A State Channel Based Approach to Address Scalability of Healthcare Data Sharing

Jahanzeb Shahid
Department of Computer Science
Università degli Studi di Milano, Milan, Italy
jahan.shahid@unimi.it; jahazeb.shahid@unicam.it

Stelvio Cimato
Department of Computer Science
Università degli Studi di Milano, Milan, Italy
stelvio.cimato@unimi.t

***Abstract*—The process of exchanging healthcare data introduces stringent requirements regarding user privacy. Federated learning (FL) is a novel model-sharing technique that aims to give additional privacy guarantees during the machine learning process. Blockchain, as a form of distributed ledger technology, possesses the characteristic of trustworthiness; however, it is deficient in terms of computational capacity with a high-latency network due to its laborious consensus protocols. In this paper, we present a distributed healthcare FL-based secure model sharing architecture to ensure healthcare data privacy and scalability.The solution relies on the state channels technique to reduce on-chain transactions, contrast architecture latency, and reduce bandwidth consumption, alleviating the burden on the blockchain. State channels can be utilized to efficiently execute the tasks of sharing federated learning models and to solve the scalability problem.**

***Index Terms*—Blockchain, State channel, Scalability, Privacy, Security, Federated Learning, Healthcare Devices**

I. Introduction

Healthcare data sharing refers to the exchange of medical and health-related data among individuals, healthcare providers, organizations, and systems, which is essential for the diagnosis, treatment, prediction, and improvement of medical quality. Different entities such as healthcare centers, hospitals, smart devices, clinics, and medical research centers record and store healthcare data [1]. However, data sharing in healthcare presents various challenges, primarily involving security, privacy, and scalability issues. In addition, over the next five years, the amount of medical data is projected to increase significantly. This is attributed to various factors, such as the conversion of traditional paper healthcare data to digital format, the greater use of electronic health records (EHR), advances in healthcare technology and an increased quantity of healthcare data generated by patients and medical devices [2].

Blockchain technology with its inherent security features like transparency, traceability, and immutability enables the sharing of data in a trustless environment. Therefore, the integration of blockchain technology with medical services is a hot topic today in research communities [3]. Federated learning (FL) has gained significant interest in the field of data sharing due to its data value aggregation, prevention of data privacy breaches, and reduced network communication burden. It has an intrinsic feature of privacy preservation, where distributed devices connected to a network share trained models with their local data to a central server or machine but not the actual data. It has been used in several situations, including the Internet of Vehicles (IoV), Industrial IoT (IIoT), and Smart City [9].

In this paper, we focus on the scenario in which a number of healthcare entities use their generated data to locally train FL models. These models are aggregated on a central node and then forwarded to the requester entity. Since registering and validating the transfer of the models on the blockchain with regular transactions can be challenging in terms of time and bandwidth usage, we propose the use of state channels to provide an off-chain solution to manage model updates, without interacting with the main blockchain for every iteration, addressing scalability issues. We develop an architecture that combines blockchain technology with FL using a Permissioned blockchain, developing a model sharing framework that encompasses both on-chain and off-chain components. We created a state-channel FL training method and evaluated the solution examining their performance in term of bandwidth and time. The paper is organized as follows. In the next section, we discuss related work. Then we introduce the architecture of the proposed solution and discuss the main components. Finally, we present our solution based on state channels and evaluate its performance.

II. Related Work

A comprehensive survey [12] that thoroughly examines the developing uses of data security and privacy for FL in the Internet of Healthcare Things (IoHT) networks. They thoroughly examine and assess the primary resolutions to the privacy and data security concerns IoHT. A scheme [13] examines the suitability of a distributed reinforcement learning approach in an FL network and evaluates the possible advantages of integrating blockchain technology in the distributed system. The proposed approach improves clinical monitoring and ensures secure communication and data privacy in a decentralized manner. A survey [14] examines the difficulties, solutions and future paths to effectively use blockchain in FL contexts with limited resources.

