A Blockchain-Based Privacy-Preserving Scheme for Sealed-Bid Auction

Zijian Zhang, *Senior Member, IEEE*, Xin Lu, Meng Li, *Senior Member, IEEE*, Jincheng An, Yang Yu, Hao Yin, Liehuang Zhu, *Senior Member, IEEE*, Yong Liu, Jiamou Liu, and Bakh Khoussainov

***Abstract*—The sealed-bid auction enables bidders to secretly send their bids to the auctioneer, which compares all bids and publishes the winning one on the bid-opening day. This type of auction is friendly for protecting the bid privacy, and sufficiently fair for all bidders if the auctioneer acts faithfully. Unfortunately, the auctioneer may not always be trustworthy. The auctioneer has the ability to deliberately leak any bid information to a part of bidders for raising the final winning price based on the investigation. Meanwhile, the auctioneer can appoint any bidder as the winner, as long as the bidder accepts a higher winning price than the current highest bid. Since bidders cannot obtain any bid information from others, to the best of our knowledge, it is difficult to prevent bid leakage from the auctioneer, and support bidders to verify the bid comparison results without disclosing the winning bid, simultaneously. To alleviate these problems, we first construct a homomorphic encryption(HE)-based bid comparison circuit. All bidders can directly compute a cipher of the winning bid by using this circuit; hence, the winning bid does not need to be exposed to all bidders. Then, we propose a blockchain-based sealed-bid scheme (BSS) by integrating the circuit with commitment and zero-knowledge proof. The auctioneer only obtains the commitments of bids before the bid-opening day, and he has to prove that the winner's bid is the same as the plaintext of the bidders' computed cipher. Thus, the auctioneer can neither leak the bid information nor publish a higher winning price during in the auction. Detailed performance analysis shows that the computational complexity of BSS is linear with the binary length of bids.**

***Index Terms*—Sealed-bid auction, privacy, homomorphic encryption, zero-knowledge proof, commitment, blockchain.**

I. Introduction

THERE are three classic types of auctions. Namely, the English auction, the Dutch auction, and the sealed-bid auction. The first type uses the open ascending price strategy, while the second type uses the open descending price strategy. All the bids have to be shared among bidders for executing the strategies in the first two types. On the contrary, the third type does not require sharing bids among bidders, and the winning bid is secretly computed and then published by the auctioneer. Since the sealed-bid auction can have the same revenue as the first two types of auctions due to the revenue equivalence theorem [1], it is better than the first two types of auction regarding. However, the value of bids is exposed, once the auctioneer deliberately leaks them. The bid leakage enables auctioneer to raise the final winning price due to the investigation. For an instance, Ivanov and Nesterov [2] analysed 600,000 Russian sealed-bid auctions from 2014 to 2018 by machine learning algorithms. Approximately 9% of the auctions suffered bids leakage, and the winning bids increases 1.5% on average. Furthermore, the auctioneer can arbitrarily change the winner to any bidder who can accept a higher price than the highest bid on the bid-opening day, because bidders cannot obtain any bid information from others. In a word, the behaviours of the auctioneer can be malicious, once bids are obtained before the bid-opening day.

Intuitively, bidders can simply share commitments of their bids with the auctioneer and each other before the bid-opening day. If so, the auctioneer cannot leak bids or change the winner, because all bids are only shown by the bidders on the opening day. Every bidder can verify the winning price and check with the commitments without the help of auctioneer. However, bids are sensitive to bidders, because they convey rich personalised information. For example, Varian [3] quantified a bidder's willingness for payment in an auction due to her historical bids. Thus, the sealed-bid auction requires to prevent bid leakage from the auctioneer, and support bidders to verify the bid comparison results without disclosing any bids, simultaneously.

The above requirements are difficult to be met because the next four properties have to be maintained at the same time. First, all bids cannot be shared to the auctioneer before the bid-opening day. Second, all bids cannot be exposed to any bidders during in the auction. Third, all the bidders can publicly verify the bid comparison result that is published by the auctioneer. Finally, the whole sealed-bid scheme must be efficient in practice. Although the first property can usually be taken by the technique of bit commitment, to the best of our knowledge, the other three

Manuscript received 2 November 2021; revised 16 July 2023; accepted 10 January 2024. Date of publication 18 January 2024; date of current version 4 September 2024. This work was supported in part by National Natural Science Foundation of China (NSFC) under Grants 62172040, 62372149, and U23A20303. *(Corresponding author: Meng Li.)*

Zijian Zhang, Xin Lu, Yang Yu, and Liehuang Zhu are with the Beijing Institute of Technology, Beijing 100811, China (e-mail: zhangzijian@bit.edu.cn; luxin@bit.edu.cn; yuyangyy@bit.edu.cn; liehuangz@bit.edu.cn).

Hao Yin is with the Changsha Institute for Computing and Digital Economy, Peking University, Beijing 100871, China (e-mail: yinhao@cert.org.cn).

Meng Li is with the Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, School of Computer Science and Information Engineering, Anhui Province Key Laboratory of Industry Safety and Emergency Technology, and Intelligent Interconnected Systems Laboratory of Anhui Province, Hefei University of Technology, Hefei 230002, China (e-mail: mengli@hfut.edu.cn).

Jincheng An and Yong Liu are with the Qi An Xin Technology Group Inc, Beijing 100000, China (e-mail: anjincheng@qianxin.com; liuyong@qianxin.com).

Jiamou Liu and Bakh Khoussainov are with the University of Auckland, Auckland 92019, New Zealand (e-mail: jiamou.liu@auckland.ac.nz; bmk@cs.auckland.ac.nz).

Digital Object Identifier 10.1109/TDSC.2024.3353540

properties have not formed a perfect solution to take yet, especially when the winning bid has to be secret for bidders.

Secure Multi-Party Computation (SMC) was first leveraged with bit commitment to protect bids from being shared among bidders as much as possible. Franklin et al. [4] first designed a seal-bid auction scheme based on verifiable signature-sharing in 1996. In this scheme, only when the number of bidders meets the minimal requirement can the cipher of bids be decrypted for computing the winning bid. Therefore, only the winning bid has to be shared among bidders. Montenegro et al. [5] designed a SMC-based sealed-bid auction system with commitment. The auctioneer first defined the range of bids, and then queried from the highest price to the lowest price in their scheme. The bidder who offered the highest bid can reply the query and reveal her bid on the bid-opening day. Other bidders did not require to expose their bids but they had to prove that their bids were less than the winner's bid. However, typical SMC has to assume the auctioneer and bidders faithfully execute the protocol and the communication model was quite complex. These make the SMC-based sealed-bid protocols not efficient in reality.

In contrast, Homomorphic Encryption (HE) was applied with blockchain and Zero Knowledge (ZK) proof to design efficiently privacy-preserving sealed-bid schemes. The winner is computed based on the ciphers of bids and fewer rounds are required to accomplish the auction while the privacy is protected because only the winner's bid is allowed to be shared among the bidders. For examples, Blass and Kerschbaum [6] proposed a maliciously-secure blockchain-based auction scheme *Strain* where blockchain played a role of public communication channel. All encrypted bids are compared pair-by-pair via a homomorphic circuit and the winner is declared by using ZK proofs. A HE-based sealed-bid scheme [7] was designed based on a hash-based commitment on blockchain. The winner was proven by revealing the commitment and the corresponding bid. A deposit mechanism was prepared to enable the bidders to faithfully execute the protocol. The auctioneer can forfeit a bidder's deposit once that bidder was reported to have cheat behaviour in the auction.

However, to build a privacy-preserving auction scheme in practical scenarios, several technical challenges have to be faced and solved. The *challenge I* is *How to compute the winner correctly over ciphertexts while not exposing the bids during the auction*? The main operation of an auction scheme is comparison, but bids in the ciphertext state make the comparison operation difficult. How to accurately find the highest bidder among multiple bidders without revealing privacy is the challenge we want to solve. The privacy information includes not only the bids of the participants before the auction starts, but also the intermediate comparison results between arbitrary participants. More than this, we need to keep the winner's bid private because in most cases the price of the transaction is also a secret, which can be published as needed. The *challenge II* is *How to efficiently execute the auction protocol with multiple participants while the underlying secure circuits mechanically compare bids*? This is from the perspective of the auction protocol's time complexity, which should show a relatively steady increase with the scale of participants. SMC-based schemes usually have high communication complexity and cannot guarantee efficient execution of privacy auctions. The advanced HE-based scheme requires multiple rounds of pairwise comparisons based on the designed arithmetic circuits, and the auction response time is long when the participant size is large. Our scheme needs to design efficient comparison circuits that support large-scale bidding. The *challenge III* is *How to construct a secure auction protocol while allowing participants to adaptively quit during auction*? Traditional schemes generally require users to participate in the protocol progress until determining the winner. Since some users leave halfway may cause the protocol to fail, it is necessary to consider an additional punishment mechanism to restrain their behavior when designing such a system. In a real-world scenario, a user would leave the auction to do something else when found the bid was not competitive. Therefore, our protocol needs to support the fact that a bidder's choice to leave the auction even after losing competitiveness will not have an impact on subsequent comparisons. Existing auction schemes cannot overcome the mentioned challenges, and thus in this paper, we use the HE technique to construct a sealed-bid auction scheme that aims to solve the challenges. We summarise our contributions as follows.

- We construct a homomorphic encryption-based bid comparison circuit for generating a cipher of the winning bid based on all of the bid-ciphers. Every bidder can obtain the generated cipher without exposing any bid. Thus, the winning bid does not need to be exposed to all bidders.
- We propose a blockchain-based sealed-bid scheme by combining the HE-based bid comparison circuit with commitment and zero knowledge proof. The auctioneer proves to all bidders that the winner's bid-cipher has the same plaintext with the generated cipher. Therefore, the auctioneer's malicious behaviours, including bid leakage and raise the winning price, can be prevented without exposing any bids to bidders.
- We formally analyse the security, and theoretically and experimentally evaluate the performance of the proposed scheme.

Organisation: The rest of the paper is organised as follows: The related works are recalled in Section II. Section III briefly introduces preliminaries and Section IV gives the models. In Section V, we show the homomorphic circuit and the blockchain-based sealed-bid auction scheme. Then, in Section VI, we present the formal security proofs. The performance analysis is given in Section VII. Finally, the conclusion is drawn in Section VIII.

II. Related Works

In this section, we introduce the related works about efficiently privacy-preserving sealed-bid auctions, and game theoretical-based bid comparison mechanism towards concrete applications.

