Decentralized Privacy using Blockchain-Enabled Federated Learning in Fog Computing

Youyang Qu, *Student Member, IEEE,* Longxiang Gao, *Senior Member, IEEE,* Tom H. Luan, *Senior Member, IEEE,* Yong Xiang, *Senior Member, IEEE,* Shui Yu, *Senior Member, IEEE,* Bai Li, and Gavin Zheng

Abstract—As the extension of cloud computing and a foundation of IoT, fog computing is experiencing fast prosperity because of its potential to mitigate some troublesome issues such as network congestion, latency, and the local autonomy. However, privacy issues and the subsequent inefficiency are dragging down the performances of fog computing. The majority of existing works hardly consider a reasonable balance between them while suffering from poisoning attacks. To address the aforementioned issues, we propose a novel blockchain-enabled federated learning (FL-Block) scheme to close the gap. FL-Block allows local learning updates of end devices exchanges with blockchain-based global learning model, which is verified by miners. Built upon this, the FL-Block enables autonomous machine learning without any centralized authority to maintain the global model and coordinate by using a PoW consensus mechanism of blockchain. Furthermore, we analyze the latency performance of FL-Block and further derive the optimal block generation rate by taking communication, consensus delays, and computation cost into consideration. Extensive evaluation results show the superior performances of FL-Block from the aspects of privacy protection, efficiency, and resistance to poisoning attack.

Index Terms—Blockchain, Federated Learning, Privacy Protection, Fog Computing.

1 Introduction

Fog computing, which works as a synthesis of cloud computing and edge computing, is fast booming due to its proficiency in various aspects, such as reducing the latency and congestion of network while enabling autonomous systems [1]. This emerging paradigm guides new research fields for data dissemination, content sharing, and delivery services [2]. With existing practice and methodologies, interdependencies of network entities can be developed in fog computing for potential promising applications [3]. Fog computing can also integrate social interactions of devices with similar objectives, interests, or even mobility patterns in the virtual community of end devices, users, and fog servers in IoTs.

Despite of the provided conveniences, privacy issues are emerging because data sharing is trending to improve the service performances. It is essential for end devices to share local data to central authorities such as fog servers for global aggregation [4]. The uploaded data usually contains some unique identifiers, location information, and trajectories, which may lead to further privacy leakage or potential financial loss [5] [6]. On the other side, privacy protection should not dramatically degrade data utility [7]. Otherwise, the optimization and prediction results will be misled by the inaccurate global aggregation results, which invalidates the advanced services of fog computing.

In addition, the communication cost keeps increasing in this big data era. If certain privacy protection mechanisms are deployed, more communication resources will be occupied, which will be another bottleneck of fog computing [8] [9]. In this scenario, there is a high demand on real-time communication as network traffic flow undergoes a myriad of instant changes [10].

To address the aforementioned issues, some existing works have made attempts from different angles. Classic privacy protection framework, differential privacy [11] with its extensions [12], is leveraged to deal with the aggregation privacy. However, it is difficult to find a universal optimal trade-off between data utility and privacy protection [7]. Encryption-based methods, such as the work conducted in [13], can preserve privacy in a satisfying degree but is not scalable to big data scenarios. Federated learning [14], which is a new-merging technology, can achieve efficiency communication by exchanging updates between local models and global models [2]. But another concern arises that the central authority may be compromized and poisoning attacks are launched by adversaries. Blockchain is introduced to this scenario to eliminate the trust issues here [15]. But strictly providing privacy protection only on blockchain and efficiency encumber the feasibility of direct applications.

Motivated by this, we propose a blockchain-enabled federated learning (FL-Block) model by replacing the central authority with a specially designed blockchain with decentralized privacy protocols. In this model, end devices will upload the local updates to the fog servers, where the global updates will be generated and stored. Since only the pointer of the global updates is saved on-chain while

- *Youyang Qu, Longxiang Gao, and Yong Xiang are with the School of Information Technology, Deakin University, VIC, Australia. Email: {quyo, longxiang.gao, yong.xiang}@deakin.edu.au.*
- *Tom H. Luan is with the School of Cyber Engineering, Xidian University, Shaanxi, China. Email: tom.luan@xidian.edu.cn.*
- *Shui Yu is with the School of Software, University of Technology Sydney, NSW, Australia. Email: shui.yu@uts.edu.au.*
- *Bai Li is with Health Chain Technology, VIC, Australia. Email: peter.li@gaiatech.io.*
- *Gavin Zheng is with Nenglian Technology, Beijing, China. Email: gavinzheng@echoin.io*

a distributed hash table (DHT) is used to save the data, the block generation efficiency could be guaranteed. With a hybrid identity generation, comprehensive verification, access control, and off-chain data storage and retrieve, FL-Block enables decentralized privacy protection while preventing single point failure. Moreover, the poisoning attack could be eliminated from the side of fog servers.

The contributions of this work are summarized as follows.

- **Decentralized Privacy:** By integrating blockchain with federated learning, FL-Block enables decentralized privacy protection and prevent single point failure in fog computing scenario. In addition, blockchain could provide incentives to federated learning participants.
- **Poisoning Attack Proof:** Poisoning attacks can be eliminated because central authority is replaced by a novel blockchain system which provides non-tempering feature. The protection mechanism is further enhanced due to the elimination of poisoning attacks.
- **High Efficiency:** Two important characteristics jointly contribute to high efficiency. The first one is federated learning only requires the exchange of training updates. The other one is only the pointer is saved on blockchain while the associated data is saved in an off-chain distributed hash table.

The remainder of this paper is organized as follows. In Section 2, we present current research on relevant research in blockchain, fog computing, and federated learning. We continue to demonstrate decentralized privacy protection model using blockchain-enabled federated learning in Section 3 and Section 4, respectively. Section 5 presents the analysis on efficiency and block generation with and without optimal setting, which is followed by evaluations on real-world data sets in Section 6. At last, we conclude and summurize this paper in Section 7.

2 Related Work

In fog computing, the environment is open to access, which poses great challenges from the perspective of privacy protection in real-world applications [16]. Hu et al. [17] devised a novel Identity-based scheme to solve device-to-device and device-to-server communications in fog computing. However, there are potential risks due to the storage of the system's master key in every end device [18]. In addition, the scalability and resultant communication overhead are not taken into consideration. Privacy protection attracts increasing volume of interest from both academia and industry [19] [20]. From the perspective of anonymity, Kang et al. [21] designed a hybrid model that enhances the privacy protection by means of pseudonyms. Employing self-certification, it is possible to manage without the compromise of the system's robustness. To further improve, Tan et al. [22] developed an end device identity authentication protocol of end devices and fog servers without certification, while Wazid et al. [23] proposed another secure authentication and key management scheme to improve key security of the users in IoT scenarios. In [24], Gu et al. proposed a customizable privacy protection model to enable the personalized privacy-preserving data sharing using Markov decision process. This enables the private data sharing between end devices and fog servers, which meets numerous privacy demands. Nevertheless, these three models requires a trusted third party, which is another bottleneck in term of decentrilized privacy protection [25].

Blockchain has a specially designed distributed ledger structure that connects blocks in chronological order. The saved data is shared and maintained by all nodes in a decentralized environment. The leading advantages of blockchain are decentralization, non-tempering, open autonomy, and anonymous traceability [26]. By using blockchain, Jiao et al. [27] proposed a novel scheme to manage resource allocation for fog or cloud computing. Then, Rowan et al. [28] designed a blockchain-enhanced PKI with an inter-end device session key establishment protocol to provide privacy protection to device-to-device communications. Peng et al. [29] presented an onboard network anonymous authentication protocol. The strength of this model is guaranteeing the anonymity of users and efficient authentication, but it fails to detect malicious end devices. In addition, Lu et al. proposed a blockchain-based reputation system and Malik proposed a blockchain-based authentication in [30] and [31], respectively. Despite the characteristic of privacy preservation, these two models are incapable of end device announcement, which results in low scalability. To address the limitation of pre-storing lots of anonymous certificates, Lu et al. [13] designed a novel model to achieve conditional privacy. This model requires frequent change of anonymous certificates to avoid linkage attack, which hinders the performance of fog computing in term of efficiency. In [32], blockchain meets distribute hash table (DHT), which decouples validation from state storage.

Federated learning is a potential promising tool, which is proposed as a decentralized learning scheme where local data is distributed to the data owner [33] [34]. Federated learning enables each data owner to have a series of local learning parameters from the local model. Rather than simply sharing the training data, the data owners share their local parameter updates with a trusted third party. Konecny et al. [35] explored the applicability of several current models and thereby devised an extension to address the sparse data issues. In [36], Konecny et al. further present a variant of federated learning which minimizes the communication cost by uploading a lessened number of local parameters. Smith et al. [34] devised a distributed learning model in term of multiple relevant tasks using federated learning. Nishio et al. [37] also gave insights on how to select clients in a resource-limited mobile edge computing scenario. Lu et al. integrated federated learning into the consensus algorithm of blockchain to save the hashrate [38]. Samarakoon et al. [39] devised a learning based intrusion detection based on sample selected extreme learning in fog computing. However, this work did not take sharing resources and device-to-device communication into consideration. To the best of our knowledge, no existing work has studied the joint use of blockchain and federated learning in the context of efficiently privacy-preserving communication in fog computing scenario.

3 Architecture of Blockchain-Enabled Federated Learning

In this section, we present the architecture of the blockchain-enabled federated learning (FL-Block). We use v_i and m_j to denote different end devices and miners, respectively. Then ds_k is used to present different data samples. In addition, we use e_l to differentiate the global model update iterations, which is also referred to as epochs.

