International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 03 | March - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Network Intrusion Detection System Using Machine Learning

Ch. Kodanda Ramu[1], M. Tarun[2], M. Praveen[3], N. Laxman[4], S. Sai Prasad[5]

[1]*Associate Professor, Department of Computer Science & Engineering, Miracle Educational Society Group of Institutions, Bhogapuram, Vizianagaram, Andhra Pradesh, India - 535216*

[2,3,4,5]*B.Tech Student , Department of Computer Science & Engineering, Miracle Educational Society Group of Institutions, Bhogapuram, Vizianagaram, Andhra Pradesh, India - 535216*

Email: kvr.chintu1978@gmail.com

Abstract - This document discusses the creation of an intelligent Network Intrusion Detection System (NIDS) using Machine Learning (ML) for improved security of computer networks. This report presents the concept for using ML techniques, which provide a solution to conventional detection methods that utilize signatures, as traditional signature-based detection methods are often incapable of detecting new or zero-day attacks. Thus, adaptive data driven methods should be employed. The system will accurately monitor and analyze incoming and outgoing traffic on the network so that each piece of traffic can be effectively classified as either normal or malicious; malicious traffic will be classified into one of four attack types, which are: Denial of Service (DoS), Probe, Remote to Local (R2L), and User to Root (U2R). The model will utilize the NSL-KDD benchmark dataset, along with a multitude of data preprocessing steps (i.e., Categorical Encoding, scaling features, etc.) in order to maximize the effectiveness of the model. Additionally, numerous ML classification algorithms (i.e., Random Forest, Decision Tree, Support Vector Machine (SVM) and K-Nearest Neighbor (KNN)) will be analyzed to determine high performance algorithms. Based upon accuracy, precision, recall and F1 score, the Random Forest classification algorithm would be the best performing algorithm. The system includes an email alerting function to notify the network administrator immediately when a critical intrusion is detected, thus enabling the network administrator to respond rapidly to the incident. Overall results indicate that the proposed ML-based NIDS is capable of detecting a significantly higher percentage of intrusions while simultaneously producing a lower percentage of false positives than traditional rule-based methods. The findings indicate that artificial intelligence can play a critical role in the protection of modern network infrastructures against more sophisticated cyber threats.

Key Words: Network Intrusion Detection System, Machine Learning, NSL-KDD Dataset, Random Forest, Cybersecurity

1.INTRODUCTION

Over time, digital connections have greatly expanded and become an increasingly important part of communication, trade, and the way companies do business [1,2]. All businesses are heavily dependent on digital systems that allow for the storage, processing, and transmission of highly sensitive information [3]. As such, this dependency has unfortunately created a vast increase in the number of cyber security threats, and cyber criminals continually develop new and sophisticated ways to circumvent the protections we have placed on our networks [4]. Therefore, we see unauthorized access, theft of data, the introduction of malware, and disruptions in normal service. Protecting the infrastructure of our organizations has gone from being "just" an IT function to being a core business function [5].

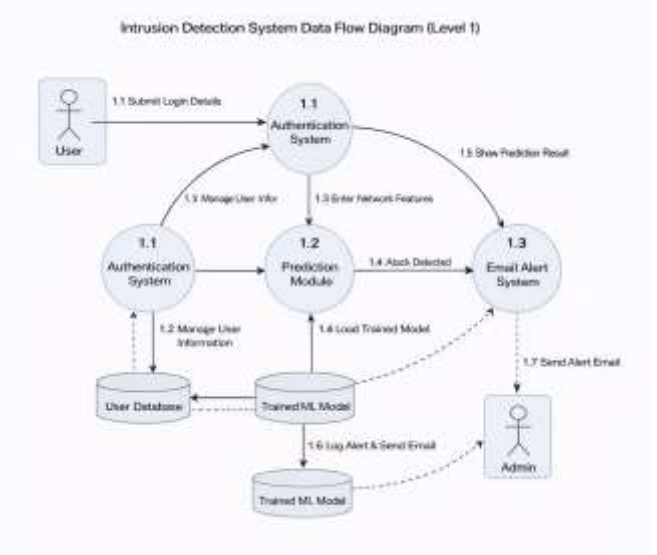

Figure. 1 Network Intrusion Detection Data Flow

One of the primary methods used to protect digital environments from cyber threats is the use of Intrusion Detection Systems (IDS). An IDS is like a watchman on the wall of an ancient city by

continuously monitoring the network traffic of an organization and the activity on its systems to determine if there are any activities that are considered suspicious or which violate the company's security policy. Historically, IDS have been part of an organization's defensive strategy for its network [10-15].

