Real-Traffic-Trained Intelligent IDS for Advanced Cyberattack Detection in Enterprise Networks

Dalila Naira Chinchay, Rodrigo Calderón Ari, Liset S. Rodriguez-Baca
Professional School of System Engineering, Cesar Vallejo University, Lima, Peru

***Abstract*—Early detection of cyberattacks remains a major challenge in enterprise networks due to encrypted traffic, protocol diversity, and highly dynamic service behavior. This study evaluates a machine learning-based intrusion detection system trained on real enterprise traffic captured over 20 working days under operational conditions. A total of 1,163,014 packets were collected and complemented with controlled attack traffic, including DDoS, brute force, botnet, SQL injection, port scanning, privilege escalation, and service exploitation scenarios. After flow-based feature extraction and preprocessing, six supervised learning models were evaluated under the same data partition and validation settings. Among them, Random Forest achieved the best overall performance, with precision, recall, and F1-score above 0.999 and an AUC of 0.9994 on the collected dataset. These findings suggest that training with real traffic can improve IDS performance under realistic enterprise conditions. However, further validation across additional organizations and time periods is required to confirm generalizability.**

***Keywords*—IDS; machine learning; cyberattacks; cybersecurity; intrusion detection**

I. Introduction

The growing reliance of organizations on digital infrastructures has intensified their exposure to increasingly sophisticated and hard-to-anticipate cyberattacks. In recent years, threats based on evasion techniques, zero-day vulnerability exploitation, and malicious automation have proven capable of easily bypassing traditional security systems. These challenges have driven the evolution of Intrusion Detection Systems (IDS), which aim to identify anomalous behaviors or attack-related patterns with ever-higher levels of accuracy.

Recent literature agrees that approaches based on artificial intelligence and machine learning offer significant advantages for early intrusion detection, particularly when dealing with unknown attacks or rapidly evolving variants. Studies such as that of Shamshair et al. [1] highlight that AI-based models can adapt to emerging threats with a level of generalization superior to mechanisms relying solely on signature-based detection. Similarly, applied research in IoT environments and enterprise networks shows that advanced techniques based on deep learning and graph-based methods have achieved improved detection of complex malicious behaviors [2], [3].

Despite these advances, a key limitation remains: most existing studies rely on synthetic or historical datasets that do not adequately represent the real behavior of modern networks. Widely used datasets such as KDD99, NSL-KDD, or UNSW-NB15 exhibit artificial structures, obsolete protocols, or traffic volumes that differ substantially from real traffic captured in an operational organization. As a result, models that achieve high performance in controlled environments often lose effectiveness when deployed in networks dominated by encrypted protocols, highly variable flows, and distributed application workloads.

Recent systematic reviews confirm this gap. Rehman et al. [4] emphasize that machine learning–based IDS requires datasets that are representative of current network traffic, while Chen et al. [5] underline that model performance largely depends on the quality of the data used during training. Although recent studies have reported promising IDS results using machine learning and deep learning, many of them rely on public benchmark datasets or controlled laboratory environments. These settings often underrepresent encrypted enterprise traffic, fluctuating service workloads, and the coexistence of modern protocols such as QUIC and TLS/HTTPS. As a result, it remains unclear to what extent the reported performance of these models can be sustained under real operational conditions.

The main contributions of this study are threefold: 1) the construction of a hybrid dataset that combines real enterprise traffic with controlled attack scenarios; 2) the comparative evaluation of six supervised machine learning models under identical preprocessing, partitioning, and validation settings; and 3) empirical evidence that real-traffic-based training can support highly accurate intrusion detection under realistic enterprise network conditions.

II. Related Work

Intrusion detection has been widely studied within the field of cybersecurity, and its evolution has been marked by the progressive incorporation of machine learning techniques. Over the past decade, numerous studies have compared traditional algorithms such as decision trees, support vector machines, and k-nearest neighbors with more recent models based on ensemble methods or deep learning, demonstrating significant improvements in accuracy, predictive capability, and the detection of hidden patterns.

Shamshair et al. [1] conducted a comparative analysis of various AI-based methods applied to zero-day attack detection, demonstrating that ensemble-based models such as Random Forest and Gradient Boosting exhibit greater stability under traffic variations and achieve better performance in real-world scenarios. In IoT environments, where network behavior tends to be more dynamic and unpredictable, Villegas et al. [2] introduced an innovative approach based on dynamic graphs and neural networks, achieving high accuracy in the identification of

complex attacks. Complementarily, Villegas et al. [3] evaluated the effectiveness of an adaptive deep learning–based IDS, reporting substantial improvements against multi-stage attacks and low-frequency malicious patterns.

A systematic review conducted by Rehman et al. [4] highlights the advantages of machine learning techniques and their applications in IDS, reinforcing arguments related to current trends and existing research gaps. In line with this perspective, Ferrag et al. [6] emphasize that training with real data strengthens the model's generalization capability and reduces the occurrence of false positives, which remain one of the main challenges of traditional IDS. Similarly, Chen et al. [5] propose a hybrid NIDS + HIDS system that improves detection performance by combining network- and host-based features, representing a modern approach to machine learning–based IDS.

Studies focused specifically on intrusion detection have also examined the comparative behavior of supervised classifiers such as Random Forest, Decision Tree, KNN, AdaBoost, and XGBoost under different traffic conditions. In general, ensemble-based models tend to provide more stable results than single classifiers when the data exhibit variability, noise, or class imbalance.

Likewise, Jijo et al. [7] provide a comprehensive review of the decision tree algorithm, highlighting its importance as a foundation for advanced models such as Random Forest and Extra Trees. In the context of critical infrastructures, Majidi et al. [8] applied ensemble techniques and autoencoders for intrusion detection in power grid networks, demonstrating their robustness against false data injection attacks.

