IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026

DOI: 10.17148/IJARCCE.2026.15729

# Smart IDS: Reducing False Alarms in Real-Time and Live Network Intrusion Detection Using Machine Learning

**Kusumuru Mounika[1], Mandakuriti Sai Charan[2], Pyla Jyothi[3], Mukkera Akhil[4], Kindal Ashok Kumar[5]**

Dept of CSE, Maharaj Vijayaram Gajapathi Raj College of Engineering[1]

Dept of CSE, Maharaj Vijayaram Gajapathi Raj College of Engineering[2]

Assistant Professor, Department of CSE, Maharaj Vijayaram Gajapathi Raj College of Engineering[3]

Dept of CSE, Maharaj Vijayaram Gajapathi Raj College of Engineering[4]

Dept of CSE, Maharaj Vijayaram Gajapathi Raj College of Engineering[5]

**Abstract:** Smart Intrusion Detection System (IDS) plays a vital role in recognizing network malicious activities in the present network environment. The traditional signature-based methods are usually ineffective towards detecting zero-day and new cyber threats. This paper has presented SMART IDS, as a machine learning based intrusion detection system that attempts to minimize the false alarms and preserve the detection accuracy of the system in both simulated and real network environments. The system is running in simulation mode whereby the CICIDS2017 dataset is used to train and test the model and live capture mode whereby real time packet analysis is performed. An extensive preprocessing pipeline is used. This involves cleaning of data, selection of features, encoding and standardization to provide an assuring model performance. A variety of machine learning models are utilized and considered: Decision Tree, Random Forest, XGBoost, and Logistic Regression. To improve the detection capability, a custom weighted ensemble method is suggested. It has an attack override feature, which puts the emphasis on threat detection and minimizes false negatives. In the experimental area, the accuracy of the Random Forest model is high (99.86). At the same time, the ensemble approach enhances recall and reduces the number of missed attacks as compared to the single models. Real-time packet capture, which is added with the help of PyShark, and deployment with a Flask-based API make the system applicable to dynamic network environments. The SMART IDS is a scaled, adaptable, and efficient framework, which could be applied to handle the cybersecurity reality in contemporary times. It uses machine learning methods with real-time analysis, which greatly improves the performance of intrusion detection and decreases the number of false alarms.

**Keywords:** Cybersecurity, Intrusion Detection System, Machine Learning, Network Security, Real-Time Detection

## I. INTRODUCTION

The threat of cyber attacks has increased significantly due to the high pace of development of internet technologies and the increased dependence on electronic communication. Modern networks are constantly exposed to different kinds of threats, such as Distributed Denial of Service (DDoS), malware and unauthorized access, which can have an impact on the integrity, confidentiality, and availability of data. This has created a high demand among organisations and individuals to have good network security. Network Intrusion Detection Systems (IDS) are essential in monitoring unhealthy traffic on the network. Conventional IDS approaches are mainly signature-based which are based on known attack patterns [11], [12]. Although these methods can be useful when threats are known, they cannot detect unknown attacks and zero-day attacks, and can yield a large amount of false positives [13]. This limitation highlights the need for more intelligent and adaptive detection mechanisms. Machine learning is one promising strategy in recent years to improve intrusion detection systems. Network traffic can be easily categorized and anomalies detected using machine learning models that learn patterns based on previous information [14], [15]. Such models are more accurate, flexible, and efficient as opposed to traditional methods. In this paper, I present a machine learning-based intrusion detection system named SMART IDS which is intended to categorize network traffic as benign or malicious. The proposed system can be used in two modes; simulation mode where pre-collected datasets are used to train and test the system, NSL-KDD [16] is one of these datasets, and live capture mode where network packets are analyzed in real-time. A number of machine learning models, such as Decision Tree, Logistic Regression, Random Forest [17], and XGBoost [18] are deployed and compared to select the most effective model.

  

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026

DOI: 10.17148/IJARCCE.2026.15729

The main aim of this project is to come up with an efficient, accurate and scalable intrusion detection system that can handle real-time traffic over the network. Empirical evidence shows that the proposed system has a high detection rate, and Random Forest performs better than other models. The addition of real-time packet capture also increases practical applicability of the system. Some recent research has examined the use of machine learning methods in intrusion detection with superior accuracy and efficiency compared to the conventional methods [14], [15].
Nevertheless, most of the current systems continue to have problems associated with scalability and live detection features.