Nonetheless, the ever-changing nature of cyber threats means that we need fast, dynamic, and intelligent solutions to these threats. Traditional systems are usually reliant on using static rules; however, when modern-day attackers target an organization's system, they are typically using dynamic and/or polymorphic attack methods to defeat traditional detection methods. To address this issue, artificial intelligence (AI) and machine learning (ML) have become very effective tools in the cybersecurity toolkit. AI and ML will process and utilize large amounts of network traffic data to automatically learn the complex patterns that represent normal network behavior, which will enable an organization to accurately identify deviations from such behaviors that would indicate that an attack is taking place. This research project looks at developing a robust ML-based Network Intrusion Detection System (NIDS) capable of not only accurately classifying a large variety of network traffic but also having an automated alerting capability that will allow for immediate administrative intervention so that the organization can be proactively protected [15-20].

1.1 Background

Network ecosystems' rapid growth comes with an equally fast-growing rise in cyber threats. To begin, the network security administrator primarily used firewalls and access control lists to filter traffic. As attacks evolved, the introduction of Intrusion Detection Systems (IDS), provided deeper layers of security. Initially, IDS was primarily signature-based and operated similarly to traditional antivirus; that is, by comparing incoming packets to large, dynamic databases of known attack signatures. While these early systems were very effective at blocking known malware and well-documented attack vectors with minimal false positive warnings, they had one major drawback: they were completely unable to detect or block new, undocumented attacks — commonly known as "zero-day exploits." Anomaly-based detection methods were developed to address this problem by establishing a baseline of normal activity on the network and flagging significant deviations from that baseline. However, many early anomaly-based systems suffered from high false alarm rates, generating alarms for benign but very unusual activity on the network. The use of machine learning in network security represents yet another evolutionary stage in that the mathematical and statistical approach to anomaly detection provides greatly increased accuracy and adaptability when analyzing historical trends.

1.2 Problem Statement

Today's networks are experiencing an ever-growing challenge from the ever-increasing number and complexity of cyberattacks. Signature-based intrusion detection systems and static firewall solutions are becoming less effective against polymorphic threats and zero-day vulnerabilities. Legacy security solutions use predetermined rules to detect intrusions and require constant manual updates and have therefore left networks vulnerable to new attack patterns. Additionally, traditional anomaly-based intrusion detection systems produce large volumes of false positive alerts that overwhelm security personnel, causing alert fatigue. Therefore, there is an immediate need for an intelligent, adaptive, and self-learning intrusion detection system that can accurately identify both known and unknown network threats and provide a minimal number of false positives.

2. LITERATURE REVIEW

The field of network intrusion detection has undergone considerable research and development over the years from basic rule-based network monitoring to today's sophisticated predictive models using artificial intelligence (AI) [1]. In the early days of network intrusion detection, researchers were primarily concerned with conceptually describing the task of monitoring the activity of a system on a continuous basis for the purpose of discovering abnormal patterns of behavior [2]. As part of this initial work, the researchers developed the fundamental concept that once a malicious act occurs, even if it has not been observed previously, it will, by definition, deviate from the norm of how the system operates (nominal operating behavior) [3]. They also produced additional concepts within their research. Subsequent researchers, in developing additional concepts, introduced data mining as a way of analyzing large sets of data, specifically as it pertains to the study of cybersecurity, demonstrating that using mathematical algorithms to analyze log files associated with network

traffic could assist in the automated identification of malicious activity [4]. This change in the way intrusion detection occurs has proved that intelligent systems can be utilized to change the position of network security from reactive to proactive [5].