3.1 Federated Learning in FL-Block

In FL-Block, the federated learning is achieved by a cluster of end devices $V = \{1, 2, ..., N_V \in V\}$ where $|V| = N_V$. The i-th device V_i possesses a set of data samples DS_i where $|DS_i| = N_i$. V_i trains its local data and the local model updates of the device V_i is uploaded to its associated miner M_i instead of a trusted third party. The miner M_i is selected from a set of miners $M = \{1, 2, ..., N_M\}$ randomly, where $|M| = N_M$. $M = V$ could be satisfied when the miners M_i are physically identical to the end devices. Otherwise, we will have $M \neq DS$. Moving on, the total number of v_i of the local model updates are verified and shared among all possible miners. Finally, the aggregated global model updates are downloaded from each miner to its corresponding device.

To further improve, the decentralized model training concentrates on solving a linear regression problem on a series of parallel data samples $DS = \cup_{i=1}^{N_V} DS_i$ where $|DS| = N_S$. The k-th data sample $ds_k \in DS$ is given as $s_k = \{x_k, y_k\}$ for a d-dimensional column vector $x_k \in R^d$ as well as a scalar value $y_k \in R$. The objective of the linear regression problem is to minimize a pre-defined loss function $f(\omega)$ with respect to a d-dimensional column vector $\omega \in R^d$, which is regarded as a global weight. To simplify this process, the pre-defined loss function $f(\omega)$ is selected as the mean squared error (MSE) in the following context.

$$f(\omega) = \frac{1}{N_D S} \sum_{i=1}^{N_V} \sum_{s_k \in S_i} f_k(\omega), \tag{1}$$

where $f_k(\omega)(x_k^T \omega - y_k)^2/2$ and the notation $(.)^T$ denotes the vector transpose operation. This could be easily extended to various loss functions under diverse neural network models with minor operations.

For purpose of solving the aforementioned issues, the learning model of the end device V_i is trained locally with the data sample set DS_i using a stochastic variance reduced gradient (SVRG) algorithm, and all local model updates of the devices are aggregated using a distributed approximate Newton (DANE) method, which will be trained to generate the global model update.

To furtehr improve this model, the ceiling of the global model is set to L epochs. For each epoch, the local model of end device V_i is updated with the number N_i of iterations. Therefore, we can have the local weight $\omega_i^{(t,l)} \in R^d$ of the end device V_i at the t_th local iteration of the l-th epoch as

$$\omega_i^{(t,l)} = \omega_i^{(t-1,l)} - \frac{\beta}{N_i} \left\{ \left[\bigtriangledown f_k(\omega_i^{(t-1,l)}) - \bigtriangledown f_k(\omega^{(l)}) \right] + \bigtriangledown f(\omega^{(l)}) \right\}, \tag{2}$$

where $\beta > 0$ is defined as a step size, $\omega^{(l)}$ is the global weight at the l-th epoch, and $\bigtriangledown f(\omega^{(l)}) = 1/N_{DS}$. $\sum_{i=1}^{N_V} \sum_{s_k \in S_i} \bigtriangledown f_k(\omega^{(l)})$ is derived from Eq. 1. We use $\omega^{(l)}$ to represent the local weight when the last local iteration of the l-th epoch is finished, i.e., $\omega_i^{(l)} = \omega_i^{(N_i,l)}$. Built upon this, the update of the global weight $\omega^{(l)}$ is formulated as

$$\omega^{(l)} = \omega^{(l-1)} + \sum_{i=1}^{N_V} \frac{N_i}{N_{DS}} \left(\omega_i^{(l)} - \omega_i^{(l-1)} \right). \tag{3}$$

The iteration process of updating the local and global weights will continue until the constraint of global weight $\omega^{(l)}$ satisfies $|\omega^{(L)} - \omega^{(L-1)}| \leq \epsilon$ is satisfies, where $\epsilon > 0$ is a small positive constant.

In the classic federated learning settings, at the l-th epoch, the end device V_i is supposed to upload its local model update $\left(\omega_i^{(l)}, \{\bigtriangledown f_k(\omega^{(l)})\}_{s_k \in S_i} \right)$ to the fog servers, with the model update size δ_m that is identically specified for each end device. The global model updates $\left(\omega^{(l)}, \bigtriangledown f(\omega^{(l)}) \right)$ with the same size δ_m are computed by the fog servers, which will be downloaded to all end devices after processing. In FL-Block, the fog server entity is substituted with a blockchain network which is detailed discussed in the following subsection.

3.2 Blockchain in FL-Block

Regarding the blockchain network settings of FL-Block, the generated blocks and the cross verification by the miners M are devised to upload truthful data of the local model updates. This is achieved by developing a specially designed distributed ledger. The protocols of the distributed ledger are discussed in the next section in detail. Each block in the distributed ledger contains body and header sectors. In the classic blockchain structure, the body sector usually contains a set of verified transactions. In FL-Block, the body stores the local model updates of the devices in V, i.e., $\left(\omega_i^{(l)}, \{\bigtriangledown f_k(\omega^{(l)})\}_{s_k \in S_i} \right)$ for the device V_i at the l-th epoch, as well as its local computation time $T_{\{local,i\}}^{(l)}$. Extended from the classic blockchain structure, the header sector stores the information of a pointer to the previous block, block generation rate λ, and the output value of the consensus algorithm, namely, the nonce. In this context, the proof of work (PoW) is used as the instance consensus algorithm, while all other consensus algorithms can be extended into this scenario. In order to store the local model updates of all the end devices, the size of each block is defined as $h + \delta_m N_V$, where h and δ_m denote the header size and model update size, respectively.

Each miner is assigned with a candidate block that contains local model updates information from its binding end devices or other miners, in the order of arrival. The writing procedure will continue until the block size is fully-filled or a maximum waiting time T_{wait} is reached from the

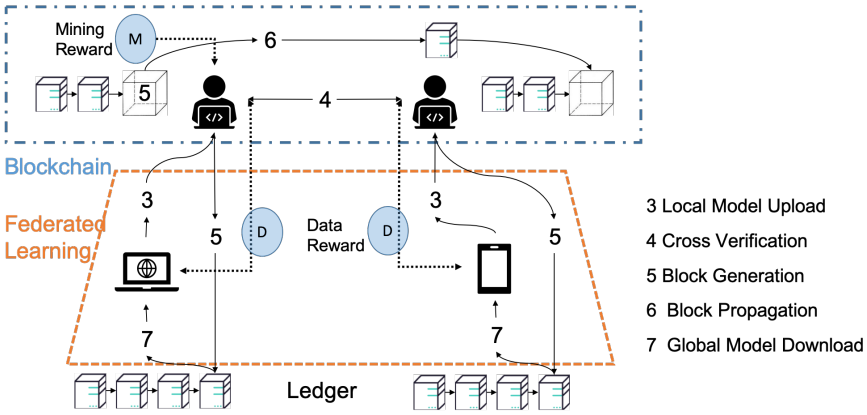


Fig. 1: Blockchain-Enabled Federated Learning Diagram

beginning of each epoch. To make it simplified, we use a sufficiently long T_{wait} so that every block is filled with local model updates of all end devices.

In the next stage, the miner keeps generating a random nonce until it becomes smaller than a target value using PoW. Once one of the miners M_1 works out the nonce, its corresponding candidate block is allowed to be generated as a new block. Intuitively, the block generation rate λ can be carefully controlled by modifying the difficulty of the deployed PoW consensus algorithm.

Furthermore, the newly-generated block is shared to all other miners for the purpose of synchronizing all existing distributed ledgers. To achieve this, all the other miners who received the newly-generated block are enforced to quit the calculation process. Then, the generated block is linked to the local ledgers maintained by the miners. However, there is a possibility of forking when another miner M_2 happens to generate a candidate block within the sharing delay of the eligible block. Some miners may falsely link the ineligible block to their local ledgers. In FL-Block, forking will mislead the end devices linking the ineligible block to train a poisoned global update and subsequently generate offtrack local updates in the next iteration.

The value of forking frequency is positively correlated with both the blockchain generate rate λ and the block sharing delay. The mitigation of forking will lead to an extra delay, which will be discussed in Section 5.

Apart from the above operation for local model updates uploading, the blockchain also provides proper rewards for data samples to the end devices, and for the verification operation to the miners, which is referred to as data reward and mining reward, respectively. The data reward of the end devices V_i is received from its corresponding miner, and the amount of the reward is linearly proportional to the size of the data sample N_i. After a block is generated by the miner M_j, the mining reward is earned from the blockchain network. Similarly, the amount of mining reward is linearly proportional to the aggregated data sample size of its all binding end devices, namely, $\sum_{i=1}^{N_{M_j}}$, where N_{M_j} denotes the number of end devices binding with the miner M_j. This will provide the incentive to the miners so that they will process more local model updates while compensating their expenditure for the data reward.

However, there is a side effect in this rewarding system. There might be malicious end devices deceiving the miners by poisoning their actual sample sizes. In this case, the miners are supposed to verify eligible local updates before storing the local model updates in their associated candidate blocks. In this context, the simplified verification operation is conducted by comparing the sample size N_i with its corresponding computation time $T^{(}_{\{local,i\}}l)$ that is assumed to be reliable, following the proof of elapsed time under Intel's SGX technology.

As shown in Fig. 1, the FL-Block operation of the end device V_i at the l-th epoch can be explained by eight procedures as follows.

