VFChain: Enabling Verifiable and Auditable Federated Learning via Blockchain Systems

Zhe Peng, Jianliang Xu, Xiaowen Chu, Shang Gao, Yuan Yao, Rong Gu, and Yuzhe Tang

***Abstract*—Advanced artificial intelligence techniques, such as federated learning, has been applied to broad areas, e.g., image classification, speech recognition, smart city, and healthcare. Despite intensive research on federated learning, existing schemes are vulnerable to attacks and can hardly meet the security requirements for real-world applications. The problem of designing a secure federated learning framework to ensure the correctness of training procedure has not been sufficiently studied and remains open. In this paper, we propose VFChain, a verifiable and auditable federated learning framework based on the blockchain system. First, to provide the verifiability, a committee is selected through the blockchain to collectively aggregate models and record verifiable proofs in the blockchain. Then, to provide the auditability, a novel authenticated data structure is proposed for blockchain to improve the search efficiency of verifiable proofs and support a secure rotation of committee. Finally, to further improve the search efficiency, an optimization scheme is proposed to support multiple-model learning tasks. We implement VFChain and conduct extensive experiments by utilizing the popular deep learning models over the public real-world dataset. The evaluation results demonstrate the effectiveness of our proposed VFChain system.**

***Index Terms*—Auditable training, blockchain, federated learning, model verification.**

I. Introduction

RECENTLY, advanced machine learning techniques, such as deep learning, have been widely applied in various areas, e.g., image reconstruction [1], speech recognition [2], smart city [3], [4], and disease diagnosis [5]–[7]. By leveraging these machine learning techniques, valuable knowledge could be derived from huge volumes of data in the wild. In order to achieve high accuracy for learning tasks, diverse datasets from many sources need to be collected and fed into machine learning models. In particular, due to the splintered landscape of data owners (or data silos) and privacy concerns, it is not easy to collect enough high-quality datasets. Thus, rather than concentrating the training data collection for models, federated learning [8], [9], a kind of distributed machine learning framework, distributes the training model to multiple data owners for individual training, while generating the ultimate model via intermediate gradients aggregation in an iterative fashion.

Although federated learning can be applied to collaboratively train a deep learning model over distributed datasets, most existing schemes are vulnerable to attacks and can hardly meet the security requirements for applications in reality. In the federated learning framework, central servers (e.g., cloud servers) are responsible for distributing, aggregating, and updating models that are eventually used in data analysis tasks. As such, these centralized servers constitute valuable targets for malicious adversaries, who might tamper with models to generate intended results or biased decisions [10]. In addition, driven by certain illegal interests, a mischievous cloud service provider might return incorrect results to users. For example, a "lazy" cloud service provider might replace the original model with a simpler but less accurate model to save its computation cost, or even worse, maliciously forge the aggregated model updates returned to users. Therefore, in order to improve the security of federated learning, it is crucial to verify the integrity and authenticity of model updates throughout the whole training procedure to avoid malicious attacks.

Recently, many works have been devoted to alleviating the data integrity problem for federated learning. These works can be classified into two types. The first type can verify the model update through a cryptographic proof generated by the central server. Xu *et al.* [11] leverage bilinear pairing to derive the integrity proof of aggregated results based on the participants' local encrypted gradients during the federated learning. Ghodsi *et al.* [12] represent neural networks as arithmetic circuits which perform computations over finite fields, and then develop an interactive proof protocol for execution verification of deep learning models on untrusted cloud. Tramer *et al.* [13] utilize trusted execution environment (i.e., the Intel SGX) to protect integrity and privacy for outsourced deep learning computations. In order to improve the efficiency of learning,

Manuscript received September 1, 2020; revised November 19, 2020; accepted December 26, 2020. Date of publication January 12, 2021; date of current version January 4, 2022. This work was supported by the Research Grants Council of Hong Kong under GRF Projects 12201520, 12200819, & 12201018, Guangdong Science and Technology Special Fund SDZX2019042 & SDZX2020036, Guangdong Basic and Applied Basic Research Foundation 2020A1515111070, and National Natural Science Foundation of China under Grant 61876151 & 62072230. Recommended for acceptance by Dr. Fan Wu. *(Corresponding author: Jianliang Xu.)*

Zhe Peng, Jianliang Xu, and Xiaowen Chu are with the Department of Computer Science, Hong Kong Baptist University, Hong Kong, Hong Kong (e-mail: pengzhe@comp.hkbu.edu.hk; xujl@comp.hkbu.edu.hk; chxw@comp.hkbu.edu.hk).

Shang Gao is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, Hong Kong (e-mail: shanggao@comp.polyu.edu.hk).

Yuan Yao is with the School of Computer Engineering, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: yaoyuan@nwpu.edu.cn).

Rong Gu is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210008, China (e-mail: gurong@nju.edu.cn).

Yuzhe Tang is with the Department of Electrical Engineering, and Computer Science, Syracuse University, New York, NY 13244 USA (e-mail: ytang100@syr.edu).

Digital Object Identifier 10.1109/TNSE.2021.3050781

the proposed scheme optimally distributes the computation to trusted and untrusted machines. However, since these schemes heavily rely on the engagement of the centralized servers, the single-point-of-failure problem still exists. Besides, previous methods either are applicable to a small variety of activation functions or require additional support of special hardware.

The other type is to leverage blockchain and smart contract technologies to conduct the verification operations for federated learning. Kim *et al.* [14] design BlockFL, a blockchain-based federated learning architecture, where mobile devices' local-learning model updates are exchanged and verified through a smart contract realized in blockchain. Bao *et al.* [15] propose FLChain to build a public auditable federated learning, in which trainers' information and verifiable training details are stored in the blockchain. An incentive mechanism is involved via smart contract to motivate participating nodes to be honest and report the misbehavior. Kang *et al.* [16] develop a reputation-aware worker selection scheme based on blockchain to improve the reliability and correctness of federated learning tasks in mobile networks. The developed approach can achieve efficient reputation management of the workers without repudiation and tampering. However, these existing schemes still have some disadvantages. Most of them just utilize the smart contract in the public blockchain, instead of the centralized servers, to conduct the model aggregation and distribution. The system performance is very limited and the data privacy of models cannot be preserved. The operation logs recorded in the blockchain are not organized well, which cannot support efficient queries for audit. Moreover, these schemes fail to consider the dynamics of participants (e.g., nodes dropping out and joining) while conducting the aggregation. As a consequence, this motivates our research problem would be: *Is it feasible to design a secure federated learning framework to ensure the integrity and authenticity of the training procedure?*

In this paper, we provide an affirmative answer through the systematic design and implementation of a verifiable and auditable federated learning framework (i.e., *VFChain*). Different from current schemes, VFChain utilizes the state-of-the-art blockchain and cryptography techniques to efficiently verify and audit the correctness of training process for the federated learning. First, in order to provide the verifiability, a verifier committee is selected through the blockchain to collectively conduct the model aggregation operations instead of the central servers and record corresponding verification proofs in the blockchain. Benefitted from the distribution property of blockchain, the robustness of the system can be enhanced and the recorded verification proofs are prevented from being tampered with by any node. Second, in order to provide the auditability, a novel authenticated data structure is proposed for blockchain to improve the search efficiency of verification proofs and support a secure rotation of committee. Finally, for the purpose of improving the efficiency, an optimization scheme is proposed to support verification for multiple-model learning tasks. To the best of our knowledge, this work represents the first attempt to cope with verifiability and auditability to build a secure federated learning framework based on the blockchain system.

Implementing such a functional system entails distinct challenges. First, the integrity and authenticity of model updates in federated learning traditionally depends on the computation by a central server, which is vulnerable to attacks. Proposing an effective scheme for defending against the problems brought by single point has a top priority to guarantee the correctness of the learning process. The second one is how to design an authenticated data structure to improve the search efficiency of verification proofs for audit. Third, it is not trivial to propose an optimization scheme to save the verification cost for multiple-model learning tasks. Repeatedly creating a freshness service for each model is both bandwidth-intensive and time-consuming.

To address the aforementioned challenges, we make the following contributions in this work:

- We propose the *VFChain* system that can provide the verifiability and auditability for the training procedure based on the blockchain to improve the security of federated learning.
- We present an effective *committee selection scheme* based on the blockchain to collectively aggregate models and record verifiable proofs in the blockchain, which can provide the verifiability.
- We propose a novel *authenticated data structure* for blockchain to improve the search efficiency of verifiable proofs and support a secure rotation of committee, which can provide the auditability.
- We design a specific *optimization scheme* to support multiple-model learning tasks. Specially, in order to improve the performance of model verification and audit, an integrative audit layer is designed to aggregate independent audit procedures.
- In addition, we evaluate our system by utilizing the popular deep learning model and the public real-world dataset. The experiment results show that VFChain can effectively provide verifiability and auditability for federated learning.

The remainder of this paper is organized as follows. We first introduce some related techniques in Section II. Then, an overview of VFChain system is described in Section III. The details of the system design and evaluation are presented in Section IV and V, respectively. In Section VI, we describe the related work. Finally, conclusion can be found in Section VII.

II. Background and Preliminaries

In this section, we briefly review some techniques behind our system, and clarify their necessity in our design.