Network Intrusion Detection Systems (NIDS) can be configured in general as two different types: signature-based detection and anomaly-based detection. In the case of signature-based NIDs, the incoming network packets received are compared and matched to a database of known threat definitions whenever there is a match [6]. If there is a match, an alert is generated. Signature-based NIDs are advantageous because of their high level of accuracy in detecting known malware, denial of service (dos) attempts, and a variety of specific injection attacks; thus, they generate relatively low numbers of false positive alerts [7].

On the other hand, intrusion detection systems based on anomalies overcome this limitation by creating a comprehensive baseline of regular activity on a network [8]. By continuously monitoring certain parameters, like traffic volume; how specific protocols are normally used; what time a user might normally log on; etc., they can identify any significant deviations from the established baseline as potential threats [9]. While anomaly detection is able to identify new forms of attacks, comprehensive examinations of the various anomaly detection systems have consistently pointed out that, one of the major shortcomings of those systems is the high number of false positives they produce [10]. Many benign events, like legitimate surges in network traffic caused by legitimate activity, or routine actions taken by administrators after the standard hours, will trigger an incorrect alarm to go off and have led to complacency in security staff from constantly responding to false alarms [11].

In overcoming the limitations of traditional methods, the academic world has embraced the use of Machine Learning (ML) algorithms with increasing interest. ML algorithms have the computing power to process and analyses large amounts of high dimensional data, automatically identifying complex patterns and relationships that define safe and malicious behaviors on a network. The field has seen a large amount of literature focused on using supervised learning as the predominant method for teaching algorithms. Algorithms such as Decision Trees have gained popularity due to their readability and ability to process data quickly [12-16].

Numerous academic papers have examined and used Support Vector Machines as an algorithm for intrusion detection. The principle behind how these algorithms function is that they construct optimal decision boundaries to differentiate amongst the numerous classes of network traffic in high-dimensional space. In multiple studies it is shown that Support Vector Machines have very high accuracy and robustness when it comes to working with network packets, even though extensive computational time is required to train them on large volume datasets. K-Nearest Neighbors method is another classification technique which works by determining the class of an incoming data point according to the relative distance between itself and the nearest n (often referred to as " K ") number of previous data points or packets that have previously been classified as belonging to one of the available classes. Although they work well for simple pattern matching applications, several studies have indicated that the use of K-Nearest Neighbors as an intrusion detection method may not be practical or ideally suited for real-time high-speed network environments due to the excessive latency associated with them during testing [17-19].

A number of research papers have brought up that Random Forests, as an ensemble learning type approach, are considered to be the best currently known method for performing IDM. Random Forests operate by aggregating outputs of numerous decision trees to lessen overfitting and achieve high levels of accuracy. The constant research supports the assertion that ensemble-type methods have a very high degree of robustness to the variability and complexity of network traffic and have excellent balance between high levels of detection and low levels of false positives [20].

2.1. Research Gaps

- Traditional systems rely on static signatures, making them ineffective against zero-day and polymorphic attacks.
- Anomaly-based methods often produce high false positives, causing alert fatigue.
- Existing systems struggle to balance accuracy with computational efficiency for real-time analysis.
- Limited integration of real-time alerting with multi-class attack classification.

2.2. Objectives

- Develop a machine learning–based NIDS to detect malicious network traffic.
- Classify attacks into DoS, Probe, R2L, and U2R categories.
- Compare performance of models like Random Forest, Decision Tree, SVM, and KNN.
- Implement real-time alerting via automated email notifications.

3. METHODOLOGY

There is a systematic methodology used when creating a Network Intrusion Detection System (NIDS), consisting of data collection, extensive data preparation, model training and performance evaluation. The foundational component of this research involves the use of a widely used dataset, namely the NSL-KDD dataset. Specifically designed to remove redundant records that existed within the first NIDS datasets, the NSL-KDD is considered an optimized benchmark for research purposes in the field of Cybersecurity. By eliminating the duplicate records from previous datasets, the NSL-KDD dataset helps to ensure that all models are evaluated with respect to their ability to identify attacks regardless of whether they are represented in the original NIDS datasets.

