HyFlaNK: A Hybrid Federated Learning Framework for Real-time Network Threat Detection

Oluyemisi Adenike Oyedemi, Renata Lopes Rosa, *Member, IEEE,* Ugochukwu Okwudili Matthew, and Demóstenes Zegarra Rodríguez, *Senior Member, IEEE*

***Abstract*—This paper proposed HyFlaNK, a hybrid federated learning threat detection framework combining Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) in a Flower-based federated learning model integrated with Apache Kafka to simulate live data ingestion, model updates, and feedback loops. The system is scalable, supports self-learning, real-time evaluation using TensorFlow/Keras for model creation and Flower for federated orchestration. Performance analysis was conducted to evaluate the model using accuracy, loss, precision, recall, F1-score, and ROC-AUC. Confusion matrices generated for the clients and global model shows good classification performance. Experimental results show consistently high performance across the local models and the aggregated global model, achieving accuracies above 99.7% and ROC-AUC of 1.0, highlighting the effectiveness and reliability of HyFlaNK. A line plot of accuracy and loss over federated rounds revealed a consistent upward trend in accuracy and a corresponding decline in loss, validating the capability of HyFlaNK to maintain high detection performance while preserving data privacy in a distributed environment. Additionally, a comprehensive performance evaluation comparing HyFlaNK with a traditional Random Forest-based approach further underscores its superior accuracy, precision, and scalability, making it a more robust solution for real-time threat detection in decentralized environments.**

***Index Terms*—Federated Learning, Real-Time Threat Detection, Apache Kafka, Data Privacy.**

I. Introduction

In this age of fast digital transformation, the development of web-based services and platforms has generated unprecedented potential for both organizations and individuals [1], [2]. This shift in technology has enabled institutions and organsations to thrive in modern digital landscape [3]. Digital transformation acts as a powerful catalyst that fuels innovation, creativity, and efficiency across sectors [4]. It has seamlessly integrated many sectors such as education [5],[6], medical applications [7], [8], and govenrment services [9].

However, this expansion has resulted in sophisticated web-based threats as a result of sopisticated attacks [9], [11]. Due to the dynamic nature of these threats, robust and adaptive systems capable of real-time detection and mitigation are required. Traditional approaches to threat detection, such as rule-based systems, signature-based detection, and centralized machine learning (ML) models, often struggle to keep pace with evolving threats [12],[15]. The increasing volume and velocity of security-related data have also overwhelmed exist-ing systems. The concept of federated learning (FL) therefore emerged as a transformative approach in various fields of machine learning [13], [14]. Unlike traditional centralized ma-chine learning, FL enables collaborative model training across distributed devices or organizations while preserving data privacy by ensuring that raw data never leaves its source [29]. This study implemented the federated learning process using the Flower framework, an open-source federated learning platform designed to support scalable and flexible client–server architectures [19]. Flower enables efficient coordination be-tween distributed clients and the central aggregation server, facilitating secure model weight exchange without sharing raw data [21]. Its modular and framework-agnostic design allows seamless integration with deep learning libraries and various federated optimization algorithms, including Federated Averaging, FedProx, and Federated SGD [20]. Furthermore, it offers the flexibility to implement new federated optimization algorithms effortlessly. Flower operates on a client-server architecture, requiring a minimum configuration of one server and at least two clients to initiate federated training.

This characteristic makes FL particularly appealing for cybersecurity applications, where sensitive data is involved, and regulatory compliance is critical [30]. Federated learning has been applied in various fields such as intelligent medical systems [18], smart city [22], and edge of computing [23]. This study therefore builds on these innovations to design a hybrid system framework that implements a real-time federated learning pipeline integrated with Apache Kafka to simulate live threat data ingestion, continuous model updates, and feedback-driven retraining. Apache Kafka is indeed a distributed publish-subscribe messaging system designed to handle high throughput and real-time data streams [25]. It is widely used in various industries to handle high volumes of data in real time [27]. Its robust ecosystem and architecture make it a popular choice for organizations seeking to build scalable data pipelines, enable real-time analytics, and manage event-driven applications [28].

The proposed system, HyFlaNK addressed these challenges in cybersecurity, including improved detection of network traffic threats, reduction of false positives and negatives,

Manuscript received September 23, 2025; revised February 17, 2026. Date of publication June 10, 2026. Date of current version June 10, 2026. The associate editor prof. Miljenko Mikuc has been coordinating the review of this manuscript and approved it for publication.

Authors are with the Computer Science Department, Federal University of Lavras, Minas Gerais, Brazil (e-mails: oyedemi.adenike@ufla.br, renata.rosa@ufla.br, ugochukwu.matthew@estudante.ufla.br, demostenes.zegarra@ufla.br).

Digital Object Identifier (DOI): 10.24138/jcomss-2025-0113

increased security context, better compliance and regulatory management, and preservation of data privacy. HyFlaNK offers a scalable and efficient solution for modern security needs by employing a modular design thereby promoting flexibility and scalability. The federated learning model was trained on network traffic datasets, comprising of various attack types. Federated Learning allows for decentralized training, where the local models (clients) simulates different devices, train on a set of data and update a global model with their weights.The trained model, integrated with Apache Kafka, allowed for real-time data ingestion, model updates, and feedback loops within the federated learning (FL) pipeline. Kafka served as the message broker that facilitated the communication between these clients and the global server. The clients continuously sent their local model updates (which are trained on local data) to the global model via Kafka.

The highlighted contributions of HyFlaNK that differentiate it from existing systems are as follows:

- A novel hybrid cybersecurity framework, HyFlaNK is developed by integrating Flower Federated Learning (FL) with Apache Kafka to enable real-time, privacy-preserving threat detection.
- The system incorporates self-learning capabilities through feedback-driven model updates, enabling dynamic adaptation to evolving cyber threats.
- The framework allows the global model to update incrementally using new streaming data from Kafka, as new data arrives, eliminating the need for full retraining and improving system responsiveness.
- HyFlaNK architecture supports privacy and regulatory compliance by preventing centralized data aggregation, aligning with standards such as GDPR and HIPAA.
- A scalable and modular design for deployment in diverse environments with multiple data sources, improving flexibility and extensibility.
- The framework enhances threat detection performance, reducing false positives and negatives when compared to traditional centralized ML systems.
- The system captures complex spatial and temporal patterns in network traffic, addressing limitations in existing stream processing-based detection methods.

The rest of this paper is organized as follows: Section II discusses related works. Section III outlines the proposed hybrid federated learning framework. Section IV presents result and discussion, while Section V presents conclusion and future works.

II. Related Works

This section reported a review of literature that was conducted on threat detection highlighting their methodologies, limitations, and how this study seeks to address those gaps.

Machine learning has been widely adopted for detecting anomalies, malware, and advanced persistent threats (APTs) [24]. Traditional approaches involve centralized training of ML models on aggregated datasets, which enables robust training but poses significant privacy and scalability challenges. For instance, [26] proposed a centralized anomaly detection system for intrusion detection using support vector machines (SVMs) and deep learning. While effective, the reliance on centralized data aggregation limits its applicability in scenarios involving sensitive or distributed data [29]. [30] explored FL for distributed malware detection across multiple organizations. Although promising, their study revealed lack of real-time analytics in their system limiting its applicability in time-sensitive environments like Security Operations Centers (SOC).

In [35], authors integrated Kafka into machine learning framework to identify and respond to threats. However, the system does not address privacy concerns, as the training data is transmitted to a centralized server, rather than being processed locally. Additionally, the model is not self-learning—it does not continuously update itself based on incoming requests. Incorporating such a feature would enable the system to dynamically adapt to new threats and improve throughput and responsiveness over time.

