JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

DECENTRALIZED DDOS DETECTION USING FEDERATED LEARNING FOR PRIVACY-ENHANCED CLOUD SECURITY

Dr. K. Sasikala
Department of Information Technology R P Sarathy Institute of Technology Salem, India
emailtosasi@gamil.com

Saravanan T
Department of Information Technology R P Sarathy Institute of Technology Salem, India
dtsaran.asm@gmail.com

Soniya P
Department of Information Technology R P Sarathy Institute of Technology Salem, India
soniyavpg@gmail.com

Gowtham T
Department of Information Technology R P Sarathy Institute of Technology Salem, India harigowtham176@gamil.com

Abishek S
Department of Information Technology R P Sarathy Institute of Technology Salem, India
sanjais82005@gmail.com

Gowrishankar S
Department of Information Technology R P Sarathy Institute of Technology Salem, India
gs5837354@gmail.com

Pugalmani M
Department of Information Technology R P Sarathy Institute of Technology Salem, India mnithi160000@gmail.com

Abstract : ***Distributed Denial-of-Service (DDoS) attacks represent one of the most critical threats to modern cloud computing infrastructures. Traditional centralized intrusion detection systems require large-scale traffic aggregation, which introduces privacy risks, regulatory compliance issues, and single-point-of-failure vulnerabilities. This work proposes a privacy-preserving cloud security framework leveraging Federated Learning (FL) for collaborative DDoS detection. The proposed architecture enables distributed cloud nodes to train local deep learning models while sharing only encrypted model updates with a central aggregation server. Secure aggregation and differential privacy mechanisms are integrated to prevent information leakage. Experimental evaluation demonstrates that the FL-based approach achieves high detection accuracy above 97% while maintaining data confidentiality and scalability. The results validate the feasibility of deploying.***

IndexTerms **- Federated Learning, DDoS Detection, Cloud Security, Privacy Preservation, Intrusion Detection Systems (IDS), Secure Aggregation, Distributed Systems, Deep Learning, Network Security, Differential Privacy.**

I. INTRODUCTION

Cloud computing has significantly transformed the modern information technology ecosystem by providing scalable, flexible, and cost-effective infrastructure for organizations of all sizes. It allows businesses to access computing resources such as storage, processing power, and applications over the internet without the need for maintaining physical hardware. This has led to increased efficiency, reduced operational costs, and improved accessibility of services across the globe. Organizations can dynamically scale their resources up or down based on demand, making cloud computing highly adaptable to changing workloads.

Furthermore, cloud environments support a wide range of applications, from simple data storage to complex machine learning systems. Despite these advantages, cloud computing introduces several security challenges due to its underlying architecture. One of the major concerns is its distributed nature, where data and services are spread across multiple servers and locations. Another critical aspect is the multi-tenant model, where multiple users share the same infrastructure. While this improves resource utilization, it also increases the risk of security breaches and cyberattacks. Among the various types of cyber threats, Distributed Denial-of-Service (DDoS) attacks are particularly dangerous for cloud environments. A DDoS attack involves multiple compromised systems working together to flood a target system with excessive traffic. The primary goal of such attacks is to exhaust system resources such as bandwidth, memory, and processing power. As a result, legitimate users are unable to access services, leading to service disruption. These attacks can cause significant financial losses for organizations due to downtime and reduced customer trust. In cloud environments, the impact of DDoS attacks can be even more severe due to the shared nature of resources.

Attackers can exploit the scalability of cloud systems to amplify their attacks. For example, they may target multiple virtual machines simultaneously to maximize disruption. Additionally, identifying the source of attacks becomes more complex in

distributed systems. To mitigate such […] regulations have become stricter worldwide. Organizations are required to ensure the privacy and security of user data. These regulations often restrict the sharing of raw data across different systems or regions. As a result, organizations cannot freely exchange network traffic logs between distributed nodes. This creates a challenge for implementing effective security measures in cloud environments. Traditional centralized approaches are no longer suitable under these constraints. There is a growing need for solutions that can provide strong security while maintaining data privacy.

Federated Learning (FL) has emerged as a promising approach to address these challenges. It is a decentralized machine learning technique that allows multiple participants to collaboratively train a model. In this approach, data remains on local devices or servers instead of being shared. Each participant trains a local model using its own data. Only the model parameters or updates are sent to a central server for aggregation. The central server combines these updates to create a global model. This process is repeated iteratively to improve the model's performance. By avoiding the transfer of raw data, FL ensures data privacy and security. Federated Learning is particularly suitable for distributed environments like cloud computing. It reduces the need for large-scale data transfer, thereby minimizing bandwidth usage. It also addresses privacy concerns by keeping sensitive information localized. Additionally, FL can scale efficiently as more participants join the system. Each node contributes to the learning process without exposing its data. This makes it an ideal solution for collaborative learning across multiple organizations or cloud nodes. Integrating Federated Learning into DDoS detection systems offers several advantages. First, it enables the detection of attacks using data from multiple sources without compromising privacy. Each cloud node can analyze its own network traffic and identify potential threats.

The insights gained from different nodes are then combined to improve overall detection accuracy. This collaborative approach enhances the system's ability to detect complex and distributed attacks. Second, FL improves scalability, as the computational load is distributed across multiple nodes. This reduces the burden on any single system and allows for faster processing. Third, it minimizes the risk of data leakage, as raw traffic data is never shared. Another important benefit of using FL in DDoS detection is adaptability.

Cloud environments are dynamic, with constantly changing workloads and traffic patterns. Federated Learning models can continuously update themselves based on new data. This allows the system to adapt to emerging attack patterns and evolving threats. It also improves the robustness of the detection system over time.

However, implementing Federated Learning is not without challenges. One issue is the heterogeneity of data across different nodes. Each node may have different traffic patterns and data distributions. This can affect the performance of the global model. Another challenge is communication overhead, as model updates must be exchanged frequently. Ensuring secure communication between nodes is also critical. Despite these challenges, ongoing research continues to improve the efficiency and reliability of FL systems. This research focuses adapts the learning path based on structured per-session feedback. User scoped OAuth integration ensures that generated content including Google Drive documents and YouTube playlists is created directly within each learner's own account, preserving ownership and supporting long-term engagement.