In a study [15], the authors explain the significant impact of blockchain-based federated learning on improving efficient healthcare. They highlight the crucial function of blockchain-managed edge nodes to prevent any potential failure with the usage of FL. An extensive survey work specifically on the state channels has been presented in [8]. The authors discussed the integration of state channels with the blockchain to improve the scalability of the network.

A survey work has been presented to improve the scalability of the blockchain [16]. The authors discussed the payment channel and transaction rollup techniques to improve the scalability of blockchain transactions in the financial sector. A technique has been presented to improve the computational and supervisory capabilities of blockchain using state channels [17]. Using state channels to generate sandboxes and instantiate federated learning tasks to implement a trust supervision mechanism based on these sandboxes.

A technique known as StateFL has been introduced, a modified BCFL (Blockchain-Based Federated Learning) architecture that employs state channels to alleviate the burden on the blockchain by decreasing the volume of on-chain transactions, improving the latency of the system and reducing transaction costs accordingly [18].

III. Overview of the Architecture

This article focuses on a common scenario in which multiple healthcare entities collaborate to share trained models based on healthcare data. Each entity has their own healthcare data resources at their respective locations. To address this situation, we have created a model-sharing framework based on FL that employs a Permissioned blockchain with state channels. In this architecture, a healthcare entity could be a hospital, research center, medical laboratory, clinic, or smart healthcare unit. There are three key role factors for the architecture.

1) FL Model Requester: A FL model requester refers to entities that need to train a FL model based on healthcare data from any other healthcare entity by agreeing on terms and conditions, as shown in Fig. 1, a research center or hospital could play this role in our framework. The FL model requester starts with the model sharing request to the blockchain. The FL model requester could serve as a representative system for the healthcare entity discussed previously, managing all model sharing requests. This system should verify its identity with the blockchain to issue a model-sharing request.

2) The Blockchain Network: The blockchain network is maintained and managed by all intermediaries. According to the situation described in this article, we utilize a Permissioned blockchain to construct the architecture for exchanging models. We implemented the Hyperledger Fabric (HLF) for this purpose.

3) Federated Learning Resource Network: FL participant nodes must be registered on the Permissioned blockchain, regardless of the device type. The identity, location, and type of stored data make up the registration information. This information will be registered as transactions on the HLF after verification. A header node in the resource network can start model sharing tasks, aggregate or quantize models, and train models for other associated nodes. The header nodes have different functions for different tasks. A model-sharing job usually starts with one model requester and involves several healthcare entities. Federated Learning (FL) trains a centralized model that uses healthcare data from several institutions. We implemented the FL resource network setup in Kubernetes, where nodes can be deployed in containers and easily managed and scaled.

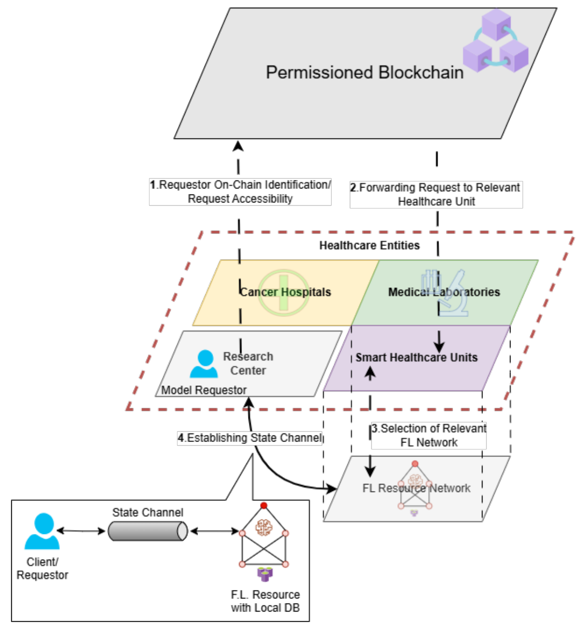

Fig. 1. Blockchain-based FL trained model sharing architecture. Different healthcare entities share FL-based healthcare data trained model over the state channels.

