

XAI Securenet: Explainable AI Techniques for Robust Intrusion Detection in IoT Infrastructures

Rushikesh Gaikwad , Karan Nair , Avinash Sahani , Ritika , Anita Nalawade

(Dept. of Electronics & Computer Science, Shah and Anchor Kutchhi Engineering College, Chembur, Maharashtra, India
Email: rushikesh.gaikwad16489@sakec.ac.in)

(Dept. of Electronics & Computer Science, Shah and Anchor Kutchhi Engineering College, Chembur, Maharashtra, India
Email:karan.nair16728@sakec.ac.in)

(Dept. of Electronics & Computer Science, Shah and Anchor Kutchhi Engineering College, Chembur, Maharashtra, India
Email: avinash.sahani16708@sakec.ac.in)

(Dept. of Electronics & Computer Science, Shah and Anchor Kutchhi Engineering College, Chembur, Maharashtra, India
Email:ritika.16462@sakec.ac.in)

(Dept. of Electronics & Computer Science, Shah and Anchor Kutchhi Engineering College, Chembur, Maharashtra, India
Email:anita.nalawade@sakec.ac.in)

Abstract:

The paper addresses the employment of machine learning models in Intrusion Detection Systems (IDS) in Cloud and iot settings, which produce high network traffic. Even though these models perform well, they tend to lack interpretability, which brings down the confidence level of security experts. The paper suggests employing shapley Additive explanations (SHAP), an explainable AI approach, to enhance the interpretability of ML models so that they are even more effective in identifying cyberattacks. This study explores the creation of a network intrusion detection system (NIDS) with a random forest classifier learned from the CIDDS001 dataset. The system produced a classification accuracy of 99.56runtime performance with predictive capability. The model was contrasted with baseline classifiers and evaluated in real-world settings with Wireshark. The research provides a solid, interpretable, and deployable NIDS solution for modern network security problems.

Keywords — Intrusion Detection System (IDS), Explainable AI (XAI), SHAP, Random Forest Classifier, Network Security, IoT, Cloud Computing, Cyberattack Detection, Machine Learning Interpretability, NIDS

I. INTRODUCTION

Growth in iot devices and cloud computing has accelerated traffic volume and complexity in the network, where security measures are urgently needed. Legacy intrusion detection systems (IDS) are inadequate against contemporary attacks. Machine learning (ML) presents an adaptive mechanism by executing automated analysis and discovering anomalies. However, ML models are nontransparent, something that security analysts lack. Explainable AI (XAI) promises to reveal the transparency behind model decisions to provide

better interpretability of ML-driven IDS for Cloud and iot domains. Network protection is essential within contemporary digital landscapes, as network threats such as dos and port scanning threaten the security landscape. Sophisticated detection strategies are required in order to spot and counter malignant behavior. NIDS is implemented to analyze traffic patterns and track anomalies. Older datasets such as KDD99 And NSL-KDD do not accurately present real-world network behavior. The CIDDS-001 dataset, released in 2017, provides a flow-based model of network interactions. Machine learning approaches, such as Random Forest Classifier, have

transformed the development of NIDS by offering high accuracy and resistance to a wide range of threats. In this paper, we use the Shapely Addictive Explanations (SHAP) [1], a game theory approach to explain the output of machine learning models by assigning importance to features, to analyze AI-based IDS.

One of the fundamental challenges in cybersecurity is the detection of network threats, and various results have been reported in the field of network intrusion detection systems (NIDSs). In particular, the most recent studies have been focused on applying the artificial intelligence (AI) technology to NIDS, and AI-based intrusion detection systems have achieved remarkable performance. Initially, the research primarily focused on applying traditional machine learning models, such as decision trees [6] (DTs) and support vector machines [7] (SVMs) to existing intrusion detection systems, and it has now been extended to deep learning approaches [8], such as convolutional neural networks (CNNs), long short-term memory (LSTM), and autoencoders.

It is very important to detect a network intrusion quickly and accurately for stable operation of the network. For this purpose, a dedicated security device called the Network Intrusion Detection System (NIDS) was proposed [9], [10]. The initial NIDS generated patterns from existing attacks and detected intrusions very quickly and accurately through pattern matching with the received packets [11]–[12].

