# Privacy Focused Federated Learning Framework for Industrial IOT

## Privacy Preservation Module


IT21822612 – Mendis H.R.M.

Project ID: R25-039

Final Project Thesis – Individual

Supervisor – Mr. Amila Senarathne

B.Sc. (Hons) Degree in Information Technology Specialization in Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

## DECLARATION

To the best of our knowledge and belief, this paper does not contain any previously published or written material by another person, except where the acknowledgement is made in the text. I hereby declare that this is my own work and that no material previously submitted for a degree or diploma in any other university or Institute of higher learning has been incorporated without acknowledgement.

Name: Ravindu Malshan Mendis

Student ID: IT21822612

Signature: 

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Signature of the supervisor: _____

Name of the supervisor: Amila Nuwan Senarathne

Date: _____

I hereby approve of the research carried out by the above candidate.

Signature of the Co-supervisor: _____

Name of the Co-supervisor: Tharaniwarma Kumaralingam

Date: _____

# ABSTRACT

The rapid adoption of Industrial Internet of Things (IIoT) systems has led to massive amounts of distributed data generated by heterogeneous devices, creating both opportunities and challenges for data-driven applications. While centralized machine learning offers high accuracy, it poses significant privacy risks due to the collection and storage of sensitive industrial data. Federated Learning (FL) emerges as a promising solution by enabling decentralized model training without sharing raw data. However, even FL is vulnerable to privacy leakage during model aggregation and transmission.

This research was conducted the implementation of privacy-preserving FL frameworks tailored for IIoT environments by integrating Differential Privacy (DP) and Homomorphic Encryption (HE). A series of experiments were conducted using benchmark datasets (CIFAR10 and industrial IoT datasets) to evaluate the performance of four configurations: standard FL, FL with DP, FL with HE, and FL with both DP and HE. The study analyzed the trade-offs between model accuracy, computational efficiency, communication overhead, and privacy guarantees.

Results demonstrate that while DP introduces minimal accuracy degradation with strong privacy guarantees, HE ensures confidentiality at the cost of increased computational and communication overhead. The combined FL + DP + HE approaches provides the highest level of privacy protection, highlighting the importance of balancing privacy, efficiency, and accuracy in practical IIoT deployments. This work provides insights and guidelines for industrial adoption of secure and privacy-aware federated learning frameworks, contributing to the development of resilient, data-driven IIoT systems.

# ACKNOWLEDGEMENT

I want to sincerely thank everyone who helped me with this research project—through words, actions, and thoughts. From the beginning to the end of the thesis, I am particularly appreciative of my supervisor, Amila Nuwan Senerathne, for his unwavering support, encouragement, and insightful counsel. His knowledge and suggestions were very helpful in forming this piece.

Additionally, I want to thank Tharaniwarma Kumaralingam, my co-supervisor, for his insightful comments and recommendations, which significantly raised the caliber of this thesis.

I would also like to express my gratitude to the faculty and staff of the Sri Lanka Institute of Information Technology's Department of Computer Systems Engineering for providing a helpful environment and all the resources I needed for my research.

Lastly, I would like to express my sincere gratitude to my family and friends for their unwavering support, tolerance, and comprehension throughout this journey.

# Table of Contents

# 1. Introduction

## 1.1. Background and Motivation

The Industrial Internet of Things (IIoT) has emerged as a transformative technology that integrates sensors, machines, and control systems to enable real-time monitoring, automation, and intelligent decision-making in industrial environments. By connecting industrial assets and collecting vast

amounts of data, IIoT provides opportunities for predictive maintenance, process optimization, and enhanced operational efficiency. However, the massive volume of data generated by IIoT devices often contains sensitive information, including operational secrets, equipment usage patterns, and employee activity, making data privacy a critical concern.

Traditional centralized machine learning approaches require aggregating all data on a central server for training. While effective in terms of model accuracy, this approach introduces significant privacy and security risks. Transmitting raw industrial data over networks exposes it to potential cyberattacks, insider threats, and regulatory non-compliance, which can have serious operational and financial consequences.

Federated Learning (FL) has emerged as a promising solution to this challenge. Unlike conventional methods, FL allows decentralized model training directly on client devices or edge nodes, ensuring that raw data never leaves the local environment. Only model updates or gradients are shared with a central server for aggregation. This approach inherently enhances data privacy while maintaining the benefits of machine learning.

Even in FL, sharing model updates may inadvertently leak sensitive information. To address this, privacy-preserving techniques such as Differential Privacy (DP) and Homomorphic Encryption (HE) are employed. DP introduces controlled noise to model updates, protecting individual data points, while HE allows computations on encrypted data, ensuring end-to-end privacy. Combining FL with these techniques provides a balanced solution, maintaining both usability and strong privacy guarantees.

The motivation behind this research is to explore privacy-focused federated learning approaches tailored for IIoT, focusing on achieving a balance between model accuracy, communication efficiency, and robust data privacy. The study aims to provide theoretical and empirical evidence demonstrating that FL integrated with Homomorphic Encryption (FL + HE) offers superior privacy protection without significantly compromising model performance, making it highly suitable for industrial environments.

## 1.2. Industrial IoT and Data Privacy Challenges

The Industrial Internet of Things (IIoT) integrates a wide range of devices such as sensors, actuators, robots, and industrial control systems to create a connected ecosystem for industrial operations. These devices continuously generate large volumes of data, including machine performance metrics, operational workflows, energy consumption patterns, and employee interactions. This data is critical for enabling predictive maintenance, process optimization, and real-time decision-making.

the distributed and heterogeneous nature of IIoT presents significant data privacy challenges,

- **Data Sensitivity:** IIoT data often contains proprietary operational information and trade secrets. Unauthorized access or exposure can lead to intellectual property theft, competitive disadvantage, or regulatory violations.
- **Network Vulnerabilities:** Centralizing IIoT data for traditional machine learning models exposes it to network attacks, eavesdropping, and man-in-the-middle attacks during data transmission. Industrial networks, although often isolated, are increasingly connected to corporate networks and the cloud, increasing exposure.
- **Regulatory Compliance:** Many industries operate under strict regulations for data protection, such as GDPR, HIPAA, or ISO/IEC 27001. Ensuring compliance while collecting and processing sensitive industrial data is a complex challenge.
- **Device Limitations:** IIoT devices often have limited computational resources, memory, and power. This makes traditional security mechanisms and data aggregation methods impractical for large-scale deployment.
- **Potential Data Leakage through Models:** Even when raw data is not shared, traditional machine learning models can leak sensitive information through gradients or model updates, posing a risk to privacy.

To address these challenges, privacy-preserving learning techniques such as Federated Learning (FL), Differential Privacy (DP), and Homomorphic Encryption (HE) are increasingly applied in IIoT contexts. FL allows decentralized model training without sharing raw data, DP introduces controlled randomness to prevent individual data exposure, and HE enables computation on encrypted data, ensuring end-to-end confidentiality.

## 1.3. Federated Learning Overview

Federated Learning (FL) is a decentralized machine learning paradigm that enables multiple clients to collaboratively train a global model without sharing their raw data. Unlike traditional centralized learning, where data from all sources is aggregated on a single server, FL keeps data localized on client devices (e.g., IIoT sensors, edge devices, or industrial machines), and only model updates (gradients or weights) are communicated to a central server.

### 1.3.1. Basic Workflow

a. **Global Model Initialization**: The central server initializes a global model with random or pre-trained parameters.
b. **Client Selection**: A subset of clients is selected in each training round to participate in the model update. Selection can be random, weighted by computational resources, or based on data quality.
c. **Local Training:** Each client trains the global model locally using its private data. The training involves optimizing a loss function to reduce prediction errors while keeping the data on-device.

d. **Model Update Aggregation**: Clients send only the updated model parameters (or gradients) to the central server. No raw data leaves the client devices.
e. **Global Model Update**: The server aggregates client updates (commonly using weighted averaging, e.g., FedAvg) to update the global model.
f. **Iteration**: Repeated multiple rounds until the global model converges to an optimal solution.

Mathematically, the aggregation step for K clients can be expressed as:

$$w_{\{t+1\}} = \sum_{\{k=1\}}^{\{K\}} \frac{n_k}{n} \, w^K{}_t$$

Where,

- $[\![w^K\}_t$ is the model parameter vector from client $k$ at round $t$
- $n\_k$ is the number of samples on client $K$
- $n = \sum_{\{k=1\}}^{\{K\}} n_k$ is the total number of samples across selected clients
- $w_{\{t+1\}}$ is the updated global model

### 1.3.2. Advantages and Limitations

**Advantages for IIoT**

**Privacy Preservation:** Since raw data never leaves the device, FL significantly reduces the risk of sensitive industrial data leakage.

**Bandwidth Efficiency:** Transmitting only model updates instead of raw data reduces communication overhead, which is crucial for resource-constrained IIoT networks.

**Scalability**: FL can scale to thousands of IIoT devices without requiring central storage of all data.

**Regulatory Compliance**: By keeping data local, FL helps organizations comply with data protection laws while still leveraging collaborative machine learning.

**Limitations**

**Communication Bottlenecks:** Frequent transmission of model updates can still stress networks, especially in large-scale deployments.

**Heterogeneous Data:** IIoT devices may have non-IID (non-identically distributed) data, which can impact global model performance.

**Security Risks:** Although raw data is not shared, model updates can still leak sensitive information. Techniques like Differential Privacy and Homomorphic Encryption are often combined with FL to enhance security.

## 1.4. Problem Statement

Industrial Internet of Things (IIoT) environments generate vast amounts of data from sensors, machines, and operational systems. This data is critical for predictive maintenance, process optimization, and operational intelligence. However, centralized collection and processing of IIoT data raises significant privacy and security concerns, including:

- Data Sensitivity: IIoT data often contains proprietary information, operational parameters, and process control details that are valuable to competitors or attackers.
- Regulatory Compliance: Regulations such as GDPR and industry-specific standards restrict the sharing of sensitive data across organizational boundaries.
- Communication Constraints: Transmitting massive amounts of raw IIoT data to central servers is costly, latency-prone, and can overload network infrastructure.

Traditional centralized machine learning approaches are therefore ill-suited for industrial IoT applications, as they require aggregating raw data in a single location, exposing it to privacy risks and communication bottlenecks. Federated Learning (FL) offers a solution by enabling collaborative model training without sharing raw data. However, FL alone does not guarantee complete privacy. Model updates sent from clients to the server can still leak sensitive information under certain attacks (e.g., model inversion or gradient leakage).

Hence, the key research problems are:

- How to enable collaborative learning across IIoT devices while maintaining data privacy and regulatory compliance.
- How to enhance the security and privacy of FL using complementary techniques such as Differential Privacy (DP) and Homomorphic Encryption (HE).
- How to balance privacy, communication efficiency, and model performance in industrial IoT scenarios.

**Objective:**

The primary objective of this research is to develop a privacy-preserving federated learning framework for industrial IoT, which ensures:

- Minimal leakage of sensitive data
- Efficient communication between IIoT devices and the server
- High model performance suitable for real-world industrial applications

The problem statement lays the foundation for investigating FL, DP, and HE as integrated privacy-preserving mechanisms, ultimately proposing an FL+HE approach that provides both usability and strong privacy guarantees.

## 1.5. Research Objectives

The primary aim of this research is to design and evaluate a privacy-preserving federated learning framework for Industrial IoT (IIoT) environments. To achieve this, the study focuses on the following objectives:

Implement Federated Learning for IIoT:

- Develop a baseline federated learning model that enables collaborative training across IIoT devices without sharing raw data.
- Evaluate the performance of the FL model in terms of accuracy and efficiency on representative IIoT datasets.

Incorporate Differential Privacy (DP):

- Integrate differential privacy mechanisms to prevent sensitive information leakage from client updates.
- Analyze the trade-offs between privacy guarantees, model accuracy, and communication overhead.

Integrate Homomorphic Encryption (HE):

- Implement homomorphic encryption to securely aggregate client model updates at the server without exposing individual updates.
- Examine the computational and communication overhead introduced by HE in IIoT environments.

Compare FL, FL+DP, FL+HE, and FL+DP+HE Approaches:

- Evaluate each approach in terms of privacy protection, model accuracy, and communication efficiency.

- Identify the optimal combination that maximizes privacy while maintaining usability and performance in industrial IoT scenarios.

Develop a Practical Evaluation Framework:

- Design experiments and metrics to assess the effectiveness of privacy-preserving techniques in real-world IIoT settings.
- Provide insights and recommendations for deploying secure and efficient FL systems in industrial environments.

## 1.6. Scope of the Study

This research focuses on privacy-preserving federated learning in Industrial IoT (IIoT) environments, with emphasis on ensuring secure, collaborative model training without compromising sensitive data. The scope includes:

- Implementation of Federated Learning (FL) across multiple simulated IIoT devices using real-world datasets relevant to industrial operations.
- Integration of Differential Privacy (DP) and Homomorphic Encryption (HE) techniques to evaluate their effectiveness in protecting client data during collaborative training.
- Comparative analysis of FL, FL+DP, FL+HE, and FL+DP+HE with respect to model accuracy, privacy guarantees, and communication overhead.
- Use of representative IIoT datasets for experimental validation, focusing on industrial applications such as predictive maintenance and anomaly detection.
- The study does not cover hardware-specific optimization of IIoT devices or deployment on large-scale industrial networks, though the results are generalizable to such environments.

