frontiers | Frontiers in Blockchain

TYPE Original Research
PUBLISHED 08 August 2022
DOI 10.3389/fbloc.2022.893747

OPEN ACCESS

EDITED BY
George M. Giaglis,
University of Nicosia, Cyprus

REVIEWED BY
Meghana Kshirsagar,
University of Limerick, Ireland
Henry Michael Kim,
York University, Canada

*CORRESPONDENCE
James E Short,
jshort@ucsd.edu

SPECIALTY SECTION
This article was submitted to Blockchain Technologies,
a section of the journal
Frontiers in Blockchain

RECEIVED 10 March 2022
ACCEPTED 29 June 2022
PUBLISHED 08 August 2022

CITATION
Short JE, Miyachi K, Toouli C and Todd S (2022), A field test of a federated learning/federated analytic blockchain network implementation in an HPC environment.
Front. Blockchain 5:893747.
doi: 10.3389/fbloc.2022.893747

A field test of a federated learning/federated analytic blockchain network implementation in an HPC environment

James E Short[1,2]*, Ken Miyachi[1,2], Christian Toouli[1,2] and Steve Todd[3]

[1]BlockLAB, San Diego Supercomputer Center, San Diego, CA, United States, [2]University of California, San Diego, San Diego, CA, United States, [3]Dell Technologies, Round Rock, TX, United States

The rapid upswing in interest in federated learning (FL) and federated analytics (FA) architectures has corresponded with the rapid increase in commercial AI software products, ranging from face detection and language translation to connected IOT devices, smartphones, and autonomous vehicles equipped with high-resolution sensors. However, the traditional client-server model does not readily address questions of data ownership, privacy, and data location in the context of the multiple datasets required for machine learning. In this paper, we report on a pilot distributed ledger and smart contract network model, designed to track analytic jobs in an HPC supercomputing environment. The test system design integrates the FL/FA model into a blockchain-based network architecture, wherein the test system records interactions with the global server and blockchain network. The design goal is to create a secure audit trail of supercomputer analytic operations and the ability to securely federate those operations across multiple supercomputer deployments. As there are still relatively few real-world applications of FL/FA models and blockchain networks in use, our system design, test deployment, and sample code are intended to provide interested researchers with exploratory tools for future research.

KEYWORDS

federated learning, federated analytics, blockchain, distributed ledger technology, data governance, HPC

Introduction

Continued advancements in data science promise to bring improved decision analytics and privacy-preserving technologies to data architectures and workflows for analytic processing. Federated learning (FL) refers to an emerging machine-learning paradigm where multiple machines train models locally on a distributed dataset and compute a global model based on local model updates (Lo et al., 2020). The federated learning model is an example of the more general approach of "bringing the code to the data, instead of the

data to the code" (Bonawitz and Eichner, 2019; Truex et al., 2019). Operationally, the FL protocol iteratively requests random clients to download a trainable machine learning (ML) model from a server, update it with their own data, and upload the changes to the model back to the server. Concurrently requesting the server to aggregate multiple client updates to improve the model. The design of this distributed data and processing architecture is intended to address problems in data locality, data privacy, and data ownership.

The rapid upswing in interest in FL architectures has corresponded with the rapid increase in commercial AI software products, ranging from face detection and language translation on consumer smartphones, to voice recognition and speech synthesis used in virtual assistants such as Amazon Alexa and Google Home. IoT devices, smartphones, and autonomous vehicles equipped with high-resolution sensors and connected to high-speed networks are seen as commercially promising data collection platforms. However, the typical mobile edge computing paradigm assumes all data resources are transferred from the IoT client device (e.g., a smartphone) to the computational platform (e.g., a data server) through a cellular network. Federated learning models, and their first derivative, federated analytics models, have been developed to address locality, privacy, and data ownership concerns in distributed machine-learning models and use cases.

In this paper, we report on a pilot distributed ledger and smart contract system, designed to track analytic jobs in an HPC supercomputing environment. The technical design integrates the federated learning (FL) and federated analytics (FA) models into a blockchain network model, wherein the system records interactions with the global server and blockchain network. The benefit of combining these technologies is the immutability of the access and modification logs, such that any breach of policy is recorded without failure and privacy can be defined across participants in the network. This benefit may provide benefits over traditional FL/FA systems wherein the privacy and security of data are part of the system infrastructure. Our goals included first, to defining, configuring and deploying a workable FL/FA model and blockchain network in an HPC supercomputing environment; second, defining and testing key operating parameters of the network; and third, assessing results and describing our system design, test procedures, and example code in sufficient detail for researchers to replicate our test environment and results. As there are still relatively few real-world, commercial applications of FL/FA models and blockchain networks in use, our system design and deployment is intended to help spark further research into the integration and application of both technologies.

