Network Intrusion Monitoring System

Yash Phadatare
Department of Computer Science and Design
New Horizon Institute of Technology & Management, University of Mumbai
Thane, India
yashphadatare217@nhitm.ac.in

Sanju Khetawat
Department of Computer Science and Design
New Horizon Institute of Technology & Management, University of Mumbai
Thane, India
sanjukhetawat217@nhitm.ac.in

Niyati Manerikar
Department of Computer Science and Design
New Horizon Institute of Technology & Management, University of Mumbai
Thane, India
niyatimanerikar217@nhitm.ac.in

Kavin Naik
Department of Computer Science and Design
New Horizon Institute of Technology & Management, University of Mumbai
Thane, India
kavinnaik217@nhitm.ac.in

Niranjan Kulkarni
Department of Computer Science and Design
New Horizon Institute of Technology & Management, University of Mumbai
Thane, India
niranjankulkarni@nhitm.ac.in

Vilas Mapare
Department of Mechatronics
New Horizon Institute of Technology & Management, University of Mumbai
Thane, India
vilasmapare@nhitm.ac.in

Priyadarshini Badgujar
Department of Computer Science and Design
New Horizon Institute of Technology & Management, University of Mumbai
Thane, India
priyadarshinibadgujar@nhitm.ac.in

Swati Patil
Department of Computer Science and Design
New Horizon Institute of Technology & Management, University of Mumbai
Thane, India
swatipatilnhitm.ac.in

***Abstract*— This web application helps to identify an attack or sense abnormal behaviour in the network and send an alert to the user and protect the user. When the user login into the portal he gets the information about the network's accuracy, f1-score, precision. This helps the user to detect how safe his network is for the system.**

Network intrusion detection systems (NIMS) play a critical role in safeguarding computer networks against various cyber threats. Traditional rule-based NIMS often struggle to keep pace with the evolving nature of attacks and the increasing complexity of network environments. In recent years, machine learning (ML) techniques have emerged as a promising approach to enhance the effectiveness of intrusion detection by enabling systems to learn and adapt to new threats.

This paper presents an overview of the application of ML techniques in network intrusion detection. We discuss the challenges faced by traditional NIMS and highlight the advantages offered by ML-based approaches, including their ability to detect anomalies, classify network traffic, and adapt to changing attack patterns. We also provide a comprehensive survey of state-of- the-art ML algorithms commonly used in NIMS, such as deep learning, support vector machines, and ensemble methods.

Keywords*— *web application, network intrusion detection, cyber threats, machine learning, anomalies, classification, deep learning.

I. INTRODUCTION

With the pervasive integration of computer networks in modern society, ensuring the security of these networks has become a paramount concern. Network intrusion, wherein unauthorized access or malicious activities occur within a network, poses significant threats to the confidentiality, integrity, and availability of sensitive data and critical systems. In response to these threats, network intrusion detection systems (NIMS) have been developed to monitor network traffic in real- time and identify suspicious or malicious activities.

Traditional NIMS rely on predefined rules or signatures to detect known attack patterns, making them effective at recognizing well-established threats. However, these rule-based systems often struggle to detect novel or sophisticated attacks that do not conform to predefined signatures. Moreover, the increasing volume and complexity of network traffic make manual rule generation and maintenance a daunting task.

In recent years, machine learning (ML) techniques have emerged as a promising approach to augment traditional NIMS by enabling systems to learn and adapt to new attack patterns autonomously. ML-based NIMS can analyze large volumes of network data, identify patterns and anomalies, and make intelligent decisions about potential threats in real-time. By leveraging the power of ML algorithms, NIMS can enhance their detection capabilities, improve accuracy, and reduce false positive rates.

This paper provides an overview of the application of ML techniques in network intrusion detection. We discuss the limitations of traditional rule-based NIMS and highlight the advantages offered by ML-based approaches. Additionally, we review state-of-the-art ML algorithms commonly used in NIMS and explore various data preprocessing techniques and feature selection methods to improve detection performance.

Furthermore, we examine the challenges associated with deploying ML- based NIMS in real-world network environments, including the selection of appropriate datasets,

evaluation metrics, and model interpretability. Through a case study, we demonstrate the practical implementation of an ML-based NIMS and evaluate its performance using real-world network traffic data.