Algorithm 1 Block-Enabled Federated Learning Algorithm

Input: End devices V_i, Fog servers, Miners M_i;
Output: Optimized Global Model;
1: Initialization;
2: **while** New data sample DS is generated **do**
3: Local model update by V_i;
4: Local model upload to fog servers;
5: Cross verification by M_i;
6: Block generation by the winning miner;
7: **if** M_i finds the nonce of the new block b_i **then**
8: Link b_i to the blockchain
9: Remove all other relevant blocks $\hat{b_i}$;
10: **end if**
11: Block propagation;
12: Global model update by fog servers;
13: Global model download to V_i;
14: **end while**
15: Output the eligible global model.

In the initialization part (l = 1), the initial parameters are uniformly randomly chosen from predefined ranges of the local and global weights $\{\omega_i^{(0)}, \omega^{(0)} \in (0, \omega_{max}]\}$ for a constant ω_{max}, and the global gradient $\bigtriangledown f(\omega^{(0)}) \in (0, 1]$.

In local model update phase, the end device V_i computes Eq. 2 with the number N_i of iterations.

In local model upload phase, the end device V_i uniformly randomly associates with the miner M_i. If $M = V$, miner M_i is selected from M V_i. The end device uploads the local model updates $\left(\omega_i^{(l)}, \{\bigtriangledown f_k(\omega^{(l)})\}_{s_k \in S_i}\right)$ and the corre-

sponding local computation time $T^{(l)}_{\{local,i\}}$ to the associated miner.

In cross verification phase, the miners broadcast the local model updates obtained from their associated end devices. At the same time, the miners verify the received local model updates from their associated devices or the other miners in the order of arrival. The truthfulness of the local model updates are validated, if the local computation time $T^{(l)}_{\{local,i\}}$ is proportional to the data sample size N_i. The verified local model updates are recorded in the miner's candidate block, until its reaching the block size $(h+\delta_m N_V)$ or the maximum waiting time T_{wait}.

In block generation phase, each miner starts running the PoW until either it finds the nonce or it receives a generated block from another miner.

In block propagation phase, denoting as $M_{\hat{o}} \in M$ the miner who first finds the nonce. Its candidate block is generated as a new block that is broadcasted to all miners. In order to avoid forking, an acknowledgment (ACK) signal is transmitted when each miner detects no forking event. Each miner waits until receiving ACK signals of all miners. Otherwise, the operation goes back and iterate from phase 2.

In global model download phase, the end device V_i downloads the generated block from its associated miner M_i.

In global model update phase, the end device V_i locally computes the global model update in Eq. 3 by using the aggregate local model updates stored in the generated block.

The one-epoch procedure continues until the global weight $\omega^{(L)}$ satisfies $\omega^{(L)} - \omega^{(L-1)} \leq \epsilon$. The centralized FL structure is vulnerable to the malfunction of fog server, which distorts the global model updates of all end devices. Compared to this, each end device in FL-Block locally computes its global model update, which is thus more robust to the malfunction of the miners that replace the fog server entity.

4 Decentralized Privacy Mechanism Based on FL-Block

This section presents the proposed decentralized privacy mechanism in fog computing built upon FL-Block. In Fig. 2, the network of fog servers is referred as distributed hash table (*DHT*). End devices communicate with fog servers through various provided services.

4.1 Blocks Establishment

To establish blocks, we orderly define hybrid identity, memory of blockchain, policy, and auxiliary functions.

4.1.1 Hybrid identity

The traditional blockchain identity uses a naive anonymous mechanism. By using a public key, every end device can generate an unlimited number of pseudo-identities if necessary. To extend it to our model, we devise a novel *hybrid identity*. This hybrid identity is a form of personalization of identities. When it is shared to different end devices or fog servers, the owner has full access to it while the other parties have limited access according to specific requirements.

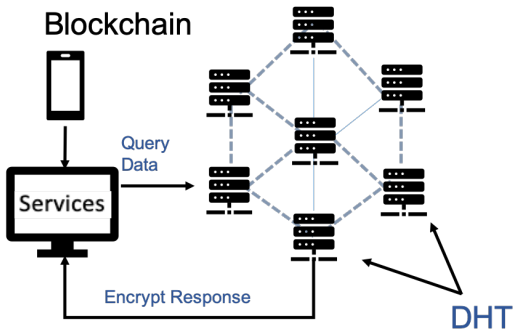


Fig. 2: Decentralized Privacy in Fog Computing Based on FL-Block

In Pro. 1, we illustrate an instance of a sole end device identity owner (u_o) and two guest end devices, namely, identity recipients (u_g). The identity contains generating key-pairs for the sole owner and two guests. In addition, a symmetric key is required to both encrypt and decrypt the data. In this way, this data is restricted to the other end devices in fog computing. The hybrid identity is defined by a 4-tuple as Equ. 4.

Protocol 1 Hybrid Identity Generation

Inputs: u_o, u_{g_1}, and u_{g_2}

Goal: $\mathrm{pk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{pk}_{sig_1}^{u_{g_i},u_o}, \mathrm{sk}_{enc_i}^{u_o,u_{g_i}}$

1. u_o execusions
   1) $(\mathrm{pk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{sk}_{sig_i}^{u_o,u_{g_i}}) \leftarrow G_{sig}()$
   2) $\mathrm{sk}_{enc_i}^{u_o,u_{g_i}} \leftarrow G_{enc}()$
   3) u_o shares $\mathrm{pk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{sk}_{enc_i}^{u_o,u_{g_i}}$ to u_{g_1} and u_{g_2}
2. u_{g_1} execusions
   1) $(\mathrm{pk}_{sig_i}^{u_{g_i},u_o}, \mathrm{sk}_{sig_i}^{u_{g_i},u_o}) \leftarrow G_{sig}()$
   2) u_{g_1} shares $\mathrm{pk}_{sig_i}^{u_o,u_{g_i}}$ to u_{g_2} and other u_{g_i}s
3. u_{g_2} execusions
   1) $(\mathrm{pk}_{sig_i}^{u_{g_i},u_o}, \mathrm{sk}_{sig_i}^{u_{g_i},u_o}) \leftarrow G_{sig}()$
   2) u_{g_2} shares $\mathrm{pk}_{sig_i}^{u_o,u_{g_i}}$ to u_{g_1} and other u_{g_i}s

All users have $(\mathrm{pk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{pk}_{sig_i}^{u_{g_i},u_o}, \mathrm{sk}_{enc_i}^{u_o,u_{g_i}})$

$$\mathrm{Hybrid}_{u,s}^{(public)} = (\mathrm{pk}_{sig_1}^{u_o,u_{g_1}}, \mathrm{pk}_{sig_2}^{u_o,u_{g_2}}, \mathrm{pk}_{sig_1}^{u_{g_1},u_o}, \mathrm{pk}_{sig_2}^{u_{g_2},u_o}) \tag{4}$$

Analogically, the integrated hybrid identity will be a 10-tuple, but we use $i|i=1,2$ as an index to simplify it to a 5-tuple as Equ. 5.

$$\mathrm{Hybrid}_{u,s} = (\mathrm{pk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{sk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{pk}_{sig_i}^{u_{g_i},u_o}, \mathrm{sk}_{sig_i}^{u_{g_i},u_o}, \mathrm{sk}_{enc_i}^{u_o,u_{g_i}}) \tag{5}$$

4.1.2 Memory of Blockchain and Policy

In term of Blockchain memory, let BM be the memory space. We have $BM : 0,1^{256} \rightarrow 0,1^N$ where $N >> 256$ and it is sufficient to store large data files. Blockchain is

like an account book containing a list of model updates with a timestamp. The first two outputs in a model update encode the 256-bit memory address pointer along with some auxiliary meta-data. The other outputs are leveraged to build the serialized document. If $L[k]$ is queried, the model update with the up-to-date timestamp will be returned. This setting allows inserting, delete, and update operation.

We define the policy P_u as a series of permissions that an end device v can gain from a specific service. For instance, if v needs to read, update, and delete a dataset, then $P_v = read, update, delete$. Any data could be safely stored as service will not break the protocols and label the data incorrectly. In addition, the service can easily observe the anomaly of end devices since all changes are visible.

4.1.3 Accessional Functions

Two accessional functions are necessary in this case, including $\mathrm{Parse}(x)$ and $\mathrm{Verify}(\mathrm{pk}_{sig}^{k}, x_p)$. $\mathrm{Parse}(x)$ continuously passes the arguments to a specific transaction. $\mathrm{Verify}(\mathrm{pk}_{sig}^{k}, x_p)$ help verify the permissions of end devices. The protocol of verify function shown in Pro. 2.

Protocol 2 Permission Verification

Inputs: $\mathrm{Verify}(\mathrm{pk}_{sig}^{k}, x_p)$

Goal: Verification Index s

1. Initialize $s = 0$
2. Initialize $a_{policy} = H(\mathrm{pk}_{sig}^{k})$
3. **if** $BM[a_{policy}] \neq \emptyset$ **then**
4. $\quad \mathrm{pk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{pk}_{sig_i}^{u_{g_i},u_o}, P_{u_o,u_{g_i}} = \mathrm{Parce}(BM[a_{policy}])$
5. $\quad$ **if** $\mathrm{pk}_{sig}^{k} == \mathrm{pk}_{sig_i}^{u_{g_i},u_o}$ **or**
6. $\quad (\mathrm{pk}_{sig}^{k} == \mathrm{pk}_{sig_i}^{u_{g_i},u_o}$ **and** $x_p \in P_{u_o,u_{g_i}})$ **then**
7. $\quad\quad s = 1$
8. $\quad$ **end if**
9. **end if**
10. **return: Verification Index** s

4.2 Blockchain Protocols Design

In this subsection, we show more detailed protocols of the proposed blockchain model. When a model update A_{access} is recorded, Pro. 3 is executed by the nodes inside the network. Analogically, when a model update A_{data} is recorded, Pro. 4 is executed by the nodes.