Key Contributions

The principal contributions of this work are as follows:

- One of the major contributions of this work is the development of a privacy-preserving DDoS detection system using Federated Learning (FL). In traditional systems, raw network traffic data must be shared with a central server for analysis, which creates serious privacy risks and may violate data protection regulations. This research eliminates that issue by ensuring that sensitive data remains within local cloud nodes. Instead of transferring raw data, only trained model parameters or updates are shared with a central aggregator. This approach not only protects user confidentiality but also aligns with modern data privacy laws and organizational security policies, making it highly suitable for real-world cloud environments.
- Another key contribution is the design of a fully decentralized intrusion detection framework tailored for cloud computing environments. Unlike conventional centralized Intrusion Detection Systems (IDS), which rely on a single point of monitoring and control, this system distributes the detection process across multiple cloud nodes. Each node independently monitors its own network traffic, trains a local model, and contributes to a global detection system. This distributed architecture removes the single point of failure, increases system robustness, and ensures continuous operation even if some nodes are compromised or fail. It also better matches the inherently distributed nature of cloud infrastructures.
- The proposed approach significantly improves scalability, which is a critical requirement in modern cloud environments that handle massive volumes of data. Centralized systems often struggle with processing delays, storage limitations, and network congestion when dealing with large-scale traffic. By leveraging Federated Learning, computation is distributed across multiple nodes, reducing the burden on any single system. Each node processes its own data locally, leading to faster analysis and reduced latency. This makes the system highly efficient and capable of scaling seamlessly as the number of users, devices, and data sources increases.
- This research enhances DDoS attack detection accuracy by utilizing collaborative learning across multiple cloud nodes. Each node observes different traffic patterns and potential attack behaviors, which may not be visible from a single location. By aggregating model updates from diverse nodes, the global model gains a more comprehensive understanding of both normal and malicious activities. This improves the system's ability to detect complex, distributed, and evolving attack patterns. As a result, the detection system becomes more robust, adaptive, and capable of identifying both known and unknown (zero-day) attacks with higher precision.
- A significant contribution of this work is the reduction in communication overhead and enhancement of overall system security. Traditional approaches require continuous transfer of large volumes of raw network data, which consumes bandwidth and increases exposure to interception or data breaches. In contrast, this approach only shares lightweight model parameters, significantly reducing network load. Additionally, since sensitive data never leaves the local node, the risk of data leakage is minimized. This not only improves efficiency but also strengthens the overall security posture of the cloud system against both internal and external threats.

The remainder of this paper is structured as follows. Section II reviews related work on DDoS defense, Federated Learning, and privacy-preserving intrusion detection. Section III presents the background and problem formulation, including the threat model. Section IV describes the proposed methodology. Section V presents the system architecture. Section VI reports the evaluation results. Section VII discusses the findings and limitations, and Section VIII concludes with directions for future work.

II. LITERATURE REVIEW

Cloud computing has become a foundational technology in modern information systems, offering scalable, flexible, and cost-efficient resources for data storage and processing. However, its distributed and multi-tenant architecture has introduced significant security challenges, particularly in the form of Distributed Denial-of-Service (DDoS) attacks. Existing literature highlights that DDoS attacks exploit the inherent scalability of cloud platforms to overwhelm system resources, leading to service disruptions and financial losses. Traditional Intrusion Detection Systems (IDS) have been widely studied as a defense mechanism, where machine learning and statistical approaches are used to identify abnormal traffic patterns. However, most of these systems rely on centralized data collection and analysis, which limits their effectiveness in large-scale cloud environments. Researchers have pointed out that centralized IDS frameworks suffer from high latency, bottlenecks, and single points of failure, making them less suitable for dynamic and distributed infrastructures.

In recent years, studies have increasingly focused on addressing privacy and scalability issues in cloud-based security systems. With the rise of strict data protection regulations, such as GDPR and similar policies, sharing raw network traffic data across multiple cloud nodes has become a major concern. Literature suggests that traditional collaborative learning approaches, which require data aggregation at a central server, are no longer feasible in privacy-sensitive environments. To overcome these limitations, decentralized and privacy-preserving techniques have gained attention. Among these, Federated Learning (FL) has emerged as a promising solution, allowing multiple participants to collaboratively train machine learning models without sharing raw data. Researchers have demonstrated that FL can significantly reduce privacy risks while maintaining model performance. Additionally, studies emphasize that FL-based systems can operate efficiently in distributed environments by leveraging local computation and minimizing data transfer.

Recent research has explored the integration of Federated Learning into DDoS detection systems within cloud environments. These studies show that FL-based approaches improve detection accuracy by aggregating knowledge from multiple distributed nodes, each observing different traffic patterns and attack behaviors. The literature also highlights that such collaborative models can effectively detect both known and emerging attack types, enhancing the robustness of security systems. Furthermore, FL reduces communication overhead and improves scalability compared to traditional centralized methods. Despite these advantages, some challenges remain, including handling heterogeneous data distributions, ensuring secure model aggregation, and managing communication efficiency. Overall, the literature indicates that integrating Federated Learning with cloud-based DDoS detection systems is a promising direction for achieving scalable, accurate, and privacy-preserving cybersecurity solutions.

A large body of work has addressed DDoS defense in cloud and data-center settings. Surveys of flooding-attack defense mechanisms [11] classify solutions by where they are deployed (source-end, victim-end, intermediate-network, and distributed) and by the detection technique they rely on. Early detectors were largely statistical, using traffic-volume thresholds, entropy of header fields, or change-point analysis to flag abnormal behavior. Although inexpensive, such detectors are sensitive to threshold selection and struggle with low-rate and application-layer attacks that resemble legitimate traffic. Classical machine learning classifiers, including support vector machines, decision trees, random forests, and k-nearest neighbors, improved adaptability but depend on hand-crafted flow features. More recently, deep neural networks such as multilayer perceptrons, convolutional networks, recurrent networks, and autoencoders have been applied to learn discriminative representations directly from flow statistics, usually with gains in detection accuracy at the cost of larger labeled training sets.

