Verifiable cross-silo federated learning

Aleksei Korneev, Jan Ramon

Verifiable cross-silo federated learning

1st Aleksei Korneev
Univ. Lille, Inria, CNRS
Centrale Lille, UMR 9189 - CRIStAL
F-59000 Lille, France
aleksei.korneev@inria.fr

2nd Jan Ramon
Univ. Lille, Inria, CNRS
Centrale Lille, UMR 9189 - CRIStAL
F-59000 Lille, France
jan.ramon@inria.fr

***Abstract*—Federated Learning (FL) is a widespread approach that allows training machine learning (ML) models with data distributed across multiple devices. In cross-silo FL, which often appears in domains like healthcare or finance, the number of participants is moderate, and each party typically represents a well-known organization. However, malicious agents may still attempt to disturb the training procedure in order to obtain certain benefits, for example, a biased result or a reduction in computational load. While one can easily detect a malicious agent when data used for training is public, the problem becomes much more acute when it is necessary to maintain the privacy of the training dataset. To address this issue, there is recently growing interest in developing verifiable protocols, where one can check that parties do not deviate from the training procedure and perform computations correctly. In this paper, we conduct a comprehensive analysis of such protocols, and fit them in a taxonomy. We perform a comparison of the efficiency and threat models of various approaches. We next identify research gaps and discuss potential directions for future scientific work.**

***Index Terms*—federated learning, FL, cross-silo FL, verification, verifiable protocols, zero knowledge proofs**

I. Introduction

Nowadays, the broad propagation of ML technologies is rapidly increasing. ML applications affect a variety of fields such as medicine, finance, marketing, education, and many others. Many ML approaches rely on a process of training on historical data: a model learns statistical patterns that later allow new predictions to be inferred. However, in some cases, data may contain private or confidential information; therefore, access to such data is limited, and applying ML must be done with extreme caution, either due to an interest in privacy of the data owners (DOs), e.g., individual persons caring about their privacy or companies caring about intellectual property, or for regulatory compliance, e.g., with the General Data Protection Regulations (GDPR).

In FL, multiple DOs, who are also sometimes referred to as clients, can train a model together, possibly under coordination of a central server, by exchanging encrypted messages without revealing their private data. As a result, researchers can benefit from a large amount of shared data and at the same time preserve privacy. However, while preserving privacy in FL allows protecting sensitive information, at the same time it produces an additional challenge in the verification of participants' behavior. Indeed, due to a possibility of malicious actions, it is important to ensure that all calculations are performed correctly even if the used data is private.

FL is often divided into two categories: cross-device and cross-silo. In the cross-device FL setting, data comes from a large number of small and usually anonymous devices with low computational capacities. Anonymity complicates penalizing clients; a single client is free to abort or to violate the procedure at any time. In contrast, in this paper we focus on cross-silo FL where the number of parties is moderate; each party is usually a well-known and large entity that is expected to cooperate in the entire training process via devices with high computing power. Each party has an incentive to care about its reputation and can be held liable if it is found to be fraudulent. For instance, cross-silo FL appears in the healthcare domain, where DOs are medical centers or hospitals that collaborate to train ML models to improve patient care. Although the need for countermeasures against malicious attacks in such setting could be reduced due to the liability of participants, verification of the calculations is still required to establish confidence in FL's performance.

Recently, dozens of research works devoted to verifiable FL have been published, proposing methods to ensure the verifiability of the parties' computations, using different infrastructures and relying on various assumptions. Nonetheless, to the best of our knowledge, verifiability in the context of cross-silo FL has not been thoroughly studied. Chao et al. in [1] studied challenges of cross-silo FL setting in details, but the verifiability property was not taken into account. In [2], [3] authors were focused on verifiability in FL, however features of the cross-silo setting were not considered and protocols' efficiency was not analysed. Mansouri et al. presented a SoK paper [4] devoted to secure aggregation protocols and included verification in the list of challenges, nevertheless, specific features of cross-silo FL were not in the scope of the paper and an efficiency analysis of verification techniques was not performed. Lastly, in [5], [6] authors studied applications of various zero knowledge proof (ZKP) schemes for ML, but these works do not address FL.

In this paper, we provide a comprehensive analysis of existing verifiable cross-silo FL protocols and identify principal research gaps. Our contributions are summarized as follows:

- to the best of our knowledge, we are the first to 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.

