Practical Multi-Key Homomorphic Encryption for More Flexible and Efficient Secure Federated Aggregation (preliminary work)

A. Pedrouzo-Ulloa[1,2], Aymen Boudguiga[2], Olive Chakraborty[2], Renaud Sirdey[2], Oana Stan[2], and Martin Zuber[2]

[1] atlanTTic Research Center, Universidade de Vigo
apedrouzo@gts.uvigo.es
[2] CEA-List, Université Paris-Saclay
alberto.pedrouzoulloa@cea.fr, name.surname@cea.fr

Abstract. In this work, we introduce a lightweight communication-efficient multi-key approach suitable for the Federated Averaging rule. By combining secret-key RLWE-based HE, additive secret sharing and PRFs, we reduce approximately by a half the communication cost per party when compared to the usual public-key instantiations, while keeping practical homomorphic aggregation performances. Additionally, for LWE-based instantiations, our approach reduces the communication cost per party from quadratic to linear in terms of the lattice dimension.

1 Introduction

As a protocol for training neural networks (NNs) without explicit sharing of learning data, Federated Learning (FL) has received a lot of attention since its inception around 2017 [8]. In a nutshell, starting from an initial common NN model, the FL protocol iteratively builds a global model by having the training data owners (i.e., the clients) locally updating the model by the partial execution of a training algorithm, and then, letting a central server aggregating these updates to generate the common model for the next round. FL can be instantiated in the *cross-device* setting, where a model is built from the data of many intermittently available and computationally constrained devices, or *cross-silo*, in which a model is built from the training sets of a reduced number of servers which are always available and computationally powerful. *This paper focuses primarily on the latter of these two settings.*

Federated Learning was initially proposed as a solution for avoiding the prohibitive communication cost of getting training data out of many user devices as well as for ensuring training data privacy. However, it is now well-known that the baseline FL protocol is not sufficient for guaranteeing the privacy of a client's training data, as the NN parameters updates exchanged throughout the protocol (seen by both the aggregation server and the other clients) leak a lot of information. As a consequence, in recent years, FL has been more deeply investigated with respect to training data privacy. In this context, performing updates aggregation by means of Homomorphic Encryption (HE) has been investigated from the viewpoint of countering the confidentiality threats *from the server* on the clients' training data. Yet, from an HE perspective, previous works (e.g. [10, 7]) have focused primarily on performance issues and implicitly assumed overly simple deployment scenarios e.g. with all the encrypted-domain calculations performed under the same HE keys and all clients sharing the same decryption key in a honest-but-curious setting. In this paper, *we introduce a lightweight communication-efficient multi-key approach suitable for the Federated Averaging rule*, allowing each client to use its own key for encryption at each round and the effective subset of clients which participated in a round to collectively decrypt the aggregated updates to further proceed with the next protocol iteration.

Main Contributions: Our proposed aggregation method is secure in the semi-honest setting and works under the Common Reference String (CRS) model. By combining secret-key RLWE-based HE and PRFs, we reduce approximately by a half the communication cost per party when comparing with its public-key counterpart. This improvement is more significant for LWE-based instantiations, in which thanks to the removal of the mask component for secret-key LWE samples, *the communication cost per party is reduced from quadratic into linear* in terms of the lattice dimension n. A high-level comparison among different available aggregation methods and ours is included in Table 1 for a FL training of $N_{\mathsf{AggRounds}}$ rounds with L participants.

Protection Method	Comm. Cost	Security Issues + other considerations	Protected inputs
Additive secret sharing	$\mathcal{O}(N_{\mathsf{AggRounds}} \cdot L^2)$	Avoids collusion with aggregator Needs new shares per each round	Same as plaintext space
Public-Key HE (single-key)	$\mathcal{O}(N_{\mathsf{AggRounds}} \cdot L)$	Collusion with aggregator Who holds the secret key?	Public-Key ctxts. (2 pol. elem. if RLWE)
Threshold HE	$\mathcal{O}(N_{\mathsf{AggRounds}} \cdot L) +$ CostPKGSetup	Avoids collusion with aggregator Requires to generate a new public key for users not collaborating	Public-Key ctxts. (2 pol. elem. if RLWE)
Proposed Method	$\mathcal{O}(N_{\mathsf{AggRounds}} \cdot L) + (L-1)^2$	Avoids collusion with aggregator 1 extra round to manage non-participating users in decryption	Secret-Key Ctxts. (1 pol. elem. if RLWE)

Table 1. Comparison between different protection methods for secure aggregation in FL.

