IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 36, No. 1, February, 2022, pp. 49-59
Journal homepage: http://iieta.org/journals/ria

Security of Federated Learning: Attacks, Defensive Mechanisms, and Challenges

Mourad Benmalek*, Mohamed Ali Benrekia, Yacine Challal

Laboratoire des Méthodes de Conception des Systèmes, Ecole nationale Supérieure d'Informatique, BP 68M, 16309, Oued-Smar Algiers, Algeria

Corresponding Author Email: m_benmalek@esi.dz

https://doi.org/10.18280/ria.360106

Received: 10 December 2021
Accepted: 18 January 2022

Keywords:
artificial intelligence, federated learning, machine learning, privacy, security

ABSTRACT

Recently, a new Artificial Intelligence (AI) paradigm, known as Federated Learning (FL), has been introduced. It is a decentralized approach to apply Machine Learning (ML) on-device without risking the disclosure and tracing of sensitive and private information. Instead of training the global model on a centralized server (by aggregating the clients' private data), FL trains a global shared model by only aggregating clients' locally-computed updates (the clients' private data remains distributed across the clients' devices). However, as secure as the FL seems, it by itself does not give the levels of privacy and security required by today's distributed systems. This paper seeks to provide a holistic view of FL's security concerns. We outline the most important attacks and vulnerabilities that are highly relevant to FL systems. Then, we present the recent proposed defensive mechanisms. Finally, we highlight the outstanding challenges, and we discuss the possible future research directions.

1. INTRODUCTION

In recent years, improvements in the implementation of Machine Learning (ML) models, have significantly increased the adoption of this technology in a wide range real-world systems that revolutionize almost all industries [1-6]. Despite ML's enormous success, many domains can only desire to benefit from it, but are unable to do so due to two significant obstacles: (1) concerns about clients' data privacy, as well as the laws and regulations that govern them, and (2) inability to develop a ML model because of insufficient data or high training overheads.

In order to overcome these obstacles, Federated Learning (FL) [7] emerges as an effective technique to exploit distributed data and computing resources, in order to collaboratively train ML models, while adhering to laws and regulations, and protecting users' data security and privacy. As consequent, ML algorithms have become further integrated in the devices of end users. This new paradigm in ML, famous as "privacy-by-design", allows a number of clients' devices to train a ML model collaboratively. A key feature would be that the clients' private data remain stored on the client's device. By conducting model training at the clients' devices, aggregate analytics could be accomplished without having to collect the clients' data themselves [8].

However, as secure as the FL seems, it by itself does not give the levels of privacy and security demanded by today's distributed systems requirements [9]. Beyond fundamental and FL-specific restrictions, the security of FL systems themselves are essential for developing networks where users can collaborate, learn, and most importantly trust. FL systems are vulnerable to a slew of new attacks and threats that target each stage of training and deployment process. Attackers can exploit flaws in FL systems in a variety of ways. For example, an attacker may maliciously corrupt training data or local model updates on clients' devices before sending them to the central server. He may also intercept the model updates exchanged between the central server and the clients' devices and replace them with malicious model updates. In order to overcome these security threats, many researchers have proposed mechanisms that defend against FL attacks and vulnerabilities.

1.1 Contributions

In this paper, we seek to provide a holistic view of FL's security concerns. The main contributions of this paper are presented as follows:

- We outline the most important vulnerabilities and attacks in FL environments, such as poisoning attacks, inference attacks, communication attacks, and free-riding attacks.
- We present the recent proposed security mechanisms that defend against the security attacks and threats in FL systems.
- We highlight the outstanding challenges, and we address the future research opportunities to improve the security of FL systems.

1.2 Paper organization

This paper is organized as follows. The basics of FL are introduced in Section 2. The major attacks and threats that are relevant to FL settings are presented in Section 3. The recent proposed security defensive mechanisms that defend against the security attacks and threats are summarized in Section 4. Section 5 identifies the research challenges and discusses the future directions towards a robust and secure FL. Finally, Section 6 gives conclusion remarks.

2. BASICS OF FEDERATED LEARNING

FL [7-13] is a ML-based framework in which numerous clients cooperate to solve a ML problem, under the supervision and the coordination of a central server usually referred to as FL server. In other words, "*it is a distributed ML strategy that generates a global model by learning from multiple decentralized edge clients. FL enables on-device training, keeping the client's local data private, and further, updating the global model based on the local model updates*" [14].

From a privacy perspective, FL complies with the "privacy-by-design" guidelines made by the European Union Agency for Network and Information Security (ENISA) [15] since the clients' private data are held locally and are not transferred to the FL server (the client's device uses meaningful data to update the local models, and model updates are aggregates of the client's private data). This has made FL a more privacy-friendly technique, attracting many communities to use it instead of the standard ML technique based on centralized data collection and centralized model training.

In order to understand the different FL security aspects presented in later sections, we give in this section a non-exhaustive overview of different concepts, techniques, and approaches used to implement this AI paradigm in practice.

2.1 Federated learning implementation

FL can be viewed as an iterative learning process in which the global model is improved with each round. FL process flow follows three steps [16]:

- **Model initialization:** each client's device receives the initial ML model from the FL server.
- **Local model training:** each client's device trains its own model with the client's local training data.
- **Aggregation of local models:** the FL server collects updated model weights, and then it aggregates them to the global model, which is subsequently updated to replace each client's local model.

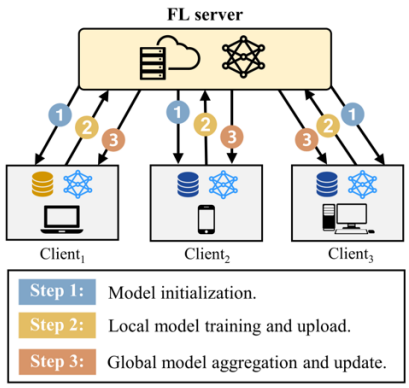

Figure 1. A schematic diagram of FL process flow

As shown in Figure 1, FL is in a continuous iterative learning process that repeats the above steps (steps 2 and 3) to maintain the global model updated across all the participants.

2.2 Network topology in federated learning

Based on network topology, the FL can be categorized into two classes: (1) Centralized FL, and (2) Fully decentralized FL [17].

