Secure Gradient Aggregation With Sparsification for Resource-Limited Federated Learning

Hasin Us Sami, *Graduate Student Member, IEEE*, and Başak Güler, *Member, IEEE*

***Abstract*— Secure aggregation is an information-theoretic mechanism for gradient aggregation in federated learning, to aggregate the local user gradients without revealing them in the clear. In this work, we study secure aggregation under gradient sparsification constraints, for resource-limited wireless networks, where only a small fraction of local parameters are aggregated from each user during training (as opposed to the full gradient). We first identify the vulnerabilities of conventional secure aggregation mechanisms under gradient sparsification. We show that conventional mechanisms can reveal sensitive user data when aggregating sparsified gradients, due to the auxiliary coordinate information shared during sparsification, even when the individual gradients are not disclosed in the clear. We then propose TinySecAgg, a novel coordinate-hiding sparsified secure aggregation mechanism to address this challenge, under formal information-theoretic privacy guarantees. Our framework reduces the communication overhead of conventional secure aggregation baselines by an order of magnitude (up to $22.5\times$) without compromising model accuracy.**

***Index Terms*— Federated learning, secure aggregation, gradient sparsification, gradient inversion, distributed learning.**

I. Introduction

FEDERATED learning (FL) is a popular paradigm for distributed training, where data-owners (users) perform training on locally collected datasets, after which the local updates (e.g., local gradients) are aggregated by a server to form a global model [1]. While popular in a variety of privacy-sensitive applications (such as healthcare) due to its on-device training architecture (data never leaves the device), FL can still reveal sensitive information about the local data samples through what is known as gradient inversion attacks [2], [3], [4], [5], [6], [7].

Secure aggregation (SA) is an information-theoretic mechanism to address this challenge without compromising model accuracy [8]. To do so, SA utilizes secret sharing principles from secure multi-party computing, where each user encodes its local gradient with additive random masks to hide its true value, and only shares the encoded gradient with the server. The additive masks cancel out upon aggregation at the server, allowing the server to correctly recover the sum of the local gradients, but without learning any further information about the individual gradients. In doing so, SA prevents the server from associating the aggregated gradients with any particular user, enhancing resilience against inversion attacks as the number of users increases [9], [10], [11]. While popular in enhancing user privacy in distributed settings, communication overhead is still a major challenge in SA, which can hinder scalability to larger networks.

Gradient sparsification is a widely adopted technique to reduce the communication overhead in FL. In this setting, each user shares only a small portion (K) of their local gradient parameters with the server (as opposed to sending the entire gradient), along with their coordinates. The parameters are selected uniformly random (rand-K), or based on their magnitude (top-K) [12], [13], [14], [15], [16], [17], [18]. The server then aggregates the received parameters using the received coordinates, where parameters from different users are aggregated if their coordinates match, to update the global model for the next training round. The two sparsification mechanisms (rand-K/top-K) provide complementary benefits; the latter enables faster convergence, while the former is more memory and compute-efficient, as the random coordinates can be generated in advance [15], [17], [19], [20], [21], [22].

In this work, we study gradient sparsification in the context of SA. We first identify the potential vulnerabilities associated with sharing coordinate information during sparsification. In particular, we show that the local data samples can be recovered from the aggregate of the gradient parameters using the coordinates shared over multiple training rounds, even if the gradients are securely aggregated (using SA) at each training round. We illustrate this phenomenon in Fig. 1, where we demonstrate the reconstruction performance of our attack, the details of which will be provided in Section IV. After a sufficient number of training iterations, the server can reconstruct the local data samples, even when only 1% of the gradient parameters are aggregated (using SA) from each user. We then propose a coordinate-hiding SA framework to address this challenge, which hides not only the gradient parameters but also their coordinates during aggregation. Our framework enables the server to aggregate the sparsified local gradients, but without learning any information about the gradient parameters (beyond their aggregate) or their coordinates, under formal information-theoretic privacy guarantees.

Manuscript received 17 November 2023; revised 10 April 2024; accepted 11 May 2024. Date of publication 20 May 2024; date of current version 14 November 2024. This research was sponsored in part by the OUSD (R&E)/RT&L under Cooperative Agreement Number W911NF-20-2-0267, NSF CAREER Award CCF-2144927, and the UCR OASIS Fellowship. The associate editor coordinating the review of this article and approving it for publication was J. Zhang. *(Corresponding author: Başak Güler.)*

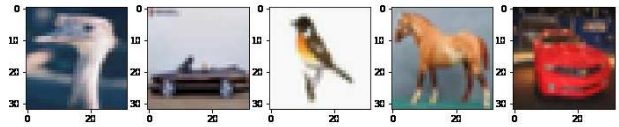
The authors are with the Department of Electrical and Computer Engineering, University of California at Riverside, Riverside, CA 92521 USA (e-mail: hsami003@ucr.edu; bguler@ece.ucr.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2024.3403475.

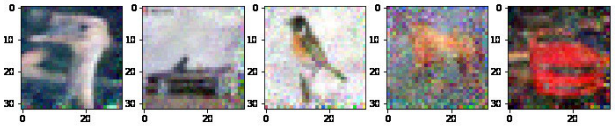
Digital Object Identifier 10.1109/TCOMM.2024.3403475

0090-6778 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

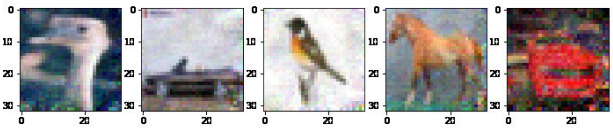
Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.



(a) Original.



(b) Recovered (top-K).



(c) Recovered (rand-K).

Fig. 1. Image reconstruction from the aggregate of sparsified local gradients. At each training round, only 1% of the local parameters are aggregated from each user. The local parameters are aggregated securely using SA at each training round, without revealing the individual parameters to the server.

Our framework builds on an offline-online trade-off, where we offload the communication-intensive operations, such as randomness generation, to a data-independent offline phase. The offline phase can take place in advance when the network load is low, or can be parallelized with other components of training. The online phase depends on the local datasets, hence should be carried out during training. We then propose an efficient SA mechanism for the online phase, to aggregate the sparsified gradients in FL, while revealing no information about the individual parameters or their coordinates (beyond their aggregated information).

Our theoretical analysis presents the formal information-theoretic privacy analysis, and provides the key trade-offs between adversary tolerance and communication/computation complexity, while our experiments demonstrate up to $22.5\times$ reduction in the communication overhead compared to SA baselines, without compromising accuracy. Our contributions can be summarized as follows:

1) We identify the vulnerabilities of sharing coordinate information in SA, and the necessity of stronger, coordinate-hiding privacy notions to enhance adversary-resilience under sparsification constraints.
2) We demonstrate successful reconstruction of local data samples from the aggregate of sparsified gradients, by utilizing only the knowledge of the rand-K/top-K coordinates of the users.
3) We propose a coordinate-hiding sparsified SA framework, TinySecAgg, to enable gradient sparsification for enhancing the scalability of SA in large model settings, but without disclosing any information about the gradient parameters or their coordinates, with formal information-theoretic privacy guarantees.
4) Through extensive experiments, we demonstrate that our framework cuts the communication cost by an order of magnitude ($22.5\times$) over conventional SA mechanisms.

II. Related Works

Gradient inversion attacks aim to recover the sensitive data by inverting the local gradients shared during training. To that end, [2] optimizes the input data to minimize the Euclidean distance to the true gradients, whereas [3] leverages a relationship between the ground-truth labels and signs of the corresponding gradients. Cosine similarity is proposed in [4] for image reconstruction, while [5] demonstrates reconstruction from a large batch of images. Another line of work considers image reconstruction by leveraging user dropouts [23] or by maliciously influencing the model parameters/architecture [6], [7], [24]. Reference [25] demonstrates inversion attacks from compressed gradients (without SA).

Secure aggregation (SA) protocols are introduced to prevent the server from gaining access to the local gradients during FL, where the local gradient of each user is protected by leveraging secure multi-party computing tools [8], [9], [10], [11], [26], [27]. Recent works [28], [29] propose rand-K/top-K sparsification techniques compatible with SA. On the other hand, in these settings the server gathers knowledge about the gradient coordinates for each user, which are vulnerable to the multi-round gradient inversion attacks introduced in Section IV. A key advantage of SA is their compatibility with complementary privacy-enhancing mechanisms such as differential privacy (DP), which can further benefit the latter by reducing the amount of DP noise that should be added to the gradients in distributed settings [30], [31], [32]. In doing so, most SA frameworks are developed under the honest-but-curious adversary model, where adversaries follow the protocol but try to gain additional information about the sensitive datasets of honest users, using the messages exchanged during protocol execution. Recent works also study SA under Byzantine adversary settings [33], [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], where the adversaries poison or modify the messages exchanged during the protocol. Although beyond our current scope, we note extending our framework to malicious adversaries as an interesting future direction.

Due to its wireless nature, communication bottleneck is a major challenge in FL. In practice, the communication overhead is handled through gradient compression techniques. Gradient sparsification allows users to send K out of d gradient parameters ($K \ll d$) selected uniformly random (rand-K) or with respect to the highest magnitude (top-K) [12], [13], [15], [16], [17], [20], [22]. Recent works have also explored the combination of two to achieve the best of both worlds [18], [45]. A complementary compression mechanism is gradient quantization, where each gradient parameter is represented with fewer number of bits to adapt to the available communication resources [14], [21], [46]. Although our focus in this work is on gradient sparsification, combining our techniques with complementary quantization approaches is also an interesting future direction.

Another related line of work is the coordinate-hiding mechanisms for Private Information Retrieval (PIR) [47], [48], [49]. These mechanisms build on a multi-server training architecture, where an encoded version of the training model is stored at multiple non-colluding servers. The goal is to enable users

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

to decode the model, perform local training, and then update the model at each server, but without revealing the true updates or the models to the servers. References [47], [48] consider the privacy leakage as a result of sharing the coordinate information of the sparse model update/gradient parameters by the users under gradient sparsification. To that end, after local training, each user transmits the encoded sparse model updates and corresponding encoded/permuted coordinates to multiple non-colluding servers. Then, each server updates its stored encoded model at the correct coordinates, but without having access to the true updates or coordinates of the user. Reference [49] allows collusions across T colluding servers, however, servers cannot collude with the users. Different from these works, in our problem the server can collude with any set of up to T users. Additionally, our focus is on a single-server FL setup, which brings an additional challenge for privacy as all encoded information has to be collected by a single server.

III. Problem Formulation

We consider a centralized FL architecture with N users, coordinated by a central server. User $i \in [N]$ holds a local dataset $\mathcal{D}_i$ with $D_i \triangleq |\mathcal{D}_i|$ samples. The goal is to train a global model $\mathbf{w} \in \mathbb{R}^d$ to minimize the global loss,

$$\min_{\mathbf{w}} F(\mathbf{w}) \triangleq \sum_{i=1}^{N} F_i(\mathbf{w}) \tag{1}$$

where $F_i(\mathbf{w})$ denotes the local loss of user i. Training is performed iteratively through global and local training rounds. At the beginning of each global round t, the server sends the current state of the global model $\mathbf{w}^t$ to the users. User $i \in [N]$ generates a local model $\mathbf{w}_i^t \leftarrow \mathbf{w}^t$, which is updated locally through E local training rounds,

$$\mathbf{w}_i^t \leftarrow \mathbf{w}_i^t - \eta \nabla F_i(\mathbf{w}_i^t) \tag{2}$$

where $\nabla F_i(\mathbf{w}_i^t)$ is the gradient evaluated on $\mathcal{D}_i$, and η is the learning rate. After E local training rounds, user i sends the (cumulative) local gradient,

$$\Delta_i^t \triangleq \mathbf{w}_i^t - \mathbf{w}^t \in \mathbb{R}^d \tag{3}$$

to the server, who then aggregates the received gradients to update the global model for the next training round,

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \frac{1}{|\mathcal{U}(t)|} \sum_{i \in \mathcal{U}(t)} \Delta_i^t \tag{4}$$

where $\mathcal{U}(t)$ denotes the set of surviving users who succeed in sending their local updates to the server at round t, as some users may drop out from the protocol due to poor wireless connectivity, or device unavailability.