The cryptographic and privacy-preserving techniques have been widely used in sealed-bid auctions. The traditional Secure Multi-Party Computation (SMC) was first leveraged with commitment to make bids private to bidders and the auctioneer as

much as possible in the early works. For an instance, Franklin [4] proposed the first sealed-bid auction protocol. In this protocol, the decryption key is distributed by applying secret sharing scheme, and several auctioneers reconstruct the key and open the bids to determine the winner together. Montenegro et al. [5] also designed a SMC and commitment-based sealed-bid auction scheme. In their scheme, the auctioneer first publishes a specific range of acceptable bids. The bidders then encrypt their bids and broadcast the commitments of the bids. The auctioneer finally polls the winner by querying from the highest to the lowest in the range. The winner reveals the commitment while other bidders provide cryptographic proofs that their bids are smaller than the winning bid under the Quadratic Residuosity Assumption (QRA). Unfortunately, this kind of works has to assume the auctioneer and bidders faithfully execute the protocols, which is usually too ideal in reality. Furthermore, a public trusted communication channel among auctioneers and bidders is also necessary but difficult for this kind of schemes.

Homomorphic Encryption (HE) was considered with blockchain and Zero Knowledge (ZK) proof to construct practical privacy-preserving sealed-bid auctions. Kosha et al. presented a blockchain-based sealed-bid auction framework, named Hawk [8]. A programmer can easily make privacy-preserving smart contracts in this framework. Benhamouda et al. [9] proposed a Hyberledger Fabric-oriented auction protocol with zero knowledge (ZK) proofs. This is a three-party protocol that has a seller and up to two buyers. With Yao's protocol for the Millionaires problem, the comparison result is secret-shared among the two bidders. Galal [10] et al. proposed an auction scheme based on ZK-SNARKs on Ethereum blockchain. It supports bidders to publicly verify the computing result of auctioneer. Bag et al. presented SEAL [11], an auctioneer-free sealed-bid auction protocol. Each bidder has to publish a ZK proof about the relationship between commitment and ciphertext of each bit. Bidders jointly evaluate the winner, and the winner reveals the secrets that are concealed in the previous commitment. Blass and Kerschbaum [6] designed *Strain*, a verifiable bid comparison circuit, for sealed-bid auctions. The bidders first encrypt their bids with their public keys and call a smart contract to put the ciphers on a blockchain system. When computing the highest bid out of all the ciphers on the blockchain, each bidder encrypts her bid with others' public keys and comparing with others' encrypted bids by herself. Each bidder outputs $n-1$ comparison results and publish them with proofs of correct calculations. Finally, the winner is determined after all results are revealed and all proofs are verified. Ma et al. presented an integer comparison circuit-based sealed-bid scheme [7]. In this scheme, each bidder commits to her bid and submits the commitment with deposit. The auctioneer chooses a bid to compare with the current highest bid in ciphertexts through the integer comparison technique. Li et al. [12] also proposed a blockchain-based sealed-bid auction scheme. It applied bulletproof, another ZK tool like ZK-SNARKs to verify the correctness of the comparison result. Unfortunately, to the best of our knowledge, the comparison result of the auctioneer cannot be verified by bidders without exposing the winning bid for the current existing works.

Apart from protecting bid privacy, it is also important to design bid selection mechanism towards the concrete applications in auctions. Jia et al. [13] integrated symmetric encryption with differential privacy and blockchain to designed a distributed auction scheme in Internet of Things (IoT). This scheme can balance the trade-off between security and social efficiency in IoT. Here, the security of this scheme was based on the security of symmetric encryption. Therefore, an excellent substitution box [14] with rigorous theoretical and experimental analysis about statistic behaviour, unpredictable and random-like behaviour was important to this kind of auction schemes. Chen et al. proposed a general framework *SAFE* [15] of auction for wireless markets. It ensures the fairness with carefully-designed modular protocols and protect the bids' privacy well. Zhu et al. [16] designed a blockchain-based two-stage scheme for spectrum sharing auction. The bidders in this scenario can be classified into two types, primary users and secondary users. Primary users can select appropriate secondary users to maximum the unit utility and throughput in the auction. Luo et al. [17] proposed a blockchain-based two-way auction mechanism for electricity auction in Internet of Electric Vehicles. They constructed a price adjustment strategy for energy allocation based on Bayesian game approach to improve the social welfare and cost performance. Ma et al. [18] also proposed a blockchain-based auction model via Bayesian game approach for federated cloud services. Two Nash Equilibriums (NEs) were achieved in the model to select cost-effective cloud server and monitor them for providing trustworthy federated cloud services. Shi et al. [19] proposed a blockchain and feedback-based auction scheme for cloud markets. In this scheme, they designed a special greedy allocation mechanism and a second-price pricing mechanism for build high-quality cloud services. From the viewpoint of privacy protection, the core mechanisms and algorithms in the aforementioned mechanisms can be implemented by the HE-based bid circuits, because the primitive operations in these auctions merely consist of arithmetical operations such as bid comparison and bid sort. Therefore, the previous efficient privacy-preserving schemes can be used in these concrete applications.

III. Preliminaries

In this paper, the lowercase Latin characters denotes the scalars, the lower bold English letters denotes the vectors and uppercase English letters denotes the matrices. We denote by $[z]_x$ the remainder of any integer z, i.e., $[z]_x = sign(z) * (z \, mod \, x/2)$ and the value is in $(-x/2, x/2)$. For vector $\boldsymbol{z}$, $[\boldsymbol{z}]_x$ denotes a vector consisting of the remainder of each coordinate. Let vector $\boldsymbol{g} = [1, 2, \ldots, 2^{\gamma-1}]$, and bit decomposition function $\boldsymbol{g}^{-1}$ is defined as $\boldsymbol{g}^{-1}(z)^T = \boldsymbol{v}^T = (v_0, v_1, \ldots, v_{\gamma-1})$ where $v_i \in \{0, 1\}$ such that $z = \langle \boldsymbol{g}, \boldsymbol{v} \rangle$. By extension, $\boldsymbol{G}^{-1}(\boldsymbol{z})$ is the function that computes the bit decomposition of every integer of vector $\boldsymbol{z}$, i.e., $\boldsymbol{g} \cdot \boldsymbol{G}^{-1}(\boldsymbol{z}) = \boldsymbol{z}$.

A. Fully Homomorphic Encryption

In this section, we recall the fully homomorphic encryption (FHE) schemes. In a homomorphic encryption scheme, a third party can perform computations on ciphertext without

KeyGen(1^λ): The $params = \{\gamma, \rho, \eta, \tau\}$ are generated according to the security setting. Firstly, randomly sample an integer $p \xleftarrow{R} \{0,1\}^\eta$. Using Eq.1 and via rejection sampling first sample an integer $x_0 \leftarrow \mathcal{D}_{\gamma,\rho}(p)^{\geq 2^{\gamma-1}}$ and then τ integers $\{x_i\}_{1\leq i\leq \tau} \leftarrow \mathcal{D}_{\gamma,\rho}(p)^{\leq x_0}$. Write vector $\boldsymbol{x} = [x_1, ..., x_\tau]$, let $pk = \{params, \boldsymbol{x}, x_0\}$ and $sk = p$.

Enc(pk, m): To encrypt the $m \in \{0,1\}$, sample a matrix $\boldsymbol{S} \xleftarrow{R} \{0,1\}^{\tau\times\gamma}$ and compute vector $\boldsymbol{c} = [m \cdot \boldsymbol{g} + \boldsymbol{x}\boldsymbol{S}]_{x_0}$.

Dec($sk, \boldsymbol{c}$): Computes $f = \boldsymbol{c} \cdot \boldsymbol{g}^{-1}\,(p/2)\,(mod\ p)$, where p is the private key. Outputs the following

$$m \leftarrow \begin{cases} 1 & if\ |f| \geq p/4 \\ 0 & if\ |f| < p/4 \end{cases}$$

Eval($pk, \mathcal{C}, \boldsymbol{c}_1, \boldsymbol{c}_2$): Input any set of ciphertexts into the binary circuit $\mathcal{C}$ and output the resulting integer modulo x_0. The operations are listed as follows:

$$\begin{aligned} \boldsymbol{c}_{add} &= Add(\boldsymbol{c}_1, \boldsymbol{c}_2) = [\boldsymbol{c}_1 + \boldsymbol{c}_2]_{x_0} \\ \boldsymbol{c}_{mult} &= Mult(\boldsymbol{c}_1, \boldsymbol{c}_2) = \left[\boldsymbol{c}_1 \boldsymbol{G}^{-1}(\boldsymbol{c}_2)\right]_{x_0} \\ \boldsymbol{c}_{flip} &= Flip(\boldsymbol{c}_1) = [\boldsymbol{g} - \boldsymbol{c}_1]_{x_0} \end{aligned}$$

Fig. 1. BHE scheme [25].

any secret information. More formally, a HE scheme contains four Probabilistic Polynomial Time (PPT) algorithms, $KeyGen$, Enc, Dec and $Eval$, where $KeyGen$ is the algorithm to generate the key pair, Enc is the encryption algorithm, Dec is the decryption algorithm, and $Eval$ is the algorithm that can execute homomorphic computation on ciphertexts [20].

As mentioned in the survey [20], there are three types of HE Schemes, *partially, somewhat* and *fully* HE schemes. In *Fully Homomorphic Encryption* (FHE) scheme, the number of operations are unlimited. In this paper, we use the Benarroch's FHE scheme [21] (BHE, for short), a third-generation FHE over integers. It's based on *Approximate Greatest Common Divisor* (AGCD) problem [22], [23], [24].

We then recall the necessary definition for FHE scheme as follows [25].

Definition 1: The distribution $\mathcal{D}_{\gamma,\rho}(p)$ parameterised by integers γ, ρ and a η-bit prime p, is supported over γ-bit integers and defined as follows [25]:

$$\begin{aligned} \mathcal{D}_{\gamma,\rho}(p) \ = \ &\{\, q \leftarrow \mathbb{Z} \cap [0, 2^\gamma/p)\,, \\ &r \leftarrow \mathbb{Z} \cap (-2^\rho/p, 2^\rho/p) : Output\ x = p \cdot q + r\} \end{aligned} \tag{1}$$

Definition 2 ((ρ, η, γ)-AGCD)): The (ρ, η, γ)-AGCD problem is to find p given oracle access to $\mathcal{D}_{\gamma,\rho}(p)$, where p is a random η-bit prime. The decisional AGCD problem is to distinguish between $\mathcal{D}_{\gamma,\rho}(p)$ and the uniform distribution on $[0, 2^\gamma) \cap \mathbb{Z}$, given oracle access to both distributions [25].

