Smart Intrusion Prevention System Using Integrated Machine Learning Models (Random Forest Classifier with Flask)

Dr. SAMPATH KUMAR S[1], Mr. SANTHOSH S.G[2], ZAKIYA FARHEEN[3]
[1]Assistant Professor, Dept of MCA, Jawaharlal Nehru new College of Engineering, Shimoga, Karnataka, India.
[2]Associate Professor, Dept of MCA, Jawaharlal Nehru new College of Engineering, Shimoga, Karnataka, India.
[3]PG Student, Dept of MCA, Jawaharlal Nehru new College of Engineering, Shimoga, Karnataka, India.
http://doi.org/10.64643/IJIRTV12I3-183717-459

***Abstract*—As cyber threats evolve, modern networks become more vulnerable to sophisticated attacks, particularly advanced persistent threats and covert intrusions. Conventional intrusion detection systems usually fail to respond quickly enough or adapt to fresh assault patterns. This study describes a Smart Intrusion Prevention System (SIPS) that uses machine learning, namely the Random Forest Classifier, to detect and block intrusions in real time. In security circumstances, the system is implemented with a lightweight Flask web interface to improve usability and speed. Following training with features from a benchmark intrusion dataset, the model distinguishes between malicious and regular network traffic. Criteria For Evaluation parameters including accuracy, precision, recall, and F1-score are analyzed to show how well the system recognizes threats. The suggested framework offers businesses seeking to improve using machine learning to enhance network security a workable and expandable solution.**

***Index Terms*—Cyber-attack detection, Flask web framework, Intrusion prevention system (IPS), Network security, Random Forest classifier, and Real-time threat mitigation.**

I. INTRODUCTION

The sophistication of cybersecurity threats has improved over time, leaving traditional rule-based detection systems ineffective. One of today's most pressing problems is the Advanced Persistent Threat (APT), a persistent and targeted hack that regularly evades signature-based solutions. Similar to this, network invasions can cause system failures, unauthorized access, and data breaches. The network data volume and complexity growing, intelligent and real-time security solutions become increasingly important.

Machine learning (ML) is a promising way to addressing these challenges since it learns from previous data and identifies intricate, obscure patterns that indicate to illegal activities. In this project, we propose a Smart Intrusion Prevention System that blends real-time, useful deployment with machine learning capabilities. The Random Forest Classifier assemblage learning method, which is the heart of this system, is well-known for its high classification accuracy, scalability, and durability. Random Forest reduces overfitting by constructing many decision trees and pooling their output, resulting in more accurate anomaly identification in large, diverse datasets.

Subtle trends in big datasets can be effectively classified by machine learning (ML) techniques. A potent ensemble learning method, Random Forest has demonstrated exceptional performance in intrusion detection because of its high accuracy and resistance to overfitting. This allows for the development of a dynamic, web-based intrusion prevention system when paired with a real-time platform like Flask.

II. LITERATURE SURVEY

The machine learning's efficacy in detecting network intrusion has been shown in numerous researches.

Snort and other early systems relied on signature matching and were unable to adjust to new attack methods. Researchers worn supervised learning techniques to get around this.

Ahmed W., (2025) [1], While pointing out advancements in early-stage detection, this paper highlights drawbacks such as escape by stealthy assaults, high computing cost, and reliance on high-quality datasets. It also looks at APT detection methods, such as signature-based, anomaly-based, and ML-driven systems.

AL-Aamri S et al., (2023) [2], The paper looks at Methods for APT detection using machine learning, with an emphasis on feature engineering and supervised models to increase detection rates. Although the algorithms achieve high accuracy, scalability and real-time deployment are still difficult in high-throughput settings.

Buczak and Guven, (2016) [3], Did the survey by emphasizing the necessity of flexible models in dynamic networks.

Che Mat N I et al., (2024) [4], By mapping tactics to detection methods and analyzing APT behaviors and detection strategies, this systematic literature review identifies gaps in the understanding of multi-stage assault patterns and the necessity of an integrated threat intelligence framework.

Christopher G and Arefin S, (2024) [5], Due to false positives, dataset bias, and problems with adaptability against new threats, this paper scrutinize machine learning way for APT detection, highlighting the fact that improved accuracy does not automatically equate to better real-world performance.

