le cnam

Conservatoire national des arts et métiers

ÉCOLE DOCTORALE SCIENCES DES MÉTIERS DE L'INGÉNIEUR

Centre d'Études et De Recherche en Informatique et Communications (CEDRIC)

Thèse de doctorat

Présentée par : **Rezak AZIZ**

Soutenue le : **18 Décembre 2025**

Discipline : **Section CNU 27**

Spécialité : **Informatique**

Exploring Verifiable and Privacy-Preserving Federated Learning through Differential Privacy and Cryptographic Protocols

THÈSE dirigée par :

Mme. Samia BOUZEFRANE Professeure, Cnam

et co-encadrée par :

M. Youakim BADR Professeur, The Pennsylvania State University

M. Pierre PARADINAS Professeur, Cnam

Jury

M. Rida KHATOUN	Professeur, Télécom Paris	Rapporteur
M Josep Domingo Ferrer	Professeur, Universitat Rovira i Virgili	Rapporteur
M. Antoine BOUTET	Assistant Professor, INSA Lyon	Examinateur
M. Damien SAUVERON	Professeur, Université de Limoges	Examinateur
M. Samia BOUZEFRANE	Professeur, Cnam	Directrice de thèse
M. Youakim BADR	Professeur, The Pennsylvania State University	Co-encadrant de thèse
M. Pierre PARADINAS	Professeur, Cnam	Co-encadrant de thèse

À mes parents,

À celles et ceux qui m'ont accompagné tout au long de ce chemin.

Acknowledgements

Je tiens à exprimer ma profonde gratitude à ma directrice de thèse, Samia Saad-Bouzefrane, pour son accompagnement constant et indéfectible tout au long de ce parcours. Son soutien, à la fois scientifique et profondément humain, a largement dépassé le cadre académique. Par sa confiance, sa bienveillance et la justesse de ses conseils, elle a contribué de manière déterminante à mon développement personnel et professionnel, me donnant la force d'avancer aussi bien dans les moments de doute que dans les phases les plus exigeantes de ce travail. Elle n'a pas seulement été une mentor académique, mais une véritable mentor de vie, dont l'influence restera durablement ancrée bien au-delà de cette thèse.

Je souhaite également exprimer ma sincère reconnaissance à mon encadrant, Youakim Badr, dont l'arrivée dans ce travail a constitué une étape déterminante de cette thèse. Sa rigueur intellectuelle, la profondeur de ses analyses et la finesse de ses remarques ont permis d'éclairer des choix scientifiques structurants et d'orienter cette recherche avec cohérence et exigence. Son accompagnement a joué un rôle essentiel dans la maturation et l'aboutissement de ce travail.

Mes remerciements les plus chaleureux et respectueux s'adressent aux membres du jury de cette thèse. Je remercie tout particulièrement Damien Sauveron, président du jury, pour l'attention portée à ce travail ainsi que pour la qualité de la conduite des échanges lors de la soutenance. J'adresse également ma sincère gratitude aux rapporteurs, Josep Domingo-Ferrer et Rida Khatoun, pour le temps consacré à l'évaluation approfondie de cette thèse. Leurs analyses rigoureuses, leurs perspectives critiques et leurs remarques pertinentes ont constitué une contribution précieuse. Je remercie enfin Antoine Boutet pour la pertinence de ses questions, la qualité de ses remarques et la richesse des échanges scientifiques.

Je souhaite également remercier Soumya Banerjee pour sa collaboration et pour la qualité de nos échanges scientifiques. Son regard, ses remarques et sa disponibilité ont contribué à enrichir les

réflexions menées dans le cadre de ce travail.

Je tiens à adresser un remerciement particulier à Stefano Secci pour son accueil au sein de l'équipe Réseaux et Objets Connectés (ROC). Intégrer cette équipe a été une expérience particulièrement enrichissante, tant sur le plan scientifique qu'humain. Sa vision, son leadership et son exigence intellectuelle ont constitué des sources constantes d'inspiration et de motivation, offrant un cadre de travail stimulant, structurant et propice à des échanges scientifiques de qualité. J'adresse également un remerciement particulier à Pierre Paradinas. La qualité des échanges que nous avons partagés a toujours été enrichissante, et sa participation au jury de cette thèse constitue une reconnaissance importante du travail accompli.

Je souhaite exprimer ma gratitude à l'ensemble de mes collègues pour leur présence, leur bienveillance et le soutien qu'ils m'ont apporté tout au long de ce parcours. Nos échanges, notamment lors des pauses café, ont contribué à instaurer un environnement de travail à la fois stimulant et humain, jouant un rôle essentiel dans l'équilibre et le bon déroulement de ce travail de recherche.

Je souhaite également remercier le personnel administratif du Conservatoire national des arts et métiers pour sa disponibilité, son professionnalisme et son accompagnement tout au long de ce parcours. Leur soutien a permis de garantir un cadre administratif et institutionnel serein, indispensable au bon déroulement de cette thèse.

Enfin, je souhaite exprimer ma reconnaissance à toutes les personnes qui, de près ou de loin, ont contribué à l'aboutissement de ce travail. Qu'il s'agisse d'un soutien scientifique, technique, administratif ou humain, chacune de ces contributions a participé, à sa manière, à la réalisation de cette thèse.

Je souhaite enfin adresser mes remerciements les plus sincères à ma famille et à mes amis pour leur soutien indéfectible tout au long de ce parcours. Je remercie tout particulièrement mes parents pour leur présence constante, leur patience et leurs encouragements sans faille. Je remercie également ma sœur et mon frère pour leur bienveillance et leur soutien. Une pensée particulière va à mon oncle Youcef et à mon cousin Hamid pour leur aide, leurs encouragements et la qualité des échanges que nous partageons, lesquels ont constitué un soutien précieux.

Abstract

Federated Learning (FL) has emerged as a distributed paradigm enabling multiple participants to collaboratively train machine learning models without sharing their raw data. By keeping data local, FL mitigates many privacy risks inherent to centralized learning architectures. However, despite this promise, recent research has revealed that exchanged gradients can still leak sensitive information about local datasets. Furthermore, most existing approaches rely on strong and often unrealistic trust assumptions toward the central server, while providing no means to verify whether privacy-preserving mechanisms have been correctly enforced. These limitations expose a critical gap between theoretical privacy guarantees and their practical implementation in real-world federated systems.

This thesis investigates how to bridge this gap by combining differential privacy (DP) with cryptographic and verifiability protocols to achieve verifiable and trust-reduced federated learning. First, we explore the use of additive homomorphic encryption to protect client updates and minimize reliance on a trusted aggregator. Second, we introduce a non-interactive verifiability protocol based on zk-SNARKs and cryptographic hashes, allowing third parties to prove and verify the correct application of DP without revealing sensitive information. Finally, we propose ProoFed, a distributed framework that leverages secret sharing to decentralize noise generation and integrate verifiable aggregation proofs in zero knowledge, thereby eliminating single points of trust.

Keywords: Federated Learning, Differential Privacy, Verifiability, Cryptography Protocols, Privacy-Preserving Machine Learning, Trust Reduction.

Résumé

L'apprentissage fédéré est apparu comme un paradigme distribué permettant à plusieurs participants d'entraîner conjointement un modèle d'apprentissage automatique sans échanger leurs données brutes. En conservant les données localement, cette approche réduit considérablement les risques liés à la divulgation d'informations sensibles, risques souvent présents dans les architectures d'apprentissage centralisées. Toutefois, malgré cette promesse de confidentialité, plusieurs travaux récents ont démontré que les gradients échangés lors du processus d'apprentissage peuvent encore révéler des informations sur les jeux de données locaux. Par ailleurs, la majorité des approches actuelles reposent sur des hypothèses de confiance fortes envers le serveur central, sans offrir de mécanismes permettant de vérifier la bonne application des mesures de protection de la vie privée. Ces limites mettent en évidence un écart important entre les garanties théoriques de confidentialité et leur concrétisation dans les déploiements réels.

Dans cette thèse, nous proposons de réduire cet écart en combinant la confidentialité différentielle avec des outils cryptographiques et des protocoles de vérifiabilité. L'objectif est d'assurer un apprentissage fédéré à la fois vérifiable et moins dépendant de la confiance envers les entités centrales. Dans un premier temps, nous explorons l'utilisation du chiffrement homomorphe additif afin de protéger les mises à jour locales tout en limitant la dépendance à un agrégateur de confiance. Dans un second temps, nous introduisons un protocole de vérifiabilité non interactif fondé sur les zk-SNARKs et les fonctions de hachage cryptographiques. Ce protocole permet de prouver et de vérifier la correcte application de la confidentialité différentielle sans révéler d'informations sensibles. Enfin, nous présentons ProoFed, un cadre distribué reposant sur le partage de secret pour décentraliser la génération du bruit et intégrer des preuves d'agrégation vérifiables en connaissance nulle, supprimant ainsi les points uniques de confiance.

Mots-clés : Apprentissage fédéré, Confidentialité différentielle, Vérifiabilité, Protocoles cryptographiques, Apprentissage automatique préservant la vie privée, Réduction de la confiance.

Contents

Appendices

List of Tables

List of Figures

Part I

General Introduction

Chapter 1

Introduction

Contents

1.1 Context

Privacy has become a central concern in the deployment of machine learning systems. In healthcare, diagnostic models rely on sensitive medical data whose exposure may compromise patient confidentiality. In finance, fraud detection and credit scoring depend on transactional records subject to strict regulations. On social platforms, recommender systems and targeted advertising exploit digital browsing traces that are often collected without transparency. These examples illustrate, on one hand, the sensitive nature of the data collected to train modern learning systems. The collection of such data is particularly risky, as it can be misused for user profiling, discrimination, or even manipulation. On the other hand, the growth of connected devices and edge computing has transformed data generation and storage. Data is no longer centralized but distributed across multiple entities such as clinics, banks, companies, and personal devices operating under heterogeneous security and policy constraints. Formally, these constraints are translated into regulatory frameworks that aim to safeguard users against such threats and ensure the responsible use of personal data. The most well-known regulations are the following:

- The **General Data Protection Regulation (GDPR)**, adopted by the European Union in 2016, addresses the broad issue of personal data collection and processing across sectors. It imposes strict obligations such as explicit user consent, data minimization, clearly defined purposes, rights of access, modification, portability, and erasure, as well as accountability of data controllers.

- The **California Consumer Privacy Act (CCPA)**, adopted in 2018, focuses on protecting consumer data in the digital economy, particularly for residents of California. It grants individuals the right to know which data are collected, to request their deletion, to opt out of data sales, and to be free from discrimination when exercising these rights. Compared to the GDPR, it has a narrower scope and emphasizes transparency and consumer control.

- The **Health Insurance Portability and Accountability Act (HIPAA)**, enacted in the United States in 1996, regulates the use and disclosure of protected health information by healthcare providers, insurers, and their business associates. It establishes national standards for data privacy and security, granting patients rights over their medical records, including access, correction, and notice of data breaches. Unlike the GDPR or CCPA, which apply broadly across sectors, HIPAA

is limited to the healthcare domain and focuses on safeguarding medical information rather than general personal data.

The growing emphasis on data protection has encouraged interest in privacy-preserving machine learning paradigms. Among them, Federated Learning (FL) has emerged as a promising solution. Introduced by McMahan et al. in 2016[1], it enables the training of a global machine learning model in a decentralized manner. In this paradigm, data remain on local devices and are never shared with third parties; instead, only model updates are communicated to a central server. At first sight, this paradigm seems to reduce the exposure of sensitive data and to comply with regulatory requirements.

However, recent studies have shown that even without sharing raw data, machine learning models can still leak sensitive information through the parameters or gradients they exchange. For instance, a recent work [2] demonstrated that large language models trained on web-scale corpora can memorize and reproduce verbatim segments of their training data, including personal information, source code, and copyrighted text. Similarly, Microsoft's Bing Chat powered by GPT-4 was tricked through a prompt injection attack into revealing its hidden system instructions, which included confidential operational rules and internal identifiers[1].

These cases are not the focus of this work, but they illustrate the growing risks of information leakage in modern machine learning systems, both during training and at deployment. In Federated Learning, this issue becomes even more pronounced due to its iterative nature and the white-box setting, where the exchanged gradients and model updates can be fully inspected by an adversary. This makes FL particularly vulnerable to inference attacks as detailed in Chapter 2. Consequently, FL provides only a first layer of protection. Many techniques to preserve privacy have been proposed in the domain of Privacy-Enhancing Technologies (PETs), but each technique comes with technical constraints and drawbacks, creating trade-offs that make their use in real-world scenarios difficult.

Therefore, despite its promise, FL and naïve PETs have not managed to provide a complete solution to the requirements imposed by regulations. Regulations are not only about restricting the exchanged data, but also about enforcing strict and auditable mechanisms to prove that privacy is truly preserved. This gap between the promises of the technology and the demands of the regulations is the main motivation of this thesis. In this context, three primary challenges need to be tackled:

[1] https://arstechnica.com/information-technology/2023/02/ai-powered-bing-chat-spills-its-secrets-via-prompt-inje

- **Information leakage from gradients**: exchanged updates may reveal sensitive information, making FL vulnerable to inference attacks and showing the limits of the "privacy by design" principle.
- **Central server dependency**: many solutions assume threat models that are not realistic in some domains, such as the presence of an honest or semi-honest server, while in practice a compromised server is a much more likely scenario.
- **Lack of auditability and verifiability**: participants and regulators have no means to verify the results and are forced to blindly trust the service provider.

1.2 Scope of the thesis

This thesis is conducted in the context of Federated Learning and its inherent limitations. The guiding idea behind this work originates from a central observation: among existing privacy-preserving mechanisms, Differential Privacy (DP) offers the only formal guarantee that remains valid even after the release of the global model. However, when applied in isolation, DP creates a trade-off between privacy and utility. Mitigating this trade-off requires an excessive level of trust in the central server responsible for enforcing differential privacy. Moreover, there currently exists no verifiable means to ensure that the claimed privacy mechanisms have been properly applied.

This thesis therefore explores how Differential Privacy can be combined with cryptographic primitives and verifiability protocols to design federated learning systems that are both privacy-preserving and trust-reduced. The scope is deliberately limited to the intersection of differential privacy, cryptographic techniques, and verifiability protocols. The objective is not to provide an exhaustive review of all privacy-preserving solutions, but rather to investigate combined approaches capable of addressing the three key issues identified in the previous section. The rationale behind these choices will be detailed in the following chapters. To make this goal concrete, the scope of the thesis is structured around three main dimensions:

- State of the art analysis of differential privacy and cryptographic methods applied to FL, emphasizing their benefits, constraints, and trade-offs.
- Design and development of combined frameworks that mitigate the drawbacks of DP by integrating cryptographic and verifiable mechanisms, aiming to achieve an equilibrium between privacy,

trust, and computational efficiency.

- Experimental evaluation on multiple datasets to assess accuracy, resource overhead, scalability, and applicability in realistic scenarios.

At the same time, some important aspects remain deliberately out of scope. This thesis does not focus on system-level issues of federated learning such as heterogeneity, client drop-out, or network optimization. The term federated learning here refers only to the classical setting with a central server. Some recent works extend the term to decentralized peer-to-peer schemes, but these are closer to privacy-preserving distributed learning than to classical FL. Other technologies such as trusted execution environments (TEEs) or blockchain are considered as complementary to the perspective of this thesis. Finally, attacks unrelated to privacy, such as poisoning or backdoors are not covered, however, the contributions should be extended to mitigate them.

1.3 Contributions

This thesis investigates how DP can be effectively combined with cryptographic techniques and verifiability mechanisms to achieve verifiable and trust reduced FL. The contributions are both conceptual and practical, encompassing theoretical analysis, protocol design, and experimental validation. They are organized as follows:

1. We do a State of the art of existing approaches that combine Differential Privacy and cryptographic primitives in the FL context. This review serves as the foundation for defining the design goals and architectural principles of the proposed frameworks.

2. We propose a framework that introduces a secure tunnel based on additive homomorphic encryption, protecting client updates throughout both transmission and aggregation. This tunnel enables the server to perform model aggregation without accessing the underlying local updates, thereby decoupling noise addition from aggregation and significantly reducing the trust required in the central server. This decoupling ensures stronger confidentiality and anonymity, while also enabling privacy amplification through both shuffling and subsampling, without imposing restrictive conditions on the privacy budget or the number of participating clients.

3. We propose a framework that introduces a non-interactive verifiability protocol based on zk-SNARKs and cryptographic hash commitments to prove and verify the correct application of central differential privacy. This protocol is designed to address two main challenges: first, managing the inherent randomness introduced by differential privacy, and second, ensuring that no additional information is disclosed, as both the raw data and the added noise must remain hidden. By integrating these zero-knowledge proofs into the differential privacy, the framework enables transparent verification of privacy preservation without compromising privacy, thereby reducing the need for blind trust in the central server and reinforcing the integrity and accountability of the entire training process.

4. We propose ProoFed, a comprehensive framework that combines differential privacy, secret sharing, and verifiable protocols to achieve privacy-preserving and trust-reduced federated learning. In this framework, noise generation is distributed among the clients using secret sharing, thereby eliminating the need to trust the central server for privacy enforcement. A dedicated verification protocol ensures that the final aggregation is performed correctly and can be audited without revealing sensitive information. By unifying privacy protection, decentralization, and verifiability within a single architecture, ProoFed strengthens both the security and transparency of the federated learning process while maintaining practical efficiency and scalability.

1.4 Thesis Outline

The manuscript is organized as follows:

- Chapter 2 introduces federated learning, presents its main categories and frameworks, and reviews the challenges related, including a focus on privacy and trust challenges by exploring the main types of attacks.

- Chapter 3 focuses on differential privacy as a formal guarantee to protect the global model in Federated Learning. We review the different ways DP can be incorporated into FL, analyze the limitations of DP-only approaches as a motivation for combined techniques, and present the state of the art together with the open challenges.

- Chapter 4 develops the first contribution for reducing trust in the federated learning process

by introducing additive homomorphic encryption. It delve into the motivations behind this contribution before detailing the design of the solution and the evaluation of the system.

- Chapter 5 presents the second contribution, which addresses the challenge of verifiability in central differential privacy. It details the design of a non-interactive protocol that proves and verifies the correct application of differential privacy using zk-SNARKs and cryptographic hash commitments, and discusses its validation through analysis and experimentation.

- Chapter 8 develops the third contribution with the design and implementation of ProoFed, a unified framework combining differential privacy, secret sharing, and verifiable aggregation protocols. It describes the overall architecture, explains how trust is distributed among clients, and evaluates the framework on multiple datasets to assess scalability and robustness.

- Chapter 7 concludes the manuscript by summarizing the main results of this thesis, discussing its limitations, and highlighting perspectives for future research.

1.5 Other Contributions

During this thesis, I have made additional contributions that are not included in this manuscript.

Supervision of a Master and Engineering student. I supervised Maya Ben Abdellatif in her master thesis entitled “Privacy-Preserving Meta-Learning using Differential Privacy”. The project explored the integration of Differential Privacy into meta-learning algorithms, with the objective of preserving privacy while maintaining the adaptability of models across heterogeneous tasks.

Scientific dissemination through poster presentation. I presented a poster based on a prior work entitled ”Secure and non-interactive k-NN classifier using symmetric fully homomorphic encryption” during the HCERES evaluation visit at the CEDRIC laboratory.

Teaching activities. I prepared and delivered two courses within the Licence Professionnelle en Informatique – Web, Mobile et Business Intelligence at the Cnam Paris. The first course focused on Algorithmics and Programming, while the second addressed Web Development. In addition, I contributed to teaching activities through the delivery of practical sessions in NEVA and IoT.

1.6 List of Publications

The work presented in this thesis has resulted in the following publications.

International Conferences

- Aziz, Rezak, Youakim Badr, and Samia Bouzefrane. "Enhancing Trust in Central Differential Privacy Using zk-SNARKs and Cryptographic Hashes." International Conference on Advanced Information Networking and Applications. Cham: Springer Nature Switzerland, 2025.

  Code: `https://github.com/rezakaziz/zkSnarkDP`
  Contribution: Main contribution by the author of this thesis.

- Aziz, Rezak, Soumya Banerjee, and Samia Bouzefrane. "Privacy Preserving Federated Learning: A Novel Approach for Combining Differential Privacy and Homomorphic Encryption." IFIP International Conference on Information Security Theory and Practice. Cham: Springer Nature Switzerland, 2024.

  Code: `https://github.com/rezakaziz/PrivatumFL`
  Contribution: Main contribution by the author of this thesis.

Journals

- Aziz, Rezak, Soumya Banerjee, Samia Bouzefrane, and Thinh Le Vinh. "Exploring homomorphic encryption and differential privacy techniques towards secure federated learning paradigm." Future internet 15, no. 9 (2023): 310.

  Status: Published in 2023
  Contribution: Main contribution by the author of this thesis.

- Aziz, Rezak, Youakim Badr, Soumya Banerjee and Samia Bouzefrane. "ProoFed: A Distributed Differential Privacy framework for Federated Learning based on Secret Additive Sharing and Verifiable Protocols" Journal of Information Security and Applications

  Status: Submitted
  Contribution: Main contribution by the author of this thesis.

Part II

Background and State Of The Art

Chapter 2

Federated Learning, Privacy and Trust Issues

Contents

2.1 Introduction

This chapter provides the background for understanding the mechanisms and limitations of FL. The goal is not to restate the motivations already discussed, but to establish a precise view of how FL operates and where its vulnerabilities arise. We begin by recalling the basic principles of machine and deep learning. We then introduce the federated learning paradigm, its core architecture, and its main categories as defined in the literature. From there, we examine the challenges that stem from its distributed nature before focusing on privacy and trust issues. These issues are not treated abstractly but through concrete examples of attacks and system assumptions that expose the weaknesses of current FL deployments. This analysis serves as a foundation for the next chapter, where privacy-enhancing and trust-reducing mechanisms are explored in depth.

2.2 Background on Machine Learning and Deep Learning

Machine Learning deals with the design of algorithms that can learn patterns from data without being explicitly programmed. Formally, given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$, drawn from an unknown distribution $\mathcal{P}(x, y)$, the goal is to find a function $f_w : \mathcal{X} \to \mathcal{Y}$ parameterized by w that minimizes the expected risk:

$$\mathcal{R}(w) = \mathbb{E}_{(x,y)\sim\mathcal{P}}\big[L(f_w(x), y)\big],$$

where L is a loss function that measures the discrepancy between predictions and true labels. Since the distribution $\mathcal{P}$ is unknown, in practice the goal is to minimize a global objective function over a dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$. For a model parameterized by w, the optimization problem is:

$$\min_{w} F(w) = \frac{1}{N}\sum_{i=1}^{N} L(f_w(x_i), y_i) \tag{2.1}$$

Machine learning can be broadly categorized into three paradigms. Supervised learning relies on labeled datasets to learn predictive mappings, typically for classification or regression; all the methods in this thesis are studied under this paradigm, although they can be extended to others. Unsupervised learning instead focuses on uncovering hidden structures in unlabeled data, through techniques such as clustering or dimensionality reduction. Reinforcement learning follows a different approach, where

an agent interacts with its environment and learns a strategy that maximizes long-term rewards, with applications ranging from robotics to autonomous driving and resource management.

Traditional machine learning models such as Logistic Regression, Support Vector Machine (SVM), and Decision Trees are efficient and interpretable, but they are outside the scope of this thesis. In addition to their limited ability to handle unstructured data such as images or text, they lack the expressive power to capture complex high-dimensional patterns. They are also less representative of current benchmarks in FL, and are not the main target of recent privacy or trust attacks. Moreover, several studies [3, 4, 5] show that model capacity directly influences learning behavior: models with limited capacity tend to extract structural patterns, whereas highly parameterized models are more likely to memorize training data. For example, small Transformers generalize without memorizing, while larger ones sacrifice generalization for memorization; similarly, GPT models can memorize up to 3.6 bits per parameter before saturating and shifting back towards generalization. For these reasons, this thesis focuses on deep learning models, which offer richer feature representations and have become the de facto standard in most FL applications.

The optimization of the equation 2.2 is typically carried out using Gradient Descent (GD), which finds a minimum by moving each step in the direction of the steepest descent, as given by the gradient. From the initial point w_0, the sequence is defined by the following iterations

$$w_{t+1} = w_t - \eta \nabla_w L(f_{w_t}(x_j), y_j),$$

where (x_j, y_j) is a minibatch, ∇_w denotes the gradient with respect to the parameters, and $\eta > 0$ is the learning rate.

In practice, one rarely computes gradients on the full dataset at each step due to the prohibitive computational cost. Instead, Stochastic Gradient Descent (SGD) and its minibatch variant are employed, where the gradient is approximated using only a subset of the training data. This introduces stochasticity in the updates, which helps escape shallow local minima and often accelerates convergence in high-dimensional spaces.

Beyond SGD, more advanced optimizers have been developed to adaptively tune learning rates and incorporate momentum. A widely used method is Adam (Adaptive Moment Estimation)[6], which maintains exponentially decaying moving averages of past gradients and their squared values. The

Adam update rule can be written as following:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_w L(f_{w_t}(x_j), y_j), \quad v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla_w L(f_{w_t}(x_j), y_j))^2,$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}, \quad w_{t+1} = w_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},$$

where $\beta_1, \beta_2 \in (0,1)$ are decay rates and $\epsilon > 0$ is a small constant ensuring numerical stability.

To sum up, deep learning comprises a wide range of architectures adapted to different data modalities and tasks. Among the most prominent are multilayer perceptrons (MLPs), Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) with their extensions such as Long Short-Term Memory (LSTM), as well as more recent families such as Transformers and Graph Neural Networks (GNNs). While this thesis does not focus on neural architectures themselves, we illustrate our proposed techniques using a subset of representative models.

2.3 Federated Learning

Federated learning (FL) is the core concept on which this thesis is built. In this section, we recall its formal foundations, present its main categories, and highlight the challenges it raises, particularly regarding privacy and trust. We also review the main open-source frameworks that support research and deployment, which serve as the practical basis for our experimental evaluation.

2.3.1 Foundations

The term federated learning was introduced by McMahan et al.[1] in 2017. The objective is to enable collaborative training of a machine learning model while keeping the data on local devices. In this paradigm, raw data never leave the client side; instead, only model updates required for aggregation are transmitted to the central server. The overall process is illustrated in Figure 2.1.

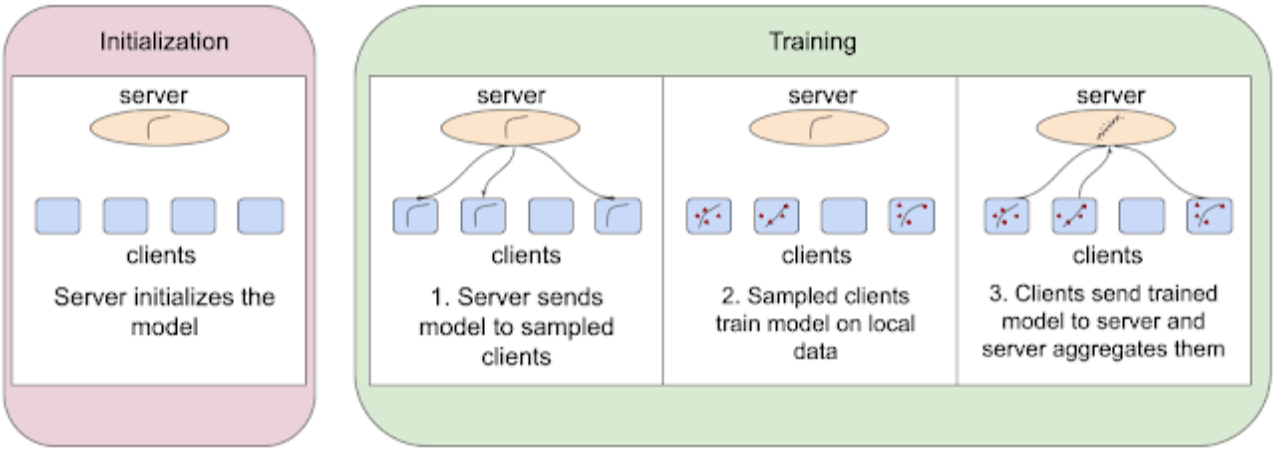


Figure 2.1: Federated learning process[7]

In classical machine learning, the goal is to minimize a global objective function over a dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$. For a model parameterized by w, the optimization problem is:

$$\min_{w} F(w) = \frac{1}{N} \sum_{i=1}^{N} l(w; x_i, y_i)$$

where l is the loss function.

In FL, the dataset is distributed among K participants. Each participant k holds a local dataset D_k of size n_k, with $\sum_1^N n_k = N$. The global objective becomes :

$$\min_{w} F(w) = \sum_{i=1}^{K} \frac{n_k}{N} F_k(w) \tag{2.2}$$

were the local objective of the client k is defined as:

$$\min_{w} F_k(w) = \frac{1}{n_k} \sum_{i=1}^{n_k} l(w; x_i^{(k)}, y_i^{(k)})$$

The key step is the global aggregation, where the server constructs the global model from client updates. To solve the global optimization problem defined in Equation 2.2, McMahan et al. [1] introduced **Federated Averaging (FedAvg)**, the most widely used aggregation algorithm. In FedAvg, the server sends the global model to selected clients, each client trains it locally, and the server aggregates the returned models as a weighted average proportional to the dataset size of each client.

Formally, the update at round t is:

$$w_{t+1} = \sum_{j=1}^{N} \frac{n_j}{\sum_{k=1}^{N} n_k} w_j^t \tag{2.3}$$

where w_j^t is the local model of client j at round t, and n_j is the number of samples held by client j.

FedAvg became popular thanks to its simplicity and communication efficiency. However, its performance in real-world settings is limited by the challenges outlined in Section 2.3.3, mainly heterogeneity, privacy, robustness, and fairness. Nanayakkara et al. [8] reviewed in 2024 several alternatives designed to address these shortcomings.

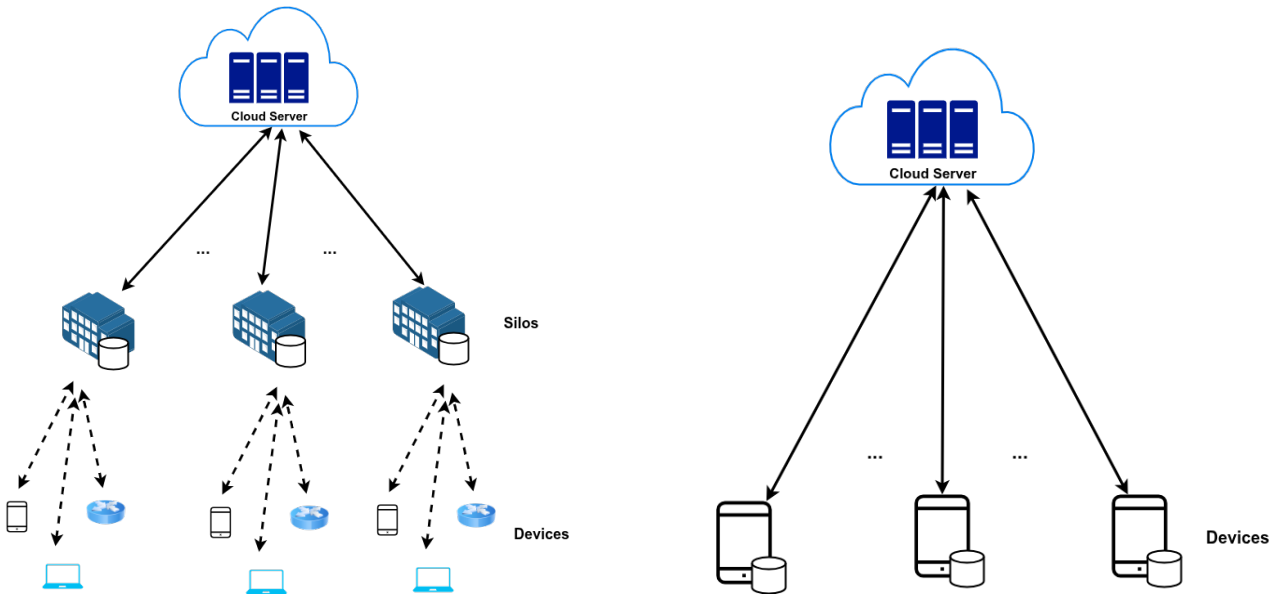
In this work, we still use FedAvg as a baseline. We only address privacy and trust as challenges, and we build our methods on top of FedAvg. Nevertheless, the proposed approaches can be extended to other aggregation mechanisms.

2.3.2 Categorization of Federated Learning

Many categorization can be found in the literature for FL according to different criterea. We categorize FL along the two main dimensions commonly adopted in the literature: by data partition and by scale and client type. Other distinctions such as communication and system constraints, privacy and robustness mechanisms, or learning objectives are considered as complementary aspects rather than primary categories.

1. **By Scale and Client Type.** Two main settings are commonly discussed in the literature[9]: cross-device and cross-silo. Figure 2.2 illustrates the differences between them. Cross-device typically involves mobile phones and IoT devices, whereas cross-silo involves organizations such as hospitals, banks, or enterprises. In the cross-silo setting, the number of clients is relatively small, but each has substantial computational capacity. By contrast, cross-device FL involves a very large number of clients, each with limited resources. Another key distinction is reliability: organizations in the cross-silo setting are generally always available for training, unlike user devices, which may frequently be offline. The methods developed in this thesis are applicable to both settings.

2. **By data partition.** Yang et al.[10] proposed a categorization by data partition: Horizontal FL and Vertical FL. Figure 2.3 shows the difference between the two settings. In Horizontal FL,



(a) Cross-silo Federated Learning (b) Cross-device Federated Learning

Figure 2.2: Comparison of cross-silo and cross-device federated learning architectures.

the datasets of the clients have the same features space. In Vertical FL, the local datasets have the same individuals, but with different features. In this work, we restrict our scope to the horizontal FL setting.