II. INTRUSION DETECTION SYSTEM

Intrusion Detection Systems (IDS) play a key role in keeping networks secure, especially in environments like Cloud and Internet of Things (IoT) where data changes quickly and there are limited resources. IDS come in two main types. The first type is signature-based IDS, which checks for attack patterns that are already known. The second type is anomaly-based IDS, which uses machine learning to identify unusual network activities. A challenge with anomaly-based IDS is that the machine learning models they use are often complicated and hard to understand. The authors in [2] identify the advantages and disadvantages of existing IDS

approaches. To do this they identified how different types of attacks are being detected. The discussed machine learning methods for IDS have not been tested or evaluated in real-time, specifically for IoTs. Many of the IDS that use ML techniques are specifically created for Wireless Sensor Networks or traditional internet architecture. This paper addresses this issue by applying explainable AI techniques, aiming to make machine learning-based IDS both effective and easier to interpret. An IDS is a cybersecurity tool that monitors network traffic, looking for signs of unauthorized access, cyber threats, or breaches of rules. It provides an extra security layer for organizations, helping them detect harmful activities as they occur, so they can take action quickly to prevent serious damage. Instead of stopping attacks directly, IDS focus on spotting suspicious or out-of-the-ordinary behavior. This makes them a vital part of a solid security plan.

A. How IDS Works

The Intrusion Detection System (IDS) constantly monitors system activities and network traffic. It tests every observation against established guidelines and patterns of known attacks, or notifies unusual activity even when no rules are violated. This capability enables it to detect system intrusions well. The IDS has a well-defined threat detection process with firm steps for dealing with security threats.

- 1) **Data Collection:** The IDS begins by collecting data in network packets, system logs, and user activity logs. This information is the basis for identifying unusual patterns that may indicate an attack.
- 2) **Traffic Analysis and Pattern Matching:** The IDS employs two fundamental tricks to analyze data
 - **Signature-Based Detection:** Compares incoming data against a list of known attack signatures, issuing an alert if there's a hit—but it only captures familiar threats.
 - **Anomaly-Based Detection:** Learns normal system behavior and raises flags when something is off, so it's perfect for catching novel, unfamiliar attacks such as zero-day attacks

- 3) Alert Generation: When the IDS catches something fishy, it generates alerts containing information on the type of attack, the targeted system, and potential entry points, providing the analyst with a swift heads-up to respond.
- 4) Incident Response and Reporting: When the IDS detects something fishy, it generates alerts containing information on the type of attack, the targeted system, and potential entry points, providing analysts a prompt heads-up to act.

III. TYPES OF INTRUSION DETECTION SYSTEMS

Two IDS categories exist to fulfill specific security requirements:

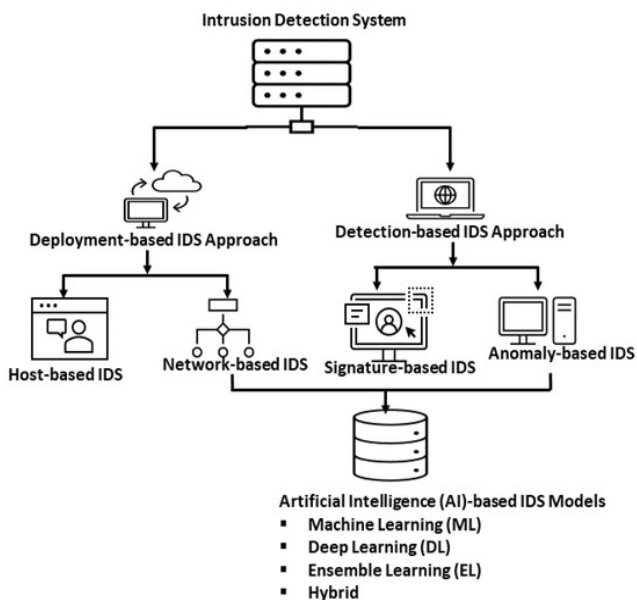


Fig. 1. XAI for Intrusion Detection System

1. **Network-Based Intrusion Detection System (NIDS):**
 - Monitors traffic across an entire network
 - The system detects suspicious activity through its examination of packet headers along with their content in the payload.
2. **Host-Based Intrusion Detection System (HIDS):**
 - Installed on individual devices (hosts) to monitor internal system activity.
 - Detects unauthorized file modifications, registry changes, or login attempts

