Journal for Educators, Teachers and Trainers
The PROFESIOLab electronic, peer-reviewed, open-access Magazine

ISSN 1989-9572

DOI:10.47750/jett.2023.14.01.074

SMART CYBERSECURITY STRATEGIES: MACHINE LEARNING FOR IOT ATTACK IDENTIFICATION AND PREVENTION

Akavaram Swapna[1],
Nanduri Shankar[1]
,Potabattula Rambabu[1]

Journal for Educators, Teachers and Trainers, Vol.14(1)

https://jett.labosfor.com/

Date of reception: 2 Dec 2022

Date of revision: 1 Jan 2023

Date of acceptance: 30 Jan 2023

Akavaram Swapna, Nanduri Shankar, Potabattula Rambabu. (2023). SMART CYBERSECURITY STRATEGIES: MACHINE LEARNING FOR IOT ATTACK IDENTIFICATION AND PREVENTION. Journal for Educators, Teachers and Trainers, Vol.14(1).815-824.

Journal for Educators, Teachers and Trainers
The PROFESIOLab electronic, peer-reviewed, open-access Magazine

Journal for Educators, Teachers and Trainers, Vol. 14(1)

ISSN 1989 –9572

https://jett.labosfor.com/

SMART CYBERSECURITY STRATEGIES: MACHINE LEARNING FOR IOT ATTACK IDENTIFICATION AND PREVENTION

Akavaram Swapna[1],Nanduri Shankar[1],Potabattula Rambabu[1]
[1]Department of Computer Science Engineering,
[1]Sree Dattha Group of Institutions, Sheriguda, Hyderabad, Telangana

ABSTRACT

The pervasive integration of Internet of Things (IoT) devices in today's networked environment has introduced several conveniences and opportunities. This technological revolution has also introduced a new category of cyber risks, as attackers use weaknesses in IoT devices to undermine user privacy, disrupt essential services, and cause chaos. Conventional security methods have demonstrated insufficiency in addressing the increasing complexity of cyber-attacks, requiring a more sophisticated and adaptable strategy. This urgency has prompted the creation of a Machine Learning Model for Cyber Attack Detection and Classification in IoT Environments (ML-IoT-CD). The necessity for a solid cybersecurity solution in IoT environments has grown essential due to the growing dependence on these devices for vital applications. Current intrusion detection systems and traditional security measures frequently lack the scalability and agility required to adapt to swiftly advancing attack methodologies. Consequently, there is an urgent need for an intelligent, automated, and proactive cyber security system that can identify and classify developing cyber threats in real time. The ML-IoT-CD paradigm seeks to address this requirement by utilizing machine learning techniques to examine extensive data produced by IoT devices. This approach may efficiently differentiate between legal and harmful activity, therefore enhancing the security posture of IoT networks.

1. INTRODUCTION

1.1 Overview

The general idea of the Internet of Things (IoT) is to allow for communication between human-to-thing or thing-to-thing(s). Things denote sensors or devices, whilst human or an object is an entity that can request or deliver a service [1]. The interconnection amongst the entities is always complex. IoT is broadly acceptable and implemented in various domains, such as healthcare, smart home, and agriculture. However, IoT has a resource constraint and heterogeneous environments, such as low computational power and memory. These constraints create problems in providing and implementing a security solution in IoT devices. These constraints further escalate the existing challenges for IoT environment. Therefore, various kinds of attacks are possible due to the vulnerability of IoT devices.

IoT-based botnet attack is one of the most popular, spreads faster and create more impact than other attacks. In recent years, several works have been conducted to detect and avoid this kind of attacks [2]–[3] by using novel approaches. Hence, a plethora of relevant of relevant models, methods, and etc. have been introduced over the past few years, with quite a reasonable number of studies reported in the research domain.

Many studies are trying to protect against these botnet attacks on the IoT environment. However, there are many gaps still existing to develop an effective detection mechanism. An intrusion detection system (IDS) is one of the efficient ways to deal with attacks. However, the traditional IDSs are often not able to be deployed for the

IoT environments due to the resource constraint problem of these devices. The complex cryptographic mechanisms cannot be embedded in many IoT devices either for the same reason. There are mainly two kinds of IDSs: the anomaly and misuse approaches. The misuse-based, also called the signature-based, approach, is based on the attacks' signatures, and they can also be found in most public IDSs, specifically Suricata [4]. Formally, the attacker can easily circumvent the signature-based approaches, and these mechanisms cannot guarantee to detect the unknown attacks and the variances of known attacks. The anomaly-based systems are based on normal data and can support to identify the unknown attacks. However, the different nature of IoT devices is being faced with the difficulty of collecting common normal data. The machine learning-based detection can guarantee detection of not only the known attacks and their variances. Therefore, we proposed a machine learning-based botnet attack detection architecture. We also adopted a feature selection method to reduce the demand for processing resources for performing the detection system on resource constraint devices. The experiment results indicate that the detection accuracy of our proposed system is high enough to detect the botnet attacks. Moreover, it can support the extension for detecting the new distinct kinds of attacks.