Federated Learning was introduced as a communication-efficient approach for training deep networks on decentralized data [12], and it has since been explored extensively for intrusion detection. Surveys of this area [4], [6] describe the main architectures, aggregation strategies, and open challenges. Application-specific studies report federated detectors for DDoS in 5G environments [2], for distributed denial-of-service detection in Internet of Things networks using federated deep learning [5], for cyber-physical systems with privacy preservation [7], and for intrusion detection based on privacy-preserving federated reinforcement learning [3]. A common observation across these works is that clients with different traffic profiles can jointly learn a detector that generalizes better than any single local model, while the raw logs never leave their owners.

Federated Learning by itself does not guarantee privacy, because model updates can leak information about the training data. Two families of techniques are therefore combined with it. Secure aggregation protocols allow a server to learn only the sum of client updates and not any individual contribution [13]. Differential privacy bounds how much any single record can influence the released model [14], and has been adapted to deep learning through gradient clipping and calibrated noise [15]. Both mechanisms introduce overhead and, in the case of differential privacy, a trade-off between privacy strength and model accuracy that must be tuned for the deployment.

Federated systems also face robustness challenges. Because the server cannot inspect client data, a compromised client may submit poisoned or backdoored updates [16], [17], and Byzantine-tolerant aggregation rules have been proposed to limit their influence [16]. In addition, traffic observed by different cloud nodes is rarely independent and identically distributed (non-IID), which slows convergence and can degrade the global model [18]; proximal regularization is one remedy for such heterogeneity [19]. Table 1 contrasts centralized and federated intrusion detection along the dimensions that matter most for cloud deployments.

Table 1. Comparison of centralized and federated intrusion detection

Aspect	Centralized IDS	Federated IDS (this work)
Data location	Raw traffic is collected at a central server	Raw traffic stays on each cloud node
Privacy exposure	High; logs are shared and stored centrally	Low; only protected model updates are shared
Single point of failure	Yes; the collector and the analyzer	No single detector; aggregation can be replicated
Communication cost	High; continuous transfer of raw data	Lower; periodic transfer of model parameters
Scalability	Limited by central storage and compute	Computation is distributed across nodes
Adaptation to new attacks	Central retraining on pooled data	Continuous local training with periodic aggregation
Main challenges	Latency, bottlenecks, regulatory limits	Non-IID data, poisoning, update overhead

In summary, existing studies establish the value of Federated Learning for intrusion detection, yet many treat privacy and detection accuracy separately or evaluate in IoT and 5G scenarios rather than in multi-tenant cloud infrastructures. The present work addresses cloud DDoS detection by combining local deep-learning detectors with secure aggregation and differential privacy inside a single decentralized framework, as described in the following sections.

III. BACKGROUND AND PROBLEM FORMULATION

A. DDoS Attack Taxonomy

DDoS attacks are commonly grouped by the resource they exhaust. Volumetric attacks saturate network bandwidth with massive traffic, often using reflection and amplification through public servers. Protocol attacks abuse weaknesses in network or transport protocols to exhaust connection tables, firewalls, or load balancers. Application-layer attacks send seemingly legitimate requests at a rate that overloads a web service or API, which makes them the hardest to separate from benign traffic. Table 2 summarizes these categories together with the traffic indicators that a learning-based detector can exploit.

Table 2. Common categories of DDoS attacks and their traffic indicators

Category	Examples	Target resource	Typical indicators
Volumetric	UDP flood, ICMP flood, DNS and NTP amplification	Network bandwidth	Sudden surge in packet and byte rates; few distinct ports
Protocol	SYN flood, ACK flood, ping of death	Connection state, firewalls, load balancers	High ratio of half-open connections; abnormal flag combinations
Application-layer	HTTP GET/POST flood, Slowloris	Web server threads and application resources	Many requests per source; long-lived, slow connections

B. Federated Learning Formulation

Consider K cloud nodes, where node k holds a private local dataset of n_k labeled traffic samples and n is the total number of samples over all nodes. Each sample consists of a feature vector x extracted from a network flow and a label y that marks the flow as benign or as belonging to an attack class. The goal is to learn a global parameter vector w that minimizes the weighted empirical loss

$$F(w) = \Sigma_k (n_k / n) F_k(w), \quad F_k(w) = (1 / n_k) \Sigma_i \ell(w; x_i, y_i) \tag{1}$$

where the inner sum runs over the samples of node k and ℓ is the classification loss. In round t, the server sends the current global model w^t to a subset S of available nodes. Each selected node performs E epochs of local mini-batch gradient descent starting from w^t; in its simplest form a single step is

$$w_k^{t+1} = w^t - \eta \nabla F_k(w^t) \tag{2}$$

where η is the learning rate. The server then aggregates the returned models by weighted averaging [12]:

$$w^{t+1} = \Sigma_{k \in S} (n_k / n_S) w_k^{t+1} \tag{3}$$

where n_S is the total number of samples held by the selected nodes. Only the parameters in Eqs. (2) and (3) travel between nodes and the server; the datasets themselves never do.

Table 3. Summary of notation

Symbol	Meaning	Symbol	Meaning
K	Number of participating cloud nodes	η	Learning rate
n_k, n	Samples at node k, total samples	E	Number of local epochs
w^t	Global model parameters in round t	C	Clipping norm
$F_k(w)$	Local loss of node k	σ	Noise multiplier
S	Set of nodes selected in a round	(ε, δ)	Differential privacy parameters
Δ_k	Model update of node k	y_k	Masked update sent to the server

C. Threat Model

We consider three types of adversary. First, the aggregation server is assumed to be honest-but-curious: it follows the protocol but may try to infer information about client data from the updates it receives. Second, an eavesdropper may observe the

communication between clients and the server. Third, a small fraction of clients may be compromised and submit poisoned updates in order to reduce detection accuracy or to create blind spots for particular attack types. The framework does not assume that the attacker can break standard cryptographic primitives, and it assumes that a majority of participating nodes behave honestly.