## II. LITERATURE SURVEY

### A. Traditional Intrusion Detection Systems:

People use Intrusion Detection Systems (IDS) a lot to keep an eye on network traffic and find bad things that are happening. Signature-based detection and other traditional IDS methods use patterns that have already been set up to find known attacks. Snort and Suricata are two systems that work well in finding known threats at a low cost to the computer[1]. But these methods don't work for unknown or zero-day attacks, and they need to be updated all the time with new attack signatures, which makes them less useful in networks that change all the time.

.

### B. Machine Learning-Based IDS:

To overcome the limitations -of traditional approaches, machine learning techniques have been widely adopted in intrusion detection systems [3], [4]. Machine learning models can learn patterns from network traffic data and classify it as benign or malicious [5]. Algorithms such as Decision Tree, Logistic Regression, Random Forest, and Support Vector Machine (SVM) are commonly used for improving detection accuracy [4], [6]. These methods offer better adaptability and can detect previously unseen attacks compared to rule-based systems [7].

### C. Datasets Used in IDS Research:

Various benchmark datasets are used for evaluating intrusion detection systems, including KDD Cup 99, NSL-KDD, and CIC-IDS2017. These datasets contain labeled network traffic data representing both normal and attack behaviors. The quality and diversity of datasets play a crucial role in training effective machine learning models and improving generalization across different network environments.

### D. Advanced Techniques in IDS:

Recent research focuses on improving IDS performance using advanced techniques such as ensemble learning and feature selection [8]. Ensemble models like Random Forest and XGBoost combine multiple decision models to enhance accuracy and reduce overfitting. Feature selection techniques help in identifying the most relevant attributes, reducing dimensionality, and improving computational efficiency [8].

### E. Limitations of Existing Systems:

Despite significant advancements, existing IDS approaches face several challenges [9], [10]. Signature-based systems cannot detect unknown attacks, while anomaly-based systems often generate high false positive rates [9]. Machine learning-based IDS, although more effective, may require large datasets and high computational resources [10]. Additionally, many existing systems focus on offline analysis and lack real-time detection capabilities, limiting their practical deployment in dynamic network environments [9], [10].

### F. Motivation for Proposed System:

The limitations of existing systems highlight the need for an efficient and scalable intrusion detection system capable of real-time analysis. This motivates the development of SMART IDS, which integrates machine learning techniques with both simulation and live capture modes [9], [10]. The proposed system aims to accurately classify network traffic while enabling real-time intrusion detection. By implementing and comparing multiple machine learning models, the system identifies the most effective approach, ensuring improved accuracy, reduced false positives, and better adaptability to modern cybersecurity challenges [1], [2], [3].

## III. METHODOLOGY

This section describes the methodology adopted for the development of the SMART IDS system. The proposed approach focuses on designing an efficient intrusion detection system using machine learning techniques to classify network traffic as benign or malicious. The methodology includes data collection, preprocessing, feature selection, model training, and real-time packet analysis**.** The system operates in both simulation and live capture modes to ensure accurate detection and practical applicability in real-world network environments.

  