The NSL-KDD dataset consists of 41 different features that describe network connection attributes and can be broken down into three broader categories: Basic Connection Attributes, Content Based Attributes, and Traffic Based Statistical Attributes. Additionally, the NSL-KDD dataset contains a target variable that classifies network traffic as either ‘Normal’ or one of four categories of attacks: Denial of Service (DoS), Probe, Remote-to-Local (R2L), or User-to-Root (U2R).

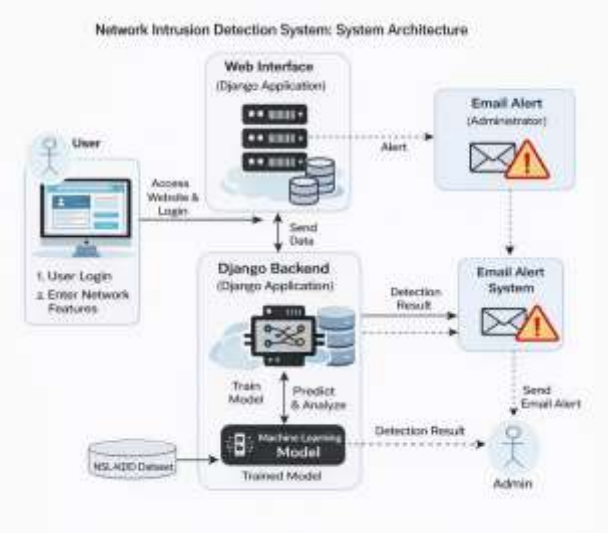

Figure. 2 Network Intrusion Detection System architecture

The preprocessing of data is an important component of this process and is a way of getting the raw data into a format that is very suitable for machine learning algorithms. In order to do this, the first step is to process any categorical variables. The categorical variables include features such as 'protocol type', 'service', and 'flag'; these features are text based, and therefore there is no way for the mathematical algorithms to directly work on the categorical data (the strings). To overcome this limitation, the labels are encoded into a numeric integer using a technique called Label Encoding so that they can now be used for calculations. After the categorical variables have been encoded, the next step in the preprocessing of the data is to apply a Standard Scaler to each of the features. The features of network traffic have a large difference in range and scale; for example, the number of bytes could range from 0 to millions. By standardizing the features to have a mean of 0 and a standard deviation of 1, it prevents features with larger numerical ranges from affecting the gradient descent during machine learning model training.

After the data is cleaned and scaled, it will be divided into training and testing data sets using an 80-20 stratified split to maintain the same distribution of attack classes in each of the sets. The training set is used to teach the algorithms about the patterns that distinguish between normal traffic versus all different types of attack vectors, while the testing set is used strictly for unbiased evaluation.

The methodology focuses on selecting and training multiple supervised machine learning classification algorithms. The selected models for implementation are the Decision Tree, Support Vector

Machine (SVM), K-Nearest Neighbors (KNN), and Random Forest classifiers. Each classifier model is trained on the preprocessed dataset to conduct both binary classification (i.e., whether the traffic is an attack or is normal) as well as multi-class classification (i.e., what type of attack the traffic resembles).

Finally, the methodology employs a robust evaluation framework. The models' predictive ability will be evaluated using standard performance metrics including Accuracy, Precision, Recall, and the F1-score. In addition, confusion matrices are created to provide a visual assessment of the true positive rate, false positive rate, true negative rate, and false negative rate of each classifier across all types of attacks. Lastly, the methodology includes the extraction of feature importance, allowing researchers to determine which particular characteristics of the network have the greatest impact on detecting malicious activity and, therefore, will be significant in determining future feature selections.

4. RESULTS AND DISCUSSIONS

To assess the performance of our proposed Network Intrusion Detection System (NIDS) based on machine learning (ML) techniques, we performed a rigorous evaluation utilising the testing subset of the NSL-KDD dataset. In order to evaluate the model, we calculated standard classification metrics, such as Accuracy, Precision, Recall, and F1-Score, and we carried out further analyses using confusion matrices to illustrate the behaviour of the model across various types of attacks.

