This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2023.3343839

An Enhanced Block Validation Framework with Efficient Consensus for Secure Consortium Blockchains

Weiquan Ni, Alia Asheralieva, Jiawen Kang, Zehui Xiong, *Member, IEEE*, Carsten Maple*, *Senior Member, IEEE* and Xuetao Wei, *Member, IEEE*

Abstract—Consortium blockchains have attracted considerable interest from academia and industry due to their low-cost installation and maintenance. However, typical consortium blockchains can be easily attacked by colluding block validators because of the limited number of miners in the systems. To address this problem, in this paper, we propose a novel block validation framework to enhance blockchain security. In the framework, the block validations are assisted and implemented by various lightweight nodes, e.g., edge devices, in addition to the typical blockchain miners. This improves the blockchain security but can cause an increased block validation delay and, thereby, reduced blockchain throughput. To tackle this challenge, we propose an effective method to select lightweight nodes based on their computing powers to maximize the blockchain throughput, and prove the uniqueness of the optimal nodes selection strategy. Security analysis and simulation results from the deployed consortium blockchain platform show that the proposed framework achieves higher throughput and security than the existing consortium blockchain models.

Index Terms—Block validation, consortium blockchain, lightweight node, security, throughput

1 Introduction

Due to enhanced security, scalability, and decentralization, consortium blockchains are witnessing increasing popularity in many distributed scenarios, e.g., mobile healthcare [1], Industrial Internet of Things (IIoT) [2], and vehicular networks [3]. In contrast to public blockchain models, consortium blockchains have a lower cost of deployment and maintenance while offering higher throughput. In practice, a consortium blockchain is maintained by a small number of federated organizational members acting as miners to process requests from users. This can reduce the overhead of resource-constrained users to engage in block mining but excludes them from accessing the blockchain data. In blockchains, various data, e.g., digital transactions, are recorded in the form of blocks as a chained list to store logical information relationships in the appended data. The process of blockchain mining consists of two stages: i) the block generation stage, during which the data is recorded into a new block; ii) the block validation stage, during which the block is exchanged and verified by miners through the application of consensus protocol, such as Practical Byzantine Fault Tolerance (PBFT) [4], Proof-of-Stake (PoS) [5] or Delegated PoS (DPoS) [6].

To be added to the blockchain, a block must receive positive validation results from the majority of miners verifying this block [4]. However, the number of miners in consortium blockchains is limited; for example, 148 and 140 miners are currently running on the popular platforms of Hyperledger Fabric [7] and Fisco Bcos [8], respectively. These systems are vulnerable to collusion among miners to accept or reject a block falsely, thereby compromising the blockchain security [9]. A common approach to mitigate the impact of colluding miners is to recruit lightweight nodes, such as edge devices, to perform block validations together with the miners [10]. In this regard, a number of prior works (e.g., [9] - [14]) have shown that such an approach can significantly enhance blockchain security through the appropriate selection of lightweight nodes. For example, the authors in [11] and [12] focused on incentive-provisioning mechanisms for lightweight nodes engaging in block validations. The authors in [13] devised a voting-based consensus algorithm called Rift, enabling each user to be a block producer or validator. The authors in [9] and [14] presented a decentralized protocol that allows lightweight nodes to verify blocks without relying on miners.

Unfortunately, the recruitment of lightweight nodes can greatly increase the block validation delay for broadcasting and verifying a block. One of the reasons is that, in general, the computing powers of lightweight nodes are highly heterogeneous and can be considerably lower than those of the miners. As a result, the rate of block validation can be significantly reduced by adding slow, resource-constrained block validators. In addition, blockchain security cannot be enhanced if the lightweight nodes are unable to return their validation results in time. To tackle this issue, the appropriate selection of lightweight nodes (e.g., based on their computing powers) is of prime importance. However,

- *W. Ni is with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China[1], and the WMG, University of Warwick, Coventry, CV4 7AL, UK[2] (e-mail:Weiquan.Ni@warwick.ac.uk).*
- *A. Asheralieva is with the Department of Computer Science, Loughborough University, Loughborough, LE11 3TU, UK (e-mail:aasheralieva@gmail.com).*
- *J. Kang is with the School of Automation, Guangdong University of Technology, Guangzhou 510006, Guangdong (e-mail:kjwx886@163.com).*
- *Z. Xiong is with the Pillar of Information Systems Technology and Design, Singapore University of Technology and Design, Singapore 487372, Singapore (e-mail:zehui_xiong@sutd.edu.sg).*
- *C. Maple (*Corresponding Author) is with the WMG, University of Warwick, Coventry, CV4 7AL, UK (e-mail:CM@warwick.ac.uk).*
- *X. Wei is with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China (e-mail:weixt@sustech.edu.cn).*

related works (e.g., [9] - [14]) did not consider the straggling of lightweight nodes and failed to address how to reduce the block validation delay, which consequently increases the blockchain throughput. For example, the works in [9] and [14] designed the protocol for collaborative block validation, while the works in [11] and [12] focused only on optimising recruitment efficiency and block broadcast delay, respectively, without analysing the total block validation delay and resulting blockchain throughput.

Accordingly, in this paper, we consider the problem of the selection of lightweight nodes with the aim of reducing the block validation delay and, thus, increasing the blockchain throughput while enhancing the blockchain security. The main contributions of this paper are as follows:

- Unlike previous works focusing on block generations, we study the throughput and security of consortium blockchains during block validations, and propose a general secure block validation framework involving lightweight nodes acting as lightweight block validators.
- To evaluate the security of the proposed framework, we comprehensively analyse the potential attacks and threats based on the threat analysis model - STRIDE. In addition, we provide countermeasures against each identified attack and threat with security levels assessment. Through quantitative evaluation, the experiment results show that our framework achieves higher security than the existing schemes.
- To improve blockchain throughputs, we formulate the optimization problem where the system utility (proportional to the throughput) is maximized by considering the limited computing powers of miners and lightweight nodes with absolute-finality consensus protocols [15] used for block validations. We prove (analytically) that the formulated optimization problem admits a unique solution, based on which we derive the optimal strategy to select lightweight nodes efficiently.
- We deploy a practical consortium blockchain system based on the kernel of the Fisco Bcos platform [8]. By applying the computing power management scheme and the strategy to select lightweight nodes, our approach achieves higher throughput than the existing schemes. Meanwhile, the selection strategy contributes to less redundant computing power of selected lightweight nodes while protecting their privacy. In addition, the experiments show that our approach can provide services to different kinds of applications.

This paper is organized as follows. Section 2 presents the background and related work. The block validation framework is detailed in Section 3. We compute the block validation delay in Section 4 and maximize the blockchain throughput in Section 5. In Section 6, we analyze the security of the proposed framework based on the STRIDE model. In Section 7, the performance of the computing power management scheme is evaluated, after which we derive a strategy to select lightweight nodes. In Section 8, we discuss the merits of the developed approach and potential application scenarios of this paper. Finally, in Section 9, we conclude this work and give our future research directions.

2 Background and Related Work

The performance of blockchain systems is commonly evaluated in terms of three metrics - decentralization, scalability, and security, representing the so-called *impossible triangle*. It is argued that only up to two of the metrics can be improved simultaneously [16]. For example, compared to public blockchains, consortium blockchains yield higher scalability but compromise on security and reduce the extent of decentralization [17]. Moreover, the operational and standby miners in consortium blockchain systems can easily collude to attack the systems due to the limited number of miners. One possible solution to address this problem is to involve a group of lightweight nodes (e.g., edge devices like laptops and smartphones) in the mining process, which will increase the total number of nodes verifying blocks to mitigate collusions among malicious miners. In this regard, many previous works (e.g., [18] - [25]) studied the process of recruiting lightweight nodes in blockchain mining. Most of these works focused on the application of lightweight nodes during block generations with the aim of improving blockchain security and scalability. For example, the authors in [18] proposed the hardware-based implementation to replace cryptographic functions in the classical consensus protocols (e.g., Proof-of-Work, PoW) to facilitate the participation of low-energy lightweight nodes (such as IoT devices) in the block generation process with reduced power consumption. In [19] and [20], the authors constructed mining pools comprising groups of collaborating lightweight nodes, while in [21] and [22], lightweight nodes are allowed to mine blocks independently. The authors in [23] proposed the consensus protocol for block mining with lightweight nodes. The authors in [24] and [25] developed schemes for offloading mining tasks with the support of lightweight nodes. However, as generally lightweight nodes have fewer resources than miners, this results in increased block delays.

In consortium blockchains, the block producer, i.e., the miner responsible for producing a new block, is mostly preselected via absolute-finality consensus protocols such as DPoS and PBFT. This significantly reduces the block generation delay but limits the capacity for decentralization. Nevertheless, limited research has been conducted involving lightweight nodes in the block validation process of consortium blockchains. In [26], the authors proposed the Proof-of-Trust consensus protocol in the crowdsourcing system, where new transactions are validated by lightweight nodes recommended by the block producer. After that, the block producer orders and records the valid transactions (obtaining high votes) into a new block that is verified by miners. Similarly, in [27], service providers and consumers, which can be lightweight nodes in consortium blockchain-based crowdsourcing systems, are allowed to validate blocks in addition to miners. In [11], the authors applied lightweight nodes for verifying blocks with the consensus protocol of PoS to improve the security of block validation. In this work, the Stackelberg game is exploited to optimize the trade-off of block broadcast delay and transaction fee from users in order to decide the number of lightweight nodes to be recruited. In [28], the authors proposed a consortium blockchain-enabled energy trading market in smart grids, where only sellers can be miners, and new blocks are verified by both sellers and consumers, i.e., lightweight nodes. In our previous works [29] - [30], we considered lightweight nodes-assisted block validations in consortium blockchains and minimized the *upper bound* of block validation delays of the system by optimizing the computing powers of lightweight nodes and miners. However, our previous schemes would still have low blockchain throughput with the existence of resource-constrained lightweight nodes. In order to solve this problem, selections of lightweight nodes must be taken into consideration to maximize the *exact* blockchain throughput.

Unfortunately, this selection strategy to increase the blockchain throughput and the details of the block validation process are not presented in-depth in the existing works. In this paper, we propose a general lightweight nodes-engaged block validation

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2023.3343839

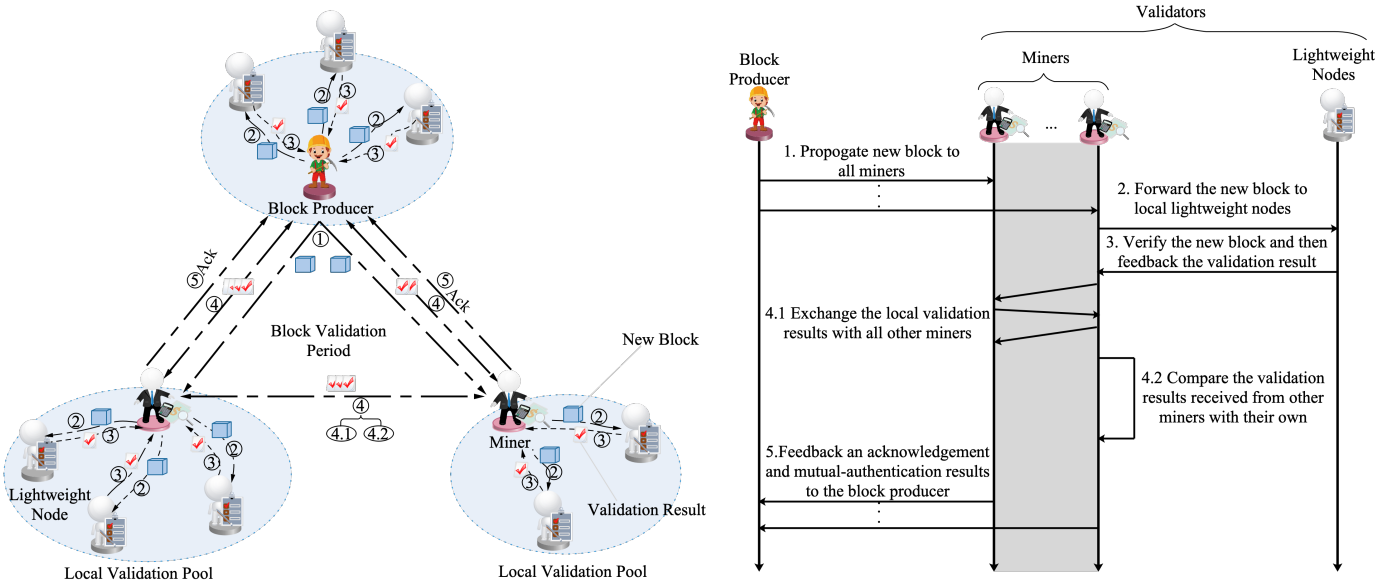


Fig. 1: Block validation framework and procedure for consortium blockchains.

framework and elaborate on the related procedure to enhance the security of consortium blockchains. Moreover, we devise an effective method to select appropriate lightweight nodes based on their computing powers for the block validations obtained from the proposed computing power management scheme; this maximizes the blockchain throughput.

