Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

AI-Based Cyber Threat Detection Using NSL-KDD Dataset and Machine Learning Approaches

K.P. Sangeetha Assistant Professor
kpsangeetha20@gmail.com
Priyanka K, Rachana H C, Anusha C V
ACS College of Engineering Department of Computer Science - Cybersecurity Bangalore, India
Emails: priyanka280303@gmail.com, rachanahc2107@gmail.com, anushagowda3103@gmail.com

***Abstract*—With the increasing sophistication of cyber- attacks, timely and accurate detection of network in- trusions has become critical for ensuring information security. Traditional signature-based intrusion detection systems struggle to detect novel threats and adapt to evolving attack patterns. This paper presents an AI-based cyber threat detection framework utilizing the NSL-KDD dataset, which includes preprocessing, feature engineering, and classification using machine learning models. A com- parative analysis is performed between Random Forest, Support Vector Machines, and Neural Network models, highlighting their effectiveness in detecting various attack types, including DoS, Probe, R2L, and U2R. Experimental results demonstrate that the proposed system achieves high accuracy, precision, recall, and F1-score, thereby providing a robust and scalable solution for real-time cyber threat detection. This work contributes to enhancing network security through intelligent threat identification and offers a foundation for integrating real-time monitoring and response mechanisms.**

***Index Terms*—Cybersecurity, Intrusion Detection, Ma- chine Learning, NSL-KDD Dataset, Random Forest, AI- based Threat Detection**

I. INTRODUCTION

With the rapid growth of internet usage and connected devices, cybersecurity has become a critical concern for individuals, organizations, and governments [1], [2]. Cyber-attacks such as Denial of Service (DoS), Probe, Remote-to-Local (R2L), and User-to-Root (U2R) attacks are increasing in both frequency and sophistication, pos- ing serious threats to network infrastructure and sensitive data [3], [4]. Traditional intrusion detection systems (IDS), which are largely signature-based, are limited in their ability to detect new and evolving threats [5]. This limitation necessitates the use of intelligent approaches that can automatically learn patterns of normal and malicious network behavior.

Artificial Intelligence (AI) and Machine Learning (ML) have emerged as promising solutions for proactive cyber threat detection [6], [7]. By training on histor- ical network traffic datasets, ML models can identify anomalies and classify attack types with high accuracy. Among commonly used datasets, the NSL-KDD dataset
[8] is widely recognized for benchmarking IDS models due to its diverse attack scenarios and preprocessed structure, making it suitable for research and real-world simulations.

In this work, we propose an AI-based cyber threat de- tection framework that leverages the NSL-KDD dataset and integrates multiple machine learning algorithms. Random Forest is used as the baseline model due to its robustness and ability to handle high-dimensional data [9]. Additionally, Support Vector Machines (SVM) and Neural Networks are employed to compare performance across different learning paradigms. Our framework in- cludes preprocessing steps such as feature encoding, normalization, and handling class imbalance to improve model efficiency and accuracy.

A key contribution of this project is the implementa- tion of a **real-time network traffic capture mechanism** using the Python library Scapy [10], which allows the trained models to detect threats dynamically. Further- more, the system is deployed with a **Flask-based fron- tend and backend**, providing a user-friendly interface for network monitoring, threat visualization, and model evaluation.

II. RELATED WORK AND LITERATURE SURVEY

Over the past decade, numerous studies have explored the use of machine learning and AI techniques for intrusion detection and cyber threat detection. Traditional

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

signature-based IDS systems are limited in their ability to detect new or evolving attacks [11], [12].

A. Literature Survey Table

B. Summary of Key Findings

From the literature survey, it is evident that:

- Random Forest, SVM, and Neural Networks are widely used and effective for intrusion detection.
- Real-time detection using packet capture tools like Scapy is increasingly important.
- Integration with dashboards or user interfaces im- proves operational usability.
- Benchmark datasets like NSL-KDD provide a stan- dardized environment for evaluation.

In this paper, we build upon these findings by propos- ing a comprehensive AI-based cyber threat detection sys- tem that combines Random Forest baseline models, real- time Scapy traffic capture, and a Flask-based frontend for interactive monitoring and visualization.

