



Article

A Survey of Security Strategies in Federated Learning: Defending Models, Data, and Privacy

Habib Ullah Manzoor¹, Attia Shabbir², Ao Chen¹, David Flynn¹ and Ahmed Zoha^{1,*}

¹ James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, UK; h.manzoor.1@research.gla.ac.uk (H.U.M.); 2928133c@student.gla.ac.uk (A.C.); david.flynn@gla.ac.uk (D.F.)
² Faculty of Computer Science, Ghulam Ishaq Khan Institute, Topi 23640, Pakistan; attiahabbir371@gmail.com
* Correspondence: ahmed.zoha@glasgow.ac.uk

Abstract: Federated Learning (FL) has emerged as a transformative paradigm in machine learning, enabling decentralized model training across multiple devices while preserving data privacy. However, the decentralized nature of FL introduces significant security challenges, making it vulnerable to various attacks targeting models, data, and privacy. This survey provides a comprehensive overview of the defense strategies against these attacks, categorizing them into data and model defenses and privacy attacks. We explore pre-aggregation, in-aggregation, and post-aggregation defenses, highlighting their methodologies and effectiveness. Additionally, the survey delves into advanced techniques such as homomorphic encryption and differential privacy to safeguard sensitive information. The integration of blockchain technology for enhancing security in FL environments is also discussed, along with incentive mechanisms to promote active participation among clients. Through this detailed examination, the survey aims to inform and guide future research in developing robust defense frameworks for FL systems.

Keywords: security; federated learning; attack; defense



Citation: Manzoor, H.U.; Shabbir, A.; Chen, A.; Flynn, D.; Zoha, A. A Survey of Security Strategies in Federated Learning: Defending Models, Data, and Privacy. *Future Internet* **2024**, *16*, 374. <https://doi.org/10.3390/fi16100374>

Academic Editors: Kuo-Yu Tsai and Kuo Chung-Wei

Received: 4 September 2024
Revised: 8 October 2024
Accepted: 12 October 2024
Published: 15 October 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Regarding the Internet of Things (IoT), big data plays a pivotal role, serving as the lifeblood that fuels innovation and operational efficiency [1,2]. The interconnected network of devices generates vast amounts of data, offering unprecedented insights into user behavior, environmental conditions, and operational patterns. This wealth of data enables organizations to make informed decisions, optimize processes, and enhance services in real time [3–5]. However, alongside these transformative benefits, the sheer volume of data generated by IoT devices presents significant challenges. Managing and processing such enormous datasets can strain traditional server capacities and lead to escalated maintenance costs [6,7]. Moreover, the transmission of large volumes of data over networks can congest communication channels, potentially causing delays and reducing the overall system efficiency. The initial setup costs for robust data servers and infrastructure can also be prohibitive, especially for smaller enterprises or in resource-constrained environments [8].

Centralized machine learning systems encounter significant security challenges due to the concentration of data in centralized servers [9–11]. The logistical and financial burdens of managing vast amounts of data from disparate sources also amplify these concerns [6]. Maintaining the integrity and confidentiality of data becomes paramount as centralized systems are vulnerable to targeted attacks aimed at breaching server security. Such breaches not only compromise sensitive information but also jeopardize the entire machine learning pipeline, leading to potential operational failures [12]. Moreover, the reliance on centralized communication networks introduces additional vulnerabilities [13]. The data transmitted over these networks can be intercepted or manipulated, posing risks to data authenticity and privacy. Furthermore, centralized architectures necessitate robust security measures

to safeguard against insider threats and unauthorized access attempts, which can exploit single points of failure within the system [14].

In response to these security challenges, alternative paradigms like FL have gained traction [15]. FL decentralizes data processing and model training, thereby distributing the risk associated with data breaches across multiple devices [16,17]. By keeping data local to where they are generated, FL mitigates the exposure of sensitive information during transmission and storage [18]. This decentralized approach not only enhances data security but also reduces the potential impact of security breaches, offering a more resilient framework for machine learning in IoT and other data-intensive applications [19,20].

FL emerges as a promising solution to the inefficiencies of centralized data systems. By distributing the model training across decentralized devices while keeping the data localized, FL addresses the concerns regarding data privacy [20–22] and server failure [20]. Despite its potential, however, FL introduces its own set of security challenges. The decentralized nature of FL systems can render them vulnerable to various attacks targeting models, data, and privacy. This paper presents an in-depth survey of the FL techniques used against data, privacy, and model attacks. Table 1 presents a comparison with the state-of-the-art surveys. Evidently, some of the presented techniques fail to mention all the defense frameworks used to defend FL.

Table 1. Comparison of key characteristics and attributes between existing state-of-the-art surveys and our survey in the field of adversarial attacks on FL.

Ref.	AT	P	BRA	R	D&R	RCS	AUA	Ho	KD	SMP	TEE	SL	PG	DP	BC	In
[23]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
[24]	✓	✗	✓	✓	✓	✗	✓	✓	✗	✓	✓	✗	✓	✓	✗	✗
[25]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓	✗	✗
[26]	✗	✗	✓	✗	✓	✗	✗	✓	✗	✗	✗	✗	✗	✓	✗	✗
[27]	✓	✗	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✓	✗	✗
[28]	✗	✗	✗	✓	✓	✗	✗	✓	✗	✗	✗	✗	✓	✓	✓	✗
[29]	✓	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓	✗	✗	✓	✗	✗
[20]	✗	✗	✓	✗	✓	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗
[30]	✓	✓	✗	✗	✗	✗	✓	✗	✓	✓	✓	✗	✗	✓	✓	✗
[31]	✗	✗	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✓	✓	✗
[18]	✗	✓	✓	✗	✓	✗	✗	✓	✓	✓	✓	✗	✗	✓	✓	✓
[32]	✗	✗	✓	✗	✓	✗	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗
[23]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗
Our	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

The main contributions of this paper are as follows:

1. The paper offers an exhaustive overview of the defense mechanisms in FL, categorizing them into pre-aggregation, in-aggregation, and post-aggregation defenses. It systematically explores how these defenses protect against data, privacy, and model attacks, making it a critical resource for researchers and practitioners.
2. The paper provides a comparative analysis of the existing surveys, identifying gaps in the literature. It demonstrates that the previous surveys have not comprehensively covered all the defense frameworks, thus positioning this paper as a more complete resource in the domain of FL security.
3. The paper not only reviews the existing defenses but also outlines the critical areas for future research. It calls for innovations in handling unreliable participants, improving the energy efficiency in FL, and developing new defense mechanisms that can adapt to the evolving threat landscape.
4. Recognizing the dynamic nature of the threats in FL environments, the paper stresses the need for continuous innovation in defense strategies. It advocates for comprehensive strategies that can adapt to new challenges, ensuring the integrity and reliability of decentralized machine learning systems.

The remainder of this paper is organized as follows and is presented Figure 1: Section 2 provides an overview of FL. Section 3 discusses the different types of FL. Section 4 offers a comprehensive overview of the various attacks that can target FL systems, categorizing them into data, model, and privacy attacks. Section 5 explores the defense strategies to mitigate these attacks, with an in-depth look at pre-aggregation, in-aggregation, and post-aggregation defenses. Section 6 presents the defense frameworks against model and data attacks, while Section 7 focuses on the privacy frameworks. Section 8 discusses the challenges, Section 9 outlines the future directions, and Section 11 concludes the paper.

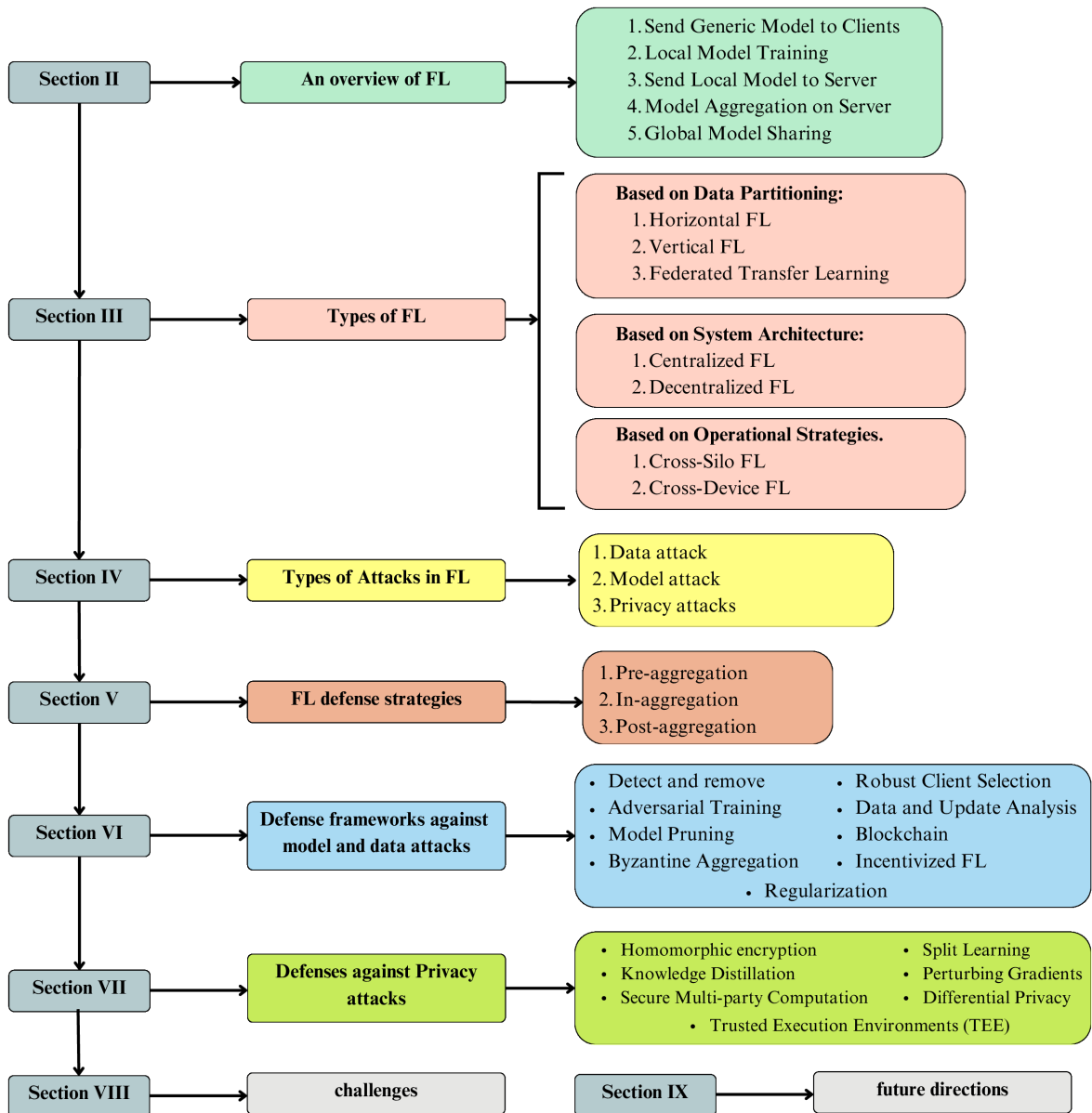


Figure 1. Paper distribution.

2. Overview of FL

In FL, the training process involving the coordination of a central server and multiple client devices (total number of devices N) can be described in five detailed steps depicted in Figure 2, incorporating mathematical expressions for the aggregation process, local training, and model sharing [19,33,34]:

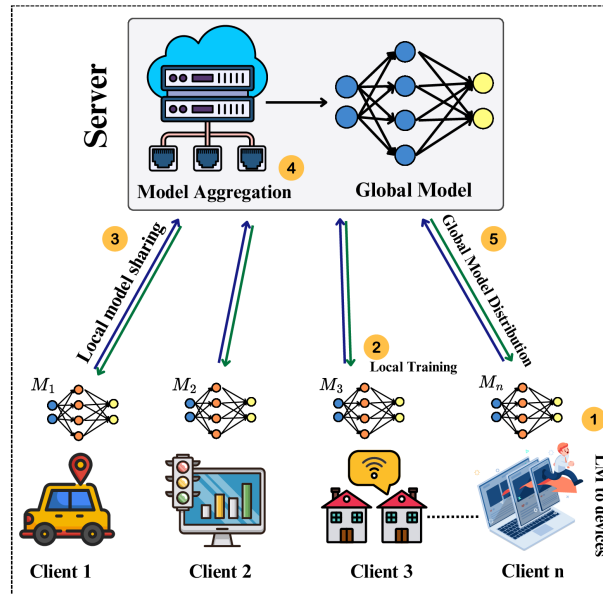


Figure 2. Overview of FL.

2.1. Send Generic Model to Clients

The central server initializes the training process by creating a generic (initial) model $\mathbf{w}^{(0)}$ and sending it to all N client devices. Mathematically, this can be represented as

$$\mathbf{w}^{(0)} \rightarrow \{\text{Client}_1, \text{Client}_2, \dots, \text{Client}_N\} \quad (1)$$

2.2. Local Model Training

Each client i independently trains the received generic model $\mathbf{w}^{(0)}$ on its local dataset \mathcal{D}_i . The local training process involves optimizing a local objective function $\mathcal{L}_i(\mathbf{w})$ using stochastic gradient descent (SGD) or another optimization algorithm. After E epochs of local training, the updated model parameters for client i are denoted as $\mathbf{w}_i^{(t)}$:

$$\mathbf{w}_i^{(t)} = \mathbf{w}^{(t-1)} - \eta \nabla \mathcal{L}_i(\mathbf{w}^{(t-1)}) \quad (2)$$

where η is the learning rate and t indicates the current round of training.

2.3. Send Local Model to Server

After completing local training, each client i sends its locally updated model parameters $\mathbf{w}_i^{(t)}$ to the central server:

$$\{\mathbf{w}_1^{(t)}, \mathbf{w}_2^{(t)}, \dots, \mathbf{w}_N^{(t)}\} \rightarrow \text{Server} \quad (3)$$

2.4. Model Aggregation on Server

The central server aggregates the local models received from the clients to form a single global model $\mathbf{w}^{(t)}$. One common method for aggregation is Federated Averaging (FedAvg), which computes the weighted average of the local model parameters based on the size of each client's dataset $|\mathcal{D}_i|$:

$$\mathbf{w}^{(t)} = \frac{1}{N} \sum_{i=1}^N \frac{|\mathcal{D}_i|}{\sum_{j=1}^N |\mathcal{D}_j|} \mathbf{w}_i^{(t)} \quad (4)$$

Alternatively, if we consider equal weighting, the aggregation simplifies to

$$\mathbf{w}^{(t)} = \frac{1}{N} \sum_{i=1}^N \mathbf{w}_i^{(t)} \tag{5}$$

2.5. Global Model Sharing

Once the aggregation is complete, the central server shares the updated global model $\mathbf{w}^{(t)}$ with all N client devices:

$$\mathbf{w}^{(t)} \rightarrow \{\text{Client}_1, \text{Client}_2, \dots, \text{Client}_N\} \tag{6}$$

The clients then use this updated global model as the starting point for the next round of local training, and the process repeats until the model reaches the desired level of accuracy and performance.

By following the above steps, FL enables collaborative model training across distributed networks while ensuring data privacy and security, leveraging mathematical techniques to ensure the integrity and efficiency of the training process.

3. Types of FL

Various types of FL exist, each suited for different applications and scenarios. This document outlines the primary types of FL, categorized by data partitioning, system architecture, and operational strategies, as depicted in Figure 3.

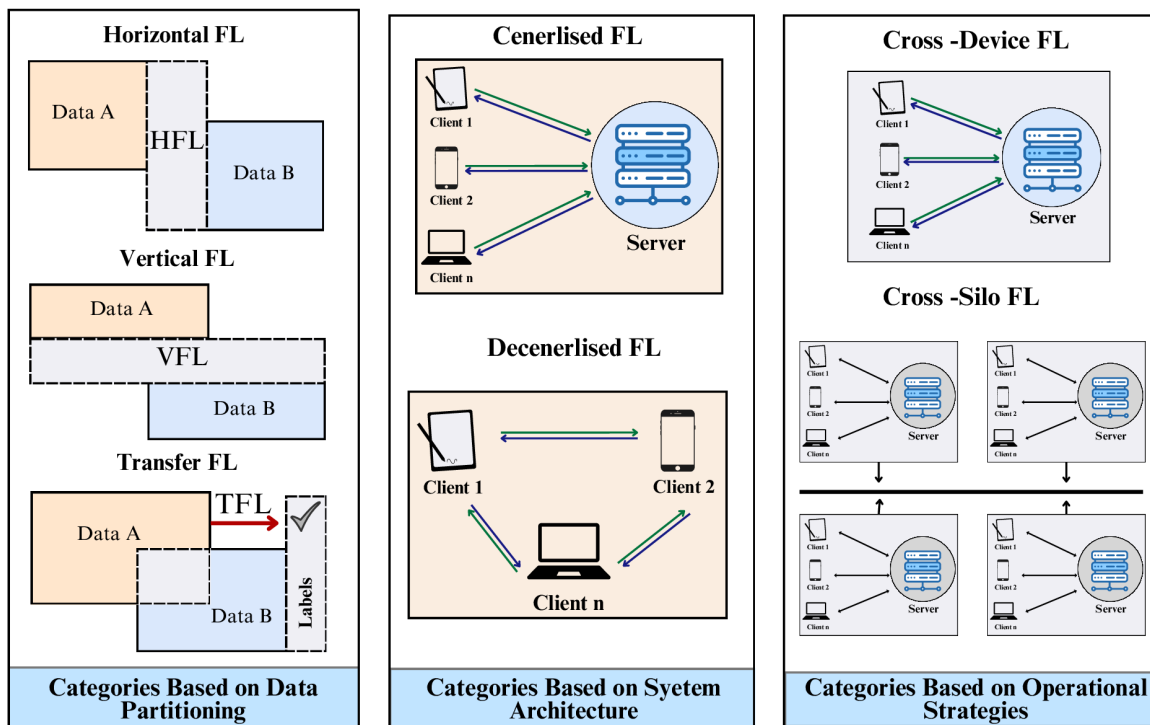


Figure 3. Types of FL.

3.1. Categories Based on Data Partitioning

3.1.1. Horizontal Federated Learning (HFL)

Horizontal FL, also known as sample-based FL, occurs when datasets from different sources share the same feature space but differ in samples [35–38]. It is applicable when organizations or devices possess data of the same type (i.e., having the same features) but of different users [39]. For instance, multiple hospitals may collaborate to train a model using patient records that contain the same set of medical features but from different patients [40].

Applications

- Healthcare: Collaborative training using patient data from different hospitals.
- Finance: Banks combining transaction records to improve fraud detection models.
- Mobile Devices: Enhancing predictive text models using data from different smartphones.

3.1.2. Vertical Federated Learning (VFL)

Vertical Federated Learning, also known as feature-based FL, involves datasets that have different feature spaces but share the same sample ID space [41,41]. This scenario occurs when different organizations or devices hold complementary information about the same set of entities [42]. For example, one organization may have demographic information about individuals, while another has purchasing behavior data [43]. By combining these features, a more comprehensive model can be trained without sharing raw data.