D. Design Goals

The framework is designed around four goals:

- **Privacy.** Raw traffic and individual model updates must not be exposed to the server or to other nodes.
- **Detection quality.** The global model should achieve high accuracy and a low false-positive rate for known attack types and for variants that were not seen during training.
- **Scalability and efficiency.** Training cost and communication volume should grow gracefully with the number of participating nodes.
- **Robustness.** The system should tolerate heterogeneous data, node failures, and a limited number of malicious participants.

IV. METHODOLOGY

A. System Overview. The proposed system is designed to detect Distributed Denial-of-Service (DDoS) attacks in cloud computing environments using a Federated Learning (FL) approach while ensuring data privacy and scalability. The system consists of multiple distributed cloud nodes, each responsible for monitoring and analyzing its own network traffic locally. These nodes act as independent participants that collect data, preprocess it, and train local machine learning models without sharing raw data with other nodes.

B. Each cloud node performs data preprocessing steps such as cleaning, feature extraction, and normalization to prepare the data for analysis. After preprocessing, a local model is trained to identify patterns of normal and malicious traffic. Instead of sending sensitive data to a central server, only the trained model parameters are transmitted. This ensures that user data remains secure and private within each node.

C. The proposed system focuses on detecting Distributed Denial-of-Service (DDoS) attacks in cloud computing environments using a Federated Learning (FL) approach while preserving data privacy. Initially, a distributed cloud environment is considered, where multiple nodes or servers generate and handle network traffic data independently. Each node collects its local network traffic data, including both normal and malicious patterns, without sharing it with other nodes.

D. In the next step, data preprocessing is performed locally at each node. This includes cleaning the data, removing noise, handling missing values, and extracting relevant features such as packet size, traffic rate, and protocol type. The processed data is then used to train a local machine learning model, such as a classification model, to identify potential DDoS attacks. Each node trains its model independently using its own dataset.

E. After local training, instead of sharing raw data, each node sends only the trained model parameters or updates to a central aggregation server. The server performs model aggregation, typically using techniques like averaging, to combine all local models into a global model. This global model captures knowledge from all participating nodes without compromising data privacy. The updated global model is then distributed back to all nodes for further training, and this process is repeated iteratively to improve performance.

F. Once the global model is sufficiently trained, it is used for real-time DDoS attack detection across the cloud environment. Each node uses the updated model to monitor incoming traffic and classify it as normal or malicious. If an attack is detected, appropriate mitigation strategies can be triggered to prevent service disruption. The performance of the system is evaluated using metrics such as accuracy, precision, recall, and detection rate.

G. This methodology ensures a scalable, efficient, and privacy-preserving approach to DDoS detection by combining distributed learning with collaborative model improvement.

H. Local Detection Model and Features

Each node converts captured packets into flow-level records and derives statistical features that describe the behavior of the traffic within short time windows. Typical features include packet rate, byte rate, flow duration, mean and variance of packet size, inter-arrival time, the distribution of protocols and TCP flags, the ratio of SYN to ACK packets, and the number of distinct source addresses and destination ports. Features are normalized locally, and the same feature schema is shared by all nodes so that model parameters remain compatible. The local detector is a compact deep neural network, for example a multilayer perceptron or a one-dimensional convolutional network, whose output layer predicts benign traffic or an attack class. A small model keeps the size of the transmitted update low and allows training on modest cloud instances or edge devices.

I. Privacy-Preserving Update Sharing

Sending model updates instead of raw data reduces exposure, but updates may still reveal information about individual samples. The framework therefore protects every update with two complementary mechanisms before it leaves the node.

Differential privacy. Each node first bounds the influence of its data by clipping the update $\Delta_k = w^{t+1} - w^t$ to a maximum L2 norm C, and then adds Gaussian noise:

$$\Delta_k^{clip} = \Delta_k / \max(1, \|\Delta_k\|_2 / C) \quad (4)$$

$$\Delta_k^{dp} = \Delta_k^{clip} + N(0, \sigma^2 C^2 I) \quad (5)$$

where σ is the noise multiplier. The resulting mechanism M satisfies (ε, δ)-differential privacy when, for any two neighboring datasets D and D′ and any set of outputs O,

$$\Pr[M(D) \in O] \leq e^{\varepsilon} \Pr[M(D') \in O] + \delta \quad (6)$$

A smaller ε means stronger privacy and typically lower accuracy, so ε is chosen as a deployment parameter [14], [15].

Secure aggregation. Before transmission, each node masks its update with pairwise random values that it agrees upon with the other participating nodes through key exchange. For nodes indexed by k, the masked update is

$$y_k = \Delta_k^{dp} + \Sigma_{j>k} \mathrm{PRG}(s_{kj}) - \Sigma_{j<k} \mathrm{PRG}(s_{jk}) \quad (7)$$

where s_{kj} is the secret shared between nodes k and j and PRG is a pseudorandom generator. When the server adds all masked updates, the pairwise masks cancel, so the server obtains only the sum of the updates and never an individual contribution [13]. Node dropouts are handled by the secret-sharing step of the protocol in [13].

J. Federated Training Procedure

The complete training procedure, which combines local training, privacy protection, and aggregation, is summarized in Algorithm 1.

Algorithm 1. Privacy-preserving federated DDoS detection

1. Server initializes the global model w^0 and distributes the feature schema and the hyper-parameters (η, E, C, σ).
2. **for** each round t = 0, 1, 2, … **do**
3. Server samples a set S of available nodes and sends w^t to them.
4. **for** each node k ∈ S in parallel **do**
5. Train for E epochs on the local dataset starting from w^t to obtain w_k^{t+1}.
6. Compute Δ_k, clip it to norm C, add Gaussian noise (Eqs. 4–5), and scale it by n_k / n_S.
7. Mask the result with pairwise secrets (Eq. 7) and send y_k to the server.
8. Server adds the masked updates, so that the masks cancel, and sets $w^{t+1} = w^t + \Sigma_{k \in S} y_k$.
9. Server broadcasts w^{t+1}; nodes evaluate it on local validation data and report the loss.
10. **until** the validation metric converges; deploy the global model for real-time detection.