An evaluation of a virtual Network Function (VNF) designed for real-time threat detection using stream processing was presented in [44]. The system relied on traditional stream processing which are not effective in capturing spatial and temporal patterns in data. Privacy was not taken into account in the training process, as it processes traffic centrally, potentially exposing sensitive data.

In [45], authors explored machine learning algorithms for cyber threat detection in the context of network security. Authors evaluate various machine learning algorithms such as Support Vector Machine (SVM), Decision Trees and Random Forest. The authors do not consider the privacy-preserving aspect of the problem or the distributed nature of modern network environments. The traditional machine learning models analyzed in the paper may struggle with generalizing over large, diverse datasets and new, unseen threats.

A preliminary framework for federated anomaly detection in distributed network environments was proposed in [31]. Their study highlighted the feasibility of using FL for privacy-preserving threat detection but did not address the challenges of real-time threat detection.

Authors in [32] developed a real-time system for detecting Distributed Denial of Service (DDoS) attacks. While effective for specific attack vectors, their system does not support self-learning, lacked the flexibility to generalize across diverse web threats, limiting its scalability and adaptability. While the aforementioned studies have advanced the state of cybersecurity, they exhibit several limitations that this research seeks to address. Centralized ML approaches struggle to handle the high volume and velocity of security data generated in modern environments [33], [34]. Also, for privacy, many existing systems rely on centralized data aggregation, raising privacy concerns and complicating compliance with regulations like General Data Protection Regulation (GDPR) and Health Insurance Portability and Accountability Act (HIPAA). Most federated learning frameworks lack integration with operational real-time analytics, hindering real-time threat visibility and response [31].

In [36], authors proposed a real-time network anomaly detection system that utilizes machine learning techniques,

specifically decision trees and clustering algorithms, to identify anomalies in network traffic. The model assumed a centralised data processing which can be a security risk. The model can not effectively capture complex patterns in network traffic. The model did not address privacy concerns associated with sharing sensitive network traffic data. The challenges in large-scale, distributed environments where multiple clients generate data simultaneously was not addressed. Work [37] proposes a lightweight federated intrusion detection model for IoT using a collaborative neural network. The model preserves privacy but does not integrate real-time streaming and adaptive retraining, which support continuous feedback-based self-learning. In [38], authors developed a model for federated setting to balance privacy and accuracy on industrial IoT datasets. The approach addresses vanishing gradients and yields high accuracy but lacks real time data streaming and feedback integration. A federated learning model for broad cybersecurity threat detection, emphasizing privacy and decentralized architecture, is explored in [39]. The work discusses general decentralized threat models without consideration for real-time network traffic detection and adaptive retraining. Paper [40] focuses on model personalization and communication efficiency, but does not address real-time streaming ingestion or feedback-driven adaptive retraining mechanisms. HyFlaNK integrates Kafka for continuous data flow and model updates, enabling live threat detection beyond offline FL training. Paper [41] introduces a federated IDS aimed at mitigating data heterogeneity via knowledge distillation and proximal regularization, explicitly handling non-IID conditions common in decentralized IoT data. The model works without real-time streaming capability or feedback loops. In contrast, HyFlaNK's architecture supports continuous monitoring and retraining based on streamed predictions. In [42], authors deployed an attention-based federated deep IDS (FetFIDS) that enhances local feature representation for more accurate attack classification in federated settings. The model does not have the capability for self-learning feedback retraining. Authors in [43] combine federated deep learning with chimp optimization for feature selection in IoT intrusion detection, aiming to improve classification accuracy over traditional ML models. Apart from lack of real-time streaming, the model does not include feedback-driven model adaptation.

This research bridges the gap by proposing HyFlaNK, a hybrid framework that leverages the privacy-preserving capabilities of FL with the ability of Kafka to handle large volumes of real-time data from multiple sources in a decentralised manner.

III. Proposed Hybrid Federated Learning Framework

Fig. 1 represents the architecture of HyFlaNK, a Flower-based federated learning model integrated with Apache Kafka for real-time data streaming and monitoring. The architecture was designed to enhance network threat detection by enabling scalable, privacy-preserving collaborative learning across distributed clients, while ensuring low-latency data processing for rapid incident response.

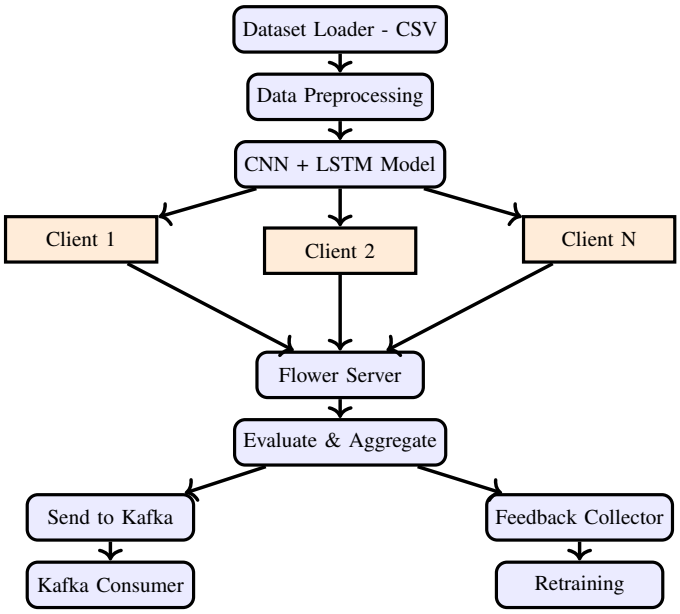

Fig. 1. Architecture of the proposed HyFlaNK Flower-based federated learning model

The system protected privacy while leveraging the hybrid capability of Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) in a federated learning system. The model enhanced security intelligence by enabling real-time detection of threats, collaborative model training without sharing of data, scalability, privacy preservation, and continuous learning from incomng requests to make the system adapt to new threats.

The CICIDS-2018 dataset downloaded from Kaggle, with features shown in Table I, consists of multiple CSV files loaded through the dataset loader. The dataset consisted of network traffic data, including various types of cyberattacks. In the data preprocessing module, several steps were performed: label encoding, normalization, feature selection using Random Forest, and splitting the dataset into training and test subsets. The CICIDS-2018 dataset was divided into five disjoint subsets, each assigned to a separate client. Each subset contains proportional representations of attack types and normal traffic to ensure realistic distributed training, maintaining diversity across clients while preventing data leakage. The dataset is evenly divided among the five clients, with each client receiving a distinct subset of the data for local training; specifically, the first four clients receive $\lfloor N/5 \rfloor$ samples each, and the fifth client receives the remaining samples, ensuring that all N samples are used and no data overlaps between clients.

The preprocessing phase is a crucial step in any machine learning pipeline, as it transforms raw data into a structured format suitable for analysis and modeling. In the setup, each client trains a hybrid CNN-LSTM model using its own partition of the dataset, enabling parallelized learning while leveraging both convolutional features and temporal dependencies in the data. A federated aggregator server, using the Federated Averaging (FedAvg) algorithm, was responsible for building a global model by aggregating local updates from individual

clients and evaluating its performance.

