EasyChair Preprint

№ 10058

BETA-FL: Blockchain-Event Triggered Asynchronous Federated Learning in Supply Chains

Mayank Gulati, Narges Dadkhah, Benedikt Groß, Gerhard Wunder, Jovan Glavonjic, Aleksandar Pavlovic and Aleksandar Tomcic

May 10, 2023

BETA-FL: Blockchain-Event Triggered Asynchronous Federated Learning in Supply Chains

Mayank Gulati*, Narges Dadkhah*, Benedikt Groß*, Gerhard Wunder*,
Jovan Glavonjic†, Aleksandar Pavlovic†, Aleksandar Tomcic†
*Cybersecurity and AI Group, Freie Universität Berlin.
{firstname.lastname}@fu-berlin.de
†VizLore Labs Foundation, Novi Sad Serbia.
{firstname.lastname}@vizlore.com

***Abstract*—BETA-FL provides a distributed federated learning framework implemented for supply chains by tightly integrating private-permissioned blockchain for the trusted alliance among various actors involved in the supply chain. With a trusted ledger as the moderator in federated learning workflow, our approach ensures protection against malicious backdoor attacks on performance from both server and clients. Additionally, our asynchronous training regime allows scalability to a large number of federated clients with small and constant delay caused due to an event-triggering scheme. We showcase the milk powder classification task as a potential use-case in the food supply chain to avoid food wastage. Finally, we facilitate a dedicated channel for regulatory bodies in our blockchain environment for inspections and audits pertaining to the functioning of the supply chain.**

***Index Terms*—Blockchain, Hyperledger Fabric (HLF), Federated Learning (FL), Decentralized machine learning, Supply Chains, Asynchronous training**

I. Introduction

A supply chain is the network of various stakeholders, operations, information, and resources involved in order to facilitate flow of goods from suppliers to customers. To establish proper functioning, the supply chain demands a huge amount of trust, cooperation, and responsibility sharing among different actors in the chain. Such expectations are hard to achieve in a conventional setting, therefore blockchain has recently emerged as the de-facto approach to provide highly trusted decentralized ledger technology to ensure security, integrity, and auditability in supply chains also covered in the previous works [1], [2], [3], [4], [5]. Nevertheless, these works do not take advantage of the data-driven approach to maximize the overall efficiency of the supply chains.

The use of machine learning (ML) in modern supply chains can enable smart decision-making by capitalizing on the insights gained from learning through enormous amount of gathered data. Unlike the typical ML, where all the datasets are combined centrally for training compromises on data privacy, federated learning (FL) provides an innovative solution for distributed learning over multiple clients, each owning their local dataset to train local models. Rather than sharing raw data samples, these clients exchange trained local model updates in form of gradients with a global server, which are then aggregated by the server and returned back to the participating clients for further training. Subsequently, the clients train locally after receiving the aggregated model updates from the server, and this process follows until convergence.

The design of supply chains inherently involves distribution and a lack of trust between systems, making the implementation of blockchain technologies particularly suitable. By utilizing a distributed ledger, all transactions between supply chain stakeholders can be managed, and any necessary data can be stored in a manner that is accessible to all participating actors under controlled conditions. Different actors holding different roles and responsibilities within a consortium of supply chain members can be moderated through smart contracts and channels (discussed in sections II-B2 & II-B1) in a closed private blockchain. Such a blockchain network through its smart contracts and read-write permissions can be configured to accommodate all complex relationships between participating actors e.g. who has rights to perform specific actions, who has rights to access specific data, what type of transactions are supported and how are they validated and agreed.

Researchers have made attempts in [6], [7], [8] to integrate blockchain with FL in order to address the limitations of traditional FL. The main issues with standard FL include the inability to detect malicious trainers, who can disrupt the training process by uploading incorrect gradients, and the vulnerabilities associated with using a centralized server, such as the risk of a single point of failure and scalability problems.