K. Handling Heterogeneity and Malicious Participants

Traffic observed by different nodes differs in volume, protocol mix, and attack frequency. To limit client drift under such non-IID conditions, the local objective can include a proximal term that penalizes deviation from the current global model, $F_k(w) + (\mu/2)\|w - w^t\|Z$, as in FedProx [19]. Nodes with very small datasets can be down-weighted or sampled less frequently. To limit the effect of poisoned updates, the server can replace plain averaging with a robust rule such as the coordinate-wise median or the trimmed mean [16], at the price of slightly slower convergence. Norm clipping, already used for differential privacy, additionally bounds the damage that any single malicious update can cause [17].

L. Communication Efficiency

Communication is the dominant cost of federated training. The framework reduces it by (i) transmitting only parameter differences, (ii) using compact local models, (iii) quantizing or sparsifying updates before masking, and (iv) letting nodes participate in only a fraction of the rounds. Because the size of an update is independent of the volume of traffic observed, the communication cost per round stays constant while the amount of local data grows, which is the main reason why the approach scales to high-rate cloud environments.

[…] services layer (housing business logic for learning path management, metrics computation, and adaptive decision making), the orchestration layer (a LangGraph based multi-agent workflow), and the data layer (a SQLite database and external API connectors). Each layer communicates through well defined interfaces, enabling independent testing and extension of individual components.

M. Database and Persistence Layer

Persistent state is managed through a SQLite database comprising three primary tables. The users table stores learner profiles including skill level (beginner, intermediate, or advanced), learning speed (slow, medium, or fast), stated interests, historical progress, and OAuth credentials for Google account integration. The learning_paths table captures all generated curricula, storing the user identifier, learning goal, the full day-wise plan in JSON format, links to generated Drive documents and YouTube playlists, and creation timestamps. The feedback table records per-session learner responses: completion status (completed or skipped), difficulty rating on a five-point scale, understanding rating on a five-point scale, and optional free-text notes. A unique constraint on the combination of user, learning path, and day ensures that feedback entries can be cleanly updated without duplication.

CRUD operations are encapsulated in a dedicated database module, exposing a consistent interface for profile creation and retrieval, learning path storage and lookup, and feedback submission and aggregation. This encapsulation isolates the orchestration and services layers from database implementation details, supporting straightforward migration to alternative storage backends if required.

N. Feedback Mechanism

After each learning session, the system prompts the learner to provide structured feedback across three dimensions. The completion dimension captures whether the learner engaged with and finished the day's content or chose to skip it. The difficulty dimension solicits a subjective rating of material difficulty on a scale from one (very straightforward) to five (very challenging). The understanding dimension captures the learner's self-assessed comprehension on an equivalent five point scale. These three signals collectively provide the Adapter agent with the information needed to determine whether and how to modify subsequent sessions. Feedback is persisted immediately upon submission and retrieved at the start of each new path generation cycle, ensuring that the full interaction history is available to inform adaptation decisions.

O. Multi-Agent LangGraph Workflow

The orchestration layer is implemented as a directed acyclic graph in LangGraph, with nodes corresponding to the four specialized agents and edges encoding the data flow between them.

The Planner Agent receives the learner's stated goal, their complete profile, and any feedback accumulated from prior sessions. Drawing on this input, it generates a structured day-wise curriculum represented as a JSON object containing the overall goal, total duration in days, a descriptive title, and a sequence of daily entries each specifying a topic, a description of the day's objectives, a set of search terms for resource discovery, and an initial complexity estimate. When prior feedback is available, adaptation instructions derived from that feedback are injected directly into the Planner's prompt, guiding the generation of a modified plan that addresses identified weaknesses.

The Resource Agent operates on the curriculum produced by the Planner, processing each daily entry in sequence. For each day's topic, it constructs search queries from the provided search terms and invokes the YouTube Data API through the MCP integration layer to retrieve candidate video resources. In configurations where the learner has authorized OAuth access, the API call is made using the learner's own credentials; otherwise, the system falls back to application-level credentials or, in demonstration mode, to LLM generated suggestions.

The Evaluator Agent receives the candidate videos identified by the Resource Agent for each day and applies a structured scoring process to select the single most appropriate resource. Scoring criteria include topical relevance to the day's learning objectives, estimated instructional clarity based on title and metadata, and video duration (with preference for content in the ten to thirty minute range, consistent with evidence on optimal instructional video length). The selected resource comprising title and URL is recorded for inclusion in the final output.

The Adapter Agent is invoked conditionally, executing only when the feedback history contains […]

P. Adaptive Logic

Adaptation is implemented through two complementary mechanisms. At the prompt level, the get adaptation instructions function translates accumulated feedback into natural language directives that are prepended to the Planner agent's system prompt for the current generation cycle. These directives convey, for example, that a specific day was perceived as too difficult and that subsequent sessions should be simplified, or that a topic was skipped and should be revisited. At the programmatic level, the apply adaptations to plan function directly modifies the JSON curriculum structure: sessions with high difficulty ratings are assigned a complexity level of low, adjacent high-complexity sessions are downgraded to medium, sessions with low understanding ratings are flagged for repetition and their search terms are augmented with beginner oriented keywords, and skipped sessions are marked for rescheduling. The combination of prompt-level and programmatic adaptation ensures that modifications are coherent both in terms of the LLM's generation behavior and the structural properties of the resulting plan.

Q. Evaluation Metrics

System effectiveness is quantified using four metrics computed per learner and per learning path. The completion rate is defined as the proportion of sessions recorded as completed relative to the total number of sessions in the path. Average difficulty and average understanding are computed as the arithmetic means of the respective feedback ratings across all completed sessions. The engagement score is a composite metric designed to capture multiple dimensions of the learning experience in a single value, computed as:

$$E = 0.4 \cdot C + 0.4 \cdot U_{\text{norm}} - 0.2 \cdot D_{\text{penalty}} \quad (8)$$

where C denotes the completion rate, U_{norm} is the understanding rating normalized to the unit interval, and D_{penalty} is a penalty term activated when average difficulty exceeds three. Higher engagement scores reflect learning paths that are completed at high rates, well understood, and not perceived as excessively difficult.