1.2. Motivation

Detecting and preventing attacks in IoT sensor data is a crucial and rewarding endeavor. IoT devices are increasingly integrated into critical infrastructure, such as power grids and healthcare systems. Ensuring their security is vital for public safety. The cyber-attacks can result in significant financial losses for individuals, businesses, and governments. Developing effective attack detection mechanisms can reduce these losses and stabilize economies.

Many IoT devices collect sensitive personal data. Detecting attacks helps protect user privacy and maintain trust in IoT technologies. Securing IoT devices fosters innovation and growth in the industry, making IoT technologies safer for deployment.IoT security expertise is in high demand, offering lucrative job opportunities in the dynamic field of cybersecurity.IoT security is a global challenge that requires collaboration and innovation to protect IoT ecosystems against cyber threats.Work in Cyber-attack detection can drive advancements in cybersecurity, machine learning, data analytics, and network security. Tackling complex problems leads to personal and intellectual growth, as you learn and adapt to new challenges.Ultimately, work in Cyber-attack detection contributes to a safer and more secure future, where IoT technology can be used without the fear of malicious attacks.

1.3 Problem Statement

The increasing proliferation of IoTdevices has led to a pressing need for robust attack detection mechanisms to safeguard the integrity, confidentiality, and availability of data generated and transmitted by these devices. The problem at hand is to develop and implement efficient and effective Cyber-attack detection solutions that can identify and mitigate a wide range of cyber threats targeting IoT sensor data.

The rapid growth of IoT devices across various domains, including critical infrastructure, healthcare, smart homes, and industrial applications, has created an extensive attack surface susceptible to cyber threats. IoT sensor data is vulnerable to various types of attacks, including but not limited to malware infections, DDoS (Distributed Denial of Service) attacks, data breaches, and physical tampering. These threats can lead to data compromise, service disruptions, and privacy breaches. Protecting the integrity and confidentiality of sensor data is crucial, as it often includes sensitive information. Unauthorized access or tampering with this data can have severe consequences.

IoT environments pose unique challenges for attack detection due to their diverse and resource-constrained nature. Traditional security measures may not be directly applicable. Many IoT devices have limited computational power, memory, and bandwidth, making it challenging to implement resource-intensive security solutions. The need for real-time or near-real-time attack detection is critical, as prompt responses are essential to prevent or mitigate the impact of attacks on IoT ecosystems. IoT deployments can range from a few devices to thousands or even millions. Solutions must be scalable to handle large-scale IoT deployments.

IoT devices often come from different manufacturers and use various communication protocols. Attack detection solutions should be able to operate in heterogeneous environments.Balancing security with privacy is a concern in IoT environments, as data collection can involve personal or sensitive information. Solutions must respect privacy regulations. The goal is to develop a holistic approach to Cyber-attack detection that covers a wide spectrum of potential threats and vulnerabilities.

2. LITERATURE SURVEY

Soe et al. [5] adopted a lightweight detection system with a high performance. The overall detection performance achieves around 99% for the botnet attack detection using three different ML algorithms, including artificial neural network (ANN), J48 decision tree, and Naïve Bayes. The experiment result indicated that the proposed architecture can effectively detect botnet-based attacks, and also can be extended with corresponding sub-engines for new kinds of attacks.

Ali et al. [6] outlined the existing proposed contributions, datasets utilised, network forensic methods utilised and research focus of the primary selected studies. The demographic characteristics of primary studies were also outlined. The result of this review revealed that research in this domain is gaining momentum, particularly in the last 3 years (2018-2020). Nine key contributions were also identified, with Evaluation, System, and Model being the most conducted.

Irfan et al. [7] classified the incoming data in the IoT, contain a malware or not. In this research, this work under sample the dataset because the datasets contain imbalance class. After that, this work classified the sample using Random Forest. This work used Naive Bayes, K-Nearest Neighbor and Decision Tree too as a comparison. The dataset that has been used in this research are from UCI Machine Learning Depository's Website. The dataset showed the data traffic from the IoT Device in a normal condition and attacked by Mirai or Bashlite.

Shah et al. [8] presented a concept called ‘login puzzle’ to prevent capture of IoT devices in a large scale. Login puzzle is a variant of client puzzle, which presented a puzzle to the remote device during the login process to prevent unrestricted log-in attempts. Login puzzle is a set of multiple mini puzzles with a variable complexity, which the remote device is required to solve before logging into any IoT device. Every unsuccessful log-in attempt increases the complexity of solving the login puzzle for the next attempt. This paper introduced a novel mechanism to change the complexity of puzzle after every unsuccessful login attempt. If each IoT device had used login puzzle, Mirai attack would have required almost two months to acquire devices, while it acquired them in 20 h.

