Verifiable cross-silo federated learning

Aleksei Korneev, Jan Ramon

Verifiable cross-silo federated learning

Aleksei Korneev[1] and Jan Ramon[1]

[1]University of Lille, Inria Lille, Magnet Team, CNRS, UMR 9189 CRIStAL

Centre de Recherche en Informatique, Signal et Automatique de Lille

Funded by the European Union

INTRODUCTION

- Federated Learning (FL) allows training ML models with data distributed across multiple devices;
- Malicious agents may attempt to disturb the training procedure in order to obtain certain benefits (e.g., a biased result or a reduction in computational load);
- To address this problem, there is recently growing interest in developing verifiable protocols, where one can check that parties do not deviate from the procedure;
- In this paper,
  - we conduct an analysis of verifiable FL protocols while studying specific challenges of the cross-silo setting;
  - we propose a new taxonomy of existing verifiable cross-silo FL protocols while analyzing their efficiency and threat models;
  - we discuss future challenges and identify research gaps.

Keywords— Federated learning, cross-silo FL, verification, verifiable protocols

BACKGROUND

Cross-silo FL properties

While analyzing the suitability of various algorithms for the cross-silo FL setting, we rely on the assumption that the setting possesses the following properties:

1. the number of participants is moderate (almost several thousands);
2. all participants have an incentive to care about their reputation;
3. all participants agree on the model to be trained (type of calculations to be executed);
4. DOs possess computationally sufficiently powerful equipment;

Verifiable FL

In the scope of this paper, we rely on a definition of Verifiable FL inspired by [1]:

Definition (Verifiable FL). FL is verifiable if selected parties are able to verify that the tasks of all participants are correctly performed without deviation.

Threat models

In order to thoroughly analyze miscellaneous flavors of the applied threat models we distinguish the following four categories:

- **honest (or trusted)**: always follows the protocol correctly and is trusted with sensitive information;
- **honest-but-curious**: always follows the protocol correctly, but is not trusted with sensitive information;
- **forger**: may try to forge different data, but otherwise follows the protocol, is not trusted with sensitive information;
- **malicious**: can arbitrary deviate from the protocol and is not trusted with sensitive information.

