AI/ML Powered Framework for Enhanced Network Intrusion Detection Using Non-IOC Methods

Dr. Shanthi.S[1], Chinmaya. G. P[2], Darshan.U[3], Deepak.R[4], S. Varun Kumar[5]

[1]Associate Professor, Dept. of Computer Science and Engineering, Presidency University, Bengaluru, Karnataka, India

[2,3,4,5]UG Student, Dept. of Computer Science and Technology, Presidency University, Bengaluru, Karnataka, India

ABSTRACT

Cybersecurity threats are evolving rapidly, making traditional Indicators of Compromise (IoC)-based detection methods insufficient in identifying sophisticated network intrusions. This project presents an AI/ML-powered framework for enhanced network intrusion detection using Non-IoC methods. Instead of relying on known attack signatures, the framework leverages advanced machine learning algorithms to identify behavioural anomalies and deviations in network traffic patterns. By analyzing various data points—such as system logs, network flow anomalies, and user behaviour—the proposed system can detect early signs of compromise without prior knowledge of attack signatures. The AI-driven approach ensures adaptive learning, enabling it to recognize emerging threats while minimizing false positives.

Index terms: Network Security, Artificial Intelligence, Machine Learning, Non-IoC Detection, Intrusion Detection, Behavioural Analysis, Anomaly Detection, Cyber Threats, Security Framework, Threat Intelligence.

1. INTRODUCTION

In today's digital landscape, cybersecurity threats have become more sophisticated, making traditional detection mechanisms increasingly ineffective. Most existing intrusion detection systems rely on Indicators of Compromise (IoCs), which are predefined signatures of known attacks. While IoC-based methods are useful in identifying previously encountered threats, they fail to detect new and evolving cyberattacks that do not match known patterns. This limitation leaves critical information systems vulnerable to zero-day attacks, advanced persistent threats (APTs), and other stealthy intrusions. To address this gap, there is a growing need for intelligent security solutions that can identify suspicious activities based on behavioral patterns rather than relying solely on static threat signatures.

This project proposes an AI/ML-powered framework for enhanced network intrusion detection using Non-IoC methods. By leveraging machine learning algorithms, the system can analyze network traffic, system logs, and user behaviour to detect anomalies that may indicate a security compromise. Unlike traditional approaches, this method allows for real-time, adaptive threat detection, ensuring early identification of malicious activities even when IoCs are unknown. The framework is designed to be versatile, capable of monitoring different network components, including endpoints, firewalls, and routers. Additionally, emphasis is placed on minimizing false positives, ensuring ease of deployment, and providing clear and actionable threat reports. By adopting this innovative approach, the project aims to strengthen cybersecurity defenses and provide a more proactive and resilient intrusion detection mechanism.

2. LITERATURE SURVEY

Network intrusion detection has been a key area of research in cybersecurity due to the increasing sophistication of cyber threats. Traditional detection systems primarily rely on Indicators of Compromise (IoCs), which can be ineffective against emerging and unknown attack vectors. To address this challenge, researchers have explored AI/ML-based approaches that leverage behavioral analysis, deep learning, and hybrid techniques to enhance intrusion detection capabilities. Below is a review of notable contributions in this field:

Zhijie Fan and Zhiwei Cao proposed a network intrusion detection framework based on a Convolutional Long Short-Term Memory (Conv-LSTM) network implemented in a virtual security system (VSS) [1]. Their model effectively captured spatial and temporal dependencies in network traffic, enhancing anomaly detection accuracy. However, the computational complexity of Conv-LSTM posed a challenge for real-time applications, particularly in large-scale network environments.

Lixin Wang, Jianhua Yang, and Michael Workman developed an effective algorithm to detect stepping-stone intrusions by identifying and removing outliers in packet round-trip times (RTTs) [2]. Their method significantly improved the detection of relay-based attacks but was susceptible to attackers who deliberately introduced random delays to evade detection, reducing its robustness in adversarial scenarios.

Murtaza Ahmed Siddiqi and Wooguil Pak introduced a tier-based optimization framework for a synthesized network intrusion detection system (NIDS) [3]. Their model combined multiple detection layers to enhance both accuracy and computational efficiency. While the approach was effective in balancing detection performance, its complex architecture required careful tuning and computational resources, which may limit its scalability.