3 Block Validation Elaboration

3.1 Secure and Scalable Block Validation Framework

In typical consortium blockchains, limited number of miners cooperate to verify new blocks during each consensus (i.e., blockchain mining) period, which can easily collude to falsely accept conflicting transactions (i.e., double-spending attacks) or reject a valid block. These attacks, i.e., miners' collusion attacks, cause serious security problems in the blockchain systems. To secure the block validation process, we study recruiting lightweight nodes acting as validators for verifying blocks. Moreover, the proposed framework provides increased scalability for consortium blockchains since the consortium blockchain nodes are preselected, and conversely, the lightweight nodes can be employed more freely and efficiently.

Fig. 1 presents the proposed block validation framework in consortium blockchains, where different consensus protocols can be applied. Prior to the block validation process, each miner constructs a local validation pool by selecting appropriate lightweight nodes based on their computing powers. The proposed framework includes three types of entities, as described below.

1) Block Producer: The block producer is responsible for generating a new block during the contemporary consensus period. It records users' transactions received from consortium blockchain nodes (CBNs) to form a new block. Notably, the proposed framework can be applied with both absolute-finality and probabilistic-finality consensus protocols [15]. In the former, one preselected block producer produces one block in the consensus period. The most typical examples are DPoS and PBFT. In the latter, multiple block producers compete to generate new blocks, possibly producing more than one block to be verified simultaneously. In this case, we adopt the first-come-first-served rule for the miners to process the blocks to relieve the computing and communicating burden of resource-constrained devices that constitute the lightweight nodes.

2) Miner: The miners take the responsibility of verifying new blocks with local lightweight nodes. Each miner picks a set of proximate lightweight nodes to facilitate secure block validations. Note that the minimal number of lightweight nodes in a validation pool can be preset to meet the security requirement of the system, and the ratio of lightweight nodes in each pool be optimized to prevent a pool from manipulating the block validations by controlling most lightweight nodes. In this paper, we only require that the recruited lightweight nodes register and possess a legal identity on the system, while further requirements will be discussed in the future. During block validation, the miners send a new block to the local lightweight nodes and collect their validation results. Furthermore, the miners must exchange the local validation results with each other, and authenticate the results they receive. The block is accepted if the miner receives positive validation results from the majority of validators (i.e., miners and lightweight nodes) in the system. Importantly, each miner must wait for the validation results from all miners due to the scale of lightweight nodes in each validation pool being dynamic and unknown. Eventually, the miner sends the authentication results as an acknowledgement to the block producer before appending the block to the local ledger.

3) Lightweight Node: The lightweight nodes are those edge devices that provide idle computing powers for block validations. In return, they can share transaction fees from users according to their contributions of resources [11]. We divide the process of block validation into three sub-steps: 1) *Verify the accuracy of the block header*; 2) *Verify the accuracy of the Merkle tree*; and 3) *Verify the validity of the transactions*. In the last sub-step, validators must verify the senders' and receivers' signatures, unspent inputs, and whether total inputs are equal to total outputs [14]. In unspent transaction output (UTXO)-based systems, the miners track the unspent inputs by checking the UTXOs of the previous related transactions. However, each transaction always has multiple inputs and outputs, and the maximum number of UTXOs per transaction can even reach 3452 in the Bitcoin system with more than 2GB of all UTXOs in the system by 2017 [31]. Hence, this method causes enormous communication and computation burden for lightweight nodes to synchronize and track the unspent tokens.

In this paper, we adopt the account-based method (as the Ethereum system [32]) for validators to track unspent tokens [33],

in which each transaction has only one input (wallet balance) and output. Specifically, global data known as the World State is used to store users' account information, including account balances and nonces indicating their transaction numbers triggered in the past. The data structure of World State is based on the Modified Merkle Patricia Tree (MMPT) [32], and each leaf is one user's account information. The World State is stored in LevelDB, and only the tree root is recorded in the block header as proof of the tree. Though the World State is a few gigabytes in size, validators can search the account information quickly based on the MMPT and download part of the tree (a few kilobytes) [34].

3.2 Unified and Secure Block Validation Procedure

Next, we introduce the details of the block validation procedure in our proposed framework to present its difference from that in typical consortium blockchains. The proposed procedure comprises five steps, as shown in Fig. 1, which are described below:

Step 1: The block producer sends the newly-produced block to miners with its public key and the signature of the block. Note that the number of block producers and new blocks produced in a consensus period relies upon the consensus protocol adopted. Besides, all messages-in-transit (e.g., blocks) are encrypted by the combination of symmetric and asymmetric encryption methods because asymmetric encryption is slower and yields more computation overhead yet higher security than symmetric encryption. The messages are encrypted by a cryptographic (symmetric) key. Then, the symmetric key is encrypted using the public key of the recipient, which gets the symmetric key using its paired private key. More detail about the encryption can be found in section 6.2.

Step 2: The miners inspect the validity and integrity of the received block using the attached public key and signature before forwarding it to local lightweight nodes. In cases of having only one block to be verified in the network, the miner can also request the block from the nearby miners if the received block is modified during transmission. If more than one verifying block exists, the miners discard those invalid blocks, and chronologically transmit the valid blocks one by one to the lightweight nodes. That is, the next block is propagated after the miners aggregate all validation results of the verifying block. The discarded block will be abortive if each miner cannot get enough positive verification results before the timeout of block validation. In order to verify the unspent input of transactions, the senders' account information in the World State (i.e., related branches of the MMPT) and the previous block header in the blockchain are also sent to the lightweight nodes.

Step 3: The lightweight nodes also check the validity and integrity of the block after obtaining the verification information. As lightweight nodes are free to access every miner, they can request the block from other miners by uploading the block header if the received block is incomplete. To allow other miners to verify the identity of selected lightweight nodes, each miner must generate (sign) verifiable acceptance proof for local lightweight nodes when they are selected. This can prevent their local miner from modifying the block intentionally. Note that the block header cannot be tampered with because this can be easily detected during the mutual authentication in Step 4. Following this process, the information in the block header is audited by the validators, including the timestamp, block producer's signature, and hash values of the previous and verified block headers [1]. In detail, the timestamp cannot be smaller than that of the last block in the blockchain. The hash values of the block headers are examined using a Hash Function [35]. After this, the validity of the Merkle tree in the verified block is inspected by generating a root of the tree with the transactions as leaves in chronological order.

Ultimately, the lightweight nodes audit the accuracy of the received World State from the miners in the same way as verifying Merkle trees, followed by verifying each transaction in the block. Specifically, the sender's and receiver's signatures in a transaction are verified using their public keys. Then, the sender's account balance in the World State is checked. Moreover, the senders have to generate a nonce for each proposed transaction, which should be consistent with the nonce of their accounts. Hence, the nonces of the transaction and the sender's account are compared while verifying the transaction. Finally, the validators add one to the nonce of the sender's account if the transaction is valid. After all transactions are verified, the validators check the proof of World State in the block header by regenerating a root of the MMPT according to the updated account information if the above validation results are positive. In contrast, the validators use *Error Codes* to mark and record each anomalous validation result detected during the verification. Eventually, the lightweight node sends its validation result to the local miner with its signature, public key and the verified block header.

Step 4: The miner gathers all the local validation results and exchanges them with other miners (Step 4.1 in Fig. 1). The results are then authenticated by each miner as long as they complete their collection of local validation results (Step 4.2 in Fig. 1). First, the miner retrieves its own validation result related to the block header to be verified. (If the received block header has been modified and cannot be found, the validation result is marked as an error directly.) Then, the validity of the received result is verified using the attached signatures with the public keys of the corresponding validators. Finally, the miner records the results if they are positive and consistent with that of the miner.

Step 5: In the absolute-finality consensus protocols, the miner submits authentication results as an acknowledgement to the block producer if most of the validation results in the network are positive. The authentication result consists of the positive validation results and error codes. Then, the miners append the valid block to the local ledger. Here, it is resource-consuming for miners to keep updating information (e.g., number) of the lightweight nodes in each pool. Instead of having complete information on lightweight nodes in the system, each miner waits for the validation results from all miners to confirm whether they receive positive validation results from the majority of validators. Last, the current consensus period is finalised when the block producer gets the acknowledgements from the majority of miners. For probabilistic-finality consensus protocols, the miner appends the verified block to the local ledger without informing the block producer when getting the positive validation results from the majority of validators. When blockchain forking occurs, and one of the sub-chains has six blocks from the forking place in advance, the remaining sub-chains are pruned and invalidated [36].

Lastly, in order to prevent malicious behaviour of lightweight nodes, reputation evaluation can be applied for each node based on their validation results, e.g., error codes detected [37]. With this, the malicious lightweight nodes can be blacklisted if their reputations are below a specific threshold. The reputation evaluation of lightweight nodes will be carried out in detail in the future.

4 Block validation Delay Analysis

In order to maximize blockchain throughputs, the strategy to select lightweight nodes (e.g., based on their computing powers) must be

taken into consideration. In this section, we analyse the total block validation delay of the system based on the computing powers of miners and lightweight nodes, with the adopted absolute-finality consensus protocols. We divide the block validation process into two stages: (1) *local block validation* of the miners and lightweight nodes, and (2) *mutual-authentication* between the miners on their local validation results. In each consensus period, the block producer is preselected for generating a new block that is then verified by $\mathcal{A}$ miners. The set of miners is represented as $\mathbb{S} = \{\mathcal{M}_i\}_{i=1}^{\mathcal{A}}$, in which $\mathcal{M}_1$ is the block producer. Furthermore, each $\mathcal{M}_i$ constructs a local validation pool $\mathbb{P}_i$ through selecting appropriated lightweight nodes in proximity, the number of which is n_i. The lightweight node j in $\mathbb{P}_i$ is denoted as $\mathcal{L}_i^j$. Besides, R_{xy} is data rate from x to y, e.g., R_{ij} is the data rate from $\mathcal{M}_i$ to $\mathcal{L}_i^j$.

4.1 Local Block Validation

After collecting and packaging transactions to a new block, the block producer delivers the block of size S_{blk} to each $\mathcal{M}_i$ with the transmission latency given by $T_{1i}^{bt} = S_{blk}/R_{1i}$, where $T_{11}^{bt} = 0$. Then, $\mathcal{M}_i$ forwards the received block and auxiliary validation information to each $\mathcal{L}_i^j$. The forwarding latency of the validation information is given by $T_{ij}^{bt} = (S_{blk} + S_{aux})/R_{ij}$, where S_{aux} is the size of auxiliary validation information.

Next, $\mathcal{L}_i^j$ verifies the block with its computing power f_{ij}^v, less than its capacity F_{ij}^{max}. We assume that the average workload of verifying unit size of the block is ν. Thus, the latency of $\mathcal{L}_i^j$ to verify the block can be given by $T^v(f_{ij}^v) = \nu S_{blk}/f_{ij}^v$.

Finally, the validation result of $\mathcal{L}_i^j$ is returned to $\mathcal{M}_i$, with an upload latency given by $T_{ji}^{ur} = S_{vr}/R_{ji}$, where S_{vr} is the size of a validation result. Note that R_{ji} can be predicted before the recruitment based on the information signal sent by the lightweight nodes to the local miners with a specific power [38].

In short, the local block validation delay of $\mathcal{L}_i^j$ is defined as

$$\begin{aligned} T_{ij}^L(f_{ij}^v) &= T_{ij}^{bt} + T^v(f_{ij}^v) + T_{ji}^{ur} \\ &= (S_{blk} + S_{aux})/R_{ij} + \nu S_{blk}/f_{ij}^v + S_{vr}/R_{ji}. \end{aligned} \quad (1)$$

The *local block validation* in $\mathbb{P}_i$ completes when the validation results in the pool are aggregated by the local miner. The local block validation delay of $\mathbb{P}_i$ is given by

$$T_i^L(\mathbf{f_i^v}) = T_{1i}^{bt} + \max_{j \in \mathbb{P}_i}\{T_{ij}^L(f_{ij}^v)\}, \quad (2)$$

where $\mathbf{f_i^v} = [f_{i1}^v, ..., f_{in_i}^v]^T$. In particular, the latency of miners for verifying the block is negligible due to the fact that they have far greater computing powers than the lightweight nodes.

4.2 Mutual-authentication

To confirm whether the block is accepted by the majority of validators, the miners must mutually authenticate the validation results in the network. Therefore, the miners collect all the local validation results and exchange with each other. The transmission latency from $\mathcal{M}_{i'}$ to $\mathcal{M}_i$ is given by $T_{i'i}^u = [(n_{i'}+1)S_{vr}]/R_{i'i}$, where $(n_{i'}+1)$ is the number of validators in $\mathbb{P}_{i'}$. We denote the workload to authenticate a validation result as ζ. The authentication latency of $\mathcal{M}_i$ for the validation results sent by $\mathcal{M}_{i'}$ is defined as

$$T_{ii'}^a(f_{ii'}^a) = [(n_{i'}+1)\zeta]/f_{ii'}^a, \quad (3)$$

where $f_{ii'}^a$ is the computing power of $\mathcal{M}_i$ on authenticating the validation results from $\mathcal{M}_{i'}$. Besides, the computing power of $\mathcal{M}_i$ for the authentication (i.e., $\sum_{\forall i' \in \mathbb{S}\setminus\{i\}} f_{ii'}^a$) should be within its capacity F_i^{max} when it receives and authenticates validation results from different miners simultaneously.