As A_{access} is used to conduct access control management, it can change the permissions of end devices granted by the service. This is accomplished by sending the policy $P_{u_o,u_{g_i}}$. If the service wants to revoke all the access permissions, the policy will be $P_{u_o,u_{g_i}}(\emptyset)$. If it is the first time to send a A_{access} with a new hybrid identity, the A_{access} will be recorded as an end device signing up to a service.

Analogously, A_{data} will manage the data manipulation operations such as read, write, update, or delete. With the assistance of the Verify() function, only the service of fog severs or the permitted end devices can access the differentially private data. In Pro. 4, we access the distributed hash table like a normal hash table. In real-world scenarios, these instructions bring about some off-chain network messages which are being sent to the distributed hash table.

Protocol 3 Access Control Management

Inputs: $\mathrm{Access}(\mathrm{pk}_{sig}^{k}, m)$

Goal: Access Control Index s

1. Initialize $s = 0$
2. Initialize $\mathrm{pk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{pk}_{sig_i}^{u_{g_i},u_o}, P_{u_o,u_{g_i}} = \mathrm{Parce}(m)$
3. **if** $\mathrm{pk}_{sig}^{k} == \mathrm{pk}_{sig_i}^{u_{g_i},u_o}$ **then**
4. $\quad L[H(\mathrm{pk}_{sig}^{k})] = m$
5. $\quad s = 1$
6. **end if**
7. **return:** s

Protocol 4 Data Load and Storage

Inputs: $\mathrm{DATA}(\mathrm{pk}_{sig}^{k}, m)$

Goal: Proper Data Manipulation

1. Initialize $c, x_p, rw = \mathrm{Parse}(m)$
2. **if** $\mathrm{Verify}(\mathrm{pk}_{sig}^{k}, x_p) = \mathrm{TRUE}$ **then**
3. **if** $\mathrm{pk}_{sig}^{k} == \mathrm{pk}_{sig_i}^{u_{g_i},u_o}$ **then**
4. $\quad \mathrm{pk}_{sig_i}^{u_o,u_{g_i}}, \mathrm{pk}_{sig_i}^{u_{g_i},u_o}, P_{u_o,u_{g_i}} = \mathrm{Parce}(L[H(\mathrm{pk}_{sig}^{u_o,u_{g_i}})])$
5. $\quad a_{x_p} = H(\mathrm{pk}_{sig}^{u_o,u_{g_i}} || x_p)$
6. $\quad$ **if** $rw = 0$ **then**
7. $\quad\quad h_c = H(c)$
8. $\quad\quad L[a_{x_p}] = L[a_{x_p}] \cup h_c$
9. $\quad\quad ds[h_c] = c$
10. $\quad\quad$ **return:** h_c
11. $\quad$ **else if** $c \in L[a_{x_p}]$ **then**
12. $\quad\quad$ **return:** $ds[h_c]$
13. $\quad$ **end if**
14. **end if**
15. **return:** $\emptyset$

4.3 Discussion on Decentralized Privacy Protection using Blockchain

We have some following assumptions in this work. The first one is on the blockchain, which is tamper-proof. This requires a large enough network to avoid untrusted end devices taking over it. The second one is that end devices can properly secure their keys during the operations, for example, using some secure services from fog servers. From here, we will illustrate how the system can protect adversaries from tampering the data stored in a blockchain network. In this scenario, we take less consideration on adversaries modifying protocols or steal personal sensitive information.

In this model, only the services have full control over the sensitive data. An adversary can hardly pretend to be an end device or corrupt the whole network since the blockchain is fully decentralized. In addition, digital signatures are required for model updates. Therefore, we hold that adversaries are not able to fabricate digital signatures or take control of the majority of the network (over 50%). Furthermore, an adversary cannot poison the data because it is stored off-chain rather on the public ledger. There are only pointers information encrypted with a hash function inside a public ledger.

Even if we consider the case that an adversary controls one or some of the nodes in the DHT network, the adversary cannot learn anything about the data. The rationale behind this is that the data is encrypted with keys that no other nodes have access. The worst case is that the adversary gains the authority and compromise a few local copies of the data, the system can still recover it since there are abundant replications distributed across the whole network.

Last but not least, the hybrid identity mechanism ensures that there is only a tiny probability that the data is poisoned because this requires the acquisition of both signing key and encryption-decryption key. If the adversaries happen to steal one of the keys, the sensitive data is still safe. In practice, we can also personalize the hybrid identity so that the compromization is restricted for the adversaries. A good instance would be different keys for a certain volume of records.

5 System Analysis

In this section, we first analyze poisoning attacks. Then, we consider a reference end device $V_o \in V$ that is randomly selected. The objective is to derive the optimal block generation rate $\lambda*$, which minimizes the learning completion latency T_o. In this work, we define T_o as the total elapsed time during L epochs at the end device V_o. This latency is linearly proportional to the l-th epoch latency, i.e., $T_o = \sum_{l=1}^{L} T_o^{(l)}$. Thus, we hereafter focus only on $T_o^{(l)}$ without loss of generality.

5.1 Poisoning Attacks and Defence

Beyond the naive approach that an adversary train the local model based on specially-designed falsified data, the enhanced attack is referred as model substitution, in which the adversary tries to replace the global model GM^{t+1} with a malicious model MM, which is shown in Eq. 6.

$$\begin{aligned} G_{t+1} &= GM^t + \frac{r}{n}\sum_{j=1}^{m}\left(LM_j^{t+1} - GM^t\right) \\ \Rightarrow MM &= GM^t + \frac{r}{n}\sum_{j=1}^{m}\left(LM_j^{t+1} - GM^t\right) \end{aligned} \tag{6}$$

in which t is the current time slot, r is the learning rate, LM is a local model, n is number the total edge devices, and m is the number participants.

Since non-IID data is used in this case, each local model may be quite different from the global model. With the convergence of the global model, the deviations cancel out bit by bit, which can be denoted as $\sum_{j=1}^{m-1}(LM_j^{t+1} - GM^t) \to 0$. Based on this, an adversary may upload a model as

$$\begin{aligned} LM_m^{t+1} &= \frac{n}{r}MM - \left(\frac{n}{r} - 1\right)GM^t - \sum_{j=1}^{m-1}\left(LM_j^{t+1} - GM^t\right) \\ &\simeq \frac{n}{r}\left(MM - GM^t\right) + GM^t. \end{aligned} \tag{7}$$

This enhanced attack increases the weights of the malicious model MM by $\eta = n/r$ to guarantee the replacement of the global model GM by MM. This attack functions better when the convergence is nearly reached. In addition, if the adversary is blind to the value of n and r, he/she can simply increase the value of η in every iteration. Although scaling by $\eta \leq n/r$ only partially replace the GM, the attack still functions well.

5.2 Single-Epoch FL-Block Latency Model

Following the FL-Block operation in Section 3.2, the end device $V_o's$ l-th epoch latency $T_o^{(l)}$ is not only determined by communication and block generation delays but also determined by the computation cost by federated learning.

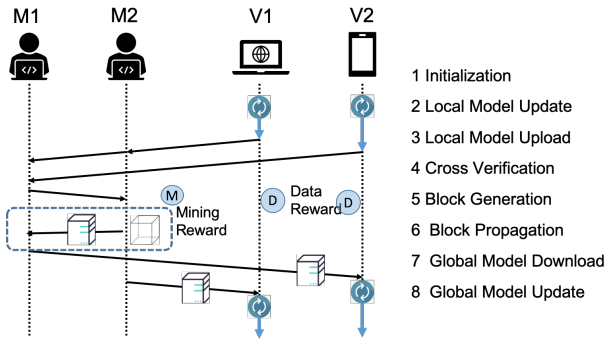


Fig. 3: Single-Epoch Operation without Forking

To begin with, the computation costs are caused by phase 2 and phase 8 defined in Section 3.2. Let δ_d denote the size of a single data sample that is given identically for all data samples. δ_d/f_c is required to process δ_d with the clock speed f_C. We formulate $T_{\{local,i\}}^{(l)}$ (local model updating delay) in phase 2 as $T_{\{local,i\}}^{(l)} = \delta_d N_o/f_c$. Analogically, we define the global model updating delay $T_{\{global,o\}}^{(l)}$ in phase 8 as $T_{\{global,o\}}^{(l)} = \delta_m N_V/f_c$.

Then, we discuss the communication delays existed in phase 3 and phase 7 between the miners and corresponding end devices. With additive white Gaussian noise (AWGN) channels, we evaluate the achievable rate using Shannon capacity. In term of phase 3, the local model uploading delay $T_{\{up,o\}}^{(l)}$ is formulated as $T_{\{up,o\}}^{(l)} = \delta_m / \Big[W_{up}\log_2(1 + \lambda_{\{up,o\}})\Big]$, where W_{up} is known as the uplink bandwidth allocation for each end device while $\lambda_{\{up,o\}}$ is the received signal-to-noise ratio (SNR) of miner M_o. Similarly, in phase 7, we define the communication delay of global model. The downloading delay $T_{\{vn,o\}}^{(l)}$ is described as $T_{\{vn,o\}}^{(l)} = h + \delta_m N_V / \Big[W_{dn}\log_2(1 + \lambda_{\{dn,o\}})\Big]$, where W_{dn} is the downlink bandwidth allocation for each device and $\lambda_{\{dn,o\}}$ is the received signal-to-noise ratio of device V_o.