Gradient sparsification is a popular compression mechanism to improve the communication efficiency in FL, where, instead of sending the entire gradient Δ_i^t to the server, each user sends only $K \ll d$ selected parameters. Sparsification typically involves a process known as *error-accumulation*, to track the cumulative error resulting from the parameters that have not been sent in previous rounds. Accordingly, the sparsification operation is given by,

$$\mathbf{x}_i^t \triangleq \mathbf{b}_i^t \odot \widetilde{\Delta}_i^t \tag{5}$$

where $\odot$ is the Hadamard product, $\mathbf{x}_i^t$ denotes the sparsified local gradient, $\mathbf{b}_i^t \in \{0,1\}^d$ is a binary mask holding the coordinates of the K parameters selected by user i, where the ℓ^{th} element is given by,

$$\mathbf{b}_i^t(\ell) \triangleq \begin{cases} 1 & \text{if user } i \text{ selects coordinate } \ell \text{ at round } t \\ 0 & \text{otherwise} \end{cases}$$

such that $\|\mathbf{b}_i^t\|_1 = K$ where $\|.\|_1$ denotes the L_1 norm, and

$$\widetilde{\Delta}_i^t \triangleq \mathbf{w}_i^t - \mathbf{w}^t + \mathbf{e}_i^{t-1} = \Delta_i^t + \mathbf{e}_i^{t-1}, \tag{6}$$

where $\mathbf{e}_i^t$ denotes the error accumulated at round t,

$$\mathbf{e}_i^t \triangleq \Delta_i^t + \mathbf{e}_i^{t-1} - \mathbf{x}_i^t = \widetilde{\Delta}_i^t - \mathbf{x}_i^t. \tag{7}$$

After constructing the sparsified local gradient $\mathbf{x}_i^t$ from (5), user i then sends the selected K parameters from $\mathbf{x}_i^t$ to the server, along with the coordinates of the selected parameters. Using the received coordinates, the server aggregates the sparsified local gradients $\mathbf{x}_i^t$ to update the global model,

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \frac{1}{|\mathcal{U}(t)|} \sum_{i \in \mathcal{U}(t)} \mathbf{x}_i^t \tag{8}$$

for the next training round. The specific structure of the binary mask $\mathbf{b}_i^t$ depends on the sparsification methodology:

A. Rand-K Sparsification

In this setting, each user selects K parameters uniformly at random (without replacement) from $\widetilde{\Delta}_i^t$, and $\mathbf{b}_i^t$ is a random binary vector where $\|\mathbf{b}_i^t\|_1 = K$, generated independently for each user $i \in [N]$.

B. Top-K Sparsification

In this setting, users send only the top K parameters with the highest magnitude to the server, and $\mathbf{b}_i^t \in \{0,1\}^d$ is a binary vector indicating the coordinates of the top K parameters from $\widetilde{\Delta}_i^t$ with the highest magnitude.

The two mechanisms (rand-K/top-K) have complementary benefits. Rand-K is more memory and communication efficient; as the binary masks $\mathbf{b}_i^t$ are sampled independently and uniformly at random, they can be generated offline in advance when the network load is low. Top-K can speed up convergence, but the coordinates depend on the gradient magnitudes, which has to be sent online during training.

1) Threat Model: In this work, our focus is on *honest-but-curious* adversaries (as is the most common threat model in SA), where adversaries do not poison the datasets, but try to reveal additional information about the local datasets of honest users using the information exchanged during protocol execution. Similar to [23], the server can freeze the global model parameters (albeit do not change them). Out of N users, up to $T < N$ users are adversarial, who may collude with each other and/or the server. The set of honest and adversarial users are denoted by $\mathcal{H}$ and $\mathcal{T} = [N] \backslash \mathcal{H}$, respectively.

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

2) Secure Aggregation : The SA problem aims at aggregating the local updates $\mathbf{x}_i^t$,

$$\mathbf{x}_{agg}^t \triangleq \sum_{i\in\mathcal{U}(t)} \mathbf{x}_i^t \tag{9}$$

but without revealing any information about the individual updates (beyond their sum). Formally, this can be stated by a mutual information condition,

$$I(\{\mathbf{x}_i^t\}_{i\in\mathcal{H}}; \mathcal{M}_{\mathcal{T}}^t | \mathbf{x}_{agg}^t, \{\mathbf{x}_i^t\}_{i\in\mathcal{T}}, \mathcal{R}_{\mathcal{T}}^t) = 0 \tag{10}$$

where $\mathcal{M}_{\mathcal{T}}^t$ denotes the set of all messages received, and $\mathcal{R}_{\mathcal{T}}^t$ denotes the randomness generated, by the adversaries and the server at round t. The correct recovery of the aggregate in (9) can be formalized by an entropy constraint,

$$H(\mathbf{x}_{agg}^t | \mathcal{M}_{\mathcal{U}(t)}^t) = 0 \tag{11}$$

where $\mathcal{M}_{\mathcal{U}(t)}^t$ denotes the set of all messages held by the surviving users $\mathcal{U}(t)$ at round t. To compute (9) under the information-theoretic privacy guarantees from (10), SA protocols enable users to *encode* their local updates $\mathbf{x}_i^t$ by using locally generated random secret masks, and send only an encoded version to the server. The encoding process hides the true value of the local updates from the server, while allowing the server to decode their sum as in (9), without learning any information about the individual updates $\mathbf{x}_i^t$. In doing so, SA protocols differ in their encoding/decoding mechanism.

A common challenge in conventional SA protocols is the communication overhead with large models, as the dimensionality of the encoded gradient sent from each user is as large as the true gradient, which prevents scalability to larger models in practice. Our goal in this work is to address this challenge, where we ask the question,

- Can gradient sparsification enhance the scalability of secure aggregation?

In this work, we answer this question in the affirmative. Sparsification can enhance the communication-efficiency of SA, but additional care should be taken to hide the coordinates, and naive approaches can do more harm than good. Specifically, as we demonstrate in the following section, the coordinate information exchanged during gradient sparsification can introduce new vulnerabilities to SA. This is due to the fact that the coordinates of sparsified local parameters vary across the users throughout the training, hence, by using the coordinates shared over multiple rounds, adversaries can reconstruct the local gradients of individual users from their sum, even when the gradients are aggregated using SA.

Our results indicate the necessity of stronger guarantees for gradient sparsification in SA, for hiding not only the local parameters, but also their coordinates. To address this challenge, we then introduce TinySecAgg, a coordinate-hiding sparse SA framework. Our framework hides both the sparsified gradient parameters and their coordinates from the server, under formal information-theoretic privacy guarantees, while significantly enhancing the communication efficiency of SA.

IV. Reconstruction From Compressed Gradients

In this section, we discuss the naive application of gradient sparsification with SA, to present the associated risks. Gradient sparsification, as described in Section III, is compatible with most well-known SA protocols [8], [9], where the key premise is to learn the sum $\mathbf{x}_{agg}^t = \sum_{i\in\mathcal{U}(t)} \mathbf{x}_i^t$ from (9), without disclosing the local updates $\mathbf{x}_i^t$. In doing so, users hide their selected parameters with additive random masks, and send the encoded parameters (and their coordinates) to the server. The additive masks are constructed in a way that they cancel out upon aggregation at the server, thus allowing the server to learn the sum $\mathbf{x}_{agg}^t$ of the local gradients, but without revealing any further information about the individual gradients $\mathbf{x}_i^t$. Though our attack in the following does not depend on the specific nature of the encoding/decoding mechanism, in App. A we also review the details of the encoding/decoding mechanisms.

In the following, we consider the frozen-model setup from [23], where the server freezes the model, as a result the local gradients Δ_i^t are stable throughout the iterations. This setting can also emerge when the model is close to convergence. We next demonstrate a gradient reconstruction mechanism that allows the server to recover the individual gradients Δ_i^t for all $i \in [N]$, using only the sum of the sparsified gradients $\mathbf{x}_{agg}^t$ and their coordinates over multiple training rounds.

Let $\tau_i^t(\ell) < t$ denote the most recent training round (prior to round t) in which user i has shared coordinate ℓ with the server. From the error accumulated up to round t, one can then rewrite (6) as follows,

$$\widetilde{\Delta}_i^t(\ell) = (t - \tau_i^t(\ell))\Delta_i(\ell) \tag{12}$$

where $\Delta_i(\ell)$ is the ℓ^{th} element of the gradient[1] Δ_i of user i as defined in (3), using which the sparsified gradient from (5) can be written as,

$$\mathbf{x}_i^t(\ell) = \mathbf{b}_i^t(\ell)(t - \tau_i^t(\ell))\Delta_i(\ell) \tag{13}$$

where $\mathbf{x}_i^t(\ell)$ denotes the ℓ^{th} element of $\mathbf{x}_i^t$.

After SA, the server learns the aggregate of the sparsified gradients for each coordinate $\ell \in [d]$,

$$\mathbf{x}_{agg}^t(\ell) = \sum_{i\in\mathcal{U}(t)} \mathbf{x}_i^t(\ell) \tag{14}$$

$$= \sum_{i\in\mathcal{U}(t)} \mathbf{b}_i^t(\ell)(t - \tau_i^t(\ell))\Delta_i(\ell) \tag{15}$$

From (15), along with the selected coordinates $\mathbf{b}_i^t$ from users $i \in \mathcal{U}(t)$, the server can construct the following,

$$\mathbf{A}\begin{bmatrix}\Delta_1(\ell)\\ \vdots \\ \Delta_N(\ell)\end{bmatrix} + \mathbf{n} = \begin{bmatrix}\mathbf{x}_{agg}^1(\ell)\\ \vdots \\ \mathbf{x}_{agg}^J(\ell)\end{bmatrix} \quad \forall \ell \in [d], \tag{16}$$

where J is the total number of training rounds, $\mathbf{n}$ denotes the noise incurred due to local training across the rounds, and $\mathbf{A}$ is a $J \times N$ matrix defined as,

$$\mathbf{A} \triangleq \begin{bmatrix}\mathbf{b}_1^1(\ell)(1 - \tau_1^1(\ell)) & \cdots & \mathbf{b}_N^1(\ell)(1 - \tau_N^1(\ell))\\ \vdots & \vdots & \vdots \\ \mathbf{b}_1^J(\ell)(J - \tau_1^J(\ell)) & \cdots & \mathbf{b}_N^J(\ell)(J - \tau_N^J(\ell))\end{bmatrix} \tag{17}$$

by letting $\mathbf{b}_i^t(\ell) = 0$ for the dropout users $i \in [N]\backslash\mathcal{U}(t)$ without loss of generality. By using $\mathbf{A}$ and the aggregate of the

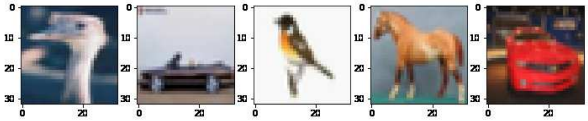
[1]One can omit the time index t from the local gradient Δ_i^t as the local gradients are stable throughout the iterations.

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

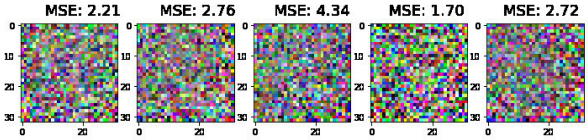
Algorithm 1 Reconstruction From Sparsified Gradients

Input: Number of users N, sparsification ratio $\frac{K}{d}$, global model $\mathbf{w}$, model size d
Output: Local gradients $\Delta_i \; \forall i \in [N]$

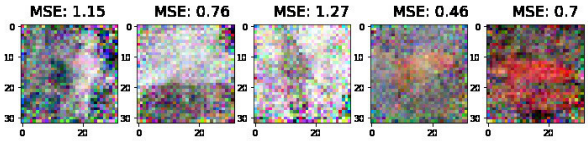
for round $t = 1, 2, \ldots, J$ **do**
  for user $i = 1, 2, \ldots, N$ in parallel **do**
    Compute the error-accumulated local gradient $\widetilde{\Delta}_i^t$ ▷ Equation (12)
    Compute the sparsified local gradient $\mathbf{x}_i^t$ ▷ Equation (13)
    Aggregate the sparsified local gradients using SA.
  end for
  Server:
  Recover the aggregate $\mathbf{x}_{agg}^t$ of the sparsified gradients $\mathbf{x}_i^t$ ▷ Equation (15)
  Send the global model $\mathbf{w}$ to the users
end for
Server:
Construct (16) using the sparse gradient aggregate $\mathbf{x}_{agg}^t$ and the selected coordinates over J training rounds ▷ Equation (16)
Recover the local gradients Δ_i by solving least-square problem, $\Delta^*(\ell) = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\left[\mathbf{x}_{agg}^1(\ell) \; \cdots \; \mathbf{x}_{agg}^J(\ell)\right]^{\mathrm{T}} \; \forall \; \ell \in [d]$



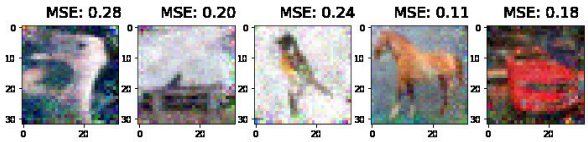
(a) Original images.



(b) Recovered images after 1 iteration.



(c) Recovered images after 200 iterations.



(d) Recovered images after 500 iterations.