Threat Model: In the semi-honest (or honest-but-curious) model, many entities ($\mathsf{E_1}, \ldots, \mathsf{E_L}$), having as secret information ($\mathsf{s_1}, \ldots, \mathsf{s_L}$), participate in a protocol P to compute a function $\mathsf{F}(\mathsf{s_1}, \ldots, \mathsf{s_L})$. Each entity $\mathsf{E}_{\mathsf{i:i\in[\![1,L]\!]}}$ tries to gather as much information as possible, but do not deviate from the protocol P (i.e., $\mathsf{E}_{\mathsf{i:i\in[\![1,L]\!]}}$ will try to recover information about the secrets $\mathsf{s}_{\mathsf{j:j\neq i}}$ of other entities). Then we say that P is secure in the semi-honest model if each $\mathsf{E}_{\mathsf{i:i\in[\![1,L]\!]}}$ has no other information than $\mathsf{F}(\mathsf{s_1}, \ldots, \mathsf{s_L})$ at the end of the protocol. Note that *assuming semi-honest adversaries in* P *does not guarantee that no parties will collude* [6].

In this work, we provide a solution for secure aggregation in FL with a semi-honest server (and up to $L-1$ semi-honest Data Owners if paired with differential privacy techniques). First, we assume a CRS model, where all Data Owners (DOs) have access to the same PRF. Using the same PRF with the same seed ensures that all DOs will generate the same mask a each round for their distinct RLWE samples. Second, we assume that DOs will have distinct secret keys. That is, each DO will encrypt her own data m_i with her own secret key s_i (but using the same mask a per round shared with other DOs). Finally, during the aggregation, the semi-honest server will compute the encrypted sum $\sum_i m_i$ with the aggregated secret key $\sum_i s_i$.

For a more realistic FL setting, our secure aggregation scheme can be seamlessly coupled with differential privacy techniques, as in [10], to cover threats coming from $L-1$ colluding semi-honest DOs (out of L) that aim at gathering information about the remaining DO data.

2 Building Blocks

Additive Secret Shares of Zero. Given L Data Owners (DOs), we can generate L uniformly random additive shares satisfying that their addition is equal to zero. The protocol is as follows:

1. The i-th DO ($\forall i$) generates a set of $(L-1)$ uniformly random elements $r_{i,j}$ for all $j \neq i$. Next, the i-th DO computes $r_{i,i} = -(\sum_{j:j\neq i} r_{i,j})$. All $r_{i,j}$ satisfy the relation $\sum_j r_{i,j} = 0$.
2. The i-th DO ($\forall i$) sends, $r_{i,j}$ to the j-th party, $\forall j$.
3. The i-th DO ($\forall i$) computes $\mathsf{share}_i = r^{(i)} = \sum_j r_{i,j}$.

Rounding polynomial elements. Let $\lfloor \boldsymbol{a} \rceil_p$ be the scaling and rounding of each coefficient of $\boldsymbol{a} \in R_q^N$ to its nearest integer, where R_q denotes the quotient polynomial ring $\mathbb{Z}_q[x]/(x^n+1)$.

Lemma 1 (Lemma 1 **[3]).** *Let* $p|q$*,* $\boldsymbol{x} \leftarrow R_q^N$ *and* $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{e} \bmod q$ *for some* $\boldsymbol{e} \in R_q^N$ *with* $\|\boldsymbol{e}\|_\infty < B < q/p$*. Then* $\Pr\left(\lfloor \boldsymbol{y} \rceil_p \neq \lfloor \boldsymbol{x} \rceil_p \bmod p\right) \leq \frac{2npNB}{q}$.

This lemma is used in our scheme (see Section 3) to remove the error term associated to each encryption. Given $(a, b = as+e+q/p\cdot m)$, we compute $\lfloor b \rceil_p = \lfloor as + e \rceil_p + m$ which, by Lemma 1, is equal to $\lfloor as \rceil_p + m$ with a certain probability $\Pr(\mathsf{Ev})$. The upper bound of the probability $\Pr(\mathsf{Ev})$ depends inversely on q.

