Securing the FL Landscape: A Comparative Analysis of Attacks in Centralized and Decentralized Models

Alberte Krogh Hansen, Alexis Donato Calin, and Kimika Uehara

December 13, 2024

1 Introduction

Modern IT infrastructures face increasing complexities and vulnerabilities due to the interconnection of networks, protocols, and devices. Federated Learning (FL) was first introduced by Google in 2016 as a way to train a centralized machine learning model while data is distributed among many devices. Prior to FL, most collaborative machine learning solutions at the time featured a centralized analysis, requiring that all the data must be gathered in a single central location. Thus, high latency and bandwidth, as well as privacy risks for sensitive information, were prominent design issues. The original paper published by Google sought to fix the problem of data utilization and privacy, and suggested the following method: send local models to each client, calculate the global minimum, and then send the resulting weights to the central server, repeating the cycle and averaging the best parameters to calculate a global model. Then, only two exchanges of small files, models and weight, were necessary to generate models across distributed devices. Initially designed for reducing the communication overhead of data sharing, FL has since found applications across domains requiring privacy-preserving collaboration [11]. Despite its benefits, due to its centralized orchestration, FL is known to be susceptible to several vulnerabilities, including single points of failure, risks to the central server's integrity, and privacy concerns. To address these challenges, Decentralized Federated Learning (DFL) has recently gained attention as an alternative that removes the central server as the sole aggregator.

This paper will assess the current FL landscape by providing a comprehensive comparison of traditional FL and DFL systems, with a particular focus on the operational differences and security implications of each approach. In Section 2, we will explore different configurations of the DFL setup and assess the security risks of components involved in the federation. In Section 3, we will identify and compare common and distinct attacks for FL and DFL systems, along with possible countermeasures. The paper will conclude by offering insights into the trade-offs between centralized and decentralized systems and discussing the current challenges faced and future research directions.

1.1 Federated Learning

FL is a machine learning technique addressing the challenges of data silos when constructing a joint model in a secure and distributed manner. FL is particularly suitable for various practical applications with stringent privacy requirements, as training data remains localized on individual servers.

Ahead of FL, is the progressive emergence of AI technology, which contributes to academic research by developing models and training through data. The problem arises when risks, including data security and data silos, become limitations for AI. Due to the urgent need for a practical and efficient technique to remedy the problems, the concept of FL arose, which makes it possible to update models locally without revealing private personal data [19].

FL operates collaboratively by executing FL algorithms across multiple distributed devices or servers under the assumption that private information remains confined to the local device. Model training occurs locally for each client, and communication between clients and the server involves parameter exchanges rather than direct data interaction. The server's role is limited to aggregating parameters to update the global model. This global model is refined through communication between clients and the server. The process involves each client training a common initial model from the central server on local data until the model converges locally. The model parameters are then encrypted and uploaded to a server, which aggregates them using the FL algorithm to update the global model for the next training round. The ultimate goal is for the global federated model to converge through continuous interaction between clients and the server.

An important challenge in FL is the communication overhead, which increases with the number of clients and represents one of the most significant barriers in FL. This overhead occurs because edge devices frequently transmit their model parameters to the central server, leading to high communication costs. To address this, improving communication efficiency has become a primary objective in FL algorithm development. Methods such as optimization algorithms, including gradient descent (GD), are designed to reduce communication costs by transmitting trained models efficiently to the central server for aggregation. The most widely applied aggregation approach in FL is Federated Averaging (FedAvg). This method involves the central server averaging the weights or gradients received from all clients. While this reduces communication overhead and is computationally efficient, it relies heavily on mutual trust between the central server and the clients.

Another problem with the clients is the varying data distribution, as any locally available data point distribution does not represent the overall data distribution. The heterogeneity in the distribution of client data leads to global model drift [3]. As solutions for the static heterogeneity and for personalized models of high quality for the client, personalized FL methods are presented. This is, for example, a multi-task FL algorithm that introduces a non-federated batch normalization layer into federated deep neural networks, allowing users to personalize the training model according to their own local data. This facilitates the performance and convergence speed of the local client model.

1.2 Decentralized Federated Learning

Decentralized Federated Learning (DFL) differs from FL by storing data locally and sending models directly between nodes, rather than relying on a central server. Eliminating the central server removes a Single Point of Failure (SPoF), reduces bottlenecks, and minimizes trust dependencies. In DFL, each node performs local model training, parameter exchange, and local model aggregation.

DFL can have two type of Federation Type [14]: cross-silo and cross-devices. Cross-silo is typically used when there are less than a hundred nodes. Each node handles a lot of data. On the other hand, cross-devices is typically used when there are more than a hundred nodes. Each node handles a small amount of data and is power restrained.

As mentioned, FL's architecture is not without flaws. One critical issue is that it is not ideal for FL to have a SPoF in the system, as a compromised server could affect the entire network [6]. This security challenge is addressed by decentralizing FL's architecture. DFL improves both the reliability and scalability of FL, as its architecture enhances communication efficiency and eliminates SPoF. Although the preparation processes in FL and DFL are similar, the exchange of model parameters and aggregation occur via peer-to-peer communication or blockchain technology. By integrating blockchain, certain advantages such as traceability and immutability emerge, but it can also leave the system vulnerable to blockchain-related security adversaries. It is important to note that DFL can exist independently of blockchain, even though DFL is a paradigm that often includes both FL and blockchain technologies [6].

2 Components of Decentralized Federated Learning

DFL's decentralized nature introduces unique system setups, altering the attack surface. This section will dissect DFL in three distinct components: network topology, decentralization and aggregation schema, and communication protocols. We will introduce current main approaches in literature for each, and analyze their security implications.