Applications

- Marketing: Combining customer demographic data with purchasing data to improve recommendation systems.
- Financial Services: Integrating credit scores from one institution with transaction histories from another to enhance risk assessment models.

3.1.3. Federated Transfer Learning (FTL)

Federated Transfer Learning is used when datasets have different feature spaces and only partially overlap in sample ID space [44]. This type of FL leverages transfer learning techniques to transfer knowledge from a source domain (with abundant labeled data) to a target domain (with limited labeled data) [45]. FTL is useful when collaborating organizations have different but related data, allowing them to benefit from each other's data without direct access.

Applications

- Cross-domain recommendation systems: Using user data from different services to improve personalized recommendations.
- Cross-organization collaborations: Enhancing machine learning models by leveraging different feature sets from collaborating organizations.

3.2. Categories Based on System Architecture

3.2.1. Centralized Federated Learning

In Centralized Federated Learning, a central server coordinates the training process. Clients train local models on their data and send updates to the central server, which aggregates these updates to form a global model. This architecture simplifies the orchestration and aggregation process but still relies on a central point for coordination.

Applications

- Enterprise environments: Where a central server can manage and coordinate model updates efficiently.
- Academic collaborations: Coordinated research projects where data are sensitive but require central oversight.

3.2.2. Decentralized Federated Learning

Decentralized FL eliminates the central server, with clients directly communicating and sharing model updates with each other [46,47]. This approach enhances resilience and privacy as there is no single point of failure or central data repository [48,49]. Decentralized FL requires more complex coordination protocols to manage peer-to-peer communications [15,50,51].

Applications

- Peer-to-peer networks: Applications in blockchain or decentralized networks where trust is distributed.

- Edge computing: IoT devices collaborating directly to update models without a central server.

3.3. Categories Based on Operational Strategies

3.3.1. Cross-Silo Federated Learning

Cross-Silo FL typically involves a small number of data silos (e.g., organizations or institutions) with relatively large datasets [52,53]. These silos collaboratively train a model, ensuring data privacy and security [54]. This type of FL often requires more stable and reliable communication infrastructure compared to cross-device FL [55].

Applications

- Enterprise collaborations: Multiple companies training a shared model using their proprietary data.
- Inter-institutional research: Academic institutions combining research data to build robust predictive models.

3.3.2. Cross-Device FL

Cross-device FL involves a large number of devices (e.g., smartphones and IoT devices) with relatively small local datasets [45]. This type of FL is characterized by high variability in device availability, computational power, and network connectivity [56]. Cross-device FL must handle these challenges efficiently to enable training on a diverse and extensive network of devices.

Applications

- Mobile applications: Improving user experience on mobile apps by training models on user behavior data from many devices.
- IoT networks: Enhancing predictive maintenance models using data from various IoT sensors and devices.

4. Types of Attacks in FL

FL is vulnerable to several types of attacks that can compromise the integrity, confidentiality, and performance of the model. These attacks can be broadly categorized into data attacks, model attacks, and privacy attacks, as presented in Figure 4. Each of these attacks affects the Federated Learning process in distinct ways and has cascading effects throughout the system.

1. Data attack.
2. Model attack.
3. Privacy attack.

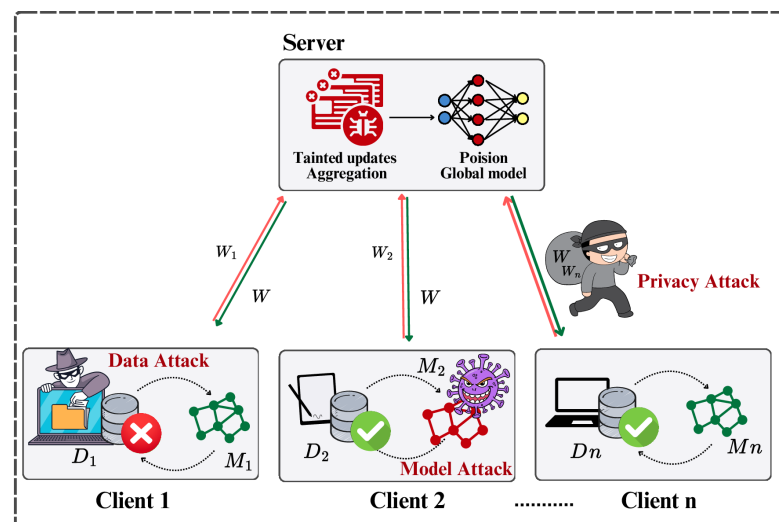


Figure 4. Types of attacks in FL.

4.1. Data Attack

Data attacks in FL aim to corrupt the training data on client devices to influence the learning process adversely. These attacks can manifest in several forms:

Data Poisoning: Malicious clients inject false or misleading data into their local datasets [17]. This can skew the model's training process, leading to a degraded or biased global model [57]. For example, in an energy forecasting network, an attacker can inject false data into the forecasting system [58].

Label Flipping: A specific type of data poisoning where the labels of the training data are intentionally flipped [59,60]. For instance, in an image classification task, images of cats could be labeled as dogs.

Backdoor Attacks: The attacker modifies the training data to introduce a backdoor into the model [61]. For example, specific triggers in the input data can cause the model to output a particular incorrect result.

The impact of data attacks in FL begins with local data corruption at the client level, where the attacker corrupts the local dataset. These corrupted data then influence the local model update during the model training phase. Subsequently, the poisoned local model is sent to the central server for aggregation. At this stage, the central server aggregates updates from all clients, including the poisoned ones, resulting in a compromised global model. Finally, this compromised global model is distributed back to all the clients, leading to degraded performance or backdoor exploitation across the entire system.

4.2. Model Attack

In model attack, malicious clients intentionally manipulate their local model updates before sending them to the central server [62,63]. The goal of model poisoning is to corrupt the global model by introducing harmful alterations into the training process. One common method of model poisoning involves modifying the gradients or parameters of the local model to push the global model toward incorrect or biased outputs [64].

The impact of model poisoning travels through several stages in the FL process. Initially, the attack occurs at the client level, where the attacker modifies the local model updates. These poisoned updates are then transmitted to the central server. During the model aggregation phase, the central server combines updates from all the clients, including the malicious ones. Because the server typically lacks the ability to differentiate between honest and malicious updates, the poisoned updates can significantly influence the global model. Finally, the compromised global model is redistributed to all the clients, propagating the effects of the attack throughout the entire FL network.

4.3. Privacy Attacks

Privacy attacks in FL are aimed at extracting sensitive information from the model updates or the aggregated model itself, thereby compromising the confidentiality of the training data [65]. Despite the inherent privacy-preserving nature of FL, which keeps raw data on local devices, these attacks exploit the communication of model updates to infer details about the data [66,67].

The impact of privacy attacks in FL travels through various stages. Initially, model updates are transmitted from clients to the central server. During this transmission, attackers intercept and analyze these updates to extract sensitive information. The severity of the privacy breach depends on the extent to which the attacker can infer details about the training data. As the extracted information is potentially disseminated, it not only compromises individual privacy but also undermines trust in the FL system.

5. Defense Frameworks in FL

Given the variety of attacks in FL, it is imperative to deploy a range of defense frameworks tailored to the specific nature of each attack, considering factors such as device configurations, FL architecture, and available resources. Each type of attack, including data, model, and privacy, exploits different vulnerabilities within the FL ecosystem, necessitating

distinct defensive strategies [68]. For instance, robust data validation and anomaly detection systems can mitigate data poisoning attacks [69], while secure aggregation protocols and byzantine fault tolerance mechanisms are more effective against model poisoning [70]. Privacy-preserving techniques, such as differential privacy and homomorphic encryption, are essential to protect against inference and membership inference attacks. Due to the diverse nature of these threats, a one-size-fits-all defense mechanism is impractical. Instead, a layered defense approach, incorporating multiple frameworks designed to address specific attack vectors, ensures comprehensive protection and enhances the overall security and resilience of FL systems [71].

To counter the emerging threats FL, researchers have proposed a variety of defensive strategies [72–74]. These defense mechanisms address adversarial attacks across distinct phases of the learning process: pre-aggregation, in-aggregation, and post-aggregation. Pre-aggregation defenses [72,73,75] focus on early identification and mitigation of malicious updates before they impact the global model. In-aggregation defense techniques [64,74,76–78] employ robust aggregation operators to mitigate adversarial effects during the global model update phase. Conversely, post-aggregation defense strategies [79–81] concentrate on repairing adversarial models after completing the FL training process, ensuring the final model’s integrity. These phase-specific defense mechanisms are crucial for enhancing the security and reliability of FL systems against backdoor attacks. The two main categories of defenses based on attack types are data and model defense, and privacy defense. These defense frameworks can be further categorized based on their working principles, as depicted in Figure 5.

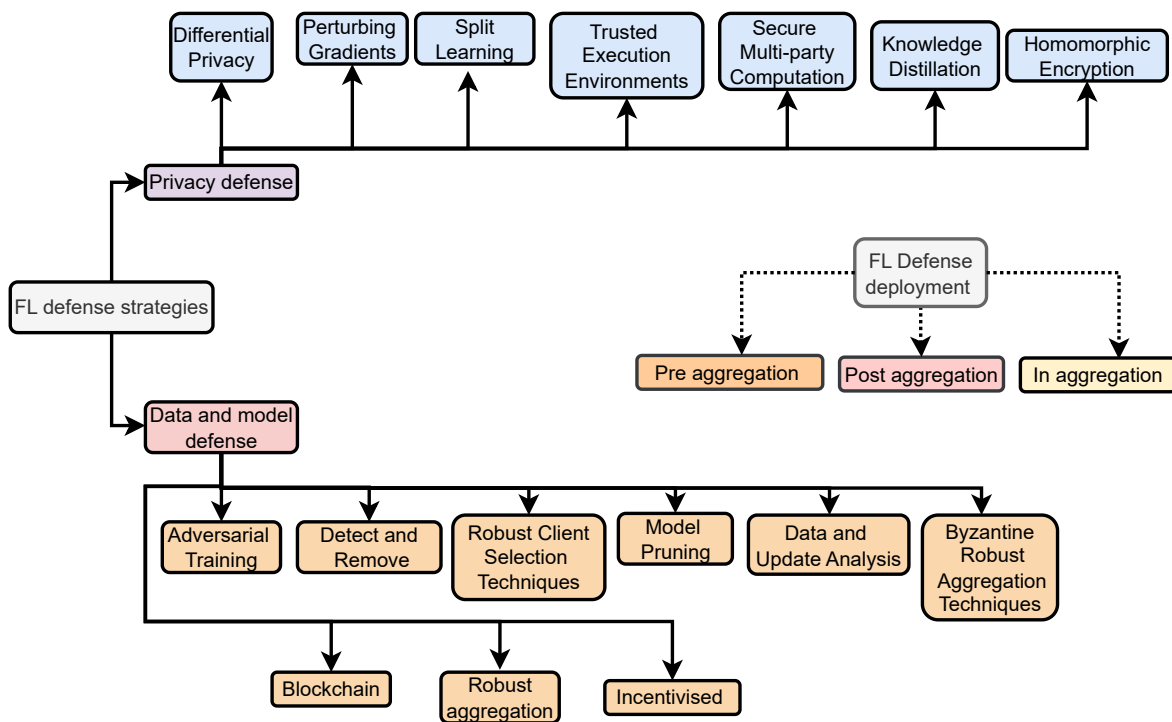


Figure 5. Types of defense strategies.

6. Defenses against Model and Data Attacks

To effectively counter the diverse threats posed by model and data attacks in FL, it is crucial to implement specialized defense mechanisms. The upcoming subsections will explore various strategies designed to detect and mitigate these attacks. We will discuss the methods employed to identify and remove malicious updates, enhance robustness through anomaly detection, and leverage secure aggregation techniques to ensure the integrity of the global model. Each approach addresses different aspects of the attack lifecycle, providing a comprehensive defense framework for FL systems.

6.1. Detect and Remove

This method is typically implemented on the server side and focuses on detecting and removing malicious updates to the model. It involves continuously monitoring incoming updates from clients and analyzing them for any signs of abnormality or inconsistency. Suspicious updates that do not conform to the expected pattern of legitimate updates are flagged. These deviations might be detected through statistical analysis, machine-learning-based anomaly detection, or by comparing the updates against a baseline of known good updates. Once identified, these flagged updates are removed from the aggregation process to prevent them from negatively impacting the model’s performance, thereby ensuring the integrity and reliability of the FL system. The block diagram of the detect and remove method is presented in Figure 6.

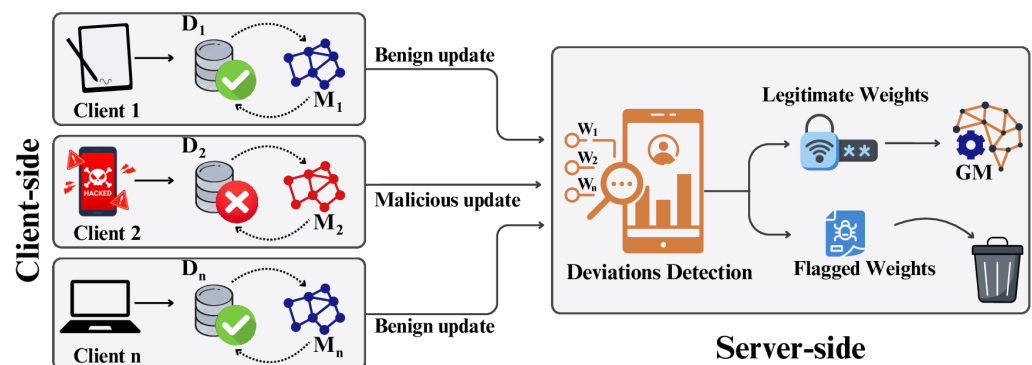


Figure 6. Visual representation of detect and remove defense strategy.

The defense framework presented in [82], called Local Malicious Factor (LoMar), is a two-phase algorithm designed to defend against poisoning attacks in FL. In Phase I, the Malicious Client Factor phase, each remote client update’s maliciousness is scored. This involves finding the k -nearest neighbors of each update to form a reference set and using kernel density estimation (KDE) to calculate a “maliciousness factor” $F(i)$ for each update. In Phase II, the Decision Threshold phase, a threshold is determined to classify the $F(i)$ scores as clean or malicious. The threshold is derived by combining the theoretical lower bound for malicious updates and the expected boundary value for clean updates in a trusted system. By applying this threshold, LoMar effectively identifies and defends against poisoning attacks in FL.

Similarly, the authors in [83] introduced a defense strategy called FederatedReverse, which involves reverse engineering, trigger generation, outlier detection, and model repair. Each participant generates local reverse triggers for each label by finding minimal perturbations that identify samples as specific labels, revealing backdoor triggers. These local triggers are sent to the central server, which aggregates them into robust global reverse triggers. The server then detects outliers among these triggers using an absolute median method, flagging abnormal scores that indicate attacks. If an attack is detected, the global reverse trigger for the infected label is added to each participant’s training data to “unlearn” the backdoor influence while preserving the main task accuracy. This approach enhances the security and reliability of FL systems.

The authors in [84] introduced a defense strategy termed Dynamic Defense Against Byzantine Attacks (DDaBA). The framework operates by dynamically deciding which client updates to aggregate and which to discard, specifically targeting potentially adversarial clients attempting to poison the model. Central to DDaBA is the use of an Induced Ordered Weighted Averaging (IOWA) operator for aggregation. This operator assigns weights to client contributions based on their performance on a validation dataset, ordering the clients from best to worst. A linguistic quantifier adjusts these weights dynamically, leveraging statistical outlier detection to identify and exclude potential adversarial clients. By adapting weights based on performance discrepancies modeled as an exponential

distribution, DDaBA effectively filters out adversarial influences while improving the overall performance of the FL model.

The SecFedNIDS framework [85] defends FL-based network intrusion detection systems against poisoning attacks with two main mechanisms. At the model level, it centrally detects and rejects poisoned local intrusion detection models by selecting critical parameters using gradient information and employing the Stochastic Outlier Selection (SOS) algorithm for real-time anomaly detection among local models. This prevents poisoned models from influencing the global model. On the client side, SecFedNIDS employs a data-level defense by identifying and filtering out poisoned traffic data using class path similarity analysis via Layer-wise Relevance Propagation.

Further, ref. [86] introduced two targeted defense strategies for FL environments. The first, Defense Against Convergence-Round Attack with Similarity Measurement (PASM), employs a pre-aggregation step to compute an interim model (wrP) before the final aggregation. By assessing the cosine similarity between wrP and each local model update (wri), updates with high similarity above a set threshold (e.g., 0.95–0.99) are identified as potentially malicious and excluded from the aggregation. This prevents backdoor attacks aiming to manipulate the model convergence. The second strategy, Defense Against Early-Round Attack with Backdoor Neuron Activation (ACCDR), uses the Data-Free TrojanNet Detector (DF-TND) to detect backdoor activations. It assesses models based on abnormal scores derived from comparisons between neuron representations of recovered patterns and original inputs. Models exceeding a threshold for abnormality are flagged as compromised and excluded from the aggregation, ensuring that the final model remains robust against early-round backdoor intrusions.

Another defense framework, called density-based anomaly detection [87], introduces a density-based algorithm to assess the anomaly scores for each client's uploaded local model, marking a model as anomalous if it stands out as denser than its closest neighbors. An anomaly score is computed based on how isolated a model is compared to its neighbors. Additionally, the framework proposes dynamically adjusting the weight of each local model during aggregation based on these anomaly scores.

BAFFLE is a defense framework designed to enhance the security of FL against backdoor attacks [61]. It operates through a feedback loop mechanism where, during each training round, a subset of clients independently validate the global model from the previous round using their private datasets. These clients assess the model for suspicious behavior by comparing its error rates across different classes with those of previously accepted "trusted" models. Significant discrepancies in error rates above a set threshold indicate potential backdoor presence, prompting the clients to vote on the model's integrity. The server then aggregates these votes and accepts the model only if the number of suspicious votes falls below a predefined quorum threshold.

Further, ref. [88] proposed a defense framework for FL against data poisoning attacks. It utilizes Kernel Principal Component Analysis (KPCA) for the dimensionality reduction of participant updates to capture data patterns and reduce noise. An algorithm (Algorithm 2) based on KPCA identifies malicious updates by measuring the deviation from the global model update. Optionally, K-means clustering may be applied after KPCA, although KPCA alone proves more effective due to its simplicity. The identified malicious participants are blacklisted or ignored in future FL rounds to prevent model poisoning. The framework includes real-time online detection to continuously monitor updates and take immediate mitigation actions like blacklisting, ensuring ongoing protection during FL training.