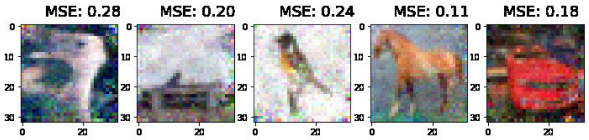
Fig. 2. (rand-K) Image reconstruction quality with varying number of training rounds. The server utilizes the coordinates to recover the local gradients.

sparsified gradients $\{\mathbf{x}_{agg}^t\}_{t\in[J]}$, the server can finally recover the local gradients $\Delta^*(\ell) \triangleq \left[\Delta_1(\ell) \; \cdots \; \Delta_N(\ell)\right]^{\mathrm{T}}$ for each coordinate $\ell \in [d]$, by solving a least squares problem,

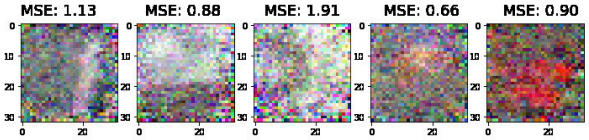
$$\Delta^*(\ell) = (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\left[\mathbf{x}_{agg}^1(\ell) \; \cdots \; \mathbf{x}_{agg}^J(\ell)\right]^{\mathrm{T}}$$

Upon recovering the local gradients $\{\Delta^*(\ell)\}_{\ell\in[d]}$, the server can apply any gradient inversion attack (e.g., [4]) to reveal the local data samples from the local gradients. The pseudocode of our attack is presented in Alg. 1.

In Figs. 2-6, we present the image reconstruction quality on a ResNet-18 model [50] trained on the CIFAR-10 dataset

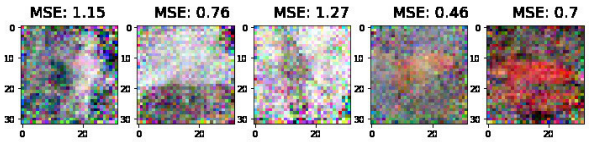


(a) Our attack.

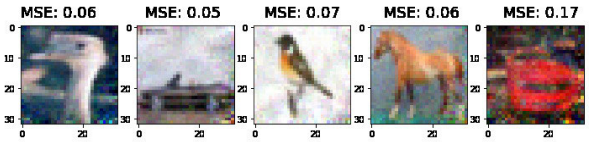


(b) Conventional multi-round attack.

Fig. 3. (rand-K) Recovered images for our attack vs. conventional multi-round attack (after 500 rounds).



(a) Recovered images with sparsification $K = 0.01d$.



(b) Recovered images with sparsification $K = 0.1d$.

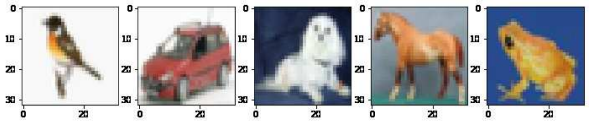
Fig. 4. (rand-K) Image reconstruction for varying sparsification level K (200 rounds).

across 5 users [51], where each user holds a single random data sample in accordance with [3], [4], and [23]. The selected parameters are aggregated using the SA protocol SecAgg from [8], while we note that our results are indifferent to the specific SA protocol used (as the final aggregated gradient is the same). After reconstructing the local gradients, gradient inversion from [4] is applied to recover the images. The reconstruction quality is measured using the mean square error (MSE) between the recovered and original image.

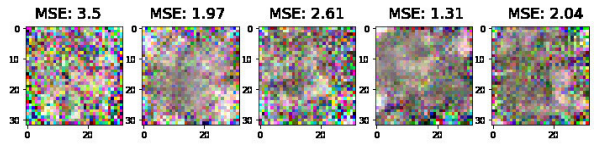
Fig. 2 demonstrates the recovered images for rand-K sparsification, with a sparsification ratio of $K = 0.01d$, i.e., only 1% of the gradient parameters are aggregated from each user. We observe that the quality of the recovered images approaches the original images as the number of training rounds increases. With increasing number of rounds, the server obtains a sufficient number of linearly independent equations in (16) for a larger number of gradient coordinates, leading to the recovery of a larger fraction of the local gradients with increased accuracy.

In Fig. 3, we further present the reconstruction performance for our attack compared to the conventional multi-round attack from [27], which leverages user participation information over multiple training rounds with a frozen global model to recover the local gradients, and can be applied to our problem by leveraging the coordinate information. We observe that the proposed attack significantly enhances the reconstruction performance over the conventional multi-round attack. This is due to the fact that the conventional multi-round attack is agnostic to error accumulation during sparsification (i.e., $t-\tau_i^t(\ell) = 1$),

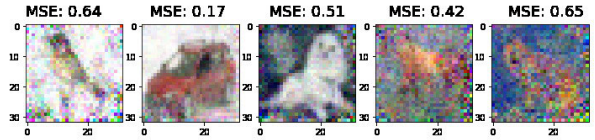
Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.



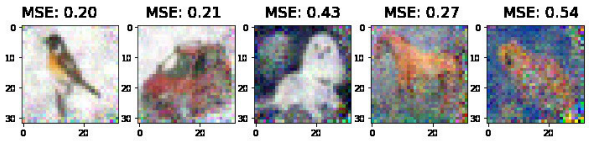
(a) Original images



(b) Recovered images after 1 iteration.

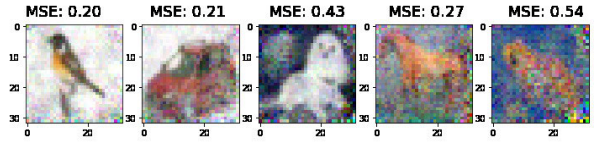


(c) Recovered images after 200 iterations.

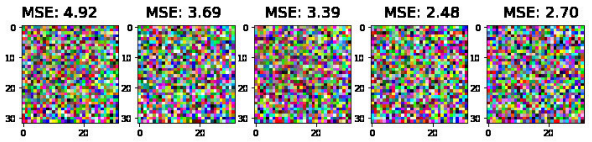


(d) Recovered images after 500 iterations.

Fig. 5. (top-K) Image reconstruction quality with respect to the number of training rounds. The server utilizes the coordinates to recover the local gradients.



(a) Our attack.



(b) Conventional multi-round attack.

Fig. 6. (top-K) Recovered images for our attack vs. conventional multi-round attack (after 500 rounds).

which hinders the reconstruction performance as observed in Fig. 3(b), as error accumulation causes the effective gradient from any given user to change over time, at any given coordinate. In Fig. 4, we present the reconstruction quality for varying levels of sparsification for rand-K. We observe that as K increases, reconstruction quality increases.

In Fig. 5, we demonstrate the reconstruction quality for top-K sparsification. At each training round, each user selects the top $K = 0.01d$ local parameters with the highest magnitude. The selected parameters are then aggregated via SA [8]. We observe that the server can recover the local gradients with higher accuracy as the number of training rounds increases, which increases the quality of the images reconstructed from the local gradients. In Fig. 6, we further illustrate the reconstruction quality of our attack compared to the conventional multi-round attack from [27]. We observe that our attack significantly improves the attack performance by incorporating error accumulation during reconstruction. In App. B, we further demonstrate the attack performance for different types of data distributions across the users.

Our key observation is that the attack can only be launched when coordinate information is available. When sufficient coordinate information is not available, reconstruction is unsuccessful, as observed in Figs. 2(b) and 5(b). Motivated by these findings, in the following, we introduce a coordinate-hiding SA mechanism, to enhance the security of SA under gradient sparsification. Our goal is to ensure the information-theoretic privacy for both the selected gradient parameters and their coordinates, which can be formalized as,

$$I(\{\mathbf{x}_i^t, \mathcal{K}_i^t\}_{i\in\mathcal{H}}; \mathcal{M}_{\mathcal{T}}^t | \mathbf{x}_{agg}^t, \{\mathbf{x}_i^t, \mathcal{K}_i^t\}_{i\in\mathcal{T}}, \mathcal{R}_{\mathcal{T}}^t) = 0 \qquad (18)$$

where $\mathcal{K}_i^t$ denotes the set of coordinates selected by user i, $\mathbf{x}_i^t$ denotes the sparsified local gradient of user i, $\mathcal{M}_{\mathcal{T}}^t$ denotes the set of all messages received by the adversaries and the server, $\mathbf{x}_{agg}^t$ denotes the gradient aggregate from (9), and $\mathcal{R}_{\mathcal{T}}^t$ denotes the set of all randomness generated by the adversaries. Our framework hides both the individual gradient parameters and coordinates received from the users during aggregation, while ensuring formal information-theoretic privacy guarantees for both the parameters and coordinates as presented in (18).

V. The TinySecAgg Framework

This section presents TinySecAgg, a coordinate-hiding rand-K gradient sparsification framework to prevent coordinate-based reconstruction attacks for SA. Our mechanism hides not only the selected gradient parameters, but also their *coordinates*, preventing the server from using the coordinate information to recover the local gradients, while ensuring the correct recovery of the aggregated gradients at each coordinate.

To do so, our framework builds on an offline-online trade-off, where we offload the communication-intensive operations, such as randomness generation, to a data-independent offline phase, which can take place in advance when the network load is low. In TinySecAgg, users only communicate an encoded version of the selected gradient parameters and their coordinates, instead of sending raw parameters and coordinates directly. At the end, the server decodes the correct aggregate of the gradient parameters for each coordinate, but without learning the parameters *or their coordinates*. As in most SA mechanisms, all operations are carried out in a finite field $\mathbb{F}_p$ of integers modulo a large prime p. We next describe the details of the offline and online phases.

A. Offline Phase

Initially, users define a one-hot vector $\mathbf{a}_k \in \{0,1\}^d$ for each $k \in [d]$, where only the k^{th} element is equal to 1, and all other elements are 0, and then partition $\mathbf{a}_k$ into M equal-sized shards,

$$\mathbf{a}_k = \begin{bmatrix} \mathbf{a}_{k1}^{\mathrm{T}} & \dots & \mathbf{a}_{kM}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}. \qquad (19)$$

As will be detailed in our theoretical analysis, parameter M controls a key trade-off between communication-efficiency and

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

adversary tolerance, in particular, selecting a larger M reduces the communication overhead, but also the adversary tolerance.

Users then agree on $N+M+T$ distinct public parameters $\{\alpha_i\}_{i\in[N]}$, $\{\beta_n\}_{n\in[M+T]}$ from $\mathbb{F}_p$. Then, each user $i\in[N]$ generates a random binary mask $\mathbf{b}_i^t\in\{0,1\}^d$ for rand-K sparsification, where K out of d elements are set to 1 uniformly at random (without replacement). Let $\mathcal{K}_i^t\triangleq\{\ell:\mathbf{b}_i^t(\ell)=1\}$ denote the (ordered) set of the K coordinates selected by user i. Then, user i generates two Lagrange polynomials,

$$\phi_{ik}(\alpha)\triangleq\sum_{n\in[M]}\mathbf{a}_{\mathcal{K}_i^t(k),n}\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha-\beta_{n'}}{\beta_n-\beta_{n'}}+\sum_{n=M+1}^{M+T}\mathbf{v}_{ikn}^t\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha-\beta_{n'}}{\beta_n-\beta_{n'}},\tag{20}$$

$$\psi_{ik}(\alpha)\triangleq\sum_{n\in[M]}\mathbf{a}_{\mathcal{K}_i^t(k),n}r_{ik}^t\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha-\beta_{n'}}{\beta_n-\beta_{n'}}+\sum_{n=M+1}^{M+T}\mathbf{u}_{ikn}^t\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha-\beta_{n'}}{\beta_n-\beta_{n'}},\tag{21}$$

for all $k\in[K]$, where $\mathcal{K}_i^t(k)$ denotes the k^{th} element of $\mathcal{K}_i^t$; $\{r_{ik}^t\}_{k\in[K]}$ are K random masks generated uniformly at random from $\mathbb{F}_p$; and $\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{k\in[K],n\in\{M+1,\dots,M+T\}}$ are generated uniformly at random from $\mathbb{F}_p^{d/M}$. The random masks $\{r_{ik}^t\}_{k\in[K]}$ will later be used to hide the true value of the gradient parameters in the online phase, whereas the random vectors $\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{k\in[K],n\in\{M+1,\dots,M+T\}}$ will hide the contents of the masks $\{r_{ik}^t\}_{k\in[K]}$ as well as the selected coordinates. Finally, user i sends the encoded vectors $\phi_{ik}(\alpha_j)$ and $\psi_{ik}(\alpha_j)$ to user $j\in[N]$, which will later be used in the online phase to ensure the correct matching of the local gradient parameters within the global model, while preventing the server from gaining explicit access to the coordinates.