Furthermore, during the *local block validation*, the lightweight nodes have the distinct local block validation delays. This enables the miner to authenticate the local validation results with its powerful computing powers while receiving them at different times and thus, the corresponding authentication latency is negligible. For authenticating the validation results from $\mathcal{M}_i'$, the latency of $\mathcal{M}_i$ includes two parts: (1) *Waiting Latency* to receive the validation results from $\mathcal{M}_{i'}$ (i.e., $T_{ii'}^w(\mathbf{f_i^v}, \mathbf{f_{i'}^v})$), and (2) *Authentication Latency* to authenticate the received results (i.e., $T_{ii'}^a(f_{ii'}^a)$). To analyse the waiting latency, we sort the miners based on the local block validation delay of their validation pools in ascending order, i.e., $T_1^L(\mathbf{f_1^v}) < T_2^L(\mathbf{f_2^v}) < ... < T_{\mathcal{A}}^L(\mathbf{f_{\mathcal{A}}^v})$. In addition, we analyse three cases of waiting latency of $\mathcal{M}_i$ as below.

Case 1: $\mathcal{M}_i$ completes collection of local validation results, which are broadcast to $\mathcal{M}_{i'}$ ($i < i'$). During the transmission, $\mathcal{M}_{i'}$ also finishes and broadcasts its collection to $\mathcal{M}_i$. In this case, they must arrange the occupation slots of the shared channel for sending their message [39]. In fact, the miners must complete receipt of the transmitted data before authenticating the results. Hence, it is more efficient that the shared channel is fully occupied by the transmitter that first proposes the transmission request. This can also ensure the stability of data transmission with reduced interference. As such, $\mathcal{M}_{i'}$ starts transmission after completing receipt of the validation results from $\mathcal{M}_i$. The waiting latency of $\mathcal{M}_i$ to get the results sent by $\mathcal{M}_{i'}$ is denoted as

$$T_{ii'}^{w_1}(\mathbf{f_i^v}) = T_i^L(\mathbf{f_i^v}) + T_{ii'}^u + T_{i'i}^u. \quad (4)$$

Case 2: Conversely, when completing its local collection, $\mathcal{M}_i$ is receiving or waiting for messages from $\mathcal{M}_{i'}$ ($i' < i$) . Hence, the waiting latency of $\mathcal{M}_i$ for the validation results sent by $\mathcal{M}_{i'}$ depends on the local block validation delay of $\mathbb{P}_{i'}$, given by

$$T_{ii'}^{w_2}(\mathbf{f_{i'}^v}) = T_{i'}^L(\mathbf{f_{i'}^v}) + T_{i'i}^u. \quad (5)$$

Case 3: Before completing its local collection, $\mathcal{M}_i$ has already got the validation results sent by $\mathcal{M}_{i'}$ ($i' < i$). That is, the waiting latency equal to zero. Yet, it must complete its local collection before authenticating the received results. In this case, the waiting latency of $\mathcal{M}_i$ to obtain the validation results sent by $\mathcal{M}_{i'}$ is replaced by the latency for finishing its local collection given by

$$T_{ii'}^{w_3}(\mathbf{f_i^v}) = T_i^L(\mathbf{f_i^v}). \quad (6)$$

Following this, $\mathcal{M}_i$ authenticates the received validation results from $\mathcal{M}_{i'}$ with the latency $T_{ii'}^a(f_{ii'}^a)$. As long as confirming the block is valid, the miner uploads the authentication results as an acknowledgement to the block producer. Nevertheless, since the number of lightweight nodes in each pool is dynamic and unknown, it is energy-intensive for miners to keep checking the total number of validators in the network. The solution is to wait for the validation results from all miners, based on which each miner can judge whether they have received positive validation results from the majority of validators. Therefore, the total authentication delay of $\mathcal{M}_i$ is defined as

$$T_i^M(\mathbf{f^v}, \mathbf{f_i^a}) = \max_{i' \in \mathbb{S}\setminus\{i\}}\{T_{ii'}^w(\mathbf{f_i^v}, \mathbf{f_{i'}^v}) + T_{ii'}^a(f_{ii'}^a)\}, \quad (7)$$

$$T_{ii'}^w(\mathbf{f_i^v}, \mathbf{f_{i'}^v}) = \begin{cases} T_i^L(\mathbf{f_i^v}) + T_{ii'}^u + T_{i'i}^u, w = w_1 \\ T_{i'}^L(\mathbf{f_{i'}^v}) + T_{i'i}^u, w = w_2 \\ T_i^L(\mathbf{f_i^v}), w = w_3 \end{cases}, \quad (8)$$

$\mathbf{f^v} = [\mathbf{f_1^v}, ..., \mathbf{f_{\mathcal{A}}^v}]$, and $\mathbf{f_i^a} = [f_{i1}^a, ..., f_{ii'}^a, ..., f_{i\mathcal{A}}^a]_{i' \neq i}^T$. In (7), the total authentication delay of $\mathcal{M}_i$ depends on the maximal

latency to receive and authenticate validation results sent by one of the miners. After authenticating all validation results, $\mathcal{M}_i$ responds to the block producer with the authentication results as an acknowledgement. The transmission latency is given by $T_{i1}^{ack} = [S_{ar} + S_{vr}(1/2(\Sigma_{i=1}^{\mathcal{A}} n_i + \mathcal{A}) + 1)]/R_{i1}$. To improve the transmission efficiency, the miners only submit the majority, i.e., $1/2(\sum_{i=1}^{\mathcal{A}} n_i + \mathcal{A}) + 1$, of positive validation results in the network to the block producer. S_{ar} denotes the size of attack records. In the end, the miners append the valid block to the local ledger. The total block validation delay of $\mathcal{M}_i$ is denoted as

$$T_i(\mathbf{f^v}, \mathbf{f_i^a}) = T_i^M(\mathbf{f^v}, \mathbf{f_i^a}) + T_{i1}^{ack}. \tag{9}$$

5 Computing Power Optimization for Efficient Block Validation

5.1 Problem Statement

First, it is necessary to optimize computing powers contributed by miners and lightweight nodes for block validations to reduce the block validation delay incurred by lightweight nodes and, thereby, improve the blockchain throughput. This can help miners exclude slow and resource-constraint lightweight nodes, which cannot return their validation results in time, during the selection process. *Second*, based on the optimal computing powers of lightweight nodes, the miners can select those with less redundant computing powers and, thus, save the recruitment cost. Note that, to ease the recruitment, we exploit a one-request one-response rule during the selection process. That is, lightweight nodes send a request to the local miner with their expected (i.e., maximal) computing power to contribute, and the miners respond with a selection decision. In this case, some lightweight nodes may contribute large computing powers while the block validation delay is subject to other slower nodes, which wastes part of their computing powers.

In consortium blockchains, the exact blockchain throughput is given by the time that the block producer receives the acknowledgements from the majority (i.e., $\lceil \frac{\mathcal{A}+1}{2} \rceil$) of miners. To compute the blockchain throughput, we sort the total block validation delay of miners in ascending order, i.e., $T_1(\mathbf{f^v}, \mathbf{f_1^a}) < ... < T_{\mathcal{A}}(\mathbf{f^v}, \mathbf{f_{\mathcal{A}}^a})$. Hence, the system utility function (i.e., throughput) is given by

$$U(\mathbf{f^v}, \mathbf{f^a_{\lceil \frac{\mathcal{A}+1}{2} \rceil}}) = T^{-1}_{\lceil \frac{\mathcal{A}+1}{2} \rceil}(\mathbf{f^v}, \mathbf{f^a_{\lceil \frac{\mathcal{A}+1}{2} \rceil}}), \tag{10}$$

where $T^{-1}_{\lceil \frac{\mathcal{A}+1}{2} \rceil}(\mathbf{f^v}, \mathbf{f^a_{\lceil \frac{\mathcal{A}+1}{2} \rceil}}) = 1/T_{\lceil \frac{\mathcal{A}+1}{2} \rceil}(\mathbf{f^v}, \mathbf{f^a_{\lceil \frac{\mathcal{A}+1}{2} \rceil}})$.

5.2 Centralized Problem Formulation

Next, we formulate the computing powers of all validators as an optimization problem, with the aim of maximizing the system utility. The optimization problem is formulated as

$$\text{maximize } U(\mathbf{f^v}, \mathbf{f^a_{\lceil \frac{\mathcal{A}+1}{2} \rceil}}) = T^{-1}_{\lceil \frac{\mathcal{A}+1}{2} \rceil}(\mathbf{f^v}, \mathbf{f^a_{\lceil \frac{\mathcal{A}+1}{2} \rceil}}) \tag{11a}$$

$$\text{s.t. } 0 < \Sigma_{i' \in \mathbb{S} \setminus \{i\}} f^a_{ii'} \leq F_i^{max}, \forall i \in \mathbb{S} \tag{11b}$$

$$0 < f^v_{ij} \leq F_{ij}^{max}, \forall j \in \mathbb{P}_i, i \in \mathbb{S} \tag{11c}$$

$$T_i(\mathbf{f^v}, \mathbf{f_i^a}) \leq T^{max}, \forall i \in \mathbb{S}. \tag{11d}$$

(11b) and (11c) indicate that the computing powers of each miner and lightweight node for the block validation cannot exceed its capacity. (11d) denotes that each miner should complete the block validation within the allowed delay of the system T^{max}.

Lemma 1: *The problem (11a) can be transformed to*

$$\text{maximize } \Sigma_{i=1}^{\mathcal{A}} T_i^{-1}(\mathbf{f^v}, \mathbf{f_i^a}) \tag{12a}$$

$$\text{subject to } (11b), (11c).$$

Lemma 2: *The objective of the optimization problem in (12a) is a strictly-concave function.*

Lemma 1 and *Lemma 2* are, respectively, proved in Appendix A and B. Based on *Lemma 2* and the linear constraints (11b) and (11c), (12a) has only one globally optimal solution [40]. To solve this problem, centralized optimization tools can be applied, provided the latest information of all lightweight nodes (e.g., real-time data rate and maximal computing power) are collected in time by the centralized entity implementing the optimization algorithm. However, continuously updating and broadcasting information of large-scale lightweight nodes can occupy numerous resources on the blockchain. Besides, appointing such a centralized entity breaks the decentralized nature of blockchains and the problem must be solved in a distributed way. Thus, in the next section, we propose a decentralized approach to solve (12a) efficiently.

5.3 Distributed Optimization Problem

Solving (12a) is challenging as $\mathbf{f^v}$ and $\mathbf{f^a}$ are highly coupled with constraint (11b). Thus, we split (12a) into two sub-problems by decoupling $\mathbf{f^v}$ and $\mathbf{f^a}$. Then, to allow miners to solve the problem locally and parallelly, we must further decouple $\mathbf{f^v}$ into $\{\mathbf{f_i^v}\}_{i=1}^{\mathcal{A}}$. Notably, each $\mathbf{f_i^a}$ in $\mathbf{f^a} = [\mathbf{f_1^a}, ..., \mathbf{f_{\mathcal{A}}^a}]$ is an independent variable in (12a). After decoupling, each miner can repeatedly optimize their local variable $(\mathbf{f_i^v}, \mathbf{f_i^a})$ until all variables converge. Importantly, in (7)-(8), each miner only share the local block validation delay of their pools (T_i^L) with others during optimization, greatly reducing data exchanged compared to centralized optimization methods (i.e., share information of all lightweight nodes). Next, we elaborate the decoupling process and each sub-problem solutions.

