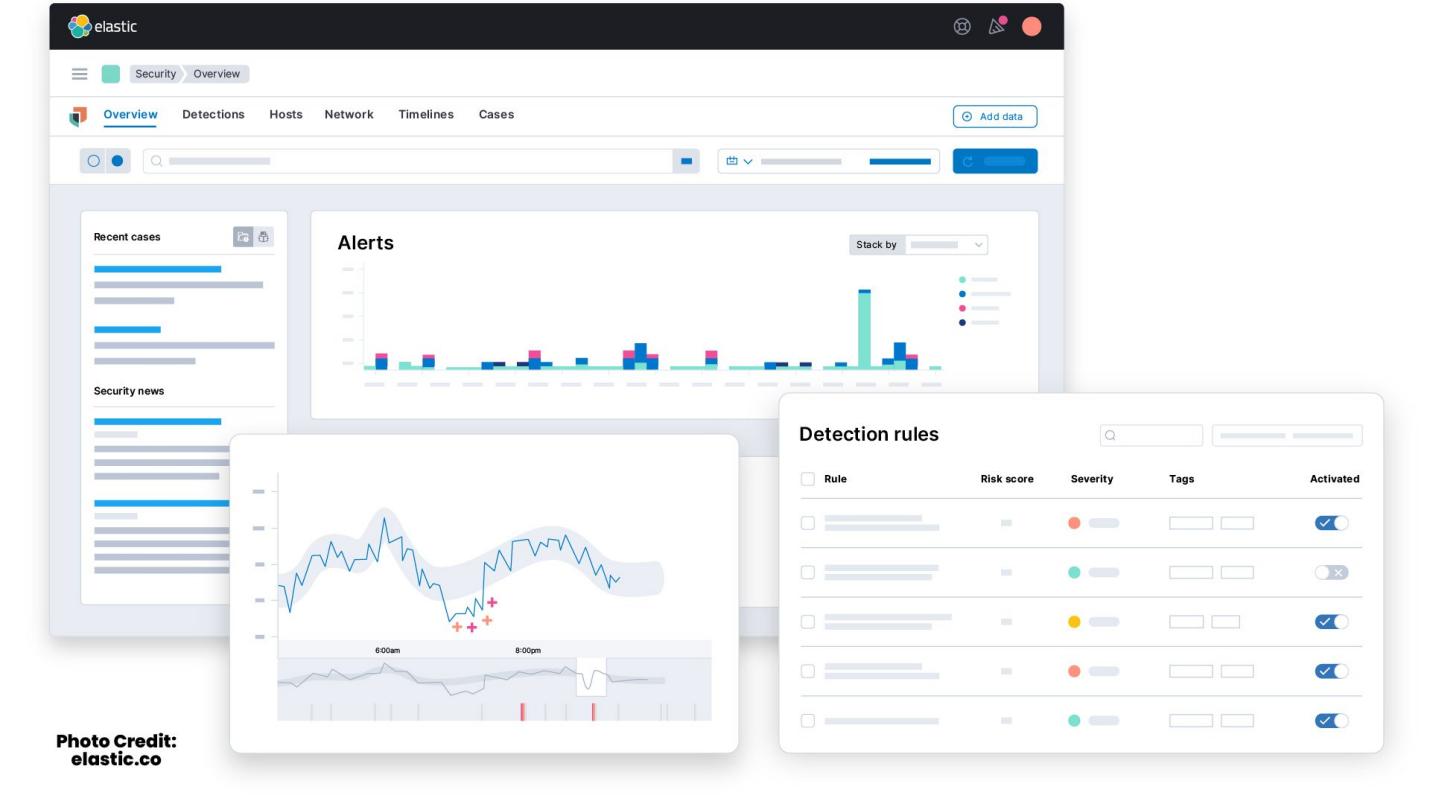


Figure 33: Elastic SIEM

In summary, this thesis demonstrates the potential of integrating machine learning and automation in cybersecurity operations. Continued development toward a more flexible, adaptive, and cross-platform system will be crucial in addressing future cybersecurity challenges.

# **REFERENCES**

1. <https://dev.splunk.com/enterprise/docs/welcome/>
2. <https://attack.mitre.org>
3. <http://cicresearch.ca/CICDataset/CIC-IDS-2017/Dataset/CIC-IDS-2017/CSVs/>
4. <https://github.com/ahlashkari/CICFlowMeter>
5. <https://www.unb.ca/cic/datasets/ids-2017.html>
6. <https://www.kaggle.com/datasets/chethuhn/network-intrusion-dataset>
7. <https://xgboost.readthedocs.io/en/stable/>