iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

BOTNET ATTACK DETECTION IN SDN USING MACHINE LEARNING

Mrs. Varsha Kumari (Research Scholar)
Department of Information Technology and Computer Applications,
Andhra University College of Engineering(A), Visakhapatnam, Andhra Pradesh, India.

Kota Bharathi (MCA Student)
Department of Information Technology and Computer Applications,
Andhra University College of Engineering(A), Visakhapatnam, Andhra Pradesh, India.

Corresponding Authors:
Varsha Kumari (Research Scholar), Email: varsha162@gmail.com
Kota Bharathi (MCA Student), Email: bharathikota5@gmail.com

ABSTRACT
The rapid growth of internet-connected devices has increased the risk of cyber threats, with botnet attacks being one of the most significant security challenges. A botnet is a network of compromised devices controlled by attackers to perform malicious activities such as Distributed Denial of Service (DDoS) attacks, spam distribution, phishing, and data theft. Traditional security systems often struggle to detect new and evolving attack patterns, creating the need for intelligent and automated detection mechanisms. This paper presents a Machine Learning-based Botnet Attack Detection System that analyzes network traffic data and classifies it as either normal traffic or botnet traffic. The dataset is preprocessed through data cleaning, categorical feature encoding, feature scaling, feature selection, feature engineering, and an 80:20 train-test split. Random Forest and XGBoost are implemented and compared using Accuracy, Precision, Recall, F1-Score, ROC-AUC, classification reports, and confusion matrices. The experimental results show that both models effectively identify malicious traffic, while XGBoost achieves slightly better overall performance.

Keywords:
Botnet Detection, Machine Learning, Random Forest, XGBoost, Network Traffic, Cybersecurity, Intrusion Detection

INTRODUCTION

The increasing use of internet-connected devices has led to a rapid rise in cybersecurity threats. Among these threats, botnet attacks are particularly dangerous because compromised devices can be coordinated to perform malicious activities such as DDoS attacks, spam distribution, phishing, and data theft. Traditional security systems commonly depend on predefined rules and signatures and therefore face difficulty when attack patterns change or previously unseen behavior appears.

Machine Learning provides a data-driven approach for identifying suspicious network behavior. In the proposed system, labeled network traffic is processed and used to train classification models that distinguish normal traffic from botnet traffic. Random Forest and XGBoost are selected for comparative evaluation because both are ensemble-based learning techniques suitable for classification tasks.

The literature survey in the project considers commonly used machine learning approaches for network intrusion and botnet detection, including Decision Tree, Random Forest, Support Vector Machine, Naive Bayes, and K-Nearest Neighbors. The project particularly focuses on Random Forest and XGBoost and evaluates their effectiveness using a common test dataset.

OBJECTIVES

- To load and preprocess network traffic data containing normal and botnet records.
- To remove duplicate, missing, irrelevant, and redundant information and transform categorical features.
- To perform feature engineering using Time Gap, Bytes Per Packet, Packet Ratio, Byte Ratio, and Total Rate.
- To train Random Forest and XGBoost models using an 80:20 training-testing split.

 © Author(s) | CC BY 4.0 International License.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

- To evaluate the models using Accuracy, Precision, Recall, F1-Score, ROC-AUC, classification reports, and confusion matrices.
- To compare the two models and select the better-performing approach for botnet attack detection.

METHODOLOGY

Data Collection and Preprocessing

The network traffic dataset contains normal and botnet traffic records with communication-related attributes such as source and destination information, protocol type, packet count, byte count, transmission rate, traffic duration, and connection state. During preprocessing, duplicate records are removed, missing values are handled, and data consistency is checked. Non-useful identifiers and network-address fields such as Fid, saddr, daddr, and timestamp fields are excluded from model training. Categorical attributes including Protocol Type, Connection State, and Web Category are encoded into numerical form.

Feature Engineering and Dataset Split

Additional traffic features are generated to improve detection performance, including Time Gap, Bytes Per Packet, Packet Ratio, Byte Ratio, and Total Rate. Numerical features are normalized using StandardScaler. The processed dataset is divided into 80% training data and 20% testing data, with stratification applied to preserve class distribution.