A. Data Storage Structure

In this scheme, there are two data storage structures: on-chain storage and off-chain storage. In federated learning, the data remains in local storage and only the trained models based on the local databases are shared with some central entity. A second important feature of FL is the distributed data structure that prevents single points of failure and user privacy disclosure.

There are two types of data stored in this structure:

1) Local Database: Healthcare data is stored in healthcare devices managed and maintained by healthcare entities, and only trained models are exchanged. To simulate the healthcare devices network to participate in the FL process, we utilized the Kubernetes architecture, where the FL participating nodes are initialized in containers. It becomes easy to scale, automatic deployment of nodes, and management of these containerized nodes.
2) Blockchain Database: Important information, such as latest model update, metadata of trained models like training results, cryptographic hashes of trained models, etc. is stored on the HLF, to realize data integrity verification and traceability of transactions.

IV. Scalability of Network

The state channel is an extended form of blockchain technology that introduces efficiency and scalability by processing some transactions off the chain. Establishing a state channel between healthcare entities for the transmission and computation of FL-based trained models on local devices reduces transaction confirmation time, saves transaction costs, and reduces on-chain storage dependency.

The training process consists of three steps:

1) State-Channel Creation
2) Off-Chain Training Iteration
3) On-Chain Settlement

A. *State-Channel Creation*

After agreement between the model requester and the FL resource network, the FL resource nodes receive the training parameters to initiate the FL training process. A state channel is established between the FL resource header node and the model requester node. A mapping of the locked on-chain state and the off-chain transactions is set, and operations in the state channel will not affect the state on the blockchain. They can also search for existing channels at first; however, if there is no channel available, then they create a new channel.

1) Technical Details for Creating State-Channel: For two healthcare entities (e.g. Cancer Hospital and Research Center), the goal is to privately exchange trained models and validate them without incurring on-chain gas cost or latency for every transaction. After validating the request to train the model using the blockchain, both parties agree on terms and conditions; a smart contract or chaincode is deployed on the blockchain signed by both parties with all terms and conditions to open and close a state channel. To establish state channels, we used the Python gRPC framework, which works as a client-server model. We installed the 'grpcio' and 'grpcio-tools' libraries. In this proposed architecture, the model requester entity can run the server to receive the model updates from the FL resource network. The server side hosts the state channel logic, such as channel management, updating states, and resolving disputes. The client-side entity interacts with the state channel server for model sharing and verification. The client and server sides deploy a protocol file with the extension '.proto' to create a state channel between two entities. We installed Python version 3.11.9 to operate the client-server model.

2) Submitting Chaincode to Permissioned Blockchain: In this architecture, we are using the HLF framework as a permissioned blockchain, and smart contracts (called chaincode in the HLF) are deployed in it. To open a state channel between two healthcare entities to share model updates trained via federated learning, a chaincode is deployed on the blockchain. This chaincode initializes and manages the state channel, stores metadata, and anchors the federated learning process for accountability and dispute resolution. This chaincode controls the state channel's life cycle and stores information such as the channel ID, the initial model state or version, the trained model requirements (quality, epochs, and the number of iterations), and security primitives.

B. *Off-Chain Training Iteration*

Based on the scenario, for each iteration, local nodes receive the initial global model from the header node and provide updated models trained on local data. To complete one iteration, the header node performs the aggregation process and sends this aggregated model to the requester node over the established state channel. We can set the number of iterations during channel creation and close the channel if the training outcomes do not fulfill the needs of the requester. All the interaction between two healthcare entities like model update, feedback, and retraining occur off-chain over the established state channel. Each new state presents the updated model hash H(M1), quality, timestamp, and digitally signed by both entities.

