Privacy and Robustness in Federated Learning: Attacks and Defenses

Lingjuan Lyu, *Member, IEEE*, Han Yu, *Senior Member, IEEE*, Xingjun Ma, Chen Chen, Lichao Sun, Jun Zhao, *Member, IEEE*, Qiang Yang, *Fellow, IEEE*, and Philip S. Yu, *Life Fellow, IEEE*

***Abstract*— As data are increasingly being stored in different silos and societies becoming more aware of data privacy issues, the traditional centralized training of artificial intelligence (AI) models is facing efficiency and privacy challenges. Recently, federated learning (FL) has emerged as an alternative solution and continues to thrive in this new reality. Existing FL protocol designs have been shown to be vulnerable to adversaries within or outside of the system, compromising data privacy and system robustness. Besides training powerful global models, it is of paramount importance to design FL systems that have privacy guarantees and are resistant to different types of adversaries. In this article, we conduct a comprehensive survey on privacy and robustness in FL over the past five years. Through a concise introduction to the concept of FL and a unique taxonomy covering: 1) threat models; 2) privacy attacks and defenses; and 3) poisoning attacks and defenses, we provide an accessible review of this important topic. We highlight the intuitions, key techniques, and fundamental assumptions adopted by various attacks and defenses. Finally, we discuss promising future research directions toward robust and privacy-preserving FL, and their interplays with the multidisciplinary goals of FL.**

***Index Terms*— Attacks, defenses, federated learning (FL), privacy, robustness.**

NOMENCLATURE

AI	Artificial intelligence.
ML	Machine learning.
FL	Federated learning.
GDPR	General data protection regulation.
i.i.d.	Independent identically distributed.
IoT	Internet of Things.
HFL	Horizontally federated learning.
VFL	Vertically federated learning.
FTL	Federated transfer learning.
H2B	HFL to businesses.
H2C	HFL to consumers.
SGD	Stochastic gradient descent.
SMC	Secure multiparty computation.
DP	Differential privacy.
CDP	Centralized differential privacy.
LDP	Local differential privacy.
DDP	Distributed differential privacy.
HE	Homomorphic encryption.
RFA	Robust federated aggregation.
GAN	Generative adversarial network.
MIA	Membership inference attack.
AT	Adversarial training.
FAT	Federated adversarial training.
API	Application programming interface.

I. INTRODUCTION

AS COMPUTING devices become increasingly ubiquitous, people generate huge amounts of data through their day-to-day usage. Collecting such data into centralized storage facilities is costly and time-consuming. Traditional centralized ML approaches cannot support such ubiquitous deployments and applications due to infrastructure shortcomings, such as limited communication bandwidth, intermittent network connectivity, and strict delay constraints [1]. Another critical concern is data privacy and user confidentiality as the usage data usually contain sensitive information [2]. Sensitive data, such as facial images, location-based services, or health

Manuscript received 13 January 2022; revised 11 August 2022; accepted 14 October 2022. This work was supported in part by Sony AI; in part by the Joint NTU-WeBank Research Centre on Fintech under Award NWJ-2020-008; in part by Nanyang Technological University, Singapore; in part by the Joint SDU-NTU Centre for Artificial Intelligence Research (C-FAIR) under Grant NSC-2019-011; in part by the National Research Foundation, Singapore, under its AI Singapore Programme, under AISG Award AISG2-RP-2020-019; in part by the RIE 2020 Advanced Manufacturing and Engineering (AME) Programmatic Fund, Singapore, under Grant A20G8b0102; in part by Nanyang Technological University through the Nanyang Assistant Professorship (NAP); and in part by the Future Communications Research & Development Programme under Grant FCP-NTU-RG-2021-014. The work of Qiang Yang was supported in part by the Hong Kong RGC Theme-Based Research Scheme under Grant T41-603/20-R. The work of Philip S. Yu was supported in part by NSF under Grant III-1763325, Grant III-1909323, Grant III-2106758, and Grant SaTC-1930941. *(Corresponding authors: Lingjuan Lyu; Han Yu; and Qiang Yang.)*

Lingjuan Lyu is with Sony AI, Tokyo 108-0075, Japan (e-mail: lingjuan.lv@sony.com).

Han Yu and Jun Zhao are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: han.yu@ntu.edu.sg; junzhao@ntu.edu.sg).

Xingjun Ma is with the School of Computer Science, Fudan University, Shanghai 200437, China (e-mail: xingjunma@fudan.edu.cn).

Chen Chen was with Sony AI, Tokyo 108-0075, Japan. He is now with the College of Computer Science, Zhejiang University, Hangzhou 310027, China (e-mail: cc33@zju.edu.cn).

Lichao Sun is with the Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA 18015 USA (e-mail: lis221@lehigh.edu).

Qiang Yang is with the Department of Artificial Intelligence (AI), WeBank, Shenzhen 518000, China, and also with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong (e-mail: qyang@ust.hk).

Philip S. Yu is with the Department of Computer Science, University of Illinois Chicago, Chicago, IL 60607 USA (e-mail: psyu@uic.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3216981.

Digital Object Identifier 10.1109/TNNLS.2022.3216981

information, can be used for targeted social advertising and recommendation, posing immediate or potential privacy risks. Hence, private data should not be directly shared without any privacy protection. As societies become increasingly aware of privacy preservation, legal restrictions, such as the GDPR, are emerging, which makes data aggregation practices less feasible [3].

In this scenario, FL (also well known as collaborative learning), which distributes model training to the devices from which data originate, emerged as a promising alternative ML paradigm [4]. FL enables a multitude of participants to construct a joint ML model without exposing their private training data [4], [5]. It can also handle unbalanced and non-i.i.d. data, which naturally arise in the real world [6]. In recent years, FL has benefited a wide range of applications such as next word prediction [6], [7], visual object detection for safety [8], entity resolution [9], recommendation [10], [11], [12], industrial IoT [13], unmanned aerial vehicles (UAVs) [14], and graph-based analysis [15], [16], [17], [18].

A. Categorization of FL Based on Distribution

Based on the distribution of data features and data samples among participants, FL can be generally classified as HFL, VFL, and FTL [19].

Under HFL, datasets owned by each participant share similar features but concern different users [20]. For example, several hospitals may each store similar types of data (e.g., demographic, clinical, and genomics) about different patients. If they decide to build an ML model together using FL, we refer to such a scenario as HFL. In this article, we further classify HFL into H2B and H2C. The main difference between H2B and H2C lies in the number of participants, FL training participation level, and technical capability, which can influence how adversaries attempt to compromise the FL system. Under H2B, there are typically a handful of participants. They can be frequently selected during FL training. The participants tend to possess significant computational power and sophisticated technical capabilities [3]. Under H2C, there can be thousands or even millions of potential participants. In each round of training, only a subset of them is selected. As their datasets tend to be small, the chance of a participant being selected repeatedly for FL training is low. They generally possess limited computational power and low technical capabilities. An example of H2C is Google's GBoard application [7].

VFL is applicable to cases in which participants have large overlaps in the sample space but differ in the feature space, i.e., different participants hold different attributes of the same records [21]. VFL mainly targets business participants. Thus, the characteristics of VFL participants are similar to those of H2B participants.

Nowadays, FTL is attracting increasing attention in industries such as finance, medicine, and healthcare. FTL deals with scenarios in which FL participants have little overlap in both the sample space and the feature space [3], [19]. In this case, transfer learning [22] techniques can be applied to provide solutions for the entire sample and feature space under a federation. FTL enables complementary knowledge to be transferred across domains in a data federation, thereby enabling a target-domain party to build flexible and effective models by leveraging rich labels from a source domain [23].

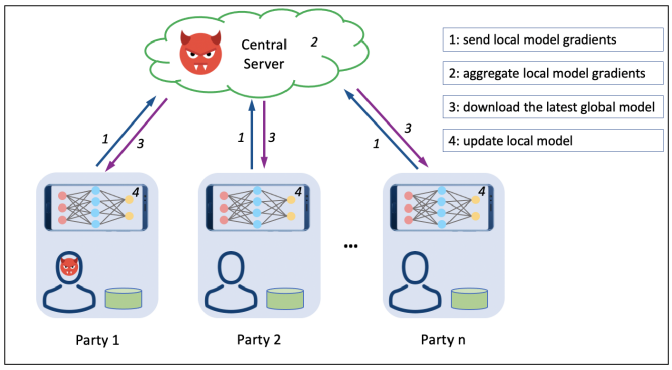


Fig. 1. Typical FL training process, in which both the (potentially malicious) FL server/aggregator and malicious participants may pose threats to the FL system.

B. Categorization of FL Based on Architectures

1) FL With Homogeneous Architectures: Sharing gradients is typically limited only to homogeneous FL architectures, i.e., the same model is shared with all participants. Participants aim to collaboratively learn a more accurate model. Specifically, model parameters w are often obtained via solving the following optimization problem: $\min_w \sum_{i=1}^{n} F(w, D_i)$, where $F(w, D_i)$ is the objective function for the local training dataset on the ith participant and characterizes how well the parameters w model the local training dataset D_i on the ith participant. Different classifiers (e.g., logistic regression and deep neural networks) use different objective functions. In FL, each participant maintains a local model for its local training dataset. The server maintains a global model via aggregating local models from n participants. Specifically, FL with homogeneous architectures performs the steps in Fig. 1. Generally, FL with homogeneous architectures comes in two forms [6]: 1) FedSGD, in which each participant sends every SGD update to the server and 2) FedAvg, in which participants locally batch multiple iterations of SGD before sending updates to the server, which is more communication efficient. These methods are all based on the mean aggregation rule that takes the average of the local model parameters as the global model.

2) FL With Heterogeneous Architectures: The most recent efforts extended FL to collaboratively train models with heterogeneous architectures [24], [25], [26]. Conventional federated model training that directly averages model weights is only possible if all local models have the same model structure. Naturally, it limits collaboration among data owners with heterogeneous model architectures. One representative work called *federated model distillation* (FedMD) [25] does not force a single global model onto local models. Instead, it is conducted in a succinct, black box, and model-agnostic manner. Each local model is updated separately; participants share the knowledge of their local models via their predictions

TABLE I
SUMMARY OF ATTACKS AGAINST SERVER-BASED FL

Attack Type		Attack Target		Attacker Role		FL Scenario		Attack Complexity		
		Model	Training Data	Participant	Server	H2B	H2C	Attack Iteration		Auxiliary Knowledge Required
								One Round	Multiple Rounds	
Robustness	Untargeted attack	YES	NO	YES	NO	YES	NO	YES	YES	YES
	Targeted attack	YES	NO	YES	NO	YES	NO	YES	YES	YES
Privacy	Infer Class Representatives	NO	YES	YES	YES	YES	NO	NO	YES	YES
	Infer Membership	NO	YES	YES	YES	YES	NO	NO	YES	YES
	Infer Properties	NO	YES	YES	YES	YES	NO	NO	YES	YES
	Infer Training Inputs and Labels	NO	YES	NO	YES	YES	NO	YES	YES	NO

on an unlabeled public set [25]. One obvious benefit of sharing logits is the reduced communication costs, without significantly affecting utility [25].

In summary, all the above sharing methods did not inherently provide defenses against privacy and poisoning attacks—two main sources of threats to FL.

C. Threats to FL

FL offers a privacy-aware paradigm of model training, which does not require data sharing and allows participants to join and leave a federation freely. Nevertheless, recent works have demonstrated that FL may not always provide sufficient privacy and robustness guarantees. Existing FL protocol designs are vulnerable to: 1) a malicious server that aims to infer sensitive information from individual updates over time, tamper with the training process, or control the view of the participants on the global parameters and 2) any adversarial participant who can infer other participants' sensitive information, tamper the global parameter aggregation, or poison the global model.

In terms of privacy, communicating gradients throughout the training process can reveal sensitive information [27], [28] and even cause deep leakage [29], either to a third party or the central server [7], [30]. Even a small portion of gradients can reveal a fair amount of sensitive information about the local data [31]. Recent works further show that, by simply observing the gradients, a malicious attacker can successfully steal the training data [29], [32].

In terms of robustness, FL systems are vulnerable to both data poisoning [33], [34] and model poisoning attacks [35], [36], [37], [38]. Malicious participants can attack the convergence of the global model or implant backdoor triggers into the global model by deliberately altering their local data (data poisoning) or their gradient uploads (model poisoning). More broadly, poisoning attacks can be categorized into: 1) untargeted attack, such as a Byzantine attack, where the adversary aims to destroy the convergence and performance of the global model [39], [40] and 2) targeted attack, such as a backdoor attack, where the adversary aims to implant a backdoor trigger into the global model, so as to trick the model to constantly predict an adversarial class on a subtask while keeping good performance on the main task [34], [35], [36].

These privacy and robustness attacks pose significant threats to FL. In centralized learning, the server is responsible for all the participants' privacy and model robustness. However, in FL, any participant can attack the server and spy on other participants, even without involving the server. Therefore, it is important to understand the principles behind these privacy and robustness attacks. The properties of the representative privacy and robustness attacks in server-based FL are summarized in Table I.

Note that all the above threats are mainly for homogeneous FL. Although heterogeneous FL is more privacy friendly compared to homogeneous FL, as sharing model prediction instead of model parameters or updates eliminates the risk of white-box inference attacks in homogeneous FL [25], [41], there is no theoretic guarantee that sharing prediction is private and secure. In fact, the predictions from local models also encode some private information [42], [43], [44], [45]. Similarly, local model predictions can also be arbitrarily manipulated by any malicious participant. However, how it is vulnerable to different privacy and poisoning attacks, as listed in Sections III and IV, remains largely unknown, which needs further investigation. Henceforth, we are mainly focusing on the homogeneous FL throughout this survey.

D. Secure FL

Attacks on FL come from either the privacy perspective when a malicious participant or the central server attempts to infer the private information of a victim participant or the robustness perspective when a malicious participant aims to compromise the global model.

To secure FL against privacy attacks, existing privacy-preserving methodologies in centralized ML have been tried in FL, including HE, SMC, and DP. However, HE and SMC may not be applicable to large-scale FL, as they incur substantial communication and computation overhead. In aggregation-based tasks, DP requires the aggregated value to contain random noise up to a certain magnitude to ensure (ϵ, δ)-DP and, thus, is also not ideal for FL. The noise addition required by DP is also hard to execute in FL. In an ideal scenario where the server (aggregator) is trusted, the server can add the noise to the aggregated gradients [7]. However, in many real-world scenarios, the participants may not trust the central server or each other. In this case, the participants would compete with each other, and all want to ensure LDP by adding as much noise as possible to their local gradients [30], [44], [46]. This tends to accumulate significant errors on the server side. DDP [30], [44], [46] can mitigate this problem to some extent when at least a certain fraction of the participants are honest and do not conduct such malicious competition.