A. Federated Learning

Federated learning [17], [18], also known as collaborative learning, can be viewed as the combination of machine learning and distributed computation essentially. Define N data owners $F_1, \ldots, F_N$, who want to collectively train a machine learning model by contributing their respective data $D_1, \ldots, D_N$. The traditional method is to collect all data together and utilize $D = D_1 \bigcup \ldots \bigcup D_N$ to train a model M_{Sum}. In contrast, a

federated-learning system is a learning process in which the data owners collaboratively train a model M_{Fed} and any training data D_i is restrained within its data owner F_i without exposure. In federated learning, a central server will maintain the global machine learning model M_{Fed} to be trained and coordinate multiple participating nodes (i.e., data owners) during the distributed training process. For each node, it will train a machine learning model (as the same one maintained by the central server) on its local data samples individually, and then upload intermediate parameters (i.e., gradients) to the central server. Upon receiving all gradients from these participants, the central server aggregates those parameters and updates the global model accordingly. After that, each node downloads the updated model from the central server and continues the training process on the same local dataset. This distributed training process will be iterated until the training error is smaller than a pre-specified threshold. In addition, the accuracy of M_{Fed}, denoted as V_{Fed}, should be very close to the accuracy V_{Sum} of M_{Sum}. Let δ be a non-negative real number. If we have

$$|V_{Fed} - V_{Sum}| < \delta, \tag{1}$$

then we say the federated learning algorithm has δ-accuracy loss.

B. Blockchain

Blockchain [19] is a distributed ledger that stores all transactions and states in the whole network. A blockchain includes a series of blocks which are linked one by one through recording a reference of the previous block. Concretely, a block mainly consists of a block header and a block body. The block header loads the metadata of the block, e.g., the timestamp and the hash value of the previous block. The block body contains a list of transactions recorded in this block [20]–[22]. Each full node in the network will maintain an integrated copy of the ledger. To achieve the data consistency [23], various consensus algorithms have been proposed, e.g., the Proof of Work (PoW) [24], the Proof of Stake (PoS) [25] and the PBFT [26].

Blockchain is firstly introduced for the consensus in Byzantine failures. The first implementation of the blockchain-based application is the Bitcoin system. By maintaining a distributed ledger (also known as the blockchain), Bitcoin system can provide a decentralized, open, Byzantine fault-tolerant transaction mechanism, and is promising to be the infrastructure for cryptocurrency network. Specifically, each block in a blockchain consists of two parts: header and records. Header contains the information of the block, including a Merkle root of all recorded transactions, a hash value of the previous block header, a nonce, etc. Records are the data (transactions in the Bitcoin system) stored in the blockchain. Blocks are chained together by headers using a cryptographic hash as a means of reference.

A blockchain network contains the following features:

Transparency: The network is accessible to all participants. Any participant can get the current state of the blockchain system based on the records in the blockchain.

Consensus: All peer nodes in the network will reach consensus on the blockchain (i.e., no intentional forks). A valid block discovered by an honest peer will be recorded on the blockchain and accepted by other peers.

Unmodified and Verifiable: All participants can verify the current state based on the records in the blockchain. Once a block is discovered and globally accepted, any further modification on this block is impossible.

C. Chameleon Hash Function

Chameleon hash function (CHF) [27] is a special kind of cryptographic hash functions that can efficiently generate the hash collision if having a trapdoor key. Given a message m and a parameter r with random value. A trapdoor key tk holder can easily find two pairs (m, r) and $(m,' r')$ with the chameleon hash function such that $\mathcal{CH}(m, r) = \mathcal{CH}(m,' r')$, where $\mathcal{CH}(\cdot)$ is the chameleon hash function. To be noticed, message m and message m' should be different, i.e., $m \neq m'$. A standard chameleon hash function (CHF) includes several efficient algorithms specified as follows:

- $(hk, tk) \leftarrow HGen(1^\kappa)$: $HGen$ is a probabilistic key generation algorithm. The input is a security parameter $\kappa \in \mathbb{N}$, while the outputs include a public hash key hk and a secret trapdoor key tk.
- $(h, \xi) \leftarrow Hash(hk, m)$: $Hash$ is a probabilistic hashing algorithm. The inputs are a public hash key hk and a message $m \in \mathbb{M}$, while the output is a pair (h, ξ) which includes a hash value h and a check string ξ.
- $d = HVer(hk, m, (h, \xi))$: $HVer$ is a deterministic verification algorithm. The inputs include a public hash key hk, a message m, a candidate hash value h, and a check string ξ. The output is a check bit d. If (h, ξ) is a valid (hash, check string) pair for the message m, then d equals 1; otherwise d equals 0.
- $\xi' \leftarrow HCol(tk, (h, m, \xi), m')$: $HCol$ is a probabilistic collision finding algorithm. The inputs include a trapdoor key tk, a valid tuple (h, m, ξ), and a new message $m' \in \mathbb{M}$. The output is a new check string ξ' such that $HVer(hk, m, (h, \xi)) = HVer(hk, m,' (h, \xi'))$.

III. System Overview

In this section, we first introduce high-level security goals that a secure federated learning system should achieve. Then, we present the system model and the threat model of our proposed system.

A. Security Goals

To address the challenges described in Section I, we propose the security goals of VFChain as follows:

- No single point of failure: A federated learning system should guarantee the security in case any single component in the system fails to return results or gets compromised by attackers.
- Local-to-global affirmation: A federated learning system should provide a strengthened guarantee to its users

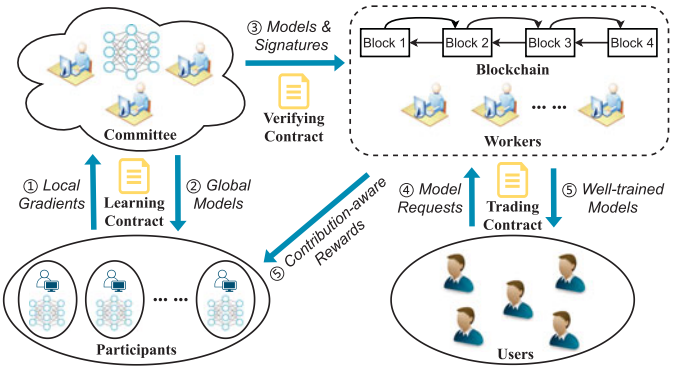


Fig. 1. System architecture of VFChain.

that the obtained well-trained models have been correctly aggregated from multiple local training models without tampering.

- Efficient model verifiability and auditability: A federated learning system should enable its users to efficiently verify the correctness of a model update and easily find the verifiable information of update with a specific version.
- Immutable model update history: A federated learning system should present a globally consistent history of global model update without tampering. In this log, each global model update corresponds to a unique log entry which cannot be deleted or amended once created.
- Evolution of authoritative keys: A federated learning system should provide the update function for authoritative keys, even when some but not the majority of the keys have been compromised.

B. System Model

Figure 1 presents the system architecture of building a verifiable and auditable federated learning framework based on blockchain. The system includes two main processes, i.e., verifiable federated learning empowered by blockchain and efficient training procedure audit. Specially, our system architecture mainly consists of the following actors: participants, workers, a blockchain-based committee, and users.

Participant: In VFChain, a *participant* is an entity defined as the same in the traditional federated learning model. Participants contribute their local training data and collaborate with each other to train a global machine learning model in return for monetary rewards.

Worker: Similar with the miners in common blockchain systems, *workers* process transactions which include related signatures of the updated model for verification and audit. Workers compete with each other to create a valid block, and the first worker completing the mining job can obtain some mining rewards. With these mining rewards, the worker could buy a well-trained model for his use through the VFChain.

Committee: A *committee* consists of a group of trustees, which are workers in the blockchain essentially. In practice, the same node in blockchain could run both the trustee and worker on a physical machine. Here we logically divide the two roles to clearly describe the designed architecture. Trustees in committee conduct, in addition to their mining tasks, the global model aggregation and verification based on the gradients collected from participants. Specifically, in order to improve the security of system, trustees are randomly chosen from workers and continuously updated based on the blockchain system.

Local Training: Each participant trains the local model with his private data independently. During each iteration, all participants generate and submit the local gradients with their signatures to the committee by launching transactions to a smart contract (called *learning contract*) in VFChain. The smart contract can be downloaded to process by workers in the committee.

Collaborative Training: Participants, who join in the same learning task, train a global machine learning model collaboratively. Specifically, after deciding a unique training model and initializing parameters, the global model is trained iteratively. In each iteration, based on the local gradients submitted by participants, trustees in the committee generate an updated global model and send it back to all participants. Meanwhile, in order to provide the verifiability and auditability for federated learning, trustees efficiently organize all generated models with verifiable signatures and send them to another smart contract (called *verifying contract*) in VFChain. Participants utilize the updated parameters in the received global model to update their local models. Then, a new iteration of model training will be triggered.

Iteration: The training procedure of a machine learning model includes multiple iterative stages (i.e., *iterations*). At the end of each iteration, all parameters in the global model will be updated once by the participant.

Round: In VFChain, the *round* refers to the procedure of creating a new block in the blockchain. The block includes the verifiable information of updated global model generated in each iteration. Specifically, in order to improve the efficiency of audit and support a secure rotation of committee, a novel authenticated data structure is proposed for blockchain.