II. PROBLEM STATEMENT

A. Problem Statement

The increasing sophistication and frequency of cyber attacks pose significant challenges to the security of computer networks. Traditional rule-based network intrusion detection systems (NIMS) are often insufficient in effectively identifying and mitigating these evolving threats. These rule-based systems rely on predefined signatures or patterns, making them susceptible to evasion by novel attack techniques that do not match known signatures. Moreover, the manual creation and maintenance of rules become impractical in dynamic network environments with a high volume of traffic.

Machine learning (ML) techniques offer a promising solution to enhance the capabilities of NIMS by enabling systems to learn from data and adapt to new attack patterns autonomously. However, the effectiveness of ML-based NIMS depends on several factors, including the choice of algorithms, the quality of training data, and the robustness of the deployed model.

Therefore, the problem statement revolves around addressing the limitations of traditional rule-based NIMS and harnessing the potential of ML techniques to develop more robust and adaptive intrusion detection systems.

B. Objective

Develop ML Models for Intrusion Detection: Create and train machine learning models capable of effectively detecting various types of network intrusions, including known and novel attacks, using labelled training datasets.

Improve Detection Accuracy: Enhance the accuracy of intrusion detection by leveraging advanced ML algorithms and feature engineering techniques to identify subtle patterns and anomalies in network traffic.

Reduce False Positives: Minimize false alarms and improve the precision of intrusion detection by optimizing ML models to distinguish between legitimate network activities and malicious behaviour.

Enhance Scalability and Efficiency: Design ML-based intrusion detection systems that can efficiently process large volumes of network traffic in real-time, ensuring low latency and minimal impact on network performance.

Enable Adaptability and Self-Learning: Develop NIMS that can adapt to evolving threats and changing network conditions by implementing algorithms for continuous learning and updating of detection mechanisms without manual intervention.

III. REVIEW OF LITERATURE

The literature surrounding network intrusion detection spans various domains, including computer science, cybersecurity, machine learning, and network engineering. Below is a summary of key findings and trends from recent research in this area:

1. Traditional Rule-Based NIMS Limitations Numerous studies have highlighted the limitations of traditional rule-based network intrusion detection systems (NIMS), including their inability to detect novel or polymorphic attacks, high false positive rates, and scalability challenges in dynamic network environments.

2. Machine Learning for Intrusion Monitoring: There is growing interest in applying machine learning (ML) techniques to enhance the capabilities of NIMS. Research has explored the effectiveness of supervised, unsupervised, and semi-supervised learning algorithms in detecting network intrusions, with a focus on improving detection accuracy and reducing false positives.

3. Feature Engineering and Selection Feature engineering plays a crucial role in extracting meaningful information from raw network traffic data. Studies have investigated various feature extraction techniques, including protocol-based features, statistical features, and time-series analysis, to capture relevant patterns indicative of malicious activities.

4. Deep Learning Approaches: Deep learning techniques, particularly convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have shown promise in learning complex patterns from network traffic data. Researchers have explored the application of deep learning architectures for intrusion detection, achieving significant improvements in detection accuracy and scalability.

5. Real-Time Processing and Scalability: Ensuring timely detection and response to security incidents is essential in dynamic network environments. Research has addressed challenges related to real-time processing and scalability of NIMS, proposing distributed architectures, stream processing frameworks, and efficient algorithms for high-speed packet analysis.

6. Evaluation Metrics and Benchmark Datasets: Standardized evaluation metrics and benchmark datasets are crucial for comparing the performance of different intrusion detection techniques. Researchers have proposed metrics such as detection rate, false positive rate, precision, recall, and F1-score, and have developed benchmark datasets such as NSL-KDD, UNSW-NB15, and CICIDS2017 for evaluating NIMS performance.

7. Practical Deployment and Integration: Beyond algorithmic advancements, research has also focused on practical considerations for deploying and integrating ML-based NIMS into operational networks. This includes compatibility with existing network infrastructure, integration with security information and event management (SIEM) systems, and usability for security analysts.

IV. IMPLEMENTATION PLAN

Implementing a network intrusion detection system (NIMS) involves several steps, including selecting appropriate hardware and software components, configuring the system, deploying sensors, and fine-tuning detection algorithms. Here's a general outline of the implementation process for a NIMS:

1. Define Requirements:
- Clearly define the requirements and objectives of the NIMS, considering factors such as the size and complexity of the network, the types of threats to be detected, compliance requirements, and available resources.

2. Select Hardware and Software:
- Choose the appropriate hardware components, including servers, network interface cards (NICs), and storage devices, based on the scalability, performance, and reliability requirements of the NIMS.

