Comments on "Privacy-Enhanced Federated Learning Against Poisoning Adversaries"

Thomas Schneider, Ajith Suresh, and Hossein Yalame

***Abstract*—Liu et al. (DOI 10.1109/TIFS.2021.3108434) recently proposed a privacy-enhanced framework named PEFL to efficiently detect poisoning behaviours in Federated Learning (FL) using homomorphic encryption. In this article, we show that PEFL does not preserve privacy. In particular, we illustrate that PEFL reveals the entire gradient vector of all users in clear to one of the participating entities, thereby violating privacy. Furthermore, we clearly show that an immediate fix for this issue is still insufficient to achieve privacy by pointing out multiple flaws in the proposed system.**

***Index Terms*—Federated Learning (FL), Homomorphic Encryption, Poisoning and Inference Attacks, Data Privacy.**

I. Introduction

Federated Learning (FL) is a new distributed machine learning approach that allows multiple entities to jointly train a model without sharing their private and sensitive local datasets with others. In FL, clients locally train models using their local training data, then send model updates to a central aggregator, which merges them into a global model. FL is used in a variety of applications such as word prediction for mobile keyboards in GBoard [1] and medical imaging [2]. Despite its benefits, FL has been shown to be susceptible to model poisoning [3] and inference attacks [4]. In model poisoning attacks, an adversary injects poisoned model updates by corrupting a subset of clients, with which the adversary can compromise the user's data privacy as well as the FL model's integrity [5], [6]. The recent work of Liu et al. [7] proposed a privacy-enhanced framework called PEFL to detect poisoning behaviors in FL. PEFL aims to prevent malicious users from inferring memberships by uploading malicious gradients and semi-honest servers from invading users' privacy. Furthermore, PEFL claims to be the first effort to detect poisoning behaviors in FL while using ciphertext and uses homomorphic encryption (HE) as the underlying technology.

In this article, we have a closer look at the PEFL system of [7] and identify multiple privacy vulnerabilities. In particular, we show that each of the three main protocols in PEFL – SecMed, SecPear, and SecAgg, reveals significant information about the user's gradients to one of the computing servers thereby compromising privacy. Furthermore, we demonstrate that combining information from the protocols enables a computation server to learn the gradient vectors of all users in clear, thereby breaking the PEFL system.

Thomas Schneider, Ajith Suresh, and Hossein Yalame are members of the Cryptography and Privacy Engineering Group (ENCRYPTO) at the Technical University of Darmstadt, Germany. E-Mail: {schneider,suresh,yalame}@encrypto.cs.tu-darmstadt.de

II. Liu et al.'s protocols are not private

In this section, we revisit the Privacy Enhanced Federated Learning (PEFL) system in [7], but with our notations for clarity. We begin with an overview of PEFL's four entities:

- *Key Generation Center (KGC):* Trusted entity managing public and private keys (pk, sk) for HE.
- *Data Owners (U_x):* Each data owner U_x, for $x \in [\mathsf{m}]$, locally trains the local model on their data and computes the gradient vector $\vec{\mathbf{g}}_x = \{g_x^1, \ldots, g_x^{\mathsf{n}}\}$. Here, m denotes the total number of users in the system and n denotes the dimension of the gradient vector.
- *Service Provider (SP):* SP receives all gradients submitted by data owners and aggregates them (usually by averaging) to produce an optimized global model.
- *Cloud Platform (CP):* CP assists SP in the computations and operates on a pay-per-use basis.

The threat model assumes that SP and CP are both semi-honest, whereas data owners can be maliciously corrupt. Furthermore, the four entities mentioned above are non-colluding.

We now examine the PEFL system in-depth, focusing on the amount of information visible to each entity. More specifically, we are interested in how much information the cloud platform (CP) learns from the protocol execution.

