Note: Right of the Bat, this paper focussed on "Poisoning" attacks. these are not failures, but nodes that "on purpose" Disrupt the FL-process. this paper is good for Background section because it Discusses core FL concepts. this paper is also good for Related work section because it discusses a different form of Disruption/failure namely poisoning atk. and a way to measure the Robustness in this Regard.

Array 23 (2024) 100360

Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/locate/array

DART: A Solution for decentralized federated learning model robustness analysis

Chao Feng [a,*], Alberto Huertas Celdrán [a], Jan von der Assen [a], Enrique Tomás Martínez Beltrán [b], Gérôme Bovet [c], Burkhard Stiller [a]

[a] Communication Systems Group CSG, Department of Informatics IfI, University of Zurich UZH, 8050 Zürich, Switzerland
[b] Department of Information and Communications Engineering, University of Murcia, 30100 Murcia, Spain
[c] Cyber-Defence Campus within Armasuisse Science & Technology, 3602 Thun, Switzerland

ARTICLE INFO

Keywords:
Decentralized federated learning
Poisoning attack
Cybersecurity
Model robustness

ABSTRACT

Federated Learning (FL) has emerged as a promising approach to address privacy concerns inherent in Machine Learning (ML) practices. However, conventional FL methods, particularly those following the Centralized FL (CFL) paradigm, utilize a central server for global aggregation, which exhibits limitations such as bottleneck and single point of failure. To address these issues, the Decentralized FL (DFL) paradigm has been proposed, which removes the client–server boundary and enables all participants to engage in model training and aggregation tasks. Nevertheless, as CFL, DFL remains vulnerable to adversarial attacks, notably poisoning attacks that undermine model performance. While existing research on model robustness has predominantly focused on CFL, there is a noteworthy gap in understanding the model robustness of the DFL paradigm. In this paper, a thorough review of poisoning attacks targeting the model robustness in DFL systems, as well as their corresponding countermeasures, are presented. Additionally, a solution called *DART* is proposed to evaluate the robustness of DFL models, which is implemented and integrated into a DFL platform. Through extensive experiments, this paper compares the behavior of CFL and DFL under diverse poisoning attacks, pinpointing key factors affecting attack spread and effectiveness within the DFL. It also evaluates the performance of different defense mechanisms and investigates whether defense mechanisms designed for CFL are compatible with DFL. The empirical results provide insights into research challenges and suggest ways to improve the robustness of DFL models for future research.

Contents

* Corresponding author.
E-mail address: cfeng@ifi.uzh.ch (C. Feng).

https://doi.org/10.1016/j.array.2024.100360
Received 29 May 2024; Received in revised form 26 August 2024; Accepted 27 August 2024
Available online 30 August 2024
2590-0056/© 2024 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

The rapid expansion of the Internet-of-Things (IoT) has led to the interconnection of more than 15 billion devices, resulting in the generation of a staggering 330 Exabytes of data on a daily basis [1]. Machine Learning (ML) has become a crucial tool for efficiently handling and analyzing such immense datasets [2]. The ML pipeline involves various stages, such as data collection, centralization, preprocessing, training, and inference. However, data aggregation into a single server or data center presents challenges, particularly in IoT environments where data is collected and stored in a distributed manner [3]. Centralizing data is challenging due to network and storage limitations, and sharing data online raises security and privacy concerns, users are wary of sharing sensitive information from IoT devices [4]. Consequently, there is a growing research interest in designing an innovative ML paradigm that effectively tackles the challenges related to data processing and privacy preservation.

Federated Learning (FL) is a privacy-preserving ML paradigm where data is distributed and retained locally across clients (*e.g.,* IoT devices) instead of centralized in a single location [5]. As illustrated in the left side of Fig. 1, the local dataset in each client is employed to feed and train the local model algorithm and evaluate the model outcomes. These local models are then sent to a central aggregation server, where a global model is generated and distributed back to the clients for further training. This iterative process continues until the global model reaches convergence or a predefined number of aggregation rounds is reached. Throughout this federated procedure, clients prioritize privacy by only transmitting the models over the network while keeping the underlying data securely stored on their devices. Additionally, the smaller size of the models compared to the raw data helps overcome bandwidth and storage limitations. Thus, FL is well-suited for implementation in distributed networks that prioritize privacy preservation, such as healthcare and autonomous driving sections [6]. Nevertheless, the incorporation of the central aggregation server, referred to as Centralized FL (CFL), introduces new risks to the system. The central aggregator becomes a single-point-of-failure risk of the CFL paradigm, leading to network congestion and processing bottleneck [3].

To cope with the single-point-of-failure risk of the CFL paradigm, a decentralized approach known as Decentralized FL (DFL) has been proposed [3]. The DFL paradigm eliminates the distinction between servers and clients, allowing all participants to serve as trainers and aggregators, as shown on the right side of Fig. 1. In this approach, participants train models using their own local data and then exchange these models with other participants through the network. Local model aggregation occurs within each participant, incorporating knowledge

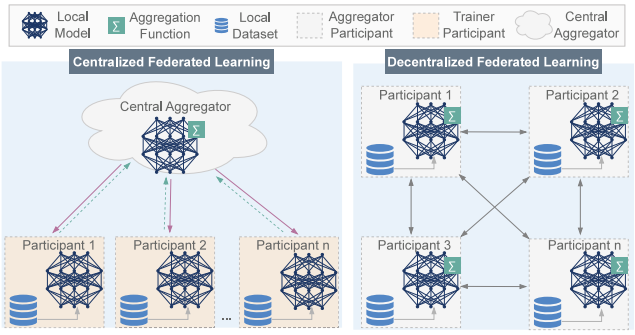

Fig. 1. Federation Aggregation Architecture of FL.

shared by others while ensuring that data privacy is preserved as only the models are shared. This decentralization allows the FL process to be fully autonomous and more resistant to network failures. Moreover, the DFL system offers increased stability by avoiding the risks associated with a single aggregator, distinguishing it from the CFL paradigm. Besides, the DFL paradigm allows for flexible interconnection using various network topologies, providing greater adaptability.

However, in addition to the benefits that CFL and DFL offer, the inherently distributed nature of this FL system makes it vulnerable to poisoning attacks [7]. These attacks involve malicious participants manipulating the training data or inserting malicious elements into their local models, resulting in inaccurate outputs [8]. These compromised models are distributed across the FL system and aggregated with benign models. This aggregation results in declining performance for benign models, reducing their robustness. These poisoning attacks can target both CFL and DFL systems. Therefore, there is a growing body of research dedicated to protecting the model robustness of FL systems and mitigating the detrimental effects of poisoning attacks [9].

Given the significance of poisoning attacks on DFL, it is crucial to prioritize the strength and security of the models. However, existing studies predominantly focus on the impact of poisoning attacks on CFL, indicating the need for greater attention to safeguarding DFL systems against such attacks. Moreover, there is a scarcity of practical tools available for effectively analyzing the robustness of DFL. Therefore, this work proposes and implements a DFL model robustness analysis solution. The contributions of this work can be summarized as follows:

- An analysis of the poisoning attacks targeting DFL. This paper offers a comprehensive overview of poisoning attacks, classifying and examining these attacks from the perspectives of both attack intention and attack strategy.

- An overview of existing defenses against poisoning attacks. This paper comprehensively reviews defense mechanisms to protect FL (both CFL and DFL) from poisoning attacks. The existing defense mechanisms are thoroughly examined, categorized, and analyzed from various perspectives.
- Design and prototype a model robustness analysis solution for DFL. This paper proposes a model robustness analysis solution for DFL, called *DART*. *DART* comprises an attack component that enables the execution of untargeted and targeted poisoning attacks, as well as a defense component that incorporates multiple defense mechanisms to protect DFL models from poisoning attacks. Meanwhile, *DART* is prototypically implemented and integrated into a real DFL platform to benchmark the performance of different attacks and the effectiveness of defense mechanisms.
- Experimental comparison of the model robustness between CFL and DFL and benchmark of the effectiveness of defense mechanisms. This paper conducts a series of experiments to compare the behavior and evaluate the robustness of CFL and DFL paradigms when subjected to targeted and untargeted attacks in MNIST, FashionMNIST, and Cifar10 datasets. Besides, this study conducts extensive experiments to benchmark the effectiveness of various defense mechanisms and offers recommendations for selecting the optimal defense mechanism.
- Analysis of the challenges and opportunities in the field of mode robustness. The paper delves into the learning lessons and highlights the research challenges from the literature review and experimental analysis while pinpointing the potential avenues for exploration in the DFL model robustness field.

In summary, the primary innovation of this work lies in its comprehensive review of existing literature on model robustness in DFL, as well as its empirical assessment of defense mechanisms against poisoning attacks. Then, this paper argues that DFL overlay network topology plays a crucial role in model robustness, underscoring the importance of considering topology when designing novel attacks or defenses. In addition, it analyzes the potential issues and challenges for improving the security of the DFL model against poisoning attacks in real-world scenarios, including both offensive and defensive considerations. Finally, this work offers recommendations for improving the robustness of the model in DFL and proposes practical roadmaps for implementation.

The subsequent sections of this paper are structured as follows: Section 2 presents an introduction to the relevant literature, while Section 3 provides a comprehensive overview of poisoning attacks and the corresponding defense mechanisms. Afterwards, Section 4 introduces the framework proposed in the paper, encompassing the overarching architecture, conception, and implementation, as well as the integration with a DFL platform. The experimental study is detailed in Section 5. Lastly, Section 6 discusses lessons learned, challenges encountered, and potential research opportunities, while Section 7 offers concluding remarks.

2. Related work

There have been a few overview and survey articles for FL model robustness studies, discussing both theoretical and practical considerations related to adversarial attacks and defense mechanisms. This section conducts a comparative analysis of these articles and presents an overview in Table 1.

[16] analyzed the security and privacy risks faced by FL from the perspectives of the sources of vulnerabilities and the types of adversarial attacks. A comparison was made between FL and other distributed ML methods regarding their vulnerability to adversarial attacks. The authors analyzed both CFL and fully-connected DFL architectures within the FL framework. However, the analysis was predominantly qualitative, lacking quantitative assessment and neglecting to compare the distinctions between CFL and DFL. Similarly, [9] presented a comprehensive assessment of the risks and threats confronting FL, outlining the impact of potential attacks. Furthermore, this survey presented an overview of defensive measures and assessed their susceptibility to compromise.

Table 1
Comparison of attributes among existing robustness analysis of FL research.

Research	CFL	DFL	Defense included	Overview of attacks	Experimental analysis
[10] 2022	✓	✓	✓	✓	x
[11] 2021	✓	x	✓	✓	✓
[12] 2022	✓	x	✓	✓	x
[13] 2021	✓	x	✓	✓	x
[14] 2023	✓	x	✓	✓	x
[15] 2022	✓	x	✓	✓	x
[9] 2022	✓	x	✓	✓	x
[16] 2021	✓	✓	✓	✓	x
[17] 2023	✓	x	x	✓	x
[18] 2022	✓	x	✓	✓	x
[4] 2023	✓	x	✓	✓	✓
[19] 2022	✓	x	✓	x	x
[8] 2023	✓	x	✓	✓	x
[20] 2021	✓	x	✓	✓	x
This work	✓	✓	✓	✓	✓

[14] summarized the security risks and impacts of adversarial attacks on FL systems and proposed a multidimensional analysis framework from the source of the attack, the attack's impact, the attack's budget, and the attack's visibility. A taxonomy of adversarial attacks is also proposed. In terms of defense mechanisms, the effects of different defense mechanisms are compared through quantitative literature analysis. Nevertheless, this survey lacks empirical analysis to validate the legitimacy of these attacks and defense measures.

[11] and [4] employed a combination of literature analysis and empirical analysis to examine and summarize the security and privacy risks associated with FL systems. The study presented an overview of attacks and defenses regarding security and privacy and utilized experimental methods to validate various types of attacks empirically. However, it is important to note that the survey solely focused on CFL and did not address the emerging challenges faced by DFL systems.

[10,13,18,20], and [17] conducted extensive research on the vulnerabilities of FL systems. They developed taxonomies of adversarial attacks. While [15] focused on analyzing the threats throughout the entire life cycle of FL, considering various stages of the FL pipeline. They classified and summarized the attacks, examined their potential impact, and analyzed the available defense strategies concurrently. [12,19] evaluated the efficacy and viability of various defense strategies, classifying and categorizing defenses according to which attacks can be effectively mitigated. [8] centered on examining poisoning attacks, where various types of such attacks were categorized, and diverse strategies and methods for carrying out these attacks were elucidated. Additionally, the research entailed an analysis of the countermeasures employed against different attacks, their effectiveness, and the vulnerability of data distribution.

In conclusion, the current research examining and analyzing the security of FL mainly focuses on CFL systems, with only a limited number of articles and surveys exploring DFL. Additionally, these works rely on qualitative analysis and theoretical taxonomies, which are lacking in quantitative and experimental analysis. Thus, there is a need for a practical-oriented module that can quantitatively evaluate and analyze the robustness of DFL models using experiments while also appraising the efficacy of different defense mechanisms.

3. Poisoning attacks and defense mechanisms

Malicious attacks pose a significant obstacle in FL, especially poisoning attacks, which are designed to undermine the robustness of the FL

model and diminish the usability of its output. Given the distributed nature of FL, detecting and mitigating poisoning attacks presents a substantial challenge. This Section provides an overview of poisoning attacks and reviews existing defense mechanisms on both CFL and DFL.

3.1. Poisoning attacks on FL

In FL systems, where nodes are distributed across various entities, adversaries can tamper with data or models to disrupt the effectiveness and robustness of benign models. In contrast to CFL, detecting malicious activity in DFL poses greater challenges due to the absence of a central entity controlling the entire process. Poisoning attacks targeting DFL can be categorized based on two viewpoints: the attack intention and attack strategy [8]. The attack intention indicates the desired outcome of the adversary, whether it is to misclassify a particular target or to undermine the overall performance. The attack strategy entails the creation of a poisoning attack through the manipulation of either the data or the model.

Poisoning attacks can be categorized into three groups based on the intentions of the adversary, namely untargeted, targeted, and backdoor attacks [7,8]. The goal of untargeted attacks is to reduce the overall performance of the model, hindering its convergence. Conversely, targeted attacks involve intentional manipulation of the training data by the adversary to cause incorrect or biased predictions for specific target inputs. The objective of the targeted attacks is to infiltrate the model in such a way that only a particular target set or class is misclassified, while the rest of the set is correctly classified [8]. Another type of attack is the backdoor attack, where the adversary inserts one or more triggers into the model during training, which can then be exploited during the inference process [21]. Essentially, the model behaves normally without the trigger, but an attacker can elicit a desired prediction or classification by presenting the trigger during the inference stage.

In the CFL paradigm, targeted attacks are data-driven processes and thus tend to be launched at the client side. Untargeted attacks in the CFL paradigm can be executed without reliance on data, enabling them to be launched from either the client or server side. In contrast, in the DFL paradigm, the traditional differentiation between client and server is eliminated, allowing any node to launch both targeted and untargeted attacks. Consequently, the attack surfaces in the DFL paradigm are increased.

From the attack strategy perspective, poisoning attacks can be divided into data poisoning and model poisoning [8]. Data poisoning involves manipulating training data to indirectly affect the model's training process, such as changing labels, *i.e.,* label flipping, or inserting malicious samples, *i.e.,* sample poisoning [4]. Both label flipping and sample poisoning can be used as untargeted attacks or targeted attacks. In the context of backdoor attacks, the attacker can choose to incorporate either semantic or artificial backdoors. Semantic backdoors use naturally existing cues, like a specific colored striped background, as triggers. Artificial backdoors are intentionally created by injecting particular triggers, such as adding certain symbols to a sample [46]. The effectiveness of data poisoning attacks is limited by the need for a large number of poisoned samples [4]. In model poisoning attacks, malicious participants can manipulate the locally trained model directly by modifying shared weights or gradients [8]. These attacks pose a greater threat as they have a higher success rate and are more difficult to detect. Model poisoning can be divided into two categories: random weights generation, which involves adding random noise to the transmitted model, and optimized weights generation, which involves optimizing the distribution of the added noise to decrease the chances of detection [4,7].

In DFL, participants possess knowledge not only of their own local data and models but also of the models belonging to their neighboring nodes. Therefore, malicious nodes can strategically optimize their attacks and execute more threatening attacks while avoiding detection, thereby heightening the complexity of designing effective defense mechanisms.