Let distribution $\mathcal{D}_{\gamma,\rho}(p)^{\geq 2^{\gamma-1}}$ denotes the distribution whose value is $\geq 2^{\gamma-1}$, analogously we define $\mathcal{D}_{\gamma,\rho}(p)^{\leq x_0}$. The BHE scheme is given as shown in Fig. 1 . It is CPA secure under the (ρ, η, γ)-decisional AGCD problem in [21].

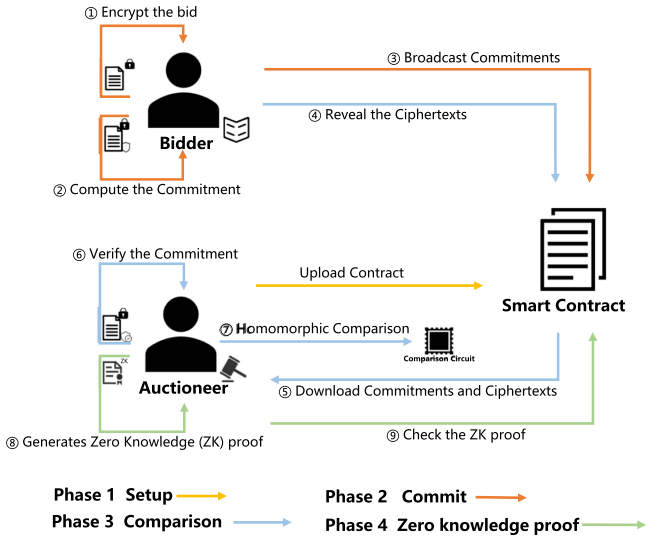


Fig. 2. Communication model.

The Bid Comparison Circuit can be designed by the BHE scheme [25]. For any two bids v_i and v_j, first convert them to binary form: $v_i = v_{i,1}||v_{i,2}||\ldots||v_{i,l}$ and $v_j = v_{j,1}||v_{j,2}||\ldots||v_{j,l}$. Then, compute $F_1 = (v_{i,1} \wedge \sim v_{j,1})$, and $F_k = (v_{i,k} \wedge \sim v_{j,k} \wedge \bigwedge_{u=1}^{k-1}(v_{i,u} = v_{j,u}))$ for all $k \in [2, l]$. Finally, the output is $\oplus_{k=1}^{l} F_k$.

If $v_i > v_j$, one of F_k (we call this bit as *the decision bit*) is 1 and all others are 0; on the contrary, all terms are 0. According to the operations in Fig. 1, the $\wedge$ operation of plaintext can be expressed by the $Mult$ operation of ciphertexts, the $\sim$ operation of plaintext can be expressed by the $Flip$ operation of ciphertexts, and the $=$ operation of plaintext can be expressed by the Add operation of ciphertexts.

Here is an example. Suppose there are two bidders in a sealed-bid auction. Two values of the bids are \$6 and \$4. The binary form of each bids is $v_1 = 110$ and $v_2 = 100$. Here $F_1 = 1 \wedge \sim 1 = 1 \wedge 0 = 0$, and $F_2 = 1 \wedge \sim 0 \wedge 1 = 1$, $F_3 = 0 \wedge \sim 0 \wedge (1 \wedge 0) = 0$. Then the output of the bid comparison is $F_1 \oplus F_2 \oplus F_3 = 1$. If we change v_1 to 100, and v_2 to 110, then the output of the bid comparison is $F_1 \oplus F_2 \oplus F_3 = 0 \oplus 0 \oplus 0 = 0$. Since all the operations of plaintexts can be computed by the corresponding operations of ciphertexts, the ciphertext of two bid comparison result can be obtained based on the operations shown in Fig. 1. In Section V, we will show the concrete procedure to compare two bids. We also construct a fully automatical circuit to extend it for comparing three and more bids there.

B. Zero-Knowledge Proof

In a ZK proof, the prover P proves to the verifier V that a statement is true without leaking secret information. The secret information is usually a secret input w such that an NP relation $R(x, w)$ is true where x is the public input. Consider there is a

two-party protocol where P sends a message a first, V replies a random t-bit string c as challenge, P then sends a message z, and V outputs accept or reject based on (a, c, z). The conversation will be called accepting conversation if V output accepts, and the definition of Σ-protocol is as follows.

Definition 3 (Σ-protocol [26]): A protocol between P and V is a Σ-protocol for a relation R if it satisfies following properties:

1) It is a public coin protocol of above 3-move form.
2) Completeness: If P and V follows the protocol on private input w to P and public input x then the verifier accepts whenever $(x, w) \in R$.
3) Special soundness: There exists a polynomial-time algorithm (usually called extractor) that can output w' such that $(x, w') \in R$ when given a pair of accepting conversations (a, c, z) and (a, c', z') with x.
4) Special honest-verifier zero-knowledge: There exists a polynomial-time simulator which on input x and a random c outputs an accepting conversation (a, c, z) with the same probability distribution as conversations between the honest P and V.

Besides the Σ-protocol, the following two computational hardness assumptions are also used in our scheme.

Definition 4 ($ISIS^{p}_{n,m,q,\beta}$ problem [27]): Given a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ and a syndrome $\boldsymbol{y} \in \mathbb{Z}_q^n$ for some parameters $\{n, m, q, \beta, p\}$ where $\beta \geq \sqrt{m}, m \geq 2n \log(q)$, the ISIS problem is to find a vector $\|\boldsymbol{x}\|_p \leq \beta$ such that $\boldsymbol{Ax} = \boldsymbol{y}\ mod\ q$.

It's proven as hard as approximating certain worst-case problems on lattices to within small factors in [27].

Definition 5 (Discrete Logarithm Assumption): The discrete logarithm assumption is to find an integer w such that $h = g^w$ when given $g \in G$ and $h \in \langle g \rangle$, where G is a group and g is a generator of G.

C. Blockchain and Smart Contract

Blockchain is a distributed ledger that has the characteristics of effective broadcasting. Valid transactions from parties in the network will be permanently recorded and can be verified publicly. Due to the consensus mechanism, the ledger is consistent for all parties.

Smart contract is a program that can be executed automatically on blockchain. One party can upload smart contracts or change the state of smart contracts by sending transactions. Ethereum [28] is a typical blockchain-based platform that allows users to create or call smart contracts. We will use Ethereum to carry out the simulation experiments.

IV. Models and Goals

Here we briefly introduce the communication model, adversary model and scheme goals for designing Homomorphic Encryption (HE)-based sealed-bid schemes.

A. Communication Model

The typical communication model for HE-based sealed-bid scheme usually contains four main phases.

The first phase is setup. A signed smart contract is uploaded by the auctioneer at the start of the auction in this phase.

The second phase is commit. Bidders 1) first encrypt their bids, and then 2) compute the commitment of the ciphertext. Following by that, bidders 3) broadcast their commitments by sending transactions to append the commitment to the blockchain. Here, every bidder can check the commitment she appended, if she wants. The execution of smart contract can be regarded as the communication channel, and it is easy to bind the submitted commitment to a bidder's address.

The third phase is comparison. The bidders 4) reveal the ciphertexts on the bid-opening day. The ciphertexts are used as the input of the designed bid comparison circuit. The auctioneer can 5) download the commitments and the ciphertexts, 6) verify the commitments, and 7) homomorphically compute the winning bid.

The last phase is zero knowledge proof. The auctioneer finally 8) generates a Zero Knowledge (ZK) proof of the computing, and 9) notifies bidders the final winner by the smart contract. The bidders can publicly verify the result by checking the ZK proof provided by the auctioneer.

B. Adversary Model

The adversary can be classified into two kinds. The first kind of the adversary is mainly from outside, while the second kind of the adversary is from inside at most. Specifically, the outside adversary is assumed to have the ability to both actively insert, delete, or modify any messages that should be sent by the auctioneer and bidders, and passively listen all the messages communicated among the auctioneer and the bidders, during in an auction.

The second kind of the adversary is from inside. It is assumed to have the ability to arbitrarily change the winner and get the endorsement from the auctioneer, and also ask any bidder to refuse to admit the commitment.

The assumptions for both kinds of the adversary are used to match the ability of the malicious auctioneer and bidders. That is, if any sealed-bid scheme is proven secure under the adversary model, with overwhelming probability, the auctioneer cannot leak bids or raise the winning price by arbitrarily changing the winner.

C. Goals

The sealed-bid auction scheme fulfils the following requirements.

Correctness: The scheme can correctly output the winner's encrypted bid and encrypted comparison results of bids. The auctioneer generates a ZK proof that every bidder can be easily verified.

Public Verifiability: Smart contract on blockchain is the only communication channel for all participants, every bidder can check the information on the public ledger. All processes of the auction are public verified.

Security: The security goal contains two aspects. On one hand, all the bids including the winning bid are not exposed to the auctioneer and bidders during in the auction. On the other hand,

with overwhelming probability, neither the auctioneer nor any bidder can change the commitment or the comparison result without being detected.

V. Scheme

The aim of the sealed-bid scheme is to compute the winner without leaking the value of bids. The core idea is to generate a valid winning ciphertext by homomorphic encryption, according to the ciphertext of bids that are submitted on blockchain by bidders. The winning bid can be publicly verified as same as one of the bid by Zero Knowledge (ZK) proof. The concrete scheme can be divided into four phases: Setup, Commit, Comparison and ZK Proof.

A. Setup

In the Setup phase, let v_i be the bid of the bidder V_i, and the binary form of v_i is: $v_i = v_{i,1}||v_{i,2}||\ldots||v_{i,l}$, where $v_{i,j} \in \{0,1\}, j \in [1,l]$. Here, the round l is a pre-defined constant value, and $i \in \mathcal{Z}$. We use $\boxplus$ represents $BHE.Add()$ and $\boxtimes$ for $BHE.Mul()$. Furthermore, $\sim \boldsymbol{c}$ denotes $BHE.Flip(\boldsymbol{c})$ where $\boldsymbol{c}$ is a BHE ciphertext.

The public key of the auctioneer is $pk = \{params, \boldsymbol{x}, x_0\}$, where $params = \{\gamma, \rho, \eta, \tau\}$, and the private key of the auctioneer is $sk = p$. The security parameter is the binary length of the private key sk. A new auction begins, when the auctioneer uploads the smart contract to blockchain.