ANALYSIS OF EXISTING APPROACHES

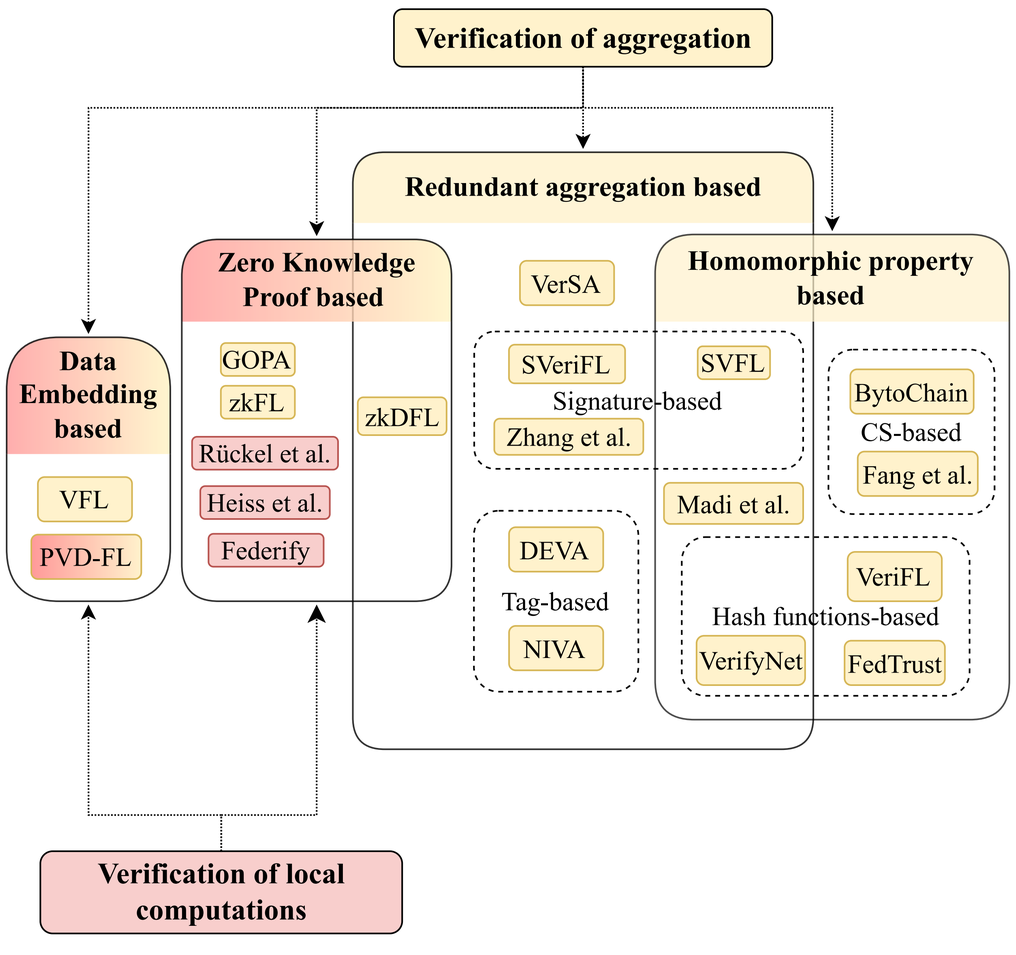

Figure 1: A taxonomy of verifiable cross-silo FL protocols. The red color corresponds to approaches focused on the verification of clients' computations, the yellow color is used for approaches focused on the aggregation verification.

Approach	Computational cost client	Computational cost server	Communication cost client	Communication cost server	Threat model client	Threat model server	Server-Client collusion	TA
VerSA [2]	$O(D)$	$O(CD)$	$O(D)$	$O(CD)$	h-b-c	[forger]	✗	✗
SVeriFL [3]	$O(D)$	$O(CD)$	$O(D)$	$O(CD)$	h-b-c	forger	✗	✓
Zhang et al. [4]	$O(D)$	$O(C)$	$O(1)$	$O(C)$	[hon]	forger	[✗]	✗
DEVA [5]	$O(CD)$	$O(CD)$	$O(CD)$	$O(CD)$	h-b-c	forger	✗	✗
NIVA [6]	$O(CD)$	$O(CD)$	$O(CD)$	$O(CD)$	[h-b-c]	[forger]	[✗]	✗
SVFL [7]	$O(D)$	$O(C)$	$O(1)$	$O(C)$	h-b-c	[forger]	✗	✗
Madi et al. [8]	$O(D)$	$O(C)$	$O(1)$	$O(D)$	hon	[forger]	✗	✗
VerifyNet [9]	$O(D)$	$O(CD)$	$O(D)$	$O(CD)$	h-b-c	forger	✗	✗
BytoChain [10]	$O(C+D)$	$O(1)$	$O(C)$	$O(1)$	mal	[mal]	[✗]	✗
Fang et al. [11]	$O(C+D)$	$O(1)$	$O(C)$	$O(1)$	[hon]	[forger*]	[✗]	✗
VeriFL [12]	$O(C+\frac{D}{E})$	$O(1)$	$O(C)$	$O(1)$	h-b-c	forger	✗	CRS
FedTrust [13]	$O(CD)$	$O(1)$	$O(CD)$	$O(1)$	hon	mal	[✗]	✗
zkDFL [14]	$O(D)$	$O(CD)$	$O(C)$	$O(C)$	hon	[forger*]	[✗]	CRS
GOPA [15]	$O(D\log C)$	N/A	$O(D\log C)$	N/A	mal*	N/A	N/A	✗
zkFL [16]	$O(CD)$	$O(CD)$	$O(1)$	$O(C\log(CD))$	[hon]	[forger]	[✗]	✗
VFL [17]	$O(D)$	$O(1)$	$O(1)$	$O(1)$	h-b-c	[forger]	[✗]	✗
PVD-FL [18]	$O(D)$	N/A	$O(D)$	N/A	[forger*]	N/A	N/A	✗

Table 1: Asymptotic complexity per epoch and threat models comparison of verifiable FL protocols with verifiable aggregation. Sections correspond to taxonomy categories (Figure 1). Notations: E – a number of epochs, C – a number of clients, D – a number of vector dimensions.

Approach	Computational cost client	Computational cost server	Communication cost client	Communication cost server	Threat model client	Threat model server	Server-Client collusion	TA
Rückel et al. [19]	$O(D)$	N/A	$O(C)$	N/A	[forger]	N/A	N/A	CRS
Heiss et al. [20]	$O(D)$	N/A	$O(C)$	N/A	[forger]	N/A	N/A	CRS
Federify [21]	$O(D)$	N/A	$O(C)$	N/A	[forger]	N/A	N/A	CRS
PVD-FL [18]	$O(D)$	N/A	$O(D)$	N/A	[forger*]	N/A	N/A	✗

Table 2: Asymptotic complexity per epoch and threat models comparison of verifiable FL protocols with verification of local models computation. Sections correspond to taxonomy categories (Figure 1). Notations: C – a number of clients, D – a number of vector dimensions.

Impact of a cross-silo setting on verification:

- Since the number of participants in cross-silo settings is moderate while ML models typically have large sizes, a dependence on D is less desirable. Taking into account that clients anyway must send their local models to a server with $O(D)$ communication cost, the overall FL complexity would become asymptotically worse only in cases when the verification overhead is larger than D.
- Many approaches rely on a blockchain infrastructure [10, 11, 14, 19, 20], however within the context of cross-silo FL, such infrastructure leads to a significant computational overhead. All miners have to execute identical calculations, resulting in a tremendous total computational burden across all participants.
- In approaches that rely on cryptographic signature schemes, a verifier is only capable of checking that server has not omitted values from other clients and has not inserted additional values in the sum. Nevertheless, a malicious server still may aggregate arbitrary signed values and successfully pass the verification.