Theoretically, while DFL improves system fault tolerance and mitigates single-point-of-failure, it also expands the attack surface and potentially exposes the system to a greater array of attack vectors. Thereby, DFL does not offer superiority over CFL in terms of model robustness.

3.2. Defense mechanisms

Most defense methods are designed for the centralized setting, which involves a less complex model aggregation process by a single aggregation server. However, in DFL, the lack of a centralized control entity and the increased array of attack vectors mentioned in Section 3.1 pose challenges when developing defense strategies for DFL. This paper categorizes the approaches for mitigating model poisoning attacks into five categories: Byzantine-robust aggregation, anomaly detection, Moving Target Defense (MTD), hybrid, and post-aggregation.

As shown in Table 2, these articles have been reviewed, compared, and analyzed across various dimensions, such as the techniques employed, the paradigms in which they are implemented (CFL or DFL), and their effectiveness in addressing both targeted and untargeted attacks.

3.2.1. Byzantine-robust aggregation

The main objective of Byzantine-robust defense techniques is to establish a robust (referred to as Byzantine resilience) aggregation rule that prevents the model's performance from being compromised by malicious updates [26,41]. These defense methods can be categorized into geometric measures, regularization, and decomposition.

There are two types of geometric approaches: coordinate-wise filtering and vector-wise filtering. The Coordinate-wise Median (*COMED*) is a defensive strategy that uses dimension-wise filtering to protect against attacks [22]. *COMED* is like the median concept but applied in high-dimensional spaces to remove outliers. This approach is effective against model replacement attacks. Another approach called *Trimmed-Mean* method [23] also uses dimension-wise filtering. It eliminates the lowest and highest values in each dimension using a trimming parameter. The underlying assumption of this approach is that malicious updates can be identified as outliers compared to benign updates.

Vector-wise filtering is a commonly used geometric measure. *RFA* [24] utilizes the geometric median as an aggregation function, which helps defend against untargeted poisoning attacks. However, this defense mechanism can be compromised if an attacker uses a singular-flipping attack to manipulate the geometric center [36]. *Krum* [25] is a widely used robust aggregation function that assigns scores to client updates based on their similarity to other updates and selects the update with the lowest score to update the global model. *Multi-Krum* is a modified version that selects multiple client updates for the next global model. *Bulyan* [26] is proposed as a two-stage aggregation protocol that overcomes the limitations of *Krum*, but experiments have shown that it performs worse than*Multi-Krum* by potentially reducing benign updates [45].

The second category of Byzantine-robust methods is the regularization-based approach. *Zeno++* [27] is an asynchronous protocol for robust CFL. This approach proposes an estimated score for gradient descent that considers the loss and update magnitude. This helps reduce the impact of malicious gradient updates. However, a drawback is that the server needs a validation dataset, which is a constraint in a federated setting. Additionally, relying on performance rather than parameters adds extra computational burden [47]. Adaptive Federated Averaging (*AFA*) [28] not only filters out potentially malicious updates but also completely blocks adversaries from participating. *AFA* employs a Hidden Markov Model (HMM) based on gradient cosine similarity to predict a client's behavior, thereby incorporating a learning component into the defense system. This predictive score

Table 2
Overview of defense mechanisms against poisoning attacks in FL.

Category	Type	Method	Technique	Paradigm	Objective	
					U	T
Byzantine-Robust Aggregation	Geometry	*COMED* [22]	Coordinate-wise median	CFL	✓	x
		TrimmedMean [23]	Filtered mean	CFL	✓	x
		RFA [24]	Geometric median	CFL	✓	x
		Krum [25]	Euclidean distance	CFL	✓	x
		Multi-Krum [25]	Euclidean distance	CFL	✓	x
		Bulyan [26]	Krum and TrimmedMean	CFL	✓	x
	Regularization	*Zeno++* [27]	Approximated gradient descent score	CFL	✓	x
		AFA [28]	Gradient similarity, Hidden Markov model	CFL	✓	x
		RSA [29]	Norm regularization	CFL	✓	x
	Decomposition	*DnC* [30]	Dimensionality Reduction (PCA) and SVD	CFL	✓	x
		RLR [31]	Learning rate decomposition	CFL	–	✓
Anomaly Detection	Validation	*ERR, LFR* [32]	Global validation	CFL	✓	✓
		PDGAN [33]	Model accuracy auditing	CFL	✓	✓
	Gradient-based	*FoolsGold* [34]	Gradient similarity (cosine)	CFL	✓	✓
		FLDetector [35]	Hessian-based gradient consistency	CFL	✓	✓
		Li et al. [36]	Spectral anomaly detection	CFL	✓	✓
		Sniper [33]	Graph clustering	CFL	–	✓
MTD		*Voyager* [37]	Manipulating of the network topology	DFL	✓	x
Hybrid Mechanism		*FLTrust* [38]	ReLU-clipped cosine similarity, norm thresholding	CFL	✓	✓
		Trusted DFL [39]	Trusted aggregation	DFL	✓	–
		FedInv [40]	Gradient-based clustering distribution divergences	CFL	✓	✓
		Norm Clipping [31]	Clipping and worker momentum	DFL	✓	✓
		DeepSight [41]	Classification and clustering with update fingerprinting	CFL	✓	✓
		FLAME [42]	Clustering (cosine similarity), adaptive clipping, noising	CFL	✓	✓
		Sentinel [43]	Model similarity, bootstrap validation and normalization	DFL	✓	✓
Post-Aggregation		Wu et al. [44]	Neuron Pruning	CFL	x	✓
		Sun et al. [45]	Weak Differential Privacy	CFL	x	✓

Untargeted Attacks (U), Targeted Attacks (T).

is then utilized as a weight factor in the aggregation process, serving as a regularization term. Another method, *RSA* [29], integrates ℓ_1-norm regularization into Stochastic Gradient Descent (SGD) to enhance robustness against Byzantine attacks.

The last class of Byzantine-robust is the decomposition-based approach. Divide-and-Conquer (*DnC*) [30] uses spectral analysis, *i.e.,* singular value decomposition, to identify and filter out poisoned updates. The evaluations show that these techniques perform well in scenarios where the data is Independent and Identically Distributed (IID). However, when dealing with non-IID datasets, *DnC* is not effective in preventing attacks if the adversary has knowledge from benign clients. Robust Learning Rate (*RLR*) [31] tackles this problem by implementing a mechanism where clients vote for the direction of the global model update based on the algebraic sign of their update vector. Each dimension of the update vector requires a certain learning threshold to be met by the total number of votes. However, a challenge in *RLR* is determining an appropriate learning threshold that the number of malicious clients cannot exceed.

3.2.2. Anomaly detection

Defense mechanisms that employ anomaly detection, also known as Byzantine detection, aim to identify and eliminate potentially harmful updates [48]. Unlike Byzantine robustness, these anomaly detection schemes do not incorporate the defense strategy into the aggregation rules.

Error Rate-based Rejection (*ERR*) and Loss Function-based Rejection (*LFR*) are designed in [32] to enhance the robustness of CFL. These methods involve evaluating the performance of client models using a validation dataset on the server side. Before combining the received models, a certain number of updates that have the most impact on either the loss or validation error are discarded. However, a drawback of these methods is that the server needs to collect a clean dataset, which goes against the privacy-preserving principles of FL. To address this limitation, *PDGAN* [33] utilizes a generative adversarial network (GAN) to generate a synthetic dataset for outlier model detection. However, it should be noted that the GAN needs to be trained before the task starts, and the server must have sufficient computing power to train an ML model in a reasonable time while performing aggregation.

FoolsGold [34] is a gradient similarity-based anomaly detection method that aims to identify grouped adversaries. It utilizes the calculation of gradient cosine similarity to detect malicious attacks that exhibit high similarity. However, *FoolsGold* is ineffective against individual adversaries. *FLDetector* [35] predicts a client's update by analyzing its past contributions. This defense mechanism has shown effectiveness against various adaptive attacks, such as scaling attacks, distributed backdoors, and untargeted model poisoning. However, the computational cost of predicting consistency is high. Another proposed defense mechanism suggested in [36] involves training a spectral anomaly detection model to identify malicious updates in a low-dimensional latent space. This approach trains an autoencoder model on benign model updates to detect anomalous models. However, selecting appropriate benign model updates for training the autoencoder model is crucial. *Sniper* [49]maps client updates into a graph-cluster using gradient similarity and effectively defends against poisoning attacks, but its performance is reduced in a more heterogeneous setting.

3.2.3. MTD-based techniques

MTD is an innovative security strategy that seeks to minimize the consequences of cyberattacks by actively or passively modifying specific system elements [50]. Proposed in [37], *Voyager* is an MTD-based aggregation protocol that reactively alters the network topology to enhance the resilience of the DFL system. The *Voyager* protocol comprises three stages: an anomaly detector, a network topology explorer, and a connection deployer. The anomaly detector assesses the received models against shared models from other participants to identify any abnormal models, thereby triggering the MTD policy. The network topology explorer aims to identify more reliable participants. Lastly, the connection deployer establishes connections with these candidate participants and exchanges their respective models to create an aggregated model.

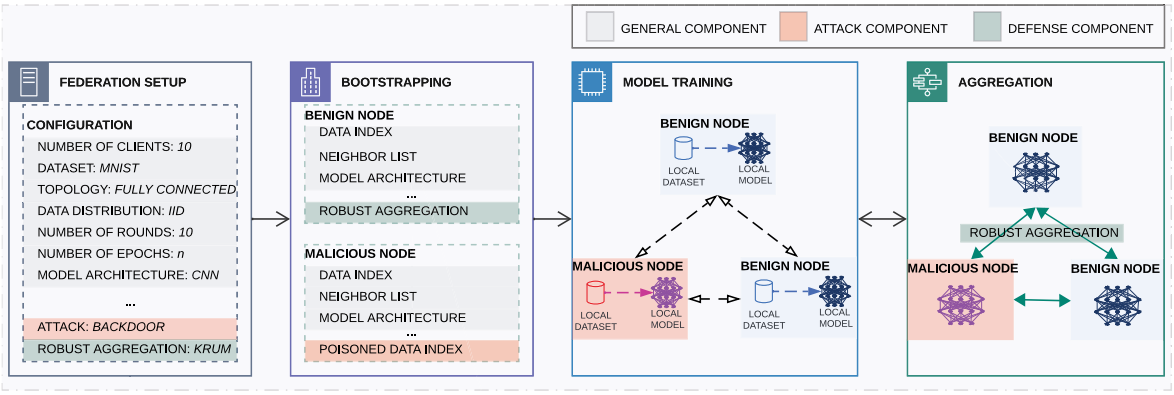

Fig. 2. *DART* within the DFL Procedure.

3.2.4. Hybrid defense techniques

Hybrid defenses combine robust aggregation and anomaly detection techniques. In the *FLTrust* approach [38], the server uses a clean dataset to train a reference model and compares it to received model updates using cosine similarity. These updates are then aggregated based on their trust scores. However, obtaining a root dataset may not be possible in a CFL architecture. *Trusted DFL* [39] introduces trusted aggregation in a decentralized architecture by evaluating nodes based on their behavior and update consistency. Each node shares its local trust score, enabling other nodes to compute a global trust score for aggregation. *FedInv* [40] addresses the limitations of *FLTrust* and *PDGAN* by using a privacy-preserving model inversion strategy to generate dummy data from clients' gradient updates. These updates are scored based on distribution divergence and aggregated using majority clustering. However, the computational complexity of *FedInv* is currently being discussed.

Norm Clipping [31] is a defense mechanism that limits the scale of model weights to protect against semantic backdoor attacks in CFL. However, it only considers the magnitude of an update and not its direction, making it difficult to determine an appropriate threshold. *DeepSight* [41] combines filtering and clipping to significantly reduce an attacker's possibilities by analyzing the neural network structure and creating fingerprints. *DeepSight* is effective but computationally complex. *FLAME* [42] proposes a hybrid approach using dynamic clustering, adaptive clipping, and noising. Client updates are clustered based on similarity, filtered for outliers, and then clipped. The aggregated model is smoothed using adaptive noising. However, *FLAME* fails when the number of attackers exceeds half of the cluster.

Sentinel [43] is a hybrid defense strategy to enhance the resilience of DFL against poisoning attacks. It introduces a three-phase aggregation protocol to strengthen security measures. In the first phase, a layer-wise average cosine similarity metric is used to identify and filter out suspicious model updates. The remaining updates are then aggregated based on local bootstrap validation loss, with aggregation weights determined through adaptive loss distance mapping. In the final phase, trusted models are normalized using the local model norm as a threshold to minimize the impact of potential stealth attacks.

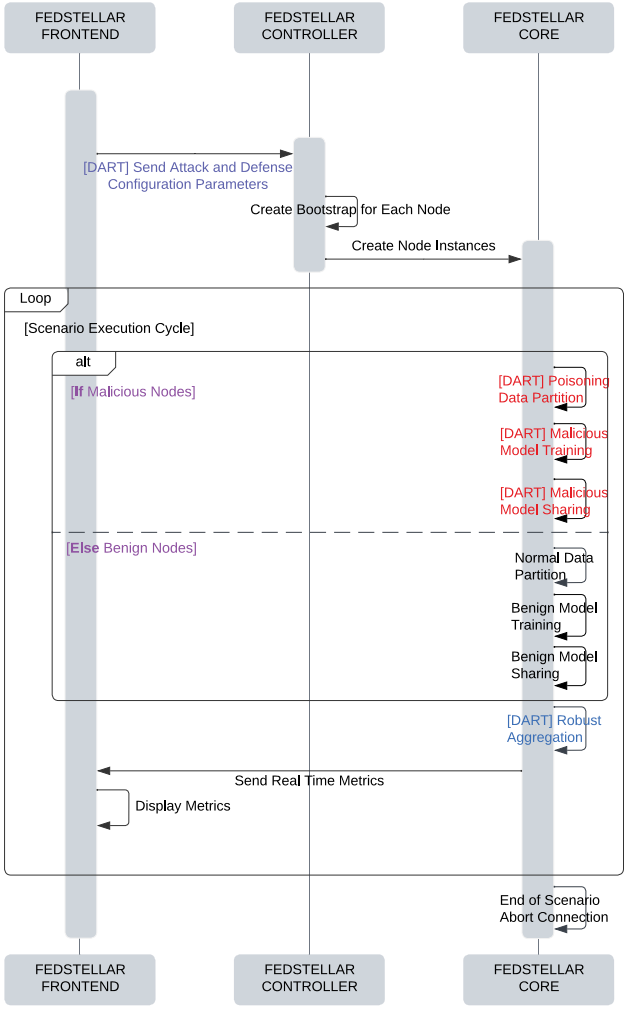

Fig. 3. Sequence Diagram Showing the Procedure of *DART*.

3.2.5. Post-aggregation techniques

Post-aggregation techniques refer to approaches that do not disrupt the process of aggregation. Instead, modifications to model updates are made after the aggregation has taken place. [44] proposed a method to defend against backdoor attacks in the CFL setting. They use neuron pruning to identify and deactivate maliciously activated neurons during an attack. Clients submit voting sequences based on their averaged activation values, and the server aggregates these votes to determine which neurons to prune. To enhance the defense, [45] suggested incorporating weak differential privacy using Gaussian noise. Adding even small amounts of noise can protect against attacks, but it may decrease

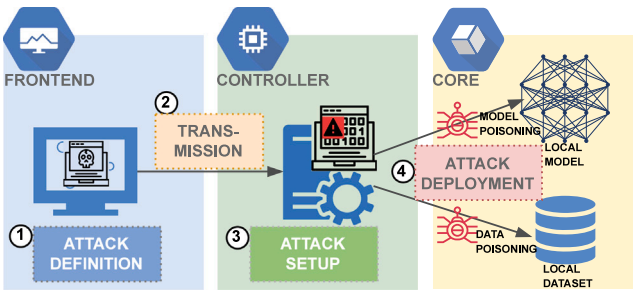

Fig. 4. Attack Component of *DART*.

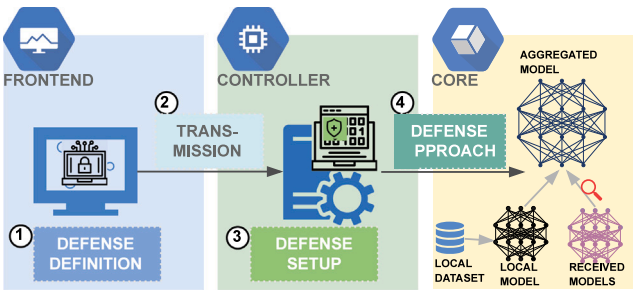

Fig. 5. Defense Component of *DART*.

the accuracy of the main task. However, determining the right amount of noise to add is challenging.