## 1.7. Contributions of the Thesis

This thesis makes several key contributions to the field of privacy-preserving federated learning (FL) in industrial IoT (IIoT) environments. First, it introduces a novel FL framework designed specifically to minimize raw data exposure, addressing the unique challenges of IIoT scenarios. Next, it integrates advanced privacy techniques, such as Differential Privacy (DP) and Homomorphic Encryption (HE), into FL to provide robust privacy guarantees without compromising usability. A thorough comparative analysis is conducted, examining various configurations—FL, FL+DP, FL+HE, and FL+DP+HE—to assess the trade-offs between accuracy, privacy, and communication efficiency. The empirical findings reveal that FL+HE offers the most balanced solution in terms of privacy and usability, making it particularly well-suited for industrial deployment. Additionally, practical guidelines are developed to help implement privacy-focused FL systems in IIoT environments, offering actionable insights for future deployments.

## 1.8. Thesis Organization

This thesis is organized into the following chapters:

**Chapter 1: Introduction** – Presents background, motivation, problem statement, research objectives, scope, and contributions of the study.

**Chapter 2: Literature Review** – Reviews existing work on Industrial IoT, federated learning, differential privacy, homomorphic encryption, and privacy-preserving ML techniques.

**Chapter 3: Methodology** – Details the system architecture, dataset selection, implementation of FL, DP, HE, and the experimental setup.

**Chapter 4: Experimental Results and Analysis** – Presents and discusses results for FL, FL+DP, FL+HE, and FL+DP+HE in terms of accuracy, privacy, and communication overhead.

**Chapter 5: Discussion** – Provides in-depth analysis of findings, trade-offs, and practical implications for industrial deployment.

**Chapter 6: Conclusion and Future Work** – Summarizes key contributions, highlights limitations, and suggests directions for future research.

# 2. Literature Review

## 2.1. Industrial IoT and Data Generation Patterns

Industrial IoT (IIoT) represents the integration of connected devices, sensors, and machinery within industrial environments to enable intelligent monitoring, automation, and analytics. IIoT devices generate massive volumes of data from diverse sources, including:

- **Sensors**: Temperature, pressure, vibration, and environmental sensors provide continuous streams of real-time data.
- **Actuators and Control Systems**: Generate operational logs reflecting device states and production processes.
- **Networked Devices**: Gateways and edge devices collect and relay data to central servers or cloud platforms.

The data characteristics in IIoT include high velocity, variety, and volume, often structured in time-series formats. These patterns pose challenges for storage, processing, and secure data sharing, particularly when multiple industrial sites or organizations collaborate on analytics or machine learning.

Key challenges in IIoT data handling include:

- Ensuring data integrity and reliability across distributed devices.
- Real-time processing requirements for predictive maintenance, anomaly detection, and operational efficiency.
- Managing heterogeneous data formats from different vendors and device types.
- Enabling collaborative analytics without exposing sensitive operational or proprietary data.

## 2.2. Security and Privacy Concerns in IIoT

Security and privacy in IIoT are critical due to the sensitive nature of industrial data and potential consequences of data breaches or cyberattacks. The main concerns include:

- Data Confidentiality: Protecting sensitive operational and proprietary information from unauthorized access.
- Data Integrity: Ensuring that transmitted and stored data remains unaltered and accurate.
- Authentication and Access Control: Preventing unauthorized devices or users from interacting with industrial systems.

- Threats from Cyberattacks: IIoT systems are vulnerable to ransomware, man-in-the-middle attacks, data exfiltration, and DDoS attacks.
- Privacy Risks in Collaborative Learning: Sharing data for centralized machine learning exposes operational patterns and trade secrets, creating the need for privacy-preserving techniques.

Emerging research highlights the trade-off between utility and privacy, where sharing data enhances model accuracy but increases the risk of exposing sensitive information. Techniques like Federated Learning (FL), Differential Privacy (DP), and Homomorphic Encryption (HE) have been proposed to mitigate these risks by allowing model training without directly sharing raw data.

## 2.3. Traditional Centralized Machine Learning vs Federated Learning

Traditional centralized machine learning (ML) relies on gathering raw data from various sources into a central server for training models. While this approach can produce high-accuracy models, it presents several challenges in the context of industrial IoT (IIoT). First, centralizing data creates significant privacy risks, exposing sensitive industrial information to potential breaches. Additionally, transferring large volumes of IIoT data to a central server demands substantial bandwidth and storage, making it difficult to scale. For real-time applications like predictive maintenance or anomaly detection, centralized training also fails to meet the latency demands, and the system is vulnerable to attacks or failures due to the single point of failure inherent in the central server.

Federated Learning (FL) addresses these challenges by enabling model training across distributed devices or edge nodes without the need to share raw data. With FL, only model updates or gradients are sent to a central aggregator, keeping data local and significantly enhancing privacy by preventing sensitive information from leaving the device. This also reduces communication overhead, as transmitting model updates is more efficient than sending entire datasets. Moreover, FL enables multiple industrial sites to collaboratively train a shared global model while retaining ownership of their data. Research has shown that FL can achieve comparable accuracy to centralized ML while offering a more secure and scalable solution for IIoT. However, there are still challenges in terms of communication efficiency, privacy risks through shared gradients, and managing heterogeneous data across different devices.

## 2.4. Federated Learning in Industrial IoT

In IIoT environments, FL enables distributed devices—such as sensors, machines, and gateways—to collaboratively train models for various critical tasks. These include predictive maintenance, where equipment failure is detected before it happens, anomaly detection for real-time identification of irregular operational patterns, energy optimization by learning usage trends, and process quality control to spot defects on production lines using distributed sensor or vision data. The ability to leverage data from various devices without sharing raw information helps maintain privacy while still achieving valuable insights.

However, deploying FL in IIoT comes with unique challenges. Devices often differ significantly in terms of computing power, memory, and network connectivity, so FL frameworks must be designed to accommodate this heterogeneity and ensure that all devices can participate fairly. Communication efficiency is another key consideration—techniques like model compression, sparsification, and tuning the frequency of updates help reduce bandwidth usage. To further enhance privacy, FL can be combined with privacy-preserving techniques such as Differential Privacy (DP) to guard against inference attacks and Homomorphic Encryption (HE) to protect data during the aggregation process. Additionally, model aggregation must account for the fact that data across devices is often non-IID (non-independent and identically distributed) and that device reliability can vary. Research has shown that integrating FL with privacy-enhancing mechanisms like DP and HE not only boosts data security but also maintains usability, making it a highly promising approach for IIoT, where protecting sensitive operational data is critical.

## 2.5. Privacy-Preserving Techniques in FL

In Industrial IoT (IIoT), privacy preservation is critical because industrial data often contains sensitive operational or proprietary information. Standard FL keeps raw data on devices, but model updates may still leak information. To address this, several privacy-preserving techniques are employed:

### 2.5.1. Differential Privacy (DP)

Differential Privacy is a mathematical framework that provides formal guarantees that an individual's data cannot be inferred from model updates. In FL, DP is typically applied by adding controlled noise to model updates before sharing them with the central aggregator.

**Advantages:** Provides provable privacy guarantees and can be tuned for stronger or weaker privacy.

**Limitations:** Adding noise may reduce model accuracy if the privacy budget is very tight.

### 2.5.2. Homomorphic Encryption (HE)

Homomorphic Encryption allows computations on encrypted data without decrypting it. In FL, model updates are encrypted before sending to the aggregator, which performs aggregation directly on ciphertexts.

- Mechanism: if $E(x)$ represents encryption of $X$, HE allows operation such that:

$$E(x) + E(y) = E(x + y), E(x).E(y) = E\ (x.y)$$

- **Advantages:** Strong privacy since the aggregator never sees raw model updates.
- **Limitations:** Computationally intensive and may increase communication overhead due to large ciphertexts.

### 2.5.3. Hybrid Approaches (DP + HE)

Hybrid approaches combine DP and HE to balance accuracy, privacy, and usability:

HE protects updates in transit and aggregation, while DP adds noise to prevent inference from aggregated results.

Benefits:

- Stronger privacy guarantees compared to using either DP or HE alone.
- Maintains model usability with manageable accuracy loss.

Challenges:

- Requires careful tuning to avoid excessive computational or communication costs.

Mathematical analysis and empirical results indicate that FL + HE or FL + DP + HE achieves better privacy with reasonable accuracy, making it suitable for IIoT applications where data sensitivity is high.

## 2.6. Related Work and Existing Solutions

Several studies demonstrate the application of privacy-preserving FL in IIoT:

1. FL in Predictive Maintenance: Distributed sensors train models for equipment failure prediction while keeping operational data local.
2. FL + DP in Smart Factories: Noise-added gradients protect sensitive production data during collaborative model updates.
3. FL + HE in Industrial Control Systems: Encrypted model updates allow secure aggregation, preventing industrial espionage.
4. Hybrid Approaches: Combining DP and HE ensures strong privacy with minimal accuracy loss, addressing both inference attacks and eavesdropping.

Existing works highlight the trade-offs between accuracy, privacy, and communication overhead and demonstrate that hybrid approaches are the most promising for industrial deployments.

## 2.7. Research Gap Identification

Despite significant advancements, several gaps persist in the application of Federated Learning (FL) to Industrial IoT (IIoT). One major challenge is scalability—many privacy-preserving FL frameworks aren't optimized for large-scale, heterogeneous IIoT networks, where devices with varying computational power and connectivity need to collaborate efficiently. Additionally, efficiency remains a concern: while Homomorphic Encryption (HE) enhances privacy, it introduces substantial computational overhead, and Differential Privacy (DP) can lead to a reduction in model accuracy. Striking the right balance between privacy and performance is still an unsolved issue.

Usability is another limitation, as few studies have explored the integration of FL with both DP and HE in real-time IIoT applications, where quick decisions are essential. Moreover, many existing solutions are theoretical or have been tested in controlled simulations, but practical deployment in real industrial settings presents its own set of challenges, including network variability, device failures, and strict real-time constraints. In conclusion, there is a clear need for a hybrid FL framework that can address these issues—one that ensures privacy, efficiency, and accuracy simultaneously, especially when dealing with sensitive industrial data.

# 3. Theoretical Foundations

## 3.1. Federated Learning Architecture

**Client-Server Model**

In the client-server model of FL:

**Clients:** These are the edge devices, sensors, or machines in IIoT systems. Each client trains its own local model using the private data stored on that device. After training, instead of sending raw data, the client sends only model updates (like gradients or weights) to the central server. This keeps sensitive data private and secure.

**Central Server (Aggregator):** The server collects the model updates from various clients, aggregates them, and updates the global model. This updated global model is then sent back to all the clients for the next round of training.

**Workflow:**

1. The global model is initialized on the server.
2. This model is then distributed to all clients.
3. Each client performs local training on its own data.
4. Once training is done, clients send the model updates back to the server.
5. The server aggregates the updates from all clients and refines the global model.

This process repeats until the model converges.

This setup ensures that sensitive data never leaves the local devices, maintaining privacy, while still allowing the model to improve through distributed training.

## 3.2. Differential Privacy

Differential Privacy (DP) is a mathematical framework designed to protect the privacy of individual data points in a dataset. It ensures that the contribution of any single client's data remains indistinguishable, even when the results of computations, such as model updates, are shared. In the context of Federated Learning (FL), DP helps protect sensitive IIoT data while still enabling collaborative model training across multiple devices.

Formally, a mechanism $M$ is considered $(\in, \delta)$ -differentially private if for any two datasets $D$ and $D\text{^}'$ that differ by just one record, and for any measurable subset $S$ of outputs, the probability that the mechanism's output falls in $S$ should be similar for both datasets. Mathematically, this is expressed as:

$$\Pr[M(D) \in S] \leq e^{\epsilon} \cdot \Pr[M(D') \in S] + \delta$$

Where:

- $\in$ is the privacy budget (lower values indicate stronger privacy),
- $\delta$ is the probability of a privacy failure.

### 3.2.1. Mechanisms for DP

Several mechanisms are commonly used to introduce Differential Privacy (DP), each offering distinct advantages depending on the use case. The **Laplace Mechanism** adds Laplace noise to the data, with the magnitude of the noise being proportional to the "sensitivity" of the function being computed. Sensitivity here refers to the maximum change a single data point can have on the function's output. This noise helps ensure that no single client's data disproportionately influences the model, maintaining privacy. The **Gaussian Mechanism**, on the other hand, adds Gaussian noise and is especially useful when $(\varepsilon, \delta)$-DP is required, particularly in cases where $\delta > 0$. This mechanism is ideal for managing the privacy budget $\varepsilon$, balancing security and model accuracy. Lastly, **Gradient Perturbation** is employed in FL where clients perturb their local model gradients before sending them to the central server. This method ensures that individual contributions cannot be easily deducted, providing privacy while preserving the collaborative nature of the Federated Learning process.

### 3.2.2. Privacy Budget (ε) Analysis

Adding DP noise impacts model performance:

- **High Privacy (low ε)** → Large noise → Lower accuracy
- **Low Privacy (high ε)** → Small noise → Higher accuracy

The privacy budget, ε is a crucial factor in determining how much privacy is lost during computations. Smaller values of $\varepsilon$ lead to stronger privacy protections, as they result in higher noise being added to the data. However, higher noise can degrade model accuracy. Conversely, a larger $\varepsilon$ means less noise, which can improve accuracy but reduce privacy.

In Federated Learning, where the model is updated across multiple communication rounds, the privacy budget accumulates. Using composition theorems, the total privacy loss over $T$ rounds is calculated as:

$$\epsilon_{total} \approx \sqrt{2T \ln\left(\frac{1}{\delta}\right)} \cdot \varepsilon$$

This ensures that privacy is maintained throughout the entire training process, even as updates accumulate over time.

