Hierarchical Federated Learning with Privacy

Varun Chandrasekaran*, Suman Banerjee*, Diego Perino#, Nicolas Kourtellis#
University of Wisconsin-Madison*, Telefonica Research#

2024 IEEE International Conference on Big Data (BigData) | 979-8-3503-6248-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/BigData62323.2024.10826023

***Abstract*—Recent work highlights how gradient-level access can lead to successful inference and reconstruction attacks against federated learning (FL). In such settings, differentially private (DP) learning is known to provide resilience. However, approaches used in the status quo (*i.e.,* central and local DP) introduce disparate utility vs. privacy trade-offs. In this work, we mitigate such trade-offs through *hierarchical FL (HFL)*. For the first time, we demonstrate that by the introduction of a new intermediary level where calibrated noise can be added, better trade-offs can be obtained; we term this *hierarchical DP (HDP)*. Our experiments with 3 different datasets (commonly used as benchmarks for FL in prior works) suggest that HDP produces models as accurate as those obtained using central DP, where noise is added at a central aggregator at a lower privacy budget.**

I. Introduction

In federated learning (FL), clients share the gradient updates associated with local learning with a central aggregator (CA) which uses them to obtain a *global model update*. FL is assumed to provide privacy [1]: the data never leaves the federated clients. However, recent work [2], [3], [4] has shown that FL is susceptible to attacks where the *private* training data can be reconstructed by observing the shared gradients. Differential privacy [5] (DP) provides guarantees on *privacy leakage* of various mechanisms, including those used by FL [6], [7], [8]. It has been shown that DP learning algorithms alleviate the aforementioned attacks [9], [10]. However, they introduce a trade-off between protecting data privacy, and *utility* of the model learned.

Thus far, DP guarantees in FL have been investigated at two levels: at the CA [11] (*i.e.,* central DP or CDP), or at the client level (*i.e.,* local DP or LDP) [12], [13], [14]. The former assumes that the clients *trust* the CA, while the latter does not, and *both approaches* introduce disparate privacy vs. utility trade-offs. In this work, we aim to understand if a compromise can be reached through an intermediary approach, by introducing *hierarchies* in the FL pipeline. We call this *hierarchical DP* (or HDP). Clients form, or belong to *zones*; updates from clients within a zone are aggregated by a *super-node* (an elected/chosen client), and the updates from zones (*i.e.,* super-nodes) are aggregated at the CA. Such hierarchies are omnipresent in daily computing systems (e.g., telecommunication networks, the internet infrastructure, etc.) [15]. These "naturally occurring" clusters of clients have embedded trust in their formation [16].

In this paper, we take the first step in analyzing how hierarchies are beneficial with respect to privacy, and what are the key technical challenges to be solved. In fact, this paper is the first to address the following challenges in this space. The first challenge is in constructing such a hierarchy. We analyze different approaches that reflect the aforementioned scenarios and discuss the trade-offs in § IV-A. The second challenge involves making modifications to the standard approaches to FL to ensure that the newly proposed HFL approach learns the same model as in the status quo. We discuss this in § IV-B. The third challenge involves formalizing an algorithm (to provide DP) to be used in an HFL ecosystem. Determining the exact mechanism has direct implications on the privacy vs. utility trade-off, and provides avenues for *privacy amplification*, which we discuss in § IV-C with our novel algorithm. We are also the first to generalize the approach by considering 7 different case scenarios of simultaneous needs for employing LDP, HDP and CDP in the same FL setting, in § V. Our final challenge is understanding the various threats faced by our new constructions (§ VII) in an experimental manner.

We first validate that HDP is robust to reconstruction adversaries. We also analyze faulty behavior at an intra- and inter-zonal granularity, and describe the information leakage by adversaries in such settings, even when an adversarial client is elected as a super-node. We also analyze prior attacks in the FL setting, and comment on the applicability of the same in the context of hierarchical FL. In particular, we look into adversaries in super-node level and discuss potential adversarial inferences they may perform. We show that HDP can thwart such attacks. Finally, we experimentally study the privacy vs. utility trade-off on different datasets (which are used in prior work related to this topic) and setups. We demonstrate that the (privacy and utility) performance of hierarchical FL sits between local and CDP. Surprisingly, we observe that the utility benefits obtained through our proposal are very close to that obtained by learning a model with CDP.

II. Background & Related Work

Federated Learning (FL) involves learning with many *federated clients* in a decentralized manner [11]. At each *round*, the *central aggregator (CA)* shares its weights with all federated clients. At any given point in time, only $k \leq n$ clients may be *online*. Each client performs training (on its private dataset) for multiple epochs (`FedAvg` [1]) or a single epoch (`FedSGD` [17]) and shares the update with the CA. Once aggregated, the CA updates its weights and shares it to all clients to be used as the starting point for the next round. FL is assumed to provide privacy by design. However, recent work invalidates this assumption [4], [3].

Differential Privacy (DP) was proposed by Dwork *et al.* [5]. Let ε be a positive real number, and $\mathcal{A}$ be a randomized algorithm that takes a dataset as input. The algorithm $\mathcal{A}$ is said to provide ε-DP if, for all datasets D_1 and D_2 that differ

on a single element, and all subsets $\mathbf{O}$ of the outcomes of running $\mathcal{A}$: $\Pr[\mathcal{A}(D_1) \in \mathbf{O}] \leq e^{\varepsilon} \cdot \Pr[\mathcal{A}(D_2) \in \mathbf{O}]$ where the probability is over the randomness of the algorithm $\mathcal{A}$. Parameter ε is also known as the *privacy budget*.

Central & Local DP: The traditional model defined earlier, also known as the *central DP* (CDP) model, implicitly assumes the existence of a trusted entity that does not deviate from protocol specification. However, such an assumption may not always hold. *Local DP* (LDP) [12] assumes that each data contributor adds the noise in-situ. Naturally, such a mechanism has limited knowledge of the overall function being computed on all the data, and overestimates the amount of noise required. The relationship between LDP and CDP is dependent on the mechanism used to achieve DP. For example, the Laplacian mechanism [5] ensures that ε-LDP also provides ε-CDP.

FL with DP: [18] propose the first approach where DP can be combined with FL to provide formal privacy guarantees. Similar ideas are proposed in the work of [10]. In both settings, however, CA is able to observe either noise-free gradients or noisy gradients from corresponding clients. To break this connection, [19], [20] propose the notion of *secure aggregation*, a variant of MPC, which provides the CA an aggregated view of all gradients (noisy/non-noisy) from the clients.

Hierarchical FL (HFL): [21] propose a mechanism to ensure communication-efficient and coordinated learning in the context of HFL. Here, the notion of hierarchies stems from the presence of clients communicating with small base stations (or cellular towers) which act as intermediaries, who further communicate with macro base stations (or the CA). Similar claims are made in the work of [22]. It is important to note that prior work focuses on improving the scalability/communication-efficiency of FL through the introduction of hierarchies, whilst ours is studying the implications on privacy.