Overall, a significant amount of ongoing research is focused on enhancing the robustness of the FL systems model. However, most of the research focuses on CFL and does not consider DFL's unique characteristics. These studies also have limitations, such as being computationally complex and sensitive to data distribution.

4. *DART* Solution

This section bridges the research gap highlighted in the previous section by designing and implementing a robustness analysis module for DFL systems. This module enables analysis of the robustness of DFL models when confronted with poisoning attacks and evaluation of the effectiveness of various defense mechanisms.

4.1. Design of the DART module

In general, the training procedure of a DFL model can be divided into four stages, as shown in Fig. 2. Initially, on the DFL platform, the user establishes the fundamental configuration of the federation, including the desired number of participants, the dataset to be used for training, the client interconnection topology, or the number of rounds of aggregation. After that, the user-defined configuration is transformed into specific bootstrapping for each individual node, encompassing details such as the actual training samples for each node, the neighbors to which they are connected, and the specific model architecture. With this configuration information, nodes start local model training while establishing communication channels with their neighboring nodes. Once the local model training is finished, each node exchanges and aggregates its respective models. These local training and model aggregation processes are repeated until the model reaches convergence or reaches a predetermined number of aggregation rounds.

However, this procedure lacks consideration for the robustness analysis of the DFL model, meaning users cannot customize and execute poisoning attacks or select and implement defense mechanisms. Therefore, this study proposes the *DART* module, consisting of two components: the attack component (highlighted red in Fig. 2) and the defense component (highlighted green in Fig. 2). When defining the desired federation, users can choose the type of poisoning attack they wish to deploy. Depending on whether the user wants to evaluate only the robustness of the DFL model or to assess the effectiveness of the defenses against poisoning attacks, the user can choose to deploy a defense mechanism. During the bootstrapping phase, the benign node selects the defense mechanism while the malicious node chooses the desired poisoning attack. In the model training node, the malicious node trains the poisoned model and the benign node trains the uninfected model. During the aggregation phase, the benign node implements defenses to filter or mitigate attacks from the malicious node.

Specifically, as depicted in the sequence diagram shown in Fig. 3, upon selecting the DFL scenario in the frontend interface, the user could also choose the desired attack and defense settings to be executed. These configuration parameters are then communicated to the controller through the REST API. Subsequently, the controller writes these configuration parameters to the bootstrap items of each node, specifying which nodes are functioning as benign nodes and which are operating as malicious nodes. The core creates instances of the nodes based on their respective configuration parameters and starts executing the scenario. A normal dataset is provided for benign nodes to train the local model, which is then shared with neighboring nodes. In contrast, malicious nodes are supplied with poisoned data and either train a malicious model through data poisoning or directly tamper with the trained local model through model poisoning, transmitting the malicious model to neighboring nodes. Next, the node follows the robust aggregation algorithm chosen in the configuration for model aggregation, proceeding to the next round of the scenario execution loop.

4.2. DART Module in Fedstellar Platform

Fedstellar [51] is a DFL platform that provides a user-friendly interface to easily deploy and train DFL models. Therefore, this paper adopts *Fedstellar* as the infrastructure and incorporates a model robustness analysis module, called *DART*. As shown in Fig. 6, *Fedstellar* consists of three layers, the frontend, the controller, and the core, corresponding to the four steps of the DFL model training process.

- **Frontend.** This layer corresponds to the federation setup step and enables the user and system interaction, allowing the user to personalize their own FL scenario through diverse configurations. Moreover, the system provides real-time monitoring functionalities, including the tracking of resource usage and model performance during the training process. Building upon this layer, *DART* provides the option to choose attacks and defense mechanisms.
- **Controller.** This layer corresponds to the bootstrapping step and functions as the central hub of the platform. It receives user configurations from the frontend, manages the federated scenario, selects appropriate learning algorithms and datasets, and establishes network connections to enable a FL process. The attack and defense are specified within this layer.
- **Core.** This layer corresponds to the training and aggregation steps. Its operations entail data preparation, model training, ensuring secure communication among participants, and storing the federated models. Therefore, this layer encompasses the primary functionality of *DART*, including the deployment and execution of attacks and defenses.

4.2.1. DART: Attack Component

The *DART* attack component consists of three parts, as illustrated in Fig. 4. The process includes four steps; first, when the user selects the DFL configuration in the Frontend, the user can also select the attack definitions, such as what type of attack is performed, the percentage

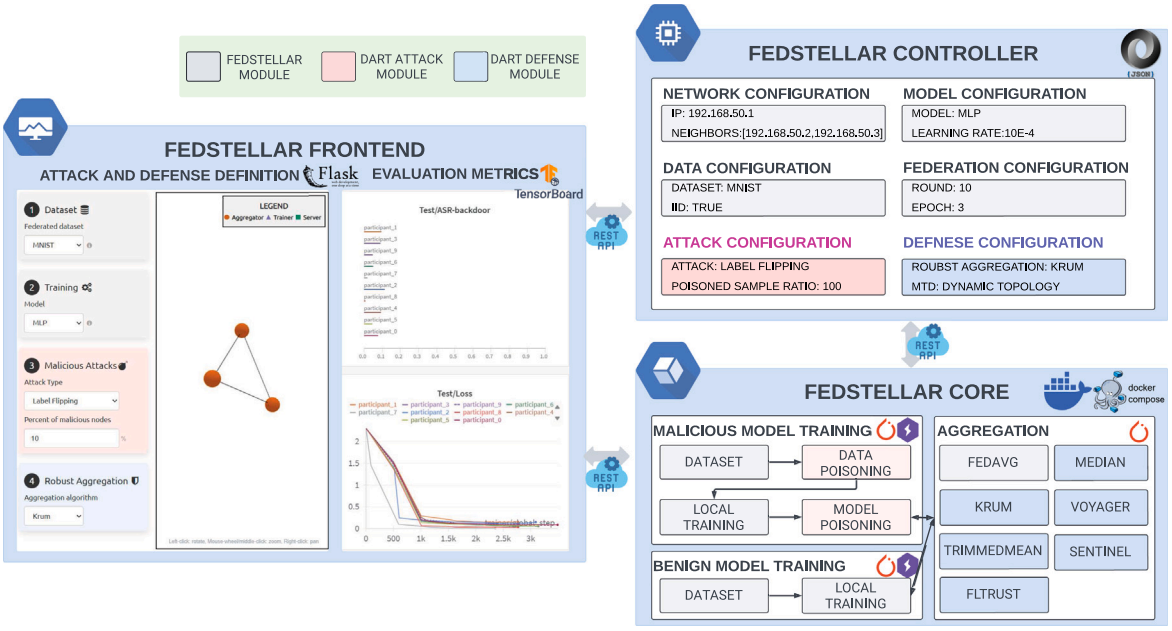

Fig. 6. Integration of *DART* Module in *Fedstellar* Platform.

of nodes to be attacked, and the percentage of noise injection. These configurations are transmitted to the Controller as JSON files. The Controller then sets up specific attacks in the nodes based on these attack configurations, such as which nodes are configured as malicious nodes and what attack strategies these nodes use.

The Controller utilizes this attack information along with other configurations to bootstrap the node's operation. In the fourth step, the Core starts the execution of these attacks. These attacks encompass two parts: one involves manipulating the data, such as altering the training data of the model to facilitate a data poisoning attack, while the other entails directly modifying the model, known as a model poisoning attack. The attack component of *DART* integrates the following poisoning attacks: Untargeted Label Flipping, Untargeted Sample Poisoning, Random Model Poisoning, Targeted Label Flipping, and Backdoor Attack.

4.2.2. *DART*: Defense Component

Similarly to the previous component, the *DART* defense component is incorporated within the three layers of *Fedstellar* and operates based on the same underlying principles, as shown in Fig. 5. Firstly, the user operating the Frontend chooses the specific defense mechanism that they wish to evaluate. In the second step, these configurations are transmitted to the backend Controller as JSON files. In the third step, the Controller converts these configurations into bootstrap for the node, including which defense mechanisms to employ. In the fourth step, the Core starts to train the local model, as well as run the defense mechanisms to protect its model from malicious models. The defense component incorporates a range of defense mechanisms, namely *Krum*, *Median*, *TrimmedMean*, *FLTrust*, *Sentinel*, and the MTD-based *Voyager*.

4.3. Implementation of *DART*

Fig. 6 illustrates the implementation and integration of *DART* within the *Fedstellar* platform. *Fedstellar* provides the fundamental DFL scenario configuration elements on the frontend, while *DART* enhances this by introducing the option for malicious attacks and defense mechanisms. Users are able to select which attacks to execute and determine the ratio of malicious nodes within the federation. Additionally, users can choose from various robust aggregation functions for defense, such as *Krum* and *TrimmedMean*. The frontend of *DART* is developed using the Flask framework [52], with user selections communicated to the backend controller through the REST API.

The controller generates bootstrapping details for each node based on the user's selections, encompassing network settings, neighbor connections, and datasets. *DART* adds configuration specifics for attacks and defenses. Malicious nodes are configured with attack settings, while benign nodes are designated as "no attack". The chosen robust aggregation function for defense is incorporated into the node's bootstrap as the model aggregation algorithm. Configuration details for each node are stored in JSON format on the file system to facilitate node bootstrap and are also transmitted to the core system through REST API.

Algorithm 1 *DART* PROCESS IN EACH NODE

Require: D_i: Local Dataset, $\mathcal{F}$: Model Training, $\mathcal{A}$: Aggregation Function, P: Neighbor List, R: Total Rounds, $\mathcal{T}$: Model Transmission, $\mathcal{DP}$: Data Poisoning, $\mathcal{MP}$: Model Poisoning.

$r \leftarrow 0$
$m_i \leftarrow$ Local Model
$M_i \leftarrow$ Neighbor Model List
if Data Poisoning **then** ▷ Poison Local Dataset
  $D_i \leftarrow \mathcal{DP}(D_i)$
end if
for $r \leq R$ **do**
  $M_i \leftarrow [\,]$
  $m_i \leftarrow \mathcal{F}(D_i, m_i)$ ▷ Local Model Training
  if Model Poisoning **then** ▷ Poison Local Model
    $M_i \leftarrow \mathcal{MP}(M_i)$
  end if
  $\mathcal{T}(M_i, P)$ ▷ Transmit Local Model to Neighbors
  for j in P **do** ▷ Receive Models from Neighbors
    $M_i \leftarrow M_i \cup \{e\}$
  end for
  $m_i \leftarrow \mathcal{A}(m_i, M_i)$ ▷ Model Aggregation
end for

The core system creates instances of individual nodes through the bootstrap information. Docker containers [53] are used to virtualize the nodes. When the docker instance is started, the nodes follow the configuration information and get the local data. The *DART* process in each node is shown in Algorithm 1. If a node is benign, it will

use the regular model training pipeline to train the model locally, transmit the local model to its neighbors, and wait for the model from its neighbors. If a node is malicious, then it will manipulate the data according to attack configuration, get the poisoned local dataset, train the malicious model, and then share it with the neighbors; or the node modifies the local model, gets the poisoned model, and shares it with the neighbors. The local model is developed using the Pytorch framework [54], with Pytorch Lightning [55] serving as the model trainer for model optimization.

Following the acquisition of models shared by neighboring nodes, the node utilizes the selected algorithm outlined in the bootstrap configuration to aggregate models. After the aggregation is completed, the next round of local model training and model aggregation is performed. Throughout the model training and testing phases, metrics, including model performance, attack, and defense effectiveness, are transmitted from core to frontend utilizing a REST API. The frontend then visualizes these metrics via TensorBoard [56].

5. Experimental analysis

This section compares and analyzes the impact of poisoning attacks on the model robustness of CFL and DFL paradigms and identifies the factors influencing their efficacy. Besides, this section benchmarks various defense mechanisms in diverse attack scenarios.

5.1. Datasets and federation setups

The following datasets and deep learning model are chosen to evaluate the aggregation algorithms implemented for the Fedstellar framework:

- **MNIST** [57] consists of handwritten digits represented by 28×28 grayscale images. It compromises 60 000 training samples and 10 000 test samples. A three layers multilayer perceptron (MLP) with a linear input layer of size 784×256, a linear hidden layer of size 256×128, and a linear output layer of size 128×10 is used for classification.
- **FashionMNIST** [58] consists of 60 000 training samples and 10 000 test samples, which are 28×28 grayscale images with 10 classes. For the FashionMNIST dataset, the same MLP as for MNIST is used.
- **Cifar10** [59] consists of 50 000 training and 10 000 test images with 10 classes of 32×32 color images. A small convolutional neural network (CNN) designed for mobile applications is used for this task [60].

The experiment configurations are listed in Table 3. The learning process is designed to run for a total of 10 rounds, with each round comprising three epochs. All experiments are performed in the federation composed of 10 clients, and the data are evenly distributed across the clients in an IID manner. All experiments are carried out on the *Fedstellar* platform in synchronous mode. The federation network has a bandwidth of 1 Mbps without loss and delay. For the experiments with different attacks, the Poisoned Node Ratios (PNR) are increased from 0% to 90%. The defense mechanisms *Krum*, *Median*, *TrimmedMean*, *FLTrust*, *Sentinal*, and *Voyager* are implemented, assessed, and compared. Furthermore, the experiments employ four different participants interconnection topologies, *i.e.,* ring, star, random, and fully connected.

5.2. Poisoning attack on CFL and DFL

This experiment first analyzes the performance of CFL and DFL in the presence of different types of poisoning attacks. When executing the attacks, it assumes that all the attacks occur during the model training process and recur in each round. These attacks encompass both data poisoning and model poisoning strategies with untargeted and targeted intentions. Experiments with malicious client proportions ranging from 0% to 90% are executed to observe the effect of different degrees of maliciousness on the model robustness.

5.2.1. Experimental analysis of untargeted attacks

Three untargeted attacks have been implemented to compare the attack performance in CFL and DFL, including Untargeted Label Flipping, Untargeted Sample Poisoning, and Random Model Poisoning attacks. The detailed attack setups are listed below:

- **Untargeted Label Flipping**: Randomly replacing all the labels corresponding to the samples in the training set such that the model is unable to map the data patterns of the samples to the correct labels. Thus, the loss function cannot correctly guide the optimization direction of the model.
- **Untargeted Sample Poisoning**: 30% of Gaussian noise is added to all of the samples in the training set to mask the correct features in the data. As a result, the model fails to recognize effective patterns in the data and cannot classify the data correctly.
- **Random Model Poisoning**: Before sending the model for aggregation, the malicious node directly adds 30% of Gaussian noise to the parameters of the trained local model to devastate the model's effectiveness.

(I) evaluation metrics for untargeted attacks. The purpose of an untargeted attack is to spread malice in the FL system and reduce the effectiveness of the models of all nodes. Therefore, the average F1-Score in the model's benign nodes is calculated as the metric to evaluate the model's robustness and the effectiveness of the untargeted attack. A lower F1-Score indicates that the untargeted attack is more effective, but the FL models have lower robustness. This experiment is carried out in CFL and DFL utilizing fully connected, ring, and star network topologies. Besides, FedAvg is employed as the aggregation function across all nodes without additional defense mechanisms.

(Ii) attack effectiveness.

Fig. 7 illustrates the impact, *i.e.,* average F1-Score, of three untargeted attacks on CFL and DFL with three datasets as the PNR increases. Overall, the effects of the Model Poisoning attack are much more destructive, with 10% of the nodes attacked having a significant reduction in the modeling effects of both CFL and DFL. With more than 30% of the nodes attacked, the model of the benign node is close to being completely destroyed and no longer able to provide meaningful inferences in all of these three datasets. The Label Flipping Attack shows a strong effect only after more than 30% of malicious nodes, and the model of benign nodes is destroyed when the percentage of malicious nodes exceeds 50%. In contrast, the effect of Sample Poisoning is weaker, and the effect of the attack is shown when the percentage of malicious nodes is greater than 50%, but even if the number of nodes attacked reaches 90%, for the MNIST dataset, the rest of the benign node model still has close to 80% of the F1-Score, which suggests that Sample Poisoning in the attack strategy is not as effective as the other two in the untargeted attack. The reason is that since the noise injected by Sample Poisoning does not always cover all of the features, the model is able to learn some patterns from limited information.

(Iii) model robustness in different paradigms and topologies.