Our framework takes it even further by providing closely integrated FL with blockchain to circumvent different kinds of backdoor attacks from server and client side. Using distributed trusted ledger as a mediator takes most of orchestration responsibilities from the server which in turn avoids potential attacks where server tries to steer gradients in certain direction or tries to collude with malicious clients. This happens as the information about which specific update belongs to which client is obfuscated from the server. Moreover, malicious activities from clients can be monitored and scrutinized with individual gradients checks before aggregation to cease them from sending poisonous updates. Additionally, the use of event-based asynchronous updates in FL can provide a flexible and

powerful architecture for building real-time systems, which can improve performance, reduce communication overhead, and enable continuous learning and adaptation. Unlike *BEAS* [7], our closely-knitted infrastructure facilitates asynchronous FL aggregation of model updates triggered by chain's events service defined based on the desired criteria (see II-B4) to provide full autonomy to the blockchain. Furthermore, our setup enables the efficient utilization of outdated gradients by preserving the record of updates on the distributed ledger, which otherwise would have been discarded.

Our work demonstrates the utilization of a private-permissioned blockchain, specifically Hyperledger Fabric (HLF [9]), in combination with FL to track the quality and safety parameters of food. Various organizations owning local datasets on quality parameters for milk powder participate in FL via a dedicated HLF channel. The objective is to enhance risk management capabilities and minimize food waste by delivering predictions regarding potential quality degradation and contamination based on measured food quality parameters and environmental conditions.

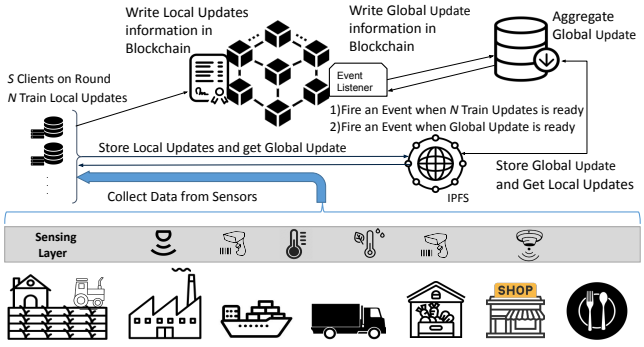

Figure. 1. Platform Architecture to demonstrate BETA β-FL with HLF, IPFS technologies used in a Food Supply Chain.

II. System Architecture

Hyperledger Fabric (HLF) [9] is the first open-source private blockchain tailored for permission-based networks within private environments that can be applicable to various business use-cases. To interact with the HLF, all participants need an identity and a valid certificate provided by certificate authorities (CA). HLF has a built-in Membership Service Provider (MSP) that enables organizations to authenticate and authorize users based on their digital certificates. This ensures that only trusted and authorized nodes can participate in the network, making it difficult for attackers to launch Sybil attacks [10] by creating fake identities. HLF provides identity management tools that allow network administrators to manage and control access to the network. For example, administrators can use role-based access controls (RBAC) to limit the privileges of users based on their roles and responsibilities, preventing malicious actors from gaining too much control over the network.

HLF network primarily consists of three types of nodes: client, peer, and ordering-service-node or orderer. A client is a type of node that creates and submits a transaction to the endorsers and broadcasts it to the ordering service. Peers are the fundamental nodes of the network. They are responsible for committing transactions and holding the ledger, which consists of transactional data. Finally, the orderer is responsible for generating a block and maintaining the consistency of the blockchain. HLF stands out from other blockchain types [11] due to its use of deterministic consensus algorithms. These algorithms ensure the accuracy and finality of any approved block by a peer, resulting in higher transactional throughput than its public blockchain counterparts. Overall, the generic components of HLF provide a highly reliable and secure platform for our work.

HLF employs Transport Layer Security (TLS) [12] to enable secure communication between nodes. This is achieved through the use of TLS certificates, which allow for Fabric nodes and clients to sign and encrypt their conversations. As a prerequisite for any channel communication within HLF, a valid TLS Certificate must be present. Additionally, the auditing of transaction initiators within HLF makes it possible to trace transactions and identify any suspicious activities.

A. Supply chain infrastructure

Our supply chain infrastructure utilizes checkpoints to monitor progress at every stage, collecting data that includes indicators of food quality degradation (such as increased microbial counts or adulteration) and contamination. This system allows for real-time insights into the condition of food products, with each checkpoint recorded on a ledger. Predictive analysis can be used to anticipate potential contamination points, while the data collected can also be used to analyze supply chain efficiency, identify bottlenecks, and optimize the process, leading to reduced costs and improved productivity. Real-time monitoring provides a continuous and accurate record of transportation and storage conditions, enhancing traceability.