The Flower server coordinated the entire federated learning process, managing communication between clients and aggregating the model weights to iteratively improve the global model without requiring direct access to client data. While HyFlaNK employs a modular threading approach to improve concurrency and manage server, feedback loop, and Kafka operations efficiently, federated learning can still be impacted if one or more clients are significantly slower than others. The system monitors client responsiveness and can proceed with partial aggregation if a client exceeds a predefined timeout threshold, thereby mitigating disruption while maintaining privacy and model accuracy. In addition to slow clients, fast malicious clients may consistently participate in aggregation and introduce poisoned updates. To mitigate this, HyFlaNK can incorporate lightweight update validation by measuring the deviation of each client update from the global model using $D_i = \|w_i - w_{global}\|$. Updates exceeding a statistical threshold derived from observed client update distributions, $D_{threshold} = \mu_D + \beta\sigma_D$, can be down-weighted or excluded. This complements the time-based response threshold $T_{threshold} = \mu_T + \alpha\sigma_T$, improving robustness against both slow and adversarial clients.

Apache Kafka was integrated to collect real-time prediction data and feed it into a Kafka consumer for visualization. After each training round, clients generated threat detection labels as predictions, which were then stored in a real-time feedback collector. This enabled continuous monitoring and assessment of the system's threat detection performance. The feedback-based retraining module periodically triggers model retraining using the data collected from real-time feedback.The real-time feedback was simulated using labeled dataset partitions to emulate validation in a production environment. Although federated learning prevents raw data from leaving client nodes, locally available labels were used to evaluate prediction correctness during training. In real-world deployments, feedback would typically be obtained through delayed validation processes such as security analyst verification, incident response confirmation, or correlation with external monitoring systems, rather than assuming immediate access to labeled streaming data.

As illustrated in Fig. 2, this process flow allows the system to detect and adapt to emerging threats, thereby improving model robustness and enhancing real-time threat detection capabilities. The top section of the figure represents the distributed clients, each training a local model on its own private data, which are then aggregated by the Flower server to form the global model. Fig. 2 illustrates the process flow executed on each client and the Flower server. The top section represents distributed clients (Client 1, Client 2, ..., Client N), where local CNN+LSTM models are trained on private data. The Flower server aggregates the client updates (FedAvg) and evaluates the global model, forming the boundary between client-side and server-side operations as shown in Fig. 3.

The multi-class HyFlaNK model defined and compiled a sequential neural network model using a combination of convolutional layers (Conv1D), LSTM layers (LSTM), and dense layers (Dense). The model was compiled using the Adam optimizer and sparse categorical cross-entropy loss function for multi-class classification.

TABLE I
FEATURE IMPORTANCE BEFORE SELECTION (TRUNCATED TO VALUES > 0.0005)

Feature	Importance
Dst Port	0.1129
Fwd Seg Size Min	0.0667
Flow Pkts/s	0.0658
Fwd Pkts/s	0.0513
Init Fwd Win Byts	0.0445
Bwd Pkts/s	0.0425
Flow IAT Max	0.0416
Flow IAT Mean	0.0376
Fwd IAT Tot	0.0349
Flow Duration	0.0321
Fwd IAT Max	0.0298
Fwd IAT Mean	0.0294
Bwd Seg Size Avg	0.0274
Flow IAT Min	0.0229
Fwd Header Len	0.0228
Fwd IAT Min	0.0211
Init Bwd Win Byts	0.0200
Bwd Header Len	0.0195
Fwd Act Data Pkts	0.0192
Subflow Fwd Byts	0.0172
PSH Flag Cnt	0.0163
ECE Flag Cnt	0.0134
Bwd Pkt Len Mean	0.0132
Pkt Size Avg	0.0124
Fwd Pkt Len Mean	0.0123
TotLen Fwd Pkts	0.0118
ACK Flag Cnt	0.0108
Fwd Pkt Len Std	0.0102
Subflow Fwd Pkts	0.0099
RST Flag Cnt	0.0096
Bwd Pkt Len Max	0.0093
Tot Fwd Pkts	0.0092
Fwd Pkt Len Max	0.0085
URG Flag Cnt	0.0075
Tot Bwd Pkts	0.0074
Flow Byts/s	0.0063
Fwd Seg Size Avg	0.0061
Pkt Len Std	0.0057
Subflow Bwd Byts	0.0050

Features with importance values below 0.0005—such as *Active Max*, *Active Mean*, *FIN Flag Count*, *Idle Std*, and others—were excluded from Table I for clarity, as they exhibited negligible contribution and were consistently close to zero. It was built on keras sequential model with 1D convolution layer with 64 filters and ReLU activation, two LSTM layers with 64 and 32 units, respectively, dense layers with 32 units and ReLU activation followed by a dropout layer to avoid overfitting, and a final dense layer with a softmax activation to output class probabilities.

The dataset, consisting of multiple CSV files representing different types of network traffic data, was loaded into the Python environment for processing and analysis. Data preprocessing was performed to transform the dataset into a meaningful and machine-learning-ready format. The process started by merging the individual CSV files into a single DataFrame, followed by filling missing values using the median strategy. The label column was encoded into integers using LabelEncoder. Feature selection was carried out to identify and retain the most relevant features in building the model. The selected features shown in Table II, with graphical representation in Fig. 4, were normalized using StandardScaler, as depicted in Fig. 5, to ensure that all features contribute equally to model development. The normalized dataset was split into training

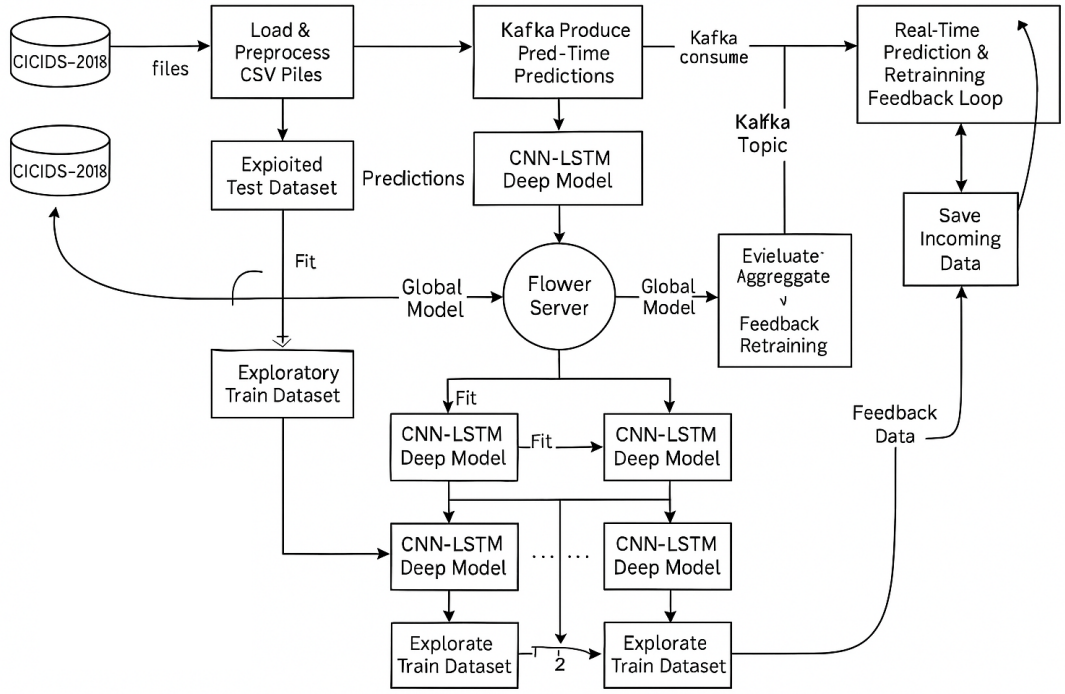

Fig. 2. The Process Flow of the HyFlaNK Hybrid Model

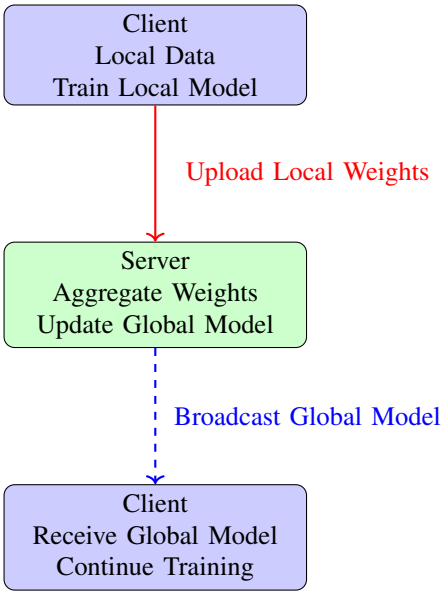

Fig. 3. Workflow showing client-side local training and server-side aggregation in HyFlaNK

and test datasets in an 80:20 ratio, respectively.