VFcoin: VFcoin, denoted as VFC, is a kind of digital asset on VFChain. In particular, a certain amount of $VFCs$ will be created by VFChain as rewards when a new block is created by workers. In VFChain, participants are rewarded with $VFCs$ for their contributions of local model training, and trustees in committee can earn $VFCs$ for conducting the global model aggregation and verification. Workers can also gain $VFCs$ for successfully creating a new block. Meanwhile, a well-trained machine learning model will cost $VFCs$ for users who can hardly train the model by themselves but hope to utilize the model. This design is reasonable since existing studies on the model-based pricing for the machine learning have found many real applications in different scenarios [28], [29].

User: Users are clients of the VFChain system, who can publish various learning tasks and request the well-trained models for real applications. Specifically, our blockchain empowered federated learning system flexibly supports two kinds of model requests: (i) long-term subscription in which a user can continuously obtain the up-to-date model for a specific learning task;

and (ii) occasional request to obtain a model with a specified version pertaining to user's interests. Benefitted from the verifiability and auditability of VFChain, we can securely provide the model-as-a-service for various users.

C. Threat Model

We assume that the participants are semi-honest adversaries and will not collude with each other, following many previous works studying the federated learning framework (e.g., [30]–[32]). In particular, each participant will follow the designed specification honestly, yet is interested in inferring other participants' training data or meaningful information about a label that he does not own. For the trustees joining in the committee selected from the blockchain, we assume that most of them will correctly perform the duty, and part of them may be corrupted by an adversary and thus generate incorrect computation results. Here, we follow the standard threat model in many blockchain systems [33], [34]. That means, we assume that, among all trustees, at least ζ of trustees are honest and could send correct computational results to others at any moment, where ζ is a constant and larger than 2/3. Moreover, without loss of generality, we also assume each trustee has equivalent computation resource [33], (namely, each node will own one unit of computation power). We assume that all users in VFChain could securely conduct the bootstrap. That means, they can securely obtain the first version of the global model and an initial public key for the system. This initial public key can be hard-coded through many secure manners, such as pre-installed in a specific hardware or a read-only device, or through a secure network connection. Finally, in model request stage, users could attempt to get the well-trained machine learning model pertaining to their interests, while are not willing to make a payment to reward the participants and trustees.

IV. System Design

In this section, we present the VFChain, a verifiable and auditable federated learning framework based on the blockchain. We begin by introducing a decentralized verification scheme for federated learning, while alleviating the participant and trustee overhead. We then improve the efficiency of training procedure audit and address the evolution of committee configurations. Finally, we reduce the traversal overhead with an optimal scheme to support verification for multiple-model learning tasks.

A. Verifiable Federated Learning

In this subsection, we introduce the decentralized verification scheme for federated learning based on the blockchain system.

1) Local Gradients Collection: In traditional federated learning framework, after collecting all gradients from participants, central server aggregates parameters to update the global model. However, this framework has two disadvantages. First, it fails to consider the dynamics of participants (e.g., nodes dropping out and joining in) because of the possible network disconnection or new connection. Second, the correctness of local gradients cannot be ensured before conducting the model aggregation.

The first step towards VFChain involves extending the trust base of local gradient submissions. Instead of using a single and shared key to sign gradients, each participant uses his individual key to make a signature for his gradient submission. In order to alleviate the dynamics of participants, for each learning task, a threshold value θ is pre-defined to specify the minimal number of valid participant signatures required to generate a global model update. At the beginning of a learning task, the participants collect all their public keys in a *policy* file, together with a threshold value θ that specifies the minimal number of valid participant signatures required to generate a global model update. Complying with our threat model, we assume that this policy file, as a trust anchor, can be obtained securely by trustees at the initial acquisition of local gradients. For example, the policy file can reside on a learning task announcement website.

In each iteration, participants send local gradients and signatures to the committee through launching transactions to a smart contract (i.e., the *learning contract*) in VFChain. When the committee verifies that at least θ participants' signatures are valid, the model aggregation is triggered. An attacker trying to forge a valid global model update needs to control θ participants' keys, which is presumably harder than gaining control over any single signing key.

2) Committee Setup and Update: In order to alleviate the single-point problem introduced by the central server in federated learning, a trustful committee is initially established and continuously updated based on the blockchain. In VFChain, the committee will collaboratively conduct the model aggregation and verification in stead of the central server.

The trust of the committee comes from the block creation in the blockchain. Workers (or winners) who successfully generate blocks in different rounds are qualified to join in the committee. In the committee setup stage, we suppose the first ϕ winners in the blockchain system form the initial committee. Then, we define a *trust value* Ω for each trustee to measure his reliability, which will be described later. When a new winner is selected in the blockchain, if his trust value Ω is larger than the minimal one in the committee, this worker becomes an eligible trustee and join in the committee. Meanwhile, the trustee with the minimal trust value will be removed form the committee.

The trust value Ω of a trustee is defined as : $\Omega = t \times c$, where t is the latest timestamp of becoming a winner (this timestamp could be obtained from the corresponding block) and c represents the amount of $VFcoins$ owned by this trustee. In this definition, we consider two properties for a reliable trustee, i.e., *liveness* and *loyalty*. Liveness means that the selected trustees can continuously perform activities in VFChain, hence keeping the committee alive. Loyalty says that trustees with more $VFcoins$ have made more contributions for the whole system, hence keeping the committee stable.

In order to derive a confederate public key and share the secret key between many trustees in the committee securely, we utilize the existing distributed key generation (DKG)

scheme. In VFChain, we choose a DKG scheme [35] for real usage, which is not conflicting with the threat model for the trustee. By using the DKG scheme, the committee can derive a confederate public key pk_C securely and each trustee $\mathcal{T}_p$ can be allocated a sub-secret s_p of the confederate secret key sk_C. In addition, the private key sk_C could be re-derived through fusing t individual sub-secrets, where t is a threshold not less than $f+1$ and f represents the amount of compromised trustees [36]. Next, pk_C is derived by the committee off-chain, thus we have to take the consensus into account, which means the confederate public key should be consistent between different trustees with the existing of compromised trustees. A feasible solution is to use the majority voting scheme to achieve the consensus [37] with a trustee selection scheme. In this scheme, we assume at least ζ of the trustees are honest [33], where the value of ζ could be 2/3. Therefore, a correct public key pk_C is collectively recorded in the blockchain by the committee. This public key could be utilized for the subsequent verification procedures.

Specially, the update of the committee will trigger the regeneration of authoritative keys. Thus, in order to improve the efficiency and security, the committee is updated at a frequency μ which can be adjusted in the real practice. That means, after every μ iterations of training, the authorised keys of the committee will be regenerated, even though no new trustees join in the committee.

3) Verifiable Model Aggregation: In each iteration of federated learning, based on the local gradients collected from participants, trustees in the committee collectively generate a global model update and send it back to all participants. Meanwhile, the related verifiable information of models will be recorded in the blockchain for audit in the future.

In VFChain, the *learning contract* is responsible for global model aggregation. To generate a global model update, each trustee checks the received participants' signatures against their public keys and the threshold θ defined in the policy file. Suppose in the i-th iteration, the committee receives n valid gradients $\mathbf{W}_j^i$ from participants $P_j, j \in \{1, \ldots, n\}$, where n is larger then the threshold θ. Each trustee aggregates these gradients by calculating

$$\mathbf{W}^i = \frac{1}{n}\sum_{j=1}^{n} \mathbf{W}_j^i, \tag{2}$$

where $\mathbf{W}^i$ is the i-th global model update. Then, the generated model update $\mathbf{W}^i$ will be verified in the committee by the rule of majority voting. Thus, the verified model updates enable users and participants to obtain the guarantee of local-to-global correspondence, due to the broad independent validation.

In addition, in order to provide the verifiability and auditability of training procedure, the committee leverages the *verifying contract* to derive and record verifiable information in the blockchain. For a global model update $\mathbf{W}^i$, the committee makes a corresponding signature $\mathcal{S}_{\mathbf{W}^i}$ with the collective keys [35]. Then, together with the valid signatures $\mathcal{S}_{\mathbf{W}_j^i}, j \in \{1, \ldots, n\}$ of local gradients, the generated global model signature will be recorded in the blockchain in each iteration. Specially, to improve the search efficiency for audit, a novel authenticated data structure is proposed to optimally store these verifiable information in the blockchain, which will be concretely illustrated in Section IV-B.

4) Model Request With Fairness: Once a model is well trained through the federated learning, a *trading contract* will label the global model in this iteration as available. Then users can send model requests to the *trading contract* to obtain the model of their interests.

In the model request, according to the rewarding policy, users need to make the payment with their $VFcoins$ to participants and trustees for their contributions in the model training procedure. The rewarding policy describes the committee's computation fee ($\mathcal{R}_c$) and the contribution-aware rewards ($\mathcal{R}_p$) for compensating participants' dedicated data and computation. Specially, our system can flexibly support two kinds of model requests, i.e., long-term subscription and occasional request. One key difference between them is that the subscriber needs to make a deposit in advance, while the occasional requester only makes the payment on demand. When the payment is settled, they both have to contact the committee to obtain the desired models.