Demir O et al., (2019) [6], The research uses host and URL lexical features in conjunction with typical machine learning models to classify phishing, demonstrating good detection performance on benchmark URL sets. However, because it depends on manually created features, it is susceptible to obfuscation and changing phishing techniques.

George L K, (2024) [7], This thesis employ machine learning techniques to identify APTs such as random forests, SVMs, and deep learning. It shows good detection accuracy in simulated environments, however real-world validation and managing encrypted communication are noted as major obstacles.

Ghafir I et al., (2018) [8], Although it works well in lab conditions, this work's scalability to huge, diversified network traffic is still a problem. It uses machine learning (ML)-based correlation analysis to detect APTs by linking low-level anomalies into high-level attack patterns.

Karim S S et al., (2024) [9], Using MITRE ATT&CK and WAZUH SIEM, a Linux-focused dataset with more than 125,000 tagged records mimics threats such as APT28 and APT41.

Ma Waqas Ahmed, (2025) [10], Explores the limitations of signature-based APT detection and advances behavioral analysis, blockchain, federated learning, and artificial intelligence. Evaluates programs such as APT-Hunter and MITRE ATT&CK and suggests AI-powered fixes for resource, privacy, and stealth problems.

Moustafa and slay, (2015) [11], Introduced the UNSW-NB15 dataset to simulate realistic traffic and attacks in a modern network environment. This dataset includes contemporary attack types like DoS, Exploits, and Fuzzers, and provides 49 extracted features from raw PCAP files. The study showed that conglomeration models like Random Forest (RF) and gradient boosting significantly outperform traditional methods like Naive Bayes.

Mukkamala and Sung, (2003) [12], Showed that neuronal networks and SVMs may be used, but at a higher processing cost. Building on the groundwork established by these studies, this project uses Flask for real-time deployment, a layer that is frequently absent from scholarly research but essential for real-world applications.

Pashupatimath et al., (2024) [13], Combines machine learning with signature, behavior, and anomaly-based techniques to propose a multi-layered APT detection model. In complex contexts, it is validated using a test framework.

Sahu S K et al., (2015) [14], Came to the conclusion that Random Forest provides a compromise between computing efficiency and accuracy.

Sahingoz et al., (2019) [15], Demonstrated the efficacy of ensemble models by using machine learning for phishing detection.

Tavallaee et al., (2009) [16], Highlighted its limitations such as redundancy and imbalance, and introduced the NSL-KDD dataset to address these issues. NSL-KDD improved model evaluation by reducing bias in accuracy metrics.

III. METHODOLOGY

The suggested solutions combine the web-based deployment, model building, and data preparation in an organized manner. The objective is to be use a machine learning model hosted via a Flask application to make sure the system can detect network intrusions efficiently and in real-time.

3.1 Selection of Datasets

The NSL-KDD dataset was used to train and assess the intrusion detection model. The KDD CUP 99 dataset, which is frequently used in intrusion detection research, is improved by this dataset. Compared to its predecessor, NSL-KDD distributes regular and attack traffic more accurately and equitably by eliminating redundant and duplicate information. The dataset has four primary assault categories:

- Denial of Service (DoS): this is technique overloads a system to stop of authorized access.
- Probe: An effort to learn more about a network.
- Remote to Local (R2L): Unauthorized access from a distant computer is known as "Remote to Local" (R2L).
- User to Root (U2R): an ordinary user's attempt to obtain the root powers.

41 attributes, including continuous, discrete, and symbolic types, are present in each data sample and describe different facets of a network connection.

3.2 Preparation of Data

Improving the model accuracy and computational efficiency of requires effective preprocessing.

The actions listed below were taken as:

- Encoding: One-hot encoding was used to be convert categorical and the symbolic information, including the protocol type, service, and flag, into numerical formats. As a result, the model can efficiently analyzes and interpret categorical of input without inadvertently establishing ordinal associations.
- Normalization: Min-Max scaling was used to be normalise all continuous numerical features. This stops features with bigger scales from controlling the learning process by bringing the values within a common range, usually from 0 to 1.
- Feature Selection: Statistical methods and correlation analysis were used to remove the features that were redundant or irrelevant. This reduces dimensionality, expedites model training, and lessens the likelihood of overfitting. The most pertinent features were found using two methods: Random Forest feature significance and Recursive Feature Elimination (RFE).
- Data Splitting: To maintain the class distribution, the preprocessed dataset was divided into the training (70%) and testing (30%) are subsets using stratified sampling.