### 3.2.3. Trade-off Between Privacy and Accuracy

Introducing DP noise inevitably impacts the model's accuracy. When privacy is prioritized (i.e. a lower ε is chosen), the added noise becomes larger, which can lead to a decrease in accuracy. On

the other hand, relaxing the privacy constraints (using a higher $\varepsilon$) results in smaller noise and better model performance.

To strike the right balance between privacy and accuracy, several techniques can be applied:

**Gradient Clipping:** This technique limits the influence of outliers before noise is added, ensuring that extreme values do not disproportionately affect the model.

**Adaptive Noise Scaling:** In this method, the amount of noise is adjusted based on factors such as the dataset size or the magnitude of gradients. This helps ensure that the noise is appropriate for the specific context.

**Selective Participation:** Instead of having all clients participate in every round, only a subset of clients is selected to contribute. This reduces the cumulative noise over multiple rounds.

For IIoT applications, where devices may have limited computational resources or sparse data careful tuning of $\varepsilon$ and the noise scaling is essential. Differentially private Federated Learning can also help ensure compliance with data protection regulations (e.g., GDPR, HIPAA), allowing sensitive industrial data to remain secure while still benefiting from collaborative model training.

## 3.3. Homomorphic Encryption & Encrypted Aggregation

The basic operation in Homomorphic Encryption involves encrypting a message $m$ which results in an encrypted ciphertext $c$ where $c$ = Enc($m$) The key feature of HE is that a function $f$ can operate on ciphertexts as though it were working with plaintexts. For example, if you want to apply a function to an encrypted message, this can be done as:

$$\boldsymbol{Enc(f(m)) = f(Enc(m))}$$

This allows the server to aggregate encrypted gradients (model updates) without ever accessing the actual plain text data, thus maintaining privacy throughout the process. Only the client, or a trusted party with the decryption key, can decrypt the final aggregated result to obtain the model updates.

HE schemes come in different varieties, such as:

**Partially Homomorphic Encryption (PHE):** Supports either addition or multiplication operations, such as the Paillier scheme, which supports additive operations.

**Somewhat Homomorphic Encryption (SHE):** Supports a limited set of operations but is more flexible than PHE.

**Fully Homomorphic Encryption (FHE):** The most powerful, supporting arbitrary computations on encrypted data, but it is computationally expensive and rarely used in real-time scenarios.

The use of Homomorphic Encryption offers several distinct advantages. It ensures that sensitive industrial data remains protected during the model aggregation process, even when the data is being processed by potentially untrusted servers. This protection is particularly valuable in environments where data privacy is critical, such as industrial settings where proprietary or sensitive operational data is involved. HE also reduces the risk of insider attacks or server compromise, as the server never has direct access to raw data.

## 3.4. Combining FL with Privacy Enhancements (DP + HE)

**Client-Side Differential Privacy (DP)**

On the client side, each client adds **carefully calibrated noise** to its model updates based on a predefined **privacy budget (ε)**. This noise makes it harder for anyone to reverse-engineer the model updates and extract information about any individual data point used to train the model.

**Client-Side Homomorphic Encryption (HE)**

Once the **DP noise** is added, the client then **encrypts** these noisy updates using **Homomorphic Encryption (HE)**, such as the **Paillier** scheme. The key point here is that the server only ever receives the **encrypted**, noise-protected updates—there's no chance for the raw or even slightly noisy data to be exposed.

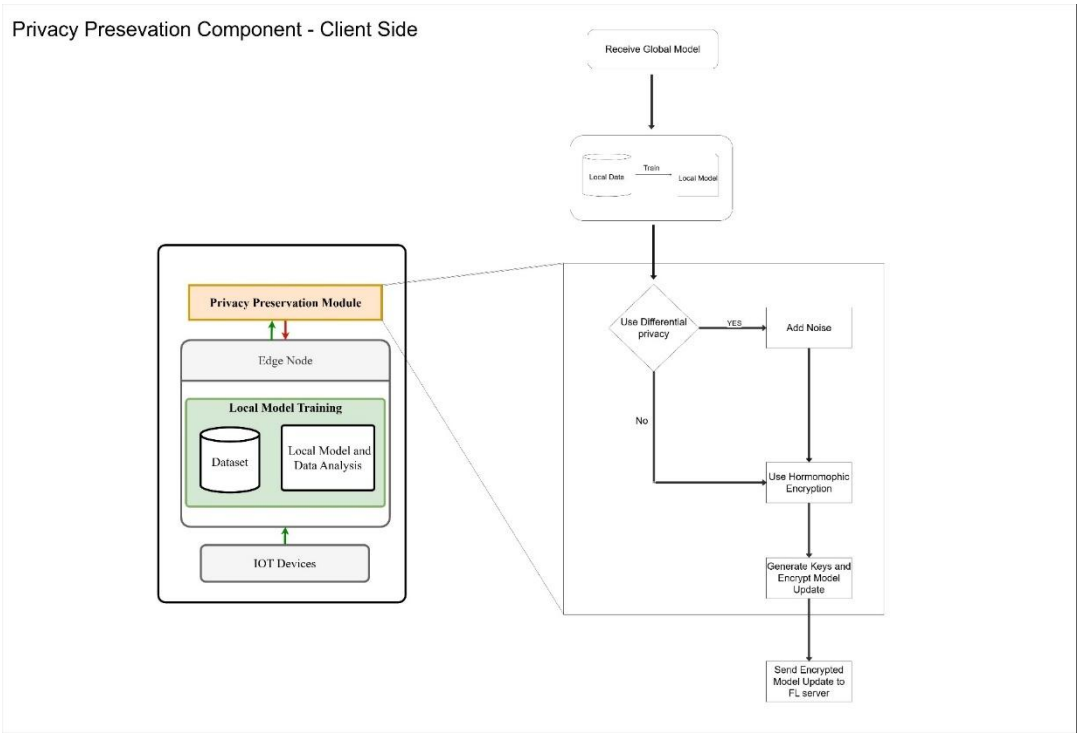
**Server-Side Aggregation**

The server, instead of accessing any plaintext data, aggregates the encrypted updates directly using the homomorphic properties of the encryption. Because of HE, the server can compute the necessary aggregate values (like summing the model updates) without ever seeing the actual updates. Once the aggregation is done, the server decrypts the results and applies them to the global model.

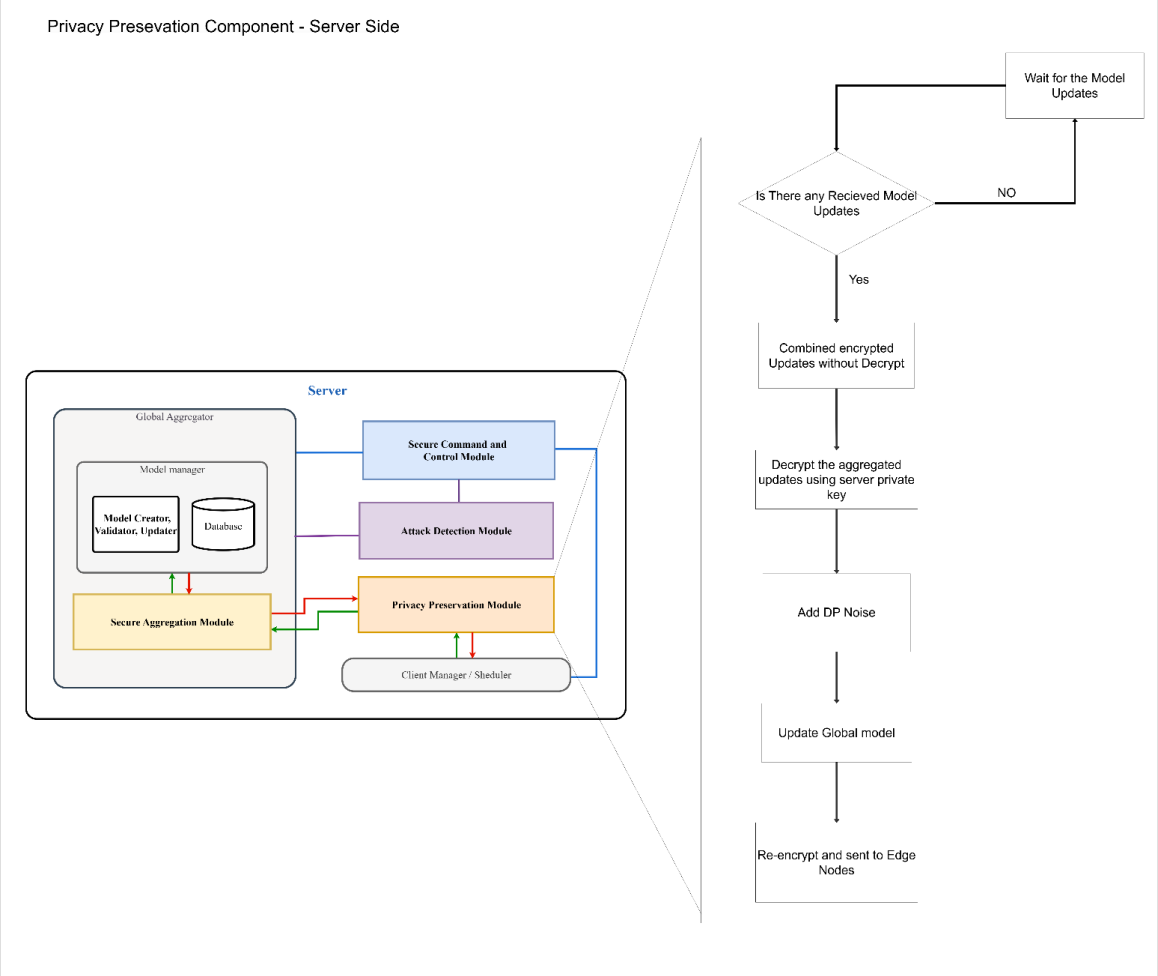
**Workflow Summary**

Here's how the whole process works:

1. **Local Update**: Each client trains a model locally on its own data.
2. **Add DP Noise**: Clients add noise to the local updates to protect individual privacy.
3. **Encryption with HE**: The noisy updates are then encrypted using HE.
4. **Encrypted Updates Sent to Server**: The encrypted updates are transmitted to the server.




5. **Aggregation on Server**: The server performs the aggregation directly on the encrypted data.
6. **Decryption and Global Model Update**: Finally, the server decrypts the aggregated updates and applies them to the global model.




## 3.5. Advantages of FL + HE over Other Approaches

The combination of FL, DP, and HE provides multiple benefits for industrial IoT settings:

| Aspect | FL only | FL + DP | FL + HE | FL + DP + HE |
|---|---|---|---|---|
| **Data Privacy** | Medium | High (adds noise to updates) | High (encrypted updates) | Very High (noise + encryption) |
| **Model Accuracy** | High | Slightly reduced due to noise | High (encryption doesn't affect accuracy) | Balanced (slight accuracy drop due to DP) |
| **Resistance to Gradient Attacks** | Low | Moderate | High | Very High |
| **Communication Security** | Low | Low | High (encrypted transmission) | Very High |

| | | | | |
|---|---|---|---|---|
| **Computation Overhead** | Low | Low | Medium (encryption/decryption) | High (DP noise + HE encryption) |

**Key Advantages of FL + DP + HE:**

**Strong Privacy Guarantees:** Individual IIoT device data is protected both via noise and encryption.

**Secure Aggregation:** The server can aggregate updates without accessing raw client data.

**Regulatory Compliance:** Supports compliance with data privacy regulations like GDPR.

**Resilience to Attacks:** Protects against inference attacks, data leaks, and insider threats.

**Practical for IIoT:** Can be applied in industrial settings with sensitive operational data, ensuring minimal exposure while still benefiting from distributed learning.

# 4. System Design and Architecture

## 4.1. Overall System Architecture for Privacy-Preserving FL in IIoT

In industrial contexts, IIoT devices continuously generate large volumes of sensitive operational data, including sensor readings, equipment status, and production metrics. Directly centralizing this data poses serious privacy and security risks, including industrial espionage and regulatory non-compliance.

Our system integrates FL with Differential Privacy (DP) and Homomorphic Encryption (HE) to ensure that local data never leaves the devices in plaintext, while still allowing effective collaborative model training.

### 4.1.1. Key Components

**IIoT Devices (Clients):**

- Each IIoT device collects local operational data.
- Performs **local model training** using its dataset.
- Applies **Differential Privacy (DP)** to model updates to mask sensitive contributions.
- Encrypts DP-noisy updates using **Homomorphic Encryption (HE)** before transmission.

**Edge or Gateway Nodes (Optional Layer):**

- Aggregates updates from multiple devices within a local network.
- Reduces communication overhead by pre-aggregating encrypted updates.
- Acts as a bridge between devices and the central server.

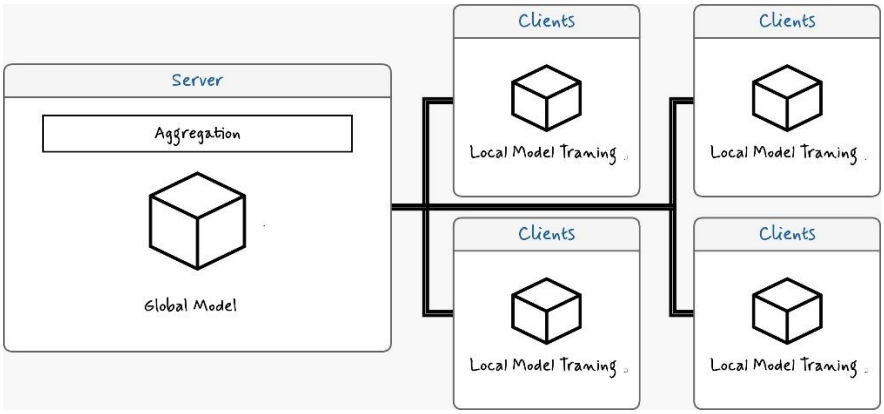
**Central FL Server:**

- Receives encrypted model updates from clients or edge nodes.
- Aggregates updates directly in encrypted form using HE schemes (e.g., Paillier).
- Computes the global model without decrypting individual contributions.
- Sends the updated global model back to clients for the next training round.