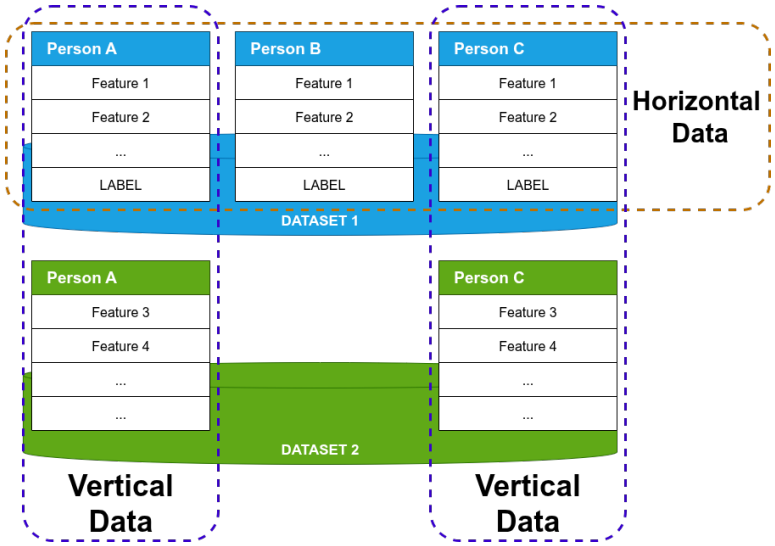


Figure 2.3: Comparison of horizontal and vertical data in federated learning architectures.

2.3.3 Challenges

While FL offers a promising paradigm for training models collaboratively without centralizing data, it also introduces a series of challenges that are not present in traditional centralized learning. In what follows, we provide the main challenges in FL:

1. **Data Related:** Unlike centralized learning, where data is assumed to be independent and identically distributed (IID) and preprocessed to reasonably satisfy such assumption, FL rarely operates under such conditions. In practice, client data is inherently non-IID, reflecting differences in user behavior, acquisition devices, or environments. This leads to imbalances in dataset sizes, skewed label distributions, and evolving patterns due to concept drift. Moreover, noisy or low-quality data further complicates the learning process. These characteristics make training in FL fundamentally more challenging than in centralized settings, as the global model must integrate heterogeneous and unstable updates without direct access to the raw data.

2. **System related:** FL also faces significant system-related challenges due to the heterogeneity of participating devices and networks. Clients often differ in terms of computational power, memory, and energy availability, ranging from powerful servers to resource-constrained mobile or IoT devices. Network conditions are also highly variable, with limited bandwidth, intermittent connectivity, and the presence of stragglers that slow down synchronous training rounds. Furthermore, client availability is unpredictable, as devices may join or leave the training process at any time. These factors introduce instability in training, reduce efficiency, and complicate the design of robust and scalable FL systems.

3. **Robustness:** as the decentralized setting opens new attack surfaces compared to centralized learning. Malicious clients may attempt to corrupt the training process through poisoning attacks, either by manipulating their local data (data poisoning) or by altering model updates (model poisoning), leading to degraded or backdoored global models. Adversaries can also exploit the aggregation process through Byzantine behavior, sending arbitrary or inconsistent updates that destabilize training. Beyond poisoning, backdoor attacks pose a serious threat, where models behave normally on standard inputs but are deliberately triggered to misclassify specific patterns. These risks highlight the need for robust aggregation, anomaly detection, and verification mechanisms to ensure the integrity and reliability of federated training.

4. **Privacy:** Despite keeping raw data on local devices, FL still faces significant privacy-related challenges. Model updates exchanged during training can unintentionally leak sensitive information about client data. Adversaries may exploit this through inference attacks, such as membership inference to determine whether a particular sample was used in training, or gradient inversion to reconstruct private inputs from shared updates.

5. **Trust and Fairness:** Another important challenge in FL is the issue of trust between participants and the central server. Since model updates are exchanged without direct access to the underlying data, clients must trust that the server performs aggregation honestly and does not misuse or inspect their contributions. Conversely, the server must also trust that clients provide genuine updates and do not attempt to manipulate the global model through malicious behavior. This lack of verifiability raises concerns about accountability and fairness, particularly in cross-organizational settings where participants may not fully trust one another.

This thesis does not address the full spectrum of challenges. Instead, it focuses on privacy and trust. The remaining challenges, such as data heterogeneity, system variability, communication efficiency, and optimization, are acknowledged but fall outside the scope of this work. Privacy and trust are examined in detail in Chapter 3.

2.3.4 Frameworks

The development of FL has given rise to several frameworks designed to facilitate research, experimentation, and large-scale deployment. This study is not exhaustive and focuses only on open-source frameworks. To narrow down the selection, we applied two criteria: the framework must have been actively updated since 2022, and it must have gathered more than 1000 stars on GitHub. The considered frameworks are detailed in the Table 2.1.

For the purpose of this study, we just summarize the capabilities of these frameworks in the following. The goal is to select a framework that is highly customizable with good documentation and that integrates seemlessly with various ML frameworks like pytorch, tensorflow and scikit learn.

1. **Federated AI Technology Enabler (FATE)[11, 12]**: is an open-source framework initiated by Webank's AI Department to support the federated AI ecosystem. FATE is built on a library, called FederatedML. The main components are used to interface with the users : FATE-Board

Table 2.1: Main open-source Federated Learning frameworks (information gathered in Sep 8th, 2025)

Framework	Initial Release	Latest Release	# Releases	Stars	Country
FATE	Jan 18, 2019	Jul 31, 2024	50	6k	China (WeBank)
TFF	Feb 20, 2019	Sep 26, 2024	91	2.4k	USA (Google)
OpenFL	Feb 1, 2021	Dec 19,2024	10	2k	USA (Intel)
PySyft	Jan 19, 2020	Jul 31, 2024	177	9.8k	USA(OpenMined)
Flower	Nov 11, 2020	Jul 29, 2025	35	6.2k	Germany (ADAP)

for visualization and monitoring, FATE-Flow for workflow management and scheduling, FATE-Serving, an industrialized serving system for FL models, designed for production environments. FATE also implements secure computation protocols such as homomorphic encryption and multi-party computation (MPC). It supports a variety of FL architectures and enables training of different machine learning models, including logistic regression, tree-based methods, and deep learning.

2. **Tensorflow Federated(TFF)[13, 14]** is an open-source framework developed by Google. It provides a server–client architecture where a central server coordinates the learning process while clients perform local training on their private data. Communication is handled through gRPC, ensuring efficiency and security. TFF is lightweight to install and mainly intended for research and experimentation.

3. **OpenFL[15]** : is a flexible software platform developed for FL, originally created through a collaboration between Intel Labs and the University of Pennsylvania. The architecture is designed to be flexible and extensible, enabling it to integrate with various machine-learning frameworks such as TensorFlow and PyTorch. The architecture follows a client–server paradigm, where a central aggregator orchestrates the training process and remote collaborators, called Collaborators, perform local computations on their private datasets. Communication between the aggregator and collaborators is secured through encrypted channels.

4. **PySyft[16, 17]** : is an open-source framework for privacy-preserving data analysis developed by the OpenMined community. It is tightly integrated with PyTorch and incorporates advanced privacy-preserving techniques such as differential privacy and secure multi-party computation. PySyft follows the principles of Remote Data Science, where code is sent to the data rather

than transferring data to the code. Its main component is the Datasite, a set of servers deployed by data owners to provide controlled and responsible access to their assets. PySyft relies on a client–server architecture in which the client, implemented as a Python interface, enables both data owners and data scientists to interact securely with the Datasite.

5. **Flower[18, 19]** is an end-to-end FL framework designed to bridge experimental research and real-world deployment. It provides a flexible and modular architecture with seamless integration into major machine-learning frameworks such as TensorFlow, PyTorch, and Keras. The architecture is organized into two main components : the server, which combines a SuperLink for managing communication and a ServerApp for customizing server behavior; and the client, which consists of a Dupernode that connects to the SuperLink and a ClientApp that defines the client-side logic of FL. This architecture makes Flower highly customizable and easy to use.

For the experiments conducted in this thesis, we retained Flower as the underlying framework. This choice is motivated by its modularity and its seamless integration with different ML Libraries.

2.4 Privacy and Trust Challenges in FL

FL is often presented as a privacy-preserving paradigm, but recent studies have shown that it introduces specific concerns regarding privacy and trust. On the privacy side, although raw datasets remain local, the gradients exchanged during training may still reveal sensitive information about client data. This risk is amplified by the large number of participants and the white-box nature of FL, where insiders can passively exploit gradients without interfering with training. On the trust side, many works assume a honest-but-curious server, whereas in practice a compromised server is a realistic threat. Similarly, clients are often assumed to behave honestly, yet malicious participants may poison updates, insert backdoors, or collude to bias the global model. These assumptions are rarely valid in sensitive domains such as healthcare or finance. As a result, the attack surface of FL is significantly broader than often acknowledged. An overview of these threats at different levels of the FL architecture is shown in Figure 2.4.

In the following, we present the main attack types, with their objectives and the techniques used summarized in Table 2.2. These attacks highlight the limitations of FL in providing privacy by design and motivate the need for additional mechanisms. In this thesis, we specifically address these threats

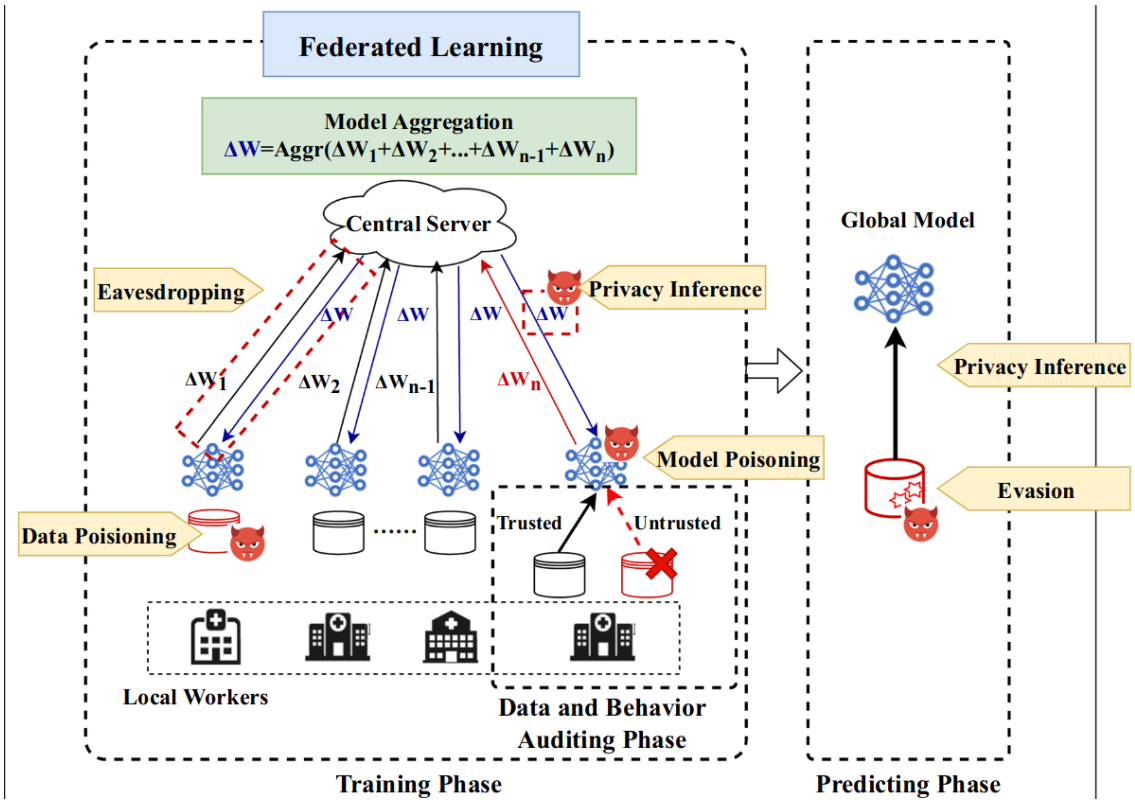


Figure 2.4: Attacks on the federated learning process [20].

through the integration of differential privacy, cryptographic protocols, and verifiability mechanisms, which are developed and evaluated in the subsequent chapters.

2.4.1 Membership Inference

Membership inference is a type of attack in machine learning that aims to figure out whether a target data point is used to train a certain ML model. More formally, given x as the target point, M as a trained model, and some external knowledge K, this attack can be defined by the following function

$$A : x, M, K \rightarrow \{False, True\}$$

Here, this function returns $True$ if the target x is in the training dataset and $False$ otherwise. This attack can be made either in a black-box setting, where the attacker has only access to an API of the model M, or in a white-box setting, where the attacker has access to the whole model.

As we can notice, the attack model A is a binary classifier, and it can be constructed using

Table 2.2: Main privacy attacks against federated learning, their targets, and common techniques.

Attack Type	**Information Leaked**	**Possible Techniques**	**Key References**
Membership inference	Presence of specific samples in the training set	Shadow models, prediction confidence analysis, gradient distribution analysis	[21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33]
Class representative inference	Generic prototypes of training classes	GAN-based generation	[34, 35]
Property inference	Statistical attributes (e.g., gender, ethnicity)	Shadow models, Meta-classifiers	[36, 37, 38, 39, 40]
Reconstruction	Raw training data or labels reconstructed from gradients	Gradient inversion optimization	[41, 42, 43, 44, 45, 46, 47, 48, 49]

different ways. These ways can be categorized into **update-based** and **trend-based** approaches [50].

Update-based attacks exploit gradients, parameters, or model snapshots exchanged during training. Nasr et al. [21] showed that the distribution of gradients for members can be distinguished from that of non-members, and Gupta et al. [22] extended this result to regression tasks. Li et al. [23] proposed two attacks that exploit differences between gradients in consecutive rounds. These works rely on the white-box setting of FL. Single-model approaches also exist. They are often based on shadow training, where a discriminator is trained to distinguish between models trained with and without the target victim's participation [24, 25, 26, 27]. Yuan et al. [51] extended this idea to federated recommender systems. Structure-modifying attacks represent a more active direction: Pichler et al. [31] exploited ReLU activation properties to tag member records.

Trend-based attacks, in contrast, analyze the temporal evolution of outputs, losses, or parameters during training. Output trajectories (confidence, entropy) have been exploited by Gu et al. [29], while loss trajectories were studied by Hu et al. [30] and Suri et al. [32]. Finally, Zhang et al. [52] demonstrated that bias trajectories in the last layer provide an effective signal for distinguishing members from non-members.

2.4.2 Class Representatives Inference

Class representatives inference aims to extract generic class-level representatives of the training data rather than reconstructing the exact records [35]. More formally, given a trained model M, the adversary attempts to generate synthetic samples $\hat{x}_c$ that capture the overall distribution of a

class c. In contrast to membership inference, which is a binary decision problem, this attack outputs prototypes or representative samples for each class.

This attack is closely related to model inversion [53], where the adversary leverages the model to reconstruct inputs that maximize confidence for a given label. When the members of a class are highly similar, the reconstructed representative may closely resemble the true training data; for example, in a facial recognition system, each class corresponds to a single individual, and the inferred representative can approximate an actual image of that person. In the context of FL, Generative Adversarial Networks (GANs) have been used to implement such attacks. Hitaj et al. [34] proposed a GAN-based strategy in which the attacker, acting as a honest-but-curious client, locally trains a GAN to generate samples that mimic the victim's data distribution while being labeled as belonging to another class. This manipulation forces the victim to refine its model in order to distinguish between real and fake samples, thereby leaking additional information about its private data. Experimental results confirmed that this approach can effectively produce representative samples that are visually close to the victim's training data. Building on this idea, Wang et al. [35] introduced mGAN-AI, a more practical and inconspicuous attack where the adversary is the server rather than a client. Unlike the client-based setting of Hitaj et al., mGAN-AI employs a multitask discriminator that not only fulfills the role of a standard GAN discriminator but also distinguishes the distribution of a victim client from those of other participants, enabling the server to reconstruct class representatives tied to specific clients and thereby breaching client-level privacy.

2.4.3 Properties Inference

The properties inference attack, first introduced by Ateniese et al. in 2013 [54], aims to extract private statistical information about the training set that is not expected to be shared and may be irrelevant to the main task. This attack mostly is based on shadow training to construct a meta-classifier capable of determining whether a property P (e.g., the ethnicity of training participants) is present in the data. Their experiments focused on centralized machine-learning models, specifically Support Vector Machines (SVMs) and Hidden Markov Models (HMMs).

In 2018, Melis et al. [36] later extended this attack to collaborative and FL. They showed that the model updates exchanged during training can inadvertently reveal sensitive properties of participants' data. By exploiting leakage from sparse embedding layers or from gradients, an adversary participating

in FL can mount both passive and active property inference attacks to recover attributes unrelated to the model's primary task. In the same year, Ganju et al. [37] studied property inference against Fully Connected Neural Networks (FCNNs) in a white-box setting. Their objective was to infer global dataset properties, such as the data collection environment or class proportions. Unlike the meta-classifier approach, which is impractical for FCNNs due to permutation symmetry across hidden layers, they introduced neuron sorting, significantly improving the attack's effectiveness. Their results underscored both the difficulty of attacking FCNNs and the success of their proposed strategies. In 2021, Zhou et al. [38] investigated generative models, in particular generative adversarial networks (GANs). They proposed a general attack pipeline demonstrating that property inference is feasible not only against discriminative models but also against generative models. Recent work tries to investigate way to extend the solutions to other topologies like federated GNN[39] and to vertical FL[39].

2.4.4 Training Samples and Labels Inference

Training samples and labels inference, also called reconstruction attacks, aim to recover the original data and their labels from the gradients or parameters exchanged during FL. By exploiting this information, adversaries attempt to recreate sensitive samples belonging to the clients, which poses a direct threat to privacy.

Among the first work in this direction, Zhu et al. [41] introduced the Deep Leakage from Gradients (DLG) to reconstruct inputs by matching gradients. Since then, several improvements have been proposed. Zhao et al. [42] showed how to extract the correct labels, while Geiping et al. [43] refined the optimization objective to scale towards more realistic architectures. Yin et al. [45] extended the attack to recover multiple samples within a batch, and Ren et al. [44] used generative models to achieve reconstructions at higher resolution and larger batch sizes.

In the FL context, reconstruction attacks continue to evolve. LOKI [46] and AWA [47] demonstrated effective recovery even with secure aggregation and multi-step local updates. FET [48] showed severe vulnerabilities for text data. Finally, Valadi et al. [49] emphasized that gradient inversion remains feasible on production grade models, depending on the architecture.

These works can be broadly categorized into **optimization-based** approaches, which iteratively adjust dummy inputs to match gradients, and **generative-model-based** approaches, which employ GANs or regression networks to synthesize data. Together, they underline the severity of reconstruction

attacks, as they can recover sensitive samples with high fidelity under increasingly realistic FL settings.

2.5 Conclusion

In this chapter, we first recalled the fundamental concepts of machine learning and deep learning, before introducing FL as the core paradigm on which this thesis builds. We described its main categories, frameworks, and challenges, with a particular focus on privacy and trust.

We then examined the main privacy threats in FL, including membership inference, class representative inference, property inference, and reconstruction attacks. These attacks demonstrate that, despite its promise, FL does not provide privacy guarantees by design and remains vulnerable to information leakage through gradients. In parallel, unrealistic trust assumptions on both the server and the clients further broaden the attack surface and undermine the reliability of the framework.

These findings underline the necessity of integrating additional mechanisms to ensure both privacy and verifiability in FL. This motivates the solutions explored in this thesis, which combine differential privacy, cryptographic protocols, and verifiability mechanisms to strengthen privacy protection and reduce trust assumptions. Chapter 3 develops this direction by reviewing existing countermeasures and setting the ground for our proposed approaches.

Key takeaways

- Federated Learning does not guarantee privacy by design.
- Gradients remain the main vector of information leakage.
- Trust assumptions on server and clients are often unrealistic.
- These challenges motivate the integration of Differential Privacy and Cryptographic Approaches which are the focus of this thesis.

Chapter 3

Privacy and Trust Enhancement in Federated Learning

Contents

3.1 Introduction to Privacy and Trust Mechanisms

As discussed in Chapter 2, Federated Learning (FL) remains vulnerable to a variety of active and passive inference attacks that exploit the exposure of model updates and the lack of verifiability in the training process. To counter these threats, several privacy-enhancing and trust-building techniques have been explored. To provide a clearer overview, Figure 3.1 organizes these approaches into two main categories: **Privacy**, which groups methods dedicated to protecting data privacy, and **Trust**, which comprises techniques ensuring integrity, verifiability, and resilience.

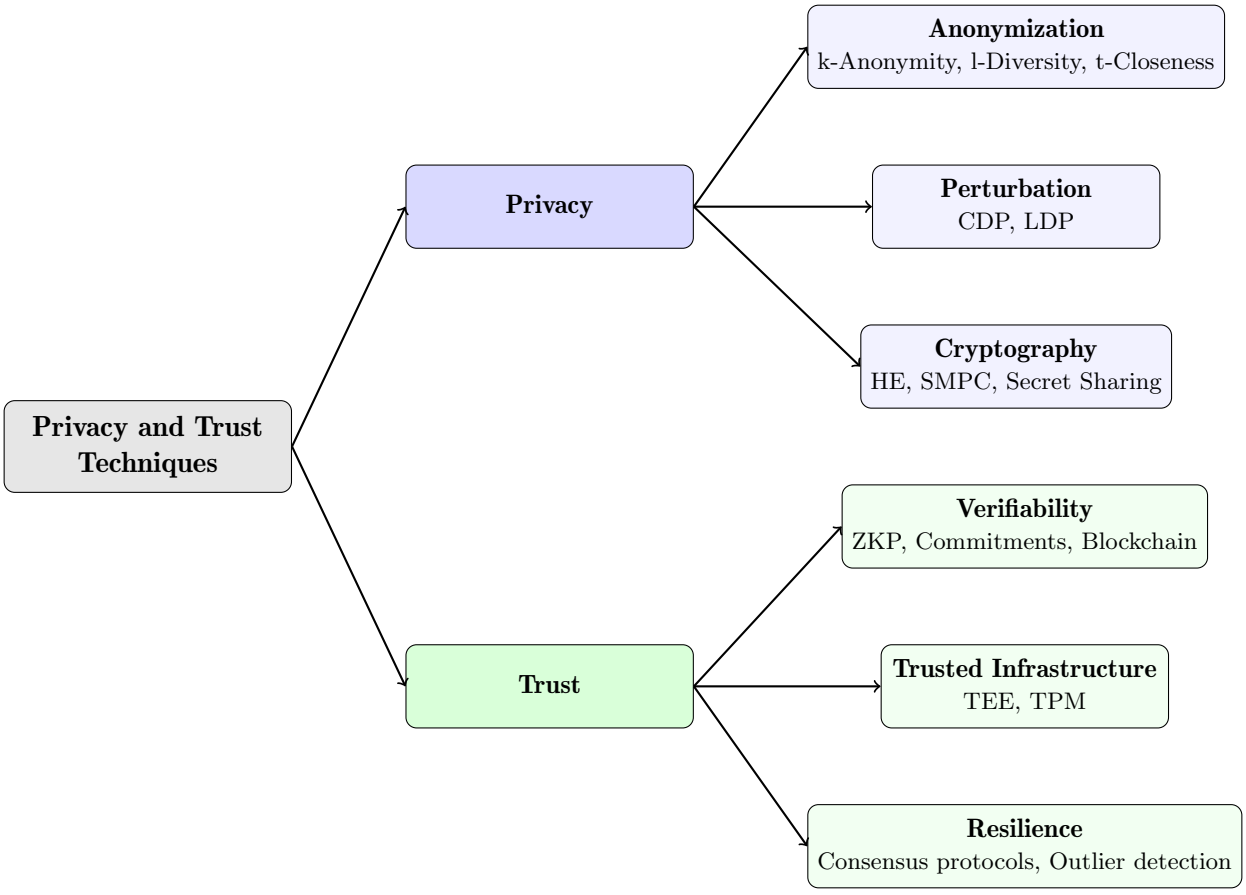


Figure 3.1: Taxonomy of Privacy and Trust-Enhancing Techniques

In this thesis, we focus on Differential Privacy (DP), since the analysis in the Table 3.1 shows that it is the only technique capable of protecting the final model in FL with lower overhead. Yet, relying on DP alone under strong assumptions degrades utility, the reason why we combine it with cryptographic techniques such as Homomorphic Encryption and Secret Sharing to mitigate this trade-off. We do

not consider anonymization methods, as they are not applicable in this context and have already shown their limits in practice. For trust, we concentrate on Zero-Knowledge Proofs (ZKP), as they directly align with privacy by ensuring verifiability without additional knowledge sharing, unlike other trust techniques that depend on hardware trust or introduce heavy overhead. This chapter introduces existing privacy and trust mechanisms and highlights their limitations when applied to FL, thereby motivating the need for the combined approach proposed in this thesis.

Table 3.1: Comparison of selected privacy-enhancing techniques in federated settings

Technique	**Protects Global Update**	**Protects Local Updates**	**Server Trust Assumptions**
Secure Aggregation	✗	✓	Honest but Curious
Secret Sharing (SS)	✗	✓	At least one honest
HE	✓(if not decrypted)	✓	Honest but Curious
Central DP	✓	✗	Trusted
Local DP	✓	✓	Untrusted

3.2 Differential Privacy

The intuition behind DP is simple: the presence or absence of a single individual in the dataset should not significantly change the output of the algorithm. To formalize this, Dwork et al.[55] consider two neighboring datasets, D_1 and D_2, that differ by only one record. An algorithm is said to satisfy DP if, when applied to D_1 or D_2, it produces outcome distributions that are nearly indistinguishable.

Definition 3.2.1. *[55] A randomized function M gives (ϵ, δ)-differential privacy if for all datasets D_1 and D_2 differing on at most one element, and all $S \subset Range(M)$,*

$$Pr[M(D_1) \in S] \leq e^{\epsilon} \times Pr[M(D_2) \in S] + \delta \tag{3.1}$$

In Definition 3.2.1, ϵ is a non-negative parameter that controls the strength of the privacy guarantee: the smaller ϵ, the stronger the protection. The parameter δ is a small probability that accounts for a possible failure of the guarantee; when $\delta = 0$, the definition corresponds to pure DP. Choosing an appropriate ϵ is highly context-dependent, and even within specific applications there is no consensus in the literature. Some works argue that claiming privacy with large values of ϵ is not realistic [56, 57], while more recent studies suggest that meaningful guarantees can still be achieved under large ϵ [58].

In this thesis, we keep ϵ as a variable and do not commit to a specific choice, leaving this decision to practitioners according to their deployment context.

Three major properties arise directly from this definition: composition, post-processing, and group privacy. These properties are the key to design powerful algorithm from basic mechanisms:

1. **Composability**: When applying DP multiple times on the same dataset, the overall privacy guarantee degrades in a controlled manner. We distinguish mainly: Sequential composition, parallel composition and advanced composition. Refer to appendix A for more details.

2. **Post-processing**: Any transformation applied after applying DP does not weaken the privacy guarantee.

3. **Group privacy**: this definition can be extended to group privacy by considering two datasets differing on at most k records instead of 1 record.

The core idea to apply DP in the real world applications is to calibrate random noise to the sensitivity of the function being computed, ensuring that no single record significantly changes the output distribution. Essentially, the sensivity, as defined in the definition 3.2.2, measures how much the output of a function can change when there is a slight change in the input data.

Definition 3.2.2. *[55] The global sensitivity Δf of a function $f : D^n \to R^d$ (with respect to l_1 metric) is the smallest number $S(f)$ such that for all $x, x' \in D^n$ which differ in a single entry*

$$\max ||f(x) - f(x')||_1 \leq S(f)$$

3.2.1 Examples of DP Mechanisms

The controlled noise is added using mechanisms that satisfy the definition 3.2.1. In this thesis, we do not aim to design or study new DP mechanisms. Instead, we rely on the Laplace/Gaussian mechanisms and their variants, which are widely used in the literature.

The Laplace mechanism achieves pure ϵ-DP by adding noise drawn from a Laplace distribution.

Definition 3.2.3. ***(Laplace mechanism [59])*** *Given any function $f : N^n \to R^k$, the laplace mechanism is defined as :*

$$M_L(x, f, \epsilon) = f(x) + Lap(\Delta f/\epsilon)$$

where $Lap(\Delta f/\epsilon)$ denotes a random variable drawn from a Laplace distribution.

The Gaussian mechanism provides an alternative to the Laplace mechanism and satisfies approximate $(\epsilon, \delta)-$DP. It adds noise sampled from a Gaussian distribution.

Definition 3.2.4. ***(Gaussian mechanism[60])*** *Given any function $f : D \to R$ over a dataset, the gaussian mechanism is defined as :*

$$M(x) = f(x) + \mathcal{N}(\sigma^2)$$

where $\mathcal{N}$ denotes a random variable drawn from a Gaussian distribution where $\sigma^2 = \frac{2(\Delta_2 f)^2 ln(2/\delta)}{\epsilon^2}$.

3.2.2 Local vs Central DP

Two main deployment models of DP are considered in the literature: the centralized model (CDP) and the local model (LDP).

In CDP, a trusted curator collects raw data from participants and then applies the DP mechanism centrally before releasing any statistics or trained models. This approach offers better utility, since the curator has access to the unperturbed data and can calibrate the noise globally. However, it requires a strong trust assumption: participants must rely on the curator to implement the mechanism correctly and not misuse the raw data prior to noise addition. In practice, this assumption is often unrealistic in sensitive domains such as healthcare or finance.

In LDP, the noise is added directly at the client side, before data leaves the user's device. This guarantees protection at the source, meaning that no central authority ever sees the original data. As a result, LDP provides stronger privacy guarantees. However, this comes at the cost of degraded utility: since each contribution is perturbed individually, the aggregated result often requires much larger datasets to achieve the same level of accuracy as CDP.

Property	**Central DP**	**Local DP**
Noise	Low (added once centrally)	High (added per user)
Utility	High (more accurate)	Low (noisy results)
Trust	High (requires trusted curator)	Low (no need for trust)

Table 3.2: Comparison between centralized and local differential privacy models.

3.2.3 Differential Privacy Amplification

DP can be enhanced through amplification techniques. The general idea is that the effective privacy loss of a mechanism is reduced when data is sampled or processed in specific ways. Two main approaches are commonly studied: subsampling and shuffling.

Amplification by Subsampling[61] reduces the effective ϵ by leveraging random participation of clients.

$$\varepsilon = \log(1 + p\,(e^{\varepsilon_0} - 1))\,, \qquad \delta' = p\,\delta.$$

Where p is the probability of a client to participate. In FL, this is naturally achieved when only a fraction of clients are selected per round. This amplification is exploited in the analysis of Chapter 8.

In the amplification by shuffling[62], each individual first applies a local randomizer, and the resulting outputs are permuted uniformly at random before analysis. The permutation breaks the link between outputs and individuals, which amplifies privacy guarantees. Shuffling assures for any $n \geq 1000$, $0 < \varepsilon_0 < \frac{1}{2}$, and $0 < \delta < \frac{1}{100}$, the algorithm A satisfies (ε, δ)-DP in the central model, with

$$\varepsilon \;=\; 12\,\varepsilon_0 \sqrt{\frac{\log(1/\delta)}{n}}.$$

Where ϵ_0 is the DP achieved by the local randomizer and n is the number of clients. In this thesis, shuffling is employed as an anonymization mechanism to strengthen the privacy guarantees of our proposed solutions, while also serving as the primary motivation for Chapter 4.

Beyond subsampling and shuffling, other forms of amplification have been proposed, including iteration-level amplification in stochastic optimization, privacy amplification by iteration [63], by compression [64], and by decentralization [65]. While less widely adopted, these approaches confirm that amplification is a general principle that can be applied in different algorithmic settings to improve the privacy–utility trade-off.

3.3 Differential Privacy in FL

The combination of DP with FL has been widely investigated in the literature, with different approaches depending on where the noise is injected. We can mainly distinguish four categories:

- **Input Data Level** [66, 67, 68]: noise is applied directly to the raw training data before local

training. This protects the data at its source but often leads to a significant drop in utility.

- **Local Model Level** [69, 70, 71, 72, 73, 74]: perturbation is added to gradients or model updates before they are sent to the server. This setting prevents the server from observing unprotected updates and is the most common scenario in Local DP.

- **Objective Function Level** [75, 76]: noise is incorporated directly into the objective function, usually by perturbing the loss or adding a regularization term. This guarantees privacy at the algorithmic level but often complicates the optimization process.

- **Global Model Level** [77, 78, 79, 80, 81]: perturbation is added after aggregation on the server side, directly to the global model parameters. This is the typical CDP setting and assumes that the server is trusted.

In this thesis, we acknowledge that in the literature the definition of DP can vary depending on the level of granularity (client-level or sample-level), which leads to different interpretations of local and central DP. For clarity, we consider Input Data Level and Local Model Level approaches as instances of Local DP, while the Global Model Level corresponds to CDP. The Objective Function Level has been less explored, mainly due to the additional complexity it introduces in optimization. In the following, we provide an overview of representative works under these different settings and discuss their limitations.

3.3.1 Centralized DP

The use of CDP in FL was first explored in 2017 by McMahan et al.[77]. They extended the FedAvg and FedSGD algorithms [1] with the moments accountant [82] to track the composition of Gaussian noise across rounds. Their approach applied classical CDP by clipping client updates to a fixed scale, thereby defining the sensitivity of the weights. The results showed that, with a sufficient number of participants, it is possible to train recurrent language models under DP while still preserving utility. In a similar independent study the same year, Geyer et al. [78] added subsampling on top of fixed clipping and confirmed that the number of clients plays a key role in the accuracy of the model.