VI. EXPLAINABLE AI AND SHAP

XAI functions as a collection of methods which enables understanding about AI system decision processes. Security analysts heavily rely on XAI technology in intrusion detection since it reveals the reasons behind data pattern classifications into safe or harmful categories. SHapley Additive Explanations stands as one leading method of XAI technology known as SHAP. The machine learning model explanation process uses SHAP by applying conceptual components from cooperative game theory. SHAP provides scores that evaluate input feature importance regarding model prediction results. This methodology generates clear explanations of specific predictions along with wider knowledge on which model characteristics drive the predictive behavior. SHAP proves invaluable for cybersecurity applications since it enables the handling of complex systems through easier functionality. The usage of randomforestclassifier feature contributions was evaluated through SHAP which operates as a tree-based library for 30 sample calculation with optimization. Explainer.shap values with its subsequent output produced a multi-dimensional array structure for the SHAP values. The evaluation results showed duration as the most determinant model characteristic followed by tcp syn and tcp fin measurements. The analysis streamlines model decision-making processes which makes the platform more robust to validation inspections and trustworthy assessments.. SHAP has a wide array of plots that can be used to visualize the contributions may be by each of the features[3]. Global explanation

A. SHAP Summary Plot for Attack Traffic

1) Structure:

- Y-Axis (Features): Orders features by significance (e.g., tcp psh, bytes, duration).
- X-Axis (SHAP Value): SHAP values (negative = malicious, positive = normal) (as per your predict.py code).
- Color Gradient: Red (high feature values), Blue (low).

- Points and Spread: Each dot represents a SHAP value for an instance, with horizontal spread showing variability.

2) *Detailed Interpretation:*

- Large values of tcp psh, bytes, and duration indicate normal traffic. Small values of these features (e.g., short durations or small number of bytes) indicate potential attacks.
- High tcp syn or DoS presence pulls in the direction of the malicious class.
- Presence of features such as tcp urg or proto IGMP has a negligible effect.

3) *Insights:*

The model is very much dependent on traffic flow and TCP flag patterns to identify attacks. Small values and the misuse of flags are very good indicators of malicious activity.

B. SHAP Summary Plot for Normal Traffic

1) *Structure:*

- Y-Axis (Features): Same features as the attack plot, ranked by importance
- X-Axis (SHAP Value): Ranges from -0.3 to 0.3, with negative values pushing toward "malicious" and positive toward "normal."
- Color Gradient: Red (high values) to blue (low values)
- CPoints and Spread: Similar to the attack plot, showing instance-level SHAP values.

2) *Detailed Interpretation:*

- Large values of tcp psh, bytes, and duration are strong indicators of normal classification.
- More positive SHAP values than in the attack plot indicate improved normal traffic handling.
- Attack-type features do not have a significant impact here.

3) *Insights:*

Normal traffic is characterized by long sessions, large volumes of data, and normal TCP behavior. The model assigns the same

features to both types of traffic but in a different context evaluates them.

V. METHODOLOGY

The approach adopts machine learning models to label network traffic as malicious or harmless in IoT and Cloud environments. Three datasets were employed: NF-ton-iot-v2, CIDDS-001, and CIDDS-002. Decision Trees (DT), Random Forest (RF), Logistic Regression (LR), and Feed-Forward Neural Networks (FFNN) were implemented. The models were tested on the basis of accuracy, precision, recall, and F1-score to achieve a balanced evaluation of performance. Subsequently to training, SHAP was used to clarify their choices and yielded feature importance scores and visuals for determining what network features contributed most significantly toward labeling traffic as malicious. The process allows a controlled training environment and testing space for ML models with an equitable measurement of performance.