Tzagkarakis et al. [9] presented an IoT botnet attack detection method based on a sparsity representation framework using a reconstruction error thresholding rule for identifying malicious network traffic at the IoT edge coming from compromised IoT devices. The botnet attack detection is performed based on small-sized benign IoT network traffic data, and thus we have no prior knowledge about malicious IoT traffic data. We present our results on a real IoT-based network dataset and show the efficacy of proposed technique against a reconstruction error-based autoencoder approach.

Meidan et al. [10] proposed a novel network-based anomaly detection method for the IoT called N-BaIoT that extracts behavior snapshots of the network and uses deep autoencoders to detect anomalous network traffic from compromised IoT devices. To evaluate the method, this work infected nine commercial IoT devices in our lab with two widely known IoT-based botnets, Mirai and BASHLITE. The evaluation results demonstrated the proposed methods ability to detect the attacks accurately and instantly as they were being launched from the compromised IoT devices that were part of a botnet.

Popoola et al. [11] proposed the federated DL (FDL) method for zero-day botnet attack detection to avoid data privacy leakage in IoT-edge devices. In this method, an optimal deep neural network (DNN) architecture is employed for network traffic classification. A model parameter server remotely coordinates the independent training of the DNN models in multiple IoT-edge devices, while the federated averaging (FedAvg) algorithm is used to aggregate local model updates. A global DNN model is produced after several communication rounds between the model parameter server and the IoT-edge devices. The zero-day botnet attack scenarios in IoT-edge devices are simulated with the Bot-IoT and N-BaIoT data sets.

Hussain et al. [12] produced a generic scanning and DDoS attack dataset by generating 33 types of scans and 60 types of DDoS attacks. In addition, this work partially integrated the scan and DDoS attack samples from three publicly available datasets for maximum attack coverage to better train the machine learning algorithms. Afterwards, this work proposed a two-fold machine learning approach to prevent and detect IoT botnet attacks. In the first fold, this work trained a state-of-the-art deep learning model, i.e., ResNet-18 to detect the scanning activity in the premature attack stage to prevent IoT botnet attacks. While, in the second fold, this work trained another ResNet-18 model for DDoS attack identification to detect IoT botnet attacks.

Abu et al. [13] proposed an ensemble learning model for botnet attack detection in IoT networks called ELBA-IoT that profiles behavior features of IoT networks and uses ensemble learning to identify anomalous network traffic from compromised IoT devices. In addition, this IoT-based botnet detection approach characterizes the evaluation of three different machine learning techniques that belong to decision tree techniques (AdaBoosted, RUSBoosted, and bagged). To evaluate ELBA-IoT, we used the N-BaIoT-2021 dataset, which comprises records of both normal IoT network traffic and botnet attack traffic of infected IoT devices.

Alharbi et al. [14] proposed Gaussian distribution used in the population initialization. Furthermore, the local search mechanism was followed by the Gaussian density function and local-global best function to achieve better exploration during each generation. Enhanced BA was further employed for neural network hyperparameter tuning and weight optimization to classify ten different botnet attacks with an additional one benign target class. The proposed LGBA-NN algorithm was tested on an N-BaIoT data set with extensive real traffic data with benign and malicious target classes. The performance of LGBA-NN was compared with several recent advanced approaches such as weight optimization using Particle Swarm Optimization (PSO-NN) and BA-NN.

Ahmed et al. [15] proposed a model for detecting botnets using deep learning to identify zero-day botnet attacks in real time. The proposed model is trained and evaluated on a CTU-13 dataset with multiple neural network designs and hidden layers. Results demonstrated that the deep-learning artificial neural network model can accurately and efficiently identify botnets.

3. PROPOSED SYSTEM

3.1 Overview

Detecting cyberattacks using a combination of data preprocessing techniques, such as Standard Scaling, and aRFC classifier is a common approach in cybersecurity. Figure 4.1 shows a cyber-attack attack detection system model.

Step 1: Preprocessing: Gather a dataset of network traffic or system logs, where each data point is labeled as either a normal activity or a cyber-attack.Preprocess the data to make it suitable for training aRFC classifier. This may include handling missing values, encoding categorical variables, and scaling numerical features.

Step 2: Standard Scaling: Extract relevant features from the dataset. Common features for cyber-attack detection may include network traffic statistics, log event patterns, and more. Feature selection or dimensionality reduction techniques can be applied if the dataset has many features. Use Standard Scaling to standardize the numerical features in the dataset. Standard Scaling is preferred over standard scaling when dealing with data that may have outliers.Standard Scaling scales the features in a way that is less affected by extreme values, making it a suitable choice for cybersecurity datasets.

Step 3: Split the Data: Divide your dataset into training, validation, and test sets. A common split might be 80% for training, and20% for testing.

Step 4: RFC Classifier: Design and build anRFC classifier.

Step 5: Training: Train the RFC classifier using the training dataset. During training, monitor performance on the validation set to avoid overfitting and adjust hyperparameters accordingly.

Step 6: Evaluation: Evaluate the trained model on the test dataset to assess its performance. Common evaluation metrics for cyber-attack detection include accuracy, precision, recall, F1-score, and confusion matrix.