In terms of comparison between the three algorithms used for classification, Random Forest demonstrated a significantly higher performance than Decision Tree, Support Vector Machine and K-Nearest Neighbour across all measured performance indicators. For all classification tasks identified within this research, Random Forest achieved a nearly perfect classification accuracy score (greater than 95 per cent) on the binary classification task of identifying normal or attack traffic; therefore, Random Forest provided a robust ability to filter general malicious activity from benign traffic while also achieving a very low false positive rate, thereby reducing the amount of alert fatigue.

In investigating multi-class classification with Random Forest, in which we needed to distinguish between DoS, Probe, R2L and U2R attacks, Random Forest consistently maintained superior performance throughout. Based upon the identified usage of confusion matrices, it was apparent that Random Forest categorised DoS and Probe attacks with an exceptional high degree of confidence. A precise recall and a close to 0.99 precision recall in this system implies that the system can detect malicious high-volume aggressive threats without blocking legitimate user requests.

On the other hand, the experimental results identified issues with very unbalanced datasets. The detection of DoS and Probe attacks was excellent, while detection of Remote-to-Local (R2L) and User-to-Root (U2R) attacks produced an F1-score that was somewhat less. This difference can primarily be attributed to there being very few records of R2L and U2R attacks in the training dataset. Since R2L and U2R attacks are inherently more stealthy than DoS and Probe attacks and occur much less frequently, the mathematical model did not have many examples from which to learn. However, despite the problem of imbalanced data, the ensemble nature of the Random Forest algorithm produced the highest level of detection accuracy possible for R2L and U2R attacks when comparing Random Forest to other baseline models.

Additionally, the implementation of the automated alert system produced a very successful set of operational results. The automated alert system is functionally able to generate an email immediately whenever the predictive module flags an incoming data packet as an attack in the live environment via the web graphical user interface (GUI).