Dealing With Long-Term Subscriptions. For some special learning tasks, the global model will be continuously trained and updated to improve the prediction accuracy, such as the Google keyboard project. Thus, in a long-term subscription, a user can consecutively obtain the up-to-date model for real applications. Once the verification information of global model update is recorded, the monetization with subscribers automatically starts. For each subscriber, the trading contract will first verify whether he has enough $VFcoin$ deposit, and then automatically deduct $VFcoins$ and record his authorization information in the blockchain for later model transmission. The deducted $VFcoins$ is transferred to participants and trustees. For simplicity, in each iteration, we consider that each trustee equally splits the reward $\mathcal{R}_c$, and each participant receives a weighted reward according to the size of his local dataset, i.e., $\frac{n_k}{\sum_k n_k} \cdot \mathcal{R}_p$, where n_k is the number of samples available locally.

Dealing With Occasional Requests. Alternatively, a user can issue an occasional request to the trading contract to specify a model pertaining to his interests. This model can be the newly updated one or a previously generated one. Once receiving an occasional request, the trading contract will automatically grant authorization to the user for model transmission after the payment is settled successfully. The rewards for trustees and participants will be calculated and distributed as in the case of subscriptions.

Monetary Penalty Scheme. Considering the fairness in incentive mechanism, in addition to the reward, we also design a monetary penalty scheme for participants and trustees. Specifically, a *time check* is appended to each function within smart contracts of VFChain. For each function, we pre-define a running time. When comes to the end timepoint of the function execution, the system will verify the results of the function. If the verification failed, it means (i) some participants

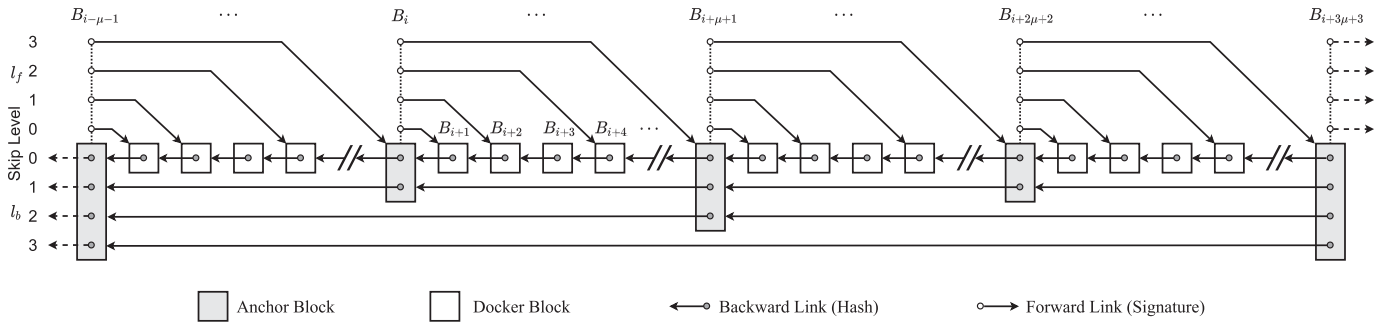


Fig. 2. Structure of a dual skip chain $\mathcal{C}_{2,3}^{2,3}$.

(or trustees) are not punctual by the end timepoint, or (ii) some participants (or trustees) generate incorrect computation results. For these two cases, VFChain will apply the monetary penalty mechanism by revoking the pre-frozen deposit of dishonest participants (or trustees) and distributing it to other honest participants (or trustees). As such, the penalty will never be exerted on honest participants (or trustees) who behave punctually and correctly, and compensation will be made for them with the existing of dishonest participants (or trustees). Therefore, VFChain can provide the fairness of incentive mechanism for participants and trustees in the model request stage.

B. Efficient Model Audit

In this subsection, we introduce the model audit scheme enabling clients to efficiently validate global model updates and signing keys. Specially, in order to improve the search efficiency, a novel authenticated data structure (i.e., dual skip chain) is proposed to optimally record verifiable information on the blockchain and support a secure rotation of committee.

1) Dual Skip Chain (DSC): Dual skip chain (DSC) is an authenticated data structure that builds hierarchical inter links between multiple related blocks in the blockchain. The DSC enables clients (including users and participants) (i) to securely traverse the blockchain in both forward and backward directions and (ii) to efficiently traverse arbitrary distances by employing multiple inter links.

The structure of DSC is illustrated in Figure 2. The DSC structure includes four major elements, (i.e., *anchor block* $\mathcal{B}_a$, *docker block* $\mathcal{B}_d$, *backward link* $\mathcal{L}_b$, and *forward link* $\mathcal{L}_f$). The anchor block $\mathcal{B}_a$ records an index of authorizations from the committee and the update of the committee, while the docker block $\mathcal{B}_d$ stores the verifiable information of model updates in each iteration of federated learning. The backward link $\mathcal{L}_b$ is the cryptographic hash of previous block, as in regular blockchain systems. The forward link $\mathcal{L}_f$ is a special cryptographic signature of future block based on the chameleon hash function [27], which is added in the current block aforehand. Specially, the link length is tied to both the block height h and the committee update frequency μ (defined in Section IV-A2), which is derived during the block creation. With the hierarchical inter links, the DSC enables a logarithmic-cost traversal over the blockchain, both backward and forward.

We denote a DSC by $\mathcal{C}_{g_b,l_b}^{g_f,l_f}$, where $g_b \geq 1$ and $l_b \geq 0$ denote *skip basis* and *skip level* for backward links, $g_f \geq 1$ and $l_f \geq 0$ are *skip basis* and *skip level* for forward links, respectively. A block $\mathcal{B}$ in DSC can be abstracted as $\mathcal{B} = (id, \pi_b, \pi_f, D, \bar{\mathcal{L}}_b, \bar{\mathcal{L}}_f)$, where id, π_b, π_f, D, $\bar{\mathcal{L}}_b$, and $\bar{\mathcal{L}}_f$ represent block identifier, the amount of backward links, the amount of forward links, payload data, the list of backward links, and the list of forward links, respectively. With different functionalities, blocks are classified into two types, i.e., anchor blocks and docker blocks.

Anchor Block $\mathcal{B}_a$. Anchor blocks are mainly used to record an index of authorizations from the committee and the updates of the committee. Concretely, an anchor block stores the periodically updated public key pk_C of the committee, hash values (i.e., backward links) of previous blocks, the signatures (i.e., forward links) of future blocks which are signed by the currently effective secret key sk_C of the committee, and a parameter r' for building the forward link. For the public key, recall that the system updates the committee and its authorised keys at the frequency μ (every μ iterations of training). Accordingly, as shown in Figure 2, the block $\mathcal{B}_i$ stores a committee's public key pk_C^i, while the block $\mathcal{B}_{i+\mu+1}$ stores a new public key $pk_C^{i+\mu+1}$ after μ iterations of training. For the signatures, the anchor block $\mathcal{B}_i$ selectively stores signatures of future blocks (for example, $\mathcal{B}_{i+1}, \mathcal{B}_{i+2}, \mathcal{B}_{i+4}, \ldots$), which are signed by the secret key sk_C^i stored in the block B_i. Specially, since these future blocks do not yet exist when the block $\mathcal{B}_i$ is created, a unique cryptographic signature of the future block will be generated in advance based on the chameleon hash function (CHF) [27]. For a future block $\mathcal{B}_j$, where $j > i$, the pre-generated signature $\mathcal{CS}_{\mathcal{B}_j}$ stored in $\mathcal{B}_i$ is derived as $\mathcal{CS}_{\mathcal{B}_j} = SIG(\mathcal{CH}(m_i, r_i), sk_C^i)$, where m_i and r_i are both randomly chosen parameters. If the future block $\mathcal{B}_j$ is also an anchor block (take the block $\mathcal{B}_{i+\mu+1}$ in Figure 2 as an example), an extra parameter r' needs to be calculated and stored to build the complete forward link from the anchor block $\mathcal{B}_i$. When the anchor block $\mathcal{B}_{i+\mu+1}$ is created, the parameter $r'_{i+\mu+1}$ will be calculated based on CHF and stored in anchor block $\mathcal{B}_{i+\mu+1}$, such that $\mathcal{CH}(m_i, r_i) = \mathcal{CH}(\mathcal{H}(\mathcal{B}_{i+\mu+1}), r'_{i+\mu+1})$, where $\mathcal{H}(\cdot)$ is a strong one-way hash function. Thus, benefitted from the consistency of hash value provided by CHF, the system can successfully build forward links for future blocks.

Docker Block $\mathcal{B}_d$. Docker blocks are mainly used to record the verifiable information of model updates. Concretely, a docker block loads the verifiable information of model update

in each iteration of federated learning, hash values (i.e., backward links) of previous blocks, and the parameter r' for building the forward link. As illustrated in Section IV-A3, the verifiable information of a global model update $\mathbf{W}^i$ generated in the i-th iteration, includes its signature $\mathcal{S}_{\mathbf{W}^i}$ made by the committee and the signatures $\mathcal{S}_{\mathbf{W}_j^i}, j \in \{1, \ldots, n\}$ of n corresponding local gradients. The hash values of previous blocks and the parameter r' for building the forward link are computed as in the case of anchor blocks.