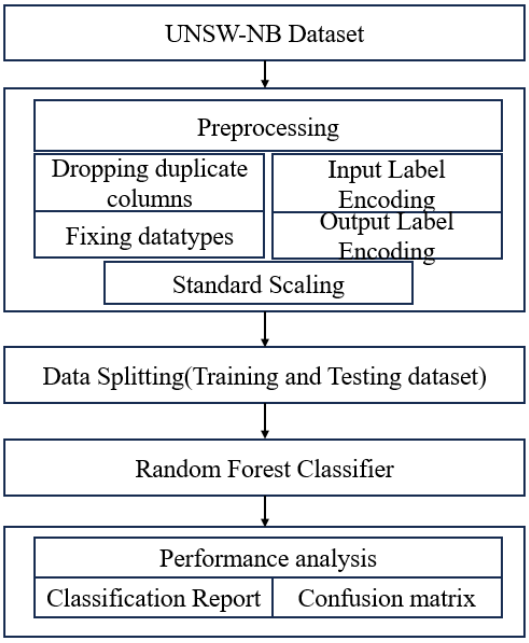


Fig. 3.1: Block diagram of proposed system.

Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

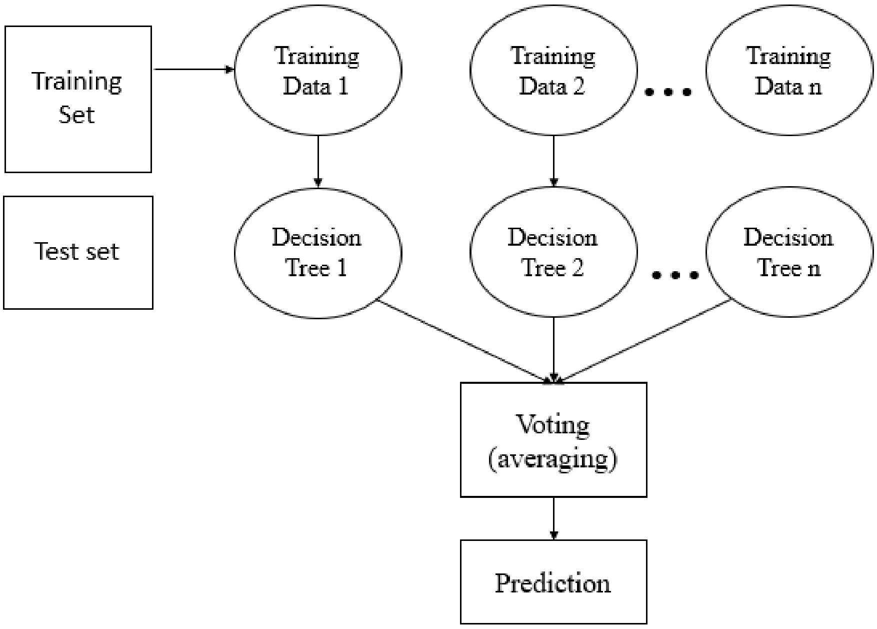


Fig. 3.2: Random Forest algorithm.

Step 1: In Random Forest n number of random records are taken from the data set having k number of records.
Step 2: Individual decision trees are constructed for each sample.
Step 3: Each decision tree will generate an output.
Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

Important Features of Random Forest

- **Diversity**- Not all attributes/variables/features are considered while making an individual tree, each tree is different.
- **Immunetothecurseofdimensionality**- Since each tree does not consider all the features, the feature space is reduced.
- **Parallelization**-Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build random forests.
- **Train-Testsplit**- In a random forest we don't have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
- **Stability**- Stability arises because the result is based on majority voting/ averaging.

Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Below are some points that explain why we should use the Random Forest algorithm

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

Types of Ensembles

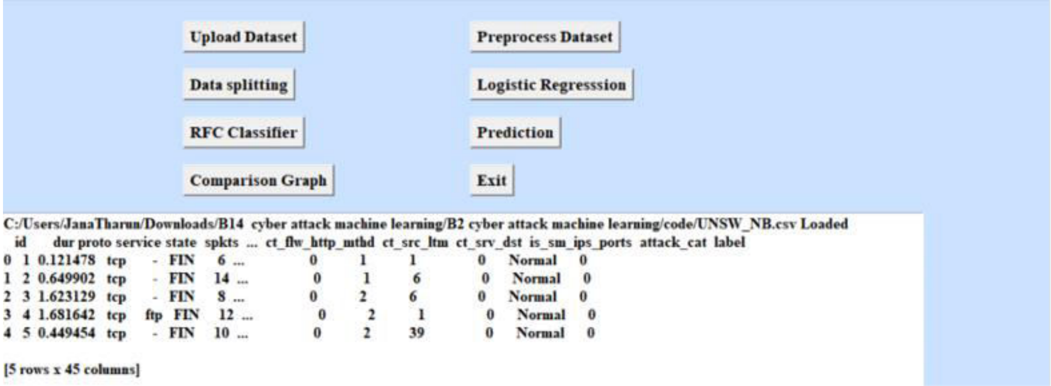
Before understanding the working of the random forest, we must look into the ensemble technique. Ensemble simply means combining multiple models. Thus, a collection of models is used to make predictions rather than an individual model. Ensemble uses two types of methods:

Bagging– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest. Bagging, also known as Bootstrap Aggregation is the ensemble technique used by random forest. Bagging chooses a random sample from the data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as row sampling. This step of row sampling with replacement is called bootstrap. Now each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation.