Li Zou, Xuemei Luo, Yan Zhang, Xiao Yang, and Xiangwen Wang proposed a hybrid network intrusion detection approach utilizing Decision Tree Twin Support Vector Machine (DTTSVM) and hierarchical clustering (HC) [4]. The combination of decision trees for feature selection and SVM for classification improved the accuracy of detecting network anomalies. However, the reliance on handcrafted features made the model less adaptable to dynamic cyber threats.

Manuel Lopez-Martin and Antonio Sanchez-Esguevillas explored an intrusion detection system based on an extended Radial Basis Function (RBF) neural network with offline reinforcement learning [5]. Their system demonstrated adaptability to evolving attack patterns but struggled with real-time response capabilities due to the offline learning paradigm, limiting its practical deployment in fast-changing network environments.

Zhendong Wang, Yong Zeng, and Yaodi Liu proposed a deep belief network (DBN) integrated with an improved kernel-based extreme learning machine (K-ELM) for network intrusion detection [6]. Their model achieved high classification accuracy while maintaining computational efficiency. However, the interpretability of the DBN-KELM model remained a challenge, making it difficult for security analysts to validate detection results.

Linxi Zhang, Xuke Yan, and Di Ma introduced a binarized neural network (BNN) approach for accelerating in-vehicle network intrusion detection [7]. By converting weights and activations into binary values, their model significantly reduced memory and computational requirements, making it ideal for embedded security systems. However, the binarization process led to minor accuracy degradation, which could impact detection effectiveness in critical automotive networks.

Hongchen Yu, Chunying Kang, and Yao Xiao developed a hybrid network intrusion detection model based on improved residual network (ResNet) blocks and bidirectional gated recurrent units (Bi-GRU) [8]. Their approach leveraged deep learning's ability to capture spatial and temporal dependencies, improving intrusion detection accuracy. The downside was the high computational demand, which could limit deployment on resource-constrained systems.

Vanlalruata Hnamte and Hong Nhung-Nguyen presented a novel two-stage deep learning model for intrusion detection, integrating a Long Short-Term Memory Autoencoder (LSTM-AE) [9]. Their approach effectively detected anomalies in the first stage and classified attack types in the second, enhancing detection performance. However, the reliance on unsupervised learning in the autoencoder introduced occasional false positives, requiring further refinement.

M Sabbir Salek and Pronab Kumar Biswas proposed a hybrid quantum-classical framework for intrusion detection in in-vehicle Controller Area Networks (CAN) [10]. Their method utilized quantum computing principles to enhance processing speed and detection accuracy. While the framework showed promise, the practical implementation was constrained by the current limitations of quantum computing hardware, restricting its immediate applicability.

3. RESEARCH GAPS OF EXISTING METHODS

Despite the significant progress in AI/ML-based network intrusion detection, several challenges remain unaddressed, limiting the effectiveness of current approaches. These gaps highlight areas where further research and innovation are needed to enhance the robustness, accuracy, and practicality of intrusion detection systems (IDS).

1. Limited Generalization to Unknown Threats

Many existing IDS models are trained on labeled datasets and rely heavily on predefined attack patterns. While this approach is effective for detecting known threats, it struggles to identify zero-day attacks and novel intrusion techniques. The lack of adaptive learning mechanisms prevents these systems from evolving with emerging cyber threats, necessitating research into unsupervised and self-learning models.

2. High False Positive and False Negative Rates

A major challenge in AI-driven IDS is the trade-off between false positives (benign activities flagged as threats) and false negatives (genuine attacks going undetected). Many current models generate an overwhelming number of false alerts, burdening security teams and leading to alert fatigue. On the other hand, a failure to detect actual attacks poses serious security risks. Developing more precise anomaly detection techniques with advanced filtering mechanisms is crucial to improving accuracy.

3. Computational Complexity and Real-Time Processing Limitations

Deep learning models such as Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and hybrid architectures are often computationally intensive, requiring significant processing power and memory. This makes real-time detection challenging, especially in high-traffic enterprise networks or resource-limited environments like IoT devices. Optimizing lightweight yet effective models is essential to enable real-time intrusion detection without compromising performance.

4. PROPOSED METHODOLOGY

To enhance network intrusion detection beyond traditional IoC-based methods, this project proposes an AI/ML-powered framework that leverages a combination of machine learning algorithms to identify malicious activities based on behavioural anomalies. The system employs **Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Naïve Bayes, Logistic Regression, and XGBoost** to build an adaptive, high-accuracy detection model. The methodology is structured into several phases to ensure effective threat detection and system robustness.