Distributed Decryption. Given $(a, b = as + e) \in R_q^2$ s.t. $s = \sum_{i=1}^{L} s_i$ where all $s_i \in R_q$, applying modulus switching [1] from q into p, we get $\left(\lfloor a \rceil_p, \lfloor b \rceil_p = \left\lfloor \lfloor a \rceil_p s + (p/q \cdot a - \lfloor a \rceil_p) \cdot s + p/q \cdot e \right\rceil\right)$. By applying Lemma 1, the error therm e is removed with a certain probability, finally having:

$$\lfloor b \rceil_p = \left\lfloor a \underbrace{s}_{\sum_i s_i} \right\rceil_p = \left\lfloor \lfloor a \rceil_p \underbrace{s}_{\sum_i s_i} + \underbrace{(p/q \cdot a - \lfloor a \rceil_p)}_{e_a} \cdot \underbrace{s}_{\sum_i s_i} \right\rceil . \tag{1}$$

From equation (1), we can obtain the magnitude of the difference $e_{\mathsf{distributed}} = \lfloor as \rceil_p - \sum_i \lfloor as_i \rceil_p$. This term must be removed for the correctness of the distributed decryption protocol executed after each aggregation round. Assuming that each s_i is bounded by B, and due to $\|e_a\|_\infty < 1/2$, the magnitude of this remaining error term is bounded by nLB.

3 Proposed scheme for secure aggregation

Current works making use of Threshold RLWE-based HE [9, 2] define a collaborative key setup phase to generate a joint public key pk associated to several secret keys s_i. This results in a pair $(\mathsf{sk} = s, \mathsf{pk} = (a, as + e))$, where each i-th DO has a s_i s.t., $\sum_{i=1}^{L} s_i = s$.

We optimize this primitive for the case of secure federated average aggregation: by assuming the CRS model, ciphertexts can be aggregated on-the-fly, similarly to real "multi-key" HE schemes. We include next a high-level description of our proposed secure aggregation primitive.

3.1 High-level description

In the CRS model, each party (a.k.a Data Owner, DO) has access to a common uniformly random polynomial term a per round. Additionally, we assume that all DOs have run the protocol described in Section 2 to generate uniformly random polynomial shares. As a consequence, each i-th DO holds $\mathsf{share}_i = r^{(i)}$. Then, each secure aggregation round is as follows:

1. DOs encrypt their inputs: The i-th DO ($\forall i$) encrypts its model update m_i with its secret key s_i as $(a, b_i) = (a, a(s_i + r^{(i)}) + e_i + q/p \cdot m_i))$, which can be compressed by a half by only sending b_i because a is publicly known (i.e., computable with $\mathsf{PRF}_K(T)$ for the T-th round).
2. Aggregation step: After receiving all b_i polynomial terms, a semi-honest aggregator can directly compute:

$$(a, \sum_i b_i) = (a, b = a(s + \underbrace{\sum_i r^{(i)}}_{0}) + e) = (a, b = a \underbrace{s}_{\sum_i s_i} + \underbrace{e}_{\sum_i e_i} + q/p \cdot \underbrace{m}_{\sum_i m_i}),$$

   which corresponds to $\mathsf{Enc}(\mathsf{sk} = s, m)$, the desired encrypted aggregation. Finally, the aggregator sends back $\mathsf{share}^{(\mathsf{agg})} = \lfloor b \rceil_{p'}$ to the DOs.
3. Distributed decryption: Given $\mathsf{Enc}(\mathsf{sk} = s, m)$ s.t. $s = \sum_i s_i$. This protocol is as follows:
   (a) The i-th DO ($\forall i$) computes $\mathsf{share}^{(i)} = \lfloor as_i \rceil_{p'}$ and makes it available to the other DOs.
   (b) All DOs compute $\left\lfloor \mathsf{share}^{(\mathsf{agg})} - \sum_i \mathsf{share}^{(i)} \right\rceil_p$, which is equal to m with probability higher than $1 - 2^{-\kappa}$, whenever the encryption parameters are chosen according to Section 4.

Semantic Security: Given a pair of independent and uniformly random terms $a, u \leftarrow R_q$, then if an algorithm $\mathcal{A}(a, \lfloor u \rceil_{p'}, \lfloor as_i \rceil_{p'})$ can distinguish between $(a, \lfloor u \rceil_{p'})$ and $(a, \lfloor as_i \rceil_{p'})$, $\mathcal{A}$ can be used to distinguish with probability $1 - 2^{-\kappa}$ the RLWE sample $(a, as_i + e)$ from the pair (a, u).