The system employed the flower Federated Learning framework, to collaboratively train a machine learning model. The federated learning model aggregated client updates using the FedAvg strategy, triggered evaluations after each round using `evaluate_fn` function. Each client ran in sequence training the global model on its local data and participating in weight aggregation. Model saving and version management, model

TABLE II
FEATURE IMPORTANCES AFTER SELECTION

Feature	Importance
Dst Port	0.1129
Fwd Seg Size Min	0.0667
Flow Pkts/s	0.0658
Fwd Pkts/s	0.0513
Init Fwd Win Byts	0.0445
Bwd Pkts/s	0.0425
Flow IAT Max	0.0416
Flow IAT Mean	0.0376
Fwd IAT Tot	0.0349
Flow Duration	0.0321

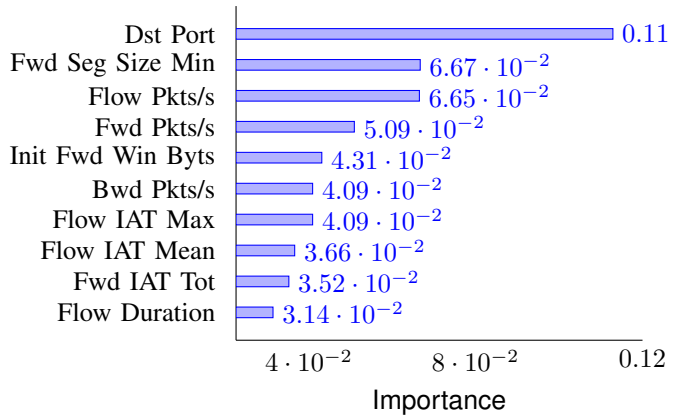

Fig. 4. Top-ranked features after selection based on their predictive importance

evaluation and metrics logging, and global model versioning were integrated into the model. This saved the global model with a unique version number to avoid overwriting existing models. The model integrated a kafka-based feedback loop to collect model predictions, gather feedback, and use it to periodically retrain the federated model.

The Kafka producer, responsible for streaming predictions, sent data to the Kafka topic `model_predictions` to enable decoupled and real-time streaming of results for feedback collection. The Kafka consumer listened to this topic, which contained both the actual class labels and the predicted classes. Upon ingestion, the predicted data was placed into a queue for visualization. The predictions streamed into Kafka were reused for incremental training, supporting active and self-learning in the model. This setup allowed for continuous model improvement, supporting both local and global evaluations. This approach facilitates the development of robust, distributed machine learning systems that can adapt and learn from real-time feedback.

IV. Result and Discussion

Table III presents the descriptive statistics of the selected features. The mean value represents the average value across all samples, while the standard deviation represents a measure of how much the values for that feature vary from the mean.

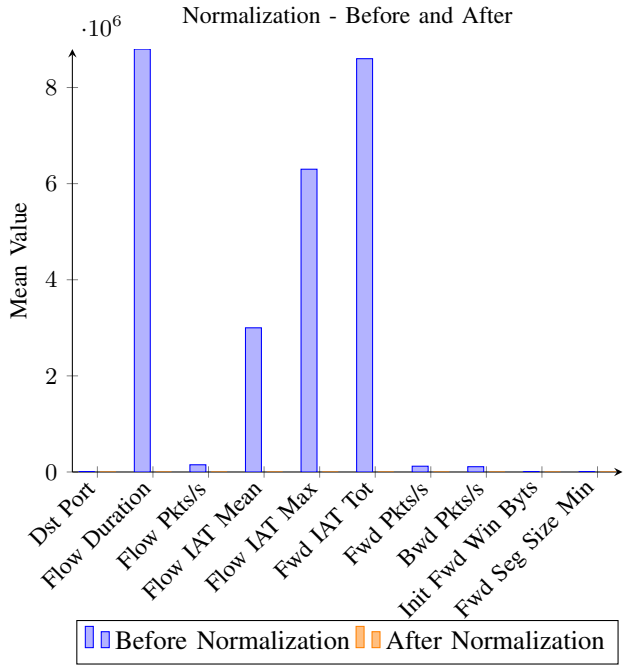

Fig. 5. Normalised Features

These statistical measures are integral in analyzing the behavioral patterns of the network, which directly inform the accuracy and performance of the HyFlaNK detection model.

The performance of each client, after training on their respective portions of the training dataset, is shown in the confusion matrix presented in Fig. 6. The confusion matrix revealed strong classification performance. The high values in true positives and true negatives suggest that the model effectively identifies network threats and normal traffic patterns.

TABLE III
Descriptive Statistics of Selected Features

Feature	Mean	Std Dev
Dst Port	6649.85	14912.79
Flow Duration	8884881.67	891418328.30
Flow Pkts/s	143513.87	426953.83
Flow IAT Mean	2921889.94	235943745.90
Flow IAT Max	6315891.83	1082044398.00
Fwd IAT Tot	8593165.05	891413467.30
Fwd Pkts/s	79375.08	254132.19
Bwd Pkts/s	64136.28	200496.68
Init Fwd Win Byts	8713.74	12959.84
Fwd Seg Size Min	20.37	9.11

Table IV provides a comprehensive performance analysis of each individual client that participated in the training process. The accuracy metric reflects the proportion of network attacks that were correctly predicted by the model. This is a key indicator of the effectiveness of HyFlaNK in identifying attack patterns.

TABLE IV
Performance Analysis of the Clients

Client	Accuracy	Loss	val_accuracy	val_loss
1	0.9988	0.006	0.9979	0.007
2	0.9986	0.007	0.9991	0.005
3	0.9987	0.006	0.9992	0.003
4	0.9986	0.007	0.9994	0.003
5	0.9985	0.007	0.9993	0.005

The results indicate that each client in the training process achieved impressive classification performance. The training loss for each client's model measures the difference between the predictions and the actual results. This loss tends toward zero for each client, indicating that the model's predictions became increasingly accurate as training progressed.The lower this value, the better the model. The validation accuracy (val_accuracy) and validation loss (val_loss) reflect the model's performance on data that was not used during training, along with the corresponding training loss. The validation loss helps monitor overfitting and underfitting. The model demonstrates robustness and stability, as evidenced by consistent validation performance, indicating it generalizes well to unseen data.

Fig. 7 shows the confusion matrices for each federated training round in the global model, illustrating the model's performance at each stage of the training process. The model correctly identified most of the positive and negative classes, with only a small number of false positives and false negatives, indicating high precision and recall. Fig. 8 presents a plot of accuracy versus loss across the global training rounds, highlighting the model's learning progression over time.

The graph demonstrates that the model's accuracy steadily increased while the loss decreased over the training epochs. Training accuracy improved consistently, and training loss reduced, indicating that the model learned effectively and progressively refined its performance.