In addition, the communication delay also exists when miners are communicating in the blockchain network, which is part of phase 4 and phase 5. Since the cross-verification time is insignificant compared with the communication costs, the cross-verification delay $T_{\{cross,o\}}^{(l)}$ in phase 4 is defined under frequency division multiple access (FDMA) as

$$T^{(l)}_{\{cross,o\}} = \max\Bigg\{T_{wait} - \Big(T^{(l)}_{\{local,i\}} + T^{(l)}_{\{up,o\}}\Big), \sum_{M_j \in M \setminus M_o} \delta_m N_{M_j} / \Big[W_m \log_2(1+\lambda_{oj})\Big]\Bigg\}, \tag{8}$$

where we use W_m to denote the bandwidth allocation per each miner link and λ_{oj} is the received SNR from the miner M_o to the miner M_j. Similarly, $M_{\hat{o}} \in M$ is regarded as the miner who first finds the nonce. Then, the total block propagation delay $T^{(l)}_{\{bp,\hat{o}\}}$ in phase 6 is formulated as $T^{(l)}_{\{bp,\hat{o}\}} = \max_{M_j \in M \setminus M_{\hat{o}}} \Big\{t^{(l)}_{\{bp,j\}}\Big\}$ under frequency division multiple access. The term $\Big\{t^{(l)}_{\{bp,j\}}\Big\} = (h = \delta_m N_V) / \Big[W_m \log_2(1+\lambda_{\hat{o}j})\Big]$ represents the block sharing delay from the mining winner $M_{\hat{o}}$ to $M_j \in M \setminus M_{\hat{o}}$, and $\lambda_{\hat{o}j}$ is received SNR from the miner $M_{\hat{o}}$ to the miner M_j.

The last considered delay is the block generation delay in phase 5. The miner $M_j \in M$'s block generation delay $T^{(l)}_{\{bg,j}$ complies with an exponential distribution with a mean value of $1/\lambda$. The block generation delay $T^{(l)}_{\{bg,\hat{o}}$ of the mining winner $M_{\hat{o}}$ is given as the delay of interest. Based on this, the latency $T^{(L)}_o$ of the l-th epoch is described as

$$T^{(l)}_o = N^{(l)}_{folk}\Big(T^{(l)}_{\{local,o\}} + T^{(l)}_{\{up,o\}} + T^{(l)}_{\{cross,o\}} + T^{(l)}_{\{bg,\hat{o}\}} + T^{(l)}_{\{bp,\hat{o}\}}\Big) + T^{(l)}_{\{vn,o\}} + T^{(l)}_{\{global,o\}}, \tag{9}$$

in which the $N^{(l)}_{folk}$ represents forking occurrences' number in the l-th epoch. The forking occurrences complies with a geometric distribution where mean is $1/(1-p^{(l)}_{folk})$ and the forking probability is $p^{(l)}_{folk}$. Extended from phase 6, the forking probability is defined as

$$p^{(l)}_{folk} = 1 - \prod_{M_j \in M \setminus M_{\hat{o}}} \Pr\Big(t^{(l)}_j - t^{(l)}_{\hat{o}} > t^{(l)}_{bp,j}\Big), \tag{10}$$

where the term $t^{(l)}_j = T^{(l)}_{\{local,j\}} + T^{(l)}_{\{up,j\}} + T^{(l)}_{\{cross,j\}} + T^{(l)}_{\{bg,j\}}$ is the cumulated delay until the miner M_j generates a block.

5.3 Optimal Generation Rate of Blocks

In this work, we target on deriving the optimal block generation rate λ^*, which can minimize the latency averaged over the consensus algorithm of the end device V_O's l-th epoch by means of the single-epoch latency expression in Eq. 9. The consensus process has an impact on three indexes, including the block sharing delay $T^{(l)}_{\{bp,\hat{o}\}}$, the block generation delay $T^{(l)}_{\{bg,\hat{o}\}}$, and the number $N^{(l)}_{folk}$ fork of forking occurrences. These three entities are inter-dependent because of the existence of mining winner M_o. Solving this requires to compare the cumulated delays for all miners and their associated end devicesfog server under their asynchronous operations that complicate the optimization.

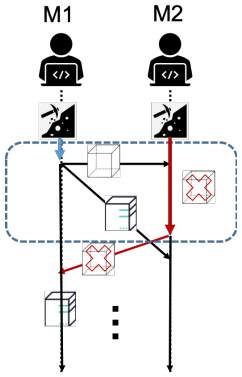


Fig. 4: Single-Epoch Operation with Forking

To eliminate the above mentioned difficulty, all miners are considered to start their consensus processes synchronously by modifying T_{wait}, with which we further derive $T^{(l)}_{\{cross,o\}} = T_{wait} - (T^{(l)}_{\{local,o\}} + T^{(l)}_{\{up,o\}})$. Under these circumstances, if the miners complete the cross verification phase earlier, they are supposed to wait until T_{wait}, which provides lower bound of the performance, for instance, latency upper bound. Through exact operations using the synchronous approximation, the optimal block generation rate λ^* could be derived in a closed-form as follows.

Proposition 1: With the PoW synchronous approximation, such as $T^{(l)}_{\{cross,o\}} = T_{wait} - (T^{(l)}_{\{local,o\}} + T^{(l)}_{\{up,o\}})$, the block generation rate λ^* minimizing the l-th epoch latency $E[t^{(l)}_o]$ averaged over the PoW process is given by

$$\lambda^* \approx 2 \times \left(T^{(l)}_{\{bp,\hat{o}\}}\left[1 + \sqrt{1 + 4N_M \times \Big(1 + T_{wait}/T^{(l)}_{\{bp,\hat{o}\}}\Big)}\right]\right). \tag{11}$$

Next, we apply both the mean $1/(1-p^{(l)}_{folk})$ of the geometrically distributed $N^{(l)}_{folk}$ and the synchronous consensus approximation to Eq. 9 and thus have

$$E[t^{(l)}_o] \approx \Big(T_{wait} + E[T^{(l)}_{\{bg,\hat{o}\}}]\Big) \Big/ \Big(1/(1-p^{(l)}_{folk})\Big) + T^{(l)}_{\{vn,o\}} + T^{(l)}_{\{global,o\}} \tag{12}$$

There are some constant delays, including T_{wait}, $T^{(l)}_{\{vn,o\}}$, and $T^{(l)}_{\{global,o\}}$, which are given in Section 3.1. Based on these, we further derive the remainder as below.

For the probability $p^{(l)}_{folk}$, using Eq. 10 with $t^{(l)}_j - t^{(l)}_{\hat{o}} = t^{(l)}_{\{bg,j\}} - t^{(l)}_{\{bg,\hat{o}\}}$ under the synchronous approximation, we obtain the fork as

$$p^{(l)}_{folk} = 1 - \mathrm{EXP}\Big(\lambda \sum_{M_j \in M \setminus M_{\hat{o}}} T^{(l)}_{\{bp,j\}}\Big), \tag{13}$$

where $T^{(l)}_{\{bp,j\}}$ is a given constant delay. Then, by jointly using $T^{(l)}_{\{bg,\hat{o}\}}$ and the complementary cumulative distribution function (CCDF) of $T^{(l)}_{\{bg,j}$ which is the exponentially distributed, CCDF of $T^{(l)}_{\{bg,\hat{o}\}}$ is formulated as

$$\Pr\left(T^{(l)}_{\{bg,\hat{o}\}} > x\right) = \prod_{j=1}^{N_M} \Pr\left(T^{(l)}_{\{bg,j\}} > x\right) = \mathrm{EXP}\left(-\lambda N_M x\right) \tag{14}$$

Moreover, we apply the total probability theorem, which yields $E[T^{(l)}_{\{bg,\hat{o}\}}] = 1/(\lambda N_M)$. The final step is to combine all these terms and re-formulize Eq. 15 as

$$E[t^{(l)}_o] \approx \left(T_{wait} + 1/(\lambda N_M)\right)\mathrm{EXP}\left(\lambda \sum_{M_j \in M \setminus M_{\hat{o}}} T^{(l)}_{\{bp,j\}}\right) + T^{(l)}_{\{vn,o\}} + T^{(l)}_{\{global,o\}}, \tag{15}$$

which is convex with respect to λ. Therefore, the optimal block generation rate λ^* is derived from the first-order necessary condition.

The accuracy of the above result with the synchronous approximation is validated by comparing the simulated λ^* without the approximation in the following section.

TABLE 1: Simulation Parameters Overview

Item	Value / Description
LTE network	ITU-R M.2135-1 Micro NLOS
Transmission power	30 dBm
Antenna gain	0 dBm
Antenna heights of the fog servers	11 m
Antenna heights of the end devices	1 m
Bandwidth	1.8 MHz
Time slot	0.5 ms
Δ	1.6
ρ_{max}	4.8
Mean throughputs	1.4Mb/s
Maximum throughputs	8.6Mb/s

6 Performance Evaluation

In this section, we testify the superiority of the proposed FL-Block model in term of privacy protection and efficiency. We simulate a fog computing environment and create a blockchain network. Based on this, we conduct experiments on machine learning tasks using real-world data sets of large scale.