B. Online Phase

After local training and sparsification, each user $i\in[N]$ generates a finite field representation of its sparsified local gradient $\mathbf{x}_i^t$, i.e.,

$$\overline{\mathbf{x}}_i^t(\ell)\triangleq f(\mathbf{x}_i^t(\ell))\quad\forall\ell\in\mathcal{K}_i^t\tag{22}$$

where the finite field transformation $f(\cdot)$ is common to all SA frameworks, whose details are provided in App. C. Next, user i broadcasts a masked gradient parameter,

$$\widehat{\mathbf{x}}_{ik}^t\triangleq\overline{\mathbf{x}}_i^t(\mathcal{K}_i^t(k))-r_{ik}^t\tag{23}$$

for each $k\in[K]$, where the true content of the K selected parameters $\mathcal{K}_i^t$ are hidden by the K random masks $r_{i1}^t,\dots,r_{iK}^t$ generated in the offline phase. After receiving (23), each user i sends a local aggregate of the encoded gradients:

$$\varphi(\alpha_i)\triangleq\sum_{j\in\mathcal{U}(t)}\sum_{k\in[K]}(\widehat{\mathbf{x}}_{jk}^t\phi_{jk}(\alpha_i)+\psi_{jk}(\alpha_i)),\tag{24}$$

to the server. The local computations $\varphi(\alpha_i)$ in (24) can be viewed as evaluations of a degree $M+T-1$ polynomial $\varphi(\alpha)$ at $\alpha=\alpha_i$. As a result, after collecting $\varphi(\alpha_i)$ from any set of $M+T$ users, the server can reconstruct the polynomial $\varphi(\alpha)$ via polynomial interpolation, and recover,

$$\mathbf{x}_{agg}^t=\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t=\begin{bmatrix}\varphi^{\mathrm{T}}(\beta_1)&\cdots&\varphi^{\mathrm{T}}(\beta_M)\end{bmatrix}^{\mathrm{T}}\tag{25}$$

which corresponds to the *true (desired) sum of the sparsified local gradients*, as will be proved in Thm. 4. After aggregating the local gradients, the server updates the global model,

$$\mathbf{w}^{t+1}\leftarrow\mathbf{w}^t-\frac{1}{|\mathcal{U}(t)|}f^{-1}(\mathbf{x}_{agg}^t)\tag{26}$$

The individual steps of our framework are presented in Alg. 2. In Fig. 7, we present the reconstruction quality for TinySecAgg under the same experimental setup as described in Section IV with sparsification levels $K=0.01d$ and $K=0.1d$. Since the server does not have access to the coordinate information with TinySecAgg, the reconstruction attack is equivalent to an inversion attack that targets the aggregated gradients to reveal the local images.

Algorithm 2 TinySecAgg

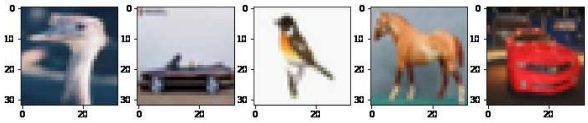
Input: Number of users N, sparsification ratio $\frac{K}{d}$, finite field representation of sparsified local gradients $\overline{\mathbf{x}}_i^t\in\mathbb{F}_p^K$ of users $i\in[N]$, model size d, number of shards M, distinct public parameters $\{\alpha_i\}_{i\in[N]}$, $\{\beta_n\}_{n\in[M]}$ in the finite field $\mathbb{F}_p$.
Output: Updated global model, $\mathbf{w}^{t+1}$

Offline phase:
for user $i=1,2,\dots,N$ in parallel **do**
  generate one-hot-encoded vectors $\mathbf{a}_k$ for $k\in[d]$
  partition $\mathbf{a}_k$ into M shards for $k\in[d]$ ▷ Equation (19)
  generate uniformly random masks r_{ik}^t for all $k\in[K]$ from $\mathbb{F}_p$
  generate uniformly random vectors $\mathbf{v}_{ikn}^t$, $\mathbf{u}_{ikn}^t$ for all $k\in[K]$, $n\in[M]$ from $\mathbb{F}_p^{\frac{d}{M}}$
  for $j=1,\dots,N\setminus\{i\}$ **do**
    Compute $\phi_{ik}(\alpha_j)$ for all $k\in[K]$ ▷ Equation (20)
    Compute $\psi_{ik}(\alpha_j)$ for all $k\in[K]$ ▷ Equation (21)
    Send $\phi_{ik}(\alpha_j)$, $\psi_{ik}(\alpha_j)$ to user j
  end for
end for
Online phase:
for user $i=1,2,\dots,N$ in parallel **do**
  Construct the masked sparsifed gradient $\widehat{\mathbf{x}}_i^t$ ▷ Equation (23)
  Broadcast $\widehat{\mathbf{x}}_i^t$
end for
Server:
Collect the local computations $\varphi(\alpha_i)$ from surviving users $i\in\mathcal{U}(t)$ ▷ Equation (24)
Recover the aggregate of the sparsified gradients $\mathbf{x}_{agg}^t$ ▷ Equation (25)
Update the global model $\mathbf{w}^{t+1}$ ▷ Equation (26)

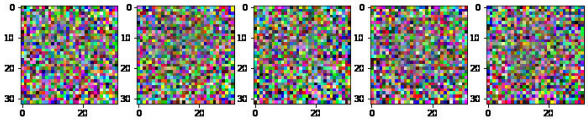
VI. Discussion

Our mechanism builds on the following key intuition. In the offline phase, each user $i\in[N]$ generates K random coordinates for sparsification, and then encodes them by using two degree $M+T-1$ polynomials $\phi_{ik}(\alpha)$ and $\psi_{ik}(\alpha)$ from (20) and (21), respectively, for each coordinate $k\in[K]$, where M denotes the total number of shards and T denotes the maximum number of adversarial users. Then, user i sends two interpolation points $\phi_{ik}(\alpha_j),\psi_{ik}(\alpha_j)$ of $\phi_{ik}(\alpha),\psi_{ik}(\alpha)$

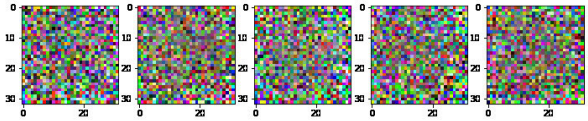
Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.



(a) Original images.



(b) Recovered images ($K = 0.01d$).



(c) Recovered images ($K = 0.1d$).

Fig. 7. Image reconstruction quality for TinySecAgg.

to user $j \in [N]$. The T random masks preserve the privacy of the coordinates against up to T colluding adversaries, as will be demonstrated in our theoretical analysis in Section VII.

The two polynomials are later used in the online phase to aggregate the gradient parameters for the selected coordinates of each user. To that end, the first polynomial $\phi_{ik}(\alpha)$ encodes whether a given coordinate $k \in [K]$ is selected by user $i \in [N]$ in the offline phase. This polynomial is then used in the online phase in (24) to extract the selected gradient parameters for each user. Specifically, after user $i \in [N]$ broadcasts the masked gradient $\widehat{\mathbf{x}}_{ik}^t$ from (23) in the online phase, each user $j \in [N]$ computes $\widehat{\mathbf{x}}_{ik}^t \phi_{ik}(\alpha_j)$. This can be viewed as an evaluation of a degree $M+T-1$ polynomial $\widehat{\mathbf{x}}_{ik}^t \phi_{ik}(\alpha)$ where,

$$\begin{aligned}
&\widehat{\mathbf{x}}_{ik}^t \phi_{ik}(\alpha_j) \\
&= \begin{bmatrix} 0 & \cdots & 0 & \overline{\mathbf{x}}_i^t(\mathcal{K}_i^t(k)) - r_{ik}^t & 0 & \cdots & 0 \end{bmatrix}^{\mathrm{T}} \\
&\quad \times \prod_{n' \in [M+T] \setminus \{n : \frac{(n-1)d}{M} \le \mathcal{K}_i^t(k) \le \frac{nd}{M}\}} \frac{\alpha_j - \beta_{n'}}{\beta_n - \beta_{n'}} \\
&\quad + \sum_{n=M+1}^{M+T} (\overline{\mathbf{x}}_i^t(\mathcal{K}_i^t(k)) - r_{ik}^t) \mathbf{v}_{ikn}^t \prod_{n' \in [M+T] \setminus \{n\}} \frac{\alpha_j - \beta_{n'}}{\beta_n - \beta_{n'}}
\end{aligned} \tag{27}$$

is the local computation of user i. Hence,

$$\begin{aligned}
&\widehat{\mathbf{x}}_{ik}^t \phi_{ik}(\beta_n) \\
&\quad = \begin{bmatrix} 0 & \cdots & 0 & \overline{\mathbf{x}}_i^t(\mathcal{K}_i^t(k)) - r_{ik}^t & 0 & \cdots & 0 \end{bmatrix}^{\mathrm{T}}
\end{aligned}$$

is non-zero only for the m^{th} element of the n^{th} shard when $(n-1)d/M \le \mathcal{K}_i^t(k) = (n-1)d/M + m \le nd/M$, and zero otherwise,

$$\widehat{\mathbf{x}}_{ik}^t \phi_{ik}(\beta'_n) = \begin{bmatrix} 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \end{bmatrix}^{\mathrm{T}}$$

for all other shards $n' \in [M] \setminus \{n\}$ such that $\mathcal{K}_i^t(k) \notin \{(n'-1)d/M, \ldots, n'd/M\}$. On the other hand, the true gradient $\overline{\mathbf{x}}_i^t(\mathcal{K}_i^t(k))$ in (27) is still masked by the random mask r_{ik}^t. This is addressed by the second polynomial $\psi_{ik}(\alpha_j)$ from the offline phase, which cancels the additive mask r_{ik}^t,

$$\begin{aligned}
&\widehat{\mathbf{x}}_{ik}^t \phi_{ik}(\alpha_j) + \psi_{ik}(\alpha_j) \\
&= \begin{bmatrix} 0 & \cdots & 0 & \overline{\mathbf{x}}_i^t(\mathcal{K}_i^t(k)) & 0 & \cdots & 0 \end{bmatrix}^{\mathrm{T}} \\
&\quad \times \prod_{n' \in [M+T] \setminus \{n : \frac{(n-1)d}{M} \le \mathcal{K}_i^t(k) \le \frac{nd}{M}\}} \frac{\alpha_j - \beta_{n'}}{\beta_n - \beta_{n'}} \\
&\quad + \sum_{n=M+1}^{M+T} \left((\overline{\mathbf{x}}_i^t(\mathcal{K}_i^t(k)) - r_{ik}^t) \mathbf{v}_{ikn}^t + \mathbf{u}_{ikn}^t \right) \\
&\quad \times \prod_{n' \in [M+T] \setminus \{n\}} \frac{\alpha_j - \beta_{n'}}{\beta_n - \beta_{n'}}
\end{aligned}$$

Then, the local sum $\varphi(\alpha_j) = \sum_{i \in \mathcal{U}(t)} \sum_{k \in [K]} (\widehat{\mathbf{x}}_{ik}^t \phi_{ik}(\alpha_j) + \psi_{ik}(\alpha_j))$ from (24) encodes the sum of the K selected coordinates from the surviving users $\mathcal{U}(t)$ in a degree $M+T-1$ polynomial. After receiving at least $M+T$ evaluation points as in (24) from the surviving users, the server can reconstruct the sum of the sparsified gradients through polynomial interpolation as in (25). Accordingly, the two polynomials generated in the offline phase ensures: 1) cancellation of the random masks, and 2) correct association between the selected gradient parameters and the corresponding coordinates in the online phase. We next demonstrate the theoretical guarantees and performance trade-offs of TinySecAgg.

VII. Theoretical Analysis

In this section, we present the formal privacy, complexity, and correctness guarantees of TinySecAgg.

Theorem 1 (Privacy): TinySecAgg ensures information-theoretic privacy for both the gradients and their coordinates against up to T adversaries, for any $|\mathcal{U}(t)| \ge T + M$,

$$I(\{\overline{\mathbf{x}}_i^t, \mathcal{K}_i^t\}_{[N] \setminus \mathcal{T}}; \mathcal{M}_{\mathcal{T}}^t | \sum_{i \in \mathcal{U}(t)} \overline{\mathbf{x}}_i^t, \{\overline{\mathbf{x}}_i^t, \mathcal{K}_i^t\}_{i \in \mathcal{T}}, \mathcal{R}_{\mathcal{T}}^t) = 0$$

where $\overline{\mathbf{x}}_i^t$ is the local gradient of user i, $\mathcal{K}_i^t$ denotes the set of coordinates selected by user i, $\sum_{i \in \mathcal{U}(t)} \overline{\mathbf{x}}_i^t$ is the gradient aggregate, whereas

$$\mathcal{M}_{\mathcal{T}}^t = \{\phi_{ik}(\alpha_j), \psi_{ik}(\alpha_j)\}_{i \in [N], j \in \mathcal{T}, k \in [K]}, \{\varphi(\alpha_i)\}_{i \in \mathcal{U}(t)}$$

denotes the set of all messages received by the adversaries and the server, and

$$\mathcal{R}_{\mathcal{T}}^t = \{r_{ik}^t\}_{i \in \mathcal{T}, k \in [K]}, \{\mathbf{v}_{ikn}^t, \mathbf{u}_{ikn}^t\}_{\substack{i \in \mathcal{T}, k \in [K], \\ n \in \{M+1, \ldots, M+T\}}}$$

denotes the set of randomness generated by the adversaries.

Proof: The proof is provided in App. D. □

Theorem 2 (Communication Complexity): The per-user communication complexity of TinySecAgg is $O(K + \frac{d}{M})$ (online) and $O(KN\frac{d}{M})$ (offline).