5.4 Sub-problem 1: Optimize $\mathbf{f^v}$ Given $\mathbf{f^a}$

Here, $\mathbf{f^a}$ is a constant after decoupling $\mathbf{f^v}$ with $\mathbf{f^a}$. Furthermore, by decoupling $\mathbf{f^v}$ into $\{\mathbf{f_i^v}\}_{i=1}^{\mathcal{A}}$, theoretically, the problem in (12a) can be solved by optimizing each $\mathbf{f_i^v}$ iteratively until convergence. In practice, each $\mathcal{M}_i$ can solve the problem in (12a) by optimizing the local variable $\mathbf{f_i^v}$ parallelly, given the constant $\{\mathbf{f_{i'}^v}\}_{i'=1}^{\mathcal{A}}$ $(i' \neq i)$. During the local optimization, each miner only exchanges the local verification delay of their pools. The local optimization problem of $\mathcal{M}_i$ can be written as

$$Q1: \text{maximize } \Sigma_{n=1}^{\mathcal{A}} T_n^{-1}(\mathbf{f_i^v}) \tag{13a}$$

$$\text{s.t. } 0 < f^v_{ij} \leq F_{ij}^{max}, \forall j \in \mathbb{P}_i, \tag{13b}$$

in which $\mathbf{f_{i'}^v}$ $(i' \neq i, i' \in \mathbb{S})$ are constant. Furthermore, (13a) denotes the sum of throughput of $\mathcal{A}$ miners regarding $\mathbf{f_i^v}$, which can be split to $\mathcal{A}$ small sub-problems, i.e., $U_n(\mathbf{f_i^v}) = T_n^{-1}(\mathbf{f_i^v}), n = 1, ..., \mathcal{A}$. Then, the miner can solve each small sub-problems parallelly and efficiently. To this end, we introduce the alternating direction method of multipliers (ADMM) to decouple the problem [41], in which (13a) is considered an ADMM-based global consensus problem [42]. In the following of this section, $\mathbf{f}$ is used to denote $\mathbf{f_i^v}$ to simplify the representation. We assume $\mathbf{f}$ is a global variable of the problem in (13a), and $\mathcal{F} = \{\mathbf{f_n}\}_{n=1}^{\mathcal{A}}$ are $\mathcal{A}$ local variables in $\mathcal{A}$ small sub-problems $U_n(\mathbf{f_n})$, respectively. The miner first solves each small sub-problems parallelly and obtains $\mathcal{F}^* = \{\mathbf{f_n^*}\}_{n=1}^{\mathcal{A}}$, which are then used to update the global variable $\mathbf{f}$. This process is repeated until $\mathbf{f}$ converges. The detail is given as follows. First, (13a) can be written in the ADMM form as:

$$Q1.1: \underset{\mathcal{F}}{\text{maximize}} \ \Sigma_{n=1}^{\mathcal{A}} T_n^{-1}(\mathbf{f_n}) \tag{14a}$$

$$\text{s.t. } \mathbf{f_n} - \mathbf{f} = 0 \tag{14b}$$

$$\mathbf{0} < \mathbf{f_n} \leq \mathbf{F}, 1 \leq n \leq \mathcal{A}. \tag{14c}$$

The constraint (14b) means that all the local variables should finally equal to the global variable, i.e., reaching consensus between each small sub-problems regarding $\mathbf{f}$. (14c) corresponds to (13b), and $\mathbf{F} = [F_{ij}^{max}]_{j\in\mathbb{P}_i}^T$. To solve this problem, ADMM uses the augmented Lagrangian [43] defined as

$$L_\beta(\mathcal{F}, \mathbf{f}, \gamma) = \Sigma_{n=1}^{\mathcal{A}}(T_n^{-1}(\mathbf{f_n}) - \gamma_n^T(\mathbf{f_n} - \mathbf{f}) - 0.5\beta||\mathbf{f_n} - \mathbf{f}||_2^2), \tag{15}$$

where $\gamma = \{\gamma_n\}_{n=1}^{\mathcal{A}}$ are dual variables or Lagrangian multipliers. The last item in (15) is a penalty item with the penalty parameter β. Here, the penalty item push each local variable towards consensus while maximizing the objective function. Though we have proved the convexity of the centralized problem (12a), the augmented Lagrangian can ease the requirement of convexity of objective functions and make the ADMM algorithm converge with no hard assumptions [42]. In the ADMM, gradient descent is exploited to solve the problem. In each round (i.e., r) of ADMM, the following two steps are repeated:

- Step 1: maximize the augmented Lagrangian L_β regarding the local variables $\mathcal{F}$ (under the condition $\mathbf{0} < \mathbf{f_n} \leq \mathbf{F}$):

$$\begin{aligned}\mathcal{F}^{\mathbf{r}} &= \underset{\mathcal{F}}{\text{argmax}}\ L_\beta(\mathcal{F}, \mathbf{f}, \gamma) \\ &= \underset{\mathcal{F}}{\text{argmax}}\ \Sigma_{n=1}^{\mathcal{A}}(T_n^{-1}(\mathbf{f_n}) - (\gamma_n^{r-1})^T(\mathbf{f_n} - \overline{\mathbf{f}}^{r-1}) - \\ &\qquad 0.5\beta||\mathbf{f_n} - \overline{\mathbf{f}}^{r-1}||_2^2),\end{aligned} \tag{16}$$

  in which $\overline{\mathbf{f}}^{r-1}$ is a global variable - the mean of variables in $\mathcal{F}^{r-1}$. Importantly, (16) can be solved parallelly as follows:

$$\begin{aligned}\forall \mathbf{f_n} \in \mathcal{F} : \underset{\mathbf{f_n}}{\text{argmax}}\ &(T_n^{-1}(\mathbf{f_n}) - (\gamma_n^{r-1})^T(\mathbf{f_n} - \overline{\mathbf{f}}^{r-1}) - \\ &0.5\beta||\mathbf{f_n} - \overline{\mathbf{f}}^{r-1}||_2^2).\end{aligned} \tag{17}$$

- Step 2: update the dual variable γ towards the negative gradient direction in a parallel manner:

$$\gamma_n^r = \gamma_n^{r-1} + \beta(\mathbf{f_n^r} - \overline{\mathbf{f}}^r). \tag{18}$$

Here, we can solve (17) with Karush–Kuhn–Tucker (KKT) conditions [44]. Note that (17) implies two types of objective functions corresponding to $T_i(\mathbf{f_n})$ and $T_{i'}(\mathbf{f_n})$. Based on Appendix C, the solution of (17) corresponding to $T_i(\mathbf{f_n})$ is given by

$$\begin{cases} \mathbf{f_n^*}[k] = \mathbf{F}[k], \forall \xi_{ki'} = \rho_{ki'} = 0 \\ \mathbf{f_n^*}[k] = \frac{\nu S_{blk}}{\mathcal{X}^* - T_{1i}^{bt} - T_k^{bt} - T_k^{ur} - T_{ii'}^u - T_{i'i}^u - T_{ii'}^a}, \xi_{ki'} \neq 0 \\ \mathbf{f_n^*}[k] = \frac{\nu S_{blk}}{\mathcal{X}^* - T_{1i}^{bt} - T_k^{bt} - T_k^{ur} - T_{ii'}^a}, \rho_{ki'} \neq 0. \end{cases} \tag{19}$$

The solution of (17) corresponding to $T_{i'}(\mathbf{f_n})$ is given by

$$\begin{cases} \mathbf{f_n^*}[k] = \mathbf{F}[k], \pi_k = 0 \\ \mathbf{f_n^*}[k] = \frac{\nu S_{blk}}{\mathcal{X}^* - T_{1i}^{bt} - T_k^{bt} - T_k^{ur} - T_{ii'}^u - T_{i'i}^a}, \pi_k \neq 0, \end{cases} \tag{20}$$

where $\mathbf{f_n^*}[k]$ is the kth element of $\mathbf{f_n^*}$. $\xi_{ki'}, \rho_{ki'}$ and π_k are Lagrange multipliers, and $\mathcal{X}^*$ is the optimal value of an intermediate variable $\mathcal{X}$ in Appendix C.

5.5 Sub-problem 2: Optimize $\mathbf{f^a}$ Given $\mathbf{f^v}$

After solving *Q1*, the intermediate-optimal value $\mathbf{f^v}$ is inserted into (12a) to optimize $\mathbf{f^a}$. (12a) can be rewritten as

$$Q2: \text{maximize}\ \Sigma_{i=1}^{\mathcal{A}} T_i^{-1}(\mathbf{f_i^a}) \tag{21a}$$

$$\text{s.t.}\ 0 < \Sigma_{i'\in\mathbb{S}\setminus\{i\}} f_{ii'}^a \leq F_i^{max}, \forall i \in \mathbb{S}. \tag{21b}$$

We can find from (7) that each $T_i^{-1}(\mathbf{f_i^a})$ in *Q2* is fully independent of each other. Thus, the problem (21a) can be solved by optimizing each $T_i^{-1}(\mathbf{f_i^a})$ separately. More importantly, each miner $\mathcal{M}_i$ can optimize $T_i^{-1}(\mathbf{f_i^a})$ locally. Here, $T_{ii'}^w(\mathbf{f_i^v}, \mathbf{f_{i'}^v})$ in (7) is a constant (denoted as $\Gamma_{ii'}$). Inserting (3) to (21a), *Q2* takes the form:

$$\underset{\mathbf{f_i^a}}{\text{maximize}}\ (\mathcal{Y} + T_{i1}^{ack})^{-1} \tag{22a}$$

$$\text{s.t.}\ 0 < \Sigma_{i'\in\mathbb{S}\setminus\{i\}} f_{ii'}^a \leq F_i^{max} \tag{22b}$$

$$\Gamma_{ii'} + [(n_{i'}+1)\zeta]/f_{ii'}^a \leq \mathcal{Y}, \forall i' \in \mathbb{S}\setminus\{i\}. \tag{22c}$$

$\mathcal{Y}$ is an intermediate variable of $\underset{i'\in\mathbb{S}\setminus\{i\}}{max}\{\Gamma_{ii'} + T_{ii'}^{ma}(f_{ii'}^{ma})\}$. Since the objective function of (12a) is a strictly concave function, the sub-problem (21a) and (22a) are also concave functions with linear constraints. Thus, the sub-problems (22a) can be solved based on the KKT conditions. Based on Appendix D, we can find the solution regarding $f_{ii'}^a$:

$$f_{ii'}^a + \sum_{i''\in\mathbb{S}\setminus\{i,i'\}} \frac{(n_{i''}+1)\zeta}{(\Gamma_{ii'} - \Gamma_{ii''}) + \frac{(n_{i'}+1)\zeta}{f_{ii'}^a}} = F_i^{max}. \tag{23}$$

6 Security Analysis and Evaluation

In this section, we comprehensively analyse and identify potential attacks and threats in our framework and provide corresponding solutions against each threat during block validations, as shown in TABLE 1 and 2. Some typical attacks and threats have been addressed in existing blockchain systems (e.g., [45]), but some are only applicable to our framework and remain to be solved; for instance, threats and attacks incurred by the involvement of lightweight nodes in block validations. In addition, we evaluate the security level of the provided countermeasures against each threat. Then, we implement an in-depth analysis of the miners' collusion attack, which is the focus of this paper. We qualitatively analyse the resilience and quantitatively evaluate the security performance of our proposed framework against the miners' collusion attack.

6.1 Threat Identification

Blockchain systems are threatened with a great variety of attacks, in which one attack probably has multiple threats [45]. To assess the security of the proposed framework, we formally analyze and identify threats produced by different attacks during block validations. To this end, we use a popular threat analysis model proposed by Microsoft, namely STRIDE, which is widely applied for threat modelling, identification and categorization in blockchains [46]. In this model, threats are classified into five categories as follows:

- **Spoofing**: Adversaries leverage a false identity or impersonate the identity of a registered entity to gain permission for implementing legal operations.
- **Tampering**: Adversaries modify data exchanged in blockchain networks or those stored in databases of the blockchain (e.g., World State and on-chain transactions/blocks).
- **Repudiation**: Adversaries exploit the inability of the blockchain system to detect and track some malicious behaviours to implement specific attacks.
- **Information Leakage**: This refers to threats that private information is accessed by adversaries without permission.
- **Denial of Service**: This threat causes a node on the blockchain system unavailable to receive and process requests of users.
- **Elevation of Privilege**: Adversaries with restricted privileges conduct unauthorized actions requiring higher licenses or implement attacks to obtain a higher license in the system.

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2023.3343839

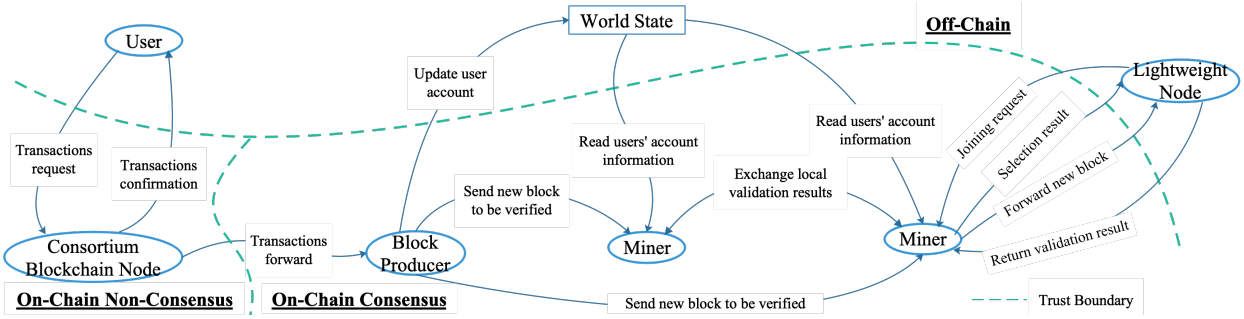


Fig. 2: Data flow diagram of our proposed framework.

The data flow diagram of our proposed framework is presented in Fig. 2, split into three trust zones (i.e., Off-Chain, On-Chain Non-Consensus, and On-Chain Consensus) based on the positions and duties of the entities. At each entity, the analysis of attacks and threats is conducted by answering the following questions:

- *What assets are processed by the entity?* Assets in the system refer to *transactions, transaction requests/confirmations, blocks, the World State, account information of users, validation results, joining requests, and selection results*.
- *Inside Attacker versus Intruder?* Inside attacker means that the investigated node itself is compromised, e.g., malicious miners in the miners' collusion attack. In contrast, an intruder, also called an outside attacker, implements attacks from outside the system, e.g., eavesdroppers in the eavesdropping attack.
- *How can each kind of threat in the STRIDE model be implemented in each step of data processing?* Giving an example of analysing threats of tampering at CBNs, we investigate how the compromised CBNs/intruders can tamper transactions while forwarding them to miners.