From an algorithmic perspective, Ding et al. [9] proposed an improved AdaBoost variant based on multiple thresholds, which enabled a reduction in overfitting when dealing with highly complex datasets. Complementarily, Zhang [10] analyzed the challenges faced by the KNN algorithm in imbalanced data environments, while Zhang, Jia, and Shang [11] explored the potential of XGBoost for handling severely imbalanced datasets, concluding that its performance can be significantly enhanced through specific weighting and regularization adjustments.

Vaarandi et al. [12] investigated the use of active learning to classify IDS alerts in real time, highlighting the advantages of incorporating adaptive methods in modern network environments. In addition, hybrid approaches have been shown to improve detection performance by integrating network-based and host-based analysis.

Hassan et al. [13] propose ensemble-based methods (boosting) combined with hyperparameter optimization and feature selection for NIDS, aligning with comparative approaches to machine learning models. Rahman et al. [14] provide an updated review of IDS techniques, machine learning models, and performance metrics in IoT environments, which is useful for contextualizing the application of machine learning in networks with real traffic.

Ba et al. [15] explore the application of machine learning models, including Random Forest and Decision Tree, for intrusion detection in industrial IoT networks, highlighting the importance of training with real data and conducting comparative evaluations of machine learning models.

Spiekermann et al. [16] examine specific intrusion detection challenges when traffic originates from virtualized networks, highlighting the need for deep learning techniques to address highly dynamic and encapsulated traffic.

Similarly, Adenusi et al. [17] conducted an extensive comparison of optimized machine learning algorithms for IDS, highlighting the advantages of ensemble and boosting methods over individual models, which supports our comparative selection of six models.

Zakariah et al. [18] conducted a comprehensive comparison of the proposed model against widely recognized industry benchmark methods, including decision-based classifier combinations and machine learning classifiers such as K-Nearest Neighbors (KNN), Deep Learning (DL), Support Vector Machine (SVM), Long Short-Term Memory (LSTM), Deep Neural Network (DNN), and ANN. The results show that the proposed model consistently outperforms these traditional techniques across all evaluation metrics. Table I is the comparative summary of recent IDS studies.

TABLE I. COMPARATIVE SUMMARY OF RECENT IDS STUDIES

Ref.	Traffic Source / Dataset	Context	Models/Approach	Key Finding	Main Limitation
[1]	Zero-day attack datasets	Cyberattack detection	AI and ensemble methods	Ensemble models showed strong stability and detection capability	Not based on live enterprise traffic
[2]	IoT network traffic	IoT	Dynamic graphs and GNNs	Effective for detecting complex attacks	Focused on IoT, not enterprise office traffic
[3]	Adaptive IDS traffic	IoT / adaptive IDS	Deep learning-based IDS	Improved detection of multi-stage and low-frequency attacks	Not a real enterprise comparative study
[4]	Multiple IDS datasets	Systematic review	ML-based IDS review	Dataset quality strongly affects IDS performance	Review only; no direct enterprise experiment
[5]	Hybrid host and network data	Hybrid NIDS + HIDS	Two-stage hybrid classifier	Improved detection using host and network features	Different from pure flow-based enterprise NIDS
[6]	Multiple cybersecurity datasets	Comparative review	Deep learning for IDS	Data quality affects generalization and false positives	Not validated on live enterprise capture
[8]	Smart grid intrusion data	Critical infrastructure	Extra Trees, Autoencoder	Robust performance in FDI detection	Domain-specific traffic

Ref.	Traffic Source / Dataset	Context	Models/Approach	Key Finding	Main Limitation
[9]	Complex classification datasets	General ML	Improved AdaBoost	Reduced overfitting in complex settings	Not specific to enterprise IDS
[10]	High-dimensional datasets	General ML	KNN	KNN faces challenges in high-dimensional data	Algorithm-focused, not IDS-specific validation
[11]	Imbalanced datasets	General ML	XGBoost	Good performance with tuning and weighting	Not centered on real enterprise traffic
[12]	IDS alert streams	Real-time alert classification	Active learning	Adaptive learning improves alert classification	Focuses on alerts, not full flow-based IDS
[13]	NIDS datasets	Network intrusion detection	XGBoost + feature selection	Boosting improves NIDS performance	No live enterprise traffic validation
[14]	IoT IDS literature	IoT	Survey of IDS techniques	Updated overview of IDS methods and metrics	Survey only; IoT-centered
[15]	IIoT intrusion data	Industrial IoT	RF, DT, and ML methods	ML is effective for IIoT intrusion detection	IIoT differs from enterprise mixed traffic
[16]	Virtualized network traffic	Virtual networks	Deep learning	Useful for dynamic encapsulated traffic	Limited transfer to enterprise production traffic
[17]	Network intrusion datasets	Data-driven IDS	Optimized ML algorithms	Ensemble methods outperform several single models	Limited evidence on live enterprise capture
[18]	NSL-KDD	Benchmark dataset	KNN, SVM, DL, LSTM, DNN, ANN	Proposed model outperformed common classifiers	NSL-KDD is not representative of modern enterprise traffic
[19]	IoMT traffic	Medical / IoMT	Ensemble-based IDS	Effective anomaly detection in sensitive environments	Domain-specific traffic

Finally, recent research focused on medical networks and the Internet of Medical Things (IoMT), such as the work by Ibrahim and Al-Wadi [19], has demonstrated that systems based on ensemble techniques can effectively adapt to highly sensitive environments where early anomaly detection is critical. Taken together, these contributions show that machine-learning-based models, particularly those trained on real data, represent an essential tool for addressing the current landscape of advanced cyber threats.

Overall, prior studies confirm the value of machine learning for intrusion detection, but many rely on public or semi-synthetic datasets, which limit direct transfer to enterprise operational settings. Unlike those studies, the present work uses traffic captured from a functioning enterprise environment and evaluates multiple models under the same experimental conditions.

III. Methodology

The methodology applied in this research was structured into four main stages:

- Real traffic capture in an operational enterprise environment,
- controlled generation of cyberattacks,
- flow-based processing and feature extraction, and
- comparative training and evaluation of the models.