Defending FL against various robustness attacks (e.g., untargeted Byzantine attack and targeted backdoor attack) is an extremely challenging task. This is due to two main

reasons. First, the defense can only be executed on the server side where only local gradients are available. This invalids many backdoor defense methods developed in the centralized ML, for example, denoising (preprocessing) methods [47], [48], [49], [50], [51], backdoor sample/trigger detection methods [52], [53], [54], [55], [56], [57], robust data augmentations [58], fine-tuning methods [58], the neural attention distillation (NAD)-based method [59], and more recent anti-backdoor learning (ABL) method based on a sophisticated learning process [60]. Second, the defense method has to be robust to both data poisoning and model poisoning attacks. Most existing robustness defenses are gradient aggregation methods mainly developed for defending against the untargeted Byzantine attackers, such as Krum/multi-Krum [40], AGGREGATHOR [61], Byzantine gradient descent (BGD) [62], median-based gradient descent [63], trimmed-mean-based gradient descent [63], and SIGNSGD [39]. These defense methods have never been tested on the targeted backdoor attacks [33], [34], [35], [36], [38]. Dedicated defense methods against both data poisoning and model poisoning attacks have been investigated, such as norm clipping [38], geometric median-based RFA [64], and robust learning rate [65]. For the collusion of Sybil attacks, contribution similarity [37] can be leveraged as a strategy for defense.

E. Motivation of This Survey and Our Contribution

Most existing surveys on FL are mostly focused on the system or protocol design [19], [66], [67]. A few surveys touched on either privacy or robustness, but did not systematically explore both, and their intersections with the other aspects in FL, such as fairness and efficiency [68], [69], [70]. A notable number of research works have been conducted on privacy and robustness. Although these works attempt to discover the vulnerabilities of FL and aim to enhance the privacy and system robustness of FL, there are very few efforts for categorizing them in a systemic manner, and privacy and robustness threats to FL have not been systematically explored. To fill in this gap, in this article, we have conducted an extensive survey on the recent advances in privacy and robustness threats to FL and their defenses. In particular, we focus on two specific threats initiated by insiders in FL systems: 1) privacy attacks that attempt to infer the victim participants' private information and 2) poisoning attacks that attempt to prevent the learning of a global model or implant triggers to control the behavior of the global model. This article mainly surveys the literature over the past five years on privacy and robustness in FL; it can be a notable inclusion to the existing literature, helping the community better understand the state-of-the-art privacy and robustness progress in FL. The limitations and the promising use cases of the existing works in the literature and open directions for future research are also offered to identify the research gaps to address the challenges of privacy and robustness in FL.

For empirical and use case analysis of privacy and robustness, interesting readers can refer to [71], [72], and [73], which showcased where and how the attacks and defense stand so far in FL. For example, Nasr et al. [71] provided a comprehensive privacy analysis of FL under both passive and active white-box inference attacks; Huang et al. [72] did a comprehensive evaluation of defenses against gradient inversion attacks in FL, including encrypting gradients, perturbing gradients, encoding inputs, and combined defenses; and Shejwalkar et al. [73] clearly showed that FL, even without any defenses, is highly robust in practice. For production cross-device FL (H2C), which contains thousands to billions of clients, poisoning attacks have no impact on existing robust FL algorithms even with impractically high percentages of compromised clients. For production cross-silo FL (H2B), which contains up to 100 clients, data poisoning attacks are completely ineffective; model poisoning attacks are unlikely to play a major risk when the clients involved are bound by contract and their software stacks are professionally maintained (e.g., in banks and hospitals).

Overall, the major contributions of this survey include the following.

1) This survey presents a comprehensive categorization of FL, and summarized threats and the corresponding protections for FL in a systematic manner.
2) Existing privacy and robustness attacks and defenses are well explored to help readers better understand the assumptions, principles, reasons, and differences of the current progress in the privacy and robustness domain of FL.
3) The conflicts between privacy and robustness, and among multiple design goals are identified; the gaps between the current works and the real scenarios in FL are summarized.
4) Future research directions will assist the community to rethink and improve their current designs toward robust and privacy-preserving FL of real practicality and impact. Meanwhile, it is suggested to integrate multidisciplinary goals in the system design of FL.

F. Survey Organization

The rest of the survey is organized as follows. Before going into an in-depth discussion on privacy and robustness in FL, in Section II, we first summarize the threat models from a general perspective and discuss the customized threat models for privacy and robustness, respectively. Section III presents a comprehensive review of the privacy attacks in FL, particularly targeting the sensitive information (class representative, membership, properties, training inputs, and labels) in HFL with homogeneous architectures. Section IV shows the detailed poisoning attacks that aim to compromise system robustness, including the untargeted and targeted poisoning attacks. Sections V and VI list the most representative privacy-preserving techniques and defense mechanisms for robustness, and current practices that have applied these techniques in FL. From the lessons learned in this survey paper, the research gaps toward realizing trustworthy FL along with directions for future research are provided in Section VII. Finally, concluding remarks are drawn in Section VIII.

For better readability, we give a diagram in Fig. 2 showing the different aspects covered in the survey. The list of

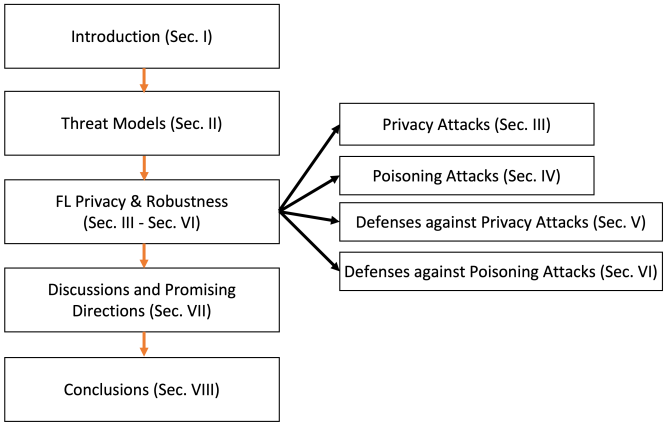


Fig. 2. Survey organization.

abbreviations used in this survey is provided in Nomenclature. Throughout this survey, we will interchangeably use participants/clients/users to represent the participants in FL.

II. Threat Models

Before reviewing attacks on FL, we first present a summary of the threat models. Generally, we categorize the threat models in FL into the following types: 1) insider versus outsider according to the role of the attacker; 2) semihonest versus malicious according to whether the attacker will obey the protocol; and 3) training phase versus test/inference phase according to the happening phase of the attack. These threat models apply to both privacy and robustness attacks.

A. Insider Versus Outsider

1) Insider Attacks: Insider attacks can be launched by either the FL server or the participants in the FL system. More concretely, the central server can observe individual updates over time and can control the view of the participants on the global parameters; any of the participants can observe the global parameter updates and can control his or her parameter uploads [71].

2) Outsider Attacks: Outsider attacks include those launched by the eavesdroppers on the communication channel between participants and the FL server, and by users of the final FL model when it is deployed as a service.

Insider attacks are generally more dangerous than outsider attacks, as it strictly enhances the capability of the adversary. Thus, our discussion of attacks against FL will focus primarily on insider attacks.

B. Semihonest Versus Malicious

1) Semihonest Setting: Adversaries are considered passive or honest-but-curious. They try to learn the private information of other participants without deviating from the FL protocol. The adversaries can only observe the received information, i.e., parameters of the global model.

2) Malicious Setting: An active or malicious adversary tries to learn the private information of the other honest participants and deviates arbitrarily from the FL protocol by modifying, replaying, or removing messages. This setting allows the adversary to conduct particularly devastating attacks.

Note that both the honest and semihonest participants will follow the protocol honestly; however, semihonest may attempt to learn or infer sensitive information from the information he or she received [43], [67]. By contrast, the malicious participant will arbitrarily betray the protocol in order to compromise the privacy of other participants or compromise the integrity of the FL model in either a targeted or untargeted manner, as indicated in Sections III and IV.

C. Training Phase Versus Inference Phase

1) Training Phase: During the training phase, the insider attacker has access to the full model, notably its architecture and parameters, and any hyperparameter that is needed to use the model for predictions [71]. Attacks conducted during the training phase attempt to learn, influence, or corrupt the FL model itself [74].

In terms of the privacy vulnerability during the training phase, the attacker can also launch a range of inference attacks on an individual participant's shared information or the aggregated information in order to compromise privacy [28], [29], [71], [75]. For example, from the shared local model parameter, the insider attacker (participant or the server) can infer class representatives [75], record membership, and properties [28], [71], even the original training inputs and labels [29]. More details can be referred to Section III.

In terms of the robustness vulnerability during the training phase, the attacker can run data poisoning attacks to compromise the integrity of the training dataset [35], [37], [76], [77], [78], [79] or model poisoning attacks to compromise the integrity of the learning process [36], [80]. The concrete attacks could be the targeted label-flipping attack [37], [81] and backdoor attack [35], or the untargeted Byzantine attack [40], [63]. More details can be referred to Section IV.

2) Test/Inference Phase: In this phase, attackers do not alter the targeted model; instead, they will query the targeted model to leak some private information or trick the targeted model to compromise robustness by producing wrong predictions. The effectiveness of such attacks is largely determined by the information that is available to the adversary about the model.

In terms of privacy vulnerability during the inference phase, the trained global model may reveal sensitive information from model predictions when deployed as a service, causing privacy leakage. In such a setting, an attacker does not have direct access to the model parameters but may be able to view input–output pairs, thus launching model stealing attacks in which model parameters can be reconstructed [82], [83], [84], [85] or MIAs that aim to determine if a particular record was used to train the model [71], [86].

In terms of the robustness vulnerability during the inference phase, the global model maintained by the server suffers from the same evasion attacks [87] as in the conventional ML setting when the target model is deployed as a service. One well-studied form of evasion attacks is the so-called adversarial examples, which seem almost indistinguishable from the original test input to a human, but can fool the

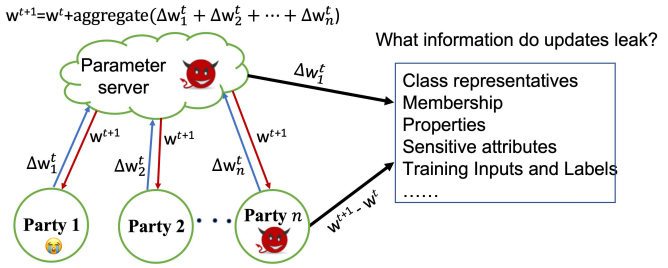


Fig. 3. Demo of privacy leakage in FL. An attacker can infer various private information about the victim participant from the received gradients or the snapshot of the FL model parameters.

trained model [88], [89]. Recent studies [90], [91], [92] have shown that FL is also vulnerable to well-crafted adversarial examples during the inference phase. One typical defense against robustness attacks is AT, in which a robust model is trained with adversarial examples and, hence, provides some robustness to white-box evasion attacks [93]. However, adapting AT methods to FL brings a host of open questions, which we discuss in Section VII.

III. Privacy Attacks

Although FL prevents the participants from directly sharing their private data, a series of works have demonstrated that exchanging gradients in FL can also leak sensitive information about the participants' private data to either passive or active attackers [28], [29], [31], [71], [94]. For example, gradients or two consecutive snapshots of the FL model parameters can leak unintended features of the participants' training data to the adversarial participants, as deep learning models tend to recognize and remember more features of the data than needed for the main learning task [95]. Fig. 3 illustrates that the set of information an adversary can infer from the gradients (i.e., $\Delta \boldsymbol{w}_1^t$) or, equivalently, the difference of two successive snapshots of the model parameters (i.e., $\boldsymbol{w}^{t+1} - \boldsymbol{w}^t$).

The reason why gradients can cause privacy leakage is that the gradients are derived from the participants' private training data, and a learning model can be considered as a representation of the high-level statistics of the dataset that it was trained on [43]. In deep learning models, gradients of a given layer are computed based on the layer's features and the error from the layer after (i.e., backpropagation). In the case of sequential fully connected layers, the gradients of the weights are the inner products of the current layer's features and the error from the layer after. Similarly, for a convolutional layer, the gradients of the weights are convolutions of the layer's features and the error from the layer after [28]. Consequently, observations of gradients can be used to infer a significant amount of private information, such as class representatives, membership, and properties of a subset of the training data. Even worse, an attacker can infer labels from the shared gradients and recover the original training samples without any prior knowledge about the training data [29]. Next, we detail the potential privacy leakage of FL according to the type of sensitive information that the attacker is targeting.

A. Inferring Class Representatives

Hitaj et al. [75] first devised an active inference attack called GAN attack against deep FL models. In this attack, a malicious participant can intentionally compromise any other participant. The GAN attack exploits the real-time nature of the FL learning process, which allows the adversarial participant to train a GAN to generate prototypical samples of the targeted private training data. The generated samples appear to come from the same distribution as the training data. Hence, the GAN attack is not targeted to reconstruct the exact training inputs but only the class representatives. It should be noted that the GAN attack assumes the entire training corpus for a given class comes from a single participant, which means that the GAN-constructed representatives are similar to the training data only when all class members are similar. This resembles model inversion attacks in the centralized ML settings [96]. Note that these assumptions are less practical in FL. Since the GAN attack requires a substantial amount of computational resources to train the GAN model, it is less suitable for H2C scenarios.

B. Inferring Membership

Given an exact data point, MIAs aim to determine if it was used to train the model [86]. For example, an attacker can infer whether a specific patient profile was used to train a classifier associated with a certain disease. FL opens new possibilities for such attacks. In FL, the adversary can infer if a particular sample belongs to the private training data of a particular participant (if the target update is from a single participant) or any participant (if the target update is the aggregate). For example, during FL model training, the nonzero gradients of the embedding layer of a deep natural language processing model trained on text data can reveal which words are in the training batches of the honest participants [28], [97].

Attackers in an FL system can conduct both active and passive MIAs [28], [71]. In the passive case, the attacker observes the updated model parameters and performs inference without modifying the learning process. In the active case, the attacker can tamper with the FL model training protocol and perform a more powerful attack against other participants. For instance, the attacker may share malicious updates and trick the FL model to expose more information about other participants' local data. One such attack is the gradient ascent attack [71], where the attacker runs gradient ascent on a target data sample and observes whether its increased loss can be drastically reduced in the next communication round; if so, the sample is very likely to be in the training set. This attack can be applied on a batch of target data samples all at the same time [71].

C. Inferring Properties

An adversary can launch both passive and active property inference attacks to infer certain properties of other participants' training data [28]. Property inference attacks assume that the adversary has auxiliary training data that are correctly labeled with the target property. A passive adversary can only

observe or eavesdrop on the gradients and perform inference by training a binary property classifier. An active adversary can exploit multitask learning to trick the FL model into learning a better separation between data with and without the target property, so as to extract more information. An adversarial participant can also infer when a property appears or disappears in the training data (e.g., identifying when a person first appears in the photographs used to train a gender classifier). The assumption of auxiliary training data in property inference attacks may limit its applicability in H2C.

D. Inferring Training Inputs and Labels

One recent work called *deep leakage from gradient* (DLG) proposes an optimization algorithm to extract both the training inputs and the labels [29]. This attack is much stronger than previous approaches. It can accurately recover the raw images and texts used to train a deep learning model. In a follow-up work [32], an analytical approach called *improved DLG* (iDLG) was proposed to extract labels based on the shared gradients and an exploration of the correlation between the labels and the signs of the gradients. iDLG can be applied to attack any differentiable models trained with cross-entropy loss and one-hot labels, which is a typical setting for classification tasks.

