Commercial Network Intrusion Detection System using Machine Learning

Dhaval Tanna
SCAI
Ira A. Fulton Schools of Engineering
Tempe, US
dtanna2@asu.edu

Rutul Patel
SCAI
Ira A. Fulton Schools of Engineering
Tempe, US
rpate196@asu.edu

Anirudh Rajhgopalann Srinivaasrajagopal
SCAI
Ira A. Fulton Schools of Engineering
Tempe, US
asrin101@asu.edu

Freya Shah
SCAI
Ira A. Fulton Schools of Engineering
Tempe, US
fshah14@asu.edu

Ishaq Ahmed Shaik
SCAI
Ira A. Fulton Schools of Engineering
Tempe, US
ishaik4@asu.edu

Jack Curcio
SCAI
Ira A. Fulton Schools of Engineering
Tempe, US
jcurcio3@asu.edu

Ritik Patel
SCAI
Ira A. Fulton Schools of Engineering
Tempe, US
ritikpatel@asu.edu

***Abstract*—The increasing and frequent occurrence of cyber threats challenges the effectiveness of traditional network security measures, especially at the network layer that controls packet routing and messages, limiting current signature and rule-based measures and enables networks to advance zero day attacks and constantly face threats. This study tests the potential of artificial intelligence models to enhance network layer security by continuously detecting abnormal and malicious behavior in real time. Although AI promises to improve detection rates, however there are still important obstacles, such as ensuring accuracy, reducing false positives, maintaining computer performance and protecting against adversary attacks through network intrusion. This research addresses a significant gap in Detection Systems (NIDS), which typically use binary classifiers and rely on their ability to report accurate response behaviors. By training and testing various machine learning and deep learning models to classify attack types, this research seeks to identify the most effective models for threat detection and response. The findings highlight both the strengths and limitations of the current model and contribute to the development of scalable, AI-driven intrusion detection and response systems designed to modernize the complex threat environment.**

***Index Terms*—Cyber threats, Network security, Network layer, Packet routing, Signature-based measures, Rule-based measures, Zero-day attacks, Artificial intelligence (AI), Anomaly detection, Real-time detection, Accuracy, False positives, Performance, Adversarial attacks, Network Intrusion Detection Systems (NIDS), Binary classifiers, Machine learning, Deep learning models, Attack classification, Threat detection, Response systems, Scalable solutions, Complex threat environment, Intrusion Response System (IRS), Intrusion Prevention System (IPS), Deep Belief Networks (DBN)**

I. Introduction

Cyber threats are constantly increasing in sophistication and frequency, and current network security measures often fall behind the newest advancements developed by intruders. With the current landscape of commercial internet systems, the network layer, which is responsible for packet transmission and routing, is a critical point for threat detection. Conventional signature-based and rule-based systems are limited in their ability to identify a variety of threats, which leaves networks exposed to zero-day exploits, advanced persistent threats and malware.

As a result, artificial intelligence models are a promising solution to increase the detection of threats through monitoring and analyzing large amounts of network data in real-time. AI models can find anomalies and malicious behaviors that go undetected by traditional systems. However, the deployment of models for network layer security also needs to consider model accuracy, computation cost, efficiency in real-time, reducing false positives and addressing adversarial attacks on the AI models themselves.

Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) are the foundations to a comprehensive network security. IDS monitors and alerts the administrators of malicious activities without intervening whereas IPS takes a proactive approach by automatically blocking or modifying harmful traffic in real time. Originally, IPS evolved from IDS to address the growing need for real-time responsive system to increasingly sophisticated network threats. Both systems rely on methods like signature-based and anomaly-

based detection, with IDS identifying deviations in the network traffic and IPS taking immediate action. Artificial intelligence offers significant advantages by enhancing IDS and IPS capabilities with adaptive and real-time analysis that can identify and respond to novel and evolving threats. By using AI-driven intrusion detection and prevention thus promises a more resilient approach to protecting network layers across commercial systems by responding to threats at both detection and prevention levels.

The research explores the potential effectiveness and challenges of intrusion detection using artificial intelligence on the network layer of commercial systems. The research dives into the ways models might improve detection and prevention of known and unknown threats. As well, the difficulties and realities of effectively training a model to detect anomalies in network traffic, including cost, time and other limitations, are considered. Through this research, the key strategies for using AI models to enhance network security are identified, with a focus on their potential applications and benefits. While deployment practices and associated challenges are acknowledged, they fall outside the scope of this study.

However, designing an effective Intrusion Response System (IRS) requires more than just identifying the occurrence of an attack; it must also accurately classify the specific type of attack to guide appropriate countermeasures. The primary goal of an IRS is not only to prevent ongoing attacks but also to collect forensic evidence, enabling informed decisions about mitigation strategies. These decisions might include isolating compromised systems, backing up critical data or implementing recovery mechanisms to minimize damage.

Most current research in the field of Network Intrusion Detection Systems (NIDS) primarily focuses on binary classification, wherein network traffic is categorized as either normal or malicious. While this approach can provide a broad level of protection, it falls short of identifying the exact nature of the intrusion, which is essential for crafting a precise response. Attackers employ various sophisticated techniques, and an IRS must be able to differentiate between them to deploy effective countermeasures. Failure to do so can lead to inadequate responses, leaving systems vulnerable to further attacks or data loss.

This research aims to address this gap by training and evaluating existing machine learning (ML), deep learning (DL) algorithms and neural networks to classify different types of attacks. By doing so, we seek to determine which models offer the highest accuracy in detecting and classifying various intrusion types. Understanding the performance of existing models will shed light on their robustness and the potential need for novel methodologies that can improve their compatibility with complex datasets. The findings from this study could provide a new direction for research focused on developing comprehensive intrusion detection and response systems capable of handling diverse and evolving threats.

II. Literature Review

IDS/IPS is of great significance in the context of network security and thus finds massive applications over years. Basically, it has evolved with the ever-increasing complication of computer threats. The domain began to develop in the 1980s as a result of the work by John Anderson and the seminal 1987 paper by Dorothy Denning [5]. Her paper introduced the concept of monitoring system behaviors and network traffic for something abnormal or hostile. While those early inventions laid the foundation for many of today's security technologies, IDS/IPS technologies continue to evolve to address the complex threat landscape with DDoS attacks and APTs.

As per Scarfone & Mell [6] IDS systems have been categorized based on how they detect the threats. Signature-based systems use predefined patterns to find the known threats, which has limited capability to identify new attacks or changed ones. On the other hand, anomaly-based systems show deviation from normal behavior and in general, these lead to a higher number of false alarms. Stateful protocol analysis goes one step further, examining the network, transport and application layers for deviation from normal protocol behavior. Network-based IDS watches at the whole parts of a network, while host-based IDS takes a look at single devices.

The main difference between IDS and IPS is what they do. IDS is passive, which means it warns administrators about suspicious activity. IPS is active, automatically stopping or changing harmful traffic right away. This ability to prevent problems is very important for stopping quick threats like malware and DDoS attacks. IPS, which comes from IDS, is more useful in changing environments where quick responses are very important. Yet both are limited, as signature-based IDS performs hardly against new threats and anomaly-based systems too often provide false alerts.