The authors in [89] proposed a defense for detecting abnormal samples, including out-of-distribution and adversarial ones. It derives a generative classifier from a pre-trained softmax neural model using Gaussian discriminant analysis (GDA), assuming that the class features follow Gaussian distributions. A confidence score based on Mahalanobis distance assesses the proximity of test samples to these distributions, gauging their probability density. Calibration techniques involve adding controlled noise to test samples and combining confidence scores from multiple layers to improve detection. The framework demonstrates

robustness under conditions like noisy labels, small training datasets, and minimal hyperparameter tuning without out-of-distribution data. It extends beyond anomaly detection to support class-incremental learning applications, showcasing its broad applicability.

6.2. Adversarial Training

Such defense frameworks are designed for data and model poisoning attacks. Since data are controlled by individual devices, these frameworks are placed at each local model. The working principle of these frameworks is based on teaching the framework identifying actual and poisoned data or updates. This is achieved by training the model on adversarial updates. The block diagram of adversarial training is presented in Figure 7. The authors in [90] presented the integration of adversarial training into the FL training process. In each communication round, the server distributes the global model to all the clients. The clients then train their local version of the model using a mix of real data and adversarial examples generated from their own private datasets. After training, the clients send both their regular model updates and the perturbed updates from the adversarial training back to the server. The server combines these updates—both natural and perturbed—from all the clients to refine the global model. The goal is that, by averaging the perturbed updates from various clients, the global model becomes more robust, similar to the effect of centralized adversarial training. This process is repeated over multiple rounds, with the clients continually training their local models and the server aggregating the updates, ultimately resulting in a robust final model.

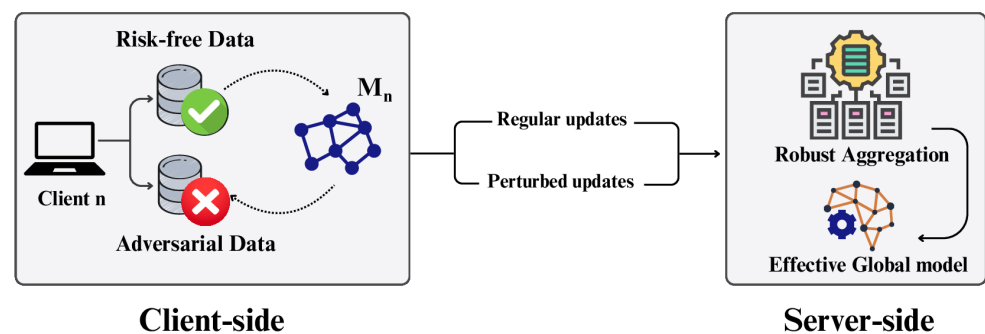


Figure 7. Visual representation of adversarial training.

The authors in [91] observed a significant decline in model performance when applying adversarial training directly within an FL setup compared to centralized training. This drop was due to model drift and non-IID data distribution. To address these issues, they proposed the FedDynAT framework. FedDynAT builds on FL algorithms like Federated Averaging (FedAvg) and Federated Curvature (FedCurv) [92] for model aggregation. Each client uses adversarial training with Projected Gradient Descent (PGD) to create adversarial examples from their local data. Unlike the traditional methods, FedDynAT employs a dynamic schedule for local training epochs (E) instead of a fixed number. The schedule starts with a high initial value (E_0) and decreases over communication rounds by a factor γE every F_E rounds, balancing convergence speed and model drift. FedDynAT uses FedCurv for model aggregation, adding a regularization term to minimize the drift between local models. The process alternates between local adversarial training with dynamic epochs, model sharing, and FedCurv aggregation until convergence is reached.

Another framework called Generative Adversarial Label Poisoner (GALP) [93] is a system to counteract label poisoning attacks in FL. It includes a generator (G) and a discriminator (D). The generator creates artificial label noise from random inputs, while the discriminator distinguishes between real and noisy labels. Noisy label classifiers, like LMNL and Masking, are trained on data poisoned by the generator, effectively “vaccinating” them against label poisoning attacks. In each round, benign clients send parameters of their vaccinated classifiers to the server, which aggregates these to update the global model. Malicious clients then initialize their local models with this global update. Since the update

includes vaccinated parameters, the malicious models are also vaccinated, reducing the impact of any local label poisoning attempts. The attacker's parameters have minimal effects on the global model. By combining GALP with noisy label classifiers, the framework ensures that the global model update is resistant to label poisoning attacks in FL.

6.3. Model Pruning

Pruning a neural network is a technique used to enhance the efficiency of the model by eliminating redundant or insignificant parameters, such as weights, neurons, or entire layers, without substantially compromising the network's performance [94]. This process involves identifying and removing those components that have minimal impacts on the network's output, thereby reducing the model's size and computational complexity [95]. The block diagram of model pruning is presented in Figure 8.

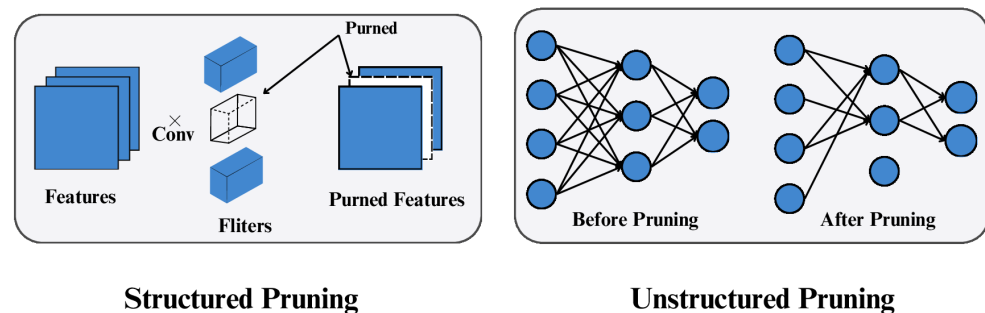


Figure 8. Model pruning.

There are two main types of pruning: structured and unstructured pruning. Structured pruning involves removing entire structures within the network, such as neurons, filters, or even layers [96]. This method simplifies the network architecture and leads to more efficient computation and memory usage as the remaining network can be more easily optimized for parallel processing and hardware acceleration. In contrast, unstructured pruning targets individual weights, removing the least important ones based on criteria such as magnitude [97]. While unstructured pruning can yield a highly sparse network, it often requires specialized hardware or libraries to fully capitalize on the sparsity as the remaining network structure is irregular.

Pruning is widely used in FL [98,99] and can significantly enhance the defense of local models by reducing the attack surface and improving model robustness [100]. By eliminating redundant or less important weights and neurons, pruning minimizes the impact of adversarial manipulations [101]. This reduction in model complexity makes it harder for attackers to inject malicious updates without detection [102]. Additionally, pruning improves the generalization ability of local models, making them less sensitive to minor perturbations introduced by attacks [103]. With fewer parameters, pruned models also reduce communication overhead during parameter sharing, enabling faster and more secure aggregation processes [17,98]. Pruning can be implemented on both the server and client sides.

6.4. Byzantine Robust Aggregation Techniques

Byzantine robust aggregation techniques in FL are designed to ensure that the global model can learn effectively even when some clients send malicious updates, either intentionally or due to faults. These techniques aim to mitigate the influence of these byzantine clients and ensure the integrity and performance of the global model. The block diagram of byzantine robust aggregation is presented in Figure 9.

One such technique is Krum [77], a robust algorithm for aggregating gradient vectors in distributed stochastic gradient descent (SGD). Krum selects the vector from clients with the smallest summed distance to its closest $n - f$ other vectors, where f represents the maximum number of byzantine clients. This selection ensures that the aggregated vector

aligns closely with the majority of the correct vectors, minimizing the impact of outliers or malicious updates. Krum satisfies the (α, f) -Byzantine resilience property, ensuring that the vector chosen aligns within an angle α of the true gradient direction. Variants like Multi-Krum combine Krum and averaging to balance resilience against byzantine failures with convergence speed.

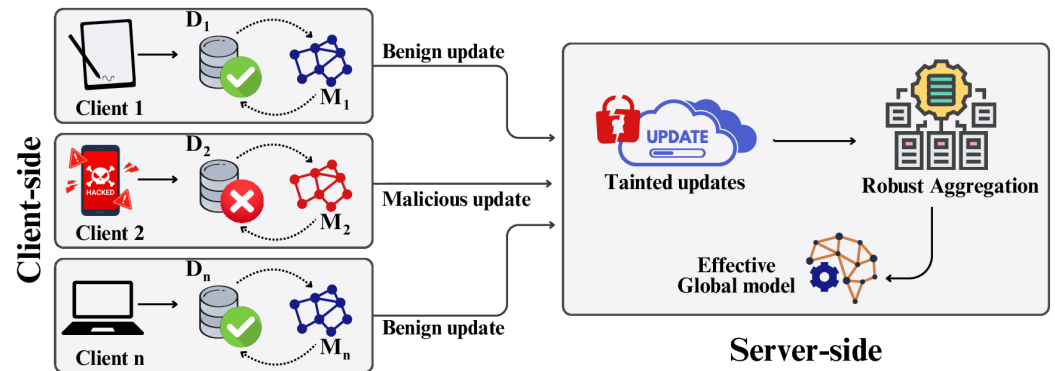


Figure 9. Visual representation of byzantine robust aggregation techniques.

Another approach, the trimmed mean [104], focuses on robust algorithms for distributed statistical learning in the presence of byzantine failures. It proposes two robust distributed gradient descent algorithms based on coordinate-wise median and trimmed mean operations, effectively handling arbitrary messages from byzantine machines. These algorithms achieve optimal statistical error rates across different loss function scenarios. A communication-efficient one-round algorithm based on a coordinate-wise median is also proposed for quadratic loss functions.

The Bulyan algorithm [105] addresses challenges in high-dimensional settings like neural networks. Bulyan aggregates gradients by majority vote per coordinate, significantly enhancing robustness and limiting the attacker’s influence to $O(1/\sqrt{d})$. This approach ensures effective learning under attack scenarios, as validated by both the theoretical analysis and empirical results.

RLR (Robust Learning Rate) [78] adjusts learning rates dynamically based on the signs of agent updates to counteract divergences caused by malicious agents with backdoors. By inverting the learning rate when necessary, RLR effectively reduces backdoor accuracy while maintaining overall model accuracy better than previous defenses. This approach enhances the collective impact of updates from honest agents on model predictions for compromised inputs.

ZeKoC [106] enhances FL’s resilience against poisoning attacks by using MaxMin initialization and a robust distance metric in K-means clustering. Before aggregation, the clusters are evaluated with a test dataset to exclude those degrading performance. This filtering detects and removes adversarial updates from compromised nodes while aggregating enough nodes for an effective global model.

ShieldFL [107] introduces a defense strategy using secure cosine similarity over encrypted gradients to combat model poisoning attacks. It employs a two-trapdoor homomorphic encryption scheme to detect poisonous gradients. ShieldFL includes a byzantine-tolerance aggregation mechanism, ensuring robustness across diverse data distributions. Gradients are weighted by confidence values rather than discarding outliers outright, preserving privacy and minimizing the influence of malicious gradients on the final model update.

The Adaptive Federated Averaging (AFA) algorithm [108] secures FL against faulty or malicious client updates with a robust aggregation rule. It detects and discards outlier updates by comparing their similarity to the median and standard deviation of the aggregated model update. A Bayesian Hidden Markov Model estimates each client’s probability of providing beneficial updates over iterations, helping to identify and block consistently malicious clients.

Similarly, FLTrust [109] introduces a robust framework for FL by establishing a root of trust through a clean training dataset collected independently by the server. FLTrust evaluates the trustworthiness of each local update using a “trust score”, determined by the similarity of its direction to that of the server model update. This approach minimizes the influence of unreliable client contributions by normalizing the magnitudes of local updates and computing the final global model update as a weighted average based on the trust scores.

These byzantine robust aggregation techniques collectively ensure the security and reliability of FL by addressing the various threats posed by malicious clients.

6.5. Robust Client Selection Techniques

Robust client selection techniques are implemented on the server side to ensure that only the most dependable and trustworthy clients contribute to the FL process. This minimizes the likelihood of incorporating malicious updates and helps to maintain the integrity of the model aggregation process. This process is illustrated in Figure 10.

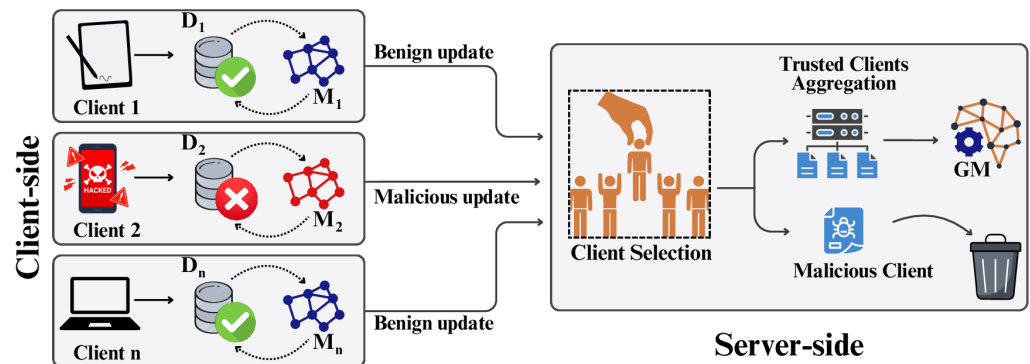


Figure 10. Visual representation of robust client selection.

Building on this concept, the authors in [110] proposed an ensemble framework to defend against malicious clients in FL. By training multiple global models using randomly sampled subsets of clients, this approach mitigates the impact of malicious updates. During inference, the ensemble model aggregates predictions via majority voting, providing resilience against a bounded number of malicious clients. A “certified security level” for each test example is derived based on the gap between the highest and second-highest label probabilities across the models, indicating robustness. This adaptable defense maintains high “certified accuracy” against adversarial behaviors, as validated empirically, demonstrating the importance of secure client selection.

Expanding on the need for diversity and accuracy in client selection, the authors in [111] introduced the DivFL framework. This approach selects a diverse subset of clients through submodular maximization to approximate the full gradient with minimal error. Using a greedy algorithm, DivFL efficiently maximizes a submodular facility location function to select representative clients. Integrated with FedAvg, DivFL substitutes averaging over all the clients by aggregating updates only from the selected diverse clients after they perform local SGD updates on their data. The framework’s convergence analysis shows that it achieves an $O(1/T)$ convergence rate similar to FedAvg, with an additional error term dependent on the gradient quality. This ensures effective performance, particularly in scenarios with non-IID and heterogeneous client data distributions, emphasizing the value of strategic client selection for robust FL.

Complementing these techniques, FedClean offers a robust defense framework against parameter poisoning attacks in FL [112]. It employs a reputation-based agent sampling strategy using Bayesian principles to assign reputation scores, prioritizing agents with higher reputations for contributing to the global model update. Additionally, FedClean features a peer truth-serum-aided hypersphere-based update selection mechanism that utilizes

a hypersphere classifier and an oracle for distinguishing between honest and malicious updates. This approach, combined with reputation-based weighted averaging aggregation and an oracle-driven detection framework, enhances security without compromising model convergence and performance stability, comparable to FedAvg.

6.6. Data and Update Analysis

This defense strategy can be implemented on both the server and client sides in FL. On the server side, it focuses on examining aggregated client updates to detect suspicious patterns or inconsistencies. Techniques such as statistical analysis, outlier detection, and machine-learning-based anomaly detection are used to identify potential malicious activity or data manipulation. On the client side, the approach involves preprocessing data, conducting data quality checks, validating, and cleaning the data to ensure their integrity before training. This dual-layered approach helps to maintain the security and reliability of the FL system by addressing potential threats at multiple points in the data lifecycle. Clients also verify their local updates before sending them to the server. This framework is graphically presented in Figure 11.

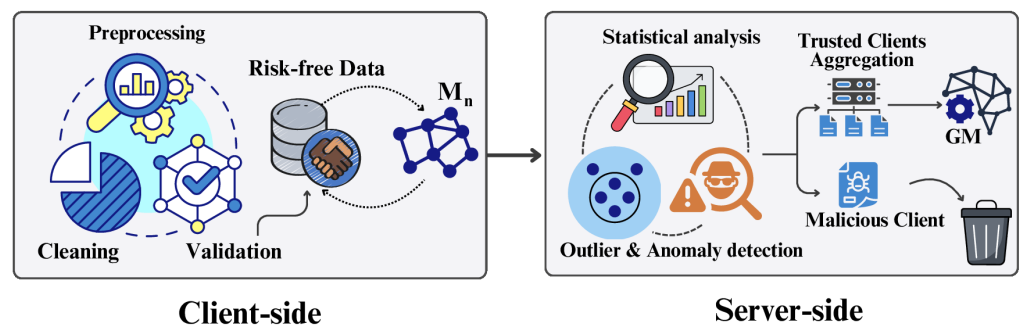


Figure 11. Visual representation of data and update analysis.

The defense strategy outlined in [113] introduces a proactive security measure named Siren, aimed at strengthening the resilience of FL systems against byzantine attacks. Siren is implemented on both the client and server sides. On the client side, each client engages in a dual process involving training and alarming. Clients reserve a portion of their local dataset as a test set to assess the performance of the global model. During each communication round, clients receive the global model weights from the server and compare them against the performance of their local model on the test dataset. If the global model's accuracy exceeds a predefined threshold compared to the local model, the client proceeds with training using the global model and informs the server that the model is reliable. However, if the global model's performance falls below the local model, the client continues using its local model and alerts the server about a potential issue with the global model. On the server side, the server gathers these alerts and leverages them to identify suspicious updates. By consolidating this feedback, the server can recognize patterns and determine if the global model has been compromised. This enables the server to exclude dubious updates and uphold the integrity of the FL process.

The defense strategy described in [114], called DeepSight, involves a multi-layered approach to protect against dynamic backdoor attacks in FL. The first layer is a classification-based filtering that assesses the homogeneity of a model's training data using a metric called "Threshold Exceedings". This enables deep analysis of individual models to classify them as benign or suspicious. The second layer employs clustering to group model updates based on similar training data, separating malicious from benign updates. Techniques like Division Differences and Normalized Energy Updates characterize the training data. The third layer combines the classification and clustering results to identify clusters containing poisoned models, deciding whether to exclude or accept each model update. The fourth layer enforces constraints on model updates through weight clipping, mitigating attacks by making any backdoor behavior negligible during aggregation. Finally, the aggregation

layer minimizes the effect of any remaining weakly trained poisoned models, ensuring their minimal impact on the overall model aggregation. This multi-layered approach provides robustness such that, even if one component fails to detect poisoned models, others can compensate to maintain integrity. By combining these techniques, DeepSight ensures the integrity and performance of the aggregated model.