Hierarchical DP (HDP): Prior work by Shi *et al.* [23] considers noise addition in a hierarchical manner. However, their proposal does not consider noise addition at every round but at specific rounds; this is a deviation from traditional approaches with serious consequences. Particularly, we remark that the privacy analysis is flawed as the authors consider situations where the gradient is clipped at only specific iterations and not all; convergence analysis of private learning mechanisms requires that the gradients be clipped and noise at all iterations. They also do not evaluate the resilience of their proposal to practical privacy attackers (such as reconstruction adversaries as we do). Finally, they do not consider the ramifications of their work in settings where DP noise is not added at specific iterations but the gradients are shared with edge-server, providing the edge-server access to unmodified data, whereas our proposal circumvents this by requiring secure aggregation to ensure that the edge-server never sees gradients from clients, but only sees an aggregate view.

III. Architectural Overview

A. Approach & Hierarchical DP

Typically in FL, there is a total of two actors at two conceptual levels of a hierarchy *i.e.,* client(s) at level 0 and the server at level 1. In our proposal, we assume the existence of three main actors at three different (conceptual) levels of the hierarchy (as noted in Figure 1). At level 0, we have the clients who hold private data. At level 2, we have the CA. The key difference lies in the middle: at level 1 (*i.e.,* the intermediate level), we introduce a new entity called the *super-node*, responsible for processing requests from the online clients in a particular region (*i.e.,zone* $i \in [s]$; s is the total number of zones). The presence of level 1 introduces a hierarchical approach for FL. An added feature in our proposal revolves around the selection of the super-node: they can either (a) be elected by a pool of their peers and thus are within the *same trust boundary/region* as their peers, or (b) be chosen as an entity in a different trust region (to both the clients and the CA). Note: in practice, there can be many intermediary layers; we stick to one for the paper's remainder.

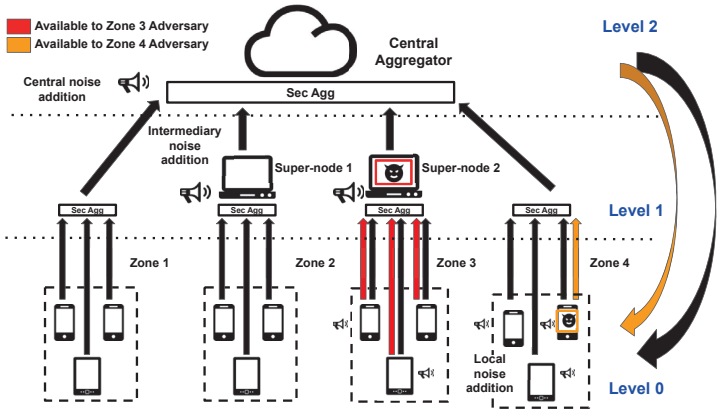


Fig. 1: **Architectural overview of our approach (§ III).**

In the status quo, DP guarantees are obtained through two mechanisms: noise addition at the CA (*i.e.,* CDP), or noise addition locally at each federated client (*i.e.,* LDP). Empirical evidence suggests that the CDP mechanism will result in a final model with higher accuracy at the expense of trusting the CA. The LDP mechanism removes this trust assumption, but requires greater noise addition at each client. To provide the best of both worlds, we propose *hierarchical DP* (or HDP).

From Figure 1, note that the functionality of the super-nodes is similar to that of the CA. Consequently, the formal definition of Hierarchical DP (HDP) is the same as that of CDP, adjusted for the participation of intra-zonal clients.

Definition III.1 (Hierarchical DP). A ε_{HDP}-hierarchical randomizer $R : \mathbf{D} \rightarrow \mathbf{O}$ is a differentially private algorithm that takes a database of arbitrary sizes (from the same zone i). That is $\Pr[R(D_i) = o] \leq e^{\varepsilon_{HDP}} \cdot \Pr[R(D_i') = o]$ for all $D_i, D_i' \in \mathbf{D}$ ($\forall i \in [s]$) such that they differ by a single row, and all $o \in \mathbf{O}$. The probability is taken over the randomness of the procedure R. Note, that the elements of the database may already be noisy due to effects of local randomizers.

Prior Work: [23] also considers noise addition in a hierarchical manner. However, they do not consider noise addition at every

round; this is a deviation from traditional approaches and results in flawed privacy analysis. Additionally, the authors consider situations where the gradient is clipped at only specific rounds; convergence analysis of private learning mechanisms requires that the gradients be clipped and noised at all rounds.

B. Threat Model

We assume a minority of ζ ($1 \leq \zeta < \frac{k}{3}$)[1] online clients are adversarial. These are honest-but-curious *i.e.,* they can not deviate from protocol specifications, but can collude. This is the commonly followed threat model in most prior works [2], [4]. Their sole goal is to infer information about the training data of a target client (or group of clients). The only knowledge these adversaries are privy to are the gradient updates shared during training. In particular, the adversaries we consider are able to view: (1) any updates *they* generate; (2) the joint update shared from the CA; and (3) the update generated at the the super-node, iff the adversary is located at said level.

Adversarial Goals. The adversary aims to: (a) identify if a particular data-point (or a particular attribute within a data-point) was used during training[2], or (b) reconstruct the data used for training by another client by observing the gradients shared. These adversaries can utilize: (a) data inference [2], or (b) data reconstruction [4], [24] to achieve their goals. Note that the reconstruction attack can capture all effects of the inference attack; once the adversary has access to the exact data used for training, it can also infer if the data-point possesses specific attributes (or not). While such an attack is computationally more expensive than inference attacks, they are more realistic.

Note: Recent work [3], [25] formalize active adversaries whose aim is to subvert different parts of the protocol. The setting they assume is one where a CA is able to maliciously modify the states shared to (different) participants so as to ensure efficient input reconstruction. We stress that such an attack deviates from our honest-but-curious adversarial model. Additionally, and more importantly, the authors of *both these works* note that obfuscating the gradient using DP (as we propose) extensively minimizes attack efficacy.

IV. Designing Hierarchies

A. Choosing Super-Nodes

1. Exploiting Inherent Hierarchies: Hierarchies exist in communication networks [26]: this information can be used to select super-nodes. For example, hierarchies are introduced by edge computing, where the edge-based base-stations [15] serve as an intermediary between the clients and the CA. Traditionally, the manufacturer of various hardware components are manufactured by different vendors [27]; we can assume that the clients, super-nodes, and CA belong to different trust regions, minimizing collusion between them.

2. Elections: Another approach is to *elect* the super-node in a *fully decentralized manner*, *i.e.,* the super-node is one of the clients. This too, ensures that the super-node is from a different trust region in comparison to the CA. First, clients are grouped into *zones*; the grouping can be based on (a) geography, (b) compute capabilities, or (c) some form of structured or unstructured P2P overlay organization [28]. Then, a distributed election protocol is run within a zone. Leader election protocols can be drawn from past literature that studied it under the context of super-node selection in P2P networks using different assumptions, metrics, etc. [29], [30]. In our setting, such a process assumes that the clients are aware of others who are active in that particular zone for the particular round, a common assumption [31], [19]. Another assumption made is that these participants can communicate the outcome of the election between each other. We do note that such assumptions induce additional communication overheads, but they do not add to the overall complexity of our proposal (as we see in Appendix V). We request the curious reader to refer to these works (and this survey [32]) for more details.