II. Background

A. FL process

In this paper, we consider a typical FL process where the data owners each own a subset of the instances of a dataset, and in order to train a model should compute the sum of vectors (e.g., gradients). We consider both settings where this aggregation is coordinated or performed by a central server [7] and settings where the data owners perform the aggregation in a decentralized way. The training procedure could be repeated several times, we refer to each iteration as an epoch.

B. Verifiable FL

In the scope of this paper, we rely on the definition of Verifiable FL proposed in [2]:

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

Following this definition, in contrast to the survey [3], we only include approaches that at least partly verify computations of the FL process. For example, we do not analyze protocols which are focused only on verification of identity, ownership, or data provenance. Moreover, we only consider methods that aim at preserving privacy, hence do not publish sensitive data. We also exclude protocols considered in [4] that aim to prevent model poisoning attacks by analyzing distribution of values submitted by parties. Such methods efficiently mitigate some attacks, but do not allow to entirely verify the correctness of individual computations or of the individual uses of the input data, e.g., an individual outlier input value is infrequent but possibly valid. Moreover, their efficiency depends on the domain and an attacker strength. On the other hand, we do include in our analysis several protocols devoted to verifiable federated private averaging and verifiable cross-device FL since the same verification techniques could be used in the cross-silo FL setting.

C. Threat models

In the scope of the considered works, authors usually rely on two widely-spread types of threat models: honest-but-curious (a.k.a. semi-honest) and malicious. According to the standard cryptography definitions, an *honest-but-curious* agent does not deviate from the protocol, but keeps a record of the protocol transcript and analyze it to gain extra information about other users, while a *malicious* adversary can deviate from the prescribed protocol instructions and follow an arbitrary strategy to obtain greater benefits. However, in the context of FL, authors often adapt these definitions with additional properties. In order to thoroughly analyze miscellaneous flavors of the applied threat models we distinguish the following four categories:

- **honest**: 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.

Since the malicious threat model is not limited to a specific type of attacks, we assume that a protocol supports the malicious threat model when its defense mechanisms are able to cope not only with forging, but also some other deviations. Additionally, we note that malicious, forger, and honest-but-curious agents can collude.

III. Analysis of existing approaches

In this section, we present a taxonomy of existing verifiable cross-silo FL protocols, analyze the efficiency of verification techniques, threat models and discuss the impact of the cross-silo setting on verification. In the scope of this section, we refer to the number of clients as C and to the number of updates dimensions sent by clients as D.

In order to ensure that a FL protocol is executed correctly, for each epoch one has to verify both the aggregation performed by a server and local computations performed by clients. We distinguish four categories of different verification techniques and describe each of them below. The full taxonomy is presented on the Figure 1. Although each approach has specific characteristics, our categories allow observing general design patterns and infer conclusions about their efficiency. For this purpose, we assess computational and communication costs both per client and per server for each method. The comparison of threat models and asymptotic complexities of protocols devoted to verification of aggregation is presented in the Table I. We emphasize that complexity metrics are calculated specifically for the verification overhead and do not reflect default FL interactions and computations. For blockchain based approaches, we assume that uploading data to the blockchain requires $O(C)$ communication overhead. Lastly, we assume that public key infrastructure, ML model weights and seeds of PRGs are initialized before the training procedure and do not require a presence of a trusted party.

A. Taxonomy description

Redundant aggregation (RA) based verification. This category consists of approaches that require the server to aggregate some redundant values in order to prove that the aggregation of clients updates is performed correctly. This feature leads to a computational cost of the server to be at least $O(C)$. Moreover, some protocols are designed under assumption that each party has one secret value, therefore a naive scaling of the approach to a FL setting where parties share multi dimensional data would lead to an additional factor D in the complexity.