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026

DOI: 10.17148/IJARCCE.2026.15729

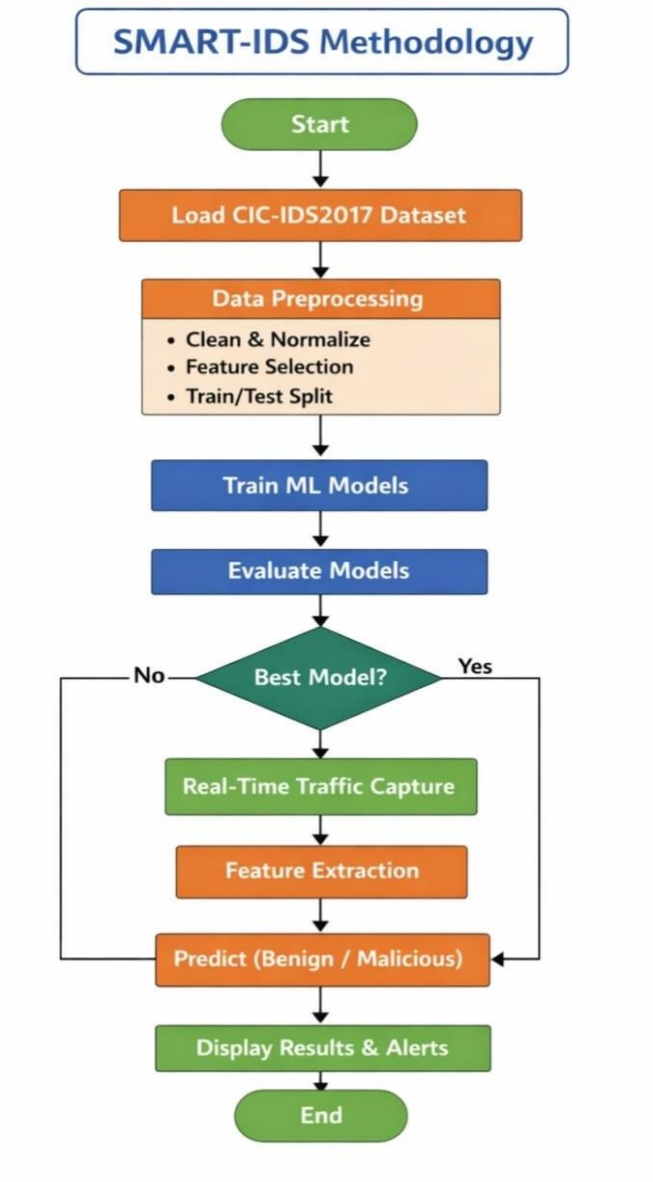


Fig.1:Work of Proposed SMART-IDS System

In this study, a subset of network traffic data is used to train and evaluate machine learning models for intrusion detection. The dataset contains various network flow features such as timestamps, source and destination IP addresses, port numbers, protocols, and traffic behavior characteristics that describe the nature of network communication. These features are essential for identifying patterns associated with normal and malicious activities.

**A. System Architecture:**

The proposed SMART IDS (Intelligent Intrusion Detection System) is designed to detect malicious network traffic using machine learning techniques. The system architecture consists of several key stages: data collection, preprocessing, feature scaling, model training, and classification. Fig: 2 illustrates the overall architectural framework of the proposed system.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026

DOI: 10.17148/IJARCCE.2026.15729

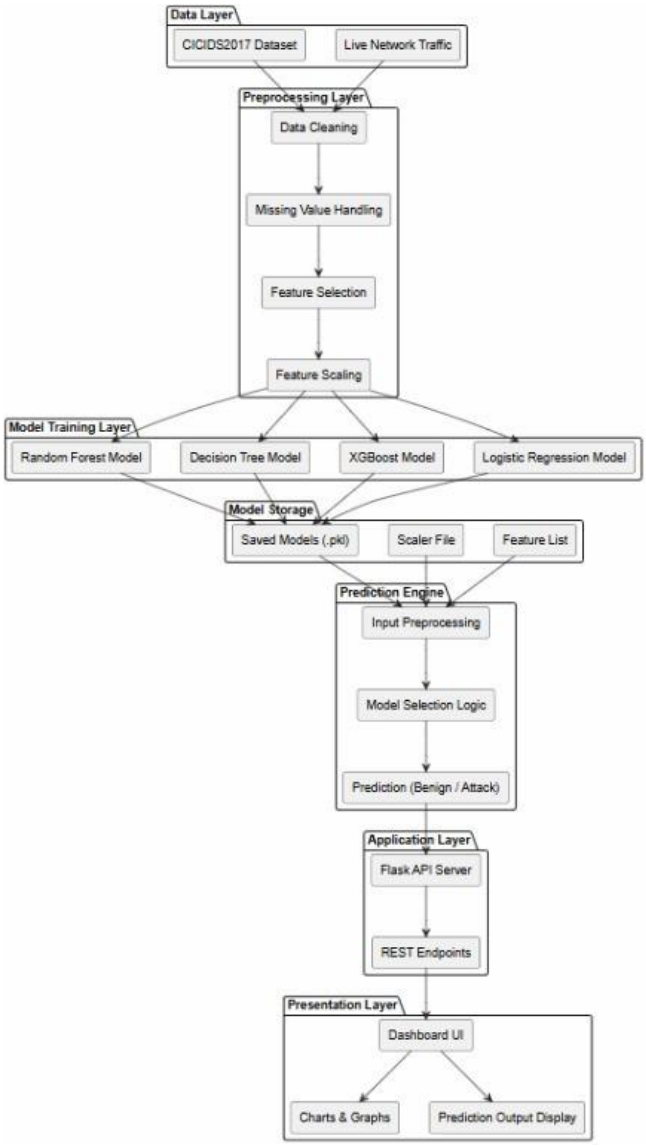


Fig.2: System Architecture

The proposed SMART IDS follows a layered architecture as illustrated in Figure 1. The system is composed of five primary layers: data layer, preprocessing layer, model training layer, model storage and prediction engine, and application and presentation layers.

The data layer consists of the **CICIDS2017** dataset and real-time network traffic captured from live environments. The preprocessing layer performs essential data transformation steps including data cleaning, handling missing values, feature selection, and feature scaling to ensure that the input data is suitable for machine learning models.