B. Commitment

A bid is encrypted as a BHE ciphertext, and the hash of ciphertext is sent to smart contract as a commitment. Hence, in this phase, bidders encrypt their bids with public key pk. Since each binary bit of the bid v_i needs to be encrypted, each bidder needs to compute l ciphertexts. More specifically, V_i selects random matrices $\boldsymbol{S}_k \leftarrow \{0,1\}^{\tau \times \gamma}$ and computes vectors $\boldsymbol{C}_i = [\boldsymbol{c}_{i,1}, \ldots, \boldsymbol{c}_{i,l}]$ where $\boldsymbol{c}_{i,k} = \mathbf{Enc}(pk, v_{i,k}) = [v_{i,k} \cdot \boldsymbol{g} + \boldsymbol{x}\boldsymbol{S}_k]_{x_0}, k \in [1,l]$. V_i sends the hash value of $\boldsymbol{C}_i$ to the smart contract. It is negligible for an auctioneer to obtain the bids by hash values in probabilistic polynomial time of the security parameter.

Suppose that the secret bid is 59 whose binary representation is 111011 and the parameter l is 8. The bidder first pads the bid to l bits. That is, the bid is 00111011. Then, apply the BHE scheme to encrypt for each bit, which is 8 vectors in total.

C. Comparison

The second phase is the comparison on the ciphertexts. On the bid-opening day, each bidder will reveal the commitment by uploading the ciphertexts of the bid. It is easy to check the validity of the ciphertext by comparing the hash value with the commitment.

In *Strain*, the comparison circuit is designed for two-party comparison, and all of the results ($n(n-1)$ in total) have to be verified by re-computing. To reduce the computation cost, we construct a circuit that outputs the highest bid when input all the bids. If we execute the bid comparison circuit homomorphically, i.e., input the BHE ciphertexts of bids, the circuit outputs a new ciphertext of winning bid. Every bidder and the auctioneer can run the bid comparison circuit, and get the same result. Technically, the specific bid comparison circuit can be explained in three steps.

Step 1: As mentioned in Fischlin's two-Party comparison technique, F_k is calculated with two bids v_i and v_j, where k is in $[1,l]$. If $v_i > v_j$, one of F_k is 1 and all others are 0; on the contrary, all terms are 0.

Now, we let $F = \oplus_{k=1}^{l} F_k$. Hence, we construct a new circuit, whose output F is 1 if $v_i > v_j$, and is 0 on the contrary. With the BHE scheme, the homomorphic circuit to compare v_i with v_j is as follows:

1. Auctioneer homomorphically computes $\neg\, v_j$ and all $\neg\,(v_{i,u} \oplus v_{j,u})$ where $u = \{1, \ldots, l\}$.

$$\begin{aligned} \mathbf{Enc}\,(pk, \neg\, v_j) &= \neg\, \boldsymbol{C}_j = (\sim \boldsymbol{c}_{j,1}, \ \ldots, \ \sim \boldsymbol{c}_{j,l}) \\ \mathbf{Enc}\,(pk, \neg\ (v_{i,u} \oplus v_{j,u})) &= \sim (\boldsymbol{c}_{i,u} \boxplus \boldsymbol{c}_{j,u}) \end{aligned} \tag{2}$$

2. Auctioneer computes l ciphertexts $F_k, k = \{1, \ldots, l\}$ and outputs $F = F_1 \boxplus F_2 \boxplus \ldots \boxplus F_l$. F is the BHE ciphertext of comparison result. It's a vector and only can be decrypted with $sk = p$. If and only if F's plaintext is 1, $v_i > v_j$.

$$\begin{aligned} F_k = \boldsymbol{c}_{i,k} \boxtimes \neg\, \boldsymbol{c}_{j,k} \boxtimes \mathbf{Enc}(pk, \neg\,(v_{i,k+1} \oplus v_{j,k+1})) \boxtimes \\ \mathbf{Enc}(pk, \neg\,(v_{i,k+2} \oplus v_{j,k+2})) \boxtimes \ldots \boxtimes \mathbf{Enc}(pk, \neg\,(v_{i,l} \oplus v_{j,l})) \end{aligned} \tag{3}$$

For example, the plaintext of the two bids involved in the comparison are 59 and 54 respectively. Their binary representations are 111011 and 110110, respectively. If the plaintext is input for the circuit, the output should be 001000, and F is calculated to be 1, which is consistent with the conclusion that $59 > 54$. Under homomorphic encryption, the F will be an ciphertext of 1.

Step 2: Next, we design a Boolean circuit that outputs the winning bid. The new circuit can be executed homomorphically and outputs a new ciphertext of winning bid. The circuit can be expressed as $exF = (v_i \wedge F) \oplus (v_j \wedge \sim F)$, where F is the output of circuit in *step 1*. As a result, it is 0 or 1, and exF is same as the winner's v. If $v_i > v_j$, F is 1 and exF is v_i; otherwise exF is v_j.

As mentioned above, in homomorphic comparison circuit, F is a BHE ciphertext of comparison result. So the auctioneer can easily compute $\sim F$.

For $k = \{1, \ldots, l\}$, auctioneer computes $\boldsymbol{c}_{new,k}$ as follows. After all computations are done, output $exF = \{\boldsymbol{c}_{new,1}, \ldots, \boldsymbol{c}_{new,l}\}$.

$$\boldsymbol{c}_{new,k} = (\boldsymbol{c}_{i,k} \boxtimes F) \boxplus (\boldsymbol{c}_{j,k} \boxtimes \sim F) \tag{4}$$

The output exF is a new ciphertext of winning bid whose plaintext is the same as the winner's submitted ciphertext. But without decryption, it's negligible to find out which of the two old ciphertexts is related to the new ciphertext.

Back to the example in the previous step, F is 1, so the Boolean circuit can be expressed as:

$$\begin{aligned} exF &= (111011 \wedge 111111) \oplus (110110 \wedge 000000) \\ &= 111011 \oplus 000000 = 111011 = 59 \end{aligned} \tag{5}$$

The output is a valid ciphertext of the winner's bid.[1]

Step 3: The circuit in *Step 2* outputs a new ciphertext of winning bid, for two bids. Here, we consecutively run such circuit for $N-1$ rounds, where N is the total number of bidders. In each round, the input is just the output of the last round. After $N-1$ rounds are run, the output is definitely a valid ciphertext of the winning bid.

D. ZK Proof

Essentially, the auctioneer (prover, P) has to prove to all bidders (verifier, V) that two BHE ciphertexts $\boldsymbol{C}_1^*$ and $\boldsymbol{C}_2^*$ have the same plaintext without exposing the matrices $\boldsymbol{S}_1^*$ and $\boldsymbol{S}_2^*$. Here, $\boldsymbol{C}_1^*$ is the output of the bid comparison circuit. Every bidder and the auctioneer can compute this ciphertext directly. $\boldsymbol{C}_2^*$ is the ciphertext of the winner's bid. The symbol $*$ is used to distinguish them with the ciphtertexts $\boldsymbol{C}_1$ from the first bidder and $\boldsymbol{C}_2$ from the second bidder. After P publishes the winner to all bidders, any bidder V can check the ledger on blockchain to get this ciphertext. The last is to generate the non-interactive ZK proof.

The concrete ZK proof concerns about three steps that are depicted in Fig. 3.

Step 1: Bid Comparison Computation (**BCC**)

P first runs the bid comparison circuit to get a ciphertext of the winning bid, $\boldsymbol{C}_1^*$. Next, since the auctioneer has the private key p, the auctioneer can also decrypt all the bidders' ciphertexts, and get the winning bid from $\boldsymbol{C}_2^*$. Let $Index_w$ indicates the index of the winner. In reality, this index can be identified by the bidder's address of blockchain. The auctioneer eventually publishes $Index_w$ to all bidders by smart contract.

Step 2: Bid Comparison Proof (**BCP**)

Assume that $k \in \{1, \ldots, l\}$. Notice that the $f_{1,k}^*$ is essentially equal to $q_{1,k}^* \cdot p + r_{1,k}^*$ for some $q_{1,k}^*$ and $r_{1,k}^*$. Similarly, $f_{2,k}^*$ is also equal to $q_{2,k}^* \cdot p + r_{2,k}^*$ for some $q_{2,k}^*$ and $r_{2,k}^*$. Therefore, P can construct three non-interactive proofs of knowledge for each round as shown in (6). This can be done by using the Σ-protocol and the Fiat-Shamir scheme [29] which are common in the area of ZK proof. Finally, P generates all the proofs for l rounds and publishes them by smart contract.

Step 3: Bid Comparison Verification (**BCV**)

By checking $Index_w$ with smart contract, every bidder can get the winner's ciphertext $\boldsymbol{C}_1^*$. Then the first verification of each bidder is to verify whether $f_{1,k}^*$ equals $\boldsymbol{c}_{1,k}^* \cdot \boldsymbol{g}^{-1}(p/2)\ mod\ p$ and $q_{1,k}^* \cdot p + r_{1,k}^*$. After that, every bidder can get $\boldsymbol{C}_2^*$ from the ledger as well. So the bidder can verify whether $f_{2,k}^*$ equals $\boldsymbol{c}_{2,k}^* \cdot \boldsymbol{g}^{-1}(p/2)\ mod\ p$ and $q_{2,k}^* \cdot p + r_{2,k}^*$. Finally, every bidder is to verify whether both $r_{1,k}^*$ and $r_{2,k}^*$ are smaller than $p/4$, or both are greater or equal to $p/4$.

$$\begin{aligned} &PK\{q_{1,k}^*, r_{1,k}^*, f_{1,k}^*, p : f_{1,k}^* = \boldsymbol{c}_{1,k}^* \cdot \boldsymbol{g}^{-1}(p/2)\ mod\ p \\ &\qquad \wedge f_{1,k}^* = q_{1,k}^* \cdot p + r_{1,k}^*\} \\ &PK\{q_{2,k}^*, r_{2,k}^*, f_{2,k}^*, p : f_{2,k}^* = \boldsymbol{c}_{2,k}^* \cdot \boldsymbol{g}^{-1}(p/2)\ mod\ p \\ &\qquad \wedge f_{2,k}^* = q_{2,k}^* \cdot p + r_{2,k}^*\} \\ &PK\{f_{1,k}^*, f_{2,k}^*, p : f_{1,k}^* = q_{1,k}^* \cdot p + r_{1,k}^* \\ &\qquad \wedge f_{2,k}^* = q_{2,k}^* \cdot p + r_{2,k}^* \\ &\qquad \wedge [(r_{1,k}^*, r_{2,k}^* < p/4) \vee (r_{1,k}^*, r_{2,k}^* \geq p/4)]\} \end{aligned} \tag{6}$$

1) BCC (p, pk):
Search all of $\{\boldsymbol{C}_i\}$ from the storage of smart contract
Run the bid comparison circuit to get $\boldsymbol{C}_1^*$
Decrypt $\boldsymbol{C}_1^*$ and all of $\{\boldsymbol{C}_i\}$
Find out $\boldsymbol{C}_2^*$ and get the winner's index $Index_w$
Publish $Index_w$ by smart contract