Hence, inputs/outputs and data processing procedures in each entity are analysed while identifying attacks and threats. In this way, the threats are classified into six categories of the STRIDE model, shown in TABLE 1 and 2. Though we focus on the security of block validations in consortium blockchains, the consensus process (from transactions triggered by users to block generation and verification at miners) is considered in the security analysis.

6.2 Security Level of Countermeasures

For each identified attack and threat, we evaluate the security level of the corresponding countermeasure, referring to the difficulty of conducting the attack by breaching the countermeasure. The security has five levels: Very Low(VL)/Low/Medium(Med)/High/Very High(VH). Some countermeasures rely on the embedded technologies in blockchains, which can robustly prevent related attacks and threats. In order to evaluate the security level of the provided countermeasures, the following robust techniques are introduced:

- *Digital Signature (DS)*: is the ciphertext of the digest of data (i.e., assets), signed/encrypted by private keys of data owners. Attaching it to data-in-transit enables recipients to audit the identity of senders and integrity of data. Recipients can use public keys of senders to decrypt digital signatures for comparing the digest with that of received data, which prevents impersonation attacks as intruders have to hack the private keys of senders to mimic their signatures. Yet, their private keys are encrypted and stored offline in local wallets, inaccessible through the network. *The offline storage of private keys* contributes to the very high security level of DS and builds up a robust resistance to attacks of data tampering/injection whenever digital signatures of the data are attached. Moreover, multi-signatures can be applied to provide higher-level security in generating digital signatures based on two-factor authentication (described below).
- *Symmetric/Asymmetric Encryption (SE/AE)*: SE/AE can encrypt data into non-readable ciphertexts to compress the data and hide its privacy. SE encrypts and decrypts data using the same cryptographic key (i.e., a symmetric key), which is extremely robust and widely used by the National Security Agency (NSA) of America. For instance, to attack the SE algorithm of AES (Advance Encryption Standard)-256, intruders have to try 2^{256} combinations to find the accurate 256-bit symmetric key [47]. Even if it can be broken theoretically, the encrypted data become outdated before being exposed. Hence, SE can protect the privacy of data with a very high security level. In contrast, AE has a pair of public/private keys used, generally, to encrypt and decrypt data, respectively. To decrypt the ciphertext, intruders must hack both the public and private keys. Yet, the ciphertext is transmitted in the network through the IP addresses of recipients without exposing or connecting with their public keys directly. And the offline storage can effectively secure private keys. Thus, AE is extremely secure in protecting the privacy of data.
  Moreover, since the complexity of AE algorithms is higher than SE, AE has a higher security level but consumes more delay and computing power than SE. In practice, AE and SE are usually used together; for example, SE is used to encrypt large-scale raw data while AE is used to encrypt the small-size symmetric key. This can produce an efficient encryption method.
- *Two-Factor Authentication (TFA)*: This refers to applying two distinct ways to audit who is signing transactions/requests to be triggered. In worst cases, the passwords stored offline can be stolen if users' devices are compromised in undetectable offline ways, or exposed unintentionally in non-technical manners. In order to prevent these attacks, two-factor authentication requires the second factor to co-sign transactions/requests with private keys. This requires users to request another cryptographic key stored in the remote servers or dynamic codes generated by the servers in real-time through uploading users' biometric information such as fingerprint, facial recognition, and so on. The online cryptographic key/dynamic code and offline private key are then used to generate multi-signatures for transactions/requests.

6.3 Resistance to Miners' Collusion Attack

Compared to the attacks and threats shown in TABLE 1 and 2, the miners' collusion attack is higher-level, deriving many lower-level attacks (e.g., double-spending attacks). Generally, all attacks can be successful only if the attack results (e.g., conflicting transactions) are not detected during block validations. Hence, block validation is paramount for the security of the blockchain system. In typical consortium blockchains, compromised miners can easily collude to accept invalid (or reject valid) blocks due to

TABLE 1: STRIDE-Based Attacks and Threats Identification and Categorization in Our Proposed Framework

ENTITY	STRIDE	Attacks: Threats Description	Countermeasure					Security†
			DS*	SE/AE*	TFA*	Others	Ref.	
CBN*	S	*Impersonation Attack*: Intruders impersonate CBNs for sending messages to the blockchain system or users.	✔	✔	✔	Strict eligibility check of CBNs makes them trustable and verifiable by the public.	[48]	VH
	T	*Transaction Modification*: Compromised CBNs alter transactions before forwarding them to miners.	✔	✔	✔	—	[49]	VH
		Confirmation Manipulation: Intruders/compromised CBNs alter transaction results (succeed/fail) or create a fake confirmation sent to users.	✗	✗	✗	Similar to simplified payment verification (SPV) to check if a transaction exists in the blockchain by the on-chain block headers.	[50]	Low
	R	*Disregard Block Validation*: A new block cannot be accepted if most CBNs do not verify the block intentionally, which is unpredictable in a short time.	✗	✗	✗	Lightweight nodes verifying blocks with CBNs increase the number of block validators and reduce the impact of CBNs on block validations.	TP*	High
		Double Spending: Compromised users submit conflict transactions spending a coin more than once via different miners with probabilistic-finality consensus protocols.	✗	✗	✗	*Elaborated in Section 6.3*	TP*	VH
	I	*Linkage Attack*: Compromised CBNs link transactions of a user to infer its identity and even location if the user always uploads transactions via the same CBN.	✗	✗	✗	Users randomly select CBNs to reduce the link frequency (inapplicable to static devices of users).	—	Med
	D	*Flood Request*: Intruders submit numerous requests to a CBN for exhausting its computing/communication resource.	✗	✗	✗	§ Machine-learning based Intrusion Detection Systems with features of request timestamps; § Distributed feature of blockchain.	[51]	VH
		Disregard Request: Compromised CBNs ignore requests of users intentionally.	✗	✗	✗	Distributed feature of blockchain.	—	VH
	E	*CBNs' Collusion Attack*: Same as the miners' collusion attack with probabilistic-finality consensus protocols.	✗	✗	✗	*Elaborated in Section 6.3*	TP*	VH
		Voting Collusion: Users with high stakes collude with miners to select malicious block producers in DPoS.	✗	✗	✗	Replace stake-based with reputation-based voting.	[52]	Med
Block Producer	S	*Impersonation Attack*: Intruders impersonate the block producer to generate incorrect blocks and attack the system.	✔	✔	✔	Strict eligibility check of block producers/CBNs makes them trustable and verifiable by the public.	[48]	VH
	T	*Man-in-the-Middle Attack*: Intruders modify new blocks delivered from block producers to miners.	✔	✔	✔	§ Block validators can request a new block from other miners in absolute-finality consensus protocols; § The manipulated block is disregarded directly and abortive in probabilistic-finality consensus protocols. (*Elaborate in Step2 of Section 3.2*)	TP*	VH
	R	*Unfair Transaction Selection*: Selfish block producers always select transactions with high fee and ignore the rest.	✗	✗	✗	Incentive-provision design based on the weighted counting sort algorithm for transaction ordering by taking multiple features of transactions into consideration.	[53]	High
		Miners' Collusion Attack: Compromised block producer colludes with miners to add invalid blocks to the blockchain.	✗	✗	✗	*Elaborated in Section 6.3*	TP*	VH
	I	*Eavesdropping and Inference Attack*: Intruders steal block information during its propagation to miners, and try to recognize users' identity based on the transaction information.	✗	✔	✗	§ Anonymous with public keys and digital certificates; § Securely store users' transactions to avoid large-scale information leakage.	—	High
	D	*Delay Block Generation*: Compromised/Malfunctioning block producers do not produce new blocks on its duty within the time threshold.	✗	✗	✗	Vote out the straggling/malfunctioning nodes.	[54]	Med
		Unfair Transaction Selection: As aforementioned.	✗	✗	✗	Incentive-provision design based on weighted counting sort algorithm.	[53]	High
	E	*Miners' Collusion Attack*: Compromised miners collude to become a powerful node towards manipulate block validations.	✗	✗	✗	*Elaborated in Section 6.3*	TP*	VH
Miner	S	*Impersonation Attack*: Intruders impersonate miners for sending messages to the blockchain system/lightweight nodes.	✔	✔	✔	Strict eligibility check of CBNs makes them trustable and verifiable by the public.	[48]	VH
	T	*Block/World State/Validation Result Alteration*: Compromised miners/intruders alter the blocks/World State sent to lightweight nodes for block validations. On the other hand, the miners can alter the validation results of local lightweight nodes sent to the other miners for mutual authentication.	✔	✗	✔	§ Lightweight nodes request the block from other miners by submitting the block header if the received block is incomplete. (*Elaborated in Step3 of Subsection 3.2*) § Miners get no rewards from an invalid validation result.	TP*	VH
	R	*Local Collusion Attack*: Compromised miners only select specific lightweight nodes for verifying blocks towards conducting local collusion attacks with lightweight nodes.	✗	✗	✗	§ Miners have no knowledge on lightweight nodes in other pools (*Elaborated in Section 6.3*); § Avoid a miner/pool to control most lightweight nodes; § Reputation evaluation on lightweight nodes.	TP*	VH
	I	*Eavesdropping Attack*: The same as that of block producer.	✗	✔	✗	Anonymous with public keys and digital certificates.	—	High
	D	*Disregard Joining Request*: Compromised miners intentionally ignore joining requests of lightweight nodes towards increasing the possibility to conduct local collusion attacks.	✗	✗	✗	§ Similar to local collusion attack; § Rational miners recruit lightweight nodes for rewards.	TP*	VH
		Disregard Validation Result: Compromised miners screen the validation results of local lightweight nodes to select those aligned with its intention of conducting further attacks.	✗	✗	✗	A verifiable joining proof is assigned to the selected lightweight nodes and thus, they can submit their validation result to other miners with the proof. (*Elaborated in Step 3 of Subsection 3.2*)	TP*	VH
	E	*Tremendous Validation Pool*: Some miners recruit most of the lightweight nodes in the system and, hence, increase the probability to control the block validation process.	✗	✗	✗	Optimize trade-off between the block validation delay and distribution of lightweight nodes in each pool given the transaction fee of users. Though [11] considered more lightweight nodes lead to the securer validation pool, the imbalance distribution of lightweight nodes in each pool can reduce the security of block validation (i.e., a pool controlling most lightweight nodes can increase the likelihood to manipulate the block validation.).	FW*	VH

TABLE 2: (*Continue.*) STRIDE-Based Attacks and Threats Identification and Categorization in Our Proposed Framework

ENTITY	STRIDE	Attacks: Threats Description	Countermeasure					Security†
			DS*	SE/AE*	TFA*	Others	Ref.	
Lightweight Node	S	*Sybil Attack*: Malicious lightweight node duplicates identities for block validation in order to obtain multiple rewards.	✗	✗	✗	We consider one-hop transmission between lightweight nodes and local miners. The attack can be detected if the miner receives multiple validation results from a device.	TP*	VH
	R	*Eclipse Attack*: Intruders block messages sent by others but local miners/intruders to the selected lightweight nodes. In this way, the local miner/intruders can alter the block/World State to be verified by lightweight nodes that are trapped into rejecting a valid block.	✔	✗	✗	Tampering block/World State breaches the integrity of block header (verifiable by DS). Lightweight nodes can record errors via verifiable *Error Codes* signed by private keys. The verified block header is attached with the validation result for mutual authentication among miners, so that the attack is prevented with the invalid validation result if the block header is inconsistent with others.	TP*	VH
	I	*Eavesdropping Attack/Man-in-the-Middle Attack*: In addition to the same eavesdropping attack happening to block producers/miners, lightweight nodes itself can be compromised to recognize users' identity based on the transaction information.	✗	✔	✗	§ Anonymous with public keys and digital certificates. § Insufficient data to infer users' identity.	[1]	VH
	D	Flood Request: Intruders send numerous requests to a lightweight node to exhaust its computing and communication resources.	✗	✗	✗	As only miners can send blocks to lightweight nodes, this attack can be easily prevented at lightweight nodes by ignoring other requests or blocks sent by other senders.	TP*	VH
	E	*Local Collusion Attack*: Compromised lightweight nodes collude with the local miner to produce wrong validation results.	✗	✗	✗	The same countermeasure against the local collusion attack for miners.	TP*	VH

* Abbreviation in Table: CBN - Consortium Blockchain Node; DS - Digital Signature; SE/AE - Symmetric/Asymmetric Encryption; TFA - Two-Factor Authentication; TP - This Paper; FW - Future Work.
† Security Level: Very Low (VL)/Low/Medium (Med)/High/Very High (VH); ✔- Applied; ✗- Not Applied.

the small number of CBNs and, thus, miners in the systems. For example, they produce a positive validation result even if they find conflicting transactions in one or more blocks. In particular, in DPoS, users select a smaller number of miners for producing and verifying blocks in a fixed period of time, which can improve the blockchain throughput but make the system more vulnerable to the miners' collusion attack. Moreover, provided a block can be valid if more than 51% of miners accept it, the small-scale collusions in the system can only destabilise and delay the block validation process. Yet, large-scale collusions can crash the blockchain systems when the number of compromised miners exceeds 51%. In the long term, the typical consortium blockchains are vulnerable to large-scale collusion attacks and must take action in advance.