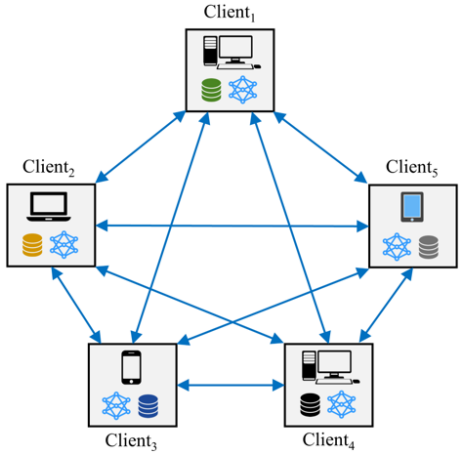

Figure 2. Fully decentralized FL

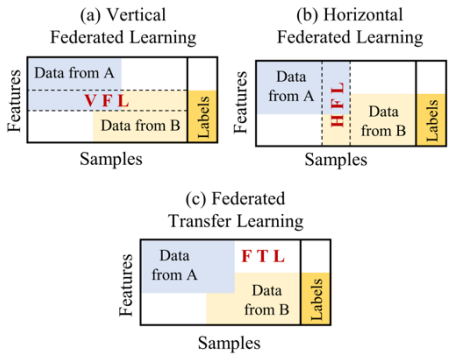

Figure 3. Data partition in FL

2.2.1 Centralized federated learning

As shown in Figure 1, even though FL is typically considered as a decentralized approach, a centralized server is required to collect clients' model updates and aggregate them to the global model. Unlike in traditional ML systems where the global model is trained on a centralized server by aggregating the clients' private data, the centralized server in FL trains the global model by only aggregating clients' updates. The Gboard keyboard application (developed by Google) is an example of centralized FL systems.

2.2.2 Fully decentralized federated learning

As shown in Figure 2, no central server is required in fully decentralized FL systems. In this type of FL systems, participants improve their models by sharing information with their neighbors using Peer-2-Peer (P2P) communications.

2.3 Data partition in federated learning

As shown in Figure 3, the distribution of data among clients classifies FL into three classes: (1) Vertical Federated Learning (VFL), (2) Horizontal Federated Learning (HFL), and (3) Federated Transfer Learning (FTL). The three classes are defined as follows [10]:

2.3.1 Vertical Federated Learning

VFL is frequently used when two datasets need to share identical sample IDs, but different feature spaces. An example for VFL approach would be a scenario from business domain, where a client A (Amazon) has information about customers' book purchases on Amazon, and client B (Goodreads) has information about customers' book reviews. Using these two

sets of datasets from different feature spaces, one may better serve the customers by using book reviews information to provide better book recommendation to the customers browsing Amazon's books.

2.3.2 Horizontal Federated Learning

In this class, there is some overlap between the features of data dispersed over multiple participants, while the data are fairly distinct in sample space. Clients in this type of FL share similar features in terms of domain, usage style of derived statistical information, or any other FL outcome. An example for HFL approach would be a scenario from medical domain, where multiple hospitals are collaborating to train a ML model (using medical images) for detecting cancer cells. Due to the laws and constraints of private medical data, medical images cannot be shared as is. However, with FL, information on such sensitive data may be safely transmitted through a secure aggregated update from each hospital.

2.3.3 Federated Transfer Learning

FTL is typically used when FL participants have little overlap in both sample and feature spaces. FTL enables to move the knowledge of one domain (the source domain) to another domain (the target domain) to achieve better learning results. An example for FTL approach would be a scenario of training a book recommendation model from the user's past browsing behavior.

2.4 Aggregation algorithms in federated learning

The aggregation algorithm can be defined as the logic that combines the locally-computed updates from all the clients participating in the training phase [18]. These algorithms play a cornerstone role in any FL system. For that, several aggregation algorithms have been proposed in the literature. We present in the following some of the most used aggregation algorithms:

2.4.1 FedAvg

The Federated Averaging algorithm (*FedAvg*) is regarded as the de facto optimization algorithm in the federated setting. This aggregation algorithm, implemented by Google [7], runs Stochastic Gradient Descent (SGD) in parallel on K devices, where K is a small fraction of the total clients' devices in the FL network. After that, the clients' devices communicate their model updates to a FL server, where the global model is built using averaging logic to compute the weighted sum of all the received updates. Although *FedAvg* has shown empirical success in heterogeneous settings, it does not entirely address the underlying challenges associated with heterogeneity [19].

2.4.2 FedProx

To address the challenges of heterogeneity in FL environments, Li et al. [19] proposed *FedProx*. As stated by the authors, "*FedProx algorithm can be viewed as a generalization and re-parametrization of FedAvg*". They proposed to add a proximal term to the local subproblem that helps to effectively limit the impact of variable local updates, and thus improve the stability of the method. Moreover, they proved that *FedProx* achieves better convergence and stability compared to *FedAvg* in heterogeneous FL environments.

2.4.3 SMC-Avg

A secure aggregation algorithm, called Secure Multi-Party Computation Averaging (*SMC-Avg*) was proposed by Bonawitz et al. [20, 21]. *SMC-Avg* algorithm is based on the concept of the Secure Multi-party Computation (SMC), which aggregates private values of clients' models without revealing information about their private values. *SMC-Avg* algorithm is suitable to deal with the problems of the mobile device-based FL networks.

3. SECURITY ATTACKS IN FEDERATED LEARNING

While FL comes with privacy guarantees regarding the protection of private data in ML settings, exchanging the model updates, as well as the large number of training iterations and communications expose the FL system to curious and malicious attackers [22-24]. Several attacks are already identified against FL systems. In this section, we present the most important attacks and vulnerabilities in FL environments. We categorize attacks against FL systems into four groups: poisoning attacks, inference attacks, communication attacks, and free-riding attacks. Table 1 summarizes the properties of attacks on FL settings.

3.1 Poisoning attacks

Typically, these attacks are undertaken by the insiders on FL systems [17, 25]. They try to prevent a model from being learned at all, or to bias the model to produce inferences that are suitable to the attacker. Regarding the attacker's capabilities, we classify poisoning attacks into two types: data poisoning attacks and model poisoning attacks [26].

Table 1. Summary of attacks on FL systems

Attacks	Key idea	Source of attacks			
		Compromised		Communication	Distributed nature of FL
		Clients	Server		
Poisoning attacks	Manipulate client's data or local model to bias the global model performance/accuracy	✓	✓	-	-
Inference attacks	Analyze the clients' updates in the goal of illegitimately gain knowledge about FL process and use this knowledge to extract meaningful insights about the training data	✓	✓	✓	-
Communication attacks	Intercept the clients' updates, then replace them with faulty or malicious updates. Moreover, communication bottlenecks can drastically destabilize the FL system	-	-	✓	✓
Free-riding attacks	Craft fake local updates with the purpose of acquiring the global shared model without really participating to the FL process	✓	-	-	-