Machine Learning Models

Random Forest creates multiple decision trees using randomized subsets of data and combines their predictions through majority voting. XGBoost builds decision trees sequentially and updates subsequent trees to correct errors made by earlier trees. Both models are trained on the same processed training data and evaluated on unseen testing data.

Evaluation Measures

Model performance is evaluated using Accuracy, Precision, Recall, F1-Score, and ROC-AUC. Confusion matrices and classification reports provide class-level information, while ROC curves show the relationship between true-positive and false-positive rates at different classification thresholds.

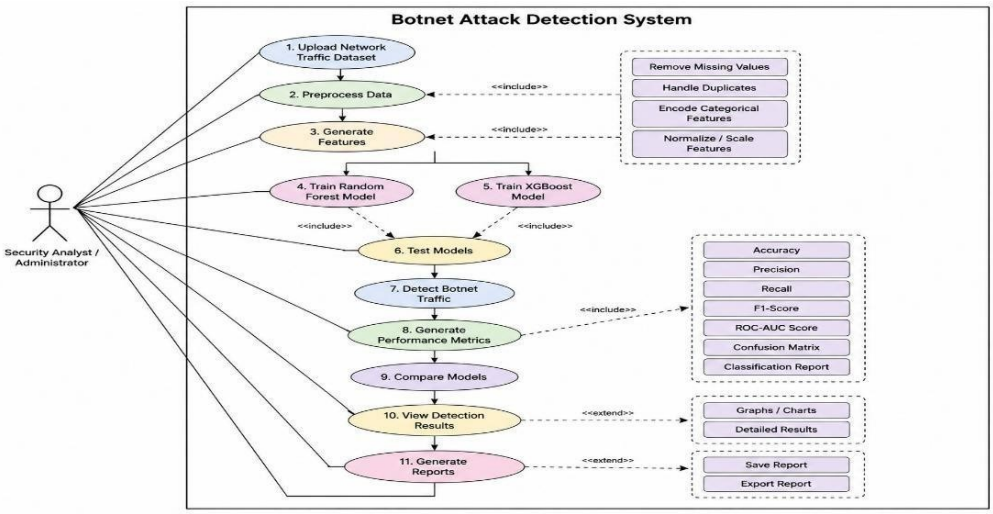


Fig. 1. Use Case Diagram

IMPLEMENTATION

The implementation is performed in Python using Pandas and NumPy for data processing, Scikit-Learn for preprocessing and Random Forest classification, XGBoost for gradient boosting classification, and Matplotlib/Seaborn for visualization. The dataset is loaded from Labeled_Dataset.csv. After removing the Label, Fid, saddr, and daddr fields as appropriate, categorical variables are encoded using pandas get_dummies. StandardScaler is fitted on the training data and then applied to the test data.

The Random Forest model uses 300 estimators, random_state=42, and parallel processing. The XGBoost model uses 300 estimators, max_depth=6, learning_rate=0.05, random_state=42, and logloss as the evaluation metric. Predictions and class probabilities are generated for the test data, followed by calculation of evaluation metrics and visualization of confusion matrices, classification reports, ROC curves, and model accuracy.

 © Author(s) | CC BY 4.0 International License.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

The processed datasets are saved as processed_X_train.csv, processed_X_test.csv, processed_y_train.csv, and processed_y_test.csv. Feature importance analysis is also performed using XGBoost to identify influential features and improve interpretability.