Table V presents a performance comparison between the Local Models (LM) and the aggregated Global Model (GM)

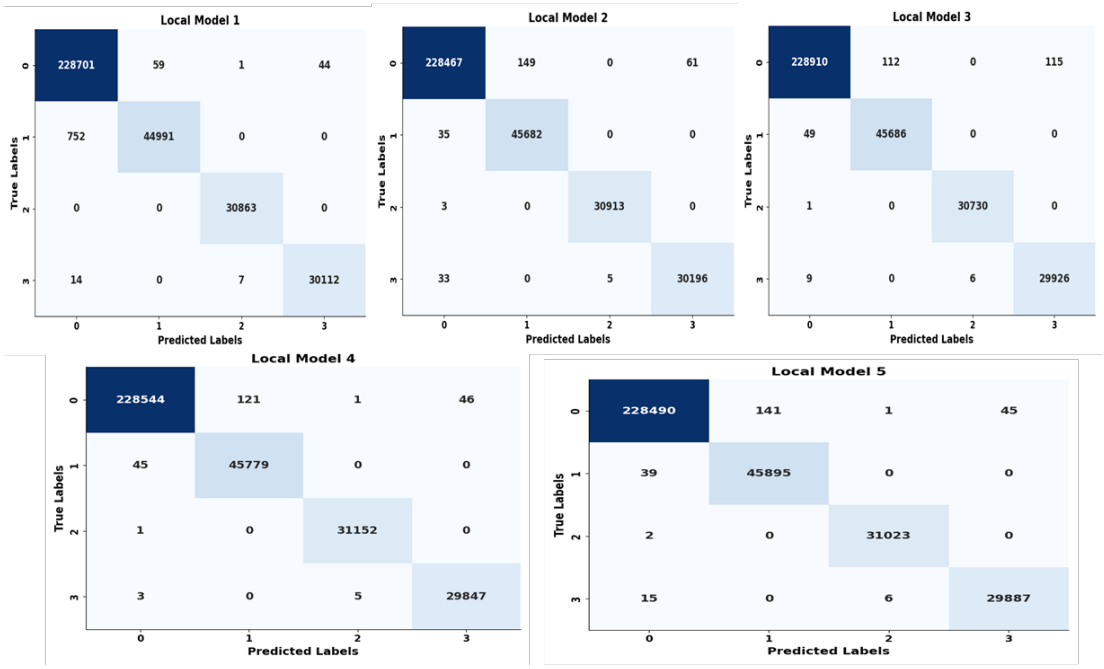

Fig. 6. Confusion Matrices of the Clients in the Training Rounds

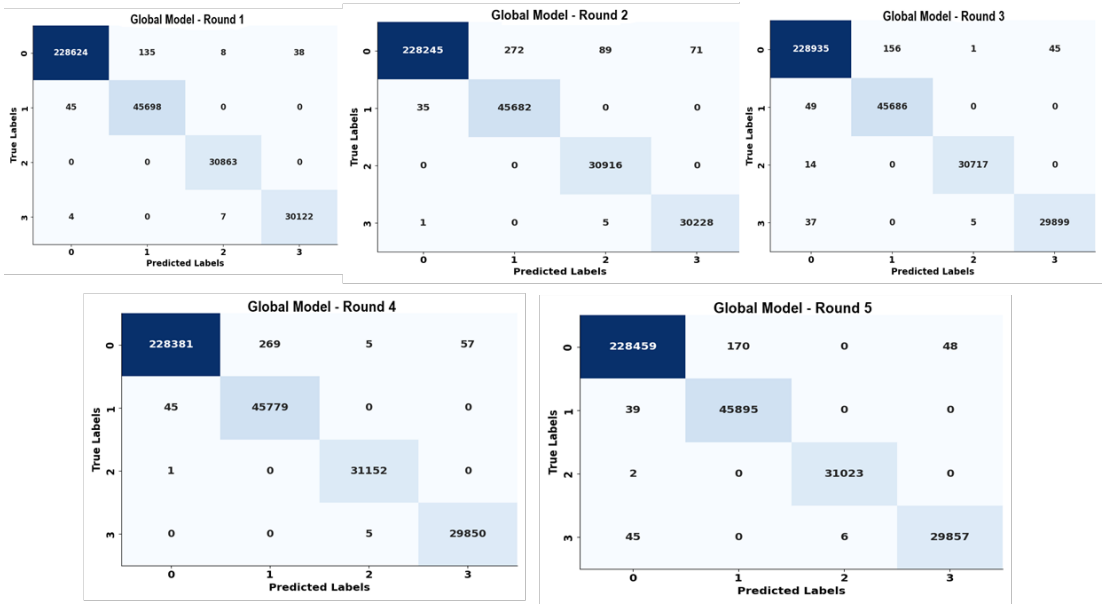

Fig. 7. Confusion Matrices for each Global Model Training Round

across training rounds. The comparison includes metrics such as accuracy, precision, recall, F1-score, ROC-AUC, and loss, providing a comprehensive evaluation of the models' performance throughout the training process in the HyFlaNK system..

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

TABLE V
PERFORMANCE COMPARISON OF LOCAL MODELS AND GLOBAL MODEL ACROSS TRAINING ROUNDS

Model	Accuracy	Prec.	Recall	F1	roc_auc	loss
LM 1	0.9974	0.9974	0.9973	0.9974	1.0000	0.0083
LM 2	0.9991	0.9991	0.9991	0.9991	1.0000	0.0044
LM 3	0.9991	0.9991	0.9991	0.9991	1.0000	0.0034
LM 4	0.9993	0.9993	0.9993	0.9993	1.0000	0.0037
LM 5	0.9993	0.9993	0.9993	0.9993	1.0000	0.0043
GM	0.9989	0.9990	0.9989	0.9989	1.0000	0.0023

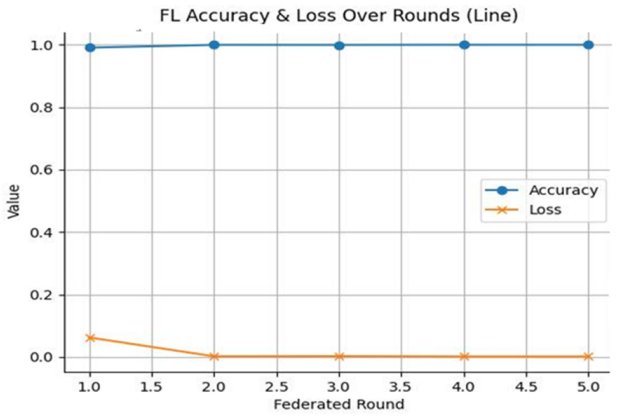

Fig. 8. Accuracy vs Loss over Global Rounds

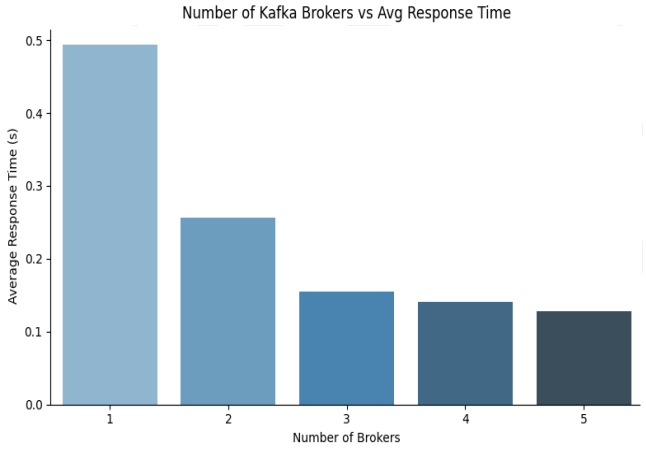

Fig. 9. Number of Brokers vs Average Response Time

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4}$$

$$AUC = \frac{1}{N_+ N_-} \sum_{i=1}^{N_+} \sum_{j=1}^{N_-} I(s_i > s_j) \tag{5}$$

Where TP means True Positives (correctly predicted positive instances), TN is True Negatives (correctly predicted negative instances), FP is False Positives (negative instances incorrectly predicted as positive), FN means False Negatives (positive instances incorrectly predicted as negative), N_+ represents the number of positive samples, N_- represents the number of negative samples, s_i represents the score for a positive sample, s_j represents the score for a negative sample, $I(s_i > s_j)$ is the indicator function, equals 1 if $s_i > s_j$, otherwise 0.