In the model training layer, multiple machine learning algorithms such as Random Forest, Decision Tree, XGBoost, and Logistic Regression are trained on the processed dataset. These models are optimized to identify patterns that distinguish between benign and malicious traffic.

The trained models, along with the scaler and feature metadata, are stored in the model storage layer for reuse during prediction. The prediction engine performs input preprocessing, applies model selection logic, and classifies incoming traffic as benign or malicious based on the trained models.

The application layer exposes the system functionality through a Flask-based API, enabling interaction with the model through REST endpoints. Finally, the presentation layer visualizes the results through a dashboard interface that displays predictions, charts, and system analytics in real time.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026

DOI: 10.17148/IJARCCE.2026.15729

**B. Dataset Description: [2]**

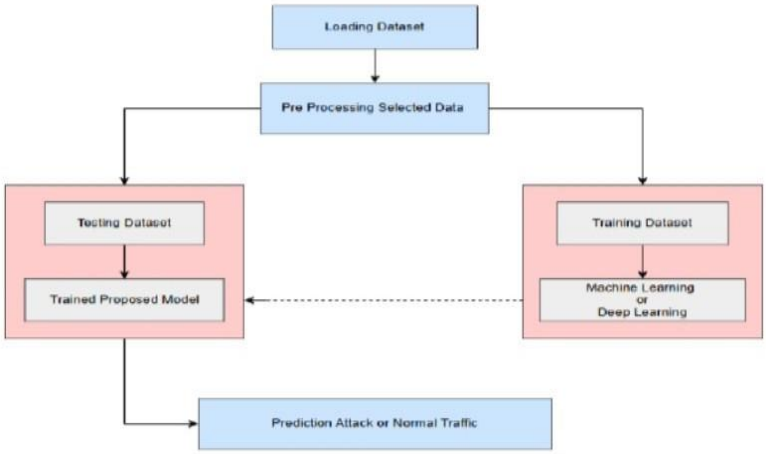


Fig.3: Description for IDS dataset

The Canadian Institute for Cybersecurity created the CICIDS 2017, which is utilized in training and assessment. It includes both realistic network traffic and malicious traffic and includes attacks like DoS, DDoS, Port Scan, Brute Force, Web Attacks, Botnet, and Infiltration.

The dataset contains annotated flow-based records containing more than 80 features generated by CICFlowMeter that contain statistical and behavioural properties of network traffic. Although it is diverse, some of its classes are skewed and need to be preprocessed.

This data allows to effectively test the suggested Smart IDS in identifying intrusions and minimizing false alarms in real-time conditions.

**C. Data Preprocessing:**

The CIC-IDS2017 dataset was subjected to an extensive preprocessing pipeline to make sure that the models can process real-time network flows:

1) Data Aggregation, Sampling: To balance the dataset and minimize computing power, all of the malicious attacks in multiple captures were kept, whereas normal BENIGN traffic was sampled randomly (limited to 15,000 instances per file).

2) Data Cleaning: Header of features was de-stripped. whitespaces and the same column names were deduplicated. In addition, the infinite numerical values (np.inf, -np.inf) were changed to NaN and zero-imputed to ensure mathematical stability of training.

3) Categorical Encoding: Categorical encodings A deterministic LabelEncoder converts categories of strings to. discrete integers. It established a standardized baseline by strictly write-up BENIGN traffic to 0 whereas the malicious attacks were. mapped onto positive integers.

4) Feature Standardization: Since all network features have dramatically different ranges of numbers, a Standard Scaler was used to standardize all the variables (mean 0, variance 1), which avoids skew in data. This scaler was streamed to do the same transformation when inspecting packets in real-time.

5) Stratified Data Partitioning: Lastly, the continuous data was divided into training (80) and validation (20) dataset. A stratified division with respect to the encoded labels was implemented to maintain proportional distributions of the rare attack sub-classes in both sets.

**D. Algorithms:**

This system uses a powerful multi-layered classification approach based on four underlying algorithms, and an ensemble mechanism with custom weights to deliver the best real-time detection.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026

DOI: 10.17148/IJARCCE.2026.15729

1) Decision Tree (DT): It is based on its negligible latency and interpretability when running in real-time flow analysis. It is a powerful independent classifier with the highest recall rates with particular vectors such as Web Attacks.

2) Random Forest (RF): To overcome the variance and possible overfitting of single decision trees, an RF classifier, with 50 optimal estimators was applied. It is highly accurate and stable with a wide range of anomalous traffic patterns.