A. Dataset Description

CIDDS-001 Dataset [4]: The Coburg Intrusion Detection Dataset (CIDDS-001) is a flow-based dataset. The dataset is used to evaluate anomaly based intrusion detection systems. A rich description of openstack and External Server network traffic constitutes the CIDDS-001 dataset which Ring et al. (2017) collected. The dataset combines typical network signals with malicious ones to deliver comprehensive knowledge about network functionality. The dataset maintains its focus on cloud traffic because researchers aim to enhance modern architecture security despite an extreme disparity between attack and normal traffic samples.

B. Data Preprocessing

The raw CIDDS-001 data underwent preprocessing as a necessary step to adapt it for machine learning applications. The preprocessing steps united information from both openstack and External Server parquet files until they merged into a standardized format. The addition of a new binary label made classification easier because it converted the task into a two-class problem. A machine learning

Intrusion Detection System (IDS) obtains its accuracy from the quality of data which exists during training and testing. IDS depends on a structured and diverse dataset which enables it to identify properly between normal and malicious traffic thus increasing its performance. The research employs CIDD-1 as its data source for this investigation given that it is known as the Coburg Intrusion Detection Data Set. CIDD-1 serves as one of the leading testing platforms for intrusion detection techniques among researchers. We describe CIDD-1 along with its preprocessing requirements that include feature selection and value encoding followed by value imputation and RF model compatibility. Our well-prepared dataset enables the creation of an accurate IDS with SHAP Additive explanations (SHAP) that provides explainable capabilities.

C. Model Development

The Random Forest algorithm proved the ideal choice for network traffic binary classification because it performs ensemble learning while displaying robustness along with high interpretability. The model received its training data from a balanced dataset where parameter optimization and settings adjustments happened through grid search resulting in 100 trees with maximum depth set at 10. The scikit-learn library facilitated simple implementation of the model which was later serialized with joblib for deployment use. The Random Forest method effectively identified varied patterns of normal and attack traffic in CIDD-001. Random Forest is an ensemble learner that trains a number of decision trees at training time and then makes predictions from their combined decisions. Each of the trees is trained on a random subset of the data and the overall prediction is made through majority voting (in case of classification) or averaging (in case of regression). This eliminates overfitting and enhances generalization, so Random Forest is especially well adapted to IDS.

1) *Working of Random Forest in IDS:* The process of applying Random Forest in IDS is as follows:

- Bootstrap Sampling: The data is split into several subsets based on random sampling with replacement. Each subset is utilized to train a separate decision tree.
- Decision Tree Training: Every tree is constructed based on a random subset of features to introduce diversity among trees.
- Majority Voting: In classification problems, the overall prediction is obtained through majority voting of all the trees.
- Feature Importance Analysis: Through Random Forest analysis administrators can view the significance of each feature which enables them to identify important performance indicators of harmful behavior.

2) *Benefits of Random Forest in IDS:*

- High Accuracy: Random Forest (RF) [5] is a supervised learning model that is used for classification. It consists of a collection of individual decision trees that operate together.
- Robustness to Overfitting: The ensemble model minimizes overfitting risks, providing consistent performance on unseen data.
- Scalability: Random Forest is capable of processing large data sets, making it ideal for high-traffic networks in cloud and IoT networks.
- Feature Interpretability: The model outputs feature importance scores, allowing analysts to see which attributes are most responsible for attack detection.
- Robust Performance: Random Forest performs better than conventional rule-based IDS and other ML models at detecting cyber threats.

3) *Comparing Random Forest with Other IDS Models :*

Model	Accuracy	Training Time	Interpretability	Remarks
Decision Tree	Medium	Fast	Moderate	Easy to understand; fast training but struggles with complex IDS tasks.
Support Vector Machine (SVM)	Very High	Very Slow	Medium	Accurate but slow and lacks explainability, limiting its IDS use.
Neural Networks	Very High	Very Low	Limited	Highly accurate; not suitable for real-time IDS due to its opaque nature.
Random Forest	High	Medium	Medium-High	Balanced in performance and interpretability; best fit for IDS applications

attributes in both count and order. The 'preprocess new data()' function extracts data from Parquet files, ensuring consistency in feature names, filling missing entries with zeros, while keeping the original formats for duration, packets, and bytes. The 'predict()' function retrieves the model from 'cidds rf model.joblib'. Distinct prediction boundaries adjust evaluation thresholds, with normal traffic at '0.30' and attack traffic at '0.70'. The predictions classify traffic as normal ('1') or malicious ('0'), generating key statistics from parquet files for analysis.