In summary, inference attacks generally assume that the adversaries possess sophisticated technical capabilities and unlimited computational resources. Moreover, most attacks assume that the adversarial participants can be selected (to update the global model) in many rounds of the FL training process. In FL, these assumptions are generally not practical in H2C scenarios but more likely to happen in H2B scenarios. These inference attacks highlight the need for gradient protection in FL, possibly through various privacy-preserving mechanisms [3] detailed in Section V.

IV. Poisoning Attacks

Different from privacy attacks that are targeting at data privacy, poisoning attacks aim to compromise the system's robustness. Depending on the attacker's objective, poisoning attacks can be broadly classified into two categories: 1) untargeted poisoning attacks [80], [81], [98], [99], [100] and 2) targeted poisoning attacks [35], [36], [101], [102], [103], [104], [105], [106].

Note that the untargeted and targeted poisoning attacks during the training phase can be mounted on both the data and the model. Fig. 4 shows that the poisoned updates can be sourced from two poisoning attacks: 1) data poisoning attack during local data collection and 2) model poisoning attack during local model training process. At a high level, both poisoning attacks attempt to modify the behavior of the target model in some undesirable way. However, due to the model-sharing nature of FL with homogeneous architectures, data poisoning attacks are generally less effective than model poisoning attacks [35], [36], [37], [38]. In fact, model poisoning subsumes data poisoning in FL settings, as data poisoning attacks eventually change a subset of updates sent to the model at any given iteration. This is functionally identical to a centralized poisoning attack in which a subset of the training data is poisoned.

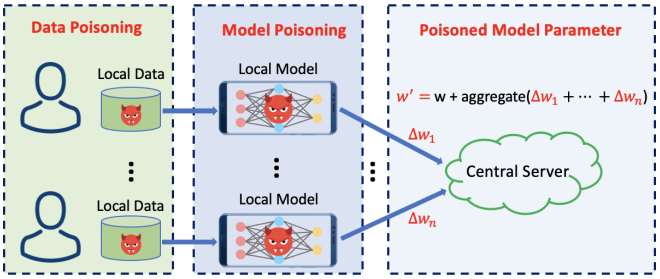


Fig. 4. Data versus model poisoning attacks in FL.

A. Untargeted Attacks

Untargeted poisoning attacks aim to arbitrarily compromise the integrity of the target model. The Byzantine attack is one type of *untargeted* poisoning attack that upload arbitrarily malicious gradients to the server so as to cause the failure of the global model [39], [40], [61], [100], [107]. A formal definition of Byzantine attack is given in Definition 1.

Definition 1 (Byzantine Attack [40], [63]): An honest participant uploads $\Delta \boldsymbol{w}_i \coloneqq \nabla F_i(\boldsymbol{w}_i)$, while a dishonest participant can upload arbitrary values

$$\Delta \boldsymbol{w}_i = \begin{cases} *, & \text{if } i\text{-th participant is Byzantine} \\ \nabla F_i(\boldsymbol{w}_i), & \text{otherwise} \end{cases} \tag{1}$$

where "$*$" represents arbitrary values and F_i represents participant i's local model objective function.

Blanchard et al. [40] showed that the aggregation of FL can be completely controlled by a single Byzantine participant if there is no defense in the FL. In particular, suppose that there are $n-1$ benign participants and a Byzantine participant; the server aggregates the gradients by $\Delta \boldsymbol{w}' = (1/n)\sum_{i=1}^{n} \Delta \boldsymbol{w}_i$, where $\Delta \boldsymbol{w}'$ is the aggregated gradient. Assume the nth participant is Byzantine; it can always make the aggregated gradient become any vector $\boldsymbol{u}$ by uploading the following gradient:

$$\Delta \boldsymbol{w}_n = n\boldsymbol{u} - \sum_{i=1}^{n-1} \Delta \boldsymbol{w}_i. \tag{2}$$

Such a simple attack exposes the vulnerability of FL against Byzantine attacks.

Chen et al. [108] discussed Byzantine attacks in Adam-based FL and proposed a camouflaged attack that can camouflage the model updates and launch effective attacks. Their proposed attack also works on other well-known optimizers, such as AdaGrad and RMSProp. Baruch et al. [109] showed that the core part of gradient descent algorithms is the direction of the descent. Specifically, for gradient descent algorithms, to guarantee the descent of the loss, the inner product between the ground-truth gradient and the robust aggregated gradient must be nonnegative

$$\langle \Delta \boldsymbol{w}, \Delta \boldsymbol{w}' \rangle \geq 0 \tag{3}$$

where $\langle \cdot, \cdot \rangle$ is the inner product operation, $\Delta \boldsymbol{w}$ is the optimal gradient, and $\Delta \boldsymbol{w}' = \text{aggregate}(\Delta \boldsymbol{w}_1, \ldots, \Delta \boldsymbol{w}_n)$ is the aggregated gradient with aggregate$(\cdot)$ being arbitrary aggregation

function. To make the aggregation fail, they proposed an "inner product manipulation attack" that can make the inner product between the ground-truth gradient and the robust aggregated gradient negative. To do this, each Byzantine participant uploaded the negative of the average benign gradients. Their proposed attack can successfully bypass coordinatewise median [63] and Krum [40]. Xie et al. [100] claimed that, by consistently applying small changes to many parameters, a Byzantine participant can perturb the model's convergence. First, they used the local data of Byzantine participants to estimate the mean and standard deviation of the distribution. Then, they analyzed the range in which changes to the parameters will not be detected by the defense, and upon choosing the maxima of this range, the convergence is averted.

B. Targeted Attacks

In targeted poisoning attacks, the learned model outputs the target label specified by the adversary for particular testing examples, e.g., predicting spams as nonspam and predicting attacker-desired labels for testing examples with a particular Trojan trigger (backdoor/Trojan attacks). However, the test error for other testing examples is unaffected. Generally, targeted attacks are more difficult to conduct than untargeted attacks as the attacker has a specific goal to achieve.

1) Label-Flipping Attack: One common example of targeted poisoning attack is the label-flipping attack [37], [81]. The labels of honest training examples of one class are flipped to another class, while the features of the data are kept unchanged. For example, the malicious participants in the system can poison their local data by flipping all 1s into 7s. A successful attack produces a model that is unable to correctly classify 1s and incorrectly predicts them to be 7s.

2) Backdoor Attack: Another realistic targeted poisoning attack is a backdoor attack, in which an adversary can modify individual features or small regions of the original training dataset to implant a backdoor trigger into the model. The model will behave normally on clean data yet will constantly predict a target class whenever the trigger (e.g., a stamp on an image) appears. For instance, a backdoor attack can cause the FL model to reach 100% accuracy on the backdoor task, e.g., to control an image classifier to assign an attacker-chosen label to images with certain features in an image-classification task, or a next-word predictor completes certain sentences with an attacker-chosen word in a word-prediction task [35].

Backdoor attacks can be further divided into two categories: dirty-label attacks [58], [103], [104], [110] and clean-label attacks [58], [79], [105], [111], [112], [113], [114]. Clean-label attacks assume that the adversary cannot change the label of any training data as there is a process by which data are certified as belonging to the correct class, and the poisoning of data samples has to be imperceptible. In contrast, in dirty-label poisoning, the adversary can introduce a number of data samples that are expected to be misclassified by the model with the desired target label into the training data. Clean-label attacks are arguably stealthier as they do not change the labels.

3) How to Poison: The targeted poisoning attack in FL can be carried out by any FL participant or via collusion on either the data or the gradients. Bhagoji et al. [36] demonstrated that a single, noncolluding malicious participant can cause the model to misclassify a set of chosen inputs with high confidence. Bagdasaryan et al. [35] pointed out that the poisoned updates can be generated by training the local model on backdoored local training data, and even a single-shot attack may be enough to inject a backdoor into the global model. Xie et al. [34] demonstrated that a global trigger pattern can be decomposed into separate local patterns and embedded into the training set of colluding adversarial participants, respectively. The impact on the FL model depends on the extent to which the backdoor participants engage in the attacks and the amount of training data being poisoned. Recent work shows that poisoning edge-case (low probability) training samples are more effective [33].

4) Remark: We remark that most of the previous research on poisoning attacks focus on Byzantine or backdoor attackers. A system that allows participants to join and leave is susceptible to Sybil attacks [115], in which an attacker gains influence by joining a system to inject c fake participants into the FL system or compromise c benign participants [37]. Sybil attacks can be launched in both the untargeted and targeted manners. For example, targeted poisoning can be conducted by Sybil clones who contribute updates toward a specific poisoning objective [37]. Concretely, Fung et al. [37] considered two types of targeted attacks by Sybil clones: label-flipping attacks and backdoor attacks.

TABLE II
PRIVACY-PRESERVING TECHNIQUES FOR FL

Privacy-preserving Techniques		Existing Works
Homomorphic Encryption		[116], [117]
DP	CDP	[118], [7]
	LDP	[119], [120] , [121], [27],[122] ,[123] , [45]
	DDP+Cryptography	[30], [46],[124]
Secure Multiparty Computation		[125] , [5]

V. Defenses Against Privacy Attacks

While privacy preservation has been extensively studied in the ML community, privacy preservation in FL can be more challenging due to the sporadic access to power and network connectivity, statistical heterogeneity in the data, and so on. Existing works in privacy-preserving FL are mostly developed based on the well-known privacy-preserving techniques, including: 1) HE, such as Paillier [126], Elgamal [127], and Brakerski-Gentry-Vaikuntanathan cryptosystems [128]; 2) SMC, such as garbled circuits [129], and secret sharing [130]; and 3) DP [131], [132]. A concise summary of privacy-preserving techniques is listed in Table II.

A. Privacy Preservation Through Homomorphic Encryption

An HE scheme allows arithmetic operations to be directly performed on ciphertexts, which is equivalent to a specific linear algebraic manipulation of the plaintext. Existing HE techniques can be categorized into: 1) fully HE; 2) somewhat HE; and 3) partially HE. Fully HE can support arbitrary computation on ciphertexts but is less efficient [128]. On the other

hand, somewhat HE and partially HE are more efficient but are specified by a limited number of operations [126], [127], [133], [134]. Partially HE schemes are more widely used in practice, including RSA [134], El Gamal [127], Paillier [126], and so on. The homomorphic properties can be described as

$$E_{pk}(m_1 + m_2) = c_1 \oplus c_2$$
$$E_{pk}(a \cdot m_1) = a \otimes c_1$$

where a is a constant, m_1 and m_2 are the plaintexts that need to be encrypted, and c_1 and c_2 are the ciphertext of m_1 and m_2, respectively.

HE is widely used and is especially useful for securing the learning process by computing encrypted data. However, doing arithmetic on the encrypted numbers comes at a cost of memory and processing time. For example, with the Paillier encryption scheme, the encryption of an encoded floating-point number (whether single or double precision) is $2m$ bits long, where m is typically at least 1024 and the addition of two encrypted numbers is 2∼3 orders of magnitude slower than the unencrypted equivalent [9]. Moreover, polynomial approximations need to be made to evaluate nonlinear functions in ML algorithms, resulting in a tradeoff between utility and privacy [116], [117]. For example, to protect individual gradients, Aono et al. [31] used additively HE to preserve the privacy of gradients and enhance the security of the distributed learning system. However, their protocol not only incurs large communication and computational overheads but also results in utility loss. Furthermore, it is not able to withstand collusion between the server and multiple participants. Hardy et al. [9] applied federated logistic regression on vertically partitioned data encrypted with an additively homomorphic scheme to secure against an honest-but-curious adversary. Overall, all these works incur extra communication and computational overheads, which limit their applications in H2C scenarios.

B. Privacy-Preservation Through SMC

SMC [129] enables different participants with private inputs to perform a joint computation on their inputs without revealing them to each other. Mohassel and Zhang [125] proposed SecureML that conducts privacy-preserving learning via SMC, where data owners need to process, encrypt and/or secret-share their data among two noncolluding servers in the initial setup phase. SecureML allows data owners to train various models on their joint data without revealing any information beyond the outcome. However, this comes at a cost of high computation and communication overhead, which may hamper participants' interest to collaborate. Bonawitz et al. [5] proposed a secure, communication-efficient, and failure-robust protocol based on SMC for secure aggregation of individual gradients. It ensures that the only information about the individual users the server learns is what can be inferred from the aggregated results. The security of their protocol is maintained under both the honest-but-curious and malicious settings, even when the server and a subset of users act maliciously—colluding and deviating arbitrarily from the protocol. That is, no party learns anything more than the sum of the inputs of a subset of honest users of a large size [5].

In general, SMC techniques ensure a high level of privacy and accuracy, at the expense of high computation and communication overhead, thereby doing a disservice to attracting participation. Another main challenge facing SMC-based schemes is the requirement for simultaneous coordination of all participants during the entire training process. Such a multiparty interaction model may not be desirable in practical settings, especially under the commonly considered participant-server architecture in FL settings. Besides, SMC-based protocols can enable multiple participants to collaboratively compute an agreed-upon function without leaking input information from any participant except for what can be inferred from the outcomes of the computation [135], [136]. That said, SMC cannot fully guarantee protection from information leakage, which requires additional DP techniques to be incorporated into the multiparty protocol to address such concerns [137], [138], [139], [140].

In summary, HE- or SMC-based approaches may not be applicable to large-scale FL scenarios as they incur substantial additional communication and computation costs. Moreover, encryption-based techniques need to be carefully designed and implemented for each operation in the target learning algorithm [141], [142]. Finally, all the cryptography-based protocols prevent anyone from auditing participants' updates to the joint model, which leaves space for the malicious participants to attack. For example, malicious participants can introduce stealthy backdoor functionality into the global model without being detected [36].

C. Privacy-Preservation Through Differential Privacy

DP was originally designed for the single database scenario, where, for every query made, a database server answers the query in a privacy-preserving manner with tailored randomization [131]. In comparison with encryption-based approaches, DP trades off privacy and accuracy by perturbing the data in a way that: 1) is computationally efficient; 2) does not allow an attacker to recover the original data; and 3) does not severely affect the utility.

The concept of DP is that the effect of the presence or the absence of a single record on the output likelihood is bounded by a small factor ϵ. As defined in Definition 2, (ϵ, δ)-approximate DP [132] relaxes pure ϵ-DP by a δ additive term, which means that the unlikely responses need not satisfy the pure DP criterion.

Definition 2 ((ϵ, δ)-DP [132]): For scalars $\epsilon > 0$ and $0 \leq \delta < 1$, mechanism $\mathcal{M}$ is said to preserve (approximate) (ϵ, δ)-DP if, for all adjacent datasets, $D, D' \in \mathcal{D}^n$ and measurable $S \in \text{range}(\mathcal{M})$

$$\Pr\{\mathcal{M}(D) \in S\} \leq \exp(\epsilon) \cdot \Pr\left\{\mathcal{M}\left(D'\right) \in S\right\} + \delta \ .$$

To avoid the worst case scenario of always violating the privacy of a δ fraction, the standard recommendation is to choose $\delta \ll 1/|D|$, where $|D|$ is the size of the database. This strategy forecloses the possibility of one particularly devastating outcome, but other forms of information leakage remain.