Proof: (Online) The online communication overhead of user $i \in [N]$ consists of: 1) $O(K)$ for broadcasting $\widehat{\mathbf{x}}_{ik}^t$ from (23), 2) $O(\frac{d}{M})$ for sending $\varphi(\alpha_i)$ from (24) to the server.

(Offline) The offline communication overhead of user $i \in [N]$ consists of: 1) $O(NK\frac{d}{M})$ to send $\phi_{ik}(\alpha_j)$ from (20) for all $k \in [K]$ to users $j \in [N]$, 2) $O(NK\frac{d}{M})$ to send $\psi_{ik}(\alpha_j)$ from (21) for all $k \in [K]$ to users $j \in [N]$. □

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

Remark 1: TinySecAgg enables a reduced online communication overhead by transferring the communication intensive operations to the offline phase, which can be performed when the network load is low.

Theorem 3 (Computation Complexity): The per-user computation complexity of TinySecAgg is $O(NK\frac{d}{M})$ *(online) and* $O(NK\frac{d}{M}\log^2(M+T)\log\log(M+T))$ *(offline).*

Proof: (Online) The online computation overhead of user $i \in [N]$ consists of: 1) $O(K)$ to compute the masked sparse gradient from (23), 2) $O(KN\frac{d}{M})$ to compute $\varphi(\alpha_i)$ in (24).

(Offline) Interpolating a polynomial of degree γ, and evaluating it at γ points incurs a computational overhead of $\gamma\log^2\gamma\log\log\gamma$ [52]. Then, the offline computation overhead of user $i \in [N]$ consists of: 1) $O(NK\frac{d}{M}\log^2(M+T)\log\log(M+T))$ to compute $\phi_{ik}(\alpha_j)$ in (20) for all $k \in [K]$ and $j \in [N]$, 2) $O(NK\frac{d}{M}\log^2(M+T)\log\log(M+T))$ to compute $\psi_{ik}(\alpha_j)$ in (21) for all $k \in [K]$ and $j \in [N]$. □

Theorem 4 (Correctness): TinySecAgg ensures the correct recovery of the aggregate $\mathbf{x}_{agg}^t = \sum_{i\in\mathcal{U}(t)} \overline{\mathbf{x}}_i^t$ *of sparsified gradients from (25), from the messages of any set* $\mathcal{U}(t)$ *of* $|\mathcal{U}(t)| \geq M+T$ *surviving users,*

$$H\Big(\sum_{i\in\mathcal{U}(t)} \overline{\mathbf{x}}_i^t | \{\widehat{\mathbf{x}}_i^t\}_{i\in\mathcal{U}(t)}, \{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}\Big) = 0 \tag{28}$$

Proof: By using (20) and (21), (24) can be written as,

$$\begin{aligned}\varphi(\alpha_i) = &\sum_{j\in\mathcal{U}(t)}\sum_{k\in[K]}\Big(\overline{\mathbf{x}}_j^t(\mathcal{K}_j^t(k))\sum_{n\in[M]}\mathbf{a}_{\mathcal{K}_j^t(k),n}\\ &\times\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha_i-\beta_{n'}}{\beta_n-\beta_{n'}}\\ &+\sum_{n=M+1}^{M+T}(\overline{\mathbf{x}}_j^t(\mathcal{K}_j^t(k))\mathbf{v}_{jkn}^t - r_{jk}^t\mathbf{v}_{jkn}^t + \mathbf{u}_{jkn}^t)\\ &\times\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha_i-\beta_{n'}}{\beta_n-\beta_{n'}}\Big),\end{aligned} \tag{29}$$

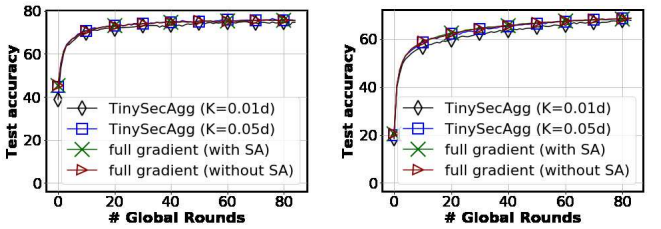
which corresponds to an evaluation point of the degree $M+T-1$ polynomial $\varphi(\alpha)$ at $\alpha=\alpha_i$. Since the polynomial $\varphi(\alpha)$ has degree $M+T-1$, it can be perfectly reconstructed using polynomial interpolation from any set of at least $M+T$ evaluation points. After reconstructing the polynomial $\varphi(\alpha)$ by using the evaluations $\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}$ where $|\mathcal{U}(t)| \geq M+T$, the server can then recover the desired computations (i.e., the true sum of the sparsified local gradients) by evaluating,

$$\begin{aligned}&\varphi(\beta_n)\\ &=\sum_{j\in\mathcal{U}(t)}\sum_{k\in[K]}\overline{\mathbf{x}}_j^t(\mathcal{K}_j^t(k))\mathbf{a}_{\mathcal{K}_j^t(k),n}\\ &=\Big[\textstyle\sum_{j\in\mathcal{U}(t)}\overline{\mathbf{x}}_j^t((n-1)\frac{d}{M}+1)\quad\cdots\quad\sum_{j\in\mathcal{U}(t)}\overline{\mathbf{x}}_j^t(n\frac{d}{M})\Big]^{\mathrm{T}}\end{aligned}$$

for all $n \in [M]$, which corresponds to the aggregate of the sparse local gradients in the n^{th} shard. Hence, the entire gradient aggregate can be recovered by concatenating the M shards as shown in (25). □

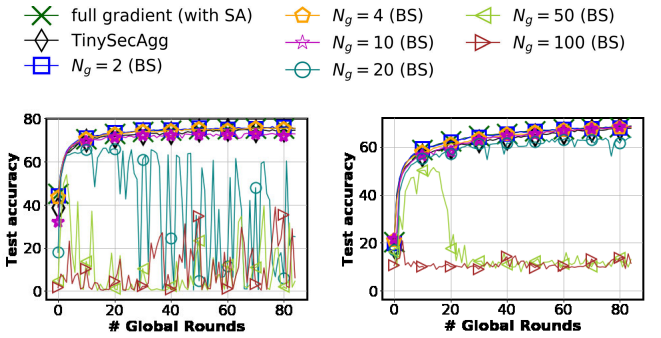
Remark 2: Parameter M *defines a trade-off between the communication overhead and adversary tolerance. Increasing* M *reduces the communication overhead* $O(K+\frac{d}{M})$ *(online) and* $O(NK\frac{d}{M})$ *(offline) as shown in Thm. 2. On the other hand, increasing* M *also leads to a reduced adversary tolerance* T *as* $T \leq |\mathcal{U}(t)| - M \leq N - M$.

Finally, the convergence guarantees follow from [12], [20].



(a) FEMNIST. (b) CIFAR-10.

Fig. 8. Test accuracy of TinySecAgg vs. full gradient aggregation (with and without SA).



(a) FEMNIST. (b) CIFAR-10.

Fig. 9. Test accuracy of TinySecAgg with respect to full gradient aggregation, and block-sparsification (BS) with varying group size N_g.

TABLE I
COMPARISON OF PER-ROUND COMMUNICATION COMPLEXITY OF TINYSECAGG WITH SA BASELINES

	Communication load (per-user)	
	Online	Offline
SecAgg	$O(N+d)$	$O(N)$
SecAgg+	$O(\log(N)+d)$	$O(\log(N))$
LightSecAgg	$O(d+d/M)$	$O(Nd/M)$
TinySecAgg	$O(K+d/M)$	$O(KNd/M)$

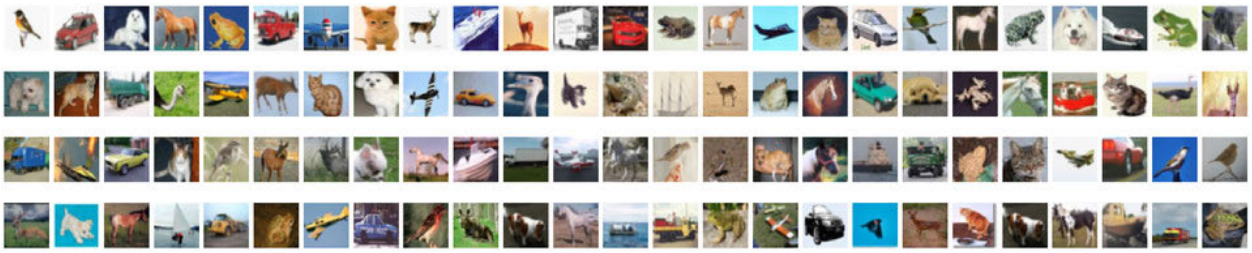
VIII. EXPERIMENTS

We first evaluate the performance of TinySecAgg with respect to conventional SA baselines SecAgg [8], SecAgg+ [9], and LightSecAgg [26], where users aggregate the full gradient as described in App. A.

A. Setup

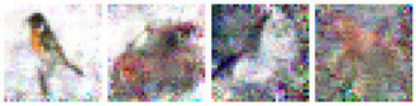
We consider a setting with $N=100$ users, where $T=\frac{N}{2}$ [8] users are adversarial. We consider image classification tasks on: 1) FEMNIST, a non-IID dataset by design, using the CNN model and data distribution from [53], 2) CIFAR-10, using the CNN model and data distribution (IID) from [1]. As our goal is to analyze the comparative performance of different frameworks on the same architecture (as opposed to optimizing model accuracy), we opt for these lightweight architectures without loss of generality. We evaluate the performance in two sparsification levels $K=0.01d$, and $K=0.05d$, with $M=40$. The additional experimental details and hyperparameters are provided in App. E.

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

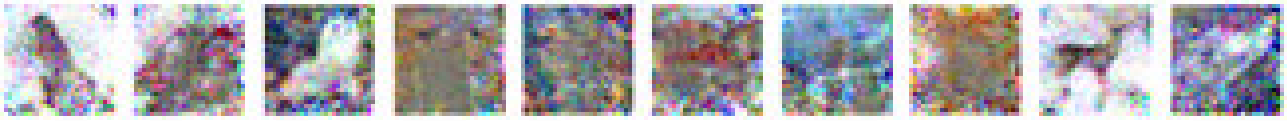


(a) Original images.

(b) Recovered images for $N_g = 2$.

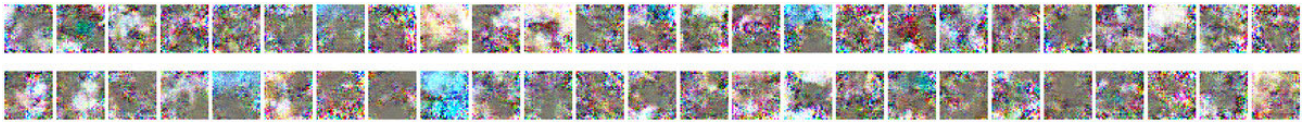


(c) Recovered images for $N_g = 4$.

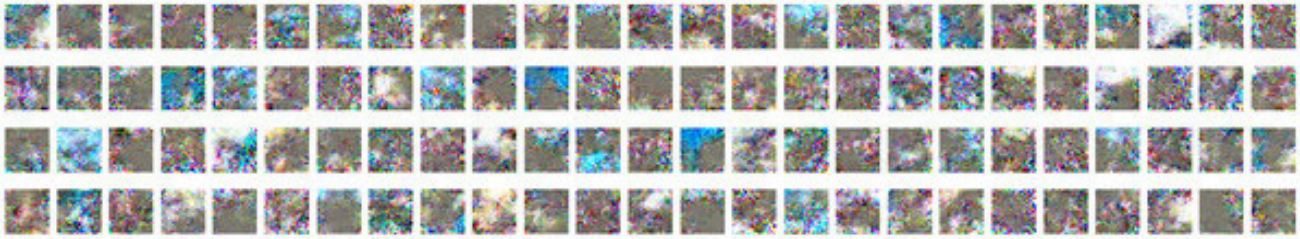


(d) Recovered images for $N_g = 10$.

(e) Recovered images for $N_g = 20$.



(f) Recovered images for $N_g = 50$.



(g) Recovered images for $N_g = 100$.

Fig. 10. Image reconstruction quality for block-sparsification with varying number of users (N_g) per group.

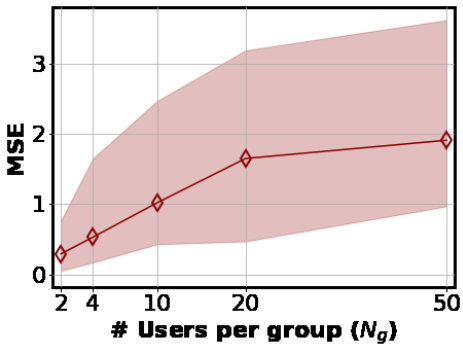


Fig. 11. Average mean squared error (MSE) with respect to the group size N_g. The shaded area represents the variation between the maximum and minimum MSE across all users.