A. Calculation of Gradients

The protocol begins with each user U_x training the model locally and obtaining the corresponding gradient vector $\vec{\mathbf{g}}_x = \{g_x^1, \ldots, g_x^{\mathsf{n}}\}$. User U_x then encrypts and sends the gradient vector to SP using CP's public key pk_c. As shown in (1), the gradient vectors corresponding to all users can be viewed as a matrix $\vec{\mathbf{G}}_{\mathsf{m}\times\mathsf{n}}$.

$$\vec{\mathbf{G}}_{\mathsf{m}\times\mathsf{n}} = \begin{array}{c} \\ U_1 \\ U_2 \\ \vdots \\ U_{\mathsf{m}} \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 \cdots & c_i \cdots & c_{\mathsf{n}-1} & c_{\mathsf{n}} \end{array} \\ \begin{pmatrix} g_1^1 & g_1^2 & \cdots & g_1^i & \cdots & g_1^{\mathsf{n}-1} & g_1^{\mathsf{n}} \\ g_2^1 & g_2^2 & \cdots & g_2^i & \cdots & g_2^{\mathsf{n}-1} & g_2^{\mathsf{n}} \\ \vdots & & \ddots & \vdots & \ddots & & \vdots \\ g_{\mathsf{m}}^1 & g_{\mathsf{m}}^2 & \cdots & g_{\mathsf{m}}^i & \cdots & g_{\mathsf{m}}^{\mathsf{n}-1} & g_{\mathsf{m}}^{\mathsf{n}} \end{pmatrix} \end{array}. \quad (1)$$

B. Median Computation using SecMed

The SP and CP use the SecMed algorithm (cf. Figure 4 in [7]) to compute the median value for each of the n coordinates. SP sends $g_j^i + r_i$ to CP for each user U_j corresponding to a coordinate c_i, where r_i denotes a random pad sampled by SP for each coordinate c_i (but the same for all users). The CP decrypts and computes the medians based on these padded values. The CP then encrypts the medians before sending them to the SP. Finally, the SP removes the pad r_i to achieve the desired results $\vec{\mathbf{g}}_y$ by utilizing the underlying encryption scheme's homomorphic property.

Leakage: The view of CP while executing SecMed is consolidated in the matrix $\vec{\mathbf{V}}_{\mathsf{SecMed}}$:

$$\vec{\mathbf{V}}_{\mathsf{SecMed}} = \begin{array}{c} \\ U_1 \\ \vdots \\ U_{\mathsf{m}} \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & \cdots & c_i & \cdots & c_{\mathsf{n}} \end{array} \\ \begin{pmatrix} g_1^1 + r_1 & \cdots & g_1^i + r_i & \cdots & g_1^{\mathsf{n}} + r_{\mathsf{n}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ g_{\mathsf{m}}^1 + r_1 & \cdots & g_{\mathsf{m}}^i + r_i & \cdots & g_{\mathsf{m}}^{\mathsf{n}} + r_{\mathsf{n}} \end{pmatrix} \end{array}. \quad (2)$$

We observe that for each coordinate c_i, CP learns a "shifted" distribution of gradients in clear across all users. This is clearly a violation of privacy [5], [6] because it leaks a lot more information to CP and thus does not meet the design goal of 'Privacy' claimed in [7]. The main source of the leakage is that SP uses *same random pad r_i for all users* with respect to a coordinate c_i. While the aforementioned leakage could be prevented by using different random pads across users, we emphasize that the use of the same pad is unavoidable for the SecMed algorithm to remain correct. In detail, the median of g_j^i values is calculated by first computing the median of $g_j^i + r_i$, then removing r_i from the result. This requires that the same r_i value be associated with each g_j^i value, or the computation's correctness will be violated.

C. Computing Pearson correlation coefficient using SecPear

Once the coordinate-wise medians are computed, the next step in PEFL is to calculate the Pearson correlation coefficient $\rho_{x,y}$ between the coordinate-wise medians $\vec{\mathbf{g}}_y$ and the gradient of the user U_x. This is achieved via the SecPear protocol (cf. Figure 5 in [7]) where SP communicates $\vec{\mathbf{g}}_x \cdot p_x$ and $\vec{\mathbf{g}}_y \cdot p_y$ to CP. The view of CP in SecPear with respect to $\vec{\mathbf{G}}_{\mathsf{m}\times\mathsf{n}}$ is $\vec{\mathbf{V}}_{\mathsf{SecPear}}$:

$$\vec{\mathbf{V}}_{\mathsf{SecPear}} = \begin{array}{c} \\ U_1 \\ \vdots \\ U_{\mathsf{m}} \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & \cdots & c_i & \cdots & c_{\mathsf{n}} \end{array} \\ \begin{pmatrix} g_1^1 \cdot p_1 & \cdots & g_1^i \cdot p_1 & \cdots & g_1^{\mathsf{n}} \cdot p_1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ g_{\mathsf{m}}^1 \cdot p_{\mathsf{m}} & \cdots & g_{\mathsf{m}}^i \cdot p_{\mathsf{m}} & \cdots & g_{\mathsf{m}}^{\mathsf{n}} \cdot p_{\mathsf{m}} \end{pmatrix} \end{array}. \quad (3)$$