6.1 Simulation Environment Description

The simulated fog computing environment is implemented on the cellular network of an urban microcell. It is consisted of 3 fog servers, and $N_v = 50$ end devices, on a single workstation. The three fog servers are located at the center of the cell with a radius of 3 kilometres, while the end devices are distributed with a normal distribution in this cell.

To model the wireless communications, we introduce the LTE networks with a popular urban channel model, which is defined in the ITU-R M.2135-1 Micro NLOS model of a hexagonal cell layout [40]. The transmission power and antenna gain of the fog server and end devices are set to be 30 dBm and 0 dBi, respectively, for simplicity. Carrier frequency is set to 2.5 GHz, and the antenna heights of the fog servers and end devices are set to 11 m and 1 m, correspondingly. We use a practical bandwidth limitation, namely, 20 RBs, which corresponds to a bandwidth of 1.8 MHz. This is assigned to an end device in each time slot of 0.5 ms. We employ a throughput model based on the Shannon capacity with a certain loss used in [41], in which Δ = 1.6 and ρ_{max} = 4.8. With these initialized settings, the mean and maximum throughputs of client θ_k are 1.4 Mb/s and 8.6 Mb/s, which are practical and feasible in LTE networks. The details of the simulation results are summarized in Table. 1 for clarity.

From the aforementioned model, we obtain the throughput and consider it as the average throughput. This throughput is leveraged to derive t in the end device selection phase. We assume the network environment is stable and thereby the federated learning process will not be impacted by network connectivity issues. In this way, we can regard the average throughput as stable outputs. To better simulate the real-world scenarios, we take a small variation of short-term throughput at the scheduled update and upload into consideration. The throughput will be sampled from the Gaussian distribution when the models' updates are shared between end devices and fog servers. The Gaussian distribution is defined by the average throughput and its $r\%$ value as the mean and standard deviation, respectively.

6.2 Global Models and Corresponding Updates

In order to guarantee the output's accuracy, the **IID** setting is considered, in which each end device will randomly sample a specific amount of data from the raw dataset.

We implement a classic CNN as the global model for all tasks. In particular, FL-Block contains six 3*3 convolution layers, including 2*32, 2*64, and 2*128 channels. In each of the channel, it is activated by ReLU and batch normalized, while every pair of them is followed by 2*2 max pooling. In additions, the six channels are followed by three fully-connected layers, including 382 and 192 units with ReLU activation and another 10 units activated by soft-max. In this case, the model will approximately have 3.6 million parameters for Fashion-MINIST and 4.6 million parameters for CIFAR-10. The size of D_m are 14.4 mb and 18.3 mb, respectively while the data type is set to be 32-bit float. Although some other deep models will have better performances, they are not the focus of this model and is not considered.

The hyper-parameters of updating global models are initialised as follows. The mini-batch size is 40. The number of epochs in each round is 40. The initial learning rate of stochastic gradient descent updates is 0.30. The learning rate decay is 0.95. We simply model the computation capability of each end device as how many data samples it could train to further update the global model. There

might be slight fluctuation due to some other tasks on this end device. We randomly decide the average capability of each end device from the interval $[10, 100]$, which will be used in the client selection phase. Consequently, the update time in client selection phase averagely ranges from 5 to 500 seconds. The computation capability depends on the Gaussian distribution in both the scheduled update phase and the upload phase. The Gaussian distribution is defined by the average throughput and its $r\%$ value as the mean and standard deviation, respectively. The range is considered to be reasonable since the workstation needs 4 seconds considering one single update with one single GPU. They may require longer update time up to 100 times if the mobile devices have a weaker computation power. Empirically, we set T_{final} to 400 seconds and T_{round} to 3 seconds.

TABLE 2: Evaluation Results of CIFAR-10

Indexes	ToA(0.5)	ToA(0.75)	Accuracy
T_{round} = 3 sec (r = 0%)	24.5	122.3	0.76
T_{round} = 3 sec (r = 10%)	26.8	136.1	0.74
T_{round} = 3 sec (r = 20%)	30.2	182.5	0.72
T_{round} = 1 sec (r = 0%)	NAN	NAN	0.50
T_{round}= 5 sec (r = 0%)	45.1	178.5	0.77
T_{round} = 10 sec (r = 0%)	80.7	312.5	0.75

ToA(x) (in seconds) is the time costed to reach a specific testing classification accuracy of x. The performance upgrades with the decrease of ToA.
Accuracy: denotes the accuracy value for the final iteration.

TABLE 3: Evaluation Results of Fashion-MINIST

Indexes	ToA(0.5)	ToA(0.75)	Accuracy
T_{round} = 3 sec (r = 0%)	15.6	39.3	0.89
T_{round} = 3 sec (r = 10%)	16.6	40.2	0.88
T_{round} = 3 sec (r = 20%)	17.3	45.5	0.91
T_{round} = 1 sec (r = 0%)	5.4	86.4	0.87
T_{round}= 5 sec (r = 0%)	25.1	60.7	0.92
T_{round} = 10 sec (r = 0%)	58.2	110.8	0.93

ToA(x) (in seconds) is the time costed to reach a specific testing classification accuracy of x. The performance upgrades with the decrease of ToA.
Accuracy: denotes the accuracy value for the final iteration.

In Table 2 and 3, we illustrate the evaluation results with the *IID* settings in terms of ToA and accuracy. Each method is executed for 10 times and the ToA and accuracy scores are measured from the averaging perspective. In term of ToA, FL-Block has excellent performances on both tasks. The architecture is sufficient to show the efficiency of the newly-designed protocols under resource-limited settings. However, the best accuracy is not the target in FL-Block. The original federated learning without deadline limitations achieved accuracies of 0.80 and 0.92 for CIFAR-10 and Fashion-MNIST, respectively. The obtained accuracies are comparable to the performances of FL-Block. Two traditional major concerns, which are the uncertainty of throughput and computation capabilities, do not have a significant impact on the FL-Block's performances.

From the aspect of the impact of T_{round}, we evaluate the changes in the classification accuracy and ToA. As shown in Fig. 5. FL-Block and Table 3, FL-Block on Fashion MNIST for different values of deadline T_{round} while maintaining T_{final} constant. It is observable that the value of T_{round} should be in a proper range without being too large or too small. Because of the smaller number of aggregation phases, longer deadlines like 20 seconds with FL-Block involved numerous end devices in each round, which leads to extremely limited performances. From the other angle, if we set a short deadline like 1 second to limit the number of end devices accessible in each round, the classification performances are also degraded in an unexpected way. Therefore, a possible promising method of selecting T_{round} would be dynamically changing it to align an abundant amount of end devices in each iteration.

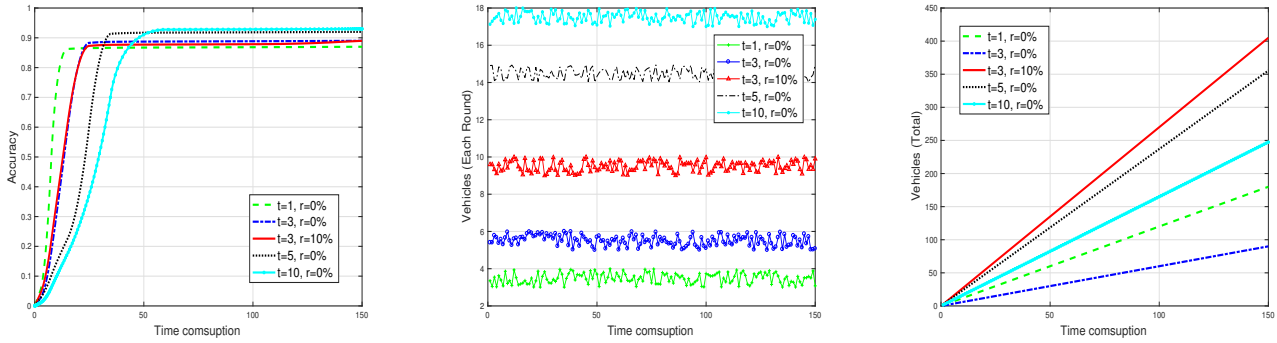
6.3 Evaluation on Convergence and Efficiency

From the perspective of convergence and efficiency, FL-Block can achieve convergence at different learning rates. Intuitively, greater learning rate results in faster convergence, which is also testified with the evaluation results. As illustrated in Fig. 6, FL-Block reaches convergences when data size is moderate, which means the scalability is excellent. Even if the data size is great, the model can still achieve fast convergence. The convergence value of FL-Block is 0.69 when the learning rate is 0.6. It is significant superior comparing to the other two ones, which converges at 0.51 and 0.41, respectively. Therefore, FL-Block fully meets the requirements of big data scenarios, in particular fog computing applications.

6.4 Evaluation on Blockchain

In Fig. 7, we show how the block generation rate λ influences the average learning completion latency of FL-Block. We can tell that the latency is convex-shaped over the generation rate λ and simultaneously decreasing with the SNRs in Fig. 7a. From Fig. 7b, we show the minimized average learning completion latency time based on the optimal block generation rate λ^*. The value of latency time derived from the proposition is greater than the simulated minimum latency by up to 2.0% .