VI. WORKING

a) Prediction model

The script employs Python to analyze network traffic data using a pre-trained Random Forest model. Initially, it loads essential libraries like 'joblib' for model retrieval, 'pandas' for data management, and 'numpy' for mathematical calculations.

It consists of two primary functions: 'preprocess new data()' for data preparation and 'predict()' for model predictions.

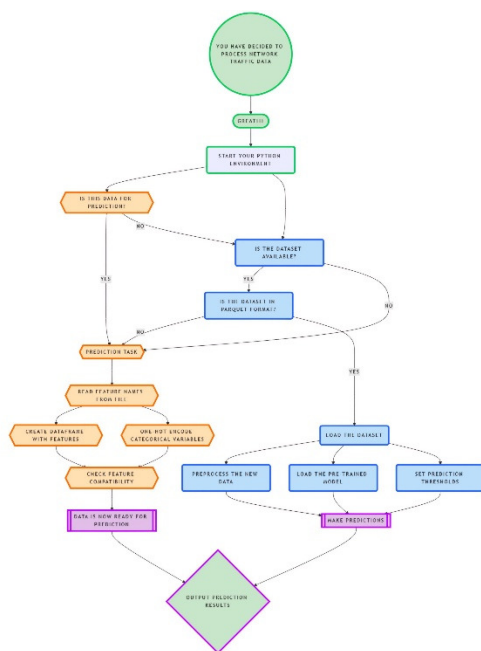


Fig. 2. Prediction model

Categorical data is encoded using one-hot encoding for the protocol field and attack type. The model requires input 'X' to align with training set

b) Training Model

The Python script develops a machine learning pipeline with a Random Forest classifier to distinguish between normal and malicious network traffic patterns. It first imports required libraries such as 'pandas' for data handling, 'numpy' for numeric operations, 'scikit-learn' for modeling, and 'shap' for result explanation. Performance monitoring incorporates inbuilt execution time measures. Employing a Parquet file as the source of data guarantees efficient processing.

Once the data has been imported, the script detects and drops missing values to ensure data integrity. The categorical columns, 'proto' and 'attack type', are encoded into numerical form using one-hot encoding prior to processing. The target variable 'is normal' is renamed to 'label normal', with malicious traffic coded as 0 and normal coded as 1. Unwanted columns are dropped to keep only key data, formatting all into 'float64'. Target variable ('y') and features ('X') are divided, then the set is divided into 20 percent test and 80 percent training.

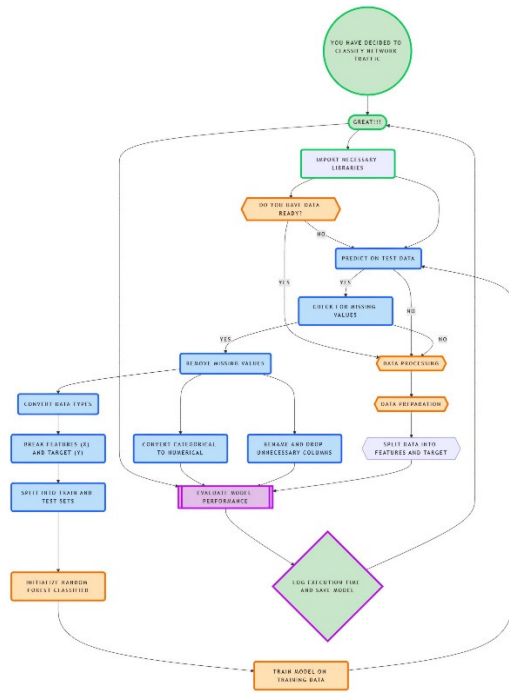


Fig. 3. Training model

The model has 50 decision trees and 10 maximum depth, using parallel processing for speed. Performance is measured after training using accuracy, precision, recall, and F1-score. SHAP values are calculated to interpret feature effect, with visualizations done if necessary. Lastly, the trained model and feature names are stored using 'joblib', together with processing time measurements for efficiency evaluation. Data preparation, modeling, evaluation, and interpretability are summarized into a complete workflow in the script for network traffic classification.