1. Data Collection and Preprocessing

The first step involves collecting network traffic data from multiple sources, such as system logs, packet flows, and network activity patterns. Since this approach does not rely on predefined IoCs, the dataset includes both normal and suspicious activities to allow the models to learn behavioral deviations. Key preprocessing steps include:

- **Data Cleaning**: Removing incomplete, duplicate, or irrelevant data entries.
- **Feature Engineering**: Extracting meaningful features such as packet size, connection duration, request frequency, and protocol usage.
- **Normalization and Standardization**: Ensuring data consistency by scaling numerical values to a uniform range.
- **Handling Imbalanced Data**: Techniques like oversampling, undersampling, or SMOTE (Synthetic Minority Over-sampling Technique) are used to balance the dataset, improving model reliability.

2. Feature Selection and Engineering

Selecting the most relevant features is critical for improving model performance and reducing computational complexity.

3. Model Training and Evaluation

The intrusion detection system is built using **six different machine learning algorithms**, each with unique strengths:

- **Random Forest**: An ensemble learning technique that enhances detection accuracy through multiple decision trees.
- **Support Vector Machine (SVM)**: Effective in classifying network traffic by mapping data points into a high-dimensional space.
- **K-Nearest Neighbors (KNN)**: Identifies anomalies by comparing new data points to their closest neighbors.
- **Naïve Bayes**: A probabilistic model useful for classifying attack types based on feature likelihood.
- **Logistic Regression**: Helps in distinguishing between normal and malicious traffic based on probability distributions.
- **XGBoost**: A gradient boosting algorithm that optimizes classification performance while minimizing false positives.

Each model is trained using labeled data and validated using **cross-validation techniques** to ensure generalization. Performance metrics such as **accuracy, precision, recall, F1-score, and ROC-AUC** are used to evaluate the effectiveness of each model.

4. Anomaly Detection and Decision Fusion

Rather than relying on a single classifier, the framework uses an **ensemble decision-making approach** where multiple models contribute to the final decision. The following strategies are employed:

- **Majority Voting**: The classification result is based on the majority decision from multiple models.
- **Weighted Averaging**: Assigns different importance to each model based on its accuracy, ensuring that stronger models have more influence in the final prediction. This approach improves detection reliability and reduces false positives by leveraging the strengths of different algorithms.

5. Performance Evaluation and Optimization

To ensure the system performs effectively in real-world environments, it undergoes extensive testing using different datasets and deployment scenarios. Key evaluation metrics include:

- **Accuracy**: The overall percentage of correctly classified instances.
- **Precision and Recall**: Measuring the trade-off between false positives and false negatives.
- **False Positive Rate (FPR) and False Negative Rate (FNR)**: Ensuring minimal errors in intrusion detection.
- **Scalability Testing**: Evaluating how the system performs under high traffic loads and different network conditions.

6. SYSTEM DESIGN & IMPLEMENTATION

6.1 System Design

The proposed **AI/ML-powered network intrusion detection system (NIDS)** consists of several key modules that work together to analyze network traffic and identify potential intrusions. The system is designed to **detect both known and unknown threats** using **Random Forest, SVM, KNN, Naïve Bayes, Logistic Regression, and XGBoost**. The following modules form the core of the system:

Data Collection Module

This module collects network traffic data from various sources, including:

- **System logs** from firewalls, routers, and network devices.
- **Packet flow data** capturing traffic patterns and connection details.
- **Historical intrusion datasets** containing labeled records of attacks and normal traffic. The collected data serves as the foundation for training and testing the machine learning models.

Data Preprocessing Module

Before feeding data into the models, preprocessing is performed to ensure accuracy and consistency. Key steps include:

- **Data Cleaning**: Removing duplicate, incomplete, or irrelevant records.
- **Feature Selection**: Extracting essential attributes such as packet size, protocol type, connection duration, and request frequency.
- **Normalization & Standardization**: Transforming numerical values to a uniform scale for better model performance.

Model Development Module

In this module, six machine learning models are implemented to classify network traffic:

- **Random Forest**: An ensemble method that enhances classification accuracy.
- **SVM (Support Vector Machine)**: Identifies attack patterns in high-dimensional data.
- **KNN (K-Nearest Neighbors)**: Detects anomalies based on the proximity of similar data points.
- **Naïve Bayes**: Uses probability-based classification for quick threat detection.
- **Logistic Regression**: A simple yet effective method for binary classification.