The Biscotti framework [115] uses a decentralized peer-to-peer approach with blockchain technology to enhance the security and privacy of FL. It addresses key vulnerabilities in FL, such as poisoning and information leakage attacks. Biscotti employs a Proof-of-Federation consensus protocol where a peer's contribution determines their role. Consistent hashing and verifiable random functions prevent Sybil attacks and randomly select peers for protocol stages. The Multi-Krum algorithm defends against poisoning attacks by rejecting updates that deviate from the majority. Differential privacy and secure aggregation protect against information leakage by adding noise to updates and obfuscating individual updates. The blockchain provides transparency and auditability to verify the training process integrity. Biscotti achieves decentralization, reducing the risk of a single point of failure, and enhances robustness against attacks. It scales efficiently with participants and handles peers joining and leaving. By integrating Multi-Krum, differential privacy, and secure aggregation, Biscotti offers a multi-layered defense against poisoning and information leakage attacks, empowering users to control their sensitive information and enhancing privacy and trust.

The authors introduced FL-Defender [69], a framework designed to counteract targeted poisoning attacks in FL. It engineers robust features by analyzing the last layer gradients of local model updates and computed cosine similarities to identify gradient direction differences indicative of attacks. Principal Component Analysis (PCA) compresses the gradient similarity matrix, enhancing attack detection. FL-Defender aggregates these features into a centroid and measures each worker's deviation from it, using accumulated similarities over training rounds to assign long-term trust scores. Updates from workers far from the centroid receive lower weights during federated averaging. This method, which focuses on the last layer gradients, was shown to be effective against label flipping and backdoor attacks across various datasets.

6.7. Blockchain

Blockchain technology plays a crucial role in enhancing security within FL environments [116–118]. It serves as a decentralized and immutable ledger that securely records all the data exchanges and model updates between participating entities, ensuring transparency and accountability [119]. Smart contracts, which automate tasks like model aggregation and verification, operate without relying on a central authority [120]. This decentralized approach mitigates the risk of single points of failure and deters malicious actors from corrupting the system [49]. By leveraging blockchain, FL safeguards against threats such as data tampering and adversarial attacks, ensuring the verifiability of contributions and prompt identification of anomalies [121]. This integration significantly bolsters the security and reliability of collaborative learning environments, addressing the inherent challenges of transparency and trust in distributed settings. The schematic representation of this process is shown in Figure 12.

For instance, in the paper by Mao et al. [122], a blockchain-based anomaly detection framework for FL was proposed to safeguard against model poisoning attacks. Incorporating a blockchain network enables identification and filtration of malicious or low-quality local model updates from compromised participants. Updates undergo validation using datasets and an accuracy threshold before aggregation, ensuring only high-quality contributions impact the model. Moreover, the framework employs incentives to deter malicious behavior, enhancing anomaly detection and protecting against accuracy compromise.

Similarly, Batool et al. [123] introduced Block-FeST, a blockchain-based federated split learning framework for anomaly detection. This framework integrates FL and split learning using Transformers to maintain data privacy and address client resource constraints.

Blockchain orchestrates FL and split learning processes, ensuring model hash transparency and immutability. A smart contract on the blockchain coordinates training, generates audit trails, and prevents tampering, thereby enhancing the anomaly detection efficiency.

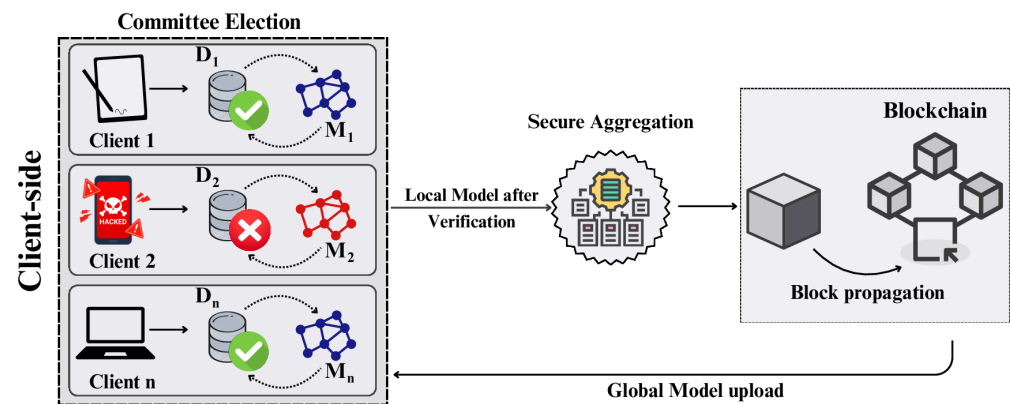


Figure 12. Visual representation of blockchain.

Furthermore, Zhang’s work [124] introduced a defense framework for IoT device failure detection using blockchain-based FL. Blockchain anchors hashed client data, ensuring data integrity and privacy. Smart contracts incentivize client participation and transparently record contributions, effectively defending against data leakage and non-participation attacks.

Moreover, Sarhan et al. [125] proposed Hierarchical Blockchain-based Federated Learning (HBFL) for cyber threat intelligence (CTI) sharing. HBFL employs blockchain to host global model aggregation and monitor updates, ensuring security and privacy while facilitating collaborative intrusion detection model training.

6.8. Incentivized Federated Learning (FL)

Incentivized FL is a methodology designed to promote active and constructive participation among distributed clients in training machine learning models while upholding data privacy and fairness standards [126,127]. Unlike traditional FL, where clients like mobile devices or IoT sensors collaborate to enhance a global model without sharing raw data, incentivization becomes crucial to motivate effective client contributions [128]. This approach ensures that all the participants are invested in the collective learning process, incentivized based on their contributions, which can include model updates, computational resources, or improvements in data quality [129]. These incentives not only boost engagement but also cultivate a cooperative environment where the clients are encouraged to adhere to protocols and perform tasks honestly [48,130]. The structure of the framework is depicted in Figure 13.

Guo et al. [131] proposed an incentivized FL framework focused on defending against and managing abnormal clients, essential for enhancing the accuracy and resilience of FL. This framework typically comprises two critical elements: an abnormal client detection module and client incentives. The detection module utilizes techniques such as outlier identification to pinpoint clients exhibiting unusual behavior due to faults or attacks, ensuring that their data updates do not disrupt the global model averages during aggregation. However, outright rejection of updates from abnormal clients risks losing valuable data and reducing their engagement. This is where client incentive mechanisms play a crucial role. These mechanisms encourage continued participation from all clients, including those identified as abnormal, in the subsequent training rounds. By assigning “credit scores” based on the client contributions and behaviors, these incentives motivate sustained involvement and improvement over time, thereby discouraging client dropout and promoting overall model integrity.

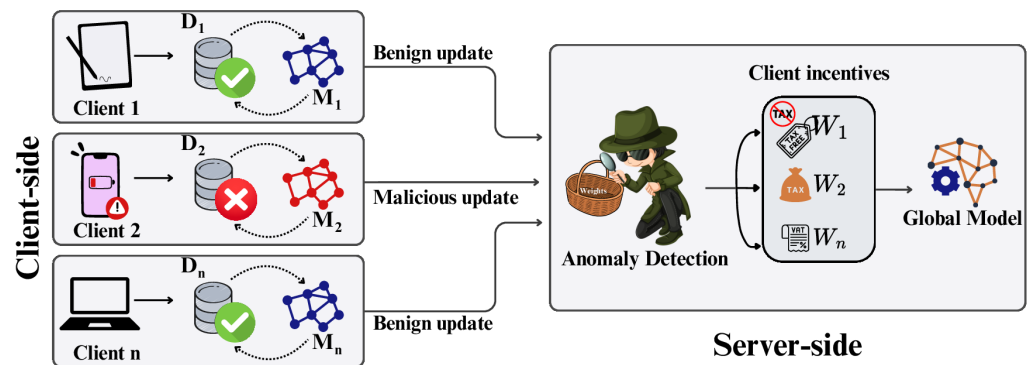


Figure 13. Visual representation of Incentivized Federated Learning.

The paper by Bai et al. [132] introduced ISPPFL (Incentive Scheme based Privacy-Preserving Federated Learning), an incentive-based defense framework for avatars in the metaverse. ISPPFL comprises three key components: (1) a privacy risk auditor that evaluates various privacy risks in the FL model; (2) a perturbation generation mechanism that creates perturbed samples using adversarial algorithms to enhance robustness against privacy attacks; and (3) an adaptive incentive mechanism that dynamically adjusts the intensity of noise generation based on the privacy risk indicator, model accuracy, and the accuracy difference between consecutive training epochs. This adaptive incentive mechanism aims to balance privacy protection and model performance. By integrating these components, ISPPFL effectively defends against privacy attacks such as Source Inference Attacks (SIAs) in Horizontal FL and Label Inference Attacks (LIAs) in Vertical FL while maintaining model accuracy. The framework is flexible and adaptable to different types of privacy risks and defense mechanisms in the metaverse context.

6.9. Regularization

Regularization serves as a defensive strategy in FL, safeguarding both the server and client sides against data and model poisoning attacks. On the server side, regularization is applied during the aggregation of model updates to prevent overfitting and improve model generalization. Techniques such as L1 and L2 regularization introduce penalty terms in the loss function during training, which helps to control the complexity of the shared model and ensures robust pattern learning from local data [133,134]. On the client side, participants implement regularization methods like dropout, batch normalization, or weight decay during their training processes [135,136]. These techniques help to prevent overfitting on local data, enhancing the overall robustness and resilience of the FL system. A diagrammatic overview of the process is provided in Figure 14.

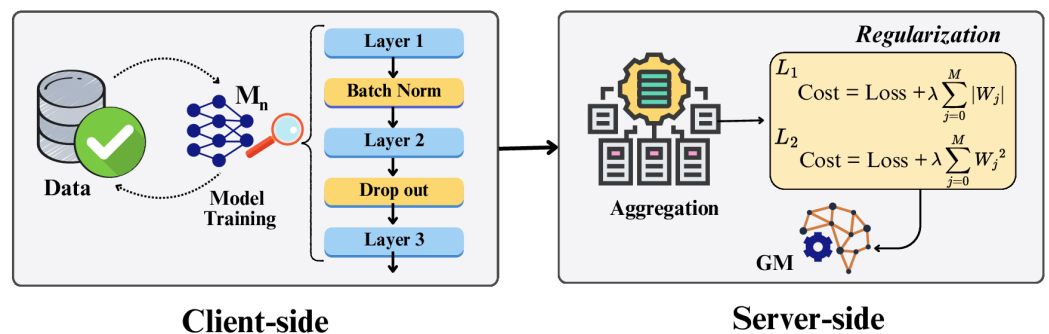


Figure 14. Visual representation of regularization.

The work by Jiang et al. [137] introduced a defense framework named Local Self-Regularization (LSR) to address the challenge of training on noisy labeled data in FL while maintaining data privacy. LSR focuses on regularizing the local training process on

each client through two main components. First, implicit regularization employs MixUp prediction to boost the model's discrimination confidence and prevent it from memorizing noisy labels. Second, explicit regularization uses self-knowledge distillation to minimize the discrepancy between the model outputs of original and augmented instances of the same sample. This method leverages data augmentation to gain additional supervision without needing an auxiliary clean dataset or compromising FL's privacy principles. By applying these regularization techniques locally, LSR aims to mitigate the performance degradation caused by noisy labels in FL environments. The authors validate the effectiveness of their approach with experiments on benchmark datasets and illustrate its potential to integrate with existing state-of-the-art methods for further performance enhancements.

Similarly, Chen et al. [138] introduced Contractible Regularization (ConTre), a defense framework for FL designed to address the challenges of non-IID data, especially in image classification tasks. ConTre works by regularizing the latent space of local models to prevent them from prematurely converging to suboptimal local optima. It achieves this by projecting input images into a latent space and using a contrastive-learning-based regularizer to differentiate between images of the same class while clustering images from different classes. ConTre can be easily integrated into existing FL frameworks without additional parameters and gradually reduces its influence during training to avoid potential side effects as the process converges. The authors demonstrate that ConTre significantly enhances the performance of various FL methods on both natural and medical image datasets under non-IID conditions. It also accelerates convergence, allowing models to reach the target accuracy in fewer communication rounds compared to methods without ConTre.

All the above-described defense frameworks are summarized in Table 2.

Table 2. Data and model defense frameworks for FL.

No.	Defense Framework	Definition	Implementation Stage	References
1	Detect and Remove	Identifies and eliminates malicious updates or compromised clients using anomaly detection or statistical analysis to protect the global model's integrity.	Pre-aggregation	[61,82–89]
2	Adversarial Training	Models are trained using adversarial examples to enhance robustness against attacks on data and model updates.	Pre-aggregation	[91,93]
3	Model Pruning	Reduces model size by removing unnecessary parameters, making it harder for adversaries to inject malicious updates while improving efficiency.	Pre-aggregation	[102,103]
4	Byzantine/Reliable Aggregation	Aggregates updates consistent with the majority to filter out outliers from potentially compromised clients.	In-aggregation	[77,78,104–109]
5	Robust Client Selection	Selects clients based on reliability, reducing the impact of malicious or unreliable clients on the global model.	Pre-aggregation	[110–112]
6	Data and Update Analysis	Analyzes local data and model updates to detect inconsistencies or suspicious behavior indicative of adversarial attacks.	Pre-aggregation/Post-aggregation	[69,113–115]
7	Blockchain	Adds transparency, immutability, and auditability to FL using blockchain, preventing tampering and ensuring trustworthy aggregation.	Post-aggregation	[122–124]
8	Incentive Federated Learning	Rewards client participation, encouraging honest contributions and dissuading malicious actions.	Pre-aggregation	[131,132]
9	Regularization	Mitigates overfitting by adding constraints to model updates, improving generalization and reducing vulnerabilities to attacks.	Pre-aggregation	[137,138]

7. Defenses against Privacy Attacks

Privacy attacks in FL pose significant risks to the confidentiality of sensitive data. To safeguard against these threats, a variety of defense mechanisms have been developed. The upcoming subsections will evaluate these strategies, highlighting methods such as differential privacy, Secure Multi-party Computation, and homomorphic encryption. We will examine how these techniques protect individual data contributions from inference and reconstruction attacks, ensuring that privacy is maintained throughout the FL process.

Each approach offers unique advantages and challenges, contributing to a robust privacy defense framework for FL systems.

7.1. Homomorphic Encryption

Homomorphic encryption is a cryptographic technique that enables computations on encrypted data without decryption, crucial for preserving data privacy and security in FL [139]. In FL systems, homomorphic encryption enables the aggregation of model updates from multiple participants while keeping individual contributions encrypted [140]. This method ensures that sensitive data remain protected throughout the aggregation process as computations are performed on ciphertexts rather than plaintexts. By supporting operations like addition and multiplication on encrypted data, FL servers can derive a global model without accessing raw, sensitive information from participating clients [141]. This capability is essential for maintaining confidentiality and fostering trust in FL environments, empowering data owners to collaborate on model training tasks while retaining control over their information [142]. Figure 15 displays the block representation of homomorphic encryption.

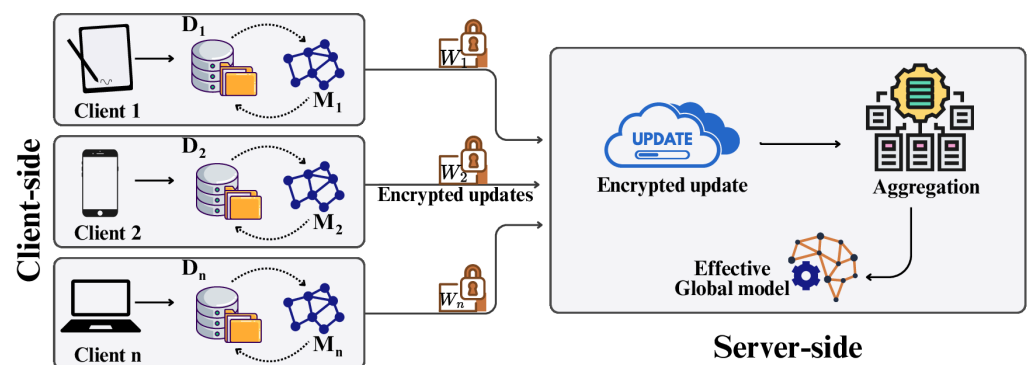


Figure 15. Visual representation of homomorphic encryption.

For instance, the authors in [143] introduced Vertical Federated Learning (VFL), where data are partitioned among parties, each holding different features of the same samples. To safeguard privacy, communications are protected using techniques such as homomorphic encryption, ensuring that individual gradients or intermediate results remain inaccessible. The framework addresses privacy threats from attackers attempting to infer private sample labels from batch-averaged gradients, employing methods like differential privacy and gradient sparsification. Additionally, a confusional autoencoder (CoAE) is utilized to obscure true labels with “soft fake labels”, preserving accuracy while concealing sensitive information.

Similarly, the Privacy-Enhancing Federated Learning (PEFL) framework proposed in [144] emphasizes data privacy and robustness against attacks in federated settings. PEFL employs homomorphic encryption (HE) to maintain the confidentiality of training data and local gradients throughout the FL process. Users encrypt their gradient vectors using the cloud platform’s public key before transmission, ensuring that secure aggregation protocols can operate within an encrypted domain without revealing individual user data. PEFL also defends against adversarial poisoning attacks through adaptive federated aggregation techniques, leveraging coordinate-wise medians to benchmark gradient similarity and mitigate the influence of outliers on model training. This approach enhances PEFL’s resilience against data poisoning attempts, reinforcing the integrity and reliability of collaborative learning in federated environments.

7.2. Knowledge Distillation

Knowledge distillation (KD) involves transferring knowledge from a large model (the “teacher”) to a smaller model (the “student”) by training the student to mimic the teacher’s

predictions [145]. This technique helps to create efficient yet accurate models suitable for various applications [146]. A visual diagram of the approach is shown in Figure 16.

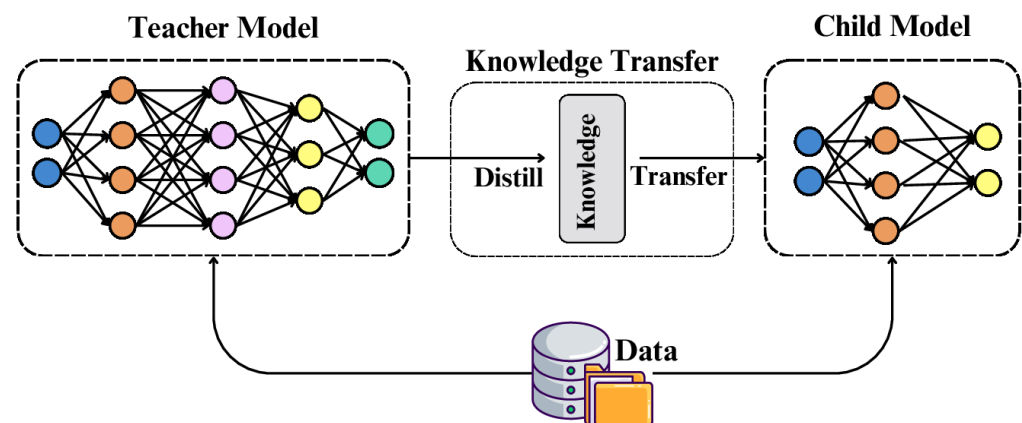


Figure 16. Visual representation of knowledge distillation.