2) BCP(p, pk, $\boldsymbol{C}_1^*$, $\boldsymbol{C}_2^*$):
For $k = 1$ to l do:
Compute $f_{1,k}^* = \boldsymbol{c}_{1,k}^* \cdot \boldsymbol{g}^{-1}(p/2)\ mod\ p$
Compute $f_{2,k}^* = \boldsymbol{c}_{2,k}^* \cdot \boldsymbol{g}^{-1}(p/2)\ mod\ p$
Compute $q_{1,k}^*, r_{1,k}^*, q_{2,k}^*, r_{2,k}^*$ by $\boldsymbol{c}_{1,k}^*, \boldsymbol{c}_{2,k}^*, p$
Generate all of the three $PK\{q_{1,k}^*, r_{1,k}^*, f_{1,k}^*, p\}$, $PK\{q_{2,k}^*, r_{2,k}^*, f_{2,k}^*, p\}$, $PK\{f_{1,k}^*, f_{2,k}^*, p\}$ by Σ-protocol and the Fiat-Shamir scheme
Publish all the Proofs of Knowledge by smart contract

3) BCV(pk, $Index_w$):
Save all of $\{\boldsymbol{C}_i\}$ from the storage of smart contract
Run the bid comparison circuit to get $\boldsymbol{C}_1^*$
Find out $\boldsymbol{C}_2^*$ according to $Index_w$
For $k = 1$ to l do:
Verify all of the three $PK\{q_{1,k}^*, r_{1,k}^*, f_{1,k}^*, p\}$, $PK\{q_{2,k}^*, r_{2,k}^*, f_{2,k}^*, p\}$, $PK\{f_{1,k}^*, f_{2,k}^*, p\}$ by Σ-protocol and the Fiat-Shamir scheme
If any verification fails, return ϱ
Otherwise, return ϯ

Fig. 3. Non-interactive ZK proof in the scheme.

Remark: The auctioneer can prove that the output of the bid comparison result is a valid ciphertext. This is not always necessary, suppose all the bidders faithfully encrypt their bids. In reality, this is the normal case. However, bidders might also like to check the validity of other bidders' ciphertexts. If so, we briefly introduce the procedure.

First, each bidder can send the auctioneer the matrix S_i that is used to encrypt the bid. An efficient and secure encryption scheme with a good substitution box [14] can be used to protect the confidentiality of the matrix. Second, the auctioneer can decrypt and save all the matrices $\{S_i\}$, $(i \in \mathcal{Z})$ of all bidders. Third, the auctioneer can compute S_1^* of $\boldsymbol{C}_1^*$ by all the $\{S_i\}$, because $\boldsymbol{C}_1^*$ can be computed by $\boxplus, \boxtimes, \sim$ on the ciphertexts of bids. Finally, the auctioneer can apply ZK proof to prove the validity of all the ciphertexts.

[1] This is an example for the classical first-price sealed bid auction. It can also support second-price sealed bid auction by running our bid comparison circuit twice. In the second running, the ciphertext of the highest bid in the first running is removed.

VI. Theoretical Analysis

Since the goal of *Public Verifiability* is relatively obvious, we mainly focus on analysing the *Correctness* and *Security* of the proposed Blockchain-based Sealed-bid Scheme (BSS) in this section. More precisely, we first prove a lemma about the ZK proof of the BSS. We then analyse the *Correctness* and *Security* of the BSS, based on the conclusion of the lemma.

Lemma 1: The Blockchain-based Sealed-bid Scheme (BSS) is a ZK proof of knowledge $sk = p$, such that $Dec(sk, \boldsymbol{c}_1) = Dec(sk, \boldsymbol{c}_2)$.

Proof: (Skeleton) In the BSS, all the operations are followed by the BHE scheme [25], as shown in Fig. 1. Therefore, the completeness can be guaranteed by the homomorphic property of the BHE scheme.

Since Σ-protocol and the Fiat-Shamir scheme are used for bidders to verify the three proofs of knowledge, $PK\{q^*_{1,k}, r^*_{1,k}, f^*_{1,k}, p\}$, $PK\{q^*_{2,k}, r^*_{2,k}, f^*_{2,k}, p\}$, $PK\{f^*_{1,k}, f^*_{2,k}, p\}$. The soundness and zero-knowledge properties can also be maintained.

Without loss of generality, we show how the Σ-protocol and the Fiat-Shamir scheme are used for proving the knowledge $PK\{q^*_{1,k}, r^*_{1,k}, f^*_{1,k}, p : f^*_{1,k} = \boldsymbol{c}^*_{1,k} \cdot \boldsymbol{g}^{-1}(p/2) \ mod \ p \wedge f^*_{1,k} = q^*_{1,k} \cdot p + r^*_{1,k}\}$ more clearly. Since all the $q^*_{1,k}, r^*_{1,k}, f^*_{1,k}, p$ can be mapped into elements in a group, the auctioneer can apply a random generator of the group as the base number, and all these elements can be set as the exponents of the base number. Similarly, the vector $\boldsymbol{c}^*_{1,k}$ contains γ numbers, which can also be mapped into γ elements in the same group. So does the vector $\boldsymbol{g}^{-1}(p/2)$, because it is the binary representation of the ciphertext $\boldsymbol{c}^*_{1,k}$. More importantly, since $z = \langle \boldsymbol{g}, \boldsymbol{v} \rangle$, and this inner product is essentially the addition of γ products. Hence, z can also be set as the exponent of the base number.

Above all, not only $q^*_{1,k}, r^*_{1,k}, f^*_{1,k}, p, \boldsymbol{c}^*_{1,k}, z, \boldsymbol{g}^{-1}(p/2)$ in the BSS can be set to the exponents of the base number, but also all the operations in the BSS can be converted to addition and multiplication. Therefore, the Σ-protocol and the Fiat-Shamir scheme can be used for $PK\{q^*_{1,k}, r^*_{1,k}, f^*_{1,k}, p\}$.

We complete the proof. □

A. Correctness

Now we prove that the scheme can correctly output a ciphertext of winning bid.

Theorem 1: The scheme mentioned above is correct for malicious bidders and auctioneer.

Proof: First of all, it should be noted that only the bidders can overwhelmingly reveal the commitments in probabilistic polynomial time of the security parameter. To prove correctness, we need to show that $\boldsymbol{c}_{1,k}$ and $\boldsymbol{c}_{2,k}$ have same plaintext for $k \in [1, l]$, where $\boldsymbol{c}_{1,k}$ is computed by circuit, and $\boldsymbol{c}_{2,k}$ the original ciphertext of v_k.

Remark that when two numbers are compared in binary form, one of $F_k = v_{i,k} \wedge \sim v_{j,k} \wedge \bigwedge_{u=1}^{k-1}(v_{i,u} = v_{j,u})$ is 1 and others are 0 if and only if $v_i > v_j$. We can prove this by induction. If $v_i > v_j$, there must be a t such that $v_{i,t} = 1, v_{j,t} = 0$ and $v_{i,k} = v_{j,k}$ for $t = \{1, 2, \ldots, k-1\}$, then $F_t = 1$ and $F = \oplus_{k=1}^{l} F_k = 1$. To prove that $F = 1 \Rightarrow v_i > v_j$, we assume that there is a t such that $F_t = 1$ and $v_i \leq v_j$. To make $F_t = 1$, $v_{i,t}$ should be 1 and $v_{j,t}$ should be 0. Because that $v_i \leq v_j$, there should be a w such that $v_{i,w} = 0, v_{j,w} = 1, w < t$ and the higher bits are all the same. This is a contradiction of assumption $F_t = 1$ which requires that all bits before t should be same. Our first homomorphic computation sub-circuit is based on this idea, it outputs F that is a ciphertext of 1 when $v_i > v_j$ and of 0 otherwise.

For second sub-circuit $exF = (v_{i,k} \wedge F) \oplus (v_{j,k} \wedge \sim F)$, it's clear that if $v_{i,k} > v_{j,k}$, then F is 1 and exF will be $v_{i,k}$; otherwise, exF will be $v_{j,k}$. Now, we assume that $v_i > v_j$, then F's plaintext is 1. For each $v_{i,k}, v_{j,k}$, evaluate them and we get the $exF = \{\boldsymbol{c}_{new,1}, \ldots, \boldsymbol{c}_{new,l}\}$ where $\boldsymbol{c}_{new,k}$ is a new ciphertext of $v_{i,k}$ after homomorphic computation.

After comparison, the auctioneer can take the three ZK proofs as we explained in (6) and Lem. 1.

We complete the proof. □

B. Security

In this subsection, we prove that the BSS scheme has the property of special honest-verifier zero-knowledge. The proof formally show why malicious bidders are hardly to obtain the secret information in the sealed-bid auction.

Lemma 2: Given parameters x_0, γ, a vector $\boldsymbol{C} \in \mathbb{Z}^{\gamma}_{x_0}$ and $m = \boldsymbol{C} \cdot \boldsymbol{z}^T \ mod \ x_0$ can be generated by the BSS scheme, such that there exists a vector $\boldsymbol{z} \in \{0, 1\}^{\gamma}$ and $y = A \cdot \boldsymbol{z}^T \ mod \ q$. It is negligible to computationally find the $\boldsymbol{z}$ based on $\boldsymbol{C}$ and m.

Proof: (skeleton) Suppose there exists a Probabilistic Polynomial Time (PPT) algorithm $\mathcal{C}$ for finding $\boldsymbol{z}$ by the inputs $\boldsymbol{A}$ and $\boldsymbol{y}$ with a non-negligible probability, an ISIS solver $\mathcal{S}_{ISIS}$ can make use of $\mathcal{C}$ to solve the $ISIS^p_{n,m,q,\beta}$ problem as shown in Definition 4. More concretely, the experiment of $\mathcal{S}_{ISIS}$ runs as below:

1) Set the parameters $n = 1, m = 3\gamma, q' = x_0$ and some β, p where $\beta \geq \sqrt{m}, m \geq 2n \log(q)$.
2) When a matrix $\boldsymbol{C}$ and a syndrome m are given by the BSS scheme, the matrix A can be formed by appending 2γ zero-columns to $\boldsymbol{C}$.
3) Send A and m to $\mathcal{C}$ as the inputs.
4) Output whatever $\mathcal{C}$ outputs.