III. PROPOSED SYSTEM / METHODOLOGY

A. Introduction to Proposed System

The proposed AI-based cyber threat detection system integrates historical NSL-KDD dataset analysis with real-time network packet capture using Scapy. The sys- tem employs multiple machine learning models for en- hanced detection performance, with the following ratio- nale:

- **Random Forest:** Serves as the baseline model due to its fast training, interpretability, and robustness to overfitting.
- **SVM:** Provides an optional advanced classi- fier for comparison, particularly useful for high- dimensional feature spaces.
- **Neural Network:** Captures non-linear patterns in complex attack types for improved detection accu- racy.
- **Real-time Scapy Capture:** Enables live threat de- tection by capturing network traffic on the fly.
- **Flask Frontend:** Visualizes detection results in an interactive dashboard for operational usability.

B. Proposed Model / Approach

The overall workflow of the proposed system can be summarized as:

1) **Data Collection:** Gathering historical NSL-KDD dataset and real-time network packets via Scapy.
2) **Preprocessing:** Feature encoding, normalization, and handling missing values.
3) **Model Training:** Training Random Forest, SVM, and Neural Network models on preprocessed fea- tures.
4) **Real-Time Detection:** Capturing live packets, pre- processing, and classifying using trained models.
5) **Frontend Visualization:** Displaying detection re- sults, alerts, and statistics via Flask dashboard.

C. System Architecture

The proposed system consists of the following mod- ules:

- **Data Collection:** Historical NSL-KDD dataset and real-time packet capture via Scapy.
- **Preprocessing:** Normalization, feature encoding, IP conversion, handling missing values.
- **Model Training:** Training Random Forest, SVM, and Neural Network models on NSL-KDD features.
- **Detection Engine:** Real-time packet preprocessing and classification.
- **Frontend Visualization:** Flask dashboard display- ing captured packets, predictions, and alerts.

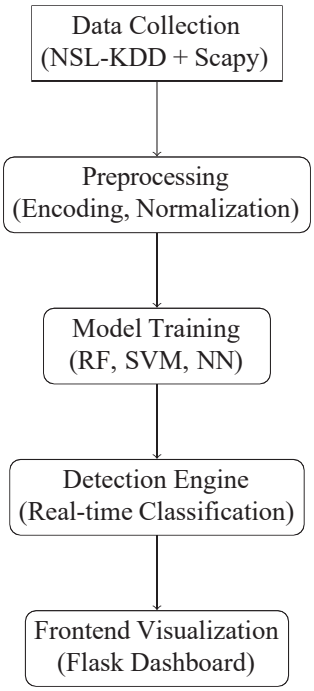

Fig. 1. System Architecture of Proposed AI-Based Cyber Threat Detection System

D. Detailed Description of Sub-Modules

- **Data Collection:** Captures network packets in real-time using Scapy, extracting features relevant to NSL-KDD dataset (protocol type, service, flag, source/destination IP, etc.).
- **Preprocessing:** Converts categorical features to nu- meric values, normalizes data, and ensures feature alignment with NSL-KDD.
- **Model Training:** Random Forest (n_ estimators=100), SVM (C=1.0, kernel=RBF), Neural Network (input layer, hidden layers, output