Fig. 8 illustrates the FL-Block's scalability in terms of the numbers N_M and N_V of miners and end devices, respectively. In Fig. 8a, the average learning completion latency is computed for N_M = 5 and N_M = 10 with and without the malfunction of miners. To capture the malfunction, Gaussian noise complying with $N \sim (-0.1; 0.1)$ is added to the aggregated local model updates of each miner with a probability of 0.5. In the scenario of without malfunction, the latency increases with the increase of N_M because of the resource consumption in block propagation delays and cross-verification operation. However, this is not always the case when there is malfunction involved. In FL-Block,



(a) Accuracy (b) end devices (Each Round) (c) end devices (Total)

Fig. 5: Effects of Different Values of Deadline T_{round}

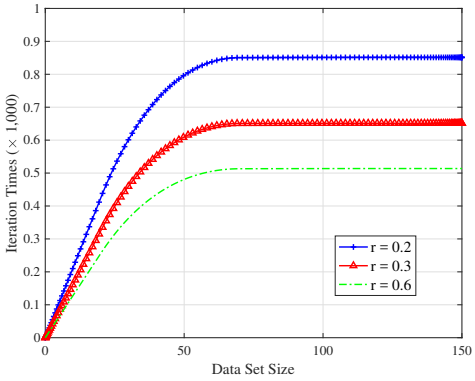
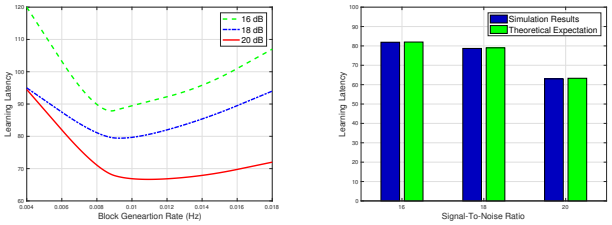
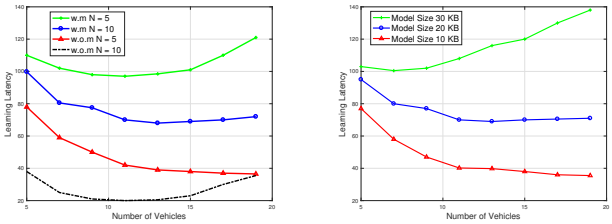


Fig. 6: Iteration Convergence v.s. Increase of Data Size



(a) λ (b) λ^*

Fig. 7: Block Generation



(a) w.m: With Malfunction; w.o.m: Without Malfunction (b) Model Size

Fig. 8: Learning Latency

the malfunction of each miner only impacts on the global model update of the corresponding end device. This kind of distortion could be eliminated by federating with other non-tempering end devices and miners. On this account, the increase of N_M may result in a shorter latency, which is testified in the provided evaluation results, for instance, N_M = 20 with malfunction.

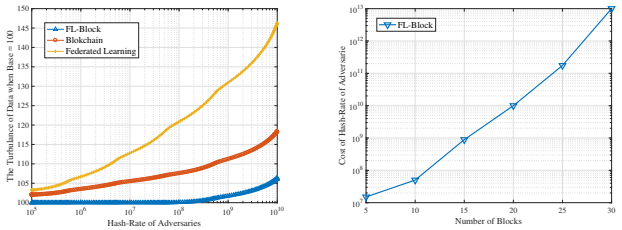
With Fig. 8a, we show that a latency-optimal number N_V of end devices are identified. A greater value of N_V enables the usage of a larger number of data samples. However, this will increase the size of each block as well as the delays of block exchange, which brings about the aforementioned convex-shaped latency. It increases each block size such as communication payload. Therefore, it leads to higher block exchange delays and consequently resulting in the convex-shaped latency concerning N_V. That is why a reasonable end device selection can potentially reduce the expected latency.

Lastly, Fig. 8b shows that the latency increases with local model size δ_m of each end device, which is intuitive but vital. For this reason, it calls for novel model compression mechanisms, which will also be part of our future work.

6.5 Evaluation on Poisoning Attack Resistance

In FL-Block, the poisoning attack could be resisted in a high-confidence way because of the nature of blockchain and the enhanced protocols we propose. In this simulation, we compare the performance of FL-Block with both classic blockchain and federated learning. The assumption here is the initialization phase of the proposed blockchain network where blocks and miners are small enough to allow these attacks to breakthrough Blockchain and thereby poisoning attack is possible to manipulate the data in a malicious way.

In Fig.9a, we use semi-logarithmic coordinate system to compare the performances of the three models. We can tell that with the increase of the adversary's hash-rate, more turbulence there will be for all three models. However, only when the hash-rate reaches a threshold, the adversary can launch a poisoning attack to the data under the protection of Blockchain. This only happens when the number of block is 10 in this simulation. If there are more blocks, it takes much more time as the time consumption increases in an exponential manner.



(a) Comparison among Three Models (b) Successful Attack Requirements based on Number of Blocks.

Fig. 9: Poisoning Attack Resistance Performance (a) Comparizon among Three Models, (b) Successful Attack Requirements based on Number of Blocks.

Fig.9b shows how much an adversary needs to breakthrough the protection with regards to different blocks. It looks like in a linearly tendency as the y axis is presented by a semi-logarithmic axis. That means the hash-rate increases in an exponential manner as well. It requires $10^{13}+$ when the block reaches 30. As the difficulty of generating a block increases over time, it provides higher and higher protection when a blockchain develops.

7 Summary and Future work

In this paper, we propose a novel blockchain-enabled federated learning model to solve the identified issues in fog computing. end devices upload the local updates to the fog servers, where the global updates will be generated and stored. Since only the pointer of the global updates is saved on-chain while a distributed hash table (DHT) is used to save to data, the block generation efficiency could be guaranteed. With a hybrid identity generation, comprehensive verification, access control, and off-chain data storage and retrieve, FL-Block enables decentralized privacy protection while preventing single point failure. Moreover, the poisoning attack could be eliminated from the aspect of fog servers. Extensive evaluation results on real-world datasets are presented to show the superiority of FL-Block.

For future work, we plan to further extend this model to more generalized scenarios by optimizing the trade-off between privacy protection and efficiency. In addition, we prepare to use game theory and Markov decision process to identify the optimal conditions in terms of computation and communication costs.

References

[1] C. Mouradian, D. Naboulsi, S. Yangui, R. H. Glitho, M. J. Morrow, and P. A. Polakos, "A comprehensive survey on fog computing: State-of-the-art and research challenges," *IEEE Communications Surveys and Tutorials*, vol. 20, no. 1, pp. 416–464, 2018.

[2] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *CoRR*, vol. abs/1807.08127, 2018.

[3] Y. Guan, J. Shao, G. Wei, and M. Xie, "Data security and privacy in fog computing," *IEEE Network*, vol. 32, no. 5, pp. 106–111, 2018.

[4] L. Ma, X. Liu, Q. Pei, and Y. Xiang, "Privacy-preserving reputation management for edge computing enhanced mobile crowdsensing," *IEEE Transactions on Services Computing*, vol. 12, no. 5, pp. 786–799, 2018.

[5] M. A. Ferrag, L. A. Maglaras, and A. Ahmim, "Privacy-preserving schemes for ad hoc social networks: A survey," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 4, pp. 3015–3045, 2017.

[6] Y. Qu, S. Yu, L. Gao, W. Zhou, and S. Peng, "A hybrid privacy protection scheme in cyber-physical social networks," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 3, pp. 773–784, 2018.

[7] Y. Qu, S. Yu, W. Zhou, S. Peng, G. Wang, and K. Xiao, "Privacy of things: Emerging challenges and opportunities in wireless internet of things," *IEEE Wireless Commun.*, vol. 25, no. 6, pp. 91–97, 2018.

[8] H. Zhu, F. Wang, R. Lu, F. Liu, G. Fu, and H. Li, "Efficient and privacy-preserving proximity detection schemes for social applications," *IEEE Internet of Things Journal*, 2017.

[9] M. M. E. A. Mahmoud, N. Saputro, P. Akula, and K. Akkaya, "Privacy-preserving power injection over a hybrid AMI/LTE smart grid network," *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 870–880, 2017.

[10] J. Hu, L. Yang, and L. Hanzo, "Energy-efficient cross-layer design of wireless mesh networks for content sharing in online social networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 9, pp. 8495–8509, 2017.

[11] C. Dwork, "Differential privacy," in *in Proceedings of ICALP 2006, Venice, Italy, July 10-14,*, 2006, pp. 1–12.

[12] L. Lyu, K. Nandakumar, B. I. P. Rubinstein, J. Jin, J. Bedo, and M. Palaniswami, "PPFA: privacy preserving fog-enabled aggregation in smart grid," *IEEE Trans. Industrial Informatics*, vol. 14, no. 8, pp. 3733–3744, 2018.

[13] R. Lu, X. Lin, H. Zhu, P. Ho, and X. Shen, "ECPP: efficient conditional privacy preservation protocol for secure vehicular communications," in *INFOCOM 2008. 27th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 13-18 April 2008, Phoenix, AZ, USA*, 2008, pp. 1229–1237.

[14] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," *Google Research Blog*, vol. 3, 2017.

[15] H. Kim, J. Park, M. Bennis, and S. Kim, "On-device federated learning via blockchain and its latency analysis," *CoRR*, vol. abs/1808.03949, 2018.

[16] J. Ni, K. Zhang, Y. Yu, X. Lin, and X. S. Shen, "Providing task allocation and secure deduplication for mobile crowdsensing via fog computing," *IEEE Transactions on Dependable and Secure Computing*, 2018.

[17] P. Hu, H. Ning, T. Qiu, H. Song, Y. Wang, and X. Yao, "Security and privacy preservation scheme of face identification and resolution framework using fog computing in internet of things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1143–1155, 2017.

[18] D. X. Song, D. A. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *2000 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 14-17, 2000*, 2000, pp. 44–55.