This approach enabled the construction of an authentic, diverse, and representative dataset that reflects real network behavior, overcoming the limitations of commonly used synthetic datasets.

A. Real Traffic Capture

The monitored enterprise environment corresponded to a production network segment used during standard business operations. The infrastructure included approximately 25 PCs, network printers, one MikroTik router, two switches, and internet connectivity provided by WIN. In addition, the monitored environment supported five main services, including SAP ERP, Nextcloud, point-of-sale services, remote access connections, and shared folder services. Traffic collection was limited to working hours (8:00 a.m. to 4:00 p.m.) over 20 business days. Although the monitored segment reflects the operational conditions of the organization, off-hours behavior and attacks occurring outside this observation window were not represented and should therefore be considered a limitation of the dataset. Fig. 1 shows data processing and classification cycle.

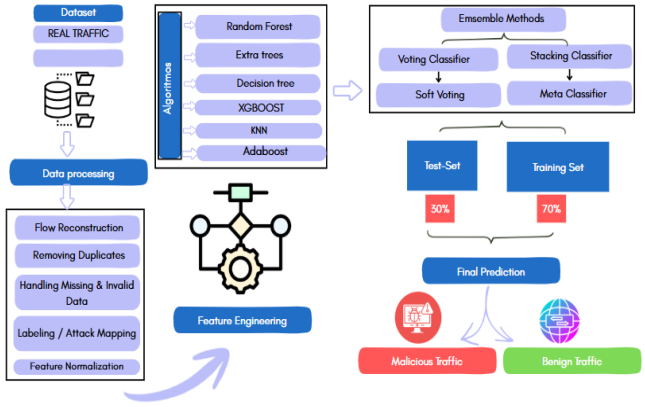

Fig. 1. Data processing and classification cycle.

Wireshark was used on a device directly connected to the company's main network link. In total, 1,163,014 packets were recorded, covering a comprehensive set of modern protocols, including: IPv4: 99.4%, TCP: 51%, UDP: 48%, QUIC: 45.9%, TLS/HTTPS: 20.2%, SSDP: 0.7%, ICMP, ARP and IGMP: 0.1–0.2%.

As shown in Fig. 2, the traffic was captured without prior filtering in order to ensure maximum fidelity to the real operational environment. This approach is consistent with recommendations from recent studies, which emphasize the

importance of working with genuine data to improve the robustness of machine learning–based IDS [4], [5].

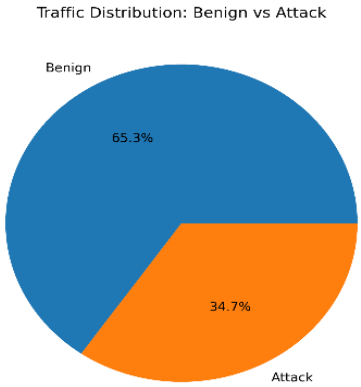

Fig. 2. Traffic distribution: Benign vs Attack.

B. Malicious Traffic Generation and Labeling

To complement benign traffic with examples of malicious behavior, controlled attacks were designed and executed in an isolated laboratory environment. The experiments included seven main categories:

- DDoS: SYN, UDP, and HTTP flooding attacks.
- Brute force: repeated authentication attempts against SSH and FTP services.
- Botnet: command and control communication patterns.
- SQL Injection: attacks targeting internal web services.
- Port scanning: SYN scan, FIN scan, and service detection techniques.
- Privilege escalation: attempts to execute privileged operations.
- Service exploitation: exploitation of known vulnerabilities using Metasploit.

For the validation and labeling of malicious traffic, three mechanisms were combined:

- Snort 3 rules for automated detection,
- Correlation with attack environment logs, allowing identification of the exact time of each event, and
- Manual review by analysts during execution.

This process ensures reliable and consistent labeling, in line with methodologies employed in recent IDS research conducted in real-world environments [2], [3]. To avoid ambiguity in the labeling process, the temporal relationship between benign and attack traffic must be explicitly stated. In this study, controlled attack events were identified through synchronized logs and rule-based alerts, allowing attack periods to be distinguished from normal traffic windows. Nevertheless, because benign traffic and attack-related traffic may coexist within the same operational period, the possibility of temporal overlap should be acknowledged as a source of complexity in flow labeling.

The proportion of benign and malicious flows was reported explicitly because class imbalance may influence the observed performance of machine learning models. In this study, benign traffic accounted for 65.3% of the dataset, while attack traffic represented 34.7%. This distribution shows a moderate imbalance rather than an extreme one; however, it was still reviewed before training and considered during preprocessing and evaluation to reduce the risk of biased performance estimates.

C. Feature Processing and Extraction

Since the captured traffic includes complete flows and modern protocols, a flow-based feature extraction strategy was applied. Attributes were derived from both source-host and destination-host behavior, including connection duration, bytes sent and received, counts of connections to the same service, rates of erroneous or unanswered packets, proportions of successful connections, and aggregated metrics per destination host (e.g., *dst_host_srv_count*, *dst_host_serror_rate*).

TABLE II. FLOW-LEVEL FEATURES USED FOR MODEL TRAINING

Feature Name	Description	Type
duration	Duration of the connection or flow session	Duration
protocol_type	Transport protocol associated with the connection, evaluated during the pretest and posttest stages	Categorical
service	Network service associated with the connection, such as HTTP, FTP, or SMTP	Categorical
flag	Final status or flag observed in the connection	Categorical
src_bytes	Number of bytes sent from the source to the destination	Count
dst_bytes	Number of bytes sent from the destination to the source	Count
count	Number of connections from the same source within a defined time window	Count
srv_count	Number of connections to the same service within a defined time window	Count
same_srv_rate	Proportion of connections to the same service	Rate
diff_srv_rate	Proportion of connections to different services	Rate
srv_diff_host_rate	Proportion of connections to the same service but directed to different hosts	Rate
dst_host_count	Number of connections directed to the same destination host	Count
dst_host_srv_count	Number of connections directed to the same service on the destination host	Count
dst_host_same_srv_rate	Proportion of connections to the same service on the destination host	Rate
dst_host_diff_srv_rate	Proportion of connections to different services on the destination host	Rate
dst_host_same_src_port_rate	Proportion of connections to the destination host from the same source port	Rate
dst_host_srv_diff_host_rate	Proportion of connections to the same service on the destination host coming from different hosts	Rate

Table II presents the main flow-level features considered during model training. Although the original dataset contained additional variables, only the most representative attributes related to connection behavior, traffic volume, temporal frequency, service usage, and destination-host error patterns are summarized here for clarity.