The process of measuring food quality involves the use of measurement checkpoints (depicted in Fig. 1), which consist of two main systems: an Internet of Things (IoT) environmental sensing system and a photonic system. The IoT sensing system comprises an IoT edge controller and sensors that measure the humidity and temperature. On the other hand, the photonic food scanner, which is based on the device from *H2020 project PhasmaFOOD*[1] that uses a combination of Ultraviolet, Visible, and Near Infrared spectrometers and a camera. These devices take measurements in quick sequence from the same point, providing a comprehensive analysis of the food quality parameters. The dataset utilized for the FL use case consists of these spectrograms that is elaborated in II-D.

B. Blockchain related Components

1) Channels: As opposed to a permissionless environment, where all information is stored on a public ledger necessitating the storage of a copy of all the data by every entity in the chain, a permissioned-based environment allows to carry out

[1]https://phasmafood.eu/

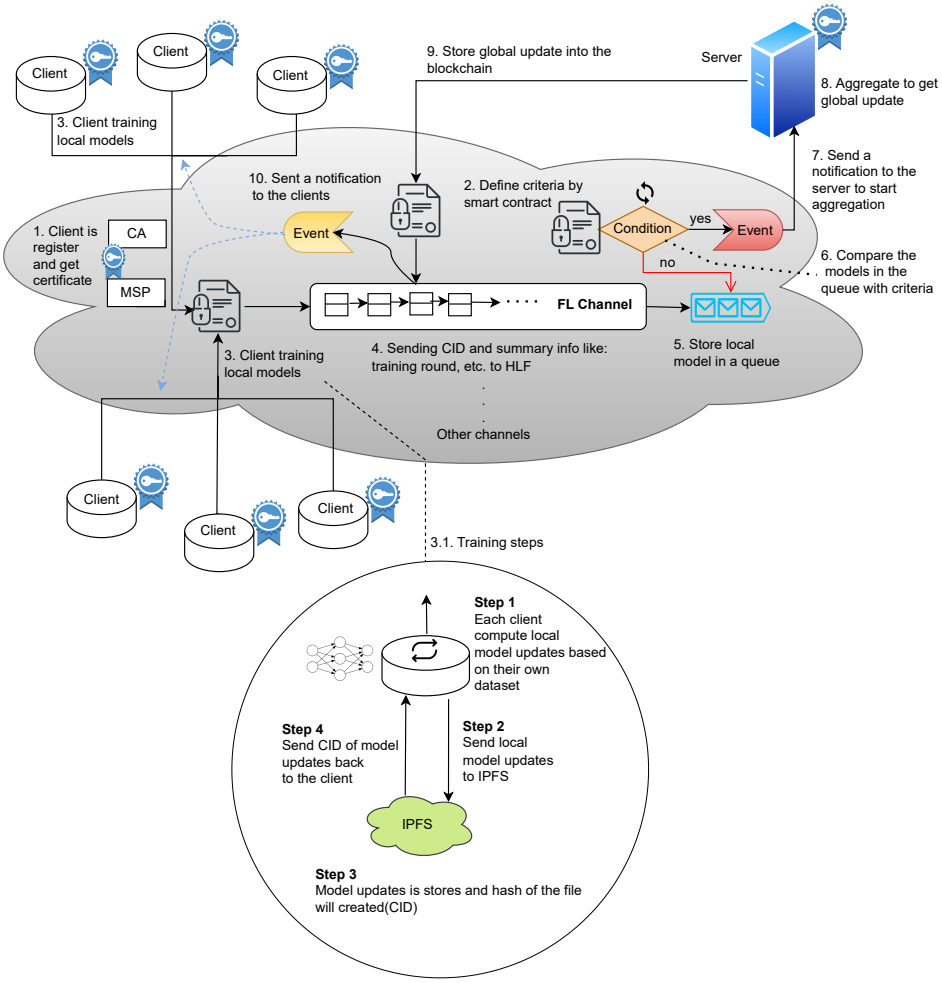

Figure. 2. Intertwined Blockchain and FL architecture.

private and confidential transactions. Therefore, HLF employs channels to provide a key abstraction for the shared information among different parties. A channel is a private "subnet" of communication pathways between two or more specified network participants. The use of HLF enables the restriction of access to individual channels, where members are only able to store information for the channel they are associated with. Within this architecture, various channels are defined for specific purposes. e.g. the Federated Learning Channel (FLC) is designated for tasks related to federated learning. The User and Resources Channel is for registering resource entities of supply chain actors, while the Supply Chains Channel is for creating checkpoints in the product cycle. Additionally, the Data Sharing Channel is available for direct transmission (if needed) of data readings from sensors.