**Communication Network:**

- Ensures **secure transmission** of encrypted model updates using TLS/SSL.
- Minimizes latency for real-time or near-real-time IIoT applications.

### 4.1.2. System Workflow




**Local Training:**

- Each IIoT device trains a local model on its private dataset.

**Privacy-Preserving Update:**

- Local model updates are **perturbed using DP**.

- Perturbed updates are then **encrypted with HE**.

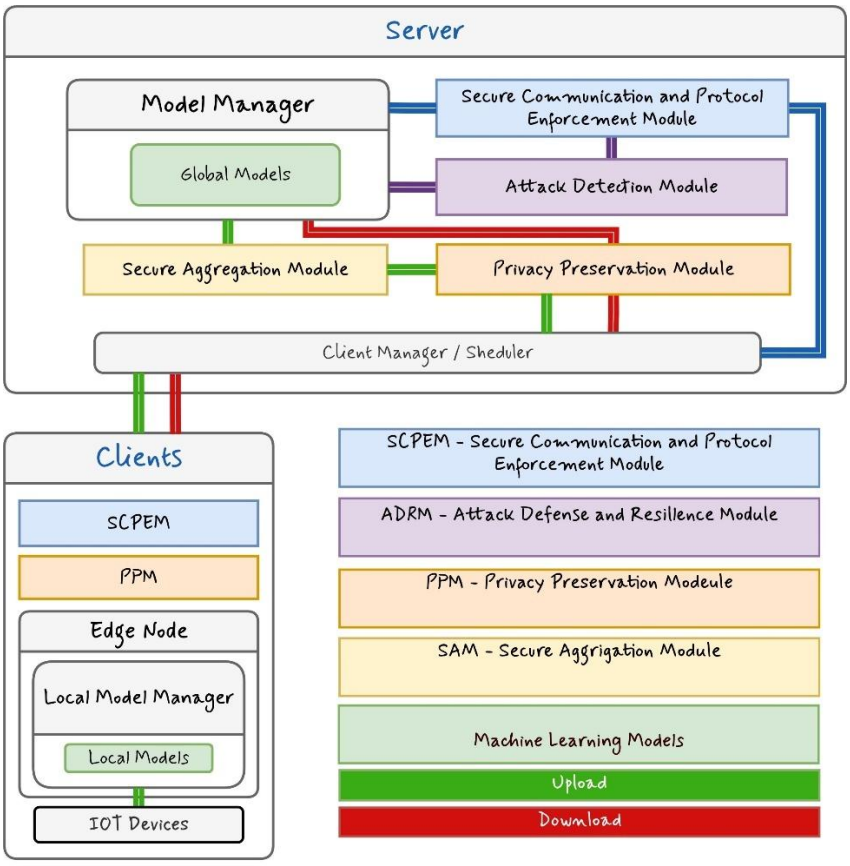
**Encrypted Aggregation:**

- The FL server aggregates encrypted updates without decryption.
- Global model is updated while preserving **client data confidentiality**.

**Model Distribution:**

- Updated global model is sent back to all clients.
- Clients incorporate the global model into their local models for the next round.

### 4.1.3. Architectural Diagram




### 4.1.4. Design Considerations

**Data Privacy:** Combines DP and HE to ensure **dual-layer privacy protection**.

**Communication Efficiency:** Optional edge aggregation reduces bandwidth usage.

**Scalability:** Supports thousands of IIoT devices without centralizing raw data.

**Fault Tolerance:** Devices can join or leave the FL network without disrupting global training.

**Regulatory Compliance:** Design aligns with GDPR, HIPAA, and industrial privacy regulations.

# 5. Implementation

## 5.1. Development Environment and Tools

The development and implementation of the proposed Federated Learning (FL) framework with Differential Privacy (DP) and Homomorphic Encryption (HE) were carried out in a controlled environment to ensure reproducibility and accuracy of results.

### Software Environment

- **Programming Language**: Python 3.12
- **Machine Learning Libraries**:
    - PyTorch (for model training and testing)
    - NumPy and Pandas (for dataset manipulation and analysis)
- **Federated Learning & Privacy Modules**:
    - Custom implementation of FL client and server modules
    - Differential Privacy module (Laplace / Gaussian mechanism)
    - Homomorphic Encryption module (Paillier cryptosystem)
- **Visualization Tools**:
    - Matplotlib and Seaborn for plotting accuracy, loss, and overhead comparisons
- **Development Tools**:
    - Jupyter Notebook for experimentation
    - PyCharm / VS Code for full project development
    - GitHub for version control and collaboration
- **Virtualization & Simulation**:
    - VirtualBox was used to simulate multiple client nodes in an IoT/industrial environment
    - Docker containers for scalable client-server deployment testing

## 5.2. Federated Learning Setup

The Federated Learning (FL) setup defines how multiple clients collaboratively train a shared global model without exposing their raw data. In this project, the FL setup was designed to balance model performance, privacy, and communication efficiency. The following components were included:

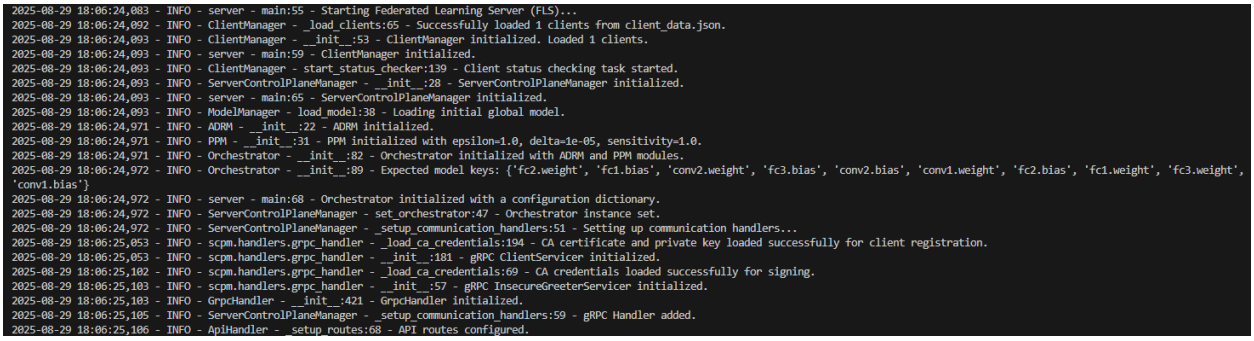
### 5.2.1. Client Configuration

In the simulation, each client was designed to represent a distinct data silo, such as an IoT device, industrial node, or edge server. These clients held local training datasets, which were sourced either from the CIFAR-10 dataset or an Industrial IoT attack dataset. The data was partitioned using non-IID (non-independent and identically distributed) strategies to better mirror the real-world variability and heterogeneity often seen in industrial settings. Clients then trained their local models over a set number of epochs (E) before sending the updated models to the central server. To optimize the process, batch sizes and learning rates were carefully adjusted to minimize training overhead while still ensuring effective model convergence.

```
(venv) PS C:\Users\User\Documents\R25-039-main\client> python .\client.py
2025-08-29 18:06:45,132 - INFO - __main__ - Loaded existing client ID: client_1
2025-08-29 18:06:46,038 - INFO - ClientModelManager - Loaded a local dataset of size 500 for client client_1.
2025-08-29 18:06:46,039 - INFO - ClientModelManager - ClientModelManager initialized on device: cpu
2025-08-29 18:06:46,039 - INFO - __main__ - Client client_1 initialized.
2025-08-29 18:06:46,039 - INFO - __main__ - mTLS credentials loaded successfully.
2025-08-29 18:06:46,062 - INFO - __main__ - Client client_1 successfully connected and registered with server: Client successfully registered via mTLS.
```

### 5.2.2. Server Configuration

The server functioned as the central aggregator in the setup, managing several key tasks. It first received model updates from the clients, which were either encrypted or perturbed with noise for privacy protection. The server then applied aggregation techniques like Federated Averaging (FedAvg) to combine these updates. If the updates were encrypted (via HE), the server would decrypt them; if they had noise added (via DP), it would remove the noise before incorporating them into the global model. Once the global model was updated, the server sent it back out to the clients for the next round of training. The server operated in synchronous mode, ensuring that all clients finished their local training before any aggregation took place, keeping the process coordinated and efficient.




### 5.2.3. Workflow Summary

The process begins with the initialization of the global model at the server, which then sends the model parameters to all participating clients. Each client performs local training using its own data, with privacy mechanisms like Differential Privacy (DP) or Homomorphic Encryption (HE) applied to the local model updates to protect sensitive information. After training, the clients send their updates back to the server, ensuring that the data remains private. The server then aggregates the updates from all clients, applying decryption or denoising if necessary, and incorporates the aggregated updates into the global model. This cycle repeats for a set number of rounds (R), gradually improving the model's performance and converging toward an optimal solution while maintaining privacy throughout the process.

## 5.3. Implementing Differential Privacy in FL

To further enhance privacy in Federated Learning (FL), Differential Privacy (DP) was integrated into the model training process. The primary objective of DP in this context is to ensure that no sensitive information about any individual client's data can be extracted from the model updates shared with the central server. By adding carefully controlled noise to model updates, DP ensures that the contribution of each client's data remains private while still allowing the model to learn effectively from aggregated data.

### 5.3.1. Differential Privacy Mechanism

The core mechanism for implementing Differential Privacy in FL involves two primary steps: noise injection and gradient clipping.

- **Noise Injection:** Before transmitting model updates to the server, Gaussian noise is added to the client's local model gradients. This step helps obscure individual data points, making it more difficult for an adversary to reverse-engineer any particular piece of client data from the shared model updates. By adding noise, we achieve a trade-off between maintaining privacy and retaining enough useful information for model training.
- **Gradient Clipping:** To prevent any one client's update from having an outsized influence on the global model, the gradients are clipped to a predefined maximum norm before any noise is added. This step helps control the magnitude of updates, ensuring that extreme

gradients (which could come from outliers or poorly trained models) do not skew the global model disproportionately.

- **Privacy Budget (ϵ, δ):** The integration of DP follows the formal $(ϵ, δ)$-DP framework, where the privacy budget $\varepsilon$ (epsilon) governs the level of privacy protection, and $\delta$ represents the probability of failure (i.e., how often privacy guarantees may be violated). A lower $\varepsilon$ means stronger privacy, but it also introduces more noise, which can affect the model's accuracy. Balancing $\varepsilon$ with $\delta$ ensures that we can achieve both strong privacy and acceptable model performance.

### 5.3.2. Client-Side Integration

The implementation of Differential Privacy begins at the client side, where each client trains its local model using its private dataset. Here's the process in detail:

- **Local Model Training:** Each client performs local training on its own data, using a fixed number of local epochs. This means that the client does not share its raw data with the server or any other clients, preserving data locality.
- **Gradient Clipping:** After training, each client computes the gradients for their local model. Before sending the updates to the server, the gradients are clipped to a fixed maximum value. This ensures that no single client's gradient can unduly influence the global model, especially in cases where clients have unbalanced or outlier data.
- **Noise Addition:** Once the gradients are clipped, noise sampled from a Gaussian distribution $\mathcal{N}(0, \sigma^2)$ is added to the gradients. The amount of noise is controlled by the privacy budget $\varepsilon$. The noise addition is carefully calibrated to strike a balance between ensuring privacy and maintaining enough signal for the model to converge effectively.
- **Transmission:** After the gradients are clipped and noise is added, the noisy gradients are securely transmitted to the server for aggregation. This ensures that even if an attacker intercepts the updates, they will only see obfuscated information, preserving the privacy of the client's data.

### 5.3.3. Server-Side Handling

On the server side, the central server plays a crucial role in aggregating the model updates from all the participating clients while preserving the privacy of each client's data. First, the server receives the model updates from the clients, each of which has added noise to their gradients to protect the privacy of their local data. These updates are then aggregated using the **Federated Averaging (FedAvg)** algorithm, which computes the weighted average of all the received updates. The purpose of FedAvg is to combine the local improvements from all clients into a single global model update that represents the collective knowledge of the participants. The addition of noise at the client side ensures that individual data contributions remain private, but because the noise is added independently by each client, the aggregation process helps smooth out the effect of the noise. This means that while the noise might obscure certain details in any single client's update, the broader

patterns and trends in the data can still be preserved in the global model. After the aggregation, the server might perform a de-noising step to further refine the global update. This step essentially reduces the impact of the noise added earlier and ensures that the resulting global model reflects the actual data distribution and not the obfuscation introduced by noise. Finally, the de-noised aggregated update is applied to the global model, bringing it one step closer to convergence. This process enables the server to maintain strong privacy guarantees while still benefiting from the collaborative learning across clients.