A. Machine Learning in Software Defined Network

With the rapid growth of network-enabled devices coupled with the rise of malicious activities, there has been an ever-increasing call for network security with advanced features. The incorporation of SDN and machine learning into network security provides a flexible and scalable solution to the emerging concern. As Nguyen(2021) [7] shared, SDN segregates the control plane from the data plane; hence, centralized management is possible, along with fast configuration changes. This flexibility is a huge improvement compared to traditional networks, which have to be manually configured on each router and switch, thus making their response for dynamic environments slower [7].

Besides simplifying network management, SDN provides considerable value related to security, by supporting various kinds of policies such as access control, intrusion detection, virus scanning, among others. For example, one such system called SANE enforces simple access control policies while it conceals topology and service information from unauthorized users [8]. SDN also easily integrates with virtual and cloud environments in order to enable dynamic policy enforcement

that can be very useful for security functions such as firewalls and elastic IP services[9].

Machine learning plays the most critical role in detecting and mitigating anomalies within SDN-based security solutions. The traditional approaches toward the problem of traffic classification are based on high-overhead processes, such as the preprocessing of traffic logs and packet captures. In contrast, SDN allows for more efficient data collection in real time using the built-in counters. Integration of SDN with ML gives far quicker responsiveness to the systems, which would be able to detect malicious flows and mitigate threats like Denial-of-Service attacks. Braga et al.[10] proved that the lightweight ML-based detection systems in SDN efficiently identify malicious traffic to reduce the collateral damage by maintaining the legitimate traffic [10][11].

However, in the implementation of ML in SDN security systems, there are a number of challenges. According to Sultana et al. (2018) [12], this includes problems such as the lack of organic and real-world training data that bridges the semantic gap from an academic proposal to a practical deployed system. The ML model also does not adapt to the variability and complexity found in real-world network data, leading to false positives. One of the open challenges in security systems is appropriate evaluation of ML models due to the lack of a uniform set of metrics for evaluation [12][13].

Another issue that complicates ML-based security model deployment is adversarial attacks. An act of attack was outlined by Tramer et al. (2016) [14] in equation-solving, model inversion, and path-finding attacks. All these possible reasons for threats take advantage of weaknesses present in the ML algorithms. Such may result in a big security breach, either in the manipulation of the decision-making process of the model or in extracting sensitive data from the very model itself [14].

Nguyen suggests several measures that would help make ML-based SDN systems more secure and robust by designing audit-able models, following security in the development, and creating cost operational models that could evaluate the trade-off between security and resource expenditure [7][15][16].

B. Supervised Machine Learning

Machine learning has been widely explored as a solution to the drawbacks of signature-based detection. The study by Osorio et.al(2021) in [1] approaches this goal on the dually classified dataset CICIDS2017 [2] with a novel approach by training GMM(Gaussian Mixture Model) and UBM(Universal Background Model) that are traditionally used in speaker detection and verification. These are probabilistic models that use a mixture of Gaussian distributions in multiple dimensions. GMM uses an Expectation Maximization Algorithm to maximize the probability(Step 2) of a set of randomly selected parameters(step 1) whereas UBM uses Maximum A Posteriori function to estimate a set of parameters for the Gaussian distribution.

[1] compares the results of GMM and UBM against the results of the traditional Random Forest algorithm. The results show that the training and prediction time was the maximum for UBM, then GMM and the quickest with Random Forest. This relation is inversely proportional to the accuracy of the models.The accuracy of GMM and UBM models initially increased with an increase in the number of Gaussian distributions but saw either a stability or reduction in accuracy after a threshold of Gaussians. The caveat in this study was that the dataset was dually classified as either BENIGN or DoS/DDoS. Osorio et. al (2021) states that "...the GMM and UBM techniques worked very well in multiclass classification or multiclass verification problems … ". The findings from this research inspire further exploration of the idea to use this novel technique to train the GMM and UBM models on a multi-classified dataset for a better prediction of the intrusion attack vector.

Al-Yaseen et al.[24] (2021) proposed a hybrid model combining Support Vector Machine (SVM) and Extreme Learning Machine (ELM). Their solution applied multi-level classification to segment attacks into various categories, achieving 95.75% accuracy. The hybrid approach reduced false positives compared to single-SVM models, illustrating the benefit of combining different algorithms for improved detection accuracy.

C. Unsupervised Machine Learning

Most Intrusion-detection systems share a common challenge, which is particularly struggling with high false-positive rates while missing actual threats. This creates a substantial operational overhead for security teams, because these systems commonly trigger thousands of erroneous alerts within a single day. Such a high volume of false alarms can overwhelm network administrators and potentially mask genuine security threats, making the development of accurate detection systems crucial. The authors of [3] proposed an improved IDS model using unsupervised machine learning, specifically, k-means clustering. This approach aims to enhance detection efficiency while minimizing both false positives and false negatives.

The results of this study imply that varying the number of clusters in the system yields different metric scores. Furthermore, when the number of clusters assigned to the model is greater than the number of data types, the detection, false negative and efficiency rates are reduced. This indicates that efficiency can be improved by fine-tuning the number of clusters. The research findings also provide scope for exploring other algorithms, such as Bayesian or Hierarchical Clustering, and comparing their results. Furthermore, a study using K-means followed by a signature detection-based strategy was proposed to further reduce the false negative rate.

D. Neural Network Approach

While machine learning offers effective solutions for predicting network intrusions with considerable accuracy, exploring neural networks may provide further improvements in precision. Lirim et al.[20] (2021) developed a Convolutional Neural Network (CNN) model integrated with a multi-layer perceptron to detect network intrusions on the UNSW-NB15

dataset. Their model leveraged hyperparameter tuning to improve performance, achieving 94.4% accuracy. However, the class imbalance within the dataset required bootstrapping to avoid skewed predictions. This study highlights the potential of CNNs to detect various attacks but underscores the challenge of rare-class detection.

Lin et al.[21] (2021) proposed a hybrid CNN-based system with offline training and online detection. They tested it on the CICIDS2017 dataset and reported an accuracy of 99.56% on raw traffic, demonstrating the importance of real-time traffic data. The results were less accurate when using extracted feature sets, indicating that raw data may yield better predictions. This study emphasizes the need for feature-rich input to maximize the effectiveness of IDS models.

Rohit et al.[22] (2021) employed an ensemble model using Naïve Bayes, PART, and Adaptive Boost on the KDDCup99 dataset. They achieved an impressive 99.97% accuracy by applying feature selection through information gain. The ensemble approach reduced both false positives and false negatives, underscoring the value of combining multiple algorithms for better predictive performance.

Yu et al.[23] (2021) introduced Few-Shot Learning (FSL), a technique for training models on limited data. Using CNN and DNN for feature extraction, they tested the model on NSL-KDD and UNSW-NB15 datasets, achieving accuracies of 92.34% and 92%, respectively. Their work demonstrates that FSL models are well-suited for scenarios with small datasets, offering a potential solution to resource-constrained training environments.