2.1 Components and Structure

2.1.1 Network Topology

The standard FL framework takes on a starred network topology, in which a central server coordinates the participating clients [16]. On the other hand, due to the absence of a central server, DFL networks can exhibit a diverse range of configurations. The prominent controlled network topologies can be seen in Figure 1 [20]:

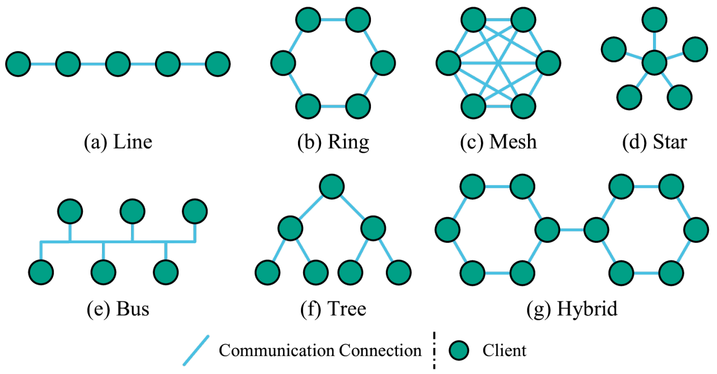

Figure 1: Illustration of network topology in decentralized federated learning by Yuan et al., 2024 [20].

- Line: A sequential pointing line is the simplest and often used as a baseline for comparison. It is unable to accommodate continuous learning of new knowledge and lacks cyclic connections, limiting the system from fully converging.
- Ring: The cyclic form allows the model to be trained between clients and acquire new knowledge from other clients. On top of its simplicity, the ring is favored for its ability to iterate indefinitely until convergence. Each client is required to transmit multiple model parameters in each communication round, introducing a possible burden on the network bandwidth.
- Mesh: A mesh is a multidirectional ring, in which each client transmits local model parameters to all other clients in every communication round. The mesh topology is hence characterized by the higher communication frequency and larger per-communication data packet overhead.
- Star: There are two different modes of the star variant. In the first mode, the central client receives, aggregates, and distributes the local models, while also generating original data and utilizing the models for perception and decision-making, unlike the Central FL model. The second mode focuses on geographic interoperability among clients, in which a client serves as the geographical center for dispersed clients in order to conserve communication resources.
- Hybrid: The hybrid topology combines elements from other variants in order to maximize adaptability. An example is two rings connected through a central client, which provides global connectivity while allowing each ring to be treated as a single entity.

In addition, researchers have suggested more complex networks to achieve complete decentralization, in which nodes spontaneously organize themselves. The following models imagine a scenario where nodes are free to cooperate with each other without control from an operator [16]:

- Erdos-Renyi (ER) model defines how to generate random graphs with a homogeneous structure, in which there is a fixed probability of an edge existing between two nodes. This framework can model how information propagates in random and well-mixed networks without a pronounced structure.
- Barabasi-Albert (BA) algorithm uses a preferential attachment mechanism to generate random scale-free networks. In this model, the probability of nodes connecting is proportional to the nodes' degree, such that the degree distribution follows a power law.
- Stochastic Block Model (SBM) generates a community-like network where nodes belong to groups based on their connectivity patterns. This model is suitable for exploring how information spreads within and between communities.

2.1.2 Decentralization and Aggregation Schema

In a fully decentralized schema, the aggregation process is distributed among the nodes without involving a central server. Model updates are aggregated through direct exchanges among clients [4]. For instance, the decentralized parallel stochastic gradient descent technique allows the model to be trained in parallel in each local node, and aggregation occurs by transmitting the trained model parameters to neighboring nodes [12]. Lalitha et al., 2018 [10] suggests a Bayesian-like approach, maintaining a belief over the model parameter space and allowing clients to update their belief from their one-hop neighbors to learn a model.

In partial aggregation, intermediate nodes perform some aggregation before updates are shared across the network, instead of each node sharing full updates between neighbors. One way to achieve this is by a dynamic assignment of aggregator roles among nodes based on communication infrastructure and data distribution [1]. Another example is a tiered architecture with clusters of edge devices, in which aggregation tasks are delegated to selected cluster heads to reduce congestion and maintain training quality [7]. Partial aggregation may be suitable in large networks when communication overhead and bandwidth are a concern. For instance, Federated Partial Gradient Aggregation splits gradients into slices, each of which is aggregated locally and with peer devices. Afterwards pulling and merging mechanisms are used to rebuild mixed updates for aggregation [8].

In blockchain-based Aggregation, a blockchain is used to verify and aggregate model updates in a decentralized and secure manner. In blockchain-based FL, a node in the blockchain represents a participant, and the blockchain stores the global model [22]. Like FL, data in a blockchain network is decentralized, as each user transfers data using blocks. Each block is uniquely identified by its hash, as well as a reference to the hash of the previous block in the chain. This complexity reduces the risk of data tampering since any alteration creates inconsistency in the chain. However, hashes also ensure data traceability in the chain. DFL utilizes blockchain to facilitate easier communication between nodes by treating model updates as data within a block, subsequently sharing the block according to a consensus mechanism. The process resembles the local model construction in FL, but the prepared updates take the form of a data block [6].