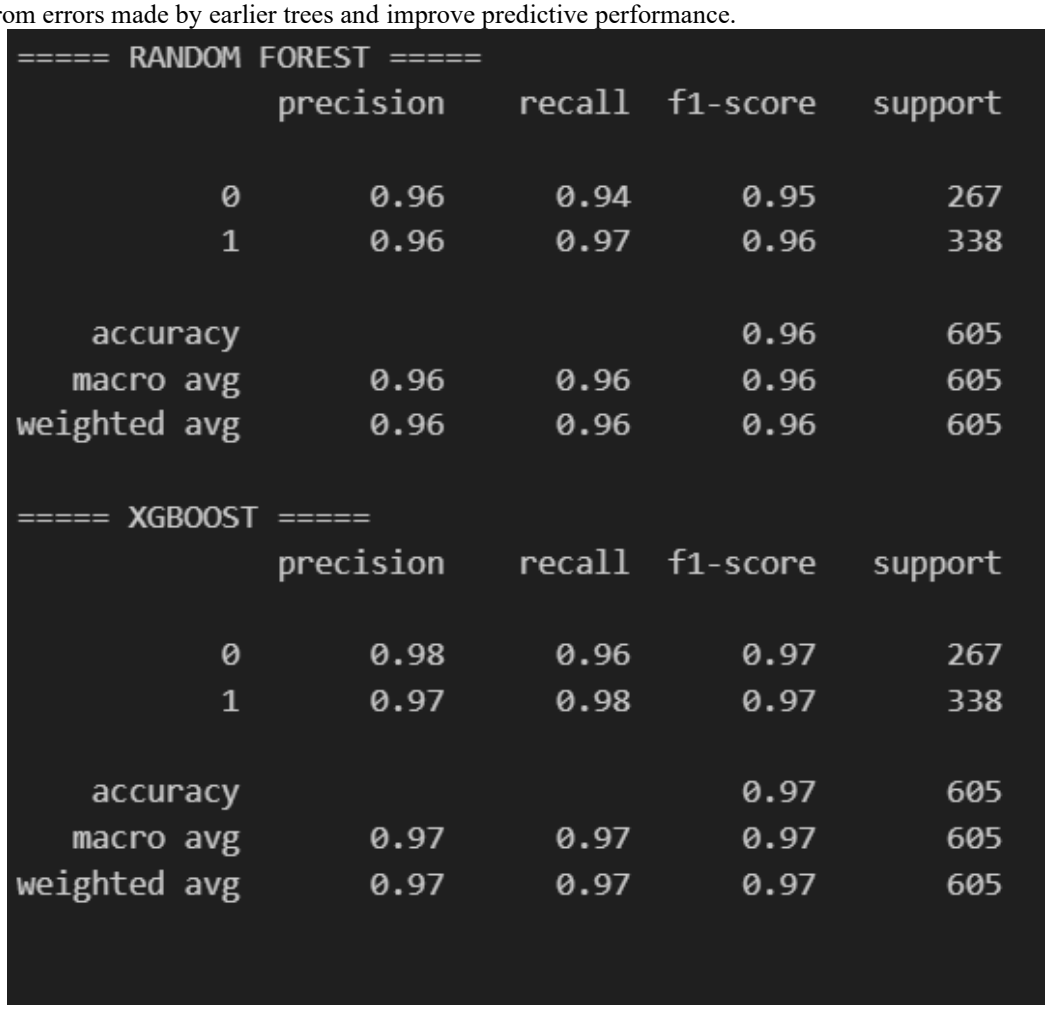
RESULTS AND DISCUSSION

The proposed system was successfully implemented and tested using network traffic data containing normal and botnet activities. Both Random Forest and XGBoost achieved high classification performance. The detailed values generated by the implementation are shown in Table 1.

Model	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*ROC-AUC*
Random Forest	0.958678	0.956268	0.970414	0.963289	0.980055
XGBoost	0.970248	0.965116	0.982249	0.973607	0.987922

Table 1. Model Performance Comparison

The comparison indicates that XGBoost outperformed Random Forest across Accuracy, Precision, Recall, F1-Score, and ROC-AUC. The improvement is consistent with the project observation that sequential boosting can learn from errors made by earlier trees and improve predictive performance.



```
===== RANDOM FOREST =====
              precision    recall  f1-score   support

           0       0.96      0.94      0.95       267
           1       0.96      0.97      0.96       338

    accuracy                           0.96       605
   macro avg       0.96      0.96      0.96       605
weighted avg       0.96      0.96      0.96       605

===== XGBOOST =====
              precision    recall  f1-score   support

           0       0.98      0.96      0.97       267
           1       0.97      0.98      0.97       338

    accuracy                           0.97       605
   macro avg       0.97      0.97      0.97       605
weighted avg       0.97      0.97      0.97       605
```