E. Deep learning approach

Shone et al. [6] (2018) suggested a new deep learning-based Network Intrusion Detection Systems using a Non-symmetric Deep Auto-Encoder (NDAE) along with a Random Forest (RF) classifier. Unlike traditional auto-encoders, only the proposed NDAE encoder phase was used in this work that provides non symmetric data dimensional reduction to decrease computational overhead. It consists of two stacked NDAEs with three hidden layers each, equal to the number of input features, using the sigmoid activation function. Then, the obtained encoded features are classified using the RF algorithm. The proposed approach was implemented using TensorFlow with GPU support and was tested on the KDD Cup '99 dataset. They said that the model accomplished an average detection accuracy of 97.85% with a significant reduction in training time, up to 97.72%, when compared to the benchmarks set by DBNs.

Shone et al. [6] (2018), extended their model proposed for the NSL-KDD dataset to overcome issues related to the diversity in modern network traffic. The authors extended their evaluation to a more fine-grained 13-class classification task. The model gave an average accuracy of 89.22% through proving to be effective for larger and more complex datasets. They indicated that their model is increased in terms of granularity, and with which, the performance tends to increase; this is because the NDAE can learn more detailed representations of features. Training time was also reduced by an average of 78.19% compared to DBNs, showing efficiency for real-time intrusion detection.

Shone et al. [6] (2018) also compared their stacked NDAE model with other deep learning-based NIDS approaches. Their model outperformed other deep neural network approaches on the NSL-KDD and KDD Cup '99 dataset in terms of accuracy, but the most shocking result was their decreased training time by as much as 98.81% in comparison with DBNs. This model responded well to the demand of efficiency required for NIDS. All these results indicate that their proposed model performed better regarding accuracy and computational efficiency, hence promising for modern network security challenges.

The authors of the paper "Collaborative Feature Maps of Networks and Hosts for AI-driven Intrusion Detection" [4] set forth a new vision for Intrusion Detection Systems by using collaborative feature maps, where they gathered information from both network and host activities. This has been one of the main deficiencies of the traditional detection systems, when either only network traffic is analyzed or, vice-versa, where reliance is put on monitoring host-based activities. One such example is in the paper [18] where the authors Iman et al. extracted the network packet features but didn't analyze host data. This merge of the two types of data greatly provides a full dataset, hence extending intrusion detection in a multi-layered space for such intrusions.

The authors [4] use deep learning methodologies while analyzing collaborative feature maps, enabling the methodology adopted by them to detect complex attack patterns very easily, which might have been difficult with less-advanced model detection. The need for a new combined dataset with a great scope of high efficiency intrusion detection was called out for when many previous work like KDD98/99 [17] and NSL-KDD [17] failed to provide up to date data. C. Beazley et al. performed exploratory analysis on unified host and network dataset, but did not analyze it.

The core innovation of the paper [4] lies in how these collaborative feature maps are processed. Initially, the authors established baseline detection capabilities using traditional machine learning models, which helped them gauge the performance of these models on existing datasets. This baseline detection provided a reference point for comparison when deep learning techniques were later applied. Deep learning inherently handles complex data structures and thus is suitable for detecting multi-step attacks incurred through network anomalies, suspicious host behavior or both. A network may not recognize abnormal traffic if it is interpreted together with unauthorized attempts to access the host system, where both occurrences together hint at an ongoing attack.

This is unlike many of the available IDS methods in literature. It proposes a uniform insight into the detection of threats, significantly improving real-time detection capability. Integrated network and host activities are bound to reduce any chances of missing critical associations which shall improve the accuracy of the threats detected. They stipulate that their deep learning-based methodology comes on top of detection

rate and hence provides much more realistic detection both for previously known and new attacks. Although several other works have been conducted on deep learning applied to intrusion detection, the novelty of the work presented here is in the first use of different data sources put together and in the use of collaborative feature maps. Some other works, as seen in [19] created a Unified Host and Network Dataset with real-world network distributions. The main issue is that it is very difficult to perform a proper analysis on that because it lacks the labels. Thus, this in-depth analysis not only looks at the constituent parts of the attacks but also the interdependencies between these components, making this study broader than others, which focus just on some attack vectors. This paper reviews a new approach to data analysis in the field of artificial intelligence and provides an overview of the directions of prospective research concerning advanced methods of cyber defense.

They have also discussed the overall increasing importance of deep learning in developing better IDS. The ground they created with collaborative feature maps [4] gives rise to advanced systems that can detect an attack effectively and prevent cyber threats in modern times. Moreover, the effectiveness of using the best in class Machine Learning models, like Random Forest, and XGBoost comes out to be the top pick if training a model on only the traditional datasets.

In conclusion, the research reviewed underscore the dynamic progression of machine learning-based intrusion detection systems (IDS). While advancements such as CNNs and ensemble learning have improved detection rates, challenges like class imbalance and real-time processing remain critical issues. The findings from this study inspire further exploration into training machine learning and deep learning models on multi-class datasets to better predict diverse intrusion attack vectors. This research will focus on hybrid models and diverse datasets to tackle these challenges, aiming to enhance detection capabilities across various attack types. The integration of Few-Shot Learning and advanced deep learning techniques will likely play a key role in the development of adaptive, real-time IDS. Additionally, as technologies like cloud computing and wireless networks expand, scalability and real-time security solutions will become even more crucial, with machine learning and artificial intelligence poised to further improve accuracy and reduce false positives.

III. Methodology

A majority of the research in the literature reviews work with dually classified datasets. The primary goal of this research is to keep the NSL-KDD and CIC-IDS2017 datasets multi-classified and analyze the effectiveness of common machine learning models when predicting a specific attack type. The benefit of correctly predicting an attack type saves network administrators time by enabling them to know what they are up against, opposed to the broad knowledge that there is an intrusion attempt. This research processes the datasets into the same binary classification used in prior research, but also keeps the datasets multi-classified. The binary classified

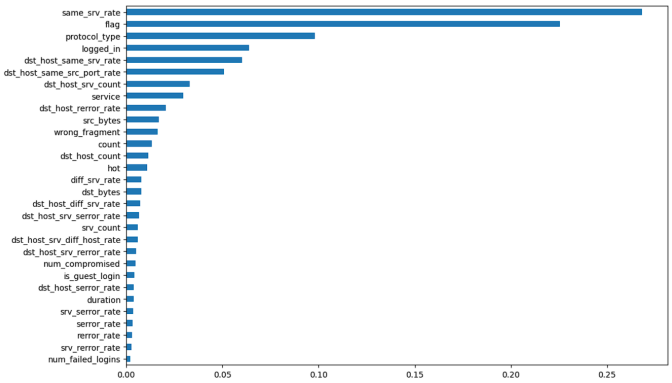

Fig. 1: Feature Importances using ExtraTreesRegressor

datasets separate the classes being predicted into network traffic labeled as either normal traffic or intrusion attempt traffic. In the multi-classified dataset there is still a class being predicted to represent normal network traffic, but there are now unique predictor classes for each type of intrusion attempt.