Election Integrity: One key requirement in our setting is that the election of the super-nodes is randomized, *i.e.,* a particular client in a zone has bounded probability of being elected as the super-node[3]. Once the super-node is down, then a new one is elected to replace it. However, recall that a small fraction ζ of online clients are adversarial, and can subvert the election protocol through collusion. To this end, different election protocols and assumptions provide different guarantees [33]. For example, to obtain conventional fault tolerance at zone i, a majority of the k_i clients must be honest [33] (*i.e.,* $\zeta_i < \frac{k_i - 1}{2} \leq \zeta$). Similarly, to obtain byzantine fault tolerance, $\zeta_i \leq \frac{k_i - 1}{3} \leq \zeta$ [33]. While there exists no formal/principled way to practically enforce such constraints, these limits serve as bounds to understand the limitations of election integrity.

Note, however, that the elected super-node has complete purview to the gradients from clients; these gradients may not be masked through the addition of noise needed for DP. To this end, we advocate for the usage of secure aggregation (SA) [20] protocols (within zones) to ensure that the super-node gets an aggregate view of the gradients from individual clients. In our setting, SA needs to be applied in all zones, and prior work suggests that SA protocols are practical (*i.e.,* introduces a tolerable time delay) for only a small number of clients [20]. Care must be taken in ensuring that the number of (online) clients per zone is also small. How this is achieved can be determined by future research. But we would like to stress that the time taken for SA within the zone in the HFL setting will be notably lower than in the status quo (without hierarchies) since the number of online clients per zone (for HFL) will be lower than the number of online clients overall (as needed by the status quo).

B. Proposed Algorithm and Correctness

Algorithm for Hierarchical FL with DP: Algorithm 1 is based on the approach taken by [18], with some key modifications to incorporate hierarchical FL for s zones. From line 6, observe

[1]Needed for fault tolerance, as we will explain later.

[2]A dataset comprises of many data-points, each of which comprises of numerous attributes.

[3]Assuming each client has a probability p of being randomly elected (where $p = \frac{1}{m}$), the probability that the client is elected in all T rounds is p^T which is very low.

1: *Parameters*
   a. user selection probability $q \in (0, 1]$
   b. per-user example cap $\hat{w} \in \mathbf{R}^+$
   c. noise scale $z \in \mathbf{R}^+$
   d. UserUpdate (for `FedAvg` or `FedSGD`)
   e. ClipFn (FlatClip or PerLayerClip)
   f. Parameter W_{min}
2: *Procedure:*
3: Initialize model θ^0
4: $w_k = \min(\frac{n_k}{\hat{w}}, 1)$ for all users k
5: **for** each round $t = 0, 1, 2, \dots$ **do**
6: **for** each zone $i = 1, 2, \dots, s$ **do**
7: $\mathcal{C}_i^t \leftarrow$ (sample users with probability q)
8: $W = \sum_{k \in \mathcal{C}_i^t} w_k$
9: **for** each user $k \in \mathcal{C}_i^t$ **in parallel do**
10: $\Delta_k^{t+1} \leftarrow \text{UserUpdate}(k, \theta^t, \text{ClipFn})$
11: **end for**
12: $\Delta(i)^{t+1} = \begin{cases} c\frac{\sum_{k \in \mathcal{C}_i^t} w_k \Delta_k^{t+1}}{q \cdot W} & \text{for FlatClip} \\ \frac{\sum_{k \in \mathcal{C}_i^t} w_k \Delta_k^{t+1}}{\max(q \cdot W_{min}, \sum_{k \in \mathcal{C}_i^t} w_k)} & \text{for PerLayerClip} \end{cases}$
13: $\mathcal{S}_i \leftarrow$ (bound on $||\Delta_k^{t+1}||_2$ for ClipFn)
14: $\sigma_i \leftarrow \left\{ \frac{z\mathcal{S}_i}{q \cdot W} \text{ for FlatClip or } \frac{2z\mathcal{S}_i}{q \cdot W_{min}} \text{ for PerLayerClip} \right\}$
15: $\theta^{t+1} \leftarrow \theta^t + \frac{1}{s} \sum_{i \in [s]} \left(\Delta(i)^{t+1} + \mathcal{N}(0, I\sigma_i^2) \right)$
16: **end for**
17: **end for**

Algorithm 1: Modified FL mechanism with zonal privacy.

that gradient aggregation first occurs at a *zonal level*. This in turn leads to re-calibration of various parameters (lines 7-14) required to provide the DP guarantee[4]. Finally, the *noised zonal gradients* are averaged and aggregated in line 15. Note that in scenarios where *uniform weighting* is applied (*i.e.,* the contribution of each client is the same), $w_i = \frac{1}{k}$. Additionally, the variance of the noise to be added (σ_i in line 14) is calculated as a function of the online clients *per zone* (as noted by he denominator term: $q \cdot W$ or $q \cdot W_{min}$), and not the total fraction of clients across all zones, as W is re-calibrated based on the selected clients per round per zone.

The proposed approach calculates the sensitivity based on the clipping bound $\mathcal{S}_i$ that is calculated across all clients across all zones; obtaining this information in practice will require additional communication between the super-nodes. Thus, this is approximated through the existence of a global clipping bound C such that $C \geq \max_{i \in [s]} \mathcal{S}_i$.

To ensure algorithmic correctness, we first generalize the construction described earlier as follows: there are k online clients, and a total of s zones (each with its own super-node). Let us assume that each zone has an equal number of clients m such that $m \cdot s = k$. The clients within each zone clip their gradients (as is commonly done [18]), and set the clipping norm to a global constant. Recall that nodes in the zone utilize SA to share an aggregated view of the gradients with the super-node. Thus, each of the s super-nodes has received an update from the corresponding online clients in their zone. These s super-nodes will forward these updates to the CA, where another round of re-scaled averaging occurs[5]. In the baseline case, this intermediary-level does not exist (and consequently, such intermediary-level forwarding does not exist); noise addition (should the approach utilize CDP) occurs at the CA. The super-node can (a) apply noise required for DP as is to the aggregated gradients from each zone (and averaging by the total number of online clients for that round will occur at the CA), or (b) average the gradients first (by the number of online clients in the zone)[6] before adding noise. Observe that the sensitivity of case (b) will be lower than that of case (a); we advocate for averaging at the super-node before noise addition.

C. Privacy Benefits

We begin by introducing preliminaries of the Gaussian mechanism. Note that the formalism below is borrowed from the work of [5].

1. ℓ_2-sensitivity: The ℓ_2-sensitivity of $g : \mathbf{N}^{|X|} \rightarrow \mathbf{R}^k$ is $\Delta_2 g = \max_{x,y \in \mathbf{N}^{|X|}} ||g(x) - g(y)||_2$ where $||x - y||_1 = 1$, $||.||_p$ denotes the p-norm, and X denotes some universe (such as the universe of all databases).

2. Gaussian Mechanism (GM): Let the privacy budget $\varepsilon \in (0, 1)$. For some constant $c^2 > 2\log(\frac{1.25}{\delta})$, the GM with parameter $\sigma \geq \frac{c\Delta_2 g}{\varepsilon}$ is (ε, δ) DP.

Based on these fundamentals, next, we proceed to define and prove our novel lemma on privacy amplification that involves the connection between LDP and CDP, when k clients are involved.

> **Lemma (Privacy Amplification):** If all k (online) clients obtains $(\varepsilon_{LDP}, \delta)$ LDP using the GM, the CA obtains $(\varepsilon_{CDP}, \delta)$ DP, where $\varepsilon_{CDP} = \frac{\varepsilon_{LDP}}{\sqrt{k}}$.