We organize our paper into five sections. First, we define the basic paradigms for federated learning (FL) and federated analytics (FA), briefly tracing their origins. Second, we summarize a selected sample of key technical articles. Third, we discuss design considerations in building our test environment and describe how we ran simulations of supercomputer jobs in this environment. We then discuss how we obtained results by capturing state information from the system, and how we analyzed state transitions and storage from smart contract execution used for access and modification logs. Finally, abstracting from lessons learned, we comment on applicable use-cases and directions for future research.

Our work was carried out at BlockLAB, the blockchain research laboratory at the San Diego Supercomputer Center, UC San Diego, with research, systems, and administrative support from Dell Technologies, and VMware and Boomi business units within Dell. We worked closely with Dell Boomi web services and VMware to execute smart contracts on the VMware blockchain service in organizing our test system environment.

Federated learning and federated analytics: technical and organizational review

Recent advances in federated learning (FL) and federated analytics (FA) continue to drive an expanding literature. The large and growing technical literature covers FL and FA system development, while the organizational literature addresses team and organizational design criteria in FL/FA implementation.

Field reviews of FL system design and software engineering

In a systematic review and classification of FL studies, Lo et al. (2020) reviewed 231 primary studies covering the software engineering aspects of FL system development. The authors followed the review methodology described in Kitchenham and Charters, (2007) organizing articles into four areas—Background Understanding, Requirement Analysis, Architecture Design, and Implementation and Analysis. A Software Development Life Cycle (SLDC) for FL was derived from combining practices in traditional software development with those of machine learning system development, as shown in Figure 1.

The primary purpose of Lo and colleagues was to cross-classify their primary sample of 231 articles into active research topics and open problems. Lo outlines key architectural components of the FL system, evaluation metrics for FL performance, and outlines open problems and future trends in FL, including enterprise and industrial-level implementation and adoption, and tradeoffs between data privacy and model system performance.

A second comprehensive review of federated learning is presented in Kairouz et al. (2019). Their review is a composite of thirty individual papers presented at the Workshop on Federated Learning and Analytics held at Google's Seattle office in June 2019. In their introduction, the editors state that a key property of FL is its inherent interdisciplinary nature, integrating models and approaches derived from, for example, machine learning, distributed optimization, cryptography, security, and differential privacy. Initially, FL addressed data locality and privacy in mobile and edge computing applications.

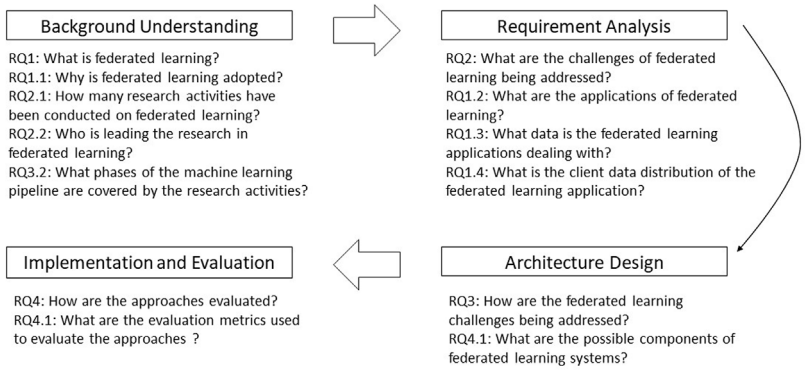

FIGURE 1
Question groups defining the FL software development life cycle (SDLC). Source: Lo et al., A systematic literature review on federated machine learning from a software engineering perspective (2017).

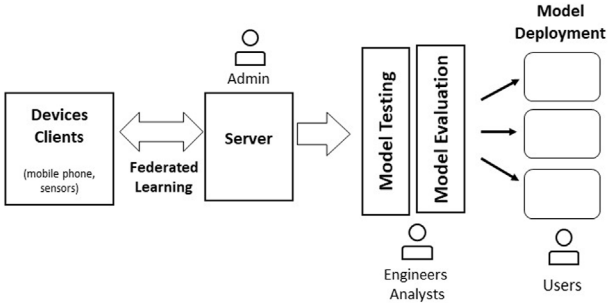

FIGURE 2
The basic federated learning model framework.