Blockchain achieves global model aggregation with a miner code or a smart contract [22]. In a typical workflow, as shown in Figure 2, the task publisher invokes a smart contract to release the FL task, and then selects clients who are willing to participate. Each participant then downloads the global model from the latest generated block, and performs local model training independently. Participant clients upload local model parameters to their corresponding blockchain nodes, which verify the parameters to prevent unreliable model updates. Once the blockchain nodes acquire the bookkeeping rights of the current training round, they generate candidate blocks to store the verified local model parameters, which are broadcast to other nodes in the blockchain for further verification. Finally, the verified nodes are added to their local ledgers and the model aggregation smart contract is invoked by the task publisher to store the resulting global model in the block [21].

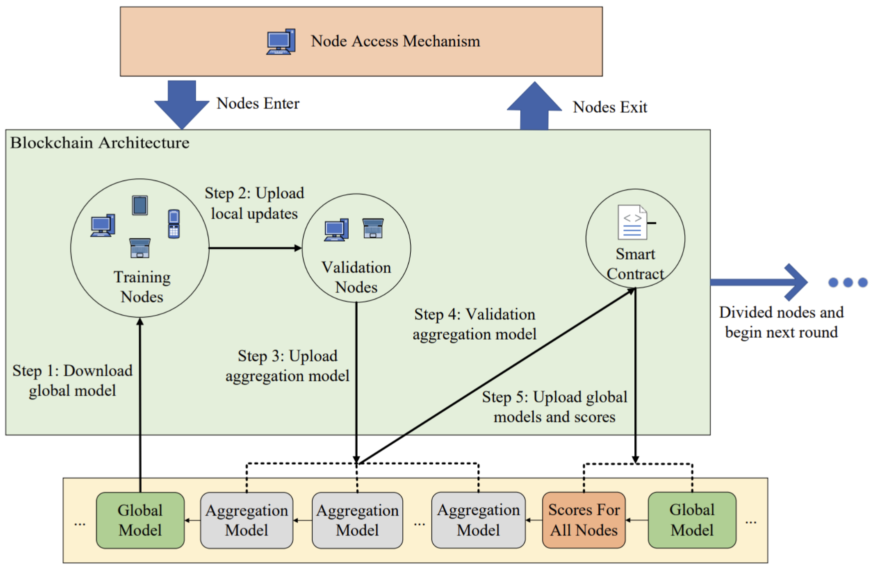

Figure 2: The workflow of blockchain-based decentralized federated learning, by Zhao et al., 2021 [22]

2.1.3 Communication Protocols

Parameter sharing consists of synchronous, asynchronous, and semi-synchronous communication schemes. In synchronous communication, participating nodes send their local model parameters to all their neighboring nodes after local training, and wait for the end of the synchronized round to transmit again. In asynchronous communication, clients update and exchange models at their own pace, which create greater flexibility but also increased communication costs and lower generalization. Finally, in semi-synchronous communication schemes, nodes perform varying number of epochs according to processing capacities and data volumes [14].

Further, common communication protocols in DFL are pointing, gossip, and broadcast. Pointing is a unidirectional, one-to-one, specified communication between two peers. Gossip protocol is a random one-peer-to-one-peer stochastic communication method. Broadcast protocol is one-peer-to-all-peers, whereby the client communicates its model to all clients [20]. There are also hybrid protocols which combine the above to accommodate for different scenarios and constraints. For instance, a broadcasting-based gossiping algorithm is a one-peer-to-neighbor-peers approach, by which a node broadcasts its value to its neighbors before gossiping, motivated by applications to wireless sensor networks [2].

2.2 Potential Security Implications

2.2.1 Network Topology

In general, fully connected networks offer improved resilience against attacks compared to other topologies which involve a central client [14]. The line and star topologies are more vulnerable to SPoF than the mesh, ring and hybrid variants. In the line variant, as the cyclic connections are limited, attackers can more easily predict and exploit client contributions. By having the central client, the star topology introduces a single point of compromise and potentially risks exposing sensitive data during central processing or transmission. In the ring topology, while the cyclic nature reduces reliance on a single node, iterative sharing of models between clients could potentially expose sensitive updates if intercepted or manipulated. In mesh and hybrid models, increased connections create higher communication overhead and may amplify the risk of intercepting or tampering with model updates. However, diversified pathways enhance security and privacy, as communication and model sharing are decentralized across all clients. Hybrid variants can also utilize dynamic topology adjustments to ensure robust communication against interference or blockages [20].

In complex networks, generally the node degree and centrality are primary determinants of security and efficiency of DFL. High-centrality nodes play a crucial role in maintaining connectivity and data flow, but when removed or disrupted, cause low-degree nodes to become isolated. Hence, poorly connected nodes are vulnerable in times of disruptions, and high-centrality nodes pose risks of targeted attacks by malicious actors [15]. In particular, BA networks feature high-degree hubs for enhancing connectivity, which create potential SPoFs and make them susceptible to target attacks. ER networks tend to be more uniform, making them less vulnerable to target attacks at the cost of slower information dissemination. Higher clustering coefficients facilitate local data retention and faster local processing, reducing exposure to external threats, but can cause bottlenecks in collaborative learning due to limited knowledge diffusion. In SBM networks, privacy is enhanced within distinct communities as external data access is limited, which in turn restricts learning efficiency and knowledge sharing. In summary, there exists a trade-off between data privacy and global learning efficiency, as high clustering and local connectivity limit external data exposure but reduce utility. There is also a dilemma between security and synchronization, as networks with low connectivity are slower to propagate data but less susceptible to targeted attacks on central nodes [16].

2.2.2 Decentralization and Aggregation Schema

Fully-decentralized aggregation eliminates the central server used for aggregation, thus getting rid of a SPoF, and increases fault tolerance as the learning system can continue functioning in cases where some nodes fail. However, without central oversight and verification mechanisms, it is challenging to determine the authenticity of updates. Direct exchange of parameters between clients can increase the risk of data leakage, if updates are intercepted or analyzed [4]. As raw data is kept local and not exchanged with other nodes, malicious actors may exploit model parameters to infer knowledge about the datasets or the model [1].