2) Smart Contracts: Smart contracts are business logic and executable contracts that define a set of rules for negotiating and these are invoked by clients who want access to the network. In our architecture, we use a different smart contract (in HLF terms, also referred as part of the *"chaincode"*) for each channel e.g. we use a chaincode (typically used to group related smart contracts for deployment) to create a transaction for a model to be stored on HLF at the FLC. Another advantage of implementing the chaincode on the FLC in our setup is the ability of accessing fine-grained models' information (local model and global model updates) of any particular round of FL from the chain in any desired manner.

3) HLF's Channel-based event services: To ensure precise control over peers' data at the channel level, our architecture employs channel-based event-triggering schemes. An asynchronous FL workflow is carried out using a dedicated FLC to read and write FL global and local updates to/from the blockchain. Events are triggered based on specific criteria outlined in II-B4, such as when the required number of local model updates are registered on the chain to enable model aggregation or when a global model update is recorded on the chain to initiate subsequent rounds of training. The ledger maintains a queue of local model updates for each FL round until the criteria is met, after which the queue is reset. The channel configuration facilitates requests to listen

to these events from outside the peer's organization. This channel-based event service not only provides a high degree of reliability but also ensures that events are not missed due to issues such as connectivity or peers joining a network that is already running.

4) Criteria: Based on pre-decided criterion events can be defined to facilitate the aggregation step in FL which is quite flexible and can be modified (as per need) before the next round of training. Only after the desired criterion is fulfilled an event is fired that would in turn allow the FL Server to do the model updates aggregation. Some examples of criteria used in our experiments are: completion of training of a minimum number of local FL clients, the inclusion of FL clients from the given organization, the inclusion of specific FL clients based on identities created by CA, a minimum number of model updates from current FL round and from previous rounds that were unused to include contributions of straggler FL clients. By delegating the responsibility of aggregating local model updates to the distributed ledger, this approach enables the ledger to determine when aggregation should occur (based on events) and which specific local model updates to use. These determinations are made in accordance with predefined criteria that have been agreed upon by all participants and outlined in the smart contract.

5) IPFS: To overcome a significant challenge posed by the high cost of on-chain storage in the blockchain when working with large and numerous FL models, we use the InterPlanetary File System (IPFS) [13]. It is an off-chain model distributed storage solution that has no single point of failure and provides content-addressed block storage to store and retrieve data efficiently. Prior works such as [14], [15] also motivate the use of IPFS to off-load huge storage demands from the chain. Since IPFS uses content identifier (CID) generated based on the content's cryptographic hash to refer to its content which also helps to avoid duplicate redundant updates. As a result, combining this with transactions on ledger aids in avoiding potential attacks such as Lazy Clients, as described in [16], in which some clients attempt to reduce their computation costs by replicating model updates from other honest clients. So, we log model update transactions created with this short CID in the HLF instead of actual FL model updates in our setup.

C. Federated Learning Implementation

FL is a distributed learning regime with physically distinct clients $\{1, 2, ..., N\}$ that work collaboratively to train a model from a large dataset $\mathcal{D}$ formed out of N independent non-overlapping data subsets $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_N$, i.e. $\mathcal{D} = \bigcup_{k=1}^{N} \mathcal{D}_k$. Each client k trains its local model w^k using its own local dataset $\mathcal{D}_k$ with m_k data samples. We follow the similar approach as in FedAvg [17], where a subset of S clients are randomly selected out of N in every round after initializing with a global model in the beginning. These clients perform local training using stochastic gradient descent (SGD) for a certain number of local iterations to update $w^k \leftarrow w^k - \eta \nabla F_k(w^k)$ with η as learning rate by solving local objective in (1):

$$F_k(w) = \frac{1}{m_k} \sum_{i \in \mathcal{D}_k} f_i(w) \tag{1}$$

where $f_i(w) = \ell(x_i, y_i)$ loss prediction on sample (x_i, y_i). Overall, global objective function in (2) is the contribution of individual local objectives weighted by fraction of local data sample size over total data sample size.

$$f(w) = \sum_{n=1}^{N} \frac{m_k}{m} F_k(w) \tag{2}$$

So, in each round r all participating client $\mathcal{C}_k$ present in S compute gradient update $g_k = \nabla F_k(w_r)$ and the server uses the gradients from these clients for aggregation and updates the global model $w_{r+1} \leftarrow w_r - \eta \sum_{k=1}^{S} \frac{m_k}{m} g_k$ and client update is given by $w_{r+1}^k \leftarrow w_r - \eta g_k$. In our setting, these model updates are administered through HLF where all local updates are written on the chain asynchronously (with hash on ledger & its corresponding updates on IPFS) and aggregation on the server is triggered when the desired criteria are fulfilled described by events. Algorithm 1 captures complete details of our implementation of BETA β-FL which is complimented by Fig 2.