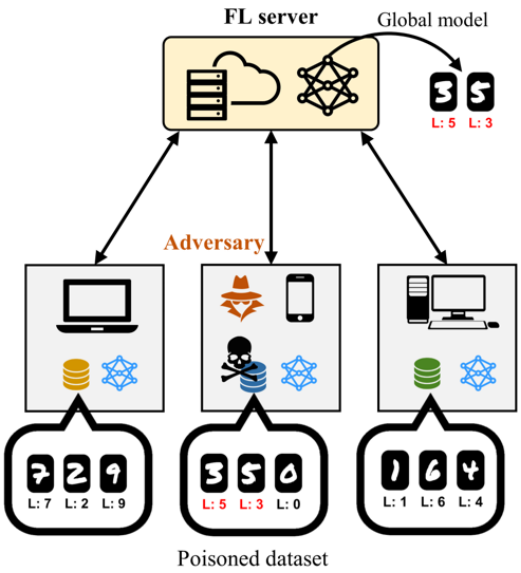

Figure 4. An example of data poisoning attack in FL systems

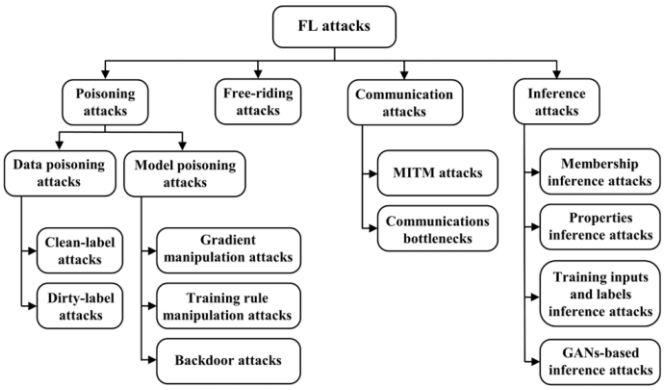

Figure 5. A taxonomy of attacks on FL systems

3.1.1 Data poisoning attacks

They are attacks that compromise the clients' training data to distort the output of the global model at inference time. As shown in Figure 4, an adversary participant may adversarially manipulate existing inputs or add poison instances to corrupt the global model's output [27]. As shown in Figure 5, we recognize two classes of attacks in this category:

Clean-label attacks. In this category of attacks, the adversary assumes that data are certified as belonging to the correct class, so, he cannot change the label of any input data, and he must craft the poisoned training data to appear as correctly labeled as the non-corrupted data. Tolpegin et al. [27] achieved 100% attack success rate on the *dog-vs-fish* classification task using the feature collision technique, where the attacker exploits the high complexity and nonlinearity of the function f that propagates an input X through the neural network to the last layer, it is possible to find an example (a dog's picture) that "collides" with the target (fish class) in feature space, while simultaneously being close to the base instance b (dog class) in input space.

Dirty-label attacks. In this category of attacks, the adversary participant can add, remove, or change any data samples he intends to misclassify with the desired target label into the training set. A very known example of dirty-label poisoning attack is label-flipping [28, 29] which has been demonstrated to be effective in traditional ML settings and become a feasible strategy to implement in FL settings.

3.1.2 Model poisoning attacks

They are attacks where malicious clients directly change the learning rule and affect gradients that they share with the FL server during the training process [26]. We can recognize several techniques in this category:

Gradient manipulation attacks. In this type of attacks, adversaries perform adversarial manipulations of the training process by manipulating local model gradients to compromise the global model performance and reduce the overall accuracy [26]. This technique can be used for example to modify an image classifier so that it assigns an attacker-chosen label to images with certain features or force a word predictor to complete certain sentences with an attacker-chosen word [22].

Training rule manipulation attacks. In this type of attacks, if the attackers have access to the model, they may be able to manipulate its output such that it has the same distribution as correct model updates, making the attack undetectable [30]. For example, Bhagoji et al. [31] added a penalty term to the objective function in order to reduce the distance between the wrong and the correct weight update distributions. This modification helped to successfully achieve a non-detectable targeted model poisoning attack.

Backdoor attacks. These attacks can be viewed as a type of model poisoning attacks. A malicious participant trains its local model with poisoned data and uploads the locally-computed updates to the FL server, embedding a backdoor to the global model after unwitting aggregative optimization [32]. Bagdasaryan et al. experiment on how backdoor attacks is implemented [22, 33].

3.2 Inference attacks

These attacks are adversarial algorithms that are capable of extracting meaningful insights about the training data via analysis of locally-computed updates [30, 34]. Inference attacks fall into four categories:

3.2.1 Membership inference attacks

Under this category, attackers aim to identify whether a specific sample belongs to a given class represented by the model and/or whether a specific sample was used to train the model [35]. For instance, an adversary can determine whether a given patient profile was used to train a classifier associated with a disease.

3.2.2 Properties inference attacks

Under this category, attackers attempt to induce properties of other clients' private data that are independent of the features which characterize the FL model classes [35]. For instance, a property inference attack would be for facial recognition models, if a class corresponds to a certain individual, the adversary task would be determining whether the individual wears glasses or not [36].

3.2.3 Training inputs and labels inference attacks

These attacks are much destructive than previous ones since they can not only determine the label of the FL model classes but also the client's training inputs. The authors showed that the proposed optimization algorithm can obtain both the training inputs and the labels in just a few rounds [37, 38].

3.2.4 GANs-based inference attacks

Under this category, powerful attacks can be performed through *Generative Adversarial Networks* (*GANs*). GANs

have been recently proposed by Hitaj et al. [39] and are still being intensively developed. The architecture of GANs is composed of two models: *Discriminator* D and *Generator* G. The GAN-based attacks exploit the real-time nature of the FL process which allows the attacker to train a GAN generating synthetic samples that are statistically representative of the training data. It should be noted that GANs generate these samples without having the right to access clients' private data. The GAN is first initialized with random noise, and at each round, it is trained to mimic the inputs in the training set of the discriminative network. Figure 6 shows an example of GANs-based inference attack.

3.3 Communication attacks

As mentioned above, FL is based on an iterative learning process in which the global model is improved with each round. In order to update the shared global model and maintain it updated across all the participants, a large number of communication messages should be exchanged between the FL server and all the participants over a given network (typically, a FL process achieves stability and convergence after a large number of communication iterations). Thus, a non-secure communication channel is considered an open vulnerability. Moreover, the communication bottlenecks can drastically destabilize the FL system [17, 40, 41].

3.3.1 Man-In-The-Middle attacks

In this type of attacks, the Man-in-the-Middle (*MITM*) intercepts the model updates exchanged between the participants and the FL server and replaces them with malicious updates [42]. Typically, a *MITM* attack is carried out through interfering with real networks or by creating fake networks that the *MITM* controls. After that, the compromised communication is frequently stripped of any encryption in order to steal, modify or redirect the model updates [43]. This attack is difficult to detect because the attacker may be silently observing or re-encrypting the hijacked communication to its designed destination once saved or modified.