Boosting– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.

4.RESULTS AND DESCRIPTION

Fig 1: This figure showcases the process of uploading a dataset within the GUI interface. Users interact with the graphical interface to select and load the dataset into the system, initiating further processing and analysis. Fig2: After preprocessing the dataset, this figure presents a count plot visualizing the distribution of data across different classes or categories. The count plot provides insights into the balance of classes within the dataset, aiding in understanding the data's composition. Fig 3: Illustrating the process of splitting the dataset into training and testing subsets, this figure depicts how the data is divided to facilitate model training and evaluation. The dataset is partitioned into separate sets for training the model and testing its performance. Fig 4: This figure displays the confusion matrix generated for the logistic regression model. The confusion matrix provides a detailed breakdown of the model's classification performance, showing the number of true positive, true negative, false positive, and false negative predictions.



```
C:/Users/JanaTharun/Downloads/B14  cyber attack machine learning/B2 cyber attack machine learning/code/UNSW_NB.csv Loaded
   id      dur proto service state  spkts  ...  ct_flw_http_mthd  ct_src_ltm  ct_srv_dst  is_sm_ips_ports  attack_cat  label
0  1  0.121478   tcp       -   FIN      6  ...                 0           1           1                0      Normal      0
1  2  0.649902   tcp       -   FIN     14  ...                 0           1           6                0      Normal      0
2  3  1.623129   tcp       -   FIN      8  ...                 0           2           6                0      Normal      0
3  4  1.681642   tcp     ftp   FIN     12  ...                 0           2           1                0      Normal      0
4  5  0.449454   tcp       -   FIN     10  ...                 0           2          39                0      Normal      0

[5 rows x 45 columns]
```

Fig 1: Uploading dataset in the GUI Interface.

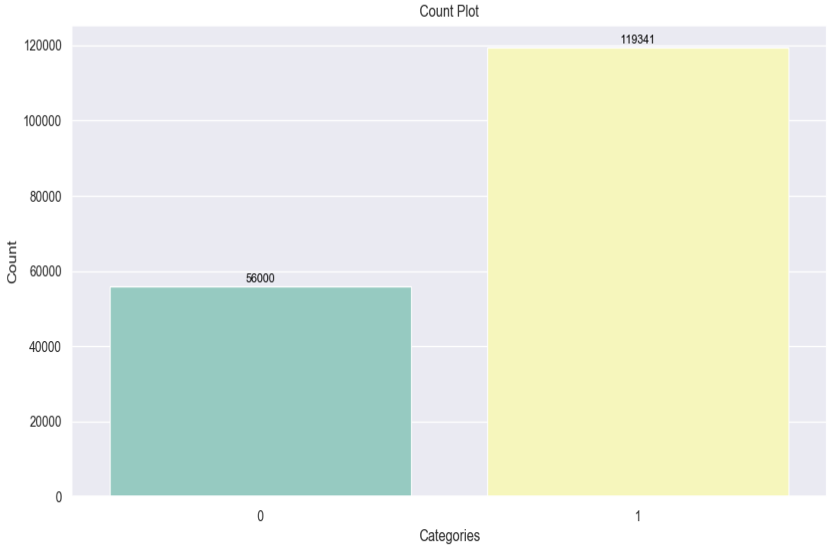
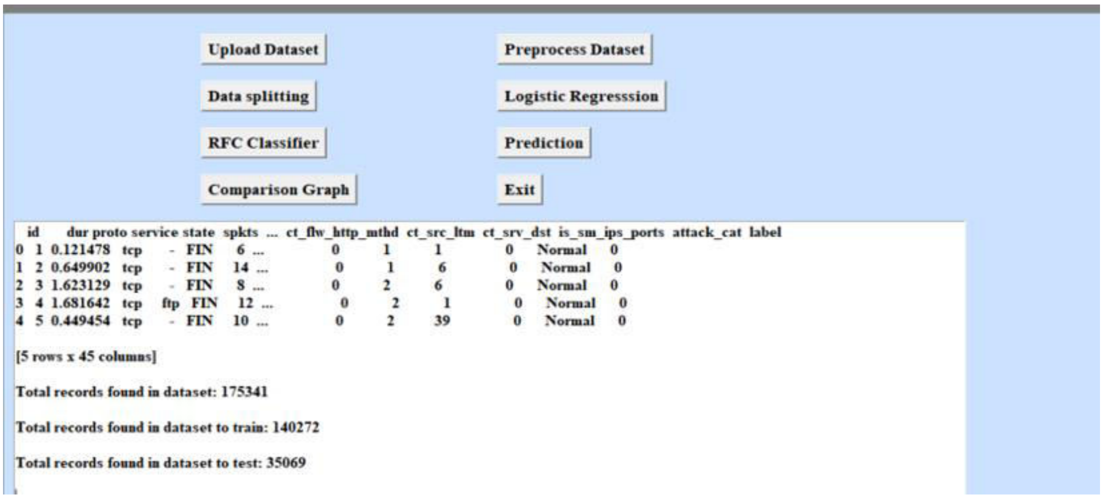