Countermeasure: Intuitively, more validators can effectively prohibit miners' collusion during block validations. To this end, numerous lightweight nodes are recruited in our framework, acting as block validators to verify blocks together with typical miners. A block can be accepted only if most of the lightweight nodes' and miners' validation results are positive. This reduces the effect of malicious miners. If compromised miners intend to accept a false block, they must convince most of the lightweight nodes to join their malicious group. Nevertheless, this is impossible in our framework with the privacy setting, i.e., each miner has no knowledge about dynamic lightweight nodes in other pools before the process of mutual authentication. Specifically, the miners only exchange the local block validation delays of their pool instead of each lightweight node while solving the distributed optimisation problem, and can obtain the information of lightweight nodes in other pools while mutually authenticating their validation results. Therefore, the attackers are unable to conduct the process of temptation since it has to be done before block validations. Moreover, the lightweight nodes in different rounds of block validations can be dynamic, which prevents linkage attacks when the attackers obtain the information of lightweight nodes in the last round.

6.4 Security Performance of the Proposed Framework

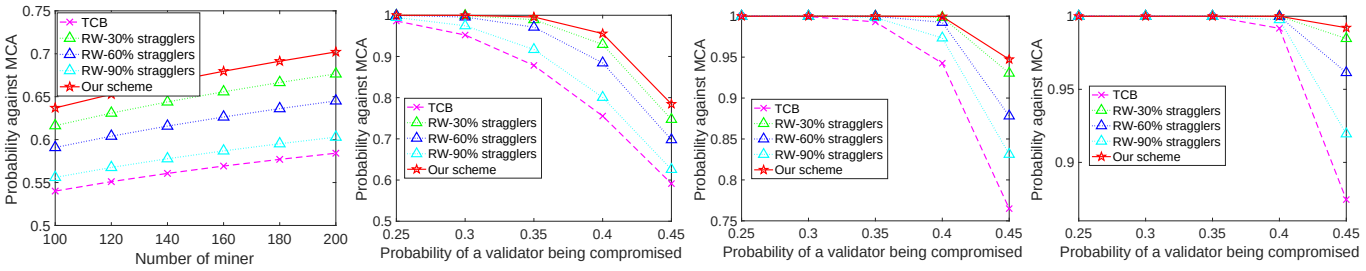
Here, we model and evaluate the security of the proposed framework against the miners' collusion attack, i.e., satisfying fault tolerance degree, as a random sampling problem with incompatible outcomes; a validator can be either honest or compromised [55]. The existing schemes are also introduced for comparison:

- *Typical consortium blockchain (TCB)*: Only miners act as block validators.
- *Related work in [11] (RW) with different percentages of stragglers*: the recruited lightweight nodes may be unable to return validation results in time due to their scarce resources (not considered when selecting them). We assume both node types (i.e., compromised/honest) of lightweight nodes have the same straggling probability. Besides, we evaluate the performance of the RW with 30%, 60% and 90% of straggling nodes.

Unlike [56] strongly assuming the probability of a block validator being compromised as $p \in [0.1, 0.3]$, we set $p \in [0.25, 0.5]$ as some lightweight nodes are highly dynamic (i.e., join randomly) and untrusted (i.e., $p = 0.5$). Besides, a secure block system should satisfy that the maximal number of compromised validators should be less than the majority of $\mathcal{D} = \lfloor \frac{\mathcal{I}+\mathcal{A}-1}{2} \rfloor$, where $\mathcal{I} = \sum_{i=1}^{\mathcal{A}} n_i$ is the number of lightweight nodes in the system. Therefore, the probability of the proposed framework against the miners' collusion attack can be calculated by using the cumulative binomial distribution given by $P = \sum_{d=0}^{\mathcal{D}} \binom{\mathcal{I}+\mathcal{A}}{d} p^d (1-p)^{\mathcal{I}+\mathcal{A}-d}$. Based on this equation, we compare the security of our framework with existing schemes, as shown in Fig. 3a - Fig. 3d. In Fig. 3a, we set $p = 0.49$, and we can find that the lightweight nodes-involved consortium blockchain systems are more robust than the typical consortium blockchain. Furthermore, the security of our framework is further enhanced with the proposed computing power management scheme, compared to the RW with different percentages of stragglers. The reason is that with straggling lightweight nodes, the number of effective validators in the system of the RW are reduced. Thus, our scheme can achieve the best security performance among the existing schemes.

In Fig. 3a, lightweight nodes are considered highly-untrusted (i.e., $p = 0.49$). In contrast, we test the framework performance in different scales of consortium blockchain systems (i.e., small, medium and large) with different trust degrees of block validators (i.e., p), as shown in Fig. 3b - Fig. 3d. The numbers of miners in the small-, medium- and large-scale systems are $(0, 20]$, $(20, 100]$, and $(100, \infty)$, respectively [57]. In these figures, our framework can ensure 100% of security in all scales of systems if $p \leq 35\%$, being the best among the existing works. When the block validators are more untrusted, the consortium blockchain systems are

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2023.3343839



(a) $p = 0.49$ (b) 20 miners (small) (c) 70 miners (medium) (d) 150 miners (large)

Fig. 3: Security of block validation against miners' collusion attacks (MCA) with (a) different numbers of miner with the probability of a validator being compromised $p = 0.49$, as well as different p in a small-, medium-, and large-scale consortium blockchain system.

more vulnerable, especially in the small-scale system, in which the system security degrades rapidly with the untrusted degree of block validators. However, the consortium blockchain systems are more resilient to the untrusted nodes with the increasing number of block validators, i.e., the large-scale consortium blockchain system is more secure than the small. More importantly, our scheme performs much better than the existing works in all cases. The reason is that the lightweight nodes acting as additional block validators can mitigate the collusions among the malicious miners, and our scheme can also ensure all lightweight nodes return their validation results in time.

7 Numerical Results

Here, we conducts extensive experiments on a practical blockchain platform to test the blockchain throughput with the framework and computing power management scheme, and derive the lightweight nodes selection strategy with their optimal computing powers.

7.1 Description of the Practical Test Platform

To evaluate the performance of the computing power management scheme, we build a practical consortium blockchain test platform based on the proposed framework. We use the same kernel as the Fisco Bcos (FB) platform, which is an open-source consortium blockchain financial platform [8]. Except for interactions between miners and lightweight nodes, the functions of miners in our platform are aligned with FB, which provides optional consensus protocols of PBFT and Raft (Replication and Fault Tolerant). Here, we adopt the PBFT. Then, we adopt the *secp256k1* as the tool for AE and digital signature, *AES-256* for SE, and *keccak256* for Hash function, which are optional and provided by the FB platform via Java SDK [58]. The algorithm details and codes can be found in [58]. Besides, our platform is enabled by Docker, and each miner is running in an independent and configurable container deployed in a powerful server with the configuration shown in TABLE 3. Based on container technology, nodes are fully independent of each other, which is the same as them running on different devices. Specifically, we deploy 200 CBNs in our platform, which can be selected as miners. Note that a large-scale consortium blockchain generally includes more than 100 CBNs [57]. In the experiments, the number of miners participating in the consensus process varies from 100 to 200. Furthermore, we deploy 2000 lightweight nodes to be selected for block validations (10 lightweight nodes can be selected by each miner) by using containers. In the experiment, if a lightweight node $\mathcal{L}_i^j$ is selected by a miner $\mathcal{M}_i$, an acceptance proof of $\mathcal{L}_i^j$ is produced by $\mathcal{M}_i$ with its private key SK_i, denoted as $\mathrm{Sign}(\mathrm{PK}_{ij}|\mathrm{Cert}_{ij}|\mathrm{VCP}, \mathrm{SK}_i)$, in which $\mathrm{Sign}(\mathrm{data}, \mathrm{SK})$ is the digital signature algorithm to sign data with a private key SK by its owner. PK_{ij} and Cert_{ij} are, respectively, the public key and digital certificate of $\mathcal{L}_i^j$ while VCP indicates the valid consensus period of the proof. Note that the acceptance proof is verifiable based on the public key of $\mathcal{M}_i$ and the Hash function. Regarding *Error Codes*, we produce a table shared by each validator to present the possible errors during block validation, each of which is assigned a unique error code. Each validator generates and signs the error code by $\mathrm{Sign}(\mathrm{E}|\mathrm{Code}|\mathrm{ID}, \mathrm{SK}_v)$ with its private key SK_v, in which E and Code are, respectively, the uniform error sign and unique code of an error. ID is the transaction or block ID (i.e., hash value).

Enabled by the container technology and kernel of the FB platform, we can configure maximal computing resources and transmission rate for each container of miners and lightweight nodes. To evaluate the proposed scheme in a practical scenario with real parameters, we simulate a consortium blockchain-enabled vehicle-to-grid network, where local aggregators (e.g., static base stations) and charging/discharging vehicles act as miners and lightweight nodes, respectively [59]. Here, we adopt the popular Tesla electric vehicles embedded with three 4-core Cortex-A72 CPUs operating at 2.2GHz [60]. In practice, vehicles possibly have different computing tasks to finish, and only contribute part of their computing powers for block validations. Thus, the available computing power of lightweight nodes ranges from 2.2GHz (1 core) to 8.8GHz (4 cores). Base stations have evolved from forwarding messages between devices to support edge computing by embedding with edge servers providing computing powers. Compared to vehicles, the computing power of base stations is much more powerful and scalable. Here, we consider the flagship Xeon server designed by Intel for base stations, which Xeon D-2700 provides 4 to 20 cores CPU running at 3.5GHz [61]. Thus, the available computing power of miners ranges from 35GHz (10 cores) to 56GHz (16 cores).

Furthermore, Tesla vehicles support wireless communication modes such as Bluetooth, Wi-Fi, and Cellular (5G). We consider 5G for communication between vehicles and base stations (unsupportable by Bluetooth and Wi-Fi due to their short communication range). Now, most electronic vehicles of different brands have supported 5G. To simulate realistic communication channels, we consider the urban environment with its practical settings, which has numerous electronic vehicles [62]. The simulation parameters of the transmission model are listed in TABLE 3. All the parameters are set according to IMT-2020 specifications established by the International Telecommunication Union (ITU) [63].

Finally, to evaluate the performance of the proposed computing resource management scheme on our established platform, we introduce a test tool (Java JDK demo) devised by the FB platform, which can examine the best performance of the system (i.e., upper bound). Using this tool, we can trigger a number of transactions automatically and simultaneously by implementing the devised smart contract and observe the consensus performance, such as blockchain throughput, energy consumed, CPU usage, and so on.

TABLE 3: Configuration of Blockchain Test Platform and Parameters of Wireless Channels between Bases Stations (BSs) and Vehicles in Urban Environment

	Parameter	Value	Parameter	Value	Parameter	Value
Platform Hardware	Architecture	x86_64	OS	Ubuntu 20.04.5 LTS	Total number of CPUs	128
	CPU	Intel(R) Xeon(R) Platinum 8352Y CPU @ 2.20GHz (32-Core CPU)			Total RAM	256GB
Wireless Channel in Urban Environment [63]	Path loss	$\mathrm{PL} = 20\log_{10}(\mathrm{f}_c) + 30\log_{10}(\mathrm{d}_{3\mathrm{d}}) + 32.4$, $\mathrm{f}_c \in$ [400,6000] MHz is carrier frequency and $\mathrm{d}_{3\mathrm{d}}$ is the 3D distance between a BS and a vehicle			Multipath fading Rician K factor (K)	μ_K=9, σ_K=3.5
	Shadow fading	Log-normal shadow fading with a standard $\sigma_{\mathrm{SF}} = 7.8$dB for outdoor/indoor vehicles				
	Building penetration loss	External Wall: $\mathrm{L}_b = 5 + 4 * \mathrm{f}_c$; Indoor: $0.5\mathrm{d}_{\mathrm{2D\text{-}in}}$, $\mathrm{d}_{\mathrm{2D\text{-}in}}$ is the indoor 2D distance; Standard Deviation: σ_{P}=0				
	Vehicle penetration loss	PL_v=PL+N(μ, σ_P^2), PL is the basic path loss, μ=9, σ_{P}=5			Average building height	[5,50] m
	BSs/Vehicles antenna gain	8 dBi / 5 dBi	BSs/Vehicles noise figure	7 dB / 10dB	Distance between vehicles and BSs	[50,5000] m
	Thermal noise density	-174 dBm/Hz	Bandwidth of a vehicle	[0.5,100] MHz		

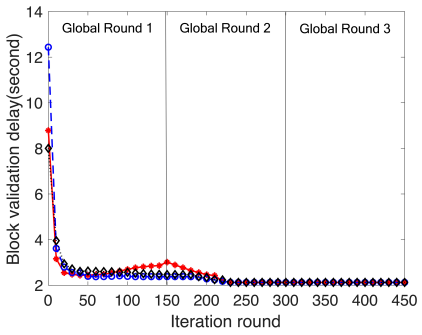


Fig. 4: Convergence of iterative solution with different initial points.

In our experiment, we trigger 20000 transactions each time (the peak transactions per second in the FB platform with PBFT [64]) while the block size is set to 10Mb unless it is specified otherwise.