From RLWE to M-LWE and LWE: As the a polynomials in the RLWE samples $(a, b = as + e)$ are generated under the CRS model with a $\mathsf{PRF}_K(\cdot)$, keys could be alternatively defined under either M-LWE or LWE assumptions without adding extra communication/computation costs for aggregation. On the one hand, we can work under the LWE assumption with the same communication cost as its RLWE counterpart, and hence removing the quadratic communication/computation overhead of public-key LWE-based solutions. On the other hand, there is an overhead for encryption and also an increase in the number of calls to $\mathsf{PRF}_K(\cdot)$ by a factor n.

4 Example instantiations and additional features

Communication costs: Table 2 includes the communication cost per party of the secure aggregation protocol. We assume that the number of model parameters $N_{\mathsf{ModelParam}}$ is high enough.

Input per DO	Decryption share per DO	Aggregator output	Decrypted result
$N_{\mathsf{ModelParam}} \cdot \log_2 q$	$N_{\mathsf{ModelParam}} \cdot \log_2 p'$	$N_{\mathsf{ModelParam}} \cdot \log_2 p'$	$N_{\mathsf{ModelParam}} \cdot \log_2 p$

Table 2. Communication costs per party in each aggregation round.

Protocol parameters $\{p, p', q, n\}$: If the event Ev represents the probability of having at least a decryption failure during $N_{\mathsf{AggRounds}}$ consecutive rounds, then by applying Lemma 1, we have:

$$\Pr(\mathsf{Ev}) \leq \frac{2 \cdot n \cdot N_{\mathsf{AggRounds}} \cdot N_{\mathsf{Ctxts.PerRound}} \cdot p' \cdot B_{\mathsf{Agg}}}{q},$$

in which bounding by $\Pr(\mathsf{Ev}) \leq 2^{-\kappa}$ with parameter κ, we have that q satisfies:

$$q \geq 2 \cdot n \cdot N_{\mathsf{AggRounds}} \cdot N_{\mathsf{Ctxts.PerRound}} \cdot p' \cdot B_{\mathsf{Agg}} \cdot 2^{\kappa}. \tag{2}$$

Finally, a last rounding step is applied after aggregating the shares, which requires $\frac{nB_{\mathsf{Agg}}p}{p'} < \frac{1}{2}$ whenever each s_i is bounded by $B_{\mathsf{Init}} = \frac{B_{\mathsf{Agg}}}{L}$. This gives the following lower bound for q:

$$q \geq 4 \cdot n^2 \cdot N_{\mathsf{AggRounds}} \cdot N_{\mathsf{Ctxts.PerRound}} \cdot p \cdot L^2 \cdot B_{\mathsf{Init}}^2 \cdot 2^{\kappa}. \tag{3}$$

Example of parameters for Federated Learning (FL): Table 3 includes two different sets of protocol parameters based on the ones provided in [10] for training in an FL context. To fix ideas in terms of performance costs, on the FEMNIST dataset [5, 4] with a 486,654 parameters model and 1000 clients, we obtain (HE-domain) aggregation times of around 27 secs for an overall time per learning round of around 10 mins (i.e., including the local training done on the clients), hence a $\approx 5\%$ overhead. This is following other studies [10] using parameters similar to those in Table 3 in the single-key setting.

Parameter	Par. set 1	Par. set 2
$\{n, N_{\mathsf{AggRounds}}, N_{\mathsf{Parties}}\}$	$\{16384, 256, 2^{12}\}$	$\{16384, 2^{20}, 2^{20}\}$
$\{N_{\mathsf{ModelParam}}, N_{\mathsf{Ctxts.PerRound}} = \lceil \frac{N_{\mathsf{ModelParam}}}{n} \rceil\}$	$\{524288, 32\}$	$\{524288, 32\}$
$\{p, p', q\}$	$\{32, 65, 242\}$ bits	$\{32, 73, 270\}$ bits
$\{\text{bit security}, \kappa\}$	$\{\approx 256, 128\}$	$\{> 192, 128\}$
$\{B_{\mathsf{init}}, B_{\mathsf{final}}\}$	$\{2^5, 2^{17}\}$	$\{2^5, 2^{25}\}$

Table 3. Example parameter sets for FL [10] (Par. set 1, approx. to [10]) and (Par. set 2, bigger than [10]).