Andrew et al. [79] proposed an adaptive clipping method in which the bound is set from a DP-estimated quantile of update norms, showing that clipping to the median norm generalizes well across

FL tasks. Zhang et al. [80] extended this direction by comparing clipping on model parameters versus gradients, and demonstrated that clipped gradients yield better utility. In contrast, Yuan et al. [81] focused on the noise side and introduced an amplitude-varying Gaussian mechanism, where the variance follows a geometric schedule across aggregations. Their central remark is that training rounds differ in sensitivity: early rounds with large updates can absorb stronger noise, while later rounds require weaker perturbations to preserve convergence, which leads to faster training and higher accuracy under the same DP budget.

Table 3.3: Comparison of central differential privacy techniques in federated learning.

Work	**Clipping**	**Mechanism**	**Takeaway**
[77]	Fixed	Gaussian	Differential Privacy in Federated Learning is feasible.
[78]	Fixed	Gaussian	More clients improve accuracy under Differential Privacy.
[79]	Adaptive	Gaussian	Adaptive clipping is better than Fixed clipping.
[80]	Adaptive	Gaussian	Gradient clipping is better than parameter clipping.
[81]	Fixed	Amplitude-varying Gaussian	Varying noise amplitude improves convergence and accuracy.

The previous solutions in Table 3.3 assumes a secure server for aggregation, which motivates the proposition of secure aggregation solutions. In this paradigm, the variance of the noise is devided across the clients, so each client add a small amount of noise to their locally clipped updates, encrypt them, and transmit them to the server, which only observes an aggregated sum that satisfies CDP. However, secure aggregation is based on modular arithemetic which is not compatible with continuous DP mechanisms [83]. This challenge has led to the design of discrete noise mechanisms like the Binomial mechanism [84], Discrete Gaussian mechanism [85, 86], Skellam mechanism [87] and Poisson Binomial mechanism [88]. It is worth to say that even if the noise is added at the client level, this is not local DP since the neighboring is considered at the global level.

3.3.2 Local DP

The use of LDP has been extensively investigated in the context of FL. The main motivation is that, in contrast to CDP, LDP ensures protection of the user's data even against a potentially malicious server, while granting clients greater flexibility in managing their own privacy. From a theoretical perspective, LDP in FL can be formulated as a high-dimensional mean estimation problem. However, applying standard LDP mechanisms in this setting results in severe utility degradation, as

discussed in Chapter 8. This is primarily due to the division of the privacy budget across multiple dimensions, which leads to a variance that grows exponentially with model size [83].

To address this challenge, several dedicated LDP mechanisms for high-dimensional FL have been proposed. Wang et al. [89] introduced the Piecewise Mechanism (PM) for collecting numerical data, which was later extended to the Hybrid Mechanism to handle mixed data types. Zhao et al. [90] advanced this line of work by proposing the Three Outputs mechanism for scenarios with tight privacy budgets, and PM-SUB, based on PM, for settings with larger budgets. Truex et al. [91] proposed an alternative approach by discretizing each local parameter and applying the exponential mechanism independently. These mechanisms were then used to perturb client-side updates, yielding the LDP-FedSGD framework.

Sun et al.[71] pointed out that fixing parameter ranges is not realistic. They proposed an adaptive LDP mechanism that adjusts to the weight ranges of different DNN layers, and added a shuffling mechanism at the parameter level to anonymize the data source. Compared to model shuffling, parameter shuffling is considered more effective against side-channel linkage attacks. This idea was later reused in mechanisms such as Adaptive-Harmony [92] and SRR-FL [93].

The shuffling of DP was studied by Girgis et al. [94, 95, 96] who addressed the challenge of poor learning performance in LDP by proposing privacy amplification through self-sampling and shuffling, where the server no longer knows which client contributes at each step, thus avoiding coordination in participant selection. Building on the same line of improving the privacy–utility trade-off, Wang et al. [97] introduced RAFLS, an adaptive FL framework based on Rényi DP and the shuffle model. Unlike Girgis et al., who focus on eliminating server knowledge of participation, RAFLS enhances utility by applying layer-wise adaptive noise injection and mitigating privacy budget explosion via parameter shuffling.

Another research direction focused on perturbing data inputs rather than parameters. Wang et al. [67] treated each data sample as an independent user and applied local perturbation before transmission, using the RAPPOR [98] mechanism to encode each feature with subsequent bit flips. Chamikara et al. [66] followed a different approach: they first extracted feature vectors from local datasets using convolutional and pooling layers of CNNs, flattened them into 1-D vectors, and then applied unary encoding combined with RAPPOR [98] perturbation before uploading to the server.

Table 3.4: Comparison of local differential privacy techniques in federated learning.

Work	DP Level	Contribution
[89]	Parameter	Proposed the first mechanisms tailored for high-dimensional FL (Piecewise and Hybrid Mechanisms).
[90]	Parameter	Introduced the Three Outputs mechanism and enhanced PM into PM-SUB for larger budgets.
[91]	Parameter	Developed LDP-FedSGD by discretizing parameters and applying the exponential mechanism.
[71]	Parameter	Proposed adaptive layer-wise LDP with parameter-level shuffling to defend against linkage attacks.
[92]	Parameter	Extended adaptive approaches with the Adaptive-Harmony mechanism for improved privacy–utility tradeoff.
[93]	Parameter	Proposed SRR-FL, strengthening robustness against inference attacks.
[94, 95, 96]	Parameter	Introducing privacy amplification through self-sampling and shuffling of LDP, removing server knowledge of client participation.
[97]	Parameter	combined Adaptive Rényi DP and shuffling with layer-wise noise to improve utility and mitigate budget explosion.
[67]	Input	Applied perturbation at the raw data level using RAPPOR, treating each data sample as a distinct user.
[66]	Input	Combined CNN-based feature extraction with RAPPOR perturbation to reduce dimensionality before sharing.

3.3.3 Discussion

DP provides a formal framework to limit information leakage in FL, but its application exposes fundamental trade-offs. In the centralized setting, the main limitation is the assumption of a trusted server. If this assumption is accepted in real deployment, the global model may leak sensitive updates before noise is applied. In the local setting, privacy is guaranteed against a malicious server, but at the cost of severe accuracy degradation, especially in high-dimensional models where the privacy budget must be divided across many parameters. Adaptive mechanisms, shuffling, and secure aggregation mitigate this problem to some extent but either rely on unrealistic system assumptions or still incur significant utility loss. Moreover, most existing works adopt relaxations of DP rather than pure DP. As pointed out in [99, 100], these relaxations should not be considered as true DP since they introduce vulnerabilities that make them prone to attacks.

These limitations motivate this thesis to avoid local DP because of the severe impact it has on utility. Instead, we argue that trust assumptions can be reduced by combining central DP with cryptographic techniques. Cryptography complements DP by protecting the confidentiality of updates during

transmission and aggregation, preventing exposure of raw information. In addition, these techniques can be used for trust by ensuring that the process was executed in the right way. When combined, cryptography ensures that the server only observes protected aggregates, while DP guarantees that even if we publish cleartext final model, no single client's data can be inferred. This hybrid design addresses the weaknesses of each approach in isolation and represents a promising direction for practical privacy-preserving FL.

3.4 Cryptography Techniques for DP in FL

To strengthen privacy in FL, this thesis combines DP with cryptographic primitives. We focus on two: Homomorphic Encryption (HE), which enables computation on encrypted data, and Additive Secret Sharing (SS), which secures updates by splitting them into random shares.

3.4.1 Homomorphic Encryption

Homomorphic encryption is a cryptographic primitive that enables arithmetic operations to be performed directly on ciphertexts without requiring decryption. The decrypted result is identical to what would have been obtained if the computation had been carried out in plaintext.

Formally, an encryption scheme is said to be homomorphic with respect to an operation $*$ if it satisfies the following property:

$$E(m_1) * E(m_2) = E(m_1 * m_2),$$

where E denotes the encryption algorithm and $m_1, m_2 \in M$, the message space.

Depending on the types and number of supported operations, HE schemes are classified as:

- **Partially Homomorphic Encryption (PHE)**: supports a single operation (addition or multiplication) an unlimited number of times. Examples include the Paillier scheme [101] (additive) and RSA[102] (multiplicative).
- **Somewhat Homomorphic Encryption (SWHE)**: supports both addition and multiplication, but only for a limited number of operations.
- **Fully Homomorphic Encryption (FHE)**: supports an unlimited number of both additions and

multiplications. This became feasible after Gentry's breakthrough to transform SWHE to FHE in 2009 [103].

In this work, we only consider additive homomorphic encryption, since our use case is limited to adding noise during aggregation. This choice is motivated by our interest on lightweight solutions in terms of time and computation.

3.4.2 Additive Secret Sharing

In this work, we employ additive secret sharing, a simple and widely used variant of secret sharing. The general notion of secret sharing was introduced by Shamir et al. [104], where a secret is split into multiple shares such that only authorized subsets of parties can reconstruct it.

Definition 3.4.1. *Let $x \in \mathbb{R}$ be a secret value. An additive secret sharing of x among n parties is a tuple of real numbers $(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$ such that:*

$$x = x_1 + x_2 + \cdots + x_n \tag{3.2}$$

Each share x_i for $i = 1, \ldots, n-1$ is drawn independently from a continuous distribution, and the final share x_n is computed to ensure the equality holds.

This special case of linear secret sharing is frequently used in secure multiparty computation protocols. Any subset of fewer than n shares provides no information about the original value x. One of the main advantages of additive secret sharing over real numbers is that addition and scalar multiplication can be performed locally and directly on the shares. For example, let x and y be two secret values shared as $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$, then the shares of the sum $x + y$ can be computed as:

$$(x_1 + y_1, \ldots, x_n + y_n)$$

3.4.3 Beyond Privacy: Trust and Verifiability

Privacy-preserving mechanisms such as DP and encryption ensure privacy but do not guarantee honest participation or correct execution. In FL, for instance, a client may deviate from the protocol or a server may manipulate aggregation results. It is therefore essential to highlight the need for verifiability: participants must be able to prove the correctness of their actions without revealing

private information. Zero-Knowledge Proofs (ZKPs) and commitment schemes provide such guarantees, bridging privacy with trust.

3.4.3.1 Zero-Knowledge Proofs

Zero-Knowledge Proofs are cryptographic protocols that allow a prover $\mathcal{P}$ to demonstrate the correctness of a statement x to a verifier $\mathcal{V}$ without revealing any additional information. Formally, given $x \in L$ for some language $L \in NP$, there exists a witness w such that (x, w) satisfies a relation R.

A ZKP must satisfy the following properties:

- **Completeness:** If $x \in L$, then for a honest prover $\mathcal{P}$, the verifier $\mathcal{V}$ accepts the proof π.
- **Soundness:** If $x \notin L$, then no dishonest prover $\mathcal{P}^*$ can convince $\mathcal{V}$ to accept π.
- **Zero-Knowledge:** The verifier learns nothing beyond the fact that x is true.

ZKPs can be broadly categorized into **interactive** and **non-interactive** proofs.

3.4.3.1.1 Interactive ZKPs. In interactive proofs, the prover and verifier exchange several messages. A central class of interactive ZKPs are the Sigma protocols, which allow a prover to demonstrate knowledge of a witness without revealing it (See Figure 3.2).

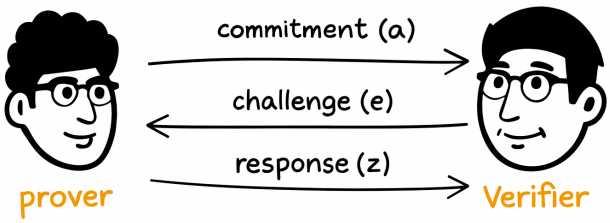


Figure 3.2: Sigma Protocols

The generic structure of a Sigma protocol consists of three moves:

1. **Commitment:** The prover selects random values and sends a commitment a to the verifier.
2. **Challenge:** The verifier sends a random challenge c.
3. **Response:** The prover computes and sends a response z, derived from the witness and challenge.

The verifier then checks a predicate over (x, a, c, z) to decide acceptance. Sigma protocols naturally satisfy completeness, soundness, and zero-knowledge, and form the basis of many practical interactive ZK systems. In this thesis, a Sigma protocol is used to prove the FL process in Chapter 8 without revealing additional information.

3.4.3.1.2 Non-Interactive ZKPs. Non-interactive proofs allow a prover to generate a single proof that can be verified without further interaction (See Figure 3.3).

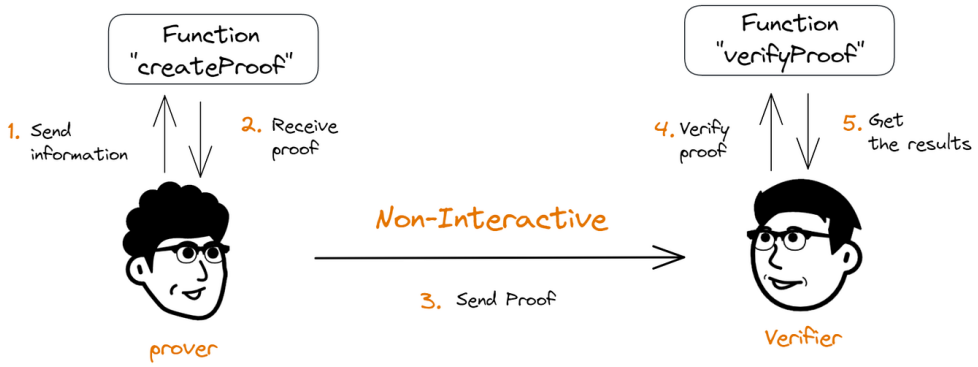


Figure 3.3: Non Interactive Proof[105]

It is worth to say that we can transform a Sigma Protocol to a non Interactive setting by applying the Fiat–Shamir heuristic [106], where the verifier's challenge is replaced by a cryptographic hash:

$$c = \mathcal{H}(x \parallel a).$$

There are also non-interactive ZKPs by design. For instance, **zk-SNARKs** (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge) are one of them and we will see their use to provide proofs about the application of DP in Chapter 5. zk-SNARKs provide the following additional guarantees:

- **Succinctness:** Proofs are short and can be verified in milliseconds.
- **Non-Interactivity:** A single proof suffices for verification.

In this work, we chose to use zk-SNARKs via ZoKrates[107] for their ease of use. However, we acknowledge that other non-interactive zero-knowledge proof libraries exist and may offer more

efficient solutions depending on the application. Many proof implementations build upon different contributions from the literature. Groth16 [108] remains the standard for minimal proof size and fast verification, and is widely used in privacy-focused cryptocurrencies such as Zcash, although it requires a trusted setup. Plonk [109] and its descendants provide universal setup conditions, improving flexibility. zk-STARKs [110], as implemented in Cairo and Winterfell, are transparent and require no trusted setup, while Bulletproofs [111] are particularly well-suited for range proofs.

3.4.3.2 Pedersen Commitments

Pedersen commitments [112] are cryptographic primitives frequently employed in zero-knowledge proofs. They allow committing to a value while keeping it hidden, with the ability to reveal it later.

Definition 3.4.2. *Let $\mathbb{G}$ be a cyclic group of prime order q, with generators $g, h \in \mathbb{G}$ such that the discrete logarithm $\log_g h$ is unknown. A Pedersen commitment to a message $m \in \mathbb{Z}_q$ with randomness $r \in \mathbb{Z}_q$ is:*

$$C = g^m h^r \in \mathbb{G}.$$

Pedersen commitments provide three important guarantees:

- **Perfect Hiding:** C reveals no information about m, even to an unbounded adversary.
- **Computational Binding:** Assuming the hardness of the discrete logarithm problem in $\mathbb{G}$, it is infeasible to open C to two different values.
- **Homomorphism:** Commitments are additively homomorphic. Given $C_1 = g^{m_1} h^{r_1}$ and $C_2 = g^{m_2} h^{r_2}$, their product is a commitment to $m_1 + m_2$:

$$C_1 C_2 = g^{m_1+m_2} h^{r_1+r_2}.$$

In the context of FL, we see the usage of Pedersen commitments in Chapter 8.

3.5 Related Work

Research at the intersection of FL, DP, and cryptography has introduced a broad spectrum of protocols to improve privacy and trust (See Table 3.5). A natural way to distinguish these works is

whether they include verification protocols or not. In this section, we give a high-level overview of selected recent contributions.

Table 3.5: Comparison of existing FL privacy and trust solutions ordered by publication year. LP: Local model Privacy, GP: Global model Privacy, RV: Real-time Verifiability, AV: Aggregation Verifiability, TV: Training Verifiability, DPV: Differential Privacy Verifiability, STA: Server Trust Assumption, CTA: Client Trust Assumption (*H: Honest, HbC: Honest-but-Curious, Ma: Malicious*), TA: Need for Trusted Authority or auxiliary nodes.

Work	**Year**	**LP**	**GP**	**RV**	**AV**	**TV**	**DPV**	**STA**	**CTA**	**TA**
Xu et al. [113]	2019	$\checkmark_{encrypted}$	$\checkmark_{ldp}$	×	×	×	×	*HbC*	*Ma*	✓
Heikkilä et al. [114]	2020	$\checkmark_{masked}$	$\checkmark_{cdp}$	×	×	×	×	*Ma*	*Ma*	✓
Wang et al. [115]	2020	$\checkmark_{ldp+ss}$	$\checkmark_{ldp}$	×	×	×	×	*HbC*	*H*	✓
Xu et al.[116]	2020	$\checkmark_{masked}$	×	✓	✓	×	×	*HbC*	*HbC*	✓
Gu et al. [117]	2021	$\checkmark_{ss}$	$\checkmark_{cdp}$	×	×	×	×	*HbC*	*Ma*	×
Guo et al.[118]	2021	$\checkmark_{masked}$	×	✓	✓	×	×	*HbC*	*HbC*	✓
Sébert et al. [119]	2022	$\checkmark_{encrypted}$	$\checkmark_{cdp}$	×	×	×	×	*HbC*	*HbC*	×
Ren et al. [120]	2022	$\checkmark_{encrypted}$	×	✓	✓	×	×	*Ma*	*Ma*	✓
Eltaras et al. [121]	2023	$\checkmark_{masked}$	×	✓	✓	×	×	*HbC*	*HbC*	✓
Zhang et al. [122]	2025	$\checkmark_{ss}$	$\checkmark_{cdp}$	✓	✓	✓	✓	*HbC*	*Ma*	✓
Korkmaz et al. [123]	2025	$\checkmark_{he+ldp}$	$\times_{partial-dp}$	×	×	×	×	*HbC*	*HbC*	×
Chen et al.[124]	2025	$\checkmark_{masked}$	×	✓	✓	×	×	*HbC*	*HbC*	×
Bellachia et al.[125]	2025	$\checkmark_{ldp}$	$\checkmark_{ldp}$	✓	✓	✓	×	*Ma*	*Ma*	×

3.5.1 Without Protocol Verification

The first class of approaches focuses on protecting privacy without enforcing protocol verification, typically by combining DP with cryptographic primitives such as secure aggregation.

3.5.1.1 CDP Based

In the centralized DP setting (CDP), most solutions rely on secure aggregation combined with encryption or secret sharing to preserve model accuracy while enforcing privacy

Heikillä et al. [114] proposed a cross-silo FL protocol that integrates distributed CDP with additive HE. Clients encrypt their local updates, which are then aggregated under encryption, with DP noise added on the server side. However, the scheme introduces heavy cryptographic overhead and remains vulnerable if the server behaves maliciously.

To improve efficiency, Gu et al. [117] replaced homomorphic encryption with additive secret sharing

across two non-colluding servers. Noise is distributed across two non-colluding servers, reducing cost compared to HE. Yet, the assumption of perfect non-collusion is unrealistic, as server misbehavior remains unverifiable.

Sébert et al. [119] identified a more subtle limitation of HE-based designs: because homomorphic encryption typically requires fixed-point arithmetic, Gaussian noise is distorted when rounded, undermining the guarantees of DP. Alternative discretized noise mechanisms, such as binomial, are difficult to extend to high-dimensional settings. To address this, they proposed a stochastic quantization operator that preserves the expected DP guarantees, and then performed homomorphic aggregation on the quantized values.

3.5.1.2 LDP Based

In contrast, LDP approaches shift the privacy burden to clients by adding noise before cryptographic protection, thereby removing the need to trust the server but often sacrificing utility.

Xu et al. [113] introduced HybridAlpha, a dropout-resilient FL framework where clients perturb their updates with LDP noise before encrypting them using Functional Encryption. In functional encryption, the server has only the possibility to learn the result of the aggregation but not the input. Functional encryption hides raw updates, but per client LDP causes severe accuracy loss. The protocol still assumes a semi-honest server, leaving it exposed to stronger adversaries.

To improve the utility of LDP, Wang et al. [115] developed the Private Encrypted Oblivious Shuffle protocol, which belongs to the shuffle model of DP. In this protocol, clients perturb their updates locally, split them into secret shares, and then encrypt them before submission to a network of shufflers. The shufflers obliviously shuffle the data before aggregation. Collusion is mitigated if at least one shuffler is honest, but reliance on external shufflers adds deployment barriers and reduces scalability.

A recent work, Korkmaz et al. [123] proposed a selective hybrid scheme in which only a fraction of model parameters are encrypted via homomorphic encryption, while the remainders are perturbed with LDP. This design reduces the cost of full encryption but weakens global protection, since non DP weights remain vulnerable to inference attacks.

In summary, existing solutions present a set of trade-offs that are still unresolved. Approaches based on distributed DP combined with homomorphic encryption aim to provide secure aggregation,

but they introduce significant computational overhead. Moreover, the alignment with DP is not straightforward: relaxations are often needed under modular arithmetic, which complicates both the guarantees and their interpretation. Centralized solutions, on the other hand, place both the noise addition and the aggregation in the hands of the same party, effectively reducing the problem to an assumption of honesty that is difficult to verify in practice.

LDP-based mechanisms eliminate the need to trust the server by injecting noise locally before cryptographic protection is applied. While this enforces privacy guarantees, it comes at the price of severe utility loss, since each client perturbs its update independently. Hybrid designs lower costs by combining partial encryption with local noise, but protection is incomplete: unprotected parameters remain exploitable. Even when more advanced primitives like functional encryption are considered, as in HybridAlpha, the approach remains inefficient, especially when the goal of the attacker is to target specific participants.

Taken together, these limitations highlight a critical gap. Current mechanisms create tradeoff between privacy and utility, and offer no systematic way to enforce honest behavior from servers and clients.

3.5.2 With Protocol Verification

A complementary direction introduces protocol verification, aiming to ensure that aggregation is performed correctly and honestly rather than relying solely on semi-honest assumptions.

3.5.2.1 Without DP

Research on verifiable aggregation in FL has evolved through several protocols that trade off efficiency, trust assumptions, and robustness. Xu et al. [116] introduced VerifyNet, a secure aggregation protocol combining double masking with verifiable computation. Aggregation integrity is proven via homomorphic hashes. Guo et al.[118] identified that the VerifyNet is communication inefficient and they propose to reduce that by replacing expensive primitives with linearly homomorphic hashes and commitments. While this improves efficiency, it still assumes a semi-trusted server and remains vulnerable to inference attacks from the global model.

Ren et al. [120] revisited both VerifyNet and VeriFL, identifying vulnerabilities in scenarios involving server-user collusion. They proposed a protocol that combines LWE-based encryption,

homomorphic hashing, and double masking to secure both local and global updates against malicious servers. However, their verification mechanism is only executed after training completes, delaying detection of misbehavior and limiting applicability in real-time adversarial settings.

Eltaras et al.[121] proposed a lightweight MPC-based aggregation protocol that outsources randomness generation to auxiliary institutions such as hospitals or banks. This approach avoids reliance on a central authority and improves efficiency compared to HE-heavy designs. However, it reintroduces dependence on external trusted entities. Chen et al. [124] extended this work to enhance robustness with protocol that integrates single masking, digital signatures, and timestamps to handle client dropout and collusion.

Overall, while these works advance the efficiency and verifiability of secure aggregation, they largely overlook protection against inference attacks launched against local and global model making them less efficient against a curious adversary. Without DP, the global model itself remains a leakage channel, which highlights a fundamental limit of these approaches.

3.5.2.2 With DP

More recent research proposed in parallel to our work attempts to unify DP, FL, and verifiability within a single framework, addressing the limitations of earlier approaches.

Bellachia et al. [125] introduce VerifBFL, a recursive zk-SNARKs framework based on Nova [126] that provides verifiability of both local training and server aggregation. Proofs are stored on-chain for auditability, but there is no mechanism to prove that DP noise was actually applied. Privacy is assumed but not verifiably enforced, which leaves the system vulnerable to clients who bypass or weaken the DP mechanism while still producing valid proofs of training.

Zhang et al. [122] point to the incompatibility between Byzantine-resilient aggregation and DP noise injection. To address this, they propose a two-layer architecture with secondary servers, separating Byzantine detection from DP enforcement. Their Noise-Immune Shamir Secret Sharing protocol, based on Pedersen commitments, enables verifiable noise addition, ensuring that central DP is provably enforced without breaking robustness guarantees. The server is modeled as honest-but-curious, while some secondary servers and clients may be malicious, which motivates the use of verifiability at each stage. Although aggregation is not accompanied by a formal zero-knowledge proof, the use of

commitments allows the aggregator to check consistency of shares, Byzantine filtering, and DP noise, thereby keeping aggregation within the scope of verifiability.

Both works use verifiability but restrict it to local DP in their solutions. As a result, they either suffer from reduced model utility due to local DP or from stronger trust assumptions ignoring malicious servers.

3.6 Conclusion

In this chapter, we provided an analysis of privacy and trust enhancing mechanisms in the context of Federated Learning. We first revisited the foundations of DP, highlighting its strengths as a formal guarantee and its limitations when applied in isolation. Central DP can provide more accurate models but depends heavily on the honesty of the server. Local DP, on the other hand, removes this dependency by shifting noise generation to the clients, but at the price of a severe degradation in utility. This trade-off between accuracy, trust, and practicality is one of the core tensions in the field.

We then explored how cryptographic techniques such as Homomorphic Encryption and Additive Secret Sharing can complement DP. Homomorphic Encryption enables computation over encrypted data, thus preventing the server from accessing sensitive updates, but its computational overhead and precision issues limit real-world scalability. Secret Sharing offers a more lightweight solution but actual solutions need heavy secure multi party computation to combine the results.

Beyond privacy, we examined mechanisms designed to enhance trust and verifiability. Zero-Knowledge Proofs, Pedersen commitments, and related techniques allow participants to verify the correctness of operations without revealing sensitive information. While promising, most existing protocols rely on simplified adversarial models and doesn't consider the verification of DP in a federated learning setting.

Taken together, these findings point to the need for combined approaches that move beyond isolated techniques. The integration of DP with lightweight cryptographic protocols and verifiability mechanisms appears as the most promising avenue to achieve robust guarantees without compromising model performance or usability. Such an approach would not only mitigate information leakage from gradients but also provide participants and regulators with tangible proofs of compliance, closing the gap between theoretical guarantees and regulatory requirements.

This analysis sets the stage for the next part, where we introduce our contributions that build upon these insights. By carefully combining differential privacy, cryptography, and verifiable protocols, we aim to design a solution that reduces trust assumptions, ensures verifiable privacy enforcement, and remains practical for real-world federated learning deployments.

Key Takeaways

- Differential Privacy provides formal guarantees but suffers from trade-offs between utility and privacy.
- Cryptographic Techniques protect updates only if they are not decrypted (combined if secret sharing).
- Differential Privacy is the only solution to protect leakage from published updates.
- The honesty of the central server is often assumed without any formal proof or verifiable guarantee.
- No single technique solves privacy, trust, and utility together.
- Differential Privacy is difficult to integrate with cryptographic protocols, as DP mechanisms are not naturally compatible with modular arithmetic.
- Verifying DP is inherently hard, since the randomness of the noise makes it difficult to prove that the mechanism has been correctly applied.
- Integrating Differential Private FL, cryptography, and verifiability is a promising direction to protect data from all the parties.

Part III

Contributions

Chapter 4

Reducing the Trust Model in FL: Homomorphic Encryption Approach

Contents

4.1 Introduction

This chapter is primarily based on our work *"Privacy Preserving Federated Learning: A Novel Approach for Combining Differential Privacy and Homomorphic Encryption"*, published at the International Conference on Information Security Theory and Practice (WISTP 2024) [127]. While the core methodology is drawn from that work, here we extend the analysis with additional evaluations and broader discussions to situate this contribution within the overall context of the thesis. This work is mainly driven by two motivations.

First motivation. As discussed in Chapter 3, model updates in federated learning remain the primary attack vector: both client updates and the aggregated global model must be protected against adversaries. DP is the only mechanism that can protect the global model from inference attacks, but in the CDP setting the two configurations used in the literature are unsatisfactory. In the first, the aggregator itself adds DP noise; however, a curious aggregator could remove or neutralize that noise after aggregation, leaving the global model unprotected. In the second, decentralized noise is combined with secure aggregation, which removes the single point of trust but introduces major practical challenges. Secure aggregation requires each client to establish pairwise keys with all others, leading to significant communication overhead; if a client drops out after masking, the cancellation of masks fails. While secret sharing can mitigate this, it adds further complexity. Moreover, when DP is applied, it is crucial to pay attention to the noise generation process. Poorly chosen randomness may produce correlated or duplicated noise streams due to the use of pseudo-random number generators, undermining privacy guarantees.

Our contribution addresses this challenge by decoupling aggregation and noise addition, while enforcing anonymization. This is achieved through a three-entity architecture composed of *clients*, a *shuffler*, and an *aggregator*. In this design, no single party can access raw updates. We accomplish this by creating a homomorphically encrypted tunnel for DP manipulation, preventing the aggregator from observing unprotected updates.

Second motivation. The second motivation comes from the LDP setting, which can be amplified by shuffling, thereby breaking the link between clients and their randomized updates. Erlingsson et al. [62] formalize strong amplification using Theorem B.1.1 (Theorem 7 in their work). However, they note:

> "The proof of Theorem 7 relies crucially on shuffling the data elements before applying the local randomizers. However, implementing such an algorithm in a distributed system requires trusting a remote shuffler with sensitive user data, thus negating the key advantage of the LDP model."

They further proved that it is possible to design a system in which amplification holds even if reports are randomized before shuffling, as formalized in Corollary B.2.1 (Corollary 9 of their work). The problem with this corollary is that it applies only if the number of clients ≥ 1000, $0 < \varepsilon_0 < 1/2$, and $\delta < 1/100$, which limits the applicability of this amplification. In this work, we propose to rely directly on the original theorem to achieve stronger and more general amplification.

In short, combining HE with DP in this system protects updates from the central server and other participants through DP noise, while simultaneously safeguarding data against the shuffler through HE.

4.2 System Design

Building on the motivations outlined above, we now describe the overall architecture of our framework.

4.2.1 High-Level Architecture

The proposed system introduces a three-entity architecture composed of *clients*, a *shuffler*, and an *aggregator*. Each entity has a distinct role and operates under the principle of separation of responsibilities to minimize trust in any single party. Clients perform local training, clip their model updates to bound sensitivity, and encrypt them using an additive homomorphic encryption scheme before transmission. The shuffler receives only encrypted updates: it permutes their parameters to anonymize contributions and homomorphically adds calibrated DP noise, without ever accessing the plaintext values. The aggregator receives the noisy and shuffled ciphertexts, decrypts them, and applies an aggregation strategy to produce the global model. Figure 5.1 summarizes the interactions between the three entities and the trust boundaries that separate them.

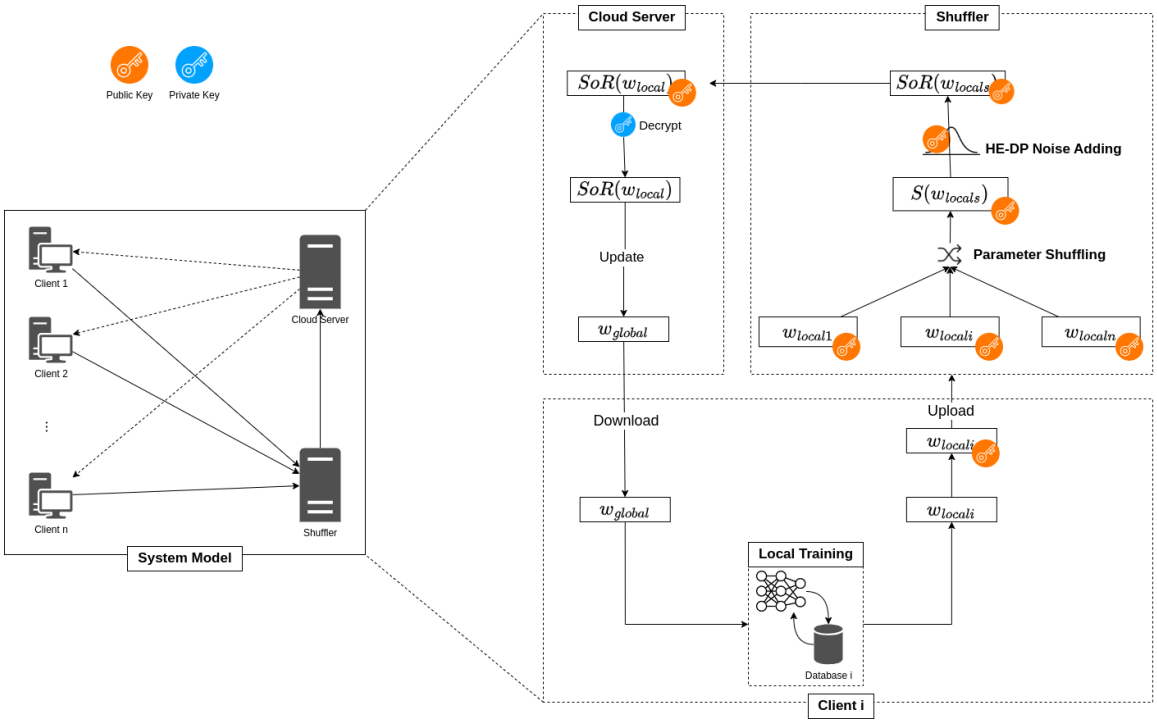


Figure 4.1: High-level architecture

4.2.2 Threat Model and Assumptions

The system is designed to protect against adversaries attempting to extract sensitive information from model updates during the federated learning process. Our focus is therefore on mitigating inference and reconstruction attacks targeting either the individual client updates or the aggregated global model. Other forms of adversarial behavior, such as poisoning or Byzantine faults, are considered out of scope for this contribution.