After each cycle, a header node sends a global model to the requested node, completing an off-chain transaction. Small transactions like model exchange in each iteration are stored off-chain, relieving the blockchain network. In the last-iteration latest model is sent to the blockchain. Healthcare entities may share models with FL-Networks off-chain without communicating with the core blockchain. This reduces the need for regular, high-volume communication with the main blockchain, improving scalability. Protocol buffers are used by gRPC for data serialization, ensuring a compact and efficient process. Moreover, it works on the HTTP/2 protocol that provides multiplexing, compression, and bidirectional streaming during transmission.

C. *On-Chain Settlement*

If a party wants to close the channel after completion of the FL process, it submits the latest model update to the HLF for the On-Chain settlement signed by both parties. If after some iterations, one party wants to close the channel due to inconsistencies or unauthorized model updates, that party can stop the operation and request the HLF for settlement by providing the latest valid trained model state. There could be two reasons for On-Chain settlement and the closure of the channel: (1) closing the channel after the completion of the FL task. (2) Closing channel in case of dispute. For the first reason, on-chain settlement begins once the cryptographic hash of the final state of the trained model has been validated. Both participants compute the hash of the final aggregated model and verify the hash value. Upon verification, the HLF receives this hash value and begins the transaction validation process. If any of the participants find errors or unauthorized updates, the latest signed model update is submitted to the blockchain to request dispute resolution. The deployed chaincode verifies the submitted state using signatures and timestamps. On the HLF, valid state is registered as a final state.

In Fig. 2, a state channel is established between two healthcare entities using the gRPC framework to iteratively exchange trained models in the FL resource network. We elaborate steps to follow to establish a state channel between two healthcare entities.

1) Healthcare entity A makes a request to the HLF to train a model by sharing information (e.g., model architecture type, number of parameters, entity-B ID) with its own identity details.

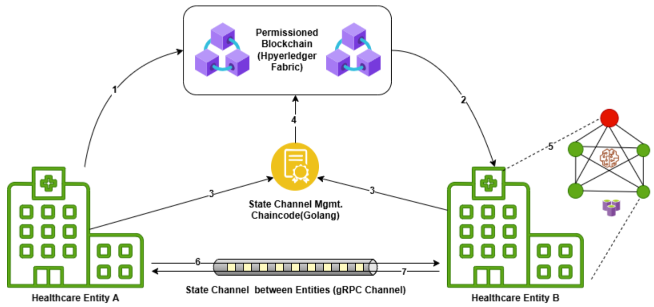

Fig. 2. Establishing of state channel between two healthcare entities to share trained models

2) The HLF performs identity validation of entity-A and analyzes the request against set policies. It checks whether this request with shared details is allowed or not to be forwarded to entity B against set policies. If it is allowed, then forward the request to entity-B with request details.
3) Both entities set terms and conditions (e.g., number of epochs, model quality, model type, data type) on the chaincode, and this chaincode is digitally signed by both entities.
4) After setting the chaincode, it is deployed on the HLF, and a state channel is established between entity A and entity B.
5) Entity B starts the training process using its FL resource network.
6) In each iteration, an aggregated model is shared via the state channel (off-chain) by entity B with its digital signature, without interacting with the HLF.
7) On successful receipt of the updated model in each iteration, entity A sends the acknowledgment to entity B with its digital signature.

V. Evaluation and Results

A. Scalability Analysis

To evaluate scalability, we consider two parameters: bandwidth and time taken to complete tasks with state channels and blockchain. In Fig. 3, there are two graphs, one showing the total time taken to complete tasks against a varying number of entities. Here, an entity refers to healthcare units such as hospitals, pharmacies, medical research centers, etc. Each entity contains 50 nodes and performs tasks with 10 iterations. As you can see, if we involve the blockchain in managing all the tasks, the process becomes more time-consuming. This is because each entity records its model update directly to the main blockchain, resulting in an increase in the number of transactions. Consequently, the consensus algorithm takes longer to register these transactions. It also leads to slower response time due to block creation. However, if we use the state channels, entities interact with the main blockchain occasionally, significantly reducing latency since the transactions are confirmed off-chain. The second plot reduces the bandwidth in the presence of state channels, as updates are only shared directly between entities with final version updates on the blockchain. In the case of the main blockchain, bandwidth usage increases due to regular transaction logging for each update of the model of every entity.