TABLE II

TOTAL COMMUNICATION OVERHEAD (MBITS) TO REACH CONVERGENCE TEST ACCURACY (75%-FEMNIST, 67%-CIFAR-10)

Framework	FEMNIST	CIFAR-10
SecAgg	738813	172304
SecAgg+	738810	172300
LightSecAgg	757277	176601
TinySecAgg ($K = 0.05d$)	56831	13150
TinySecAgg ($K = 0.01d$)	43760	7645

Table I compares the per-round communication complexity of TinySecAgg with the SA baselines. In Table II, we present the total online communication overhead to reach the target test accuracy at convergence. As the model size d is large (in the order of 10^6) compared to the number of users N, all SA baselines incur similar communication overhead (as they communicate the full gradient), whereas TinySecAgg reduces the cost by up to $22.5\times$ over the best baseline.

In Fig. 8, we further compare the convergence of TinySecAgg (SA with sparsification) with respect to SA without sparsification (i.e., full gradient aggregation with SA), when

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

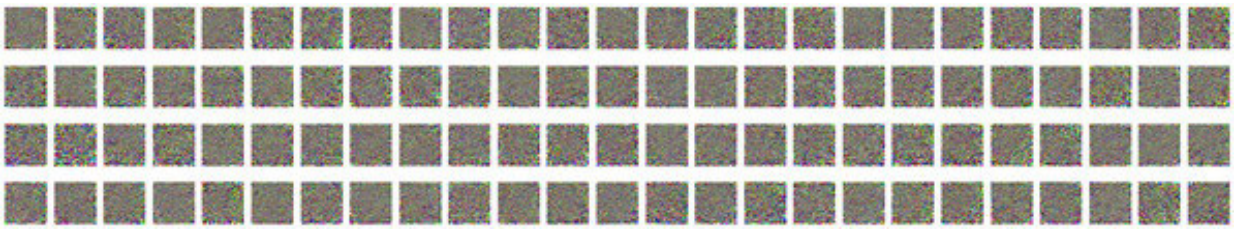


Fig. 12. Image reconstruction quality for TinySecAgg ($N = 100$ users).

10% of the users randomly drop out at each round. As detailed in App. A, all SA baselines carry out the same training operations and hence have the same test accuracy, as they only differ in their encoding mechanism, not in training. We also demonstrate the convergence of the same model architecture, but without sparsification or SA (i.e., full gradient aggregation without SA). This represents our target accuracy, as the finite-field conversion in SA can degrade accuracy, as detailed in App. C. We observe that TinySecAgg achieves comparable accuracy to all baselines (with or without SA).

As detailed in Section IV, rand-K/top-K variants of conventional SA frameworks are vulnerable to multi-round gradient reconstruction attacks. To that end, we next consider another SA variant, where we partition users in groups of size N_g and then assign users within each group to a set of $K \ll d$ randomly generated coordinates. The key advantage is that now all users in a given group participate at the same set of selected coordinates, which can be a potential solution to avoid multi-round reconstruction attacks, while still enhancing communication-efficiency. In the following, we will call this variant *block-sparsification*.

We then adapt LightSecAgg to block-sparsification, as this baseline is compatible with block-sparsification and has demonstrated enhanced communication-efficiency over the alternative baselines. From [54, Theorem 1], for a given set of users $\mathcal{S}_g$ of size $N_g = |\mathcal{S}_g|$, an upper bound on the mutual information between the local dataset $\mathcal{D}_i$ of user $i \in \mathcal{S}_g$ and the gradient aggregate $\sum_{i\in\mathcal{S}_g} \Delta_i^t$ for block-sparsification can be obtained as,

$$I\left(\mathcal{D}_i; \sum_{i\in\mathcal{S}_g} \Delta_i^t\right) \leq O\left(\frac{1}{N_g B}\right) \tag{30}$$

where Δ_i is the local gradient of user i as defined in (3), and B is the batch size for training. As such, a smaller group size increases the information leakage from the gradient aggregate.

From (30), we expect a gradual increase in the privacy leakage as the number of users in each group decreases. To investigate the impact of the group size on the model accuracy, in Fig. 9, we present the test accuracy with a learning rate of $\eta = 0.001$ for all frameworks, including TinySecAgg and the baselines, with varying group size N_g for block sparsification. For a fair comparison, we have set the sparsification level as $K = 0.01d$ for TinySecAgg and $K = 0.03d$ for block-sparsification, to ensure an equal online communication overhead per round for both frameworks, which is given by $O(K + \frac{K}{M})$ for block-sparsification with LightSecAgg. We observe that model performance degrades as the number of users per group increases for block-sparsification, and performance starts to degrade gradually starting with $N_g = 20$ users per group. Accordingly, Fig. 9 suggests a trade-off between model accuracy and privacy leakage as a function of the group size N_g in block sparsification. Smaller group sizes achieve better accuracy by trading-off with increased privacy leakage.

In Fig. 10, we further explore the impact of the varying number of users per group on the image reconstruction quality from the gradient aggregate, for block-sparsification. Note that by using the coordinate information over multiple training rounds, the entire gradient aggregate of each group can be recovered by performing our multi-round reconstruction attack from Section IV. For ease of exposition, we illustrate the results for the first group, while in Fig. 11 we further report the average, minimum, and maximum mean squared error (MSE) between ground-truth images and reconstructed images across all groups and users. In both Figs. 10 and 11, we observe that reducing the number of users per group increases the information leakage from the aggregate of the gradients, as expected from the theoretical results from (30). Finally, in Fig. 12, we present the attack performance for TinySecAgg, under the same experiment setup. We observe that reconstruction fails for all $N = 100$ users. Accordingly, TinySecAgg retains the model performance while reducing the information leakage of the local data samples.

IX. Conclusion

In this work, we study gradient sparsification in the context of SA. We identify the vulnerabilities of sharing coordinate information in gradient sparsification, and introduce a coordinate-hiding SA mechanism, TinySecAgg, to enhance the reliability of SA in resource-limited settings. Our framework provides an order of magnitude improvement in communication-efficiency without degrading model accuracy. Future directions include extending our mechanisms to Byzantine adversaries, where the server can maliciously poison the model parameters to recover the local gradients of the users.

Appendix A
Conventional Secure Aggregation Protocols

In this section, we review the underlying principles of conventional SA mechanisms. Due to space constraints, we present the key steps of two representative frameworks.

A. SecAgg

The SecAgg framework [8] is the first SA protocol for large-scale secure gradient aggregation, whose underlying

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

principles have been widely adopted by follow-up works [9]. At the outset, SecAgg leverages pairwise additive random masks to hide the local gradients. The pairwise masks cancel out upon aggregation, allowing the server to recover the true sum of the gradients, but without observing them in the clear. The protocol consists of the following offline and online phases. All operations are carried out in a finite field $\mathbb{F}_p$.

B. Offline Phase

In the offline phase, each pair of users $i, j \in [N]$ agree on a *pairwise* random seed, s_{ij} through a Diffie-Hellman type key exchange protocol [55], by utilizing private-public key pairs (s_i^{SK}, s_i^{PK}), (s_j^{SK}, s_j^{PK}) belonging to users i and j respectively. The seed s_{ij} is then expanded into a d-dimensional random vector within the finite field $\mathbb{F}_p$, using a pseudo-random generator (PRG), to create a *pairwise mask*,

$$\mathbf{r}_{ij} \triangleq \text{PRG}(s_{ij}) \tag{31}$$

In addition, user i generates a *private* random seed, s_i, which is then expanded into a d-dimensional random vector within $\mathbb{F}_p$, to create a *private mask* $\mathbf{r}_i \triangleq \text{PRG}(s_i)$. Each user $i \in [N]$ then secret shares its private key s_i^{SK} and private seed s_i with all other users using Shamir's $N/2$-out-of-N secret sharing [56]. Secret sharing ensures that the secrets s_i^{SK} and s_i can be perfectly reconstructed from any set of $\frac{N}{2}$ shares, whereas any set of less than $\frac{N}{2}$ shares reveals no information about the secret. The secret shares are later used in the online phase while recovering the final gradient aggregate.

C. Online Phase

In the online phase, each user $i \in [N]$ first maps its local update $\mathbf{x}_i \in \mathbb{R}_p^d$ to the finite field $\mathbb{F}_p$ as detailed in App. C, and obtains its finite field representation $\overline{\mathbf{x}}_i \in \mathbb{F}_p^d$. Then, user i masks $\overline{\mathbf{x}}_i$ using the pairwise and private masks generated in the offline phase,

$$\widehat{\mathbf{x}}_i \triangleq \overline{\mathbf{x}}_i + \mathbf{r}_i + \sum_{j\in[N]:i<j} \mathbf{r}_{ij} - \sum_{j\in[N]:i>j} \mathbf{r}_{ij} \quad \text{mod } p \tag{32}$$

and sends the masked gradient $\widehat{\mathbf{x}}_i$ to the server. Upon receiving the masked gradients $\widehat{\mathbf{x}}_i$ from the surviving users $i \in \mathcal{U}$, the server computes their aggregate $\sum_{i\in\mathcal{U}} \widehat{\mathbf{x}}_i$. Upon aggregation, the pairwise masks of all the surviving users cancel out. On the other hand, if some users drop out from the protocol and fail to send their masked gradient to the server, their pairwise masks will not cancel out. To learn the true aggregate of the gradients, the server has to remove the pairwise additive masks corresponding to the dropped users, and the private masks corresponding to the surviving users from the aggregate of the masked gradients. To do so, the server collects from the surviving users, the secret shares of the random seeds corresponding to the pairwise masks of dropped users, and the private masks of surviving users, reconstructs the corresponding masks, and removes them from the aggregate of the masked gradients,

$$\sum_{i\in\mathcal{U}} \widehat{\mathbf{x}}_i - \sum_{i\in\mathcal{U}} \mathbf{r}_i - \sum_{i\in\mathcal{D}} \sum_{j:i<j} \mathbf{r}_{ij} + \sum_{i\in\mathcal{D}} \sum_{j:i>j} \mathbf{r}_{ij} \quad \text{mod } p \tag{33}$$

A more recent variant of SecAgg is the SecAgg+ protocol from [9], which reduces the communication overhead by using a graph-based topology for distributing the secret shares across the users. SecAgg and its variants are fully compatible with rand-K/top-K sparsification, but are vulnerable to the multi-round reconstruction attacks described in Section IV.

D. LightSecAgg

The LightSecAgg protocol [26] replaces the dropout recovery mechanism of SecAgg with a one-shot recovery mechanism, to address the communication and computation complexity of pairwise masking, which increases as the number of dropped users increases. The protocol follows the following offline and online phases.

1) Offline Phase: Each user $i \in [N]$ generates a uniformly random mask $\mathbf{z}_i \in \mathbb{F}_p^d$, partitioned into M equal-sized shards:

$$\mathbf{z}_i = \begin{bmatrix} \mathbf{z}_{i1} & \dots & \mathbf{z}_{iM} \end{bmatrix}^{\mathrm{T}} \tag{34}$$

Next, user i encodes the random mask, by generating a degree $M+T-1$ polynomial,

$$h_i(\alpha) \triangleq \sum_{n\in[M]} \mathbf{z}_{in} \prod_{n'\in[M+T]\setminus\{n\}} \frac{\alpha-\beta'_n}{\beta_n-\beta'_n} + \sum_{n=M+1}^{M+T} \mathbf{v}_{in} \prod_{n'\in[M+T]\setminus\{n\}} \frac{\alpha-\beta'_n}{\beta_n-\beta'_n}, \tag{35}$$

where $\{\mathbf{v}_{in}\}_{n\in\{M+1,\dots,M+T\}}$ are uniformly random vectors, and sends an encoded mask, $\widetilde{\mathbf{z}}_{ij} \triangleq h_i(\alpha_j)$ to user $j \in [N]$.

2) Online Phase : User i sends a masked local gradient $\widehat{\mathbf{x}}_i \triangleq \overline{\mathbf{x}}_i + \mathbf{z}_i$ to the server, where $\overline{\mathbf{x}}_i \in \mathbb{F}_p^d$ denotes the finite field representation of the local gradient. After receiving the masked gradients from the set of surviving users, the server collects the aggregate of the encoded masks,

$$\widetilde{\mathbf{z}}_i \triangleq \sum_{i\in\mathcal{U}} \widetilde{\mathbf{z}}_{ij} \tag{36}$$

from the surviving users $i \in \mathcal{U}$, where each encoded mask aggregate is an evaluation of the degree $M+T-1$ polynomial,

$$h(\alpha) \triangleq \sum_{n\in[M]} \sum_{i\in\mathcal{U}} \mathbf{z}_{in} \prod_{\substack{n'\in[M+T]\\ \setminus\{n\}}} \frac{\alpha-\beta'_n}{\beta_n-\beta'_n} + \sum_{n=M+1}^{M+T} \sum_{i\in\mathcal{U}} \mathbf{v}_{in} \prod_{\substack{n'\in[M+T]\\ \setminus\{n\}}} \frac{\alpha-\beta'_n}{\beta_n-\beta'_n} \tag{37}$$

After collecting $M+T$ evaluations, the server reconstructs the polynomial $h(\alpha)$ through polynomial interpolation, and decodes the aggregate of the random masks,

$$\sum_{i\in\mathcal{U}} \mathbf{z}_i = \begin{bmatrix} \sum_{i\in\mathcal{U}} \mathbf{z}_{i1} & \dots & \sum_{i\in\mathcal{U}} \mathbf{z}_{iM} \end{bmatrix}^{\mathrm{T}} \tag{38}$$

Finally, the server recovers the aggregate of the local gradients by subtracting (38) from the aggregate of the masked gradients $\sum_{i\in\mathcal{U}} \widehat{\mathbf{x}}_i$. As this protocol encodes the entire gradient into M shards, it is not compatible with rand-K/top-K sparsification. However, a sparsified variant can be obtained by using the block-sparsification approach described in Section VIII.