Proof: Assume the existence of a global sensitivity bound C, and k online clients each of which utilize the GM to independently add noise to the gradients (denoted g_j for $j \in [k]$) being computed to achieve $(\varepsilon_{LDP}, \delta)$ LDP guarantees. Thus, each client utilizes constants c_j for $j \in [k]$ which satisfy the above condition (in the definition of the GM), and samples noise from distributions with standard deviation σ_j for $j \in [k]$, *i.e.,* each client returns the following value: $g_j + \eta_j, \forall j \in [k]$ where $\eta_j \sim \mathcal{N}(0, \sigma_j^2), \forall j \in [k]$. The CA computes $\hat{g} = \frac{1}{k} \sum_{j=1}^{k} (g_j + \eta_j)$ (*e.g.,* as required by `FedAvg` [11]). Note that g_j is the gradient calculated by client j multiplied by the size of client j's dataset, and then divided by the total dataset size of all online clients. We know that the sum of two Gaussian variables is Gaussian. Extending this, we know that $\sum_{j=1}^{k} \eta_j = \eta$ since $\sum_{j=1}^{k} \mathcal{N}(0, \sigma_j^2) = \mathcal{N}(0, \sum_{j=1}^{k} \sigma_j^2) = \mathcal{N}(0, \hat{\sigma}^2)$ (where $\eta \sim \mathcal{N}(0, \hat{\sigma}^2)$). We also know that $\hat{\sigma}^2 =$

[4]For more details on the various forms of clipping that can be employed, refer the original work [18].

[5]In the case with no intermediaries, the aggregator will average the gradient based on the number of online parties k. To preserve correctness in the setting with intermediaries, the aggregator first re-scales the gradient by the number of online clients per zone, and then performs averaging based on the total number of online clients. This preserves algorithmic correctness.

[6]We assume that the SA protocol returns just the sum, and not the average. The protocol can easily be modified to return the average as well.

	Aggregator	Super-node	Client	Privacy @ Center
C1	✗	✗	✓	$\frac{\varepsilon}{\sqrt{k}}$
C2	✗	✓	✗	$\frac{\varepsilon}{\sqrt{s}}$
C3	✗	✓	✓	$\frac{\varepsilon}{\sqrt{\beta \cdot s \cdot m + (1-\beta) \cdot s}}$
C4	✓	✗	✗	ε
C5	✓	✗	✓	$\frac{\varepsilon}{\sqrt{\alpha . k + 1}}$
C6	✓	✓	✗	$\frac{\varepsilon}{\sqrt{\beta . s + 1}}$
C7	✓	✓	✓	$\frac{\varepsilon}{\sqrt{\beta \cdot s \cdot m + \alpha \cdot s + 1}}$

TABLE I: Different configurations that are possible with our proposed hierarchical scheme. The 7 indicates no noise addition, and the 3 indicates noise addition. The privacy budget presented in the last column is (a) devoid of any amplification induced due to shuffling (through the presence of SA), and (b) what is viewed at the CA. We will explain when these cases are realizable in § V.

$\sum_{j=1}^{k} \sigma_j^2 \geq \frac{\Delta_2 g^2}{\varepsilon^2} \sum_{j=1}^{k} c_j^2$. Setting $c_j = C$ and $\sigma_j = \sigma$ for all $j \in [k]$, we see that the CA is $(\frac{\varepsilon_{LDP}}{\sqrt{k}}, \delta)$ DP, if each client is $(\varepsilon_{LDP}, \delta)$ LDP.

Amplification from Shuffling [34], [35]: Amplification allows for stronger privacy (measured at the CA) when noise is added at a lower level of the hierarchy (be it clients, super-nodes, or both). Note that the stage where the noise is added dictates the denominator term in the privacy budget (assuming amplification induced only due to the sum of GMs): $\sqrt{k}$ if noise is added at each of the clients, and $\sqrt{s}$ if noise is added at each of the super-nodes. Since $s < k$, the privacy budget when the noise is added at the super-nodes is more than when added at the client-level. Intuitively, this suggests that a classifier learnt using HDP is going to be more utilitarian than one learnt with LDP. In § VII, we empirically validate this claim.

Why Use Super-Nodes? Because of the aforementioned amplification result, one might wonder why we need to utilize super-nodes to begin with: the clients themselves should be able to re-calibrate the amount of noise they have to add locally to achieve the desired privacy level at the center. However, as discussed earlier in § IV-A, super-nodes who may (or not) reside in a different trust boundary than the client or the CA, are naturally existing in several settings (*e.g.,* home or office router, telco antenna, etc.) and would be only consequential to examine their participation in an FL process such as HFL. Additionally, the utility benefits are better than that of the re-calibrated LDP (as we will see in § VII-A). Finally, the use of super-nodes provides better *scalability* of the aforementioned secured aggregation step due to the small number of clients within a zone.

V. Generalization

In this section, we provide a generalization of the technique discussed in this paper, with the corresponding privacy budgets (using the basic composition theorem [5]). Assume a total of k online clients per round, there are k_i clients per zone such that $\sum_{i \in [s]} k_i = k$. To ease discussion, let us assume $k_i = m$ (for all i). Several cases are possible, and analyzed next, for the first time in the HFL with HDP setting:

Case 1 corresponds to pure LDP when *all clients* add the required noise to obtain ε LDP. From the CA's perspective, the overall privacy is $\frac{\varepsilon}{\sqrt{k}}$. This is visualized in Figure 1, towards the far right (in the absence of SA).

Case 2 corresponds to pure HDP when *all super-nodes* add the required noise to obtain ε HDP. From the CA's perspective, the overall privacy is $\frac{\varepsilon}{\sqrt{s}}$. This is visualized in Figure 1 in zone 2 (in the absence of SA by the super-nodes). The privacy budget calculated here is in an ideal setting, where the different clients trust their corresponding super-node with de-noised gradients.

Case 3 is realizable when a fraction of all online clients choose to add the required noise for LDP, but the remainder do not. To ensure that the remaining clients receive privacy, the super-node corresponding to these clients needs to add noise to ensure DP. To simplify the math, let us assume that clients corresponding to $\beta \cdot s$ super-nodes choose to utilize LDP and the remaining $(1-\beta) \cdot s$ super-nodes provide HDP. If each zone has m clients, then at each of the $\beta.s$ super-nodes, we observe $\frac{\varepsilon}{\sqrt{m}}$ DP. From the CA's perspective, the total privacy is $\frac{\varepsilon}{\sqrt{\beta \cdot s \cdot m + (1-\beta) \cdot s}}$ where the second term in the sum is the contribution of the $(1-\beta).s$ super-nodes that provide ε HDP, and the first term is due to the $\beta \cdot s \cdot m$ clients that provide ε LDP, and the sum itself is due to basic composition.

Note: For the setup described in § III, all clients want ε LDP guarantees (*i.e.,* $\beta = 1$) *and* the super-nodes provide ε HDP guarantees (*i.e.,* two layers of noise addition), the overall privacy at the center is $\frac{\varepsilon}{\sqrt{m \cdot s}}$

Case 4 corresponds to the setting of pure CDP when *only the CA* adds noise to obtain ε CDP. This is visualized in Figure 1, towards the far left.

Case 5 corresponds to the setting where a fraction α of clients do not trust the CA and consequently wish to achieve ε LDP. To provide privacy to the remainder, the CA also adds noise. Thus, the total privacy budget from the CA's perspective is $\frac{\varepsilon}{\sqrt{\alpha . k + 1}}$, where the first term is due to the α fraction which achieves LDP.