Leakage: Similar to the problem with SecMed above, here CP learns the correlation between each coordinate in the gradient vector $\vec{\mathbf{g}}_x$. SP uses the *same random pad p_x for all coordinates*, which causes leakage. However, using different pads for the coordinates does not address the issue since the use of the same pad is required for the SecPear algorithm to remain correct (cf. Proposition 1 in [7]). More elaborately, for $d_x = \vec{\mathbf{g}}_x \cdot p_x$ and $d_y = \vec{\mathbf{g}}_y \cdot p_y$, computation of $\rho_{x,y}$ involves computing the covariance $Cov(d_x, d_y)$ and the standard deviations $\sigma(d_x)$ and $\sigma(d_y)$. As shown in Proposition 1 in [7], the correctness of $\rho_{x,y}$ relies on the following observations:

$$Cov(d_x, d_y) = p_x p_y \cdot Cov(\vec{\mathbf{g}}_x, \vec{\mathbf{g}}_y),$$
$$\sigma(d_x) = p_x \cdot \sigma(\vec{\mathbf{g}}_x) \ , \ \sigma(d_y) = p_y \cdot \sigma(\vec{\mathbf{g}}_y).$$

If different pads are used for the coordinates, the above relations do not hold, and hence $\rho_{d_x,d_y} = \frac{Cov(d_x,d_y)}{\sigma(d_x)\sigma(d_y)} \neq \rho_{x,y}$.

D. Aggregating the gradients using SecAgg

SecAgg, the final stage in PEFL, aggregates the gradients after scaling them with a factor based on the Pearson correlation coefficient calculated in SecPear. SP communicates $\vec{\mathbf{g}}_x + s_x$ to CP for this purpose, as shown in $\vec{\mathbf{V}}_{\mathsf{SecAgg}}$:

$$\vec{\mathbf{V}}_{\mathsf{SecAgg}} = \begin{array}{c} \\ U_1 \\ \vdots \\ U_{\mathsf{m}} \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & \cdots & c_i & \cdots & c_{\mathsf{n}} \end{array} \\ \begin{pmatrix} g_1^1 + s_1 & \cdots & g_1^i + s_1 & \cdots & g_1^{\mathsf{n}} + s_1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ g_{\mathsf{m}}^1 + s_{\mathsf{m}} & \cdots & g_{\mathsf{m}}^i + s_{\mathsf{m}} & \cdots & g_{\mathsf{m}}^{\mathsf{n}} + s_{\mathsf{m}} \end{pmatrix} \end{array}. \quad (4)$$

Leakage: Again, similar to SecMed, $\vec{\mathbf{V}}_{\mathsf{SecAgg}}$ reveals a "shifted" distribution of each user's gradient values across all the coordinates to CP. When combining information from $\vec{\mathbf{V}}_{\mathsf{SecPear}}$ (3) and $\vec{\mathbf{V}}_{\mathsf{SecAgg}}$ (4), a more significant leakage occurs. Consider the gradient at coordinates i, j for user U_x. From $\vec{\mathbf{V}}_{\mathsf{SecPear}}$, CP learns $a_1 = g_x^i \cdot p_x$ and $a_2 = g_x^j \cdot p_x = (g_x^i \cdot \delta_x^{ij}) \cdot p_x$ where $\delta_x^{ij} = g_x^j / g_x^i$. Similarly, CP learns $b_1 = g_x^i + s_x$ and $b_2 = g_x^j + s_x = (g_x^i + \Delta_{ij}^x) + s_x$ from $\vec{\mathbf{V}}_{\mathsf{SecAgg}}$, where $\Delta_{ij}^x = g_x^j - g_x^i$. Given that CP can compute both δ_x^{ij} and Δ_{ij}^x in clear, CP learns g_x^i and g_x^j by solving the equations. For instance, $a_2 = (g_x^i + \Delta_{ij}^x) \cdot p_x = a_1 + \Delta_{ij}^x \cdot p_x$ reveals p_x. Using this method, CP learns the entire gradient matrix $\vec{\mathbf{G}}_{\mathsf{m}\times\mathsf{n}}$, thereby breaching the PEFL system's privacy.