The privacy community generally categorizes DP into the following three categories as per different trust assumptions

TABLE III
COMPARATIVE ANALYSIS AMONG CDP, LDP, AND DDP

DP type	Trusted aggregator?	Who should add noise?	Privacy Guarantee	Error Bound
CDP [118], [7]	Yes	aggregator	aggregated value	$O(\frac{1}{\epsilon})$
LDP [27], [143]	No	user	locally released value	$O(\frac{\sqrt{n}}{\epsilon})$
DDP [30], [44]	No	user	aggregated value	$O(\frac{1}{\epsilon})$

and noise sources: CDP, LDP, and DDP. A comprehensive comparison among CDP, LDP, and DDP is listed in Table III.

1) Centralized Differential Privacy: CDP was originally designed for the centralized scenario where a trusted database server, which is entitled to see all participants' data in the clear, wishes to *answer queries or publish statistics* in a privacy-preserving manner by randomizing query results [42], [131], [144]. When CDP meets FL, CDP assumes a trusted aggregator, who is responsible for adding noise to the aggregated local gradients to ensure record-level privacy of the whole data of all participants [7], [118]. However, CDP is geared to tackle thousands of users for training to converge and achieve an acceptable tradeoff between privacy and accuracy [7], resulting in a convergence problem with a small number of participants [145]. Moreover, CDP can achieve acceptable accuracy only with a large number of participants, thus not applicable to H2B with relatively a small number of participants.

Meanwhile, the assumption of a trusted server in CDP is ill-suited in many applications as it constitutes a single point of failure for data breaches and saddles the trusted curator with legal and ethical obligations to keep the user data secure. When the aggregator is untrusted, which is often the case in distributed scenarios, LDP [146] or DDP is needed [138], [147] to protect the privacy of individuals.

2) Local Differential Privacy: LDP [146] offers a stronger privacy guarantee, and data owners perturb their private information to satisfy DP locally before reporting it to an untrusted data curator [122], [123], [148]. A comprehensive survey of LDP can be referred to [149]. A formal definition of LDP is given in Definition 3.

Definition 3 ((ϵ, δ)-LDP): A randomized algorithm $\mathcal{M}$ satisfies (ϵ, δ)-LDP ((ϵ, δ)-LDP) if and only if, for any input v and v', we have

$$\Pr\{\mathcal{M}(v) = o\} \leq \exp(\epsilon) \cdot \Pr\{\mathcal{M}(v') = o\} + \delta$$

for $\forall o \in \text{Range}(\mathcal{M})$, where $\text{Range}(\mathcal{M})$ denotes the set of all possible outputs of the algorithm $\mathcal{M}$. Furthermore, $\mathcal{M}$ is said to preserve (pure) ϵ-LDP if the condition holds for $\delta = 0$.

Although the randomized response [150] and its variants [151] have been widely used to provide LDP when individuals disclose their personal information, we remark that all the randomization mechanisms used for CDP, such as Laplace and Gaussian mechanisms [132], can be individually used by each participant to ensure LDP in isolation. However, in the distributed scenario, without the help of cryptographic techniques, each participant has to add enough calibrated noise to ensure LDP. The attractive privacy properties of LDP, thus, come with a huge utility degradation, especially with billions of individuals. Under the CDP model, the aggregator releases the aggregated value with an expected additive error of at most $\Theta(1/\epsilon)$ to ensure ϵ-DP (e.g., using the Laplace mechanism [132]). In contrast, under the LDP model, at least $\Omega(\sqrt{n}/\epsilon)$ additive error in expectation must be incurred by any ϵ-DP mechanism for the same task [146], [152]. This gap is essential for eliminating the trust in the centralized server and cannot be removed by algorithmic improvement [153].

To protect FL with homogeneous architectures, in which model parameters or gradients are shared, for example, Shokri and Shmatikov [154] first applied LDP to distributed learning/FL, in which each participant individually adds noise to its gradients before releasing to the server, thus ensuring LDP. However, their privacy bounds are given per-parameter, and a large number of parameters prevents their method from providing a meaningful privacy guarantee [42]. Other approaches that are also considered to apply LDP to FL can only support shallow models, such as logistic regression, and only focus on simple tasks and datasets [119], [120], [121]. Bhowmick et al. [27] presented a viable approach to large-scale local private model training and introduced a relaxed version of LDP by limiting the power of potential adversaries. Due to the high variance of their mechanism, it requires more than 200 communication rounds and incurs much higher privacy costs, i.e., MNIST ($\epsilon = 500$) and CIFAR-10 ($\epsilon = 5000$). Note that ϵ required in [27] is relatively large, as they considered only privacy protection against *reconstruction attacks* instead of membership attacks. Their obtained results suggested that using LDP mechanisms with *large* ϵ may still provide decent protection against reconstruction. Li et al. [143] proposed locally differentially private algorithms in the context of metalearning, which might be applicable to FL with personalization. However, it only provides provable learning guarantees in convex settings. Truex et al. [123] applied *condensed LDP* (α-CLDP) into FL. However, α-CLDP results in a weak privacy guarantee. Another contemporary work called LDP-FL [122] achieves better performance on both effectiveness and efficiency than [123] with a special communication design for deep learning approaches.

To protect FL with heterogeneous architectures, in which model predictions are shared, one naive approach is adding the locally differentially private random noise to the predictions like in previous works. Although the privacy concern is mitigated with random noise perturbation, it brings a new problem with a substantial tradeoff between privacy budget and model utility. Sun and Lyu [45] filled in this gap by proposing a novel framework called FEDMD-NFDP, which integrated a novel noise-free DP (NFDP) mechanism into FedMD. The LDP guarantee of NFDP roots in the local data sampling process, which explicitly eliminates noise addition and privacy cost explosion issues in previous works.

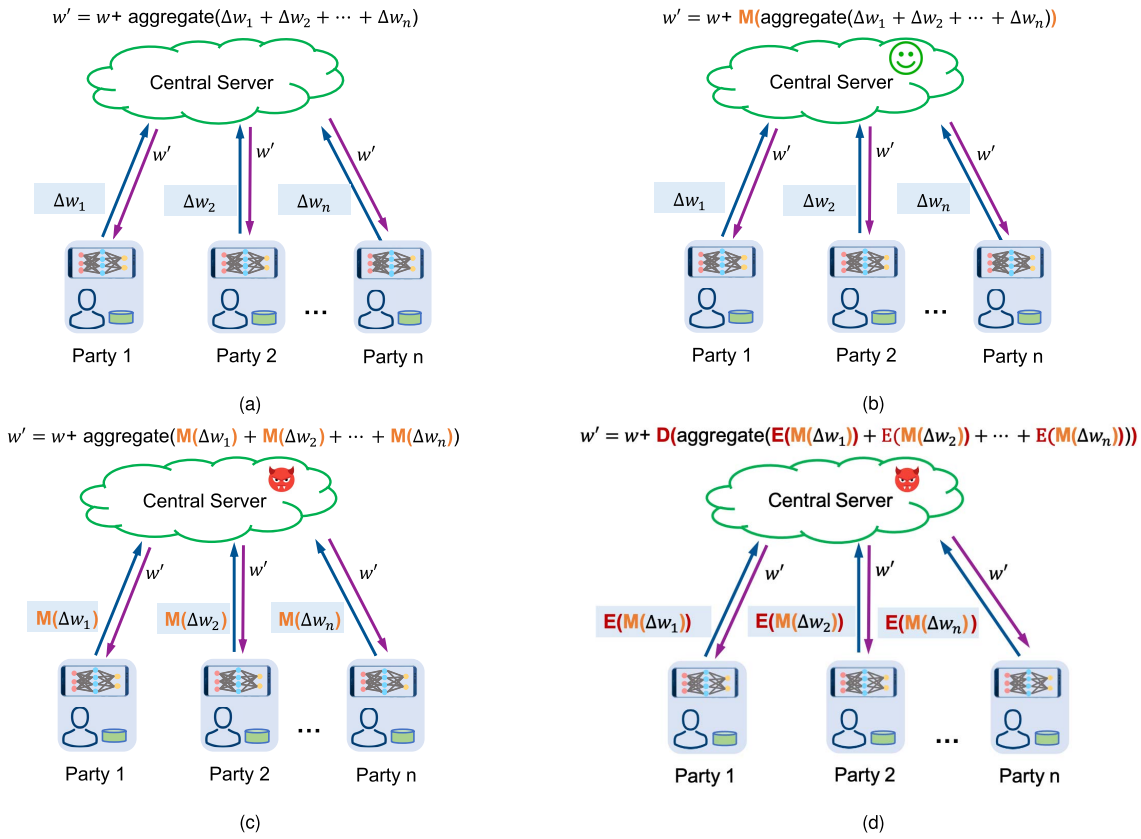


Fig. 5. Illustration of FL without privacy and with different DP mechanisms. M denotes a DP mechanism used to privatize the data. (b) In centralized DP, the central server is trusted. (c) In LDP, the central server is not trusted; gradients are perturbed to ensure LDP before forwarding to the central server. (d) In distributed DP, the central server is also not trusted; gradients are perturbed via DP mechanism M and encrypted via encryption operation E to ensure privacy before forwarding to the central server, which needs to finally decrypt (D) the aggregated ciphertext. (a) FL without privacy. (b) Centralized DP: FL with a trusted server. (c) Local DP: FL without a trusted server. (d) Distributed DP with SMC: FL without a trusted server.

3) Distributed Differential Privacy: DDP bridges the gap between LDP and CDP while ensuring the privacy of each individual by combining with cryptographic protocols [30], [137], [138], [139], [140]. Therefore, DDP avoids placing trust in any server and offers better utility than LDP. Theoretically, DDP offers the same utility as CDP, as the total amount of noise is the same.

The notion of DDP reflects the fact that the required noise in the target statistic is sourced from multiple participants [147]. Approaches to DDP that implement an overall additive noise mechanism by summing the same mechanism run at each participant (typically with less noise) necessitate mechanisms with stable distributions—to guarantee proper calibration of known end-to-end response distribution—and cryptography for hiding all but the final result from participants [30], [46], [124], [137], [138], [139], [147]. Stable distributions include Gaussian distribution, Binomial distribution [44], and so on, i.e., the sum of Gaussian random variables still follows a Gaussian distribution, and the sum of Binomial random variables still follows a Binomial distribution. DDP utilizes this nice stability to permit each participant to randomize its local statistic to a lesser degree than would LDP. However, in DDP, only the sum of the individually released statistics is (ϵ, δ)-differentially private but not the individually released statistic. Therefore, DDP necessitates the help of SMC to maintain utility and ensure aggregator obliviousness, as evidenced in [30], [46], [125], [138], [139], [140], and [141].

An illustration of FL without privacy and with different DP mechanisms is given in Fig. 5. Another parallel line of work for privacy-preserving distributed learning is to transfer the knowledge of the ensemble of multiple models to a student model [42], [144], [155], [156]. For example, Hamm et al. [155] first created labeled data from auxiliary unlabeled data, then used the labeled auxiliary data to find an empirical risk minimizer, and, finally, released a differentially private classifier using output perturbation [157]. Similarly, Papernot et al. [42], [144] proposed *private aggregation of teacher ensembles* (PATE) to first train an ensemble of teachers on disjoint subsets of private data and then perturb the knowledge of the ensemble of teachers by adding noise to the aggregated teacher votes before transferring the knowledge to a student. Finally, a student model is trained on the aggregate output of the ensemble such that the student learns to accurately mimic the ensemble. PATE requires a lot of participants to achieve reasonable accuracy, and each participant needs to have enough data to train an accurate model, which might not hold in the FL system, where the data distribution of participants might be highly unbalanced, making this approach unsuitable to the FL system.

VI. Defenses Against Poisoning Attacks

Robustness to poisoning attacks is a desirable property in FL. To address poisoning attacks, many robust aggregation schemes are proposed in the literature. Known defenses to poisoning attacks in a centralized setting, such as robust losses [158] and anomaly detection [98], assume control of the participants or explicit observation of the training data. Neither of these assumptions is applicable to FL in which the server only observes the model parameters/updates that are sent as part of the iterative ML algorithm [37]. We summarize the defenses against untargeted and targeted attacks as follows.

A. Defenses Against Untargeted Attacks

For Byzantine-resilient aggregation, an algorithm is Byzantine fault tolerant [40] if its convergence is robust even when a large portion of participants is adversarial. In the following, we list several representative attempts that try to defend against untargeted Byzantine attacks.

1) AUROR: Shen et al. [159] introduced a statistical mechanism called AUROR to detect malicious users while generating an accurate model. AUROR is based on the observation that indicative features (most important model features) from the majority of honest users will exhibit a similar distribution, while those from malicious users will exhibit an anomalous distribution. It then uses k-means to cluster participants' updates across training rounds and discards the outliers, i.e., contributions from small clusters that exceed a threshold distance are removed. The accuracy of a model trained using AUROR drops by only 3% even when 30% of all the users are adversarial.

2) Krum: Blanchard et al. [40] proposed Krum, in which the top f contributions to the model that is furthest from the mean participant contribution are removed from the aggregation. Krum uses the Euclidean distance to determine which gradient contributions should be removed. It can theoretically withstand poisoning attacks of up to 33% adversaries among the participants, i.e., given n agents of which f is Byzantine, Krum requires that $n \geq 2f + 3$. Krum is resistant to attacks by omniscient adversaries—aware of a good estimate of the gradient—who send the opposite vector multiplied by a large factor. It is also resistant to attacks by adversaries who send random vectors drawn from a Gaussian distribution (the larger the variance of the distribution, the stronger the attack). Multi-Krum is a variant of Krum, which intuitively interpolates between Krum and averaging, thereby combining the resilience properties of Krum with the convergence speed of averaging. Essentially, Krum filters outliers based on the entire update vector but does not filter coordinatewise outliers.

3) Coordinatewise Statistics: To address this issue, Yin et al. [63] proposed two robust distributed gradient descent algorithms: one based on coordinatewise median and the other based on the coordinatewise trimmed mean. Unfortunately, median-based rules can incur a prohibitive computational overhead in large-scale settings [163]. Guerraoui et al. [160] proposed a meta-aggregation rule called Bulyan, a two-step meta-aggregation algorithm based on the Krum and trimmed median, which filters malicious updates followed by computing the trimmed median of the remaining updates. Median and geometric-median-based robust aggregation rules are also extensively explored in [165] and [166]. Pillutla et al. [64] proposed a robust aggregation approach called RFA by replacing the weighted arithmetic mean with an approximate geometric median, so as to reduce the impact of the contaminated updates. Unfortunately, RFA can only handle a few types of poisoning attackers but is not applicable to Byzantine attacks.