Case 6 is similar to **Case 5**, in that only a fraction β of super-nodes add noise to achieve ε HDP. To provide privacy to the remainder, the CA also adds noise. Thus, the total privacy budget from the CA's perspective is $\frac{\varepsilon}{\sqrt{\beta . s + 1}}$, where the first term is due to the β fraction which achieves HDP.

Case 7 corresponds to the scenario where (a) clients in $\beta.s$ zones do not trust any entity and wish to achieve ε LDP, (b) clients in $\alpha.s$ zones trust the super-nodes, and thus the super-nodes achieves ε HDP, and (c) for the clients in the remaining $(1 - \alpha - \beta) \cdot s$ zones, the CA is responsible for providing privacy. Thus, from the CA's perspective, the total privacy is $\frac{\varepsilon}{\sqrt{\beta \cdot s \cdot m + \alpha \cdot s + 1}}$, where the first term is due to (a), and the second term is due to (b).

Note: All the calculations above are in the absence of the amplification due to SA. Should it be considered, the only

change would be a removal of the square root over all terms.

VI. Implementation

We implement our proposed approach using a combination of `tensorflow-federated`[7] to provide the components required for FL, and `tensorflow-privacy`[8] to provide the machinery required for private learning. To ensure correct accounting, we modify the accounting libraries in `tensorflow-privacy` to accurately reflect sampling probability and number of iterations (in this case, rounds). To evaluate the efficacy of our approach, we consider the three datasets listed in Table II; only the EMNIST dataset is modified to exhibit the non-i.i.d property that is commonly associated with FL. Note that the objective of our evaluation is to understand the advantageous utility vs. privacy trade-offs introduced by our scheme (relative to other privacy schemes). To this end, how non-i.i.d the data is distributed between the clients is not an important compounding factor. The evaluation we will discuss can be considered as an average-case evaluation of the proposed HDP approach.

Dataset	Size	# Samples	# Classes	n
EMNIST [36]	$28 \times 28 \times 1$	382705	10	3383
CIFAR-10 [37]	$32 \times 32 \times 3$	60000	10	500
CIFAR-100 [38]	$32 \times 32 \times 3$	60000	100	500

TABLE II: Dataset characteristics.

The datasets follow a standard 80:20 split. All our experiments were executed on a server with 2 NVIDIA Titan XP with 128 GB RAM and 48 CPU cores running Ubuntu 20.04.2. Due to computational constraints, and issues in `tensorflow-federated` related to GPU execution[9], our experimental setup is conservative; we only perform a single run of each configuration. We choose representative architectures from prior work for the datasets we consider [1], [18]. In particular, we consider (a) a shallow 1 hidden layer DNN for EMNIST, and (b) 2 convolution layers followed by 2 fully connected layers for both CIFAR-10 and CIFAR-100. We will release all code used for our experiments on request.

For all experiments (unless explicitly specified otherwise), we choose a setup where $k = 100$ users are randomly sampled per federated round (this amounts to different values of the sampling/user-selection probability for different datasets). `FedAvg` [1] is used as the algorithm of choice, and each client performs local training for 5 epochs. FL is performed for 200 rounds. All participating clients use SGD as the learning algorithm, with a learning rate of 0.02 (this includes clients that are super-nodes). The server's learning rate is set to 1. These parameters were chosen based on the prescribed guidelines from the authors of `tensorflow-federated` and from prior work [1], [18].

[7]https://github.com/tensorflow/federated/tree/v0.16.1

[8]https://github.com/tensorflow/privacy

[9]https://github.com/tensorflow/federated/issues/832

VII. Experimental Evaluation

Through our experimental evaluation, we wish to answer the following questions:

1) Does HDP provide advantageous privacy vs. utility trade-offs in comparison to the baseline scenarios of using (a) LDP and/or (b) CDP? Note that for this evaluation, we assume that the clients trust the corresponding super-nodes (*i.e.,* it is a best-case evaluation of the efficacy of the approach)
2) Does HFL create a new attack surface for an adversary wishing to perform data (or membership) inference?

Recall from § IV-C, that HFL (with the GM for DP) leads to privacy amplification. Thus, to ensure a fair comparison, we only report the privacy budget (ε) calculated at the CA level in our results.

Our results suggest that:

1) *The theoretical intuition is corroborated in empirical measurements of utility*. As expected, HDP provides an advantageous trade-off between utility and privacy, and outperforms the LDP baseline.
2) HFL is also more resilient to data reconstruction attacks [39]. In the scenario where the super-node is adversarial, then it is able to perform perfect reconstruction (under some assumptions about the batch-size of data used for client-side learning, and/or in the absence of SA). However, if the client is adversarial, we observe that reconstruction efforts are less successful than in the scenario with CDP.

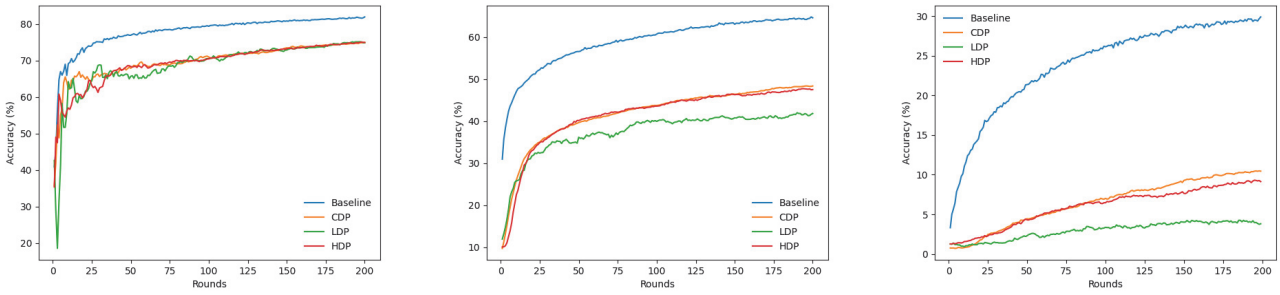
Incorrect Baselines: One might consider an approach that utilizes only SA as a baseline. However, such an approach results in privacy leakage in active adversarial models [25], [3]. Thus, while such approaches do not result in utility degradation (*i.e.,* the models learnt in the presence of the cryptographic aggregation protocols are the same without it), they are not (differentially) private.

A. Privacy vs. Utility Trade-Off

We first train without DP to get an estimate of the 2-norm of the gradients. This helps us estimate the clipping norm (C) to be used during DP training. We also observe the training duration (*i.e.,* exact epoch number) at which the validation accuracy saturates. Once this is obtained, we configure the noise multiplier (z) so as to enable DP training and log the privacy expenditure (measured by ε_{CDP}).

LDP, HDP and CDP Privacy vs. Utility: To ensure a fair comparison of privacy vs. utility, we convert the privacy expenditure in all settings to that of the CDP privacy expenditure (using the formulation presented in § IV-C)[10]. As an example, recall that if we wish to achieve $\varepsilon_{LDP} = 2$ for $k = 10$ clients, the corresponding privacy budget (from the perspective of the CA is $\hat{\varepsilon}_{CDP} = \frac{2}{\sqrt{10}}$ (when the clients do not utilize

[10]Note that the privacy budget values we report are the ones obtained without the amplification due to shuffling; SA is known to be communication inefficient [19], [20] and we wish to provide insight on the privacy vs. utility trade-offs in its absence as an average case estimate.