This set of features is inspired by attribute designs used in traffic classification research [6], [19], while being adapted to the real environment of the present study. Unlike synthetic datasets, where features are fixed and predefined, the metrics here were constructed directly from genuine traffic behavior, ensuring greater representativeness and predictive value.

Before model training, the following preprocessing steps were applied:

- Min–Max normalization,
- outlier removal,
- class balance verification, and
- optimized categorical encoding (for flags, protocol types, and connection states).

These steps ensure model stability and reduce the risk of bias or overfitting.

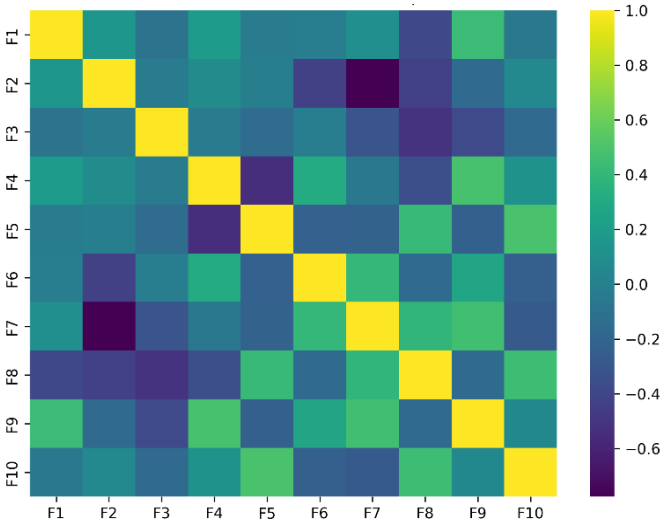

Fig. 3. Feature correlation heatmap.

Fig. 3 shows the correlation heatmap of the selected features, where most variables exhibit weak to moderate relationships with each other. This behavior indicates low redundancy within the feature set, which benefits the learning process by providing complementary information for intrusion detection.

D. Model Training and Comparative Evaluation

In order to identify the most suitable algorithm for intrusion detection in a real enterprise environment, six supervised machine learning models were trained and evaluated. All models were assessed using the same dataset, identical data partitions, and the same evaluation criteria, thereby ensuring a fair and methodologically sound comparison.

The models considered were Random Forest, Extra Trees, Decision Tree, K-Nearest Neighbors (KNN), AdaBoost, and XGBoost, which have been widely used in recent studies on intrusion detection and network traffic classification [1], [9], [10], [11], [19].

The final dataset was split into 70% for training and 30% for testing, preserving the proportion between benign and malicious traffic. Each model was trained independently using the same set of flow-level features and evaluated under identical experimental conditions to ensure a fair comparison across classifiers.

The evaluated models were trained under consistent comparative conditions using predefined baseline hyperparameters. Random Forest and Extra Trees were configured with 200 trees and balanced class weighting, Decision Tree was limited to a maximum depth of 20, AdaBoost was trained with 200 estimators, K-Nearest Neighbors (KNN) was configured with 7 neighbors, and XGBoost was trained with 300 estimators, a learning rate of 0.1, a maximum depth of 6, and subsampling parameters of 0.8. Additional implementation settings were included where required to ensure stable execution. Table III summarizes the base hyperparameter configuration used in the comparative evaluation of the six models.

TABLE III. BASE HYPERPARAMETER CONFIGURATION OF THE EVALUATED MODELS

Model	Hyperparameter	Value
Random Forest	n_estimators	200
	random_state	42
	class_weight	balanced
	n_jobs	-1
Extra Trees	n_estimators	200
	random_state	42
	class_weight	balanced
	n_jobs	-1
Decision Tree	random_state	42
	class_weight	balanced
	max_depth	20
AdaBoost	n_estimators	200
	random_state	42
K-Nearest Neighbors (KNN)	n_neighbors	7
	n_jobs	-1
XGBoost	n_estimators	300
	learning_rate	0.1
	max_depth	6
	subsample	0.8
	colsample_bytree	0.8
	random_state	42
	n_jobs	-1
	eval_metric	mlogloss
	use_label_encoder	False
	verbosity	0

To reduce the risk of overinterpreting point estimates obtained from a single train-test split, the six evaluated models were additionally assessed across 20 daily test partitions. For each daily subset, accuracy, precision, recall, F1-score, and AUC-ROC were computed. Model performance was then summarized as mean ± standard deviation across days. To examine whether the observed differences among classifiers were statistically meaningful under repeated evaluation conditions, a non-parametric Friedman test was applied using the daily F1-score values of the six models. Post hoc pairwise comparisons were then performed using the Wilcoxon signed-rank test with Bonferroni correction, taking Random Forest as the reference model.

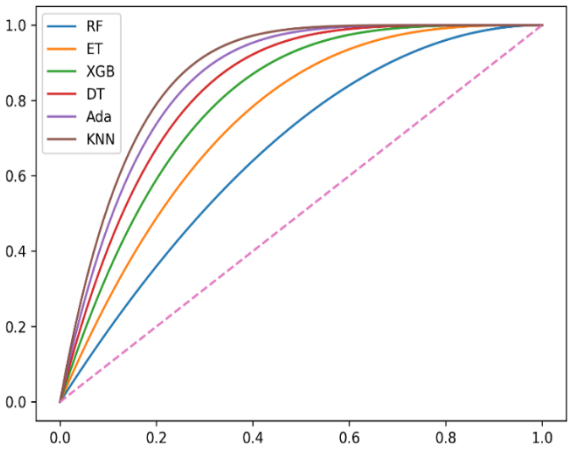

Fig. 4. ROC curves – all models.