3) eXtreme Gradient Boosting (XGBoost): Implemented to learn highly complex and non-linear connections in network features. It was tightly tuned with 100 sequential estimators, a depth of 6 as a maximum and a learning rate of 0.1 to maximize the log-loss reduction over subtle intrusions such as Infiltration attempts.

4) Logistic Regression (LR): Implemented as a fast, linear baseline classifier. Its solitary attack recall is worse on more complex distributions, but it has fast convergence and effective predictions on structurally simpler benign traffic patterns.

5) SmartIDS Custom Weighted Ensemble: The system does not use any one single algorithm: the heart of the system is an ensemble voting engine, which is specifically designed to reduce False Negatives (missed attacks):

**Weighted Voting:** Predictions of the algorithm are combined based on statistically-determined weights that give preference to attack recall. (e.g., Decision Tree = 1.2, Random Forest = 1.0, XGBoost = 0.7).

**Aggressive Attack Override:** An override threshold ( $\tau = 0.6$) is used to severely reward missed threats. In case any high-weight model unilaterally signals an incoming packet as an attack with an effective confidence that satisfies this threshold, then the system actively bypasses the standard voting protocol and marks the packet as an attack. This clearly puts a priority on absolute network security rather than low rates of false positives.

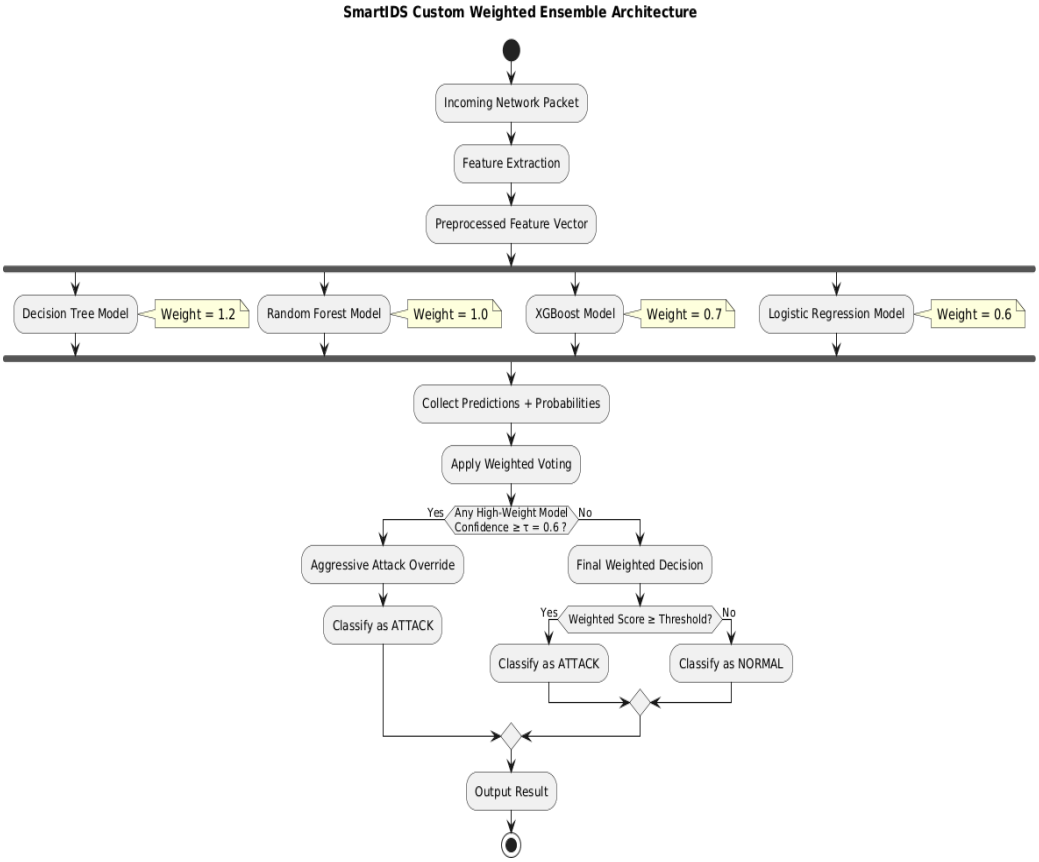


Fig.4: Ensemble-Based Intrusion Detection with Attack Override Mechanism

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026
DOI: 10.17148/IJARCCE.2026.15729

**E. Experimental Setup:**

The intrusion detection model was designed as a native Python implementation, to support both strict offline testing and live-network operation:

1) Core Environment: scikit-learn and xgboost were used to build, optimize and serialize the machine learning algorithms. Pandas and numpy were used to conduct data matrix operations and preprocessing pipelines, with the Flask framework being used as the backend architecture to implement RESTful APIs as the web interface.