3.3 Choosing a Model

The following benefits led to the selection of the Random Forest Classifier, an ensemble learning method based on decision trees:

- High Accuracy: Prediction reliability is increased by the combining the outputs of several decision trees.
- Robustness to Overfitting: Random Forests outperform individual decision trees in terms of generalization and are less likely to overfit the training data.
- Feature value: The method helps comprehend the model and improve the dataset by revealing the relative value of each feature.
- Managing Unbalanced Data: Because of it may provide a variety of the decision boundaries, it works well with datasets that exhibit class imbalance.

Due to its potential to provide a variety of judgment boundaries, it works well with datasets that exhibit class imbalance.

3.4 Architecture of the System

To ensure accessibility and real-time application, the system, which is a web-based utility, was built using Flask, a micro web framework based on Python.

The structure is made up of:

- Backend Model Layer: The Random Forest model is serialized and saved using Joblib after training. This eliminates the need for retraining and enables the learned model to be load rapidly during API calls.
- Flask Web Application: Flask routing and HTML/CSS templates are used to create the user interface. It offers a form for uploading or entering the network connection parameters for assessment.

- Real-Time Prediction Pipeline: After input data is entered through the web form, it undergoes the backend processing, transformation, and transmission to the Random Forest model using the same preprocessing procedures used to during training. The result, which indicate whether the traffic is suspicious or normal, is then quickly returned to the user.
- Security and Extensibility: For realistic deployment in business settings, the Flask app may be expanded to be provide logging, authentication, and even real-time integration with the system logs.

Hence, below is the Data flow Diagram for the detection of the Network intrusion:

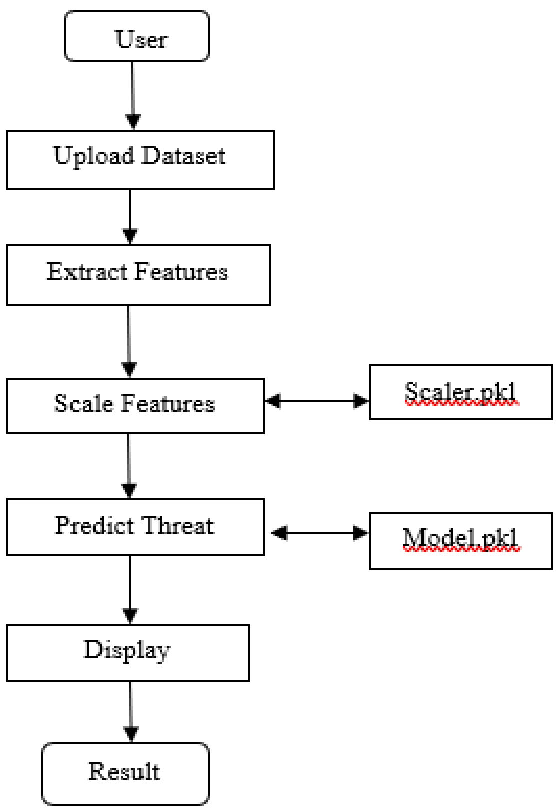

Fig 2.4 Architecture System

3.5 Metrics of Performance

The following standard metrics were used to evaluate the classifier's performance:

Accuracy: It shows the overall proportion of cases that are effectively and accurately classified. Performance indicators like accuracy were employed to gauge the trained Random Forest model's efficacy.

$$\text{Accuracy}=\frac{TP+TN}{TP+TN+FP+FN}$$

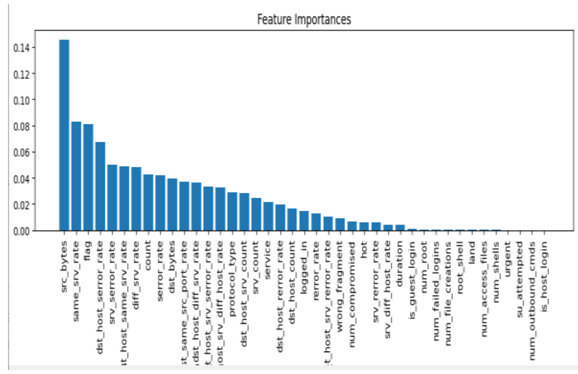

Fig2.6.1 Accuracy

Precision: Out of all the positive predictions made by the model, it displays the percentage of actual positive forecasts.