From the table, Local Model 1 had an accuracy of 0.9974 which implies that it correctly predicted 99.74% instances of attacks, a precision of 0.9974 meaning that 99.74% of instances predicted as positive were actually positive, recall of 0.9973 meaning that 99.73% of the actual positive attacks were correctly identified. An f1-score of 0.9974 indicates a strong balance between precision and recall. A roc-auc of 1.0 indicate that the model perfectly distinguised between classes of attacks. A low value of loss 0.0083 indicates that the prediction of the model was very close to the true labels.These evaluation metrics remained consistent across the federated rounds, highlighting the model's stability and reliability throughout the training process.

Fig. 9 illustrates the relationship between the number of brokers and the average response time, showing how the response time changes as the number of brokers varies. The brokers are responsible for handling the ingestion, storage, and retrieval of messages in a distributed system while the average response time in seconds measures how long it takes for a message to travel through the system, from ingestion to retrieval. A lower response time indicates a more responsive system. As the number of broker increases, the average response time decreases. This implies that the model can scale to handle more incoming data streams without sacrificing performance.

Fig. 10 shows the relationship between the number of brokers and throughput. Throughput refers to the number of messages processed per second by the system. As the number of brokers increases, the system's ability to handle a larger volume of messages in real-time improves, resulting in higher throughput.

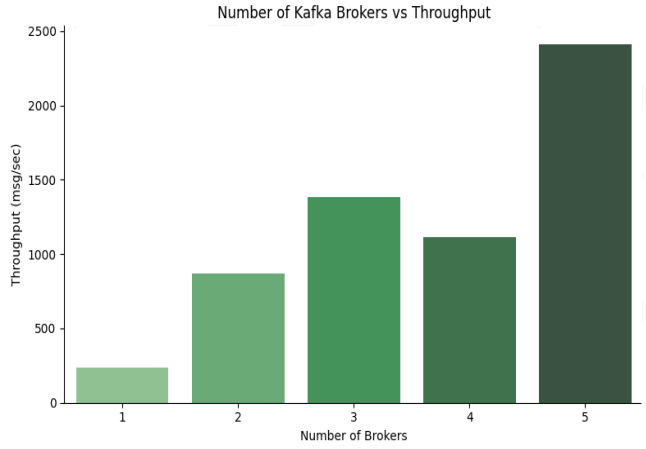

Fig. 10. Number of Brokers vs Throughput

A reduction in throughput was observed with the addition of the 4th broker, likely due to resource contention, such as network or CPU limitations. However, with the introduction of the 5th broker, system optimizations were implemented, leading to improved performance and a subsequent increase in throughput. Table VI provides a summary of the analysis offering a detailed overview of the system's performance across different broker configurations.

The system demonstrated the ability to process more messages per second, a crucial capability in high-velocity environments such as threat detection. This improvement in scalability and performance ensures that the system can efficiently process and analyze security data in real time, which is essential for swiftly detecting and responding to web-based threats. The enhanced throughput enables faster decision-

TABLE VI
SCALABILITY ANALYSIS OF KAFKA BROKERS: RESPONSE TIME VS THROUGHPUT

No. of Brokers	Av. Resp. Time(s)	Throughput *(msg/sec)*
1	0.49	233.80
2	0.26	871.36
3	0.15	1385.74
4	0.14	1113.08
5	0.13	2414.38

making, improving the system's effectiveness in mitigating potential security risks.

Both a qualitative comparison and a quantitative evaluation of the proposed HyFlaNK model is shown, highlighting its architectural advantages and empirical performance against a Random Forest-based threat detection model as the primary benchmark. The comparison emphasizes HyFlaNK's suitability for decentralized environments through federated learning, adaptive retraining, and enhanced detection performance.

A. Qualitative Comparison

Table VII presents a qualitative comparison between the proposed HyFlaNK architecture and the real-time threat detection framework developed by [35] focusing on architectural and operational capabilities relevant to dynamic threat detection environments. While the system integrates Kafka-based real-time streaming using a Random Forest model, it relies on a static, pre-trained model and lacks adaptive learning and privacy-preserving mechanisms. In addition, the approach does not leverage federated learning, limiting its scalability in decentralized settings. In contrast, HyFlaNK introduces a federated deep learning model (CNN+LSTM) integrated with Kafka for real-time streaming and includes a self-learning feedback loop. This enables continuous model improvement without exposing sensitive data. While HyFlaNK supports adaptive retraining, the current experiments were performed on pre-partitioned dataset samples. Evaluation of system performance under evolving threat scenarios and newly emerging attacks is planned as part of future work. HyFlaNK also supports adaptive retraining and is designed to scale efficiently across multi-broker Kafka environments. These enhancements make HyFlaNK a more robust, scalable, and future-ready framework for real-time threat detection in distributed environments.

TABLE VII
QUALITATIVE COMPARISON OF JASPIN ET AL. [35] AND HYFLANK (PROPOSED)

Feature	*Jaspin et al. [35]*	*HyFlaNK (Proposed)*
Federated Learning (FL)	No	Yes
Real-time Streaming	Yes	Yes
Feedback Mechanism	No	Yes (adaptive / self-learning)
Privacy Preservation	No	Yes
Model Update	Static (pre-trained)	Adaptive (online retraining)
Learning Model Type	Random Forest (ML)	CNN+LSTM (Deep Learning)
Scalability	Moderate	High
Accuracy	99.4%	99.89%
ROC-AUC	Not Reported	1.000

B. Quantitative Evaluation

Table VIII presents a quantitative evaluation between the proposed HyFlaNK model and the baseline Random Forest model used by [35]. The evaluation was based on standard performance metrics including Accuracy, Precision, Recall, F1-score, ROC AUC, and Loss.

TABLE VIII
QUANTITATIVE EVALUATION OF RANDOM FOREST AND HYFLANK (PROPOSED)

Model	Accuracy	Precision	Recall	F1-score	ROC AUC	Loss
Random Forest	0.9761	0.9798	0.9575	0.9686	0.9991	0.0031
HyFlaNK (Proposed)	0.9989	0.9990	0.9989	0.9989	1.0000	0.0023

HyFlaNK outperforms the Random Forest model across all metrics, achieving an accuracy of 99.89%, compared to 97.61% for Random Forest. Similarly, HyFlaNK achieves near-perfect scores in precision (99.90%), recall (99.89%), and F1-score (99.89%), while also attaining a perfect ROC AUC of 1.000 and a lower loss of 0.0023. In contrast, the Random Forest model exhibits reduced detection strength, with lower recall (95.75%) and a higher loss of 0.0031, indicating comparatively less reliable predictions. These results demonstrate the superior detection performance, generalization, and robustness of the HyFlaNK architecture in real-time web threat environments.