D. Experimental Setup

The experiment simulates a use case aimed at identifying adulteration in skimmed milk powder during its journey through the supply chain. The blockchain network records the milk powder's progress through the supply chain with the help of defined measurement checkpoints. The milk powder is susceptible to various methods of adulteration, such as the addition of neutral fillers to increase its volume. To detect any such fillers, we employ spectrometry, comparing the spectral images of the adulterated samples with those of the pure skimmed milk powder.

Our blockchain implementation is created using HLF, with smart contracts written in *node.js*. We make use of HLF infrastructure to define channels such as FLC, etc. and channel based event services. Additionally, we rely on *CouchDB* as the underlying blockchain database. Our evaluation is performed on a Google Cloud Server, where each entity is deployed in separate Docker containers. The server comprises 2 CPUs, 8 GB of RAM, and 35 GB of HDD.

In general, we have the following organizations: Supply Chain Actors (Seed sellers, Farmers, Distributors, Processors, Storage Owners), Wholesalers Retailers, Consumers, and Regulatory bodies. We have two peers per organization: one for endorsing and other for committing. These peers are nodes on the network that host a copy of the ledger and participate in the consensus process to validate transactions and update the ledger. Endorsing peer validates and endorses transactions, while committing peer receives and validates the endorsed transaction before updating their copy of the ledger. These peers play crucial role in the consensus process to ensure all copies of the ledger are consistent and tamper-proof.

Algorithm 1: BETA β-FL Algorithm

1: **Input:** Set of N participating clients, where client $\mathcal{C}_k$ holds its own local dataset $\mathcal{D}_k$ for local training with local model parameter w^k.
Parameter: learning rate η, sampling fraction $\frac{S}{N}$, local epochs E, local minibatch size B, total rounds R.
2: **Initialise** Randomly sample S clients out of total N and initialise global model weights (w_0) at $round = 0$.
3: **for** each round $r = 1, 2, ..., R$ **do**
4: Define local model event's condition for triggering aggregation, see II-B4.
5: **for** each client $\mathcal{C}_k$ from set of S clients in parallel **do**
6: Train client on local dataset $\mathcal{D}_k$ with batch size B and learning rate η for E epochs.
7: After training log the local gradients (g_k) on IPFS and its hash on HLF along with other attributes: client id, round number, dataset size, etc. and populate the queue.
8: **end for**
9: **if** Criterion is fulfilled **then**
10: Fire local model event to trigger gradients aggregation process at server. Gradients utility is checked before to avoid redundant, null updates, etc.
11: **end if**
12: **if** Aggregation is completed **then**
13: Log current global model w_r on IPFS with its hash on HLF and fire global model event to initialise local models with w_r by reading it from IPFS with hash provided from HLF.
Now, $r = r + 1$ and proceed from Step 4.
14: **end if**
15: **end for**
16: **Output:** Entire history of local and global updates on IPFS and its corresponding summary on HLF.

In this research, we employed a *Hamamatsu C12880MA*[2] spectrometer under UV illumination for visible spectra (400-700 nm) of pure and adulterated skimmed milk powder. These spectrograms form our dataset (seen in Fig. 3) with 4 labeled classes with varying amounts of adulteration from pure to completely adulterated achieved by mixing with several protein powders (e.g. whey, soy, and hemp proteins) in different concentrations.

To simplify matters, we have divided the independent and identically distributed (IID) dataset into four client organizations - producers, storage owners, retailers, and wholesalers - with the same feature and label space. This horizontal split can be expanded to include more clients and organizations. It should be noted that relying solely on training data from a single client would not be sufficient due to the limited amount of labeled data available at each client. This highlights the importance of collaborative learning through FL, which enables the utilization of knowledge learned from other clients. Each of the four clients trains a model to perform a classification task. The model comprises three one-dimensional Convolutional layers, with a kernel size of 10 and a stride of 2, followed by a Linear layer for the four labeled classes. The implementation is done using PyTorch [18].