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

(a) Original images.

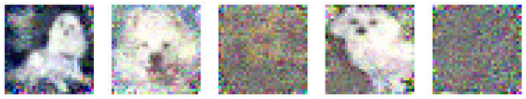
(b) Recovered images (rand-K).

(c) Recovered images (top-K).

Fig. 13. Reconstruction quality with non-IID distribution.

(a) Original images.

(b) Recovered images (rand-K).



(c) Recovered images (top-K).

Fig. 14. Reconstruction quality with IID distribution.

APPENDIX B
ATTACK PERFORMANCE FOR IID VS NON-IID DATA DISTRIBUTION

In this section, we demonstrate the attack performance between IID and non-IID data distribution across the users, for the experimental setup in Section IV, with $N = 5$ users and sparsification parameter $K = 0.01d$, where the attack is run for 500 training rounds. In Fig. 13, we first demonstrate the reconstruction performance for the non-IID setting where the data sample for each user is sampled from a different class.

To further investigate the reconstruction performance between IID versus non-IID data distribution across the users, we next consider the IID setting where each user is assigned a data sample from the same class. We demonstrate our results in Fig. 14, where we observe that for both top-K and rand-K, majority of the images can be reconstructed. We observe that the attack performance is greater in the non-IID setting for top-K. This suggests a higher degree of separation between the local gradients in the non-IID setting, whereas in the IID setting there may be overlaps between the top-K parameters between different users due to the similarity across the images. As real-world FL mechanisms often operate under non-IID distributions, our results highlight the need for strong defenses for FL.

APPENDIX C
DETAILS OF THE FINITE FIELD TRANSFORMATION

The local update $\mathbf{x}_i^t(\ell)$ of each user i is converted to the finite field as follows,

$$\overline{\mathbf{x}}_i^t(\ell) \triangleq f(\mathbf{x}_i^t(\ell)) \mod p \quad \forall \ell \in \mathcal{K}_i^t \tag{39}$$

where $f(\cdot)$ is a stochastic rounding function [34],

$$f(x) \triangleq \begin{cases} \lfloor qx \rfloor \mod p & \text{probability} \quad 1-(qx-\lfloor qx \rfloor) \\ \lfloor qx \rfloor + 1 \mod p & \text{otherwise} \end{cases}$$

such that q is a tuning parameter that quantifies the rounding loss. The modulo operation represents the positive and negative integers within the first and second half of the finite field $\mathbb{F}_p$ respectively, known as two's complement representation. After decoding the aggregated gradients, the server updates the global model as in (26), where $f^{-1} : \mathbb{F}_p \to \mathbb{R}$ maps the finite field representation back to the real domain,

$$f^{-1}(x) = \begin{cases} x/q & \text{if } 0 \le x < \dfrac{p-1}{2} \\ (x-p)/q & \text{if } \dfrac{p-1}{2} \le x \le p \end{cases} \tag{40}$$

APPENDIX D
PROOF OF THEOREM 1

We now present the proof of information-theoretic privacy against any set of $|\mathcal{T}| = T$ colluding users (the same analysis holds for any $|\mathcal{T}| < T$). The set of honest users is denoted by $\mathcal{H} = [N] \backslash \mathcal{T}$. For the tractability of the theoretical analysis, we let the set of adversarial users be $[T]$ without loss of generality. Throughout the analysis, we consider the worst-case scenario where all messages are communicated across the users, i.e., users declared as dropped are only delayed, and their messages are eventually received by the adversaries. We assume the number of surviving users to be equal to the recovery threshold, i.e., $|\mathcal{U}(t)| = M + T$, while noting that the same analysis can be applied also for a larger number of surviving users. Then, the mutual information condition from Thm. 1 can be written as follows,

$$\begin{aligned} &I(\{\overline{\mathbf{x}}_i^t, \mathcal{K}_i^t\}_{i\in\mathcal{H}}; \{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}}, \{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}}, \{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}}, \\ &\quad \{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)} | \sum_{i\in\mathcal{U}(t)} \overline{\mathbf{x}}_i^t, \{\overline{\mathbf{x}}_i^t, \mathcal{K}_i^t\}_{i\in\mathcal{T}}, \{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}}, \\ &\quad \{\mathbf{v}_{ikn}^t, \mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}}) \\ &= H(\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}}, \{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}}, \{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}}, \\ &\quad \{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)} | \sum_{i\in\mathcal{U}(t)} \overline{\mathbf{x}}_i^t, \{\overline{\mathbf{x}}_i^t, \mathcal{K}_i^t\}_{i\in\mathcal{T}}, \{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}}, \\ &\quad \{\mathbf{v}_{ikn}^t, \mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}}) - H(\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}}, \\ &\quad \{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}}, \{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}}, \{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)} | \sum_{i\in\mathcal{U}(t)} \overline{\mathbf{x}}_i^t, \\ &\quad \{\overline{\mathbf{x}}_i^t, \mathcal{K}_i^t\}_{i\in[N]}, \{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}}, \{\mathbf{v}_{ikn}^t, \mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}}) \end{aligned} \tag{41}$$

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

For the second term in (41), we have:

$$
\begin{aligned}
&H(\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\\
&\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\\
&\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}})\\
&=H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\\
&\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\\
&\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}}) \qquad (42)\\
&=H(\{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\\
&\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\{r_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\\
&\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}})+H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\\
&\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}}) \qquad (43)\\
&=H(\{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\\
&\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\{r_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\\
&\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}})+H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}}) \qquad (44)\\
&=H(\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\\
&\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\{r_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\\
&\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}})+H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}}) \qquad (45)\\
&=H(\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\\
&\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\{r_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}},\\
&\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+H(\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}}) \qquad (46)\\
&=H(\{\mathbf{y}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}},\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\\
&\{r_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}},\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})\\
&+H(\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}}) \qquad (47)\\
&=H(\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]},\{r_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\\
&\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\,k\in[K]\\n\in\{M+1,\ldots,M+T\}}},\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}},\{\mathbf{y}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})\\
&+H(\{\mathbf{y}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+H(\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}}) \qquad (48)\\
&=0+H(\{\mathbf{y}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+H(\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})\\
&+H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}}) \qquad (49)\\
&=H(\{\mathbf{v}_{ikn}^t\}_{\substack{i\in\mathcal{H},k\in[K]\\n\in\{M+1,\ldots,M+T\}}})+H(\{\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{H},k\in[K]\\n\in\{M+1,\ldots,M+T\}}})\\
&+H(\{r_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}}) \qquad (50)\\
&=(N-T)TK\frac{d}{M}\log p+(N-T)TK\frac{d}{M}\log p\\
&+(N-T)K\log p \qquad (51)
\end{aligned}
$$

where (42) follows from the fact that given $\overline{\mathbf{x}}_i^t$ and $\mathcal{K}_i^t$, the only remaining uncertainty in $\widehat{\mathbf{x}}_{ik}^t$ is due to the uncertainty in r_{ik}^t, for all $k\in[K]$; (43) follows from chain rule of entropy; (44) follows from the independence of the random masks, $\{r_{ik}^t\}_{i\in\mathcal{H},k\in[K]}$ generated by the honest users. In (45),

$$\mathbf{m}_{ijk}^t\triangleq\sum_{n=M+1}^{M+T}\mathbf{v}_{ikn}^t\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha_j-\beta_{n'}}{\beta_n-\beta_{n'}} \qquad (52)$$

for all $i\in\mathcal{H},j\in\mathcal{T},k\in[K]$. Next, (46) follows from the chain rule of entropy and independence of uniformly random vectors. In (47), we define,

$$\mathbf{y}_{ijk}^t\triangleq\sum_{n=M+1}^{M+T}\mathbf{u}_{ikn}^t\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha_j-\beta_{n'}}{\beta_n-\beta_{n'}} \qquad (53)$$

whereas (48) again follows from the chain rule and independence of the vectors generated uniformly at random from the finite field $\mathbb{F}_p$. We next define $\gamma_{jn}\triangleq\prod_{n'\in[M+T]\setminus\{n\}}\frac{\alpha_j-\beta_{n'}}{\beta_n-\beta_{n'}}$, which represents the Lagrange coefficients in (52), (53). Then,

$$\begin{bmatrix}\mathbf{m}_{i1k}^t & \cdots & \mathbf{m}_{iTk}^t\end{bmatrix}=\begin{bmatrix}\mathbf{v}_{i,k,M+1}^t\cdots\mathbf{v}_{i,k,M+T}^t\end{bmatrix}\mathbf{X}, \qquad (54)$$

$$\begin{bmatrix}\mathbf{y}_{i1k}^t & \cdots & \mathbf{y}_{iTk}^t\end{bmatrix}=\begin{bmatrix}\mathbf{u}_{i,k,M+1}^t\cdots\mathbf{u}_{i,k,M+T}^t\end{bmatrix}\mathbf{X}, \qquad (55)$$

where

$$\mathbf{X}\triangleq\begin{bmatrix}\gamma_{1,M+1} & \cdots & \gamma_{T,M+1}\\ \vdots & \ddots & \vdots\\ \gamma_{1,M+T} & \cdots & \gamma_{T,M+T}\end{bmatrix}. \qquad (56)$$

Note that $\mathbf{X}$ is a $T\times T$ MDS (Maximum Distance Separable) matrix (hence is invertible) according to the MDS property of Lagrange coefficients [57]. As such, one can compute $\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{i\in\mathcal{H},n\in\{M+1,\ldots,M+T\},k\in[K]}$ given $\{\mathbf{m}_{ijk}^t\}_{i\in\mathcal{H},j\in\mathcal{T},k\in[K]}$, and $\{\mathbf{y}_{ijk}^t\}_{i\in\mathcal{H},j\in\mathcal{T},k\in[K]}$ using (54) and (55) respectively. Next, (49) holds as there is no uncertainty in $\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}$ given $\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in[N]}$, $\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},k\in[K]\\n\in\{M+1,\ldots,M+T\}}}$, $\{\mathbf{m}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}}$, and $\{\mathbf{y}_{ijk}^t\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}}$. Finally, (51) follows from the entropy of

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

uniform random variables. Next, the first term in (41) can be bounded as follows,

$$
\begin{aligned}
&H(\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{T}},\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}})\\
&=H(\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{T}},\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}}) \qquad (57)\\
&\le H(\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}})+H(\{\phi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})\\
&\quad+H(\{\psi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+H(\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\\
&\quad\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{T}},\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}},\\
&\quad\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}}) \qquad (58)\\
&=H(\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}})+H(\{\phi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})\\
&\quad+H(\{\psi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+H(\{\varphi(\beta_n)\}_{n\in[M]},\{\varphi(\alpha_i)\}_{i\in\mathcal{T}}|\\
&\quad\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{T}},\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}},\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\psi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}}) \qquad (59)\\
&=H(\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}})+H(\{\phi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})\\
&\quad+H(\{\psi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}})+0 \qquad (60)\\
&\le(N-T)K\log p+(N-T)TK\frac{d}{M}\log p\\
&\quad+(N-T)TK\frac{d}{M}\log p \qquad (61)
\end{aligned}
$$

where (57) holds since there is no uncertainty in $\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}}$ given $\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{T}}$, $\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}}$, (58) follows from the chain rule of entropy and that conditioning cannot increase entropy. Equation (59) follows from the fact that $\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}$ correspond to $M+T$ evaluations of a degree $M+T-1$ polynomial, $\varphi(\alpha)$ for $\alpha\in\{\alpha_i\}_{i\in\mathcal{U}(t)}$. By leveraging polynomial interpolation, one can uniquely recover a polynomial of degree $M+T-1$ from any set of $M+T$ evaluation points. Therefore, there is a bijective mapping between the $M+T$ interpolation points $\{\varphi(\beta_n)\}_{n\in[M]},\{\varphi(\alpha_i)\}_{i\in[T]}$ and the $M+T$ local evaluations $\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}$. Note that there is no uncertainty in $\{\varphi(\beta_n)\}_{n\in[M]}$ given $\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t$, which follows from equation (25). Next, there is no uncertainty in $\{\varphi(\alpha_i)\}_{i\in[T]}$ given $\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t$, $\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{T}},\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}}$, $\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}}$, $\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in\mathcal{H}\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in\mathcal{H}\\j\in\mathcal{T}\\k\in[K]}}$ from which (60) follows. Finally, (61) holds since uniform distribution maximizes entropy. By combining (51), (61), and (41),

$$
\begin{aligned}
&I(\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{H}};\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{T}},\\
&\quad\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}})\\
&\le(N-T)K\log p+(N-T)TK\frac{d}{M}\log p\\
&\quad+(N-T)TK\frac{d}{M}\log p-(N-T)TK\frac{d}{M}\log p\\
&\quad-(N-T)TK\frac{d}{M}\log p-(N-T)K\log p\\
&=0 \qquad (62)
\end{aligned}
$$

Then, from (62) and the non-negativity of mutual information,

$$
\begin{aligned}
&I(\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{H}};\{\widehat{\mathbf{x}}_{ik}^t\}_{\substack{i\in[N]\\k\in[K]}},\{\phi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\{\psi_{ik}(\alpha_j)\}_{\substack{i\in[N]\\j\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\varphi(\alpha_i)\}_{i\in\mathcal{U}(t)}|\sum_{i\in\mathcal{U}(t)}\overline{\mathbf{x}}_i^t,\{\overline{\mathbf{x}}_i^t,\mathcal{K}_i^t\}_{i\in\mathcal{T}},\{r_{ik}^t\}_{\substack{i\in\mathcal{T}\\k\in[K]}},\\
&\quad\{\mathbf{v}_{ikn}^t,\mathbf{u}_{ikn}^t\}_{\substack{i\in\mathcal{T},\ k\in[K]\\n\in\{M+1,\dots,M+T\}}})=0 \qquad (63)
\end{aligned}
$$

which concludes the proof.