- Select NIMS software or platforms that best meet the requirements of the deployment, considering factors such as detection capabilities, ease of use, compatibility with existing infrastructure, and support for customization.

3. Configure NIMS Software:
- Install and configure the selected NIMS software on the designated server or servers.
- Configure the NIMS settings, including network interfaces, monitoring policies, alert thresholds, logging options, and integration with other security tools or systems.

4. Deploy Sensors:
- Deploy NIMS sensors strategically throughout the network to monitor traffic at critical points, such as network borders, internal segments, and key assets.

- Configure the sensors to capture and analyze network traffic according to the defined policies and detection rules.

5. Test and Validation:
- Conduct thorough testing and validation of the NIMS deployment to ensure that it effectively detects and responds to security threats without causing disruptions to normal network operations.
- Perform real-world testing with simulated attacks, test data sets, or controlled experiments to evaluate the NIMS performance under various conditions.

6. Training and Documentation:
- Provide training and documentation for network administrators, security analysts, and other stakeholders involved in operating and maintaining the NIMS.
- Document the NIMS configuration, deployment procedures, troubleshooting guidelines, and best practices to facilitate ongoing management and support.

7. Ongoing Monitoring and Maintenance:
- Monitor the performance and effectiveness of the NIMS on an ongoing basis, including monitoring alert logs, reviewing system health metrics, and analyzing detected incidents.
- Perform regular maintenance tasks, such as software updates, signature updates, rule refinements, and hardware upgrades, to keep the NIMS up-to-date and resilient against emerging threats.

8. Continuous Improvement:
- Continuously evaluate and improve the NIMS based on feedback from security analysts, incident response activities, and changes in the threat landscape.
- Implement lessons learned from security incidents, security advisories, and research findings to enhance the effectiveness and efficiency of the NIMS over time.

V. SYSTEM TESTING

System testing for a network intrusion detection system (NIMS) involves validating the functionality, performance, security, and reliability of the system to ensure that it effectively detects and responds to security threats. Here's an overview of the various types of system testing that can be conducted for a NIMS:

1. Functional Testing:
- Use Case Testing: Verify that the NIMS accurately detects and alerts on various types of network intrusions, including known attacks, unknown attacks, and anomalous behavior, as specified in the use cases.
- Alert Validation: Validate the accuracy and completeness of intrusion alerts generated by the NIMS, ensuring that they contain relevant information such as severity level, timestamp, source/destination IP addresses, and description of the detected intrusion.

2. Performance Testing:
- Throughput Testing: Measure the NIMS's ability to process network traffic at different throughput rates while maintaining acceptable detection accuracy and minimal latency.
- Scalability Testing: Evaluate the NIMS's ability to scale horizontally or vertically to accommodate increasing network traffic volumes, additional sensors, or higher computational loads.

3. Security Testing:
- Penetration Testing: Conduct penetration tests and vulnerability assessments to identify potential weaknesses or security vulnerabilities in the NIMS deployment, such as misconfigurations, access control issues, or software vulnerabilities.
- False Positive/Negative Testing: Assess the NIMS's performance in terms of false positive and false negative rates, identifying scenarios where legitimate network traffic is incorrectly flagged as malicious or malicious activity goes undetected.

4. Reliability Testing:
- Fault Tolerance Testing: Evaluate the NIMS's ability to withstand hardware failures, software errors, or network disruptions without compromising its ability to detect and respond to security threats.
- Redundancy Testing: Test failover mechanisms, redundant components, and high availability configurations to ensure continuous monitoring and protection in the event of system failures.

5. Usability Testing:

- User Interface Testing: Assess the usability and intuitiveness of the NIMS's user interface, ensuring that security analysts can efficiently navigate, configure, and interact with the system.

- Configuration Testing: Validate the ease of configuring and customizing NIMS settings, policies, and detection rules to meet the organization's specific security requirements and operational needs.

6. Integration Testing:

- Compatibility Testing: Verify interoperability with existing network infrastructure components, security appliances, management systems, and third-party security tools, ensuring seamless integration and data exchange.

- SIEM Integration Testing: Test integration with security information and event management (SIEM) systems for centralized log collection, correlation, and analysis of NIMS alerts and security events.

7. Compliance Testing:

- Regulatory Compliance Testing: Validate that the NIMS deployment complies with relevant regulatory requirements, industry standards, and best practices for network security, data protection, and privacy, such as GDPR, HIPAA, or PCI DSS.