After running the experiment $\mathcal{S}_{ISIS}$, an output $\boldsymbol{z}$ can be obtained overwhelmingly. This output $\boldsymbol{z}$ meets the equation that $m = A \cdot \boldsymbol{z}^T \ mod \ q$. Thus, the $ISIS^p_{n,m,q,\beta}$ problem can be solved, because $\mathcal{C}$ can find the output with a non-negligible probability.[2]

Formally, the equation $\mathcal{S}_{ISIS} = 1$ is regarded as that the solver $\mathcal{S}_{ISIS}$ wins, and the equation $\mathcal{C} = 1$ is defined as that the PPT algorithm $\mathcal{C}$ successfully finds the correct output. We have that $Pr[\mathcal{S}_{ISIS} = 1] = Pr[\mathcal{C} = 1]$.

We complete the proof.

Lemma 3: It is negligible to computationally compute whether $\Omega = \boldsymbol{c}_0$ or $\Omega = \boldsymbol{c}_1$, given pk and challenge $\Omega \in$

[2]More rigorously, the ISIS problem here is a special instance $ISIS^{\infty}_{n,m,q,\beta}$ in [27].

$\{\boldsymbol{c}_0, \boldsymbol{c}_1\}$, where $\boldsymbol{c}_0 = HE.Enc(pk, m_0)$ and $\boldsymbol{c}_1 = HE.Enc(pk, m_1)$ for any m_0, m_1.

Proof: (skeleton)

This lemma is essentially equivalent to prove that the BSS scheme is secure for the Indistinguishability Chosen Plaintext Attack (IND-CPA).

Formally, we have to construct five hybrid arguments and use the Lemma 2 together to prove this lemma.

Hybrid 0. We define the distribution $(pk, \boldsymbol{c_0})$ in H_0 as follows. Let $pk \leftarrow KeyGen(1^\lambda)$, and let $\boldsymbol{c_0} = HE.Enc(pk, m_0)$.

Hybrid 1. We define the distribution $(pk, \boldsymbol{c_0})$ in H_1 as follows. Let $pk = \{params, \boldsymbol{u}, u_0\}$. Since $\boldsymbol{u}, u_0$ are uniformly selected; and hence, the pk here is independent with pk in H_0. Let $\boldsymbol{c_0} = BSS.Enc(pk, m_0)$. $(pk, \boldsymbol{c_0})_{H_0}$ and $(pk, \boldsymbol{c_0})_{H_1}$ are computationally indistinguishable, because of the hardness of finding the vector $\boldsymbol{z}$ by Lemma 2. Hence, for some negligible function $negl_1$, we have

$$|Pr_{H_0}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0}) = 1) - Pr_{H_1}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0}) = 1)| \leq negl_1(\lambda)$$

Hybrid 2. We define the distribution $(pk, \boldsymbol{c_0})$ in H_2 as follows. Let $pk = \{params, \boldsymbol{u}, u_0\}$. Let $\boldsymbol{c_0} = [m \cdot \boldsymbol{g} + \boldsymbol{v}]_{u_0}$, where $v \leftarrow \mathbb{Z}^\gamma_{u_0}$. Since $\boldsymbol{v}$ is completely random, the probability of success for $\mathcal{A}$ in this hybrid is exactly 1/2. Hence, for some negligible function $negl_2$, we have

$$|Pr_{H_1}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0}) = 1) - Pr_{H_2}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0}) = 1)| \leq negl_2(\lambda)$$

Hybrid 3. We define the distribution $(pk, \boldsymbol{c_1})$ in H_3 as follows. Let $pk = \{params, \boldsymbol{u}, u_0\}$. Let $\boldsymbol{c_1} = BSS.Enc(pk, m_1)$. Since $\boldsymbol{v}$ in H_2 is completely random, the probability of success for $\mathcal{A}$ in this hybrid is also exactly 1/2. Hence, for some negligible function $negl_3$, we have

$$|Pr_{H_2}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0}) = 1) - Pr_{H_3}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_1}) = 1)| \leq negl_3(\lambda)$$

Hybrid 4. We define the distribution $(pk, \boldsymbol{c_1})$ as follows. Let $pk \leftarrow KeyGen(1^\lambda)$, and let $\boldsymbol{c_1} = BSS.Enc(pk, m_1)$. Since $(pk, \boldsymbol{c_1})_{H_3}$ and $(pk, \boldsymbol{c_1})_{H_4}$ are computationally indistinguishable, because of the hardness of finding the vector $\boldsymbol{z}$ by Lemma 2. Hence, for some negligible function $negl_4$, we have

$$|Pr_{H_3}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_1}) = 1) - Pr_{H_4}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_1}) = 1)| \leq negl_4(\lambda)$$

Now, we put all of the hybrid arguments together. For some negligible function $negl$, we can conclude that

$$|Pr_{H_0}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0}) = 1) - Pr_{H_4}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_1}) = 1)| \leq negl_1(\lambda) + negl_2(\lambda) + negl_3(\lambda) + negl_4(\lambda) \leq negl(\lambda)$$

In fact, there is still 'Last-Mile' to arrive in the IND-CPA experiment. That is, if $|Pr_{H_0}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0}) = 1) - Pr_{H_4}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_1}) = 1)|$ is negligible for any c_0 and c_1, the BSS scheme is secure in the IND-CPA experiment.

The IND-CPA experiment of the BSS scheme $PrivK^{cpa}_{\mathcal{A},BSS}(\lambda)$ executes as below.

1) An encryption key pk is generated by running $BSS.KeyGen$.
2) The adversary $\mathcal{A}$ can access an oracle $BSS.Enc$ to encrypt any plaintext.
3) The adversary $\mathcal{A}$ outputs a pair of messages m_0, m_1 of the same length.
4) A uniform bit $b \in \{0, 1\}$ is chosen, and $c \leftarrow BSS.Enc(m_b)$ is computed and sent to $\mathcal{A}$.
5) The adversary $\mathcal{A}$ continues to access to $BSS.Enc$, and outputs a bit b'.
6) The output of the experiment is set to be 1 if $b = b'$ and 0 otherwise.

We define that the adversary $\mathcal{A}$ succeeds if the output of $PrivK^{cpa}_{\mathcal{A},BSS}(\lambda)$ is 1.

If $Pr[PrivK^{cpa}_{\mathcal{A},BSS}(\lambda) = 1]$ is non_negligible, we can apply this experiment to construct a simulator $\mathcal{S}$ for distinguishing the $\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0})$ and $\mathcal{A}(1^\lambda, pk, \boldsymbol{c_1})$.

The simulator $\mathcal{S}$ calls the $\mathcal{A}$ to run the distinguishability experiment D as below.

1) The simulator $\mathcal{S}$ generates pk according to $BSS.KeyGen$.
2) $\mathcal{S}$ can encrypt the plaintexts from $\mathcal{A}$ by running $BSS.Enc$.
3) When receiving the m_0, m_1 from $\mathcal{A}$, $\mathcal{S}$ encrypts them to get $\boldsymbol{c_0}$ and $\boldsymbol{c_1}$.
4) When $\mathcal{S}$ is given $\boldsymbol{c_b}$, send it to the $\mathcal{A}$.
5) Output whatever $\mathcal{A}$ outputs.

Since $Pr[PrivK^{cpa}_{\mathcal{A},BSS}(\lambda) = 1] = Pr[D(\mathcal{S}) = 1]$ and $|Pr_{H_0}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_0}) = 1) - Pr_{H_4}(\mathcal{A}(1^\lambda, pk, \boldsymbol{c_1}) = 1)|$ is negligible for any c_0, c_1,

$$\begin{aligned}
&Pr[D(\mathcal{S}) = 1] \\
&= 1/2 Pr[Pr[D(\mathcal{S}) = 1] | c = c_0] \\
&\quad + 1/2 Pr[Pr[D(\mathcal{S}) = 1] | c = c_1] \\
&= 1/2(1 - Pr[D(\mathcal{S}) = 1] | c = c_0] + Pr[D(\mathcal{S}) = 1] | c = c_1]) \\
&\leq 1/2 + 1/2 | Pr[D(\mathcal{S}) = 1] | c = c_0] \\
&\quad - Pr[D(\mathcal{S}) = 1] | c = c_1] |) \\
&\leq 1/2 + negl(\lambda)
\end{aligned}$$

We complete the proof. □

Lemma 4: With negligible probability, malicious bidders C can learn no more than the inputs $\boldsymbol{C}_{c_i} = \{\boldsymbol{c}_{c_i,1}, \ldots, \boldsymbol{c}_{c_i,l}\}$ and the circuit's output $\boldsymbol{C}_w$, where $c_i \in C$ and $|C| = \delta, \delta < \epsilon$.

Proof: (skeleton) We first assume that all malicious bidders collude with each other. In order to prove this lemma, we need to show that they can learn the secret key in the ZK proof with negligible probability. Second, we need to show that they can learn the bids of honest bidders with negligible probability, and it can be concluded as the following situations:

1) If two honest bidders exchange their bids, the adversary (colluding bidders) is negligible to determine which bid corresponds to which bidder.

2) Assume a bidder is not the winner, she is neither to change any bit of her bids, nor does she to know the *position of the deciding bit of the winner's bid*.

For case 1, it's notable that, according to Lemma 2, with m and $\boldsymbol{C}$, with negligible probability, the adversary can learn any information about the secret key. Furthermore, according to the DL problem in Definition 5, it's computationally hard to compute secret p with $h' = g^p$. Hence, the adversary learns almost nothing about the secret key in the ZK proof.

To prove the security of bids, we first consider the scenario in which the colluding set of bidders C is assumed to contain the winner, i.e., the adversary knows the highest bid v_w of all bidders. As mentioned before, with negligible probability, the adversary can learn the secret key, so she is negligible to get the bids of honest bidders and only knows they are smaller than v_w without knowing the relations between other bids.

More specifically, we assume that the honest bid V_{h_1} submits a v_1-cryptogram $\boldsymbol{C}_{h_1}$ and V_{h_2} submits v_2-cryptogram $\boldsymbol{C}_{h_2}$. It's clear that if v_1 and v_2 are exchanged, and they submit new ciphertexts $\boldsymbol{C}'_{h_1}$ and $\boldsymbol{C}'_{h_2}$, the intermediate parameters F_k of the circuit where $k \in [h_1, h_2]$ may change, as well as the final output. However, according to Lemma 3, with negligible probability, the adversary can distinguish between $\boldsymbol{A}$ and $\boldsymbol{B}$ where

$$\boldsymbol{A} = (pk, \boldsymbol{C}_1, \ldots, \boldsymbol{C}_{h_1}, \boldsymbol{C}_{h_1+1}, \ldots, \boldsymbol{C}_{h_2-1}, \boldsymbol{C}_{h_2}, \ldots, \boldsymbol{C}_n)$$

$$\boldsymbol{B} = \left(pk, \boldsymbol{C}_1, \ldots, \boldsymbol{C}'_{h_1}, \boldsymbol{C}'_{h_1+1}, \ldots, \boldsymbol{C}'_{h_2-1}, \boldsymbol{C}'_{h_2}, \ldots, \boldsymbol{C}_n\right).$$

Furthermore, $\boldsymbol{F}_A$ and $\boldsymbol{F}_B$ are also computationally distinguishable, where

$$\boldsymbol{F}_A = (pk, F_1, \ldots, F_{h_1}, F_{h_1+1}, \ldots, F_{h_2-1}, F_{h_2}, \ldots, F_n)$$

$$\boldsymbol{F}_B = \left(pk, F_1, \ldots, F'_{h_1}, F'_{h_1+1}, \ldots, F'_{h_2-1}, F'_{h_2}, \ldots, F_n\right).$$

In other words, it is negligible to know which of two honest bidders exchange their bids without decrypting Fs.