Appendix E
Additional Experimental Details

Our experiments include randomness generated for rand-K sparsification. For the performance comparison of different SA frameworks in terms of model convergence and total communication overhead, to ensure a fair comparison, we use a fixed seed (equal to the index of the training round) in each training round for all frameworks.

The remaining hyperparameters are $E=5$ for the number of local training iterations, $\eta=0.001$ for the learning rate, 25 for the batch size, $q=2^{20}$ for the tuning parameter, and $p=2^{32}-5$ for the finite field size, where the parameters of the masked sparsified gradient along with the corresponding masked locations are transmitted by using 32 bits. The hyperparameters are selected in accordance with the prior works, [1], [26], and we do not require additional hyperparameter tuning. The number of shards is $M=40$. Experiments are run on AMD Ryzen 3960X CPU and NVIDIA RTX4000. For the experiments on the multi-round reconstruction attack, we assign each user a single data sample from CIFAR-10 dataset [51] for consistency with the prior works [4], [23]. For performance evaluation across different frameworks, we perform training on CIFAR-10 and FEMNIST datasets, which has been widely used in literature for similar purpose [26].

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2017, pp. 1–10.

[2] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2019.

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

[4] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients—How easy is it to break privacy in federated learning?" in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020, pp. 1–11.

[5] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via GradInversion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 16332–16341.

[6] Y. Wen, J. Geiping, L. Fowl, M. Goldblum, and T. Goldstein, "Fishing for user data in large-batch federated learning via gradient magnification," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2022, pp. 1–17.

[7] L. H. Fowl, J. Geiping, W. Czaja, M. Goldblum, and T. Goldstein, "Robbing the fed: Directly obtaining private data in federated learning with modified models," in *Proc. 10th Int. Conf. Learn. Represent. (ICLR)*, 2022, pp. 1–25.

[8] K. Bonawitz et al., "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2017, pp. 1175–1191.

[9] J. H. Bell, K. A. Bonawitz, A. Gascón, T. Lepoint, and M. Raykova, "Secure single-server aggregation with (poly) logarithmic overhead," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2020, pp. 1253–1269.

[10] Y. Ma, J. Woods, S. Angel, A. Polychroniadou, and T. Rabin, "Flamingo: Multi-round single-server secure aggregation with applications to private federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2023, pp. 477–496.

[11] Z. Liu, H.-Y. Lin, and Y. Liu, "Long-term privacy-preserving aggregation with user-dynamics for federated learning," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 2398–2412, 2023.

[12] S. U. Stich, J. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2018, pp. 1–12.

[13] D. Alistarh, T. Hoefler, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli, "The convergence of sparsified gradient methods," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 1–11.

[14] P. Jiang and G. Agrawal, "A linear speedup analysis of distributed deep learning with sparse and quantized communication," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2530–2541.

[15] H. Xu et al., "GRACE: A compressed communication framework for distributed machine learning," in *Proc. IEEE 41st Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2021, pp. 561–572.

[16] A. Xu and H. Huang, "Detached error feedback for distributed SGD with random sparsification," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2020, pp. 1–26.

[17] R. Hu, Y. Gong, and Y. Guo, "Federated learning with sparsified model perturbation: Improving accuracy under client-level differential privacy," 2022, *arXiv:2202.07178*.

[18] J. Wang et al., "CocktailSGD: Fine-tuning foundation models over 500Mbps networks," in *Proc. 40th Int. Conf. Mach. Learn. (ICML)*, 2023, pp. 36058–36076.

[19] Y. Lin, S. Han, H. Mao, Y. Wang, and B. Dally, "Deep gradient compression: Reducing the communication bandwidth for distributed training," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–14.

[20] A. Xu, Z. Huo, and H. Huang, "Step-ahead error feedback for distributed training with compressed gradient," in *Proc. 35th AAAI Conf. Artif. Intell. (AAAI)*, 2021, pp. 10478–10486.

[21] G. Yan, T. Li, S.-L. Huang, T. Lan, and L. Song, "AC-SGD: Adaptively compressed SGD for communication-efficient distributed learning," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 9, pp. 2678–2693, Sep. 2022.

[22] A. N. Sahu, A. Dutta, A. M. Abdelmoniem, T. Banerjee, M. Canini, and P. Kalnis, "Rethinking gradient sparsification as total error minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 1–14.

[23] M. Lam, G. Wei, D. Brooks, V. J. Reddi, and M. Mitzenmacher, "Gradient disaggregation: Breaking privacy in federated learning by reconstructing the user participant matrix," in *Proc. 38th Int. Conf. Mach. Learn. (ICML)*, 2021, pp. 1–10.

[24] D. Pasquini, D. Francati, and G. Ateniese, "Eluding secure aggregation in federated learning via model inconsistency," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2022, pp. 2429–2443.

[25] H. Yang, M. Ge, K. Xiang, and J. Li, "Using highly compressed gradients in federated learning for data reconstruction attacks," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 818–830, 2023.

[26] J. So et al., "LightSecAGG: A lightweight and versatile design for secure aggregation in federated learning," in *Proc. Mach. Learn. Syst.*, 2022, pp. 1–27.

[27] J. So, R. E. Ali, B. Güler, J. Jiao, and A. S. Avestimehr, "Securing secure aggregation: Mitigating multi-round privacy leakage in federated learning," in *Proc. AAAI Conf. Artif. Intell.*, 2023, pp. 1–10.

[28] I. Ergün, H. U. Sami, and B. Güler, "Communication-efficient secure aggregation for federated learning," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2022, pp. 3881–3886.

[29] S. Lu, R. Li, W. Liu, C. Guan, and X. Yang, "Top-k sparsification with secure aggregation for privacy-preserving federated learning," *Comput. Secur.*, vol. 124, Jan. 2023, Art. no. 102993.

[30] P. Kairouz, Z. Liu, and T. Steinke, "The distributed discrete Gaussian mechanism for federated learning with secure aggregation," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 5201–5212.

[31] W.-N. Chen, A. Ozgur, and P. Kairouz, "The Poisson binomial mechanism for unbiased federated learning with secure aggregation," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 1–17.

[32] T. Stevens, C. Skalka, C. Vincent, J. Ring, S. Clark, and J. Near, "Efficient differentially private secure aggregation for federated learning via hardness of learning with errors," in *Proc. 31st USENIX Secur. Symp. (USENIX Secur.)*, 2022, pp. 1379–1395.

[33] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, robust, and scalable computation of aggregate statistics," in *Proc. Symp. Netw. Syst. Design Implement.*, 2017, pp. 1–25.

[34] J. So, B. Güler, and A. S. Avestimehr, "Byzantine-resilient secure federated learning," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2168–2181, Jul. 2021.

[35] A. Panda, S. Mahloujifar, A. N. Bhagoji, S. Chakraborty, and P. Mittal, "SparseFed: Mitigating model poisoning attacks in federated learning with sparsification," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 7587–7624.

[36] A. R. Chowdhury, C. Guo, S. Jha, and L. van der Maaten, "EIFFeL: Ensuring integrity for federated learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2022, pp. 2535–2549.

[37] J. Bell et al., "ACORN: Input validation for secure aggregation," Cryptol. ePrint Arch., Paper 2022/1461. [Online]. Available: https://eprint.iacr.org/2022/1461

[38] X. Cao, Z. Zhang, J. Jia, and N. Z. Gong, "FLCert: Provably secure federated learning against poisoning attacks," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 3691–3705, 2022.

[39] N. Wang, Y. Xiao, Y. Chen, Y. Hu, W. Lou, and Y. T. Hou, "FLARE: Defending federated learning against model poisoning attacks via latent space representations," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, 2022, pp. 946–958.

[40] B. Zhao, P. Sun, T. Wang, and K. Jiang, "FedInv: Byzantine-robust federated learning by inversing local model updates," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 1–9.

[41] H. Lycklama, L. Burkhalter, A. Viand, N. Küchler, and A. Hithnawi, "RoFL: Robustness of secure federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2023, pp. 453–476.

[42] M. Rathee, C. Shen, S. Wagh, and R. A. Popa, "ELSA: Secure aggregation for federated learning with malicious actors," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2023, pp. 1961–1979.

[43] J. Le, D. Zhang, X. Lei, L. Jiao, K. Zeng, and X. Liao, "Privacy-preserving federated learning with malicious clients and honest-but-curious servers," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 4329–4344, 2023.

[44] L. Bangalore, M. H. F. Sereshgi, C. Hazay, and M. Venkitasubramaniam, "Flag: A framework for lightweight robust secure aggregation," in *Proc. ACM Asia Conf. Comput. Commun. Secur. (CCS)*, 2023, pp. 14–28.

[45] L. P. Barnes, H. A. Inan, B. Isik, and A. Özgür, "RTop-k: A statistical estimation approach to distributed SGD," *IEEE J. Sel. Areas Inf. Theory*, vol. 1, no. 3, pp. 897–907, Nov. 2020.

[46] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "QSGD: Communication-efficient SGD via gradient quantization and encoding," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 1–12.

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.

[47] S. Vithana and S. Ulukus, "Private read update write (PRUW) in federated submodel learning (FSL): Communication efficient schemes with and without sparsification," *IEEE Trans. Inf. Theory*, vol. 70, no. 2, pp. 1320–1348, Feb. 2024.

[48] S. Vithana and S. Ulukus, "Private read-update-write with controllable information leakage for storage-efficient federated learning with top r sparsification," *IEEE Trans. Inf. Theory*, vol. 70, no. 5, pp. 3669–3692, May 2024.

[49] Z. Jia and S. A. Jafar, "X-secure T-private federated submodel learning with elastic dropout resilience," *IEEE Trans. Inf. Theory*, vol. 68, no. 8, pp. 5418–5439, Aug. 2022.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[51] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, Canada, 2009.

[52] K. S. Kedlaya and C. Umans, "Fast polynomial factorization and modular composition," *SIAM J. Comput.*, vol. 40, no. 6, pp. 1767–1802, Jan. 2011.

[53] S. Caldas et al., "LEAF: A benchmark for federated settings," 2018, *arXiv:1812.01097*.

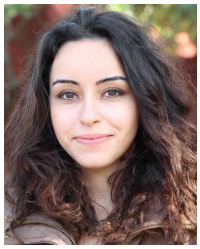
[54] A. R. Elkordy, J. Zhang, Y. H. Ezzeldin, K. Psounis, and A. S. Avestimehr, "How much privacy does federated learning with secure aggregation guarantee?" in *Proc. Priv. Enhancing Technol. (PETS)*, 2023, pp. 1–16.

[55] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[56] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[57] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and S. A. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security, and privacy," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1–11.

Hasin Us Sami (Graduate Student Member, IEEE) received the B.Sc. degree in electrical and electronic engineering from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2019. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of California at Riverside. His research interests include federated and distributed machine learning, information theory, secure and private computing, and wireless networks.



Başak Güler (Member, IEEE) received the B.Sc. degree in electrical and electronics engineering from Middle East Technical University (METU), Ankara, Turkey, and the Ph.D. degree from the Wireless Communications and Networking Laboratory, The Pennsylvania State University, in 2017. From 2018 to 2020, she was a Post-Doctoral Scholar with the University of Southern California. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, University of California at Riverside. Her research interests include information theory, distributed computing, machine learning, and wireless networks. She has received an NSF CAREER Award in 2022.

Authorized licensed use limited to: Univ of Calif Riverside. Downloaded on April 08,2025 at 16:38:01 UTC from IEEE Xplore. Restrictions apply.