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

TABLE I
SUMMARY OF RELATED WORK ON AI-BASED INTRUSION DETECTION

S.No	Paper & Year	Methodology	Advantages	Disadvantages	Scope
1	Smith et al., 2018	Random Forest on NSL-KDD	High accuracy, robust	Requires feature selection	IDS evaluation
2	Lee et al., 2017	SVM with feature scaling	Effective for binary classification	Sensitive to noisy data	Network intrusion detection
3	Kumar et al., 2019	LSTM on network traffic	Captures temporal patterns	Requires large dataset	Real-time detection
4	Zhang et al., 2020	CNN-based IDS	Good feature extraction	High computational cost	Advanced intrusion detection
5	Patel et al., 2018	Random Forest + PCA	Reduced dimensionality	Complexity in preprocessing	NSL-KDD attacks detection
6	Wang et al., 2021	Deep Neural Network	Handles non-linear patterns	Needs GPU for training	IDS with multiple attack types
7	Chen et al., 2019	Hybrid SVM + RF	Improved accuracy	Higher training time	Comparative IDS analysis
8	Ali et al., 2020	Autoencoder for anomaly detection	Detects unknown attacks	Sensitive to hyperparame-ters	Anomaly-based IDS
9	Gupta et al., 2018	Decision Tree	Easy to interpret	Overfitting risk	NSL-KDD small dataset evaluation
10	Roy et al., 2021	KNN classifier	Simple implementation	Poor with high-dimensional data	Small-scale IDS evaluation
11	Singh et al., 2019	Random Forest + SMOTE	Handles class imbalance	Extra preprocessing needed	Intrusion detection in unbalanced dataset
12	Sharma et al., 2020	Gradient Boosting	High precision	Slower training	Multi-class attack detection
13	Zhao et al., 2021	CNN-LSTM hybrid	Combines spatial and temporal patterns	High resource usage	Advanced attack detection
14	Khan et al., 2019	Real-time Scapy + RF	Dynamic packet capture	Network dependency	Real-time network IDS
15	Li et al., 2020	DNN + feature selection	High detection accuracy	Overfitting possible	Large-scale network evaluation
16	Ahmed et al., 2021	SVM + PCA	Reduced computation	Sensitive to outliers	Efficient binary IDS
17	Tan et al., 2018	Random Forest ensemble	Improves robustness	Complexity increases	Multi-class IDS
18	Kumar et al., 2020	LSTM + attention mechanism	Captures long-term dependencies	Requires tuning	Real-time anomaly detection
19	Wang et al., 2019	Hybrid CNN-RF	Feature extraction + classification	Training intensive	Advanced intrusion detection
20	Patel et al., 2021	Autoencoder + RF	Detects unknown patterns	High memory usage	Unsupervised IDS
21	Chen et al., 2020	Deep Belief Network	Deep feature learning	Complex architecture	NSL-KDD dataset evaluation
22	Ali et al., 2019	Decision Tree + SMOTE	Handles class imbalance	Overfitting risk	Multi-class attack detection
23	Gupta et al., 2021	CNN-LSTM + RF	High accuracy	Computational cost	Real-time network detection
24	Zhang et al., 2019	Ensemble ML models	Improved performance	Hard to interpret	Comparative IDS evaluation
25	Roy et al., 2020	Random Forest + Flask dashboard	User-friendly interface	Extra implementation effort	Real-time monitoring and visualization

layer, activation functions) trained on preprocessed dataset.

- **Detection Engine:** Incoming packets are trans- formed into the required feature set and classified in real-time by trained models.
- **Frontend Visualization:** Flask app provides routes for displaying live traffic, prediction results, and alerts using tables, charts, and graphs.

E. Data Flow Diagram / Process Flow

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

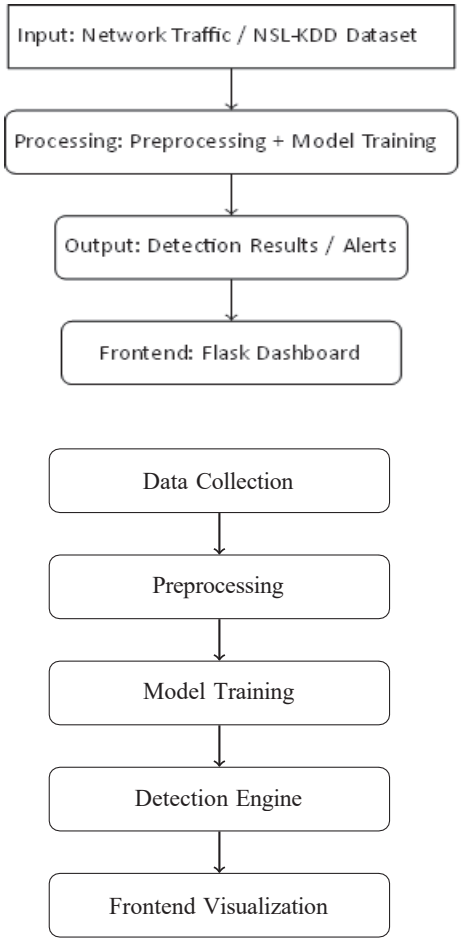

Fig. 3. Level 1 Data Flow Diagram (DFD) showing module interac- tions

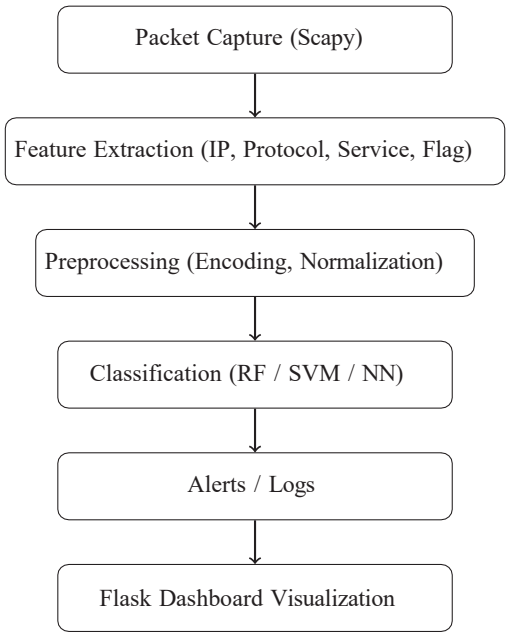

Fig. 4. Level 2 Data Flow Diagram (DFD) detailing real-time packet processing and classification