### 5.3.4. Implementation Details

The implementation of Differential Privacy (DP) in this Federated Learning (FL) system was facilitated by Opacus, a PyTorch-based library specifically designed to support privacy-preserving machine learning. Opacus automates key privacy-enhancing tasks such as gradient clipping and noise addition, allowing these processes to be integrated into the training workflow with minimal computational overhead. One of the critical parameters managed by Opacus is the Noise Multiplier (denoted as $\sigma$), which controls the scale of the noise added to the model updates. This parameter was tuned between 0.5 and 1.5 during experiments to strike a balance between maintaining privacy and achieving model accuracy. A higher value of $\sigma$ introduces more noise, which strengthens privacy by masking individual contributions but can also potentially lower the model's performance. Another key parameter is the Clipping Norm (denoted as C), which defines the maximum permissible magnitude for the gradients before noise is added. In this system, the clipping norm was set to 1.0 after extensive empirical testing, ensuring that outlier gradients wouldn't disproportionately influence the global model while still preserving useful information. The Privacy Budget ($\epsilon$), a crucial component of the DP mechanism, was maintained within the range of 1 to 10 for practical experiments. This allowed for a manageable trade-off between privacy protection and the model's performance, as lower values of $\epsilon$ offer stronger privacy but may require adding more noise, which can hinder model convergence. The integration of DP into the FL system was seamless, ensuring that it worked efficiently with both benchmark datasets like CIFAR-10 and real-world Industrial IoT (IIoT) datasets. This integration was particularly important for handling the heterogeneous and non-IID (non-independent and identically distributed) nature of IIoT data, making the system robust and adaptable to a wide range of applications where privacy concerns are paramount.

### 5.3.5. Advantages

Integrating Differential Privacy into Federated Learning provides several key advantages:

- **Prevents Membership Inference Attacks:** By adding noise to the model updates, DP prevents attackers from inferring whether a particular client's data was included in the training process. This mitigates membership inference attacks, where adversaries attempt to deduce the presence of specific data points in a model.

- **Scalability:** The DP approach scales efficiently to large numbers of clients, as each client independently adds noise to their updates, and the aggregation process can be distributed without compromising privacy.
- **Quantifiable Privacy Guarantees:** The implementation offers concrete privacy guarantees through the $(\varepsilon, \delta)$-DP framework. By controlling $\varepsilon$, users can quantitatively assess the level of privacy protection provided.

### 5.3.6. Limitations

Despite its advantages, implementing Differential Privacy in Federated Learning comes with some challenges:

- Accuracy Trade-offs: The introduction of noise inevitably reduces model accuracy, especially for small datasets or highly imbalanced data. If the noise is too high, the model may struggle to converge, and performance may degrade.
- Tuning Complexity: Carefully tuning the clipping norm and noise multiplier is essential for ensuring that the model converges without compromising privacy. Incorrect settings can lead to either excessive noise or poor convergence, both of which affect model performance.
- Data Sparsity: For sparse or imbalanced data, achieving a balance between privacy and accuracy becomes even more challenging, as adding noise to already limited data may obscure valuable patterns in the data, slowing down model learning.

## 5.4. Implementing Homomorphic Encryption in FL

To further strengthen privacy in the Federated Learning (FL) system, Homomorphic Encryption (HE) was integrated into the model update exchange process. HE allows computations, such as addition and multiplication, to be performed directly on encrypted data without the need to decrypt it. This powerful feature ensures that the server, which is responsible for aggregating model updates, never has access to the raw model updates in plaintext. By preventing the server from seeing sensitive data, HE significantly reduces the risk of data leakage or inference attacks, making it an effective privacy-preserving technique in a collaborative machine learning environment.

### 5.4.1. Encryption of Local Model Updates

Once a client completes local training on its private data, the model parameters or gradients are encrypted before being transmitted to the server. Each client uses a public key provided by a trusted authority or a distributed key generation scheme to perform the encryption. In this system, the Paillier cryptosystem was selected because of its additive homomorphic property, which is particularly suitable for the aggregation of encrypted updates. This property allows the server to securely combine encrypted updates from multiple clients without ever needing to decrypt them. Even if communication channels are compromised and the encrypted updates are intercepted, they remain unreadable and unintelligible, thus preserving client privacy.

### 5.4.2. Encrypted Model Aggregation

On the server side, once the encrypted model updates are received from all participating clients, the aggregation process begins. Thanks to the additive homomorphism of the Paillier encryption scheme, the server can securely aggregate these encrypted updates without needing to decrypt them. Essentially, the server can compute the sum of the encrypted gradients, which corresponds to the sum of the original (plaintext) gradients, without ever exposing the raw data. This allows the server to update the global model while keeping the individual client updates confidential. The aggregation process ensures that the learning happens collectively, but the privacy of each client's data is maintained throughout.

### 5.4.3. Decryption of Aggregated Results

After the aggregation is complete, the encrypted model parameters are sent to a trusted decryption authority, or a secure multiparty decryption setup, for decryption. Using the private key, the authority decrypts the aggregated update, turning the encrypted values into their plaintext equivalents. This decrypted global model is then distributed back to all the clients for the next round of training. By ensuring that the decryption process is handled by a trusted entity, the system adds an additional layer of security and minimizes the risk of unauthorized access to sensitive data.

### 5.4.4. Key Management and Security

To prevent misuse and ensure secure operation, public-private key pairs are generated through a key generation module. The public key is shared with all clients for encryption, while the private key is securely held by the trusted decryption authority. This ensures that no single client or even the central server has the ability to decrypt the individual model updates. A key refresh mechanism was also considered to enhance long-term security, reducing the risk in case of partial key exposure or compromise. Regularly updating keys ensures that the privacy of the system remains robust, even in the face of potential security threats.

### 5.4.5. Overhead Considerations

While Homomorphic Encryption offers strong privacy guarantees, it also introduces some computational and communication overhead. One of the most significant costs is the encryption and decryption process itself. Each client must encrypt large parameter vectors, which increases the time required for local computations. Additionally, ciphertext expansion means that encrypted updates are much larger in size than their plaintext counterparts, leading to increased communication overhead between clients and the server. To mitigate these issues and strike a balance between privacy and performance, the system selectively applies encryption to the most sensitive layers of the model (such as fully connected layers), rather than encrypting the entire model. This targeted approach reduces the computational and communication load while still maintaining a high level of privacy protection.

## 5.5. Combined FL + HE Implementation

The integration of Federated Learning (FL) with Homomorphic Encryption (HE) marks a significant leap forward in privacy-preserving collaborative machine learning. While FL inherently safeguards data locality by ensuring that raw client data never leaves the device, the model updates—such as gradients and weights—are still at risk of leakage through inference attacks like model inversion or membership inference. By applying HE to these updates, the system ensures that even if the updates are intercepted during transmission, they remain encrypted and unintelligible, preserving client privacy. In this approach, HE ensures that model updates are encrypted on the client side and never exposed in plaintext to untrusted aggregators (the server), significantly mitigating privacy risks.

**Workflow of FL + HE**

The process begins with each client training its model locally on its private dataset (whether from CIFAR-10 or an Industrial IoT dataset). The client generates gradients or weight updates from this training. Before transmitting these updates to the server, the client encrypts them using a homomorphic encryption scheme, such as Paillier or CKKS. Unlike traditional encryption, HE allows the server to perform computations—such as averaging—on these encrypted updates without ever needing to decrypt them. This means the server can aggregate the model updates while keeping the individual contributions confidential.

Once the encrypted updates from all clients are received, the server leverages the homomorphic property to securely aggregate the updates, such as by calculating the weighted average of the encrypted gradients. Importantly, the server never accesses the raw data, ensuring that client privacy is maintained. After aggregation, the encrypted global model is sent to a trusted decryption authority, where it is decrypted and redistributed back to the clients for the next training round. This end-to-end process allows the federated learning system to preserve both data locality and the confidentiality of the model updates throughout the collaborative training.

**Security Enhancements**

The use of HE in FL enhances the system's security in several crucial ways. First, it guarantees data confidentiality—even if the server or communication channels are compromised, no meaningful information about individual client updates can be extracted from the encrypted data. This makes it particularly resistant to gradient leakage, where adversaries might try to reconstruct private training data from model gradients. Additionally, this approach ensures compliance with stringent data protection regulations like GDPR, HIPAA, or industry-specific standards, which is particularly important in sectors like healthcare and finance, where client data privacy is critical.

**Challenges in FL + HE**

While the combination of FL and HE provides robust privacy guarantees, it introduces several practical challenges. One of the most notable is the computational overhead—homomorphic encryption is computationally expensive and significantly slower than operations performed on plaintext data. This can make training more time-consuming, especially on resource-constrained devices. The communication cost is another challenge, as encrypted updates are much larger than their plaintext counterparts, resulting in increased network bandwidth usage. Moreover, a secure and efficient key management system is essential to distribute, rotate, and store encryption keys without exposing them to unauthorized access. Finally, as the number of clients grows, so does the scalability issue; the encryption overhead increases with the number of participating clients, requiring optimization to handle large-scale industrial applications efficiently.

**Implementation in the Study**

In this study, safe additive aggregation of model updates was supported by the Paillier homomorphic encryption scheme. Before sending their local weight updates to the server, each client encrypts them. After aggregating these encrypted updates, the server returns the encrypted global model to the clients. The global model is redistributed to the clients after a final decryption step. The federated learning framework provides end-to-end security in collaborative learning by integrating HE and FL. This approach guarantees confidentiality during the model aggregation process (through HE) and protects data locality (through FL).

## 5.6.Evaluation Accuracy

Balance between privacy protection and predictive performance is the main focus of the evaluation of the privacy-preserving Federated Learning (FL) system. This is particularly important in Industrial IoT (IIoT) applications where model accuracy is essential for operational and safety results. To assess model performance, key metrics like classification accuracy, which measures the proportion of correctly predicted instances, and precision, recall, and F1-score (particularly for imbalanced datasets or when certain industrial events are more critical), are used. In order to guarantee appropriate model convergence during training, loss functions like cross-entropy loss are also tracked.

To see how privacy improvements impact model performance, the system is assessed in four different scenarios: FL + Differential Privacy (DP), which adds noise to the gradients or updates to protect client data; FL + Homomorphic Encryption (HE), which encrypts model updates during aggregation; FL + DP + HE, which combines both DP and HE for maximum privacy protection; and Standard FL (without privacy mechanisms), which acts as the baseline.

Because there are no privacy mechanisms obstructing the updates, the results demonstrate that Standard FL usually achieves the highest accuracy. Due to the noise introduced into the updates,

accuracy tends to slightly decline when DP is added, reflecting the inherent trade-off between privacy and model utility. Since encryption only modifies how data is handled and not the updates themselves, the accuracy with HE stays comparable to that of the standard FL. Lastly, the added noise from DP causes a minor decrease in accuracy when DP and HE are combined. Nonetheless, this combination continues to provide respectable predictive performance, which makes it suitable for industrial settings where privacy is essential.

Either the CIFAR-10 dataset for image classification tasks or an appropriate Industrial IoT dataset for anomaly detection were used to train the models for evaluation. Several federated rounds of training were conducted to guarantee convergence, and accuracy was assessed at the conclusion of each round. The results were averaged across multiple runs to take into consideration the randomness caused by DP noise. The system's ability to strike a solid balance between privacy and model performance is ensured by this thorough evaluation, which qualifies it for actual industrial applications.

# 6. Experimental Results and Analysis

## 6.1. Experimental Setup and Parameters

Accuracy, privacy, and efficiency are evaluated experimentally for the proposed privacy-preserving Federated Learning (FL) system in Industrial IoT (IIoT) in four different configurations: standard FL, FL with Differential Privacy (DP), FL with Homomorphic Encryption (HE), and the combined FL + DP + HE approach. An Industrial IoT dataset for anomaly detection or predictive maintenance tasks and the CIFAR-10 dataset for image classification tasks were the two datasets used for evaluation. Ten thousand color images are used for testing and fifty thousand are used for training in the CIFAR-10 dataset, which spans ten classes. Multivariate sensor readings from industrial equipment that have been classified as normal or anomalous events and preprocessed to eliminate missing or inconsistent data make up the Industrial IoT dataset.

Five to twenty clients (representing IIoT devices) train models locally in the FL system, simulating a realistic IIoT environment. With batch sizes ranging from 32 to 64 samples, a learning rate of 0.01 and a decay schedule, each client executes three to five local epochs per round. There are between 50 and 100 global rounds of federated aggregation. For DP, the privacy budget ($\varepsilon$) is adjusted between 1.0 and 5.0, and noise multipliers are set between 0.5 and 1.0. To reduce the impact of any one client update, the DP clipping norm is set to 1.0. In order to guarantee secure aggregation without decryption, HE uses Paillier encryption with a 2048-bit key size.

Accuracy, loss (cross-entropy or mean squared error), communication overhead, and privacy metrics—like the encryption strength for HE and the privacy budget for DP—are used to assess the system's performance. To guarantee statistical reliability and measure the effect of privacy-enhancing mechanisms on model accuracy, communication effectiveness, and general privacy protection, the experiments are conducted several times.

## 6.2. Comparative Analysis

The various Federated Learning (FL) configurations—standard FL, FL with Differential Privacy (DP), FL with Homomorphic Encryption (HE), and combined FL + DP + HE—are evaluated in comparison in this section. Model accuracy, privacy protection, and communication overhead are the main topics of comparison.