(a) EMNIST (b) CIFAR-10 (c) CIFAR-100

Fig. 2: We plot validation accuracy of training with differential privacy for 3 scenarios: (i) CDP (in orange), (ii) LDP (in green), and (iii) HDP (in red), in comparison to training without privacy (in blue). This is done across three datasets: (a) EMNIST, (b) CIFAR-10 and (c) CIFAR-100. Observe that HDP achieves better utility than LDP, and often close to CDP across datasets.

SA); the hat is used to denote that the privacy is calculated from a particular vantage point. Training was performed for each configuration (*i.e.,* LDP, HDP, and CDP) such that the privacy budget we were willing to expend was the same (*i.e.,* $\varepsilon_{LDP} = \varepsilon_{HDP} = \varepsilon_{CDP} = \varepsilon$). In Figure 2, the privacy is aligned *i.e.,* all three schemes aim to achieve the same privacy expenditure (*i.e.,* $\varepsilon_{LDP} = \varepsilon_{HDP} = \varepsilon_{CDP}$) = 3.06 for EMNIST, and ($\varepsilon_{LDP} = \varepsilon_{HDP} = \varepsilon_{CDP}$) = 24.80 for CIFAR-10. Thus, Table III contains corresponding values of the privacy budget achieved at the end of training (measured through $\hat{\varepsilon}_{CDP}$). Our *baseline* is one without DP training.

We consider a scenario where there are $s = 10$ super-nodes. Observe that learning simple tasks such as EMNIST (refer Figure 2a) with DP is achievable in all three settings; prior work has demonstrated that learning with privacy for EMNIST can be as performant as learning without privacy [18] *i.e.,* the accuracy degradation induced by LDP in comparison to CDP is minimal. However, as expected, HDP provides advantageous trade-offs (*i.e.,* the privacy budget is lower than CDP for comparable accuracy). The results are more interesting for complex datasets such as CIFAR-10 and CIFAR-100. First, observe that for the particular configuration we choose (*i.e.,* $n = 500$, $k = 100$), baseline accuracy is $\sim 63\%$ and $\sim 28\%$ for the two CIFAR versions, respectively. Note that these values are comparable to those achieved by [18] (refer Fig. 4). Training with CDP degrades this accuracy further. However, as expected, HDP is able to provide advantageous trade-offs. Note that CIFAR-100 is a more complex learning task than CIFAR-10, and yet (a) HDP achieves similar utility to CDP, and (b) much higher utility than LDP; *e.g.,* $\varepsilon_{HDP} = 3.06$ produces a more utilitarian model than $\varepsilon_{LDP} = 3.06$ for EMNIST.

The Influence of s: From our analysis, we can see that increasing the number of super-nodes (s) makes the scheme more private (the privacy budget is inversely proportional to the value of the number of parties where the noise is being added; in the case of HDP, this is s). To better understand this hyperparameter, we consider an experimental setting where we vary the value of k to 100, 200, and 300. Across these 3 settings, we vary the value of s to one of $\{10, 20, 30, 40, 50\}$ across all

Dataset	LDP	HDP	CDP
EMNIST	0.30	0.96	3.06
CIFAR-10	2.48	7.48	24.80
CIFAR-100	2.48	7.48	24.80

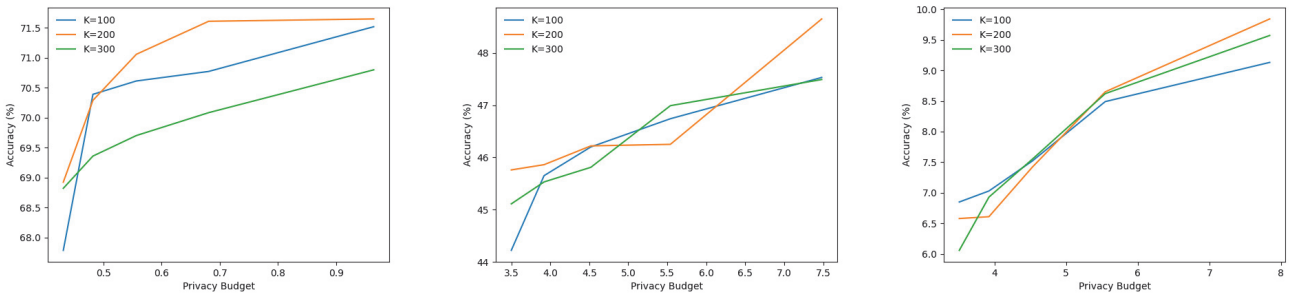
TABLE III: Privacy expenditure (measured by $\hat{\varepsilon}_{CDP}$) across datasets and DP methods, at the end of 200 rounds. Note: the values for CIFAR-10 and CIFAR-100 are the same as the values of n and k are the same in both settings, and both datasets have the same size and are trained for the same duration.

datasets. All other hyperaprameters were kept the same as in the earlier experiment. We then measure the validation accuracy and vary the configurations of C and z to obtain the privacy expenditure. We plot the relationship between the fully trained model's validation accuracy and the privacy budget expended to achieve it in Figure 3. Across all datasets we observe a common trend: the validation accuracy calculated increases as the privacy expenditure ($\hat{\varepsilon}_{CDP}$) does (as s decreases). For the datasets we consider, increasing the value of k does not increase the validation accuracy substantially. This is *not* indicative of a more general trend; one would assume that increasing the number of participants would result in a more accurate model.
Take-away: HDP provides advantageous privacy vs. utility trade-offs in comparison to both CDP and LDP. Additionally, increasing the value of s provides better privacy, which in-turn leads to lower utility.

B. Attacks on Federated Learning

We wish to understand if HFL introduces a new attack surface: through the introduction of adversarial entities at the super-node level. In this subsection, we will discuss (a) the capability of attackers in the status quo, (b) if the success of attacks in the status quo increases in the new hierarchical scheme, and (c) if any new attacks are possible due to the introduction of hierarchies.

Level 0 Adversaries: Here, we consider scenarios where the adversaries are at level 0, and aim to perform data



(a) EMNIST (b) CIFAR-10 (c) CIFAR-100

Fig. 3: We plot validation accuracy as a function of privacy (obtained by varying the value of s) for 3 scenarios using HDP: (i) $k = 100$ (in blue), (ii) $k = 200$ (in orange), and (iii) $k = 300$ (in green), across the three datasets: (a) EMNIST, (b) CIFAR-10, and (c) CIFAR-100. Observe that the utility obtained by HDP improves with increasing privacy budget across all configurations.