Backward Link $\mathcal{L}_b$. A backward link is made through the cryptographic hash of previous block, as in regular blockchain systems. During the creation of a block, its identifier id is set as the cryptographic hash of D, $\bar{\mathcal{L}}_b$, and $\bar{\mathcal{L}}_f$, i.e., $id = \mathcal{H}(D, \bar{\mathcal{L}}_b, \bar{\mathcal{L}}_f)$, all known at this point. First, all blocks will be linked backward to generate a regular blockchain. Second, in order to improve the efficiency of backward traversal, we build extra backward links between anchor blocks. In an anchor block $\mathcal{B}_i$, $\bar{\mathcal{L}}_b$ stores exactly π_b backward links. Concretely, a link at skip level $1 \leq l \leq \pi_b - 1$ in the block $\mathcal{B}_i$ points to the last anchor block having at least $l + 1$ backward links. In DSC, this block is $\mathcal{B}_{i-j}$, where $j = (g_b)^{l-1} \cdot (\mu + 1)$. The value of π_b is determined as

$$\pi_b = max\{l : 2 \leq l \leq l_b + 1 \wedge 0 \equiv \left\lceil \frac{i}{\mu + 1} \right\rceil \ mod \ (g_b)^{l-2}\}. \tag{3}$$

Forward Link $\mathcal{L}_f$. A forward link is made through a pre-generated signature of future block based on CHF. In order to improve the efficiency of forward traversal, we build two types of forward links for both anchor blocks and docker blocks, i.e., *anchor-to-anchor link* and *anchor-to-docker link*. For the first type, in order to support a secure rotation of committee and evolution of authoritative keys, the forward link is added between each two adjacent anchor blocks. This link is set at the l_f-th skip level in each anchor block. For the second type, to easily track the authorization of model updates by a specific secret key, an exponentially number of forward links are built between an anchor block and related docker blocks. A link at skip level $0 \leq l \leq l_f - 1$ in the anchor block $\mathcal{B}_i$ points to the docker block $\mathcal{B}_{i+j}$, where $j = (g_f)^l$. Since we only connect related docker blocks for an anchor block, the maximum value of l_f can be set as $max\{l_f : (g_f)^{l_f - 1} \leq \mu\}$. As such, in each anchor block, the total amount of forward links is $\pi_f = l_f + 1$.

2) Properties of DSC: The DSC provides a hierarchical inter-link structure to build effective connections between correlative blocks in the blockchain. Beyond the standard properties of blockchain, the DSC offers four useful features as follows.

First, to support the secure rotation of committee in VFChain, the DSC enables effective evolution of authoritative keys. Complying with our threat model, we assume the initial public key of committee can be correctly recorded in the genesis block as a trust root. Then, the consistency of public key evolution is protected by the DSC. The execution of committee update is related with the successful creation of a block, which can be examined by all workers in the blockchain. To update the committee, trustees first collectively generate a new public key and sign it with the currently effective secret key, as usual during a new verifiable model aggregation. As such, the committee can delegate trust from the outdated key to the new one. When the workers have added the new key to the DSC, the new key becomes active and supersedes the older counterpart. Anyone following the DSC can make sure that a threshold of the trustees has approved the new authoritative key. With this scheme, the committee can rotate authoritative keys regularly and, securely revoke potentially compromised keys, if needed.

Second, the DSC enables clients to securely traverse the blockchain in both forward and backward directions from any reference point. In DSC, we build two types of inter links between various blocks for two traversal directions, respectively. On one hand, the backward links pay more attention to the authoritative key updates, which can be used to easily track the history of committee updates and conduct efficient backward audit. On the other hand, the forward links mainly focus on the training model updates, which can be used to efficiently track the verification information of global model generated with the same key in an iteration and conduct the forward audit.

Third, the DSC enables a logarithmic-cost traversal over the blockchain, both backward and forward. In DSC, we build the backward links with a rough granularity, while building the forward links with a fine granularity. Thus, the lengths of backward links and forward links have different distributions. Concretely, beyond the inherent backward links with the shortest length in the blockchain, most of the backward links between various anchor blocks have relatively longer lengths. By contrast, a majority of the forward links are built for the docker blocks located in two adjacent anchor blocks, which have relatively shorter lengths. Moreover, both types of inter links are built in an exponential manner. For example, a client has a correct hash of an existing block and hopes to obtain a future or past block in the DSC from an untrusted source (for example, a nearby peer node). To cryptographically verify the target block and all links pointing to it, the client needs to download only a logarithmic number of additional and intermediate blocks. Thus, with the DSC, clients can efficiently traverse arbitrary distances by employing multiple hierarchical inter links.

Moreover, the DSC enables two clients verify their owned blocks with each other. Suppose that two resource-constrained clients have two blocks in the DSC, but cannot access to any database containing a full counterpart of DSC. For example, workers exchange verification information of model updates through peer-to-peer while disconnected from any other worker in VFChain. If these workers have cached a logarithmic number of additional blocks with their respectively owned blocks and these blocks have intersections, then the two workers can successfully validate their target blocks with each other. For model updates, forward verification is important when an out-of-date worker obtains a newer update from a peer. The backward verification is also useful in the scenario of secure model version rollback.

C. Optimization for Multiple-Model Tasks

In this subsection, we illustrate the optimization scheme for learning tasks consisting of multiple training models.

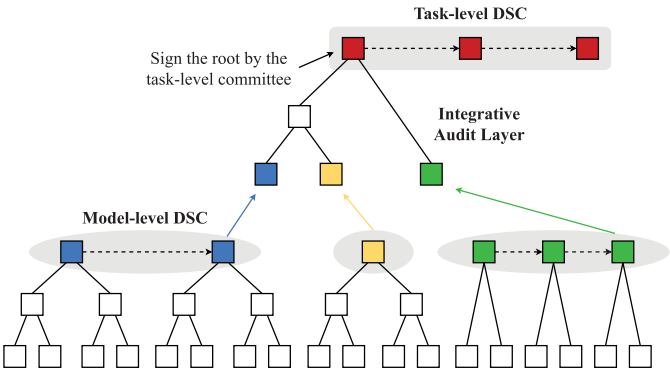


Fig. 3. Constructing an integrative audit layer in VFChain.

Specially, in order to improve the performance of model verification and audit, we design an integrative audit layer in the VFChain framework to aggregate independent audit procedures.

In many real business intelligence (BI) [38] applications, such as advertising recommendation [39], [40] and risk management [41], a data analysis task may need to simultaneously train multiple models to cooperatively generate the finally discriminative prediction results. In order to keep track of each model training, a naive approach is to construct a specialized DSC for each model to record the verifiable model updates, which is maintained by a separate group of workers. When a user conducts the separate verifications for new updated global models, he would have to frequently contact all the respective update logs and follow their configurations in the corresponding DSC. However, this method is bandwidth-intensive and time-consuming for the user, and requires the maintainers of each DSC to run a freshness service. Thus, to alleviate this burden, we further enhance and optimize the VFChain system to effectively support multiple-model learning tasks.

A new *integrative audit layer* for multiple models is designed and introduced into the VFChain framework. This layer is responsible for collecting and validating verification information about all the models included in the learning task, and finally providing the audit service to different users. As shown in Figure 3, a task-level committee is established to verify and aggregate multiple independent DSCs to build a task-level DSC (t-DSC). In the t-DSC, each entry is a snapshot of a task state. To derive a snapshot, the committee first requests the latest data from the individual model, including verifiable signatures and hashes of global model and local gradients. Then, the committee verifies the correctness of signatures against the corresponding model DSC. Finally, the committee builds a Merkle tree based on the hashes to summarize all model versions in the snapshot, and collectively signs the root.

This architecture can help realize the gradual upgrade of multiple-model training tasks based on some open-source and pre-trained models, such as VGG, ResNet, and DenseNet. Models, that do not yet have their own DSCs, can still be included in the integrative audit layer through their hash values with the latest version. Moreover, the task-level committee will run an aggregation timestamp service to prevent the replay attacks. The committee will be certain that users can receive the latest version of all individual models and a consistent task state. Then, users can request the latest signed task-level snapshot from the committee and check their outdated models using the Merkle proofs. If such models really exist, the user can access the individual DSC, knowing the hash values of the latest blocks.

In addition, a multiple-model learning task can have several different aggregation layers potentially. Each layer could represent a type of distribution, e.g., based on the development stage (release, test, and tuning) of models. The update history of individual model is maintained in the unique DSC, while the committee could additionally tag each version with its distribution affiliation for the learning task. For example, when a new release tag is added to a update of model, users can be notified with a new release distribution. Meanwhile, the participants could continue to train the model and publish a new (e.g., test) version of model in the DSC. Moreover, the timeliness can also be assured by maintaining a separate timestamp service for each distribution.

V. Implementation and Evaluation

In this section, we present the implementation and evaluation of our proposed VFChain system.

A. Experimental Methodology

We evaluate the performance of our blockchain-based verifiable and auditable federated learning framework via a proof-of-concept implementation. To implement the proposed prototype, we build a blockchain to simulate the VFChain. Blockchain nodes are regarded as trustees and workers. They participate in generating, auditing and trading the verifiable models for federated learning with three pre-defined smart contracts, i.e., *Learning Contract*, *Verifying Contract* and *Trading Contract*. Generated transactions are serialized in the blockchain.