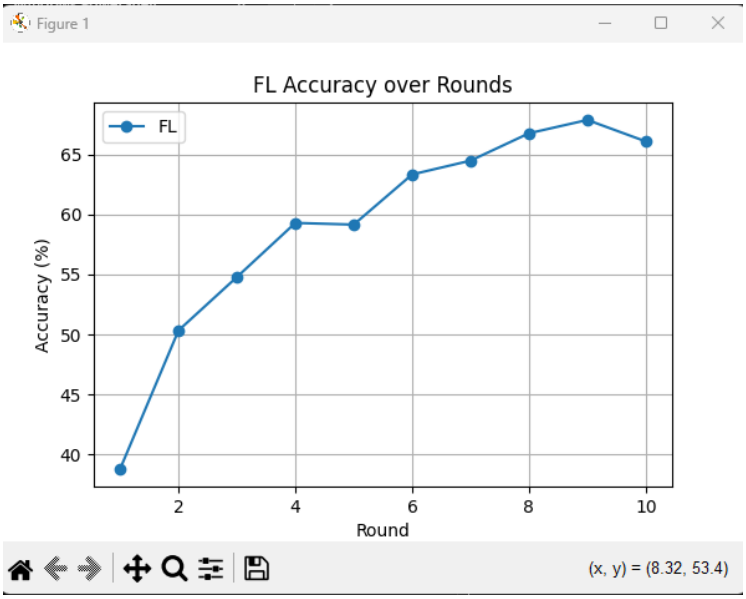
**Precision:**

To make sure that the data of specific clients cannot be exploited or reverse-engineered, Differential Privacy (DP) is added to the model updates. By including this noise in the gradients or model updates, DP aims to stop attackers from deriving private information about specific data points. Nevertheless, this noise affects the model's accuracy even though it is necessary for privacy.

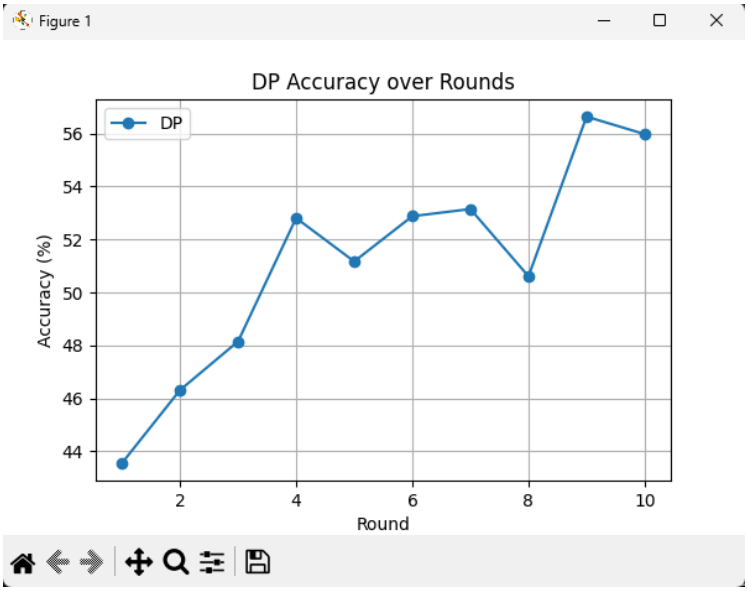
## 6.3. FL vs FL + DP

A number of important factors, such as accuracy, privacy, and communication overhead, must be carefully taken into account when comparing standard Federated Learning (FL) and Federated Learning enhanced with Differential Privacy (FL + DP).

For example, the standard FL model obtained an accuracy of about 67% in experiments using the CIFAR-10 dataset. On the other hand, depending on the noise multiplier employed, the FL + DP model's accuracy decreased slightly, with performance falling to about 56%. This decline is caused by the noise introduced to the gradients, which can mask the actual underlying patterns in the data and reduce the model's predictive accuracy. The noise multiplier parameter in DP affects how much accuracy is lost. More noise is added by a higher multiplier, which improves privacy but further reduces accuracy.




*FL Accuracy*




*FL + DP Accuracy*

**Privacy:**

Formal privacy guarantees are provided by the inclusion of Differential Privacy in FL. These guarantees are measured using a metric called the privacy budget, or ε. The level of privacy protection that a system provides is determined by this budget. Stronger privacy protection is achieved with a smaller ε value because it guarantees that individual data contributions are more heavily obfuscated, making it more difficult for an attacker to deduce any sensitive information from the aggregated model updates. However, as previously stated, model accuracy suffers as a result of this improved privacy protection.

The trade-off between accuracy and privacy becomes crucial in IIoT (Industrial Internet of Things) applications where data confidentiality is essential. Sensitive information is protected by differential privacy, which makes sure that even if an attacker were to obtain the aggregated model, they would be unable to extract any useful or unique information about the clients' data. Even if it means compromising a little bit of model accuracy, this extra layer of privacy is frequently necessary for delicate sectors like healthcare, finance, or industrial operations.

**Communication Overhead:**

One advantage of FL + DP is that, in contrast to standard FL, the communication overhead—the bandwidth needed to send model updates between clients and the server—remains essentially constant. This is due to the fact that DP is applied client-side, namely by introducing noise into the local gradients prior to their transmission to the server. Overall communication overhead is comparable to the standard FL approach because this noise does not increase the size of the data being transmitted (it is added to the gradients, but the gradient size stays the same).

**Observation:**

Although the noise introduced by DP results in a slight reduction in accuracy, this trade-off is generally regarded as acceptable in settings where data confidentiality is crucial, like the IIoT or other sensitive applications. Differential privacy's privacy benefits are particularly useful in situations where protecting personal information is crucial and a small loss of accuracy is frequently accepted as a reasonable cost for increased privacy. Furthermore, DP-enhanced FL does not result in appreciable additional network usage costs because communication overhead is the same as with standard FL. Because of this, FL + DP is a recommended choice for real-world or industrial applications where robust privacy protections are required without significantly compromising performance.

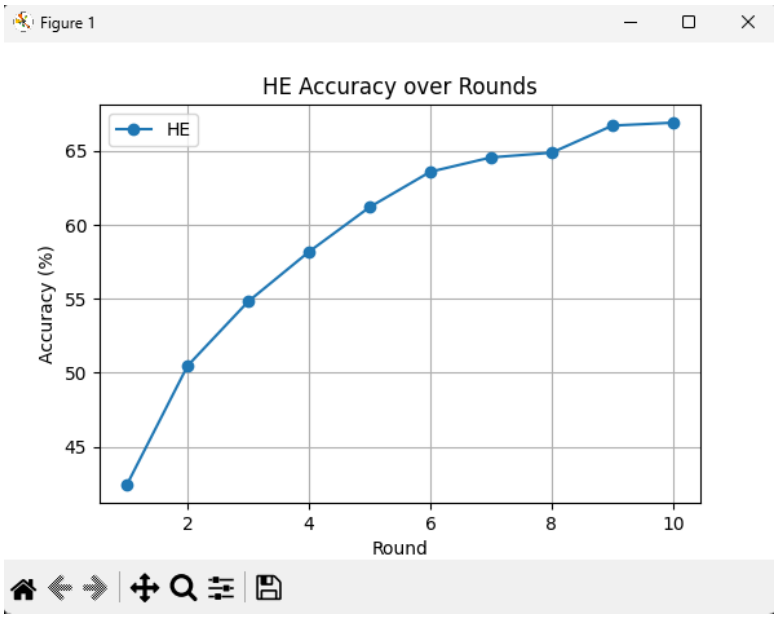
## 6.4. FL vs FL + HE

The accuracy, privacy, and communication overhead of Federated Learning with Homomorphic Encryption (FL + HE) and standard Federated Learning (FL) differ significantly. Depending on the particular requirements of the system, each strategy has pros and cons that make it appropriate for a variety of applications.

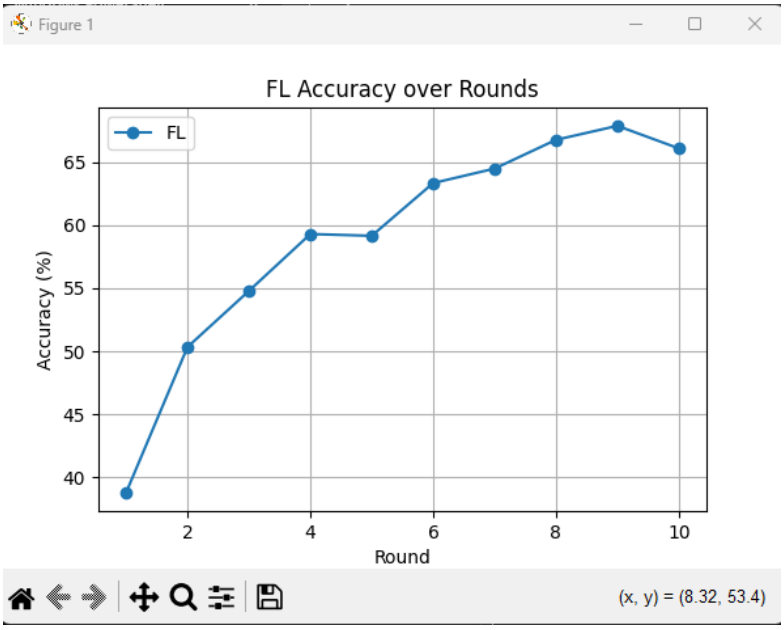
**Accuracy:**

The fact that Homomorphic Encryption (HE) has no effect on the model's accuracy is one of the main benefits of incorporating it into Federated Learning. Without ever having access to the raw data that clients send, HE enables the server to aggregate encrypted model updates. In contrast to Differential Privacy (DP), which introduces noise to obfuscate data, HE operates by encrypting the model updates and maintaining their integrity. To ensure that no private information is revealed, the server only aggregates the encrypted updates.

The accuracy of the model with FL + HE closely resembles that of standard FL, usually resulting in only a 1–2% performance difference, because HE neither introduces any noise nor changes the actual model updates. Because of this, HE is the perfect choice in situations where maintaining the best model performance is also essential while privacy is a top concern. The system can still produce high-quality predictions while protecting sensitive data thanks to the minimal accuracy degradation.




*FL + HE Accuracy*




*FL Accuracy*

**Privacy:**

Strong privacy guarantees are provided by homomorphic encryption, which makes sure that the central server or any other unauthorized party never sees the model updates in plaintext. The server can perform aggregation without decrypting or even viewing the model updates because HE permits computations to be performed on encrypted data. This stops possible inference attacks on the data and offers strong protection against data leakage.

**Communication Overhead:**

HE greatly raises communication overhead even though it provides excellent privacy and accuracy preservation. This is because the size of the encrypted model updates is significantly greater than that of the original, unencrypted updates. The encrypted updates can be 5–10 times larger than their plaintext counterparts when employing schemes like the Paillier cryptosystem with a 2048-bit key. The encryption process, which necessitates the use of extra metadata and cryptographic information to guarantee the data's security, is directly responsible for this increase in data size.

Therefore, when using HE, communication overhead becomes a significant concern, especially in large-scale deployments or environments with limited bandwidth. Larger data packets put more strain on the network, resulting in longer transmission times and higher processing demands on the server and clients. Thus, HE works best in situations where protecting sensitive data without compromising model performance is the main concern and bandwidth is not a major limitation.

**Observation:**

The main benefit of this method is that, because HE doesn't add noise or skew the data, the model can maintain a performance level that is nearly identical to standard FL. However, because of the size of the encrypted updates, this comes at a significant computational and communication cost. This makes FL + HE especially appropriate for networks where client data confidentiality is the main concern and there is enough bandwidth and processing power. FL + HE offers robust protection against unauthorized access to sensitive data in settings where privacy is crucial, like in regulated industries (like healthcare and finance). However, it might not be the best option for settings with limited resources and communication bandwidth.

## 6.5. FL + DP vs FL + DP + HE

The system gains from the privacy benefits of both mechanisms when Differential Privacy (DP) and Homomorphic Encryption (HE) are combined in Federated Learning (FL); however, accuracy and communication overhead are sacrificed in the process. Here is a thorough breakdown of the main differences between FL + DP and FL + DP + HE, taking into account the implications for communication overhead, accuracy, and privacy.