In proposing a broader definition, the authors define two FL settings, "cross-device" and "cross-silo", respectively:

Federated learning is a machine learning setting where multiple entities (clients) collaborate in solving a machine learning problem under the coordination of a central server or service provider. Each client's raw data are stored locally and not exchanged or transferred; instead, focused updates intended for immediate aggregation are used to achieve the learning objective.

The basic federated learning model

The basic FL model, typically developed by a model engineer for a particular application, is shown in Figure 2. The primary activities, stakeholders, and workflow break down into six steps:

1 Problem identification: the model engineer defines the problem to be solved using FL.
2 Device (client) instrumentation: in a mobile—edge computing application, the clients (an application running on the mobile phone) are instrumented to store the necessary training data locally. For example, text messages or photos.
3 Simulation prototyping: the model engineer may prototype model architectures and test learning parameters using a proxy dataset.
4 Model training: multiple FL tasks are started to train different models, using different optimization parameters.
5 Model evaluation: after the specified period of time necessary to train the models, the models are analyzed to select the best candidates defined by selection criteria.
6 Model deployment: deployment includes standard activities, including debugging, quality tests and A/B testing (comparing two variations of a model against "best" criteria), and activities defined by context, specifically the criteria used to determine "best" and fit to purpose.

Research questions

Algorithmic questions remain open in the real-world usability of decentralized approaches for machine learning. Some questions are analogous to the special case of federated learning with a central server, while others are specific to the fully decentralized or "trustless" model.[1] Kairouz et al. (2019) summarize four key areas:

1 Effect of network topology and asynchrony on decentralized SGD (stochastic gradient descent, a smoothing function):

1 "Trustless" systems, typically associated with blockchains, are systems where "trust" is distributed among different actors in the blockchain network using an economic model that incentivizes actors to cooperate with the rules defined by the consensus algorithm. However, the term is often criticized as being imprecise.

denser networks encourage faster consensus and give better error convergence rates, at the cost of incurring communication delays that increase with the number of nodes.

2 Local-update decentralized SGD: this is basically the problem of designing an efficient convergence model when running an optimization—smoothing function (SGD) in a decentralized network.

3 Personalization and trust mechanisms: this is a related class of problems posed by the fully decentralized environment. The authors summarize: "The use of incentives or mechanism design in combination with decentralized learning is an emerging and important goal, which may be harder to achieve in the setting without a trusted central server."

4 Gradient compression and quantization methods: a final algorithmic challenge is in redesigning compression-communication protocols, originally designed and optimized for centralized server systems, for the fully decentralized environment. The performance and cost of the FL approach are severely affected by network performance and transmission costs.

The work reported here focuses on topic areas 2 and 3. In future research, we plan to address topics 1 and 4.

Practical questions in combining FL/FA and blockchain technology

Combining FL and blockchain technology, in principle, could enable decentralization of the global server by using smart contracts to do model aggregation. Participating clients executing the smart contracts could secure the use of FL models and data throughout the system. However, there is the logical inconsistency that data available on public blockchain platforms such as Ethereum is available to all network clients by default, and a principal motivation of the decentralized FL protocol is the protection of data privacy. How might existing privacy-preserving protocols be modified to fit the decentralized FL scenario? Kairouz et al. (2019) identify three approaches: first, to prevent the participating nodes from exploiting individually submitted model updates, existing secure aggregation protocols could be used. A practical secure aggregation protocol already used in cross-device FL was proposed by Bonawitz and Eichner, 2019; effectively handling dropped out participants at the cost of complexity of the protocol.

Alternatively, another approach might have each client stake a deposit of cryptocurrency on the blockchain and suffer a penalty if they drop out during the execution. A third approach to achieving secure aggregation would be to use confidential smart contracts—for example, what is enabled by the Oasis Protocol (Oasis Protocol Foundation, 2022) executing within secure enclaves. In this system, each client could submit an encrypted local model update, wherein only specific secure hardware could decrypt and aggregate the encrypted model through remote attestation. (Cheng et al., 2019). Utilizing blockchain technology with FL/FA could provide a secure, automated, and auditable process to update models.

Configuring the federated analytics/ blockchain test environment

In this section, we present the Federated Analytics (FA) test environment created in BlockLAB with Dell partners VMware and Boomi. We first discuss the key considerations in designing an FA/blockchain system in an HPC environment. We then discuss configuring and testing the system under test (SUT).

What is needed: FA/blockchain design considerations

As summarized in Figure 3, designing and configuring a FA/blockchain system involves a number of considerations appropriate for the SUT test environment. The initial consideration is to determine the system components and benefits under test. Generally, federated analytics jobs will consist of many participants sharing the same resource or many participants contributing to the same resource. In our test case, we modeled our use cases along the lines of multiple researchers using the SDSC HPC system for conducting work. In this scenario, FA benefits would include generating a secure audit trail of what the supercomputer was used for, and the blockchain controlling access to analytic and operational data generated by the FA jobs running in the SDSC HPC system.