For different FL paradigms, the CFL and DFL with fully connected topology exhibit a similar decrease in the model F1-Score across the attacks. However, the model robustness of DFL is affected by the manner in which participants are interconnected, *i.e.,* the topology. The results from Fig. 7 indicate that DFL with the ring and star topology exhibit reduced levels of model robustness in counter when faced with untargeted poisoning attacks.

[37] suggests that a benign node's connections to malicious nodes follow a hypergeometric distribution. Nodes with fewer average neighbors in a DFL network are more susceptible to poisoned attacks. Compared to fully connected, star and ring topologies are sparser, resulting in a lower decrease in F1-Score when facing untargeted attacks. To validate this hypothesis, this paper conducts an experiment on different DFL connected by random networks, using the Watts–Strogatz [61]

Table 3
Configuration of experiments.

Configuration		Value
Federation	Number of Clients	10
	Total Rounds	10
	Epochs in Each Round	3
	Paradigm	CFL, DFL
	Data Distribution	IID
Robustness	Attacks	Untargeted Label Flipping Untargeted Sample Poisoning Random Model Poisoning Targeted Label Flipping Backdoor Attack
	Poisoned Node Ratios (PNR)	0%, 10%, 30%, 50%, 70%, 90%
	Aggregation Function	*FedAvg*, *Krum*, *Median*, *TrimmedMean*, *FLTrust*, *Sentinal*, *Voyager*
Network	Bandwidth	1 Mbps
	Delay	0 ms
	Topology	Fully connected, Ring, Star, Random

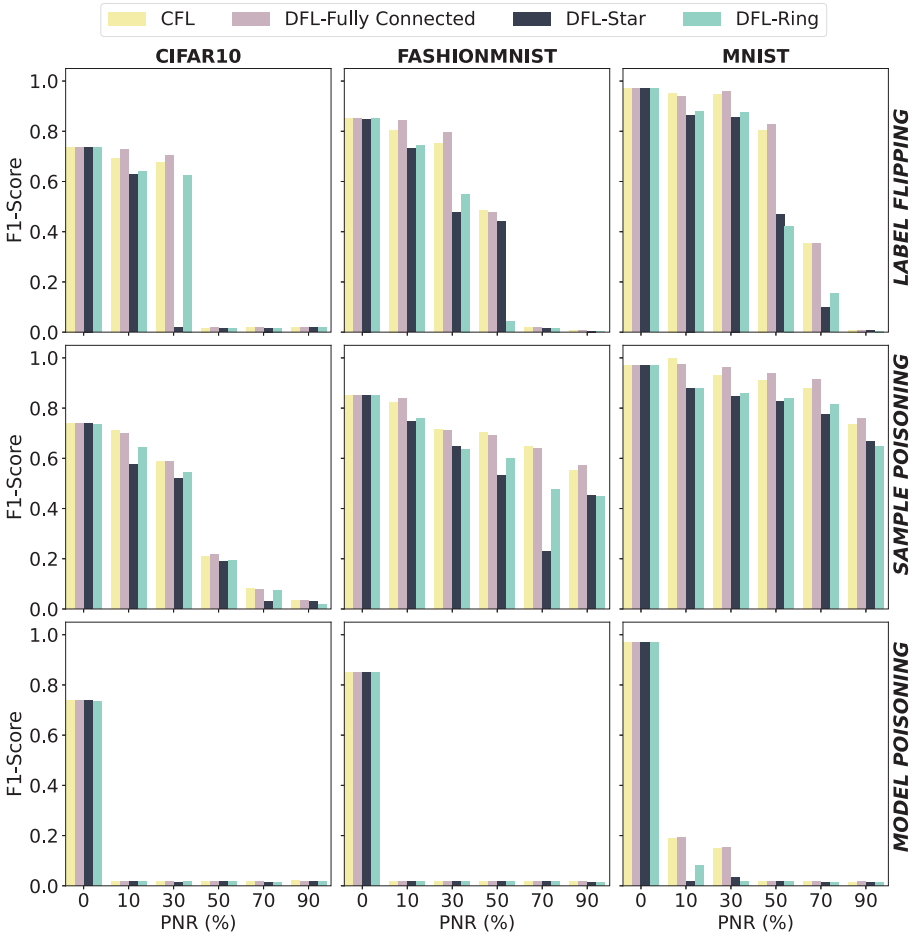

Fig. 7. Average F1-Score Results for Untargeted Poisoning Attacks.

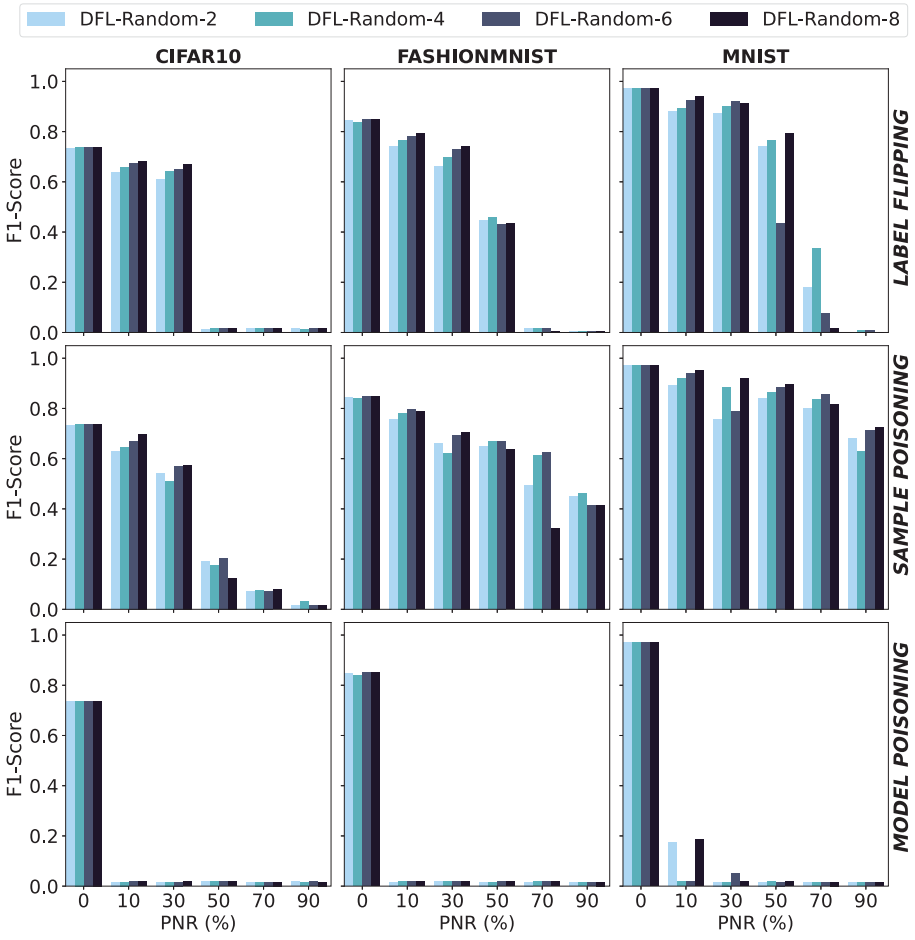

Fig. 8. Average F1-Score Results for Untargeted Poisoning Attacks with Random Topologies.

model to generate random small-world networks with average neighbor numbers of 2, 4, 6, and 8.

Fig. 8 illustrates the average F1-Score of three untargeted attacks on the random topology of DFL with different average connected neighbors with three datasets as the PNR increases. The DFL model with 8 neighbors performs the best against the three attacks, followed by 6 neighbors, while 2 neighbors perform the worst. These results demonstrate that the network sparsity significantly influences the model robustness of DFL. Specifically, a denser network, characterized by a higher average number of connected neighbors, enhances the robustness of the model against poisoning attacks. This is because nodes need to aggregate fewer models when the average number of connected neighbors is small. Therefore, once aggregated with a malicious model, the benign node model will be greatly affected by the malicious ones. Whereas for a dense network, such as fully connected, although there is a high probability for a benign node to aggregate with a malicious node since the aggregation occurs in a large number of models, the maliciousness is thereby diluted and demonstrates better robustness.

In conclusion, when confronted with untargeted attacks, CFL and DFL exhibit similar performance. The model robustness of DFL is influenced by the network topology, with denser networks yield greater robustness. In terms of attacks, Model Poisoning Attacks prove to be the most efficient across various attack strategies. Label Flipping produces comparable outcomes but necessitates a higher proportion of malicious nodes. Conversely, Sample Poisoning Attacks are less effective than the aforementioned strategies regarding untargeted attacks.

5.2.2. Experimental analysis of targeted attacks

In this section, two targeted attacks are designed and implemented: Targeted Label Flipping and Backdoor Attack. The detailed attack setups are listed below:

- **Targeted Label Flipping**: Replacing only specific labels, specifically, replacing all the label 7 with 4 in the training dataset. Thus, for the model, the feature patterns of the training samples originally belonging to label 7 will be recognized as label 4. Consequently, all the samples that should have corresponded to 7 will be incorrectly classified as label 4, whereas for the other labels, their classification results are not affected.
- **Backdoor Attack**: A 10×10 pixels of an X shape watermark is added to the top right corner of the samples that belong to label 4 in the training dataset, such that the model learns the pattern corresponding to label 4 as the implanted watermark. In the testing phase, when this watermark does not appear, the model behaves normally, while the samples containing the watermark are misclassified as label 4.

(I) evaluation metrics for targeted attacks.

This experiment first evaluates the overall performance of the model, *i.e.,* the F1-Score, when subjected to these targeted attacks, as shown in Fig. 9. The results show that even when 90% of the nodes in both CFL and DFL models are suffering from targeted attacks, the overall performance on the model is not significantly dropped compared with no attack presence. This finding suggests that targeted attacks are

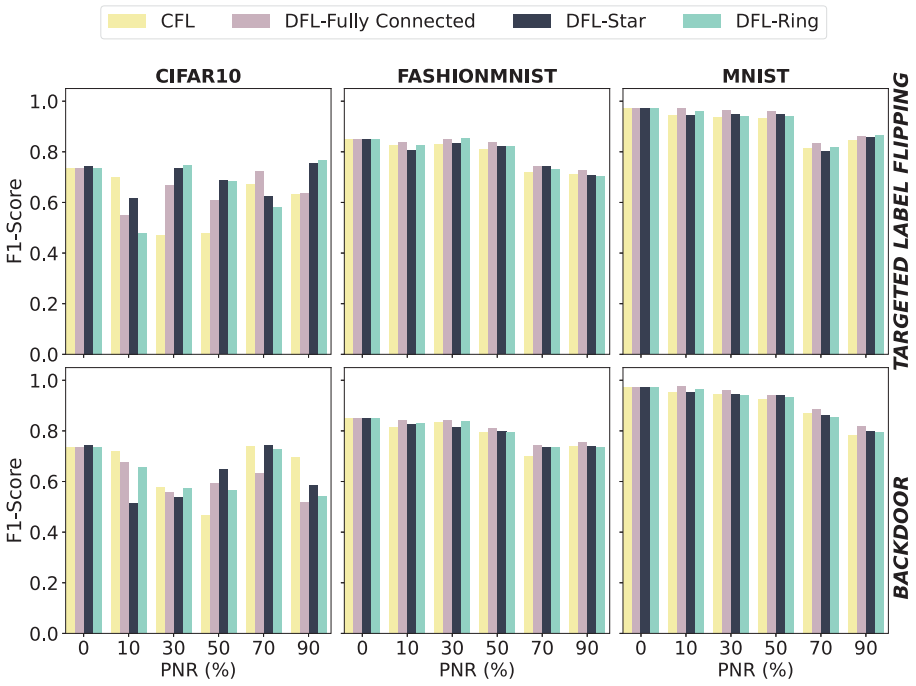

Fig. 9. Average F1-Score Results for Targeted Poisoning Attacks.

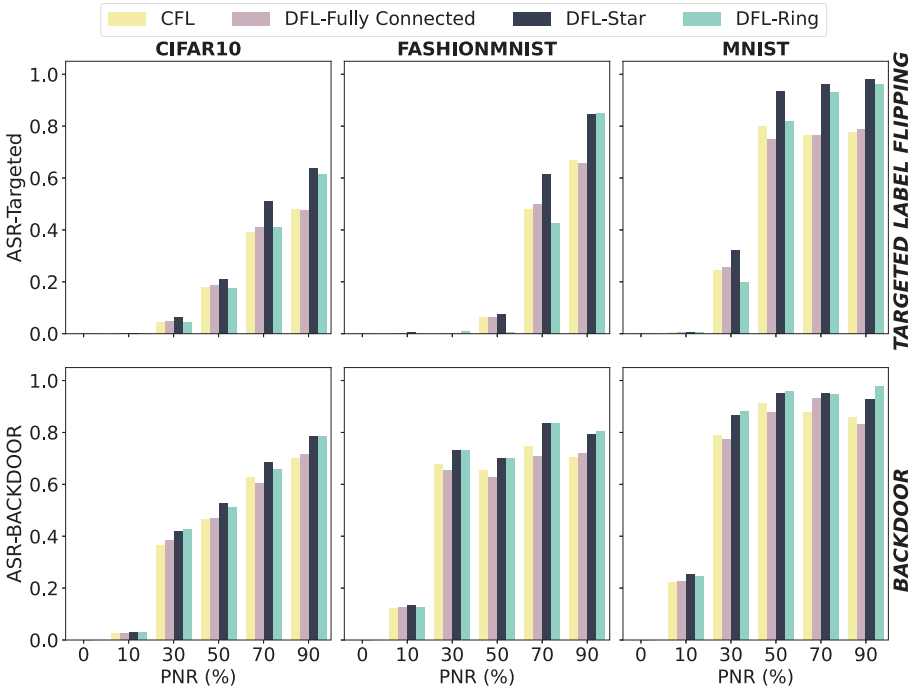

Fig. 10. Average ASR Results for Targeted Poisoning Attacks.

more challenging to be detected through overall performance metrics. Therefore, this section adopts Attack Success Rate (ASR) to measure the effectiveness of targeted attacks.

The objective of Targeted Label Flipping is to cause misclassification of a source label l_{src} to a desired target label l_t. The ASR-Targeted is determined by the number of samples in which the true label $y = l_{src}$ is incorrectly predicted as the target label $\hat{y} = l_t$, as described in Eq. (1). c_{ij} represents the count of samples with the true label y_i and predicted label $\hat{y}_j$, and the set L represents the labels present in the dataset.

$$ASR - Targeted = \frac{c_{src,t}}{\sum_{j=0}^{|L|} c_{src,j}} \tag{1}$$

The Backdoor attack involves the adversary inserting a trigger into local data samples that are associated with a particular target label l_t. The effectiveness of the Backdoor attack is evaluated using ASR-Backdoor, which is defined in Eq. (2). Here, c_{ij} denotes the number

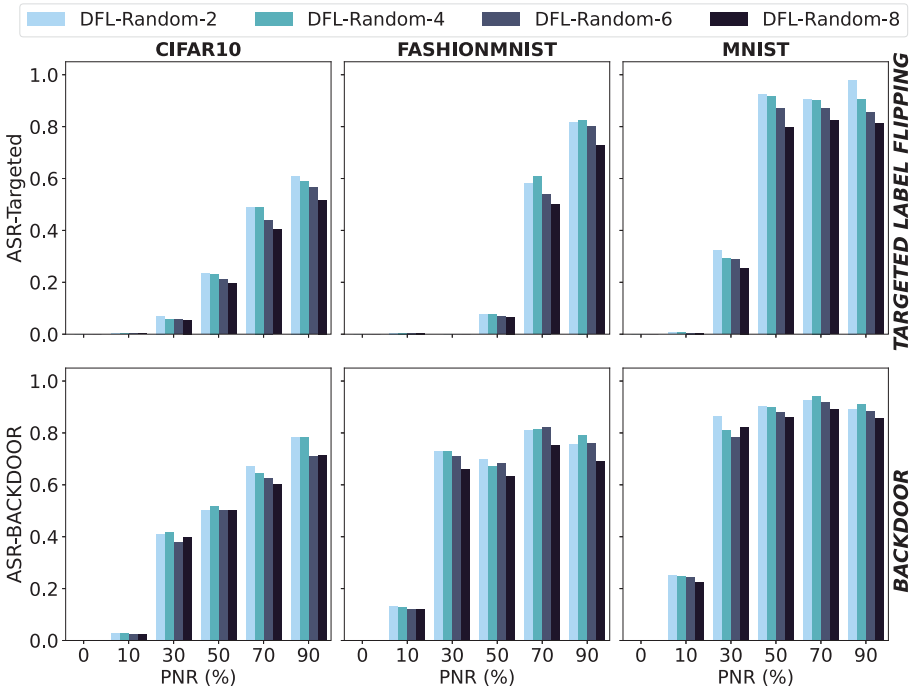

Fig. 11. Average ASR Results for Targeted Poisoning Attacks with Random Topologies.

of samples that have a true label y_i but a predicted label $\hat{y}_j$, while L represents the set of labels in the dataset under consideration.

$$ASR - Backdoor = \frac{\sum_{j=0}^{|L|} c_{j,t} - c_{t,t}}{|D| - c_{t,t}} \tag{2}$$

The ASR metrics are calculated as the mean result across all benign participants, with higher values indicating greater effectiveness of the targeted attack.