For case 2, we assume that the adversary can detect that an honest bidder V_h changes some bits of bid v_h. Since only the auctioneer can decrypt the comparison result Fs output by the circuit, if the adversary can detect who changes bid, it must be a scenario that the winner changes after modification of bids and the auctioneer makes a proof to claim a new winner. Recalling our two-party comparison strategy, if we input v_w as first input and v_h as second input, there will be a *deciding bit position* t. If she wants to be raise a higher bid which is more than the current winner's bid, then she should change the higher-order than t. This is a contradiction of the case 2.

Now, let us consider the scenario that the winner is not in the set C of the colluding bidders. The adversary only knows their inputs and the output $\boldsymbol{C}_w$. Similar to the scenario that C contains winner, exchanging two honest bidder's bids may result in changes in the F_ks, but the winner cannot be changed as only the auctioneer can make a proof. Hence, the adversary cannot find out which of the honest bidder's bids exchanged because of computational distinguishability.

Similarly, the proof of situation 2 when C does not contain winner can also be made by intuition, and we can conclude that the adversary cannot detect honest bids' change if the winner is still the same.

We complete the proof. □

Theorem 2: The BSS scheme is secure to resist against malicious auctioneer and bidders.

Proof: (skeleton) Based on Lemma 4, if the set of colluding bidders contains the winner, then they can learn the highest bid of the auction, but they are negligible to learn other bids of honest bidders; if the winner is an honest bidder, then the colluding bidders learn almost nothing more than their own inputs and the output of the circuit. Based on the computationally difficult problem, in either case, the bidder is negligible to obtain the secret key. Hence, we can conclude that this theorem is correct.□

In addition, a rigorous analysis of zero-knowledge can be performed by examining the zero-knowledge simulators associated with the underlying Σ-protocol and BHE scheme. These simulators, denoted as *Sim*, play a crucial role in demonstrating the zero-knowledge property. Specifically, we consider any probabilistic polynomial-time (PPT) distinguisher D that aims to distinguish honestly generated proofs by the simulators, and we analyze the success probability of D in achieving this task.

Let us assume that D interacts with the following processes: the key generation algorithm $KeyGen(1^\lambda)$, which generates a public-private key pair (pk, sk), the Σ-protocol prover $\Sigma.Prover(pk, sk, g, f'_1, f'_2)$, which takes as input the public and private keys, along with some auxiliary information (g, f'_1, f'_2), and produces a proof π, and finally, the simulator $\mathrm{Sim}(td, sk, pk, g)$, which interacts with the distinguisher by providing a simulated proof π^*.

To analyze the zero-knowledge property, we consider the probability that the distinguisher D outputs 1, denoted as $D(\pi, \pi^*) = 1$, when interacting with the processes mentioned above. The fundamental requirement for achieving zero-knowledge is that this probability should be negligible in terms of the security parameter λ. Formally, we have:

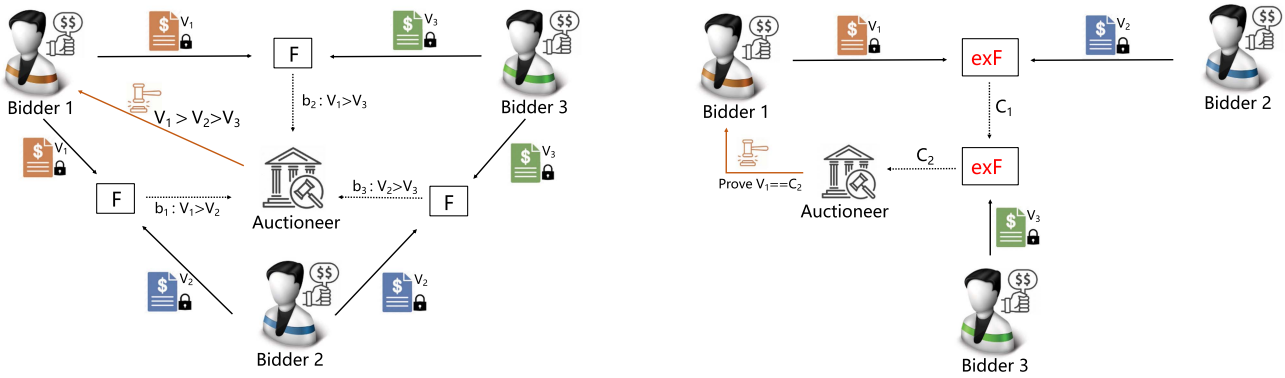
$$PrKeyGen(1^\lambda) \to (pk, sk), \Sigma.Prover(pk, sk, g, f'_1, f'_2)$$
$$\to \pi, \mathrm{Sim}(td, sk, pk, g) \to \pi^* | D(\pi, \pi^*) = 1 \leq negl(\lambda)$$

where $negl(\lambda)$ represents a negligible function of the security parameter λ. This inequality ensures that any efficient(PPT) distinguisher D is negligible to differentiate between the honestly generated proofs and the simulated proofs. Hence, the zero-knowledge property is preserved.

VII. Performance Analysis

In this section, we adopt theoretical complexity to analyze the performance of the schemes from three aspects: block latency, computation overhead, and storage cost. We also take experiments to compare Strain and BSS schemes in practical protocol executing scenarios.

Before starting the performance analysis, we demonstrate the difference between Strain and BSS in protocol execution by taking an example, where 3 bidders publish their bid (v_1, v_2, v_3) respectively, and for simplicity assuming the relation of $v_1 > v_2 > v_3$ holds. The goal of protocols is to find the max bid, i.e., v_1, among them without revealing its value.



(a) The protocol process of Strain scheme

(b) The protocol process of BSS scheme

Fig. 4. Differences between BSS and Strain protocols.

Fig. 4(a) shows the process of conducting the Strain scheme, which outputs the comparison results for any two bidders. When all bidders finish the comparison to others, we get whole sorting results. For example, b_1 is the comparison result of v_1 and v_2, where it means $v_1 > v_2$. We can also get the comparison results, which are b_2 and b_3, and finally decide the max value is v_1.

Fig. 4(b) shows the process of conducting the BSS scheme, which directly outputs the max bid during the comparisons. The winner needs to prove that the max bid is equal to the bid it holds. Thus, we can concatenate the pairwise comparison to reduce the number of comparisons. For example, c_1 is the computation result of v_1 and v_2 via the function, which has $c_1 == v_1$. Then we take c_1 and v_3 into the same function, getting the max value c_2 as output, which still has $c_2 == v_1$. The BSS scheme uses fewer rounds than the Strain scheme to get the same effect, and we verify it through experiments.

A. Complexity

Block Latency: The BSS starts by all bidders sending the commitments of their own bids. All bidders can do it in parallel, and it requires one block latency. After mining the block, all participants can reveal the commitments and download the ciphertexts to run the bid comparison circuit. The auctioneer plays the role of prover and publishes a ZK proof after comparison. This also requires one block latency. In the final block, the verifiers return the verification result. This can also be done in parallel. In summary, the Blockchain-based Sealed-bid scheme requires a total of 3 blocks latency: 1 block for committing to bids and 2 blocks for computing the winner. The latency is a constant in both bit length l of the bid and the number of bidders n while typical MPC protocols always require at least $O(l)$ rounds of interaction and *Strain* requires 4 blocks latency: 1 blocks for committing, 3 blocks for comparisons and determining the winner.

Computation Overhead: Our scheme has three phases, operations in the first two phases are mainly BHE functions and operations in ZK proof including encryption and binary operations.

We measure the computation cost in the terms of the number of BHE operations and the number of computations in ZK proof. During the *Commit* phase, each bidder needs to compute a ciphertext by **Enc** function for each bit of bids. Hence, the total number for generating ciphertext $\boldsymbol{C}$ of a bid is l. In the *Comparison* phase, the auctioneer must run the circuit, but the bidder can choose to run or not. To compute the relation of two ciphertexts, it will need 3 $Mult$, 2 $Flip$ and 1 Add functions to compute one F_k and require l Add to calculate F by F_k.

Hence, the total number of operations to compute F is $7l$. Computing the winner with F and $\boldsymbol{C}$ in *Step 2* requires 1 $Flip$, l Add and $2l$ $Mult$, and the total number is $3l + 1$. The circuit is equivalent to running the above two steps $n - 1$ times, where n is the number of bidders. Hence, the total number of operations in *Comparison* phase is $(n - 1)(10l + 1)$. In the ZK proof (non-interactive version) phase, the prover (auctioneer) computes 6 numbers $q^*_{1,k}, r^*_{1,k}, f^*_{1,k}$ and $q^*_{2,k}, r^*_{2,k}, f^*_{2,k}$. The verifier needs to verifies 7 equations.

Since each round of ZK proof generation and ZK proof verification is required for each bit for each bid; thereby, the total computation cost for ZK proof can be summarised as shown in Table I.

In *Strain*, a bidder needs to call l times of encryption to publish a ciphertext. In a two-party comparison, the bidder first encrypts his bid under another public key, and then calculates the result through $6l$ operations (we ignore shuffling for simplicity), and takes l decryption to get one result of comparison. So, the total number of operations in *Comparison* phase for one bidder is $8(n - 1)l$, which is slightly better than ours. Take γ as soundness parameter of ZK proof of *Strain*, generating one proof for a two-party comparison takes $5l\gamma$ operations, so a bidder needs $5(n - 1)l\gamma$ operations. For auctioneer (verifier), he re-computes the comparison and verify the ZK-proof, which takes $n(n - 1)(6l + 3l\gamma)$ operations. The communication heavy is worse than our scheme.