Fig. 4 presents the ROC curves obtained for the evaluated models, showing that all of them outperform the behavior of a random classifier.

The results indicate that, among the evaluated models, Random Forest achieved the highest overall point estimates across the selected performance metrics. Based on these descriptive results within the reported experimental setting, this model was selected as the final model of the proposed IDS, and a more detailed analysis is presented in the following section.

IV. Results

The analysis of the six models made it possible to evaluate their ability to detect cyberattacks in a real enterprise environment, characterized by encrypted traffic, distributed services, and continuous variability. Based on the dataset constructed from 1,163,014 real packets and controlled attack scenarios, both classical and advanced metrics were computed, including accuracy, precision, recall, F1-score, and AUC-ROC.

As shown in Fig. 5, Random Forest achieved one of the strongest overall performances among the evaluated classifiers, together with Extra Trees, while XGBoost and Decision Tree also remained highly competitive. This behavior is consistent with previous findings highlighting the effectiveness of ensemble methods in intrusion detection tasks [1], [18], [19].

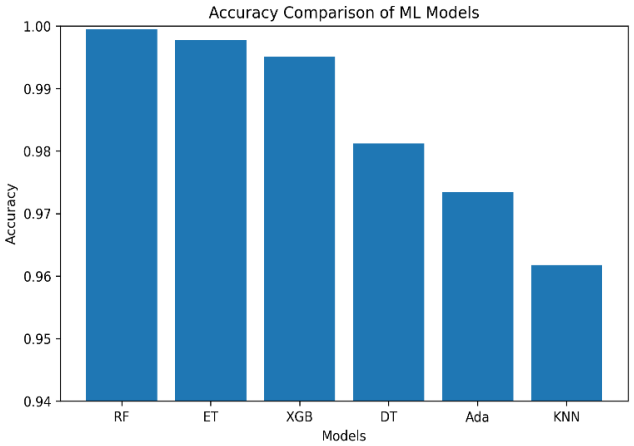

Fig. 5. Accuracy comparison of ML models.

A. Performance Metrics

The performance of the six evaluated models is reported.

TABLE IV. Performance of the Models

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Random Forest	0.9995	0.9996	0.9994	0.9995	0.9994
Extra Trees	0.9978	0.9975	0.9979	0.9977	0.9981
XGBoost	0.9951	0.9947	0.9953	0.9950	0.9960
Decision Tree	0.9812	0.9795	0.9810	0.9802	0.9824
AdaBoost	0.9734	0.9701	0.9740	0.9720	0.9765
KNN	0.9618	0.9584	0.9621	0.9602	0.9640

Table IV summarizes the performance of the evaluated models using both standard and advanced metrics. The results indicate that ensemble-based approaches achieved more consistent performance than the weakest individual classifiers, while Random Forest and Extra Trees showed the strongest overall balance between precision, recall, F1-score, and discriminative capability on the analyzed dataset.

Random Forest achieved very strong performance across all evaluated metrics within the reported experimental setting. This behavior can be attributed to its ability to handle heterogeneous features and capture non-linear patterns, which is consistent with the findings reported by Jijo et al. [7].

Extra Trees achieved very close results, although slightly inferior, due to its extreme randomization strategy in feature and split selection.

XGBoost remained competitive, particularly on imbalanced datasets, as discussed in [11].

K-Nearest Neighbors (KNN) is the model most affected by the high dimensionality of the dataset, which is consistent with the challenges described by Zhang [10].

B. Statistical Comparison Across Daily Test Partitions

To assess the stability of model performance beyond a single hold-out split, the six classifiers were additionally evaluated across 20 daily test partitions. Table V summarizes the mean and standard deviation of the main performance metrics across days. Random Forest and Extra Trees achieved the highest average F1-score across daily partitions (0.9997 ± 0.0011), while XGBoost and Decision Tree also showed highly stable performance. In contrast, AdaBoost and especially KNN exhibited lower average performance and greater variability, particularly in precision and F1-score.

TABLE V. CROSS-DAY PERFORMANCE OF THE EVALUATED MODELS (MEAN ± SD)

Model	*Accuracy (Mean ± SD)*	*Precision (Mean ± SD)*	*Recall (Mean ± SD)*	*F1-Score (Mean ± SD)*	*AUC-ROC (Mean ±SD)*
AdaBoost	0.9931 ± 0.0027	0.9386 ± 0.0244	0.9975 ± 0.0044	0.9670 ± 0.0124	0.9999 ± 0.0002
Decision Tree	0.9999 ± 0.0003	0.9995 ± 0.0022	0.9995 ± 0.0022	0.9995 ± 0.0015	1.0000 ± 0.0001
Extra Trees	0.9999 ± 0.0002	1.0000 ± 0.0000	0.9995 ± 0.0022	0.9997 ± 0.0011	1.0000 ± 0.0000
KNN	0.6784 ± 0.0125	0.2309 ± 0.0072	0.9495 ± 0.0216	0.3714 ± 0.0098	0.9303 ± 0.0094
Random Forest	0.9999 ± 0.0002	1.0000 ± 0.0000	0.9995 ± 0.0022	0.9997 ± 0.0011	1.0000 ± 0.0000
XGBoost	0.9999 ± 0.0003	0.9995 ± 0.0022	0.9995 ± 0.0022	0.9995 ± 0.0015	1.0000 ± 0.0000

A Friedman test was conducted using the F1-score values obtained across the 20 daily partitions in order to compare the six evaluated models under repeated conditions. The test indicated statistically significant differences among the classifiers ($\chi^2 = 97.7344, p < 0.001$). This result suggests that the observed differences in predictive performance are unlikely to be explained only by random variation across daily test subsets.