B. Communication Architecture

In this section, we briefly discuss the communication architecture of the HLF within peers that participate in the transaction validation process. This discussion is significant because it pertains to the scalability issue of blockchain technology, where the consensus time required to validate a transaction is lengthy. In case of numerous iterative transactions, such as updates of the FL models, time and bandwidth increase significantly.

In Fig. 3, there are three nodes (1.Endorsement node, 2. Orderer node and 3.Commit node) in the HLF that participate in the model validation process after the FL task is completed. Once the task is completed, the model requester node sends the transaction proposal (1 in Fig.3) to the endorsement node. The state channel management chaincode is also deployed on the endorsement node. The endorsement node converts this proposal into a transaction after validating the proposal from the deployed chanicode and sends it back (2 in Fig.3) to the model requester node. The model requester node then sends this transaction to the (3 in Fig.3) orderer node with updated model, where a block is created. Afterwards, the orderer node sends this created block to (4 in Fig.3) the endorsement and the commit nodes, which store the block in the ledger. Thereafter, the block is committed to the HLF.

The endorsement peer nodes receive the proposal and create transactions. However, the commit peer nodes or non-endorsment nodes do not participate in the transaction creation and only participate in the transaction validation when a block is committed. Let us suppose a trained model of size 1 MB is proposed to the HLF for clarity. The model requester node uploads the 1 MB trained model to the endorsement peer node (Transaction Endorsement in Fig.4) to propose a transaction execution and validation procedure. Chaincode execution transforms the submitted proposal into a transaction after validation. The endorsing peer node digitally signs the transaction with its private key. The model requester receives the transaction from the endorsing peer and sends the 1 MB trained model included in the transaction to the orderer node (Transaction Submission in Fig.4), where the Raft consensus mechanism is running. It creates a block and replicates it to the endorsement and commit nodes (Raft Replication in Fig. 4). Here, the submitted model is copied twice for validation to endorsement and commit nodes. Here, the HLF has one endorsement, one commit, and one orderer node. The final (Peer Gossip in Fig.4) sends the produced block to other peer nodes until all peers get it. Consequently, to validate a trained model size of 1 MB, it will consume 5 to 6 MB of bandwidth on the HLF to validate.

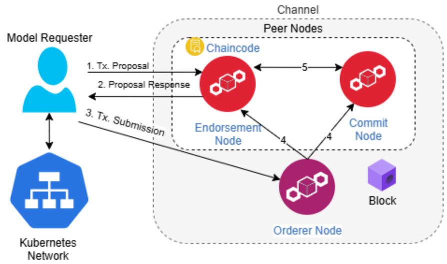

Fig. 3. Peers Communication with Raft Consensus Algorithm

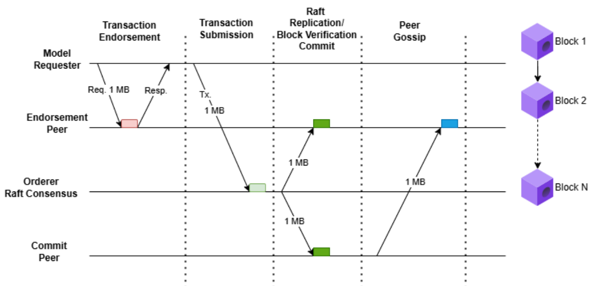

Fig. 4. Consumption of bandwidth in Raft consensus algorithm.