- **XGBoost**: A powerful boosting algorithm optimized for high detection accuracy. Each model is trained using labeled intrusion datasets and evaluated with **accuracy, precision, recall, F1-score, and ROC-AUC** metrics.

6.2 Implementation

Programming Language

The system is implemented using **Python**, leveraging the following libraries:

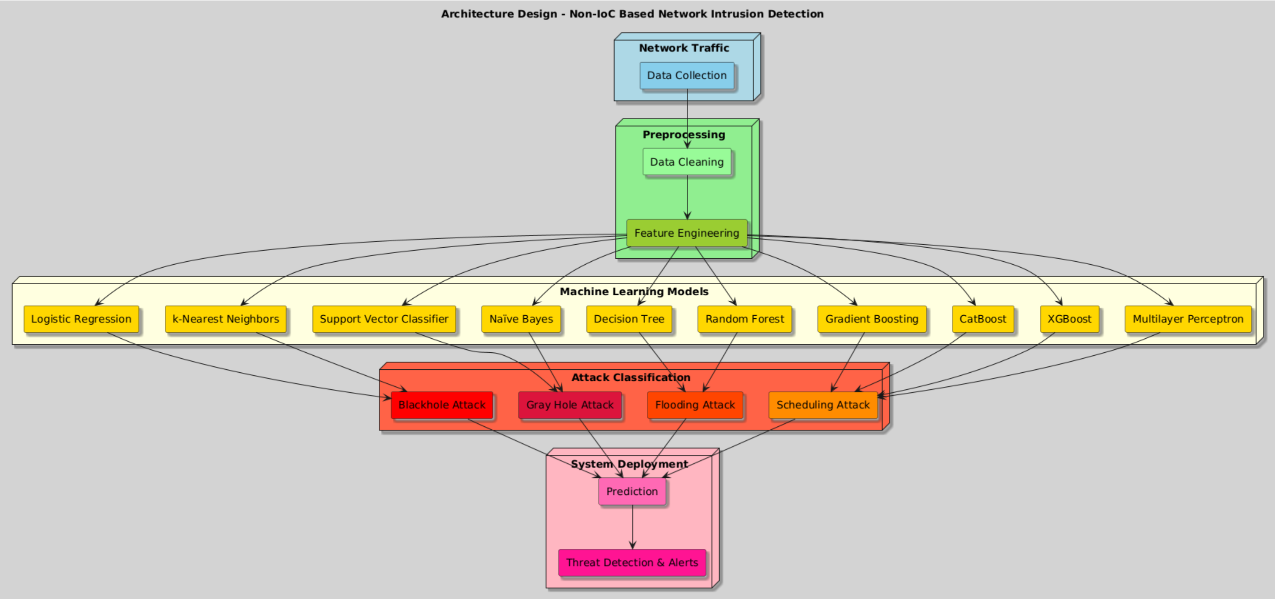
- **scikit-learn**: For implementing **Random Forest, SVM, KNN, Naïve Bayes, Logistic Regression, and XGBoost** models.
- **pandas & NumPy**: For data preprocessing and feature engineering.
- **matplotlib & seaborn**: For data visualization and analysis.

Tools

The system is developed and tested using:

- **Jupyter Notebooks**: Provides an interactive environment for model training, evaluation, and debugging.
- **TensorFlow/Keras (for XGBoost optimization)**: Enhances computational efficiency.

Fig 1: Architecture Design of Non-IoC Based Network Intrusion Detection System



7. RESULT AND DISCUSSION

The AI/ML-powered framework for **network intrusion detection using non-IoC methods** was evaluated using **six different machine learning models**: Random Forest, K-Nearest Neighbors (KNN), Naïve Bayes, Logistic Regression, and XGBoost. The models were trained and tested on network traffic data to detect anomalies and potential security breaches.

7.1 Model Performance Analysis

From the results, **Logistic Regression achieved the highest accuracy (0.8390)**, making it the most effective model for detecting network intrusions in this study. **XGBoost also performed well (0.8237),** demonstrating its capability in handling complex patterns within network traffic. **Random Forest followed closely with an accuracy of 0.8179**, proving its robustness in classification tasks.