We assume adversaries can be either outsiders or insiders. Outsiders can be easily prevented by providing secure channels for all communications using encryption. On the other hand, insiders are participants in the learning process. The insider roles include:

- **The Aggregator** is honest in performing the aggregation but curious in analyzing the received updates. If it gains access to the exact DP noise values, it may attempt to remove or neutralize them to recover raw updates.
- **The Shuffler** is honest in executing shuffling and noise addition but curious about the updates it processes.

- **Clients** are honest in following the protocol: they train locally, clip their updates, encrypt them, and send them to the shuffler.

We assume that all communications between entities are secure. The public key of the aggregator is correctly distributed to all clients during setup. Static keys are used in this work, while in practice they should follow best practices. Finally, we assume no collusion between the shuffler and the aggregator, which is essential to ensure that no single entity can both access raw updates and remove the injected noise. This will be discussed more in the deployment aspects in Section 4.4

4.2.3 Workflow Overview

At a high level, the workflow of this system is organized into three phases: local preparation by clients, perturbation and anonymization by the shuffler, and aggregation by the server. Algorithm 1 summarizes these steps.

Algorithm 1 HE + Parameter Shuffling with Centralized DP Noise

Require: Clients $\{1..n\}$, number of rounds T, public key pk, secret key sk (aggregator)
Ensure: Global model $w^{(T)}$

for $t = 1$ **to** T **do**
  Client i**:**
    Train local model and compute update $w_i^{(t)}$
    Clip update to $\|w_i^{(t)}\|_2 \leq C$
    Encrypt update $c_i^{(t)} = \mathsf{Enc}_{\mathsf{pk}}(w_i^{(t)})$
    Send $c_i^{(t)}$ to shuffler
  Shuffler:
    Sample noise vector $z \sim \mathcal{N}(0, \sigma^2 I)$
    Encrypt noises $e = \mathsf{Enc}_{\mathsf{pk}}(z)$
    Compute encrypted noisy updates $c_{\text{noisy}}^{(t)} = c^{(t)} \oplus e$
    Shuffle $\{c_{\text{noisy}}^{(t)}\}$ across clients by parameter
    Forward shuffled noisy ciphertexts to aggregator
  Aggregator:
    Decrypt $c_{\text{noisy}}^{(t)}$ with sk
    Aggregate decrypted noisy updates using FedAvg
    Update global model $w^{(t+1)}$
end for

Key Management. In this system, the aggregator is responsible for generating the HE key pair (pk, sk). The public key pk is distributed to all clients and to the shuffler, enabling them to encrypt

model updates or noise contributions, while the secret key sk remains exclusively with the aggregator. This setup ensures that only the aggregator can decrypt ciphertexts, whereas the shuffler can add noise homomorphically without learning raw updates. For the scope of this work we assume a static key pair securely generated and distributed at initialization. However, in practice, public keys must be authenticated and the secret key must be protected using standard security practices.

Local Preparation

Each client trains a local model on its private data during a federated round. To ensure the updates are compatible with differential privacy, they are first clipped to a fixed bound. This clipping is essential for calibrating differential privacy noise by bounding the sensitivity. The updates are then encrypted with an additive homomorphic encryption scheme based on the previously generated key.

Because the shuffler only receives ciphertexts, it cannot inspect or infer sensitive information from the updates. At the same time, the homomorphic property allows the shuffler to manipulate ciphertexts without decryption. This provides a clear separation between data possession and data readability.

Perturbation And Anonymization

The shuffler is responsible for sampling noise vectors, encrypting them under the same public key, and combining them homomorphically with the encrypted updates. Importantly, the shuffler does not hold the secret key and therefore cannot decrypt client updates.

Noise can be incorporated either through LDP or through distributed CDP mechanisms. It is important to note that these two approaches do not provide the same level of protection. In both cases, if models are forwarded directly to the server, the server may still attempt to extract sensitive information by tracking updates across multiple federated rounds. For instance, it could cluster models to distinguish a particular user based on the magnitude of their weights or the accuracy of their local model.

To mitigate this risk, we introduce parameter-wise permutation across client updates (see Figure 4.2). Instead of forwarding each client's update as a whole, the shuffler reorders parameters so that contributions cannot be directly attributed to specific clients. Unlike approaches that shuffle entire models or layers, this strategy breaks the link not only between users and their models, but

also between parameters and their original sources. The aggregated model remains functional, but individual contributions are anonymized.

Figure 4.2: Example of Parameter Shuffling

Noise Calibration

The noise can be calibrated to offer LDP or CDP guarantees. The sensitivity of the updates is fixed based on the clipping used in the previous section. In the two cases the noise behaves in the same way. For example, using the Gaussian noise, for each parameter and each client, we draw independent samples from $\mathcal{N}(0, \sigma_i^2)$. When aggregated, the noise follows the standard Gaussian property:

$$\sum_i \mathcal{N}(0, \sigma_i^2) \sim \mathcal{N}\left(0, \sum_i \sigma_i^2\right).$$

So, we should calibrate the noise in a way that the variance across the process reaches a target value $\sigma^2_{\text{target}} = \sum_i \sigma_i^2$. This way we reduce the total amount of noise, while never giving the central entity direct access to the data. The privacy guarantee still holds thanks to the post-processing property of DP.

After that, the aggregator decrypts the received ciphertexts using its secret key and performs secure aggregation following an aggregation strategy.

4.3 Evaluation

The evaluation of the proposed system is divided into two parts. The first is a theoretical analysis covering the complexity and effective privacy guarantees. The second is an experimental validation focusing on the convergence of the global model.

4.3.1 Theoretical Analysis

The theoretical analysis covers the system complexity, the calibration and amplification of DP noise, and the effectiveness of parameter-wise shuffling.

System Complexity

A key consideration in federated learning protocols is the system overhead introduced by privacy-preserving mechanisms. Table 4.1 compares plaintext aggregation, secure aggregation, and homomorphic encryption. We denote by n the number of participating clients and by P the number of model weights. Plaintext aggregation has linear cost in both n and P but provides no privacy. Secure aggregation adds a quadratic term in n due to pairwise mask exchanges. This cost is negligible for large P but can dominate when P is small, since mask size does not depend on the model dimension. Homomorphic encryption preserves linear complexity but introduces a $\approx 128\times$ ciphertext expansion, making it the most bandwidth-intensive option. In practice, HE dominates for large models, while secure aggregation may be more costly for very small models and large client populations.

Table 4.1: Per-round communication overhead of different schemes.

Scheme	**Communication Overhead**	**Complexity**
Plaintext	$n \times P \times 4$ bytes	$O(nP)$
Secure Aggregation	$n \times P \times 4 \ + \ n(n-1) \times 32$ bytes	$O(nP) + O(n^2)$
HE-2048 (Ours)	$n \times P \times (512 + 4)$ bytes	$O(nP)$

DP Noise and Amplifications

The analysis in this section is based on the relation between the variance of the aggregation and the equations in Section 3.2.3. With this design, amplification can be achieved by subsampling in the central DP setting and by shuffling in the local DP setting. Figures 4.3 and 4.4 illustrate how amplification affects the relation between the Gaussian noise standard deviation σ and the privacy budget ε.

In the central model (Figure 4.3), subsampling clearly reduces the privacy cost. The smaller the participation rate, the larger the gain. For example, with 10% of the clients ($q = 0.1$), the noise needed to achieve the same ε is much lower compared to full participation. Increasing q weakens this effect,

but the overall benefit remains significant.

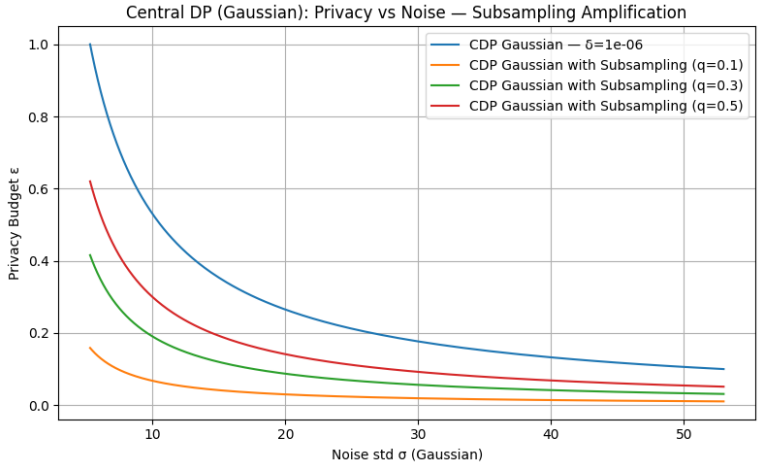


Figure 4.3: Relation between privacy budget and noise deviation in CDP with subsampling amplification.

In the local model (Figure 4.4), amplification through shuffling depends on the number of clients. Without shuffling, the variance of the noise is high, leading to reduced utility. As the population grows, amplification strengthens and the effective noise quickly decreases for the same privacy budget. Large-scale deployments therefore benefit naturally from this effect. However, amplification in this case is only valid for $\epsilon < \frac{\log(n/2)}{3}$ as stated by Erlingson et al.[62].

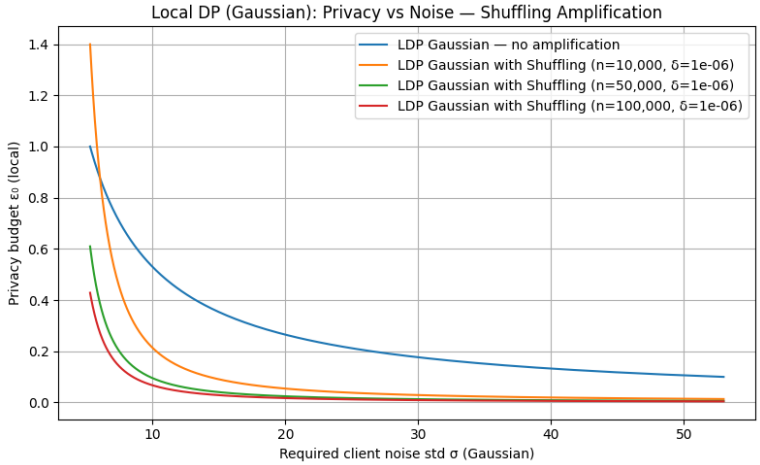


Figure 4.4: Relation between privacy budget and noise deviation in LDP model with shuffling amplification.

Parameter Shuffling Analysis

When parameters are permuted independently across clients, reconstructing a full client updates reduces to selecting one value from each of the d parameter columns. This gives n^d possible candidate vectors, of which exactly n correspond to true clients. The brute-force effort scales as n^d trials; with evaluation speed R candidates per second, the expected time is

$$T = \frac{n^d}{R} \quad \text{seconds}$$

For example, with $n = 10$ clients and $d = 100$ parameters, exhaustive search requires approximately 10^{91} seconds at 10^9 trials per second, making brute force infeasible. For more realistic deep learning models with thousands or millions of weights, the search space becomes astronomically large.

4.3.2 Experimental Validation

Evaluating the proposed design at scale quickly becomes infeasible due to the high computational cost of homomorphic encryption. Using 2048-bit keys on the machine available for our experiments, a single training round with only two clients required around 600 seconds for the considered CNN model. This overhead made it impractical to extend the evaluation to larger client populations or more complex models. For this reason, the experimental analysis focuses mainly on convergence behavior under different privacy settings, while runtime and communication overhead are discussed theoretically in the previous section.

Simulations are conducted using the FEMNIST[1] dataset using a CNN with two convolutional layers, one fully connected layer, and a softmax output. This deep learning architecture is commonly used in the literature[119]. The Gaussian mechanism is applied with $\delta = 10^{-5}$ and clipping bound $C = 1$, while ϵ is variable. The implementation is based on PyTorch and Flower with Paillier encryption, and shuffling is simulated using Python libraries.

Figure 4.5 shows accuracy over rounds. With moderate noise ($\sigma = 0.05$), the model converges close to the non-private baseline. Higher noise ($\sigma = 0.1$) slows convergence and lowers final accuracy. Under the LDP setting, convergence does not improve with additional rounds, confirming the limits of LDP in this context.

[1] https://huggingface.co/datasets/flwrlabs/femnist

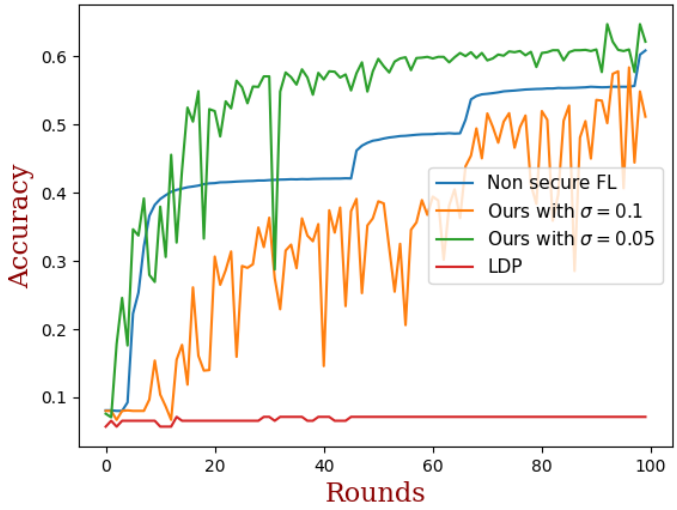


Figure 4.5: Impact of the number of rounds on accuracy

Table 4.2 reports the effect of the client population. With $\sigma = 0.05$, about 50 clients are sufficient to reach baseline accuracy, while with $\sigma = 0.1$ at least 100 clients are needed.

Table 4.2: Accuracy vs. number of clients

Clients	$\sigma = 0.05$	$\sigma = 0.1$	NoDP
12	0.08	0.19	0.52
20	0.13	0.07	0.53
50	0.59	0.07	0.62
100	0.64	0.55	0.61
150	0.73	0.58	0.60

4.4 Discussion and Limitations

The proposed design reduces the trust required in federated learning by separating responsibilities between clients, a shuffler, and an aggregator. Unlike secure aggregation protocols that rely on pairwise masking and are fragile under client dropouts, our approach leverages homomorphic encryption and differential privacy to prevent any single entity from accessing raw updates or removing the injected noise. The shuffler introduces anonymity and enables amplification through shuffling, while the aggregator remains limited to decrypting noisy updates and computing the global average. This architecture therefore achieves strong privacy guarantees with lower protocol complexity than existing

secure aggregation schemes based on masking.

The main cost of this design lies in implementation complexity and computational overhead. Homomorphic encryption introduces ciphertext expansion and additional computation at the client, shuffler, and aggregator. While encryption and decryption are standard operations, the extra burden comes from the shuffler, which must perform homomorphic additions, generate noise, and randomize contributions. These costs are acceptable in small to medium-scale deployments, but scaling to very large models remains challenging. A second critical issue is the placement of the shuffler: privacy guarantees depend on the non-collusion assumption between the aggregator and the shuffler. The following aspects are essential for secure deployment:

1. **Independence**: The shuffler must operate independently of the aggregator. This reduces collusion risk and enforces separation of trust. Deployment inside a Trusted Execution Environment (TEE) and external certification can further guarantee its neutrality.

2. **Randomization**: The shuffling algorithm must introduce sufficient randomness to prevent the aggregator or adversaries from inferring the order of contributions.

3. **Scalability**: The shuffler should scale with the number of clients, ensuring that computation and communication remain efficient in large deployments.

4. **Auditability and Transparency**: Operations of the shuffler must be auditable, allowing external verification and fostering trust among participants.

5. **Authentication and Authorization**: Strong authentication and strict access control are required to prevent unauthorized interaction with the shuffler.

6. **Resilience and Fault Tolerance**: The shuffler must tolerate faults and attacks, with mechanisms to continue or recover operation under adverse conditions.

7. **Legal and Regulatory Compliance**: The shuffler must comply with privacy laws and data protection regulations. Certification under recognized frameworks strengthens its legitimacy and acceptance.

In practice, several deployment options can be considered for the shuffler. One possibility is to operate it under a governmental or regulatory authority, which may strengthen independence and

ensure compliance with privacy laws. However, governments may lack the infrastructure to support such a service. Alternative options include consortium-based shufflers operated jointly by stakeholders, independent certification authorities, or implementations within TEEs, each of which provides a different balance between trust, scalability, and practicality.

Limitations

Despite its advantages, the design has clear limitations. First, the threat model assumes honest-but-curious adversaries and does not address poisoning or Byzantine behavior, which would require the addition of robust aggregation methods. It also assumes that the server behaves honestly in the aggregation step, but provides no mechanism to verify its actions, leaving the process without guarantees of verifiability. Second, key management is simplified to static keys, while real deployments demand applying best practices such as key rotation and revocation. Third, the computational and communication cost of homomorphic encryption remains high and prevents the system from scaling to large models, even if optimizations or hardware acceleration can reduce the burden. Fourth, the use of the standard Gaussian mechanism for differential privacy is another limitation, since it is not fully compatible with homomorphic encryption and points to the need for noise-generation methods adapted to encrypted domains as stated in Chpater 3. Finally, the shuffler remains a single point of availability, since its failure or compromise would disrupt the system, making redundant or distributed designs more appropriate in practice.

4.5 Conclusion

This chapter presented a federated learning framework that combines homomorphic encryption, shuffling, and differential privacy to reduce the trust required in the central aggregator. By separating aggregation, anonymization, and noise addition, the design prevents any single entity from accessing raw updates or removing the injected noise. Theoretical analysis showed that the framework preserves linear complexity while benefiting from privacy amplification through subsampling and shuffling. Experimental results confirmed that the model converges under moderate noise with sufficient client participation, but also highlighted the performance cost of homomorphic encryption.

Despite these guarantees, the approach has clear limitations. The threat model assumes honest-

but-curious participants and does not address poisoning or Byzantine attacks. Noise is added using the standard Gaussian mechanism, which is not fully compatible with encrypted computation. Most critically, the computational overhead of homomorphic encryption makes the system impractical for large models or large client populations. The shuffler also remains a single point of availability, and its deployment in practice would require strong guarantees of independence and trust.

This contribution should therefore be regarded as a proof of concept, demonstrating the feasibility of combining HE and DP to reduce trust in federated learning, rather than a system ready for deployment. Its limitations in scalability motivate the exploration of lighter-weight alternatives in Chapter 8, while the lack of verifiability motivates the exploration of cryptographic proofs, which is the focus of Chapter 5.

Key Takeaways

- The proposed solution makes curious adversaries ineffective by combining HE, DP, and shuffling.
- The design assumes honest behavior of the entities in the process. These assumptions are strong and deserve further discussion.
- Experiments show convergence under moderate noise, encouraging the use of privacy amplification in DP and having many participants.
- The solution supports two kinds of amplification: Amplification by Subsampling and Amplification by Shuffling.
- The solution reduce the need to establish pairwise keys between clients eliminating the risk of clients dropout.
- Amplification by shuffling is achieved under the general solution for any ϵ and any number of clients, unlike the original work[62] that required $0 < \epsilon < 0.5$ with more than 1000 clients.
- This solution should be seen as a proof of concept. Its scalability limits motivate the exploration of lighter alternatives in later chapters.

Chapter 5

Non Interactive Verifiability of Differential Privacy

Contents

5.1 Introduction

This chapter is mainly based on our publication "Enhancing Trust in Central Differential Privacy using zk-SNARKs and Cryptographic Hashes", presented at the International Conference on Advanced Information Networking and Applications (AINA 2025) [128]. While the original work was developed in a general scope, this chapter extends the discussion by exploring how the proposed framework can be adapted to the federated learning context.

First motivation. As discussed in Chapter 3, DP provides a formal and well-established mechanism for limiting information leakage. Its practical deployment often depends on assumptions of trust that are difficult to justify in real systems. In the CDP setup, the server is responsible for adding noise according to the declared privacy budget ϵ. However, real-world implementations frequently relax these guarantees to maintain higher utility [129, 99, 100]. More critically, a profit-driven server can intentionally manipulate the noise generation process: it may reduce or clip the noise to improve model accuracy, adjust the privacy budget ϵ to alter the noise scale, or even bypass the agreed noise mechanism altogether. Such manipulations directly weaken privacy and remain undetectable without verifiable enforcement. Without transparency or cryptographic proof, users must therefore rely entirely on the server's honesty. This situation motivates the need for mechanisms that can verify compliance with declared privacy parameters.

Second motivation. Current approaches [130, 131, 132] for auditing differential privacy systems are insufficient to establish trust. As stated by Shamsabadi et al. [133], post-hoc auditing techniques can reveal violations but cannot guarantee correctness. They rely on partial access to model updates, leak additional information, and require substantial computational effort. These methods provide only lower bounds on privacy loss and cannot certify that a claimed privacy guarantee truly holds.

Third motivation. Establishing verifiability through cryptographic proofs is far from straightforward. Noise generation involves arithmetic over continuous or large integer domains, while most zero-knowledge proof systems operate over finite fields with modular constraints. These challenges make the direct translation of privacy-preserving mechanisms into verifiable statements non trivial. Recent research has explored the integration of ZKPs into DP mechanisms to provide verifiable privacy guarantees [134, 135, 136]. These works demonstrate the potential of cryptographic proofs and commitments to verify that the injected noise complies with declared privacy parameters. However,

most of these approaches focus on simplified mechanisms such as randomized response [134, 136] for LDP or binomial noise generation [135] for CDP. Table 5.1 provides a comparative overview of similar studies combining DP and ZKPs. The comparison highlights differences in the DP setting, proof interactivity, noise mechanisms, and the resulting computational or communication overhead.

Table 5.1: Comparison of Privacy-Preserving Mechanisms Combining DP and ZKPs

Work	**DP Setting**	**Type of ZKP**	**Noise Mechanism**	**Comments**
[134]	LDP	Non-Interactive	Randomized Response	No Experiments
[135]	CDP	Interactive	Binomial	Communication/Computation Overhead
[136]	LDP	Non-Interactive	Randomized Response	Only for binary queries (not extensible)
Ours	**CDP**	**Non-Interactive**	**Discrete Laplace**	**-**

These limitations emphasize the need for a verifiable approach to CDP in which the noise generation process itself is subject to cryptographic proof. In this work, we design a framework that integrates zk-SNARKs and cryptographic hashes to provide non-interactive verification that the correct noise was applied, based on unbiased randomness and consistent parameters. This verifiability ensures that both clients and data owners can confirm that the declared privacy guarantees have been honestly enforced, without relying on external audits or post-hoc inference.

5.2 System Design

Building on the motivations outlined above, the goal is to make the noise generation and application process in CDP verifiable through cryptographic proofs and without disclosing more information about the data.

5.2.1 High Level Architecture

The proposed framework extends the laplace mechanism for CDP model by integrating verifiable computation components. It introduces a cryptographically verifiable workflow in which each privacy-preserving operation is accompanied by a zero-knowledge proof attesting to its correctness. The proposed system relies on a three-entity architecture composed of data owners, a server, and a client. Each entity plays a specific role and the responsibilities are separated to avoid placing full trust in a single party. Data owners validate queries against a predefined set of allowed operations and provide

the data needed for computation. The server executes the validated queries, generates Laplace noise in a distributed maneer, and generates proofs using zk-Snarks to certify the correctness of both the query processing and the noise generation. The client receives only the final noisy output along with the corresponding proofs, which can be publicly verified without access to the raw data.

This design ensures that data owners control what queries are allowed, the server performs computation without being blindly trusted, since every step is verifiable, and the client and data owners can independently check compliance with differential privacy guarantees. Figure 5.1 shows the interactions between these entities.

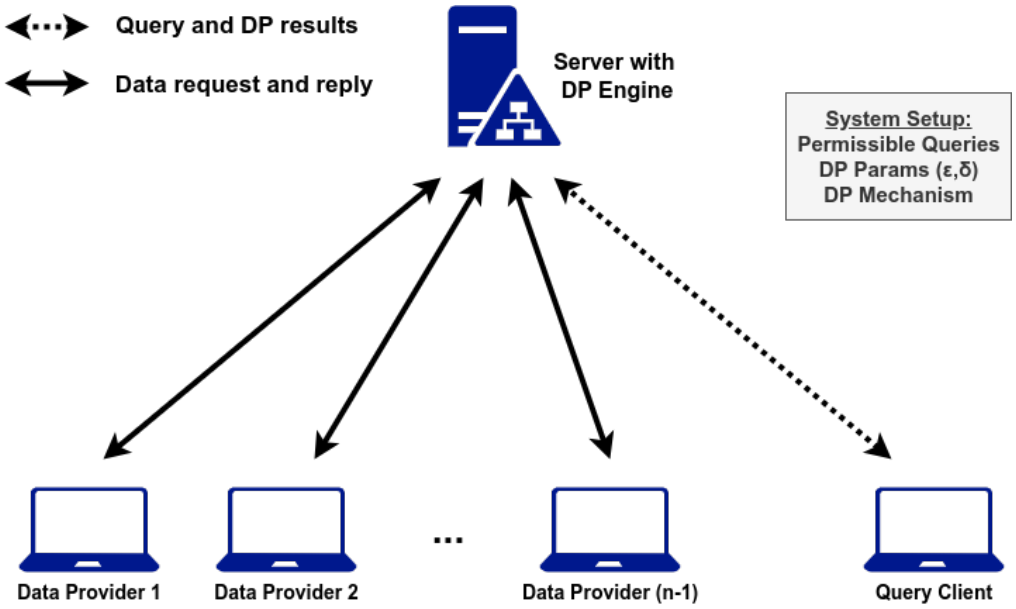


Figure 5.1: High Level Architecture

5.2.2 Threat Model and Assumptions

The system is designed to protect against adversaries attempting to weaken or bypass differential privacy guarantees during data processing. The main focus is on preventing a malicious server from tampering with noise generation or misreporting privacy parameters, as well as ensuring that data owners and the client can verify the integrity of computations. Other adversarial behaviors such as data poisoning, model poisoning, or Byzantine faults are outside the scope of this work.

We assume the following adversarial behavior and trust assumptions:

1. The Server is profit driven and may act maliciously. It is responsible for executing queries and generating noise, but it might misreport privacy parameters, reduce or skip noise addition, or tamper with query results. Its behavior must therefore be verifiable by external parties.

2. Data Owners are honest. They validate queries against a predefined set and provide their data. They do not collude with the server or the client.

3. The Client is curious and potentially malicious. It may attempt to learn more than intended from the noisy results of the server. It may also try to submit unauthorized queries in order to manipulate the process or infer private information. However, the client must not collude with the server, since such collusion would break the separation of responsibilities that ensures both privacy and verifiability.

We assume that commitments are published on an append-only public board accessed by all parties. We assume, also, that all communications between entities are encrypted. Proving and verifying keys for zk-SNARKs are established through a secure multi-party computation setup, preventing the existence of a single trusted party. The absence of collusion between the server and the client is essential, as their cooperation would eliminate all verification guarantees and expose raw data.

5.2.3 Detailed Workflow and Proof Generation

We first describe the operational workflow, which is devided into four stages: query submission and validation, noise generation, data transmission and processing, and finally result distribution. This workflow is designed to be easily verifiable using zk-Snark. Therefore, we then show how each step becomes publicly verifiable using non-interactive zk-SNARKs. This separation keeps the system description intuitive while making the trust guarantees explicit.

5.2.3.1 Operational Workflow

The operational workflow describes the sequence followed by the involved parties. It defines how data owners' contributions are collected, aggregated, and protected through noise addition under the declared privacy parameters. The computation proceeds through four main stages.

Stage 1: Query Submission and Validation. The objective of this phase is to ensure that only authorized queries are executed by the system. First, the client submits a query Q. Then, each data owner checks whether Q belongs to the predefined set of allowed queries. For example, a counting query such as "How many users are active in the system?" would be valid, while a query requesting raw individual records would be rejected. Only validated queries are processed.

Stage 2: Noise Generation. The objective of this stage is to ensure that differential privacy noise is generated correctly while remaining efficient within zk-SNARK circuits. Several methods exist to sample noise from a Laplace distribution, such as Inverse Transform Sampling, Rejection Sampling, and Gaussian Approximation. However, these approaches rely on complex functions, such as logarithms, exponentials or floating-point arithmetic. These functions are not suited for zk-SNARK circuits due to high constraint counts and complex range handling. To overcome these limitations, we adopt the Discrete Laplace Mechanism, known to satisfy ϵ-DP [137]. We know that we can easily generate the discrete laplace distribution as the difference between two independent geometric distribution [138]. In this work, we use Algorithm 2.

Algorithm 2 Efficient Laplace Noise Generation

Require: b (scale parameter of the Laplace distribution)
Ensure: η (Laplace-distributed noise)
1: Each data owner provides two independent geometric samples k_i, k'_i.
2: The server computes $g_1 = \min(k_1, \ldots, k_n)$ and $g_2 = \min(k'_1, \ldots, k'_n)$ with $p = 1 - \exp(-1/b)$.
3: Compute the Laplace noise: $\eta = g_1 - g_2$.
4: **Return:** the noise η.

Each g_1 and g_2 follow a geometric distribution with updated success probability $p' = 1-(1-p)^n = 1 - e^{-1/b}$, where $b = \Delta Q/\epsilon$ denotes the Laplace scale. The resulting noise is $\eta = g_1 - g_2$ is discrete-Laplace distributed and can be efficiently verified inside a zk-SNARK circuit.

Stage 3: Data Transmission and Query Processing. Once noise has been generated, the system proceeds to collect and process the validated data under verifiable commitments. Each data owner submits its private data D_i to the server with a zk-SNARK friendly hash C_i (e.g., MiMC Hashes [139], Pedersen Commitments) published on a public board, ensuring integrity and non-repudiation. Once the server receives all data, it executes the validated query Q as $R = Q(D_1, D_2, \ldots, D_n)$.

To preserve scalability and maintain consistency with federated settings, this work focuses on decomposable queries [140, 141], which can be expressed as the composition of sub-queries on disjoint data partitions.

Definition 5.2.1 (Decomposability)**.** *A query Q is decomposable if it can be broken into sub-queries $Q_1, Q_2, \ldots, Q_n$ such that:*

$$Q(D) = Compose(Q(D_1), Q(D_2), \ldots, Q(D_n)),$$

where: D is the whole dataset, D_i are partitions of D, Q is the query.

This property enables parallel computation and naturally aligns with the federated learning paradigm, where each sub-query Q_i can be interpreted as the local model update of a client.

Stage 4: Result Distribution. Finally, the server releases the noisy result $R_\eta = R + \eta$ to the client. This result is accompanied by the corresponding commitments and the proofs of correctness.

5.2.3.2 Proving Workflow

This subsection specifies how each functional stage becomes verifiable through zk-SNARK. Each proof certifies the correctness of one computational step while preserving data confidentiality.

Laplace Noise Proof π_{noise}. To verify the correctness of the discrete Laplace noise generation, the server constructs a proof π_{noise} certifying that: $\eta = g_1 - g_2$. The NP statement is:

$$\exists\, g_1, g_2, r_{g_1}, r_{g_2} \text{ such that } \begin{cases} C_{g_1} = \mathsf{Com}(g_1; r_{g_1}), \\ C_{g_2} = \mathsf{Com}(g_2; r_{g_2}), \\ C_\eta = \mathsf{Com}(\eta; r_\eta), \\ \eta = g_1 - g_2. \end{cases}$$

While this proof validates the difference computation, it does not by itself ensure that g_1 and g_2 are generated in the right way. For this, a second proof π_{min} is required for each g_i. This proof guarantees that the committed value corresponds to the true minimum among a set of geometric samples, without disclosing any sample or its index. Each data owner provides commitments $C_{k_i} = \mathsf{Com}(k_i; r_i)$ for its geometric samples. The server privately computes $g = \min(k_1, k_2, \ldots, k_n)$ and publishes a commitment $C_g = \mathsf{Com}(g; r_g)$ together with a proof π_{min}. The proof π_{min} attests the following NP statement:

$$\exists g, r_g, \{k_i, r_i\}_{i=1}^n \text{ such that } \begin{cases} C_g = \mathsf{Com}(g; r_g), & \\ C_{k_i} = \mathsf{Com}(k_i; r_i), & \forall i \in \{1, \ldots, n\}, \\ \prod_{i=1}^n (k_i - g) = 0, & \text{(membership condition)}, \\ k_i - g \geq 0, & \forall i \in \{1, \ldots, n\} \text{ (minimality condition)}. \end{cases}$$

The membership condition ensures that g equals one of the committed inputs, while the minimality condition guarantees that g is no greater than any other committed value.

Since SNARKs operate over finite fields with positive integer values in $[0, p-1]$ where p is a large prime number, the non-negativity constraint is enforced by expressing each difference $d_i = k_i - g$ in its binary representation inside the circuit:

$$d_i = \sum_{j=0}^{\ell-1} 2^j b_{i,j},$$

where each $b_{i,j}$ is a binary variable constrained by

$$b_{i,j} \cdot (b_{i,j} - 1) = 0, \quad \forall j \in \{0, \ldots, \ell - 1\}.$$

These constraints guarantee that $d_i \in [0, 2^\ell)$, thus ensuring that $k_i \geq g$. Further examples on this encoding are provided in Appendix C.

Query Execution Proof $\pi_{compose}$. This proof verifies that the server executed the validated query correctly and aggregated all data-owner inputs as specified.

$$\exists R, r_R, \{D_i, r_i\}_{i=1}^n \text{ such that } \begin{cases} C_i = \mathsf{Com}(D_i; r_i), & \forall i \in \{1, \ldots, n\}, \\ C_R = \mathsf{Com}(R; r_R), & \\ Compose(D_1, \ldots, D_n) - R = 0, & \end{cases}$$

This ensures that all inputs contributed to the final result, and that no value was substituted, excluded, or tampered with during computation.