First, we use Hyperledger Fabric (version 1.4) and Docker (version 18.09.0) to simulate the VFChain for its adaptability and simplification. The smart contract is implemented with the Go programming language. Specifically, Fabric is an open source project originally designed by IBM and now hosted by the Linux Foundation. It has been widely used in real business, such as finance, supply chain, healthcare, government, identity, etc. In Fabric network, peer nodes have three different roles, such as order node, endorser node, and anchor node. Beyond these three roles, we add a new role for peer nodes, i.e., the trustee, introduced in Section IV-A2. And each node is built in the docker environment to simulate a real node. Specially, in VFChain, the nodes in the committee serve as trustees, and other nodes serve as workers.

Second, we build the federated learning environment with PyTorch (version 1.6.0) and Python (version 3.6.9). In order to realize the interaction between the federated learning and the blockchain, we implement a specified application program interface (API) for Fabric. Through the API, participants involved in the federated learning can submit the generated local gradients to the committee and, then receive a updated

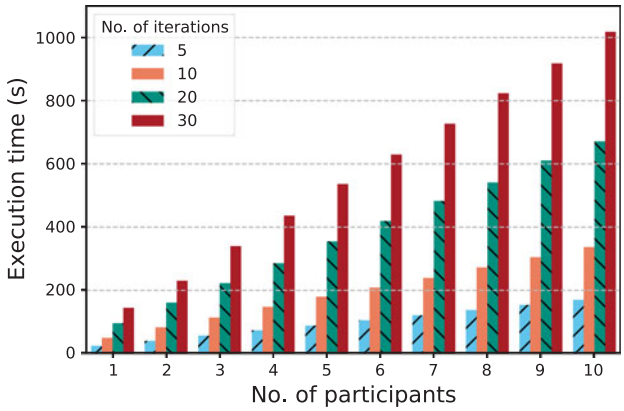


Fig. 4. Execution time cost w.r.t. no. of participants.

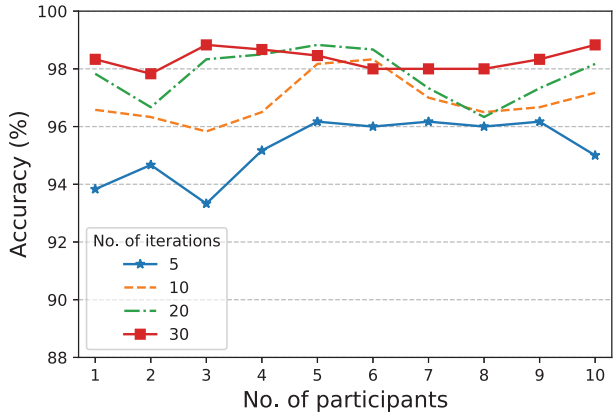


Fig. 5. Model accuracy w.r.t. no. of participants.

global model to continue the local training, thus successfully interact with the committee.

The experiments of evaluation are carried out on a desktop computer with a 2.0 GHz quad-core Intel Xeon processor and 32 GB of RAM, with Ubuntu 18.04 LTS operating system. In our experiments, we choose the popular MNIST database which includes a training set of 60000 examples and a test set of 10000 examples. We randomly extract 5000 samples from the training set as the verification examples. To comprehensively evaluate the system performance, we conduct multiple training experiments with 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10 participants, respectively. In a specific experiment with i participants, each participant will randomly extract $55000/i$ training samples and $5000/i$ verification samples, and then feed them into the training model. We use a deep learning model, i.e., Convolution Neural Network (CNN), as our training model with a structure as follows: input layer, convolution layer, maxpooling layer, fully connected layer, and output layer. For the training parameters, the learning rate is set as 0.01 and the batch size is set as 100. Moreover, experiments are also conducted with different numbers of iterations, including 5, 10, 20, and 30, respectively. In each iteration, a participant will train the local model with $55000/i$ training samples and $5000/i$ verification samples.

B. Performance Evaluation

We first evaluate the performance of verifiable federated learning through three different aspects, including execution time, accuracy, and throughput. Then, we further evaluate the performance of model audit scheme, and the effectiveness of optimization scheme for multiple-model tasks.

Evaluation on Execution Time. For the execution time, we measure the total execution time of VFChain, which mainly includes the time cost of training the model with multiple participants, and the time cost of generating and recording the model verification information into the blockchain. Experiments are conducted with different amounts of participants joining in the federated learning task. Moreover, in order to comprehensively evaluate the system performance, we also create four extra experimental scenarios with 5, 10, 20, and 30 iterations in the federated learning, respectively. Figure 4 plots the total execution time of the system with regard to the number of participants within four iteration scenarios. The system will cost $40.09\ s$ execution time, if we only have 2 participants and conduct 5 iterations of learning. And when 10 participants conduct 30 iterations in federated learning, the system needs $1018.02\ s$ to complete the whole training and recording procedure. From this figure we can find that, with more participants joining the federated learning task, the system needs more execution time. This is because if we have more participants, in each iteration, the system needs to aggregate more local gradients and subsequently record more verification information into the blockchain. In addition, more number of iteration will naturally cause much execution time cost.

Evaluation on Accuracy. In terms of the model accuracy, we measure the classification accuracy on the test dataset, with the finally generated global model after many iterations of federated training in VFChain. Experiments are conducted with different amounts of participants joining in the federated learning task. Moreover, as we did in the evaluation on execution time, we also create four extra experimental scenarios with 5, 10, 20, and 30 iterations in the federated learning, respectively. Figure 5 shows the classification accuracy of the generated global model with regard to the number of participants within four iteration scenarios. The accuracy of the model will achieve 98.83%, if we have 10 participants and conduct 30 iterations of learning. And when only 1 participant conducts 5 iterations in federated learning, the accuracy of the model will be reduced to 93.83%. Specially, when we have only 1 participant, the federated learning will be declined to a traditional model training manner. From this experiment we can find that, when we conduct more iterations of training, the model accuracy could be improved. This is because with more iterations, the training procedure can more sufficiently learn the features from the training dataset and thus generate a more discriminately model for classification. In addition, we find that the accuracy will be slightly improved with more participants engaged. This is because in each iteration, participants will randomly select training samples from the training set. If we choose a small number of participants and iterations, some examples in the training set may not be selected and fed into the model for training. More different samples for training will definitely generate a more accurate model.

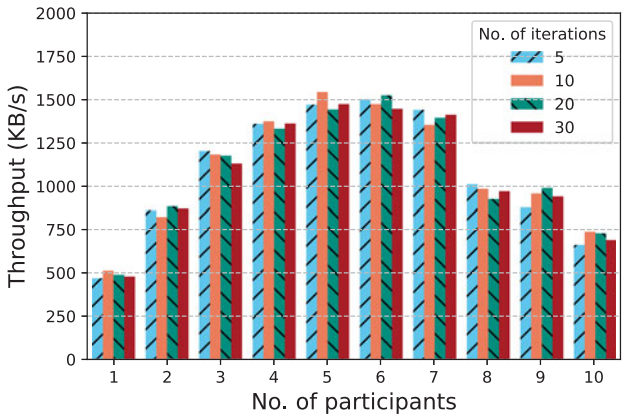


Fig. 6. System throughput w.r.t. no. of participants.

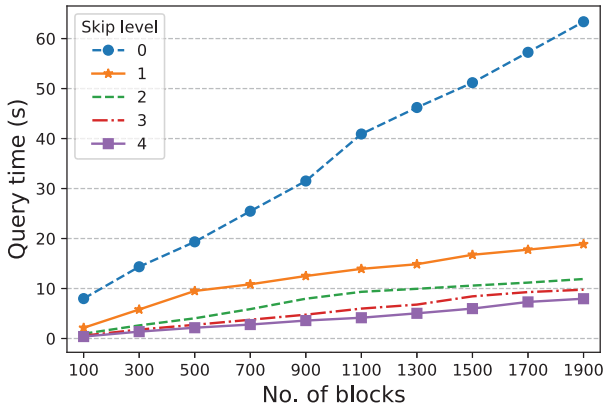


Fig. 7. Query time w.r.t. no. of blocks.

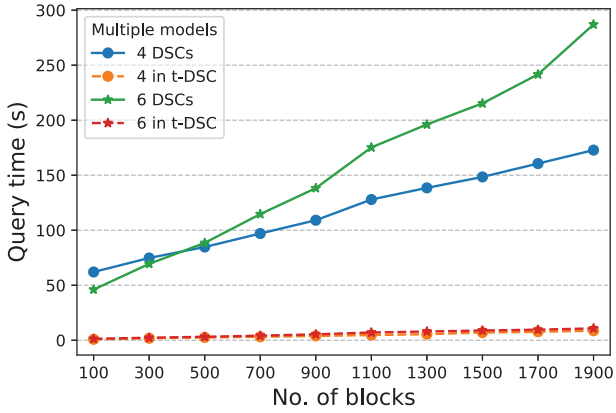


Fig. 8. Query time w.r.t. no. of blocks.