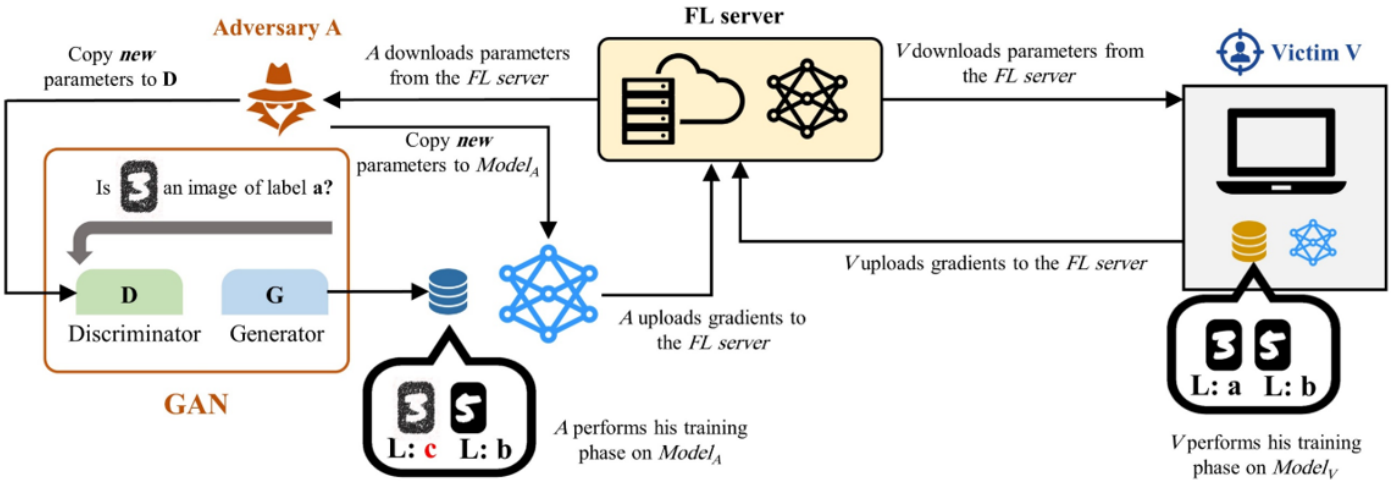

Figure 6. An example of GANs-based inference attack in FL systems

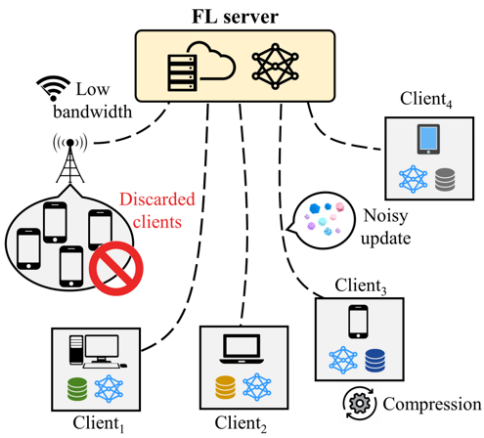

Figure 7. Communications bottlenecks in FL systems

3.3.2 Communications bottlenecks

As shown in Figure 7, communication bottlenecks can drastically destabilize the FL system because they increase the number of participants who drop out. Further, discarding clients depending on their connection state causes eventual biases in the global shared model and affects the aggregation of individual updates. Furthermore, techniques that seek to decrease the communication overhead [44-46], such as compression, can be exploited in a destructive way to inject noise in individual updates and deteriorate their quality.

3.4 Free-riding attacks

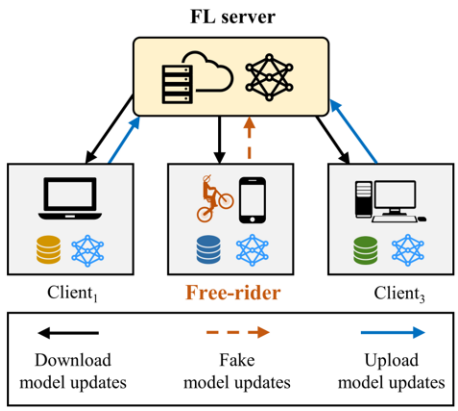

Figure 8. An example of free-riding attack in FL systems

Free-rider attacks consist in crafting fake local updates with the purpose of acquiring the global shared model without really participating to the FL process [47, 48]. Free-rider is generally referred to an individual who benefits from services,

public goods, or resources, of a communal nature, but do not pay for them [49]. In free-riding attacks, there could be two main motives to submit fake updates: (1) a client may want to save local CPU cycles or other computing resources, also, (2) a client may not have the required data, or is concerned about data privacy violations, so that local data are not available for model training [48]. As shown in Figure 8, the strategy of a free-rider, to obtain the final aggregated model, consists in participating in FL cycle by mimicking local updating through the sharing of opportune crafted parameters.

4. DEFENSES IN FEDERATED LEARNING

Recent studies into FL security have tried to stress-test existing techniques for preventing private information extraction and model corruption. In this section, we present a review of the recent proposed mechanisms that defend against the security attacks and threats raised in Section 3. Figure 9 and Table 2 summarize the prominent types of defensive mechanisms in FL.

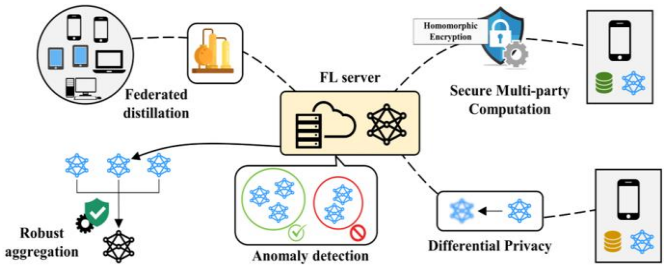

Figure 9. An overview of defensive mechanisms in FL systems

4.1 Differential privacy

The basic goal of Differential Privacy (*DP*) is to ensure that, with high probability, no single record in a given client's dataset can be meaningfully discriminated from the other records [50-52]. The basic idea behind this technique is to introduce noise to the client's sensitive attributes before sharing individual updates with the FL server [52]. As a consequence, each client's privacy is protected. Meanwhile, the statistical data quality loss caused by the introduced noise of each client is rather minor compared with the greater data privacy protection. In FL environment, *DP* distorts client updates so that the existence or absence of any given record in a client's private data has no major impact on the update shared by the client.

Based on DP, McMahan et al. [53] proposed *DP-FedAvg*, a noised version of *FedAvg* (presented earlier in Section 2.4.1) that satisfies user-level differential privacy. The main goal of *DP-FedAvg* is to provide a strong guarantee that the trained model protects the privacy of clients' data without affecting model quality. Later, Augenstein et al. [54] proposed *DP-FedAvg-GAN* with the purpose to protect the clients' training data against GANs-based attacks.

Ma et al. showed that *DP* can be used as defense against data poisoning attacks [55-57]. Further, Bagdasaryan et al. [22] demonstrated that *DP* applied to clients' models can successfully defend against backdoor attacks, but the required noise levels significantly baffle the model's learning ability (i.e., the more we apply noise the higher we protect data but the utility decreases drastically). Furthermore, Lecuyer et al. [58] studied the possibility of using *DP* as a defense mechanism against inference attacks.

4.2 Secure multi-party computation

This technique (SMC) was originally proposed with the purpose of creating methods for distrustful parties to jointly compute a function over their inputs while keeping them private [59]. In FL environment, *SMC*, which is based on cryptographic methods, is used to protect the privacy of client data.