A. Implementation based on NSL-KDD

This first implementation phase included the use of NSL-KDD [25]. The dataset contains various attack types, and each record in the dataset is represented by 41 features that describe various network aspects. Considering all of the attack instances and non-attack instances, the dataset has a fairly normal distribution, making it a very good quality of dataset for binary classification. Moreover, NSL-KDD is an improved version of the previous KDD Cup 1999 dataset [17], with reduced redundancy and improvement in terms of a more balanced dataset.

The foremost step in this implementation's methodology was some general preprocessing. Standard data cleaning and validation was performed, including checking for null or duplicate records and clearing them. Some of the features specifically- protocol type, service, flag and labels were of object data type. Hence, label encoding was performed on those. We have implemented both- Binary classification and Multi-class classification on this dataset. To do that, separate copies of the original dataset were used and manipulated as per the need. The Machine Learning models that were implemented are- Gradient Boosting Classifier, Support Vector Classifier, Decision Tree Classifier, Random Forest Classifier, and K-Nearest Neighbors Classifier. (and ADABoost, XGBoost)

1) Binary Classification: For the binary classification, the attacks were merged so that the final data only had normal and intrusion classes, or in other words, every other class except 'normal' was converted to 'attack' or 'intrusion'. The dataset was then split into train and test data, and then it was passed onto StandardScaler for feature standardization. Since the NLS-KDD dataset has over one hundred thousand entries, each with 41 features, it was safe to pick top features that are more relevant in order to reduce training time. Figure (1) shows the 30 most relevant features that were derived

using ExtraTreesRegressor and only those features were used in training and testing. Those features were kept on the X and the target variable was kept on the Y, for training purposes. We also kept all 41 of the features on a copy of the dataset and performed the rest of the steps in a similar manner to see what impact they have on the prediction.

Also, we tried experimenting with different combinations of preprocessing steps, as well as various implementation techniques. Firstly, for binary classification, the dataset was split into 80% and 20% for the training and testing sets respectively using the same random seed. As a result, the models for both binary classification and multi-class classification are trained and tested on the same dataset entries, with the only difference being the class being predicted. The same tests were also performed using a split of 60% and 40% for the training and testing respectively.

2) Multi-class Classification: For multiclass classification, at first every attack type in the target variable was kept intact in the data. However, some of the labels were so low in quantity that it was disrupting the whole training process. Hence, all the labels present below a certain threshold in the dataset were removed. Since this was performed on a different copy of the original dataset, label encoding was performed again, followed by scaling. We also implemented the same two variants of train test split as mentioned previously. After the train test split, the data was fed to the ML models. Several types of machine learning models are used to analyze their effectiveness on the multi-class datasets. We also organized attack types into broad categories as an experiment to make the multi-class classification problem much simpler and more interpretable and see how this affects the results. Besides this, some other dimensionality reductions were done using Recursive Feature Elimination on a copy of data, which helped improve the performance and efficiency of the models.

B. Implementation based on CIC-IDS

The dataset CIC-IDS2017 comprises of multiple CSV files corresponding to each day of the week, with a particular emphasis on the data collected on Wednesday. The dataset encompasses a total of 692,703 rows and 79 columns, including the target variable, which consists of 6 distinct classes. This comprehensive dataset provides a robust foundation for both binary and multi-class classification tasks

1) Data Pre-processing Steps: To get the CIC–IDS2017 dataset ready for training the model, a few crucial steps in preprocessing were involved. First up, we needed to convert the target variable which started as categorical labels into numerical form that would work with machine learning algorithms. This was done through label encoding. Then we ran into some extremely high values in certain features—some even close to infinity. To keep these outliers from throwing off the dataset, we clipped them to a set threshold which helped keep things stable. When it came to handling missing data, we decided not to remove the null values. Instead, we filled them with zeros. This way, even missing values could still provide signals in the dataset which might reveal important patterns for the model. Finally, with around 600,000 entries in total, we split the data into training, cross-validation and test sets in a 60-20-20 ratio. This split gave us a solid chunk of data for training while leaving enough for cross-validation and testing to evaluate the model's accuracy effectively.

2) Existing Research: Previous research and work on this dataset has implemented the Random Forest model in a binary classification context achieving an accuracy of 96.7% [26]. When we used the Random Forest model to train a multi-class classification framework on the CIC-IDS2017 dataset, the model achieved an accuracy of 99% which is in contrast, better than the accuracy that was achieved for binary classified data using the same model. This improvement underscores the model's enhanced capabilities to observe patterns among various vulnerabilities.

3) Models Trained: We developed a multi-class classification model by deploying the Decision Tree, Random Forest and XGBoost algorithms. These models were compared against a random model to evaluate their performance. Considering a random model as a benchmark is essential in the model evaluation process as it provides us with a baseline for the performance comparison as it quantifies on the fact that how much better the trained models are in comparison to some random predictions ensuring that they capture meaningful patterns in the data. This approach helps identify overfitting issues and hence guide the model selection by highlighting which models consistently outperform the random benchmark. It also enhances the way we interpret data by offering a clear standard for assessing the effectiveness of our models that allows users to understand the model's capabilities and limitations in a more informed manner.

4) Model Training Information: All models mentioned here were trained with a variety of parameters, including different criteria (Gini and Entropy), maximum depths, maximum features and the total number of estimators. The highest accuracy that we achieved in the multi-class setting was above 99%. Hyperparameter tuning was specifically conducted on the max_depth parameter to optimize the model performance.

IV. Results

A. Dataset: NSL-KDD

1) AdaBoost Classifier: In AdaBoost, there lies a wide difference between binary and multi-class classifications. For binary classification, the model showed 78.25% (refer fig. 2a), which tells about its capability to do well on normal and attack traffic. However, at the same time, there is a huge reduction in the multi-class classification accuracy (refer fig. 2b) to 62.26%, which signifies increased difficulty when classifying among many types of attacks. This performance drop can be attributed to the high similarity between features of various attack types, making it difficult for AdaBoost to accurately classify them. While it did reasonably well on binary classification problems, here AdaBoost could not generalize so well once feature overlaps and higher complexity of decision boundaries came into play with multi-classes.