In [10], [9] authors proposed to check that the result of the updates aggregation is correct by means of cryptographic signatures schemes based on bilinear pairings. In both works, the

Approach	Computational cost		Communication cost		Threat model		Server-Client collusion	TA
	client	server	client	server	client	server		
VerSA [8]	$O(D)$	$O(CD)$	$O(D)$	$O(CD)$	h-b-c	[forger]	✗	✗
SVeriFL [9]	$O(D)$	$O(CD)$	$O(D)$	$O(CD)$	h-b-c	forger	✗	✓
Zhang et al. [10]	$O(D)$	$O(C)$	$O(1)$	$O(C)$	[hon]	forger	[✗]	✗
DEVA [11]	$O(CD)$	$O(CD)$	$O(CD)$	$O(CD)$	h-b-c	forger	✗	✗
NIVA [12]	$O(CD)$	$O(CD)$	$O(CD)$	$O(CD)$	[h-b-c]	[forger]	[✗]	✗
SVFL [13]	$O(D)$	$O(C)$	$O(1)$	$O(C)$	h-b-c	[forger]	✗	✗
Madi et al. [14]	$O(D)$	$O(C)$	$O(1)$	$O(D)$	hon	[forger]	✗	✗
VerifyNet [15]	$O(D)$	$O(CD)$	$O(D)$	$O(CD)$	h-b-c	forger	✗	✗
BytoChain [16]	$O(C+D)$	$O(1)$	$O(C)$	$O(1)$	mal	[mal]	[✗]	✗
Fang et al. [17]	$O(C+D)$	$O(1)$	$O(C)$	$O(1)$	[hon]	[forger*]	[✗]	✗
VeriFL [18]	$O(C+\frac{D}{E})$	$O(1)$	$O(C)$	$O(1)$	h-b-c	forger	✗	CRS
FedTrust [19]	$O(CD)$	$O(1)$	$O(CD)$	$O(1)$	hon	mal	[✗]	✗
zkDFL [20]	$O(D)$	$O(CD)$	$O(C)$	$O(C)$	hon	[forger*]	[✗]	CRS
GOPA [21]	$O(D\log C)$	N/A	$O(D\log C)$	N/A	mal*	N/A	N/A	✗
zkFL [22]	$O(CD)$	$O(CD)$	$O(1)$	$O(C\log(CD))$	[hon]	[forger]	[✗]	✗
VFL [23]	$O(D)$	$O(1)$	$O(1)$	$O(1)$	h-b-c	[forger]	[✗]	✗
PVD-FL [24]	$O(D)$	N/A	$O(D)$	N/A	[forger*]	N/A	N/A	✗

TABLE I: Asymptotic complexity and threat models comparison of FL protocols with verifiable aggregation. Sections correspond to taxonomy categories (Figure 1). Server-Client collusion column shows if a method allows client and server to collude to bypass the verification. Notations: E – a number of epochs, C – a number of clients, D – a number of vector dimensions, TA – trusted authority. A symbol "*" corresponds to a threat model applied to a fraction of agents. Square brackets denote threat models and collusion markers that are inferred after analyzing the verification method and are not explicitly described in the corresponding paper.

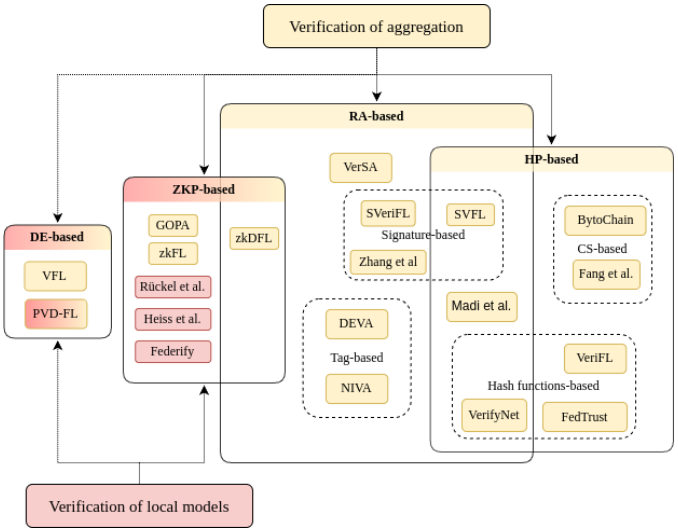

Fig. 1: A taxonomy of verifiable cross-silo FL protocols. The red color corresponds to approaches focused on the verification of clients' computations, the yellow color is used for approaches focused on the aggregation verification.

server has to compute a redundant aggregation of signatures. Indeed, such schemes allow ensuring that the result is obtained from data signed by all other clients, however a malicious server may aggregate arbitrary signed values (e.g., values from previous epochs) and successfully pass the verification with a fabricated resulting value. As a result, by relaxing the threat model, both approaches become leaders in their category from the complexity perspective.