Noise Addition Proof π. This proof ensures that the final released result R_η has been correctly computed as $R + \eta$.

$$\exists\, R, \eta, r_R, r_\eta \text{ such that } \begin{cases} C_R = \mathsf{Com}(R; r_R), \\ C_\eta = \mathsf{Com}(\eta; r_\eta), \\ R_\eta = R + \eta. \end{cases}$$

Together, these four proofs establish a complete chain of verifiability, as illustrated in Figure 5.2. The structure can be implemented either as a single, unified proof encompassing the entire verification process or as a set of more granular sub-proofs, each dedicated to a specific stage of computation. In the approach adopted here, the Laplace Noise Proof ensures the correctness of the generated noise through the sub-proofs π_{min1}, π_{min2}, and π_{noise}, while the Query Proof $\pi_{compose}$ guarantees the correct composition of this verified noise with the committed query result. This modular design is again conceived to make the relation easier to the Chapter 4 (See the section 5.4 for more details).

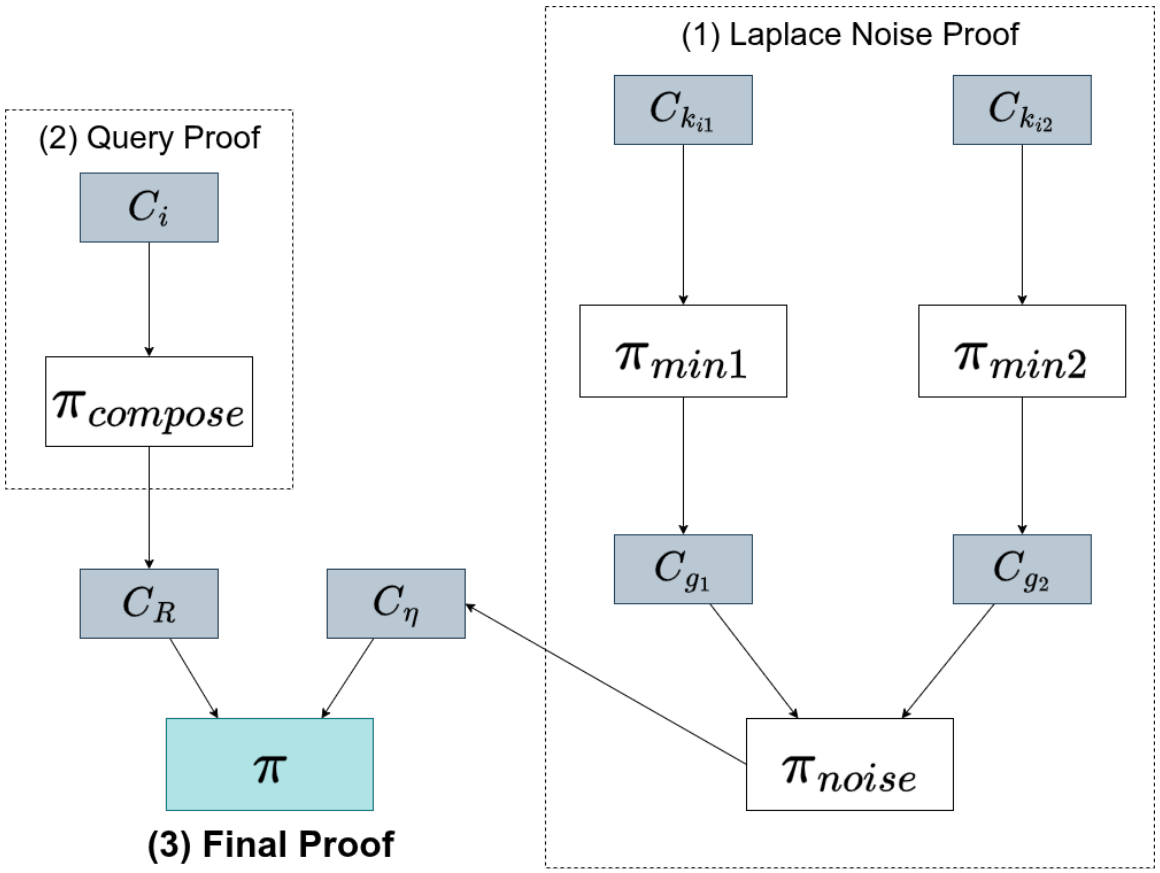


Figure 5.2: Verifiability Chain

5.3 Evaluation

This section presents the experimental evaluation of the proposed framework. The objective is to assess its computational efficiency, scalability, and statistical correctness of noise generation, while highlighting the trade-offs between verifiability and performance.

We specifically focus on counting queries, following the methodology of Biswas and Cormode[135]. Counting queries were simulated for each data owner, as the specific data content was irrelevant to the tests. The simulations were implemented using Python 3 and ZoKrates for zk-SNARK proof generation and verification. Experiments ran on an Intel i5 processor with 32 GB RAM and Ubuntu 24.04. We measured execution time, resource usage, and proof verification, ensuring noise accuracy matched the theoretical Laplace distribution.

Performance Evaluation

To measure the computational efficiency of the proposed framework, we first considered a scenario with 12 data owners.

Table 5.2 shows that the computational load is mainly concentrated in the π_{min} proof, which is the most demanding due to the verification of minimality conditions across multiple inputs. This step naturally involves a higher number of constraints (9216), explaining its dominant share in the overall runtime. In contrast, the other proofs (π_{noise}, $\pi_{compose}$, and $\pi_{addition}$) remain lightweight. Verification stays consistently fast across all stages, confirming that public validation remains efficient and easily parallelizable.

Overall, the framework completes the full verification process in around one second for 12 data owners.

Table 5.2: Performance Metrics for Each Phase

Phase	**Server Time (ms)**	**Verifier Time (ms)**	**# Constraints**
Stage 1	49.01	3.66	-
Stage 2 : π_{min}	404.92	11.32	9216
Stage 2: π_{noise}	94.83	11.62	2338
Stage 3: $\pi_{compose}$	183.69	10.43	3344
Stage 4: $\pi_{addition}$	98.76	9.37	2338
Overall	**1433.03**	-	**17236**

Scalability Analysis

We evaluate how the number of clients affects two key metrics: execution time and the number of constraints in the zkSNARK circuit. By increasing the number of clients, we observe the system's performance and complexity under varying loads.

As shown in Figure 5.3, the execution time and the number of constraints both increase significantly with the number of clients. This highlights the challenges of scaling zk-SNARK-based systems, where handling more clients leads to higher computational overhead and circuit complexity. The results emphasize the importance of optimizing for scalability while maintaining efficiency and security.

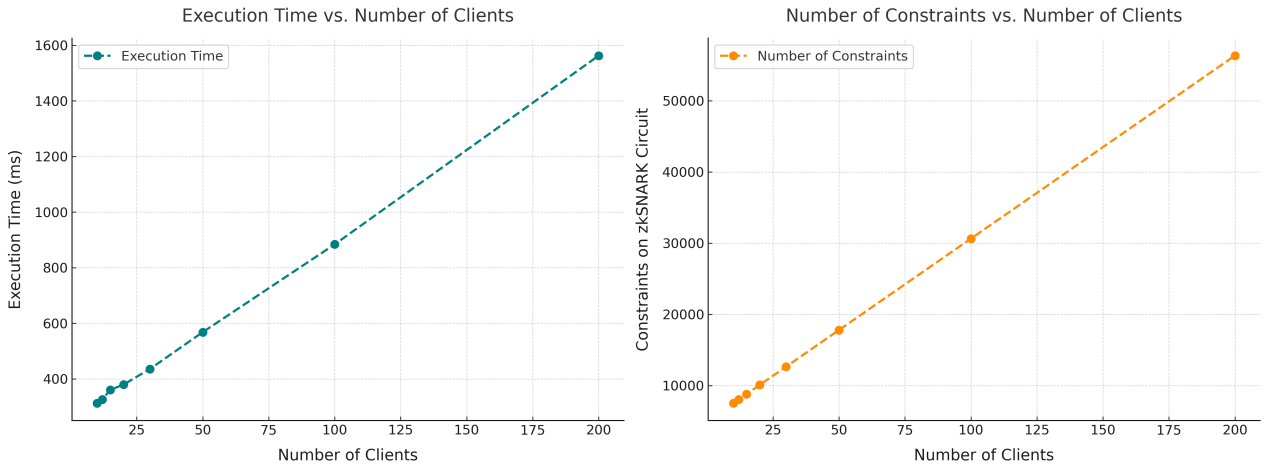


Figure 5.3: Execution Time and zkSNARK Circuit Constraints vs. Number of Clients

Noise Distribution Verification

To demonstrate the practicality of generating Laplace noise, we generated 1,000 random values following the protocol described in the section 5.2.3.1, parameterized by the privacy budget ϵ and sensitivity of 1, and we compared it to the theoretical distribution.

Figure 5.4 shows that the histogram of the generated noise aligns closely with the theoretical Laplace distribution. This confirms that the noise generation method is practical and accurately maps to the expected distribution.

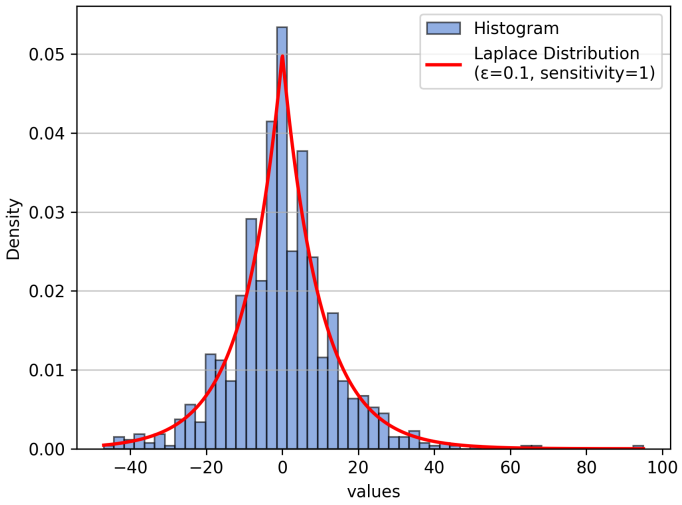


Figure 5.4: Histogram of generated noise and comparison with the target laplace distribution

5.4 Discussion

The proposed framework integrates cryptographic verifiability into the Discrete Laplace Mechanism under CDP by integrating zk-SNARKs directly into the noise generation and query execution process. Unlike traditional DP systems that rely entirely on the honesty of a central server, this design allows each step to be publicly verifiable without revealing raw data. The approach thus bridges the gap between privacy and transparency by ensuring that declared privacy budgets are enforced.

From an architectural standpoint, the system separates responsibilities between three entities: data owners, a server, and a client. Data owners validate and authorize queries; the server performs computations and generates cryptographic proofs; and the client verifies outputs without accessing sensitive data. This separation of duties eliminates single points of trust and ensures accountability through cryptographic evidence. Moreover, the use of discrete Laplace noise instead of continuous sampling enables efficient implementation inside zk-SNARK circuits, which are otherwise constrained by field arithmetic and modular operations.

This architecture was intentionally designed to align with the federated learning setting introduced in the previous chapter. In that context, the entities of the CDP framework can be directly mapped to the components of the federated architecture: *data owners* correspond to the *clients*, the *server* plays a similar role to the *shuffler*, and the *client* in the CDP model can be mapped to the *aggregating*

server.

However, adapting this verifiable CDP model to the previous federated design requires rethinking the location of proof generation. In the current framework, the proof of composition is generated at the CDP server level, which corresponds to the shuffler in the previous architecture. Since aggregation in the federated framework is performed by the aggregating server, this proof should instead be shifted to that entity to ensure consistency with the system's trust model and data flow. This adjustment would align the verifiable noise generation and aggregation proofs with the responsibilities of each component, preserving both correctness and transparency.

Limitations

Despite its promising results, the proposed framework exhibits several limitations that must be addressed to ensure scalability, flexibility, and real-world applicability.

1. **Scalability:** The computational cost of zk-SNARK proof generation grows linearly with the number of data owners. Although verification remains efficient, large-scale deployments would require circuit optimization, proof aggregation, or parallel proving mechanisms to remain practical.

2. **Limited Query Set:** The framework currently supports decomposable queries, such as counting or summation, which can be easily verified under the discrete Laplace mechanism. However, the evaluation was limited to simple counting queries, leaving the expressiveness of the framework untested for more complex analytical tasks. Extending experiments to more complex computations should be considered in order to validate the solution.

3. **Trust and Non-Collusion Assumptions:** The model assumes honest-but-curious data owners and a non-colluding relationship between the server and the client. In real-world deployments, enforcing non-collusion is difficult and may require additional organizational or regulatory controls. Incorporating verifiable secret sharing or multi-party computation protocols could mitigate this limitation by reducing trust dependencies between entities.

4. **Privacy Scope:** The framework enforces CDP guarantees against the client, ensuring that the final client cannot infer sensitive data from noisy outputs. However, it does not protect privacy

against the server itself, which retains access to raw data during computation. Achieving full end-to-end privacy would require extending the model with encryption, secure aggregation, or multi-party computation mechanisms. This is where a more exhaustive framework combining the work from chapter 4 is required. However, using HE introduces significant computational bottlenecks and heavy additional overhead. These constraints complicate the seamless integration of this framework within the previous Chapter.

5. **Performance Overhead:** Even though verification is fast, proof generation remains resource-intensive, both in time and memory consumption. This overhead is acceptable for proof-of-concept or medium-scale deployments but may be prohibitive for large-scale federated systems.

5.5 Conclusion

This chapter introduced a verifiable framework for Central Differential Privacy that integrates zero-knowledge proofs through zk-SNARKs and cryptographic commitments to ensure the integrity of the noise generation and application process. The main contribution lies in demonstrating that verifiability and DP can effectively coexist within a unified design. Each privacy-preserving operation is accompanied by a cryptographic proof certifying its correctness, ensuring that declared privacy guarantees are enforced without exposing raw data. The adoption of the discrete Laplace mechanism, instead of its continuous counterpart, enables efficient implementation inside zk-SNARK circuits.

Experimental results confirmed the technical feasibility of this approach. Proof verification remains lightweight and parallelizable, while the noise generation process accurately follows the target Laplace distribution. These results show that the proposed framework maintains both statistical soundness and public verifiability with acceptable computational overheads for small scale deployments.

Nonetheless, several limitations persist. The scalability of zk-SNARK proofs remains a key challenge, as proving time grows linearly with the number of participants. The framework currently supports a restricted set of decomposable queries, such as counting and summation, leaving complex analytical operations for future exploration. Furthermore, the trust model assumes honest-but-curious data owners and non-colluding entities conditions that are difficult to guarantee in real-world deployments. Achieving end-to-end privacy would require integrating this framework with secure aggregation, encryption, or multi-party computation techniques.

Based on these findings and limitations, the next chapter explores interactive proofs and secret sharing to propose a unified verifiable and privacy aware federated learning framework using lightweight commitments.

Key Takeaways

- The proposed framework embeds zk-SNARK proofs into the CDP workflow, enabling public verifiability of both the noise generation and query execution processes without revealing raw data.
- The integrity of the inputs are ensured through cryptographic commitments, creating a verifiability chain through the proofs.
- Experimental results confirm that verifiability can be achieved with low verification cost and acceptable proving time for small scale deployments, while the generated noise accurately follows the discrete Laplace distribution.
- Despite its effectiveness, the approach faces scalability challenges, supports only decomposable queries, and relies on non-collusion and honest-but-curious assumptions among entities.
- This work serves as a proof of concept demonstrating that verifiability and DP can coexist within a unified design, laying the groundwork for future extensions using interactive proofs, secret sharing, for end-to-end privacy-preserving federated learning frameworks.

Chapter 6

A Framework for Private and Verifiable Federated Learning

Contents

6.1 Introduction

The previous chapters explored two complementary strategies to enhance privacy and trust in FL.

In Chapter 4, we investigated how homomorphic encryption could canceal the updates from every participant, allowing the aggregation of model updates while preserving DP. This approach successfully limited the exposure of individual updates through the separation of the responsibility of adding noise from the aggregation process. This approach used homomorphic encryption in order to canceal the model updates from the shuffler while DP is used to protect against final clients and aggregation server. However, it introduced significant computational and communication overheads, which limit its scalability in realistic cross-device environments. Moreover, the shuffler played a central role in orchestrating applying the DP mechanism, without any verification process, leaving residual trust dependencies unresolved.

In Chapter 5, we shifted the focus toward verifiability by integrating zk-SNARKs and cryptographic commitments into the differential privacy workflow. This method provided strong formal guarantees that the declared privacy budget had been correctly enforced, without revealing either raw data or the added noise. Nevertheless, the approach relied on zk-SNARKs, which, while powerful, are computationally expensive and difficult to scale as the number of clients or the model dimension grows. Furthermore, the verification process still assumed that the server is not curious but .it may behave in a Byzantine manner. Therefore, it must provide proofs of correct behavior. The main challenge was how to certify the right generation of the noise which was hard because of the inheretent randomness considered on it.

These limitations reveal an underlying tension: proposed solutions tend to favor either strong privacy via encryption and DP or strong verifiability via cryptographic proofs, but combining both solutions is hard without sacrificing scalability. To bridge this gap, we propose in this chapter a lightweight and distributed alternative that removes the need for any trusted central entity, while keeping the computational and communication costs practical for federated learning environments.

The proposed framework, named ProoFed, relies on additive secret sharing and distributed noise generation to achieve end-to-end privacy under the central differential privacy model, without employing heavy cryptographic primitives such as homomorphic encryption or zk-SNARKs.

6.2 System Design

This section details the architecture and workflow of the proposed system, focusing on the integration of additive secret sharing and distributed noise generation for privacy-preserving federated learning.

6.2.1 High Level Architecture

Figure 6.1 presents the high-level architecture of the proposed privacy-preserving federated learning framework. The design integrates three complementary mechanisms: *differential privacy*, *additive secret sharing*, and *cryptographic commitments*. Together, these components ensure that model updates remain confidential, verifiable, and unlinkable throughout the training process.

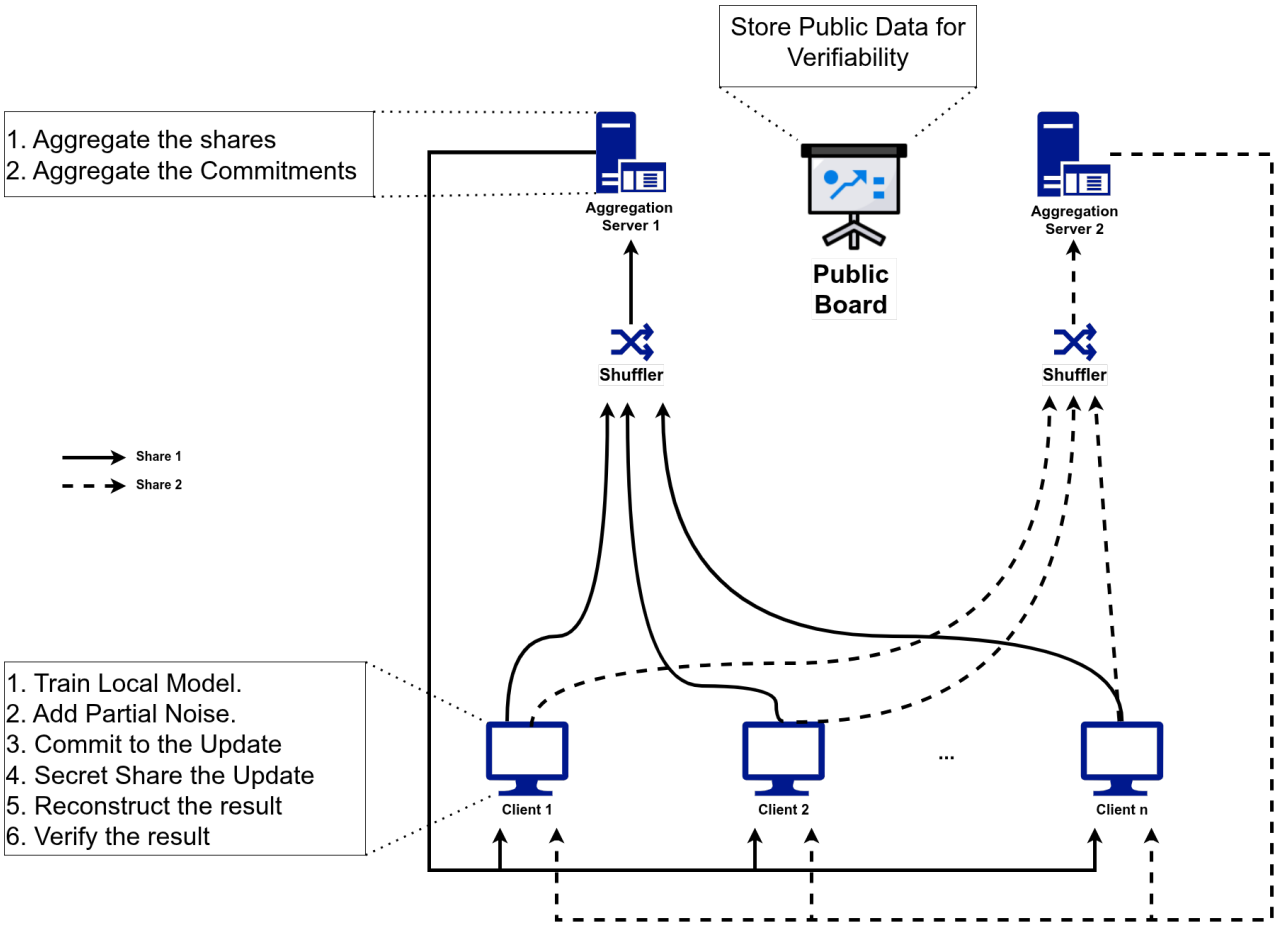


Figure 6.1: High Level Architecture of ProoFed

The framework begins with the deployment of two non-colluding servers that act as independent aggregation entities. Each client trains a local model on its private dataset and adds differential privacy noise to a designated segment of the model update, ensuring that noise contributions are distributed across clients. To preserve integrity without revealing the actual update, every client computes a cryptographic commitment to its local model before applying additive secret sharing. The resulting shares are sent to distinct shufflers, while the corresponding commitments are transmitted in parallel.

The shufflers serve to anonymize communication by dissociating each share from its origin. This operation strengthens privacy guarantees by preventing inference attacks that could exploit timing or metadata correlations. Once shuffling is complete, the shares and commitments are forwarded to the aggregation servers, which independently compute partial sums. The aggregated results are then returned to the clients, who reconstruct the global model and verify its correctness using the combined commitments and the public randomness available on the shared board.

Remark. The architecture can be extended to any number k of independent servers, provided that at least two do not collude. Such a configuration can be realized through Trusted Execution Environments or by dynamically selecting a subset of participating clients to act as aggregation nodes in each training round.

6.2.2 Threat Model and Assumptions

We consider a federated learning setting involving potentially malicious or semi-honest participants, both on the client and server side. Our goal is to secure the training process against leakage, manipulation, and inference, under realistic adversarial conditions. We explicitly address the following threats:

1. **Semi-Honest Clients:** Clients that follow the protocol correctly but may attempt to infer sensitive information about other participants. This can be done either through collusion with a server, or by exploiting the iterative nature of training to infer information from successive global models.

2. **Malicious Server:** A central server may attempt to:

   (a) *Reconstruct local models* from the shares provided by clients.

(b) *Recover individual data points* by manipulating model parameters or aggregations, or through collusion with other clients or servers;

(c) *Tamper with the global model* by omitting updates from selected clients, injecting forged updates, or otherwise altering the aggregated model to serve an adversarial goal.

To address these threats, we make the following assumptions:

- We assume the presence of at least two non-colluding aggregation entities, which may be realized in practice through independent public cloud providers, random subsets of selected clients or Trusted Execution Environments.
- Any communication between clients, shufflers, and servers is assumed to be encrypted.
- While we consider semi-honest clients in our threat model, we do not address actively malicious clients who may submit poisoned updates, conflicting gradients, or malformed commitments.
- We do not model client dropouts, network failures, or asynchronous execution.

6.2.3 Workflow Overview

In this section, we dive into the detailed design of ProoFed, a protocol ensuring that client updates remain hidden, noise is generated collaboratively, and aggregation is verifiable. Our framework, ProoFed, builds a chain of trust which enables strong privacy guarantees while ensuring better model utility. The notations used during this section is described in Appendix D.1.

6.2.3.1 Private Partition of a vector of n elements

As said previously, each client is responsible for adding noise to a specific weight of the model update. To ensure privacy, it is essential that each client knows only its own selected indexes, while no other party can learn them.

To achieve this, we propose a privacy-preserving consensus protocol that allows all clients to select disjoint subsets of coordinates independently, while avoiding overlap and preserving index.

Algorithm 3 Unique Random Coordinate Assignment

Require: n clients $C_1, C_2, \ldots, C_n$, number of random values m, large field $\mathbb{F}$
Ensure: Each client C_i holds m globally unique random numbers

Initialization and Anonymous Sharing:
for each client C_i **do**
  Randomly generate $S_i = \{s_{i,1}, \ldots, s_{i,m}\} \subset \mathbb{F}$
  Send S_i anonymously to the public board B through shufflers
end for
$B \leftarrow \bigcup_{i=1}^{n} S_i$
Duplicate Checking:
repeat
  for each client C_i **do**
    for each $s_{i,j} \in S_i$ **do**
      if $s_{i,j}$ appears more than once in B **then**
        C_i samples a new $s'_{i,j} \in \mathbb{F} \setminus B$
        Update $S_i \leftarrow (S_i \setminus \{s_{i,j}\}) \cup \{s'_{i,j}\}$
        Publish $s'_{i,j}$ on the public board B
      end if
    end for
  end for
until all elements in B are unique
Global Consensus:
All clients agree on B as the globally unique vector
Each client sorts B in ascending order
for each $s_{i,j} \in S_i$ **do**
  Find the position $p_{i,j}$ of $s_{i,j}$ in B
  Record $p_{i,j}$ as a private coordinate
end for

Expected number of iterations for consensus Let n denote the number of clients, each selecting m indices uniformly at random from a finite field $\mathbb{F}$ of size N. The probability of collisions decreases exponentially with N, and when $N \gg nm$, the expected number of iterations until all selected indices are globally unique is close to one.

6.2.3.2 Hiding Local Updates from Adversaries

To protect local updates from adversaries, each client encodes its model update $w_i \in \mathbb{R}^d$ using an additive secret sharing scheme. The client generates k random shares $w_i^{(1)}, w_i^{(2)}, \ldots, w_i^{(k)} \in \mathbb{R}^d$ that satisfy the following formula.

$$w_i = \sum_{j=1}^{k} w_i^{(j)}. \tag{6.1}$$

Each share $w_i^{(j)}$ is transmitted independently to a distinct shuffler, whose task is to anonymize the origin of the shares. This mechanism ensures that no individual server, and no coalition of up to $k-1$ servers, can reconstruct the original update w_i.

Let $\{w_1^{(j)}, w_2^{(j)}, \ldots, w_n^{(j)}\}$ denote the set of shares received by shuffler j. After processing, each shuffler forwards the permuted and delayed shares to its corresponding aggregator, which computes the partial sum as following.

$$\mathbf{w}^{(j)} = \sum_{i=1}^{n} w_i^{(j)}.$$

The global model update is then reconstructed by combining the results from all aggregators:

$$\mathbf{w} = \sum_{j=1}^{k} \mathbf{w}^{(j)} = \sum_{j=1}^{k} \sum_{i=1}^{n} w_i^{(j)} = \sum_{i=1}^{n} \sum_{j=1}^{k} w_i^{(j)} = \sum_{i=1}^{n} w_i.$$

Because each aggregator only observes a partial sum $\mathbf{w}^{(j)}$, it cannot recover any individual client's full update. This construction remains secure against up to $n-1$ colluding servers, as demonstrated in Appendix D.2. The security and anonymity of the protocol further improve as the number of participating clients grows, since the mixing of shares increases entropy and reduces the possibility of inferring or correlating sensitive information across the system.

6.2.3.3 Achieving Differential Privacy

Local updates are already protected through secret sharing, as detailed in Section 6.2.3.2. The remaining challenge is to protect the aggregated global model from inference attacks that may exploit aggregated statistics to extract sensitive information, even when raw updates remain hidden.

To overcome this challenge, a collaborative mechanism is proposed to achieve Central Differential Privacy without relying on a fully trusted server. Each client contributes noise to a specific portion of its update vector $w_i \in \mathbb{R}^d$, ensuring that every coordinate of the global model is noised exactly once. This controlled partitioning of the update space prevents redundant noise accumulation while maintaining consistent privacy guarantees across clients.

The noised local update is defined as

$$\tilde{\mathbf{w}}_i = w_i + e_i$$

where e_i is a sparse noise vector, containing non-zero values only within the client's designated segment. The noise components are sampled from a differentially private distribution to ensure statistical indistinguishability of individual contributions.

Although this design limits the total amount of noise introduced, it does not completely conceal individual updates, since only part of each vector is perturbed and the partitioning scheme remains fixed. For this reason, the differential privacy mechanism must be combined with secret sharing and anonymized communication, as discussed in Section 6.2.3.2, to ensure end-to-end protection of client data.

After the aggregation phase, the cumulative noise is represented as

$$\eta = \sum_{i=1}^{n} e_i$$

and the resulting differentially private global model is expressed as

$$\tilde{\mathbf{w}} = \sum_{i=1}^{n} \tilde{\mathbf{w}}_i = \sum_{i=1}^{n} w_i + \sum_{i=1}^{n} e_i = \mathbf{w} + \eta.$$

This formulation guarantees that each coordinate of the global model receives exactly one instance of noise, thereby maintaining strong differential privacy while preserving model utility.

6.2.3.4 Verifiability Mechanisms

To ensure the correctness of the global aggregation without compromising the privacy of individual updates, we employ homomorphic commitments and then propose a protocol for zero knowledge proof of opening.

One solution that may be interesting is to commit on each weight of the update and send it to the server. However, the limit of such solution is seen directly since it will take a lot of time to generate the commitments and the proofs. What we propose here, is as follows:

Each client computes the mean of all the weights of the local model and then commits to its value using a homomorphic commitment scheme resulting on a commitment $C_i = \text{Commit}(\tilde{\mathbf{w}}_i; r_i)$. The

commitment is then published to the public board while breaking the link between each commitment and its client. We can publicly aggregate the commitments. For our solution, we propose to use Pedersen commitments. As stated in Section 6.2.2, the clients are considered as honest to achieve this part.

At the end of the protocol, the aggregating server sends to the clients the global share of the aggregation, which can be combined by the clients to get the global model update.

Now, we consider n provers $\mathcal{P}_1, \ldots, \mathcal{P}_n$, each holding a private mean of the local update $\mathbf{W}_i \in \mathbb{Z}_q$ along with private randomness $r_i \in \mathbb{Z}_q$.

The goal of the protocol is to allow the provers to jointly prove, in zero knowledge, that a publicly known value $\tilde{\mathbf{w}} = \sum_{i=1}^{n} \tilde{\mathbf{w}}_i$ is indeed the sum of their secret shares, and that their aggregated commitment satisfies:

$$C = \prod_{i=1}^{n} C_i = g^{\tilde{\mathbf{w}}} h^R \quad \text{where } R = \sum_{i=1}^{n} r_i$$

without revealing neither individual $\mathbf{w}_i$ nor r_i.

The process is done in the following steps.

1. **Commitment Phase:** Each prover $\mathcal{P}_i$ samples a random ephemeral value $\rho_i \in \mathbb{Z}_q$ and computes a temporary commitment:

   $$A_i = h^{\rho_i}$$

   These commitments are sent to the public board. The public board aggregates:

   $$A = \prod_{i=1}^{n} A_i = h^{\rho}, \quad \text{with } \rho = \sum_{i=1}^{n} \rho_i$$

2. **Challenge Phase:** A challenge value $c \in \mathbb{Z}_q$ is computed using a cryptographic hash function in the Fiat–Shamir heuristic:

   $$c = H(A, C, \tilde{\mathbf{w}})$$

   where $C = \prod_{i=1}^{n} C_i$ and $\tilde{\mathbf{w}}$ is the publicly known sum of the secret shares.

3. **Response Phase:** Each prover computes their response:

   $$z_i = \rho_i + c \cdot \tilde{\mathbf{w}}_i \mod q$$

and sends it to the coordinator. The coordinator aggregates:

$$z = \sum_{i=1}^{n} z_i = \rho + c \cdot \tilde{\mathbf{w}} \mod q$$

4. **Verification Phase:** The verifier checks that:

$$h^z \stackrel{?}{=} A \cdot (h^{\tilde{\mathbf{w}}})^c$$

and accepts the proof if the equality holds.

Due to practical reasons while working with limited float numbers, the solution can tolerate an error in the computation. This can easily done by verifiying all the neighboring sums are correct.

6.3 Evaluation

The proposed framework, ProoFed, is evaluated through both theoretical analysis and experimental validation.

6.3.1 Trust and Privacy Analysis

This section provides both a theoretical and conceptual analysis of the privacy, confidentiality, and verifiability guarantees offered by ProoFed. The objective is to show that the framework preserves confidentiality of local updates, achieves differential privacy at the global level, and enables verifiable aggregation against potentially malicious or semi-honest participants.

6.3.1.1 Confidentiality of Local Updates

The confidentiality of local updates is ensured through multiple complementary mechanisms: additive secret sharing to protect individual updates, anonymization via share shuffling to break source associations, and partial observability at the aggregation level to prevent any server from reconstructing a client's contribution.