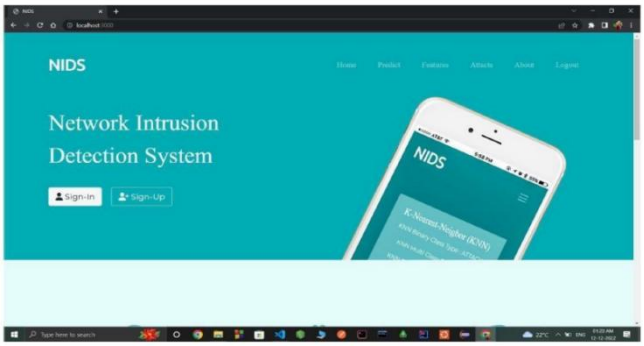
8. Documentation and Training:

- Documentation Review: Review documentation, training materials, and user guides to ensure completeness, accuracy, and relevance to the NIMS deployment, facilitating ongoing management, support, and knowledge transfer.

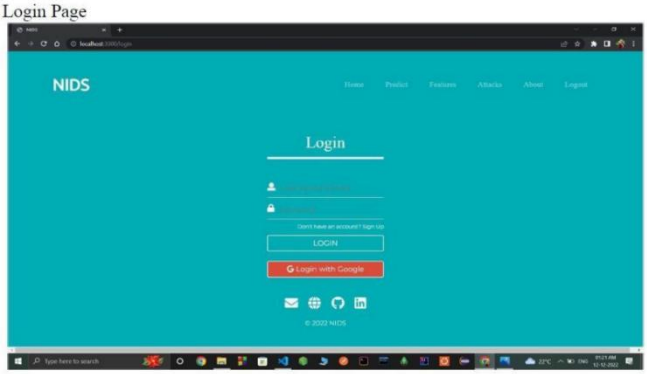
VI. OUTPUT

Mobile Compatible Application Screen

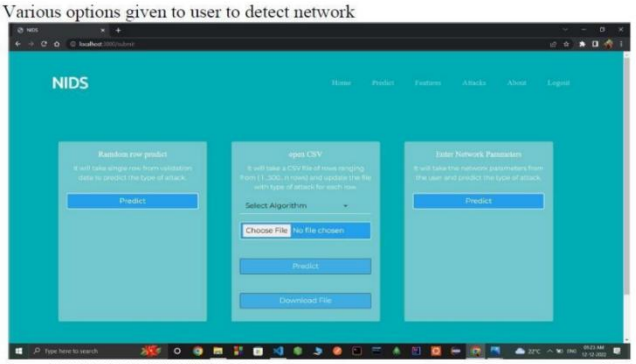
Desktop/Laptop Compatible Application screen:



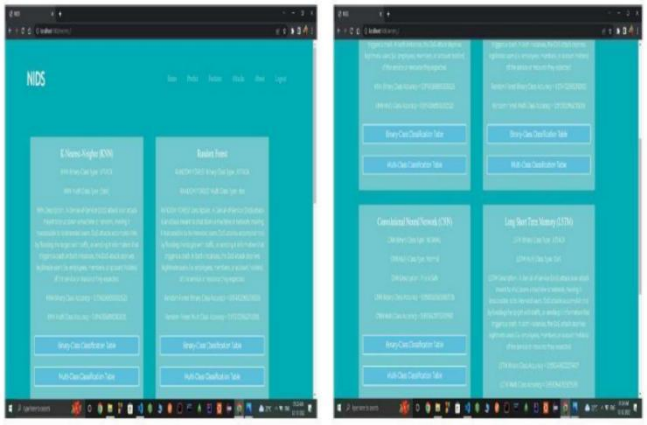
Login Page



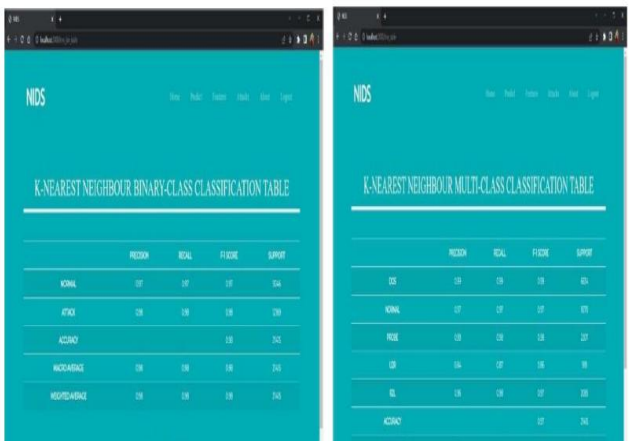
Various options given to user to detect network