All RA-based approaches cope with a forger server while considering clients to be honest or honest-but-curious. Additionally, in [9], integrity of data shared by clients is verified, this threat is processed by addition of a TA.

Homomorphic property (HP) based verification. This category covers verification techniques which rely on the HP of underlying primitives: hash functions [18], [19] and commitment schemes [16], [17]. The general idea of such protocols is the following: clients compute hashes/commitments from their data and share results with each other, then all clients may verify the result of aggregation, i.e. check that this result corresponds to the aggregation of hashes/commitments through homomorphism. As a consequence, both computational and communication costs of the server are $O(1)$ if the ciphertext length does not depend on C or D. Clients have to compute a hash/commitment in $O(D)$ from their data and the aggregation of hashes/commitments from other clients in $O(C)$. Since clients have to exchange messages with each other, the communication cost per client is at least $O(C)$. Exceptionally, in FedTrust [19] client costs have an additional $O(D)$ factor, as the hash is calculated for each component separately.

In [16], [17] authors rely on a blockchain infrastructure. While the complexity metrics for the verification overhead are the same as for other approaches from this category, the total computational cost of such approaches is much larger due to the replication of computations. Moreover, such approaches also have specific threat models, since they rely on blockchain security. For instance, Fang et al. [17] assume 70% of stake holders to be honest.

There are also several protocols that lie at the intersection of the RA- and HP-based categories [10], [15], [14]. In such approaches server has to perform a redundant aggregation, however the verification of the aggregation is also based

on homomorphic properties. In terms of threat models and complexities these approaches do not differ from concurrent works from RA- and HP-based categories.

ZKP-based verification. The third category contains approaches which are based on ZKPs. The core principle could be described as follows: a party performs calculations and at the same time computes the proof, which is shared along with the result of calculations; other parties can later run the proof verification algorithm to ensure that the result was computed correctly. In contrast to previous categories, advanced ZKPs allow proving arbitrary computations, therefore such methods are suitable for proving the correctness of both aggregation of clients updates [22], [20], [21] and computation of these updates [25], [26], [27]. Moreover, recent ZKP schemes provide a proof size that is sublinear in the amount of computations to prove.

In protocols focused on aggregation, authors build on different infrastructures and ZKP schemes. In GOPA [21], authors introduced a decentralized gossip approach where nodes publish proofs of their computations using Σ-protocols. In zkFL [22] authors apply more modern ZKP scheme, Halo2, and provide two version of the protocol – a centralized FL setting and a blockchain based one. In zkDFL [20] authors rely on blockchain infrastructure and Groth16 scheme. Moreover, authors use different techniques to prove that each update was sent by one of the clients. Consequently, differences in the settings and chosen ZKP schemes result in different complexity metrics while applied threat models are similar.

In contrast to the verifiable aggregation protocols, approaches focused on DO's computation verification [27], [26], [25] have almost identical design: all protocols use the Groth16 scheme to achieve verifiability in a blockchain based setting. However, authors focus on verification of different ML models: a linear regression model in [27], a naive Bayes classifier in [25] and a feedforward neural network in [26]. In all three approaches, aggregation is performed by a smart contract, therefore the correctness of the aggregation relies on the blockchain security assumptions.

Data Embedding (DE) based verification. This category covers protocols, where participants embed additional values into their data before sharing it with untrusted parties; later, the result of calculations performed by an untrusted source is assumed to be correct if the corresponding additional values are computed correctly. The embedding principle leads to an increase in the size of the transmitted data and the complexity of the outsourced calculations, which depends on the size of embedded values, and require more expensive data preprocessing.

B. Features of cross-silo FL verification

One can notice that in Table I there are mainly two parameters determining communication cost: C and D, however there is a difference in their impact on the cross-silo setting. Since the number of participants in such setting is moderate while ML models typically have large sizes, a dependence on D is less desirable. Besides, taking into account that clients must send their local models to a server, the overall FL complexity would only be asymptotically worse in cases when communication cost depends on CD, such as in [11], [12], [19]. We note that VFL approach [23] achieves the best asymptotic complexity in comparison to others.

We also observed that many approaches rely on a blockchain infrastructure [17], [16], [20], [27], [26]. This strategy offers several advantages. For instance, smart contracts enforce transparent and verifiable distribution of incentives [27]. Additionally, the use of smart contracts also makes a presence of a distinct server unnecessary, thereby replacing a single party trust with blockchain trust guarantees. 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. At the same time, in cross-silo FL there is typically at least one party interested in obtaining the results of training, thus there is no a strong need for a decentralized infrastructure.