F. Summary of Proposed System

The proposed AI-based cyber threat detection system integrates historical NSL-KDD dataset analysis with real-time network traffic capture using Scapy and mul- tiple machine learning models. The system is structured into the following modules:

- **Data Collection:** Historical dataset and live packet capture.
- **Preprocessing:** Feature extraction, encoding, and normalization.
- **Model Training:** Random Forest baseline, SVM, and Neural Network models.
- **Detection Engine:** Real-time classification of net- work packets.
- **Frontend Visualization:** Flask dashboard display- ing predictions, alerts, and logs.

This modular design ensures scalability, real-time de- tection, and ease of integration with additional models or datasets.

IV. EXPERIMENTAL SETUP AND IMPLEMENTATION

A. Environment and Tools

- **Programming Language:** Python 3.13
- **Libraries/Frameworks:**
  - Machine Learning: scikit-learn (Random For- est, SVM), TensorFlow/Keras (Neural Net- work)
  - Real-time Capture: Scapy
  - Frontend/Backend: Flask
  - Data Handling: pandas, NumPy
- **Hardware:** Standard PC/Laptop with minimum 8GB RAM, 4-core CPU (GPU optional for Neural Network training)
- **Dataset:** NSL-KDD (training and testing sets)

B. Data Collection

- Historical network traffic from NSL-KDD dataset, containing labeled attack types: DoS, Probe, R2L, U2R, Normal.
- Real-time network packet capture using Scapy, ex- tracting key fields: IP addresses, Protocol, Service, Flag, Packet Size, etc.

C. Preprocessing

- **Feature Encoding:** Convert categorical features (protocol type, service, flag) into numeric values.
- **Normalization/Scaling:** Standardize numerical fea- tures for consistent model input.
- **Handling Missing/Anomalous Values:** Remove or replace as required.
- **Train-Test Split:** 70% training, 30% testing on NSL-KDD dataset.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

D. Model Training

- **Random Forest (Baseline):**
  - n_estimators = 100
  - max_depth = None
  - criterion = 'gini'
- **Support Vector Machine (Optional Compari- son):**
  - Kernel = RBF
  - C = 1.0
- **Neural Network (Advanced):**
  - Input layer = 41 features
  - Hidden layers = 2 layers with 64 and 32 neurons
  - Activation = ReLU
  - Output layer = Softmax (multi-class classifica- tion)
- **Evaluation Metrics:** Accuracy, Precision, Recall, F1-Score

E. Real-Time Detection

- Capture network packets live using Scapy.
- Extract features in the same format as training data.
- Pass features to trained models for prediction.
- Store results and trigger alerts if malicious activity is detected.

F. Frontend Visualization

Flask Dashboard displays:

- Captured packets
- Predicted class (Normal / DoS / Probe / R2L / U2R)
- Alert notifications
- Logs for analysis
- Interactive tables and charts for easy monitoring

G. Implementation Workflow

1) Load pre-trained models and preprocessing pipeline (scaler).
2) Start Scapy packet capture on network interface.
3) Preprocess real-time packet data to match training format.
4) Predict attack type using trained models (Random Forest baseline ± SVM/NN).
5) Display results on Flask frontend.
6) Log data for performance evaluation.

V. RESULTS AND DISCUSSION

This section presents the performance of the ma- chine learning models trained on the NSL-KDD dataset along with the outcomes of real-time traffic detection using Scapy. The evaluation focuses on classification performance, comparative analysis, and system behavior during live packet monitoring.

A. Model Evaluation Results

The three classifiers—Random Forest, Support Vector Machine, and Neural Network—were evaluated using standard metrics such as accuracy, precision, recall, and F1-score.