In partial aggregation mechanisms, security and privacy are enhanced as intermediate nodes act as filters for malicious updates, and individual client updates are not directly exposed to the entire network. However, if an intermediate node is compromised, the system is at risk of malicious updates being injected or amplified which affect the downstream aggregation. The trusted nodes may introduce a dependency and become target for adversaries [1].

Blockchain has several positive security implications, such as ensuring that all updates are tamper-proof and traceable, and the consensus mechanisms and the immutable ledger keeping a record of validated updates. However, incorporating blockchain as an underlying infrastructure into DFL introduces the risk of blockchain-based attacks, which exploit underlying mechanisms such as the consensus, private and public keys, smart contracts, and the network layer. Poor implementation or lack of security protection of these components and infrastructure allow attackers to manipulate or gain unauthorized access to compromised blocks' contents and associated assets [6]. Attacks on blockchain technology are detailed in section 3.3.3.

2.2.3 Communication Protocols

Synchronous communication offers higher reliability due to structured synchronization. However, this scheme can be vulnerable to delays caused by slow or idle participants, as well as targeted attacks on synchronization points. In asynchronous communication, as nodes communicate independently without waiting for others, the system can have faster convergence but suffer higher communication costs and risks of stale data. Since there is no global synchronization to validate updates, and updates may be delayed or duplicated, asynchronous communication can introduce vulnerabilities to poisoning attacks and data inconsistency [14]. In addition, asynchronous DFL makes ensuring secure aggregation challenging, as secure aggregation protocols require all clients to be incorporated in the aggregation step [6]. Finally, semi-synchronous communication achieves a balance between speed and resource efficiency by combining elements of both synchronous and asynchronous methods, but can be vulnerable to adversarial attacks during the aggregation phase [14].

Pointing communication protocol reduces the attack surface by limiting the exposure of model updates to specified peers, and offers higher traceability of community pathways in case of malicious activity. On the other hand, the pointing protocol makes the system more susceptible to eavesdropping or interception, if communication is not properly encrypted and protected, hence suffers scalability issues as managing secure keys and communication channels can be complex in large networks. In the gossip protocol, randomized communication makes it harder for attackers to target specific nodes, increasing robustness, as well as enhances fault tolerance as the network can still function even if some peers are compromised. However, with random communication paths, there is less traceability of malicious sources, and due to the increased exposure of model updates with multiple random peers, a compromised node can spread malicious updates rapidly across the network. The broadcast protocol ensures high transparency as all peers receive the same updates, making it easier to detect anomalies in the network, as well as high robustness due to the fact that updates are still propagated in case of compromised peers. In resource-constrained environments, high communication overhead means that the network is more prone to denial-of-service attacks. Furthermore, if the broadcasting peer is malicious, its updates can influence the entire network, and thus the protocol creates a larger attack surface for adversarial attacks [20].

3 Attacks and Defenses

Following this introduction and analysis of FL and DFL systems, along with the potential security implications for components involved in DFL, this section will dive into each approach and conduct a comparative analysis of the various attacks and corresponding defense mechanisms.

3.1 Attacks on Federated Learning

One must also be aware of the security risks that FL entails, as security can guarantee the confidentiality and integrity of data. There are different types of attacks that can compromise the security of the federated model and the local participants' model.

3.1.1 Poisoning Attacks

The first attack is the poisoning attack, which tends to manipulate, destroy and pollute the client's local training dataset or model. It can therefore be subdivided into categories of data poisoning, which manipulates clients' data, and model poisoning, which affects the client and the global model, producing a specific error output [19]. A poisoning attack mainly involves an attack on a single user or server.

A data poisoning attack can occur in various ways. One method involves manipulating the source tags or characteristics of local client data with the aim of influencing the training outcomes of connected models. The challenge with data poisoning is that it is often harder to detect than model poisoning attacks because it can be difficult to determine whether a client participates in FL with good intentions or malicious intent [3]. Furthermore, the attacker in data poisoning attacks only interferes with the data collection process of the FL systems client and does not directly harm the FL-trained model, making detection even more complex.

Data poisoning attacks can generally be classified into two main categories:

- Clean-label attacks: These assume that the client adversary cannot alter the labels of any training data sample [19]. This constraint exists because data must be verified as belonging to the correct class through a validation process, meaning that the poisoning must be subtle.
- Dirty-label attacks: The adversary inserts multiple copies of the data samples they wish to misclassify, using a target label of their choice in the training set [19].

In contrast, model poisoning attacks target local model updates before they are sent to the central server. The design of FL itself introduces vulnerabilities to model poisoning, as the global model is exposed to clients during all training stages and can be intercepted during communication [19]. In general, model poisoning attacks are more effective than data poisoning in FL settings because a single non-collaborative, malicious participant can cause the global model to misclassify a selected set of inputs with high confidence.

3.1.2 Byzantine Attacks

The second security attack that acts as a critical threat in FL is the Byzantine attack, which focuses on the collaboration between multiple users in a distributed learning environment. Unlike poisoning attacks, which typically target a single user or server, Byzantine attackers control several clients, also known as Byzantine users, who upload corrupted or fake data. Byzantine users may upload bottled data due to unreliable communication channels. To detect this type of attack, reliability indicators are used to evaluate whether knowledge has been transmitted by clients [3]. The result is often a manipulated global model that fails to converge or produces wrong outputs, impacting the integrity of the system.