AdaBoost Classification Report:

	precision	recall	f1-score	support
normal	0.67	0.97	0.79	9711
attack	0.97	0.64	0.77	12833
accuracy			0.78	22544
macro avg	0.82	0.81	0.78	22544
weighted avg	0.84	0.78	0.78	22544

(a) Accuracy results for AdaBoost Binary Classification

warezmaster	0.00	0.00	0.00	944
worm	0.00	0.00	0.00	2
xlock	0.00	0.00	0.00	9
xsnoop	0.00	0.00	0.00	4
xterm	0.00	0.00	0.00	13
accuracy			0.62	22544
macro avg	0.03	0.05	0.04	22544
weighted avg	0.40	0.62	0.49	22544

AdaBoost Accuracy: 62.26%

(b) Accuracy results for AdaBoost Multiclass Classification

Fig. 2: Accuracies for AdaBoost Classifier (NSL-KDD)

XGBoost Classification Report:

	precision	recall	f1-score	support
normal	0.69	0.97	0.81	9711
attack	0.97	0.67	0.79	12833
accuracy			0.80	22544
macro avg	0.83	0.82	0.80	22544
weighted avg	0.85	0.80	0.80	22544

(a) Accuracy results for XGBoost Binary Classification

warezmaster	0.00	0.00	0.00	944
worm	0.00	0.00	0.00	2
xlock	0.00	0.00	0.00	9
xsnoop	0.00	0.00	0.00	4
xterm	0.00	0.00	0.00	13
accuracy			0.71	22544
macro avg	0.27	0.35	0.28	22544
weighted avg	0.55	0.71	0.61	22544

XGBoost Accuracy: 71.40%

(b) Accuracy result for XGBoost Multiclass Classification

Fig. 3: Accuracies for XGBoost Classifier (NSL-KDD)

2) XGBoost Classifier: Compared with AdaBoost, XGBoost showed better results in binary classification and multi-class classification. The binary classification accuracy resulted in 80.09% (refer fig. 3a), which is slightly higher than those obtained by AdaBoost due to its efficiency in handling feature interactions that build better decision boundaries. For the multi-class classification problem, an accuracy of 71.4% was achieved (refer fig. 3b), which was notably higher compared to the one of AdaBoost. The improved performance of XGBoost in multi-class classification can be attributed to its ability to handle feature complexities and mitigate class imbalance more effectively, resulting in better differentiation between multiple attack types. XGBoost managed to hold relatively high accuracy in both cases, showing robustness for both simpler and more complex tasks, unlike AdaBoost

3) Support Vector Classifier (SVC): According to fig. 4a The binary classification performed by a linear kernel Support Vector Classifier achieved an accuracy of 94.64%. The relatively modest accuracy reflects the limitation of a linear kernel in capturing the complex, non-linear relationships between network attack and normal traffic patterns. However, in the case of multi-class classification (refer fig. 4b), SVC performed slightly better with an accuracy score of 98.02% despite its limited linear kernel. This increase probably suggests that the model took advantage of the multi-class margin optimization, where the linear kernel's structure was able to partially accommodate the added complexity of multiple classes. Nonetheless, a relatively low performance of SVC is seen overall, in both the classification tasks. This further suggests that it struggles to handle the complex and nonlinear decision boundaries present in the dataset. Thus, though efficient, SVC may need a nonlinear kernel to compete with the accuracies of the other models. This might require further research.

4) Gradient Boosting Classifier (GBC): In binary classification, the Gradient Boosting Classifier performed very well, with 99.33% accuracy (refer fig. 5a); thus, it outperformed SVC in terms of accuracy. In the multi-class classification task, GBC demonstrated 99.14% accuracy (refer fig. 5b), which is a little lower than the results in binary classification. The added complexity of more classes can be one reason for that. While GBC did not achieve the top performance for either task, the small drop in accuracy between binary and multi-class classifications attests to its strong generalization capabilities and good suitability for complex environments with high-dimensional data.

5) Decision Tree Classifier (DTC): According to fig. 6a, the Decision Tree Classifier had a very high accuracy of 99.59%, showcasing the capability of capturing complicated decision boundaries that were necessary for differentiating between attack and normal traffic. It was also the best-performing model in binary classification tasks. When performing multi-class classification, DTC remained among the best with an accuracy of 99.37% (refer fig. 6b), which is close to the result in binary classification. The slight performance degradation here shows that the Decision Tree was not disturbed too much by the added complexity of more classes. In fact, it was able to efficiently make fine-grained distinctions between various types of attack variations.

6) Random Forest - [NLS-KDD]: The random forest model trained on the binary classified dataset had an accuracy of 99.77% (refer fig. 7a), and the random forest model trained on the multiclass classified dataset had an accuracy of 99.71% (refer fig. 7b). There was a slight decrease in the precision, recall, and f1-score for the multi-class model. Overall, the accuracy and other metrics of the multi-class model is almost the same compared to the binary model, suggesting that

Accuracy: 0.9464263671151709

Classification Report:

	precision	recall	f1-score	support
intrusion	0.99	0.99	0.99	42096
normal	0.99	1.00	0.99	47015
accuracy			0.99	89111
macro avg	0.99	0.99	0.99	89111
weighted avg	0.99	0.99	0.99	89111

(a) Accuracy results for SVC Binary Classification

Accuracy: 0.9802044640953418

Classification Report:

	precision	recall	f1-score	support
Other	0.70	0.09	0.15	325
apache2	0.92	0.99	0.95	137
back	0.93	0.97	0.95	710
guess_passwd	0.91	0.45	0.61	278
ipsweep	0.96	0.92	0.94	2447
mscan	0.88	0.73	0.80	186
neptune	1.00	1.00	1.00	28721
nmap	0.73	0.95	0.82	1019
normal	0.98	0.99	0.99	47153
pod	0.97	0.96	0.97	142
portsweep	0.97	0.98	0.98	1981
processtable	0.97	1.00	0.98	126
satan	0.93	0.93	0.93	2599
smurf	0.98	1.00	0.99	1912
teardrop	0.98	0.99	0.99	598
warezclient	0.80	0.86	0.82	598
warezmaster	0.96	0.57	0.72	179
accuracy			0.98	89111
macro avg	0.92	0.85	0.86	89111
weighted avg	0.98	0.98	0.98	89111

(b) Accuracy results for SVC Multiclass Classification

Fig. 4: Accuracies for Support Vector Classifier (NSL-KDD)

Accuracy: 0.9933453782361324

Classification Report:

	precision	recall	f1-score	support
intrusion	0.99	0.99	0.99	42096
normal	0.99	1.00	0.99	47015
accuracy			0.99	89111
macro avg	0.99	0.99	0.99	89111
weighted avg	0.99	0.99	0.99	89111

(a) Accuracy results for GBC Binary Classification

Accuracy: 0.9914488671432259

Classification Report:

	precision	recall	f1-score	support
Other	0.74	0.50	0.60	325
apache2	0.98	0.96	0.97	137
back	1.00	1.00	1.00	710
guess_passwd	0.99	0.49	0.65	278
ipsweep	0.99	0.99	0.99	2447
mscan	0.75	0.82	0.78	186
neptune	1.00	1.00	1.00	28721
nmap	1.00	0.95	0.97	1019
normal	0.99	1.00	0.99	47153
pod	0.99	0.60	0.75	142
portsweep	0.99	0.99	0.99	1981
processtable	0.84	0.97	0.90	126
satan	0.97	0.98	0.97	2599
smurf	1.00	1.00	1.00	1912
teardrop	0.98	1.00	0.99	598
warezclient	0.98	0.96	0.97	598
warezmaster	0.93	0.90	0.91	179
accuracy			0.99	89111
macro avg	0.95	0.89	0.91	89111
weighted avg	0.99	0.99	0.99	89111

(b) Accuracy results for GBC Multiclass Classification

Fig. 5: Accuracies for Gradient Boosting Classifier (NSL-KDD)

predicting the attack type with this model does not decrease performance in any significant way. The multi-class model had the most difficulty determining attacks in the "Other" category, likely due to the small sample size and variability in the feature values of the various attack types grouped into this category.