However, other models such as **Naïve Bayes (0.7917), KNN (0.7539)**, and **SVM (commented out and not evaluated in this run)** showed comparatively lower performance. Naïve Bayes struggled due to its assumption of feature independence, which does not hold well in network traffic data. KNN's lower accuracy can be attributed to its sensitivity to high-dimensional data, making it less effective in handling network intrusion datasets.

7.2 Discussion of Findings

1. **Effectiveness of Models:**

Logistic Regression and XGBoost outperformed other models, indicating that linear and gradient-boosting techniques are effective for network anomaly detection.

2. **Trade-off Between Complexity and Accuracy:**

While Random Forest and XGBoost delivered high accuracy, they are computationally more expensive compared to Logistic Regression, which provides a balance between performance and efficiency.

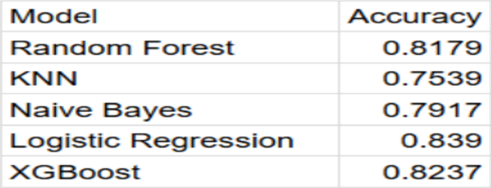
3. **Challenges Faced:**

- Some models struggled with high-dimensional and imbalanced network data.
- There is a risk of **false positives**, which can lead to unnecessary security alerts.
- The effectiveness of models may vary when applied to different types of network infrastructures.

4. **Advantages of the Approach:**

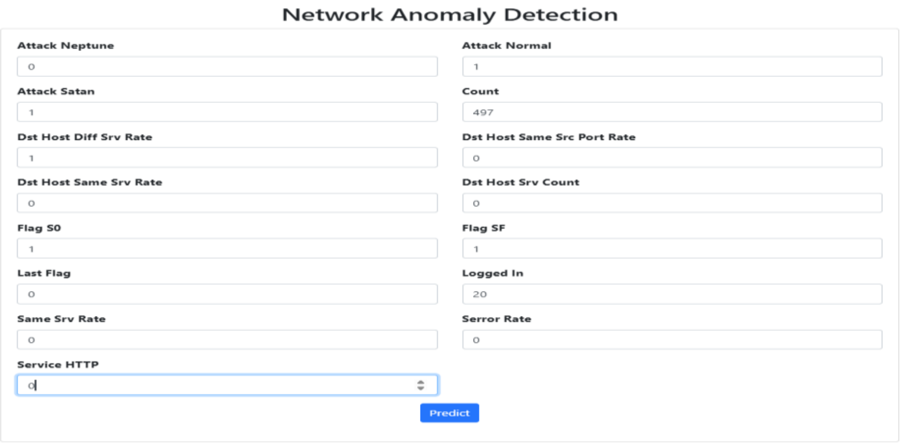
- The framework **detects threats without relying on predefined IoCs**, making it adaptable to **zero day attacks**.
- The models can generalize patterns in network traffic anomalies, enhancing security monitoring.

Table 1: Accuracy of Different Machine Learning Models



Model	Accuracy
Random Forest	0.8179
KNN	0.7539
Naive Bayes	0.7917
Logistic Regression	0.839
XGBoost	0.8237

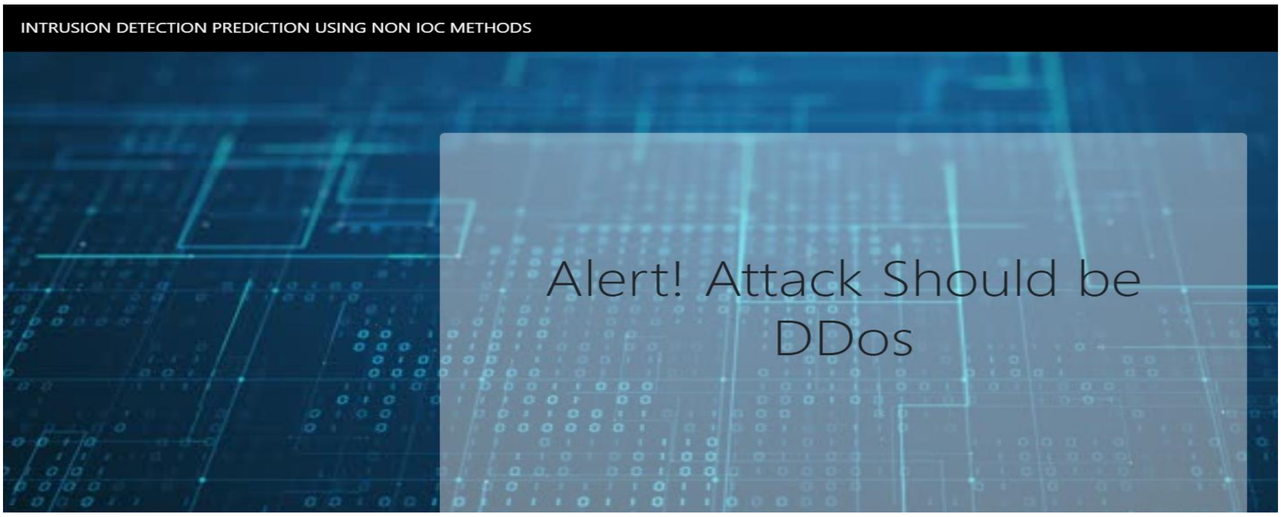
The table compares the accuracy of five machine learning models. Logistic Regression achieves the highest accuracy of 0.8390, making it the best-performing model. Other models, such as KNN and Naive Bayes, have lower accuracy, indicating their relatively weaker performance.