RESEARCH GAPS AND FUTURE CHALLENGES

- There are currently no verifiable FL protocols that fully support verification of both computations performed by clients and server at the same time;
- The verifiable aggregation is primarily studied for the most popular type of aggregation – averaging of vectors possessed by DOs. Nevertheless, other U-statistics with a kernel of the degree two or larger (e.g. Kendall rank correlation coefficient) could be applied;
- There are no protocols that are robust to a collusion between client and server to bypass the verification;
- The repetitive nature of FL training is usually overlooked while developing a verifiable protocol. However, we believe that leveraging this characteristic can lead to optimizations;
- There are no works exploring the applicability of recursive ZKP schemes and Incrementally Verifiable Computation (IVC) in the context of FL.

As future work, we plan to analyze the impact of various ZKP schemes on the complexities of FL settings.

Scheme	Parameters size	Proving	Verification	Proof Size
Dory	$O(\|C\|)$	$O(\|C\|)$	$O(\log\|C\|)$	$O(\log\|C\|)$
Gemini (time-efficient)	$O(\|C\|)$	$O(\|C\|)$	$O(\log\|C\|)$	$O(\log\|C\|)$
SuperSonic, DARK-fix	$O(1)$	$O(\|C\|\log\|C\|)$	$O(\log\|C\|)$	$O(\log\|C\|)$
Bulletproofs	$O(\|C\|)$	$O(\|C\|)$	$O(\|C\|)$	$O(\log\|C\|)$
Compressed Σ-protocol	$O(\|C\|)$	$O(\|C\|)$	$O(\|N\|)$	$O(\log(\|C\|))$
Groth16	$O(\|C\|)$	$O(\|C\|\log\|C\|)$	$O(\|N\|)$	$O(1)$
Sonic	$O(\|C\|)$	$O(\|C\|\log\|C\|)$	$O(N)$	$O(1)$
Dew	$O(1)$	$O(\|C\|^2)$	$O(\log\|C\|)$	$O(1)$

Table 3: Asymptotic comparison of ZKP schemes with logarithmic and constant proof size complexity. C – computation expressed as a circuit, $|C|$ – number of gates, $|N|$ – length of inputs.

References

• **1.** Zhang, Y. & Yu, H. 2022, in International Joint Conference on Artificial Intelligence • **2.** Hahn, C., et al. 2023, IEEE Transactions on Dependable and Secure Computing, 20, 36 • **3.** Gao, H., et al. 2023, Information Sciences, 622, 98 • **4.** Zhang, X., et al. 2020, in ICC 2020 - 2020 IEEE International Conference on Communications (ICC), 1–6 • **5.** Tsaloli, G., et al. 2021, in Information Security: 24th International Conference, ISC 2021, Virtual Event, November 10–12, 2021, Proceedings (Berlin, Heidelberg: Springer-Verlag), 296–319 • **6.** Brunetta, C., et al. 2021, Non-Interactive, Secure Verifiable Aggregation for Decentralized, Privacy-Preserving Learning, Cryptology ePrint Archive, Paper 2021/654 • **7.** Luo, F., et al. 2024, IEEE Transactions on Mobile Computing, 23, 850 • **8.** Madi, A., et al. 2021, in 2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS), 1–8 • **9.** Xu, G., et al. 2020, IEEE Transactions on Information Forensics and Security, 15, 911 • **10.** Li, Z., et al. 2021, IEEE Network, 35, 295 • **11.** Fang, C., et al. 2022, Computer Communications, 186, 1 • **12.** Guo, X., et al. 2021, IEEE Transactions on Information Forensics and Security, 16, 1736 • **13.** Hsu, C.-F., et al. 2022, in 2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR), 318–323 • **14.** Ahmadi, M. & Nourmohammadi, R. 2023, zkDFL: An efficient and privacy-preserving decentralized federated learning with zero-knowledge proof, `https://synthical.com/article/a8c00457-dadd-4207-bd23-7edaf0188617` • **15.** Sabater, C., et al. 2022, Machine Learning, 111 • **16.** Wang, Z., et al. 2023, arXiv preprint arXiv:2310.02554 • **17.** Fu, A., et al. 2022, IEEE Transactions on Industrial Informatics, 18, 3316 • **18.** Zhao, J., et al. 2022, IEEE Transactions on Information Forensics and Security, 17, 2059 • **19.** Rückel, T., et al. 2022, Computer Networks, 202, 108621 • **20.** Heiss, J., et al. 2022, in 2022 IEEE International Conference on Blockchain (Blockchain) (Los Alamitos, CA, USA: IEEE Computer Society), 194–201 • **21.** Keshavarzkalhori, G., et al. 2024, IEEE Access, 12, 3240