An early and widely discussed baseline federated learning use case is where the FL model enables mobile phones to collectively learn a shared prediction model while keeping all the training data on the device (McMahan and Ramage, 2017). The design consideration in this case is to decouple the ability to do machine learning from the need to store data in a centralized data store, for example, in the cloud. To do this, a machine learning model is downloaded onto user phones in the FL network, then executed locally, with the results from the model sent back to the analyst. Sharing only the model results, not the data, helps to ensure privacy while allowing for improved models.

Our system under test (SUT) involved a series of design considerations. The first was to determine the access structure of the blockchain system. A public blockchain would allow anyone to run nodes that verify the data, which is problematic in this case due to the possibility that malicious actors in the network could send false information. Access control is an important requirement of FL/FA and, therefore, we chose the alternate model of a permissioned blockchain. Here, network participants are first required to be verified. Participant roles are then assigned

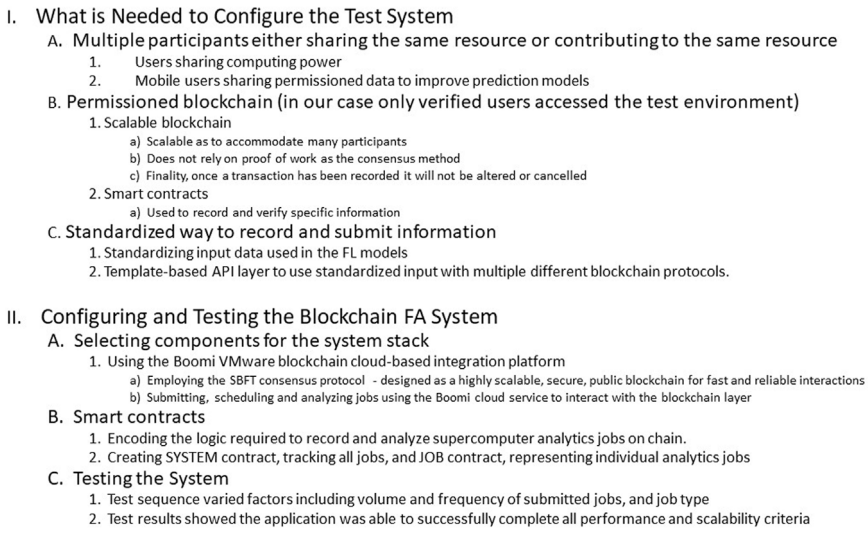

FIGURE 3
Design considerations in configuring an FA Blockchain system.

to determine the level of data visibility each participant in the network is permitted.

The second design consideration was to define how smart contracts should be organized and created. Smart contracts refer to self-executing code that specifies the exchange rules for network transactions. We chose a very straightforward smart contract design that performed access control and the storing of an audit trail of data access and modifications. The secure nature of the access control and audit trail offer significant improvements to the data privacy and security of FL/FA models by utilizing self-executing programmatic rules to enforce privacy and security.

A third set of design considerations refers to how data producers are connected on the blockchain. It is desirable to design the test system to work only when there is "good" data being reported out as transactions are executed. "Good" data refers to data that is accurate—a measure of data quality—and "truthful"—a composite measure of node consensus that the data submitted to the chain is verifiable, reliable, and accessible by all participants in the network (Awan et al., 2019; Lo et al., 2020; Mugunthan et al., 2020). The most direct way to connect actions performed by users is by making API calls to a network server that, in turn, then submits the updated data to the blockchain. Further, it is desirable to build these API calls into the underlying test system so that all actions are automatically reported to the transaction ledger.[2]

The integration of data quality metrics associated with access control and historical data access and modification logs enables high-trust analytics regarding information in the system without exposing the underlying data itself. This is an improvement over traditional FL/FA systems where data quality is very difficult to determine without first being exposed to the raw data. Furthermore, blockchain technology has the potential to utilize Zero-Knowledge proofs, which could further abstract the data from a data quality score.

Creating and testing the FA—blockchain test environment

The BlockLAB development team first created and tested a system for recording analytic jobs performed on the SDSC supercomputer and then wrote that data onto a blockchain. The team then partnered with Dell units Boomi and VMware to use the VMware blockchain stack hosted in a cloud offering and fronted by Boomi's web interface. Our test system architecture is shown in Figure 4. Smart contracts created by the Dell and BlockLAB development teams enabled analytic job information to be transmitted and stored in the VMware ledger (Project Concord, 2018; Gueta et al., 2019; VMware Blockchain Group, 2022).