(Ii) attack effectiveness.

Fig. 10 presents the average ASR metrics of Targeted Label Flipping and Backdoor attacks on CFL and DFL with three datasets as the PNR increases. Based on the results, in terms of attack effectiveness, the Backdoor Attack, which utilizes manipulated data samples, is found to be more effective than the Targeted Label Flipping attack, which uses manipulated labels. From the experimental results, it becomes evident that more than 30% of nodes injected with a backdoor can lead to a backdoor injection success rate of over 50% in all three datasets. Targeted Label Flipping requires at least 50% of the nodes to be attacked to achieve similar results, and if the PNR is below 30%, the influence on the models is considered negligible.

(Iii) model robustness in different paradigms and topologies.

As in the case of untargeted attacks, CFL and DFL with fully connected topology demonstrate similar model robustness against targeted attacks. When the poisoned node proportion is low, the influence of this type of attack on the overall performance of the federation is insignificant, particularly in the case of Targeted Label Flipping.

This section evaluates the impact of different topologies on targeted poisoning attacks by implementing the same setups of random topology experiments of untargeted attacks. The results of this experiment are shown in Fig. 11. Similar to untargeted attacks, topology has an impact on the effectiveness of targeted attacks. A denser network facilitates the spread of targeted attacks, but the aggregation of more models also has a negative effect on the maliciousness. Thus, a higher ASR is observed in a low-density network than in a high-density network. However, for the Backdoor Attack, once the PNR exceeds 50%, the impact of topology becomes less significant.

To summarize, Backdoor attacks by manipulating data are more effective than Target Label Flipping attacks by manipulating labels. Meanwhile, similar to untargeted attacks, the robustness of the model against targeted poisoning attacks for CFL and DFL with fully connected topology is roughly the same. Besides, the topology of the network plays a crucial role in the spread of maliciousness. When benign nodes are directly connected to malicious nodes, the more sparse the network, the more vulnerable they are to attacks.

5.2.3. Benchmark of defense mechanisms for untargeted poisoning attacks

This section conducts extensive experiments to assess the effectiveness of defense mechanisms against poisoning attacks. It specifically looks at how well these mechanisms, initially designed for CFL, can adapt to DFL. The experiments benchmark these defense mechanisms across diverse datasets against various attack strategies.

This benchmark includes a variety of defense mechanisms, including:

- ***FedAvg***: Averaging over the received models, designed for CFL, without providing additional protection.
- ***FLTrust***: Filtering anomalous models using ReLU-clipped cosine similarity, designed for CFL.
- ***Krum***: Finding the model with the shortest distance to all other models as the aggregated model, designed for CFL.
- ***Median***: Finding the median model of all models as the aggregated model, designed for CFL.
- ***Sentinel***: Combining similarity and loss-based methods to filter and regularize models, designed for DFL.
- ***TrimmedMean***: Filtering the extreme values in the received models and then averaging them to get the aggregated model, designed for CFL.
- ***Voyager***: An MTD-based defense that isolates malicious nodes by means of dynamic topology, designed for DFL.

The first experiment seeks to evaluate the effectiveness of these defense mechanisms in mitigating Untargeted Label Flipping, Untargeted Sample Poisoning, and Random Model Poisoning attacks. The effectiveness of the defense mechanisms is assessed using the F1-Score, with higher values indicating better protection against adversarial attacks.

Table 4
Benchmark of average F1-Score results for defense mechanisms in mitigating untargeted poisoning attacks.

		[PNR (%)]	Label Flipping					Model Poisoning					Sample Poisoning				
			10	30	50	70	90	10	30	50	70	90	10	30	50	70	90
CIFAR10	CFL	*FedAvg*	**69.2%**	**67.8%**	1.7%	1.9%	1.9%	1.8%	1.8%	2.0%	1.7%	2.0%	**71.2%**	58.7%	20.9%	8.0%	3.3%
		FLTrust	60.1%	61.0%	59.6%	56.2%	29.5%	41.4%	54.7%	50.2%	1.8%	2.9%	63.2%	59.0%	49.8%	39.6%	2.2%
		Krum	62.7%	63.6%	**61.8%**	**60.8%**	30.2%	52.9%	**63.6%**	**61.8%**	1.8%	1.9%	65.2%	**63.3%**	**61.8%**	**43.8%**	1.9%
		Median	68.9%	52.7%	55.0%	47.3%	**32.5%**	**60.5%**	40.3%	18.2%	**14.3%**	6.3%	62.7%	43.1%	57.4%	1.8%	**27.1%**
		Sentinel	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
		TrimmedMean	60.6%	54.9%	53.0%	47.3%	32.2%	2.6%	22.5%	7.3%	1.8%	**8.0%**	68.1%	54.6%	13.7%	32.7%	4.2%
		Voyager	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	DFL-Fully Connected	*FedAvg*	**72.9%**	**70.4%**	1.8%	1.8%	1.9%	1.8%	1.8%	1.9%	1.8%	2.0%	69.9%	58.9%	21.9%	8.0%	3.2%
		FLTrust	51.9%	52.2%	34.4%	1.8%	1.8%	52.3%	51.3%	1.9%	1.8%	1.8%	68.1%	61.9%	38.5%	2.5%	1.8%
		Krum	65.6%	65.8%	43.8%	2.0%	1.8%	66.0%	**67.7%**	1.8%	1.8%	1.9%	66.5%	65.3%	47.8%	1.8%	1.8%
		Median	66.1%	68.4%	12.5%	1.8%	1.9%	**70.1%**	18.7%	1.8%	1.7%	1.6%	70.3%	61.2%	39.0%	3.6%	2.3%
		Sentinel	66.4%	66.0%	**67.0%**	**65.9%**	**67.4%**	67.1%	65.5%	**67.1%**	**66.8%**	**67.4%**	66.1%	**67.4%**	**65.4%**	**66.4%**	**67.4%**
		TrimmedMean	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.9%	1.7%	**72.1%**	56.7%	11.9%	6.1%	1.8%
		Voyager	63.8%	64.6%	65.0%	62.8%	65.5%	65.4%	64.0%	65.2%	64.3%	64.3%	65.0%	65.9%	63.3%	63.9%	65.2%
	DFL-Ring	*FedAvg*	64.0%	62.6%	1.6%	1.6%	1.8%	1.7%	1.6%	1.6%	1.6%	1.7%	64.1%	54.2%	19.4%	7.4%	1.7%
		FLTrust	52.8%	51.3%	21.8%	1.8%	1.8%	53.1%	52.2%	3.5%	1.8%	1.7%	64.8%	62.2%	45.3%	5.1%	4.6%
		Krum	65.9%	65.3%	27.1%	1.9%	1.9%	67.0%	**66.8%**	4.0%	1.8%	1.7%	67.2%	64.8%	51.2%	2.3%	5.4%
		Median	**72.7%**	**70.5%**	63.7%	1.9%	1.8%	**71.0%**	4.6%	1.8%	2.0%	1.9%	69.6%	66.3%	59.3%	8.5%	4.3%
		Sentinel	65.7%	65.6%	**66.3%**	**65.1%**	**65.4%**	67.0%	66.1%	**66.5%**	**66.5%**	**66.6%**	64.5%	**67.0%**	**66.3%**	**65.5%**	**67.0%**
		TrimmedMean	1.8%	1.8%	1.8%	1.8%	1.9%	2.1%	1.9%	1.8%	1.7%	1.7%	**70.2%**	61.5%	27.4%	15.9%	1.8%
		Voyager	63.2%	62.9%	64.2%	63.6%	65.2%	64.2%	63.0%	65.1%	65.0%	63.8%	62.6%	64.5%	64.1%	64.5%	64.6%
	DFL-Star	*FedAvg*	62.9%	1.7%	1.6%	1.6%	1.7%	1.6%	1.6%	1.7%	1.6%	1.7%	57.5%	52.2%	19.0%	3.1%	3.0%
		FLTrust	53.8%	49.3%	52.5%	1.7%	1.8%	50.1%	51.5%	1.8%	1.9%	2.0%	62.2%	59.3%	53.2%	4.6%	1.8%
		Krum	69.1%	**63.9%**	65.4%	1.8%	1.8%	64.8%	**67.2%**	1.9%	1.9%	2.0%	63.6%	62.7%	60.4%	1.7%	1.8%
		Median	**71.2%**	24.5%	36.7%	1.8%	1.9%	**71.0%**	1.9%	1.9%	1.8%	1.6%	**71.1%**	36.1%	30.5%	4.9%	2.1%
		Sentinel	67.5%	65.5%	**66.4%**	**65.5%**	**65.6%**	66.8%	66.3%	**67.0%**	**67.0%**	**67.6%**	66.2%	**67.8%**	**68.4%**	**66.3%**	**67.1%**
		TrimmedMean	2.0%	1.8%	1.8%	1.8%	1.9%	3.5%	1.9%	1.7%	1.8%	2.1%	70.5%	60.0%	34.6%	16.7%	2.2%
		Voyager	63.7%	65.3%	65.8%	63.8%	64.6%	65.6%	64.2%	64.9%	64.3%	65.1%	65.0%	64.6%	64.9%	63.7%	64.0%
FASHIONMNIST	CFL	*FedAvg*	80.5%	75.0%	48.5%	1.7%	0.5%	1.8%	1.9%	1.8%	1.8%	1.7%	**82.1%**	71.4%	70.4%	64.5%	55.2%
		FLTrust	71.6%	69.0%	77.3%	59.5%	40.8%	76.6%	76.0%	72.5%	46.0%	28.6%	70.3%	68.8%	67.7%	61.9%	46.0%
		Krum	72.4%	68.9%	**80.9%**	60.8%	**42.0%**	**81.0%**	**81.8%**	**80.8%**	44.4%	26.1%	72.6%	69.9%	67.9%	62.7%	42.4%
		Median	**81.4%**	77.9%	80.0%	62.0%	0.3%	80.2%	70.7%	67.7%	23.0%	39.2%	80.9%	**81.2%**	78.8%	**68.9%**	40.5%
		Sentinel	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
		TrimmedMean	81.3%	**80.8%**	80.0%	**65.7%**	41.2%	72.6%	65.8%	63.5%	**59.1%**	**40.7%**	80.3%	78.8%	**80.2%**	65.7%	**66.6%**
		Voyager	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	DFL-Fully Connected	*FedAvg*	84.5%	79.4%	47.9%	1.8%	0.5%	1.8%	1.8%	1.8%	1.9%	1.8%	83.8%	71.2%	69.2%	64.1%	57.0%
		FLTrust	80.8%	67.0%	2.0%	0.8%	1.6%	67.8%	64.1%	63.9%	1.8%	1.8%	80.3%	78.6%	14.7%	10.6%	11.4%
		Krum	83.7%	69.4%	0.3%	0.4%	0.5%	**83.2%**	**83.0%**	**82.5%**	1.8%	1.9%	**83.7%**	**81.6%**	2.9%	1.9%	1.3%
		Median	84.1%	**79.6%**	55.9%	0.7%	0.7%	75.8%	20.6%	1.7%	1.8%	2.0%	83.4%	76.5%	64.1%	47.0%	9.4%
		Sentinel	77.9%	75.6%	**79.3%**	**74.3%**	**75.2%**	77.2%	75.5%	76.0%	**75.6%**	**76.6%**	76.7%	76.3%	**77.8%**	**74.7%**	**75.4%**
		TrimmedMean	**84.7%**	78.5%	9.2%	2.3%	6.1%	13.9%	1.9%	1.8%	1.9%	1.8%	82.4%	76.3%	69.5%	48.1%	55.7%
		Voyager	75.2%	73.7%	75.7%	73.3%	74.9%	75.3%	73.8%	73.3%	74.2%	73.7%	74.2%	72.6%	75.5%	72.9%	72.7%
	DFL-Ring	*FedAvg*	74.4%	55.0%	4.4%	1.6%	0.2%	1.7%	1.6%	1.6%	1.6%	1.5%	75.8%	63.6%	60.0%	47.6%	44.7%
		FLTrust	81.3%	69.8%	12.1%	0.7%	61.0%	64.2%	36.2%	1.8%	1.8%	1.8%	80.6%	45.4%	46.2%	27.1%	16.6%
		Krum	83.9%	71.1%	0.9%	0.5%	77.3%	84.2%	45.9%	1.9%	1.8%	1.9%	83.1%	40.3%	40.8%	21.4%	7.3%
		Median	**84.7%**	**83.5%**	55.0%	41.5%	0.0%	**85.2%**	50.7%	1.9%	1.9%	1.9%	**84.9%**	73.2%	69.6%	75.3%	34.9%
		Sentinel	82.9%	76.7%	**75.8%**	**78.3%**	**77.2%**	81.3%	**75.1%**	**76.3%**	**75.8%**	**75.6%**	76.2%	**77.6%**	**77.3%**	**76.2%**	**77.2%**
		TrimmedMean	84.6%	79.8%	59.0%	1.5%	0.8%	1.8%	5.3%	1.9%	1.9%	1.9%	84.3%	74.6%	72.1%	53.9%	56.4%
		Voyager	74.2%	75.0%	75.0%	75.4%	74.7%	72.7%	73.4%	74.7%	75.6%	72.8%	73.0%	74.3%	74.8%	73.3%	75.5%
	DFL-Star	*FedAvg*	73.1%	47.6%	44.1%	1.6%	0.5%	1.6%	1.6%	1.6%	1.7%	1.5%	74.6%	64.9%	53.1%	22.8%	45.3%
		FLTrust	81.8%	14.8%	68.0%	26.5%	0.4%	65.3%	66.0%	23.4%	1.7%	1.7%	81.8%	76.7%	77.6%	11.6%	12.1%
		Krum	80.7%	0.6%	**80.2%**	27.6%	0.4%	**83.5%**	83.3%	30.0%	1.8%	1.7%	**84.4%**	**80.9%**	**81.6%**	2.2%	3.3%
		Median	84.2%	**84.4%**	77.6%	0.2%	0.3%	**85.0%**	31.5%	1.8%	1.9%	1.8%	72.8%	72.1%	62.9%	27.3%	73.4%
		Sentinel	78.1%	75.0%	76.9%	**76.0%**	**76.7%**	76.8%	76.9%	**76.6%**	**76.7%**	**76.9%**	77.3%	75.8%	77.9%	**76.4%**	**75.7%**
		TrimmedMean	**84.6%**	73.0%	40.5%	29.7%	0.2%	1.8%	3.8%	1.9%	1.8%	1.9%	83.4%	79.2%	61.6%	49.9%	52.8%
		Voyager	74.5%	73.6%	74.1%	72.2%	73.5%	73.6%	74.6%	73.9%	74.5%	74.0%	74.4%	73.5%	76.6%	75.1%	73.7%
MNIST	CFL	*FedAvg*	**95.3%**	**94.7%**	80.5%	35.3%	0.8%	18.8%	15.0%	1.8%	1.6%	1.5%	**99.8%**	93.0%	90.9%	**88.0%**	**73.6%**
		FLTrust	88.6%	89.9%	87.2%	67.6%	43.9%	88.3%	86.2%	88.8%	62.6%	24.8%	92.0%	81.0%	77.2%	71.7%	46.8%
		Krum	93.5%	92.8%	93.2%	69.5%	44.7%	93.3%	**93.2%**	**93.9%**	64.6%	21.8%	92.7%	79.5%	77.2%	70.2%	47.4%
		Median	94.1%	93.5%	**95.4%**	70.0%	44.4%	**93.9%**	85.2%	83.7%	**68.4%**	**48.3%**	94.4%	**93.3%**	92.9%	71.4%	48.0%
		Sentinel	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
		TrimmedMean	93.7%	80.7%	91.1%	**88.0%**	**46.0%**	90.6%	78.0%	77.1%	66.3%	40.9%	**93.3%**	92.7%	93.3%	83.1%	53.9%
		Voyager	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	DFL-Fully Connected	*FedAvg*	94.1%	95.9%	82.7%	35.2%	0.9%	19.5%	15.4%	1.8%	1.8%	1.6%	**97.3%**	96.3%	93.9%	91.3%	75.8%
		FLTrust	93.0%	90.0%	48.7%	32.8%	0.3%	78.6%	77.2%	57.2%	1.7%	1.7%	88.3%	90.0%	19.9%	21.9%	18.7%
		Krum	95.8%	95.6%	47.4%	30.5%	0.3%	95.6%	**95.8%**	72.7%	1.7%	1.8%	95.6%	93.2%	2.1%	4.6%	2.2%
		Median	97.1%	**96.0%**	75.3%	18.4%	0.1%	96.7%	76.0%	1.6%	1.7%	1.8%	95.3%	94.6%	**93.2%**	60.4%	2.2%
		Sentinel	95.1%	85.9%	**88.9%**	**88.4%**	**89.2%**	**97.8%**	89.2%	**88.3%**	**87.7%**	**87.6%**	95.6%	87.4%	**86.0%**	86.9%	**87.9%**
		TrimmedMean	**97.3%**	88.1%	67.4%	45.2%	0.4%	23.2%	1.7%	1.8%	1.8%	1.9%	96.0%	**96.8%**	92.7%	**93.2%**	80.3%
		Voyager	93.8%	84.8%	86.2%	84.7%	85.6%	94.4%	86.1%	86.0%	84.8%	84.7%	94.1%	85.0%	83.2%	84.4%	87.2%
	DFL-Ring	*FedAvg*	88.0%	87.5%	42.3%	15.5%	0.4%	8.2%	1.7%	1.7%	1.5%	1.4%	88.0%	85.9%	84.0%	81.3%	64.6%
		FLTrust	94.9%	19.1%	18.2%	75.2%	0.4%	27.8%	74.5%	17.9%	1.8%	1.8%	94.3%	88.0%	78.2%	85.6%	16.5%
		Krum	96.1%	1.6%	1.2%	**94.3%**	0.3%	36.0%	**93.8%**	23.0%	1.9%	1.9%	95.6%	90.4%	80.7%	86.3%	2.0%
		Median	96.9%	**95.7%**	86.8%	72.1%	0.4%	94.8%	67.6%	1.8%	1.8%	1.9%	**97.0%**	95.7%	86.4%	74.9%	2.0%
		Sentinel	96.5%	87.4%	87.1%	88.4%	**86.5%**	**96.7%**	88.4%	**87.2%**	**88.3%**	**88.2%**	90.7%	84.5%	87.3%	87.8%	**90.3%**
		TrimmedMean	**97.4%**	95.5%	**88.3%**	5.9%	1.1%	1.9%	1.8%	1.8%	1.8%	1.9%	96.9%	**96.8%**	**95.3%**	**90.5%**	73.0%
		Voyager	95.2%	84.0%	83.9%	85.9%	84.9%	83.7%	85.3%	83.9%	85.0%	86.7%	86.5%	83.8%	86.0%	86.6%	86.2%
	DFL-Star	*FedAvg*	86.5%	85.7%	46.8%	9.9%	0.8%	1.7%	3.4%	1.6%	1.5%	1.4%	88.0%	84.5%	82.6%	77.4%	66.8%
		FLTrust	89.2%	93.4%	89.6%	10.2%	0.7%	59.9%	55.4%	43.3%	1.7%	1.7%	91.8%	93.4%	87.1%	18.2%	17.4%
		Krum	95.2%	95.0%	**94.5%**	0.3%	0.6%	74.8%	69.6%	55.6%	1.8%	1.8%	94.4%	94.2%	**93.5%**	2.0%	2.3%
		Median	**97.2%**	**96.5%**	67.9%	50.0%	1.1%	**97.2%**	1.7%	1.7%	1.9%	1.7%	**97.1%**	93.8%	79.4%	70.7%	2.1%
		Sentinel	86.9%	85.6%	87.0%	**88.5%**	**90.0%**	86.7%	**88.0%**	**88.3%**	**88.2%**	**87.7%**	87.3%	87.9%	87.5%	87.7%	84.4%
		TrimmedMean	91.8%	95.5%	85.2%	51.2%	1.0%	6.7%	6.9%	1.8%	1.8%	1.7%	**97.1%**	**96.0%**	91.0%	**89.0%**	82.2%
		Voyager	85.0%	84.2%	86.8%	85.4%	87.4%	85.3%	86.2%	86.4%	85.0%	84.4%	84.5%	84.4%	83.0%	86.6%	**84.7%**