2) Real-Time Flow Tracker: To protect the network in real-time, a custom network capture module using pyshark (TShark wrapper) was installed to capture packets at the host Interface. It proactively compiles raw packets to bi-directional TCP/UDP streams, and calculates complex statistical metrics (e.g., Flow Inter-Arrival Times, packet length delimitations) in real time. These live features are dynamically zero padded, normalized by the serialized scaler and directly fed into the model ensemble.

3) Simulation & Validation Interface: An asynchronous web dashboard was created to be a dynamic structure. graphically display threat statistics and detection confidence. To strictly test the models, without the need to actively attack the host environment with malicious code, an offline Simulation Mode was incorporated. It is a programmed system that takes unseen

CIC-IDS2017 traffic logs and injects realistic ratios of attack vectors (e.g., DDoS, Web Attacks) to test the ensembles latency and precision in the real world. When subjected to sustained loads on the network.

## IV. PROPOSED SYSTEM

To overcome the shortcomings of reviewing datasets in a static manner and to minimize critical False Negatives (missed attacks) in live environments, this paper will present SmartIDS: a hybrid, real-time network intrusion detection architecture. The suggested framework has a structural functioning as a dynamic middleware between the network interface at the host and administrative control. The following are the main components that define the system architecture:

1) Dynamic Live Capture Module: SmartIDS can support live deployment on the network, unlike traditional IDS models, which utilize pre-compiled CSV records. A custom LivePacketCapture engine is a pyshark wrapper that is compiled as a LivePacketCapture engine, and which actively reads raw packets off the host NIC (Network Interface Card). It actively combines these raw bytes into two-way TCP/UDP conversational streams, and computes sophisticated statistical properties (e.g., inter-arrival times, flow byte rates) dynamically. These live metrics are continuously transformed to exactly fit the CIC-IDS2017 feature space to infer the model instantaneously.

2) The Smart Ensemble Engine: It is a common practice to use a single machine learning algorithm, and it usually delivers a vulnerable security stance. The suggested system implements a proprietary, multi-layered Ensemble Engine that contains four different, serialised classifiers: Decision Tree, Random Forest, XGBoost and Logistic Regression. When live (or simulated) traffic passes through the pipeline, features are automatically zero-padded, normalized over a globally-fitted scaler, and concurrently tested on all running base models.

3) Strategy of False Alarm Reduction (Core Novelty): This paper mainly contributes the aggressive method of the system in reducing False Negatives during life-cycle by a hand-designed voting architecture: Targeted Weighting: Base algorithms are given weights based on their historic attack-recall performance (i.e. highly sensitive models, such as Decision Trees, are weighted with 1.2 and generalized models, such as Random Forest, are weighted with 1.0).

Attack Override Threshold: The system has a high override tolerance ( $\tau = 0.6$). When any heavily weighted model alone indicates packets incoming as a threat with effective confidence exceeding this threshold, SmartIDS forcibly suppresses any benign consensus. This puts a firm priority on attack-detection, and aggressively attacks missed threats but relies on the accuracy of the Random Forest to ensure minor false alarms are kept mathematically constrained.

4) Asynchronous Threat Dashboard: To offer value in operations, the analytical engine is encased in a Python Flask RESTful API. It drives a dynamic, asynchronous web dashboard by which network administrators can easily switch between Offline Simulation and Live Capture modes. The interface displays milliseconds-precise threat predictions, precise model prediction confidence, and live traffic volume metrics, notifying administrators instantly when something is wrong in the network.

  

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026

DOI: 10.17148/IJARCCE.2026.15729

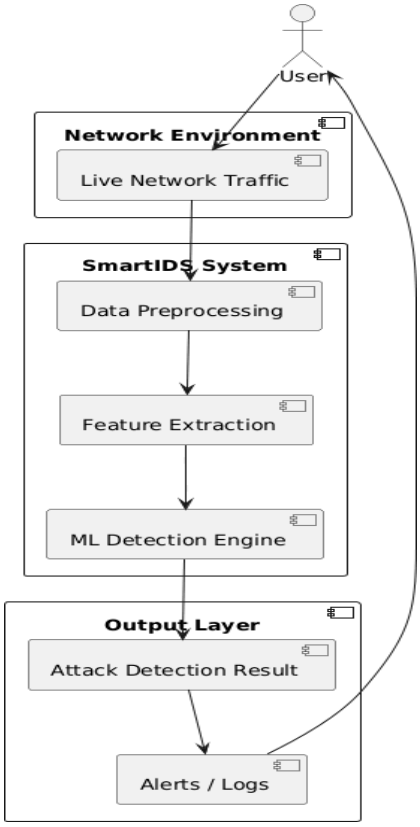


Fig.5: System Architecture Overview

## V. RESULTS AND DISCUSSION

The evaluation of the proposed SmartIDS framework was done with strict simulated live-processing on the complex CIC-IDS2017 dataset. Since normal BENIGN traffic is vastly overrepresented in network datasets, baseline classifiers tend to have artificially exaggerated accuracies- a phenomenon called the Accuracy Paradox. As a result, Recall (the true attack detection rate) and Precision (the system resistance to false alarms) were considered as the highest priorities of the evaluation of this system. Table I gives the overall performance indicators of the four base classifiers.