Fig2: Count plot after preprocessing dataset



```
  id     dur proto service state  spkts  ...  ct_flw_http_mthd  ct_src_ltm  ct_srv_dst  is_sm_ips_ports  attack_cat  label
0  1  0.121478   tcp       -   FIN      6  ...                 0           1           1                0      Normal      0
1  2  0.649902   tcp       -   FIN     14  ...                 0           1           6                0      Normal      0
2  3  1.623129   tcp       -   FIN      8  ...                 0           2           6                0      Normal      0
3  4  1.681642   tcp     ftp   FIN     12  ...                 0           2           1                0      Normal      0
4  5  0.449454   tcp       -   FIN     10  ...                 0           2          39                0      Normal      0

[5 rows x 45 columns]

Total records found in dataset: 175341

Total records found in dataset to train: 140272

Total records found in dataset to test: 35069
```

Fig 3: Splitting the dataset for Model Training and Testing.

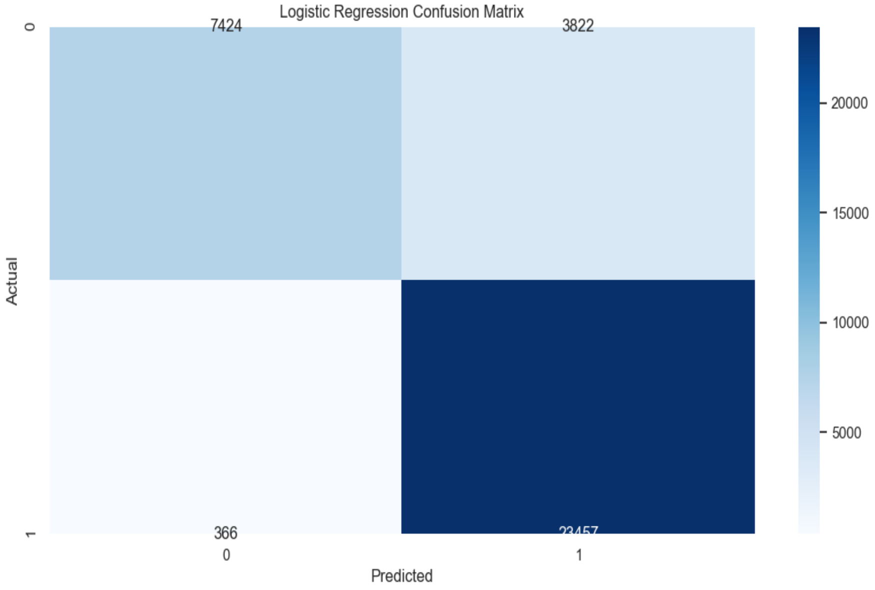


Fig 4: confusion matrix of logistic Regression

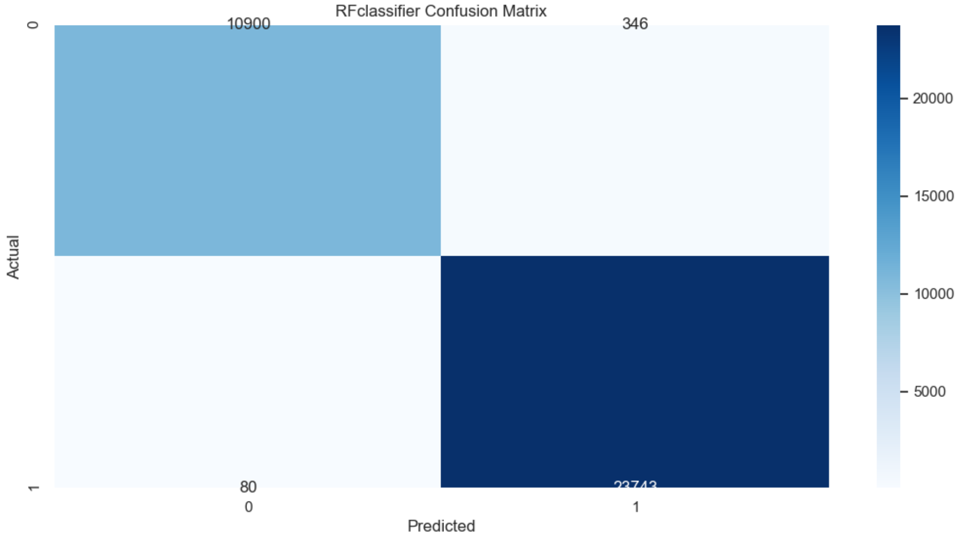
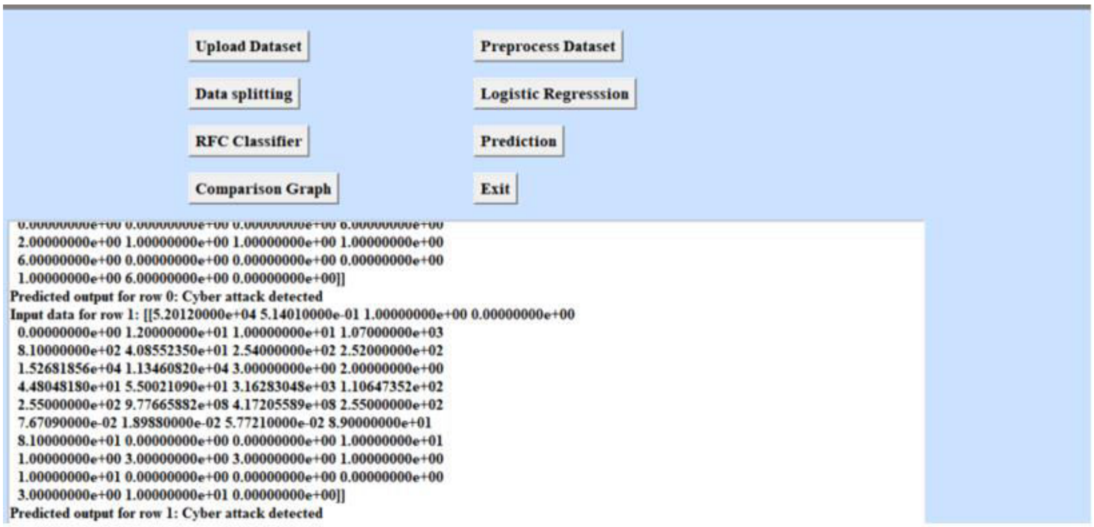