VII. EVALUATION METRICS

The performance of an IDS system was evaluated using various metrics, including accuracy, precision, recall, F1- score, and ROC-AUC. Accuracy measured overall correctness, precision measured the ratio of true positives to predicted positives, and recall measured the rate of actual positives correctly identified. These metrics were calculated with 5-fold cross-validation to ensure reliability across varying data splits. Confusion matrices provided fine-grained insights into classification results.

The normal traffic prediction is mainly dependent on SHAP values, which emphasizes the significance of bytes, TCP flags, and duration, whereas attack features have negligible influence. The effect of high or low values is expressed by plot colors, depicting significant traffic features that influence predictions of normal behavior. Improvement in network monitoring systems can be made by detecting key predictive features for better detection of normal as well as anomalous traffic. The analysis compares different traffic properties to determine the effect of such properties on a machine learning-based model identifying traffic as an attack. The model becomes more transparent since SHAP values illustrate how various features are important in terms of making a prediction. The TCP Push flag (tcp_psh) is the most important, with changes having a major impact on attack identification, and bytes sent, connection time, tcp_syn, and tcp_ack are also important features. Secondary impactful features are proto_ICMP, packet numbers, and Type of Service (tos). On the other hand, features such as attack id, attack type portscan, and attack type bruteforce have low SHAP values, contributing minimal impact. This analysis supports strengthening cybersecurity through enhanced strategies