The VMware blockchain core natively supports pluggable state machine replication (SMR) atop a Byzantine fault-tolerant (BFT) consensus protocol. It enables blockchain nodes (defined as "members" in the VMware documentation), who may not necessarily trust one another, to share data or transact in a permissioned business network. Further, the blockchain

2 Attached in the supplemental section is the set of API endpoints we created to interact with the test environment. These endpoints allow the user to fetch information about the supercomputer jobs and also create new users for the system and new jobs for the supercomputer.

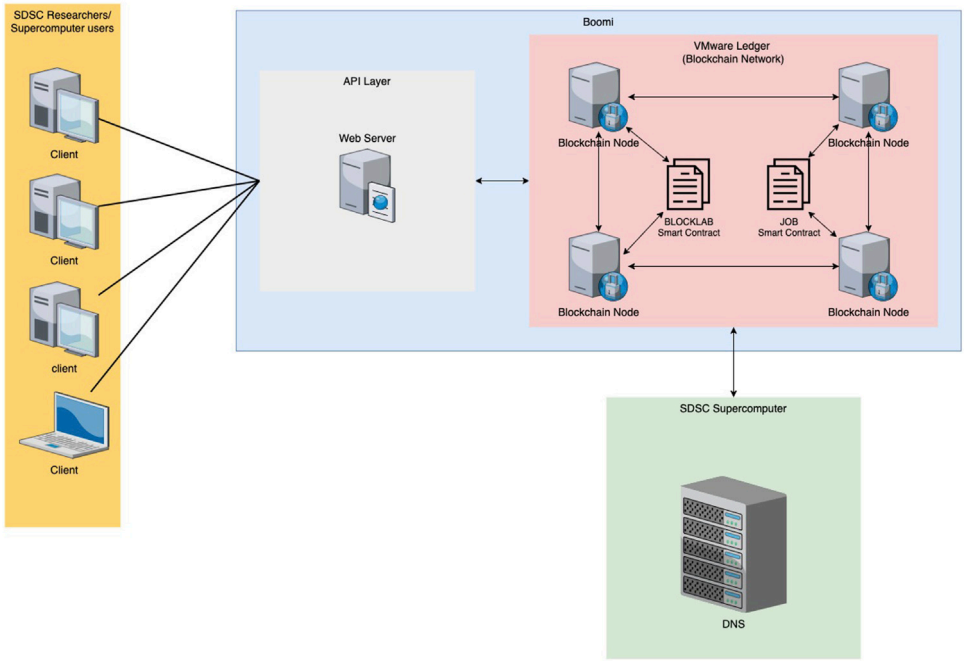

FIGURE 4
System under test.

architecture is designed to enable the network to operate correctly even if a subset of nodes behaves maliciously. The system also allows for an increase in the number of nodes in the network, improving network availability and tolerance to faults.

At a high level, the test environment created using the VMware blockchain platform is a distributed trust platform. By "distributed trust," we refer to a programming language-based control system that can verify "who is asking for what," using a comparison approach that matches up the user or the role the user adopts with policies that distribute access rights and authorizations to that user or role. Trust platforms have become essential components of blockchain networks, where potentially large numbers of people are spread geographically, making multiple requests for information, sometimes for the first time. In this scenario, conventional system-security and verification approaches are inadequate (Boomi, 2022).

Smart contracts: creating "blocklab" and "job" smart contracts

"Smart Contracts" are self-executing digital exchange agreements that reside and run on a distributed ledger/blockchain service. The "smart" aspect of a smart contract refers to the autonomous execution of software code. Each smart contract exposes a set of publicly accessible functions that are called by applications (APIs). The 'contract' aspect of smart contracts refers to the execution of the transaction once all pre-defined rules are met (Levi and Lipton, 2018; Cheng et al., 2019).

The BlockLAB and VMware development teams wrote two smart contracts to enable analytic job information to be transmitted and stored in the blockchain ledger. The first, called "BLOCKLAB," kept track of all analytic jobs submitted, and the second, called "JOB," represented each job submitted, its job parameters, and the result. As summarized in Figure 5, instantiating the smart contracts, establishing distributed data streams for the blockchain, and verifying the data and blockchain network connectivity involved a series of design considerations and programming steps[3].