Fig 5: Confusion Matrix of Proposed Random Forest Classifier.



```
 0.00000000e+00 0.00000000e+00 0.00000000e+00 6.00000000e+00
 2.00000000e+00 1.00000000e+00 1.00000000e+00 1.00000000e+00
 6.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00
 1.00000000e+00 6.00000000e+00 0.00000000e+00]]
Predicted output for row 0: Cyber attack detected
Input data for row 1: [[5.20120000e+04 5.14010000e-01 1.00000000e+00 0.00000000e+00
 0.00000000e+00 1.20000000e+01 1.00000000e+01 1.07000000e+03
 8.10000000e+02 4.08552350e+01 2.54000000e+02 2.52000000e+02
 1.52681856e+04 1.13460820e+04 3.00000000e+00 2.00000000e+00
 4.48048180e+01 5.50021090e+01 3.16283048e+03 1.10647352e+02
 2.55000000e+02 9.77665882e+08 4.17205589e+08 2.55000000e+02
 7.67090000e-02 1.89880000e-02 5.77210000e-02 8.90000000e+01
 8.10000000e+01 0.00000000e+00 0.00000000e+00 1.00000000e+01
 1.00000000e+00 3.00000000e+00 3.00000000e+00 1.00000000e+00
 1.00000000e+01 0.00000000e+00 0.00000000e+00 0.00000000e+00
 3.00000000e+00 1.00000000e+01 0.00000000e+00]]
Predicted output for row 1: Cyber attack detected
```

Fig 6:Model Prediction on Test Data.

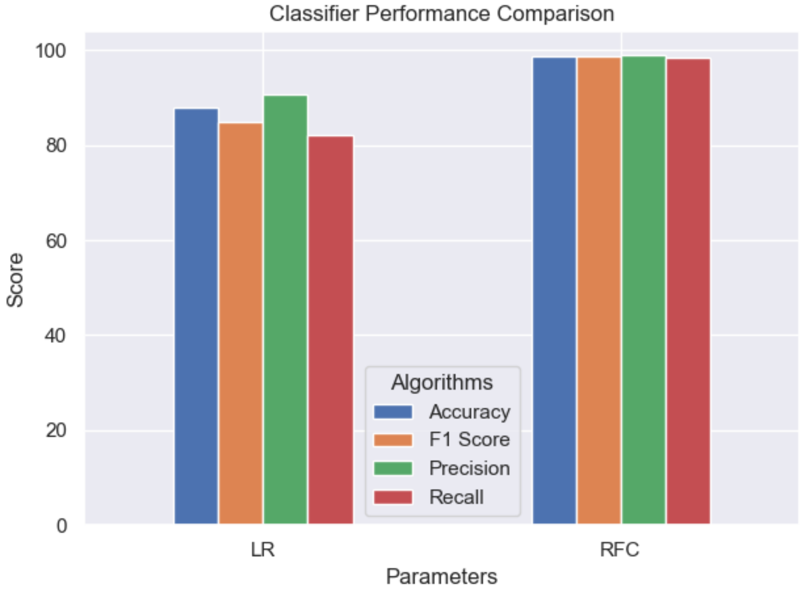


Fig 6: Performance Comparison Graph of LR and RFC Models.