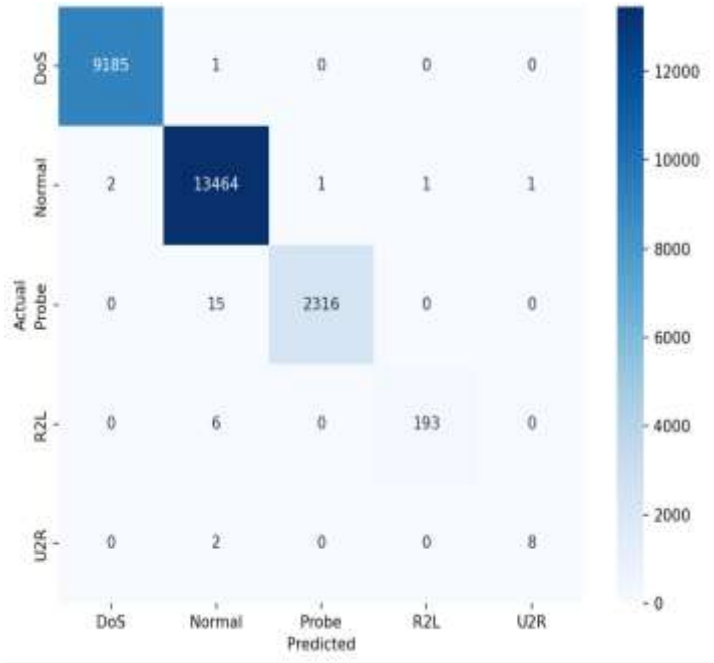

Figure. 3 Confusion of Random Forest

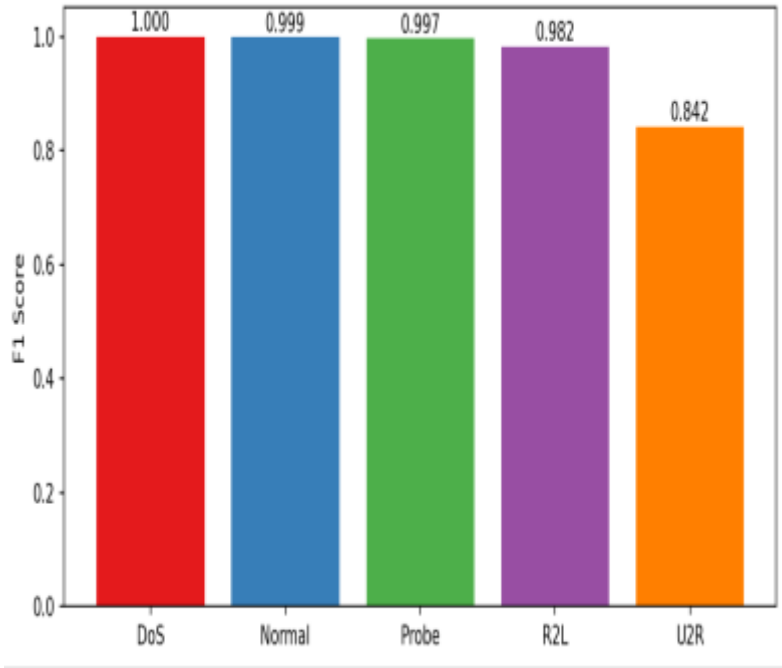

Figure. 4 Per Class F1- Random Forest

Figure. 5 Application User Interface

5.CONCLUSIONS

In conclusion, we have successfully created and implemented a highly efficient Network Intrusion Detection System (NIDS) using advanced machine-learning methods as a result of this research. The NSL-KDD Dataset was used for our work, showing the superiority of data-driven methods over traditional signature-based approaches. The dataset was subjected to extensive preprocessing and feature scaling to prepare it for in-depth mathematical analysis.

After testing the various algorithms, Random Forest was the best-performing classifier because it had high accuracy and precision and provided strong F1 Scores for both binary and multi-class threat classifications. The NIDS was able to detect various types of complex cyber threats such as denial of service (DoS), probing, remote-to-local (R2L), and user-to-root (U2R) attacks while also greatly reducing the number of false positives produced by the system.

As a result of implementing an automated email alerting system within our NIDS, it has turned our analytical model into a proactive defensive tool. This means that network administrators can receive real-time notifications of high-priority security events.

Our research demonstrates how artificial intelligence plays an important role in extensive cybersecurity solutions. We also provide a scalable, intelligent, and reliable framework that can continuously adapt to the changing landscape of digital threats and protect sensitive network infrastructures from them.

REFERENCES

[1]. W. Stallings, Network Security Essentials: Applications and Standards, 6th Edition, Pearson Education, 2017.

[2]. K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," National Institute of Standards and Technology (NIST), Special Publication 800-94, 2007.

[3]. M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A Detailed Analysis of the KDD Cup 99 Dataset," Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009.

[4]. I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016.

[5]. T. M. Mitchell, Machine Learning, McGraw-Hill Education, 1997.

[6]. J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, 3rd Edition, Morgan Kaufmann, 2011.

[7]. S. Axelsson, "Intrusion Detection Systems: A Survey and Taxonomy," Technical Report, Chalmers University of Technology, Sweden, 2000.

[8]. D. E. Denning, "An Intrusion Detection Model," IEEE Transactions on Software Engineering, Vol. 13, No. 2, pp. 222–232, 1987.

[9]. N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," Military Communications and Information Systems Conference, 2015.

[10]. Python Software Foundation, Python Documentation, Available at: https://www.python.org

[11]. T. Fawcett, "An Introduction to ROC Analysis," Pattern Recognition Letters, Vol. 27, No. 8, pp. 861–874, 2006.

[12]. I. H. Witten, E. Frank, and M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, 3rd Edition, Morgan Kaufmann, 2011.

[13]. C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[14]. I. Sommerville, Software Engineering, 10th Edition, Pearson Education, 2015.

[15]. A. S. Tanenbaum and D. J. Wetherall, Computer Networks, 5th Edition, Pearson Education, 2011.

[16]. OWASP, "OWASP Top 10 – The Ten Most Critical Web Application Security Risks," Available at: https://owasp.org

[17]. S. B. Kotsiantis, "Supervised Machine Learning: A Review of Classification Techniques," Informatica, Vol. 31, pp. 249–268, 2007.

[18]. Scikit-learn, Machine Learning in Python – Scikit-learn Documentation, Available at: https://scikit-learn.org

[19]. TensorFlow, TensorFlow Machine Learning Framework Documentation, Available at: https://www.tensorflow.org

[20]. Kaggle, Network Intrusion Detection Datasets, Available at: https://www.kaggle.com