7) K-Nearest Neighbors (KNN) - [NLS-KDD]: We used both the binary classified dataset and the multi classified dataset to train our K-Nearest Neighbors (KNN) model which resulted in an accuracy of 98.59% (refer fig. 8a) and 98.08% (refer fig. 8b) respectively. When working on the model with multi-class data, we observed a significant drop in the precision, recall and f1 score. Due to a small sample size, the categories "other", "apache2" and "guess_passwd" hampered the performance of our multi-class model by making it difficult to determine the attacks. Although the accuracy might nearly be similar to the binary model, the reduction in other metrics suggests that there is a decrease in performance when the multi class model is predicting the type of attack.

B. Dataset: CIC-IDS2017

1) Random Forest: The Random Forest model which was trained in a binary classification setting achieved an impressive accuracy of 96.7% [26], while the model that was trained in a multi-class setting reached an notable accuracy of 99.7% (refer fig. 9). This indicates the model's proficiency in managing the

Accuracy: 0.9958590970811684

Classification Report:

	precision	recall	f1-score	support
intrusion	1.00	1.00	1.00	42096
normal	1.00	1.00	1.00	47015
accuracy			1.00	89111
macro avg	1.00	1.00	1.00	89111
weighted avg	1.00	1.00	1.00	89111

(a) Accuracy results for DTC Binary Classification

Accuracy: 0.9936932589691508

Classification Report:

	precision	recall	f1-score	support
Other	0.73	0.66	0.70	325
apache2	0.98	0.95	0.97	137
back	0.99	1.00	1.00	710
guess_passwd	0.97	0.95	0.96	278
ipsweep	1.00	0.99	0.99	2447
mscan	0.90	0.82	0.86	186
neptune	1.00	1.00	1.00	28721
nmap	0.98	0.99	0.98	1019
normal	1.00	1.00	1.00	47153
pod	0.98	0.93	0.95	142
portsweep	0.98	0.98	0.98	1981
processtable	0.97	0.99	0.98	126
satan	0.97	0.97	0.97	2599
smurf	1.00	1.00	1.00	1912
teardrop	0.98	1.00	0.99	598
warezclient	0.96	0.95	0.96	598
warezmaster	0.85	0.92	0.88	179
accuracy			0.99	89111
macro avg	0.96	0.95	0.95	89111
weighted avg	0.99	0.99	0.99	89111

(b) Accuracy results for DTC Multiclass Classification

Fig. 6: Accuracies for Decision Tree Classifier (NSL-KDD)

complexities in the data in a multi-class classification setting. The multi-class scenario presents a promising opportunity for further enhancement to this dataset. By analyzing the classes where the model exhibits suboptimal performance, we can identify specific areas for improvement. Incorporating additional data or features related to these underperforming classes could significantly improve the model's performance, making it more robust and effective in real-world applications.

2) Decision Tree: According to fig. 10, the Decision Tree model performs really well in detecting common classes like benign traffic and DoS Hulk, thus achieving high precision and recall for these categories. However, it struggles with

Classification Report:

	precision	recall	f1-score	support
intrusion	1.00	1.00	1.00	13970
normal	1.00	1.00	1.00	15734
accuracy			1.00	29704
macro avg	1.00	1.00	1.00	29704
weighted avg	1.00	1.00	1.00	29704

(a) Accuracy results for RF Binary Classification

Classification Report:

	precision	recall	f1-score	support
Other	0.89	0.74	0.81	108
apache2	1.00	1.00	1.00	46
back	1.00	1.00	1.00	237
guess_passwd	1.00	0.94	0.97	93
ipsweep	1.00	1.00	1.00	816
mscan	1.00	0.98	0.99	62
neptune	1.00	1.00	1.00	9574
nmap	0.99	0.99	0.99	340
normal	1.00	1.00	1.00	15718
pod	0.96	1.00	0.98	47
portsweep	0.99	1.00	1.00	660
processtable	1.00	1.00	1.00	42
satan	0.98	0.99	0.99	866
smurf	1.00	1.00	1.00	637
teardrop	0.99	1.00	0.99	199
warezclient	1.00	0.98	0.99	199
warezmaster	1.00	0.88	0.94	60
accuracy			1.00	29704
macro avg	0.99	0.97	0.98	29704
weighted avg	1.00	1.00	1.00	29704

(b) Accuracy results for RF Multiclass Classification

Fig. 7: Accuracies for Random Forest (NSL-KDD)

the minority classes such as DoS GoldenEye, DoS slowloris and especially Heartbleed where low precision, recall and F1-scores indicate frequent misclassifications. This disparity suggests that the model is biased more towards majority classes resulting in a weaker detection for rare attacks. The overall accuracy is 89.04% but the lower macro-averaged F1-score of 0.69 highlights the need for additional strategies like class balancing or more complex models to enhance the detection of minority attack types and ensure balanced performance across all the available classes.

3) XGBoost: According to fig. 11, the XGBoost model exhibits outstanding performance on the CIC-IDS2017 dataset achieving perfect precision, recall and F1-scores for most classes including BENIGN, DoS Hulk, DoS Slowhttptest and DoS slowloris. With an overall accuracy of 99.79% the model effectively identifies all the major attack types and benign traffic. However, for the rare Heartbleed attack, while the precision remains high at 1.00, recall drops to 0.50 resulting in a lower F1-score of 0.67. This discrepancy suggests that despite the model's strong classification capabilities, very rare attack types may still benefit from additional data or specialized techniques for more consistent detection. The macro average F1-score of 0.94 confirms balanced performance across all the

Classification Report:

	precision	recall	f1-score	support
intrusion	0.99	0.98	0.98	13970
normal	0.99	0.99	0.99	15734
accuracy			0.99	29704
macro avg	0.99	0.99	0.99	29704
weighted avg	0.99	0.99	0.99	29704

(a) Accuracy results for KNN Binary Classification

Classification Report:

	precision	recall	f1-score	support
Other	0.70	0.24	0.36	108
apache2	0.80	0.89	0.85	46
back	0.91	0.96	0.93	237
guess_passwd	0.75	0.65	0.69	93
ipsweep	0.94	0.96	0.95	816
mscan	0.87	0.42	0.57	62
neptune	1.00	1.00	1.00	9574
nmap	0.87	0.94	0.90	340
normal	0.98	0.99	0.99	15718
pod	0.96	0.98	0.97	47
portsweep	0.94	0.97	0.96	660
processtable	1.00	0.81	0.89	42
satan	0.95	0.94	0.94	866
smurf	0.98	1.00	0.99	637
teardrop	0.98	0.99	0.98	199
warezclient	0.91	0.83	0.87	199
warezmaster	0.86	0.50	0.63	60
accuracy			0.98	29704
macro avg	0.91	0.83	0.85	29704
weighted avg	0.98	0.98	0.98	29704

(b) Accuracy results for KNN Multiclass Classification

Fig. 8: Accuracies for K-Nearest Neighbors (NSL-KDD)

	precision	recall	f1-score	support
BENIGN	1.00	1.00	1.00	88006
DoS GoldenEye	0.98	0.99	0.99	2059
DoS Hulk	1.00	1.00	1.00	46215
DoS Slowhttptest	0.99	0.99	0.99	1100
DoS slowloris	1.00	0.99	0.99	1159
Heartbleed	1.00	0.50	0.67	2
accuracy			1.00	138541
macro avg	0.99	0.91	0.94	138541
weighted avg	1.00	1.00	1.00	138541

Fig. 9: Accuracy results for Random Forest (CIC-IDS2017)

classes with minor exceptions for rare cases.