7.2 Convergence of Iterative Solution

We evaluate the convergence of the iterative distributed solution regarding *Sub-problem 1* and *Sub-problem 2*. In this experiment, we deploy 150 miners with 10 lightweight nodes in each validation pool. In a global round, each miner optimizes their local variables (i.e., $(\mathbf{f_i^v}, \mathbf{f_i^a})$) independently and parallelly. After the local optimization, the miners share the block validation delay of their validation pools with each other. Given the block validation delays of all pools (i.e., $T_i^L, \forall i$), each $\mathcal{M}_i$ starts the next global round of optimization until all variables (i.e., $(\mathbf{f_i^v}, \mathbf{f_i^a}), \forall i$) converge. The convergence of the solution can also be denoted as the consensus of the block validation delay of the system reached by each miner. From Fig. 4, given different initial computing powers of miners and lightweight nodes, the solution can converge fast within three global rounds. Specifically, the points within each global round are the block validation delay of the system computed by each miner after they optimize their local variables. The results prove that the solution can efficiently optimize the computing powers of miners and lightweight nodes with the small number of signalling.

7.3 Comparison on the Blockchain Throughput

For examining the blockchain throughput of our computing power management scheme, the comparable schemes are introduced:

- *Globally-optimal*: Exploiting the Global Traversal method to search through the strategy space for the optimal computing powers of miners and lightweight nodes for the block validation.
- *Typical consortium blockchain (TCB)*: Only miners verify the block while their computing powers are also optimized.
- *Related work in [11] (RW)*: The RW only considers the block propagation delays in block validations without optimizing computing powers of miners and lightweight nodes.
- *Random scheme*: Randomly initializing the computing powers of miners and lightweight nodes for the block validation. To avoid special cases, we run each experiment of the random scheme ten times and average out the results.

In Fig. 5a, the block validation delays of the schemes (except the random scheme) have various degrees of increase with the number of miners/validation pools. Compared to the other schemes, our scheme is almost equal to the globally-optimal. Though TCB has the least block validation delay, it is vulnerable with very low security level, as shown in Fig. 3a - Fig. 3d. In general, our scheme can reduce block validation delay incurred by the lightweight nodes and, thereby, increase blockchain throughput (shown in Fig. 5b) with the enhanced security of the system.

7.4 Impacts of Parameter Settings

Here, we examine the blockchain throughput with different parameter settings in order to evaluate the applicability of the proposed computing power management scheme. In Fig. 5c, with different workloads to verify the unit size of blocks (i.e., ν), the blockchain throughputs of our scheme are always close to the optimal values and greatly exceed that of the RW and the random scheme. Though the TCB has the highest blockchain throughput, it compromises the security as aforementioned.

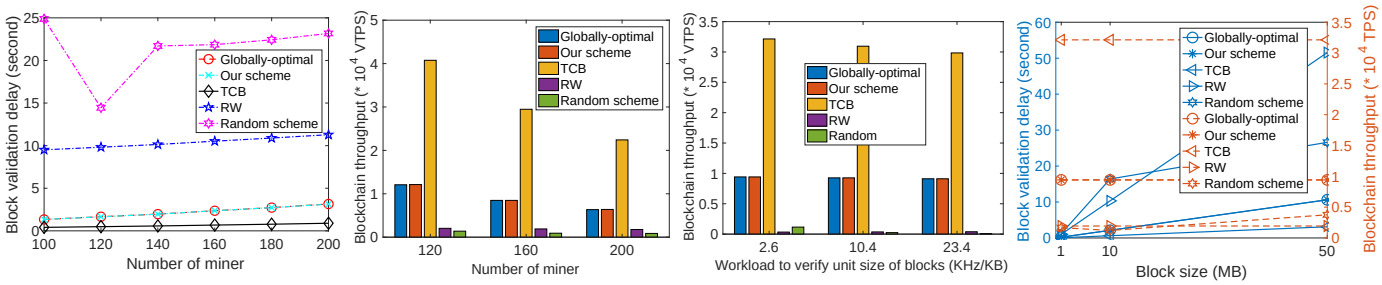
Another experiment is conducted on the effect of block sizes (i.e., S_{blk}) on the blockchain throughput/block validation delay. In Fig. 5d, the block validation delays of different schemes increase with the block sizes. Yet, our scheme increases at a much slower rate than the RW and random scheme. More importantly, with distinct block sizes, our scheme has the nearly-equal block validation delay and blockchain throughput as the optimal values. In this figure, the blockchain throughputs of the schemes, except the random scheme, also slightly increase with the block sizes.

7.5 Selection of Lightweight Nodes

During selection, the one-request one-response rule is used to ease the communication overhead of lightweight nodes. Miners initially collect the amount of computing power that lightweight nodes plan to contribute, and then respond with selection results based on the computing power management scheme. With the management scheme, the optimal computing powers of lightweight nodes for block validations can be obtained. This facilitates the system to select lightweight nodes that maximize the blockchain throughput with the smallest redundant computing powers according to the following process. In cases that the block validation delay exceeds the threshold of the system after optimization, the slowest lightweight nodes in each validation pool with zero and almost zero redundant computing powers are excluded from the validation pool. Once the block validation delay satisfies the threshold of the system, the miners select lightweight nodes with less redundant computing powers based on the quotas allocated by the system.

Here are the system settings: the maximal consensus delay is 3s, and each of 150 miners selects lightweight nodes from 10 candidates in their validation pools with the aforementioned selection method. The performance of the selection of lightweight nodes can be assessed in terms of 1) the number of lightweight nodes that return validation results successfully, 2) the block validation delay, 3) the total redundant computing power, and 4) the probability density function (PDF) of available computing powers of selected lightweight nodes. In term 2), the block validation

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2023.3343839



(a) Impact of numbers of miners (b) Impact of numbers of miners (c) Impact of ν (d) Impact of S_{blk}.

Fig. 5: Blockchain throughput/block validation delay with different numbers of miners, workload to verify unit size of block ν and block size S_{blk}.

delay is either less than 3s (all selected lightweight nodes return validation results successfully) or equal to 3s (otherwise). To better present the performance about 1), 2), and 3), we introduce two auxiliary metrics called block validation delay per lightweight node BPL=T/N_l and redundant computing power per lightweight node RPL=f^r/N_l. T and f^r are the block validation delay and total redundant computing power of the system, respectively. N_l is the number of lightweight nodes returning validation results. Note that BPL is an auxiliary metric here without any practical meaning.

With the auxiliary metrics, Fig. 6a presents the RPL of the system in the block validation. In this figure, the RPLs of different schemes increase with the number of selected lightweight nodes in each validation pool. This is due to the fact that the miners have no choice but to select the resource-rich lightweight nodes when the quota is large. Yet, the RPL of the RW is much larger than our scheme that is almost equal to the globally-optimal. The reason is that without optimizing the computing powers of lightweight nodes, [11] randomly selects lightweight nodes, leading to the considerable redundant computing powers.

Given the maximal consensus delay of the system, Fig. 6b can reflect the number of stragglers of the system by comparing the result of schemes with the benchmark. The benchmark is given by Ben=3s/N_{sl}, where N_{sl} is the number of selected lightweight nodes expected to return validation results. If BPL $<$ Ben, then all lightweight nodes return validation results successfully. If BPL $>$ Ben, then some lightweight nodes are straggling. Besides, the higher value (BPL-Ben) has, the more lightweight nodes are straggling. From Fig. 6b, we can find that our scheme can ensure all lightweight nodes return their validation results before timeout. However, this can hardly be achieved by the RW.

To observe the distribution of selected lightweight nodes, in Fig. 6c, all schemes are reluctant to select the resource-constraint lightweight nodes to increase the blockchain throughput. As for resource-rich lightweight nodes, our scheme intends to select those with less computing power to reduce the redundant computing power and cost while the RW randomly selects lightweight nodes with enormous redundant computing power. Note that our scheme selects the same lightweight nodes with the globally-optimal.

8 Discussion

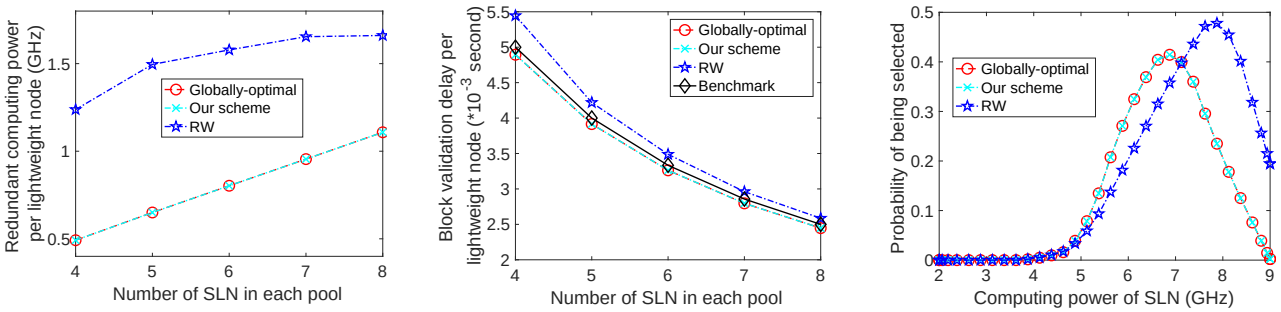
Merits of the distributed selection method: 1) *Small signalling*: The method to select lightweight nodes is developed with the proposed computing power management scheme. In section 5.3, we devise a decentralized solution to optimize the utility function and maximize the blockchain throughput of the system, in which miners only exchange block validation delays of local validation pools during optimization. The decentralized solution requires a far smaller amount of signalling compared to the centralized method. This can avoid network congestion and contribute to resource-saving blockchain systems. 2) *Privacy Protection*: During the optimization, information of lightweight nodes is only used in local validation pools instead of sharing in the blockchain system, which shrinks the attack surface of the information. Furthermore, after optimizing the computing powers of miners and lightweight nodes, the selection method of lightweight nodes is implemented in a distributed manner through the iterative method. During the selection, each miner selects local lightweight nodes without having knowledge about lightweight nodes in other pools.

Applicability of the developed scheme: Here, potential application scenarios of the proposed framework and computing power management scheme are discussed based on the above experiment. The security analysis proves that our framework performs best among the existing works with different numbers of miners in the system, which is applicable to popular large-scale consortium blockchain systems such as Hyperledger Fabric and Fisco Bcos and improve the block validation security. When the probability of a block validator being compromised is less than 35%, our framework can ensure 100% of security for block validation in the small-, medium-, and large-scale systems. This indicates that the scheme can be applied in many small-scale consortium blockchain systems such as Contour Blockchain with 12 banks as miners in the system, Batavia Bank Consortia with only 5 members in their ecosystem [65], and medium-scale consortium blockchain systems such as Pharmaledger with 30 members [66].

In section 7.4, we examine the computing power management scheme with different parameter settings, including the workload to verify the unit block size and block size. Based on the experiment results, our scheme can be applied to the block validations of various systems (e.g., consortium blockchain enabled resource sharing [67], medical data sharing [68], energy trading [69], etc.) requiring different workloads to verify the unit block size or having different block sizes, with enhanced security and higher throughput. The way to verify different kinds of transactions is out of the scope here and will be discussed in future works.

9 Conclusion

We studied the security and throughput of consortium blockchains. First, we improved the security of typical consortium blockchains by devising a general and secure block validation framework involving lightweight validators, and investigated the threats and attacks in the framework with the countermeasures and security level evaluation. The numerical results showed that our framework has the highest security in the existing works. Then, we proposed an effective method to select lightweight nodes based on their optimal computing powers obtained from the proposed computing power management scheme. Extensive experiments showed that this scheme achieved a higher blockchain throughput compared to the existing schemes. Besides, the scheme can effectively



(a) RPL (b) BPL (c) Probability Distribution(PDF)

Fig. 6: (a) Redundant computing power (RPL), (b) Block validation delay (BPL) per selected lightweight node (SLN), and (c) PDF of SLN.

facilitate the selection of lightweight nodes and reduce the redundant computing power of selected lightweight nodes while maximizing the blockchain throughput. Finally, we discussed the potential application scenarios based on the proposed framework and lightweight nodes selection method with the advantage of saving the signalling and protecting the privacy of lightweight nodes. In future works, the detection of malicious miners and lightweight nodes and the evaluation of their reputations will be investigated. Application of the proposed framework and scheme in some specific areas will be researched in-depth.

REFERENCES

[1] W. Ni *et al.*, "Healchain: A decentralized data management system for mobile healthcare using consortium blockchain," in *IEEE Chin. Control Conf.*, 2019, pp. 6333–6338.

[2] Z. Li, *et al.*, "Consortium blockchain for secure energy trading in industrial internet of things," *IEEE Trans. Ind. Inform.*, vol. 14, no. 8, pp. 3690–3700, 2017.

[3] J. Kang *et al.*, "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE IoT J.*, vol. 6, no. 3, pp. 4660–4670, 2018.

[4] Z. Zheng *et al.*, "An overview of blockchain technology: Architecture, consensus, and future trends," in *IEEE BigData Congr.*, 2017, pp. 557–564.

[5] S. King *et al.*, "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake," *self-published paper, August*, vol. 19, p. 1, 2012.

[6] D. Mingxiao *et al.*, "A review on consensus algorithm of blockchain," in *IEEE Int. Conf. Syst., man, Cybern.*, 2017, pp. 2567–2572.