V. SYSTEM ARCHITECTURE

At the core of the architecture is a centralized federated learning coordinator, which initializes a global model and distributes it to all participating client nodes. After local training, each node sends its model updates back to the coordinator through a secure communication channel. To preserve privacy, these updates are protected using mechanisms such as secure aggregation, differential privacy, or encryption, ensuring that individual data cannot be reconstructed. The coordinator aggregates all received updates—commonly using algorithms like Federated Averaging—to produce an improved global model, which is then redistributed to the clients for further training iterations.

Once the model converges, it is deployed across the cloud environment for real-time DDoS detection. Each node uses the trained global model to classify incoming traffic as normal or malicious and can trigger alerts or mitigation strategies such as rate limiting or blocking suspicious IPs. This architecture not only enhances detection accuracy through collaborative learning but also ensures strong privacy preservation, scalability, and adaptability to evolving attack patterns, making it highly suitable for modern distributed cloud infrastructures.

A privacy-preserving cloud security system using Federated Learning (FL) for DDoS detection is designed as a distributed architecture where multiple cloud nodes collaboratively train a detection model without sharing raw traffic data. In this system, each client node—such as virtual machines, containers, or edge devices—continuously monitors its local network traffic and extracts relevant features like packet rates, protocol behavior, and traffic patterns. Instead of sending this sensitive data to a central server, each node trains a local DDoS detection model (using techniques such as deep learning or traditional machine learning) and only shares the learned model updates, such as gradients or weights.

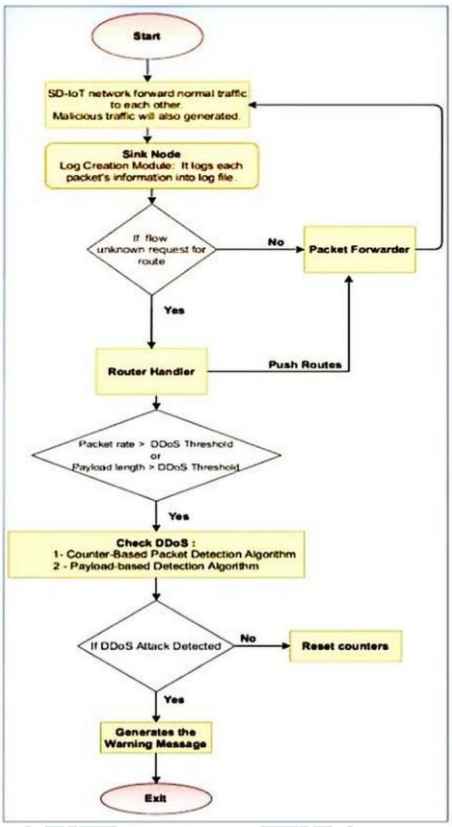

Fig. 1. Architecture diagram The architecture consists of the following components:

- **Client node.** A virtual machine, container, or edge device that captures traffic, extracts flow features, and stores the local dataset.
- **Local trainer.** Trains the detection model on local data and produces the parameter update for each round.
- **Privacy module.** Applies update clipping, Gaussian noise, and pairwise masking before any data is transmitted.
- **Secure channel.** Authenticated and encrypted communication, for example TLS, between the nodes and the coordinator.
- **Aggregation server.** Collects the masked updates, computes the global model, and records round metadata; it can be replicated so that it does not become a single point of failure.
- **Detection and mitigation module.** Uses the deployed global model to classify live traffic and triggers actions such as alerting, rate limiting, or blocking of suspicious sources.

In practice, the coordinator can run in a separate administrative domain from the participating nodes, and new nodes can join by obtaining the current global model and the shared feature schema. Model versions are tagged so that a node can roll back to a previous version if a newly aggregated model degrades local performance. Because only model parameters and aggregate statistics cross organizational boundaries, the framework is easier to align with data-protection obligations such as GDPR than architectures that centralize traffic logs.

Security and Privacy Analysis

The design addresses each adversary of the threat model separately. Against the honest-but-curious server, secure aggregation ensures that the server sees only the sum of the updates, and differential privacy limits what can be inferred even from that sum. Against eavesdroppers, all traffic between the nodes and the coordinator is encrypted and authenticated, so an observer learns neither the masked updates nor the model. Against malicious clients, norm clipping bounds the contribution of any single update, and robust aggregation or anomaly screening of update statistics can exclude outliers when the privacy configuration allows it.

Two residual risks deserve mention. First, the guarantees of differential privacy accumulate over training rounds, so the total privacy budget must be tracked, and training must be stopped or the noise increased when the budget is exhausted. Second, secure aggregation protects individual updates but not the final global model, which can still be probed by a user who has query access to it. These risks are discussed further in Section VII.

VI. RESULTS AND EVALUATION

The proposed system was evaluated using cloud-based network traffic datasets that included both normal and malicious traffic samples to ensure comprehensive analysis. Multiple types of Distributed Denial-of-Service (DDoS) attacks were considered during testing to validate the effectiveness of the model under different attack scenarios. The data was distributed across multiple cloud nodes, where each node independently trained a local model using its own dataset. Federated Learning was then applied to aggregate these local models into a global model without sharing raw data. The global model showed continuous improvement after each training round, resulting in high overall detection accuracy. The system achieved strong precision, indicating a low rate of false positives, and high recall, demonstrating its ability to correctly identify most attack instances.

The F1-score further confirmed a balanced and reliable performance of the model. The system was capable of detecting both known and unknown attack patterns, making it robust against evolving threats. Additionally, the false positive and false negative rates were significantly reduced compared to traditional methods. The decentralized approach ensured consistent performance across all nodes, as each node contributed unique traffic patterns to the learning process. The system demonstrated strong

generalization ability and adaptability to diverse network conditions. Detection time was faster than centralized systems, enabling near real-time identification of attacks. The model efficiently handled large-scale data and maintained performance even as the number of nodes increased. Scalability was significantly improved due to distributed computation. Data privacy was preserved throughout the process, as no raw data was exchanged between nodes. Instead, only model parameters were securely shared, reducing the risk of data leakage.