In FL, KD plays a crucial role in enhancing the model performance by enabling the creation of efficient student models directly on decentralized devices [147]. A powerful teacher model, trained either centrally or collaboratively, distills its knowledge to student models located on individual devices [148,149]. This approach alleviates the computational burden on edge devices while preserving high accuracy and generalization, thereby optimizing the overall efficiency of FL systems [150].

FedKD (Federated Knowledge Distillation) [151] further advances privacy preservation by focusing on the exchange of updates from smaller “mentee” models instead of transmitting full, large mentor models. This approach significantly reduces the volume of information shared, thereby minimizing potential privacy risks. Moreover, FedKD employs gradient encryption before uploading the local gradients to the server, ensuring that the sensitive patterns within the gradients are obscured and protected. Gradient compression using techniques like Singular Value Decomposition (SVD) also contributes to minimizing sensitive information leakage. FedKD dynamically adjusts compression precision, prioritizing higher precision during convergence when gradients contain less private information. This holistic approach in FedKD ensures robust privacy preservation while maintaining effective model performance.

Additionally, the FEDGEN framework [152] emphasizes privacy by focusing on learning a generator model solely from the prediction layers of the local user models, thereby avoiding direct access to private input data. By sharing only prediction layers rather than entire models, FEDGEN reduces privacy risks as the prediction layers contain less information about the underlying private training data. The generator model in FEDGEN operates within a compact latent space, significantly less information-dense than high-dimensional input spaces, further minimizing the information shared and lowering the privacy breach risks. An extended version of FEDGEN retains the feature extraction layers locally while sharing only prediction layers, reducing the communication overhead and enhancing the privacy protection by keeping sensitive data securely on local devices.

Similarly, the FedFTG framework [153] innovatively enhances privacy through data-free knowledge distillation. FedFTG uses pseudo-data generated by models instead of real client data, ensuring that the actual data are never transmitted, thus preserving user privacy. The pseudo-data capture only high-level patterns and cannot reconstruct the original training samples, safeguarding the individual client data attributes. This method effectively keeps real data local and private, mitigating privacy breach risks.

7.3. Secure Multi-Party Computation

Secure Multi-party Computation (SMPC) allows multiple parties to collaboratively compute a function on their individual inputs while ensuring the confidentiality of personal

information [154,155]. This technique plays a critical role in maintaining data privacy across applications such as financial transactions, healthcare analytics, and decentralized decision-making processes [156]. By enabling computations without disclosing sensitive data, SMPC ensures that information remains protected throughout distributed networks. A graphical representation of the process is illustrated in Figure 17.

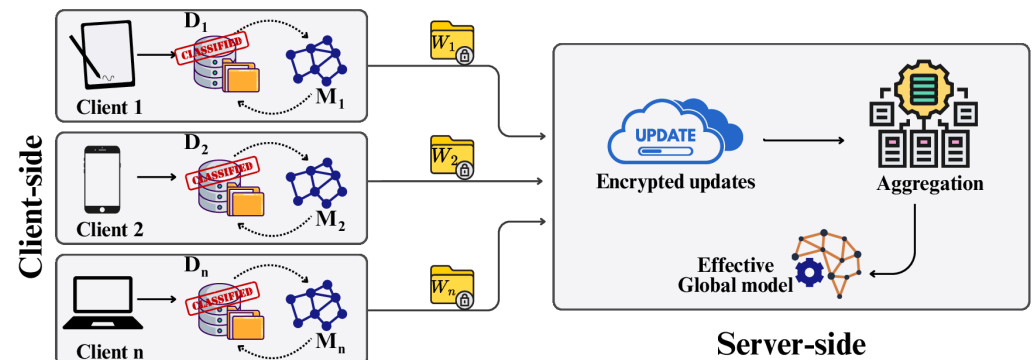


Figure 17. Visual representation of Secure Multi-party Computation.

Augmented Multi-party Computation (AMPC), introduced by Zhang et al. [157], enhances privacy in FL. AMPC aims to safeguard local training data and models from other participants and the central server, assuming the server is “honest-but-curious”. Utilizing MPC, AMPC distributes local models into secret shares, with public shares for the server and private shares locally, thereby preventing gradient leakage and enhancing the overall security.

Additionally, Mugunthan et al. [158] explored privacy in FL by combining secure MPC with differential privacy. Their protocol allows clients to securely compute the weighted averages of local model updates while keeping their data and model weights confidential from the server. To counter collusion risks among clients, the protocol integrates differential privacy mechanisms that mask the individual weights and maintain the privacy of the aggregated results through noise addition.

Byrd et al. used a similar approach of combining differential privacy and MPC to protect participant privacy in FL [159]. Differential privacy adds noise to model weights to protect against information disclosure, while MPC encrypts weights using shared values between client pairs. This dual-layered defense ensures that even colluding clients receive only noisy approximations, strengthening the privacy defenses in FL scenarios. These methodologies collectively advance the privacy protections necessary for secure collaborative computations in distributed environments.

7.4. Split Learning

Split learning divides the machine learning model between the client and server, ensuring that sensitive data remain on the client side while enabling distributed model updates [160,161]. This approach protects against attacks aiming to steal or analyze private information by minimizing data exposure during model updates [162,163]. By keeping data decentralized, split learning not only enhances privacy but also facilitates collaborative model training across distributed networks [164]. Figure 18 presents the block schematic of split learning.

For instance, SplitLearn [165] enhances model privacy through distributed training mechanisms. It partitions the neural network model between clients and servers, ensuring that no single entity can access the complete model. During training, only intermediate representations such as activations and gradients are exchanged, preventing the reconstruction of the entire model from any single source. This decentralized approach supports flexible deployment options based on trust levels among the participants, maintaining robust privacy protections while enabling incremental training by new clients.

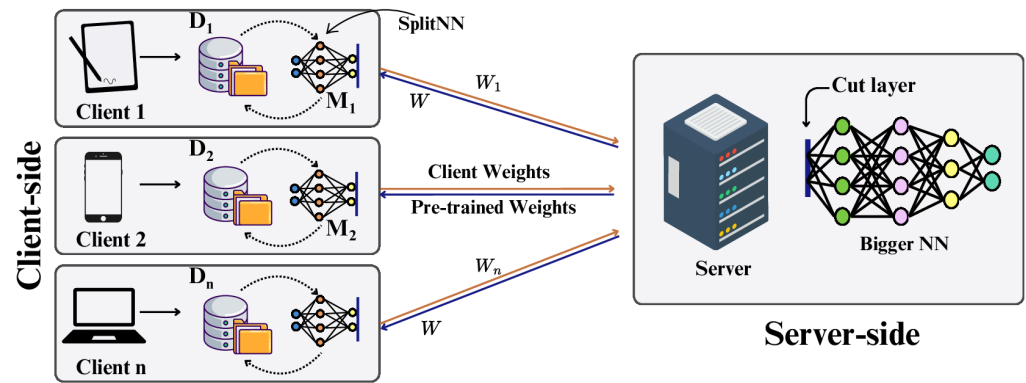


Figure 18. Visual representation of split learning.

Additionally, Federated Split Learning (FSL) [166] addresses security concerns by focusing on securing intermediate data exchanged during training. It applies privacy-aware loss functions and integrates differential privacy techniques to add noise, preventing attackers from reconstructing the original training data. FSL achieves strong accuracy–privacy trade-offs compared to the traditional methods, underscoring its effectiveness in protecting sensitive data and ensuring reliable FL processes.

Moreover, Splitfed Learning (SFL) [167] incorporates robust frameworks to safeguard sensitive information. SFL divides the model between client and server, ensuring that neither party has access to the complete model parameters, thereby bolstering privacy. It integrates differential privacy techniques by perturbing the model gradients with calibrated noise and incorporates PixelDP noise layers to enhance the privacy protections during data transmission. Together, these approaches fortify the privacy of shared data in FL, providing robust defense against privacy breaches.

7.5. Perturbing Gradients

Perturbing gradients involves adding random noise to gradients during model training to protect sensitive data from unauthorized access [168,169]. This technique disrupts gradient-based attacks used by adversaries to reverse-engineer models or extract confidential information [170,171]. By obscuring the original gradient structure and details, perturbing gradients makes it challenging for attackers to infer sensitive data or manipulate model parameters, thereby enhancing security in scenarios where data privacy is paramount [172]. Figure 19 outlines the process block diagram of perturbing gradients.

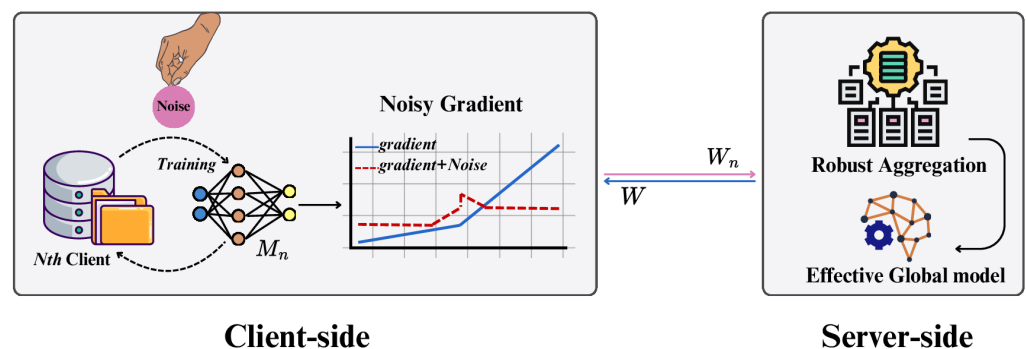


Figure 19. Visual representation of perturbing gradients.

For example, Liao et al. [173] introduced a privacy framework for over-the-air Federated Learning (OtA FL) that utilizes spatially correlated additive perturbations to enhance privacy during wireless transmission of model updates. These perturbations cancel out at the central server, ensuring strong (ϵ, δ) -differential privacy against eavesdroppers without compromising the learning accuracy. The framework optimizes the perturbation

parameters to balance privacy and model convergence, offering robust privacy guarantees supported by the theoretical analysis and simulations.

Similarly, Wang et al. [174] proposed a privacy framework to combat gradient leakage attacks in FL. Their approach assesses the information leakage risk across the model layers and applies local random perturbations to gradients before transmission to the server. This strategy complicates attackers’ efforts to identify and exploit noisy gradients, bolstering privacy protection. Global gradient compensation techniques ensure model convergence, maintaining robust privacy while preserving the learning performance.

In another approach, Sun et al. [175] developed the Soteria framework to address the privacy risks posed by data representations learned during FL. By perturbing these representations before transmission, Soteria minimizes privacy leakage while preserving model utility. It strategically adds noise to degrade the quality of reconstructed data, ensuring analytical guarantees on privacy defense and algorithm convergence. The experimental results demonstrate Soteria’s effectiveness in significantly improving privacy protection without sacrificing model accuracy.

Additionally, Lee et al. [176] proposed a privacy-preserving framework for FL that integrates a digestive neural network (DgstNN) to transform original data representations. This transformation maximizes the disparity between the gradients of original and transformed data, rendering gradients indecipherable to attackers while minimizing classification error. The framework optimizes privacy–utility trade-offs using predefined parameters, enhancing the model privacy against inference attacks compared to the traditional differentially private methods.

7.6. Differential Privacy

Differential privacy [177] provides a robust framework for safeguarding privacy in FL by injecting random noise into model updates, such as gradients, before they are transmitted to the central server [178]. This noise, calibrated to the sensitivity of the data, ensures that individual data points cannot be discerned from the updates, effectively preventing inference attacks [179]. By maintaining confidentiality regarding data contributions, even in collaborative learning settings, differential privacy enables a controlled balance between data privacy and model accuracy. A block illustration of this technique is provided in Figure 20.

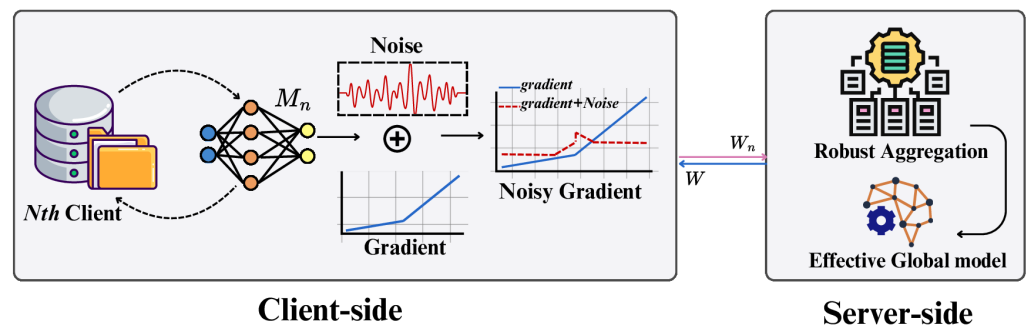


Figure 20. Visual representation of differential privacy.

For example, Wei et al. [177] introduced NbAFL (“noising before model aggregation FL”), leveraging differential privacy to enhance privacy preservation in FL. Clients perturb their locally trained model parameters with Gaussian noise before uploading to the server, ensuring that sensitive information remains obfuscated during transmission. The server aggregates these noisy parameters into a global model, potentially adding further noise based on privacy budget constraints. Theoretical analyses validate NbAFL’s differential privacy guarantees and demonstrate its utility in maintaining model performance across varying privacy levels and client configurations.

Additionally, Lecuyer et al. [180] applied differential privacy to address adversarial examples in deep neural networks through PixelDP. By treating each input example as

a database protected by differential privacy, PixelDP adds noise based on sensitivity at the neural network layer, thereby ensuring that differential privacy holds for the entire network's outputs. This approach provides certified robustness against adversarial attacks, limiting the impact of minor input changes on the network predictions and bolstering security in complex real-world datasets.

7.7. Trusted Execution Environments (TEEs)

In FL, Trusted Execution Environments (TEEs) provide secure enclaves within client devices or servers, ensuring that sensitive model training processes are protected from tampering or unauthorized access [168]. This security measure is vital in preventing privacy breaches where adversaries might attempt to intercept or manipulate model updates exchanged between clients and the central server. By securely executing computations within TEEs, FL systems safeguard the confidentiality and integrity of participant data, thereby fostering trust among clients and encouraging greater participation in collaborative learning tasks [181]. TEEs in FL play a critical role in maintaining the privacy of individual data contributions while enabling effective and secure collective model training across distributed networks. The schematic representation of this process is shown in Figure 21.

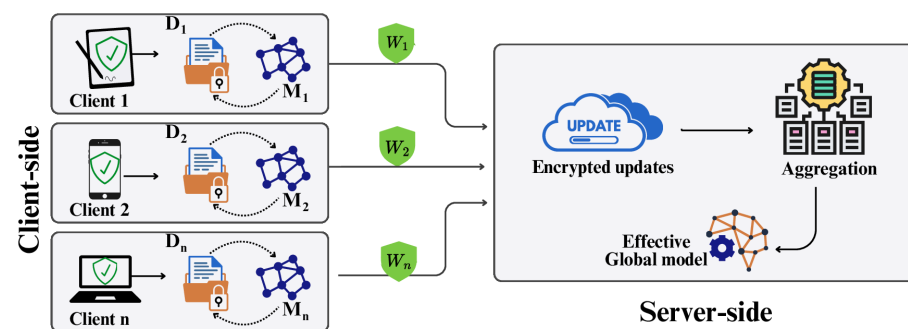


Figure 21. Visual representation of Trusted Execution Environments (TEEs).

Further, ref. [182] introduces a privacy-preserving FL scheme to defend against causative attacks using TEEs like Intel SGX. The participants and the aggregation server run their algorithms within secure TEEs, ensuring the integrity of the local training and global aggregation processes. The framework includes a training-integrity protocol, implementing verified programs within TEEs, and employs remote attestation to verify the integrity of the enclave execution. This setup detects and excludes manipulated gradients, preventing tampering with the global model and ensuring secure, reliable FL.

Another paper [183] presents Privacy-preserving Federated Learning (PPFL), a defense framework designed to protect against privacy attacks in FL systems. PPFL utilizes TEEs on both server and client devices to conceal model and gradient updates, as well as trained layers, from adversaries, thereby preventing attacks aimed at reconstructing private data or inferring properties and membership from trained models or gradients. Training is conducted in a greedy layer-wise manner, with each layer independently trained within TEEs until convergence, overcoming the memory constraints of TEEs. To defend against membership inference attacks on the final model, the last layer and its outputs remain within the client's TEE after training. Secure communication channels are established between the server and client TEEs, with data transmission encrypted using keys known only within the TEEs, ensuring data confidentiality and integrity. Evaluation shows that, by maintaining the privacy of the training process within TEEs, PPFL effectively mitigates known attacks like data reconstruction and property inference.

All the above-described privacy frameworks are summarized in Table 3.

Table 3. Privacy techniques in FL.

No.	Defense Framework	Definition	References
1	Homomorphic encryption	A cryptographic technique that enables computation on encrypted data without decryption, ensuring privacy in FL.	[143,144]
2	Knowledge Distillation	A method where a smaller “student” model learns from a larger “teacher” model, enabling privacy-preserving model compression in FL.	[151–153]
3	Secure Multi-party Computation	A protocol that allows multiple parties to compute a function over their inputs while keeping those inputs private.	[157–159]
4	Split Learning	A method where the model is split between client and server, ensuring data remain local while only intermediate activations are shared.	[165–167]
5	Perturbing Gradients	A technique that adds noise to gradients before sharing, reducing the risk of exposing private data in FL.	[173–176]
6	Differential Privacy	A method that adds calibrated noise to data or gradients to limit the risk of identifying individuals in FL.	[177,180]
7	Trusted Execution Environments	Secure hardware components that process data in an isolated, tamper-proof environment, ensuring confidentiality in FL.	[182,183]

8. Discussion

Each of the defense methods discussed has its own advantages and disadvantages based on the specific scenarios they address, such as data and model poisoning, privacy attacks, or resource constraints. While some techniques offer robust security measures, they may also introduce computational or performance trade-offs that require careful consideration in Federated Learning systems. The advantages and disadvantages of these techniques are summarized in Table 4.

Table 4. Advantages and disadvantages of defense techniques in Federated Learning.