[7] Ibm. [online]. https://rb.gy/lxokzo. [Accessed on Feb. 25, 2023].

[8] Fisco. [online]. http://fisco-bcos.org/. [Accessed on Feb. 25, 2023].

[9] M. Al-Bassam *et al.*, "Fraud proofs: Maximising light client security and scaling blockchains with dishonest majorities," *arXiv preprint arXiv:1809.09044*, vol. 160, 2018.

[10] C. Decker *et al.*, "Information propagation in the bitcoin network," in *IEEE P2P Process.*, 2013, pp. 1–10.

[11] J. Kang *et al.*, "Incentivizing consensus propagation in proof-of-stake based consortium blockchain networks," *IEEE Wireless Commun. Lett.*, vol. 8, no. 1, pp. 157–160, 2018.

[12] Y. Jiang *et al.*, "6g oriented blockchain based internet of things data sharing and storage mechanism," *J. Commun.*, vol. 41, no. 10, p. 48, 2020.

[13] F. A. Khan *et al.*, "Rift: A high-performance consensus algorithm for consortium blockchain," *Int. J. Recent Technol. Eng.*, pp. 989–997, 2019.

[14] S. Cao *et al.*, "Cover: Collaborative light-node-only verification and data availability for blockchains," in *IEEE Int. Conf. Blockchain*, 2020, pp. 45–52.

[15] S. Zhang *et al.*, "Analysis of the main consensus protocols of blockchain," *ICT express*, vol. 6, no. 2, pp. 93–97, 2020.

[16] Q. Zhou *et al.*, "Solutions to scalability of blockchain: A survey," *IEEE Access*, vol. 8, pp. 16 440–16 455, 2020.

[17] L. Yang *et al.*, "Secure off-chain payment in consortium blockchain system," in *Int. Conf. Netw. Appl.*, 2020, pp. 259–264.

[18] W. Yan *et al.*, "Pcbchain: Lightweight reconfigurable blockchain primitives for secure iot applications," *IEEE Trans. Very Large Scale Integration (VLSI) Syst.*, vol. 28, no. 10, pp. 2196–2209, 2020.

[19] R. Doku *et al.*, "Lightchain: On the lightweight blockchain for the internet-of-things," in *IEEE Int. Conf. Smart Comput.*, 2019, pp. 444–448.

[20] B. Seok *et al.*, "A lightweight hash-based blockchain architecture for industrial iot," *Applied Sci.*, vol. 9, no. 18, p. 3740, 2019.

[21] B. Ghimire *et al.*, "Sharding-enabled blockchain for software-defined internet of unmanned vehicles in the battlefield," *IEEE Netw.*, vol. 35, no. 1, pp. 101–107, 2021.

[22] G. Spathoulas *et al.*, "Security and privacy in the internet of things using blockchain technology," in *15th Int. Conf. Distrib. Comput. Sensor Syst.*, 2019, pp. 284–290.

[23] X. Yuan *et al.*, "Efficient byzantine consensus mechanism based on reputation in iot blockchain," *Wireless Commun. Mobile Comput.*, vol. 2021, 2021.

[24] N. Zhao *et al.*, "Coalition game-based computation resource allocation for wireless blockchain networks," *IEEE IoT J.*, vol. 6, no. 5, pp. 8507–8518, 2019.

[25] S. Seng *et al.*, "User matching on blockchain for computation offloading in ultra-dense wireless networks," *IEEE Trans. Netw. Sci. Eng.*, 2020.

[26] J. Zou *et al.*, "A proof-of-trust consensus protocol for enhancing accountability in crowdsourcing services," *IEEE Trans. Services Comput.*, vol. 12, no. 3, pp. 429–445, 2018.

[27] H. Afzaal *et al.*, "Formal modeling and verification of a blockchain-based crowdsourcing consensus protocol," *IEEE Access*, vol. 10, pp. 8163–8183, 2022.

[28] J. Yang *et al.*, "Compensation for power loss by a proof-of-stake consortium blockchain microgrid," *IEEE Trans. Ind. Inform.*, vol. 17, no. 5, pp. 3253–3262, 2020.

[29] W. Ni *et al.*, "Fast and secure consortium blockchains with lightweight block verifiers," in *Third Int. Conf. Blockchain Comput. Appl.*, 2021, pp. 11–18.

[30] N. Weiquan *et al.*, "Throughput-efficient blockchain for internet-of-vehicles," in *IEEE Globecom Workshops*, 2021, pp. 1–6.

[31] Delgado-Segura *et al.*, "Analysis of the bitcoin utxo set," in *Int. Conf. Financial Cryptography Data Secur.* Springer, 2018, pp. 78–91.

[32] G. Wood *et al.*, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[33] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, 2014.

[34] Diving into ethereum's world state. [online]. https://rb.gy/j7peiq. [Accessed on Feb. 25, 2023].

[35] Blockchain. [online]. http://www.ibm.com. [Accessed on Feb. 25, 2023].

[36] C. S. Lai *et al.*, "Blockchain applications in microgrid clusters," in *Smart Grids and Big Data Analytics for Smart Cities*. Springer, 2021, pp. 265–305.

[37] H. Yang *et al.*, "User-centric blockchain for industry 5.0 applications," in *IEEE Int. Conf. Commun. Workshops*, 2022, pp. 25–30.

[38] Y. Zhu *et al.*, "Blockchain-empowered decentralized storage in air-to-ground industrial networks," *IEEE Trans. Ind. Inform.*, vol. 15, no. 6, pp. 3593–3601, 2019.

[39] T. Huynh *et al.*, "Joint downlink and uplink interference management for device to device communication underlaying cellular networks," *IEEE Access*, vol. 4, pp. 4420–4430, 2016.

[40] Frontline convex optimization solvers [online]. https://www.solver.com/convex-optimization. [Accessed on Feb. 25, 2023].

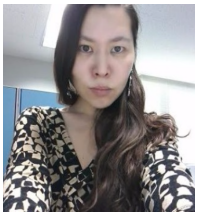
[41] A. Asheralieva *et al.*, "Optimizing age of information and security of the next-generation internet of everything systems," *IEEE IoT J.*, 2022.

[42] S. Boyd *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations Trends® Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[43] M. V. Afonso, J. M. Bioucas-Dias, and M. A. Figueiredo, "An augmented lagrangian approach to the constrained optimization formulation of

imaging inverse problems," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 681–695, 2010.
[44] G. Gordon and R. Tibshirani, "Karush-kuhn-tucker conditions," *Optim.*, vol. 10, no. 725/36, p. 725, 2012.
[45] S. Ahmadjee *et al.*, "A study on blockchain architecture design decisions and their security attacks and threats," *ACM Trans. Softw. Eng. Methodology (TOSEM)*, vol. 31, no. 2, pp. 1–45, 2022.
[46] C. Hebert *et al.*, "Secure blockchain in the enterprise: A methodology," *Pervasive Mobile Comput.*, vol. 59, p. 101038, 2019.
[47] Understanding AES-256 encryption. [online]. https://www.n-able.com/blog/aes-256-encryption-algorithm. [Accessed on Feb. 25, 2023].
[48] Consortium blockchain tutorial. [online]. https://rb.gy/y0aay9. [Accessed on Feb. 27, 2023].
[49] Electrum. [online]. https://electrum.readthedocs.io/en/latest/index.html. [Accessed on Feb. 25, 2023].
[50] I. Malakhov *et al.*, "On the use of proof-of-work in permissioned blockchains: Security and fairness," *IEEE Access*, vol. 10, pp. 1305–1316, 2021.
[51] S. Saha *et al.*, "Towards an optimal feature selection method for ai-based ddos detection system," in *IEEE 19th Annu. Consum. Commun. & Netw. Conf.*, 2022, pp. 425–428.
[52] J. Kang *et al.*, "Toward secure blockchain-enabled internet of vehicles: Optimizing consensus management using reputation and contract theory," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2906–2920, 2019.
[53] D. M. Doe *et al.*, "Incentive mechanism design for mitigating frontrunning and transaction reordering in decentralized exchanges," *IEEE Access*, 2023.
[54] S. M. S. Saad *et al.*, "Comparative review of the blockchain consensus algorithm between proof of stake (pos) and delegated proof of stake (dpos)," *in Int. J. Innovative Comput.*, vol. 10, no. 2, 2020.
[55] K.-K. Eleftherios *et al.*, "Omniledger: A secure, scale-out, decentralized ledger via sharding," in *IEEE Symp. Secur. Privacy*, 2018, pp. 583–598.
[56] L. Kai *et al.*, "Groupchain: Towards a scalable public blockchain in fog computing of iot services computing," *IEEE Trans. Services Comput.*, vol. 13, no. 2, pp. 252–262, 2020.
[57] A. Asheralieva *et al.*, "Throughput-efficient lagrange coded private blockchain for secured iot systems," *IEEE IoT J.*, vol. 8, no. 19, pp. 14 874–14 895, 2021.
[58] Cryptography module. [online]. https://shorturl.at/fovZ6. [Accessed on Nov. 20, 2023].
[59] M. Kim *et al.*, "A secure charging system for electric vehicles based on blockchain," *Sensors*, vol. 19, no. 13, p. 3028, 2019.
[60] Fsd chip-tesla. [online]. https://en.wikichip.org/wiki/tesla_(car_company)/fsd_chip. [Accessed on Feb. 25, 2023].
[61] Xeon server performance. [online]. https://rb.gy/gyodul. [Accessed on Feb. 25, 2023].
[62] A. Asheralieva *et al.*, "Optimal contract design for joint user association and intercell interference mitigation in heterogeneous lte-a networks with asymmetric information," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 5284–5300, 2016.
[63] ITU, "Guidelines for evaluation of radio interface technologies for imt-2020," Tech. Rep. Report ITU-R M.2412-0, Oct. 2017.
[64] Performance of fisco bcos. [online]. https://rb.gy/bfibgg. [Accessed on Feb. 25, 2023].
[65] 101 blockchains. [online]. https://101blockchains.com/blockchain-consortium/. [Accessed on Feb. 25, 2023].
[66] Pharmaledger member. [online]. https://pharmaledger.eu/about-us/members/. [Accessed on Feb. 25, 2023].
[67] S. Wang *et al.*, "Consortium blockchain for secure resource sharing in vehicular edge computing: A contract-based approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1189–1201, 2020.
[68] M. Wang *et al.*, "Medshare: A privacy-preserving medical data sharing system by using blockchain," *IEEE Trans. Services Comput.*, 2021.
[69] M. U. Hassan *et al.*, "Deal: Differentially private auction for blockchain-based microgrids energy trading," *IEEE Trans. Services Comput.*, vol. 13, no. 2, pp. 263–275, 2019.

Weiquan Ni is currently a Ph.D student in the joint Ph.D program of the Department of Science and Engineering, Southern University of Science and Technology, China and the WMG, University of Warwick, UK. His research interests mainly focus on blockchain, security and resource/data management/pricing in wireless communications and networks.



Alia Asheralieva obtained her Ph.D. from the University of Newcastle, Australia, in 2014. From 2015, she was with Graduate School of Information Science and Technology, Hokkaido University, Japan. From 2017, she was with Information Systems Technology and Design Pillar, Singapore University of Technology and Design. She is currently with the Department of Computer Science and Engineering, Southern University of Science & Technology, China. Her main research interests span many areas of communications and networking, including cloud/edge computing, blockchains, Internet of Things, optimization, game theory, and machine learning.

Jiawen Kang received the M.S. degree and the Ph.D. degree from the Guangdong University of Technology, China, in 2015 and 2018. He is currently a full professor at the Guangdong University of Technology. He was a postdoc at Nanyang Technological University from 2018 to 2021, Singapore. His research interests mainly focus on blockchain, security, and privacy protection in wireless communications and networking.

Zehui Xiong is currently an Assistant Professor in the Pillar of Information Systems Technology and Design, Singapore University of Technology and Design. He received the PhD degree in Nanyang Technological University, Singapore. He was the visiting scholar at Princeton University and University of Waterloo. His research interests include wireless communications, network games and economics, blockchain, and edge intelligence. He is now serving as the editor or guest editor for many leading journals including IEEE Transactions. He is the Founding Vice Chair of Special Interest Group on Wireless Blockchain Networks in IEEE Cognitive Networks Technical Committee.

Carsten Maple is the Principal Investigator of the NCSC-EPSRC Academic Centre of Excellence in Cyber Security Research at the University and Professor of Cyber Systems Engineering in WMG. He is also a co-investigator of the PETRAS National Centre of Excellence for IoT Systems Cybersecurity where he leads on Transport & Mobility. He is a Fellow of the Alan Turing Institute, the National Institute for Data Science and AI in the UK, where he is a principal investigator on a $5 million project developing trustworthy national identity to enable financial inclusion.

Xuetao Wei received the Ph.D. degree in computer science from the University of California, Riverside, CA, USA, in 2013. Since 2019, he has been an Associate Professor with the Southern University of Science and Technology, Shenzhen, China. He was an Assistant Professor and then promoted to an Associate Professor with the University of Cincinnati, OH, USA. His research interests include industrial metaverse, blockchain, and Internet of Things.