3.1.3 Server Exploitation

Due to the centralized structure of FL, it has the inherent weakness of making the central server more vulnerable to exploitation attacks [13]. Malicious or compromised servers can significantly affect the global model by corrupting it. The model can be easily manipulated, or the data from the private client can be extracted. This allows adversaries to exploit the shared computational resources for creating malicious objectives during the training of a machine learning model.

3.2 Defenses for Federated Learning

Defense mechanisms are critical for protecting FL systems from a variety of attacks and reducing associated risks.

3.2.1 Mitigating Poisoning Attacks

Since direct access to data is restricted, most defenses against model corruption focus on ensuring that the trained model learns to recognize the underlying statistical distribution of the actual training data. While it cannot be guaranteed that the trained model is entirely free from malicious data or updates, defenses aim to minimize the influence of such updates on the global model. Defending against data and model poisoning in FL involves identifying and removing outliers by calculating the similarity between data samples, often through anomaly detection. This approach is considered a proactive defense mechanism that explicitly detects malicious updates and prevents their impact on the system [3]. Such defenses are particularly effective against untargeted adversarial attacks.

3.2.2 Mitigating Byzantine Attacks

Researches have developed various strategies to detect and mitigate Byzantine attacks in FL. One approach is a credibility-based one that evaluates the reliability of knowledge transmitted by clients. The approach integrates secure two-party computation protocols to ensure privacy during parameter aggregation, effectively identifying Byzantine attackers and safeguarding the global model [19]. Another strategy is a blockchain based FL framework that uses validators to execute parallel validation workflow and detect Byzantine attacks through consistency checks. The approach both improves model verification efficiency through Byzantine-resistant validation and ensures transparency via blockchain technology.

3.2.3 Mitigating Server Exploitation

Server exploitation poses a significant challenge in FL, as attackers can gain direct access to the global model from the server, thereby expanding the attack surface[3]. Additionally, the server influences clients´ views of the shared model, meaning it plays a central role in determining the outcome of the training process. Servers also control when clients can access or modify the model during FL training, enabling opportunities for designing new defense schemes. These schemes could measure a model's average case or worst case vulnerability to attacks, offering a way to mitigate server exploitation risks effectively.

3.3 Attacks on Decentralized Federated Learning

DFL's architecture is not without its flaws. The primary threats to DFL involve privacy breaches and the performance of the global model. Unlike centralized FL, where the main threat lies with the server (the core of aggregation, communication, and validation processes), DFL faces threats targeting clients or miners. Various operational models and methods can compromise the system's performance or privacy.

3.3.1 Privacy Breach Attacks

Privacy breach attacks in FL and DFL involve exposing sensitive data through model updates or gradients. Although data is decentralized in DFL, nodes may still access parameter information and gradients.

Model Inversion exploits shared model parameters to reconstruct private data. FL is vulnerable to this kind of attack due to the aggregation done on the central server, where an attacker can access global model updates. In DFL, attackers communicate with each node by exploiting peer-to-peer communications in order to get the model parameters. Gradient leakage serves as a way to reconstruct the local data from a DFL node by analyzing shared gradient information [20].

Privacy can also be compromised by Membership Interference Attacks, which aims to reveal personal information associated with certain samples. By using membership interference, one can determine if a sample is a member of a specific class or learn more about its characteristics [20].

3.3.2 Poisoning Attacks

As noted earlier, FL is vulnerable to Data Poisoning, an attack which involves the use of poisoned local training data, such as the addition of noise or altering data labels to mislead the global model during aggregation. However in DFL, where there is no centralized server to verify participants, the threats intensify. Attackers can exploit the decentralized nature by impersonating legitimate participants and introducing artificially generated or maliciously manipulated data. This lack of centralized oversight in DFL makes it harder to detect and isolate poisoned datasets, thereby amplifying the attack's potential impact. [20].

Model poisoning in FL often involves directly modifying a local client's model parameters in an attempt of degrading the global model's performance or embedding hidden backdoor. Due to the lack of a central authority in DFL to manage model updates, this threat becomes more dangerous. Malicious participants can manipulate their local models to propagate poisoned parameters more freely across the network. This decentralized environment provides opportunities for attackers to modify the purpose of the model or insert subtle changes that remain undetected, ultimately compromising the integrity of the distributed system. Model Poisoning is less covered by research, but this attack is even more powerful than Data Poisoning attacks, indeed Model Poisoning can take advantage of the fact that malicious participants can directly influence the performance of federated global model [20].

3.3.3 Blockchain-based Attacks

Blockchain-based attacks exploit the critical infrastructure of blockchain, which is used to achieve decentralized in a FL system. Consensus attacks are a way to gain majority control of the DFL system's consensus mechanisms. This is typically done through Sybil attacks by creating fake nodes to increase influence or through 51% attacks which is when an attacker manages to take control of 51 percent of the nodes, giving him control over the whole consensus mechanism. Further, while cryptography secures private and public keys, flaws in key-signing mechanisms could allow a private key to be hijacked via its corresponding public key. This would enable a potential hacker to gain full access to the corresponding data in the blockchain [6].

As blockchain has no control over the network layer and relies on the security of the internet service provider. Routing attacks exploits network protocols vulnerabilities. If an attacker gains access to the network it can tamper the data with packet routing and disrupt data transfer or alter blockchain structures. Privacy Poisoning is an attack in which hackers exploit the legal constraints of the blockchain by inserting personal data into a data block, making the system non-compliant according to the European general data protection regulation. Distributed Denial of Service (DDoS) disrupts communication between nodes in the DFL system. DDoS agents can continuously generate false transactions, filling the blockchain buffer with so-called spam data. This results in degraded blockchain functionality [6].