The final steps in building and testing the blockchain network implement the following sequence of steps: the program sequence starts when an analytic job is submitted to the supercomputer, triggering a transaction containing the name of the analytic job, the requester of the job, and the submission

3 Historically, smart contracts automated the execution of a set of pre-defined rules between a set of agreeing nodes. Trigger events invoke the execution of these agreements via methods codified in the smart contract (thus enforcing the rules). In short, through conditional logic, the smart contract verifies and enforces the execution of a digital transaction between two or more nodes. As smart contracts run on top of a distributed ledger platform, all interactions are immutably recorded. Ledger entries generate an auditable record for all parties involved in the transaction.

1. Smart Contract Instantiation
   a) Connecting to the Boomi blockchain
   b) Instantiating the BLOCKLAB and JOB smart contracts
2. Interacting with Smart Contracts
   a) Creating new contracts
   b) Administering the contracts
   c) Querying the contracts
3. Distributed data method for recording on the blockchain
   a) Storing supercomputer analytic jobs data
4. Processing of data for the blockchain
   a) Ensuring the correct formatting
   b) Determining policies for how much sensitive data can be added
5. Establishing distributed data streams for the blockchain
   a) Transmitting analytic job data from the supercomputer to the blockchain
6. Verification of data and connectivity
   a) Verifying that all submitted analytic jobs and job parameters have been successfully transmitted
   b) Testing the network connection between the supercomputer and the blockchain

FIGURE 5
Design consideration in building the Blockchain network.

date to be run against the BLOCKLAB contract. The BLOCKLAB contract then instantiated a new JOB contract, corresponding to the submitted job. Once instantiated, JOB was updated with parameters including the job input data, the analytic model and resources requested. Once the supercomputer job was completed, the JOB contract was updated with the analytic job output parameters. Both BLOCKLAB and JOB contracts could be accessed through a set of published APIs and, in principle, could be integrated into the larger analytics job workflow. As each contract ran on the distributed ledger, all interactions in the test environment were recorded immutably on the DTL (distributed transaction ledger) and made available for verifiability and reporting.

Experimental results and assessment

Over a several-month testing period, the BlockLAB development team tested and verified system parameters by submitting over 500 API calls to the blockchain network. The API calls simulated a wide range of analytic jobs and system resources required by researchers at SDSC using the COMET supercomputer. The blockchain system worked as designed: the analytic jobs were successfully transmitted from the SDSC servers to the VMware Boomi web service using smart contracts and were recorded to the VMware blockchain distributed ledger for subsequent analysis and reporting. Excerpts of the code used are presented in the Supplementary Material.

In partnering with commercially developed technologies from Boomi and VMware, SDSC and BlockLAB researchers avoided a number of complexities. Researchers were not constrained to specific blockchain implementations, allowing the team to future-proof their application logic (e.g., no code changes for different blockchain implementations). System features built into the Boomi and the VMware blockchain platform eliminated the need for the SDSC development team to interact with the ledger directly. By this, the team was able to focus resources on the use case and test environment, job classifications, SUT system architecture and test parameters, and the successful posting of data to the blockchain and distributed ledger environment.

In reviewing the results, the two development teams reconsidered three important project goals. First, the primary design goal was to build and successfully test a workable FL/FA/blockchain system in an HPC environment. A second key objective was to create and test a workable and efficient smart contract implementation in the test environment. Third, a key objective was to open up discussion of future use cases and operational scenarios presented by the pilot. We will comment on this last objective in our concluding section.

Key takeaways from our pilot system design and testing include:

1 Blockchain applied to Federated Learning/Federated Analytics can improve model and data consistency through smart contract automation, and trust through the transparency and public nature of the blockchain.
2 Tracking and securing the executed code to train models can ensure consistency throughout the FL/FA lifecycle because each successive run of analytic jobs on the training datasets in the distributed data architecture is "known" to be consistent, and therefore "trusted". The distributed ledger and distributive nodes' reaching consensus ensures consistency is improved, which is a contributing factor in "trust".
3 Data can be verified/proven consistent in the distributed storage model, and/or zero-knowledge proofs can be employed to ensure data integrity used in the FL/FA architecture without exposing the underlying data[4].
4 Provenance in the FL/FA architecture appears to be a key blockchain use case, in that concerns around implementing blockchain in FL/FA can be mitigated through a variety of traditional security techniques that anonymize users in the system.