In this direction, Google proposed a secure aggregation algorithm [60] that securely aggregates the clients' updates by using *SMC* to compute the weighted averages of received updates. Upon receiving a sufficient number of clients' updates, the FL server can decrypt the average update. This is possible because client updates are transferred via additive secret sharing. As a consequence, the clients' private data are protected.

In the same direction, Xu et al. [61] proposed a privacy-preserving approach, called *VerifyNet*, that aim to realize secure gradient aggregation and verification. This approach employs a double-masking method (based on Shamir's secret sharing and homomorphic hash function) making it difficult for malicious adversaries to infer training data. Moreover, *VerifyNet* guarantees that the clients may verify the FL server's results, ensuring the FL server's reliability.

Table 2. Summary of defensive mechanisms in FL systems

Defensive mechanisms	Key idea	Attacks
Differential Privacy	Introduce noise to the client's sensitive data before sharing individual updates with the FL server	Data poisoning attacks Backdoor attacks Inference attacks
Secure Multi-party Computation	Encrypt clients' uploaded parameters	Inference attacks MITM attacks Free-riding attacks
Anomaly detection	Analyze clients' updates to identify misbehaving clients	Model poisoning attacks Data poisoning attacks
Robust aggregation	Detect malicious individual updates during training process	Inference attacks Model poisoning attacks Data poisoning attacks
Federated distillation	Transfer knowledge from a fully trained model to another model	Communications bottlenecks MITM attacks Inference attacks GANs-based attacks

Although SMC-based methods [60-64] provide a secure aggregation of the protected clients' updates, they induce significant extra communication overhead among clients which may be unaffordable for some devices and networks. Moreover, they make countermeasures to security attacks (such as model poisoning attacks) ineffective, and attacks become difficult to detect by the FL server.

4.3 Anomaly detection

This category of defenses (also called *outlier detection*) uses analytical and statistical methods to identify events that do not conform to an expected pattern or activity. In order to identify misbehaving clients in FL settings, anomaly detection mechanisms could be used. For that, the FL server analyzes individual updates and their impact on the global shared model to discover attacks such as poisoning attacks. However, these mechanisms are most likely to fail when it comes to targeted backdoor attacks.

Chen et al. [65] proposed an anomaly detection-based technique, in which the FL server can reconstruct the clients' updated models and compare the model performance metrics against a validation dataset with respect to the model obtained by aggregating all updates except that of the client. After that, any client updates that decrease model performance, according to some criteria or threshold, are marked as outliers.

Cao et al. [66] proposed another defense technique called *Sniper*. The proposed approach can recognize honest clients and decrease the success rate of poisoning attacks to 2% even when multiple attackers are colluded. In *Sniper*, the FL server identifies legitimate clients by solving a maximum clique problem in a graph constructed with clients' shared updates as vertices and if the Euclidean distance between two vertices is small enough, then there exists an edge between them. The FL server then finds the maximum clique in the graph, and aggregates vertices (local models) in the clique to get the global FL model.

In another work, Fung et al. [67] presented *FoolsGold*, a novel defensive mechanism to cope with poisoning attacks. In their work, the authors defined poisoning sybils as malicious clients creating multiple fake identities to mount more powerful poisoning attacks on FL and transfer fake updates to the FL server. After that, they presented their defensive technique that leverages client similarity to identify poisoning sybils based on the diversity of client updates, because in the distributed learning process, each client's private data has a unique distribution, while sybils aim to the same objective and will share updates that appear more similar to each other than non-malicious clients. On the contrary to other defensive techniques, this mechanism does not require any changes of the protocol executed on client-side. In addition, it doesn't need prior knowledge of the number of malicious clients.

Many other research works have been developed in this category [29, 68]. In these studies, the proposed approaches aim to detect a deviation in individual updates from each client, as well as the verification of honesty of training inputs.

4.4 Robust aggregation

As mentioned above, robust aggregation algorithms play a cornerstone role in any FL system, and several algorithms [69-72] have been proposed in the literature. Theses algorithms are used to detect and discard faulty model updates during the training process. Moreover, robust aggregation algorithms should be able to sustain clients' dropout and communications instabilities. They can also address the challenges of heterogeneity in FL environments [19].

In addition to algorithms presented in Section 2.4, Lu and Fan [71] proposed an aggregation algorithm that uses Gaussian distribution to measure clients' potential contributions. Further, they proposed layer-wise optimizing steps, so the aggregation works well on different functional units in the neural network. Furthermore, they showed that the proposed algorithm achieves better convergence and stability compared to the well-known aggregation algorithm *FedAvg* [7]. Moreover, this algorithm outperforms *FedAvg* in terms of robustness against attacks.

4.5 Federated distillation

Federated Distillation [73-76] (also called *Federated Knowledge Distillation*) is regarded as an alternative of the model compression method. As mentioned above, a large number of communication messages should be exchanged between all the clients and the FL server to update the global shared model and maintain it updated across all the clients. However, these exchanged messages can drastically destabilize the FL system. For that, federated distillation is a compelling FL solution in which a fully trained model transfers knowledge to a small model step by step on what needs to be done. The idea of sharing knowledge only instead of model parameters can be used to improve the security and privacy of the clients' private data. Moreover, this concept helps to save communication and reduce computation overheads.

In this direction, Li and Wang [76] proposed an algorithm of federated distillation called *FedMD*. In this algorithm, the authors seek to transfer knowledge from a fully trained model to a smaller model. Typically, the knowledge is a pre-trained model's logit, transferred to a small model for compression. The knowledge can also be a collection of other small models' logits, in that the collection of forecasts is often more accurate than individual predictions.

5. CHALLENGES AND FUTURE OPPORTUNITIES

To complete our overview, we discuss the outstanding challenges, and we address the future research opportunities to improve the robustness of FL environments, summarized in the following recommendations:

5.1 Ensuring and building trust

When private data is stored on clients' devices, FL servers have little scope for manual verification. Thus, the question that might arise is: *how can the FL server trust reports, such as model updates, from clients*? Cryptographic primitives may offer promise for secure calculations using private data. For example, Zero-Knowledge Proofs may be used to ensure that participants are transferring individual updates with pre-specified properties to defend against backdoor attacks and model corruption attacks, while avoiding the disclosure of clients' private data. However, it is important to understand better how to effectively implement and apply these cryptographic primitives and protocols, especially in large-scale FL systems.

5.2 Ensuring traceability

Ensuring traceability of the global model throughout the lifecycle of the FL process is another major challenge in FL settings. For example, if a model parameter is modified or updated during the training process, it is important to have backward tracking ability to determine which client's update caused that change. In this direction, we believe that blockchain can provide attractive solutions for FL due to its unique features, such as traceability, immutability, and decentralization [77]. By using blockchain, any update events and client actions are transparently tracked by all network entities. Moreover, a model parameter modification or update can be easily traced through blockchain transaction logs [78].