3.4 Defenses for Decentralized Federated Learning

Compared to FL, DFL removes the SPoF, and doesn't rely on direct access to raw data, but the nature of collaborative model training also introduces vulnerabilities. The main vulnerabilities of DFL are privacy leakage, model poisoning and blockchain attacks. We will discuss what strategies and technologies can be used to mitigate theses vulnerabilities.

3.4.1 Data Privacy Preservation

DFL can function by sharing model parameters, including gradients, weights, and metal-level information. However, gradients and meta-level information can be used to find details about the training data and model updates. In order to prevent this issue and improve privacy, DFL should only share model weights [14].

Research indicates that privacy leakage is one of overarching challenges that FL faces [20], as FL entails certain risks of parameter data being exposed. In addition, there are various technologies for protecting privacy. These are all security encryption technologies that are intended to protect information:

- Secure Multi-Party Computing (SMC): This technology is used to protect sensitive input data from each party to the process by encrypting parameters. The security model, which usually involves several parties, and ensures security through a zero knowledge simulation framework, allows participants to retain control over their data while only knowing the input and output of operations.

- Differential Privacy (DP): Is an approach that introduces random noise to drown the original encrypted user data, making it impossible for attackers to reverse the original data from the database.
- Homomorphic Encryption (HE): Is an encryption algorithm that allows users to perform specific algebraic operations on ciphertext directly. When operating on an encrypted model using HE technique, all participants in these operations must share both the same encryption key as well as decryption key.

3.4.2 Mitigating Poisoning Attacks

Similar to its role in preserving data privacy, Differential Privacy (DP) can also be employed to mitigate poisoning attacks. Additionally, it can be complemented with other technologies such as [20]:

- Anomaly Detection: This approach aims to analyze the contribution of nodes to detect anomalies or outliers. These anomalies could indicate that someone is attacking the DFL through data poisoning or model poisoning attacks.
- Pruning: In order to eliminate backdoor, pruning drops some neurons of the network. This way the structure is more accurate and efficient. This is not without consequences because it can also lead to a loss of model capacity.
- Federated distillation: Federated distillation (previously known as knowledge distillation) is a technique used to optimize the models by using a neural network as a reference for training, instead of learning directly from data. It helps mitigate poisoning attacks by reducing the reliance on raw, and potentially poisoned data.

Sentinel [5] is an aggregation function that can be used to secure DFL against poisoning attack. It relies on a combination of similarity filtering, bootstrap validation, and normalization to ensure that only reliable updates contribute to the shared model and to maintain the integrity of the learning process.

3.4.3 Mitigating Blockchain Attacks

In combination with previously mentioned defense mechanisms, blockchain-based protection could help mitigate attacks in DFL systems. The following are potential defenses for the underlying blockchain architecture and infrastructure: robust aggregation, Trusted Execution Environment (TEE), and Verifiable DFL.

Sophisticated aggregation and consensus mechanisms, in combination with same techniques used in FL, can help protect DFL systems. Protocols such as Proof-of-Work (PoW) and Proof-of-Stake (PoS) control the participation of nodes in both generating updates and aggregating parameters, thus are known to drastically increase the difficulty of consensus attacks for attackers, as well as poisoning attacks in blockchain [6].

TEE offers a secure, tamper-resistant environment for executing code and managing sensitive data. TEE's capabilities include hiding model parameters on local devices and sophisticated cryptographic safeguards. Though it is an under-explored study area, integrating TEE as a method of secure smart contract data in blockchain-based systems is a potential way to ensure that data and model parameters remain confidential, interactions between nodes are secure, and data access privileges are protected [3].

In verifiable DFL systems, only trustworthy participants are allowed to engage in the training process. Randomly selected workers, based on consensus protocols, verify the model updates generated by trainers before aggregation, and participants are rewarded for their reliability [9]. Further, mechanisms like verifiable random functions or reliability rankings could be used to determine trustworthy trainers. Additionally, systems like VFChain employ trainer signatures and randomly rotate committees to verify aggregated models, maintaining a transparent, tamper-resistant ledger of training records [17].

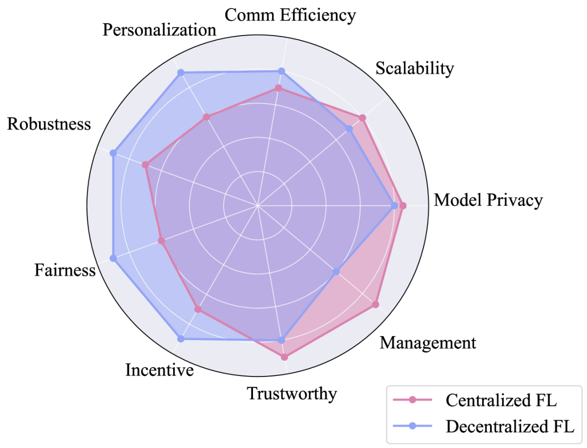

Figure 3: Comparative analysis between CFL and DFL. Taken from Yuan et al., 2024 [20].

3.5 Discussion

Yuan et al., 2024 [20] argue that both FL and DFL exhibit distinct strengths and weaknesses, as illustrated in Figure 3. The radar chart highlights how these two approaches perform across metrics such as communication, scalability, robustness, personalization, fairness and model privacy.

FL's centralized structure leads to greater management, with the central server providing secure aggregation and verification structures. By exchanging only model parameters instead of raw data, FL reduces bandwidth requirements while at the same time protecting user privacy. This structure is particularly effective in applications where trust in the central server can be assumed as it ensures data remains localized [20].