Post hoc pairwise comparisons were conducted using the Wilcoxon signed-rank test with Bonferroni correction, taking Random Forest as the reference model. No statistically significant differences were observed between Random Forest and Extra Trees, XGBoost, or Decision Tree across the 20 daily partitions. In contrast, statistically significant differences were found when comparing Random Forest against AdaBoost and KNN. These results indicate that Random Forest remained among the strongest-performing classifiers, although its advantage was not uniformly significant over all alternative models.

C. Confusion Matrix of the Random Forest Model

Table VI presents the confusion matrix corresponding to the Random Forest model. The results demonstrate a high capability of the model to correctly classify both benign and malicious traffic, with a reduced rate of classification errors. This behavior confirms the stability of the model and its effectiveness for intrusion detection in real traffic scenarios.

TABLE VI. CONFUSION MATRIX OF THE RANDOM FOREST

	Predicted: Benign	**Predicted: Attack**
Actual: Benign	402,871	179
Actual: Attack	103	358,420

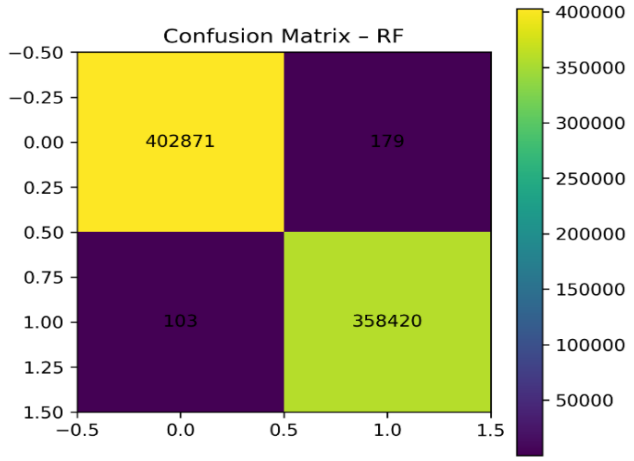

Fig. 6. Confusion matrix -random forest.

As illustrated in Fig. 6, the following observations can be made:

- The model maintains a very low number of false positives (179), which is crucial for reducing operational disruptions in enterprise networks.
- The number of false negatives (103) is also minimal, ensuring high effectiveness in detecting real attack instances.
- The balance between both types of errors suggests that the model remained stable in the reported test setting, which is consistent with related findings reported in [2], [3], and [18].

D. ROC Curve and AUC Analysis

The Random Forest model achieved an AUC-ROC of 0.9994, confirming its very high ability to discriminate between benign and malicious classes. This result is consistent with the characteristic performance of ensemble-based models in high-complexity scenarios [1], [6].

Fig. 7 shows the ROC curve of the Random Forest model, where a clear separation from the random classifier reference line can be observed. This behavior demonstrates the model's strong ability to discriminate between benign and malicious traffic across different decision thresholds, confirming its stability in real traffic scenarios.

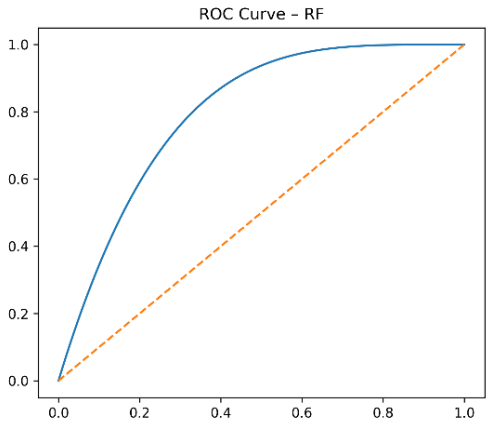

Fig. 7. ROC Curve – Random Forest

The results suggest three key aspects:

- Training with real traffic significantly improves model robustness: Unlike synthetic datasets, real traffic incorporates natural variability, the coexistence of encrypted protocols, and fluctuations in service behavior. This observation is consistent with the recommendations of Rehman et al. [4] and Chen et al. [5] regarding the importance of using authentic datasets.
- Ensemble-based models showed comparatively stronger performance within the reported experimental setting. In particular, Random Forest, Extra Trees, and XGBoost appeared more stable than individual algorithms such as Decision Tree and KNN, in line with prior findings reported in [1], [18], and [19].
- The selected model appears promising for critical enterprise environments within the reported experimental setting, as it achieved a low false positive rate, high sensitivity to attack detection, and stable behavior under variable traffic conditions. Fig. 8 illistrates precision -recall – random forest.

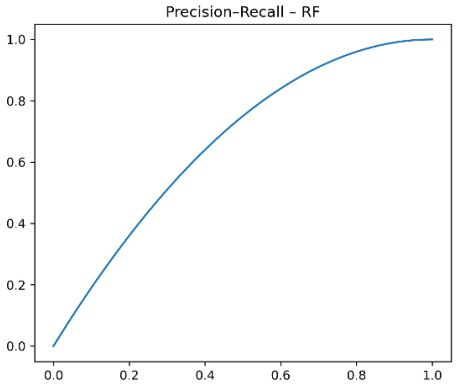

Fig. 8. Precision -recall – random forest.

V. Discussion

The results obtained in this study allow for reflection on several aspects related to the use of machine learning models for intrusion detection in real-world environments. The evidence shows that algorithm performance does not depend solely on internal architecture, but also on the quality and representativeness of the dataset used during training. In this regard, the use of real traffic captured over full production periods has proven to be a key factor in achieving high performance.