5.3 Ensuring a trade-off between security/privacy and performance/accuracy

As presented in this paper, many defensive approaches for FL security have been designed, each of which is proposed to address different security/privacy and performance/accuracy objectives. However, each one of them has its own pros and cons. Thus, the question that might arise is: *how to ensure a better trade-off between security/privacy and performance/accuracy*? On the one hand, designing efficient FL approaches should not undermine the robustness of the proposed mechanisms. If the approach is not secure against the different attacks, an attacker can affect the FL process. For example, if the encryption level in SMC-based mechanisms or the quantity of noise in DP-based mechanisms is not enough, the clients that participate to the FL process still suffer from the risk of privacy leakage. On the other hand, if the encryption level is too high or too much noise added to the exchanged updates, the FL model severally suffers from low accuracy.

5.4 Deploying decentralized federated learning

In the traditional FL systems, a third party (which is the FL server) is required for system initialization, supervision, and global model aggregation. However, a setting where no central server is required in the system is a potential framework for collaboration among applications that do not trust any third party. For example, we can consider a strategy where each client that participates to the FL system could be elected as a server using a round robin method. It would be interesting to explore if existing attacks and vulnerabilities on the traditional FL still apply in this decentralized scenario, as well as new attack surfaces that may be opened. Therefore, the defensive mechanisms of decentralized FL should also be investigated.

6. CONCLUSION

As FL is becoming widely used in many practical applications that aim to preserve users' privacy, protecting the security of this new paradigm becomes an urgent need. In this paper, we have presented a survey on security concerns in FL settings. Specifically, we have revisited existing security attacks and threats towards FL, such as poisoning attacks, inference attacks, communication attacks, and free-riding attacks. Furthermore, we have presented the current defensive techniques based on differential privacy, secure multi-party computation, anomaly detection, robust aggregation, and federated distillation. We showed that there is not yet a defensive mechanism that fulfills all the security/privacy and performance/accuracy objectives and that there is still much work to be done. After that, we have presented four interesting research topics in this field. We hope that such survey can serve as a valuable reference for researchers in both FL and security fields.

REFERENCES

[1] Bkassiny, M., Li, Y., Jayaweera, S.K. (2013). A survey on machine-learning techniques in cognitive radios. IEEE Communications Surveys & Tutorials, 15(3): 1136-1159. https://doi.org/10.1109/surv.2012.100412.00017

[2] Ucci, D., Aniello, L., Baldoni, R. (2019). Survey of machine learning techniques for malware analysis. Computers & Security, 81: 123-147. https://doi.org/10.1016/j.cose.2018.11.001

[3] Joshua, E.S.N., Chakkravarthy, M., Bhattacharyya, D. (2020). An extensive review on lung cancer detection using machine learning techniques: A systematic study. Revue d'Intelligence Artificielle, 34(3): 351-359. https://doi.org/10.18280/ria.340314

[4] Lin, W., Hu, Y., Tsai, C. (2012). Machine learning in financial crisis prediction: A survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(4): 421-436. https://doi.org/10.1109/tsmcc.2011.2170420

[5] Ketepalli, G., Bulla, P., (2020). Review on generative deep learning models and datasets for intrusion detection systems. Revue d'Intelligence Artificielle, 34(2): 215-226. https://doi.org/10.18280/ria.340213

[6] Hossain, E., Khan, I., Un-Noor, F., Sikander, S.S., Sunny, M.S.H. (2019). Application of big data and machine learning in smart grid, and associated security concerns: A review. IEEE Access, 7: 13960-13988. https://doi.org/10.1109/access.2019.2894819

[7] McMahan, H.B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A. (2017). Communication-efficient learning of deep networks from decentralized data. The 20th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA. pp. 1273-1282.

[8] Aledhari, M., Razzak, R., Parizi, R.M., Saeed, F. (2020). Federated learning: A survey on enabling technologies, protocols, and applications. IEEE Access, 8: 140699-140725. https://doi.org/10.1109/Access.2020.3013541

[9] Bouacida, N., Mohapatra, P. (2021). Vulnerabilities in federated learning. IEEE Access, 9: 63229-63249. https://doi.org/10.1109/access.2021.3075203

[10] Yang, Q., Liu, Y., Chen, T., Tong, Y. (2019). Federated machine learning: Concept and Applications. ACM Transactions on Intelligent Systems and Technology, 10(2): 1-9. https://doi.org/10.1145/3298981

[11] Guizani, M. (2020). A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. IEEE Internet Things Journal, 8(7): 5476-5497. https://doi.org/10.1109/jiot.2020.3030072

[12] Lim, W.Y.B, Luong, N.C., Hoang, D.T., Jiao, Y., Liang, Y.C., Yang, Q., Niyato, D., Miao, C. (2020). Federated learning in mobile edge networks: A comprehensive survey. IEEE Communications Surveys & Tutorials, 22(3): 2031-2063. https://doi.org/10.1109/comst.2020.2986024