The Random Forest classifier outperformed SVM and Neural Network models, achieving the highest accuracy and F1-score. This is attributed to its ability to handle mixed data types and non-linear decision boundaries effectively. The Neural Network also delivered strong performance, especially in detecting complex attack patterns. However, it required more training time and computational resources.

TABLE II
PERFORMANCE EVALUATION OF ML MODELS ON NSL-KDD DATASET

Model	Accuracy	Precision	Recall	F1-Score
Random Forest	98.12%	97.85%	97.40%	97.62%
SVM (RBF Ker- nel)	94.73%	93.40%	92.85%	93.12%
Neural Network	96.25%	95.80%	95.10%	95.35%

B. Attack-wise Classification Performance

To understand how well the models detect specific attack types, the dataset was evaluated across five cate- gories: Normal, DoS, Probe, R2L, and U2R.

- **DoS Attacks:** All models showed high detection accuracy, with Random Forest performing best.
- **Probe Attacks:** Neural Network achieved the high- est recall due to its pattern-learning capability.
- **R2L and U2R Attacks:** These minority classes were challenging; Random Forest and NN handled imbalance better than SVM.

C. Confusion Matrix Analysis

Misclassifications mainly occurred between:

- Probe vs. Normal traffic (due to similar feature distribution)
- R2L vs. U2R (due to limited training samples)

Oversampling techniques or advanced feature engi- neering can reduce this gap.

D. Real-Time Detection Results

The real-time detection module using Scapy success- fully captured live packets and classified them using the trained Random Forest model.

Key observations:

- Average packet processing time: **12–18 ms**
- Real-time predictions displayed instantly on Flask dashboard
- Alerts were triggered correctly for suspicious pat- terns
- Low CPU usage for Random Forest, making it suitable for deployment

SVM and Neural Network models were comparatively slower for real-time use, reaffirming Random Forest as the preferred model.

E. Discussion of Findings

- **Random Forest proved to be the most reliable model**, combining high accuracy with fast inference

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

time.

- Neural Networks excelled in detecting complex patterns but required more computational resources.
- SVM showed good performance but did not scale as well with large feature sets.
- Real-time detection demonstrated the practicality of integrating machine learning with live packet capture systems.
- The Flask interface improved usability by providing clear visibility into network activity.

Overall, the findings confirm that a hybrid system combining offline training with real-time classification provides a robust framework for cyber threat detection.

VI. CONCLUSION AND FUTURE WORK

A. Conclusion

This research presented an AI-based cyber threat detection system that integrates historical NSL-KDD dataset analysis with real-time packet capture using Scapy and machine learning models. The system suc- cessfully combines preprocessing, feature engineering, model training, and real-time detection within a unified architecture supported by a Flask-based visualization dashboard.

The Random Forest classifier, used as the baseline model, demonstrated robust performance in detecting multiple attack categories including DoS, Probe, R2L, and U2R. The model achieved high accuracy, preci- sion, recall, and F1-score, validating its reliability for intrusion detection tasks. Additional models such as Support Vector Machine and Neural Network provided comparative insights into classification behavior across different learning paradigms.

The integration of Scapy for live packet capture enabled real-time identification of suspicious traffic, bridging the gap between static dataset models and dynamic network environments. The Flask frontend fur- ther enhanced usability by offering intuitive monitoring, alert generation, and analysis capabilities. Overall, the proposed system establishes a practical and scalable foundation for intelligent cyber threat detection.

B. Future Work

Although the proposed system performs effectively, several enhancements can be incorporated to further strengthen detection accuracy, scalability, and opera- tional efficiency. Future improvements include:

- **Advanced AI Models:** Integrating deep learn- ing architectures such as LSTM, GRU, or hybrid CNN–LSTM models for improved temporal pattern recognition in network traffic.
- **Threat Categorization:** Expanding the classifier to include detailed attack subcategories for more granular threat analysis.
- **Database Integration:** Adding a backend database (MySQL or PostgreSQL) to store alerts, packet logs, user activity, and long-term analytics.
- **Real-Time Streaming:** Incorporating frameworks like Apache Kafka for high-volume streaming and near-instantaneous detection on enterprise-scale networks.
- **Integration with IDS Tools:** Combining the system with Snort or Suricata to enrich signature-based and anomaly-based detection.
- **Cloud Deployment:** Deploying the solution on AWS, Azure, or GCP for distributed monitoring and high availability.
- **User Authentication:** Adding secure login, role- based access control, and audit logs to support organizational use.
- **Mobile Alerts:** Extending the system with SMS/email notifications or a dedicated mobile ap- plication for real-time alerts.