4) Weakness of Current Defenses: In spite of their robustness guarantees, recent inspections revealed that previous Byzantine-robust FL mechanisms are also quite brittle and can be easily circumvented. Bhagoji et al. [36] showed that targeted model poisoning of deep neural networks is effective even against the Byzantine-robust aggregation rules, such as Krum and coordinatewise median. Baruch et al. [109] and Xie et al. [100] showed that, while the Byzantine-robust aggregation rules may ensure that the influence of the Byzantine workers in any single round is limited, the attackers can couple their attacks across the rounds, moving weights significantly away from the desired direction, and, thus, achieve the goal of lowering the model quality. Xu and Lyu [166] demonstrated that multi-Krum is not robust against untargeted poisoning. This is because multi-Krum is based on the distance between each gradient vector and the mean vector, while the mean vector is not robust against untargeted poisoning. Fang et al. [80] showed that aggregation rules (e.g., Krum [40], Bulyan [160], trimmed mean [63], coordinatewise median [63], and other median-based aggregators [62]) that were claimed to be robust against Byzantine failures are not effective in practice against optimized local model poisoning attacks that carefully craft local models on the compromised participants such that the aggregated global model deviates the most toward the inverse of the direction along which the global model would change when there are no attacks. All these highlight the need for more effective defenses against Byzantine attackers in FL.

5) Other Possibilities: Other works investigate Byzantine robustness from different lenses. Chen et al. [163] presented DRACO, a framework for robust distributed training via algorithmic redundancy. DRACO is robust to arbitrarily malicious computing nodes while being orders of magnitude faster than state-of-the-art robust distributed systems. However, DRACO assumes that each participant can access other participants' data, limiting its practicability in FL. Su and Xu [94] proposed to robustly aggregate the gradients computed by the Byzantine participants based on the filtering procedure proposed by Steinhardt et al. [167]. Bernstein et al. [39] proposed SIGNSGD, which is combined with a majority vote to enable participants to upload elementwise signs of their gradients to defend against three types of half "blind" Byzantine adversaries: 1) adversaries that arbitrarily rescale their stochastic gradient estimate; 2) adversaries that randomize the sign of each coordinate of the stochastic gradient; and 3) adversaries that invert their stochastic gradient estimate.

B. Defenses Against Targeted Attacks

Existing defenses against the targeted backdoor attacks can be categorized into two types: detection methods and erasing methods [168].

TABLE IV
STATE-OF-THE-ART DEFENSES AGAINST FL POISONING. n IS THE NUMBER OF PARTICIPANTS. NOTE THAT SOME DEFENSES HAVE NO THEORETIC BREAKING POINT

Defense	Technique	IID Data	Non-IID Data	Breaking Point	Targeted Poisoning	Untargeted Poisoning
AUROR [159]	Clustering	✓	×	NA	×	✓
Krum/Multi-Krum [40]	Euclidean distance	✓	×	$(n-2)/2n$	×	✓
Coordinate-wise Median [63]	Coordinate-wise median	✓	×	1/2	×	✓
Bulyan [160]	Krum + trimmed median	✓	×	$(n-3)/4n$	×	✓
RFA [64]	Geometric median	✓	×	NA	×	✓
FoolsGold [37]	Contribution similarity	✓	✓	NA	✓	×
Sun *et al.* [38]	Norm-bounding and DP	✓	✓	NA	✓	✓ [73]
Wu *et al.* [161]	Pruning	✓	✓	NA	✓	×
CRFL [162]	Clipping and smoothing	✓	✓	NA	✓	×

1) Detection: Detection methods exploit activation statistics or model properties to determine whether a model is backdoored [169], [170] or whether a training/test example is a backdoor example [52]. There are a number of detection algorithms that are designed to detect which inputs contain a backdoor, and which parts of the model (its activation functions specifically) are responsible for triggering the adversarial behavior of the model, in order to remove the backdoor [47], [52], [53], [103], [171]. These algorithms rely on the statistical difference between the latent representations of backdoor-enabled and clean (benign) inputs in the poisoned model. These backdoor detection algorithms can, however, be bypassed by maximizing the latent indistinguishability of backdoor-enabled adversarial inputs and clean inputs [172].

2) Erasing: While detection can help identify potential risks, the backdoored model still needs to be purified/erased since the potential impact of backdoor triggers remains uncleared in the backdoored models. The erasing methods take a step further and aim to purify the adverse impacts on models caused by the backdoor triggers. The current state-of-the-art erasing methods are mode connectivity repair (MCR) [173] and NAD [59]. MCR mitigates the backdoors by selecting a robust model in the path of loss landscape, while NAD leverages knowledge distillation to erase triggers. Other previous methods, including fine-tuning, denoising, and fine-pruning [171], have been shown to be insufficient against the latest attacks [58], [174]. Another more recent work called ABL [60] aims to train clean models given backdoor-poisoned data. The overall learning process is framed as a dual task of learning the clean and the backdoor portions of data. Based on this process, ABL can: 1) help isolate backdoor examples at an early training stage and 2) break the correlation between backdoor examples and the target class at a later training stage.

3) Backdoor Defenses in FL: Despite the promising backdoor defense results in the centralized setting, it is still unclear whether these defenses can be smoothly adapted to the FL setting, especially in the non-i.i.d. setting. For backdoor defense in FL, Sun et al. [38] showed that clipping the norm of model updates and adding Gaussian noise can mitigate backdoor attacks that are based on the model replacement paradigm. Andreina et al. [175] incorporated an additional validation phase in each round of FL to detect backdoor. However, none of these provides certified robustness guarantees. Certified robustness for FL against backdoor attacks remains largely unexplored. Xie et al. [162] provided the first general framework called *certifiably robust FL* (CRFL) to train CRFL models against backdoors.

4) Sybil Defenses in FL: In addition to backdoors, the targeted attack can also be launched by Sybil clones [37]. To defend against the targeted poisoning attack by Sybil clones, Fung et al. [37] exploited the characteristic behavior that Sybils are more similar to each other than the similarity observed amongst the honest clients and proposed FoolsGold: a new defense scheme against FL Sybil attacks by adapting the learning rate of participants based on contribution similarity. Note that FoolsGold does not bound the expected number of attackers by assuming that attackers can spawn a large number of Sybils, rendering assumptions about proportions of honest participants unrealistic [40]. In addition, FoolsGold requires no auxiliary information beyond the learning process and makes fewer assumptions about participants and their data. The robustness of FoolsGold holds for different distributions of participant data, varying poisoning targets, and various Sybil strategies and can be applied successfully on both FedSGD and FedAvg.

5) Summary: We list the most representative defenses against poisoning attacks in FL in Table IV. Some of them have breaking points, i.e., the fraction of malicious participants, and robustness guarantees cannot be provided if the fraction of malicious participants is larger than the breaking point.

6) Remark: Note that both the untargeted and targeted poisoning attacks are less effective in settings with infrequent participation like H2C [35]. Moreover, under practical production FL environments, Shejwalkar et al. [73] have shown that FL, even without any defenses, is highly robust in practice. For production cross-device FL (H2C), which contains thousands to billions of clients, poisoning attacks have no impact on existing robust FL algorithms even with impractically high percentages of compromised clients. For production cross-silo FL (H2B), which contains up to 100 clients, data poisoning attacks are completely ineffective; model poisoning attacks are unlikely to play a major risk when the clients involved are bound by contract and their software stacks professionally maintained (e.g., in banks, hospitals). Some exceptional cross-silo scenarios are most likely with a strong incentive (e.g., financial) causing multiple parties to be willing to risk a breach of contract by colluding or for one party to hack thereby risking criminal liability. Therefore, we conclude that

these poisoning attacks are more likely to happen in some exceptional H2B scenarios.

VII. Discussion and Promising Directions

There are still potential vulnerabilities that need to be addressed in order to improve the privacy and robustness of FL systems. Moreover, there are multiple design goals that are equally important with privacy and robustness and, thus, need to be considered simultaneously in FL. In this section, we outline research directions that we believe are promising.

7) Curse of Dimensionality: Large models, with high dimensional parameter vectors, are particularly susceptible to privacy and security attacks [176]. Most FL algorithms require overwriting the local model parameters with the global model. This makes them susceptible to poisoning attacks, as the adversary can make small but damaging changes in the high-dimensional models without being detected. Almost all of the well-designed Byzantine-robust aggregators [40], [63], [64] still suffer from the curse of dimensionality. Specifically, the estimation error scales up with the size of the model in a square-root manner. Thus, sharing model parameters may not be a strong design choice in FL; it opens all the internal states of the model to inference attacks and maximizes the model's malleability by poisoning attacks. To address these fundamental shortcomings of FL, it is worthwhile to explore whether sharing gradients is essential. Instead, sharing less sensitive information (e.g., SIGNSGD [39]) or only sharing model predictions [25], [45], [176] in a black-box manner may result in more robust privacy protection in FL.

8) Rethinking Current Privacy Attacks: There are several inherent weaknesses in current attacks that may limit their applicability in FL [177]. For example, the GAN attack assumes that the entire training corpus for a given class comes from a single participant, and only in the special case where all class members are similar, GAN-constructed representatives are similar to the training data [75]. These assumptions are less practical in FL. For DLG [29] and iDLG [32], both works: 1) adopt a second-order optimization method called L-BFGS that is more computationally expensive compared with first-order optimizations;2) are only applicable to gradients computed on minibatches of data, i.e., at most $B = 8$ in DLG and $B = 1$ in iDLG, which is not the real case for FL, in which gradient is normally shared after at least 1 epoch of local training; and 3) used untrained model, neglecting gradients over multiple communication rounds. Attacking FL systems in a more efficient manner and under more practical settings remain largely unexplored. In addition, whether current attacks still work in FL that uses adaptive optimization methods [178], such as SGDM and Adam, remains unknown.

9) Rethinking Current Defenses: FL with secure aggregation for the purpose of privacy is more susceptible to poisoning attacks as individual updates cannot be inspected. Similarly, it is still unclear if AT, one state-of-the-art defense approach against adversarial attacks in conventional ML [93], [179], [180], can be adapted to FL, as AT was developed primarily for i.i.d. data and remains unclear for its performance in non-i.i.d. settings. Moreover, AT is computationally expensive and may hurt the performance [181], which may not be feasible for the H2C scenario. In terms of DP-based methods [7], [121], [182], [183], [184], [185], record-level DP bounds the success of membership inference but does not prevent property inference applied to a group of training records [28]. Participant-level DP, on the other hand, is geared to work with thousands of users for training to converge and achieving an acceptable tradeoff between privacy and accuracy [7]. The FL model fails to converge with a small number of participants, making it unsuitable for H2B scenarios. Furthermore, DP may hurt the accuracy of the learned model [186], which may not be appealing to the industry. Further work is needed to investigate if participant-level DP can protect FL systems with fewer participants. It is also worthwhile to explore whether we can use the condensed data [187] rather than the raw data for local model training in order to better protect privacy.

10) Optimizing Defense Mechanism Deployment: When deploying defense mechanisms to check if any adversary is attacking the FL system, the FL server will need additional computational costs. In addition, different types of defense mechanisms may exhibit different effectiveness against different attacks and incur different costs. It is important to study how to optimize the timing of deploying defense mechanisms or the announcement of deterrence measures. Game theoretic research holds promise in addressing this challenge.

11) Test-Phase Privacy in FL: This survey mainly focuses on the training phase attacks and defenses in FL, considering the more attack possibilities opened by the distributed training property of FL systems. In fact, FL is also vulnerable to both privacy and robustness attacks during the test/inference phase by the users of the final FL model when it is deployed as a service.

In terms of privacy vulnerability, the trained global model may reveal sensitive information from model predictions when deployed as a service, causing privacy leakage. In such a setting, an adversary does not have direct access to the model parameters but may be able to view input–output pairs. Previous studies have shown a series of privacy leakage given only black-box access to the trained models, such as: 1) model stealing attacks in which model parameters can be reconstructed by an adversary who only has access to an inference/prediction API based on those parameters [82], [83], [84], [85] and 2) MIAs that aim to determine if a particular record was used to train the model [86]. FL models face a similar dilemma during model deployment for testing purposes. The development of effective defenses against privacy leakage during model deployment calls for further investigations.

12) Test-Phase Robustness in FL: In terms of robustness vulnerability, recent studies [90], [91], [92] have shown that FL is also vulnerable to well-crafted adversarial examples. During inference time, the attackers can add a very small perturbation to the test data, making the test data almost indistinguishable from natural data and yet classified incorrectly by the global model. For federated robustness against adversarial examples, Zizzo et al. [90] and Hong et al. [91] proposed to apply AT to FL, i.e., FAT, in order to achieve adversarial robustness in FL. Zizzo et al. [90] noticed that conducting AT on all participants leads to divergence of the model. To solve this problem, they conducted AT on only a proportion of

participants for better convergence. Another recent work by Hong et al. [91] considered hardware heterogeneity in FL, i.e., only limited users can afford AT. Hence, they conduct AT on only a proportion of participants that have powerful computation resources while standard training on the rest of the participants. Shah et al. [92] investigated the impact of communication rounds in FAT and proposed a dynamic AT. The training of all the above FAT works is unstable, which potentially hurts the convergence and performance. Moreover, AT typically requires significant computation and a longer time to converge, and it is unclear how it performs in non-i.i.d. settings. Chen et al. [188] took the first step to investigate FAT under non-i.i.d. setting with label skewness. However, how to speed up AT in FL may be required in the future. Overall, there exist difficulties in applying AT to the federated setting. This motivates future works to explore more effective approaches to maintain both natural accuracy and robustness in FL.

In addition to the adversarial examples, recent works [83], [84] have validated that the API services (the victim/target model) can be easily stolen and are vulnerable to adversarial example transferability attacks. It would be interesting to explore whether the collaboratively built global model in FL is also facing a similar problem and how to effectively claim the ownership of the trained model [189].

13) Relationship With GDPR: GDPR[1] defines six-core principles as rational guidelines for service providers to manage personal data, including: 1) lawfulness, fairness, and transparency; 2) purpose limitation; 3) data minimization; 4) accuracy; 5) storage limitation; and 6) integrity and confidentiality (security). GDPR also requires data controllers to provide the following rights for data subjects if capable (the GDPR Articles 12–23): 1) right to be informed; 2) right of access; 3) right to rectification; 4) right to erasure (right to be forgotten); 5) right to restrict processing; 6) right to data portability; 7) right to object; and 8) rights in relation to automated decision making and profiling. Although FL has emerged as a prospective solution that facilitates distributed collaborative learning without disclosing original training data, unfortunately, FL is not naturally compliant with the GDPR [190], as pointed out by a recent survey [190], which has dedicated to surveying the relationship between FL and GDPR requirements. For example, the secure aggregation mechanism in FL amplifies the lack of transparency and fairness in FL systems and, thus, fails to fully comply with the GDPR requirements of fairness and transparency; malicious participants in FL may conduct either data or model poisoning attacks for an unauthorized purpose, and local ML model parameters obtained from participants are no longer minimal for the original purpose. These possible attacks, which lead to noncompliance with the GDPR, should be addressed. Henceforth, it is worthwhile to explore approaches to empower FL-based systems to follow the GDPR regulatory guidelines and, thus, fully comply with the GDPR.