C. Heterogeneous Model Architecture

In the scenario under discussion, multiple healthcare entities are involved, each sharing their different trained models from various healthcare devices. The reason is the different computational and memory resources of the healthcare devices. Some devices, such as wearable, have very limited computational resources, and some healthcare devices, such as PCR or MRI machines, may have GPU's built into their architecture. We are utilizing four distinct model architectures to train the models for healthcare devices. These are 1DCNN, GRU, MobilNetV2, and ResNet-18. In Table I, we present a comparison of these model architectures with their generated and quantized model sizes for each iteration of the FL sharing process.

TABLE I
DIFFERENT MODEL ARCHITECTURE WITH DIFFERENT MODEL SIZES

Model Arch.	No. of Param.	Model Size (MB)	Quantized Size
IDCNN	500K	1 -2	0.5 - 0.8
GRU	1M	2 - 5	1 - 2
MobilNetV2	3.4M	10 - 14	3.5 - 5.6
ResNet-18	11M	30 - 40	8 - 11

D. Model Sizes and Scalability

The size of the trained model is compressed in each iteration with quantization. Here, we try to explore what the size of the model could be for each iteration of the FL process with different models and how state channels can provide scalability with these variant sizes in terms of time and bandwidth compared to the blockchain [21]. The HLF typically uses the Raft consensus algorithms by default for validation purposes. It is lightweight and performs well when a large number of nodes are participating in the consensus process. Here, we discuss the time taken and bandwidth consumed to share a FL-based trained over the state channel and the blockchain. In case of using the state channel, the final version of the trained model is validated on the blockchain but not for every iteration. If the network bandwidth is 100 Mbps (Mega Bits per second) or 12.5 MBps (Mega Bytes per second), then the model upload time can be calculated for the GRU model architecture with model size of 1 to 2 for each iteration.

In federated learning, the size of the updated model depends on the number of parameters in the model and the precision of the parameters, which is typically 32-bit floats = 4 bytes per parameter. Some architectures use 16-bit floats for precision, which significantly decreases the model sizes.For example, in the case of 1DCNN model architecture, there are typically 500,000 parameters that are exchanged to train the updated model. If each parameter is 4 bytes using the 32-bit float precision value, then the model size is approximately 500,000 x 4 bytes = 2,000,000 bytes $\simeq$ 2MB.

The number of devices in FL does not matter because each device trains its own model locally and simply shares updated parameters. The aggregator node collects updates from all devices and creates a global model with the same size and structure each round. In this evaluation, the actual comparison is between using the blockchain with state channels and using the blockchain without the state channel. Therefore, the actual time consumption to validate a shared model occurs while performing a validation procedure of a blockchain transaction. There are two parameters, time and bandwidth, that we are taking to compare the proposed architecture with the default HLF architecture. In Fig. 5, the evaluation of the computation time and bandwidth usage between the HLF and the state channel mechanisms in different model architectures: 1D CNN, GRU, MobilNetV2 and ResNet-18 reveals a clear performance advantage of state channels in federated learning scenarios.

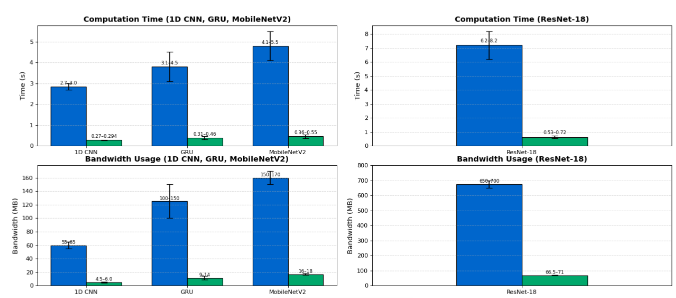

Fig. 5. Blockchain and State Channel Time and Bandwidth Comparison for different Model Architecture.

In Fig. 6 and 7, we look at how well the state channel works as we add more entities, with each entity using 30 to 50 healthcare devices and training models that have different architectures. It can be observed that with an increasing number of healthcare entities, the HLF is taking more time and consuming bandwidth to validate transactions compared to the state channels.