Table 5
Benchmark of average ASR-targeted results for defense mechanisms in mitigating targeted label flipping Attack.

	[PNR (%)]	CIFAR10					FASHIONMNIST					MNIST				
		10	30	50	70	90	10	30	50	70	90	10	30	50	70	90
CFL	*FedAvg*	0.3%	4.5%	18.0%	39.0%	47.9%	0.4%	0.0%	6.4%	48.0%	66.9%	0.5%	24.6%	79.9%	76.6%	77.5%
	FLTrust	0.2%	**4.2%**	**16.8%**	34.6%	**41.7%**	0.0%	1.6%	11.1%	17.6%	39.8%	1.0%	**0.8%**	15.6%	**21.6%**	48.4%
	Krum	0.3%	4.4%	17.1%	38.6%	41.8%	0.0%	0.0%	11.5%	19.2%	40.7%	1.0%	0.7%	16.5%	22.5%	49.6%
	Median	0.3%	4.3%	21.1%	46.7%	44.2%	0.0%	0.0%	**2.7%**	15.9%	**39.0%**	0.8%	1.2%	**3.0%**	22.9%	48.3%
	Sentinel	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	TrimmedMean	0.3%	4.8%	25.5%	**31.7%**	44.8%	0.0%	8.8%	9.9%	**14.8%**	41.2%	1.1%	1.2%	15.5%	22.9%	**47.3%**
	Voyager	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
DFL-Fully Connected	*FedAvg*	0.3%	4.9%	18.7%	41.0%	47.5%	0.4%	0.0%	6.2%	49.8%	65.9%	0.4%	25.5%	74.9%	76.5%	78.8%
	FLTrust	0.2%	4.3%	16.2%	35.6%	41.8%	0.0%	19.7%	12.8%	82.9%	82.1%	0.7%	0.9%	91.7%	91.5%	90.6%
	Krum	0.2%	4.4%	16.6%	42.7%	52.6%	0.0%	25.0%	0.0%	89.4%	84.1%	0.7%	0.4%	97.3%	96.3%	97.0%
	Median	0.2%	4.5%	15.3%	38.3%	60.0%	0.0%	0.6%	25.7%	76.3%	86.4%	0.7%	5.1%	22.2%	89.3%	97.8%
	Sentinel	0.3%	**0.2%**	**0.2%**	**0.2%**	**0.2%**	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	0.1%	**1.3%**	**1.2%**	**0.8%**	**1.6%**
	TrimmedMean	0.2%	4.4%	17.1%	47.1%	45.6%	0.0%	0.8%	67.1%	75.9%	86.9%	0.7%	3.0%	75.8%	96.8%	95.3%
	Voyager	0.2%	4.3%	16.6%	36.7%	42.3%	0.3%	0.0%	5.4%	44.0%	57.6%	0.4%	22.4%	64.7%	65.0%	68.7%
DFL-Ring	*FedAvg*	0.1%	4.3%	17.6%	40.9%	61.6%	0.0%	0.8%	0.4%	42.7%	84.9%	0.6%	19.9%	81.9%	93.1%	96.3%
	FLTrust	0.2%	4.3%	16.5%	35.8%	43.3%	0.0%	0.1%	41.2%	57.8%	82.1%	0.9%	12.3%	78.8%	17.1%	91.9%
	Krum	0.3%	6.5%	20.0%	32.5%	48.8%	0.0%	0.0%	47.9%	56.6%	84.5%	0.9%	15.3%	97.7%	1.0%	95.2%
	Median	0.2%	5.1%	15.3%	42.1%	42.6%	0.0%	0.0%	62.1%	59.7%	72.6%	0.8%	4.3%	56.3%	96.0%	94.5%
	Sentinel	0.3%	**0.2%**	**0.2%**	**0.2%**	**0.2%**	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	**0.5%**	**1.0%**	**1.0%**	**0.8%**	**1.1%**
	TrimmedMean	0.2%	5.0%	17.9%	47.4%	41.4%	0.0%	0.3%	19.9%	70.1%	83.5%	0.9%	1.8%	9.8%	96.8%	95.3%
	Voyager	0.2%	4.4%	16.2%	35.1%	41.8%	0.3%	0.0%	5.5%	44.3%	57.9%	0.4%	22.1%	65.0%	66.1%	68.5%
DFL-Star	*FedAvg*	0.3%	6.3%	20.9%	51.3%	63.7%	0.5%	0.0%	7.6%	61.5%	84.5%	0.6%	32.2%	93.7%	96.3%	98.0%
	FLTrust	0.2%	4.3%	16.1%	35.8%	41.4%	15.9%	0.0%	79.7%	88.1%	85.2%	0.4%	1.8%	17.9%	17.4%	94.1%
	Krum	0.3%	5.3%	19.0%	45.5%	39.1%	20.3%	0.0%	89.1%	91.6%	92.1%	0.4%	1.6%	1.4%	**0.7%**	99.1%
	Median	0.3%	5.7%	20.8%	46.4%	48.0%	4.1%	64.5%	56.4%	78.3%	0.0%	0.8%	0.8%	98.2%	93.5%	97.8%
	Sentinel	0.3%	**0.2%**	**0.2%**	**0.2%**	**0.2%**	0.0%	**0.0%**	**0.0%**	**0.0%**	**0.0%**	0.5%	**1.0%**	**1.0%**	0.9%	**1.2%**
	TrimmedMean	0.3%	5.6%	18.2%	42.8%	42.0%	0.0%	0.2%	60.9%	79.8%	78.6%	0.6%	3.1%	95.0%	95.4%	98.0%
	Voyager	0.2%	4.2%	16.0%	36.2%	42.3%	0.3%	0.0%	5.5%	44.0%	56.2%	0.4%	22.5%	64.6%	65.4%	65.1%

Table 6
Benchmark of average ASR-backdoor results for defense mechanisms in mitigating backdoor attack.

	[PNR (%)]	CIFAR10					FASHIONMNIST					MNIST				
		10	30	50	70	90	10	30	50	70	90	10	30	50	70	90
CFL	*FedAvg*	**2.5%**	36.3%	46.5%	62.7%	69.8%	12.3%	67.7%	65.4%	74.4%	70.3%	22.2%	78.9%	91.1%	87.6%	85.9%
	FLTrust	3.9%	**8.3%**	**6.4%**	**33.2%**	**49.6%**	**1.0%**	21.4%	29.1%	39.5%	70.5%	3.2%	53.1%	60.2%	57.9%	63.7%
	Krum	2.5%	34.3%	48.8%	51.0%	68.3%	**1.0%**	25.9%	34.6%	46.4%	73.5%	**2.5%**	59.4%	65.3%	**57.6%**	**57.1%**
	Median	2.7%	33.6%	34.6%	59.6%	73.2%	1.2%	4.9%	20.8%	**14.9%**	**42.6%**	5.4%	**11.3%**	**24.1%**	59.0%	97.1%
	Sentinel	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	TrimmedMean	2.5%	35.3%	39.9%	65.5%	70.1%	**1.0%**	**4.7%**	**13.0%**	23.4%	66.5%	6.9%	31.1%	36.9%	70.0%	97.0%
	Voyager	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
DFL-Fully Connected	*FedAvg*	2.6%	38.3%	46.7%	60.2%	71.6%	12.3%	65.2%	62.7%	70.8%	72.1%	22.7%	77.2%	87.8%	93.1%	83.1%
	FLTrust	3.5%	**18.3%**	61.2%	60.3%	91.8%	10.8%	35.9%	74.6%	63.7%	75.6%	36.1%	77.6%	94.1%	68.8%	96.1%
	Krum	**2.3%**	36.9%	**37.0%**	58.5%	61.5%	9.9%	32.9%	78.2%	60.5%	78.1%	37.5%	78.5%	93.3%	65.4%	99.6%
	Median	2.6%	31.3%	45.2%	60.5%	71.0%	3.1%	61.0%	74.4%	89.1%	84.5%	12.0%	70.5%	97.8%	96.8%	98.5%
	Sentinel	**2.3%**	38.5%	36.8%	53.5%	67.5%	**1.0%**	**1.0%**	**1.2%**	**1.0%**	**1.0%**	**3.2%**	**5.4%**	**2.5%**	**3.2%**	**6.9%**
	TrimmedMean	2.6%	36.9%	44.5%	**48.4%**	65.5%	17.1%	53.4%	72.5%	85.3%	74.6%	39.8%	92.5%	98.8%	97.9%	99.0%
	Voyager	**2.3%**	32.8%	40.9%	52.1%	**62.4%**	10.7%	57.2%	54.6%	62.2%	62.7%	19.2%	68.1%	78.8%	80.7%	70.3%
DFL-Ring	*FedAvg*	2.9%	42.4%	51.1%	65.6%	78.5%	12.5%	73.1%	70.1%	83.6%	80.2%	24.5%	88.2%	95.8%	94.6%	97.5%
	FLTrust	3.0%	32.3%	64.5%	80.8%	85.9%	3.0%	28.2%	64.0%	84.9%	80.7%	11.7%	93.0%	92.3%	96.4%	90.3%
	Krum	3.0%	**31.1%**	62.0%	53.6%	69.1%	1.4%	23.5%	65.7%	90.6%	83.6%	5.6%	98.6%	97.7%	99.3%	90.8%
	Median	2.7%	38.9%	42.7%	**44.1%**	70.3%	6.3%	54.1%	72.2%	86.6%	86.7%	18.8%	67.0%	78.5%	98.7%	99.1%
	Sentinel	2.5%	34.5%	50.9%	76.3%	81.3%	**1.2%**	**1.0%**	**1.2%**	**1.0%**	**1.0%**	**2.5%**	**7.5%**	**5.3%**	**11.5%**	**5.8%**
	TrimmedMean	2.8%	41.4%	53.9%	52.5%	61.8%	9.6%	50.6%	60.9%	79.1%	81.9%	38.7%	89.5%	98.0%	99.7%	98.5%
	Voyager	**2.2%**	32.5%	**40.9%**	52.2%	**61.1%**	11.2%	58.6%	54.5%	61.4%	63.1%	19.6%	65.9%	76.4%	80.9%	72.1%
DFL-Star	*FedAvg*	2.8%	41.8%	52.6%	68.3%	78.5%	13.2%	73.1%	70.1%	83.6%	79.1%	25.2%	86.7%	95.0%	95.0%	92.6%
	FLTrust	4.7%	44.6%	61.9%	72.9%	87.5%	1.7%	55.8%	70.4%	72.5%	78.7%	7.5%	76.0%	91.1%	96.0%	84.6%
	Krum	2.6%	35.4%	42.3%	56.2%	80.8%	**0.9%**	59.4%	70.8%	73.6%	81.3%	**2.4%**	72.8%	92.1%	97.4%	88.0%
	Median	**2.1%**	40.2%	**38.6%**	61.8%	66.1%	2.8%	58.1%	73.0%	85.8%	68.0%	41.9%	88.2%	97.7%	98.5%	97.6%
	Sentinel	2.9%	**31.7%**	44.9%	56.4%	76.9%	2.7%	**4.9%**	**5.4%**	**6.4%**	**7.1%**	13.0%	**6.0%**	**11.7%**	**13.0%**	**5.9%**
	TrimmedMean	2.2%	46.1%	56.2%	52.6%	76.8%	5.3%	59.3%	74.0%	80.0%	80.0%	28.9%	94.6%	99.1%	98.7%	98.6%
	Voyager	2.3%	33.0%	40.1%	**51.8%**	**61.0%**	10.5%	57.6%	54.7%	61.6%	62.6%	19.6%	68.2%	77.9%	79.8%	72.6%

For those defense mechanisms originally designed for CFL, such as *FedAvg*, *FLTrust*, *Krum*, *Median*, and *TrimmedMean*, they are tested on both CFL and DFL. For the defense mechanisms designed for DFL, *Sentinel* and *Voyager*, since they do not work on CFL, the testing is done on DFL only.