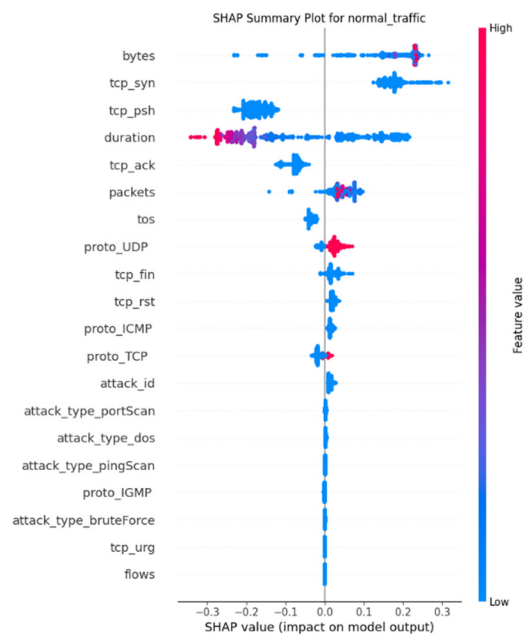


Fig. 4. SHAP Summary Plot for Normal Traffic

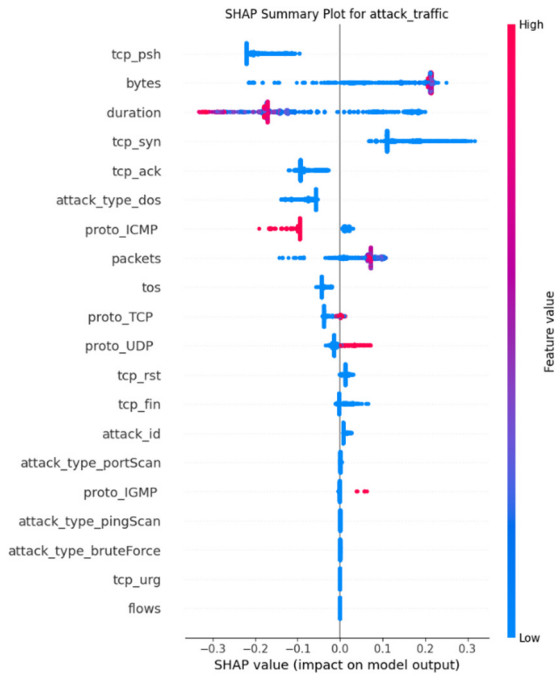


Fig. 5. SHAP Summary Plot for Attack Traffic

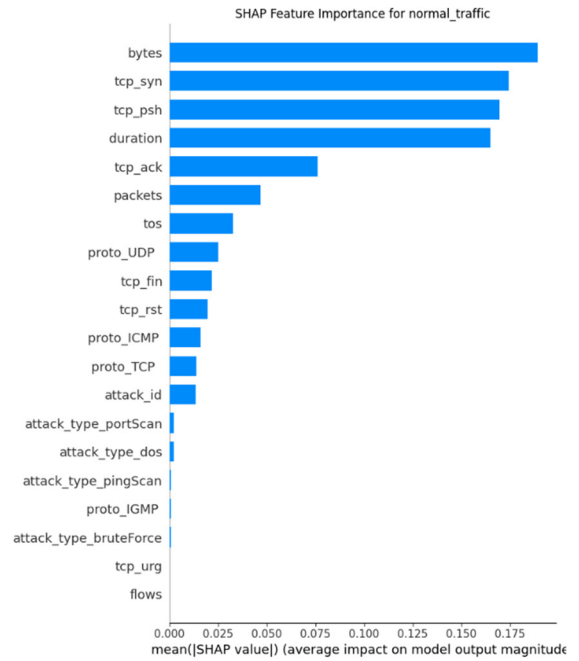


Fig. 6. shap summary bar normal traffic

The SHAP (Shapley Additive Explanations) feature importance plot shows important network traffic features that drive model predictions for normal patterns. Every model feature is given a SHAP value, which represents its contribution to prediction results. The higher the SHAP value, the greater the impact on the predictions. The story identifies the critical features in determining normal traffic patterns, where bytes, tcp syn, and tcp psh are the top three features in making correct predictions. On the other hand, tcp urg and flows have limited predictive power. The amount of data in terms of bytes plays a crucial role in distinguishing normal from abnormal network activity. Decision-making is predominantly based on TCP flags, duration of connection, and packet numbers as important determinants. Less significance is accorded to properties such as the type of service and other network protocols, in addition to the termination signals for TCP connections. Features associated with attacks, like portscan, dos, and bruteforce types, contribute little to defining normal traffic behaviors. In sum, the discussion highlights certain features that are indispensable for successful prediction.

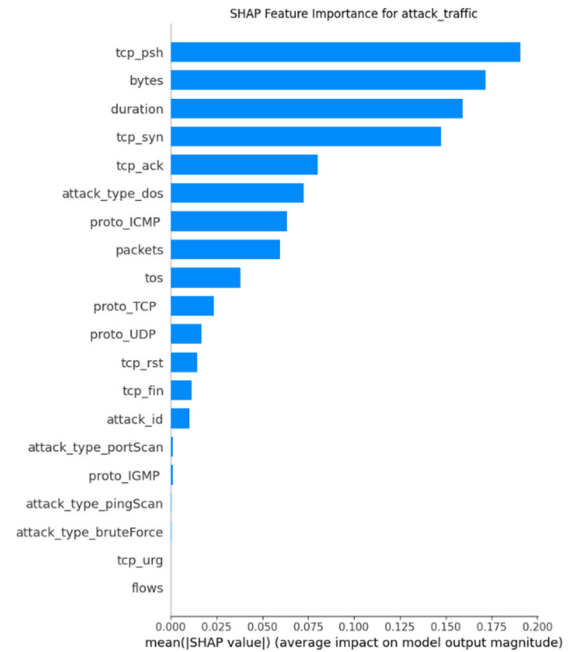


Fig. 7. shap summary bar attack traffic

The SHAP analysis identifies salient features of machine learning models for identifying normal network traffic, highlighting the importance of simple network behavior indicators and TCP flag indicators, with minimal contribution from attack features. It helps to comprehend normal traffic

features and improves prediction accuracy. Using the SHAP Feature Importance visualization tool, it evaluates the effect of features such as TCP Push Flag, data volume, and session duration, important for detecting attacks. Attackers usually use TCP Push Flag for transmitting data in denial-of-service attacks. Evaluating unique protocol features and fast packet transmission enhances detection, while less significant features also play a role in cyber analysis. Generally, the analysis guides improved intrusion detection systems against cyberattacks.