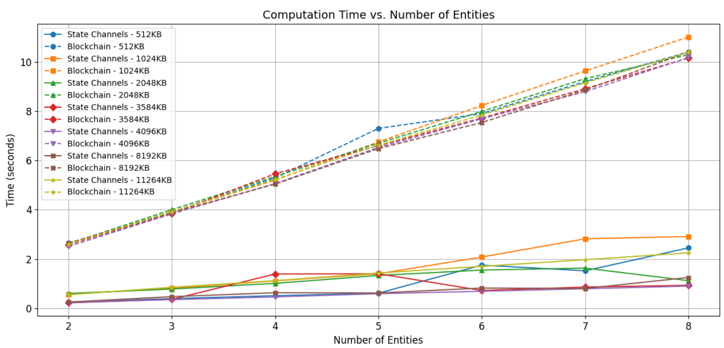

Fig. 6. Blockchain and State Channel Time Comparison with Increasing Healthcare Entities.

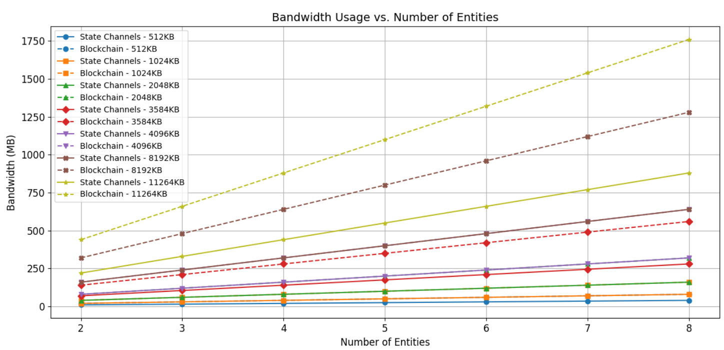

Fig. 7. Blockchain and State Channel Bandwidth Comparison with Increasing Healthcare Entities.

VI. Conclusion

In this paper, we proposed a scalability solution by integrating state channels with the Hyperledger Fabric blockchain. Different healthcare entities trained models on their heterogeneous devices using federated learning. These trained models are exchanged iteratively via established state channels to improve scalability and decrease the blockchain consensus time and network bandwidth consumption. In the end, the results demonstrate significant advantages in terms of time and bandwidth when using state channels with blockchains.

Acknowledgment

This work was supported in part by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the Italian MUR. Neither the European Union nor the Italian MUR can be held responsible for them.

References

[1] Mayer, A. Henrique, C. A. da Costa, and R.R. Righi. "Electronic health records in a Blockchain: a systematic review," Health informatics J., vol. 26, issue. 2, pp. 1273–1288, 2020

[2] Jafar, Uzma, M. J. A. Aziz, and Z. Shukur, "Blockchain for electronic voting system—review and open research challenges," J. MDPI Sensors, vol. 21, no. 17, 2021.

[3] Wang, Yajing, J. He, N. Zhu, Y. Yi, Q. Zhang, H. Song, and R. Xue, "Security enhancement technologies for smart contracts in the blockchain: A survey," Trans. on Emerging Telecommunications Technologies, vol. 32, no. 12, 2021.

[4] Jafar, Uzma, M. J. Aziz, and Z. Shukur, "Blockchain for electronic voting system—review and open research challenges,"J. MDPI Sensors, vol. 21, no. 17, 2021.

[5] Gudgeon, Lewis, P. M.Sanchez, S. Roos, P. McCorry, and A. Gervais, "Sok: layer-two blockchain protocols," in Financial Cryptography and Data Security: 24th International Conference, Kota Kinabalu, Malaysia, Springer International Publishing, pp. 201–226, February 2020.

[6] Albaseer, Abdullatif, B. S. Ciftler, M. Abdallah, and A. Fuqaha, "Exploiting unlabeled data in smart cities using federated edge learning," in 2020 International Wireless Communications and Mobile Computing (IWCMC), IEEE, pp. 1666–1671, 2020.