However, the centralized nature of FL creates a SPoF, making the system highly dependent on the security and reliability of the server. This vulnerability is reflected in FL´s lower robustness compared to DFL. Additionally, FL faces challenges in fairness and scalability. The global model aggregated by the server might not perform equally well across all clients, especially when client data is highly heterogeneous [20]. Similarly, the server's communication resources can become a bottleneck in large-scale deployments with many clients. Despite these challenges, FL excels in environments where management simplicity and trustworthiness are prioritized.

In contrast, DFL eliminates dependency on a central server, enabling direct peer-to-peer communication and avoiding SPoF vulnerabilities. This decentralized architecture can continue functioning even if some nodes fail. In that way, DFL shows fairness, as its dynamic and adaptable topologies accommodate diverse client needs and also external conditions. The radar chart illustrates DFL advantage in personalization, where its decentralized structure supports dynamic updates and localized optimization, allowing clients to tailor the model to their specific data distributions. However, these benefits come at the cost of scalability, as peer-to-peer exchanges may require more bandwidth and computational resources [20]. Additionally, DFL´s complexity introduces challenges in management and security, particularly in ensuring model consistency and defending against malicious activities such as poisoning attacks.

The radar chart highlights the trade-offs between the two approaches. FL is better suited for scenarios prioritizing efficiency, simplicity and trust in a central entity, while DFL excels in applications requiring robustness, adaptability and fairness in highly distributed or dynamic environments. The choice between FL and DFL depends on the specific priorities of the application. FL offers greater simplicity and efficiency but is constrained by its centralized vulnerabilities. On the other hand, DFL provides a more adaptable and robust framework but requires innovative solutions to address its challenges in communication, security and management.

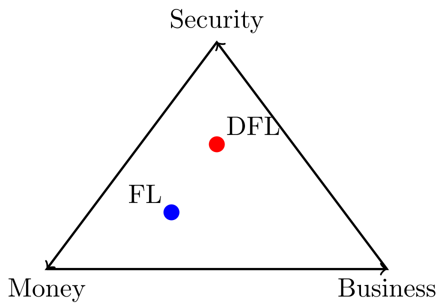

Figure 4: Illustration of the interplay between Money, Business, and Security for both FL and DFL.

Figure 4 represents the fundamental trilemma of choosing a solution for a company. Money, Business and Security are depicted as the vertices of a triangle, the solution can prioritize at most 2 aspects while compromising on the third.

Currently we believe that FL emphasizes cost efficiency while DFL is more security focused. Neither is inherently better or worse for the business. At first impression FL might seems more costly to use due to the cost for buying and running the central server, but the process of developing a DFL network requires more people with competencies and thus is effectively more expensive to implement.

4 Challenges and Future Directions

Our research indicates that learning in a fully decentralized manner currently faces significant challenges in terms of security and privacy. As DFL is an emerging paradigm, there is no consensus in literature on frameworks for deploying DFL architectures, and many ideas are still theoretical [14]. Future research should study the complex interplay between network topology and data distribution in DFL, leading to more nuanced and personalized network dynamics and model aggregation strategies [16]. Integration of blockchain has received academic interest as viable defenses against privacy exposure, data poisoning, and malicious attacks, and offers existing infrastructure and mechanisms to enhance trustworthiness among clients [20]. Reducing communication overhead and increasing scalability appear to be a major bottleneck which often is in trade-off with privacy and security. There is a need for more empirical studies concerning each component of DFL, and tailoring the DFL framework to diverse real-world applications is now an important challenge. The possible improvements in data privacy and security with decentralization makes DFL an attractive candidate for scenarios like healthcare, Industrial IoT (IIoT) operations, mobile edge computing, and smart cities [14].

A novel approach by Zhang et al., 2024 [18] also introduces a lightweight energy disaggregation model using grouped convolution and channel shuffle techniques, reducing the number of parameters and computations. Tests on the REFIT dataset demonstrated comparable performance to traditional methods like FedAvg, as mentioned earlier, while drastically cutting communication costs and energy usage. This solution aligns with the comparison of FL and DFL, since the decentralized architectures can overcome traditional FL´s communication vulnerabilities while maintaining performance in resource constrained environments.

DFL is currently considered a state-of-the-art approach compared to FL. Its use is restricted by several factors, primarily: the difficulty of deploying a DFL network, the processing power of each nodes (which may be insufficient depending on the application) and by communication challenges between nodes.

While the difficulty of deploying such network might remain, the processing power and transfer rate is constantly evolving. We can illustrate this idea with the shift in the AI field: Large Language Models (LLM) started as a service hosted on a big, powerful and remote server, now the trend is to run AI locally, with computer processor marketed as being AI compatible (Microsoft Copilot, Apple Intelligence). On the other hand, improvements on processor are not as dramatic as they once were, and experts have predicted the death of Moore's Law, due to the increasing difficulty of reducing the size of transistors. Programmers will not be able to rely on processing improvement, but will be forced to optimize their code more. As for the communication problem, the constant increase of transfer rate with new technologies such as 5G and in the future 6G.

5 Conclusion

Our research highlighted the trade-offs between centralization and decentralization in federated learning frameworks, offering insights into designing more secure and powerful distributed machine learning systems. DFL presented a paradigm shift by eliminating single points of failure but its flexibility in network topology, aggregation, and communication protocols introduced unique challenges in ensuring security and data privacy. The comparative analysis of FL and DFL revealed their distinct strengths and security vulnerabilities, with attacks such as poisoning, Byzantine faults, and server exploitation posing significant risks to FL, and privacy breaches, model poisoning, and blockchain vulnerabilities threatening DFL. Defensive strategies are pivotal in mitigating these attacks, and as research progresses to address the current limitations, DFL has the potential to redefine collaborative machine learning.