V. CONCLUSION AND FUTURE WORKS

The research introduced HyFlaNK, a federated learning system integrated with real-time streaming technologies, marking a significant advancement toward decentralized and adaptive threat detection. The HyFlaNK model not only ensures data privacy but also supports asynchronous learning, making it ideal for environments with sensitive or distributed datasets. In particular, HyFlaNK preserves privacy by keeping raw data on local clients, transmitting only model updates to the server. Increasing the number of participating clients further enhances privacy, as each client contributes only a small portion of the dataset, limiting potential exposure of individual records. HyFlaNK is scalable, privacy-preserving, and has demonstrated high accuracy and responsiveness without the need to centralize data. This architecture highlighted the effectiveness of streaming infrastructure, specifically Kafka, in machine learning workflows within HyFlaNK, where model predictions, retraining triggers, and feedback are handled in near real-time. The modular threading approach used by HyFlaNK for the server, feedback loop, and Kafka consumer/producer improves concurrency, allowing the system to remain resilient under heavy load.

Furthermore, the capabilities of HyFlaNK can be expanded by integrating with mobile applications and implementing robust mechanisms for handling broker failures or model corruption. These enhancements would further strengthen the system's resilience, making it more adaptable to dynamic environments and ensuring continuous operation even in the face of potential system disruptions.

Additionally, while the current study focuses on DDoS detection using the CICIDS-2018 dataset, the hybrid

CNN–LSTM architecture employed in HyFlaNK is designed to capture both spatial feature representations and temporal traffic patterns, making it adaptable to a broader range of cyber threats such as brute-force, botnet, and infiltration attacks. Therefore, comparable performance trends are expected when evaluated on datasets containing diverse attack categories, provided appropriate retraining is conducted. Future investigations will assess the generalizability of HyFlaNK to datasets containing diverse threat types. This includes addressing potential biases between regular and malicious traffic and implementing preprocessing or weighting strategies to ensure robust detection performance across varying threat distributions. Furthermore, future work will experimentally evaluate the adaptive retraining capability of HyFlaNK by introducing previously unseen attack types during the training process to simulate evolving threat scenarios. This will enable systematic assessment of incremental learning performance, adaptation speed, and potential catastrophic forgetting in federated environments, thereby providing quantitative validation of the self-learning feedback mechanism under realistic dynamic threat conditions.

REFERENCES

[1] A. J. Mahardhani, "The role of public policy in fostering technological innovation and sustainability," *Journal of Contemporary Administration and Management (ADMAN)*, 2023, vol. 1, no. 2, pp. 27–53.

[2] Singh, A. P., Tyagi, M., Vasistha, G., Sikarwar, N., Malik, N., Singh, M. K., & Verma, V. K., "Social media application development," in *Applications of Artificial Intelligence in 5G and Internet of Things*, CRC Press, 2025, pp. 1-6.

[3] Masoud and Basahel, "The effects of digital transformation on firm performance: The role of customer experience and IT innovation," *Digital*, 2023, vol. 3, no. 2, pp. 109–126.

[4] F. Candelon and M. Reeves, *The rise of AI-powered companies*, 2022, Walter de Gruyter GmbH & Co KG.

[5] L. Chen, P. Chen, and Z. Lin, "Artificial intelligence in education: A review," *IEEE Access*, 2020, vol. 8, pp. 75264-75278, doi: 10.1109/ACCESS.2020.2988510.

[6] Aldoseri, A., Al-Khalifa, K. N., and Hamouda, A. M., "AI-powered innovation in digital transformation: Key pillars and industry impact," *Sustainability*, 2024, vol. 16, no. 5, doi.org/10.3390/su16051790.

[7] S. Kaur, J. Singla, L. Nkenyereye, S. Jha, D. Prashar, G. P. Joshi, S. El-Sappagh, M. S. Islam, and S. M. R. Islam, "Medical diagnostic systems using artificial intelligence (AI) algorithms: Principles and perspectives," *IEEE Access*, 2020, vol. 8, pp. 228049-228069, doi: 10.1109/ACCESS.2020.3042273.

[8] S. Paul, M. Riffat, A. Yasir, M. N. Mahim, B. Y. Sharnali, I. T. Naheen, and Kulkarni, A., "Industry 4.0 applications for medical/healthcare services," *Journal of Sensor and Actuator Networks*, 2021, vol. 10, no. 3, 43, doi.org/10.3390/jsan10030043.

[9] O. S. Al-Mushayt, "Automating e-government services with artificial intelligence," *IEEE Access*, 2019, vol. 7, pp. 146821-146829, doi: 10.1109/ACCESS.2019.2946204.

[10] M. Fossi, G. Egan, K. Haley, E. Johnson, T. Mack, T. Adams, and P. Wood. "Symantec internet security threat report trends for 2010," *Volume XVI*, 2011.

[11] M. Shafiq, Z. Gu, O. Cheikhrouhou, W. Alhakami, and H. Hamam, "The rise of 'Internet of Things': Review and open research issues related to detection and prevention of IoT-based security attacks," *Wireless Communications and Mobile Computing*, 2022, vol. 2022, no. 1, p. 8669348, doi: 10.1155/2022/8669348.

[12] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks*, 2007, vol. 51, no. 12, pp. 3448–3470, doi: 10.1016/j.comnet.2006.11.001.

[13] M. A. Almaiah, R. B. Sulaiman, U. Islam, Y. Badr, and F. A. El-Qirem, "Federated Learning in Healthcare: A Bibliometric Analysis of Privacy, Security, and Adversarial Threats (2021–2024)," *SHIFRA*, 2025, vol. 2025, pp. 46–61.

[14] P. K. Myakala, P. Naayini, and S. Kamatala, "A Survey on Federated Learning for TinyML: Challenges, Techniques, and Future Directions," *Partners Universal International Innovation Journal*, 2025, vol. 3, no. 2, pp. 97–114.

[15] K. Shaukat, S. Luo, V. Varadharajan, I. A. Hameed, and M. Xu, "A survey on machine learning techniques for cyber security in the last decade," *IEEE Access*, 2020, vol. 8, pp. 222310–222354.

[16] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artificial Intelligence and Statistics (AISTATS)*, 2017, pp. 1273–1282.

[17] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2019, vol. 10, no. 2, pp. 1–19.

[18] B. Liang, J. Cai, and H. Yang, "A new cell group clustering algorithm based on validation & correction mechanism," *Expert Systems with Applications*, 2022, vol. 193, pp. 116410.

[19] I. Kholod, E. Yanaki, D. Fomichev, E. Shalugin, E. Novikova, E. Filippov, and M. Nordlund, "Open-source federated learning frameworks for IoT: A comparative review and analysis," *Sensors*, vol. 21, no. 1, p. 167, 2020.

[20] M. Hkima, I. Boudali, and T. Abdellatif, "Federated learning frameworks: A survey," in *2024 IEEE/ACS 21st International Conference on Computer Systems and Applications (AICCSA)*, 2024, pp. 1–6.

[21] B. Yin, H. Yin, Y. Wu, and Z. Jiang, "FDC: A secure federated deep learning mechanism for data collaborations in the Internet of Things," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6348–6359, 2020.

[22] T. Long and Q.-S. Jia, "Matching uncertain renewable supply with electric vehicle charging demand—A bi-level event-based optimization method," *Complex System Modeling and Simulation*, 2021, vol. 1, no. 1, pp. 33–44.

[23] H. Zhou, G. Yang, H. Dai, and G. Liu, "PFLF: Privacy-preserving federated learning framework for edge computing," *IEEE Transactions on Information Forensics and Security*, 2022, vol. 17, pp. 1905–1918.

[24] R. M. Rajendran and B. Vyas, "Detecting APT using machine learning: Comparative performance analysis with proposed model," in *Southeast-Con 2024*, 2024, pp. 1064–1069.