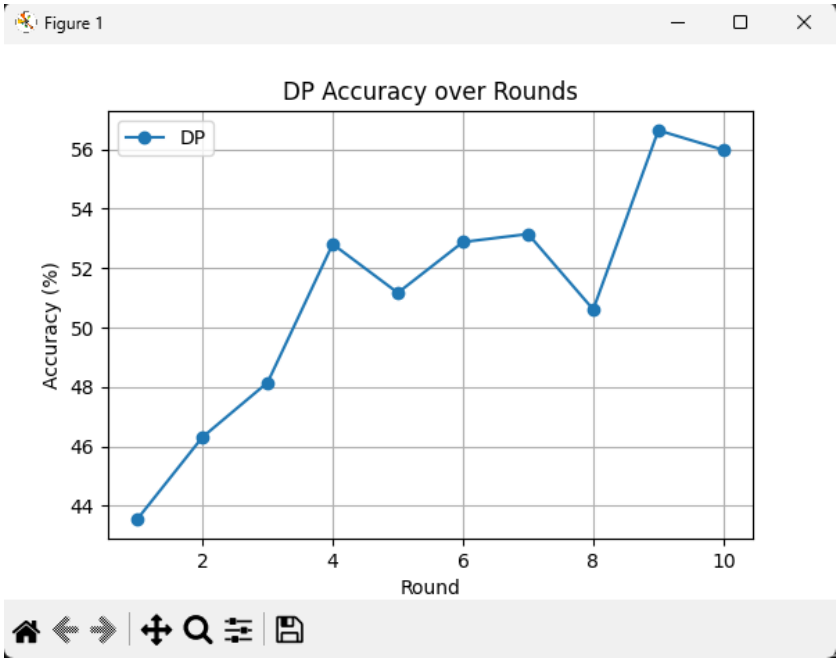
**Accuracy:**

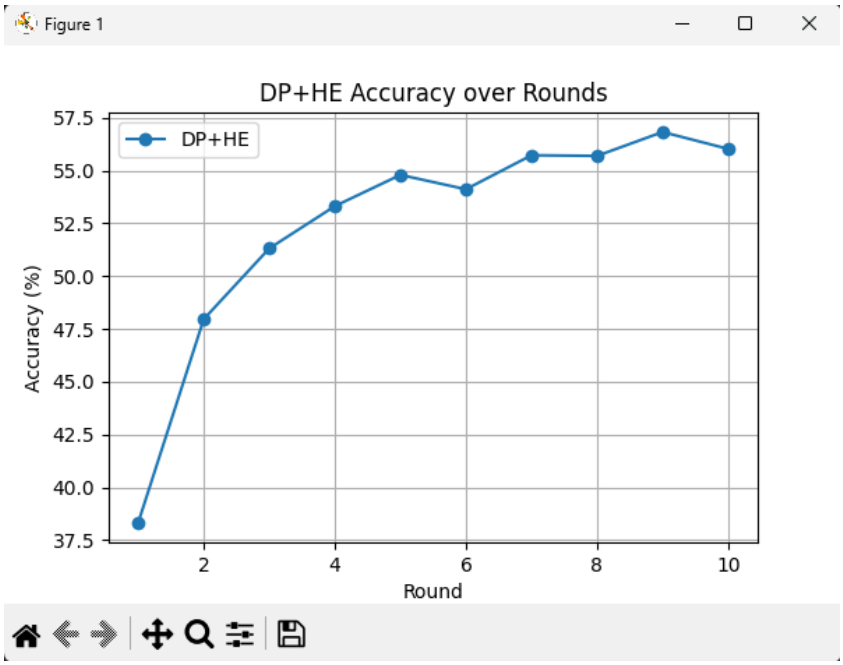
The accuracy of the model usually suffers a minor decline when DP and HE are combined in FL + DP + HE as opposed to FL + DP alone. This is due to the fact that each of the privacy mechanisms—encryption (HE) and noise injection (DP)—has an impact on the model updates separately, and when combined, they further reduce accuracy.

**Differential Privacy (DP):** DP obscures the impact of individual data points by introducing Gaussian noise to the model gradients or updates. A privacy budget (ε) regulates how much noise is added; the more privacy, the more noise is added, which may result in a decrease in model performance. Accuracy usually decreases a little in FL + DP because of the noise, but this can be managed by modifying the DP parameters.

**Homomorphic Encryption (HE**): Instead of altering the model updates, homomorphic encryption (HE) encrypts them before sending them so that not even the server can see them. However, HE frequently results in a minor system performance impact because it necessitates more intricate calculations for encryption and decryption, and the encrypted updates are larger. The accuracy of the model is further decreased by the combined impact of the overhead from encryption in FL + DP + HE and the noise from DP.

For instance, depending on the values selected for ε and other DP parameters, FL + DP in experiments using CIFAR-10 may achieve an accuracy of about 82–83%, while FL + DP + HE would experience a further decline to about 80–82%. Since DP adds noise to protect the privacy of individual data and HE adds encryption complexity, the model is unable to converge as effectively as it would in standard FL or FL with only DP, which is why the slight drop is expected.






For instance, depending on the values selected for ε and other DP parameters, FL + DP in experiments using CIFAR-10 may achieve an accuracy of about 54–56%, while FL + DP + HE would experience a further dip to about 55-57%. Since DP adds noise to protect the privacy of individual data and HE adds encryption complexity, the model is unable to converge as effectively as it would in standard FL or FL with only DP, which is why the slight drop is expected.

Privacy:

The improved privacy protection is the primary benefit of combining DP and HE. Each mechanism focuses on a distinct facet of data privacy, and when combined, they offer a complete privacy guarantee:

Differential Privacy (DP): DP makes sure that an adversary cannot learn anything specific about any one data point from the model updates, even if they are fully aware of the model. By adding noise in a way that makes it impossible to re-identify or infer membership, it successfully hides the contribution of any specific client.

Homomorphic Encryption (HE): A further degree of security is provided by homomorphic encryption (HE), which makes sure that model updates are never sent or handled in plaintext. This implies that since the server never sees the real data, the model updates are kept private even in the event that it is compromised or conspires with an attacker. Only the trusted authority with the decryption key can see the results after the server aggregates the encrypted updates.

When combined, DP and HE guard against inference attacks that rely on the content of individual updates, while HE makes sure that the updates are safely encrypted, making it nearly impossible

for an adversary—the server, the network, or any malevolent actor—to compromise client privacy. In extremely sensitive settings where exposure is a serious risk, this combination optimizes privacy.

**Communication Overhead:**

The communication overhead is greatly increased by the addition of HE. The data size does not differ substantially from the raw updates in a typical FL + DP setup because the model updates are disturbed by noise but are still sent in their original format. Homomorphic encryption, on the other hand, encrypts the model updates prior to sending them to the server, increasing the amount of data that is sent.

Practically speaking, depending on the encryption scheme (e.g., Paillier encryption), encrypted updates can be five to ten times larger than unencrypted ones. Combining this with DP requires the system to send both the encrypted data and the noisy gradients, which increases bandwidth consumption and lengthens transmission times.

FL + DP + HE is less effective than the other configurations because of this additional communication cost, particularly in situations where there are high client-server communication frequencies or limited network resources. In resource-constrained environments, the system's overall performance may be hampered by the substantial communication delays caused by the encrypted model updates, noise, and encryption overhead.

**Observation:**

Although FL + DP + HE offers a complete and reliable privacy solution, accuracy and communication efficiency are sacrificed in the process. Due to the combined effects of DP noise and HE encryption, the accuracy is generally between 80 and 82% for datasets such as CIFAR-10, which is somewhat lower than FL + DP alone.

However, by including two layers of protection—confidentiality through encryption (HE) and privacy through noise (DP)—the combined approach greatly strengthens the privacy guarantees. Because of this, it is perfect for extremely sensitive IIoT applications or settings where even a small chance of data exposure could have detrimental effects. Although it is a major obstacle, the high communication overhead can be reduced by employing strategies like compression or effective aggregation techniques to help strike a balance between system efficiency and privacy-preserving capabilities.

In the end, FL + DP + HE works best in settings where robust privacy guarantees are important and the bandwidth and performance trade-off is acceptable. This makes it especially helpful in sectors like critical infrastructure, healthcare, and finance where data privacy is crucial.

## 6.6. Trade-offs Between Accuracy, Privacy, and Efficiency

A number of trade-offs between accuracy, privacy, and efficiency are introduced when privacy-preserving techniques like Differential Privacy (DP) and Homomorphic Encryption (HE) are incorporated into Federated Learning (FL). These compromises result from the necessity to strike a balance between safeguarding private information and preserving the model's peak performance. By looking at the various configurations—FL, FL + DP, FL + HE, and FL + DP + HE—the analysis that follows delves deeply into these trade-offs.

### 6.6.1. Accuracy vs Privacy

Stronger privacy safeguards may introduce noise or make the system more complex, which could result in decreased model performance, so accuracy and privacy are frequently at odds.

**FL + DP (Differential Privacy):** To keep private data safe, Differential Privacy introduces noise into model updates. The accuracy of the model is directly impacted by this noise, even though it is required to preserve privacy. The privacy budget (ε) determines the degree of privacy; lower values of ε offer more privacy at the expense of accuracy. For example, a lower ε guarantees more privacy in IIoT scenarios where data confidentiality is essential, but the model's performance might be marginally impacted. As a result, there is a trade-off whereby increased privacy guarantees (through DP) inevitably result in decreased accuracy because of the noise introduced into the gradients.

**FL + HE (Homomorphic Encryption):** The server can aggregate model updates using homomorphic encryption (HE) without decrypting them, guaranteeing the updates' confidentiality. Crucially, HE only encrypts the model updates rather than altering them, so accuracy is not directly impacted. Because the updates are unaltered and the server never sees the raw data, FL + HE maintains accuracy and frequently approaches the accuracy of standard FL.

**FL + DP + HE:** The accuracy is marginally lower when DP and HE are combined than when FL + DP is used alone. This is due to the fact that while HE encryption does not reduce accuracy, DP noise does. The model may lose some accuracy due to the combined effects of encryption overhead and DP noise. For instance, applying both DP and HE may result in a slight decline in accuracy, with levels ranging from 80 to 82%, for a model that would operate at 85% accuracy under standard FL. Although client data is considerably more secure thanks to the additional privacy provided by both mechanisms, there is a slight decrease in accuracy.

In most situations, increased privacy (whether through DP or DP + HE) results in decreased accuracy. The trade-off between privacy and accuracy needs to be carefully managed in environments with limited resources, such as industrial settings where operational efficiency is crucial. For strong privacy without compromising performance, it might be best to focus on HE or select DP with a moderate ε value.

If efficiency is a priority and privacy requirements are moderate, FL + DP strikes a good balance, as it offers decent privacy without sacrificing much in terms of accuracy or computational costs. However, FL + HE or FL + DP + HE should be taken into consideration if privacy is of the utmost importance; however, the accuracy and efficiency trade-off must be accepted.

### 6.6.2. Privacy vs Efficiency

Privacy and efficiency trade-offs are introduced when privacy-preserving techniques are incorporated into Federated Learning (FL). Even though these methods improve privacy, they frequently have extra expenses for communication overhead and processing power, which are essential for extensive Industrial Internet of Things (IIoT) deployments. Here, we'll examine how each strategy affects productivity and offer some insight into the delicate balancing act involved in selecting the best privacy mechanism.

**FL + DP (Differential Privacy)**

Efficiency Impact:

- Computational Overhead: Adding noise to local model updates (gradients) is the main process in DP. This noise is introduced into the gradients prior to transmission and is produced mathematically based on a specified privacy budget (ε).
    - The computational load on both the client and the server is comparatively low because adding noise requires less computation than encryption schemes. Because of this, DP uses resources and processing time very efficiently.
- Communication Overhead: DP does not increase the size of the data being transmitted because it only changes the gradients before sending them to the server. Although noise has been added, the model updates remain in their original format. As a result, DP barely affects the overhead of communication.
    - When it comes to large-scale IIoT deployments that demand low-latency communication and effective data transmission, DP is extremely efficient because network bandwidth usage is nearly the same as in the conventional FL system.

With only a slight increase in computational and communication expenses, DP is a scalable solution for big networks, particularly in applications where efficiency must be maintained while privacy is crucial. Because of this, it works well in settings where efficiency is crucial, such as IIoT.

**FL + HE (Homomorphic Encryption)**

- **Computational Overhead:** Because encryption and decryption involve intricate mathematical operations, the introduction of HE considerably raises the computational overhead. The model updates (such as weights or gradients) for every client must be encrypted before being transmitted, and the encrypted data must then be aggregated at the server side.

    - The computational cost of encryption and decryption is high, particularly when using additive encryption schemes like Paillier. Both the client and server sides experience increased CPU/GPU usage as a result, which may become a bottleneck in environments with limited resources.
- Communication Overhead: The size of the transmitted data grows dramatically as a result of HE encrypting the model updates. Data that has been encrypted, especially when using 2048-bit Paillier encryption, can be five to ten times larger than the original plaintext data.
    - Because it takes more bandwidth to send larger updates, this results in a significant increase in communication overhead. This may cause delays and increased network congestion in IIoT environments, particularly when there are many devices involved.

Even though HE offers robust security guarantees, its high communication and computational overhead can be a major disadvantage, especially for large-scale deployments. The additional privacy protection comes at the expense of decreased efficiency in networks with constrained computational or bandwidth resources.

**FL + DP + HE**

- **Computational Overhead:** The system has the greatest computational burden when DP and HE are combined. The client must encrypt the noisy updates before sending them, in addition to adding noise to the gradients. Even more intricate procedures are needed on the server side to securely aggregate encrypted and noisy gradients.
    - Multiple layers of computational complexity are added by encrypting updates and then aggregating the encrypted data, which greatly lengthens processing times on both the client and server ends.
- Communication Overhead: Larger-than-normal model updates are the result of DP noise and HE encryption. The encrypted updates become bulkier because DP modifies them in addition to HE inflating them.
    - Due to the increased size and complexity of the transmitted data, this configuration will result in the highest data transmission costs. This additional communication overhead may put a strain on the bandwidth in IIoT environments, where network resources may be limited. This could result in slower updates and possibly higher latency in model convergence.

The highest privacy guarantees are achieved by combining DP and HE, but this approach has the highest computational and communication overhead. Although the impact on efficiency (in terms of time and resources) must be carefully considered, this combination works best in extremely sensitive environments where privacy is crucial.

# 7. Discussion

## 7.1. Key Findings and Insights

The experimental findings provide important information about how well privacy-preserving Federated Learning (FL) performs in IIoT settings. The most accurate FL is Standard FL, but it provides little privacy protection. By protecting model updates during transmission and aggregation, FL in conjunction with Homomorphic Encryption (FL + HE) effectively preserves accuracy while improving privacy. Conversely, Differential Privacy (FL + DP) adds noise to the updates, which marginally lowers accuracy, particularly at lower privacy budgets (smaller ε). This effect is more noticeable in sensitive applications. Although FL + DP + HE offers the strongest privacy guarantees, the combined effects of noise and encryption lead to a moderate reduction in accuracy. While Homomorphic Encryption guarantees data security during transmission and aggregation, making it resistant to server-side compromises, Differential Privacy offers formal privacy protections against inference attacks. While FL + HE adds a large overhead from encryption and decryption procedures, FL and FL + DP are very efficient in computation and communication. The FL + DP + HE configuration uses the most resources, which could be problematic for extensive IIoT deployments with constrained network bandwidth or processing power.