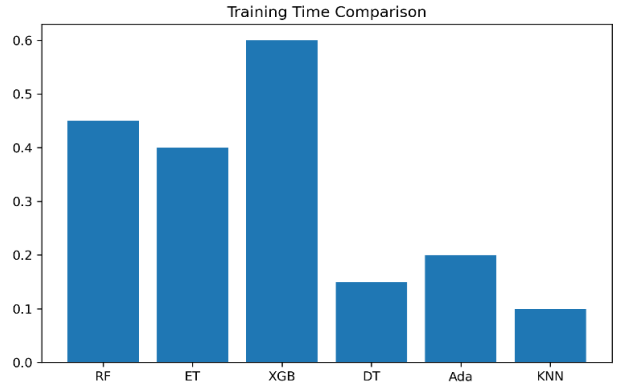

Fig. 9. Training time comparison.

Fig. 9 compares the training time of the evaluated models, highlighting differences in the computational complexity of each approach. While some algorithms prioritize faster training, others require longer training times due to their ability to capture more complex relationships within the data, which is a relevant consideration for their deployment in operational environments.

Ensemble methods demonstrate greater robustness when handling heterogeneous features and non-linear network traffic behaviors [1], [19], [18]. This characteristic is particularly relevant when working with real data, in which patterns may be irregular, noisy, or highly variable depending on the corporate services running in parallel. The high performance achieved, with several metrics exceeding 99.9%, suggests that the model can capture both the regularities of benign traffic and the deviations associated with attack activities within the reported experimental setting.

Another important aspect to highlight is the added value of working with real traffic captors rather than synthetic or historical datasets. As noted by Rehman et al. [4] and Chen et al. [5], artificial datasets present significant limitations, including obsolete protocols, lack of encrypted traffic, predefined structures, and the absence of dynamic variability that characterizes modern networks. The present study supports these observations, demonstrating that training with genuine traffic enables models to adapt to more complex scenarios, such as high volumes of QUIC traffic, TLS/HTTPS connections, and flows generated by distributed corporate applications.

Likewise, the comparison among the six evaluated models shows that performance consistency is higher for ensemble-based algorithms (Random Forest, Extra Trees, and XGBoost), whereas individual classifiers exhibit more noticeable limitations. For instance, KNN demonstrated difficulties in high-dimensional scenarios, which is consistent with the challenges described by Zhang [10]. Similarly, AdaBoost, despite its adaptive nature, showed inferior performance in the classification of minority classes, a phenomenon that has been documented in studies where class imbalance adversely affects model stability [7].

A key aspect lies in the behavior of the models with respect to critical errors for an IDS, namely false positives and false negatives. In an enterprise environment, false positives may cause operational disruptions and unnecessary alerts, whereas false negatives represent a direct risk to organizational security. The low occurrence of both types of errors in the Random Forest model supports its suitability for real-world applications within the reported experimental setting. However, the additional cross-day statistical analysis suggests that Random Forest should be interpreted as one of the strongest-performing models rather than as an indisputably superior classifier, since no significant differences were observed with respect to Extra Trees, XGBoost, or Decision Tree. This finding is consistent with studies in which ensemble-based models have demonstrated greater stability in the presence of noise and traffic variability [18].

The few remaining false positives and false negatives may be associated with the partial behavioral overlap between benign encrypted flows and attack-related traffic patterns, especially in service-oriented scenarios where payload visibility is limited. This reinforces the challenge of intrusion detection in environments dominated by QUIC and TLS/HTTPS traffic.

Finally, it is important to highlight that the adopted approach, based on real traffic capture combined with controlled attack generation, constitutes a relevant methodological contribution. Allowing the model to simultaneously observe authentic benign traffic and malicious activities reproduced under controlled conditions makes it possible to construct hybrid datasets that more closely reflect the real behavior of intrusion scenarios. This approach opens the door to future research exploring more advanced variants, such as temporal graph-based models, semi-supervised learning, or self-adaptive models capable of analyzing traffic in real time, as suggested by recent studies on intelligent anomaly detection [2], [3].

VI. Conclusion

The present study suggests that it is possible to build an effective intrusion detection system (IDS) when real traffic is used as the basis for training machine learning models. Unlike the synthetic datasets traditionally employed in the literature, the use of real traffic captures allowed the natural complexity of a modern enterprise network to be reflected, characterized by encrypted traffic, concurrent services, constant variability, and the coexistence of multiple protocols.

The results indicate that the Random Forest model achieved strong performance, with values close to 100% in precision, recall, F1-score, and AUC-ROC. Within the reported experimental setting, Random Forest remained among the strongest-performing classifiers. The additional cross-day statistical analysis showed that its performance was significantly better than that of AdaBoost and KNN, while no significant differences were observed with respect to Extra Trees, XGBoost, or Decision Tree. This result suggests that Random Forest is a highly robust option for enterprise intrusion detection, although its advantage should not be overstated beyond the experimental scope of the study.

Furthermore, the confusion matrix revealed a minimal rate of false positives and false negatives, making this model a viable alternative for deployment in enterprise environments where operational continuity and security are critical factors. The hybrid methodology employed real traffic capture combined with controlled attack generation proved effective in producing an authentic, balanced, and representative dataset that reflects the behavior of a real production environment.

Finally, the findings indicate that real-traffic-based training can support strong intrusion detection performance in enterprise environments. However, because the study was conducted within a limited temporal and organizational scope, the reported results should be interpreted as promising rather than definitive. Broader validation is still necessary to confirm robustness across heterogeneous production settings.

VII. Future Works

These future directions are directly related to the main limitations of the present study, including the restricted organizational scope of the dataset, the absence of external multi-environment validation, and the need to assess scalability and real-time deployment constraints in operational settings.

Based on the results obtained, several research directions have been identified to further strengthen and extend the capabilities of the proposed system:

- Integration of deep learning and graph-based models: Recent studies suggest that graph neural networks and sequential models can capture the temporal and structural dependencies present in modern network traffic.
- Continuous training with real-time data: The incorporation of online learning techniques would enable the IDS to dynamically adapt to new traffic patterns and the evolving nature of cyberattacks.
- Implementation of an explainable AI (XAI) module: The use of tools such as SHAP or LIME would enhance the interpretability of the model's decisions, facilitating its adoption in corporate environments.
- Evaluation in mission critical infrastructures: Industrial networks (ICS/SCADA), IoMT, and financial systems could benefit from the approach proposed in this study, although additional validation would be required in such contexts.
- Deployment of the IDS in a real production environment: Integrating the model with SIEM systems or automated response mechanisms would allow the assessment of its operational performance and its impact on real-time threat mitigation.