Fig. 2. Classification Report Comparison

The classification reports show balanced precision and recall for both normal and botnet classes. For Random Forest, the normal class has precision 0.96, recall 0.94, and F1-score 0.95, while the botnet class has precision 0.96, recall 0.97, and F1-score 0.96. XGBoost reports precision 0.98, recall 0.96, and F1-score 0.97 for normal traffic, and precision 0.97, recall 0.98, and F1-score 0.97 for botnet traffic.

 © Author(s) | CC BY 4.0 International License.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

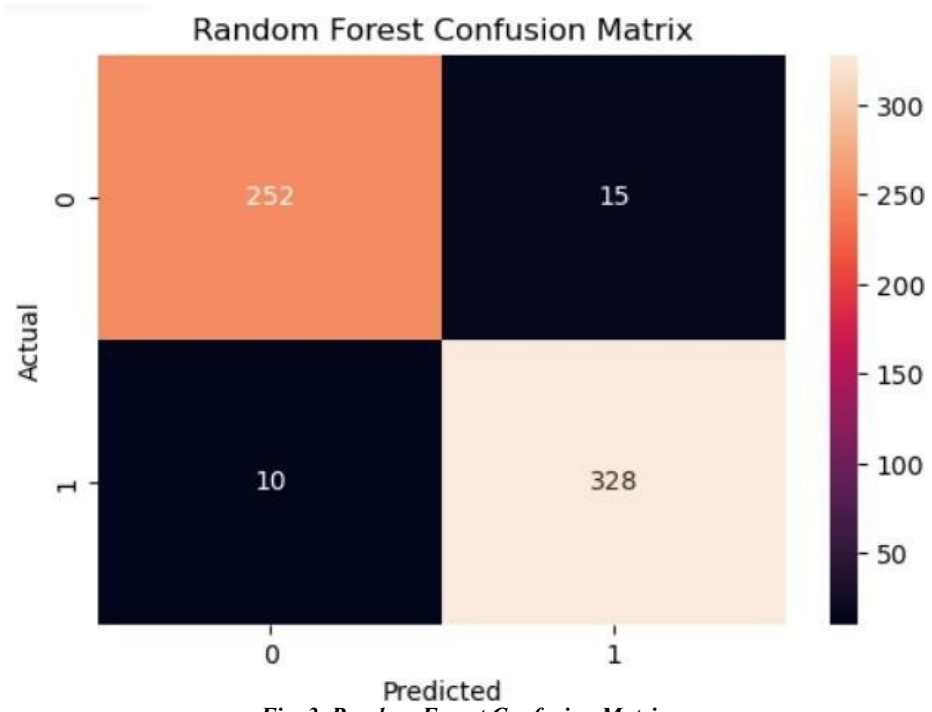


Fig. 3. Random Forest Confusion Matrix

The Random Forest confusion matrix contains 252 correctly classified normal records and 328 correctly classified botnet records. It also shows 15 false positives and 10 false negatives.