(I) defense effectiveness.

The average F1-Score of the defense mechanisms under three datasets, three attack strategies, and different PNRs is shown in Table 4. Overall, *Sentinel* achieves the best defense against untargeted attacks and is able to maintain the maximum model robustness, with *Voyager* coming in second. However, when the percentage of poisoned nodes exceeds 50%, most defenses prove to be ineffective. This is due to Byzantine-robust aggregation techniques, such as*Krum*, *Median*, and *TrimmedMean*, being more likely to select malicious models over benign ones in such scenarios. Although *FLTrust* demonstrates a good result at low PNR, the results show that it does not exhibit an advantage for Byzantine-robust aggregation techniques. These defense strategies exhibit similar outcomes to *FedAvg* across different network topologies and datasets. Meanwhile, the defense mechanism is more effective against Sample Poisoning and Label Flipping than Model Poisoning. Except for *Sentinel* and *Voyager*, most defense mechanisms are ineffective in mitigating Model Poisoning attacks.

(Ii) compatibility in DFL.

In contrast to CFL, the network topologies influence the effectiveness of Byzantine defense strategies and *FLTrust* in DFL. A smaller average number of connected neighbors in the network results in malicious nodes posing a greater threat to their directly connected benign nodes. In such cases, benign nodes are more likely to aggregate with more than 50% of malicious neighbors, rendering Byzantine-robust aggregation techniques ineffective. It can be seen from the results in Table 4 that the results in the ring and star topologies are worse than the fully connected ones. Therefore, those defense mechanisms designed for CFL have some difficulties adapting to DFL, and they are more effective in dense topologies.

To conclude, when the FL system contains a small number of malicious nodes, all defense mechanisms prove to be effective. However, if the percentage of malicious nodes exceeds 50%, the Byzantine-robust defenses and *FLTrust* become ineffective. In contrast, the DFL-focused defense mechanisms, Voyager and Sentinel, are capable of effectively countering attacks from a high percentage of malicious nodes. Furthermore, the network's topology also plays a role in the effectiveness of the defenses, with Byzantine-robust defenses and *FLTrust* being less effective in sparse networks but performing better in dense networks.

5.2.4. Benchmark of defense mechanisms for targeted poisoning attacks

As analyzed in Section 5.2.2, targeted attacks are more difficult to detect. Therefore, this section experimentally benchmarks the performance of defense mechanisms listed in Section 5.2.3 under targeted attacks, including Targeted Label Flipping and Backdoor Attacks.

(I) defense effectiveness for targeted label flipping.

Overall, *Krum*, *TrimmedMean*, *Median*, *Voyager*, and *FLTrust* are not effective against Targeted Label Flipping attacks, as shown in the Table 5. The reason is that since the Targeted Label Flipping attacks have a small impact on the overall model, it is difficult for distance-based or extreme value filtering defenses to work. In contrast, *Sentinel*'s loss-based layer-wise normalization mechanism is much more effective against Targeted Label Flipping, and the experimental results show that *Sentinel* can effectively prevent the spread of Targeted Label Flipping attacks.

(Ii) defense effectiveness for backdoor attack.

According to the data presented in the Table 6, the defense mechanisms exhibit weaknesses when confronted with the Backdoor Attack. In general, *Sentinel* successfully counters the Backdoor Attack on the MNIST and FashionMNIST datasets. However, it fails to effectively address this attack on the Cifar10 dataset, which task is more complex and has a larger number of model parameters. The reason behind this failure is that in more complex models, the implanted backdoor has a limited effect on both the similarity and the loss of the model. Therefore, neither the Byzantine-robust defense mechanisms nor the hybrid *Sentinel* and *FLTrust* approaches are able to mitigate this type of attack effectively.

In conclusion, compared to untargeted attacks, targeted attacks are more difficult to defend, and the Byzantine-robust defense is often ineffective. On the contrary, the combination of model similarity and loss-based layer-wise normalization, *i.e., Sentinel*, can effectively defend against targeted attacks.

6. Lessons learned, open challenges, and research opportunities

Drawing on the analysis of the diverse model robustness of DFL in previous sections, this section endeavors to address the lessons learned and challenges that have surfaced in the research of DFL model robustness. Additionally, it suggests potential avenues for further research.

6.1. Lessons learned

Through the literature research and experimental analysis on both attack and defense sides, the following lessons were summarized:

- There is limited research on model robustness for DFL. The current research on model robustness is mainly focused on CFL. In addition, there is a lack of research on designing attacks optimized for DFL from an attack perspective or designing defense mechanisms to enhance the robustness of DFL models from a defense perspective.
- The model robustness degradation caused by untargeted attacks is more tangible than with targeted attacks. The latter have an insignificant impact on the overall effectiveness of the model and are more challenging to detect by common metrics. When considering attack strategy, sample poisoning is found to be less effective in untargeted attacks, but it exhibits higher efficacy in targeted attacks. In terms of the extent of harm inflicted by the attacks, untargeted attacks, particularly those involving model poisoning, result in significant damage.
- Topology plays a crucial role in determining the success of an attack in DFL. A densely connected network is more vulnerable to the spread of malicious attacks, while a sparse network can result in more severe consequences when under attack.
- The majority of defense mechanisms are ineffective in a high percentage (*i.e.,* more than 50%) of malicious nodes. Meanwhile, defense mechanisms designed for CFL encounter challenges when applied to DFL. Byzantine-robust countermeasures, such as *Krum* and *TrimmedMean*, have limited effectiveness in sparse DFL networks.
- Most defense mechanisms are insufficient when mitigating targeted attacks, either in CFL or DFL. The limited effect of targeted attacks on the model's parameters poses challenges for distance- or similarity-based defense mechanisms.

6.2. Application in practical scenarios

In general, current research on model robustness for FL, especially DFL, still tends to focus on the general problem rather than practical scenarios, i.e., most of the studies are validated on commonly used benchmark datasets, such as MNIST and FashionMNIST. There is a limited amount of research exploring the impacts of poisoning attacks on the robustness of FL models in practical scenarios, particularly in the fields of IoT, finance, and healthcare.

[62] discussed the feasibility of implementing poisoning attacks on FL systems in IoT environments and evaluated the hazards caused by poisoning attacks in an IoT Intrusion Detection System. [63] proposed a framework for Federated Reinforcement Learning (FRL) to recognize malware attacks and automatically implement mitigation strategies in IoT environments. They verified the robustness of their proposed FRL framework to data poisoning and model poisoning attacks. [64] and [65] provided an overview of the various problems associated with FL applications in medical and healthcare scenarios, including the problems associated with heterogeneities, and also the security problems associated with poisoning attacks. [66] proposed a framework for data poisoning attacks against healthcare FL, by comparing local models of different rounds and models from other nodes. In [67], a distributed backdoor attack approach was proposed and validated on a dataset in the financial domain. The research on FL in practical applications has garnered significant attention, yet there remains a noticeable absence of discourse surrounding the robustness of its models to both offensive and defensive sides.

Through the experimental validation in this paper, there are the following insights for the deployment of FL in practice scenarios, especially in the fields of IoT, healthcare, and finance:

- The decentralized nature of DFL results in a lack of model auditing and verification mechanisms for the federation, leading to potential security concerns that participants should be mindful of. While the FedAvg algorithm demonstrates strong performance in non-attacked scenarios, the presence of a malicious node within the federation can compromise the security of all nodes. To mitigate these risks, aggregation methods specifically tailored for DFL are advised, especially in scenarios where heightened security is required.
- Since overlay topology greatly impacts the model robustness of DFL, nodes should carefully consider which neighbors to connect and aggregate with. The remarkable performance of the *Voyager* algorithm, which relies on altering the connections with neighbors in DFL, indicates that employing a dynamic topology can effectively reduce the impact of poisoning attacks.
- Due to the minimal alterations made by targeted attacks, particularly backdoor attacks, conventional methods of model auditing that focus on performance or similarity between nodes have shown limited efficacy. Nevertheless, these attacks can have significant consequences in critical sectors like healthcare and finance. Therefore, additional audit analysis should be counted for targeted attacks, such as analyzing the stability of model results across categories.

6.3. Real-world applicability

When deploying the DART method for evaluating the robustness of DFL models in a real-world environment, the following difficulties pose limitations and difficulties:

- **Limited Datasets.** As mentioned in Section 6.2, current evaluations of DFL model robustness focus on commonly used datasets and lack domain datasets, especially real-world datasets. However, datasets may contain sensitive information, and sharing them directly may bring privacy and security concerns. Hence, the process of desensitizing and restructuring authentic data using techniques like data augmentation and synthetic data generation presents viable options [68]. These methods serve to eliminate private information within the raw data, while maintaining the raw data's characteristics and patterns. This, in turn, allows for a more thorough examination of the potential challenges in deploying DFL models in practical settings.
- **Realistic Attacks.** In the research on poisoning attacks, it is commonly assumed that malicious nodes can coordinate with each other to target benign nodes. However, in real-world scenarios of DFL, individual nodes often have different affiliations that complement each other. This poses a challenge for attackers, as the lack of synergy between malicious nodes can diminish the impact of their attacks or even cancel out the effect brought by the attack. Combining poisoning attacks with Sybil attacks can have a more significant impact, particularly when a large number of manipulated nodes are present in a DFL. Experimental findings indicate that a high proportion of malicious nodes within a federation can render defense mechanisms ineffective. Thus, if the Sybil attack has more than half of the manipulated nodes in the federation, defenses, including Krum and TrimmedMean, fail. Besides, a combination of poisoning attacks and Eclipse attacks can isolate a target benign node from other benign nodes. In such a case, the target node is surrounded by malicious nodes, and most of the defenses are ineffective. The same attacker can execute these two types of attacks and does not require collaboration with other attackers. However, they require attackers to know the overlay topology of the DFL in advance. Consequently, a topology manipulation defense mechanism can effectively defend against such attacks.
- **Collaboration of Defenses.** Decentralization not only poses a synchronization difficulty for the attacker but also a collaboration difficulty for the defender. Ideally, if a benign node in a DFL detects an attack, it should notify the other benign nodes to take appropriate defense measures. However, these collaborative messages can be intercepted and tampered with by the attacker, making collaboration among defenders difficult in real-world scenarios.DFL participants can adopt a more "selfish" defense, i.e., proactive defenses, to protect their models from attacks and not rely on notifications from other nodes. Possible proactive strategies include dynamically changing the aggregation algorithm, or proactively changing connected neighbors.

6.4. Ethical considerations

Although DFL offers advantages regarding privacy and effectiveness, it still has some ethical concerns:

- **Data Privacy.** Although in DFL, only the parameters or gradients of the model are transmitted through the network, studies have shown that this can still cause privacy leakage [10], [9], [14]. An attacker can infer privacy information from model parameters, model gradients, or model outputs, including the hyperparameters of model training, or restore the original training data. This type of attack is called an inference attack, bringing privacy concerns to DFL. Since the successful implementation of inference attacks relies on model overfitting, pruning and data augmentation techniques can be used to mitigate inference attacks. Besides, differential privacy or homomorphic encryption techniques can be used to protect the privacy-preserving capabilities of DFL models.
- **Fairness and Bias.** As a data-driven process, the bias of the data brings about the bias of the DFL model. Therefore, the fairness and bias of the DFL model are also important aspects of the ethical consideration of DFL. [69] proposed a qualitative analysis method for FL fairness, which can effectively measure the fairness of FL models.
- **Sustainability.** The training of DFL models consumes energy and produces greenhouse gases. Thus, it poses concerns in terms of sustainability. [70] proposed a framework for measuring the sustainability of FL, which calculates the sustainability of FL models by considering multiple aspects such as grid efficiency, hardware efficiency, and computational efficiency.

- **Compliance with Regulations.** DFL is required to comply to a range of data protection regulations, such as General Data Protection Regulation (GDPR) [71] and Health Insurance Portability and Accountability Act (HIPAA) [72]. These regulations were not specifically tailored to address the challenges faced by DFL, making compliance a complex undertaking. The concept of the right to be forgotten, as outlined in GDPR, presents particular challenges for DFL, as the deletion of user data from one node may not guarantee its removal from other nodes through model aggregation. Research on machine unlearning [73] is gaining attention as a potential avenue for DFL to align with data protection requirements.

6.5. Open challenges and opportunities

Regarding the lessons learned, application in practical scenarios, and ethical considerations, the following challenges and opportunities for model robustness research are identified:

- **Increased Attack Surfaces.** In CFL, only the central aggregator has a global view of the whole federation and has model information of all nodes. However, in DFL, depending on the topology, any node may have model information about all other nodes in the federation, which actually increases the security vulnerabilities. These increased attack surfaces pose a persistent challenge in preserving the robustness of DFL models. Therefore, it is crucial to customize defense mechanisms specifically for DFL systems. The robustness of DFL heavily relies on topology, suggesting that incorporating dynamic topology could serve as a viable defense mechanism. Additionally, leveraging nodes' local data and models within DFL systems can enhance the effectiveness of defense mechanisms against attacks.
- **Optimized Attacks.** In DFL, nodes receive models from their neighbors. Therefore, it is possible for malicious nodes to exploit the models received from their neighboring nodes as knowledge to optimize their attack strategies. By incorporating their own knowledge of defense mechanisms, these malicious nodes can manipulate the attack direction and increase the likelihood of achieving their objectives. This increases the difficulty and challenge for defense. In benign nodes, dynamic changes in aggregation functions can increase the difficulty of the attacker learning about the defense mechanism and increase the cost of executing the optimized attack. Besides, techniques such as homomorphic encryption can effectively increase the cost of such attacks.
- **Balance between Overhead and Robustness.** Incorporating defense mechanisms can enhance the robustness of DFL models, but it also presents challenges like increased network overhead, high computational resource usage, and scalability limitations. For example, when employing filtering-based strategies as a defense approach, setting a threshold too low may lead to ineffective filtering of malicious models, while setting the threshold too high may result in the exclusion of valuable models and the loss of benefits from information sharing. In terms of network traffic, defenses with inter-node connection operations may introduce communication overhead since they may change the overlay topology. Besides, more complex defenses may necessitate a higher number of computations, resulting in computation overhead and increased communication latency. This can lead to prolonged convergence times for the federation. It is crucial to balance overhead and robustness when designing defenses. MTD is a dynamic defense framework that enables defenders to adjust security levels based on specific requirements rather than striving for absolute security. As such, MTD is suggested as a design framework for DFL defenses to manage the trade-off between security and overhead effectively. Viable MTD strategies include model backup and rollback, dynamic topology, and dynamic aggregation. Furthermore, conducting data auditing and validation for each node before starting federated training can effectively protect against data poisoning attacks.
- **Data Heterogeneity.** As a data-driven process, performance of DFL relies on how the data is distributed across the nodes. It is often assumed that the data follows the IID settings when investigating the robustness of DFL models. Algorithms like Krum are based on the assumption that models trained on similar data should be highly similar, but anomalies are not. However, this assumption is invalidated in non-IID scenarios where data distribution across nodes is highly varsion. In non-IID settings, model similarity decreases, making it challenging to differentiate between benign and malicious models using similarity or clustering. In these cases, data-independent metrics can be utilized to detect malicious models. Rather than solely depending on model similarity, utilizing intrinsic model characteristics, such as eXplainable AI (XAI) metrics, can improve the performance of defense mechanisms while preserving the knowledge exchanged by non-IID nodes. Besides, multi-view learning has shown promising results in the fields of finance [74] and IoT [75]. It may represent a viable approach to enhancing the robustness of DFL models in non-IID environments.

6.6. Roadmap for future work

Based on the preceding discussion, the following aspects can serve as a guiding framework for future studies aimed at strengthening the robustness of DFL models:

- **Topology is the Key.** As mentioned several times above, overlay topology is the key factor affecting model robustness in DFL. Therefore, adopting the strategy of dynamic topology, either proactive or reactive, can mitigate the effects of different kinds of poisoning attacks while optimizing the model performance of DFL.
- **Dynamic Aggregation.** Advanced attacks optimize attack strategies for specific aggregation functions. Therefore, using dynamic aggregation functions, such as changing the aggregation algorithm in every round, can effectively prevent the knowledge of aggregation strategies from being identified by the attackers.
- **Endogenous Properties of Models.** Metrics such as model similarity strongly depend on the distribution of the data and thus are inefficient in the non-IID environment. The endogenous properties of the model, such as convergence and entropy and other metrics used in XAI, are a viable way to enhance the robustness of the model in the non-IID environment.
- **Data Augmentation and Synthetic Data.** Utilizing techniques such as data augmentation and synthetic data generation can maintain the integrity of the original data while also ensuring data privacy. Consequently, these approaches can be employed in DFL to generate synthetic data that balances data distribution among nodes. By leveraging the synthetic data generated, it is possible to approximate non-IID as IID, thereby enabling the detection and mitigation of poisoning attacks.