[13] Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T., Yu, H. (2019). Federated Learning. Morgan & Claypool.
[14] Imteaj, A., Thakker, U., Wang, S., Li, J., Amini, M.H. (2021). A survey on federated learning for resource-constrained IoT devices. IEEE Internet of Things Journal, 9(1): 1-24. https://doi.org/10.1109/jiot.2021.3095077
[15] Danezis, G., Domingo-Ferrer, J., Hansen, M., Hoepman, J.H., Metayer, D.L., Tirtea, R., Schiffner, S. (2015). Privacy and data protection by design-from policy to engineering. arXiv preprint arXiv:1501.03726.
[16] Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., Gao, Y. (2021). A survey on federated learning. Knowledge-Based Systems, 216: 106775. https://doi.org/10.1016/j.knosys.2021.106775
[17] Mothukuri, V., Parizi, R.M., Pouriyeh, S., Huang, Y., Dehghantanha, A., Srivastava, G. (2021). A survey on security and privacy of federated learning. Future Generation Computer Systems, 115: 619-640. https://doi.org/10.1016/j.future.2020.10.007
[18] Nilsson, A., Smith, S., Ulm, G., Gustavsson, E., Jirstrand, M. (2018). A performance evaluation of federated learning algorithms. The 2nd Workshop on Distributed Infrastructures for Deep Learning, Rennes, France, pp. 1-8. https://doi.org/10.1145/3286490.3286559
[19] Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V. (2018). Federated optimization in heterogeneous networks, arXiv preprint arXiv:1812.06127.
[20] Bonawitz, K., Ivanov, B., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K. (2016). Practical secure aggregation for federated learning on user-held data. arXiv preprint arXiv:1611.04482.
[21] Bonawitz, K., Ivanov, B., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. The 24th ACM SIGSAC Conference on Computer and Communications Security, Dallas, Texas, USA, pp. 1175-1191. https://doi.org/10.1145/3133956.3133982
[22] Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V. (2020). How to backdoor federated learning. The 23rd International Conference on Artificial Intelligence and Statistics, Palermo, Italy, pp. 2938-2948.
[23] Sun, G., Cong, Y., Dong, J., Wang, Q., Liu, J. (2020). Data poisoning attacks on federated machine learning. arXiv preprint arXiv:2004.10020.
[24] Weng, H., Zhang, J., Xue, F., Wei, T., Ji, S., Zong, Z. (2020). Privacy leakage of real-world vertical federated learning. arXiv preprint arXiv:2011.09290.
[25] Muñoz-González, L., Biggio, B., Demontis, A., Paudice, A., Wongrassamee, V., Lupu, E.C., Roli, F. (2017). Towards poisoning of deep learning algorithms with back-gradient optimization. The 10th ACM Workshop on Artificial Intelligence and Security, Dallas, Texas, USA, 27-38.
[26] Shejwalkar, V., Houmansadr, A. (2021). Manipulating the byzantine: optimizing model poisoning attacks and defenses for federated learning. The Network and Distributed System Security Symposium. https://dx.doi.org/10.14722/ndss.2021.23xxx
[27] Tolpegin, V., Truex, S., Gursoy, M.E., Liu, L. (2020). Data poisoning attacks against federated learning systems. The European Symposium on Research in Computer Security, Guildford, United Kingdom, pp. 480-501. https://doi.org/10.1007/978-3-030-58951-6_24
[28] Biggio, B., Nelson, B., Laskov, P. (2012). Poisoning attacks against support vector machines. The 29th International Conference on International Conference on Machine Learning, Edinburgh, Scotland, pp. 1467-1474.
[29] Fung, C., Yoon, C.J., Beschastnikh, I. (2020). The limitations of federated learning in sybil settings. The 23rd International Symposium on Research in Attacks, Intrusions and Defenses, 301-316.
[30] Kairouz, P., McMahan, H.B., Avent, B., et al. (2021). Advances and open problems in federated learning. Foundations and Trends® in Machine Learning, 14(1-2): 1-210. http://dx.doi.org/10.1561/2200000083
[31] Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S. (2019). Analyzing federated learning through an adversarial lens. The International Conference on Machine Learning, California, USA, pp. 634-643.
[32] Yin, Z., Yuan, Y., Guo, G.P., Zhou, P. (2021). Backdoor attacks on federated learning with lottery ticket hypothesis. arXiv preprint arXiv:2109.10512.
[33] Sun, Z., Kairouz, P., Suresh, A.T., McMahan, H.B. (2019) Can you really backdoor federated learning? arXiv preprint arXiv:1911.07963.
[34] Lee, H., Kim, J., Hussain, R., Cho, S., Son, J. (2021). On defensive neural networks against inference attack in federated learning. The 2021 IEEE International Conference on Communications, Montreal, Canada, pp. 1-6. https://doi.org/10.1109/icc42927.2021.9500936
[35] Melis, L., Song, C., De-Cristofaro, E., Shmatikov, V. (2019). Exploiting unintended feature leakage in collaborative learning. The 40th IEEE Symposium on Security and Privacy, CA, USA, pp. 691-706. https://doi.org/10.1109/SP.2019.00029
[36] Fredrikson, M., Jha, S., Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. ACM SIGSAC Conference on Computer and Communications Security, Colorado, USA, pp. 1322-1333. https://doi.org/10.1145/2810103.2813677
[37] Zhu, L. Han, S. (2020). Deep leakage from Gradients. In: Yang Q., Fan L., Yu H. (eds) Federated Learning. Lecture Notes in Computer Science, vol 12500. Springer, Cham. https://doi.org/10.1007/978-3-030-63076-8_2
[38] Zhao, B., Mopuri, K.R., Bilen, H. (2020). iDLG: Improved deep leakage from gradients. arXiv preprint arXiv:2001.02610.
[39] Hitaj, B., Ateniese, G., Perez-Cruz, F., Deep models under the GAN: Information leakage from collaborative deep learning. The 24th ACM SIGSAC Conference on Computer and Communications Security, Texas, USA, pp. 603-618. https://doi.org/10.1145/3133956.3134012
[40] Konecný, J., McMahan, H.B., Yu, F.X., Richtarik, P., Suresh, A.T., Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. NIPS Workshop on Private Multi-Party Machine Learning, Barcelona, Spain, pp. 1-8.
[41] Smith, S.L., Kindermans, P.J., Ying, C., Le, Q.V. (2018). Don't decay the learning rate, increase the batch size. The 6th International Conference on Learning Representations, Vancouver, Canada, pp. 1-11.
[42] Wang, D., Li, C., Wen, S., Nepal, S., Xiang, Y. (2021). Man-in-the-middle attacks against machine learning classifiers via malicious generative models. IEEE