References

[1] Hadeel Abd El-Kareem Abd El-Moaty Saleh, Ana Fernández Vilas, Manuel Fernández-Veiga, Yasser El-Sonbaty, and Nashwa El-Bendary. Using decentralized aggregation for federated learning with differential privacy. In *Proceedings of the 19th ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, &; Ubiquitous Networks on 19th ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, &; Ubiquitous Networks*, MSWiM '22, page 33–39. ACM, October 2022.

[2] Tuncer Aysal, Mehmet Yildiz, Anand Sarwate, and Anna Scaglione. Broadcast gossip algorithms for consensus. *Signal Processing, IEEE Transactions on*, 57:2748 – 2761, 08 2009.

[3] Nader Bouacida and Prasant Mohapatra. Vulnerabilities in federated learning. *IEEE Access*, 9:63229–63249, 2021.

[4] Diego Cajaraville-Aboy, Ana Fernández-Vilas, Rebeca P. Díaz-Redondo, and Manuel Fernández-Veiga. Byzantine-robust aggregation for securing decentralized federated learning, 2024.

[5] Chao Feng, Alberto Huertas Celdrán, Janosch Baltensperger, Enrique Tomás Martínez Beltrán, Pedro Miguel Sánchez Sánchez, Gérôme Bovet, and Burkhard Stiller. Sentinel: An aggregation function to secure decentralized federated learning, 2024.

[6] Ehsan Hallaji, Roozbeh Razavi-Far, Mehrdad Saif, Boyu Wang, and Qiang Yang. Decentralized federated learning: A survey on security and privacy. *IEEE Transactions on Big Data*, 10(2):194–213, April 2024.

[7] Chuang Hu, Huang Huang Liang, Xiao Ming Han, Bo An Liu, Da Zhao Cheng, and Dan Wang. Spread: Decentralized model aggregation for scalable federated learning. In *Proceedings of the 51st International Conference on Parallel Processing*, ICPP '22, New York, NY, USA, 2023. Association for Computing Machinery.

[8] Jingyan Jiang and Liang Hu. Decentralized federated learning with adaptive partial gradient aggregation. *CAAI Transactions on Intelligence Technology*, 5, 09 2020.

[9] Hyesung Kim, Jihong Park, Mehdi Bennis, and Seong-Lyun Kim. Blockchained on-device federated learning. *IEEE Communications Letters*, 24(6):1279–1283, 2020.

[10] Anusha Lalitha. Fully decentralized federated learning. 2018.

[11] Léo Lavaur, Marc-Oliver Pahl, Yann Busnel, and Fabien Autrel. The evolution of federated learning-based intrusion detection and mitigation: A survey. *IEEE Transactions on Network and Service Management*, 19(3):2309–2332, 2022.

[12] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent, 2017.

[13] Habib Ullah Manzoor, Sajjad Hussain, David Flynn, and Ahmed Zoha. Centralised vs. decentralised federated load forecasting in smart buildings: Who holds the key to adversarial attack robustness? *Energy and Buildings*, 324:114871, 2024.

[14] Enrique Tomás Martínez Beltrán, Mario Quiles Pérez, Pedro Miguel Sánchez Sánchez, Sergio López Bernal, Gérôme Bovet, Manuel Gil Pérez, Gregorio Martínez Pérez, and Alberto Huertas Celdrán. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. *IEEE Communications Surveys & Tutorials*, 25(4):2983–3013, 2023.

[15] Luigi Palmieri, Chiara Boldrini, Lorenzo Valerio, Andrea Passarella, and Marco Conti. Exploring the impact of disrupted peer-to-peer communications on fully decentralized learning in disaster scenarios, 2023.

[16] Luigi Palmieri, Chiara Boldrini, Lorenzo Valerio, Andrea Passarella, and Marco Conti. Impact of network topology on the performance of decentralized federated learning, 2024.

[17] Zhe Peng, Jianliang Xu, Xiaowen Chu, Shang Gao, Yuan Yao, Rong Gu, and Yuzhe Tang. Vfchain: Enabling verifiable and auditable federated learning via blockchain systems. *IEEE Transactions on Network Science and Engineering*, 9(1):173–186, 2022.

[18] Yuan Sheng, Feng Gao, and Kangjia Zhao. Efficient communication for decentralized federated learning: An energy disaggregation case study. *School of Control Science and Engineering, Shandong University*, 2024.

[19] Jie Wen, Zhixia Zhang, Yang Lan, Zhihua Cui, Jianghui Cai, and Wensheng Zhang. A survey on federated learning: challenges and applications. *Int. J. Mach. Learn. Cybern.*, 14(2):513–535, 2023.

[20] Liangqi Yuan, Ziran Wang, Lichao Sun, Philip S. Yu, and Christopher G. Brinton. Decentralized federated learning: A survey and perspective, 2024.

[21] Haoran Zhang, Shan Jiang, and Shichang Xuan. Decentralized federated learning based on blockchain: concepts, framework, and challenges. *Computer Communications*, 216:140–150, 2024.

[22] Shuang Zhao, Yalun Wu, Rui Sun, Xiaoai Qian, Dong Zi, Zhiqiang Xie, Endong Tong, Wenjia Niu, Jiqiang Liu, and Zhen Han. Blockchain-based decentralized federated learning: A secure and privacy-preserving system. In *2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, pages 941–948, 2021.