7. Conclusion

DFL is gaining prominence as a novel paradigm for achieving stable, collaborative, and privacy-preserving ML. Its distinctive architecture renders it vulnerable to various forms of malicious attacks, particularly poisoning attacks. There is a growing interest in researching model robustness in FL. However, existing research, which explores both offensive and defensive approaches, mainly concentrates on investigating the CFL paradigm. Therefore, there is a great need for studies on model robustness for DFL paradigm.

Therefore, this paper provides a comprehensive literature review of poisoning attacks that target model robustness in the DFL paradigm, along with the corresponding defense mechanisms. Additionally, a novel module named *DART* is introduced to assess the robustness of DFL models, which is then implemented and integrated into a DFL platform. Through a series of thorough experiments, this study contrasts the behavior of CFL and DFL when subjected to various poisoning attacks, identifying the critical factors that influence the spread and effectiveness of attacks within DFL. Furthermore, it benchmarks the efficacy of different defense mechanisms and explores the compatibility of CFL-based defense strategies with DFL.

The empirical findings offer valuable insights into the challenges of research and propose potential directions to enhance the robustness of DFL models for future research.

CRediT authorship contribution statement

Chao Feng: Writing – review & editing, Writing – original draft, Methodology, Conceptualization. **Alberto Huertas Celdrán:** Writing – review & editing, Writing – original draft, Methodology. **Jan von der Assen:** Writing – review & editing. **Enrique Tomás Martínez Beltrán:** Writing – review & editing. **Gérôme Bovet:** Project administration, Funding acquisition. **Burkhard Stiller:** Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work has been partially supported by *(a)* the Swiss Federal Office for Defense Procurement (armasuisse) with the CyberMind and DATRIS (CYD-C-2020003) projects and *(b)* the University of Zürich UZH.

References

[1] Duarte F. Number of IOT devices (2023–2030). Explod Top 2023. URL: https://explodingtopics.com/blog/number-of-iot-devices, Last Visit December 2023.
[2] Silva PR, Vinagre J, Gama J. Towards federated learning: An overview of methods and applications. WIREs Data Min Knowl Discov 2023.
[3] Beltran ETM, Perez MQ, Sanchez PMS, Bernal SL, Bovet G, Perez MG, Perez GM, Celdran AH. Decentralized federated learning: fundamentals, state-of-the-art, frameworks, trends, and challenges. IEEE Commun Surv Tutor 2023;25(4):2983–3013.
[4] Rodríguez-Barroso N, Jimenez-López D, Luzón MV, Herrera F, Martínez-Camara E. Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges. Inf Fusion 2023;90:148–73.
[5] Research G. Federated learning: Collaborative machine learning without centralized training data. 2017.
[6] Huertas Celdran A, Sanchez Sanchez PM, Feng C, Bovet G, Perez GM, Stiller B. Privacy-preserving and syscall-based intrusion detection system for IoT spectrum sensors affected by data falsification attacks. IEEE Internet Things J 2023;10(10):8408–15.
[7] Tian Z, Cui L, Liang J, Yu S. A comprehensive survey on poisoning attacks and countermeasures in machine learning. ACM Comput Surv 2023;55(8):1–35.
[8] Xia G, Chen J, Yu C, Ma J. Poisoning attacks in federated learning: A survey. IEEE Access 2023;11:10708–22.
[9] Lyu L, Yu H, Ma X, Chen C, Sun L, Zhao J, Yang Q, Yu PS. Privacy and robustness in federated learning: Attacks and defenses. IEEE Trans Neural Netw Learn Syst 2022;1–21.
[10] Benmalek M, Benrekia MA, Challal Y. Security of federated learning: Attacks, defensive mechanisms, and challenges. Revue des Sciences et Technologies de l'Information - Série RIA : Revue d'Intelligence Artificielle 2022;36(1):49–59.
[11] Blanco-Justicia A, Domingo-Ferrer J, Martínez S, Sánchez D, Flanagan A, Tan KE. Achieving security and privacy in federated learning systems: Survey, research challenges and future directions. Eng Appl Artif Intell 2021;106:104468.
[12] Chen Y, Gui Y, Lin H, Gan W, Wu Y. Federated learning attacks and defenses: A survey. In: 2022 IEEE international conference on big data (big data). 2022, p. 4256–65.
[13] Jere MS, Farnan T, Koushanfar F. A taxonomy of attacks on federated learning. IEEE Secur Priv 2021;19(2):20–8.
[14] Kumar KN, Mohan CK, Cenkeramaddi LR. The impact of adversarial attacks on federated learning: A survey. IEEE Trans Pattern Anal Mach Intell 2023;1–20.
[15] Liu P, Xu X, Wang W. Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. Cybersecurity 2022;5(1):1–19.
[16] Mothukuri V, Parizi RM, Pouriyeh S, Huang Y, Dehghantanha A, Srivastava G. A survey on security and privacy of federated learning. Future Gener Comput Syst 2021;115:619–40.
[17] Nair AK, Raj ED, Sahoo J. A robust analysis of adversarial attacks on federated learning environments. Comput Stand Interfaces 2023;86:103723.
[18] Qammar A, Ding J, Ning H. Federated learning attack surface: taxonomy, cyber defences, challenges, and future directions. Artif Intell Rev 2022;1–38.
[19] Wang Z, Kang Q, Zhang X, Hu Q. Defense strategies toward model poisoning attacks in federated learning: A survey. In: 2022 IEEE wireless communications and networking conference. 2022, p. 548–53.
[20] Zhang J, Li M, Zeng S, Xie B, Zhao D. A survey on security and privacy threats to federated learning. In: 2021 international conference on networking and network applications (naNA). 2021, p. 319–26.
[21] Bagdasaryan E, Veit A, Hua Y, Estrin D, Shmatikov V. How to backdoor federated learning. In: ArXiv:1807.00459 [cs]. 2019, p. 1–15.
[22] Yin J, Cui X, Li K. A reputation-based resilient and recoverable P2P botnet. In: 2017 IEEE second international conference on data science in cyberspace (DSC). 2017, p. 275–82.
[23] Yin D, Chen Y, Kannan R, Bartlett P. Byzantine-Robust distributed learning: Towards optimal statistical rates. In: Proceedings of the 35th international conference on machine learning. 2018, p. 5650–9.
[24] Pillutla K, Kakade SM, Harchaoui Z. Robust aggregation for federated learning. 2022, arXiv:1912.13445.
[25] Blanchard P, El Mhamdi EM, Guerraoui R, Stainer J. Machine learning with adversaries: Byzantine tolerant gradient descent. In: Proceedings of the 31st international conference on neural information processing systems. California, USA; 2017, p. 118–28.
[26] Mhamdi EME, Guerraoui R, Rouault S. The hidden vulnerability of distributed learning in byzantium. 2018, arXiv preprint arXiv:1802.07927.
[27] Xie C, Koyejo S, Gupta I. Zeno++: Robust fully asynchronous SGD. 2021, arXiv:1903.07020 [cs, stat].
[28] Muñoz-Gonzalez L, Co KT, Lupu EC. Byzantine-Robust federated machine learning through adaptive model averaging. 2019, arXiv:1909.05125.
[29] Li L, Xu W, Chen T, Giannakis GB, Ling Q. RSA: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. Proc AAAI Conf Artif Intell 2019;33(01):1544–51.
[30] Shejwalkar V, Houmansadr A. Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning. In: Proceedings 2021 network and distributed system security symposium. 2021.
[31] Ozdayi MS, Kantarcioglu M, Gel YR. Defending against backdoors in federated learning with robust learning rate. Proc AAAI Conf Artif Intell 2021;35(10):9268–76.
[32] Fang M, Cao X, Jia J, Gong NZ. Local model poisoning attacks to Byzantine-Robust federated learning. 2021, arXiv:1911.11815 [cs].
[33] Zhao Y, Chen J, Zhang J, Wu D, Teng J, Yu S. PDGAN: A novel poisoning defense method in federated learning using generative adversarial network. In: Algorithms and architectures for parallel processing. 2020.
[34] Fung C, Yoon CJM, Beschastnikh I. Mitigating sybils in federated learning poisoning. 2020.
[35] Zhang Z, Cao X, Jia J, Gong NZ. FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients. 2022, arXiv:2207.09209 [cs].
[36] Li S, Cheng Y, Wang W, Liu Y, Chen T. Learning to detect malicious clients for robust federated learning. 2020, arXiv:2002.00211.
[37] Feng C, Celdran AH, Vuong M, Bovet G, Stiller B. Voyager: MTD-based aggregation protocol for mitigating poisoning attacks on DFL. IEEE/IFIP Netw Oper Manag Symp 2024.
[38] Cao X, Fang M, Liu J, Gong NZ. FLTrust: Byzantine-robust federated learning via trust bootstrappin. In: Proceedings 2021 network and distributed system security symposium. 2021.
[39] Gholami A, Torkzaban N, Baras JS. Trusted decentralized federated learning. In: 2022 IEEE 19th annual consumer communications & networking conference (CCNC). Las Vegas, NV, USA: IEEE; 2022, p. 1–6.
[40] Zhao B, Sun P, Wang T, Jiang K. FedInv: Byzantine-robust federated learning by inversing local model updates. Proc AAAI Conf Artif Intell 2022;36:9171–9, Number: 8.
[41] Rieger P, Nguyen T, Miettinen M, Sadeghi A. DeepSight: Mitigating backdoor attacks in federated learning through deep model inspection. In: Proceedings network and distributed system security symposium. 2022.

[42] Nguyen T, Rieger P, Chen H, Yalame H, Möllering H, Fereidooni H, Marchal S, Miettinen M, Mirhoseini A, Zeitouni S, Koushanfar F, Sadeghi A, Schneider T. FLAME: Taming backdoors in federated learning. 2021.
[43] Feng C, Celdran AH, Baltensperger J, Beltran ETM, Bovet G, Stiller B. Sentinel: An aggregation function to secure decentralized federated learning. 2023, arXiv:2310.08097.
[44] Wu C, Yang X, Zhu S, Mitra P. Mitigating backdoor attacks in federated learning. 2021, arXiv:2011.01767.
[45] Sun Z, Kairouz P, Suresh AT, McMahan HB. Can you really backdoor federated learning?. 2019, arXiv:1911.07963.
[46] Nguyen TD, Nguyen T, Nguyen PL, Pham HH, Doan K, Wong KS. Backdoor attacks and defenses in federated learning: Survey, challenges and future research directions. 2023, arXiv:2303.02213.
[47] Guo S, Zhang T, Yu H, Xie X, Ma L, Xiang T, Liu Y. Byzantine-resilient decentralized stochastic gradient descent. 2021, arXiv:2002.08569.
[48] Zhang Y, Zeng D, Luo J, Xu Z, King I. A survey of trustworthy federated learning with perspectives on security, robustness, and privacy. 2023, arXiv:2302.10637 [cs].
[49] Cao D, Chang S, Lin Z, Liu G, Sun D. Understanding distributed poisoning attack in federated learning. In: IEEE 25th international conference on parallel and distributed systems. 2019.
[50] Cai G, Wang B, Hu W, Wang T. Moving target defense: State of the art and characteristics. Front Inf Technol Electron Eng 2016;17(11):1122–53.
[51] Beltran ETM, Gómez aLP, Feng C, Sanchez PMS, Bernal SL, Bovet G, Perez MG, Perez GM, Celdran AH. Fedstellar: A platform for decentralized federated learning. Expert Syst Appl 2024;242:122861.
[52] Flask. Flask. 2024, URL: https://flask.palletsprojects.com/en/3.0.x/. Last Visit July 2024.
[53] Merkel D. Docker: Lightweight linux containers for consistent development and deployment. 2014, URL:https://www.docker.com/. [Accessed 9 july 2024].
[54] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Köpf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S. Pytorch: An imperative style, high-performance deep learning library. 2019, arXiv:1912.01703.
[55] Falcon W, team TPL. Pytorch lightning. 2019, URL: https://www.pytorchlightning.ai, Last Visit July 2024.
[56] Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, Kudlur M, Levenberg J, Monga R, Moore S, Murray DG, Steiner B, Tucker P, Vasudevan V, Warden P, Wicke M, Yu Y, Zheng X. TensorFlow: A system for large-scale machine learning. 2016, URL: https://www.tensorflow.org/tensorboard, arXiv:1605.08695. Last Visit July 2024.
[57] LeCun Y, Cortes C. MNIST handwritten digit database. 2010.
[58] Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. 2017, arXiv:1708.07747 [cs, stat].
[59] Krizhevsky A. Learning multiple layers of features from tiny images. 2009.
[60] Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. 2017, arXiv preprint arXiv:1704.04861.
[61] Watts DJ, Strogatz SH. Collective dynamics of 'small-world'networks. nature 1998;393(6684):440–2.
[62] Nguyen TD, Rieger P, Miettinen M, Sadeghi AR. Poisoning attacks on federated learning-based IoT intrusion detection system. In: Proc. workshop decentralized iot syst. secur. 79, 2020.
[63] Feng C, Celdran AH, Sanchez PMS, Kreischer J, von der Assen J, Bovet G, Perez GM, Stiller B. CyberForce: A federated reinforcement learning framework for malware mitigation. 2023, arXiv:2308.05978.
[64] Yoo JH, Jeong H, Lee J, Chung TM. Federated learning: Issues in medical application. In: Future data and security engineering: 8th international conference, FDSE 2021, virtual event, November 24–26, 2021, proceedings 8. Springer; 2021, p. 3–22.
[65] Ali M, Naeem F, Tariq M, Kaddoum G. Federated learning for privacy preservation in smart healthcare systems: A comprehensive survey. IEEE J Biomed Health Inform 2022;27(2):778–89.
[66] Kuo T-T, Pham A. Detecting model misconducts in decentralized healthcare federated learning. Int J Med Inform 2022;158:104658.
[67] Xie C, Huang K, Chen PY, Li B. Dba: Distributed backdoor attacks against federated learning. In: International conference on learning representations. 2019.
[68] Frid-Adar M, Klang E, Amitai M, Goldberger J, Greenspan H. Synthetic data augmentation using GAN for improved liver lesion classification. In: 2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018). 2018, p. 289–93. http://dx.doi.org/10.1109/ISBI.2018.8363576.
[69] Sánchez Sánchez PM, Huertas Celdrán A, Xie N, Bovet G, Martínez Pérez G, Stiller B. Federatedtrust: A solution for trustworthy federated learning. Future Gener Comput Syst 2024;152:83–98. http://dx.doi.org/10.1016/j.future.2023.10.013, URL: https://www.sciencedirect.com/science/article/pii/S0167739X23003886.
[70] Celdran AH, Feng C, Sanchez PMS, Zumtaugwald L, Bovet G, Stiller B. Assessing the sustainability and trustworthiness of federated learning models. 2023, arXiv:2310.20435.
[71] Parliament E, of the European Union C. Regulation (EU) 2016/679 of the European parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/EC (general data protection regulation). 2016, URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj, Last Visit July 2024.
[72] US Department of Health and Human Services. Health insurance portability and accountability act of 1996 (HIPAA). 1996, URL: https://www.hhs.gov/hipaa/for-professionals/privacy/index.html, Last Visit July 2024.
[73] Bourtoule L, Chandrasekaran V, Choquette-Choo CA, Jia H, Travers A, Zhang B, Lie D, Papernot N. Machine unlearning. In: 2021 IEEE symposium on security and privacy. 2021, p. 141–59. http://dx.doi.org/10.1109/SP40001.2021.00019.
[74] Xu C, Zhao W, Zhao J, Guan Z, Song X, Li J. Uncertainty-aware multi-view deep learning for internet of things applications. IEEE Trans Ind Inf 2023;19(2):1456–66. http://dx.doi.org/10.1109/TII.2022.3206343.
[75] Liu Y, Xu C, Chen L, Yan M, Zhao W, Guan Z. TABLE: Time-aware balanced multi-view learning for stock ranking. Knowl-Based Syst 2024;112424. http://dx.doi.org/10.1016/j.knosys.2024.112424, URL: https://www.sciencedirect.com/science/article/pii/S095070512401058X.