E. Practical and probabilistic attacks

Another practical attack on PEFL would be to allow CP to register as an honest user $U_{\mathsf{m}+1}$ in the PEFL system and submit its gradients. This action does not violate the semi-honest assumption of CP in the PEFL threat model and may represent scenarios in which CP has some side channel information about some user gradients. Knowing $\vec{\mathbf{g}}_{\mathsf{m}+1}$, CP learns r_i for all $i \in [\mathsf{n}]$ and hence the gradient matrix $\vec{\mathbf{G}}_{\mathsf{m}\times\mathsf{n}}$ given in (1) in clear from (2) and thereby breaking the privacy of the entire PEFL system.

Considering the matrices $\vec{\mathbf{V}}_{\mathsf{SecMed}}$ and $\vec{\mathbf{V}}_{\mathsf{SecPear}}$ together, we note that it is sufficient for CP to be aware of just one gradient, say g_x^i, to compromise the system's privacy. In particular, g_x^i will allow CP to learn the random pad r_i corresponding to the i-th coordinate c_i in $\vec{\mathbf{V}}_{\mathsf{SecMed}}$, revealing the gradients of all users at that coordinate. Similarly, knowing g_x^i allows CP to learn the random pad p_x corresponding to user U_x in $\vec{\mathbf{V}}_{\mathsf{SecPear}}$ and reveals the gradient vector $\vec{\mathbf{g}}_x$ to CP in clear. CP will now learn the entire gradient matrix $\vec{\mathbf{G}}_{\mathsf{m}\times\mathsf{n}}$ by combining the information from $\vec{\mathbf{V}}_{\mathsf{SecMed}}$ and $\vec{\mathbf{V}}_{\mathsf{SecPear}}$.

Finally, we note that CP can launch probabilistic attacks by looking for similar values in the matrices $\vec{\mathbf{V}}_{\mathsf{SecMed}}$ and $\vec{\mathbf{V}}_{\mathsf{SecPear}}$ and attempting to correlate the random pads. This is possible in PEFL because CP is aware of the correlation between different rows of $\vec{\mathbf{V}}_{\mathsf{SecMed}}$ as well as columns of $\vec{\mathbf{V}}_{\mathsf{SecPear}}$.

ACKNOWLEDGEMENTS

This project received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No.850990 PSOTI) and was co-funded by the Deutsche Forschungsgemeinschaft (DFG) – SFB1119 CROSSING/236615297 and GRK2050 Privacy & Trust/251805230.

REFERENCES

[1] B. McMahan and D. Ramage, "Federated learning: Collaborative Machine Learning without Centralized Training Data," in *Google AI*, 2017.

[2] M. Sheller, A. Reina, B. Edwards, and J. Martin, "Federated Learning for Medical Imaging," in *Intel AI*, 2018.

[3] T. D. Nguyen, P. Rieger, H. Chen, H. Yalame, H. Möllering, H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, S. Zeitouni, F. Koushanfar, A.-R. Sadeghi, and T. Schneider, "FLAME: Taming Backdoors in Federated Learning," in *USENIX Security*, 2022.

[4] H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, H. Möllering, T. D. Nguyen, P. Rieger, A. R. Sadeghi, T. Schneider, H. Yalame, and S. Zeitouni, "SAFELearn: Secure Aggregation for private FEderated Learning," in *DLS*, 2021.

[5] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning," in *CCS*, 2017.

[6] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting Gradients - How easy is it to break privacy in federated learning?" in *NeurIPS*, 2020.

[7] X. Liu, H. Li, G. Xu, Z. Chen, X. Huang, and R. Lu, "Privacy-Enhanced Federated Learning Against Poisoning Adversaries," *IEEE TIFS*, 2021.