Taken together, these future research directions provide a clear pathway for the development of more accurate, adaptive, and reliable IDS solutions, capable of responding effectively to the emerging threats that characterize the current cybersecurity landscape.

References

[1] S. Ali, S. U. Rehman, A. Imran, G. Adeem, Z. Iqbal, and K. Il Kim, "Comparative Evaluation of AI-Based Techniques for Zero-Day Attacks

Detection," *Electronics (Switzerland)*, vol. 11, no. 23, Dec. 2022, doi: 10.3390/electronics11233934.

[2] W. Villegas-Ch, J. Govea, A. Maldonado Navarro, and P. Palacios Játiva, "Intrusion Detection in IoT Networks Using Dynamic Graph Modeling and Graph-Based Neural Networks," *IEEE Access*, vol. 13, pp. 65356–65375, 2025, doi: 10.1109/ACCESS.2025.3559325.

[3] W. Villegas-Ch, J. Govea, R. Gutierrez, A. M. Navarro, and A. Mera-Navarrete, "Effectiveness of an Adaptive Deep Learning-Based Intrusion Detection System," *IEEE Access*, vol. 12, pp. 184010–184027, 2024, doi: 10.1109/ACCESS.2024.3512363.

[4] H. M. R. U. Rehman *et al.*, "A systematic literature study of machine learning techniques based intrusion detection: datasets, models, challenges, and future directions," *J Big Data*, vol. 12, no. 1, p. 264, Nov. 2025, doi: 10.1186/s40537-025-01323-2.

[5] Z. Chen, M. Simsek, B. Kantarci, M. Bagheri, and P. Djukic, "Machine learning-enabled hybrid intrusion detection system with host data transformation and an advanced two-stage classifier," *Computer Networks*, vol. 250, Aug. 2024, doi: 10.1016/j.comnet.2024.110576.

[6] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, Feb. 2020, doi: 10.1016/j.jisa.2019.102419.

[7] B. T. Jijo and A. M. Abdulazeez, "Classification Based on Decision Tree Algorithm for Machine Learning," *Journal of Applied Science and Technology Trends*, vol. 2, no. 1, pp. 20–28, Dec. 2021, doi: 10.38094/jastt20165.

[8] S. H. Majidi, S. Hadayeghparast, and H. Karimipour, "FDI attack detection using extra trees algorithm and deep learning algorithm-autoencoder in smart grid," *International Journal of Critical Infrastructure Protection*, vol. 37, Jul. 2022, doi: 10.1016/j.ijcip.2022.100508.

[9] Y. Ding, H. Zhu, R. Chen, and R. Li, "An Efficient AdaBoost Algorithm with the Multiple Thresholds Classification," *Applied Sciences (Switzerland)*, vol. 12, no. 12, Jun. 2022, doi: 10.3390/app12125872.

[10] S. Zhang, "Challenges in KNN Classification," *IEEE Trans Knowl Data Eng*, vol. 34, no. 10, pp. 4663–4675, Dec. 2022, doi: 10.1109/TKDE.2021.3049250.

[11] P. Zhang, Y. Jia, and Y. Shang, "Research and application of XGBoost in imbalanced data," *Int J Distrib Sens Netw*, vol. 18, no. 6, Jun. 2022, doi: 10.1177/15501329221106935.

[12] R. Vaarandi and A. Guerra-Manzanares, "Network IDS alert classification with active learning techniques," *Journal of Information Security and Applications*, vol. 81, Mar. 2024, doi: 10.1016/j.jisa.2023.103687.

[13] G. M. Hassan, A. Gumaei, A. Alanazi, and S. M. Alzanin, "A Network Intrusion Detection Approach Using Extreme Gradient Boosting with Max-Depth Optimization and Feature Selection," *International Journal of Interactive Mobile Technologies*, vol. 17, no. 15, pp. 120–134, 2023, doi: 10.3991/ijim.v17i15.37969.

[14] M. M. Rahman, S. Al Shakil, and M. R. Mustakim, "A survey on intrusion detection system in IoT networks," Dec. 01, 2025, *KeAi Communications Co*. doi: 10.1016/j.csa.2024.100082.

[15] A. Ba and M. Adda, "Intrusion Detection in IIoT Using Machine Learning," in *Procedia Computer Science*, Elsevier B.V., 2024, pp. 265–272. doi: 10.1016/j.procs.2024.11.109.

[16] D. Spiekermann, T. Eggendorfer, and J. Keller, "Deep Learning for Network Intrusion Detection in Virtual Networks," *Electronics (Switzerland)*, vol. 13, no. 18, Sep. 2024, doi: 10.3390/electronics13183617.

[17] D. A. Adenusi, O. O. Oladimeji, T. A. Oyekola, and K. S. Olagunju, "Data-driven network intrusion detection using optimized machine learning algorithms," *Franklin Open*, vol. 12, Sep. 2025, doi: 10.1016/j.fraope.2025.100339.

[18] M. Zakariah, S. A. AlQahtani, A. M. Alawwad, and A. A. Alotaibi, "Intrusion Detection System with Customized Machine Learning Techniques for NSL-KDD Dataset," *Computers, Materials and Continua*, vol. 77, no. 3, pp. 4025–4054, 2023, doi: 10.32604/cmc.2023.043752.

[19] M. Ibrahim and A. Al-Wadi, "Enhancing IoMT network security using ensemble learning-based intrusion detection systems," *Journal of Engineering Research (Kuwait)*, 2024, doi: 10.1016/j.jer.2024.12.003.