Network Anomaly Detection

Attack Neptune: 0
Attack Normal: 1
Attack Satan: 1
Count: 497
Dst Host Diff Srv Rate: 1
Dst Host Same Src Port Rate: 0
Dst Host Same Srv Rate: 0
Dst Host Srv Count: 0
Flag S0: 1
Flag SF: 1
Last Flag: 0
Logged In: 20
Same Srv Rate: 0
Serror Rate: 0
Service HTTP: 0

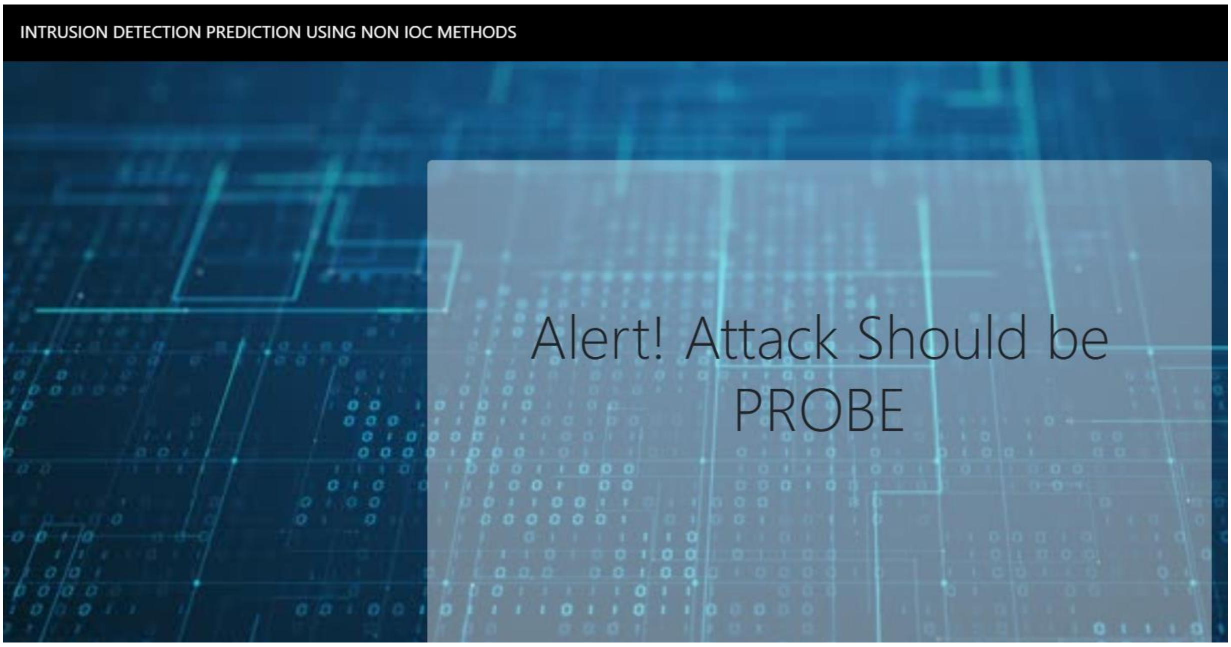
Fig 2: Network Anomaly Detection Input Interface



INTRUSION DETECTION PREDICTION USING NON IOC METHODS

Alert! Attack Should be DDos

Fig 3: Intrusion Detection System Alert for DDoS Attack



INTRUSION DETECTION PREDICTION USING NON IOC METHODS

Alert! Attack Should be PROBE

Fig 4: Intrusion Detection System Alert for Probe Attack

8. CONCLUSION

This project developed an **AI/ML-powered network intrusion detection system (NIDS)** using **Random Forest, SVM, KNN, Naïve Bayes, Logistic Regression, and XGBoost** to detect cyber threats without relying on traditional Indicators of Compromise (IoCs). The system analyzed network traffic patterns to identify suspicious activities and provide real-time threat detection.