[2]www.hamamatsu.com/eu/en/product/optical-sensors/spectrometers/mini-spectrometer/C12880MA.html

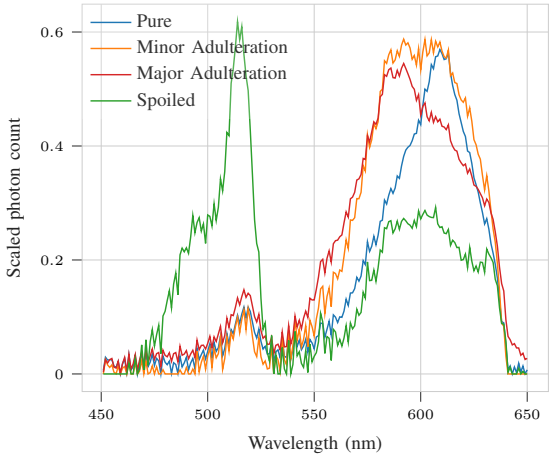

Figure. 3. Spectrogram dataset for milk powder with 4 classes.

TABLE I
AVERAGE ACCURACY ON LAST ROUND (10TH) IN EXPERIMENTS

Exp Nr.	Global	Client 1	Client 2	Client 3	Client 4
Experiment 1[a]	0.967	0.9686	0.9686	0.9682	0.967
Experiment 2[b]	0.967	0.9634	0.9684	0.9656	0.9642
Experiment 3[c]	0.972	0.9709	0.9724	0.9704	0.968

[a]Synchronous FL training without blockchain.
[b]Synchronous FL training with reading & writing models on blockchain.
[c]BETA β-FL.

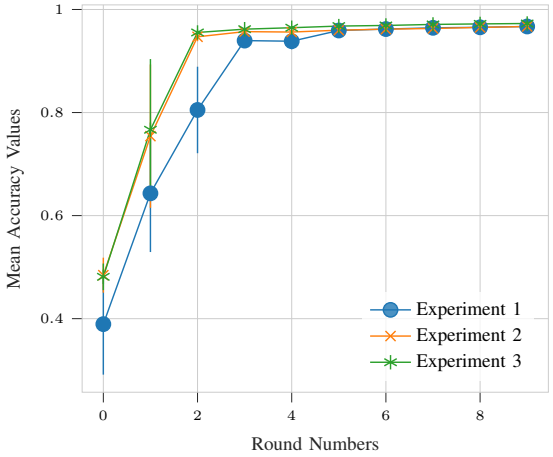

Figure. 4. Mean and standard deviation of Accuracy on Global Server.

III. Results and Conclusions

We conducted three types of experiments: standard synchronous FL without blockchain, synchronous FL training

with model updates stored on HLF and IPFS, and our BETA β-FL which allows asynchronous training using event-triggered services. All experiments utilized the same hyperparameters, with local epochs (E) set to 5, mini-batch size (B) at 32, and learning rate (η) of 0.001, trained for 10 FL rounds. Results from Fig. 4 and Table I show that there is little difference in accuracy between the three experiments. However, our approach offers a slight improvement as it avoids redundant model updates. Evaluation of the global model was conducted on a separate validation milk powder dataset.

To analyze the experiments, the FL pipeline was divided into different parts to determine the execution times of various operations, including local client training, writing trained local updates to HLF via IPFS, and the delay caused by using the events service in the BETA β-FL approach. Results showed that the average local training time per client is 0.833 seconds, while writing these local gradients to HLF via IPFS takes about 8.736 seconds, and the average delay caused by the events service is 2.5134 seconds. These findings suggest that clients should train more local epochs before sharing updates to the ledger via IPFS. The asynchronous setup parallelizes operations using the event service, which offsets the fixed cost of the delay in listening to events happening twice in every round. Overall, parallelizing these operations achieved significant gains, despite the training time and time to write/read model updates to/from IPFS scaling with the size of model parameters.

The highlight of our BETA β-FL approach is that it provides reliable, secure, and auditable FL infrastructure that defends against malicious threats from both server and clients. Our utilization of a decentralized HLF within an asynchronous setup results in a solution that is highly scalable, trustworthy, and tamper-proof, albeit with a minor overhead related to events service. This approach grants HLF more authority, which opens up avenues for future research, such as utilizing previously unused stale updates from slow clients to achieve faster convergence. Additionally, other smart contracts can be written to update hyperparameters of slow-performing clients, such as reducing their local epochs, changing the batch size, etc. to accelerate their training.