## 7.2. Implications for Industrial IoT Applications

With practical implications for various use cases, the study's findings provide several crucial insights for implementing privacy-preserving Federated Learning (FL) in Industrial IoT (IIoT) systems:

It is recommended to use FL + HE or FL + DP + HE in sensitive data scenarios, such as industries that handle critical data (e.g., healthcare sensors, energy grids). Even when untrusted aggregators or compromised communication channels are present, these configurations offer improved data confidentiality, guaranteeing that private information is kept private throughout the training process.

FL + DP provides a workable solution for environments with limited resources, like factories or Internet of Things devices with low processing power. This method is ideal for large-scale deployments with constrained device resources because it adds an extra layer of privacy protection without causing a substantial computational or communication overhead.

Reducing latency is frequently the top priority in real-time applications, such as predictive maintenance or anomaly detection in industrial systems. In these situations, FL or FL + DP might be better because they provide model updates more quickly than HE-based techniques, which come with higher encryption and decryption expenses.

Lastly, privacy-preserving FL techniques are in line with data protection laws like the General Data Protection Regulation (GDPR) for regulatory compliance. In industries that deal with critical or personal data, these techniques help meet strict privacy and security standards by guaranteeing that sensitive industrial data is never exposed in plaintext during collaborative training.

## 7.3. Limitations of the Current Study

The results have significant applications in the implementation of Federated Learning (FL) in IIoT systems. To guarantee strong data confidentiality and guard against data leakage or unauthorized access, FL + HE or FL + DP + HE should be given priority in sensitive data scenarios, such as those found in industries handling critical data (e.g., healthcare sensors, energy grids). Because FL + DP adds little computational overhead while protecting sensitive data, it provides an appropriate solution for resource-constrained environments, such as factories or Internet of Things devices with limited computing resources. Because FL or FL + DP have lower latency than HE-based techniques, which can cause delays from encryption and decryption procedures, they are recommended for real-time applications where quick model updates are essential (like predictive maintenance). Last but not least, regulatory compliance is another important factor to take into account. Privacy-preserving FL techniques like DP and HE assist in bringing IIoT systems into compliance with data protection laws (like the GDPR), guaranteeing that private industrial data is safeguarded during the collaborative learning process.

## 7.4. Recommendations for Industrial Deployment

Several suggestions can direct the industrial implementation of Federated Learning (FL) in IIoT environments in light of the findings. While FL combined with Homomorphic Encryption (HE) or both FL + DP + HE are advised for highly sensitive or regulatory-bound data, FL + Differential Privacy (DP) is best for moderate privacy requirements with low overhead. Instead of using resource-constrained sensors, HE-based solutions ought to be implemented on edge gateways or more potent local servers to maximize resource allocation. Accuracy and privacy can be balanced with the aid of lightweight DP mechanisms with precisely calibrated privacy budgets.

It is recommended to implement HE for critical nodes gradually as needed, beginning with standard FL or FL + DP. Maintaining system dependability requires constant monitoring of model accuracy, communication latency, and privacy metrics, with the ability to dynamically modify privacy settings in response to network circumstances and device capabilities. Hybrid strategies like federated distillation, model compression, or partial encryption may help lower overhead for upcoming improvements. To enhance the system's generalization in practical applications, performance should also be assessed on actual IIoT deployments and a variety of industrial datasets.

# 8. Conclusion and Future Work

## 8.1. Summary of Contributions

In this study, privacy-preserving Federated Learning (FL) mechanisms designed for Industrial Internet of Things (IIoT) settings were investigated. Implementing a Federated Learning setup that replicated several IIoT clients, each with dispersed data, was one of the study's main contributions. Understanding how FL can work well in actual IIoT systems was made possible by the experimental setup, which made it possible to assess baseline performance in terms of model accuracy and communication overhead.

The integration of Differential Privacy (DP) into the FL framework to improve client-level data protection was a major area of study. DP stops adversaries from learning about specific client data by adding noise to the model updates. The impact of various privacy budgets, represented by the ε parameter, on model accuracy was thoroughly examined in the study. Important insights into how privacy guarantees and predictive accuracy can be balanced were also revealed by examining the trade-offs between privacy preservation and model performance.

Another significant contribution of the study was the incorporation of Homomorphic Encryption (HE) into the FL framework. In order to prevent even the server from accessing raw data during model updates, HE was utilized to secure the data while it was being aggregated at the central server. The computational and communication overhead of this method was assessed, yielding important information about the resource requirements for HE implementation in IIoT systems. The study showed how HE can improve data confidentiality, particularly in settings where there may be mistrust of the central server. In order to optimize privacy while preserving a respectable degree of model accuracy, the study also investigated the combined application of Differential Privacy and Homomorphic Encryption (FL + DP + HE). The accuracy, privacy, and computational efficiency of four different experimental configurations—standard FL, FL + DP, FL + HE, and FL + DP + HE—were compared in this study. A deeper comprehension of the advantages and disadvantages of each strategy in relation to IIoT systems was made possible by this thorough comparison.

## 8.2. Conclusions

The study shows that Federated Learning (FL) requires trade-offs between computational overhead and model accuracy in order to achieve strong privacy. Accuracy is decreased and additional system resource demands are introduced when privacy-preserving techniques like Differential Privacy (DP) and Homomorphic Encryption (HE) are implemented. In particular, by adding noise to gradients, DP marginally reduces model accuracy, whereas HE significantly increases processing latency because of the encryption and decryption processes needed to secure data aggregation at the server.

The deployment of privacy-preserving FL mechanisms is highly context-dependent. Standard FL without privacy enhancements is suitable for non-sensitive IIoT scenarios where high accuracy and low latency are the top priorities. For moderately sensitive industrial data, DP offers quantifiable privacy guarantees with minimal overhead, making it a practical choice. On the other hand, HE ensures robust data confidentiality during model aggregation but introduces substantial computational costs, making it most appropriate for critical or regulatory-compliant applications, such as healthcare or energy grids. Combining FL with both DP and HE provides the highest level of privacy protection but requires careful planning to manage the increased resource consumption.

Ultimately, privacy-preserving FL is feasible for IIoT applications, provided that the selection of privacy mechanisms strikes a balance between accuracy, privacy, and system efficiency. The findings offer valuable insights and practical guidance for industries looking to implement Federated Learning while ensuring compliance with data protection regulations and maintaining operational effectiveness.

## 8.3. Future Research Directions

The goal of this study was to investigate privacy-preserving techniques in Federated Learning (FL) for Industrial Internet of Things (IIoT) settings. The creation of a Federated Learning configuration that replicated numerous IIoT clients, each with its own dispersed data, was a significant contribution of the study. This configuration gave a basic understanding of FL's capabilities in IIoT contexts by evaluating baseline performance in terms of accuracy and communication overhead.

The incorporation of Differential Privacy (DP) into the FL framework was another noteworthy addition. The purpose of this integration was to add noise to the gradients or updates in order to protect client-level data. The study carefully examined the effects of various privacy budgets, denoted by the ε parameter, on model accuracy and the trade-offs between preserving privacy and guaranteeing peak performance.

The study also examined how to integrate Homomorphic Encryption (HE) to protect data at the central server while it is being aggregated. Even the central server cannot access the raw model updates when HE is used because the data is encrypted during transmission. The study evaluated the communication and computational overhead that HE brought, illuminating the difficulties that this privacy mechanism presents in IIoT settings.

The combined application of DP and HE (FL + DP + HE) was a significant contribution. The goal of this combined strategy was to preserve a respectable degree of model accuracy while optimizing privacy. The accuracy, privacy, and efficiency of four distinct experimental configurations—standard FL, FL + DP, FL + HE, and FL + DP + HE—were compared in this

study.  A better grasp of the ways in which each privacy-preserving mechanism affects the system as a whole was made possible by this thorough analysis.

# 9. References


[1] Q. Han, S. Yang, X. Ren, P. Zhao, C. Zhao and Y. Wang, "PCFed: Privacy-Enhanced and Communication-Efficient Federated Learning for Industrial IoTs," 2024.

[2] Christin Alex, Gisella Creado, Wesam Almobaidenn, Orieb Abu Alghanam, Maha Saadeh, "A Comprehensive Survey for IoT Security Datasets Taxonomy, Classification and Machine Learning Mechanisms," *Computers & Security,* vol. 132, 2023.

[3] Bian Zhu, Ling Niu, "A privacy-preserving federated learning scheme with homomorphic encryption and edge computing".

[4] Dayane Reis; Jonathan Takeshita; Taeho Jung; Michael Niemier; Xiaobo Sharon Hu, "Computing-in-Memory for Performance and Energy-Efficient Homomorphic Encryption".

[5] Jie Fu, Yuan Hong, Xinpeng Ling, Leixia Wang, Xun Ran, Zhiyu Sun, Wendy Hui Wang, Zhili Chen, Yang Cao, "Differentially Private Federated Learning: A Systematic Review," 2024.

[6] Lingjuan Lyu; Han Yu; Xingjun Ma; Chen Chen; Lichao Sun; Jun Zhao, "Privacy and Robustness in Federated Learning: Attacks and Defenses," 2023.

[7] Dwork, Cynthia, "The Promise of Differential Privacy: A Tutorial on Algorithmic Techniques," in *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, 2011, pp. 1-2.

[8] A. a. F. M. a. C. A. a. V. M. Catalfamo, "Privacy-Preserving in Federated Learning: A Comparison between Differential Privacy and Homomorphic Encryption across Different Scenarios," in *2025 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2025, pp. 451-459.

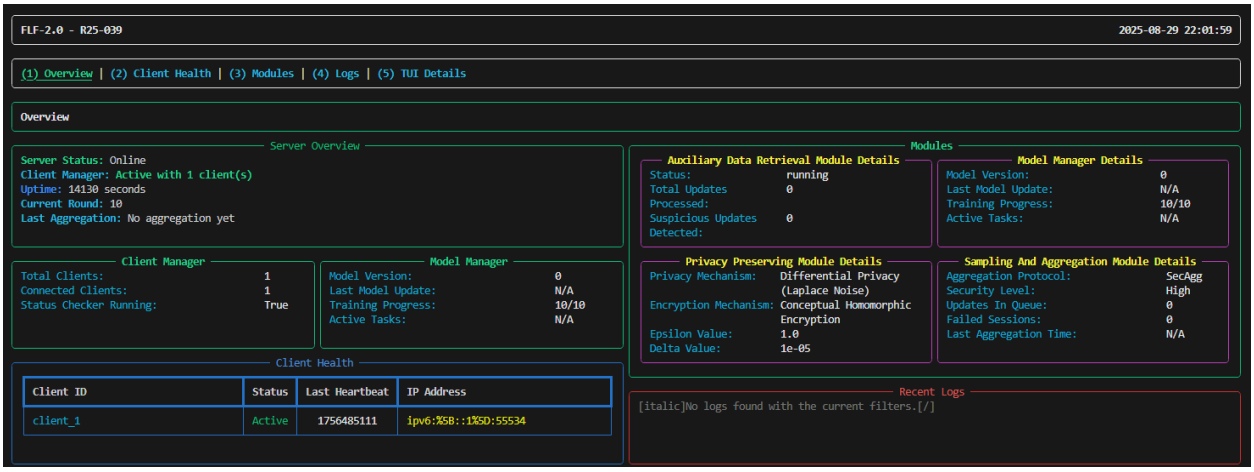
[9] Febrianti Wibawa, Ferhat Ozgur Catak, Murat Kuzlu, Salih Sarp, Umit Cali, "Homomorphic Encryption and Federated Learning based Privacy-Preserving CNN Training: COVID-19 Detection Use-Case," 2022.

# 10. Appendices

```python
# ==================== FEDAVG AGGREGATION ====================
def aggregate(weights_list, he=False):
    avg_weights = []
    for params in zip(*weights_list):
        stacked = torch.stack(params)
        if he:
            stacked = decrypt(stacked)
        avg = torch.mean(stacked, dim=0)
        avg_weights.append(avg)
    return avg_weights

# ==================== UPDATE GLOBAL MODEL ====================
def set_weights(model, weights):
    for p, w in zip(model.parameters(), weights):
        p.data = w.clone()

# ==================== EVALUATION ====================
def evaluate(model, testloader):
    model.eval()
    correct, total = 0, 0
    with torch.no_grad():
        for inputs, labels in testloader:
            inputs, labels = inputs.to(device), labels.to(device)
            outputs = model(inputs)
            _, predicted = torch.max(outputs, 1)
            total += labels.size(0)
            correct += (predicted == labels).sum().item()
    return 100 * correct / total
```




```python
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

# ==================== MODEL ====================
class SimpleCNN(nn.Module):
    def __init__(self):
        super().__init__()
        self.conv1 = nn.Conv2d(3, 32, 3, padding=1)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(32, 64, 3, padding=1)
        self.fc1 = nn.Linear(64 * 8 * 8, 256)
        self.fc2 = nn.Linear(256, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = x.view(-1, 64 * 8 * 8)
        x = F.relu(self.fc1(x))
        return self.fc2(x)

# ==================== DP HELPER ====================
def add_dp_noise(tensor, epsilon=1.0):
    noise = torch.normal(0, 1/epsilon, size=tensor.size()).to(tensor.device)
    return tensor + noise

# ==================== HE HELPER ====================
def encrypt(tensor):
    return tensor * 1e3  # simulate large encrypted numbers
def decrypt(tensor):
    return tensor / 1e3
```