Almost all verifiable aggregation approaches from our analysis rely on honest or honest-but-curios clients threat model. The only exceptions with support of malicious clients are: the blockchain based approach BytoChain [16], where a committee of verifiers is tasked to check properties of clients uploads, therefore partially mitigating poisoning attacks, and decentralized protocols where clients perform aggregation themselves [21], [24]. Indeed, weaker threat models are common in cross-silo FL. However, in real world conditions the use of such threat models is not always advisable.

IV. Conclusions and future directions

In this paper, we analyzed verifiable cross-silo FL protocols while focusing on their efficiency and threat models. We proposed a taxonomy based on applied verification techniques and discussed how cross-silo FL characteristics impact the verification. Finally, in this section, we discuss several challenges which have not yet been fully addressed by the research community.

Firstly, most protocols target malicious behavior of either the server or clients, while only few protocols cope with both threats. We believe that consideration of such cases is important for development of practical protocols. Secondly, we observed that verifiable aggregation is primarily studied for the most popular type of aggregation – averaging of vectors possessed by DOs. Nevertheless, in certain settings, other U-statistics with kernel of degree two or larger (e.g. Kendall rank correlation coefficient) could be applied [28], introducing new challenges in the verification process. Thirdly, to the best of our knowledge, there are no protocols that support collusion between client and server to bypass the verification. However, in real world scenarios such collusion might occur. Fourthly, we noticed that the iterative nature of FL training is usually overlooked when designing a verifiable protocol. Nevertheless, this property opens up a possibility of developing various optimizations. For instance, in [18], authors used this property to combine verification of multiple epochs

together, thereby reducing the computational cost. We believe that new optimizations also could be developed for protocols based on other verification techniques. Lastly, to the best of our knowledge, there are no works conducting a thorough analysis of the applicability of various ZKP schemes for the cross-silo FL setting.

ACKNOWLEDGMENTS

This project was partially supported by the 'Chair TIP' project funded by ANR, I-SITE, INRIA and MEL, and the Horizon Europe TRUMPET project grant no. 101070038.

REFERENCES

[1] C. Huang, J. Huang, and X. Liu, "Cross-silo federated learning: Challenges and opportunities," 2022.

[2] Y. Zhang and H. Yu, "Towards verifiable federated learning," in *International Joint Conference on Artificial Intelligence*, 2022.

[3] A. Tariq, M. A. Serhani, F. Sallabi, T. Qayyum, E. S. Barka, and K. A. Shuaib, "Trustworthy federated learning: A survey," 2023.

[4] M. Mohamad, M. Önen, W. Ben Jaballah, and M. Conti, "Sok: Secure aggregation based on cryptographic schemes for federated learning," in *PETS 2023, 23rd Privacy Enhancing Technologies Symposium, 10-15 July 2023, Lausanne, Switzerland (Hybrid Conference)*, IACR, Ed., Lausanne, 2023, iACR.

[5] M. Labs, "Zero-knowledge proof meets machine learning in verifiability: A survey," https://drive.google.com/file/d/1tylpowpaqcOhKQtYolPlqvx6R2Gv4IzE/view, 2023.

[6] Z. Xing, Z. Zhang, J. Liu, Z. Zhang, M. Li, L. Zhu, and G. Russello, "Zero-knowledge proof meets machine learning in verifiability: A survey," 2023.

[7] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konecný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021. [Online]. Available: http://dx.doi.org/10.1561/2200000083

[8] C. Hahn, H. Kim, M. Kim, and J. Hur, "Versa: Verifiable secure aggregation for cross-device federated learning," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 1, pp. 36–52, 2023.