Conclusion and future research

Motivated by the growing interest in the integration of federated learning and blockchain technologies, this paper presents a system design, test implementation, and example code to provide its interested researchers exploratory tools for study and experimentation in FL/FA blockchain. The integration of FL/FA and blockchain technologies provide their own unique

4 In cryptography, a zero-knowledge proof or zero-knowledge protocol is a method by which one party (the prover) can prove to another party (the verifier) that a given statement is true while the prover avoids conveying any additional information apart from the fact that the statement is indeed true.

challenges in the design and testing of pilot systems. As the data engineering and smart contract design considerations, for example, do not have an obvious "federated version" that would be efficient or necessarily comprehensive under the assumptions motivating this work. However, as major technology firms have deployed federated learning in production, and several startups have been founded with the objective of using federated learning to address privacy and data collection challenges, these activities suggest that FL/FA blockchain will continue to gain traction in a range of industry use cases and in interdisciplinary academic research (Li et al., 2021; Salim et al., 2021; Unal et al., 2021).

In defining our future research objectives, the authors plan to extend this work into topics including federated optimization (Konecny et al., 2016), FL algorithmic performance benchmarking (Nilsson, A. et al., 2018), and blockchain performance evaluation (Egbedion et al., 2021). Here again, the challenge will be to design performance criteria, appropriate SUTs, and assessment methods that will integrate across FL optimization problems and blockchain performance. Optimization and performance research in FL has tended to focus on the efficiency of different algorithms in resolving communications inefficiencies (time, energy) in the distributed data model. With this emphasis, researchers are often concerned with sparse data, where inefficiencies may occur on a small subset of nodes or data points.

Researchers addressing blockchain performance have tended to cluster performance evaluation into four groups, benchmarking, monitoring, experimental analysis, and simulations. An early benchmarking example is BlockBench, a framework designed for evaluating private blockchains in terms of performance metrics on throughput, latency, scalability, and fault-tolerance (Tuan Anh Dinh et al., 2017). More generally, performance metrics are typically divided into two groups: macro and micro. Macro metrics provide an overview of the system's performance for users at the application level, such as, for example, transaction throughput, latency, scalability, or fault tolerance. Micro metrics track the performance of different sub-processes of transactions or specific layers in the blockchain developer model, including peer discovery rate, transaction propagating rate, contract execution time, or encryption and hash function efficiency (Yu et al., 2017; Dickerson, et al., 2020; Fan et al., 2020). As a general rule, both macro and micro metrics are evaluated under well-specified workloads.

We intend to pursue this work to address two use cases that are receiving widespread attention in both academic and industry publications. In the IoT space, smart home energy management and in the financial domain, credit score monitoring and reporting. Both use cases present design challenges that draw on the pilot work presented in this paper and extend into the design and analysis of their operational performance. Use case scenarios are important for requirements specification and the description of high-level functionalities. Specifically, we will be interested in identifying patterns of requirements that are common across different test system implementations. We envision multiple challenges in the design, implementation, and assessment of pilot field tests in our work, and in the overall goal of assessing how such systems are implemented in practice.

Data availability statement

The original contributions presented in the study are included in the article/Supplementary Material; further inquiries can be directed to the corresponding author.

Author contributions

JS: Conceptualization, project lead, experimental design, writing—original draft preparation. KM: Conceptualization, test architecture, data engineering, and programming lead, writing—review and editing. CT: Conceptualization, data engineering and programming, writing—review and editing. ST: Conceptualization, Dell-VMware-Boomi integration, writing—review and editing.

Funding

This research was provided by block grants made to the BlockLAB, the San Diego Supercomputer Center, UC San Diego.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors, and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

Supplementary material

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fbloc.2022.893747/full#supplementary-material

References

Awan, S., Li, F., Luo, B., and Liu, M. (2019). "Poster: A reliable and accountable privacy-preserving federated learning framework using the blockchain," in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 2561–2563.

Bonawitz, K., and Eichner, H. (2019). Towards federated learning at scale: System design," in *Proceedings of the 2nd SysML Conference* Palo Alto, CA., USA.

Boomi (2022). AtomSphere platform. Available at: https://boomi.com/platform/ (Accessed March 8, 2022).

Cheng, R., Zhang, F., Kos, J., He, W., Hynes, N., Johnson, N., et al. (2019). "Ekiden: A platform for confidentiality-preserving, trustworthy, and performant smart contracts," in *2019 IEEE European symposium on security and privacy (EuroS&P)*, 185–200. doi:10.1109/EuroSP.2019.00023

Dickerson, T., Gazzillo, P., Herlihy, M., and Koskinen, E. K. (2020). Adding concurrency to smart contracts. *Distrib. Comput.* 33, 209–225. doi:10.1007/s00446-019-00357-z

Egbedion, B., Wimmer, H., and Rebman, C. M., Jr. (2021). Exploring blockchain performance with CPUHEAVY microbenchmark on smart contracts. *Issues Inf. Syst.* 22 (4), 305–319. doi:10.48009/4_iis_2021_330-345

Fan, C., Ghaemi, S., Khazaei, H., and Musilek, P. (2020). Performance evaluation of blockchain systems: A systematic survey. *IEEE Access* 8, 126927–126950. doi:10.1109/ACCESS.2020.3006078

Gueta, G. G., Abraham, I., Grossman, S., Malkhi, D., Pinkas, B., Reiter, M. K., et al. (2019). *Sbft: a scalable and decentralized trust infrastructure*. Cornell University. arXivarXiv:1804.01626v3 (Accessed Jan 2, 2019).