Session keys: It is easy to define session keys, as the s_i terms of $s_i + r^{(i)}$ can be changed in each aggregation round. Alternatively, other options are possible, e.g., by using $s_i + u \cdot r^{(i)}$, where u is a uniformly random element changed each round and generated by $\mathsf{PRF}_K(\cdot)$.

Flexible decryption structure: If a DO does not collaborate for decryption, the aggregator and the rest of DOs are able to "fix" their encryptions with an extra communication round, enabling: (1) to remove the model update of the missing party in the aggregation result, and (2) to decrypt under a different subset of secret keys.

General Linear Combination of Model Parameter Updates: The aggregation can be generalized to work for any linear combination of encrypted model updates. For this purpose, the generated additive shares in Section 2 have to *satisfy the zero equality for the desired linear combination.*

5 Conclusions

This work presents a lightweight aggregation protocol for the federated learning under the assumption of semi-honest parties, with less bandwidth requirements than existing protocols and a more flexible setup. In the future, we intend to implement and test this secure aggregation approach when deployed for a practical use case of Federated Learning (such as the one from [7]). Moreover, we want to go beyond the assumption of honest-but-curious data owners by extending the protocol with methods for verifiable encryption/decryption.

Acknowledgements

This work was partially funded by the European Union's Horizon Europe Framework Programme for Research and Innovation Action under project TRUMPET (proj. no. 101070038), by the European Regional Development Fund (FEDER) and Xunta de Galicia under project "Grupos de Referencia Competitiva" (ED431C 2021/47), by FEDER and MCIN/AEI under project FELDSPAR (TED2021-130624B-C21).

This work was funded in part by the EU-funded ENCRYPT under the Horizon Europe Framework Programme under grant agreement Nr. 101070670 as well as by Agence Nationale de la Recherche (France) under grant Plan France 2030/ANR-22-PECY-0003 (SecureCompute).

The first author is currently a visiting researcher at CEA-List, Université Paris-Saclay, being also funded by the European Union "NextGenerationEU/PRTR" by means of a Margarita Salas grant of the Universidade de Vigo.

Funded by the European Union. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

References

1. M. R. Albrecht, J. Faugère, R. Fitzpatrick, and L. Perret. Lazy modulus switching for the BKW algorithm on LWE. In *PKC 2014*, volume 8383 of *Lecture Notes in Computer Science*, pages 429–445. Springer, 2014.
2. A. Aloufi, P. Hu, Y. Song, and K. E. Lauter. Computing blindfolded on data homomorphically encrypted under multiple keys: A survey. *ACM Comput. Surv.*, 54(9):195:1–195:37, 2022.
3. C. Baum, D. Escudero, A. Pedrouzo-Ulloa, P. Scholl, and J. R. Troncoso-Pastoriza. Efficient protocols for oblivious linear function evaluation from ring-lwe. *J. Comput. Secur.*, 30(1):39–78, 2022.
4. S. Caldas, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar. LEAF: A benchmark for federated settings. *CoRR*, abs/1812.01097, 2018.
5. G. Cohen, S. Afshar, J. Tapson, and A. van Schaik. EMNIST: extending MNIST to handwritten letters. In *IJCNN 2017*, pages 2921–2926. IEEE, 2017.
6. Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. Cryptology ePrint Archive, Paper 2008/197, 2008. https://eprint.iacr.org/2008/197.
7. A. Madi, O. Stan, A. Mayoue, A. Grivet-Sébert, C. Gouy-Pailler, and R. Sirdey. A secure federated learning framework using homomorphic encryption and verifiable computing. pages 1–8, 2021.
8. H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. 2017.
9. C. Mouchet, J. R. Troncoso-Pastoriza, J. Bossuat, and J. Hubaux. Multiparty homomorphic encryption from ring-learning-with-errors. *Proc. Priv. Enhancing Technol.*, 2021(4):291–311, 2021.
10. A. G. Sébert, R. Sirdey, O. Stan, and C. Gouy-Pailler. Protecting data from all parties: Combining FHE and DP in federated learning. *CoRR*, abs/2205.04330, 2022.