[19] R. Lu, X. Lin, T. H. Luan, X. Liang, and X. S. Shen, "Pseudonym changing at social spots: An effective strategy for location privacy in vanets," *IEEE Trans. Vehicular Technology*, vol. 61, no. 1, pp. 86–96, 2012.

[20] H. Li, D. Liu, Y. Dai, T. H. Luan, and S. Yu, "Personalized search over encrypted data with efficient and secure updates in mobile clouds," *IEEE Trans. Emerging Topics Comput.*, vol. 6, no. 1, pp. 97–109, 2018.

[21] J. Kang, R. Yu, X. Huang, and Y. Zhang, "Privacy-preserved pseudonym scheme for fog computing supported internet of vehicles," *IEEE Trans. Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2627–2637, 2018.

[22] H. Tan, D. Choi, P. Kim, S. B. Pan, and I. Chung, "Secure certificateless authentication and road message dissemination protocol in vanets," *Wireless Communications and Mobile Computing*, vol. 2018, pp. 7 978 027:1–7 978 027:13, 2018.

[23] M. Wazid, A. K. Das, V. Odelu, N. Kumar, M. Conti, and M. Jo, "Design of secure user authenticated key management protocol for generic iot networks," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 269–282, 2018.

[24] B. S. Gu, L. Gao, X. Wang, Y. Qu, J. Jin, and S. Yu, "Privacy on the edge: Customizable privacy-preserving context sharing in hierarchical edge computing," *IEEE Transactions on Network Science and Engineering*, 2019.

[25] H. Li, R. Lu, J. V. Misic, and M. M. E. A. Mahmoud, "Security and

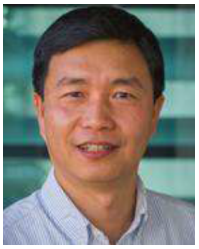
privacy of connected vehicular cloud computing," *IEEE Network*, vol. 32, no. 3, pp. 4–6, 2018.
[26] L. Ma, Q. Pei, Y. Qu, K. Fan, and X. Lai, "Decentralized privacy-preserving reputation management for mobile crowdsensing," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2019, pp. 532–548.
[27] Y. Jiao, P. Wang, D. Niyato, and K. Suankaewmanee, "Auction mechanisms in cloud/fog computing resource allocation for public blockchain networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 9, pp. 1975–1989, 2019.
[28] S. Rowan, M. Clear, M. Gerla, M. Huggard, and C. M. Goldrick, "Securing vehicle to vehicle communications using blockchain through visible light and acoustic side-channels," *CoRR*, vol. abs/1704.02553, 2017.
[29] N. Ruan, M. Li, and J. Li, "A novel broadcast authentication protocol for internet of vehicles," *Peer-to-Peer Networking and Applications*, vol. 10, no. 6, pp. 1331–1343, 2017.
[30] Z. Lu, W. Liu, Q. Wang, G. Qu, and Z. Liu, "A privacy-preserving trust model based on blockchain for vanets," *IEEE Access*, vol. 6, pp. 45 655–45 664, 2018.
[31] N. Malik, P. Nanda, A. Arora, X. He, and D. Puthal, "Blockchain based secured identity authentication and expeditious revocation framework for vehicular networks," in *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications / 12th IEEE International Conference On Big Data Science And Engineering, TrustCom/BigDataSE 2018, New York, NY, USA, August 1-3, 2018*, 2018, pp. 674–679.
[32] M. Bernardini, D. Pennino, and M. Pizzonia, "Blockchains meet distributed hash tables: Decoupling validation from state storage," *arXiv preprint arXiv:1904.01935*, 2019.
[33] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, 2019.
[34] V. Smith, C. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017, pp. 4424–4434.
[35] J. Konecný, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *CoRR*, vol. abs/1610.02527, 2016.
[36] J. Konecný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *CoRR*, vol. abs/1610.05492, 2016.
[37] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *2019 IEEE International Conference on Communications, ICC 2019, Shanghai, China, May 20-24, 2019*, 2019, pp. 1–7.
[38] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial iot," *IEEE Transactions on Industrial Informatics*, 2019.
[39] X. An, X. Zhou, X. Lü, F. Lin, and L. Yang, "Sample selected extreme learning machine based intrusion detection in fog computing and MEC," *Wireless Communications and Mobile Computing*, vol. 2018, 2018.
[40] M. Series, "Guidelines for evaluation of radio interface technologies for imt-advanced," *Report ITU*, vol. 638, 2009.
[41] M. R. Akdeniz, Y. Liu, M. K. Samimi, S. Sun, S. Rangan, T. S. Rappaport, and E. Erkip, "Millimeter wave channel modeling and cellular capacity evaluation," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1164–1179, 2014.

Youyang Qu received his B.S. degree of Mechanical Automation in 2002 and M.S. degree of Software Engineering in 2015 from Beijing Institute of Technology, respectively. He is currently pursuing the Ph.D. degree at School of Information Technology, Deakin University. His research interests focus on dealing with security and customizable privacy issues in Social Networks, Machine Learning, IoT, and Big Data. He is active in communication society and has served as a TPC Member for IEEE flagship conferences including IEEE ICC and IEEE Globecom.

Longxiang Gao received a Ph.D. in Computer Science from Deakin University, Australia. He is currently a Senior Lecturer at the School of Information Technology, Deakin University. Before joining Deakin University, he was a post-doctoral research fellow at IBM Research and Development Australia. His research interests include data processing, mobile social networks, Fog computing and network security.

Dr. Gao has over 40 publications, including patents, monographs, book chapters, and journal and conference papers. Some of his publications have been published in the top venues, such as IEEE TMC, IEEE IoT, IEEE TDSC and IEEE TVT. He received the 2012 Chinese Government Award for Outstanding Students Abroad (Ranked No.1 in Victoria and Tasmania consular districts). Dr. Gao is a Senior Member of IEEE and is active in IEEE Communication Society. He has served as the TPC co-chair, publicity co-chair, organization chair and TPC member for many international conferences.

Tom H. Luan received the B.E. degree from the Xi'an Jiaotong University, China in 2004, MPhil. degree from the Hong Kong University of Science and Technology, and Ph.D. degree from the University of Waterloo, all in Electrical and Computer Engineering. From 2013 to 2017, he was with the Deakin University, Australia, as a Lecturer in Mobile and Apps. Since 2017, he is with the School of Cyber Engineering at the Xidian University as a Professor. Dr. Luan's research interests are on edge computing, vehicular networks, wireless multimedia, mobile cloud computing.



Yong Xiang received his B.E. and M.E. degrees from the University of Electronic Science and Technology of China, China, and PhD degree from The University of Melbourne, Australia. He is a Professor at the School of Information Technology, Deakin University, Australia. He is also the Associate Head of School (Research) and the Director of the Artificial Intelligence and Data Analytics Research Cluster. He has obtained a number of research grants (including several ARC Discovery and Linkage grants from the Australian Research Council) and published numerous research papers in high-quality international journals and conferences. He is the coinventor of two U.S. patents and some of his research results have been commercialised. Dr Xiang is the Editor/Guest Editor of several international journals. He has been invited to give keynote speeches and chair committees in a number of international conferences, review papers for many international journals and conferences, serve on conference program committees, and chair technical sessions in conferences. Dr. Xiang is a senior member of the IEEE.

Shui Yu is currently a full Professor of School of Computer Science, University of Technology Sydney, Australia. Dr Yu's research interest includes Security and Privacy, Networking, Big Data, and Mathematical Modelling. He has published two monographs and edited two books, more than 200 technical papers, including top journals and top conferences, such as IEEE TPDS, TC, TIFS, TMC, TKDE, TETC, ToN, and INFOCOM. Dr Yu initiated the research field of networking for big data in 2013. His h-index is 41. Dr Yu actively serves his research communities in various roles. He is currently serving the editorial boards of IEEE Communications Surveys and Tutorials, IEEE Communications Magazine, IEEE Internet of Things Journal, IEEE Communications Letters, IEEE Access, and IEEE Transactions on Computational Social Systems. He has served many international conferences as a member of organizing committee, such as publication chair for IEEE Globecom 2015, IEEE INFOCOM 2016 and 2017, TPC chair for IEEE BigDataService 2015, and general chair for ACSW 2017. Dr Yu is a final voting member for a few NSF China programs in 2017. He is a Senior Member of IEEE, a member of AAAS and ACM, the Vice Chair of Technical Committee on Big Data of IEEE Communication Society, and a Distinguished Lecturer of IEEE Communication Society.

Bai Li received his PhD in Computer Science from Deakin University, Australia in 2012. He is currently the CTO of the Health Chain Technology Pty. Ltd. Dr Li has been working in IT industry for more than 10 years. In his career, he worked on many cutting edge technologies in fields like AR, IT security, big data, AI, industry IoT and Blockchain. Dr Li leads a team focusing on Blockchain R&D in healthcare and IoT sectors at Health Chain Technologies Pty. Ltd. Dr. Li has more than 10 publications, including journal and conference paper and book chapter. Some of his publications have been published the top journals.

Gavin Zheng completed his BSc and MSc in Huazhong University of Science and Technology. He also got a MIS from McGill University and a MBA from Rotman school, UofT. Currently, he is the chief scientist of Nenglian Technology – A blockchain solution provider based in Beijing, China. Now, he leads the team of Echoin which is a public blockchain project aiming to build aN ecology system for energy industry. His main research interests are blockchain related area: VM, Smart Contract, decentralized storage, Zero Knowledge proof. He is also active in quantitative financing and Rust programming language.