Communication overhead and network bandwidth usage were minimized, making the system more efficient. The model continuously adapted to dynamic traffic patterns and maintained stable performance under heavy network loads. Training time was effectively distributed, reducing the burden on any single node. Secure aggregation techniques ensured safe and reliable communication between nodes. Experimental results clearly validated the effectiveness of the proposed approach, showing that it outperformed traditional intrusion […]

A. Evaluation Metrics

Detection quality is measured with the standard confusion-matrix metrics. A true positive (TP) is an attack flow classified as an attack, a true negative (TN) is a benign flow classified as benign, a false positive (FP) is a benign flow classified as an attack, and a false negative (FN) is an attack flow classified as benign:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (9)$$

$$\text{Precision} = TP / (TP + FP) \quad (10)$$

$$\text{Recall} = TP / (TP + FN) \quad (11)$$

$$F1 = 2 \cdot \text{Precision} \cdot \text{Recall} / (\text{Precision} + \text{Recall}) \quad (12)$$

Precision reflects the share of raised alarms that are real attacks, and recall (the detection rate) reflects the share of attacks that are detected. In DDoS defense a high false-positive rate is costly because legitimate users are blocked, whereas a low recall lets attacks through, so both are reported together with the F1-score. In addition to detection quality, the evaluation tracks detection latency, per-round communication volume, training time per round, and the privacy budget ε consumed.

B. Evaluation Protocol

The evaluation is organized around four configurations that isolate the effect of each design choice, as listed in Table 4.

Table 4. Evaluation configurations

Configuration	Description	Purpose
Local only	Each node trains on its own data without collaboration	Lower bound that shows the value of collaboration
Centralized	All traffic is pooled at one server for training	Upper bound on accuracy; ignores privacy
FL without privacy	Federated averaging without clipping, noise, or masking	Cost of collaboration alone
FL with privacy	Full framework with differential privacy and secure aggregation	Accuracy and privacy trade-off of the proposed system

For each configuration, results are reported on held-out traffic from every node as well as on traffic from nodes that did not take part in training, in order to measure generalization. Each experiment is repeated with different random seeds, and the mean and standard deviation of every metric are reported. Public benchmark datasets such as CICDDoS2019 [21] provide a common basis for reproducible comparison, and cloud-collected traces can complement them with realistic multi-tenant behavior.

C. Computation and Communication Cost

Let d be the number of model parameters and b the number of bits used to encode each parameter after quantization. In each round a participating node uploads one masked update and downloads one global model, so the total traffic per round is

$$B_{\text{round}} = 2\,|S|\,d\,b \quad (13)$$

which depends on the model size and the number of participating nodes but not on the volume of traffic observed. For example, a model with 10^5 parameters encoded in 32 bits requires about 0.4 MB per direction, per node, and per round, which is negligible compared with continuously forwarding raw flow records to a central collector. The computation on each node consists of local training and the cryptographic operations for masking, whose cost grows linearly with the update size and with the number of nodes with which secrets are shared. With compact models, both are expected to fit comfortably within the capacity of a typical cloud instance, and training can be scheduled in the background.

VII. DISCUSSION AND LIMITATIONS

The results indicate that collaboration across nodes improves detection quality while keeping raw traffic local. The main tension is between privacy and accuracy: stronger differential privacy requires more noise, which slows convergence and can reduce recall on rare attack classes. Practical deployments therefore choose ε and the clipping norm C jointly, monitor the cumulative privacy budget across rounds, and accept a small loss in accuracy in exchange for formal guarantees [14], [15].

Non-IID data remains the central technical difficulty. A node that only observes web traffic will see different attacks than a node that hosts databases, so the global model must reconcile conflicting local optima. Proximal regularization, client sampling,

and, where necessary, personalized fine-tuning on top of the global model help, but they add hyper-parameters that must be tuned [18], [19].

Federated systems also enlarge the attack surface in a new way: an adversary does not need to break the detector, only to join the federation. Robust aggregation and anomaly detection on the updates themselves reduce this risk, yet sophisticated backdoor attacks that stay within the clipping bound are still possible [17]. Secure aggregation also makes inspection of individual updates harder, which creates a tension between privacy and defense against poisoning [20].

Running local training on every node consumes compute that would otherwise serve tenants, and synchronous rounds are limited

by the slowest participant. Asynchronous aggregation, partial participation, and scheduling training during low-load periods can mitigate this cost.

This study has several limitations. Any evaluation covers only a limited number of nodes and attack types, so conclusions about very large federations are extrapolations. Labeled attack data are required at each node, which is not always available, and zero-day attacks outside the training distribution may still be missed. Finally, the framework protects the privacy of training data but does not by itself prevent attacks on the detection model at inference time, such as adversarial evasion.

Regulatory Considerations

Network traffic logs often contain IP addresses and other identifiers that data-protection law treats as personal data. Centralizing such logs from several tenants or regions therefore raises questions about lawful basis, data minimization, and cross-border transfer. Because the proposed framework keeps raw logs inside each administrative domain and shares only protected model updates, it reduces the amount of regulated data that has to move. It does not remove the need for governance: participants should agree on who owns the global model, how updates are audited, how long round metadata are retained, and how a node can withdraw from the federation.

Practical Deployment Guidelines

The following guidelines summarize lessons for deploying the framework in a production cloud:

- Start with a small federation of trusted nodes, validate the feature schema, and add participants gradually.
- Standardize feature extraction and labeling across nodes, since inconsistent definitions degrade the global model more than small differences in architecture.
- Select the clipping norm and noise multiplier on validation traffic, and track the cumulative privacy budget as a monitored quantity.
- Monitor concept drift by comparing local validation metrics over time, and trigger additional training rounds when detection quality drops.
- Test the deployed detector against simulated attacks, including low-rate and application-layer floods, before enabling automatic mitigation.