[25] N. Narkhede, G. Shapira, and T. Palino, *Kafka: The Definitive Guide: Real-Time Data and Stream Processing at Scale*, O'Reilly Media, Inc., 2017.

[26] N. Dey, R. Deepika, K. Tekuri, and U. Sanjana, "Advancements in Machine Learning for Anomaly Detection in Cyber Security," in *International Conference on Intelligent Computing and Big Data Analytics*, 2024, pp. 163–178, Springer.

[27] S. Rani, "Tools and techniques for real-time data processing: A review," *International Journal of Science and Research Archive*, 2025, vol. 14, no. 1, pp. 1872–1881.

[28] R. Manchana, "Event-Driven Architecture: Building Responsive and Scalable Systems for Modern Industries," *International Journal of Science and Research (IJSR)*, 2021, vol. 10, no. 1, pp. 1706–1716.

[29] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[30] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2019, vol. 10, no. 2, pp. 1–19.

[31] A. Dehlaghi-Ghadim, T. Markovic, M. Leon, D. Söderman, and P. E. Strandberg, "Federated learning for network anomaly detection in a distributed industrial environment," in *2023 International Conference on Machine Learning and Applications (ICMLA)*, 2023, pp. 218–225.

[32] S. Rajesh, M. Clement, S. B. Sooraj, S. H. Al Shifan, and J. Jyothi, "Real-time DDoS attack detection based on machine learning algorithms," *Proceedings of the Yukthi*, 2021.

[33] M. Evangelou and N. M. Adams, "An anomaly detection framework for cyber-security data," *Computers & Security*, 2020, vol. 97, pp. 101941.

[34] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, and others, "Advances and open problems in federated learning," *Foundations and Trends in Machine Learning*, 2021, vol. 14, no. 1–2, pp. 1–210.

[35] K. Jaspin, Y. S. Shinanth, and others, "Real Time Network Threat Detection Using Machine Learning and Kafka in Middleware," in *2024 4th International Conference on Ubiquitous Computing and Intelligent Information Systems (ICUIS)*, 2024, pp. 1423–1429. doi: 10.1109/ICUIS64676.2024.10867200.

[36] S. Zhao, M. Chandrashekar, Y. Lee, and D. Medhi, "Real-time network anomaly detection system using machine learning," in *2015 11th Inter-*

national Conference on the Design of Reliable Communication Networks (DRCN), 2015, pp. 267–270. doi: 10.1109/DRCN.2015.7149025.

[37] S. T. Musharraf, Z. Asif, M. Saleem, M. Z. Hussain, and M. Z. Hasan, "A hybrid lightweight and explainable federated learning model for real-time intrusion detection in resource-constrained IoT environments," *Spectrum of Engineering Sciences*, pp. 490–500, 2025.

[38] N. Chaurasia, M. Ram, P. Verma, N. Mehta, and N. Bharot, "A federated learning approach to network intrusion detection using residual networks in industrial IoT networks," *The Journal of Supercomputing*, vol. 80, no. 13, pp. 18325–18346, 2024.

[39] M. A. Hossain and M. S. Islam, "Towards decentralized cybersecurity: A novel privacy-preserving federated learning approach for botnet attack detection," *Blockchain: Research and Applications*, p. 100355, 2025.

[40] G. Singh, K. Sood, P. Rajalakshmi, and Y. Xiang, "Sentinel: Dynamic knowledge distillation for personalized federated intrusion detection in heterogeneous IoT networks," in *IEEE Internet of Things Journal*, 2026.

[41] H. Peng, C. Wu, and Y. Xiao, "Fd-ids: Federated learning with knowledge distillation for intrusion detection in non-iid iot environments," in *Sensors*, vol. 25, no. 14, pp. 4309, 2025.

[42] S. Ghosh, A. S. M. M. Jameel, and A. E. Gamal, "FetFIDS: A Feature Embedding Attention based Federated Network Intrusion Detection Algorithm," in *arXiv preprint arXiv:2508.09056*, 2025.

[43] A. Karunamurthy, K. Vijayan, P. R. Kshirsagar, and K. T. Tan, "An optimal federated learning-based intrusion detection for IoT environment," in *Scientific Reports*, vol. 15, no. 1, pp. 8696, 2025.

[44] M. A. Lopez, A. G. Pastana, O. C. M. B. Duarte, and G. Pujolle, "An evaluation of a virtual network function for real-time threat detection using stream processing," in *2018 Fourth International Conference on Mobile and Secure Services (MobiSecServ)*, 2018, pp. 1–5. doi: 10.1109/MOBISECSERV.2018.8311440.

[45] H. M. Farooq and N. M. Otaibi, "Optimal machine learning algorithms for cyber threat detection," in *2018 UKSim-AMSS 20th International Conference on Computer Modelling and Simulation (UKSim)*, 2018, pp. 32–37. doi: 10.1109/UKSim.2018.00018.

Oluyemisi Adenike Oyedemi received her B.Tech. degree in Computer Engineering from Ladoke Akintola University of Technology, Oyo State, NIgeria in 2003; M.Sc. degree, and Ph.D. degree both in Computer Science from Obafemi Awolowo University, Ile-Ife, Osun State, Nigeria in 2017 and 2022 respectively. She is a senior lecturer in the Department of Computer Science, University of Ilesa, Ilesa, Osun State, Nigeria. She is on her postdoctoral studies with the Department of Computer Science, Federal University of Lavras, Brazil from November 2024 to date. Her research interests include Machine Learning, intelligent system, cybersecurity, and fraud detection systems. Dr. Oyedemi is a member of the Nigeria Computer Society (NCS), Association of Professional Women Engineers (APWEN), Nigeria Women in Information Technology (NIWIIT), Nigerian Society of Engineers (NSE), and Council for the Regulation of Engineering in Nigeria (COREN).

Renata Lopes Rosa received the M.S. degree from the University of São Paulo in 2009 and the Ph.D. degree from the Polytechnic School of the University of São Paulo in 2015 (EPUSP).

She is currently an Adjunct Professor with the Department of Computer Science, Federal University of Lavras, Brazil. She has more than ten years of professional experience. Her current research interests include computer networks, telecommunication systems, machine learning, quality of experience of multimedia service, cybersecurity, social networks, and recommendation systems. Prof. Rosa is a member of IEEE.

Ugochukwu Okwudili Matthew holds Masters in Computer Applications from Bayero University Kano, Nigeria. He is presently a Postgraduate student in the Department of Computer Science, Federal University of Lavras, Brazil. His research interest include Artificial Intelligence, Machine Learning, Big Data Science, Cloud Computing, Internet of Things, Data Mining, Multimedia and E-Learning Education. Mr. Ugochukwu is a member of Nigeria Computer Society (NCS), Nigeria Institute of Management (NIM), International Association of Computer Science & Information Technology (IACSIT), European Alliance for Innovation (EAI), International Association of Engineers of Computer Society (IAENG-CS) and Teaching & Education Research Association (TERA).

Demóstenes Zegarra Rodríguez received his B.S. degree in electronic engineering from the Pontifical Catholic University of Peru, the M.S. degree, and the Ph.D. degree from the University of São Paulo in 2009 and 2013, respectively. In 2018–2019, he had a post-doctoral position at the Technical University of Berlin, specifically in the Quality and Usability Laboratory. He is currently an Adjunct Professor with the Department of Computer Science, Federal University of Lavras, Brazil. He has solid knowledge in Telecommunication Systems and Computer Science based on 15 years of professional experience in major companies. His research interests include QoS and QoE in multimedia services, architect solutions in telecommunication systems, intrusion detection systems, and cybersecurity. Prof. Demostenes is a member of the Brazilian Telecommunications Society, and a senior member of IEEE.