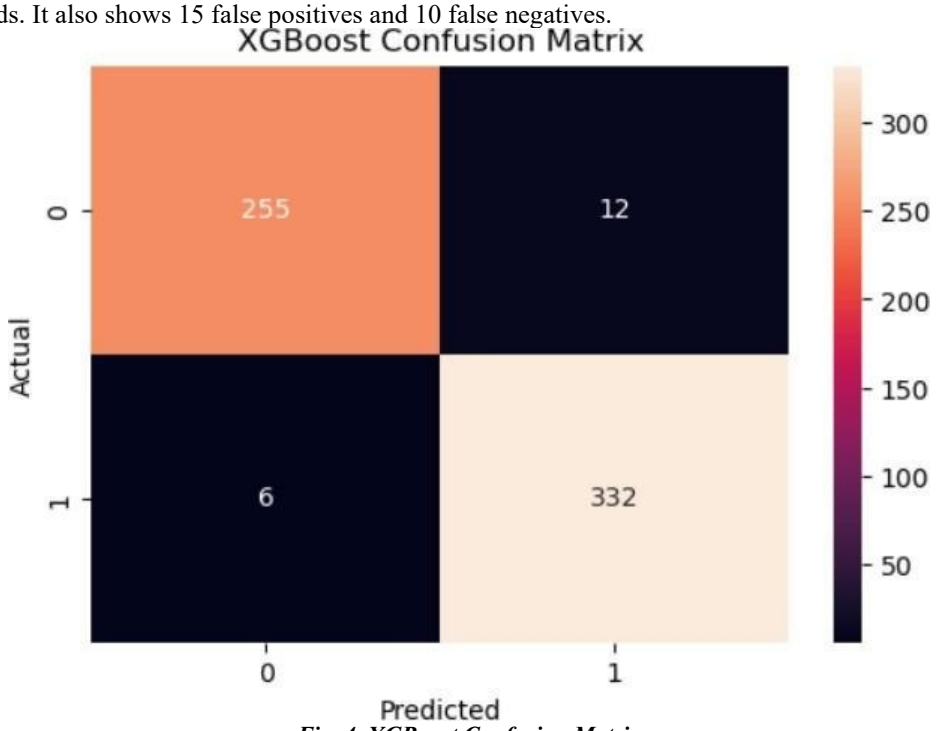


Fig. 4. XGBoost Confusion Matrix

The XGBoost confusion matrix contains 255 correctly classified normal records and 332 correctly classified botnet records, with 12 false positives and 6 false negatives. Thus, XGBoost produces fewer misclassifications than Random Forest on the evaluated test set.

 © Author(s) | CC BY 4.0 International License.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