Proposition 6.3.1 (Confidentiality under Additive Secret Sharing)**.** *Let $\mathcal{A} \subset \{1, \ldots, k\}$ denote a subset of at most $k-1$ servers controlled by an adversary, and let $\mathcal{S}_{\mathcal{A}} = \{\mathbf{w}_i^{(j)} : j \in \mathcal{A}\}$ be the shares observed by this adversary. Then: $I(\mathbf{w}_i; \mathcal{S}_{\mathcal{A}}) = 0$.*

In practice, this implies that even if one aggregator and several clients collude, they cannot infer another client's contribution. The anonymity of communication channels further reduces the probability of source re-identification, as validated by Proposition 6.3.2. Shufflers break the association between each share and its origin, as well as between correlated shares belonging to the same client. This randomization disrupts timing and ordering correlations that could otherwise leak identity information.

Proposition 6.3.2 (Anonymity through Shuffling)**.** *Let M denote a message observed on the network and ID the identity of the sending client. After shuffling, the mutual information between these random variables satisfies: $I(M; ID) \approx 0$.*

At any time t, an external observer perceives random, unlinkable messages, thus preventing any meaningful correlation between content and sender. Each aggregator processes only a partial sum of the client updates: $\mathbf{w}^{(j)} = \sum_{i=1}^{n} \mathbf{w}_i^{(j)}$, and the final global model is reconstructed as:

$$\tilde{\mathbf{w}} = \sum_{j=1}^{k} \mathbf{w}^{(j)} = \sum_{i=1}^{n} \mathbf{w}_i.$$

6.3.1.2 Differential Privacy of the Global Model

We now prove that our protocol is differentially private. To keep it simple in explanation, let's consider that we are sampling noise from a Laplace distribution and we want to prove that out protocol maps exactly to the Laplace mechanism applied to the sum of client updates, and thus satisfies ε-differential privacy.

Let:

- $D = \{w_1, \ldots, w_n\}$ be the collection of local updates from n clients, each $w_i \in \mathbb{R}^d$;
- $f(D) = \sum_{i=1}^{n} w_i \in \mathbb{R}^d$ be the global model update function;
- $D \sim D'$ be neighboring datasets differing in one client's update.

For simplicity, we consider our protocol as follows :

1. Each client is responsible for adding Laplace noise to exactly one coordinate j_i of its local vector

w_i, generating a sparse noise vector:

$$Y_i = (0, \ldots, \underbrace{X_i}_{j_i\text{-th position}}, \ldots, 0), \quad \text{with } X_i \sim \text{Lap}\left(\frac{\Delta}{\varepsilon}\right)$$

2. The total noise vector added to the global update is:

$$\eta = \sum_{i=1}^{n} Y_i = (X_1, X_2, \ldots, X_d)$$

where all X_j are mutually independent.

3. The final output of the protocol is:

$$\mathcal{M}(D) = f(D) + \eta = \sum_{i=1}^{n} x_i + \sum_{i=1}^{n} Y_i$$

Each coordinate of η is a sample from $\text{Lap}(\Delta/\varepsilon)$, and the components are independent. Therefore, $\mathcal{M}(D)$ is exactly the Laplace mechanism as defined above.

6.3.1.3 Verifiability of Aggregation

We now prove that our aggregation verification protocol satisfies completeness, soundness, and zero-knowledge. Here, we use the term server to designate all the aggregators.

Definition 1 (Completeness) If all clients are honest and the server correctly aggregates their committed updates, then the verifier always accepts the proof.

Proof. Assume each client i publishes commitments of the form $C_i = g^{w_i} h^{r_i}$ and the server computes:

$$C = \prod_{i=1}^{n} C_i = g^w h^R, \quad \text{with } w = \sum_{i=1}^{n} w_i, \quad R = \sum_{i=1}^{n} r_i.$$

Every client collaborates to publicly sample a random $\rho \in \mathbb{Z}_q$, computes $A = h^\rho$, derives the challenge $c = H(A, C, w)$, and publishes with $z_i = \rho + cw_i \mod q$ in order to publicly reconstruct $z = \rho + cw \mod q$. The verifier checks whether:

$$h^z \stackrel{?}{=} A \cdot (h^w)^c.$$

This holds because:

$$h^z = h^{\rho + cw} = h^\rho \cdot h^{cw} = A \cdot (h^w)^c.$$

Hence, the verification succeeds when the aggregation is correct. □

Definition 2 (Soundness) If the server miscomputes the aggregate $w' \neq \sum w_i$, then it cannot produce a valid proof, except with negligible probability.

Proof. Suppose the server claims a false sum w', attempting to convince the verifier that:

$$C = g^{w'} h^{R'} \quad \text{with } C = \prod_{i=0}^{n} C_i = g^w h^R, \text{ but } w' \neq w.$$

Due to the binding property of Pedersen commitments under the discrete logarithm assumption, such a representation is infeasible.

To forge a valid proof, the server must have control on A and z such that:

$$h^z = A \cdot (h^{w'})^e = h^{\rho + cw'}.$$

But this implies:

$$z = \rho + cw' \neq \rho + cw,$$

and hence $h^z \neq A \cdot (h^w)^c$, which causes verification to fail.

Since c is derived using the Fiat–Shamir heuristic after A is fixed, the server cannot adaptively choose a fake z. Therefore, soundness holds. □

Definition 3 (Zero-Knowledge) For any coalition of at most $k-1$ (dishonest) servers and any verifier, there exists a simulator that, given only the public inputs w and C, can produce a proof transcript that is computationally indistinguishable from a real proof execution, without access to any secret shares $w_i^{(j)}$ or randomness r_i.

Proof. A simulator Sim can generate a valid-looking transcript (A, c, z) without access to any secrets as follows:

1. Sample $c \leftarrow \mathbb{Z}_q$, $z \leftarrow \mathbb{Z}_q$ uniformly.
2. Compute $A = h^z \cdot (h^w)^{-c} = h^{z-cw}$.
3. Output (A, c, z).

This simulated transcript satisfies the verification equation: $h^z = A \cdot (h^w)^c$. Because the challenge c is computed via a hash function modeled as a random oracle, the simulated transcript is computationally

indistinguishable from one generated in a real execution. Additionally, Pedersen commitments are perfectly hiding, ensuring that no information about w_i is leaked through C_i. □

6.3.2 Experimental Results

We evaluate ProoFed under multiple experimental settings as summarized in Table 6.1, and compare its performance against three baseline approaches: vanilla Federated Learning, Local Differential Privacy, and Central Differential Privacy.

All experiments were implemented in PyTorch using the Flower federated learning framework and executed on a workstation equipped with an AMD Threadripper PRO 7955WX processor, 250 GB of RAM, and two NVIDIA RTX 6000 GPUs.

The evaluation examines several key aspects of the proposed framework. We first study the convergence behavior and the resulting model accuracy under different privacy budgets, highlighting the trade-off between privacy protection and utility. The computational overhead introduced by the verification mechanisms is also measured to assess efficiency and scalability. Furthermore, we investigate the impact of the number of participating clients on both performance and system load. Finally, we evaluate how the framework scales with increasing model complexity and size, providing a comprehensive assessment of ProoFed's practical applicability.

Table 6.1: Summary of datasets and experimental configuration.

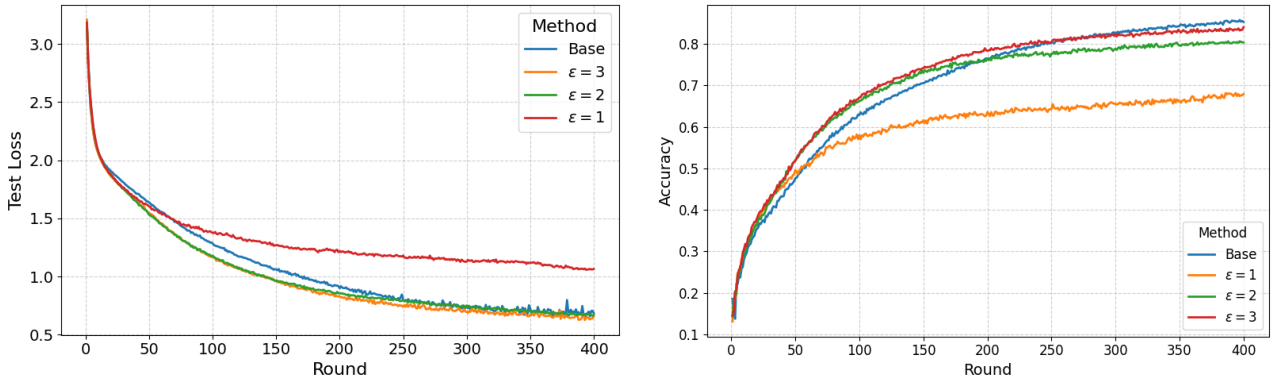
Dataset	**Task Type**	**Model Used**	**#Clients**	**#Rounds**
CIFAR-10	Multi-class	ResNet18/34/50	50 / 70 / 100 / 120	400
CIFAR-100	Multi-class	ResNet18	120	500
Adult Income	Binary classification	Fully connected	70	100
MiniImagenet	Multi-class	ResNet18	120	600

Training configuration: All experiments use the Adam optimizer with a learning rate of 0.001, a local epoch setting of 1, a batch size of 32 per client and a Gaussian Mechanism with $\epsilon \in \{1, 2, 3\}$ and $\delta = 10^{-3}$.

6.3.2.1 Impact of Training Rounds

We use the CIFAR-10 dataset with 120 clients, combined with data augmentation to increase data variability and enable each client to simulate subsampling by working with different data instances in each round. A ResNet-18 model is used, and all clients participate in every round.

Figure 6.2 presents the model performance across communication rounds for various privacy levels, showing that even with noise injection the model converges. Notably, Proofed can achieve comparable performance to the setting without privacy as shown in Figure 4 for the $\epsilon = 3.0$.



(a) Loss vs Rounds.

(b) Accuracy vs Rounds.

Figure 6.2: Performance of the model on CIFAR10 across different privacy budgets (ε) under IID settings (Number of Clients=120)

We acknowledge that the privacy budget considered in these experiments are high but if we consider amplification by subsampling and shuffling, the amplification is the same as the one seen in Chapter 3.

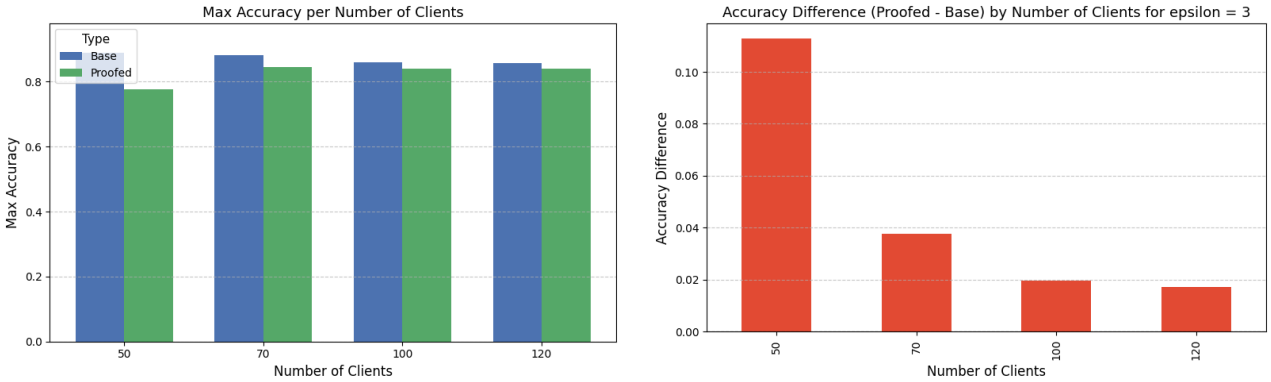
6.3.2.2 Effect of the number of clients

In this experiment, we investigate how varying the number of participating clients affects the performance and efficiency of our protocol. The setup uses the CIFAR-10 dataset with a variable number of clients ranging from 50 to 120.

Impact on Model Performance. Figures 6.3a and 6.3b illustrate the outcomes. We observe that having more clients generally improves the model's performance and reduces the gap between the performance of our approach and the baseline without any privacy. This improvement is due to the

aggregation process, where the variance of the noise is effectively reduced by a factor of n^2, thus improving on model utility.

- **Figure 6.3a** shows the maximum accuracy achieved when changing the number of clients while using a fixed privacy budget of $\varepsilon = 3.0$.
- **Figure 6.3b** presents the performance gap between our method and the baseline. The gap decreases as the number of clients increases.



(a) Max accuracy per number of clients

(b) Difference in accuracy (Proofed - Base)

Figure 6.3: Effect of the number of clients on the utility of the model.

Impact on Server Overhead. We also analyze the computational overhead on the server side as the number of clients increases. Figure 6.4 shows the time per round including training aggregation, proofing and client training times. Although the runtime increases with the number of clients, the additional cost introduced by the verification protocol remains minimal and stable. This demonstrates that our verification mechanism scales efficiently.

Figure 6.4 demonstrates that the more client we have the more it takes time in each round. Notably, we see that the verification process adds minimal overhead on the server side.

6.3.2.3 Effect of Model Size on Computation Time

In this experiment, we examine how the size of the model affects the overall computation time. We use again the CIFAR-10 dataset with a fixed number of 120 clients, 400 communication rounds, and identical training configurations.

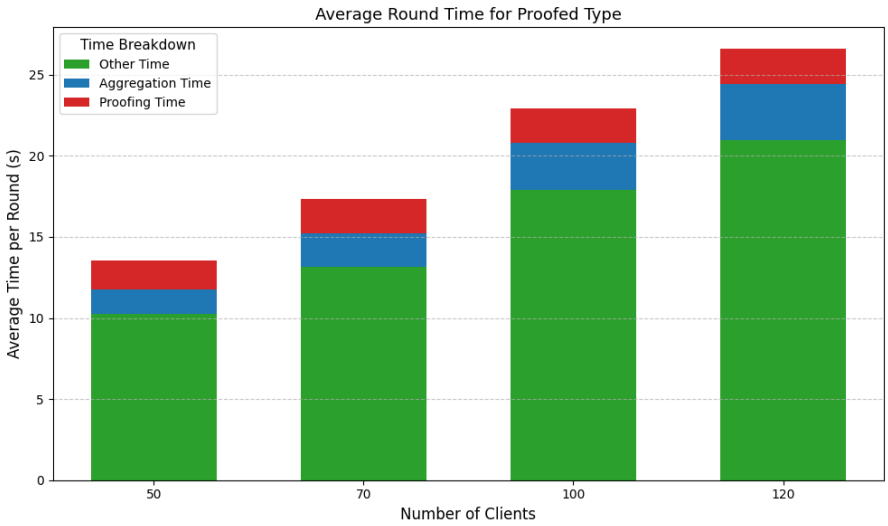


Figure 6.4: Server overhead across increasing numbers of clients.

We evaluate our protocol using multiple variants of the model architecture: GoogleNet (6 millions), ResNet-18 (11.7 millions), ResNet-34 (21.8 millions). These architectures differ in their number of parameters which allows us to analyze scalabiliy according to the model complexity.

Figure 6.5 shows the impact of increasing model size on the total execution time per round. As expected, the overall training time increases with the model complexity. However, in the server side there is no effect on the verification time of our protocol. This is due to the fact that each client uses only the mean of its model in the process of verification.

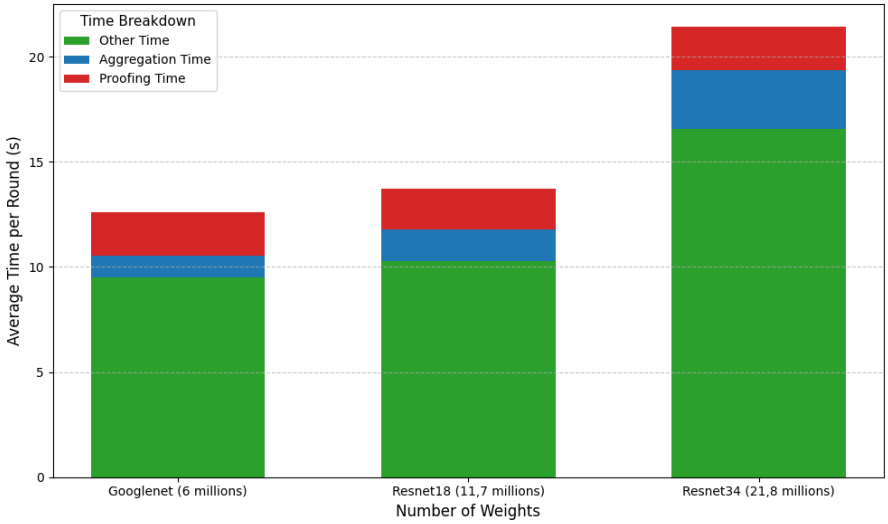


Figure 6.5: Effect of model size on the execution time on Cifar10.

6.3.2.4 Comparison Across Configurations

We report accuracy under different privacy budgets $\varepsilon \in \{\infty, 3.0, 2.0, 1.0\}$. Table 6.2 summarizes the results across four datasets: CIFAR-10, CIFAR-100, Adult Income, and Mini-ImageNet.

The results confirm that our method offers a comparable accuracy to the CDP setting. Compared to LDP, which suffers from high utility loss, and CDP, which assumes a trusted server, our method maintains both privacy and model utility without relying on trust assumptions.

Table 6.2: Accuracy (%) across different privacy levels and datasets.

Dataset	**Setting**	**Method**	$\varepsilon = \infty$	$\varepsilon = 3.0$	$\varepsilon = 2.0$	$\varepsilon = 1.0$
CIFAR-10	Clients: 120 Rounds: 400 Loc-Epochs: 1	LDP CDP ProoFed	85.33 85.33 85.33	20.03 83.77 84.06	15.58 80.49 80.34	15.04 69.07 67.91
CIFAR-100	Clients: 120 Rounds: 500 Loc-Epochs: 1	LDP CDP ProoFed	49.54 49.54 49.54	1.82 49.67 46.58	2.11 41.08 38.85	1.44 6.91 8.05
Adult Income	Clients: 70 Rounds: 100 Loc-Epochs: 1	LDP CDP ProoFed	83.82 83.82 83.82	32.09 80.23 81.41	25.24 78.71 77.61	63.33 72.48 70.18
Mini-ImageNet	Clients: 120 Rounds: 600 Loc-Epochs: 1	LDP CDP ProoFed	45.43 45.43 45.43	1.94 44.83 44.36	1.62 40.71 40.44	1.46 13.76 13.54

To complement this analysis, Figure 6.6 compares the average client-side execution time per round across all configurations. As expected, our method incurs higher latency than the LDP setting, since local differential privacy requires only local noise addition and involves no verification process. The CDP configuration also exhibits minimal client-side overhead, as all privacy-related computations are delegated to the server. In contrast, our approach introduces a moderate increase in execution time due to the use of secret sharing and the generation of zero-knowledge proofs. Nevertheless, the overall computational cost remains practical and well within acceptable limits for real-world federated learning deployments.

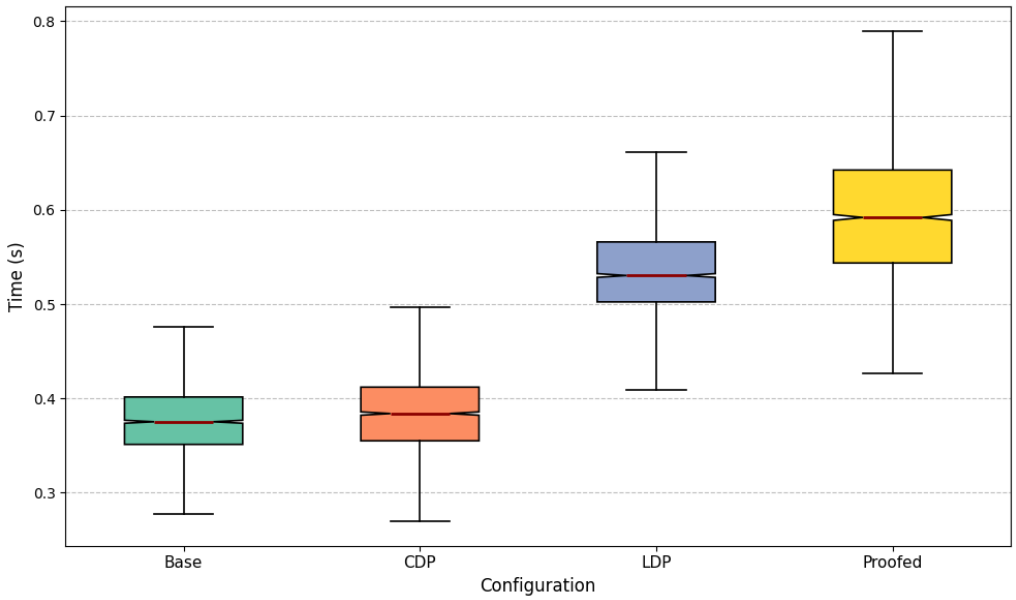


Figure 6.6: Average client-side execution time per configuration.

6.4 Discussion and Limitations

The ProoFed framework combines secret sharing, zero-knowledge proofs, and differential privacy to strengthen the security and trustworthiness of the federated learning process. We evaluated its effectiveness through extensive experiments, assessing its performance, scalability, and efficiency.

First, we analyzed the impact of the number of training rounds, and our results demonstrate that the proposed scheme converges reliably to an optimal model, with only a minimal trade-off between privacy and utility. Furthermore, incorporating data subsampling to our solution improves privacy guarantees while enabling prolonged training over multiple rounds.

Second, we studied the influence of the number of participating clients. As expected, increasing the number of clients results in a reduction of the gap between the privacy-preserving setup and the baseline (non-private) setup, while maintaining equivalent formal privacy guarantees. Moreover, the computational overhead introduced by our framework remains minimal, both on the client and server sides, thereby supporting its practical applicability.

In addition, we demonstrated that the verification mechanism on the server side is independent of the model size, as each client transmits a constant-size commitment for model validation. While training and aggregation times do scale with model complexity, this behavior is inherent to the

federated learning framework.

Finally, we compared our framework with the standard Central Differential Privacy baseline. Our results indicate comparable utility levels, with the added advantage of removing the trust assumption on the central server. This improvement comes at the cost of increased execution time relative to CDP alone.

Real World Applicability. To adapt our protocol within existing federated learning frameworks, only minor modifications are required. First, a public board accessible by all clients must be instantiated. This board serves as a medium for publishing commitments and reaching consensus. At each round, an additional step is inserted to allow clients to negotiate consensus on their masked local weights, following the protocol described in Section 6.2.3.1. Second, the client logic must be extended to generate verifiable updates, including commitments and associated zero-knowledge proofs. The aggregation of commitments can be performed either through the public board or by the server. If the aggregation is handled by the server, its logic must also be extended to combine the commitments posted by clients. In our experiments, we simulated the public board using a shared file accessible by all processes and implemented commitment aggregation directly in the server. The required modifications are lightweight and do not require to trust a third party.

Limitations of the solution. Our proposed framework has a few limitations that must be acknowledged, with the potential to be addressed in future work to achieve full robustness. First, the verifiable protocol introduces a non-negligible computational overhead on the client side, due to the generation of cryptographic commitments, masks, and zero-knowledge proofs. While this overhead remains lightweight compared to homomorphic encryption, it may still be a problem for resource-constrained devices. Second, our protocol currently focuses on the verifiability of aggregation and does not include detection mechanisms for malicious clients injecting poisoned updates, conflicting gradients, or sybil attacks. However, it is easy to replace FedAvg to another technique taking into account poisoning attacks like RSA[142]. Third, we assume that all selected clients participate synchronously in each round, which may not hold in practice, especially in mobile or edge scenarios with unstable connectivity. Finally, we do not empirically evaluate the robustness of our system to client drop out or network failures. A real-world deployment would require fault-tolerant mechanisms, such as timeouts or fallback

aggregation paths, to ensure resilience under partial client dropout.

6.5 Conclusion

This chapter introduced ProoFed, a comprehensive framework designed to reconcile privacy, verifiability, and scalability in federated learning. The proposed approach combines additive secret sharing, central differential privacy, and zero-knowledge proofs, thereby removing the need to trust a central server while preserving the overall utility of the global model.

We demonstrated that ProoFed ensures the confidentiality of local updates, enables verifiable aggregation, and maintains a balanced trade-off between computational cost and model performance. Experimental evaluations across multiple datasets confirmed the robustness and practicality of our framework, showing that it achieves comparable accuracy to traditional centralized differential privacy while offering stronger trust guarantees.

Overall, this chapter represents a key step toward building a federated learning paradigm that is secure, transparent, and verifiable, laying the foundation for trustworthy distributed AI systems.

Key Takeaways

- ProoFed achieves privacy, trust, and scalability in Federated Learning by combining Central Differential Privacy, Additive Secret Sharing, and Zero-Knowledge Proof of Opening.
- The design removes the need for a fully trusted server by distributing both noise generation and aggregation verification among clients and non-colluding servers.
- Confidentiality of local updates, verifiability of aggregation, and tradeoffs utility-privacy reduction are simultaneously achieved, as confirmed by theoretical analysis and empirical evaluation.
- The proposed proving protocol remains lightweight and independent of model size. However, the number of participants are adding overhead.
- The framework currently assumes synchronous participation and honest clients, extending it to tolerate client dropouts and adversarial behaviors is part of future work.

Part IV

General Conclusion

Chapter 7

General Conclusion and Perspectives

Contents

7.1 Summary of contributions

This thesis started from a fundamental observation: federated learning, although designed to protect data by keeping it decentralized, does not guarantee privacy or trust by design. Gradients exchanged between clients and the server can still leak sensitive information, and the entire training process often depends on strong and unrealistic trust assumptions regarding the honesty of the central aggregator. At the same time, privacy regulations such as the GDPR require not only that personal data remain protected, but also that the protection mechanisms themselves be verifiable and auditable. This creates a gap between what current FL systems provide and what real-world deployments actually require.

In Chapter 3, we presented a state of the art on differential privacy and cryptographic methods in FL. This review placed differential privacy as the only mechanism capable of preventing information leakage from the global model in federated learning, while also highlighting its privacy–utility trade-off. The chapter then discussed how cryptographic approaches could reduce trust assumptions and, consequently, mitigate this trade-off. This analysis established the foundation for the combined approaches developed in the later chapters. Therefore, the main problem addressed in this thesis is how to ensure that federated learning preserves privacy even when the aggregation server and participants are not fully trusted, and how to provide verifiable evidence that differential privacy mechanisms have been correctly applied.

In Chapter 4, we proposed to reduce the trust placed in the server using homomorphic encryption. Unlike previous works, the proposed framework does not aggregate the entire model in the encrypted domain. Importantly, this design allows the application of Theorem 7 from Erlingsson et al. [62], which provides privacy amplification by shuffling in the general context, not only in the relaxed setting described by their Corollary 9. Moreover, the framework supports both central and local DP configurations. It also enables secure aggregation while removing the computational overhead caused by pairwise masking exchanges, thus handling client dropouts since masks are generated after participation. While this solution effectively separates noise addition from aggregation, it does not yet include verification of the correct application of differential privacy.

To address this gap, Chapter 5 tackled the lack of auditability in existing privacy mechanisms. We introduced a non-interactive proof protocol that verifies the correct application of differential privacy

without revealing data or noise values. The protocol combines zk-SNARKs with cryptographic hash commitments, enabling public verification of DP compliance by any participant or external auditor. This contribution demonstrated that verifiability and differential privacy can coexist, and that privacy guarantees can be made auditable in practice. However, it also revealed limitations in scalability when combined with the previous approach from Chapter 4, due to the heavy computational overhead of using two key-based cryptographic mechanisms simultaneously. In particular, zk-SNARKs are not easily adaptable to varying numbers of clients across rounds, and key generation through multi-party computation adds significant complexity.

These challenges motivated the exploration of alternative verification techniques in Chapter 8. We proposed ProoFed, a generalized framework designed to overcome the previous limitations. ProoFed distributes the generation of differential privacy noise among clients using additive secret sharing, thereby removing the need for a trusted central entity and enabling verifiable, privacy-preserving aggregation within a fully distributed setting.

7.2 Global Discussion

The results presented in the previous chapters collectively highlight the evolution of this work toward a federated learning framework that integrates privacy, trust reduction, and verifiability in a coherent manner. Each approach contributes a different layer of assurance, and their comparison reveals how design choices interact across the axes of trust, verifiability, scalability, and utility. The overall observation is that improving one of these properties inevitably affects the others, and the strength of a given mechanism depends on how well it balances this multi-objective trade-off. In particular, cryptographic primitives effectively reduce trust assumptions but introduce computational and communication overhead, while verifiability mechanisms improve auditability at the cost of additional complexity. The challenge, therefore, is not to maximize a single property but to reach a sustainable equilibrium that maintains privacy guarantees without sacrificing system practicality. This balance gradually emerged through the successive designs developed in the thesis, leading to a final architecture that reconciles privacy and verifiability within an efficient distributed setting. Table 7.1 summarizes the comparative characteristics of the three frameworks proposed across the thesis.

Table 7.1: Comparison of the three proposed frameworks across key dimensions.

Dimension	**Chapter 4**	**Chapter 5**	**Chapter 8**
Server Trust	Not Trusted	Semi Trusted	Not Trusted
Local Model Privacy	Yes (HE)	No	Yes (SS)
Global Model Privacy	Yes	Yes	Yes
Verifiability	None	zk-SNARKs	ZKP of Commitment Opening
Scalability	Low	Moderate	Improved

7.3 Future Research Directions

The contributions of this thesis open several promising research directions for extending the proposed verifiable and privacy-preserving frameworks toward more realistic, interpretable, and accountable federated learning systems. While the present work focused on synchronous, privacy-centered architectures, future research should address the following key challenges.

7.3.1 Asynchronous Verifiable and Differentially Private Federated Learning

The framework proposed in Chapter 8 assumes synchronous participation of all clients in each training round, thereby excluding communication-related challenges from its scope. However, real-world federated learning systems are inherently asynchronous: clients may join or leave dynamically, operate under heterogeneous communication rates, and experience intermittent connectivity.

Extending the proposed framework to an asynchronous setting would require revisiting both the privacy mechanisms and the verification protocols to accommodate partial participation, delayed updates, and straggler effects. A promising direction involves designing Timing-Safe Release Protocols that coordinate the disclosure of model updates without compromising either privacy or verifiability under asynchronous communication.

7.3.2 Human-Centric and Interpretable Privacy Guarantees

As discussed throughout this thesis, the privacy budget ε was treated as a variable. While differential privacy offers formal mathematical guarantees, the practical meaning of ε remains abstract and difficult to communicate to stackholders. This limitation does not make the concept unusable, it motivates the development of more interpretable privacy frameworks.

An analogous challenge exists in artificial intelligence, where deep learning models are often viewed as "black boxes" yet remain useful despite their limited interpretability. Similarly, future research should aim to make privacy guarantees more transparent and comprehensible. One promising direction is the notion of explainable privacy, where the numerical value of ε is translated into intuitive notions of risk or data exposure. In practice, this could involve empirically linking ε to measurable quantities such as the success probability of inference attacks, reconstruction risk, or expected data leakage. Developing such human-centered interpretations would greatly facilitate stakeholder understanding and adoption of differential privacy mechanisms.

7.3.3 Verifiable Tracking of Privacy Budget in Federated Learning

A key limitation of this work lies in the absence of explicit tracking of cumulative privacy loss across training rounds. While this aspect was neglecting in the current thesis, it represents a critical avenue for future exploration.

Actually, existing privacy accounting methods such as the Moment Accountant [82] and RDP Accountant [143] struggle to model the dynamic and heterogeneous nature of federated environments. Moreover, they often rely on relaxed definitions of differential privacy. Future work should focus on designing verifiable privacy composition methods that can account for heterogeneity and partial participation while integrating privacy amplification by iteration [63] and privacy amplification by subsampling [144].

An additional open problem concerns continual federated learning, where models evolve over time or across data streams. In such settings, privacy loss may accumulate not only across training rounds but also across model versions and tasks. Addressing this challenge requires developing temporal and continual privacy frameworks capable of reasoning about long-term and cross-task exposure.

7.3.4 Future Extensions of the Proposed Frameworks

Building upon the architectures and experimental results proposed in this thesis, several extensions could further enhance the practicality, robustness, and scalability of verifiable and privacy-preserving federated learning systems:

- **Robustness Against Poisoning Attacks:** The current frameworks assume benign client behavior.

Future work should focus on defending against adversarial contributions such as model poisoning, backdoors, or data manipulation. Integrating robust and verifiable aggregation rules would enhance both trust and security. However, this remains challenging due to the statistical nature of model update filtering, which can be difficult to reconcile with cryptography. Actually, adapting the robust aggregation method named RSA[142] seems the easiest way.

- **Verifiability on the Client Side:** To achieve end-to-end trust, verifiability should extend beyond the server to include client-side verification. In Chapter 5, the proposed zk-SNARK-based framework demonstrated how succinct non-interactive proofs can ensure the integrity and accountability of the aggregation process. However, zk-SNARKs require a trusted setup phase involving key generation, which introduces potential security and transparency concerns. Future research should therefore explore alternative proof systems such as zk-STARKs [110], which offer comparable succinctness and non-interactivity while eliminating the need for key negotiation and improving auditability through transparent setup procedures. Building upon the architecture presented in Chapter 8, another promising direction involves designing distributed proofs to construct a fully self-verifiable federated learning framework, where verification tasks are decentralized among clients.

- **Experimental Expansion:** Further empirical validation is required under adversarial and resource-constrained conditions to assess the robustness and practicality of the proposed frameworks. Although this thesis employed a simplified public bulletin board for result verification, future research could investigate more sophisticated implementations of this component, such as blockchain-based solutions. A decentralized ledger would enhance transparency, immutability, and resistance to tampering, thereby strengthening the auditability and trustworthiness of federated learning processes. In parallel, experimental evaluations should consider scenarios involving poisoning and backdoor attacks, communication cost analysis, and scalability testing across heterogeneous client populations.