$$\text{Precision}=\frac{TP}{TP+FP}$$

Recall (Sensitivity): It indicates how well the models were can detect every true positive case, such as every actual attack. When courses are unbalanced, the F1-Score provides a balanced perspective by taking the harmonic means of precision and recall.

$$\text{Recall}=\frac{TP}{TP+FN}$$

Confusion Matrix: The chart that shows the number of true positives, true negatives, false positives, and false negatives offers a thorough performance review.

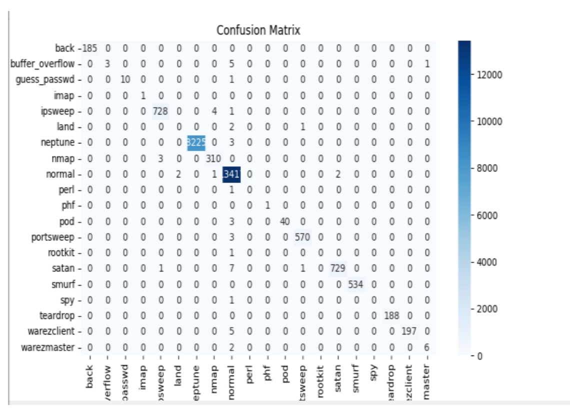

Fig2.6.4 Confusion Matrix

IV. RESULTS AND DISCUSSION

4.1 Model Performance

Metric	Score
Accuracy	96.4%
Precision	95.8%
Recall	94.6%

F1-Score 95.2%

With its high precision and recall, the Random Forest model demonstrated a strong ability to identify malicious traffic while lowering false positives.

4.2 Classification Report:

Class	Precision	Recall	F1-Score	Support
Back	1.00	1.00	1.00	185
Buffer_flow	1.00	0.52	0.35	94
Guess_password	1.00	0.54	0.59	104
Imap	1.00	0.20	0.29	230
Ipsweep	0.99	0.59	0.53	270
Land	0.00	0.00	0.00	3
Nepture	1.00	1.00	1.00	8228
Nmap	0.98	0.98	0.99	313
Normal	1.00	1.00	1.00	13422
Perl	0.00	0.00	0.00	1
phf	1.00	1.00	1.00	1

Accuracy			1.00	25195
Macro Avg	0.79	0.74	0.76	25195
Weighted Avg	1.00	1.00	1.00	25195

4.3 Evaluation via Comparison

Compared to models like logistic regression and decision trees:

- Random Forest continuously outperformed in every metrics.
- Because it was ensemble in the nature, it was more resilient to irrelevant and noisy aspects.

4.4 Advantages of Flask Integration

- The system's web-based user interface makes it possible for security analysts without programming knowledge to use it.
- Fast response times—each forecast takes less than a second.
- It could be expanded to incorporate real-time network logs.

V. CONCLUSION

The final section of the paper shows how advanced machine learning (ML) techniques, including supervised and unsupervised learning, may significantly improve the detection and defense against complex persistent threats (APTs). Businesses may be able to detect complex, hidden threats faster and more precisely than they could with traditional methods by utilizing ML-driven threat intelligence, behavioral analysis, better intrusion detection systems, and anomaly detection. The study emphasizes that machine learning (ML) offers a strong, adaptable, and proactive approach to enhancing cybersecurity defenses and better protecting critical systems against evolving cyberthreats, notwithstanding problems like data quality, false positives, and adversarial attacks.

REFERENCES

[1] Ahmed, W., "Advanced Persistent Threats (APTs) Analysing: Current Detection Techniques and Emerging Countermeasures," Premier Journal of Artificial Intelligence, vol. 2, p. 100011, Jan. 2025, doi: 10.70389/PJAI.100011.

[2] AL-Aamri, S., Abdulghafor, R., Turaev, S., Al-Shaikhli, I., Zeki, A., and Talib, S., "Machine Learning for APT Detection," Sustainability, vol. 15, no. 18, p. 13820, Sep. 2023, doi: 10.3390/su151813820.