No.	Defense Technique	Advantages	Disadvantages
1	Detect and Remove	Efficient detection of malicious updates using anomaly detection; prevents poisoned updates from influencing the global model.	Relies on statistical methods, which may struggle with sophisticated attacks.
2	Adversarial Training	Enhances model robustness by training on adversarial examples; effective for defending against data and model poisoning.	Performance degradation in non-IID data distributions, especially in federated setups.
3	Model Pruning	Improves computational efficiency and reduces model size, making it harder for attackers to insert malicious updates.	Potential loss in accuracy if essential parameters are pruned.
4	Byzantine Robust Aggregation	Filters out malicious updates by identifying outliers and focusing on majority consensus.	Computationally intensive, slowing down convergence in large-scale FL systems.
5	Robust Client Selection	Enhances the integrity of the aggregation process by selecting reliable clients, reducing the impact of malicious ones.	May exclude valuable data from legitimate but infrequent clients.
6	Data and Update Analysis	Provides comprehensive defense by analyzing both client updates and local data for consistency.	Requires significant computational resources, slowing the training process.
7	Homomorphic Encryption	Enables computations on encrypted data, protecting sensitive information during aggregation.	Computationally expensive and slower; challenging for resource-constrained clients.
8	Knowledge Distillation	Reduces the computational load on edge devices while maintaining performance by transferring knowledge from larger models.	Can introduce information leakage; vulnerable to inference attacks.
9	Secure Multi-Party Computation	Enables secure collaborative computation without revealing individual data, ensuring high privacy.	Complex to implement and resource-intensive.
10	Blockchain Integration	Adds transparency and immutability to FL, preventing tampering and providing auditability.	High overhead in terms of computation and storage; not scalable for large networks.
11	Incentivized Federated Learning	Encourages active and honest participation by rewarding clients based on contributions.	Requires careful incentive design to avoid gaming.
12	Regularization	Prevents overfitting and improves generalization, especially in environments with noisy data.	Over-regularization can reduce model accuracy.
13	Split Learning	Enhances privacy by keeping sensitive data on client devices and only sharing intermediate representations.	Requires sophisticated coordination, which can introduce delays.
14	Perturbing Gradients	Protects against gradient-based attacks by adding noise to gradients; ensures differential privacy.	Too much noise can degrade model performance, requiring careful tuning.
15	Differential Privacy	Provides strong privacy guarantees by adding noise to model updates, protecting individual contributions.	Trade-off between privacy and model utility; too much noise can reduce accuracy.
16	Trusted Execution Environments (TEEs)	Ensures integrity and privacy by isolating computation inside secure hardware enclaves.	Limited memory and high overhead restrict scalability for large models.

9. Challenges

Data Privacy Regulations: Adhering to diverse and evolving data privacy regulations across different regions adds complexity to the implementation of FL. Ensuring compliance while maintaining the efficiency and effectiveness of FL systems requires continuous monitoring and adaptation to new legal requirements.

Interoperability: Ensuring seamless interaction between different FL frameworks and devices from various manufacturers is crucial. The diversity of the devices and platforms involved in FL creates a complex ecosystem where interoperability can be a significant challenge. Different devices may use varied hardware configurations, operating systems, and network protocols, which can complicate the integration process. To achieve interoperability, the development of standard protocols and interfaces is essential. These standards should facilitate seamless communication and collaboration between devices and frameworks regardless of their underlying technologies. For instance, standardized APIs (Application Programming Interfaces) can enable different FL implementations to exchange data and model updates efficiently. Additionally, the adoption of common communication protocols can ensure that devices with varying capabilities can participate in the FL process without compatibility issues.

Usability and Deployment: Enhancing the adoption of FL systems, particularly in resource-constrained environments, requires simplifying the deployment and management processes. The complexity of setting up FL frameworks often acts as a barrier, especially in environments lacking technical expertise. User-friendly interfaces and automated configuration tools are pivotal in overcoming these challenges. Interfaces like graphical user interfaces (GUIs) offer intuitive controls and step-by-step guidance, simplifying FL setup and management. Automated tools adjust the system parameters dynamically based on the available resources, optimizing efficiency and reducing manual tasks. Comprehensive documentation and support further assist in troubleshooting, ensuring smoother integration and wider adoption of FL, thereby maximizing its benefits across diverse environments.

Communication Overhead: The transmission of model updates between clients and the central server in FL leads to significant communication overhead. This issue is exacerbated in large-scale deployments where numerous devices participate, causing delays and potential bottlenecks in the network. Efficient communication protocols and compression techniques are needed to mitigate these challenges and ensure timely updates without compromising model performance.

Heterogeneity of Devices: FL involves a variety of devices with different computational capabilities, network conditions, and data distributions. This heterogeneity can affect the convergence of the global model and the overall system performance. Developing adaptive algorithms that can handle such diversity is crucial to maintain the robustness and effectiveness of FL systems.

Security of Edge Devices: Despite the inherent privacy-preserving nature of FL, several security and privacy challenges persist. Attacks such as data poisoning, model poisoning, and inference attacks can compromise the integrity and confidentiality of the system. Robust defense mechanisms, including differential privacy, homomorphic encryption, and Secure Multi-party Computation, are essential to protect against these threats. Additionally, the deployment of Trusted Execution Environments (TEEs) can enhance security by ensuring that computations are performed in a secure and isolated manner.

Real-Time Adaptation: The dynamic nature of threats in FL necessitates the development of real-time adaptive defense mechanisms. These systems should be capable of detecting and responding to new and evolving attack vectors promptly. Machine learning techniques can be employed to predict and counteract potential attacks, thereby enhancing the resilience of FL systems.

Scalability of Defense Mechanisms: As the number of participating devices increases, the scalability of the defense mechanisms becomes a critical concern. Techniques that are computationally intensive or require significant communication overhead may not be practical for large-scale FL systems. Research should focus on developing lightweight and

scalable defense strategies that can efficiently handle a growing number of clients without degrading performance.

10. Future Directions

10.1. Enhanced Privacy-Preserving Techniques

In light of the ongoing advancements and the challenges faced in the field of FL defense techniques, several future directions can be explored to enhance the robustness and efficiency of these systems.

Homomorphic Encryption: Future work should focus on making homomorphic encryption more efficient and scalable for real-time applications in FL. Homomorphic encryption enables computations on encrypted data without requiring decryption, thus preserving data privacy. However, the computational cost is currently prohibitive for large-scale applications. Optimizing this encryption method could significantly improve privacy-preserving FL systems.

Differential Privacy: Integrating differential privacy into FL remains a significant research area. There is a need to balance the trade-off between data utility and privacy. Developing adaptive mechanisms that can dynamically adjust privacy parameters based on the sensitivity of the data could lead to more effective privacy preservation without compromising model performance.

10.2. Advanced Byzantine-Resilient Aggregation

Robust Aggregation Techniques: There is a need to develop more robust aggregation methods to withstand sophisticated byzantine attacks, where malicious clients might send incorrect model updates. Research should aim at refining techniques such as robust weighted aggregation and adaptive outlier detection to be applicable in diverse FL environments, ensuring the integrity of the global model.

Dynamic Defense Mechanisms: Implementing dynamic defense mechanisms that can adapt to evolving attack patterns and client behaviors is essential. This includes utilizing machine learning techniques to predict and counteract potential byzantine attacks in real time, enhancing the resilience of FL systems.

10.3. Scalable and Efficient Secure Multi-Party Computation (SMPC)

Enhancing the scalability of SMPC protocols is crucial to support large-scale FL systems. The current SMPC implementations often face high computational and communication overheads, limiting their practicality. Future research should focus on optimizing these protocols to make them more efficient and scalable, enabling their use in extensive FL networks.

10.4. Trust and Incentive Mechanisms

Developing robust trust management systems is vital for evaluating and verifying the credibility of participating clients. Trust mechanisms can help to maintain the integrity of FL systems by ensuring that only trustworthy participants contribute to the model training. Additionally, designing incentive mechanisms that encourage honest participation while deterring malicious behavior could enhance the overall security framework of FL systems.

10.5. Federated Learning Protocol Standardization

As FL continues to grow, there is a need for standardized protocols and frameworks to ensure interoperability between different systems and devices. Developing common standards can facilitate better collaboration and integration across various platforms, enhancing the overall security and efficiency of FL implementations.

10.6. Explainable Federated Learning

Developing methods to make FL models more interpretable and explainable is another important direction. By understanding the decision-making process of these models, it becomes easier to detect and mitigate malicious behaviors. Explainability can also build

trust among participants by providing transparency in how the aggregated model is being trained and updated.

10.7. Federated Learning with Unreliable Participants

In real-world scenarios, not all the participants can be assumed to be reliable or have stable connectivity. Research should explore mechanisms for FL that can handle intermittent participation and varying levels of reliability among clients, ensuring that the global model remains robust despite these challenges.

10.8. Energy-Efficient Federated Learning

Given the diverse range of devices involved in FL, energy efficiency is a critical concern. Incorporating energy-efficient techniques into defense mechanisms can enhance their practicality and adoption. For instance, lightweight cryptographic protocols can be designed to provide robust security without imposing heavy computational loads on client devices. Additionally, optimizing secure aggregation techniques to minimize communication overhead can conserve energy while maintaining the integrity and confidentiality of model updates. Future research should focus on creating a balance between energy efficiency and robust defense strategies. By doing so, FL systems can achieve high levels of security and privacy without compromising on the sustainability and performance of the participating devices. This holistic approach ensures that FL remains an effective and viable solution for decentralized learning across diverse and resource-constrained environments.

11. Conclusions

The evolution of FL has brought forth substantial advancements in data privacy and decentralized model training. However, these benefits come with heightened security risks that necessitate robust defense mechanisms. This survey categorizes and reviews the current defense strategies against various attacks on FL systems. Pre-aggregation, in-aggregation, and post-aggregation defenses play critical roles in mitigating adversarial threats at different stages of the learning process. Furthermore, advanced cryptographic techniques like homomorphic encryption and differential privacy are essential for protecting sensitive information. The integration of blockchain technology enhances the security and transparency of FL, while incentive mechanisms ensure active and honest participation from clients. Moving forward, continuous innovation and comprehensive strategies are imperative to address the evolving threats in FL environments, ensuring the integrity and reliability of decentralized machine learning.

Author Contributions: Conceptualization, H.U.M. and A.Z.; methodology, H.U.M. and A.C.; software, H.U.M.; validation, A.S., A.C. and D.F.; formal analysis, H.U.M.; investigation, H.U.M. and A.C.; resources, A.Z.; data curation, H.U.M.; writing—original draft preparation, H.U.M.; writing—review and editing, A.S., A.C., D.F. and A.Z.; visualization, H.U.M.; supervision, D.F. and A.Z.; project administration, A.Z.; funding acquisition, A.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by Engineering and Physical Sciences Research Council [grant number: EP/Y002288/1].

Data Availability Statement: Authors did not use any data in this paper.

Acknowledgments: During the preparation of this work, the authors utilized ChatGPT to proofread the manuscript. Following the use of this tool/service, the authors thoroughly reviewed and edited the content as necessary. They hereby take full responsibility for the publication's content.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AT	Adversarial Training
P	Pruning
BRA	Byzantine/Reliable Aggregation
R	Regularization
D&R	Detect and Remove
RCS	Robust Client Selection
AUA	Data and Update Analysis
Ho	Homomorphic Encryption
KD	Knowledge Distillation
SMP	Secure Multi-Party Aggregation
TEE	Trusted Execution Environment
SL	Split Learning
PG	Perturbing Gradients
DP	Differential Privacy
BC	Blockchain
In	Incentive Federated Learning

References

- Badi, S. IoT and Big Data Analytics: Revolutionizing Business and Society with Advanced Insights. *Int. J. Appl. Math. Comput. Sci.* **2024**, *3*, 42–56.
- Almutairi, R.; Bergami, G.; Morgan, G. Advancements and Challenges in IoT Simulators: A Comprehensive Review. *Sensors* **2024**, *24*, 1511. [[CrossRef](#)] [[PubMed](#)]
- Ahmed, E.; Yaqoob, I.; Hashem, I.A.T.; Khan, I.; Ahmed, A.I.A.; Imran, M.; Vasilakos, A.V. The role of big data analytics in Internet of Things. *Comput. Netw.* **2017**, *129*, 459–471. [[CrossRef](#)]
- Basit, A.; Manzoor, H.U.; Akram, M.; Gelani, H.E.; Hussain, S. Machine learning-assisted anomaly detection for power line components: A case study in Pakistan. *J. Eng.* **2024**, *2024*, e12405. [[CrossRef](#)]
- Manzoor, H.U.; Khan, A.R.; Al-Quraan, M.; Mohjazi, L.; Taha, A.; Abbas, H.; Hussain, S.; Imran, M.A.; Zoha, A. Energy management in an agile workspace using ai-driven forecasting and anomaly detection. In Proceedings of the 2022 4th Global Power, Energy and Communication Conference (GPECOM), Cappadocia, Turkey, 14–17 June 2022; pp. 644–649.
- Allioui, H.; Mourdi, Y. Exploring the full potentials of IoT for better financial growth and stability: A comprehensive survey. *Sensors* **2023**, *23*, 8015. [[CrossRef](#)] [[PubMed](#)]
- Alzubaidi, L.; Bai, J.; Al-Sabaawi, A.; Santamaría, J.; Albahri, A.S.; Al-dabbagh, B.S.N.; Fadhel, M.A.; Manoufali, M.; Zhang, J.; Al-Timemy, A.H.; et al. A survey on deep learning tools dealing with data scarcity: Definitions, challenges, solutions, tips, and applications. *J. Big Data* **2023**, *10*, 46. [[CrossRef](#)]
- Tso, F.P.; Jouet, S.; Pezaros, D.P. Network and server resource management strategies for data centre infrastructures: A survey. *Comput. Netw.* **2016**, *106*, 209–225. [[CrossRef](#)]
- Dash, S.; Shakyawar, S.K.; Sharma, M.; Kaushik, S. Big data in healthcare: Management, analysis and future prospects. *J. Big Data* **2019**, *6*, 1–25. [[CrossRef](#)]
- Liu, B.; Ding, M.; Zhu, T.; Xiang, Y.; Zhou, W. Adversaries or allies? Privacy and deep learning in big data era. *Concurr. Comput. Pract. Exp.* **2019**, *31*, e5102. [[CrossRef](#)]
- Agrawal, S.; Chowdhuri, A.; Sarkar, S.; Selvanambi, R.; Gadekallu, T.R. Temporal weighted averaging for asynchronous federated intrusion detection systems. *Comput. Intell. Neurosci.* **2021**, *2021*, 5844728. [[CrossRef](#)]
- Hasan, J. Security and privacy issues of federated learning. *arXiv* **2023**, arXiv:2307.12181.
- Force, J.T. *Security and Privacy Controls for Information Systems and Organizations*; Technical Report; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2017.
- Kang, H.; Liu, G.; Wang, Q.; Meng, L.; Liu, J. Theory and application of zero trust security: A brief survey. *Entropy* **2023**, *25*, 1595. [[CrossRef](#)]
- Manzoor, H.U.; Hussain, S.; Flynn, D.; Zoha, A. Centralised vs. decentralised federated load forecasting in smart buildings: Who holds the key to adversarial attack robustness? *Energy Build.* **2024**, *324*, 114871. [[CrossRef](#)]
- Manzoor, S.; Mian, A.N. Robust Federated Learning-based Content Caching over Uncertain Wireless Transmission Channels in FRANs. In Proceedings of the 2021 19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt), Philadelphia, PA, USA, 18–21 October 2021; pp. 1–6.
- Liu, P.; Xu, X.; Wang, W. Threats, attacks and defenses to federated learning: Issues, taxonomy and perspectives. *Cybersecurity* **2022**, *5*, 4. [[CrossRef](#)]
- Zhang, J.; Zhu, H.; Wang, F.; Zhao, J.; Xu, Q.; Li, H. Security and privacy threats to federated learning: Issues, methods, and challenges. *Secur. Commun. Netw.* **2022**, *2022*, 2886795. [[CrossRef](#)]