Transactions on Dependable and Secure Computing, 18(5): 2074-2087. https://doi.org/10.1109/tdsc.2020.3021008
[43] Datta, S. (2020). Vulnerabilities of smart homes. In Chatterjee, P., Benoist, E., Nath, A. (eds). Applied approach to privacy and security for the internet of things. IGI-Global, Hershey, Pennsylvania, USA, pp. 1-25.
[44] Chen, Y., Sun, X., Jin, Y. (2020). Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation. IEEE Transactions on Neural Networks and Learning Systems, 31(10): 4229-4238. https://doi.org/10.1109/tnnls.2019.2953131
[45] Chen, Z., Liao, W., Hua, K., Lu, C., Yu, W. (2021). Towards asynchronous federated learning for heterogeneous edge-powered internet of things. Digital Communications and Networks, 7(3): 317-326. https://doi.org/10.1016/j.dcan.2021.04.001
[46] Nishio, T., Yonetani, R. (2019). Client selection for federated learning with heterogeneous resources in mobile edge. The 2019 IEEE International Conference on Communications, Shanghai, China, pp. 1-7. https://doi.org/10.1109/ICC.2019.8761315
[47] Lin, J., Du, M., Liu, J. (2019). Free-riders in federated learning: Attacks and defenses. arXiv preprint arXiv:1911.12560.
[48] Fraboni, Y., Vidal, R., Lorenzi, M. (2021). Free-rider attacks on model aggregation in federated learning. The 24th International Conference on Artificial Intelligence and Statistics, San Diego, California, pp. 1846-1854.
[49] Baumol, W.J. (2014). Welfare Economics and the Theory of the State. In Rowley, C.K., Schneider, F. (eds), The Encyclopedia of Public Choice. Springer, Boston, MA. https://doi.org/10.1007/978-0-306-47828-4_214
[50] Dwork, C. (2018). Differential privacy: A survey of results. The International Conference on Theory and Applications of Models of Computation, Xi'an, China, pp. 1-19. https://doi.org/10.1007/978-3-540-79228-4_1
[51] Dwork, C., McSherry, F., Nissim, K., Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. Theory of Cryptography Conference, New York, USA, pp. 265-284. https://doi.org/10.1007/11681878_14
[52] Dwork, C., Roth, A. (2014). The Algorithmic Foundations of Differential Privacy. Now Publishers Inc., Hanover, MA, USA.
[53] McMahan, H.B., Ramage, D., Talwar, K., Zhang, L. (2018). Learning differentially private recurrent language models. The 6th International Conference on Learning Representations, Vancouver, BC, Canada, pp. 1-14.
[54] Augenstein, S., McMahan, H.B., Ramage, D., Ramaswamy, S., Kairouz, P., Chen, M., Mathews, R.Y., Arcas, B.A. (2020). Generative models for effective ML on private, decentralized Datasets. The 8th International Conference on Learning Representations, Addis Ababa, Ethiopia, pp. 1-26. https://doi.org/10.48550/arXiv.1911.06679
[55] Ma, Y., Zhu, X., Hsu, J. (2019). Data poisoning against differentially-private learners: Attacks and defenses. The 28th International Joint Conference on Artificial Intelligence, Macao, China, pp. 4732-4738. https://doi.org/10.48550/arXiv.1903.09860
[56] Abadi, M., Chu, A., Goodfellow, I., McMahan, H.B., Mironov, I., Talwar, K., Zhang, L. (2016). Deep learning with differential privacy. The 23rd ACM SIGSAC conference on computer and communications security, Vienna, Austria, pp. 308-318. https://doi.org/10.1145/2976749.2978318
[57] Geyer, R.C., Klein, T., Nabi, M. (2019). Differentially private federated learning: A client level perspective. International Conference on Learning Representations, New Orleans, Louisiana, USA, pp. 1-9.
[58] Lecuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., Jana, S. (2019). Certified robustness to adversarial examples with differential privacy. The 40th Symposium on Security and Privacy, San Francisco, CA, USA, pp. 656-672. https://doi.org/10.1109/SP.2019.00044
[59] Canetti, R., Feige, U., Goldreich, O., Naor, M. (1996). Adaptively secure multi-party computation. ACM Symposium on Theory of computing, Pennsylvania, USA, pp. 639-648.
[60] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. The 24th ACM SIGSAC Conference on Computer and Communications Security, Dallas, Texas, USA, pp. 1175-1191. https://doi.org/10.1145/3133956.3133982
[61] Xu, G., Li, H., Liu, S., Yang, K., Lin, X. (2019). VerifyNet: Secure and verifiable federated learning. IEEE Transactions on Information Forensics and Security, 15: 911-926. https://doi.org/10.1109/tifs.2019.2929409
[62] Phong, L.T., Aono, Y., Hayashi, T., Wang, L., Moriai, S. (2018). Privacy-preserving deep learning via additively homomorphic encryption. IEEE Transactions on Information Forensics and Security, 13(5): 1333-1345. https://doi.org/10.1109/tifs.2017.2787987
[63] Hao, M., Li, H., Xu, G., Liu, S., Yang, H. (2019). Towards efficient and privacy-preserving federated deep learning. The 2019 IEEE International Conference on Communications, Shanghai, China, pp. 1-6. https://doi.org/10.1109/ICC.2019.8761267
[64] Hao, M., Li, H., Luo, X., Xu, G., Yang, H., Liu, S. (2020). Efficient and privacy-enhanced federated learning for industrial artificial intelligence. IEEE Transactions on Industrial Informatics, 16(10): 6532-6542. https://doi.org/10.1109/tii.2019.2945367
[65] Chen, Y., Su, L., Xu, J. (2017). Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. ACM on Measurement and Analysis of Computing Systems, 1(2): 1-25. https://doi.org/10.1145/3154503
[66] Cao, D., Chang, S., Lin, Z., Liu, G., Sun, D. (2019). Understanding distributed poisoning attack in federated learning. The 25th International Conference on Parallel and Distributed Systems, Tianjin, China, pp. 233-239. https://doi.org/10.1109/icpads47876.2019.00042
[67] Fung, C., Yoon, C.J., Beschastnikh, I. (2020). The limitations of federated learning in sybil settings. The 23rd International Symposium on Research in Attacks, Intrusions and Defenses, San Sebastian, pp. 301-316.
[68] Tran, D., Li, J., Mądry, A. (2018). Spectral signatures in backdoor attacks. The 32nd International Conference on Neural Information Processing Systems, Montréal, Canada, pp. 8011-8021. https://doi.org/10.48550/arXiv.1811.00636
[69] Pillutla, K., Kakade, S.M., Harchaoui, Z. (2019). Robust

aggregation for federated learning. arXiv preprint arXiv:1912.13445. https://doi.org/10.48550/arXiv.1912.13445

[70] Grama, M., Musat, M., Muñoz-González, L., Passerat-Palmbach, J., Rueckert, D., Alansary, A. (2020). Robust aggregation for adaptive privacy preserving federated learning in healthcare. arXiv preprint arXiv:2009.08294. https://doi.org/10.48550/arXiv.2009.08294

[71] Lu, Y., Fan, L. (2020). An efficient and robust aggregation algorithm for learning federated CNN. The 3rd International Conference on Signal Processing and Machine Learning, Beijing, China, pp. 1-7.

[72] Ang, F., Chen, L., Zhao, N., Chen, Y., Wang, W., Yu, F.R. (2020). Robust federated learning with noisy communication. IEEE Transactions on Communications, 68(6): 3452-3464. https://doi.org/10.1109/tcomm.2020.2979149

[73] Seo, H., Park, J., Oh, S., Bennis, M., Kim, S.L. (2020). Federated knowledge distillation. arXiv preprint arXiv:2011.02367.

[74] Jiang, D., Shan, C., Zhang, Z. (2020). Federated learning algorithm based on knowledge distillation. International Conference on Artificial Intelligence and Computer Engineering, Beijing, China, pp. 163-167. https://doi.org/10.1109/icaice51518.2020.00038

[75] Zhu, Z., Hong, J., Zhou, J. (2021). Data-free knowledge distillation for heterogeneous federated learning. The 38th International Conference on Machine Learning, pp. 12878-12889. https://doi.org/10.48550/arXiv.2105.10056

[76] Li, D., Wang, J. (2019). FedMD: Heterogenous federated learning via model distillation. International Workshop on Federated Learning for User Privacy and Data Confidentiality, Vancouver, Canada, pp. 1-8. https://doi.org/10.48550/arXiv.1910.03581

[77] Babu, B.V.S., Babu K.S. (2021). The purview of blockchain appositeness in computing paradigms: A survey. Ingénierie des Systèmes d'Information, 26(1): 33-46. https://doi.org/10.18280/isi.260104

[78] Nguyen, D.C., Ding, M., Pham, Q.V., Pathirana, P.N., Le, L.B., Seneviratne, A., Li, J., Niyato, D., Poor, H.V. (2021). Federated learning meets blockchain in edge computing: opportunities and challenges. IEEE Internet of Things Journal, 8(16): 12806-12825. https://doi.org/10.1109/jiot.2021.3072611