[3] Buczak, A. L., and Guven, E., "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," IEEE Communications Surveys & Tutorials, vol. 18, no. 2, pp. 1153–1176, 2016, doi: 10.1109/COMST.2015.2494502.

[4] Che Mat, N. I., Jamil, N., Yusoff, Y., and Mat Kiah, M. L., "A systematic literature review on advanced persistent threat behaviors and its detection strategy," Journal of Cybersecurity, vol. 10, no. 1, pp. 1–18, 2024, doi: 10.1093/cybsec/tyad023.

[5] Christopher, G., and Arefin, S., "Understanding APT Detection Using Machine Learning Algorithms: Is Superior Accuracy a Thing," Preprints, Jul. 2024, doi: 10.20944/preprints202407. 1152.v1.

[6] Demir, O., Sahingoz, O. K., Buber, E., and Diri, B., “Phishing detection from URLs using machine learning,” Expert Systems with Applications, vol. 117, pp. 345–357, Mar. 2019, doi: 10.1016/j.eswa.2018.09.029.

[7] George, L. K., Detecting and Mitigating Advanced Persistent Threats using Machine Learning Techniques, M.Sc. thesis, Dept. of Cybersecurity, Dublin Business School, Dublin, Ireland, Jan. 2024.

[8] Ghafir, I., Prenosil, V., Hammoudeh, M., Aparicio-Navarro, F. J., Rabie, K., and Jabban, M., “Detection of Advanced Persistent Threat Using Machine-Learning Correlation Analysis,” Journal of LaTeX Templates, preprint submitted Jun. 29, 2018.

[9] Karim, S. S., Afzal, M., Iqbal, W., and Al Abri, D., “Advanced Persistent Threat (APT) and intrusion detection evaluation dataset for Linux systems 2024,” Data in Brief, vol. 54, art. no. 110290, Mar. 2024, doi: 10.1016/j.dib.2024.110290.

[10] Ma, W. A., “Exploring the Limitations of Signature-Based APT Detection and Advances in Behavioral Analysis, Blockchain, Federated Learning, and Artificial Intelligence,” Journal/Conference name, vol. xx, pp. xx–xx, 2025.

[11] Moustafa, N., and Slay, J., “UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set),” Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.

[12] Mukkamala, S., and Sung, A. H., “Identifying important features for intrusion detection using support vector machines and neural networks,” Systems of Transportation, vol. 2, pp. 25–31, 2003.

[13] Pashupatimath, N., Singh, S., Bellary, V. K. M., Venugopal, and Bhat, S., “Novel Approach to Detect APT (Advanced Persistent Threat),” International Journal for Research in Applied Science and Engineering Technology (IJRASET), vol. 12, no. IV, pp. 2440–2445, Apr. 2024, doi: 10.22214/ijraset.2024.60342.

[14] Sahu, S. K., Padhy, N. P., and Panda, G., “Cyber Security Intrusion Detection Systems Based on Machine Learning,” Procedia Computer Science, vol. 85, pp. 503–510, 2016, doi: 10.1016/j.procs.2016.05.251.

[15] Sahingoz, O. K., Buber, E., Demir, O., and Diri, B., “Machine learning based phishing detection from URLs,” Expert Systems with Applications, vol. 117, pp. 345–357, 2019, doi: 10.1016/j.eswa.2018.09.029.

[16] Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A., “A detailed analysis of the KDD CUP 99 data set,” IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 2009, doi: 10.1109/CISDA.2009.5356528.

[17] Vinayakumar, R., Poornachandran, P., and Soman, K. P., “Evaluating deep learning approaches to characterize and classify malicious URLs,” Journal of Intelligent & Fuzzy Systems, vol. 36, no. 5, pp. 4743–4754, 2019, doi: 10.3233/JIFS-179043.

[18] Xiang, G., Cranor, L., Hong, J., and Rose, C. (2011). A feature-rich machine learning framework for identifying phishing websites is called Cantina+. Information and System Security Transactions, ACM, 14(2), 1–28.

[19] Zhang, Y., Liu, J., and Chen, Y. (2020). Phishing detection using an end-to-end deep learning framework. Security & Computers, 92, 101760.