19. Manzoor, H.U.; Khan, A.R.; Flynn, D.; Alam, M.M.; Akram, M.; Imran, M.A.; Zoha, A. Fedbranched: Leveraging federated learning for anomaly-aware load forecasting in energy networks. *Sensors* **2023**, *23*, 3570. [[CrossRef](#)]
20. Blanco-Justicia, A.; Domingo-Ferrer, J.; Martínez, S.; Sánchez, D.; Flanagan, A.; Tan, K.E. Achieving security and privacy in federated learning systems: Survey, research challenges and future directions. *Eng. Appl. Artif. Intell.* **2021**, *106*, 104468. [[CrossRef](#)]
21. Manzoor, S.; Mian, A.N.; Zoha, A.; Imran, M.A. Federated learning empowered mobility-aware proactive content offloading framework for fog radio access networks. *Future Gener. Comput. Syst.* **2022**, *133*, 307–319. [[CrossRef](#)]
22. Khan, A.R.; Manzoor, H.U.; Ayaz, F.; Imran, M.A.; Zoha, A. A privacy and energy-aware federated framework for human activity recognition. *Sensors* **2023**, *23*, 9339. [[CrossRef](#)]
23. Kumar, K.N.; Mohan, C.K.; Cenkeramaddi, L.R. The Impact of Adversarial Attacks on Federated Learning: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *46*, 2672–2691. [[CrossRef](#)]
24. Sikandar, H.S.; Waheed, H.; Tahir, S.; Malik, S.U.; Rafique, W. A detailed survey on federated learning attacks and defenses. *Electronics* **2023**, *12*, 260. [[CrossRef](#)]
25. Hallaji, E.; Razavi-Far, R.; Saif, M. Federated and transfer learning: A survey on adversaries and defense mechanisms. In *Federated and Transfer Learning*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 29–55.
26. Nair, A.K.; Raj, E.D.; Sahoo, J. A robust analysis of adversarial attacks on federated learning environments. *Comput. Stand. Interfaces* **2023**, *86*, 103723. [[CrossRef](#)]
27. Rodríguez-Barroso, N.; Jiménez-López, D.; Luzón, M.V.; Herrera, F.; Martínez-Cámara, E. Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges. *Inf. Fusion* **2023**, *90*, 148–173. [[CrossRef](#)]
28. Yang, M.; He, Y.; Qiao, J. Federated learning-based privacy-preserving and security: Survey. In Proceedings of the 2021 Computing, Communications and IoT Applications (ComComAp), Shenzhen, China, 26–28 November 2021; pp. 312–317.
29. Bouacida, N.; Mohapatra, P. Vulnerabilities in federated learning. *IEEE Access* **2021**, *9*, 63229–63249. [[CrossRef](#)]
30. Mothukuri, V.; Parizi, R.M.; Pouriyeh, S.; Huang, Y.; Dehghantanha, A.; Srivastava, G. A survey on security and privacy of federated learning. *Future Gener. Comput. Syst.* **2021**, *115*, 619–640. [[CrossRef](#)]
31. Xia, G.; Chen, J.; Yu, C.; Ma, J. Poisoning attacks in federated learning: A survey. *IEEE Access* **2023**, *11*, 10708–10722. [[CrossRef](#)]
32. Benmalek, M.; Benrekia, M.A.; Challal, Y. Security of federated learning: Attacks, defensive mechanisms, and challenges. *Revue Sciences Technologies l'Information-Série RIA Revue d'Intelligence Artificielle* **2022**, *36*, 49–59. [[CrossRef](#)]
33. Manzoor, H.U.; Jafri, A.; Zoha, A. Adaptive single-layer aggregation framework for energy-efficient and privacy-preserving load forecasting in heterogeneous Federated smart grids. *Internet Things* **2024**, *28*, 101376. [[CrossRef](#)]
34. Shabbir, A.; Manzoor, H.U.; Arshad, K.; Assaleh, K.; Halim, Z.; Zoha, A. Sustainable and Lightweight Defense Framework for Resource Constraint Federated Learning Assisted Smart Grids Against Adversarial Attacks. *Authorea Prepr.* **2024**. [[CrossRef](#)]
35. Manzoor, H.U.; Jafri, A.; Zoha, A. Lightweight Single-Layer Aggregation Framework for Energy-Efficient and Privacy-Preserving Load Forecasting in Heterogeneous Smart Grids. *Authorea Prepr.* **2024**. [[CrossRef](#)]
36. Qi, P.; Chiaro, D.; Guzzo, A.; Ianni, M.; Fortino, G.; Piccialli, F. Model aggregation techniques in federated learning: A comprehensive survey. *Future Gener. Comput. Syst.* **2023**, *150*, 272–293. [[CrossRef](#)]
37. Liu, Y.; Huang, A.; Luo, Y.; Huang, H.; Liu, Y.; Chen, Y.; Feng, L.; Chen, T.; Yu, H.; Yang, Q. Fedvision: An online visual object detection platform powered by federated learning. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 13172–13179.
38. Khan, A.R.; Manzoor, H.U.; Rais, R.N.B.; Hussain, S.; Mohjazi, L.; Imran, M.A.; Zoha, A. Semantic-Aware Federated Blockage Prediction (SFBP) in Vision-Aided Next-Generation Wireless Network. *Authorea Prepr.* **2024**. [[CrossRef](#)]
39. Brecko, A.; Kajati, E.; Koziolek, J.; Zolotova, I. Federated learning for edge computing: A survey. *Appl. Sci.* **2022**, *12*, 9124. [[CrossRef](#)]
40. Gao, D.; Ju, C.; Wei, X.; Liu, Y.; Chen, T.; Yang, Q. Hhhfl: Hierarchical heterogeneous horizontal federated learning for electroencephalography. *arXiv* **2019**, arXiv:1909.05784.
41. Liu, Y.; Kang, Y.; Zou, T.; Pu, Y.; He, Y.; Ye, X.; Ouyang, Y.; Zhang, Y.Q.; Yang, Q. Vertical federated learning: Concepts, advances, and challenges. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 3615–3634. [[CrossRef](#)]
42. Bharati, S.; Mondal, M.; Podder, P.; Prasath, V. Federated learning: Applications, challenges and future directions. *Int. J. Hybrid Intell. Syst.* **2022**, *18*, 19–35. [[CrossRef](#)]
43. Wu, Y.; Cai, S.; Xiao, X.; Chen, G.; Ooi, B.C. Privacy preserving vertical federated learning for tree-based models. *arXiv* **2020**, arXiv:2008.06170. [[CrossRef](#)]
44. Saha, S.; Ahmad, T. Federated transfer learning: Concept and applications. *Intell. Artif.* **2021**, *15*, 35–44. [[CrossRef](#)]
45. Dai, S.; Meng, F. Addressing modern and practical challenges in machine learning: A survey of online federated and transfer learning. *Appl. Intell.* **2023**, *53*, 11045–11072. [[CrossRef](#)]
46. Manzoor, H.U.; Hussain, S.; Flynn, D.; Zoha, A. Centralised vs. Decentralised Federated Load Forecasting: Who Holds the Key to Adversarial Attack Robustness? *Authorea Prepr.* **2024**. [[CrossRef](#)]
47. Yuan, L.; Wang, Z.; Sun, L.; Philip, S.Y.; Brinton, C.G. Decentralized federated learning: A survey and perspective. *IEEE Internet Things J.* **2024**. [[CrossRef](#)]
48. Gu, X.; Sabrina, F.; Fan, Z.; Sohail, S. A review of privacy enhancement methods for federated learning in healthcare systems. *Int. J. Environ. Res. Public Health* **2023**, *20*, 6539. [[CrossRef](#)] [[PubMed](#)]

49. Issa, W.; Moustafa, N.; Turnbull, B.; Sohrabi, N.; Tari, Z. Blockchain-based federated learning for securing internet of things: A comprehensive survey. *ACM Comput. Surv.* **2023**, *55*, 1–43. [[CrossRef](#)]
50. Gabrielli, E.; Pica, G.; Tolomei, G. A survey on decentralized federated learning. *arXiv* **2023**, arXiv:2308.04604.
51. Ye, H.; Liang, L.; Li, G.Y. Decentralized federated learning with unreliable communications. *IEEE J. Sel. Top. Signal Process.* **2022**, *16*, 487–500. [[CrossRef](#)]
52. Huang, C.; Huang, J.; Liu, X. Cross-silo federated learning: Challenges and opportunities. *arXiv* **2022**, arXiv:2206.12949.
53. Müller, K.; Bodendorf, F. Cross-silo federated learning in enterprise networks with cooperative and competing actors. *Hum. Side Serv. Eng.* **2023**, *108*, 244–253.
54. Huang, C.; Tang, M.; Ma, Q.; Huang, J.; Liu, X. Promoting collaborations in cross-silo federated learning: Challenges and opportunities. *IEEE Commun. Mag.* **2023**, *62*, 82–88. [[CrossRef](#)]
55. Liu, K.; Hu, S.; Wu, S.Z.; Smith, V. On privacy and personalization in cross-silo federated learning. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 5925–5940.
56. Wang, T.; Du, Y.; Gong, Y.; Choo, K.K.R.; Guo, Y. Applications of federated learning in mobile health: Scoping review. *J. Med. Internet Res.* **2023**, *25*, e43006. [[CrossRef](#)]
57. Zhao, L.; Li, J.; Li, Q.; Li, F. A federated learning framework for detecting false data injection attacks in solar farms. *IEEE Trans. Power Electron.* **2021**, *37*, 2496–2501. [[CrossRef](#)]
58. Shabbir, A.; Manzoor, H.U.; Ahmed, R.A.; Halim, Z. Resilience of federated learning against false data injection attacks in energy forecasting. In Proceedings of the 2024 International Conference on Green Energy, Computing and Sustainable Technology (GECOST), Miri Sarawak, Malaysia, 17–19 January 2024; pp. 245–249.
59. Lv, Z.; Cao, H.; Zhang, F.; Ren, Y.; Wang, B.; Chen, C.; Li, N.; Chang, H.; Wang, W. Awfc: Preventing label flipping attacks towards federated learning for intelligent iot. *Comput. J.* **2022**, *65*, 2849–2859. [[CrossRef](#)]
60. Jebreel, N.M.; Domingo-Ferrer, J.; Sánchez, D.; Blanco-Justicia, A. LFighter: Defending against the label-flipping attack in federated learning. *Neural Netw.* **2024**, *170*, 111–126. [[CrossRef](#)] [[PubMed](#)]
61. Andreina, S.; Marson, G.A.; Möllering, H.; Karame, G. Baffle: Backdoor detection via feedback-based federated learning. In Proceedings of the 2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS), Washington, DC, USA, 7–10 July 2021; pp. 852–863.
62. Manzoor, H.U.; Arshad, K.; Assaleh, K.; Zoha, A. Enhanced Adversarial Attack Resilience in Energy Networks through Energy and Privacy Aware Federated Learning. *Authorea Prepr.* **2024**. [[CrossRef](#)]
63. Zhou, X.; Xu, M.; Wu, Y.; Zheng, N. Deep model poisoning attack on federated learning. *Future Internet* **2021**, *13*, 73. [[CrossRef](#)]
64. Manzoor, H.U.; Khan, A.R.; Sher, T.; Ahmad, W.; Zoha, A. Defending federated learning from backdoor attacks: Anomaly-aware fedavg with layer-based aggregation. In Proceedings of the 2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Toronto, ON, Canada, 5–8 September 2023; pp. 1–6.
65. Mohammadi, S.; Balador, A.; Sinaei, S.; Flammini, F. Balancing privacy and performance in federated learning: A systematic literature review on methods and metrics. *J. Parallel Distrib. Comput.* **2024**, *192*, 104918. [[CrossRef](#)]
66. Yuan, X.; Ma, X.; Zhang, L.; Fang, Y.; Wu, D. Beyond class-level privacy leakage: Breaking record-level privacy in federated learning. *IEEE Internet Things J.* **2021**, *9*, 2555–2565. [[CrossRef](#)]
67. Yang, H.; Ge, M.; Xue, D.; Xiang, K.; Li, H.; Lu, R. Gradient leakage attacks in federated learning: Research frontiers, taxonomy and future directions. *IEEE Netw.* **2023**, *38*, 247–254. [[CrossRef](#)]
68. Hallaji, E.; Razavi-Far, R.; Saif, M.; Wang, B.; Yang, Q. Decentralized federated learning: A survey on security and privacy. *IEEE Trans. Big Data* **2024**, *10*, 194–213. [[CrossRef](#)]
69. Jebreel, N.M.; Domingo-Ferrer, J. FI-defender: Combating targeted attacks in federated learning. *Knowl.-Based Syst.* **2023**, *260*, 110178. [[CrossRef](#)]
70. Wainakh, A.; Zimmer, E.; Subedi, S.; Keim, J.; Grube, T.; Karuppayah, S.; Sanchez Guinea, A.; Mühlhäuser, M. Federated learning attacks revisited: A critical discussion of gaps, assumptions, and evaluation setups. *Sensors* **2022**, *23*, 31. [[CrossRef](#)] [[PubMed](#)]
71. Bao, G.; Guo, P. Federated learning in cloud-edge collaborative architecture: Key technologies, applications and challenges. *J. Cloud Comput.* **2022**, *11*, 94. [[CrossRef](#)] [[PubMed](#)]
72. Fung, C.; Yoon, C.J.; Beschastnikh, I. Mitigating sybils in federated learning poisoning. *arXiv* **2018**, arXiv:1808.04866.
73. Shen, S.; Tople, S.; Saxena, P. Auror: Defending against poisoning attacks in collaborative deep learning systems. In Proceedings of the 32nd Annual Conference on Computer Security Applications, Los Angeles, CA, USA, 5–9 December 2016; pp. 508–519.
74. Sun, Z.; Kairouz, P.; Suresh, A.T.; McMahan, H.B. Can you really backdoor federated learning? *arXiv* **2019**, arXiv:1911.07963.
75. Nguyen, T.D.; Rieger, P.; De Viti, R.; Chen, H.; Brandenburg, B.B.; Yalame, H.; Möllering, H.; Fereidooni, H.; Marchal, S.; Miettinen, M.; et al. {FLAME}: Taming backdoors in federated learning. In Proceedings of the 31st USENIX Security Symposium (USENIX Security 22), Boston, MA, USA, 10–12 August 2022; pp. 1415–1432.
76. Pillutla, K.; Kakade, S.M.; Harchaoui, Z. Robust aggregation for federated learning. *IEEE Trans. Signal Process.* **2022**, *70*, 1142–1154. [[CrossRef](#)]
77. Blanchard, P.; El Mhamdi, E.M.; Guerraoui, R.; Stainer, J. Machine learning with adversaries: Byzantine tolerant gradient descent. *Adv. Neural Inf. Process. Syst.* **2017**, *30*. Available online: https://papers.nips.cc/paper_files/paper/2017/hash/f4b9ec30ad9f68f89b29639786cb62ef-Abstract.html (accessed on 1 September 2024).

78. Ozdayi, M.S.; Kantarcioglu, M.; Gel, Y.R. Defending against backdoors in federated learning with robust learning rate. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 9268–9276.
79. Manzoor, H.U.; Khan, M.S.; Khan, A.R.; Ayaz, F.; Flynn, D.; Imran, M.A.; Zoha, A. FedClamp: An Algorithm for Identification of Anomalous Client in Federated Learning. In Proceedings of the 2022 29th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Glasgow, UK, 24–26 October 2022; pp. 1–4.
80. Wu, C.; Zhu, S.; Mitra, P. Federated unlearning with knowledge distillation. *arXiv* **2022**, arXiv:2201.09441.
81. Wu, C.; Yang, X.; Zhu, S.; Mitra, P. Mitigating backdoor attacks in federated learning. *arXiv* **2020**, arXiv:2011.01767.
82. Li, X.; Qu, Z.; Zhao, S.; Tang, B.; Lu, Z.; Liu, Y. Lomar: A local defense against poisoning attack on federated learning. *IEEE Trans. Dependable Secur. Comput.* **2021**, *20*, 437–450. [[CrossRef](#)]
83. Zhao, C.; Wen, Y.; Li, S.; Liu, F.; Meng, D. Federatedreverse: A detection and defense method against backdoor attacks in federated learning. In Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security, Online, 22–25 June 2021; pp. 51–62.
84. Rodríguez-Barroso, N.; Martínez-Cámara, E.; Luzón, M.V.; Herrera, F. Dynamic defense against byzantine poisoning attacks in federated learning. *Future Gener. Comput. Syst.* **2022**, *133*, 1–9. [[CrossRef](#)]
85. Zhang, Z.; Zhang, Y.; Guo, D.; Yao, L.; Li, Z. Secfednids: Robust defense for poisoning attack against federated learning-based network intrusion detection system. *Future Gener. Comput. Syst.* **2022**, *134*, 154–169. [[CrossRef](#)]
86. Lu, S.; Li, R.; Liu, W.; Chen, X. Defense against backdoor attack in federated learning. *Comput. Secur.* **2022**, *121*, 102819. [[CrossRef](#)]
87. Wan, W.; Lu, J.; Hu, S.; Zhang, L.Y.; Pei, X. Shielding federated learning: A new attack approach and its defense. In Proceedings of the 2021 IEEE Wireless Communications and Networking Conference (WCNC), Nanjing, China, 29 March–1 April 2021; pp. 1–7.
88. Li, D.; Wong, W.E.; Wang, W.; Yao, Y.; Chau, M. Detection and mitigation of label-flipping attacks in federated learning systems with KPCA and K-means. In Proceedings of the 2021 8th International Conference on Dependable Systems and Their Applications (DSA), Yinchuan, China, 5–6 August 2021; pp. 551–559.
89. Lee, K.; Lee, K.; Lee, H.; Shin, J. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Adv. Neural Inf. Process. Syst.* **2018**, *31*. Available online: https://papers.nips.cc/paper_files/paper/2018/hash/abdeb6f575ac5c6676b747bca8d09cc2-Abstract.html (accessed on 1 September 2024).
90. Zizzo, G.; Rawat, A.; Sinn, M.; Buesser, B. Fat: Federated adversarial training. *arXiv* **2020**, arXiv:2012.01791.
91. Shah, D.; Dube, P.; Chakraborty, S.; Verma, A. Adversarial training in communication constrained federated learning. *arXiv* **2021**, arXiv:2103.01319.
92. Shoham, N.; Avidor, T.; Keren, A.; Israel, N.; Benditkis, D.; Mor-Yosef, L.; Zeitak, I. Overcoming forgetting in federated learning on non-iid data. *arXiv* **2019**, arXiv:1910.07796.
93. Hallaji, E.; Razavi-Far, R.; Saif, M.; Herrera-Viedma, E. Label noise analysis meets adversarial training: A defense against label poisoning in federated learning. *Knowl.-Based Syst.* **2023**, *266*, 110384. [[CrossRef](#)]
94. Reed, R. Pruning algorithms—A survey. *IEEE Trans. Neural Netw.* **1993**, *4*, 740–747. [[CrossRef](#)]
95. Geng, X.; Gao, J.; Zhang, Y.; Xu, D. Complex hybrid weighted pruning method for accelerating convolutional neural networks. *Sci. Rep.* **2024**, *14*, 5570. [[CrossRef](#)]
96. Anwar, S.; Hwang, K.; Sung, W. Structured pruning of deep convolutional neural networks. *ACM J. Emerg. Technol. Comput. Syst. (JETC)* **2017**, *13*, 1–18. [[CrossRef](#)]
97. Liao, Z.; Quéru, V.; Nguyen, V.T.; Tartaglione, E. Can Unstructured Pruning Reduce the Depth in Deep Neural Networks? In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 2–6 October 2023; pp. 1402–1406.
98. Jiang, Y.; Wang, S.; Valls, V.; Ko, B.J.; Lee, W.H.; Leung, K.K.; Tassioulas, L. Model pruning enables efficient federated learning on edge devices. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *34*, 10374–10386. [[CrossRef](#)] [[PubMed](#)]
99. Liu, S.; Yu, G.; Yin, R.; Yuan, J. Adaptive network pruning for wireless federated learning. *IEEE Wirel. Commun. Lett.* **2021**, *10*, 1572–1576. [[CrossRef](#)]
100. Zhangheng, L.; Chen, T.; Li, L.; Li, B.; Wang, Z. Can Pruning Improve Certified Robustness of Neural Networks? *Trans. Mach. Learn. Res.* **2022**. Available online: <https://www.semanticscholar.org/paper/Can-pruning-improve-certified-robustness-of-neural-Li-Chen/6f0b89a3ce7c835dc42afe798b9424471f4ca585> (accessed on 1 September 2024).
101. Zhang, C.; Yang, S.; Mao, L.; Ning, H. Anomaly detection and defense techniques in federated learning: A comprehensive review. *Artif. Intell. Rev.* **2024**, *57*, 1–34. [[CrossRef](#)]
102. Meng, M.H.; Teo, S.G.; Bai, G.; Wang, K.; Dong, J.S. Enhancing Federated Learning Robustness Using Data-Agnostic Model Pruning. In Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining, Osaka, Japan, 25–28 May 2023; Springer: Cham, Switzerland, 2023; pp. 441–453.
103. Jiang, X.; Borcea, C. Complement sparsification: Low-overhead model pruning for federated learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 8087–8095.
104. Yin, D.; Chen, Y.; Ramchandran, K.; Bartlett, P.L. Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
105. El Mhamdi, E.M.; Guerraoui, R.; Rouault, S. The Hidden Vulnerability of Distributed Learning in Byzantium. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Dy, J., Krause, A., Eds.; Proceedings of Machine Learning Research; PMLR: New York, NY, USA, 2018; Volume 80, pp. 3521–3530.