[9] H. Gao, N. He, and T. Gao, "Sverifl: Successive verifiable federated learning with privacy-preserving," *Information Sciences*, vol. 622, pp. 98–114, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025522014359

[10] X. Zhang, A. Fu, H. Wang, C. Zhou, and Z. Chen, "A privacy-preserving and verifiable federated learning scheme," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[11] G. Tsaloli, B. Liang, C. Brunetta, G. Banegas, and A. Mitrokotsa, "Deva: Decentralized, verifiable secure aggregation for privacy-preserving learning," in *Information Security: 24th International Conference, ISC 2021, Virtual Event, November 10–12, 2021, Proceedings*. Berlin, Heidelberg: Springer-Verlag, 2021, p. 296–319. [Online]. Available: https://doi.org/10.1007/978-3-030-91356-4_16

[12] C. Brunetta, G. Tsaloli, B. Liang, G. Banegas, and A. Mitrokotsa, "Non-interactive, secure verifiable aggregation for decentralized, privacy-preserving learning," Cryptology ePrint Archive, Paper 2021/654, 2021. [Online]. Available: https://eprint.iacr.org/2021/654

[13] F. Luo, S. Al-Kuwari, and Y. Ding, "Svfl: Efficient secure aggregation and verification for cross-silo federated learning," *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 850–864, 2024.

[14] 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," in *2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)*, 2021, pp. 1–8.

[15] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 911–926, 2020.

[16] Z. Li, H. Yu, T. Zhou, L. Luo, M. Fan, Z. Xu, and G. Sun, "Byzantine resistant secure blockchained federated learning at the edge," *IEEE Network*, vol. 35, no. 4, pp. 295–301, 2021.

[17] C. Fang, Y. Guo, J. Ma, H. Xie, and Y. Wang, "A privacy-preserving and verifiable federated learning method based on blockchain," *Computer Communications*, vol. 186, pp. 1–11, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366422000081

[18] X. Guo, Z. Liu, J. Li, J. Gao, B. Hou, C. Dong, and T. Baker, "Verifl: Communication-efficient and fast verifiable aggregation for federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1736–1751, 2021.

[19] C.-F. Hsu, J.-L. Huang, F.-H. Liu, M.-C. Chang, and W.-C. Chen, "Fedtrust: Towards building secure robust and trustworthy moderators for federated learning," in *2022 IEEE 5th International Conference on Multimedia Information Processing and Retrieval (MIPR)*, 2022, pp. 318–323.

[20] M. Ahmadi and R. Nourmohammadi, "zkdfl: An efficient and privacy-preserving decentralized federated learning with zero-knowledge proof," https://synthical.com/article/a8c00457-dadd-4207-bd23-7edaf0188617, 11 2023.

[21] C. Sabater, A. Bellet, and J. Ramon, "An accurate, scalable and verifiable protocol for federated differentially private averaging," *Machine Learning*, vol. 111, 10 2022.

[22] Z. Wang, N. Dong, J. Sun, and W. Knottenbelt, "zkfl: Zero-knowledge proof-based gradient aggregation for federated learning," *arXiv preprint arXiv:2310.02554*, 2023.

[23] A. Fu, X. Zhang, N. Xiong, Y. Gao, H. Wang, and J. Zhang, "Vfl: A verifiable federated learning with privacy-preserving for big data in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3316–3326, 2022.

[24] J. Zhao, H. Zhu, F. Wang, R. Lu, Z. Liu, and H. Li, "Pvd-fl: A privacy-preserving and verifiable decentralized federated learning framework," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2059–2073, 2022.

[25] G. Keshavarzkalhori, C. Pérez-Solà, G. Navarro-Arribas, J. Herrera-Joancomartí, and H. Yajam, "Federify: A verifiable federated learning scheme based on zksnarks and blockchain," *IEEE Access*, vol. 12, pp. 3240–3255, 2024.

[26] J. Heiss, E. Grunewald, S. Tai, N. Haimerl, and S. Schulte, "Advancing blockchain-based federated learning through verifiable off-chain computations," in *2022 IEEE International Conference on Blockchain (Blockchain)*. Los Alamitos, CA, USA: IEEE Computer Society, aug 2022, pp. 194–201. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/Blockchain55522.2022.00034

[27] T. Rückel, J. Sedlmeir, and P. Hofmann, "Fairness, integrity, and privacy in a scalable blockchain-based federated learning system," *Computer Networks*, vol. 202, p. 108621, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128621005132

[28] J. Bell, A. Bellet, A. Gascon, and T. Kulkarni, "Private protocols for u-statistics in the local model and beyond," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, S. Chiappa and R. Calandra, Eds., vol. 108. PMLR, 26–28 Aug 2020, pp. 1573–1583. [Online]. Available: https://proceedings.mlr.press/v108/bell20a.html