V. Discussion

Several of the machine learning models show potentially significant consistency in key metrics (accuracy, precision, recall and f1-score) between binary and multi classification tasks that indicate they may be useful in classifying the attack type in network intrusion detection systems. Notably, the Decision Tree Classifer, Random Forest and Gradient Booster models had the best translation from the dual classified dataset to the multi-class dataset. On the contrary, AdaBoost and K-

	precision	recall	f1-score	support
BENIGN	1.00	0.84	0.91	88006
DoS GoldenEye	0.39	0.95	0.55	2059
DoS Hulk	0.83	0.99	0.91	46215
DoS Slowhttptest	0.86	0.81	0.84	1100
DoS slowloris	0.29	0.84	0.43	1159
Heartbleed	0.50	0.50	0.50	2
accuracy			0.89	138541
macro avg	0.64	0.82	0.69	138541
weighted avg	0.92	0.89	0.90	138541

Fig. 10: Accuracy results for Decision Tree (CIC-IDS2017)

	precision	recall	f1-score	support
BENIGN	1.00	1.00	1.00	88006
DoS GoldenEye	1.00	0.99	0.99	2059
DoS Hulk	1.00	1.00	1.00	46215
DoS Slowhttptest	1.00	0.99	0.99	1100
DoS slowloris	1.00	0.99	0.99	1159
Heartbleed	1.00	0.50	0.67	2
accuracy			1.00	138541
macro avg	1.00	0.91	0.94	138541
weighted avg	1.00	1.00	1.00	138541

Fig. 11: Accuracy results for XGBoost (CIC-IDS2017)

Nearest Neighbours performed the worst and had a significant drop in performance with the multi-class predictions.

AdaBoost and K-Nearest Neighbours could not keep up its accuracy with the classification complexity increase. In binary classification, AdaBoost performed with an accuracy of 78% when detecting between general attacks and normal traffic, but when it came to multi-class the model's accuracy dropped to 62%, suggesting difficulty in distinguishing between different attack types. Likewise, KNN faced a performance reduction. There was not a significant loss in accuracy, but the precision, recall and F1-score of the model dropped substantially in the multi-class setting. The above mentioned models explain the shift in complexities introduced when moving beyond binary classification, where the class boundaries become non-linear, leading to reduced accuracy.

A comparison between classifiers, Decision Tree Classifier (DTC), Random Forest, Gradient Boosted Classifier (GBC) models showed promising performance. DTC was successful in both tasks maintaining near peak accuracy with 99.59% in binary and 99.37% in multiclass classification. The graph of the precision and recall for Decision Tree model on multiclass classification can be seen in Figure 12. This makes the DTC suitable for cases with a need for high precision. GBC, with a binary classification accuracy of 99.33% and for the multiclass task has a slight dip to 99.14%, demonstrated solid generalizability while handling complex classification without overfitting.

XGBoost and Random Forest consistently showed higher accuracy across both linear and binary and multi-class data. XGBoost is able to maintain consistent performance even

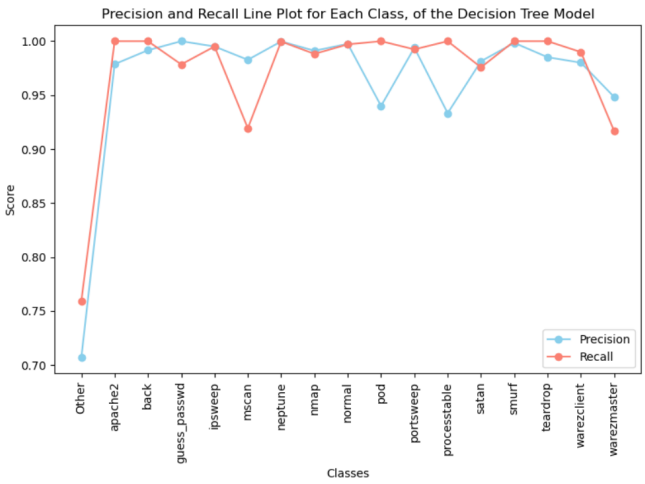

Fig. 12: Precision and Recall for Decision Tree Classifier

with feature complexities and class imbalances, while Random Forest has near-perfect accuracies in both Datasets. This level of consistency shows the model's robustness and adaptability across complex environments showing they are ideal candidates for both binary and multi-class classifications in environments which have little or no room for error like network intrusion detection. Although there are risks of overfitting the dataset which could be avoided by implementing a 60-40 data split while using log loss functions to track training progress and monitoring model performance over time.

The imbalance of classes posed challenges to some classifications, mostly in multi-class classifications. The current main issue in the existing network datasets like CIC-IDS2017 is that many of the class labels are under-represented (example, attacks like Heartbleed and Apache2), which result in low accuracy, recall and F1 scores in models such as Decision Tree and KNN. To address these issues, well-balanced datasets can be created that can help build better Intrusion Detection Systems. Also, new datasets can incorporate newer attacks that were absent in the previous generation network datasets. Artificial Intelligence can be used for this purpose, creating a dataset that better captures a range of attacks.

To refine the present findings, future research could explore deep learning strategies and neural networks to improve complex and high-level information processing in a new way. On the other hand we could also generate more representative datasets covering a wider range of network attack scenarios. Models like the Support Vector Classifier, which showed improvement potential with non-linear kernels, could also be revisited. Exploring kernel alternatives could offer improved results, especially in high-dimensional spaces with non-linear data separations.

Key observations across both binary and multi-class classifications are that the Decision Tree Classifier consistently exhibited the highest accuracy, which reinforces its adaptability and precision in identifying subtle network traffic distinctions. The Gradient Boosting Classifier performed reliably on both tasks and was only slightly sensitive to the classification complexity, which suggests good generalizability. On the other hand, the efficient Support Vector Classifier showed constraints in performance, indicating it might need non-linear kernels to behave at its best on high-dimensional, non-linearly separable datasets such as KDD. From the experiments, in the final analysis, all three classifiers gave reliable results, yet the Decision Tree Classifier and Gradient Boosting Classifier performed better for tasks having complex distributions of classes. The Decision Tree, with high precision and stability, is recommended for tasks requiring detailed decision-making, while Gradient Boosting provides a strong alternative with robust generalization across binary and multi-class classifications. The Support Vector Classifier can be a good alternative in simpler cases, and it even has a chance to be improved by using kernels.