Fig 5: Presented here is the confusion matrix for the proposed Random Forest Classifier model. Similar to Fig 4, this matrix offers insights into the classification performance of the Random Forest model, enabling assessment of its accuracy and efficacy. Fig 6: This figure showcases the model's predictions on the test data. Users can observe the model's predicted outcomes compared to the actual labels, evaluating its performance in real-world scenarios. Fig 7: The performance comparison graph of the Logistic Regression (LR) and Random Forest Classifier (RFC) models is depicted here. This graph provides a visual representation of the models' performance metrics, aiding in the comparison and selection of the most suitable model for the task at hand.

5. CONCLUSION

The approach of using standard scaling and aRFC classifier for cyber-attack detection is a promising one. It leverages advanced machine learning techniques to identify malicious activities in network traffic or system logs. By preprocessing the data effectively and training a RFC model, it is possible to achieve accurate and timely detection of cyber threats. However, it's important to note that the effectiveness of such a system depends on various factors, including the quality and diversity of the training data, the design of the RFC architecture, and the continuous monitoring and updating of the model.

REFERENCES

[1] S. Dange and M. Chatterjee, "Iot botnet: The largest threat to the iot network" in Data Communication and Networks, Cham, Switzerland:Springer, pp. 137-157, 2020.

[2] J. Ceron, K. Steding-Jessen, C. Hoepers, L. Granville and C. Margi, "Improving IoT botnet investigation using an adaptive network layer", Sensors, vol. 19, no. 3, pp. 727, Feb. 2019.

[3] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, et al., "N-baiot-network-based detection of iot botnet attacks using deep autoencoders", IEEE Pervas. Comput., vol. 17, no. 3, pp. 12-22, 2018.

[4] Shah, S.A.R.; Issac, B. Performance comparison of intrusion detection systems and application of machine learning to Snort system. Futur. Gener. Comput. Syst. 2018, 80, 157–170.

[5] Soe YN, Feng Y, Santosa PI, Hartanto R, Sakurai K. Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture. Sensors. 2020; 20(16):4372. https://doi.org/10.3390/s20164372

[6] I. Ali et al., "Systematic Literature Review on IoT-Based Botnet Attack," in IEEE Access, vol. 8, pp. 212220-212232, 2020, doi: 10.1109/ACCESS.2020.3039985.

[7] Irfan, I. M. Wildani and I. N. Yulita, "Classifying botnet attack on Internet of Things device using random forest", IOP Conf. Ser. Earth Environ. Sci., vol. 248, Apr. 2019.

[8] Shah, T., Venkatesan, S. (2019). A Method to Secure IoT Devices Against Botnet Attacks. In: Issarny, V., Palanisamy, B., Zhang, LJ. (eds) Internet of Things – ICIOT 2019. ICIOT 2019. Lecture Notes in Computer Science(), vol 11519. Springer, Cham. https://doi.org/10.1007/978-3-030-23357-0_3

[9] C. Tzagkarakis, N. Petroulakis and S. Ioannidis, "Botnet Attack Detection at the IoT Edge Based on Sparse Representation," 2019 Global IoT Summit (GIoTS), Aarhus, Denmark, 2019, pp. 1-6, doi: 10.1109/GIOTS.2019.8766388.

[10] Y. Meidan et al., "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders," in IEEE Pervasive Computing, vol. 17, no. 3, pp. 12-22, Jul.-Sep. 2018, doi: 10.1109/MPRV.2018.03367731.

[11] S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh and O. Jogunola, "Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT-Edge Devices," in IEEE Internet of Things Journal, vol. 9, no. 5, pp. 3930-3944, 1 March1, 2022, doi: 10.1109/JIOT.2021.3100755.

[12] F. Hussain et al., "A Two-Fold Machine Learning Approach to Prevent and Detect IoT Botnet Attacks," in IEEE Access, vol. 9, pp. 163412-163430, 2021, doi: 10.1109/ACCESS.2021.3131014.

[13] Abu Al-Haija Q, Al-Dala'ien M. ELBA-IoT: An Ensemble Learning Model for Botnet Attack Detection in IoT Networks. Journal of Sensor and Actuator Networks. 2022; 11(1):18. https://doi.org/10.3390/jsan11010018

[14] Alharbi A, Alosaimi W, Alyami H, Rauf HT, Damaševičius R. Botnet Attack Detection Using Local Global Best Bat Algorithm for Industrial Internet of Things. Electronics. 2021; 10(11):1341. https://doi.org/10.3390/electronics10111341

[15] Ahmed, A.A., Jabbar, W.A., Sadiq, A.S. et al. Deep learning-based classification model for botnet attack detection. J Ambient Intell Human Comput 13, 3457–3466 (2022). https://doi.org/10.1007/s12652-020-01848-9