106. Chen, Z.; Tian, P.; Liao, W.; Yu, W. Zero knowledge clustering based adversarial mitigation in heterogeneous federated learning. *IEEE Trans. Netw. Sci. Eng.* **2020**, *8*, 1070–1083. [[CrossRef](#)]
107. Ma, Z.; Ma, J.; Miao, Y.; Li, Y.; Deng, R.H. ShieldFL: Mitigating model poisoning attacks in privacy-preserving federated learning. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 1639–1654. [[CrossRef](#)]
108. Muñoz-González, L.; Co, K.T.; Lupu, E.C. Byzantine-robust federated machine learning through adaptive model averaging. *arXiv* **2019**, arXiv:1909.05125.
109. Cao, X.; Fang, M.; Liu, J.; Gong, N.Z. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *arXiv* **2020**, arXiv:2012.13995.
110. Cao, X.; Jia, J.; Gong, N.Z. Provably secure federated learning against malicious clients. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 6885–6893.
111. Balakrishnan, R.; Li, T.; Zhou, T.; Himayat, N.; Smith, V.; Bilmes, J. Diverse client selection for federated learning via submodular maximization. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022.
112. Kumar, A.; Khimani, V.; Chatzopoulos, D.; Hui, P. Fedclean: A defense mechanism against parameter poisoning attacks in federated learning. In Proceedings of the ICASSP 2022—2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 23–27 May 2022; pp. 4333–4337.
113. Guo, H.; Wang, H.; Song, T.; Hua, Y.; Lv, Z.; Jin, X.; Xue, Z.; Ma, R.; Guan, H. Siren: Byzantine-robust federated learning via proactive alarming. In Proceedings of the ACM Symposium on Cloud Computing, Seattle, WA, USA, 1–4 November 2021; pp. 47–60.
114. Rieger, P.; Nguyen, T.D.; Miettinen, M.; Sadeghi, A.R. Deepsight: Mitigating backdoor attacks in federated learning through deep model inspection. *arXiv* **2022**, arXiv:2201.00763.
115. Shayan, M.; Fung, C.; Yoon, C.J.; Beschastnikh, I. Biscotti: A blockchain system for private and secure federated learning. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *32*, 1513–1525. [[CrossRef](#)]
116. Jiang, Y.; Ma, B.; Wang, X.; Yu, G.; Yu, P.; Wang, Z.; Ni, W.; Liu, R.P. Blockchain Federated Learning for Internet of Things: A Comprehensive Survey. *ACM Comput. Surv.* **2023**, *56*, 258. [[CrossRef](#)]
117. Liu, K.; Yan, Z.; Liang, X.; Kantola, R.; Hu, C. A survey on blockchain-enabled federated learning and its prospects with digital twin. *Digit. Commun. Netw.* **2022**, *10*, 248–264. [[CrossRef](#)]
118. Cai, Z.; Chen, J.; Fan, Y.; Zheng, Z.; Li, K. Blockchain-empowered Federated Learning: Benefits, Challenges, and Solutions. *arXiv* **2024**, arXiv:2403.00873.
119. Saleh, A.M.S. Blockchain for secure and decentralized artificial intelligence in cybersecurity: A comprehensive review. *Blockchain Res. Appl.* **2024**, *5*, 100193. [[CrossRef](#)]
120. Alsamhi, S.H.; Myrashova, R.; Hawbani, A.; Kumar, S.; Srivastava, S.; Zhao, L.; Wei, X.; Guizan, M.; Curry, E. Federated learning meets blockchain in decentralized data-sharing: Healthcare use case. *IEEE Internet Things J.* **2024**, *11*, 19602–19615. [[CrossRef](#)]
121. Ali, A.; Al-Rimy, B.A.S.; Tin, T.T.; Altamimi, S.N.; Qasem, S.N.; Saeed, F. Empowering precision medicine: Unlocking revolutionary insights through blockchain-enabled federated learning and electronic medical records. *Sensors* **2023**, *23*, 7476. [[CrossRef](#)]
122. Mao, Q.; Wang, L.; Long, Y.; Han, L.; Wang, Z.; Chen, K. A blockchain-based framework for federated learning with privacy preservation in power load forecasting. *Knowl.-Based Syst.* **2024**, *284*, 111338. [[CrossRef](#)]
123. Batool, Z.; Zhang, K.; Zhu, Z.; Aravamuthan, S.; Aivodji, U. Block-FeST: A blockchain-based federated anomaly detection framework with computation offloading using transformers. In Proceedings of the 2022 IEEE 1st Global Emerging Technology Blockchain Forum: Blockchain & Beyond (iGETBlockchain), Irvine, CA, USA, 7–11 November 2022; pp. 1–6.
124. Zhang, W.; Lu, Q.; Yu, Q.; Li, Z.; Liu, Y.; Lo, S.K.; Chen, S.; Xu, X.; Zhu, L. Blockchain-based federated learning for device failure detection in industrial IoT. *IEEE Internet Things J.* **2020**, *8*, 5926–5937. [[CrossRef](#)]
125. Sarhan, M.; Lo, W.W.; Layeghy, S.; Portmann, M. HBFL: A hierarchical blockchain-based federated learning framework for collaborative IoT intrusion detection. *Comput. Electr. Eng.* **2022**, *103*, 108379. [[CrossRef](#)]
126. Zhang, T.; Gao, L.; He, C.; Zhang, M.; Krishnamachari, B.; Avestimehr, A.S. Federated learning for the internet of things: Applications, challenges, and opportunities. *IEEE Internet Things Mag.* **2022**, *5*, 24–29. [[CrossRef](#)]
127. Hassija, V.; Chawla, V.; Chamola, V.; Sikdar, B. Incentivization and aggregation schemes for federated learning applications. *IEEE Trans. Mach. Learn. Commun. Netw.* **2023**, *1*, 185–196. [[CrossRef](#)]
128. Rafi, T.H.; Noor, F.A.; Hussain, T.; Chae, D.K. Fairness and privacy preserving in federated learning: A survey. *Inf. Fusion* **2024**, *105*, 102198. [[CrossRef](#)]
129. Khan, A.F.; Wang, X.; Le, Q.; Khan, A.A.; Ali, H.; Ding, J.; Butt, A.; Anwar, A. Pi-fl: Personalized and incentivized federated learning. *arXiv* **2023**, arXiv:2304.07514.
130. Khajehali, N.; Yan, J.; Chow, Y.W.; Fahmideh, M. A Comprehensive Overview of IoT-Based Federated Learning: Focusing on Client Selection Methods. *Sensors* **2023**, *23*, 7235. [[CrossRef](#)]
131. Guo, H.; Mao, Y.; He, X.; Zhang, B.; Pang, T.; Ping, P. Improving Federated Learning through Abnormal Client Detection and Incentive. *CMES-Comput. Model. Eng. Sci.* **2024**, *139*. Available online: <https://www.sciencedirect.com/org/science/article/pii/S1526149223001261> (accessed on 1 September 2024). [[CrossRef](#)]
132. Bai, Y.; Xing, G.; Wu, H.; Rao, Z.; Peng, C.; Rao, Y.; Yang, W.; Ma, C.; Li, J.; Zhou, Y. ISPPFL: An incentive scheme based privacy-preserving federated learning for avatar in metaverse. *Comput. Netw.* **2024**, *251*, 110654. [[CrossRef](#)]

133. Shi, Y.; Zhang, Y.; Zhang, P.; Xiao, Y.; Niu, L. Federated learning with L1 regularization. *Pattern Recognit. Lett.* **2023**, *172*, 15–21. [[CrossRef](#)]
134. Tun, Y.L.; Thwal, C.M.; Park, Y.M.; Park, S.B.; Hong, C.S. Federated learning with intermediate representation regularization. In Proceedings of the 2023 IEEE International Conference on Big Data and Smart Computing (BigComp), Jeju, Republic of Korea, 13–16 February 2023; pp. 56–63.
135. Kim, J.; Kim, G.; Han, B. Multi-level branched regularization for federated learning. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MD, USA, 17–23 July 2022; pp. 11058–11073.
136. Acar, D.A.E.; Zhao, Y.; Navarro, R.M.; Mattina, M.; Whatmough, P.N.; Saligrama, V. Federated learning based on dynamic regularization. *arXiv* **2021**, arXiv:2111.04263.
137. Jiang, X.; Sun, S.; Wang, Y.; Liu, M. Towards federated learning against noisy labels via local self-regularization. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, 17–21 October 2022; pp. 862–873.
138. Chen, Z.; Wu, Z.; Wu, X.; Zhang, L.; Zhao, J.; Yan, Y.; Zheng, Y. Contractible regularization for federated learning on non-iid data. In Proceedings of the 2022 IEEE International Conference on Data Mining (ICDM), Orlando, FL, USA, 28 November–1 December 2022; pp. 61–70.
139. Aziz, R.; Banerjee, S.; Bouzeffrane, S.; Le Vinh, T. Exploring homomorphic encryption and differential privacy techniques towards secure federated learning paradigm. *Future Internet* **2023**, *15*, 310. [[CrossRef](#)]
140. Park, J.; Lim, H. Privacy-preserving federated learning using homomorphic encryption. *Appl. Sci.* **2022**, *12*, 734. [[CrossRef](#)]
141. Yan, G.; Lyu, S.; Hou, H.; Zheng, Z.; Song, L. Towards Quantum-Safe Federated Learning via Homomorphic Encryption: Learning with Gradients. *arXiv* **2024**, arXiv:2402.01154.
142. Munjal, K.; Bhatia, R. A systematic review of homomorphic encryption and its contributions in healthcare industry. *Complex Intell. Syst.* **2023**, *9*, 3759–3786. [[CrossRef](#)] [[PubMed](#)]
143. Liu, Y.; Zou, T.; Kang, Y.; Liu, W.; He, Y.; Yi, Z.; Yang, Q. Batch label inference and replacement attacks in black-boxed vertical federated learning. *arXiv* **2021**, arXiv:2112.05409.
144. Liu, X.; Li, H.; Xu, G.; Chen, Z.; Huang, X.; Lu, R. Privacy-enhanced federated learning against poisoning adversaries. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4574–4588. [[CrossRef](#)]
145. Sengupta, A.; Dixit, S.; Akhtar, M.S.; Chakraborty, T. A Good Learner can Teach Better: Teacher-Student Collaborative Knowledge Distillation. In Proceedings of the The Twelfth International Conference on Learning Representations, Virtual Event, 25–29 April 2023.
146. Lan, W.; Cheung, Y.m.; Xu, Q.; Liu, B.; Hu, Z.; Li, M.; Chen, Z. Improve Knowledge Distillation via Label Revision and Data Selection. *arXiv* **2024**, arXiv:2404.03693.
147. Liu, L.; Zhang, J.; Song, S.; Letaief, K.B. Communication-efficient federated distillation with active data sampling. In Proceedings of the ICC 2022-IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 201–206.
148. Sun, G.; Shu, H.; Shao, F.; Racharak, T.; Kong, W.; Pan, Y.; Dong, J.; Wang, S.; Nguyen, L.M.; Xin, J. FKD-Med: Privacy-Aware, Communication-Optimized Medical Image Segmentation via Federated Learning and Model Lightweighting through Knowledge Distillation. *IEEE Access* **2024**, *12*, 33687–33704. [[CrossRef](#)]
149. Gad, G.; Fadlullah, Z. Federated learning via augmented knowledge distillation for heterogenous deep human activity recognition systems. *Sensors* **2022**, *23*, 6. [[CrossRef](#)]
150. Li, H.; Ge, L.; Tian, L. Survey: Federated learning data security and privacy-preserving in edge-Internet of Things. *Artif. Intell. Rev.* **2024**, *57*, 130. [[CrossRef](#)]
151. Wu, C.; Wu, F.; Lyu, L.; Huang, Y.; Xie, X. Communication-efficient federated learning via knowledge distillation. *Nat. Commun.* **2022**, *13*, 2032. [[CrossRef](#)]
152. Zhu, Z.; Hong, J.; Zhou, J. Data-free knowledge distillation for heterogeneous federated learning. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 18–24 July 2021; pp. 12878–12889.
153. Zhang, L.; Shen, L.; Ding, L.; Tao, D.; Duan, L.Y. Fine-tuning global model via data-free knowledge distillation for non-iid federated learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 10174–10183.
154. Zhao, C.; Zhao, S.; Zhao, M.; Chen, Z.; Gao, C.Z.; Li, H.; Tan, Y.A. Secure multi-party computation: Theory, practice and applications. *Inf. Sci.* **2019**, *476*, 357–372. [[CrossRef](#)]
155. Kaaniche, N.; Laurent, M.; Belguith, S. Privacy enhancing technologies for solving the privacy-personalization paradox: Taxonomy and survey. *J. Netw. Comput. Appl.* **2020**, *171*, 102807. [[CrossRef](#)]
156. Alghamdi, W.; Salama, R.; Sirija, M.; Abbas, A.R.; Dilnoza, K. Secure Multi-Party Computation for Collaborative Data Analysis. In *Proceedings of the E3S Web of Conferences*; EDP Sciences: Les Ulis, France, 2023; Volume 399, p. 04034.
157. Zhang, C.; Ekanut, S.; Zhen, L.; Li, Z. Augmented multi-party computation against gradient leakage in federated learning. *IEEE Trans. Big Data* **2022**. [[CrossRef](#)]
158. Mugunthan, V.; Polychroniadou, A.; Byrd, D.; Balch, T.H. Smpai: Secure multi-party computation for federated learning. In Proceedings of the NeurIPS 2019 Workshop on Robust AI in Financial Services, Vancouver, BC, Canada, 13 December 2019; MIT Press: Cambridge, MA, USA, 2019; Volume 21.
159. Byrd, D.; Polychroniadou, A. Differentially private secure multi-party computation for federated learning in financial applications. In Proceedings of the First ACM International Conference on AI in Finance, New York, NY, USA, 15–16 October 2020; pp. 1–9.

160. Pham, N.D.; Phan, T.K.; Abuadbba, A.; Gao, Y.; Nguyen, D.; Chilamkurti, N. Split learning without local weight sharing to enhance client-side data privacy. *arXiv* **2022**, arXiv:2212.00250.
161. Xu, X.; Yang, M.; Yi, W.; Li, Z.; Wang, J.; Hu, H.; Zhuang, Y.; Liu, Y. A Stealthy Wrongdoer: Feature-Oriented Reconstruction Attack against Split Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 12130–12139.
162. Duan, Q.; Hu, S.; Deng, R.; Lu, Z. Combined federated and split learning in edge computing for ubiquitous intelligence in internet of things: State-of-the-art and future directions. *Sensors* **2022**, *22*, 5983. [[CrossRef](#)] [[PubMed](#)]
163. Erdoğan, E.; Küpçü, A.; Çiçek, A.E. Unsplit: Data-oblivious model inversion, model stealing, and label inference attacks against split learning. In Proceedings of the 21st Workshop on Privacy in the Electronic Society, Los Angeles, CA, USA, 7 November 2022; pp. 115–124.
164. Fan, M.; Chen, C.; Wang, C.; Zhou, W.; Huang, J. On the Robustness of Split Learning against Adversarial Attacks. *arXiv* **2023**, arXiv:2307.07916.
165. Otoum, S.; Guizani, N.; Mouftah, H. On the feasibility of split learning, transfer learning and federated learning for preserving security in ITS systems. *IEEE Trans. Intell. Transp. Syst.* **2022**, *24*, 7462–7470. [[CrossRef](#)]
166. Turina, V.; Zhang, Z.; Esposito, F.; Matta, I. Federated or split? A performance and privacy analysis of hybrid split and federated learning architectures. In Proceedings of the 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), Chicago, IL, USA, 5–10 September 2021; pp. 250–260.
167. Thapa, C.; Arachchige, P.C.M.; Camtepe, S.; Sun, L. Splitfed: When federated learning meets split learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Online, 22 February–1 March 2022; Volume 36, pp. 8485–8493.
168. Yaacoub, J.P.A.; Noura, H.N.; Salman, O. Security of federated learning with IoT systems: Issues, limitations, challenges, and solutions. *Internet Things Cyber-Phys. Syst.* **2023**, *3*, 155–179. [[CrossRef](#)]
169. Kim, H.G.; Shin, J.; Choi, Y.H. Human-Unrecognizable Differential Private Noised Image Generation Method. *Sensors* **2024**, *24*, 3166. [[CrossRef](#)]
170. Wang, S.; Zhu, T.; Liu, B.; Ming, D.; Guo, X.; Ye, D.; Zhou, W. Unique Security and Privacy Threats of Large Language Model: A Comprehensive Survey. *arXiv* **2024**, arXiv:2406.07973.
171. Radanliev, P.; Santos, O. Adversarial Attacks Can Deceive AI Systems, Leading to Misclassification or Incorrect Decisions. *Preprints* **2023**, 2023092064. [[CrossRef](#)]
172. Wei, W.; Liu, L. Trustworthy distributed ai systems: Robustness, privacy, and governance. *ACM Comput. Surv.* **2024**. Available online: <https://dl.acm.org/doi/10.1145/3645102> (accessed on 1 September 2024). [[CrossRef](#)]
173. Liao, J.; Chen, Z.; Larsson, E.G. Over-the-air federated learning with privacy protection via correlated additive perturbations. In Proceedings of the 2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL, USA, 27–30 September 2022; pp. 1–8.
174. Wang, J.; Guo, S.; Xie, X.; Qi, H. Protect privacy from gradient leakage attack in federated learning. In Proceedings of the IEEE INFOCOM 2022-IEEE Conference on Computer Communications, London, UK, 2–5 May 2022; pp. 580–589.
175. Sun, J.; Li, A.; Wang, B.; Yang, H.; Li, H.; Chen, Y. Soteria: Provable defense against privacy leakage in federated learning from representation perspective. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 9311–9319.
176. Lee, H.; Kim, J.; Hussain, R.; Cho, S.; Son, J. On defensive neural networks against inference attack in federated learning. In Proceedings of the ICC 2021-IEEE International Conference on Communications, Montreal, QC, Canada, 14–23 June 2021; pp. 1–6.
177. Wei, K.; Li, J.; Ding, M.; Ma, C.; Yang, H.H.; Farokhi, F.; Jin, S.; Quek, T.Q.; Poor, H.V. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 3454–3469. [[CrossRef](#)]
178. Ponomareva, N.; Hazimeh, H.; Kurakin, A.; Xu, Z.; Denison, C.; McMahan, H.B.; Vassilvitskii, S.; Chien, S.; Thakurta, A.G. How to dp-fy ml: A practical guide to machine learning with differential privacy. *J. Artif. Intell. Res.* **2023**, *77*, 1113–1201. [[CrossRef](#)]
179. Dwork, C. Differential privacy. In *International Colloquium on Automata, Languages, and Programming*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1–12.
180. Lecuyer, M.; Atlidakis, V.; Geambasu, R.; Hsu, D.; Jana, S. Certified robustness to adversarial examples with differential privacy. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 19–23 May 2019; pp. 656–672.
181. Yu, D.; Xie, Z.; Yuan, Y.; Chen, S.; Qiao, J.; Wang, Y.; Yu, Y.; Zou, Y.; Zhang, X. Trustworthy decentralized collaborative learning for edge intelligence: A survey. *High-Confid. Comput.* **2023**, *3*, 100150. [[CrossRef](#)]
182. Chen, Y.; Luo, F.; Li, T.; Xiang, T.; Liu, Z.; Li, J. A training-integrity privacy-preserving federated learning scheme with trusted execution environment. *Inf. Sci.* **2020**, *522*, 69–79. [[CrossRef](#)]
183. Mo, F.; Haddadi, H.; Katevas, K.; Marin, E.; Perino, D.; Kourtellis, N. PPFL: Privacy-preserving federated learning with trusted execution environments. In Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services, Virtual, 24 June–2 July 2021; pp. 94–108.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.