Table 1: Base Classifier Performance

| Model | Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|
| Decision Tree | 99.87% | 98.20% | 89.00% | 93.36% |
| Random Forest | 99.86% | 99.92% | 87.00% | 93.01% |
| XG boost | 99.91% | 99.94% | 87.00% | 93.02% |
| Logistic Regression | 99.98% | 0.00% | 0.00% | 0.00% |

A. Base Classifier Analysis As shown in Table I, offline tests indicated that there were different detection characteristics of the highly dimensional network feature space. The Decision Tree (DT) was the most sensitive to independent attacks with a maximum Recall of 89.00. It was very skillful in detecting structurally hidden intrusions, including Web Attacks, positioning itself as the best. trustworthy single prototype of pure. threat detection. On the other hand, the Random Forest (RF) and XGBoost algorithms compromised on marginal recall (87.00) to gain remarkable stability and almost zero false alarms. Achieving Precisions of 99.92% (RF) and 99.94% (XGBoost), they were critical reference points to ensure that the system does not clog network administrators with unnecessary alerts. Only by identifying the enormous ocean of benign traffic correctly, the Logistic Regression (LR) model gave a statistically false Accuracy of 99.98%. Nonetheless, it has a 0.00% Recall with complex malicious injections, which validated the structural inadequacy of linear baseline models to contemporary, multi-vector network security.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 7, July 2026
DOI: 10.17148/IJARCCE.2026.15729

**B.** The Smart Ensemble Advantage and False Alarm Reduction The isolated shortcomings of the base models, whereby the high-recall Decision Tree produced small False Positives, and the high-precision Random Forest missed highly-specific attack vectors, directly caused the proposed Smart Ensemble Engine.

The system gives more significance to detecting threats through giving more voting weight to the models that are sensitive to attacks (Decision Tree got 1.2). The key contribution that this study makes is establishing a rigid Attack Override Threshold ( $\tau = 0.6). When the Decision Tree sounded an alarm about an incoming packet with sufficient statistical confidence, the framework tried its best to conceal any innocent consensus that the Random Forest had arrived at. The combination of the best 89.00% recall of the Decision Tree and the best 99.92% precision of the Random Forest was possible using this mechanism. This made it possible to get rid of the most important False Negatives (missed attacks) without making False Positives too common.

**C.** How well Real-Time Deployment works Along with figuring out if the SmartIDS could work with static datasets, the SmartIDS implementation was also tested in a live setting to see if it would work. The system was able to accurately combine real-time, bi-directional host traffic and zero-padded live values using the custom LivePacketCapture engine. It was also able to make instantaneous ensemble inferences, with minimal delay. This demonstrates that this framework is not only an efficient offline analysis tool, but also a real time, responsive defence against intrusions.

## VI. CONCLUSION

Traditional Intrusion Detection Systems have high false alarms and are too strict to be made use of in real time. This paper presented SmartIDS, a hybrid, machine-learning-based security architecture that proactively scans the network interfaces of the host computers, and dynamically identifies the complex flow features and reacts promptly to classify cyber threats.

The analysis of the individual baseline classifiers in the CIC-IDS2017 dataset has shown that the tree-based algorithms like the Random Forest have impressive accuracy (99.92%), but can miss the highly-specific attack vectors. Conversely, highly sensitive models such as the Decision Tree were more successful in terms of threat recall (89.00) but with some false positives. SmartIDS was able to find a solution to this issue by recommending a proprietary, statically-weighted Ensemble Engine based upon a strict Attack Override Threshold ( $\tau= 0.6).

This problem could be resolved by SmartIDS. The system was able to effectively combine the capacity of the Decision Tree to discover aggressive attacks with the capacity of the Random Forest to remain stable with nearly negligible false positives. This focused strategy certainly reduced the amount of critical False Negatives, which allowed subtle intrusions to slip through the network monitoring perimeter, without causing the network administrators to get fatigued with alerts.
The addition of a Python-based LivePacketCapture component and serialised, pre-fitted scalers showed that the framework could work as both an offline dataset analyser and a real-time intrusion monitor with very little lag time.