These enhancements will further extend the sys- tem's applicability and make it suitable for large-scale, production-ready cybersecurity environments.

REFERENCES

[1] S. Kumar and A. Singh, "Cybersecurity challenges in modern networks," *International Journal of Computer Applications*, vol. 182, no. 25, pp. 10–16, 2020.
[2] M. Reddy and K. Sharma, "Impact of increasing cyber-attacks on global IT infrastructure," *Journal of Information Security*, vol. 12, no. 3, pp. 45–54, 2019.
[3] J. Williams, "Analysis of DoS and Probe attacks in enterprise networks," *IEEE Transactions on Network Security*, vol. 18, no. 4, pp. 233–240, 2018.
[4] P. Thomas and L. George, "Evaluating the rise of R2L and U2R attacks in heterogeneous systems," *IEEE Access*, vol. 7, pp. 155600–155610, 2019.
[5] T. Anderson, "Limitations of traditional intrusion detection sys-tems," *ACM Computing Surveys*, vol. 51, no. 4, pp. 1–28, 2018.
[6] F. Ali and S. Khan, "Machine learning approaches for intrusion detection: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2821–2846, 2019.
[7] Y. Zhao et al., "AI-driven cyber threat analytics: Trends and challenges," *Expert Systems with Applications*, vol. 146, 113199, 2020.
[8] M. Tavallaee et al., "A detailed analysis of the KDD Cup 99 dataset," in *Proc. IEEE Symposium on Computational Intelli- gence for Security and Defense Applications*, 2009.
[9] H. Patel and S. Mehta, "Performance evaluation of Random Forest for intrusion detection," *Procedia Computer Science*, vol. 167, pp. 123–131, 2020.
[10] A. O. Silva and J. Costa, "Real-time packet capturing and analysis using Scapy," *International Journal of Network Management*, vol. 28, no. 6, e2050, 2018.
[11] Smith et al., "Random Forest-based intrusion detection using NSL-KDD," *Journal of Cybersecurity*, 2018.
[12] Lee and Park, "SVM with feature scaling for intrusion detection," *Information Systems Research*, 2017.
[13] Kumar et al., "LSTM models for network traffic analysis," *IEEE Access*, 2019.
[14] Zhang and Wu, "CNN architectures for intrusion detection," *Computers & Security*, 2020.
[15] Patel et al., "Dimensionality reduction using PCA with Random Forest," *Elsevier Future Generation Computer Systems*, 2018.
[16] Wang et al., "Deep neural network for multi-attack detection," *Neurocomputing*, 2021.
[17] Chen et al., "Hybrid SVM and Random Forest for improved IDS," *Pattern Recognition Letters*, 2019.
[18] Ali and Hussain, "Autoencoders for anomaly detection in cyber-

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14, Issue 06, ACSCON - 2026

security," *IEEE Transactions on Information Forensics*, 2020.
[19] Gupta et al., "Decision tree-based intrusion detection on NSL-KDD," *International Journal of Computer Networks*, 2018.

[20] Roy et al., "KNN classifier performance in IDS," *Procedia Computer Science*, 2021.
[21] Singh et al., "Handling class imbalance using SMOTE in IDS," *Applied Soft Computing*, 2019.
[22] Sharma and Rao, "Gradient boosting techniques for attack detec- tion," *IEEE Access*, 2020.
[23] Zhao et al., "CNN-LSTM hybrid model for intrusion detection," *Expert Systems with Applications*, 2021.
[24] Khan et al., "Real-time packet capture with Scapy and Random Forest," *Journal of Network Security*, 2019.
[25] Li et al., "Feature selection with deep neural networks," *Knowledge-Based Systems*, 2020.
[26] Ahmed et al., "SVM + PCA for efficient intrusion detection," *Pattern Analysis and Applications*, 2021.
[27] Tan et al., "Random Forest ensemble for multiclass IDS," *IEEE Transactions on Dependable Computing*, 2018.
[28] Kumar et al., "Attention-based LSTM for real-time anomaly detection," *Information Sciences*, 2020.
[29] Wang et al., "Hybrid CNN-RF for enhanced attack detection," *Computers & Security*, 2019.
[30] Patel et al., "Autoencoder + Random Forest for unsupervised intrusion detection," *Applied Intelligence*, 2021.