VI. Conclusion

In this study, we assessed the effectiveness of various machine learning models on a multi-class dataset and contrasted these findings with existing research, where models are commonly trained on binary-class datasets for network intrusion detection. Our analysis employed classifiers such as Random Forest (RF), Support Vector (SV), Gradient Boosting (GB), Decision Tree (DT), K-Nearest Neighbors (KNN), AdaBoost and XGBoost, trained and tested on the NSL-KDD and CIC-IDS-2017 datasets. By evaluating model performance on multi-class data, we gained deeper insights into each model's strengths and limitations in handling the more complex, realistic scenarios that a multi-class approach represents for Intrusion Detection Systems (IDS).

The results reveal that most models achieve high accuracy even when operating on multi-class data, supporting the view that machine learning can serve as a reliable option for constructing robust Intrusion Response Systems (IRS). Notably, the Decision Tree model consistently demonstrated the highest accuracy among all classifiers, underscoring its effectiveness in accurately identifying a wide range of network intrusions while maintaining a low false-positive rate. This makes Decision Trees particularly suited for dynamic intrusion detection tasks that require precision and reliability in diverse threat landscapes.

Building on these findings, future work should explore the use of deep learning architectures and advanced neural networks on larger and more complex datasets that exhibit minimal class imbalance and significant randomness. This focus would allow for further investigation into how deep learning methods can enhance detection accuracy in environments where data is less structured and more unpredictable. Additionally, supervising the training process to prevent overfitting will be crucial to ensuring that the models generalize effectively to unseen data rather than following an overly idealized curve.

As well, further research should test these models in real time to analyze their potential to be used in practical applications. Certain model types may not be efficient enough to

keep up with real network traffic. There is also the potential for new traffic patterns to emerge over time in a real world environment, so seeing how the models perform is critical. The efficiency of retraining should be analyzed to prevent long term losses in accuracy in these scenarios. By gaining an understanding of the performance of these models in applied network intrusion detection systems, the most effective models become known.

To further improve the robustness of NIDS frameworks, incorporating hybrid models that blend signature-based and anomaly-based detection methods could offer a more balanced and adaptable solution. This approach may optimize both detection accuracy and operational efficiency, paving the way for the development of a comprehensive and reliable Intrusion Detection System. Expanding on this research, the ultimate goal is to develop a fully integrated, resilient intrusion detection solution capable of real-time, adaptive threat management in diverse network environments.

REFERENCES

[1] J. S. M. Osorio, J. A. V. Tejada and J. F. B. Vega, "Detection of DoS/DDoS attacks: the UBM and GMM approach," 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), Bordeaux, France, 2021, pp. 866-871.

[2] "Network Intrusion dataset (CIC-IDS-2017)" [Online] Available on Kaggle: https://www.kaggle.com/datasets/chethuhn/network-intrusion-dataset

[3] S. Duque and M. N. Omar, "Using Data Mining Algorithms for developing a model for Intrusion Detection Systems," 2015 Complex Adaptive Systems Conference, Missouri University of Science Technology, San Jose, CA.

[4] J. Liu, M. Simsek, B. Kantarci, M. Bagheri and P. Djukic, "Collaborative Feature Maps of Networks and Hosts for AI-driven Intrusion Detection," GLOBECOM 2022 - 2022 IEEE Global Communications Conference, Rio de Janeiro, Brazil, 2022, pp. 2662-2667, doi: 10.1109/GLOBECOM48099.2022.10000985.

[5] D. E. Denning, "An Intrusion Detection Model," IEEE Transactions on Software Engineering, vol. SE-13, no. 2, pp. 222–232, Feb. 1987.

[6] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," National Institute of Standards and Technology, 2007.

[7] T. Nguyen, "The Challenges in SDN/ML Based Network Security: A Survey," North Carolina State University, 2021.

[8] M. Casado, T. Garfinkel, A. Akella, M. J. Freedman, D. Boneh, N. McKeown, and S. Shenker, "SANE: a protection architecture for enterprise networks," 15th USENIX Security Symposium, 2006.

[9] G. Stabler, A. Rosen, S. Goasguen, and K.-C. Wang, "Elastic IP and security groups implementation using OpenFlow," Proceedings of the 6th international workshop on Virtualization Technologies in Distributed Computing, 2012.

[10] R. Braga, E. Mota, and A. Passito, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," Conference on Local Computer Networks (LCN), 2010.

[11] K. Giotis, G. Androulidakis, and V. Maglaris, "Leveraging SDN for efficient anomaly detection and mitigation on legacy networks," 2014 3rd European Workshop on Software-Defined Networks, 2014.

[12] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," Springer - Special Issue on Software Defined Networking, 2018.

[13] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," 2010 IEEE Symposium on Security and Privacy, 2010.

[14] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing Machine Learning Models via Prediction APIs," 25th USENIX Security Symposium, 2016.

[15] "Microsoft Security Development Lifecycle."

[16] T. Tulabandhula and C. Rudin, "Machine Learning with Operational Costs," Journal of Machine Learning Research, vol. 14, 2013.

[17] "KDD Cup 1999 Data" [Online] Available: https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[18] I. Sharafaldin, A. Habibi Lashkari, and A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," 2018, pp. 108-116, 10.5220/0006639801080116.

[19] M. J. M. Turcotte, A. D. Kent, and C. Hash, "Unified Host and Network Data Set," arXiv:1708.07518 [cs], Aug. 2017.

[20] Lirim et al., "Convolutional Neural Networks for Intrusion Detection in UNSW-NB15 Dataset," Applied Sciences, vol. 12, no. 22, 2021.

[21] Lin et al., "Hybrid Intrusion Detection System Using Convolutional Neural Networks and Suricata," Journal of Cybersecurity, vol. 8, no. 4, 2021.

[22] A. Rohit et al., "Ensemble Learning for Improved Intrusion Detection: A Study on KDDCup99 Dataset," International Journal of Network Security, vol. 15, no. 3, 2021.

[23] Z. Yu et al., "Few-Shot Learning for Intrusion Detection in NSL-KDD and UNSW-NB15 Datasets," Cyber Defense Review, vol. 6, no. 2, 2021.

[24] A. Al-Yaseen et al., "Hybrid SVM-ELM Model for Multi-Level Intrusion Detection," Journal of Information Security and Applications, vol. 58, 2021.

[25] NSL-KDD Data, [Online] Available: https://www.kaggle.com/datasets/kiranmahesh/nslkdd/data.

[26] C. C. Yang, "Intrusion Detection System - Network Intrusion Dataset (CIC-IDS-2017)," [Online] Available: https://www.kaggle.com/code/chuckcharlesyang/intrusion-detection-system.

[27] S. I. Serengil, "A Step by Step Adaboost Example," December 26, 2021. [Online] Available: https://sefiks.com/2018/11/02/a-step-by-step-adaboost-example/#google_vignette.

[28] Hemavatisabu, "Data Pre-processing with Sklearn Using Standard and Minmax Scaler," February 3, 2022. [Online] Available: https://www.geeksforgeeks.org/data-pre-processing-wit-sklearn-using-standard-and-minmax-scaler/?ref=header_outind.