Evaluation on Throughput. For the throughput, we measure the total size of data processed in the system, which mainly consists of the data processed by different nodes in each iteration of the federated learning, and the data processed in the blockchain for generating and recording the verification information. Experiments are conducted with different amounts of participants joining in the federated learning task. Moreover, in order to comprehensively evaluate the system throughput, we also create four extra experimental scenarios with 5, 10, 20, and 30 iterations in the federated learning, respectively. Figure 6 illustrates the total throughput of the system with regard to the number of participants within four iteration scenarios. The throughput of the system can be larger than 1.5 MB/s with 5 participants conducting 10 iterations of learning. And if 10 participants conduct 30 iterations in federated learning, the throughput of the system will be reduced to 691 KB/s. With this figure, we can find that the throughput of the system will increase with the number of participants and then fall down in some degree. The reason is that, for each iteration, more data will be generated with more participants, and then the system will meet with the performance bottleneck and the data congestion. In addition, the number of iteration will have less effect on the throughput since the data size will dramatically change in each iteration.

Evaluation on Model Audit. For the model audit, we measure the total query time of VFChain, which mainly refers to the search time of blocks over the designed authenticated data structure (i.e., DSC). Experiments are conducted with different numbers of blocks. Moreover, in order to comprehensively evaluate the performance of DSC, we also create five extra experimental scenarios with 0, 1, 2, 3, and 4 skip levels in the DSC, respectively. Specially, the DSC with the 0 skip level is actually the original blockchain, which can be viewed as a baseline. Figure 7 plots the total query time of the system with regard to the number of blocks within five skip level scenarios. The system will cost 63.37 s query time with the baseline (i.e., 0 skip level), if we query 1900 various blocks over the blockchain. And when we query the same amount of blocks over the DSC with 4 skip levels, the query time will be rapidly reduced to 7.96 s. From the experiment results, we can also find that, with more skip levels equipped in the DSC, the system could return the query results more quickly. This is because with more skip levels, the system could neglect more blocks when conduct the query over the DSC. In addition, the query time will rationally increase with more number of blocks in the query.

Evaluation on Multiple-Model Optimization. For the multiple-model learning tasks, we compare the total query time with or without the multiple-model optimization scheme. Concretely, we compare the search times of blocks over the task-level DSC (t-DSC) in the designed integrative audit layer and the individual DSCs. Experiments are also conducted with different numbers of blocks. Without loss of generality, we totally derive 6 different models from the CNN through changing the number of neurons in internal layers. In order to evaluate the effectiveness of the optimization scheme, we conduct the comparison experiments for 4 models and 6 models with t-DSC and DSC, respectively. Figure 8 plots the total query time through the integrative audit layer with regard to the number of blocks within four scenarios. In 4 models scenario, the system will cost 172.82 s query time for 1900 blocks through 4 individual DSCs, while the query time for the same amount of blocks could be reduced to 8.647 s with the t-DSC. And in the 6 models scenario, the system will cost 287.016 s query time for 1900 blocks through 6 individual DSCs, while the query time for the same amount of blocks could be reduced to 10.780 s with the t-DSC. In addition,

from the experiment results, we can also find that the query time will prominently increase with more models using the individual DSCs.

VI. Related Work

Among a large number of works in the literature of verifiable federated learning, the design of VFChain is closely related to the following categories of research.

Security of Federated Learning: Recently, the security of federated learning has arisen a surge of interests in research community. On one hand, many effective attacks have been proposed. For example, to generate incorrect analysis results, attackers might compromise a fraction of data owners' machines to inject the adversarial samples [42] or poison the model updates [43], [44]. On the other hand, some works have proposed various methods to improve the security and preserve the privacy for federated learning systems. Weng *et al.* [45] present an incentive mechanism based on blockchain to force correct behaviours from the participating nodes and provide a privacy-preserving scheme for model updates in distributed deep learning. Pokhrel *et al.* [46] leverage blockchain to exchange and verify model updates for federated learning to protect the data privacy in autonomous vehicle scenarios. In this work, the authors also analyze the end-to-end delay to provide insights for communication efficiency improvement. Mugunthan *et al.* [47] utilize the differential privacy technique to the model contribution and accordingly reward agents with the quality of their contributions. Meanwhile, the privacy of the underlying dataset can be protected and be resilient to various malicious adversaries. However, different from the existing efforts, our proposed VFChain attempts to provide the verifiability and auditability for training procedure to improve the security of federated learning.

Blockchain Technology: Blockchain technology, as a decentralized and immutable ledger with time order, has been exposed to considerable attention in both research community and industry [48]. Some latest Blockchain techniques, including Ethereum and Hyperledger, introduce smart contract to support various decentralized applications. In terms of the consensus protocol in the blockchain, the latest Algorand protocol [37] has been proposed as a hybrid consensus protocol based on the PoS and BFT. The Algorand protocol can work in the permissioned scene with the existing of a synchronous network. Different from the PoW-based protocols, Algorand can provide a deterministic guarantee for block creation, in which a valid block added in the blockchain would not be removed in the future. Many researchers have used this emerging technique to solve some specific security problems in different real application scenarios, such as the software update management [49], cloud storage [50] and machine learning system [51]. In order to improve the security of blockchain system, some researchers propose many new types of attacks and corresponding defense schemes in blockchain systems [52], [53]. Considering the privacy of data stored in the blockchain, many new methods have been developed to protect the transaction privacy by employing effective cryptographic tools for blockchain, such as Zerocash [54], Zerocoin [55] and Hawk [56]. In addition, for the data query problem of blockchain, some schemes have been proposed to improve the query efficiency in different blockchain systems, such as vChain [57], VQL [58], and GEM2-Tree [59]. Benefitted from the decentralization and immutability properties of blockchain, we could build a verifiable and auditable federated learning framework based on the blockchain system.

VII. Conclusion

In this paper, we propose VFChain, a blockchain-based verifiable and auditable federated learning framework. Compared with previous works, VFChain represents the first attempt to cope with verifiability and auditability to build a secure federated learning framework based on the blockchain system. To realize this system, a committee is selected through the blockchain to collectively aggregate models and record verifiable proofs in the blockchain. Then, a novel authenticated data structure is proposed for blockchain to improve the search efficiency of verifiable proofs and support a secure rotation of committee. Finally, we design an optimization scheme to support multiple-model learning tasks to further improve the efficiency. We implement VFChain and conduct extensive experiments based on the popular deep learning model and public real-world dataset. The evaluation results demonstrate the effectiveness of VFChain.

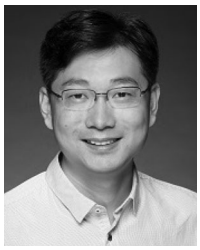
References

[1] S. Gu *et al.*, "Learned dynamic guidance for depth image reconstruction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 10, pp. 2437–2452, Oct. 2020.
[2] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.
[3] Z. Peng, Y. Yao, B. Xiao, S. Guo, and Y. Yang, "When urban safety index inference meets location-based data," *IEEE Trans. Mobile Comput.*, vol. 18, no. 11, pp. 2701–2713, Nov. 2019.
[4] Z. Peng, S. Gao, B. Xiao, G. Wei, S. Guo, and Y. Yang, "Indoor floor plan construction through sensing data collected from smartphones," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4351–4364, Dec. 2018.
[5] Y. Yuan and M. Q.-H. Meng, "Automatic bleeding frame detection in the wireless capsule endoscopy images," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 1310–1315.
[6] Y. Yuan, A. Hoogi, C. F. Beaulieu, M. Q.-H. Meng, and D. L. Rubin, "Weighted locality-constrained linear coding for lesion classification in ct images," in *Proc. IEEE 37th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, 2015, pp. 6362–6365.
[7] P. Hu *et al.*, "Bluememo: Depression analysis through twitter posts," in *Proc. IJCAI*, 2020, pp. 5252–5254.
[8] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, 2017, pp. 1273–1282.
[9] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang, "Learning differentially private recurrent language models," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–14.
[10] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," 2019, *arXiv:1908.09635*.
[11] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 911–926, 2019.
[12] Z. Ghodsi, T. Gu, and S. Garg, "Safetynets: Verifiable execution of deep neural networks on an untrusted cloud," in *Proc. NIPS*, 2017, pp. 4672–4681.
[13] F. Tramer and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–19.