14) Threats and Protections of VFL and FTL: This survey mainly focuses on the threats to HFL; there are some recent exploratory efforts on threats and protections of VFL and FTL.

[1]https://gdpr-info.eu

For VFL, Secureboost [191] considered user privacy and data confidentiality in VFL and presented an approach to train a high-quality tree-boosting model collaboratively. A recent work called FederBoost [192] pointed out that Secureboost is expensive since it requires cryptographic computation and communication for each possible split; thus, they proposed a vertical FederBoost, which does *not* require any cryptographic operation. Another recent work by Jin et al. [193] uncovered the risk of *catastrophic data leakage in vertical FL* (CAFE) through a novel algorithm that can perform large-batch data leakage with high data recovery quality and theoretical guarantees. They empirically demonstrated that CAFE can recover large-scale private data from the shared aggregated gradients in VFL settings, overcoming the batch limitation problem in current data leakage attacks.

For FTL, Gao et al. [24] proposed an end-to-end privacy-preserving multiparty learning approach with two variants based on HE and secret-sharing techniques, respectively, in order to build a heterogeneous FTL (HFTL) framework. Liu et al. [23] adopted two secure approaches, namely, HE and secret sharing for preserving privacy. The HE approach is simple but computationally expensive. By contrast, the secret-sharing approach offers the following advantages: 1) there is no accuracy loss and 2) computation is much faster than the HE approach. The major drawback of the secret-sharing approach is that one has to offline generate and store many triplets before online computation.

Overall, there is still a large space for VFL and FTL. It is worth further investigation as to whether existing threats in HFL are all valid in VFL and FTL or if there are new threats and countermeasures in VFL and FTL.

15) Vulnerabilities to Free-Riding Participants: In FL systems, there may exist free-riders who aim to benefit from the global model but do not want to contribute any useful information, thus compromising collaborative fairness [183], [184], [194]. The main incentives for free-riders include: 1) the participant does not have any data to train the local model; 2) the participant is too concerned about its privacy and, thus, chooses to release fake updates; and 3) the participant does not want to consume or does not have any local computation power to train the local model. In the current FL paradigm [6], all participants receive the same federated model at the end of the collaborative training, regardless of their individual contributions. This makes the paradigm vulnerable to free-riding participants [166], [183], [184], [195], [196]. How to prevent free-riding remains an open challenge. Incentivized FL (via allocating different reputations to different participants and penalizing unreliable or malicious participants) [194], [197] would be an important direction to help address free-riding problem and possibly the before-mentioned privacy and poisoning attack problems in Sections III and IV.

16) Reliability of FL Over Wireless Network: When FL systems are deployed in the real world, unreliable data may be uploaded by mobile devices (i.e., workers). The workers may perform unreliable updates intentionally, e.g., the data poisoning attack, or unintentionally, e.g., low-quality data caused by energy constraints or high-speed mobility [198]. Similarly, when FL meets UAVs, reliability is a key factor that may

affect performance [14]. Therefore, finding out trusted and reliable workers for FL tasks becomes critical. The concept of reputation could be used for reliable worker selection strategy design in order to keep the low-quality devices from affecting the learning efficiency and accuracy [198].

17) Extension to Decentralized FL: Decentralized FL is an emerging research area, where there is no single central server in the system [3], [7], [183], [184]. Decentralized FL is potentially more useful for H2B scenarios where the business participants do not trust any third party. In this paradigm, each participant could be elected as a server in a round-robin manner. The recent emerging swarm learning [199] can be deemed as a decentralized FL framework, which unites edge computing, blockchain-based peer-to-peer networking, and coordination while maintaining confidentiality without the need for a central coordinator. It is interesting to investigate whether existing threats to server-based FL still apply in decentralized FL.

18) Efficient FL With Single Round Communication: In addition to privacy and robustness, communication cost is another major concern that may hinder the large-scale implementation of FL [200]. One-shot FL has recently emerged as a promising approach for communication efficiency. It allows the central server to learn a model in a single communication round. Despite the low communication cost, existing one-shot FL methods are mostly impractical or face inherent limitations, e.g., a public dataset is required, participants' models are homogeneous, additional data/model information needs to be uploaded, and unsatisfactory performance [201], [202], [203]. Recent work proposed a more practical data-free approach named DENSE for a one-shot FL framework with heterogeneity [204]. Other alternative one-shot FL approaches with practical assumptions are worthwhile to explore, considering the alluring communication efficiency and less privacy and robustness attack surfaces exposed in one-shot FL.

19) Achieving Multiple Objectives Simultaneously: There are no existing works that can satisfy multiple goals simultaneously: 1) fast algorithmic convergence; 2) good generalization performance; 3) communication efficiency; 4) fault tolerance; 5) privacy preservation; and 6) robustness to targeted, untargeted poisoning attacks, and free-riders. Previous works have attempted to solve multiple objectives at the same time. For example, Lyu et al. [183], [184] addressed collaborative fairness and privacy simultaneously; Xu and Lyu et al. [166] proposed a *robust and fair FL* (RFFL) framework to address both collaborative fairness and Byzantine robustness. However, it is important to highlight that there is an inherent conflict between privacy and robustness: defending against robustness attacks usually requires complete control of the training process or access to the training data [37], [40], [63], [159], [205], [206], which goes against the privacy requirements of FL. Although using encryption or DP-based techniques can provide provably privacy preservation, they are not robust to poisoning attacks and may produce models with undesirably poor privacy-utility tradeoffs. Agarwal et al. [30] combined DP with model compression techniques to reduce communication costs and obtain privacy benefits simultaneously. It remains largely unexplored, and there exist large gaps as to how to simultaneously achieve all the above six objectives.

VIII. Conclusion

Although FL is still in its infancy, it will continue to thrive and will be an active and important research area in the foreseeable future. As FL evolves, so will the privacy and robustness threats to FL. It is of vital importance to provide a broad overview of current attacks and defenses on FL so that future FL system designers are well aware of the potential vulnerabilities in the current designs and help them clear roadblocks toward the real-world deployment of FL. This survey serves as a concise and accessible overview of this topic, and it would greatly help our understanding of the privacy and robustness attack and defense landscape in FL. Global collaboration on FL is emerging through a number of workshops at leading AI conferences.[2] The ultimate goal of developing a general-purpose FL defense mechanism that can be robust against various attacks without degrading model performance will require interdisciplinary effort from the wider research community.

Acknowledgment

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the National Research Foundation, Singapore.

References

[1] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, Jan. 2018.

[2] M. Abadi et al., "Deep learning with differential privacy," in *Proc. CCS*, 2016, pp. 308–318.

[3] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," *Synth. Lect. Artif. Intell. Mach. Learn.*, vol. 13, no. 3, pp. 1–207, 2019.

[4] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*.

[5] K. Bonawitz et al., "Practical secure aggregation for privacy-preserving machine learning," in *Proc. CCS*, 2017, pp. 1175–1191.

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

[7] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *Proc. ICLR*, 2018, pp. 1–14.

[8] Y. Liu et al., "Fedvision: An online visual object detection platform powered by federated learning," in *Proc. IAAI*, 2020, pp. 13172–13179.

[9] S. Hardy et al., "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," 2017, *arXiv:1711.10677*.

[10] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, "Communication-efficient federated learning via knowledge distillation," *Nature Commun.*, vol. 13, no. 1, pp. 1–8, 2022.

[11] C. Wu, F. Wu, L. Lyu, Y. Huang, and X. Xie, "FedCTR: Federated native ad CTR prediction with cross platform user behavior data," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 4, pp. 62:1–62:19, 2022.

[12] J. Cui, C. Chen, L. Lyu, C. Yang, and W. Li, "Exploiting data sparsity in secure cross-platform social recommendation," in *Proc. NIPS*, 2021, pp. 10524–10534.

[13] J. Li, L. Lyu, X. Liu, X. Zhang, and X. Lyu, "FLEAM: A federated learning empowered architecture to mitigate DDoS in industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 18, no. 6, pp. 4059–4068, Jun. 2021.

[2] http://www.federated-learning.org/

[14] H. Yang, J. Zhao, Z. Xiong, K.-Y. Lam, S. Sun, and L. Xiao, "Privacy-preserving federated learning for UAV-enabled networks: Learning-based joint scheduling and resource management," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 10, pp. 3144–3159, Oct. 2021.

[15] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie, "FedGNN: Federated graph neural network for privacy-preserving recommendation," 2021, *arXiv:2102.04925*.

[16] C. Chen et al., "Vertically federated graph neural network for privacy-preserving node classification," in *Proc. IJCAI*, 2022, pp. 1–9.

[17] X. Ni, X. Xu, L. Lyu, C. Meng, and W. Wang, "A vertical federated learning framework for graph convolutional network," 2021, *arXiv:2106.11593*.

[18] C. Wu, F. Wu, L. Lyu, T. Qi, Y. Huang, and X. Xie, "A federated graph neural network framework for privacy-preserving personalization," *Nature Commun.*, vol. 13, no. 1, pp. 1–10, Dec. 2022.

[19] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.

[20] M. Kantarcioglu and C. Clifton, "Privacy-preserving distributed mining of association rules on horizontally partitioned data," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 9, pp. 1026–1037, Sep. 2004.

[21] J. Vaidya and C. Clifton, "Privacy preserving association rule mining in vertically partitioned data," in *Proc. KDD*, 2002, pp. 639–644.

[22] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Dec. 2009.

[23] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang, "A secure federated transfer learning framework," *IEEE Intell. Syst.*, vol. 35, no. 4, pp. 70–82, Jul./Aug. 2020.

[24] D. Gao, Y. Liu, A. Huang, C. Ju, H. Yu, and Q. Yang, "Privacy-preserving heterogeneous federated transfer learning," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 2552–2559.

[25] D. Li and J. Wang, "FedMD: Heterogenous federated learning via model distillation," 2019, *arXiv:1910.03581*.

[26] R. Liu et al., "No one left behind: Inclusive federated learning over heterogeneous devices," in *Proc. KDD*, 2022, pp. 1–9.

[27] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, "Protection against reconstruction and its applications in private federated learning," 2018, *arXiv:1812.00984*.

[28] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 691–706.

[29] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Proc. NIPS*, 2019, pp. 14747–14756.

[30] N. Agarwal, A. T. Suresh, F. X. X. Yu, S. Kumar, and B. McMahan, "CpSGD: Communication-efficient and differentially-private distributed SGD," in *Proc. NIPS*, 2018, pp. 7564–7575.

[31] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 5, pp. 1333–1345, May 2018.

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

[33] H. Wang et al., "Attack of the tails: Yes, you really can backdoor federated learning," in *Proc. NIPS*, 2020, pp. 1–15.

[34] C. Xie, K. Huang, P. Chen, and B. Li, "DBA: Distributed backdoor attacks against federated learning," in *Proc. ICLR*, 2020, pp. 1–19.

[35] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2938–2948.

[36] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *Proc. ICML*, 2019, pp. 634–643.

[37] C. Fung, C. J. Yoon, and I. Beschastnikh, "The limitations of federated learning in Sybil settings," in *Proc. 23rd Int. Symp. Res. Attacks, Intrusions Defenses (RAID)*, 2020, pp. 301–316.

[38] Z. Sun, P. Kairouz, A. T. Suresh, and H. B. McMahan, "Can you really backdoor federated learning?" 2019, *arXiv:1911.07963*.

[39] J. Bernstein, J. Zhao, K. Azizzadenesheli, and A. Anandkumar, "SignSGD with majority vote is communication efficient and fault tolerant," in *Proc. ICLR*, 2019, pp. 1–20.

[40] P. Blanchard et al., "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Proc. NIPS*, 2017, pp. 119–129.

[41] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-IID private data," 2018, *arXiv:1811.11479*.

[42] N. Papernot, M. Abadi, U. Erlingsson, I. Goodfellow, and K. Talwar, "Semi-supervised knowledge transfer for deep learning from private training data," in *Proc. ICLR*, 2017, pp. 1–16.

[43] L. Lyu, "Privacy-preserving machine learning and data aggregation for Internet of Things," Ph.D. dissertation, Dept. Elect. Electron. Eng., Univ. Melbourne, Melbourne, VIC, Australia, 2018.

[44] L. Lyu et al., "Distributed privacy-preserving prediction," in *Proc. Int. Conf. Syst., Man, Cybern.*, 2020.

[45] L. Sun and L. Lyu, "Federated model distillation with noise-free differential privacy," in *Proc. IJCAI*, 2021.

[46] S. Truex et al., "A hybrid approach to privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Secur.*, 2019, pp. 1–11.

[47] Y. Liu, Y. Xie, and A. Srivastava, "Neural trojans," in *Proc. IEEE Int. Conf. Comput. Design (ICCD)*, Nov. 2017, pp. 45–48.

[48] B. G. Doan, E. Abbasnejad, and D. C. Ranasinghe, "Februus: Input purification defense against trojan attacks on deep neural network systems," in *Proc. Annu. Comput. Secur. Appl. Conf.*, 2020, pp. 897–912.

[49] S. Udeshi, S. Peng, G. Woo, L. Loh, L. Rawshan, and S. Chattopadhyay, "Model agnostic defence against backdoor attacks in machine learning," *IEEE Trans. Rel.*, vol. 71, no. 2, pp. 880–895, Jun. 2022.

[50] M. Villarreal-Vasquez and B. Bhargava, "ConFoc: Content-focus protection against trojan attacks on neural networks," 2020, *arXiv:2007.00711*.

[51] Y. Li, T. Zhai, B. Wu, Y. Jiang, Z. Li, and S. Xia, "Rethinking the trigger of backdoor attack," 2020, *arXiv:2004.04692*.

[52] B. Tran, J. Li, and A. Madry, "Spectral signatures in backdoor attacks," in *Proc. NIPS*, 2018, pp. 8000–8010.

[53] B. Chen et al., "Detecting backdoor attacks on deep neural networks by activation clustering," 2018, *arXiv:1811.03728*.

[54] D. Tang, X. Wang, H. Tang, and K. Zhang, "Demon in the variant: Statistical analysis of DNNs for robust backdoor contamination detection," in *Proc. 30th USENIX Secur. Symp. (USENIX Security)*, 2021, pp. 1541–1558.

[55] E. Soremekun, S. Udeshi, and S. Chattopadhyay, "Exposing backdoors in robust machine learning models," 2020, *arXiv:2003.00865*.

[56] A. Chan and Y.-S. Ong, "Poison as a cure: Detecting & neutralizing variable-sized backdoor attacks in deep neural networks," 2019, *arXiv:1911.08040*.

[57] E. Chou, F. Tramer, and G. Pellegrino, "SentiNet: Detecting localized universal attack against deep learning systems," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, May 2020, pp. 48–54.

[58] Y. Liu, X. Ma, J. Bailey, and F. Lu, "Reflection backdoor: A natural backdoor attack on deep neural networks," in *Proc. ECCV*. Springer, 2020, pp. 182–199.

[59] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Neural attention distillation: Erasing backdoor triggers from deep neural networks," in *Proc. ICLR*, 2021, pp. 1–19.