Our architecture is built on a private permissioned blockchain (HLF), which ensures data confidentiality among multiple supply chain stakeholders who may be competitors or do not trust each other. However, organizations may need to provide supply chain information to regulators to comply with legal requirements, as mentioned in [2]. To address this, our platform includes a dedicated channel for regulatory bodies to access relevant information and evaluate supply chain performance. Furthermore, starting in 2023, the German federal government plans to implement a new supply chain act[3], which mandates companies to establish grievance mechanisms and report on their activities to ensure improved human rights protection.

[3]https://www.bundesregierung.de/breg-en/federal-government/supply-chain-act-1872076

IV. Acknowledgements

This work is funded by the German Federal Ministry of Education and Research (BMBF) under TinyPART (16KIS1395K) and FT-Chain (01DS21011) projects.

References

[1] S. Malik, S. S. Kanhere, and R. Jurdak, "Productchain: Scalable blockchain framework to support provenance in supply chains," in *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, pp. 1–10, 2018.

[2] E. Wagner, R. Matzutt, J. Pennekamp, L. Bader, I. Bajelidze, K. Wehrle, and M. Henze, "Scalable and privacy-focused company-centric supply chain management," in *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pp. 1–5, 2022.

[3] R. W. Ahmad, K. Salah, R. Jayaraman, I. Yaqoob, M. Omar, and S. Ellahham, "Blockchain-based forward supply chain and waste management for covid-19 medical equipment and supplies," *IEEE Access*, vol. 9, pp. 44905–44927, 2021.

[4] A. Park and H. Li, "The effect of blockchain technology on supply chain sustainability performances," *Sustainability*, vol. 13, no. 4, 2021.

[5] H. Wu, J. Cao, Y. Yang, C. L. Tung, S. Jiang, B. Tang, Y. Liu, X. Wang, and Y. Deng, "Data management in supply chain using blockchain: Challenges and a case study," in *2019 28th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–8, 2019.

[6] X. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, "Flchain: A blockchain for auditable federated learning with trust and incentive," in *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*, pp. 151–159, 2019.

[7] A. Mondal, H. Virk, and D. Gupta, "Beas: Blockchain enabled asynchronous amp; secure federated machine learning," 2022.

[8] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2020.

[9] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, S. Muralidharan, C. Murthy, B. Nguyen, M. Sethi, G. Singh, K. Smith, A. Sorniotti, C. Stathakopoulou, M. Vukolić, S. W. Cocco, and J. Yellick, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, (New York, NY, USA), Association for Computing Machinery, 2018.

[10] J. R. Douceur, "The sybil attack," in *Peer-to-Peer Systems: First InternationalWorkshop, IPTPS 2002 Cambridge, MA, USA, March 7–8, 2002 Revised Papers 1*, pp. 251–260, Springer, 2002.

[11] Z. Wang and Q. Hu, "Blockchain-based federated learning: A comprehensive survey," *arXiv preprint arXiv:2110.02182*, 2021.

[12] H. Krawczyk, K. G. Paterson, and H. Wee, "On the security of the tls protocol: A systematic analysis," in *Advances in Cryptology – CRYPTO 2013* (R. Canetti and J. A. Garay, eds.), (Berlin, Heidelberg), pp. 429–448, Springer Berlin Heidelberg, 2013.

[13] J. Benet, "IPFS - content addressed, versioned, P2P file system," *CoRR*, vol. abs/1407.3561, 2014.

[14] N. Kawaguchi, "Application of blockchain to supply chain: flexible blockchain technology," *Procedia Computer Science*, vol. 164, pp. 143–148, 2019.

[15] V. Mothukuri, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and K.-K. R. Choo, "Fabricfl: Blockchain-in-the-loop federated learning for trusted decentralized systems," *IEEE Systems Journal*, vol. 16, no. 3, pp. 3711–3722, 2022.

[16] C. Ma, J. Li, L. Shi, M. Ding, T. Wang, Z. Han, and H. V. Poor, "When federated learning meets blockchain: A new distributed learning paradigm," *IEEE Computational Intelligence Magazine*, vol. 17, no. 3, pp. 26–33, 2022.

[17] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016.

[18] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.