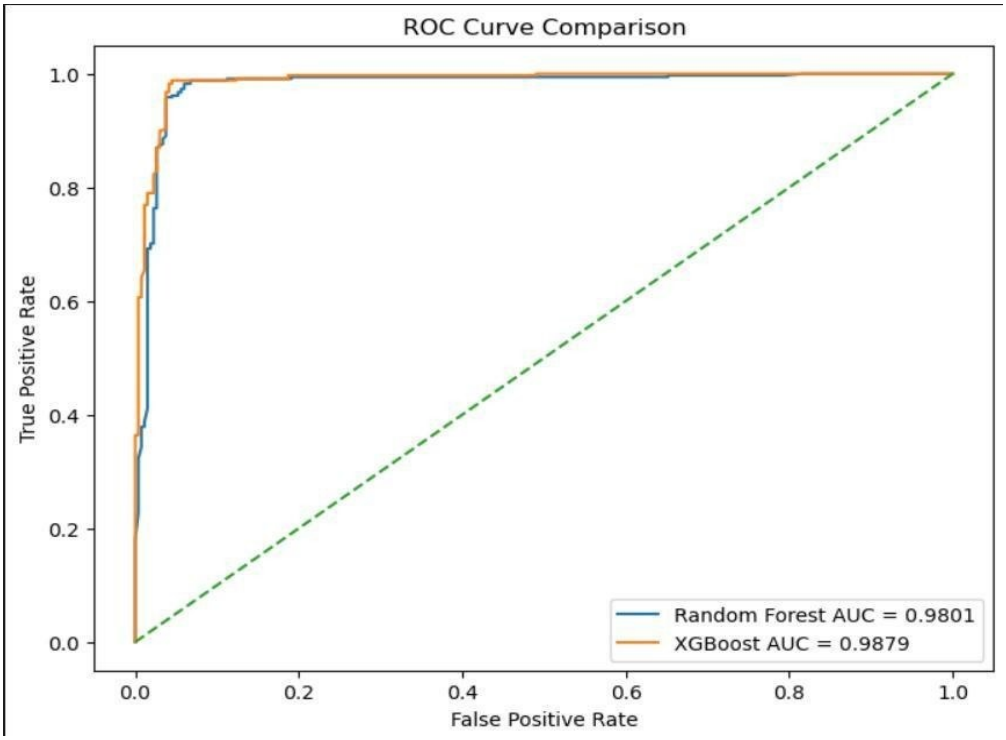


Fig. 5. ROC Curve Comparison

The ROC curves demonstrate strong discrimination between normal and botnet traffic. The generated AUC values are 0.980055 for Random Forest and 0.987922 for XGBoost, confirming the superior discrimination capability of XGBoost for the evaluated dataset.

CONCLUSION

A Machine Learning-based Botnet Attack Detection System was developed to classify network traffic into normal and botnet categories. The workflow includes data cleaning, categorical encoding, feature selection, feature engineering, scaling, model training, testing, and performance evaluation. Random Forest and XGBoost were implemented and compared using multiple evaluation measures. Both models achieved strong results, but XGBoost produced the best overall performance with 0.970248 accuracy, 0.965116 precision, 0.982249 recall, 0.973607 F1-score, and 0.987922 ROC-AUC. The results demonstrate the potential of machine learning for automated botnet detection and network security monitoring.

FUTURE SCOPE

- Collect larger and more diverse network traffic datasets to improve model generalization.
- Integrate real-time network traffic monitoring for live botnet detection.
- Explore deep learning techniques such as ANN, CNN, and LSTM for improved detection accuracy.
- Implement automated alert and notification mechanisms for security administrators.
- Extend the system to detect multiple cyber-attacks such as DDoS, phishing, ransomware, and malware infections.
- Deploy the model in cloud and enterprise environments for large-scale network monitoring.
- Develop a web-based dashboard for visualizing network traffic and attack statistics.
- Continuously update the model with new attack patterns to handle evolving cybersecurity threats.

REFERENCES

[1] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, 4th ed., Morgan Kaufmann Publishers, 2016.
[2] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, 3rd ed., 2011.
[3] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd ed., O'Reilly Media, 2022.
[4] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, Springer, 2017.
[5] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

 © Author(s) | CC BY 4.0 International License.

iJETRM
INTERNATIONAL JOURNAL OF ENGINEERING TECHNOLOGY RESEARCH & MANAGEMENT (IJETRM)
https://ijetrm.com/

[6] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, pp. 785–794, 2016.
[7] S. García, M. Grill, J. Stiborek, and A. Zunino, "An Empirical Comparison of Botnet Detection Methods," Computers & Security, vol. 45, pp. 100–123, 2014.
[8] W. Stallings, Network Security Essentials: Applications and Standards, 7th ed., Pearson Education, 2017.
[9] W. Stallings, Cryptography and Network Security: Principles and Practice, 8th ed., Pearson Education, 2023.
[10] N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," Military Communications and Information Systems Conference (MilCIS), 2015.
[11] Scikit-Learn Documentation, Machine Learning in Python, https://scikit-learn.org
[12] XGBoost Documentation, XGBoost Python Package Documentation, https://xgboost.readthedocs.io
[13] Pandas Documentation, Python Data Analysis Library, https://pandas.pydata.org
[14] NumPy Documentation, Scientific Computing with Python, https://numpy.org
[15] Matplotlib Documentation, Visualization with Python, https://matplotlib.org
[16] Seaborn Documentation, Statistical Data Visualization, https://seaborn.pydata.org
[17] Python Software Foundation, Python Programming Language Documentation, https://www.python.org
[18] Open Web Application Security Project (OWASP), Cybersecurity and Network Security Resources, https://owasp.org
[19] National Institute of Standards and Technology (NIST), Cybersecurity Framework and Security Guidelines, https://www.nist.gov
[20] Cisco Networking Academy, Introduction to Cybersecurity and Network Threats, Cisco Systems Inc.

 © Author(s) | CC BY 4.0 International License.