[14] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Commun. Lett.*, vol. 24, no. 6, pp. 1279–1283, Jun. 2020.
[15] X. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, "Flchain: A. blockchain for auditable federated learning with trust and incentive," in *Proc. IEEE 5th Int. Conf. Big Data Comput. Commun.*, 2019, pp. 151–159.
[16] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Apr. 2020.
[17] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.
[18] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020.
[19] I. Eyal, A. E. Gencer, E. G. Sirer, and R. Van Renesse, "Bitcoin-ng: A scalable blockchain protocol," in *Proc. USENIX NSDI*, 2016, pp. 45–59.
[20] G. Wood, "Ethereum: A. secure decentralised generalised transaction ledger," *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, 2014.
[21] D. Ron and A. Shamir, "Quantitative analysis of the full bitcoin transaction graph," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2013, pp. 6–24.
[22] N. Z. Aitzhan and D. Svetinovic, "Security and privacy in decentralized energy trading through multi-signatures, blockchain and anonymous messaging streams," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 5, pp. 840–852, Sep./Oct..
[23] C. Decker, J. Seidel, and R. Wattenhofer, "Bitcoin meets strong consistency," in *Proc. ACM ICDCN*, 2016, pp. 1–10.
[24] S. Barber, X. Boyen, E. Shi, and E. Uzun, "Bitter to better-how to make bitcoin a better currency," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2012, pp. 399–414.
[25] A. Kiayias, A. Russell, B. David, and R. Oliynykov, "Ouroboros: A. provably secure proof-of-stake blockchain protocol," in *Proc. CRYPTO*, 2017, pp. 357–388.
[26] M. Castro and B. Liskov, "Practical byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, 2002.
[27] M. Bellare and T. Ristov, "A characterization of chameleon hash functions and new, efficient designs," *J. Cryptol.*, vol. 27, no. 4, pp. 799–823, 2014.
[28] L. Chen, P. Koutris, and A. Kumar, "Towards model-based pricing for machine learning in a data marketplace," in *Proc. ACM SIGMOD*, 2019, pp. 1535–1552.
[29] A. B. Kurtulmus and K. Daniel, "Trustless machine learning contracts; evaluating and exchanging machine learning models on the ethereum blockchain," 2018, arXiv:1802.10185.
[30] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. IEEE Symp. Secur. Privacy*, 2019, pp. 691–706.
[31] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Secur. Privacy*, 2019, pp. 739–753.
[32] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: Information leakage from collaborative deep learning," in *Proc. ACM CCS*, 2017, pp. 603–618.
[33] M. Zamani, M. Movahedi, and M. Raykova, "Rapidchain: Scaling blockchain via full sharding," in *Proc. ACM CCS*, 2018, pp. 931–948.
[34] C. Cai, Y. Zheng, A. Zhou, and C. Wang, "Building a secure knowledge marketplace over crowdsensed data streams," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/TDSC.2019.2958901.
[35] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," *J. Cryptol.*, vol. 20, no. 1, pp. 51–83, 2007.
[36] E. Kokoris-Kogias, E. C. Alp, L. Gasser, P. Jovanovic, E. Syta, and B. Ford, "CALYPSO: Private data management for decentralized ledgers," in *Proc. IEEE Symp. Very Large-Scale Data Anal. Vis.*, 2021, pp. 1–19.
[37] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proc. ACM SOSP*, 2017, pp. 51–68.
[38] S. Chaudhuri, U. Dayal, and V. Narasayya, "An overview of business intelligence technology," *Commun. ACM*, vol. 54, no. 8, pp. 88–98, 2011.
[39] C.-C. Chan, Y.-C. Lin, and M.-S. Chen, "Recommendation for advertising messages on mobile devices," in *Proc. ACM WWW*, 2014, pp. 235–236.
[40] C. Zeng, Q. Wang, S. Mokhtari, and T. Li, "Online context-aware recommendation with time varying multi-armed bandit," in *Proc. ACM SIGKDD*, 2016, pp. 2025–2034.
[41] D. Wu and J. R. Birge, "Risk intelligence in big data era: A. review and introduction to special issue," *IEEE Trans. Cybern.*, vol. 46, no. 8, pp. 1718–1720, Aug. 2016.
[42] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *Proc. IEEE Symp. Secur. Privacy*, 2018, pp. 19–35.
[43] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *Proc. ICML*, 2019, pp. 634–643.
[44] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. AISTATS*, 2020, pp. 2938–2948.
[45] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *IEEE Trans. Dependable Secure Comput.*, to be published.
[46] S. R. Pokhrel and J. Choi, "Federated learning with blockchain for autonomous vehicles: Analysis and design challenges," *IEEE Trans. Commun.*, vol. 68, no. 8, pp. 4734–4746, Aug. 2020.
[47] V. Mugunthan, R. Rahman, and L. Kagal, "Blockflow: An accountable and privacy-preserving solution for federated learning," 2020, *arXiv:2007.03856*.
[48] M. Ben-Or and A. Hassidim, "Fast quantum byzantine agreement," in *Proc. ACM STOC*, 2005, pp. 481–485.
[49] K. Nikitin *et al.*, "Chainiac: Proactive software-update transparency via collectively signed skipchains and verified builds," in *Proc. USENIX Secur.*, 2017, pp. 1271–1287.
[50] S. Hu, C. Cai, Q. Wang, C. Wang, X. Luo, and K. Ren, "Searching an encrypted cloud meets blockchain: A. decentralized, reliable and fair realization," in *Proc. of IEEE Conf. Comput. Commun.*, 2018, pp. 792–800.
[51] M. Shen, X. Tang, L. Zhu, X. Du, and M. Guizani, "Privacy-preserving support vector machine training over blockchain-based encrypted iot data in smart cities," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7702–7712, Oct. 2019.
[52] S. Gao, Z. Li, Z. Peng, and B. Xiao, "Power adjusting and bribery racing: Novel mining attacks in the bitcoin system," in *Proc. ACM CCS*, 2019, pp. 833–850.
[53] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.*, 2014, pp. 436–454.
[54] E. B. Sasson *et al.*, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, 2014, pp. 459–474.
[55] I. Miers, C. Garman, M. Green, and A. D. Rubin, "Zerocoin: Anonymous distributed e-cash from bitcoin," in *Proc. IEEE Symp. Secur. Privacy*, 2013, pp. 397–411.
[56] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *Proc. IEEE Symp. Secur. Privacy*, 2016, pp. 839–858.
[57] C. Xu, C. Zhang, and J. Xu, "vchain: Enabling verifiable boolean range queries over blockchain databases," in *Proc. ACM SIGMOD*, 2019, pp. 141–158.
[58] Z. Peng, H. Wu, B. Xiao, and S. Guo, "VQL: Providing query efficiency and data authenticity in blockchain systems," in *Proc. IEEE 35th Int. Conf. Data Eng. Workshops*, 2019, pp. 1–6.
[59] C. Zhang, C. Xu, J. Xu, Y. Tang, and B. Choi, "Gem2-tree: A. gas-efficient structure for authenticated range queries in blockchain," in *Proc. IEEE 35th Int. Conf. Data Eng.*, 2019, pp. 842–853.

Zhe Peng received the B.S. degree from Northwestern Polytechnical University, Xi'an, China, in 2010, the M.S. degree from the University of Science and Technology of China, Hefei, China, in 2013, and the Ph.D. degree in computer science from the Hong Kong Polytechnic University, Hong Kong, in 2018. He is currently a research Assistant Professor with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. His research interests include blockchain system, mobile computing, and data security and privacy.

Jianliang Xu received the B.Eng. degree in computer science and engineering from Zhejiang University, Hangzhou, China, and the Ph.D. degree in computer science from the Hong Kong University of Science and Technology, Hong Kong. He is currently a Professor with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. He has authored or coauthored more than 200 technical papers in his research fields, with an H-index of 48, which include big data management, blockchain, mobile computing, and data security, and privacy. He has been an Associate Editor for IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and the *Proceedings of the VLDB Endowment*.

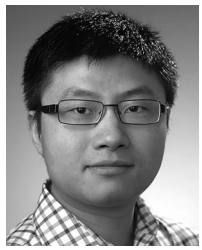


Xiaowen Chu received the B.E. degree in computer science from Tsinghua University, Beijing, China, in 1999 and the Ph.D. degree in computer science from the Hong Kong University of Science and Technology, Hong Kong, in 2003. He is currently a Full Professor with the Department of Computer Science, Hong Kong Baptist University, Hong Kong. His research interests include distributed and parallel computing and wireless networks. He is an Associate Editor for IEEE ACCESS and IEEE INTERNET OF THINGS.

Shang Gao received the B.S. degree from Hangzhou Dianzi University, Hangzhou, China, in 2010, the M. E. degree from Southeast University, Nanjing, China, in 2014, and the Ph.D. degree from The Hong Kong Polytechnic University, Hong Kong, in 2019. He is currently a Research Assistant Professor with the Department of Computing, The Hong Kong Polytechnic University. His research interests include information security, network security, data privacy, blockchain security, and applied cryptography.

Yuan Yao received the B.S., M.S., and Ph.D. degrees in computer science from Northwestern Polytechnical University (NPU), Xi'an, China, in 2007, 2009, and 2015, respectively. Prior to joining the Faculty with NPU, he was a Postdoctoral Researcher with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. He is currently an Associate Professor with the School of Computer Science, Northwestern Polytechnical University. His research interests include the area of real-time and embedded systems, swarm intelligence, cross-layer design in vehicular ad hoc networks, and security in vehicular networks.

Rong Gu received the Ph.D. degree in computer science from Nanjing University, Nanjing, China, in December 2016. He is currently an Associate Researcher with State Key Laboratory for Novel Software Technology, Nanjing University. His research interests include distributed systems and big data.



Yuzhe Tang received the B.S. and M.S. degrees in computer science and engineering from Fudan University, Shanghai, China, in 2006 and 2009, respectively, and the Ph.D. degree in computer science from the Georgia Institute of Technology, Atlanta, GA, USA, in 2014. He is an Assistant Professor with the Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, NY, USA. He has authored or coauthored technical research papers in prestigious conference proceedings and journals, including NDSS, ACSAC, ACM/IFIP Middleware, IEEE ICDCS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE ICDE, EDBT, ACM CIKM, and IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING. His research interests include security measurement, security protocol, trusted computing, cost-effective defense, and broadly distributed systems security. He was the recipient of the Best Paper Award in CCGRid 2015 and IEEE Cloud 2012.