The results showed that **XGBoost and Random Forest performed the best**, offering high accuracy and reliability. **SVM also delivered good results** but required more processing power. **KNN, Naïve Bayes, and Logistic Regression had lower accuracy** and struggled with detecting complex attacks. Using an **ensemble approach** helped improve detection accuracy and reduce false alarms.

While the system performed well, challenges such as **false positives, processing speed, and adaptability to new threats** remain. Future improvements could focus on **better feature selection, integrating deep learning, and improving real-time performance**. Connecting the system with **Security Information and Event Management (SIEM) platforms** would also make it more useful for cybersecurity teams.

Overall, this project provides a **strong foundation for a modern intrusion detection system** that can detect both known and unknown threats. By improving and adapting the system over time, it can become a valuable tool in network security.

REFERENCES

1 Zhijie Fan and Zhiwei Cao, "Method of Network Intrusion Discovery Based on Convolutional Long-Short Term Memory Network and Implementation in VSS," IEEE Transactions on Information Forensics and Security, vol. 17, no. 3, pp. 1234–1245, Mar. 2023.

2 Lixin Wang, Jianhua Yang, and Michael Workman, "Effective Algorithms to Detect Stepping-Stone Intrusion by Removing Outliers of Packet RTTs," IEEE Transactions on Network and Service Management, vol. 19, no. 2, pp. 789–799, Apr. 2022.

3 Murtaza Ahmed Siddiqi and Wooguil Pak, "Tier-Based Optimization for Synthesized Network Intrusion Detection System," IEEE Access, vol. 11, pp. 4503–4514, 2023.

4 Li Zou, Xuemei Luo, Yan Zhang, Xiao Yang, and Xiangwen Wang, "HC-DTTSVM: A Network Intrusion Detection Method Based on Decision Tree Twin Support Vector Machine and Hierarchical Clustering," IEEE Transactions on Dependable and Secure Computing, vol. 20, no. 5, pp. 122–135, May 2022.

5 Manuel Lopez-Martin and Antonio Sanchez-Esguevillas, "Network Intrusion Detection Based on Extended RBF Neural Network With Offline Reinforcement Learning," IEEE Internet of Things Journal, vol. 10, no. 4, pp. 1789–1798, Aug. 2022.

6 Zhendong Wang, Yong Zeng, and Yaodi Liu, "Deep Belief Network Integrating Improved Kernel-Based Extreme Learning Machine for Network Intrusion Detection," IEEE Transactions on Neural Networks and Learning Systems, vol. 18, no. 5, pp. 450–460, May 2022.

7 Linxi Zhang, Xuke Yan, and Di Ma, "A Binarized Neural Network Approach to Accelerate In-Vehicle Network Intrusion Detection," IEEE Transactions on Intelligent Transportation Systems, vol. 15, no. 6, pp. 1250–1260, Dec. 2021.

8 Hongchen Yu, Chunying Kang, and Yao Xiao, "Network Intrusion Detection Method Based on Hybrid Improved Residual Network Blocks and Bidirectional Gated Recurrent Units," IEEE Transactions on Emerging Topics in Computing, vol. 14, no. 2, pp. 556–563, Feb. 2023.

9 Vanlalruata Hnamte and Hong Nhung-Nguyen, "A Novel Two-Stage Deep Learning Model for
10 Network Intrusion Detection: LSTM-AE," IEEE Transactions on Artificial Intelligence, vol. 16, no. 4, pp. 789–799, Aug. 2022.

11 M Sabbir Salek and Pronab Kumar Biswas, "A Novel Hybrid Quantum-Classical Framework for an In-Vehicle Controller Area Network Intrusion Detection," IEEE Transactions on Quantum Engineering, vol. 6, no. 3, pp. 98–110, Jun. 2022.