VIII. CONCLUSION

This report identifies the essential function of Explainable AI (XAI), particularly with the use of SHAP (Shapley Additive Explanations), in making machine learning models within Intrusion Detection Systems (IDS) more explainable. Through the use of SHAP, cybersecurity professionals have concise understanding of how such models determine decisions, which enhances confidence and incident detection and response—vital steps since cybersecurity errors are expensive. The research employs the CIDDS-001 dataset with a Random Forest Classifier and has a whopping 99.56 percent accuracy in separating normal from malicious traffic on challenging IoT and cloud environments, being effective against port scanning and DDoS attacks.

It also highlights that conventional IDS techniques are not keeping pace with the rapidly changing cyber threats of today and recommends that this XAI-augmented machine learningbased NIDS is a safer bet. Future development may concentrate on more intelligent feature selection, class imbalance correction, and accelerating real-time performance, perhaps by experimentation with varied datasets and sophisticated methods. For businesses, these results are a treasure trove for planning or updating IDS, allowing security teams to make intelligent, swift decisions as the digital landscape grows. The combination of high accuracy and simple explanation places this NIDS as one of the best contenders in battling cyber attacks and making networks secure.

REFERENCES

- [1] B. Lutevich, "What is an intrusion detection system (IDS)? Definition from searchsecurity," Oct 2021. [Online]. Available: <https://www.techtarget.com/searchsecurity/definition/intrusion-detection-system>.
- [2] A. Taha, A. Erbad, and M. Guizani, "A survey on machine learning-based approaches for internet traffic classification," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 472–511, 2021.
- [3] V. Trevisan, "Using shap values to explain how your machine learning model works," Jul 2022. [Online]. Available: <https://towardsdatascience.com/using-shap-values-to-explain-how-your-machine-learning-model-works-723db3b4c137>.
- [4] M. Ring, S. Wunderlich, D. Gradl, D. Landes, and A. Hotho, "Flow- based benchmark data sets for intrusion detection," in *Proceedings of the 16th European conference on cyber warfare and security*. ACPI, 2017, pp. 361–369.
- [5] T.-T.-H. Le, H. Kim, H. Kang, and H. Kim, "Classification and explanation for intrusion detection systems based on ensemble trees and shap method," *Sensors*, vol. 22, no. 3, p. 1154, 2022. (2002) The IEEE website. [Online]. Available: <http://www.ieee.org/>
- [6] J. R. Quinlan, C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning). San Mateo, CA, USA: Morgan Kaufmann, 1993. *FLEXChip Signal Processor (MC68175/D)*, Motorola, 1996.
- [7] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge, U.K.: Cambridge Univ. Press, 2000.
- [8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
- [9] A. Borkar, A. Donode, and A. Kumari, "A survey on intrusion detection system (IDS) and internal intrusion detection and protection system (IIDPS)," in *Proc. Int. Conf. Inventive Comput. Informat. (ICICI)*, Nov. 2017, pp. 949–953, doi: 10.1109/ICICI.2017.8365277..
- [10] Z. Zhou, C. Zhongwen, Z. Tiecheng, and G. Xiaohui, "The study on network intrusion detection system of snort," in *Proc. Int. Conf. Netw. Digit. Soc.*, May 2010, pp. 194–196, doi: 10.1109/ICNDS.2010.5479341
- [11] M. F. Zolkipli and A. Jantan, "A framework for malware detection using combination technique and signature generation," in *Proc. 2nd Int. Conf. Comput. Res. Develop.*, May 2010, pp. 196–199, doi: 10.1109/ICCRD.2010.25
- [12] V. Gupta, M. Singh, and V. K. Bhalla, "Pattern matching algorithms for intrusion detection and prevention system: A comparative analysis," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2014, pp. 50–54, doi: 10.1109/ICACCI.2014.6968595..