Dataset	LDP	HDP	CDP	No DP
EMNIST	0.571	0.599	0.62	0.68
CIFAR-10	0.423	0.452	0.494	0.596
CIFAR-100	0.447	0.46	0.474	0.564

TABLE IV: Efficacy of data reconstruction: reconstruction, measured by LPIPS (higher is better), is least effective when LDP is used for DP training; HDP provides next best resilience.

reconstruction (of a particular client). Recall that in FL, the CA adds the aggregated (client or super-node calculated) gradients to its own weights before propagating new updates for the next round. Based on the generalization proposed in § V, one can observe that the noise addition required to provide DP can occur at (one or all of the) three levels: (a) the clients themselves add DP noise (*i.e.,* LDP), (b) the CA adds noise (*i.e.,* CDP), and (c) the super-nodes add noise (*i.e.,* a HDP scheme). Regardless of which level adds the noise, we assume that noise addition is performed and an adversarial client (at level 0) receives the *noisy update* from the CA, and wishes to use this information to enable data reconstruction. To do so, the client is able to subtract its contribution from the aggregated weight update shared, and run reconstruction attacks using the remaining information (the strategy proposed by [2]). To perform reconstruction, we implement the attack proposed by [39], using the source code presented by the authors for the datasets and models considered in the earlier section. We measure the reconstruction capabilities using the Learned Perceptual Image Patch Similarity (LPIPS) metric [40] (larger values are more realistic). As noted in prior work, reconstruction attacks are largely inefficient for large batch sizes (used for local learning) and large values of k.

To simplify the setup, and highlight the merit of our scheme, we consider a simple setup of $s = 2$ and $k = 4$, and where one of these clients is adversarial and wishes to learn the data of the other, when the batch size used is 1 (and no SA is used). The results are presented in Table IV. Observe that HDP provides better resilience than CDP, but is worse than LDP. This is explained by the privacy guarantees provided by HDP (which, again, lies between LDP and CDP).

Level 1 adversaries: Privacy attacks at the super-node level can be caused by the adversary having complete purview to the gradients from each client in that zone. While we advocate for the usage of SA to alleviate this issue, we discuss the outcomes if such a protocol can not be deployed. In such situations, as denoted by Figure 1, the super-node can have direct access to gradients from individual clients and can perform data reconstruction attacks (to determine membership). We describe what may happen in such scenarios: (1) In the pure LDP setting, clients add noise to the gradients shared upstream. Thus, the super-node adversary has a noisy view of each of the per-client gradient it receives. In such settings, attacks such as the one by [2] will be rendered ineffective (as this attack requires exact gradient knowledge), and data reconstruction attacks become less effective, as these attacks rely on correct gradient information (refer Table IV). (2) In scenarios where clients *do not* add noise, and the super-node is required to perform noise addition (*e.g.,* HDP), data reconstruction attacks are possible if the super-node is adversarial as it has direct purview to the gradients from the clients. However, if SA is utilized, the adversarial super-node is provided an *aggregated* gradient; the efficacy of these attacks is reduced *i.e.,* reconstruction is not of high fidelity (refer to the *No DP* column in Table IV).

To make level 1 adversaries less effective, (a) either client-level noise needs to be added, or (b) an aggregated gradient (via SA) needs to be shared to the super-node. LDP (*i.e.,* client-level noise addition) is a very computationally efficient procedure, and its impacts on utility are well understood (*i.e.,* it induces an unfavorable utility vs. privacy trade-off). Protocols like SA, on the other hand, do not harm the utility of the model (*i.e.,* the utility of the model in the absence of SA is the same with it). However, it greatly increases the communication cost associated with the protocol; each client needs to communicate with all other clients (for each aggregation, in the worst case), leading to communication complexity that is quadratic in the number of clients. We measured the time required to perform SA using the `tff.learning.secure_aggregator` module available

as part of `tensorflow-federated`, and for the models we described earlier. Per round, we needed between 2 and 30 seconds (time increases as a function of model size). These numbers were consistent with the numbers presented earlier.

Level 3 adversaries: If the CA is adversarial, the setting is the same as the level 2 adversary. Both LDP and HDP in conjunction with SA can reduce such an adversary's efficacy of performing data reconstruction. However, there is a strategic advantage of performing SA at the super-node level. In a simple setting, assume the existence of s zones each with m clients, and $s << k$ (where k is the total number of online clients). It is clear to see how the communication cost associated with SA is lower in this setting (it is quadratic in s and not k).

Other Attacks: Collusion of super-nodes (operating at different zones) will not yield a stronger privacy attack as prior work has shown that access to client-level gradients is sufficient to yield good reconstruction (*i.e.,* more information is not necessarily better). Thus, we strongly do not believe this to be a problem. Similarly, for attacks like poisoning in FL, we stress that approaches to safeguard such attacks in the status quo will neatly be compatible with our approach; our primary contribution is a new approach to facilitate noise addition at a different location in FL.

Take-away: In addition to SA, (a) to defeat level 0 adversaries, either LDP or HDP suffices; (b) to defeat level 1 adversaries, LDP suffices; and (c) to defeat level 2 adversaries, LDP or HDP suffices. However, HDP provides better utility than LDP, and is preferred wherever possible.

VIII. Discussion & Open Questions

Influence of k: The privacy accounting in federated learning stems from multiple factors, primary of which is the sampling probability; this determines the number of clients that are online in each federated round. Contrary to our expectation, our experimental results suggest that increasing the value of k does not have a strong effect on the accuracy of the final model learnt. We conjecture that this maybe the case due to the i.i.d distribution of data associated with both the CIFAR-10 and CIFAR-100 datasets. While EMNIST is non-i.i.d, the learning task by itself is too simple to merit substantial utility difference across the three settings we consider. However, a small value for the sampling probability *i.e.,* a small value of k greatly improves the privacy of the model learnt (*i.e.,* small values of ε). We leave performing a more in-depth analysis of the influence of k and ϵ for future work.

Privacy Amplification: New approaches for privacy amplification (such as randomized check-ins [41]) can easily be incorporated with our scheme[11]. Oftentimes, amplification is a byproduct of composing a process that provides randomization with the actual function that needs to be made differentially private. In our scheme, such a randomization effect is observed through secure aggregation (which enables shuffling). In the scenario where the $s = k$, there is no amplification provided as the central aggregator views individual client gradients. However, when $s < k$, observe that the central aggregator is only provided an aggregate view (of all gradients from a particular zone). Determining if other sources of amplification may exist is subject to future work.

Information Leakage: If the number of clients per zone exceeds the number of federated rounds, then in expectation, each client will be elected a super-node only once. A more detailed understanding is needed to ascertain how much information is leaked by exposing a super-node *only once* to aggregate gradient information multiple times.

Run-time: In our work, we measure the privacy vs. utility trade-offs of the proposed hierarchical ecosystem, assuming that elections and secure aggregation are implemented using state-of-the-art approaches, with individual microbenchmarks available in the corresponding works. Microbenchmarks providing run-times of the overall scheme, highlighting time taken for both elections and secure aggregation, as a function of both n and k, will help understand the practicality of the scheme.

Practical Topologies & Data Distribution: In our current work, we prototype the proposal using simple topologies where each super-node is responsible for the same number of clients. One can theoretically show that varying the number of clients per super-node will not influence the privacy guarantees of the approach. However, this will have an impact on the utility of the final model learnt.

Scalability: The introduction of hierarchies into the FL process coupled with DP is not based on artificial assumptions or construction of a system. Such hierarchies already exist in the design and operation of many distributed systems and network topologies used today (DNS, CDNs, ASs, etc.) and is only a consequential step to study how such hierarchies can inter-play with the FL process. Thus, and as shown here, they can be used to improve model performance vs. privacy tradeoff. They can also reduce potentially necessary communication costs such as while executing secured aggregation: small number of clients in each zone can perform this step in a more scalable fashion and allow faster protocol convergence for model update sharing.