Part V

Résumé étendu de la thèse

Chapter 8

Résumé

Contents

8.1 Introduction générale et problématique

8.1.1 Contexte général

L'essor des techniques d'apprentissage automatique a profondément transformé de nombreux domaines applicatifs, tels que la santé, la finance, les systèmes de recommandation ou encore les services numériques personnalisés. Cette évolution repose en grande partie sur la capacité des modèles à exploiter des volumes massifs de données, souvent sensibles, issues de sources hétérogènes et distribuées. Toutefois, la centralisation de ces données soulève d'importantes préoccupations en matière de confidentialité, de sécurité et de conformité réglementaire.

Ces préoccupations ont conduit à l'adoption de cadres juridiques de plus en plus stricts visant à encadrer la collecte, le traitement et l'exploitation des données personnelles. Des réglementations telles que le Règlement Général sur la Protection des Données (RGPD) en Europe, le California Consumer Privacy Act (CCPA) aux États-Unis ou encore le Health Insurance Portability and Accountability Act (HIPAA) dans le domaine de la santé imposent des exigences fortes en matière de minimisation des données, de limitation des finalités, de transparence et de responsabilité. Ces textes ne se limitent pas à restreindre les flux de données, mais exigent également que les organisations soient en mesure de démontrer, de manière vérifiable et auditable, le respect des principes de protection de la vie privée.

Dans ce contexte, l'apprentissage fédéré (*Federated Learning*) est apparu comme un paradigme prometteur permettant d'entraîner des modèles d'apprentissage automatique de manière collaborative sans nécessiter le partage des données brutes. En conservant les données sur les sites locaux et en ne communiquant que des mises à jour de modèles, cette approche semble à première vue mieux alignée avec les exigences réglementaires, en réduisant la centralisation et l'exposition directe des données sensibles.

Cependant, de nombreux travaux récents ont montré que cette hypothèse de confidentialité par conception est insuffisante. Les gradients et paramètres échangés lors de l'apprentissage fédéré peuvent révéler des informations sur les jeux de données locaux, rendant le système vulnérable à diverses attaques par inférence. Dès lors, si l'apprentissage fédéré constitue une avancée importante vers des architectures plus respectueuses de la vie privée, il ne répond pas à lui seul aux exigences légales et réglementaires en matière de protection des données, qui impliquent non seulement la limitation des fuites d'information, mais également la capacité à prouver et à auditer les garanties de confidentialité

effectivement mises en œuvre.

Pour pallier les fuites d'information observées dans l'apprentissage fédéré, la confidentialité différentielle s'est imposée comme l'un des rares mécanismes offrant des garanties formelles de protection de la vie privée, y compris après la publication du modèle entraîné. Toutefois, son intégration dans les systèmes d'apprentissage fédéré met en évidence plusieurs limites structurelles. L'ajout de bruit introduit un compromis délicat entre confidentialité et utilité des modèles, pouvant dégrader significativement leurs performances lorsque des niveaux élevés de protection sont requis.

Par ailleurs, la majorité des approches reposant sur la confidentialité différentielle supposent l'existence d'un serveur central honnête ou semi-honnête, chargé d'appliquer correctement les mécanismes de protection. Cette hypothèse de confiance forte constitue une faiblesse majeure, notamment dans des environnements multi-organisations ou soumis à des exigences réglementaires strictes. En pratique, les participants ne disposent d'aucun moyen effectif pour vérifier que les mécanismes annoncés sont correctement implémentés et exécutés.

8.1.2 Problématique et objectifs

Au-delà de la question de la confidentialité, un enjeu central concerne la transparence et l'auditabilité des systèmes d'apprentissage fédéré. L'absence de mécanismes de vérification empêche les participants, ainsi que les autorités de régulation, de s'assurer du respect effectif des garanties de confidentialité, créant un écart significatif entre les promesses théoriques et leur mise en œuvre réelle.

Dans ce contexte, la problématique centrale de cette thèse peut être formulée comme suit : *comment concevoir des systèmes d'apprentissage fédéré capables de fournir des garanties formelles de confidentialité, tout en réduisant les hypothèses de confiance envers les entités centrales et en permettant la vérification effective de la bonne application des mécanismes de protection de la vie privée ?*

Pour répondre à cette problématique, cette thèse explore la combinaison de la confidentialité différentielle avec des primitives cryptographiques et des protocoles de vérifiabilité. L'objectif est de proposer des cadres intégrés permettant à la fois de protéger les mises à jour locales, de limiter la confiance accordée au serveur central, et de rendre vérifiables les propriétés de confidentialité annoncées, sans divulguer d'informations sensibles. Le présent chapitre de résumé étendu vise ainsi à

synthétiser l'ensemble des travaux présentés dans ce manuscrit, en mettant en évidence leur cohérence et leur progression scientifique.

8.2 Enseignements de l'etat de l'art

L'apprentissage fédéré (*Federated Learning*) a été introduit comme une alternative aux architectures d'apprentissage centralisées afin de permettre l'entraînement collaboratif de modèles d'apprentissage automatique tout en conservant les données sur les sites locaux. Dans ce paradigme, un serveur central orchestre le processus d'apprentissage en distribuant un modèle global aux participants, lesquels effectuent un entraînement local avant de renvoyer des mises à jour agrégées pour former un nouveau modèle global.

Cette approche présente plusieurs avantages, notamment la réduction des risques liés à la centralisation des données et une meilleure conformité aux réglementations en matière de protection de la vie privée. Toutefois, malgré ces bénéfices apparents, l'apprentissage fédéré introduit de nouvelles contraintes liées à la distribution des données, à l'hétérogénéité des participants et à la nature itérative des échanges de paramètres. Ces contraintes exposent le système à des vulnérabilités spécifiques qui ne sont pas présentes dans les approches centralisées.

8.2.1 Menaces sur la vie privée dans l'apprentissage fédéré

Contrairement à une idée largement répandue, le fait de ne pas partager les données brutes ne garantit pas une protection complète de la vie privée. Les mises à jour de modèles échangées dans l'apprentissage fédéré peuvent contenir des informations sensibles exploitables par des adversaires. Plusieurs types d'attaques par inférence ont ainsi été mis en évidence, notamment les attaques par inférence d'appartenance, les attaques par reconstruction de données ou encore l'inférence de propriétés statistiques des jeux de données locaux.

Ces attaques exploitent la nature en boîte blanche de l'apprentissage fédéré, où les gradients et paramètres sont directement accessibles aux entités participantes ou au serveur central. Elles montrent clairement que l'apprentissage fédéré ne constitue qu'une première couche de protection et qu'il ne peut, à lui seul, satisfaire les exigences élevées de confidentialité requises dans des domaines sensibles tels que la santé ou la finance.

8.2.2 Confidentialité différentielle dans l'apprentissage fédéré

Face à ces limitations, la confidentialité différentielle s'est imposée comme un mécanisme fondamental pour fournir des garanties formelles de protection de la vie privée. Elle repose sur l'introduction d'un bruit aléatoire contrôlé afin de limiter l'impact de toute donnée individuelle sur le résultat final du modèle. Cette propriété garantit qu'un adversaire ne peut pas déterminer, avec une probabilité significative, si une donnée spécifique a été utilisée lors de l'entraînement.

Dans le contexte de l'apprentissage fédéré, la confidentialité différentielle peut être appliquée selon différents modèles, notamment la confidentialité différentielle centrale et la confidentialité différentielle locale. Chacun de ces modèles présente des avantages et des inconvénients en termes de protection, de coût computationnel et d'utilité des modèles. Les travaux existants ont montré que l'application naïve de la confidentialité différentielle entraîne souvent une dégradation notable des performances, en particulier lorsque des niveaux de confidentialité élevés sont exigés.

8.2.3 Hypothèses de confiance et absence de vérifiabilité

Un autre aspect critique mis en évidence par l'état de l'art concerne les hypothèses de confiance sous-jacentes aux systèmes d'apprentissage fédéré. La majorité des approches existantes supposent un serveur central honnête ou semi-honnête, chargé d'appliquer correctement les mécanismes de confidentialité différentielle et d'effectuer l'agrégation des mises à jour de manière fidèle. Cette hypothèse est cependant incompatible avec des scénarios réalistes impliquant des entités indépendantes ou concurrentes.

De plus, les mécanismes de confidentialité proposés dans la littérature sont rarement accompagnés de moyens permettant d'en vérifier l'application effective. L'absence de vérifiabilité empêche les participants et les autorités de régulation de s'assurer que les garanties annoncées sont respectées, renforçant ainsi la dépendance à une confiance implicite envers le serveur central.

8.2.4 Enjeux ouverts et positionnement

L'analyse de l'état de l'art met en évidence un écart persistant entre les garanties théoriques offertes par les mécanismes de confidentialité et leur mise en œuvre pratique dans les systèmes d'apprentissage fédéré. Les approches existantes traitent souvent la confidentialité, la confiance et la vérifiabilité

comme des problématiques indépendantes, sans proposer de cadre unifié permettant de les aborder conjointement.

C'est dans ce contexte que s'inscrit cette thèse, qui vise à dépasser les limitations identifiées en explorant la combinaison de la confidentialité différentielle avec des primitives cryptographiques et des protocoles de vérifiabilité. L'objectif est de proposer des approches capables de réduire les hypothèses de confiance, de rendre les garanties de confidentialité auditables et de préserver l'utilité des modèles, ouvrant ainsi la voie à des systèmes d'apprentissage fédéré plus transparents et mieux adaptés aux exigences des environnements sensibles.

8.3 Contributions et approches proposées

Cette section synthétise les principales contributions de cette thèse, développées dans les Chapitres 4, 5 et 6. L'objectif commun de ces contributions est de réduire les hypothèses de confiance dans les systèmes d'apprentissage fédéré, tout en garantissant des propriétés formelles de confidentialité et en introduisant des mécanismes de vérifiabilité permettant d'auditer le processus d'apprentissage.

8.3.1 Réduction de la confiance par chiffrement homomorphe

La première contribution de cette thèse, développée au Chapitre 4, vise à réduire les hypothèses de confiance envers le serveur central dans les systèmes d'apprentissage fédéré, en particulier dans le contexte de la confidentialité différentielle centrale. Comme mis en évidence dans l'état de l'art, confier au serveur l'ajout du bruit de confidentialité différentielle constitue une faiblesse majeure, puisqu'un serveur curieux peut potentiellement neutraliser ce bruit après l'agrégation et compromettre les garanties de confidentialité.

Pour répondre à cette limitation, un cadre reposant sur le chiffrement homomorphe additif est proposé, permettant de protéger les mises à jour locales tout au long de leur transmission et de leur agrégation. L'architecture introduite repose sur une séparation explicite des rôles entre trois entités : les clients, un shuffler et un agrégateur. Les clients chiffrent leurs mises à jour locales à l'aide d'une clé publique avant transmission, tandis que le shuffler opère exclusivement sur des données chiffrées afin d'assurer à la fois l'anonymisation des contributions et l'ajout du bruit de confidentialité différentielle.

Grâce aux propriétés du chiffrement homomorphe additif, l'agrégation peut être effectuée sans

accès aux valeurs sous-jacentes, ce qui empêche l'agrégateur d'observer les mises à jour non protégées. Cette conception établit un découplage clair entre l'agrégation des mises à jour et l'application de la confidentialité différentielle, limitant ainsi les capacités d'observation et d'analyse du serveur central dans le modèle de menace considéré. Elle permet également d'exploiter des mécanismes d'amplification de la confidentialité, notamment par sous-échantillonnage dans le cadre central et par mélange des paramètres dans le cadre local, sans imposer de contraintes restrictives sur le nombre de participants ou sur le budget de confidentialité.

Les analyses théoriques et expérimentales présentées dans ce chapitre montrent que cette approche permet de renforcer significativement la confidentialité des mises à jour locales et l'anonymat des contributions, tout en maintenant des performances de convergence comparables à celles des approches classiques d'apprentissage fédéré sous des niveaux de bruit modérés. Cette contribution constitue ainsi une première étape vers des architectures d'apprentissage fédéré moins dépendantes de la confiance, tout en mettant en évidence les limites pratiques liées au coût du chiffrement homomorphe, qui motivent l'exploration de solutions complémentaires dans les chapitres suivants.

8.3.2 Vérifiabilité non interactive de la confidentialité différentielle

La deuxième contribution de cette thèse, développée au Chapitre 5, s'attaque à une limitation fondamentale des mécanismes de confidentialité différentielle en pratique : l'absence de moyens permettant de vérifier que le bruit requis par la confidentialité différentielle est effectivement généré et appliqué conformément aux paramètres annoncés. Dans le cadre de la confidentialité différentielle centrale, cette absence de vérifiabilité impose une hypothèse de confiance forte envers le serveur, lequel peut intentionnellement ou non affaiblir les garanties de confidentialité en modifiant le budget de confidentialité, en réduisant le bruit injecté ou en contournant le mécanisme prévu.

Pour répondre à cette problématique, cette thèse introduit un cadre de vérifiabilité non interactive reposant sur l'intégration de preuves à divulgation nulle de connaissance (zk-SNARKs) et d'engagements cryptographiques au sein du processus de confidentialité différentielle. Le principe consiste à associer à chaque étape critique — génération du bruit, exécution de la requête et ajout du bruit au résultat — une preuve cryptographique attestant de sa conformité, sans révéler ni les données sensibles ni les valeurs de bruit utilisées. Afin de rendre ces preuves compatibles avec des circuits zk-SNARKs, la confidentialité différentielle repose sur le mécanisme de Laplace discret, dont la génération peut être

efficacement vérifiée à l'aide de preuves de minimalité et de composition.

Ce cadre permet ainsi de transformer la confidentialité différentielle, traditionnellement fondée sur des hypothèses de confiance implicites, en une propriété vérifiable a posteriori. Les clients et les propriétaires de données peuvent vérifier publiquement que les paramètres de confidentialité annoncés ont été respectés, sans recourir à des audits ex post ni à des techniques d'inférence approximatives. Toutefois, cette approche reste limitée par le coût computationnel des zk-SNARKs et par son périmètre fonctionnel, restreint aux requêtes décomposables et à des déploiements de taille modérée. Ces limitations motivent l'exploration de solutions plus légères et plus distribuées, développées dans la contribution suivante.

8.3.3 ProoFed : un cadre unifié, distribué et vérifiable

La troisième contribution de cette thèse, développée au Chapitre 6, propose *ProoFed*, un cadre d'apprentissage fédéré visant à concilier confidentialité, vérifiabilité et réduction des hypothèses de confiance, tout en évitant le recours à des primitives cryptographiques lourdes telles que le chiffrement homomorphe ou les zk-SNARKs. Cette contribution s'inscrit dans la continuité des limites identifiées dans les chapitres précédents, en particulier la dépendance à des entités centrales de confiance et les surcoûts de calcul associés aux approches existantes.

ProoFed repose sur une architecture distribuée combinant le partage de secrets additif, la génération collaborative du bruit de confidentialité différentielle et des mécanismes d'engagement cryptographique. Les mises à jour locales sont découpées en parts distribuées entre plusieurs serveurs non collusifs, de sorte qu'aucune entité isolée ne puisse reconstruire les contributions individuelles. La confidentialité différentielle centrale est obtenue de manière collaborative, chaque client ajoutant du bruit à une portion spécifique de son vecteur de mise à jour, garantissant que chaque coordonnée du modèle global est bruitée exactement une fois, tout en limitant l'accumulation inutile de bruit.

Afin de renforcer la confiance dans le processus d'agrégation, ProoFed intègre des mécanismes de vérifiabilité fondés sur des engagements homomorphes et des preuves à divulgation nulle de connaissance. Ces mécanismes permettent de vérifier que l'agrégation globale correspond bien à la somme des mises à jour engagées par les clients, sans révéler les contributions individuelles ni les valeurs de bruit utilisées. La vérifiabilité porte ainsi sur la correction de l'agrégation et le respect du protocole, sans supposer un comportement honnête du serveur.

Les analyses théoriques et les résultats expérimentaux présentés dans ce chapitre montrent que ProoFed atteint des performances comparables à celles de la confidentialité différentielle centrale classique, tout en supprimant l'hypothèse d'un serveur pleinement digne de confiance. Cette contribution illustre la faisabilité d'un apprentissage fédéré à la fois privé et vérifiable, au prix de compromis raisonnables en termes de complexité protocolaire et de synchronisation, ouvrant la voie à des systèmes d'apprentissage fédéré plus transparents et mieux adaptés aux environnements sensibles.

8.3.4 Synthèse comparative des contributions

Pris dans leur ensemble, les trois cadres proposés dans cette thèse abordent de manière progressive les limites identifiées dans l'état de l'art. La première contribution réduit l'exposition des mises à jour locales, la seconde introduit la vérifiabilité des garanties de confidentialité, et la troisième combine ces dimensions dans une architecture distribuée et cohérente.

Cette progression permet d'explorer différents compromis entre confidentialité, confiance, vérifiabilité et efficacité, et fournit une base solide pour la conception de systèmes d'apprentissage fédéré mieux adaptés aux exigences des environnements sensibles et réglementés.

8.4 Validation expérimentale, discussion et perspectives

Cette section synthétise les résultats expérimentaux obtenus pour les différentes approches proposées dans cette thèse, avant d'en dégager les principaux enseignements, les limites identifiées et les perspectives de recherche ouvertes.

8.4.1 Cadre expérimental général

Les différentes contributions de cette thèse ont été évaluées à l'aide de cadres expérimentaux adaptés à leurs objectifs respectifs, visant à analyser l'impact des mécanismes proposés sur la confidentialité, l'utilité des modèles et les coûts computationnels. Les expériences ont été menées sur plusieurs jeux de données représentatifs et des modèles d'apprentissage profond couramment utilisés dans la littérature, afin de refléter des scénarios d'apprentissage fédéré réalistes.

Les évaluations prennent en compte des paramètres clés tels que le nombre de participants, le nombre de tours d'entraînement, la taille des modèles et les niveaux de confidentialité différentielle.

Les métriques analysées incluent principalement la précision des modèles, la vitesse de convergence, ainsi que les temps de calcul et de communication côté client et côté serveur. Une attention particulière est également portée aux surcoûts induits par les mécanismes cryptographiques et de vérifiabilité intégrés aux différents cadres proposés.

8.4.2 Résultats expérimentaux et enseignements

Les résultats expérimentaux montrent que les approches proposées permettent de préserver une utilité satisfaisante des modèles tout en renforçant significativement les garanties de confidentialité et de confiance. Dans le cadre de la réduction de la confiance par chiffrement homomorphe, les expériences confirment que la protection des mises à jour locales et la séparation des responsabilités n'empêchent pas la convergence des modèles, sous réserve de niveaux de bruit modérés et d'un nombre de participants suffisant. Ces résultats mettent cependant en évidence des limites claires en termes de coûts computationnels, en particulier lorsque la taille des modèles ou le nombre de clients augmente.

En ce qui concerne la vérifiabilité non interactive de la confidentialité différentielle, les expériences démontrent la faisabilité pratique de l'intégration de preuves cryptographiques attestant de la bonne application du bruit, avec un coût de vérification faible et indépendant de la taille des données. Le surcoût principal réside dans la génération des preuves, dont la complexité croît avec le nombre de participants, ce qui limite la scalabilité de cette approche dans des déploiements à très grande échelle.

Les résultats obtenus avec le cadre ProoFed mettent en évidence l'intérêt d'une approche distribuée de la confidentialité différentielle et de la vérifiabilité. La génération collaborative du bruit et l'utilisation du partage de secrets permettent de réduire significativement les hypothèses de confiance, tout en conservant des performances comparables à celles de la confidentialité différentielle centrale classique. Les expériences confirment également que les mécanismes de vérifiabilité intégrés introduisent un surcoût limité et indépendant de la taille des modèles, ce qui renforce la pertinence de cette approche pour des environnements d'apprentissage fédéré réalistes.

8.4.3 Discussion globale et limites

L'analyse transversale des résultats met en évidence les compromis inhérents aux systèmes d'apprentissage fédéré préservant la vie privée. Le renforcement des garanties de confidentialité, de réduction de la confiance et de vérifiabilité s'accompagne nécessairement de coûts supplémentaires

en termes de calcul, de communication et de complexité protocolaire. Le choix d'un mécanisme donné dépend donc fortement des contraintes et des exigences du domaine applicatif ciblé.

Par ailleurs, les contributions de cette thèse se concentrent sur des scénarios d'apprentissage fédéré synchrones et reposant sur des hypothèses de participation honnête des clients. Des problématiques telles que l'asynchronisme, les déconnexions fréquentes, l'hétérogénéité marquée des données ou les attaques actives de type empoisonnement ne sont pas traitées de manière approfondie. Ces limites, volontairement assumées, définissent le cadre d'application des résultats présentés et motivent les perspectives de recherche identifiées.

8.4.4 Perspectives de recherche

Les travaux présentés ouvrent plusieurs perspectives de recherche. Une première direction concerne l'extension des mécanismes proposés à des scénarios asynchrones ou plus fortement décentralisés, afin de mieux refléter les contraintes des environnements réels à grande échelle. Une autre perspective porte sur le suivi, la composition et la vérification dynamique du budget de confidentialité différentielle au fil des itérations d'apprentissage, en lien avec les exigences réglementaires et les cadres d'auditabilité émergents.

Enfin, l'intégration des mécanismes de confidentialité et de vérifiabilité à des approches visant à améliorer l'interprétabilité, la transparence et l'acceptabilité des modèles constitue une piste prometteuse pour favoriser l'adoption de l'apprentissage fédéré dans des domaines sensibles nécessitant un haut niveau de confiance, de responsabilité et de conformité réglementaire.

8.5 Conclusion du résumé étendu

Ce chapitre a présenté un résumé étendu des travaux menés dans cette thèse, en mettant en évidence la progression scientifique adoptée pour répondre aux limites de l'apprentissage fédéré en matière de confidentialité, de confiance et de transparence. Partant du constat que la décentralisation des données ne suffit pas à garantir la protection de la vie privée, cette thèse a exploré des mécanismes complémentaires visant à rendre les garanties de confidentialité effectives, vérifiables et compatibles avec des contraintes pratiques.

Les contributions présentées montrent que la confidentialité différentielle demeure le seul mécanisme

capable de protéger formellement le modèle global contre les attaques par inférence, mais que son efficacité dépend fortement des hypothèses de confiance accordées aux entités centrales. À travers l'utilisation du chiffrement homomorphe, de preuves cryptographiques et de mécanismes distribués fondés sur le partage de secrets, cette thèse propose plusieurs stratégies permettant de réduire ces hypothèses, d'introduire des garanties de vérifiabilité et d'améliorer la transparence des processus d'apprentissage.

Les résultats expérimentaux et les analyses comparatives mettent en évidence les compromis inhérents entre confidentialité, utilité des modèles, vérifiabilité et coûts computationnels. Aucune approche ne permet de maximiser simultanément toutes ces dimensions, mais les cadres proposés montrent qu'il est possible d'atteindre un équilibre satisfaisant en combinant de manière raisonnée des mécanismes cryptographiques et des techniques de confidentialité différentielle.

En conclusion, ce résumé étendu met en lumière que l'avenir de l'apprentissage fédéré préservant la vie privée repose sur des architectures hybrides et distribuées, capables de réconcilier garanties formelles, auditabilité et contraintes opérationnelles. Les travaux présentés constituent une étape vers des systèmes d'apprentissage fédéré plus fiables, plus responsables et mieux alignés avec les exigences réglementaires et sociétales actuelles.

Bibliography

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

[2] N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramer, and C. Zhang, "Quantifying memorization across neural language models," in *The Eleventh International Conference on Learning Representations*, 2022.

[3] C. Song, T. Ristenpart, and V. Shmatikov, "Machine learning models that remember too much," in *Proceedings of the 2017 ACM SIGSAC Conference on computer and communications security*, 2017, pp. 587–601.

[4] J. Barron and D. White, "Too big to think: Capacity, memorization, and generalization in pre-trained transformers," *arXiv preprint arXiv:2506.09099*, 2025.

[5] J. X. Morris, C. Sitawarin, C. Guo, N. Kokhlikyan, G. E. Suh, A. M. Rush, K. Chaudhuri, and S. Mahloujifar, "How much do language models memorize?" *arXiv preprint arXiv:2505.24832*, 2025.

[6] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[7] J. Hun Ro, "Fedjax: Federated learning simulation with jax," Oct 2021. [Online]. Available: https://blog.research.google/2021/10/fedjax-federated-learning-simulation.html

[8] S. I. Nanayakkara, S. R. Pokhrel, and G. Li, "Understanding global aggregation and optimization of federated learning," *Future Generation Computer Systems*, vol. 159, pp. 114–133, 2024.

[9] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, M. Raykova, H. Qi, D. Ramage, R. Raskar, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and Open Problems in Federated Learning," Mar. 2021, arXiv:1912.04977 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1912.04977

[10] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *CoRR*, vol. abs/1902.04885, 2019. [Online]. Available: http://arxiv.org/abs/1902.04885

[11] Y. Liu, T. Fan, T. Chen, Q. Xu, and Q. Yang, "Fate: an industrial grade platform for collaborative learning with data protection," *J. Mach. Learn. Res.*, vol. 22, no. 1, Jan. 2021.

[12] "GitHub - FederatedAI/FATE: An Industrial Grade Federated Learning Framework — github.com," https://github.com/FederatedAI/FATE, [Accessed 08-09-2025].

[13] "TensorFlow Federated — tensorflow.org," https://www.tensorflow.org/federated?hl=fr, [Accessed 08-09-2025].

[14] "GitHub - google-parfait/tensorflow-federated: An open-source framework for machine learning and other computations on decentralized data. — github.com," https://github.com/google-parfait/tensorflow-federated, [Accessed 08-09-2025].

[15] G. A. Reina, A. Gruzdev, P. Foley, O. Perepelkina, M. Sharma, I. Davidyuk, I. Trushkin, M. Radionov, A. Mokrov, D. Agapov *et al.*, "Openfl: An open-source framework for federated learning," *arXiv preprint arXiv:2105.06413*, 2021.

[16] A. Ziller, A. Trask, A. Lopardo, B. Szymkow, B. Wagner, E. Bluemke, J.-M. Nounahon, J. Passerat-Palmbach, K. Prakash, N. Rose *et al.*, "Pysyft: A library for easy federated learning," in *Federated learning systems: Towards next-generation AI*. Springer, 2021, pp. 111–139.

[17] "GitHub - OpenMined/PySyft: Perform data science on data that remains in someone else's server — github.com," https://github.com/OpenMined/PySyft, [Accessed 08-09-2025].

[18] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. De Gusmão *et al.*, "Flower: A friendly federated learning research framework," *arXiv preprint arXiv:2007.14390*, 2020.

[19] "GitHub - adap/flower: Flower: A Friendly Federated AI Framework — github.com," https://github.com/adap/flower, [Accessed 08-09-2025].

[20] P. Liu, X. Xu, and W. Wang, "Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives," *Cybersecurity*, vol. 5, no. 1, p. 4, Feb. 2022. [Online]. Available: https://doi.org/10.1186/s42400-021-00105-6

[21] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning," in *2019 IEEE Symposium on Security and Privacy (SP)*, May 2019, pp. 739–753, arXiv:1812.00910 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1812.00910

[22] U. Gupta, D. Stripelis, P. K. Lam, P. Thompson, J. L. Ambite, and G. Ver Steeg, "Membership inference attacks on deep regression models for neuroimaging," in *Medical Imaging with Deep Learning*. PMLR, 2021, pp. 228–251.

[23] J. Li, N. Li, and B. Ribeiro, "Effective passive membership inference attacks in federated learning against overparameterized models," in *The Eleventh International Conference on Learning Representations*, 2023.

[24] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership Inference Attacks against Machine Learning Models," Mar. 2017, arXiv:1610.05820 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1610.05820

[25] J. Zhang, J. Zhang, J. Chen, and S. Yu, "Gan enhanced membership inference: A passive local attack in federated learning," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[26] A. Luqman, A. Chattopadhyay, and K.-Y. Lam, "Membership inference vulnerabilities in peer-to-peer federated learning," in *Proceedings of the 2023 Secure and Trustworthy Deep Learning Systems Workshop*, 2023, pp. 1–5.

[27] J. Chen, J. Zhang, Y. Zhao, H. Han, K. Zhu, and B. Chen, "Beyond model-level membership privacy leakage: an adversarial approach in federated learning," in *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, 2020, pp. 1–9.

[28] X. Yuan, X. Ma, L. Zhang, Y. Fang, and D. Wu, "Beyond Class-Level Privacy Leakage: Breaking Record-Level Privacy in Federated Learning," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2555–2565, Feb. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9456909/

[29] Y. Gu, Y. Bai, and S. Xu, "CS-MIA: Membership inference attack based on prediction confidence series in federated learning," *Journal of Information Security and Applications*, vol. 67, p. 103201, Jun. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212622000801

[30] H. Hu, Z. Salcic, L. Sun, G. Dobbie, and X. Zhang, "Source Inference Attacks in Federated Learning," Sep. 2021, arXiv:2109.05659 [cs]. [Online]. Available: http://arxiv.org/abs/2109.05659

[31] G. Pichler, M. Romanelli, L. R. Vega, and P. Piantanida, "Perfectly accurate membership inference by a dishonest central server in federated learning," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 4290–4296, 2023.

[32] A. Suri, P. Kanani, V. J. Marathe, and D. W. Peterson, "Subject Membership Inference Attacks in Federated Learning," Sep. 2022, arXiv:2206.03317 [cs]. [Online]. Available: http://arxiv.org/abs/2206.03317

[33] W. Zhang, S. Tople, and O. Ohrimenko, "Leakage of dataset properties in Multi-Party machine learning," in *30th USENIX Security Symposium (USENIX Security 21)*. USENIX Association, Aug. 2021, pp. 2687–2704. [Online]. Available: https://www.usenix.org/conference/usenixsecurity21/presentation/zhang-wanrong

[34] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: Information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 603–618. [Online]. Available: https://doi.org/10.1145/3133956.3134012

[35] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019*

- *IEEE Conference on Computer Communications*, 2019, p. 2512–2520. [Online]. Available: https://doi.org/10.1109/INFOCOM.2019.8737416

[36] L. Melis, C. Song, E. D. Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," 2018.

[37] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 619–633. [Online]. Available: https://doi.org/10.1145/3243734.3243834

[38] J. Zhou, Y. Chen, C. Shen, and Y. Zhang, "Property inference attacks against gans," *CoRR*, vol. abs/2111.07608, 2021. [Online]. Available: https://arxiv.org/abs/2111.07608

[39] J. Liu, B. Chen, B. Xue, M. Guo, and Y. Xu, "Piafgnn: Property inference attacks against federated graph neural networks." *Computers, Materials & Continua*, vol. 82, no. 2, 2025.

[40] D. Yao, S. Li, X. Gong, S. Hou, and G. Pan, "Urvfl: Undetectable data reconstruction attack on vertical federated learning," *arXiv preprint arXiv:2404.19582*, 2024.

[41] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/60a6c4002cc7b29142def8871531281a-Paper.pdf

[42] B. Zhao, K. R. Mopuri, and H. Bilen, "idlg: Improved deep leakage from gradients," *arXiv preprint arXiv:2001.02610*, 2020.

[43] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients - how easy is it to break privacy in federated learning?" in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 16 937–16 947. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/c4ede56bbd98819ae6112b20ac6bf145-Paper.pdf

[44] H. Ren, J. Deng, and X. Xie, "Grnn: Generative regression neural network—a data leakage attack for federated learning," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 4, may 2022. [Online]. Available: https://doi.org/10.1145/3510032

[45] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through Gradients: Image Batch Recovery via GradInversion," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN, USA: IEEE, Jun. 2021, pp. 16 332–16 341. [Online]. Available: https://ieeexplore.ieee.org/document/9577731/

[46] J. C. Zhao, A. Sharma, A. R. Elkordy, Y. H. Ezzeldin, S. Avestimehr, and S. Bagchi, "Loki: Large-scale data reconstruction attack against federated learning through model manipulation," in *2024 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2024, pp. 1287–1305.

[47] Y. Song, Z. Wang, and E. Zuazua, "Approximate and weighted data reconstruction attack in federated learning," *arXiv preprint arXiv:2308.06822*, 2023.

[48] Y. Gao, Y. Xie, H. Deng, and Z. Zhu, "Gradient inversion attack in federated learning: Exposing text data through discrete optimization," in *Proceedings of the 31st International Conference on Computational Linguistics*, 2025, pp. 2582–2591.

[49] V. Valadi, M. Åkesson, J. Östman, S. Toor, and A. Hellander, "From research to reality: Feasibility of gradient inversion attacks in federated learning," *arXiv preprint arXiv:2508.19819*, 2025.

[50] L. Bai, H. Hu, Q. Ye, H. Li, L. Wang, and J. Xu, "Membership inference attacks and defenses in federated learning: A survey," *ACM Computing Surveys*, vol. 57, no. 4, pp. 1–35, 2024.

[51] W. Yuan, C. Yang, Q. V. H. Nguyen, L. Cui, T. He, and H. Yin, "Interaction-level membership inference attack against federated recommender systems," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 1053–1062.

[52] L. Zhang, L. Li, X. Li, B. Cai, Y. Gao, R. Dou, and L. Chen, "Efficient membership inference attacks against federated learning via bias differences," in *Proceedings of the 26th international symposium on research in attacks, intrusions and defenses*, 2023, pp. 222–235.

[53] M. Fredrikson, S. Jha, and T. Ristenpart, "Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. Denver Colorado USA: ACM, Oct. 2015, pp. 1322–1333. [Online]. Available: https://dl.acm.org/doi/10.1145/2810103.2813677

[54] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, and G. Felici, "Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers," *Int. J. Secur. Netw.*, vol. 10, no. 3, p. 137–150, sep 2015. [Online]. Available: https://doi.org/10.1504/IJSN.2015.071829

[55] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Advances in Cryptology - EUROCRYPT 2006*, S. Vaudenay, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 486–503.