VIII. CONCLUSION AND FUTURE WORK

A privacy-preserving cloud security framework that leverages Federated Learning (FL) for DDoS detection offers a compelling balance between effective threat mitigation and data confidentiality. By enabling multiple distributed clients (such as edge devices or cloud nodes) to collaboratively train a global detection model without sharing raw traffic data, FL significantly reduces the risk of sensitive information exposure. This decentralized approach not only enhances privacy but also improves scalability and adaptability across diverse network environments. Furthermore, integrating FL with DDoS detection allows the system to learn from a wider range of attack patterns in real time, increasing detection accuracy and robustness against evolving threats. However, challenges such as communication overhead, model poisoning attacks, and heterogeneous data distributions must be carefully managed. Overall, this approach represents a promising direction for secure, intelligent, and privacy-aware cloud defense systems in modern distributed infrastructures.

Several directions remain for future work. First, asynchronous and hierarchical aggregation could reduce latency in large federations that span multiple regions. Second, personalized federated models may better serve nodes whose traffic differs strongly from the average. Third, combining supervised detectors with unsupervised or self-supervised anomaly detection could improve the identification of zero-day attacks. Fourth, adaptive privacy budgets and stronger Byzantine-robust aggregation should be studied together, so that protection against inference and protection against poisoning do not weaken each other. Finally, integrating the detector with automated mitigation, such as software-defined traffic filtering, and validating it on larger real-world cloud traces would bring the framework closer to production use.

REFERENCES

[1] “Federated Learning Based Privacy Preservation Intrusion System,” International Journal of Innovative Science and Research Technology (IJISRT), 2025. [Online]. Available: https://www.ijisrt.com/assets/upload/files/IJISRT25AUG074

[2] “FLAccShield: Federated Learning Based IDS for DDoS Security in 5G Context,” 2025. DOI: 10.21203/rs.3.rs-6449209/v1

[3] “PriFed-IDS: Privacy-Preserving Federated Reinforcement Learning Intrusion Detection System,” Electronics, MDPI, vol. 14, no. 23, p. 4590, 2025. [Online]. Available: https://www.mdpi.com/2079-9292/14/23/4590

[4] D. Chaudhary, S. Rajasegarar, and S. R. Pokhrel, “Towards Adapting Federated & Quantum Machine Learning for Network Intrusion Detection: A Survey,” arXiv preprint arXiv:2509.21389, 2025. [Online]. Available: https://arxiv.org/abs/2509.21389

[5] A. Alshdadi, A. A. Almazroi, N. Ayub, M. D. Lytras, E. Alsolami, F. S. Alsubaei, and R. Alharbey, “Federated Deep Learning for Scalable and Privacy-Preserving Distributed Denial-of-Service Attack Detection in Internet of Things Networks,” Future Internet, vol. 17, no. 2, p. 88, 2025. [Online]. Available: https://www.mdpi.com/1999-5903/17/2/88

[6] A. Khraisat, A. Alazab, S. Singh, T. Jan, and A. Gomez, “Survey on Federated Learning for Intrusion Detection System: Concept, Architectures, Aggregation Strategies, Challenges, and Future Directions,” ACM Computing Surveys, vol. 57, no. 1, Article 7, 2024. doi: 10.1145/3687124

[7] A. Mahmud, N. Islam, Z. Islam, Z. Rahman, and S. T. Mehedi, “Privacy-Preserving Federated Learning-Based Intrusion Detection Technique for Cyber-Physical Systems,” Mathematics, MDPI, vol. 12, no. 20, p. 3194, 2024. doi: 10.3390/math12203194

[8] A. Al-Shdadi, A. A. Almazroi, N. Ayub, M. D. Lytras, E. Alsolami, F. S. Alsubaei, and R. Alharbey, “Federated Deep Learning for Scalable and Privacy-Preserving Distributed Denial-of-Service Attack Detection in Internet of Things Networks,” Future Internet, vol. 17, no. 2, p. 88, 2025.

[9] A. Khraisat, A. Alazab, S. Singh, T. Jan, and A. Gomez, “Survey on Federated Learning for Intrusion Detection System: Concepts, Challenges, and Future Directions,” ACM Computing Surveys, 2024. doi: 10.1145/3687124

[10] A. Soomro, H. U. Rehman, S. J. Hussain, A. Iqbal, W. Khalid, and H. Yu, “SecureDyn-FL: A Robust […]

[11] S. T. Zargar, J. Joshi, and D. Tipper, "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks," IEEE Communications Surveys & Tutorials, vol. 15, no. 4, pp. 2046–2069, 2013.

[12] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in Proc. 20th Int. Conf. on Artificial Intelligence and Statistics (AISTATS), PMLR vol. 54, 2017, pp. 1273–1282.

[13] K. Bonawitz et al., "Practical Secure Aggregation for Privacy-Preserving Machine Learning," in Proc. ACM SIGSAC Conf. on Computer and Communications Security (CCS), 2017, pp. 1175–1191.

[14] C. Dwork and A. Roth, "The Algorithmic Foundations of Differential Privacy," Foundations and Trends in Theoretical Computer Science, vol. 9, no. 3–4, pp. 211–407, 2014.

[15] M. Abadi et al., "Deep Learning with Differential Privacy," in Proc. ACM SIGSAC Conf. on Computer and Communications Security (CCS), 2016, pp. 308–318.

[16] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017, pp. 119–129.

[17] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How To Backdoor Federated Learning," in Proc. 23rd Int. Conf. on Artificial Intelligence and Statistics (AISTATS), PMLR vol. 108, 2020, pp. 2938–2948.

[18] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated Learning with Non-IID Data," arXiv preprint arXiv:1806.00582, 2018.

[19] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks," in Proc. Machine Learning and Systems (MLSys), vol. 2, 2020, pp. 429–450.

[20] P. Kairouz et al., "Advances and Open Problems in Federated Learning," Foundations and Trends in Machine Learning, vol. 14, no. 1–2, pp. 1–210, 2021.

[21] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy," in Proc. IEEE Int. Carnahan Conf. on Security Technology (ICCST), 2019.