[60] Y. Li, X. Lyu, N. Koren, L. Lyu, B. Li, and X. Ma, "Anti-backdoor learning: Training clean models on poisoned data," in *Proc. NIPS*, 2021, pp. 14900–14912.

[61] G. Damaskinos, E. M. El Mhamdi, R. Guerraoui, A. H. A. Guirguis, and S. L. A. Rouault, "Aggregathor: Byzantine machine learning via robust gradient aggregation," in *Proc. SysML*, 2019, pp. 81–106.

[62] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *ACM Meas. Anal. Comput. Syst.*, vol. 1, no. 2, p. 44, 2017.

[63] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proc. ICML*, 2018, pp. 5650–5659.

[64] K. Pillutla, S. M. Kakade, and Z. Harchaoui, "Robust aggregation for federated learning," *IEEE Trans. Signal Process.*, vol. 70, pp. 1142–1154, 2022.

[65] M. S. Ozdayi, M. Kantarcioglu, and Y. R. Gel, "Defending against backdoors in federated learning with robust learning rate," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 10, 2021, pp. 9268–9276.

[66] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.

[67] P. Kairouz et al., "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*.

[68] Q. Li et al., "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Trans. Knowl. Data Eng.*, early access, Nov. 2, 2021, doi: 10.1109/TKDE.2021.3124599.

[69] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Gener. Comput. Syst.*, vol. 115, pp. 619–640, Feb. 2021.

[70] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowl.-Based Syst.*, vol. 216, Mar. 2021, Art. no. 106775.

[71] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 739–753.

[72] Y. Huang, S. Gupta, Z. Song, K. Li, and S. Arora, "Evaluating gradient inversion attacks and defenses in federated learning," in *Proc. NIPS*, 2021, pp. 7232–7241.

[73] V. Shejwalkar, A. Houmansadr, P. Kairouz, and D. Ramage, "Back to the drawing board: A critical evaluation of poisoning attacks on production federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2022, pp. 1354–1371.

[74] B. Biggio, B. Nelson, and P. Laskov, "Support vector machines under adversarial label noise," in *Proc. ACML*, 2011, pp. 97–112.

[75] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2017, pp. 603–618.

[76] C. Miao, Q. Li, H. Xiao, W. Jiang, M. Huai, and L. Su, "Towards data poisoning attacks in crowd sensing systems," in *Proc. 18th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Jun. 2018, pp. 111–120.

[77] C. Miao, Q. Li, L. Su, M. Huai, W. Jiang, and J. Gao, "Attack under disguise: An intelligent data poisoning attack mechanism in crowdsourcing," in *Proc. World Wide Web Conf.*, 2018, pp. 13–22.

[78] H. Zhang et al., "Data poisoning attack against knowledge graph embedding," 2019, *arXiv:1904.12052*.

[79] G. Sun, Y. Cong, J. Dong, Q. Wang, L. Lyu, and J. Liu, "Data poisoning attacks on federated machine learning," *IEEE Internet Things J.*, vol. 9, no. 13, pp. 11365–11375, Jul. 2022.

[80] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proc. 29th USENIX Secur. Symp. (USENIX)*, 2020, pp. 1605–1622.

[81] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," 2012, *arXiv:1206.6389*.

[82] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction APIs," in *Proc. USENIX Secur. Symp.*, 2016, pp. 601–618.

[83] X. He, L. Lyu, L. Sun, and Q. Xu, "Model extraction and adversarial transferability, your BERT is vulnerable!" in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2021, pp. 2006–2012.

[84] Q. Xu, X. He, L. Lyu, L. Qu, and G. Haffari, "Beyond model extraction: Imitation attack for black-box NLP APIs," in *Proc. COLING*, 2022, pp. 1–12.

[85] X. He et al., "CATER: Intellectual property protection on text generation APIs via conditional watermarks," in *Proc. NIPS*, 2022, pp. 1–19.

[86] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 3–18.

[87] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar, "Can machine learning be secure?" in *Proc. ICCS*, 2006, pp. 16–25.

[88] B. Biggio et al., "Evasion attacks against machine learning at test time," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2013, pp. 387–402.

[89] C. Szegedy et al., "Intriguing properties of neural networks," 2013, *arXiv:1312.6199*.

[90] G. Zizzo, A. Rawat, M. Sinn, and B. Buesser, "FAT: Federated adversarial training," 2020, *arXiv:2012.01791*.

[91] J. Hong, H. Wang, Z. Wang, and J. Zhou, "Federated robustness propagation: Sharing robustness in heterogeneous federated learning," 2021, *arXiv:2106.10196*.

[92] D. Shah, P. Dube, S. Chakraborty, and A. Verma, "Adversarial training in communication constrained federated learning," 2021, *arXiv:2103.01319*.

[93] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. ICLR*, 2018, pp. 1–28.

[94] L. Su and J. Xu, "Securing distributed gradient descent in high dimensional statistical learning," 2018, *arXiv:1804.10140*.

[95] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Commun. ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[96] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2015, pp. 1322–1333.

[97] S. Guo, C. Xie, J. Li, L. Lyu, and T. Zhang, "Threats to pre-trained language models: Survey and taxonomy," 2022, *arXiv:2202.06862*.

[98] B. I. P. Rubinstein et al., "ANTIDOTE: Understanding and defending against poisoning of anomaly detectors," in *Proc. 9th ACM SIGCOMM Internet Measurement Conf.*, 2009, pp. 1–14.

[99] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2018, pp. 19–35.

[100] C. Xie, O. Koyejo, and I. Gupta, "Fall of empires: Breaking Byzantine-tolerant SGD by inner product manipulation," in *Uncertainty in Artificial Intelligence*. PMLR, 2020, pp. 261–270.

[101] B. Nelson et al., "Exploiting machine learning to subvert your spam filter," in *Proc. LEET*, vol. 8, 2008, pp. 1–9.

[102] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar, "Adversarial machine learning," in *Proc. 4th ACM Workshop Secur. Artif. Intell.*, 2011, pp. 43–58.

[103] X. Chen, C. Liu, B. Li, K. Lu, and D. Song, "Targeted backdoor attacks on deep learning systems using data poisoning," 2017, *arXiv:1712.05526*.

[104] T. Gu, B. Dolan-Gavitt, and S. Garg, "BadNets: Identifying vulnerabilities in the machine learning model supply chain," 2017, *arXiv:1708.06733*.

[105] A. Shafahi et al., "Poison frogs! Targeted clean-label poisoning attacks on neural networks," in *Proc. NIPS*, 2018, pp. 6103–6113.

[106] Y. Liu et al., "Trojaning attack on neural networks," in *Proc. NDSS*, 2018, pp. 1–17.

[107] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, Jul. 1982.

[108] C. Chen, J. Zhang, A. K. H. Tung, M. Kankanhalli, and G. Chen, "Robust federated recommendation system," 2020, *arXiv:2006.08259*.

[109] G. Baruch, M. Baruch, and Y. Goldberg, "A little is enough: Circumventing defenses for distributed learning," in *Proc. NIPS*, 2019, pp. 8632–8642.

[110] T. A. Nguyen and A. Tran, "Input-aware dynamic backdoor attack," in *Proc. NIPS*, 2020, pp. 3454–3464.

[111] L. Muñoz-González et al., "Towards poisoning of deep learning algorithms with back-gradient optimization," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, 2017, pp. 27–38.

[112] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *Proc. ICML*, 2017, pp. 1885–1894.

[113] S. Zhao, X. Ma, X. Zheng, J. Bailey, J. Chen, and Y.-G. Jiang, "Clean-label backdoor attacks on video recognition models," in *Proc. CVPR*, Jun. 2020, pp. 14443–14452.

[114] Y. Zeng, M. Pan, H. A. Just, L. Lyu, M. Qiu, and R. Jia, "Narcissus: A practical clean-label backdoor attack with limited information," 2022, *arXiv:2204.05255*.

[115] J. R. Douceur, "The Sybil attack," in *Proc. Int. Workshop Peer-to-Peer Syst.*, 2002, pp. 251–260.

[116] Y. Aono, T. Hayashi, L. T. Phong, and L. Wang, "Scalable and secure logistic regression via homomorphic encryption," in *Proc. 6th ACM Conf. Data Appl. Secur. Privacy*, Mar. 2016, pp. 142–144.

[117] M. Kim, Y. Song, S. Wang, Y. Xia, and X. Jiang, "Secure logistic regression based on homomorphic encryption: Design and evaluation," *JMIR Med. Informat.*, vol. 6, no. 2, p. e19, Apr. 2018.

[118] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*.

[119] T. T. Nguyên, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin, "Collecting and analyzing data from smart device users with local differential privacy," 2016, *arXiv:1606.05053*.

[120] N. Wang et al., "Collecting and analyzing multidimensional data with local differential privacy," in *Proc. IEEE 35th Int. Conf. Data Eng. (ICDE)*, Apr. 2019, pp. 638–649.

[121] Y. Zhao et al., "Local differential privacy-based federated learning for Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8836–8853, Nov. 2020.

[122] L. Sun, J. Qian, X. Chen, and P. S. Yu, "LDP-FL: Practical private aggregation in federated learning with local differential privacy," in *Proc. IJCAI*, 2021, pp. 1–9.

[123] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "LDP-fed: Federated learning with local differential privacy," in *Proc. 3rd ACM Int. Workshop Edge Syst., Analytics Netw.*, Apr. 2020, pp. 61–66.

[124] L. Lyu, "Lightweight crypto-assisted distributed differential privacy for privacy-preserving distributed learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[125] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 19–38.

[126] P. Paillier et al., "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Eurocrypt*, vol. 99, 1999, pp. 223–238.

[127] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 4, pp. 469–472, Jul. 1985.

[128] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st Annu. ACM Symp. Symp. theory Comput. (STOC)*, 2009, pp. 169–178.

[129] A. C. Yao, "Protocols for secure computations," in *Proc. 23rd Annu. Symp. Found. Comput. Sci.*, Nov. 1982, pp. 160–164.

[130] D. Demmler, T. Schneider, and M. Zohner, "ABY—A framework for efficient mixed-protocol secure two-party computation," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2015, pp. 1–15.

[131] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory Cryptography Conf.*, 2006, pp. 265–284.

[132] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, Aug. 2014.

[133] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Proc. Annu. Cryptol. Conf.*, 2012, pp. 643–662.

[134] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[135] S. Goryczka and L. Xiong, "A comprehensive comparison of multiparty secure additions with differential privacy," *IEEE Trans. Dependable Sec. Comput.*, vol. 14, no. 5, pp. 463–477, Oct. 2015.

[136] M. S. Riazi, C. Weinert, O. Tkachenko, E. M. Songhori, T. Schneider, and F. Koushanfar, "Chameleon: A hybrid secure computation framework for machine learning applications," in *Proc. Asia Conf. Comput. Commun. Secur.*, May 2018, pp. 707–721.

[137] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption," in *Proc. ACM SIGMOD Int. Conf. Manage. data*, Jun. 2010, pp. 735–746.

[138] E. Shi, H. Chan, E. Rieffel, R. Chow, and D. Song, "Privacy-preserving aggregation of time-series data," in *Proc. Annu. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, 2011, pp. 1–17.

[139] G. Ács and C. Castelluccia, "I have a dream! (differentially private smart metering)," in *Proc. Int. Workshop Inf. Hiding*. Berlin, Germany: Springer, 2011, pp. 118–132.

[140] L. Lyu, K. Nandakumar, B. Rubinstein, J. Jin, J. Bedo, and M. Palaniswami, "PPFA: Privacy preserving fog-enabled aggregation in smart grid," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3733–3744, Aug. 2018.

[141] V. Chen, V. Pastro, and M. Raykova, "Secure computation for machine learning with SPDZ," 2019, *arXiv:1901.00329*.

[142] P. Mohassel and P. Rindal, "ABY 3: A mixed protocol framework for machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Oct. 2018, pp. 35–52.

[143] J. Li, M. Khodak, S. Caldas, and A. Talwalkar, "Differentially private meta-learning," 2019, *arXiv:1909.05830*.

[144] N. Papernot, S. Song, I. Mironov, A. Raghunathan, K. Talwar, and U. Erlingsson, "Scalable private learning with pate," in *Proc. ICLR*, 2018, pp. 1–34.

[145] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," 2020, *arXiv:2003.02133*.

[146] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *Proc. 54th IEEE Annu. Symp. Found. Comput. Sci.*, Oct. 2013, pp. 429–438.

[147] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2006, pp. 486–503.

[148] L. Lyu, X. He, and Y. Li, "Differentially private representation for NLP: Formal guarantee and an empirical study on privacy and fairness," in *Proc. Findings Assoc. Comput. Linguistics (EMNLP)*, 2020, pp. 1–11.

[149] M. Yang, L. Lyu, J. Zhao, T. Zhu, and K.-Y. Lam, "Local differential privacy and its applications: A comprehensive survey," 2020, *arXiv:2008.03686*.

[150] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *J. Amer. Statist. Assoc.*, vol. 60, no. 309, pp. 63–69, 1965.

[151] U. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proc. CCS*, 2014, pp. 1054–1067.

[152] T. H. Chan, E. Shi, and D. Song, "Optimal lower bound for differentially private multi-party aggregation," in *Proc. Eur. Symp. Algorithms*, 2012, pp. 277–288.

[153] T. Chan, K.-M. Chung, B. M. Maggs, and E. Shi, "Foundations of differentially oblivious algorithms," in *Proc. 30th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2019, pp. 2448–2467.

[154] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2015, pp. 1310–1321.

[155] J. Hamm, Y. Cao, and M. Belkin, "Learning privately from multiparty data," in *Proc. ICML*, 2016, pp. 555–563.

[156] L. Lyu and C.-H. Chen, "Differentially private knowledge distillation for mobile analytics," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 1809–1812.

[157] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *J. Mach. Learn. Res.*, vol. 12, pp. 1069–1109, Mar. 2011.

[158] B. Han, I. W. Tsang, and L. Chen, "On the convergence of a family of robust losses for stochastic gradient descent," in *Machine Learning and Knowledge Discovery in Databases*. 2016, pp. 665–680.

[159] S. Shen, S. Tople, and P. Saxena, "A uror: Defending against poisoning attacks in collaborative deep learning systems," in *Proc. 32nd Annu. Conf. Comput. Secur. Appl.*, Dec. 2016, pp. 508–519.

[160] R. Guerraoui et al., "The hidden vulnerability of distributed learning in byzantium," in *Proc. ICML*, 2018, pp. 3521–3530.

[161] C. Wu, X. Yang, S. Zhu, and P. Mitra, "Mitigating backdoor attacks in federated learning," 2020, *arXiv:2011.01767*.

[162] C. Xie, M. Chen, P.-Y. Chen, and B. Li, "CRFL: Certifiably robust federated learning against backdoor attacks," in *Proc. ICML*, 2021, pp. 11372–11382.

[163] L. Chen, H. Wang, Z. Charles, and D. Papailiopoulos, "Draco: Byzantine-resilient distributed training via redundant gradients," in *Proc. ICML*, 2018, pp. 903–912.