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoj, A. N., et al. (2019). *Advances and open problems in federated learning. Papers compiled from the Workshop on federated learning and analytics*. Seattle WA. June 17–18th, 2019.

Kitchenham, B., and Charters, S. (2007). *Guidelines for performing systematic literature reviews in software engineering*. Durham: Keele University and Durham University Joint Report. Technical Report EBSE 2007-001.

Konecny, J., McMahan, H. B., Ramage, D., and Richtarik, P. (2016). Federated optimization: Distributed machine learning for on-device intelligence. Available at: https://arxiv.org/abs/1610.02527 (Accessed June 6, 2022).

Levi, S. D., and Lipton, A. B. (2018). *An introduction to smart contracts and their potential and inherent limitations*. Cambridge, MA: Harvard Law School Forum on Corporate Governance. Available at: https://corpgov.law.harvard.edu/2018/05/26/an-introduction-to-smart-contracts-and-their-potential-and-inherent-limitations/ (Accessed May 26, 2018).

Li, J., Shao, Y., Wei, K., Ding, M., Ma, C., Shi, L., et al. (2021). Blockchain assisted decentralized federated learning (BLADE-FL): Performance analysis and resource allocation. *IEEE Trans. Parallel Distrib. Syst.* 33, 2401–2415. doi:10.1109/TPDS.2021.3138848

Lo, S. K., Liu, Q., Wang, C., Pail, H-Y., and Zhu, L. (2020). A systematic literature review on federated machine learning: From A software engineering perspective. *ACM Comput. Surv.* 37 (4), 111:1–111:39. doi:10.1145/3450288

McMahan, B., and Ramage, D. (2017). Federated learning: Collaborative machine learning without centralized training data. Available at: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html (Accessed April 6, 2017).

Mugunthan, V., Rahman, R., and Kagal, L. (2020). "BlockFLow: An accountable and privacy-preserving solution for federated learning," in *MIT CSAIL*. Preprint. 2007.03856v1.pdf (arxiv.org) (Accessed March 8, 2022).

Oasis Protocol Foundation (2022). Oasis network primer. Available at: https://docs.oasis.dev/oasis-network-primer/(Accessed March 8, 2022).

Project Concord (2018). *Open source generic state machine replication library*. Available at: https://projectconcord.io/(Accessed March 8, 2022).

Salim, S., Turnbull, B., and Moustafa, N. (2021). A blockchain-enabled explainable federated learning for securing internet-of-things-based social media 3.0 networks. *IEEE Trans. Comput. Soc. Syst.*, 1–17. Early Access. doi:10.1109/TCSS.2021.3134463

Truex, S., Baracaldo, N., Anwar, A., Steinke, T., Ludwig, H., Zhang, R., et al. (2019). *A hybrid approach to privacy-preserving federated learning*. Cornell University. arXivarXiv:1812.03224v2 (Accessed Aug 14, 2019).

Tuan Anh Dinh, T., Wang, J., Chen, G., Liu, R., Beng, C. O., and Tan, K. L. (2017). *Blockbench: A framework for analyzing private blockchains*. Chicago, IL, USA. SIGMOD'17, May 14-19. doi:10.1145/3035918.3064033Accessed June 20, 2022)

Unal, D., Hammoudeh, M., Asif Khan, M., Abuarqoub, A., Epiphaniou, G., and Hamila, R. (2021). Integration of federated machine learning and blockchain for the provision of secure big data analytics for internet of things. *Comput. Secur.* 109, 102393. doi:10.1016/j.cose.2021.102393

VMware Blockchain Group (2022). Introducing VMware blockchain. Available at: https://octo.vmware.com/vmware-blockchain-launch/(Accessed March 8, 2022).

Yu, L., Tsai, W. T., Li, G., Yao, Y., Hu, C., and Deng, E. (2017). "Smart-contract execution with concurrent block building," in *Proc. IEEE symposium: Service-oriented syst. Eng. (SOSE)*, 160–167.