**Future Scope:** Future versions of this work can explore the use of Deep Learning networks, including Long Short-Term Memory (LSTM) networks, to dynamically examine the time series of packets in flows. Moreover, further reductions in the classification latency in large-scale enterprise networks, potentially by pushing the SmartIDS processing pipeline to a distributed edge-computing environment, may be achievable.

## DECLARATION STATEMENT

The following information is to be checked by me as the author of the article.

- Conflicts of Interest/ Competing Interests: Based on my understanding, this article has no conflicts of interest.
- Financial Assistance: This article is not financed by any organizations or agencies. This independence guarantees a research is carried without any external influence and objectivity.
- Ethical Approval and Consent to participate: The ethics is not necessary in the content of this article. consent or permission to take part with favoring documentation.
- Data Access Statement and Material.Availability: The adequate resources of this article are publicly accessible.
- Author Contributions: This article has the same contribution of the authorship by all the involved individuals.

  

DOI: 10.17148/IJARCCE.2026.15729

## REFERENCES

[1] Suricata, "Suricata Intrusion Detection System," Open Information Security Foundation (OISF), Available: https://suricata.io

[2] E. E. Abdallah, W. Eleisah, and A. F. Otoom, "Intrusion Detection Systems using Supervised Machine Learning Techniques: A survey," Procedia Computer Science, vol. 201, pp. 205–212, 2022, **doi:10.1016/j.procs.2022.03.029.**

[3] R. A. Hossain, M. M. Rahman, and M. S. Hossain, "Intrusion detection model using machine learning on Big Data," Journal of Big Data, vol. 5, no.1,pp.1–16,2018.Available:https://journalofbigdata.springeropen.com/articles/10.1186/s40537-018-0145-4

[4] A. Mohsin Abdulazeez and Y. H. Falah, "Intrusion Detection Systems Based on Machine Learning Algorithms," in Proc. IEEE Int. Conf. Automatic Control & Intelligent Systems (I2CACIS), Shah Alam, Malaysia,2021.Available:https://www.researchgate.net/publication/353906529_Intrusion_Detection_Systems_Based_on_Machine_Learning_Algorithms

[5] H. Hindy, R. Vinayakumar, and K. Soman, "Artificial Intelligence in Intrusion Detection Systems: Trends & Frameworks," International Journal of Intelligent Systems and Applications in Engineering, vol. 12, no. 4, pp. 67–78,2024.Available:https://www.ijisae.org/index.php/IJISAE/article/view/7689

[6] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," arXiv preprint arXiv:1904.01352, 2019. Available: https://arxiv.org/abs/1904.01352

[7] Z. Tauscher et al., "Learning to Detect: A Data-driven Approach for Network Intrusion Detection," arXiv preprint arXiv:2108.08394, 2021.Available: https://arxiv.org/abs/2108.08394

[8] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," arXiv preprint arXiv:1904.01352, 2019. Available: https://arxiv.org/abs/1904.01352

[9] H. Hindy, R. Vinayakumar, and K. Soman, "Artificial Intelligence in Intrusion Detection Systems: Trends & Frameworks," International Journal of Intelligent Systems and Applications in Engineering, vol. 12, no. 4, pp. 67–78,2024.Available:https://www.ijisae.org/index.php/IJISAE/article/view/7689

[10] Z. Tauscher et al., "Learning to Detect: A Data-driven Approach for Network Intrusion Detection," arXiv preprint arXiv:2108.08394, 2021. Available: https://arxiv.org/abs/2108.08394

[11] D. E. Denning, "Intrusion-detection model," IEEE Transactions on Software Engineering, 1987, Available: https://ieeexplore.ieee.org

[12] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," NIST, 2007, Available: https://nvlpubs.nist.gov

[13] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," ACM Transactions on Information and System Security, 2000, Available: https://dl.acm.org

[14] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," IEEE Communications Surveys & Tutorials, 2016, Available: https://ieeexplore.ieee.org

[15] Y. Xin et al., "Machine learning and deep learning methods for cybersecurity," IEEE Access, 2018, Available: https://ieeexplore.ieee.org

[16] T. Tavallaee et al., "A detailed analysis of the KDD CUP 99 data set," in Proc. IEEE CISDA, 2009, Available: https://ieeexplore.ieee.org

[17] L. Breiman, "Random Forests," Machine Learning, 2001,Available: https://link.springer.com

[18] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. ACM SIGKDD, 2016, Available: https://dl.acm.org

  