[7] Qu, Xidi, S. Wang, Q. Hu, and X. Cheng, "Proof of federated learning: a novel energy-recycling consensus algorithm," IEEE T. on Parallel and Distributed Systems, vol. 32, no. 8, 2021, pp. 2074–2085.

[8] Negka, L. D., and G. P. Spathoulas, "Blockchain state channels: a state of the art," IEEE Access, vol. 9, 2021, pp. 160277–160298.

[9] Albaseer, Abdullatif, B. S. Ciftler, M. Abdallah, and A. Al-Fuqaha. "Exploiting unlabeled data in smart cities using federated edge learning," in 2020 International Wireless Communications and Mobile Computing (IWCMC), IEEE, 2020, pp. 1666–1671.

[10] Alrebdi, Norah, A. Alabdulatif, C. Iwendi, and Z. Lian, "SVBE: searchable and verifiable blockchain-based electronic medical records system," Scientific Reports, vol. 12, no. 1, 2022.

[11] Deepa, Natarajan, Q. Pham, D. C. Nguyen, S. Bhattacharya, B. Prabadevi, T. R. Gadekallu, P. K. R. Maddikunta, F. Fang, and P. N. Pathirana, "A survey on blockchain for big data: approaches, opportunities, and future directions,"J. Future Generation Computer Systems, vol. 131, 2022, pp. 209–226.

[12] Coelho, Kristtopher, M. Nogueira, A. B. Vieira, E. F. Silva, and J. Augusto, M. Nacif. "A survey on federated learning for security and privacy in healthcare applications," J. Computer Communications vol. 207, 2023, pp. 113–127.

[13] Singh, M. Biken, H. Singh, and A. Pratap, "Energy-efficient and privacy-preserving blockchain based federated learning for smart healthcare system," IEEE Trans. on Services Computing, vol. 17, no. 5, 2023, pp. 2392–2403.

[14] Moore, Ervin, A. Imteaj, S. Rezapour, and M. H. Amini, "A survey on secure and private federated learning using blockchain: Theory and application in resource-constrained computing," IEEE J. Internet of Things Journal, vol. 10, no. 24, 2023, pp.21942–21958.

[15] Myrzashova, Raushan, S. H. Alsamhi, A. V. Shvetsov, A. Hawbani, and X. Wei, "Blockchain meets federated learning in healthcare: a systematic review with challenges and opportunities," J. Internet of Things Journal, IEEE, vol. 10, no. 16, 2023, pp. 14418–14437.

[16] Rebello, G. Antonio, G. F. Camilo, L. A. C. de Souza, M. Potop-Butucaru, M. D. de Amorim, M. E. M. Campista, and L. H. MK. Costa, "A survey on blockchain scalability: from hardware to layer-two protocol," J. Communications Surveys and Tutorials, IEEE, 2024.

[17] Zhang, Fan, S. Guo, X. Qiu, S. Xu, F. Qi, and Z. Wang, "Federated learning meets blockchain: state channel-based distributed data-sharing trust supervision mechanism," J. Internet of Things Journal, IEEE, vol. 10, no. 14, 2021, pp. 12066–12076.

[18] Iacoban, I. Paula, "StateFL: state channels powered federated learning," 2024.

[19] Zhu, Saide, Z. Cai, H. Hu, Y. Li, and W. Li, "zkCrowd: a hybrid blockchain-based crowdsourcing platform," IEEE Trans. on Industrial Informatics, vol. 16, no. 6, 2019, pp. 4196–4205.

[20] Yazdinejad, Abbas, A. Dehghantanha, H. Karimipour, G. Srivastava, and R. M. Parizi, "A robust privacy-preserving federated learning model against model poisoning attacks,"Trans. on Information Forensics and Security, IEEE, 2024.

[21] Nguyen-Phan, Trinh, V. Lemieux, and T. Bui, "A decentralized information governance Framework for implementing blockchain in healthcare: an exploratory analysis," 2025.