[56] J. Tang, A. Korolova, X. Bai, X. Wang, and X. Wang, "Privacy loss in apple's implementation of differential privacy on macos 10.12," 2017. [Online]. Available: https://arxiv.org/abs/1709.02753

[57] B. Jayaraman and D. Evans, "Evaluating differentially private machine learning in practice," 2019. [Online]. Available: https://arxiv.org/abs/1902.08874

[58] A. Lowy, Z. Li, J. Liu, T. Koike-Akino, K. Parsons, and Y. Wang, "Why does differential privacy with large epsilon defend against practical membership inference attacks?" 2024. [Online]. Available: https://arxiv.org/abs/2402.09540

[59] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*. Springer, 2006, pp. 265–284.

[60] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and trends® in theoretical computer science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[61] T. Steinke, "Privacy amplification by subsampling," https://differentialprivacy.org/subsampling/, 2025, [Accessed 12-09-2025].

[62] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta, "Amplification by shuffling: From local to central differential privacy via anonymity," in *Proceedings*

of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms. SIAM, 2019, pp. 2468–2479.

[63] V. Feldman, I. Mironov, K. Talwar, and A. Thakurta, "Privacy amplification by iteration," in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2018, pp. 521–532.

[64] W.-N. Chen, D. Song, A. Ozgur, and P. Kairouz, "Privacy amplification via compression: Achieving the optimal privacy-accuracy-communication trade-off in distributed mean estimation," *Advances in Neural Information Processing Systems*, vol. 36, pp. 69 202–69 227, 2023.

[65] E. Cyffers and A. Bellet, "Privacy amplification by decentralization," in *International conference on artificial intelligence and statistics*. PMLR, 2022, pp. 5334–5353.

[66] M. A. P. Chamikara, D. Liu, S. Camtepe, S. Nepal, M. Grobler, P. Bertok, and I. Khalil, "Local Differential Privacy for Federated Learning," Aug. 2022, arXiv:2202.06053 [cs]. [Online]. Available: http://arxiv.org/abs/2202.06053

[67] C. Wang, X. Wu, G. Liu, T. Deng, K. Peng, and S. Wan, "Safeguarding cross-silo federated learning with local differential privacy," *Digital Communications and Networks*, vol. 8, no. 4, pp. 446–454, Aug. 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352864821000961

[68] P. Lai, H. Phan, L. Xiong, K. Tran, M. Thai, T. Sun, F. Dernoncourt, J. Gu, N. Barmpalios, and R. Jain, "Bit-aware randomized response for local differential privacy in federated learning," 2022.

[69] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized Federated Learning With Differential Privacy," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9530–9539, Oct. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9082603/

[70] L. Ni, P. Huang, Y. Wei, M. Shu, and J. Zhang, "Federated Learning Model with Adaptive Differential Privacy Protection in Medical IoT," *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1–14, Nov. 2021. [Online]. Available: https://www.hindawi.com/journals/wcmc/2021/8967819/

[71] L. Sun, J. Qian, and X. Chen, "LDP-FL: Practical Private Aggregation in Federated Learning with Local Differential Privacy," May 2021, arXiv:2007.15789 [cs]. [Online]. Available: http://arxiv.org/abs/2007.15789

[72] X. Shen, H. Jiang, Y. Chen, B. Wang, and L. Gao, "PLDP-FL: Federated Learning with Personalized Local Differential Privacy," *Entropy*, vol. 25, no. 3, p. 485, Mar. 2023, number: 3 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/1099-4300/25/3/485

[73] L. Cui and X. Wu, "Aldp-fl for adaptive local differential privacy in federated learning," *Scientific Reports*, vol. 15, no. 1, p. 26679, 2025.

[74] S. Kiani, N. Kulkarni, A. Dziedzic, S. Draper, and F. Boenisch, "Differentially private federated learning with time-adaptive privacy spending," *arXiv preprint arXiv:2502.18706*, 2025.

[75] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol. 12, no. null, p. 1069–1109, Jul. 2011.

[76] O. Choudhury, A. Gkoulalas-Divanis, T. Salonidis, I. Sylla, Y. Park, G. Hsu, and A. Das, "Differential privacy-enabled federated learning for sensitive health data," *CoRR*, vol. abs/1910.02578, 2019. [Online]. Available: http://arxiv.org/abs/1910.02578

[77] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private language models without losing accuracy," *CoRR*, vol. abs/1710.06963, 2017. [Online]. Available: http://arxiv.org/abs/1710.06963

[78] R. C. Geyer, T. Klein, and M. Nabi, "Differentially Private Federated Learning: A Client Level Perspective," Mar. 2018, arXiv:1712.07557 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1712.07557

[79] G. Andrew, O. Thakkar, B. McMahan, and S. Ramaswamy, "Differentially private learning with adaptive clipping," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 455–17 466, 2021.

[80] X. Zhang, X. Chen, M. Hong, Z. S. Wu, and J. Yi, "Understanding clipping for federated learning: Convergence and client-level differential privacy," in *International Conference on Machine Learning, ICML 2022*, 2022.

[81] X. Yuan, W. Ni, M. Ding, K. Wei, J. Li, and H. V. Poor, "Amplitude-varying perturbation for balancing privacy and utility in federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1884–1897, 2023.

[82] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[83] J. Fu, Y. Hong, X. Ling, L. Wang, X. Ran, Z. Sun, W. H. Wang, Z. Chen, and Y. Cao, "Differentially private federated learning: A systematic review," 2024. [Online]. Available: https://arxiv.org/abs/2405.08299

[84] N. Agarwal, A. T. Suresh, F. Yu, S. Kumar, and H. B. Mcmahan, "cpsgd: Communication-efficient and differentially-private distributed sgd," 2018. [Online]. Available: https://arxiv.org/abs/1805.10559

[85] L. Wang, R. Jia, and D. Song, "D2p-fed: Differentially private federated learning with efficient communication," *arXiv preprint arXiv:2006.13039*, 2020.

[86] P. Kairouz, Z. Liu, and T. Steinke, "The distributed discrete gaussian mechanism for federated learning with secure aggregation," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5201–5212.

[87] N. Agarwal, P. Kairouz, and Z. Liu, "The skellam mechanism for differentially private federated learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5052–5064, 2021.

[88] W.-N. Chen, A. Ozgur, and P. Kairouz, "The poisson binomial mechanism for unbiased federated learning with secure aggregation," in *International Conference on Machine Learning*. PMLR, 2022, pp. 3490–3506.

[89] N. Wang, X. Xiao, Y. Yang, J. Zhao, S. C. Hui, H. Shin, J. Shin, and G. Yu, "Collecting and analyzing multidimensional data with local differential privacy," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 638–649.

[90] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, and K.-Y. Lam, "Local differential privacy-based federated learning for internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8836–8853, 2020.

[91] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy," in *Proceedings of the third ACM international workshop on edge systems, analytics and networking*, 2020, pp. 61–66.

[92] J. Zhao, M. Yang, R. Zhang, W. Song, J. Zheng, J. Feng, and S. Matwin, "Privacy-enhanced federated learning: A restrictively self-sampled and data-perturbed local differential privacy method," *Electronics*, vol. 11, no. 23, p. 4007, 2022.

[93] M. Varun, S. Feng, H. Wang, S. Sural, and Y. Hong, "Towards accurate and stronger local differential privacy for federated learning with staircase randomized response," in *Proceedings of the Fourteenth ACM Conference on Data and Application Security and Privacy*, 2024, pp. 307–318.

[94] A. Girgis, D. Data, and S. Diggavi, "Renyi Differential Privacy of The Subsampled Shuffle Model In Distributed Learning," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 29 181–29 192. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/f44ec26e2ac3f1ab8c2472d4b1c2ea86-Abstract.html

[95] A. M. Girgis, D. Data, and S. Diggavi, "Differentially Private Federated Learning with Shuffling and Client Self-Sampling," in *2021 IEEE International Symposium on Information Theory (ISIT)*. Melbourne, Australia: IEEE, Jul. 2021, pp. 338–343. [Online]. Available: https://ieeexplore.ieee.org/document/9517906/

[96] A. Girgis, D. Data, S. Diggavi, P. Kairouz, and A. T. Suresh, "Shuffled Model of Differential Privacy in Federated Learning," in *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. PMLR, Mar. 2021, pp. 2521–2529, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v130/girgis21a.html

[97] S. Wang, K. Gai, J. Yu, L. Zhu, H. Wu, C. Wei, Y. Yan, H. Zhang, and K.-K. R. Choo, "Rafls: Rdp-based adaptive federated learning with shuffle model," *IEEE Transactions on Dependable and Secure Computing*, 2024.

[98] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014, pp. 1054–1067.

[99] A. Blanco-Justicia, D. Sánchez, J. Domingo-Ferrer, and K. Muralidhar, "A critical review on the use (and misuse) of differential privacy in machine learning," *ACM Computing Surveys*, vol. 55, no. 8, pp. 1–16, 2022.

[100] L. Del Vasto-Terrientes, D. Sánchez, and J. Domingo-Ferrer, "Differential privacy in practice: Lessons learned from 10 years of real-world applications," *IEEE Security & Privacy*, 2025.

[101] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology—EUROCRYPT'99: International Conference on the Theory and Application of Cryptographic Techniques Prague, Czech Republic, May 2–6, 1999 Proceedings 18*. Springer, 1999, pp. 223–238.

[102] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, p. 120–126, Feb. 1978. [Online]. Available: https://doi-org.proxybib-pp.cnam.fr/10.1145/359340.359342

[103] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, ser. STOC '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 169–178. [Online]. Available: https://doi.org/10.1145/1536414.1536440

[104] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, p. 612–613, Nov. 1979. [Online]. Available: https://doi-org.proxybib-pp.cnam.fr/10.1145/359168.359176

[105] K. Momin. (2023, Apr) A non-mathematical introduction to zero knowledge proof. Accessed: 2025-10-22. [Online]. Available: https://kayprasla.medium.com/a-non-mathematical-introduction-to-zero-knowledge-proof-c1a4a269e308

[106] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Advances in Cryptology — CRYPTO' 86*, A. M. Odlyzko, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1987, pp. 186–194.

[107] J. Eberhardt and S. Tai, "Zokrates - scalable privacy-preserving off-chain computations," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2018, pp. 1084–1091.

[108] J. Groth, "On the size of pairing-based non-interactive arguments," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2016, pp. 305–326.

[109] A. Gabizon, Z. J. Williamson, and O. Ciobotaru, "Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge," *Cryptology ePrint Archive*, 2019.

[110] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable, transparent, and post-quantum secure computational integrity," *Cryptology ePrint Archive*, 2018.

[111] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: Short proofs for confidential transactions and more," in *2018 IEEE symposium on security and privacy (SP)*. IEEE, 2018, pp. 315–334.

[112] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Advances in Cryptology — CRYPTO '91*, J. Feigenbaum, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 129–140.

[113] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "HybridAlpha: An Efficient Approach for Privacy-Preserving Federated Learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, Nov. 2019, pp. 13–23, arXiv:1912.05897 [cs]. [Online]. Available: http://arxiv.org/abs/1912.05897

[114] M. A. Heikkilä, A. Koskela, K. Shimizu, S. Kaski, and A. Honkela, "Differentially private cross-silo federated learning," *arXiv preprint arXiv:2007.05553*, 2020.

[115] T. Wang, B. Ding, M. Xu, Z. Huang, C. Hong, J. Zhou, N. Li, and S. Jha, "Improving utility and security of the shuffler-based differential privacy," *Proc. VLDB Endow.*, vol. 13, no. 13, p. 3545–3558, sep 2020. [Online]. Available: https://doi.org/10.14778/3424573.3424576

[116] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "VerifyNet: Secure and Verifiable Federated Learning," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 911–926, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8765347/

[117] X. Gu, M. Li, and L. Xiong, "PRECAD: privacy-preserving and robust federated learning via crypto-aided differential privacy," 2021. [Online]. Available: https://arxiv.org/abs/2110.11578

[118] X. Guo, Z. Liu, J. Li, J. Gao, B. Hou, C. Dong, and T. Baker, "VeriFL: Communication-Efficient and Fast Verifiable Aggregation for Federated Learning," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1736–1751, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9285303

[119] A. G. Sébert, R. Sirdey, O. Stan, and C. Gouy-Pailler, "Protecting Data from all Parties: Combining FHE and DP in Federated Learning," May 2022, arXiv:2205.04330 [cs]. [Online]. Available: http://arxiv.org/abs/2205.04330

[120] Y. Ren, Y. Li, G. Feng, and X. Zhang, "Privacy-Enhanced and Verification-Traceable Aggregation for Federated Learning," *IEEE Internet of Things Journal*, vol. 9, no. 24, pp. 24 933–24 948, Dec. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9846908/

[121] T. Eltaras, F. Sabry, W. Labda, K. Alzoubi, and Q. AHMEDELTARAS, "Efficient Verifiable Protocol for Privacy-Preserving Aggregation in Federated Learning," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 2977–2990, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10121168/

[122] C. Zhang, J. Weng, J. Weng, Y. Zhong, J.-N. Liu, and C. Deng, "Robust and secure federated learning with verifiable differential privacy," *IEEE Transactions on Dependable and Secure Computing*, vol. 22, no. 5, pp. 5713–5729, 2025.

[123] A. Korkmaz and P. Rao, "A selective homomorphic encryption approach for faster privacy-preserving federated learning," *arXiv preprint arXiv:2501.12911*, 2025.

[124] S. Chen, T. Zhou, H. Xie, and X. Yang, "VSEPDA: Verifiable secure and efficient privacy-preserving data aggregation protocol for image classification in federated learning," *Journal of Information Security and Applications*, vol. 90, p. 104039, May 2025, publisher: Elsevier BV. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2214212625000778

[125] A. A. Bellachia, M. A. Bouchiha, Y. Ghamri-Doudane, and M. Rabah, "Verifbfl: Leveraging zk-snarks for a verifiable blockchained federated learning," 2025. [Online]. Available: https://arxiv.org/abs/2501.04319

[126] A. Kothapalli, S. Setty, and I. Tzialla, "Nova: Recursive zero-knowledge arguments from folding schemes," in *Annual International Cryptology Conference*. Springer, 2022, pp. 359–388.

[127] R. Aziz, S. Banerjee, and S. Bouzefrane, "Privacy preserving federated learning: A novel approach for combining differential privacy and homomorphic encryption," in *IFIP International Conference on Information Security Theory and Practice*. Springer, 2024, pp. 162–177.

[128] R. Aziz, Y. Badr, and S. Bouzefrane, "Enhancing trust in central differential privacy using zk-snarks and cryptographic hashes," in *International Conference on Advanced Information Networking and Applications*. Springer, 2025, pp. 163–176.

[129] J. Domingo-Ferrer, D. Sánchez, and A. Blanco-Justicia, "The limits of differential privacy (and its misuse in data release and machine learning)," *Communications of the ACM*, vol. 64, no. 7, pp. 33–35, 2021.

[130] M. Nasr, J. Hayes, T. Steinke, B. Balle, F. Tramèr, M. Jagielski, N. Carlini, and A. Terzis, "Tight auditing of differentially private machine learning," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 1631–1648.

[131] M. Jagielski, J. Ullman, and A. Oprea, "Auditing differentially private machine learning: How private is private sgd?" *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 205–22 216, 2020.

[132] F. Tramer, A. Terzis, T. Steinke, S. Song, M. Jagielski, and N. Carlini, "Debugging differential privacy: A case study for privacy auditing," *arXiv preprint arXiv:2202.12219*, 2022.

[133] A. S. Shamsabadi, G. Tan, T. I. Cebere, A. Bellet, H. Haddadi, N. Papernot, X. Wang, and A. Weller, "Confidential-dpproof: Confidential proof of differentially private training," in *ICLR 2024-12th International Conference on Learning Representations*, 2024.

[134] G. M. Garrido, M. Babel, and J. Sedlmeir, "Towards Verifiable Differentially-Private Polling," Jun. 2022, arXiv:2206.07220 [cs]. [Online]. Available: http://arxiv.org/abs/2206.07220

[135] A. Biswas and G. Cormode, "Interactive proofs for differentially private counting," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, 2023, pp. 1919–1933.

[136] D. M. Davidow, Y. Manevich, and E. Toch, "Privacy-Preserving Payment System With Verifiable Local Differential Privacy," 2023, publication info: Published elsewhere. Minor revision. https://aftconf.github.io/aft23/index.html. [Online]. Available: https://eprint.iacr.org/2023/126

[137] C. L. Canonne, G. Kamath, and T. Steinke, "The discrete gaussian for differential privacy," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 676–15 688, 2020.

[138] S. Inusah and T. J. Kozubowski, "A discrete analogue of the laplace distribution," *Journal of statistical planning and inference*, vol. 136, no. 3, pp. 1090–1102, 2006.

[139] M. Albrecht, L. Grassi, C. Rechberger, A. Roy, and T. Tiessen, "Mimc: Efficient encryption and cryptographic hashing with minimal multiplicative complexity," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2016, pp. 191–219.

[140] J. L. Bentley and J. B. Saxe, "Decomposable searching problems i. static-to-dynamic transformation," *Journal of Algorithms*, vol. 1, no. 4, pp. 301–358, 1980.

[141] Y. Yu, P. K. Gunda, and M. Isard, "Distributed aggregation for data-parallel computing: Interfaces and implementations," in *ACM Symposium on Operating Systems Principles (SOSP)*, October 2009, pp. 247–260. [Online]. Available: https://www.microsoft.com/en-us/research/publication/distributed-aggregation-for-data-parallel-computing-interfaces-and-implementations/

[142] L. Li, W. Xu, T. Chen, G. B. Giannakis, and Q. Ling, "RSA: byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," *CoRR*, vol. abs/1811.03761, 2018. [Online]. Available: http://arxiv.org/abs/1811.03761

[143] Y.-X. Wang, B. Balle, and S. P. Kasiviswanathan, "Subsampled rényi differential privacy and analytical moments accountant," in *The 22nd international conference on artificial intelligence and statistics*. PMLR, 2019, pp. 1226–1235.

[144] B. Balle, G. Barthe, and M. Gaboardi, "Privacy amplification by subsampling: Tight analyses via couplings and divergences," *Advances in neural information processing systems*, vol. 31, 2018.

[145] J. P. Near and C. Abuah, *Programming Differential Privacy*, 2021, vol. 1. [Online]. Available: https://uvm-plaid.github.io/programming-dp/

[146] F. D. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, 2009, pp. 19–30.

[147] C. Dwork, G. N. Rothblum, and S. Vadhan, "Boosting and differential privacy," in *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 2010, pp. 51–60.

Appendix A

Appendix For Chapter 3

Contents

Composition in DP allows combining multiple differential private mechanisms to preserve privacy while enabling more complex data analysis. It offers a way to bound the total privacy cost while releasing multiple result on the same dataset[145]. There are two types of composition: sequential composition and parallel composition.

A.1 Sequential Composition [60]

Sequential composition is a fundamental property in DP because it offers a way to design of algorithms that consult the data multiple times. The basic sequential composition for an ϵ-differential privacy is defined by the theorem A.1.1.

Theorem A.1.1. *Let $\mathcal{M}_\infty$ be an ϵ_1-differentially private algorithm, and let $\mathcal{M}_2$ be an ϵ_2-differentially private algorithm. Then their combination, defined to be $\mathcal{M}_{1,2}$ by the mapping: $\mathcal{M}_{1,2}(x) = (\mathcal{M}_1(x), \mathcal{M}_2(x))$ is $\epsilon_1 + \epsilon_2$-differentially private.*

This means that the privacy loss accumulates as we apply multiple differentially private mechanisms to the same dataset.

The sequential composition theorem can be applied repeatedly to combine k differentially private mechanisms. Formally, we obtain the following corollary:

Corollary A.1.1. *Let $\mathcal{M}_i$ be an ϵ_i-differentially private algorithm with $i \in K = [1..k]$. Then if $\mathcal{M}_K$ is defined to be $\mathcal{M}_K(x) = (\mathcal{M}_1(x), \ldots, \mathcal{M}_k(x))$ is $(\sum_{i=1}^{k} \epsilon_i)$-differentially private.*

The sequential composition theorem A.1.1 can be extended to the approximate differential privacy setting.

Theorem A.1.2. *Let $\mathcal{M}_i$ be an (ϵ_i, δ_i)-differentially private algorithm with $i \in K = [1..k]$. Then if $\mathcal{M}_K$ is defined to be $\mathcal{M}_K(x) = (\mathcal{M}_1(x), \ldots, \mathcal{M}_k(x))$ is $(\sum_{i=1}^{k} \epsilon_i, \sum_{i=1}^{k} \delta_i)$-differentially private.*

In ADP, we should take care of the parameter δ in addition to the ϵ. This is why, the choice of the two parameters need a careful attention based on the specific application and the desired privacy and accuracy.

A.2 Parallel Composition [146]

Unlike sequential composition, which combines multiple differentially private mechanisms in the same dataset, parallel composition is based on the idea of dividing the dataset into disjoint chunks and apply differentially private mechanism on each chunk separately.The formal statement of this property is given by the theorem A.2.1.

Theorem A.2.1. *Let $\mathcal{M}_i$ each provide ϵ-differential privacy. Let D_i be arbitrary disjoint subsets of the input domain D. The sequence of $\mathcal{M}_i(X \cap D_i)$ provides ϵ-differential privacy.*

This type of composition gives lower bound of privacy loss than the sequential mechanism. An example of where parallel composition is automatically satisfied is a query that returns a histogram of the dataset.

A.3 Advanced Composition [147]

While parallel and sequential composition are fundamental properties of both approximate and pure differential privacy, approximate differential privacy allows another type of composition. It is called advanced composition.

This type of composition deals with mechanisms that are instances of $k-$fold adaptive composition. A $k-$fold adaptive composition is a sequence of mechanisms $m_1, ..., m_k$ such that :

- Each mechanism m_i may be chosen based on the outputs of all previous mechanisms $m_1, \ldots, m_{i-1}$
- The input to each mechanism is both a private dataset and all outputs of previous mechanisms.

In fact, many iterative algorithms, such as gradient descent or k-means, are nearly always instances of $k-$fold adaptive composition. By using this type of composition, we can provide a tighter bound on the privacy than the basic sequential composition. Formally, this composition was defined in [147] by the following theorem:

Theorem A.3.1. *For every $\epsilon > 0, \delta, \delta' > 0$, and $k \in \mathbb{N}$, the class of (ϵ, δ)-differentially private mechanisms is $(\epsilon', k\delta + \delta')$-differentially private under k-fold adaptive composition, for*

$$\epsilon' = \sqrt{2k \ln(\frac{1}{\delta'})}.\epsilon + k.\epsilon.\epsilon_0$$

where $\epsilon_0 = e^{\epsilon} - 1$

Using this composition permit to tightly bound the privacy loss after sequential composition. Figure A.1 compares the bounds provided by basic composition and advanced composition while executing k mechanisms. We can see clearly that as the number of mechanisms k increases, the bounds provided by advanced composition are more accurate than the basic composition. However, even with advanced composition, the bounds are loose bounds.

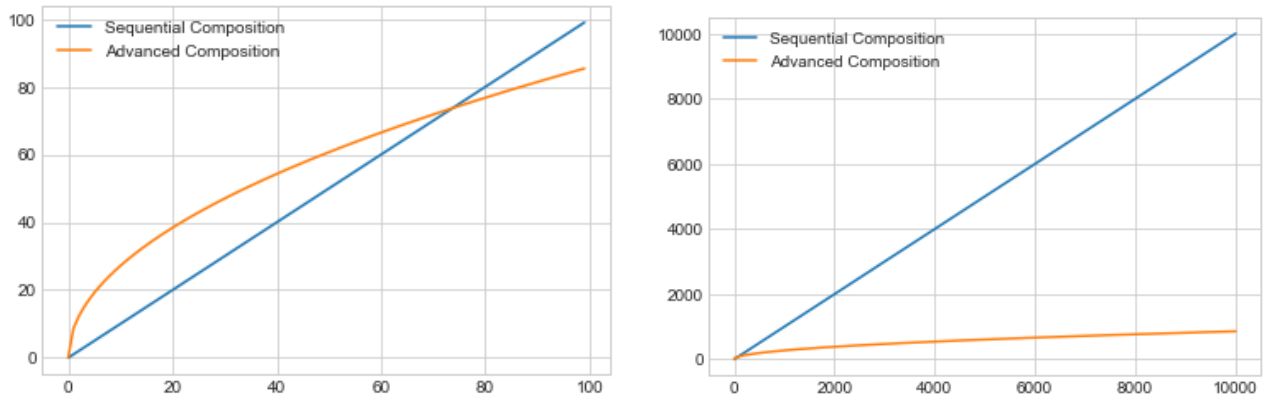


Figure A.1: Advanced composition vs basic composition[145]

Appendix B

Appendix For Chapter 4

This appendix presents the theoretical foundations of the privacy amplification results discussed in Chapter 4. We restate the key results from Erlingsson *et al.* [62], namely *Theorem 7* and *Corollary 9*, which formalize the amplification of local differential privacy guarantees through random shuffling.

B.1 Amplification by Shuffling

Erlingsson *et al.* introduced the following fundamental result, showing that random shuffling of locally differentially private (LDP) outputs yields stronger guarantees in the central model of differential privacy.

Theorem B.1.1 (Amplification by Shuffling [62])**.** *Let* $A_{\mathrm{ldp}}^{(i)} : \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(i-1)} \times \mathcal{D} \to \mathcal{S}^{(i)}$ *for* $i \in [n]$ *be a sequence of algorithms such that each* $A_{\mathrm{ldp}}^{(i)}$ *is* ε_0*-differentially private for all values of the auxiliary inputs in* $\mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(i-1)}$*.*

Let $A_{\mathrm{sl}} : \mathcal{D}^n \to \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(n)}$ *be the algorithm that, given a dataset* $x_{1:n} \in \mathcal{D}^n$*, samples a uniform random permutation* π *over* $[n]$*, then sequentially computes*

$$z_i = A_{\mathrm{ldp}}(z_{1:i-1}, x_{\pi(i)}), \quad \text{for } i = 1, 2, \ldots, n,$$

and outputs $z_{1:n}$*.*

Then, for any integer $n > 1$ *and any* $\varepsilon_0, \delta > 0$*, the algorithm* A_{sl} *satisfies* (ε, δ)*-differential privacy in the central model, where*

$$\varepsilon \leq \varepsilon_1 \sqrt{2n \log(1/\delta)} + n\varepsilon_1(e^{\varepsilon_1} - 1),$$

for $\varepsilon_1 = \frac{2e^{2\varepsilon_0}(e^{\varepsilon_0} - 1)}{n}$*.*

In particular:

$$\varepsilon \leq \frac{e^{2\varepsilon_0}(e^{\varepsilon_0}-1)\sqrt{8\log(1/\delta)}}{\sqrt{n}} + \frac{6e^{4\varepsilon_0}(e^{\varepsilon_0}-1)^2}{n},$$

when $\varepsilon_0 \leq \ln(n/4)/3$*, and for any* $n \geq 1000$*,* $0 < \varepsilon_0 < 1/2$*, and* $0 < \delta < 1/100$*,*

$$\varepsilon \leq 12\varepsilon_0\sqrt{\frac{\log(1/\delta)}{n}}.$$

B.2 Amplification with Pre-Shuffled Randomization

The proof of Theorem B.1.1 assumes that data are shuffled before randomization. However, trusting a remote shuffler negates the benefits of LDP. Erlingsson et al. further established that amplification still holds when the randomization occurs *before* shuffling, under certain conditions.

Corollary B.2.1 (Amplification with Pre-Shuffled Randomization [62])**.** *Let* $A_{\text{ldp}}^{(i)} : \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(i-1)} \times \mathcal{D} \to \mathcal{S}^{(i)}$ *for* $i \in [n]$ *be a sequence of algorithms such that each* $A_{\text{ldp}}^{(i)}$ *is* ε_0*-differentially private for all auxiliary inputs. Let* $A_{\text{post}} : \mathcal{D}^n \to \mathcal{S}^{(1)} \times \cdots \times \mathcal{S}^{(n)}$ *be the algorithm that, given a dataset* $D \in \mathcal{D}^n$*, computes* $z_{1:n} = A_{\text{local}}(D)$*, samples a random uniform permutation* π*, and outputs* $z_{\pi(1)}, \ldots, z_{\pi(n)}$*.*

Let $S \subseteq [n]$ *be a set of indices such that for all* $i, j \in S$*,* $A_{\text{ldp}}^{(i)} \equiv A_{\text{ldp}}^{(j)}$*. Then, for* $|S| \geq 1000$*,* $0 < \varepsilon_0 < 1/2$*, and* $0 < \delta < 1/100$*, the algorithm* A_{post} *satisfies* (ε, δ)*-differential privacy at index* $i \in S$ *in the central model, where*

$$\varepsilon = 12\varepsilon_0\sqrt{\frac{\log(1/\delta)}{|S|}}.$$

This corollary establishes that privacy amplification can still occur even if local randomization precedes shuffling. However, the guarantee holds meaningfully only under restrictive conditions (large $|S|$, small ε_0), which may limit practical applicability.

Appendix C

Appendix For Chapter 5

Overview

Inside zk-SNARK circuits, all variables live in a finite field $\mathbb{F}_p$. Since finite fields have no notion of signed integers, proving that a value is non-negative ($x \geq 0$) requires constraining it to lie within a valid binary range $[0, 2^k - 1]$. This is done using bit-decomposition.

Mathematical Principle

To encode non-negativity, we express x as a sum of binary variables:

$$x = \sum_{i=0}^{k-1} b_i 2^i, \qquad b_i \in \{0, 1\}.$$

Each constraint $b_i(b_i - 1) = 0$ guarantees that b_i is boolean, so the reconstructed x is always an unsigned integer in $[0, 2^k - 1]$. Thus, x is implicitly non-negative.

Illustration 1: Positive Value

Consider $x = 42$. Its binary decomposition is:

$$42 = 2^5 + 2^3 + 2^1 = 32 + 8 + 2.$$

The bit vector $(b_{31}, \ldots, b_0)$ contains:

$$b = (0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, \ldots, 0).$$

During recomposition:

$$\texttt{recomposed} = \sum_{i=0}^{31} 2^i b_i = 42,$$

and the constraint `assert(recomposed == x)` holds. Thus, $x = 42$ satisfies the non-negativity condition and the circuit accepts it.

Illustration 2: Modular Negative Value

Now consider the value $x = p - 1$, which represents -1 in $\mathbb{F}_p$ (for example, in BN254, $p \approx 2^{254}$). In binary, $p - 1$ is a 254-bit sequence of all ones:

$$p - 1 = (1, 1, 1, \ldots, 1)_2.$$

When the circuit attempts to decompose x using only 32 bits, it fails to satisfy the equality:

$$\texttt{recomposed} = \sum_{i=0}^{31} 2^i b_i \neq x,$$

because $p - 1$ cannot be represented within the 32-bit range $[0, 2^{32} - 1]$. Therefore, the constraint `assert(recomposed == x)` fails, and the proof generation aborts.

Appendix D

Appendix For Chapter 8

D.1 Notation Summary

We summarize below the main mathematical notations used throughout the paper for clarity and consistency:

- n: Total number of clients participating in federated learning.
- d: Dimension of the model update vectors.
- $w_i \in \mathbb{R}^d$: Local model update vector computed by client i.
- $\tilde{w}_i$: Noisy version of client i's update.
- $\mathbf{w} = \sum_{i=1}^{n} w_i$: Aggregated global model update before adding noise.
- $\tilde{\mathbf{w}} = \sum_{i=1}^{n} \tilde{w}_i = \mathbf{w} + \boldsymbol{\eta}$: Aggregated noisy global model update.
- $\mathbf{e}_i \in \mathbb{R}^d$: Sparse noise vector added by client i on a private coordinate subset.
- $\boldsymbol{\eta} = \sum_{i=1}^{n} \mathbf{e}_i$: Total noise added across all clients.
- $w_i^{(j)}$: The j-th additive share of the local update w_i in secret sharing.
- $C_i = g^{w_i} h^{r_i}$: Pedersen commitment to w_i using randomness $r_i \in \mathbb{Z}_q$.
- $C = \prod_{i=1}^{n} C_i = g^{w} h^{R}$: Aggregated commitment over all clients.
- S_i: Set of private indices chosen by client i to add noise (disjoint across clients).

- k: The size of S_i.
- $r_i \in \mathbb{Z}_q$: Random value used in the commitment by client i.
- ρ: Ephemeral randomness used in the zero-knowledge proof of opening.
- z: ZKP response computed as $z = \rho + cw$.
- c: Fiat–Shamir challenge hash $c = H(A, C, w)$ in the ZKP protocol.

Group elements g, h are fixed public generators of a cyclic group of prime order q, used for Pedersen commitments.

D.2 Security under $(n-1)$ Collusion

Theorem D.2.1 (Security under $(n-1)$ Collusion)**.** *Consider a system with n clients. Suppose each honest client uses additive secret sharing to split k of its weights among all n participants. Then, even if $(n-1)$ clients collude, they cannot reconstruct the honest client's values on at least those k protected weights.*

Proof. Let client_H be the honest client. For each weight $w_j^{(H)}$, $j \in S$, client_H generates shares $s_{H,1,j}, \ldots, s_{H,n,j} \in \mathbb{Z}_p$ satisfying

$$\sum_{r=1}^{n} s_{H,r,j} \equiv w_j^{(H)} \mod p.$$

client_H keeps $s_{H,H,j}$ and sends the other $n-1$ shares to the other clients.

The colluding clients know $\{s_{H,c,j} : c \neq H\}$. However, since the final share $s_{H,H,j}$ is uniformly random (as $w_j^{(H)} - \sum_{c \neq H} s_{H,c,j} \mod p$), the distribution of $\{s_{H,c,j} : c \neq H\}$ is uniform and independent of $w_j^{(H)}$.

Thus, the colluding clients gain no information about $w_j^{(H)}$.

Therefore, for each of the k weights $j \in S$, $w_j^{(H)}$ remains information-theoretically secure from the colluders. This completes the proof. □