Other Attacks: Collusion of super-nodes (operating at different zones) will not yield a stronger privacy attack as prior work has shown that access to client-level gradients is sufficient to yield good reconstruction [25], [3] (*i.e.,* more information is not necessarily better). Similarly, for attacks like poisoning in FL, we stress that approaches to safeguard such attacks in the status quo will neatly be compatible with our approach; our primary contribution is a new approach to facilitate noise addition at a different location in FL.

Comparisons: Shi *et al.* [23] also do not evaluate the resilience of their proposal to practical privacy attackers (such as reconstruction adversaries as we do). They also do not consider the ramifications of their work in settings where DP noise is not added at specific rounds but the gradients are shared with edge-server, providing the edge-server access to unmodified data, whereas our proposal circumvents this by requiring secure aggregation to ensure that the edge-server never sees gradients from clients, but only sees an aggregate view.

[11]Depending on the level applied, we obtain different amplification factor

IX. Conclusion

In our work, we propose an approach for hierarchical FL through super-node election from federated clients. We also propose extensions for how it can be retrofitted to provide DP guarantees. Our experiments suggest that the proposed approach provides more advantageous privacy vs. utility trade-offs compared to approaches in the status quo, while providing resilience to inference adversaries.

X. Acknowledgments

This research was partially supported by the Spanish Ministry of Economic Affairs and Digital Transformation under agreement TSI-063000-2021-63 (MAP-6G), and the EU Horizon 2020 program under grant agreement 101070473 (FLUIDOS). The views and opinions expressed are those of the authors only and do not necessarily reflect those of the funding agencies.

References

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

[2] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Symposion on Security & Privacy*. IEEE, 2019, pp. 691–706.

[3] F. Boenisch, A. Dziedzic, R. Schuster, A. S. Shamsabadi, I. Shumailov, and N. Papernot, "When the curious abandon honesty: Federated learning is not private," *arXiv preprint arXiv:2112.02918*, 2021.

[4] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients–how easy is it to break privacy in federated learning?" *arXiv preprint arXiv:2003.14053*, 2020.

[5] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy." *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[6] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *ACM CCS*, 2016.

[7] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate, "Differentially private empirical risk minimization," *JMLR*, vol. 12, no. Mar, pp. 1069–1109, 2011.

[8] X. Wu, F. Li, A. Kumar, K. Chaudhuri, S. Jha, and J. Naughton, "Bolt-on differential privacy for scalable stochastic gradient descent-based analytics," in *ACM SIGMOD*, 2017.

[9] L. Song and P. Mittal, "Systematic evaluation of privacy risks of machine learning models," in *USENIX*, 2021.

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

[11] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[12] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What can we learn privately?" *SIAM Journal on Computing*, vol. 40, no. 3, pp. 793–826, 2011.

[13] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[14] Y. Zhao, J. Zhao, M. Yang, T. Wang, N. Wang, L. Lyu, D. Niyato, and K.-Y. Lam, "Local differential privacy-based federated learning for internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8836–8853, 2020.

[15] P. Liu, D. Willis, and S. Banerjee, "Paradrop: Enabling lightweight multi-tenancy at the network's extreme edge," in *SEC*. IEEE/ACM, 2016, pp. 1–13.

[16] S. Marti and H. Garcia-Molina, "Taxonomy of trust: Categorizing p2p reputation systems," *Computer Networks*, vol. 50, no. 4, pp. 472–484, 2006.

[17] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *ACM CCS*, 2015, pp. 1310–1321.

[18] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private language models without losing accuracy," *arXiv preprint arXiv:1710.06963*, 2017.

[19] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for federated learning on user-held data," *arXiv preprint arXiv:1611.04482*, 2016.

[20] ——, "Practical secure aggregation for privacy-preserving machine learning," in *ACM CCS*, 2017, pp. 1175–1191.

[21] M. S. H. Abad, E. Ozfatura, D. Gunduz, and O. Ercetin, "Hierarchical federated learning across heterogeneous cellular networks," in *ICASSP*. IEEE, 2020.

[22] J. Yuan, M. Xu, X. Ma, A. Zhou, X. Liu, and S. Wang, "Hierarchical federated learning through lan-wan orchestration," *arXiv preprint arXiv:2010.11612*, 2020.

[23] L. Shi, J. Shu, W. Zhang, and Y. Liu, "Hfl-dp: Hierarchical federated learning with differential privacy," in *GLOBECOM*. IEEE, 2021, pp. 1–7.

[24] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," *arXiv preprint arXiv:2104.07586*, 2021.

[25] D. Pasquini, D. Francati, and G. Ateniese, "Eluding secure aggregation in federated learning via model inconsistency," *arXiv preprint arXiv:2111.07380*, 2021.

[26] J. Petrek and V. Sledt, "A large hierarchical network star—star topology design algorithm," *European transactions on telecommunications*, vol. 12, no. 6, pp. 511–522, 2001.

[27] G. G. Gueta, I. Abraham, S. Grossman, D. Malkhi, B. Pinkas, M. Reiter, D.-A. Seredinschi, O. Tamir, and A. Tomescu, "Sbft: a scalable and decentralized trust infrastructure," in *IEEE/IFIP DSN*, 2019, pp. 568–580.

[28] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network schemes," *IEEE Communications Surveys & Tutorials*, vol. 7, no. 2, pp. 72–93, 2005.

[29] A. M. Mahdy, J. S. Deogun, and J. Wang, "A dynamic approach for the selection of super peers in ad hoc networks," in *IEEE ICN*, 2007.

[30] L. Qing, F. Xuan-li, H. Yu-ke, and H. Wan-jie, "Super-peer selection algorithm based on ahp in mobile peer-to-peer network," in *7th International Conference on Information Technology: IoT and Smart City*, 2019, p. 305–309.

[31] C. A. Choquette-Choo, N. Dullerud, A. Dziedzic, Y. Zhang, S. Jha, N. Papernot, and X. Wang, "Capc learning: Confidential and private collaborative learning," *arXiv preprint arXiv:2102.05188*, 2021.

[32] F. S. Gharehchopogh and H. Arjang, "A survey and taxonomy of leader election algorithms in distributed systems," *Indian Journal of Science and Technology*, vol. 7, no. 6, p. 815, 2014.

[33] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Transactions on Computer Systems*, vol. 20, no. 4, pp. 398–461, 2002.

[34] A. Bittau, Ú. Erlingsson, P. Maniatis, I. Mironov, A. Raghunathan, D. Lie, M. Rudominer, U. Kode, J. Tinnes, and B. Seefeld, "Prochlo: Strong privacy for analytics in the crowd," in *26th Symposium on Operating Systems Principles*, 2017, pp. 441–459.

[35] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta, "Amplification by shuffling: From local to central differential privacy via anonymity," in *ACM-SIAM Symposium on Discrete Algorithms*, 2019, pp. 2468–2479.

[36] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *IJCNN*. IEEE, 2017, pp. 2921–2926.

[37] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[38] ——, "Cifar-100 (canadian institute for advanced research)." [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html

[39] L. Zhu and S. Han, "Deep leakage from gradients," in *Federated Learning*. Springer, 2020, pp. 17–31.

[40] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *IEEE CVPR*, 2018, pp. 586–595.

[41] B. Balle, P. Kairouz, H. B. McMahan, O. Thakkar, and A. Thakurta, "Privacy amplification via random check-ins," *arXiv preprint arXiv:2007.06605*, 2020.