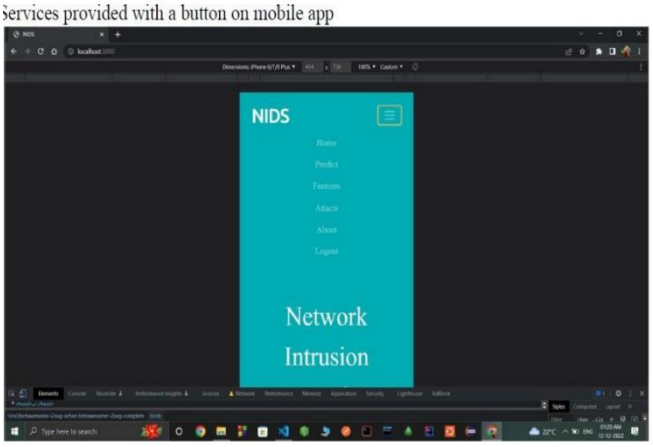
Details about each algorithm and Prediction



Classification table for the trained and testing data:



Services provided with a button on mobile app



VII. SYSTEM REQUIREMENTS

1. Hardware Requirements::

For Development we need a machine of following configuration:

- Processor : Intel core i3 10th Gen
- RAM : 4GB & above
- HDD : 256 GB.
- Systems : Laptop or Desktop
- Storage : Memory Card
- Connectivity : Sim Card, Wi-Fi, Bluetooth

2. Software Requirements:

For Development we need a machine of following configuration:

- Operating System : Windows 8/10/11.
- Programming Language : flutter, dart language
- Development IDE :Visual Studio Code
- Version: 1.75
- Other Software's: Android Studio
- Database : SQLite

ACKNOWLEDGMENT

We are heartily grateful to our project guide Dr. Niranjan Kulkarni (Department of Computer Science and Design) for his being a constant source of encouragement at various stages of the completion of this project. We want to thank our parents and lord, because without their blessing, perhaps we could do nothing.

We wish to thank all my friends, colleagues, students, brother professionals and the campus staff (Department of Computer Science and Design), who have help us with the critical review of this project including my friends. Their Debugging skills and grateful appreciated. Special thanks to my Head of the Computer Science and Design Department Dr. Niranjan Kulkarni for his advice and encouragement and creative ideas for this project. We must be thankful to the various authors who have contributed.

REFERENCES

[1] Network intrusion detection system by applying ensemble model for smart home, 2024, International Journal of Electrical and Computer Engineering.

[2] Zhang, Weibao. (2024). A Survey on Network Security Traffic Analysis and Anomaly Detection Techniques. International Journal of Emerging Technologies and Advanced Applications.

[3] Nakip M, Gelenbe E. Online Self-Supervised Deep Learning for Intrusion Detection Systems. IEEE Transactions on Information Forensics and Security, 2024.

[4] A systematic literature review for network intrusion detection system (IDS), 2023, International Journal of Information Security.

[5] Machine Learning-Based Intrusion Detection System: An Experimental Comparison, 2023, Journal of Computational and Cognitive Engineering.

[6] Nitasha Sahani, Ruoxi Zhu, Jin-Hee Cho, and Chen-Ching Liu. 2023. Machine Learning-based Intrusion Detection for Smart Grid Computing: A Survey. ACM Trans. Cyber-Phys. Syst. 7, 2, Article 11 (April 2023).

[7] Design of Intrusion Detection System based on Cyborg intelligence for security of Cloud Network Traffic of Smart Cities, 2022, Journal of Cloud Computing.

[8] A Study of Network Intrusion Detection Systems Using Artificial Intelligence/Machine Learning, 2022, Applied Sciences (Switzerland).

[9] Cyber Intrusion Detection System Based on a Multiobjective Binary Bat Algorithm for Feature Selection and Enhanced Bat Algorithm for Parameter Optimization in Neural Networks, 2022, IEEE Access.

[10] [X. Y. Li, R. Tang and W. Song, "Intrusion Detection System Using Improved Convolution Neural Network," 2022 11th International Conference of Information and Communication Technology (ICTech)), Wuhan, China, 2022.

IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.