[164] C. Xie, O. Koyejo, and I. Gupta, "Generalized Byzantine-tolerant SGD," 2018, *arXiv:1802.10116*.

[165] D. Alistarh, Z. Allen-Zhu, and J. Li, "Byzantine stochastic gradient descent," in *Proc. NIPS*, 2018, pp. 4613–4623.

[166] X. Xu and L. Lyu, "A reputation mechanism is all you need: Collaborative fairness and adversarial robustness in federated learning," 2020, *arXiv:2011.10464*.

[167] J. Steinhardt, M. Charikar, and G. Valiant, "Resilience: A criterion for learning in the presence of arbitrary outliers," 2017, *arXiv:1703.04940*.

[168] Y. Li, Y. Jiang, Z. Li, and S.-T. Xia, "Backdoor learning: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 22, 2022, doi: 10.1109/TNNLS.2022.3182979.

[169] B. Wang et al., "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 707–723.

[170] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Deepinspect: A black-box trojan detection and mitigation framework for deep neural networks," in *IJCAI*, vol. 2019, pp. 4658–4664.

[171] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses*, 2018, pp. 273–294.

[172] T. J. L. Tan and R. Shokri, "Bypassing backdoor detection algorithms in deep learning," in *Proc. IEEE Eur. Symp. Secur. Privacy (EuroS&P)*, Sep. 2020, pp. 175–183.

[173] P. Zhao, P.-Y. Chen, P. Das, K. N. Ramamurthy, and X. Lin, "Bridging mode connectivity in loss landscapes and adversarial robustness," 2020, *arXiv:2005.00060*.

[174] Y. Yao, H. Li, H. Zheng, and B. Y. Zhao, "Latent backdoor attacks on deep neural networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 2041–2055.

[175] S. Andreina, G. A. Marson, H. Mollering, and G. Karame, "BaFFLe: Backdoor detection via feedback-based federated learning," in *Proc. IEEE 41st Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2021, pp. 852–863.

[176] H. Chang, V. Shejwalkar, R. Shokri, and A. Houmansadr, "Cronus: Robust and heterogeneous collaborative learning with black-box knowledge transfer," 2019, *arXiv:1912.11279*.

[177] C. Chen, L. Lyu, H. Yu, and G. Chen, "Practical attribute reconstruction attack against federated learning," *IEEE Trans. Big Data*, early access, Mar. 15, 2022, doi: 10.1109/TBDATA.2022.3159236.

[178] J. Jin, J. Ren, Y. Zhou, L. Lyu, J. Liu, and D. Dou, "Accelerated federated learning with decoupled adaptive optimization," in *Proc. ICML*, 2022, pp. 10298–10322.

[179] Y. Wang, X. Ma, J. Bailey, J. Yi, B. Zhou, and Q. Gu, "On the convergence and robustness of adversarial training," in *Proc. ICML*, vol. 1, 2019, p. 2.

[180] Y. Wang, D. Zou, J. Yi, J. Bailey, X. Ma, and Q. Gu, "Improving adversarial robustness requires revisiting misclassified examples," in *Proc. ICLR*, 2019, pp. 1–14.

[181] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, and A. Madry, "Robustness may be at odds with accuracy," in *Proc. ICLR*, 2019, pp. 1–14.

[182] L. Lyu, J. C. Bezdek, X. He, and J. Jin, "Fog-embedded deep learning for the Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4206–4215, Jul. 2019.

[183] L. Lyu et al., "Towards fair and privacy-preserving federated deep models," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 11, pp. 2524–2541, Mar. 2020.

[184] L. Lyu, Y. Li, K. Nandakumar, J. Yu, and X. Ma, "How to democratise and protect AI: Fair and differentially private decentralised deep learning," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1003–1017, Apr. 2020.

[185] L. Lyu, J. C. Bezdek, J. Jin, and Y. Yang, "FORESEEN: Towards differentially private deep inference for intelligent Internet of Things," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2418–2429, Oct. 2020.

[186] X. Pan, M. Zhang, S. Ji, and M. Yang, "Privacy risks of general-purpose language models," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2020, pp. 1314–1331.

[187] T. Dong, B. Zhao, and L. Lyu, "Privacy for free: How does dataset condensation help privacy?" in *Proc. ICML*, 2022, pp. 1–19.

[188] C. Chen, Y. Liu, X. Ma, and L. Lyu, "CalFAT: Calibrated federated adversarial training with label skewness," in *Proc. NIPS*, 2022, pp. 1–16.

[189] X. He, Q. Xu, L. Lyu, F. Wu, and C. Wang, "Protecting intellectual property of language generation APIs with lexical watermark," in *Proc. AAAI*, 2022, pp. 1–9.

[190] N. Truong, K. Sun, S. Wang, F. Guitton, and Y. Guo, "Privacy preservation in federated learning: An insightful survey from the GDPR perspective," *Comput. Secur.*, vol. 110, Nov. 2021, Art. no. 102402.

[191] K. Cheng et al., "SecureBoost: A lossless federated learning framework," *IEEE Intell. Syst.*, vol. 36, no. 6, pp. 87–98, Dec. 2021.

[192] Z. Tian, R. Zhang, X. Hou, J. Liu, and K. Ren, "FederBoost: Private federated learning for GBDT," 2020, *arXiv:2011.02796*.

[193] X. Jin, P.-Y. Chen, C.-Y. Hsu, C.-M. Yu, and T. Chen, "Catastrophic data leakage in vertical federated learning," in *Proc. NIPS*, vol. 2021, pp. 994–1006.

[194] X. Xu, L. Lyu, X. Ma, C. Miao, C. S. Foo, and B. K. H. Low, "Gradient driven rewards to guarantee fairness in collaborative machine learning," in *Proc. NIPS*, 2021, pp. 16104–16117.

[195] L. Lyu, X. Xu, Q. Wang, and H. Yu, "Collaborative fairness in federated learning," in *Federated Learning*. Springer, 2020, pp. 189–204.

[196] Q. Yang, L. Fan, and H. Yu, *Federated Learning: Privacy Incentive*. Springer, 2020.

[197] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y.-C. Liang, and D. I. Kim, "Incentive design for efficient federated learning in mobile networks: A contract theory approach," in *Proc. IEEE VTS Asia Pacific Wireless Commun. Symp. (APWCS)*, Aug. 2019, pp. 1–5.

[198] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Feb. 2020.

[199] S. Warnat-Herresthal et al., "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, 2021.

[200] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang, and D. Niyato, "Federated learning for 6G communications: Challenges, methods, and future directions," *China Commun.*, vol. 17, no. 9, pp. 105–118, Sep. 2020.

[201] N. Guha, A. Talwalkar, and V. Smith, "One-shot federated learning," 2019, *arXiv:1902.11175*.

[202] Q. Li, B. He, and D. Song, "Practical one-shot federated learning for cross-silo setting," 2020, *arXiv:2010.01017*.

[203] D. K. Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 2611–2620.

[204] J. Zhang, C. Chen, B. Li, L. Lyu, S. Wu, J. Xu, S. Ding, and C. Wu, "A practical data-free approach to one-shot federated learning with heterogeneity," 2021, *arXiv:2112.12371*.

[205] M. Barreno, B. Nelson, A. D. Joseph, and J. D. Tygar, "The security of machine learning," *Mach. Learn.*, vol. 81, no. 2, pp. 121–148, 2010.

[206] J. Steinhardt, P. W. W. Koh, and P. S. Liang, "Certified defenses for data poisoning attacks," in *Proc. NIPS*, 2017, pp. 3517–3529.

Lingjuan Lyu (Member, IEEE) received the Ph.D. degree from The University of Melbourne, Melbourne, VIC, Australia.

She is currently a Senior Research Scientist and the Team Leader with Sony AI, Tokyo, Japan. She had published over 50 papers in top conferences and journals, including NeurIPS, ICLR, ICML, Nature, and so on. Her current research interest is trustworthy artificial intelligence (AI).

Dr. Lyu was a winner of the IBM Ph.D. Fellowship Worldwide. Her works had won several best paper awards and oral presentations from top conferences.

Han Yu (Senior Member, IEEE) received the Ph.D. degree from the School of Computer Science and Engineering, NTU, Singapore.

He is a Nanyang Assistant Professor (NAP) in the School of Computer Science and Engineering (SCSE), Nanyang Technological University (NTU), . He held the prestigious Lee Kuan Yew Post-Doctoral Fellowship (LKY PDF), from 2015 to 2018. He has published over 200 research papers and book chapters in leading international conferences and journals. He is a coauthor of the book Federated Learning - the first monograph on the topic of federated learning. His research focuses on trustworthy federated learning.

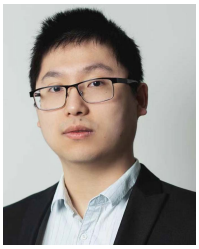
Dr.Yu is a Senior Member of CCF. His research works have won multiple awards from conferences and journals.

Xingjun Ma received the bachelor's degree from Jilin University, Changchun, China, the master's degree from Tsinghua University, Beijing, China, and the Ph.D. degree from The University of Melbourne, Melbourne, VIC, Australia.

He was a Lecturer with the School of Information Technology, Deakin University, Geelong, VIC, Australia. He was a Post-Doctoral Research Fellow with the School of Computing and Information Systems, The University of Melbourne. He is currently an Associate Professor of computer science with Fudan University, Shanghai, China. He works in the areas of adversarial machine learning, deep learning, artificial intelligence (AI) security, and data privacy.

Chen Chen received the B.S. degree in computer science from the Chu Kochen Honors College, Zhejiang University, Hangzhou, China, in 2017, where he is currently pursuing the Ph.D. degree with the College of Computer Science.

He is currently an Intern with Sony AI, Tokyo, Japan. His research interests include federated learning, adversarial training, multilabel learning, and recommendation systems.



Lichao Sun received the Ph.D. degree in computer science from the University of Illinois Chicago, Chicago, IL, USA, in 2020, under the supervision of Prof. Philip S. Yu.

He is currently an Assistant Professor with the Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA, USA. He has published more than 45 research articles in top conferences and journals, such as CCS, USENIX-Security, NeurIPS, KDD, ICLR, the Advancement of AI (AAAI), the International Joint Conference on AI (IJCAI), ACL, NAACL, TII, TNNLS, and TMC. His research interests include security and privacy in deep learning and data mining. He mainly focuses on artificial intelligence (AI) security and privacy, social networks, and NLP applications.

Jun Zhao (Member, IEEE) received the bachelor's degree from Shanghai Jiao Tong University, Shanghai, China, in July 2010, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University (CMU), Pittsburgh, PA, USA, in May 2015, affiliating with CMU's renowned CyLab Security & Privacy Institute.

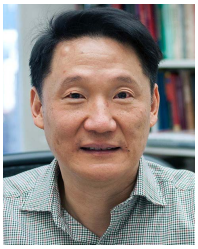
He is currently an Assistant Professor with the School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore. Before joining NTU first as a Post-Doctoral Researcher with Xiaokui Xiao and then as a Faculty Member, he was a Post-Doctoral Researcher and Arizona Computing PostDoc Best Practices Fellow with Arizona State University, Tempe, AZ, USA. His research interests include communication networks, security/privacy, and artificial intelligence (AI).

Dr. Zhao's coauthored papers received the Best Paper Award (IEEE TRANSACTIONS) by the IEEE Vehicular Society (VTS) Singapore Chapter in 2019 and the Best Paper Award in the EAI International Conference on 6G for Future Wireless Networks (EAI 6GN) 2020.

Qiang Yang (Fellow, IEEE) received the B.Sc. degree in astrophysics from Peking University, Beijing, China, in 1982, and the M.Sc. degree in astrophysics and the Ph.D. degree in computer science from the University of Maryland, College Park, MD, USA, in 1985 and 1989, respectively.

He was a Faculty Member with the University of Waterloo, Waterloo, ON, Canada, from 1989 to 1995, and Simon Fraser University, Burnaby, BC, Canada, from 1995 to 2001. He was the Founding Director of Huawei's Noah's Ark Lab, Hong Kong, from 2012 to 2014 and a Co-Founder of 4Paradigm Corporation, Beijing, an artificial intelligence (AI) platform company. He is currently the Head of the AI Department and the Chief AI Officer of WeBank, Shenzhen, China, and a Chair Professor with the Computer Science and Engineering (CSE) Department, The Hong Kong University of Science and Technology (HKUST), Hong Kong, where he was a former Head of the CSE Department and the Founding Director of the Big Data Institute from 2015 to 2018. He is the author of several books, including *Intelligent Planning* (Springer), *Crafting Your Research Future* (Morgan & Claypool), and *Constraint-Based Design Recovery for Software Engineering* (Springer). His research interests include AI, machine learning, and data mining, especially in transfer learning, automated planning, federated learning, and case-based reasoning.

Dr. Yang has served as an Executive Council Member of the Advancement of AI (AAAI) from 2016 to 2020. He is a fellow of several international societies, including ACM, AAAI, IEEE, IAPR, and AAAS. He was a recipient of several awards, including the 2004/2005 ACM KDDCUP Championship, the ACM SIGKDD Distinguished Service Award in 2017, and the AAAI Innovative AI Applications Awards in 2018 and 2020. He was the Founding Editor-in-Chief of the *ACM Transactions on Intelligent Systems and Technology* (ACM TIST) and IEEE TRANSACTIONS ON BIG DATA (IEEE TBD). He has served as the President of the International Joint Conference on AI (IJCAI) from 2017 to 2019.



Philip S. Yu (Life Fellow, IEEE) received the B.S. degree in electrical engineering (E.E.) from the National Taiwan University, New Taipei, Taiwan, in 1992, the M.S. and Ph.D. degrees in E.E. from Stanford University, Stanford, CA, USA, in 1976 and 1978, respectively, and the M.B.A. degree from New York University, New York, NY, USA, in 1982.

He is currently a Distinguished Professor of computer science with the University of Illinois Chicago (UIC), Chicago, IL, USA, and also holds the Wexler Chair in Information Technology. Before joining UIC, he was with IBM, USA, where he was the Manager of the Software Tools and Techniques Department, Watson Research Center. He has published more than 1200 papers in refereed journals and conferences. He holds or has applied for more than 300 U.S. patents. His research interest is on big data, including data mining, data stream, database, and privacy.

Dr. Yu is a fellow of the ACM. He was a recipient of the ACM SIGKDD 2016 Innovation Award for his influential research and scientific contributions to mining, fusion, and anonymization of big data, the IEEE Computer Society's 2013 Technical Achievement Award for pioneering and fundamentally innovative contributions to the scalable indexing, querying, searching, mining, and anonymization of big data, and the Research Contributions Award from IEEE International Conference on Data Mining (ICDM) in 2003 for his pioneering contributions to the field of data mining. He received the ICDM 2013 10-Year Highest-Impact Paper Award and the EDBT Test of Time Award in 2014. He was the Editor-in-Chief of *ACM Transactions on Knowledge Discovery from Data* from 2011 to 2017 and IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING from 2001 to 2004.