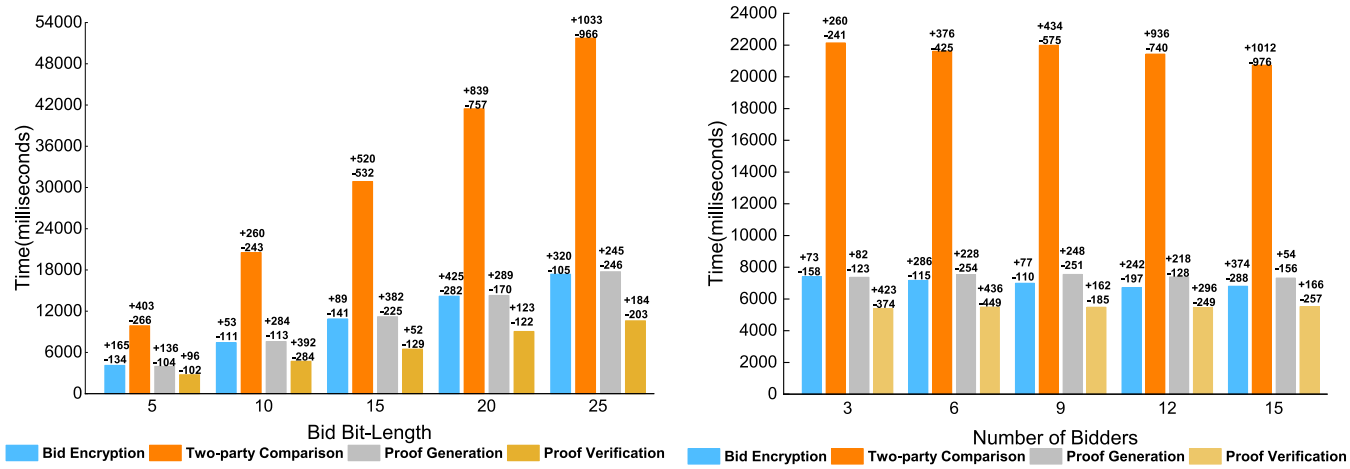
Storage Cost: A bidder needs to generate l ciphertext in the *Commit* phase. Each ciphertext is a BHE ciphertext which is a γ-tuple. Thus each bidder will generate information of size $l\gamma$ during the first phase. Since the computing circuit runs off the

TABLE I
COMPUTATION OVERLOAD

		Commit	Comparison			ZK proof		
	Item		two-party comparison	two-party winner	total	proof generation	proof verification	total
BSS	Bidder	l	-	-	-	-	$7l$	$8l$
	Auctioneer	-	$7l$	$3l+1$	$(n-1)(10l+1)$	$6l$	-	$(n-1)(10l+1)+6l$
Strain	Bidder	l	$7l$	l	$8(n-1)l$	$5(n-1)l\gamma$	-	$(n-1)(5l\gamma+8l)+l$
	Auctioneer	-	-	-	-	-	$n(n-1)(6l+3l\gamma)$	$n(n-1)(6l+3l\gamma)$

TABLE II
STORAGE COST

		Commit	Comparison			ZK proof		
	Item		two-party comparison	two-party winner	total	proof generation	proof verification	total
BSS	Bidder	$l\gamma$	-	-	-	-	$(2\gamma+6)l+3$	$(3\gamma+6)l+3$
	Auctioneer	$nl\gamma$	-	-	-	$(2\gamma+6)l+2$	-	$(2\gamma+n\gamma+6)l+2$
Strain	Bidder	l	$3l$	l	$4(n-1)l$	$5(n-1)l\gamma$	-	$(n-1)5l(\gamma+1)+l$
	Auctioneer	-	-	-	-	-	$n(n-1)(5l\gamma+2l)$	$n(n-1)(5l\gamma+2l)$



(a) Computation time needed for $n = 3$

(b) Computation time needed for $l = 10$

Fig. 5. Time cost of the BSS scheme.

chain, the participant who wants to run the circuit should download all the ciphertexts, so the cost is n times of the ciphertext size. In ZK proof phase, the prover first needs to save an element (the generator) of G. For each bit of bid, the prover generates a statement of 2 BHE ciphertexts and 6 elements of $\mathbb{Z}$. Besides, the prover needs to publish an index of the winner. So, the total storage needs $1 + (2\gamma + 6)l + 1 = (2\gamma + 6)l + 2$ numbers. The verifier downloads and saves the above proofs, and publishes whether accepting the result or not (true or false). Thus, the total storage needs $(2\gamma + 6)l + 2 + 1 = (2\gamma + 6)l + 3$ numbers. Table II shows the storage overhead on the bidder (act as a verifier in ZK proof) and auctioneer.

In *Comparison* phase of *Strain*, a bidder needs to download all the ciphertexts, generates $n - 1$ ciphertexts of his own bid under others' public key, and outputs $n - 1$ comparison results. Each ciphertext or comparison result is a l-tuple, so the total number is $4(n - 1)l$. In ZK proof phase, the bidder needs 5γ elements for one bit of two-party comparison result. The size of proof for one bidder is $5(n - 1)l\gamma$ in total. The auctioneer needs download the proofs and the all ciphertexts of comparison. So the cost equals $n(n - 1)(5l\gamma + 2l)$, which is much more than ours when n is big.

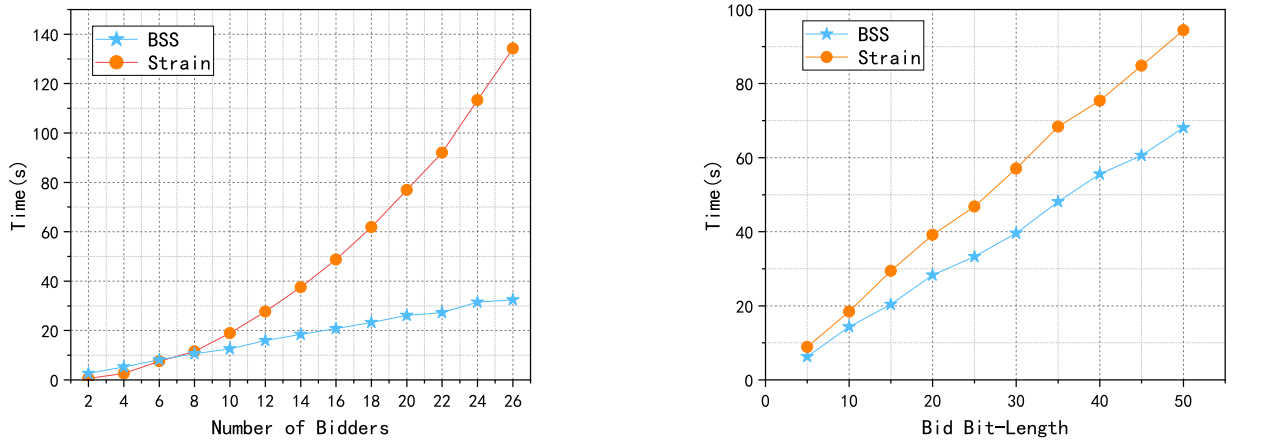
We have implemented the core scheme using JAVA. The experiment was carried out on a laptop with Intel(R) Core(TM) i5-9400 (clocked at 2.90 GHz with 8 GB RAM). The parameter $\rho = 64, \gamma = 1024, \eta = 512$ and $\tau = 2048$.

Based on the experimental results, we plotted two graphs which depict the average time needed to encrypt a bid, compare two bids, generate proof and verify the proof. First, we fix the number of bidders n at 3 and get Fig. 5(a) for different values of the bit length of bid l. Fig. 5(b) is for different number of bidders where we fix the bit length of a bid at 10. The figure shows the average time and indicates the difference between the average time and the maximum and minimum values.

It can be observed from Fig. 5 that the average time cost for encryption, proof generation and verification is irrespective to the number of bidders n. The time increases linearly with the increasing of the length of the bid l, given the number of bidders.

TABLE IV
EXECUTION TIME FOR DIFFERENT BID CIPHERTEXT LENGTHS ($n = 10$)

l	5 bits	10 bits	15 bits	20 bits	25 bits	30 bits	35 bits	40 bits	45 bits	50 bits
Strain	6.306s	14.307s	20.380s	28.319s	33.281s	39.610s	48.097s	55.564s	60.595s	68.088s
39.61 BSS	8.862s	18.433s	29.442s	39.183s	46.814s	57.046s	68.343s	75.409s	84.850s	94.414s



(a) Execution time for different number of bids ($l = 10$ bits) (b) Execution time for different bid ciphertext lengths (n=10)

Fig. 6. Comparison of the execution efficiency of BSS and Strain.

TABLE III
EXECUTION TIME FOR DIFFERENT NUMBER OF BIDS ($l = 10$ BITS)

n	2	4	6	8	10	12	14	16	18	20	22	24	26
Strain	2.615s	5.229s	8.119s	10.581s	12.541s	15.995s	18.445s	20.767s	23.251s	26.143s	27.239s	31.472s	32.465s
BSS	0.629s	2.607s	7.546s	11.528s	18.939s	27.685s	37.508s	48.736s	61.877s	76.931s	92.022s	113.336s	134.242s

Additionally, Fig. 5 also shows that the time required to encrypt is very close to the time of proof generation. This is because, in the process of proof generation, the prover needs to generate a ciphertext for each bit. Comparing with encryption, it takes less time to generate the rest statements.

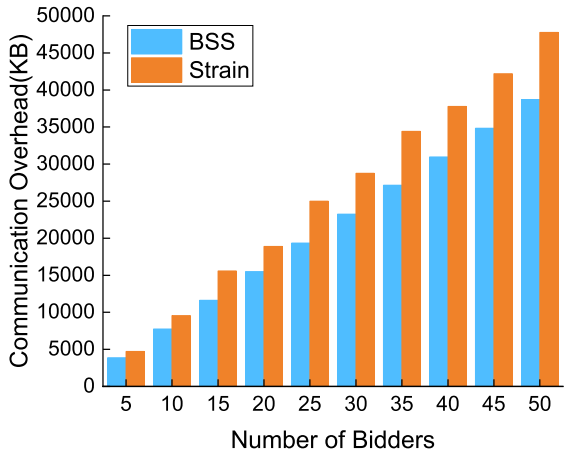
We also deployed a smart contract in Ethereum testnet to test the practicality of the BSS scheme. An auction can be executed in 16 transactions. It costs 40385 gas to send a SHA256 hash value for commitment, $1323044 * 16$ gas to send a ciphertext, and $177473 * 16$ gas to publish the ZK proof. The median gas price at the time of writing is 103 gwei (31 Oct 2021). The total cost can be computed as $(16 * (1323044 + 177473) + 40385) * 103 * 0.000000001$ and is about 2.47 ETH. This cost can be reduced to 0.29 ETH, if the blockchain merely records the commitment and the proof, and all the ciphertexts are saved in an off-chain way.

B. Performance

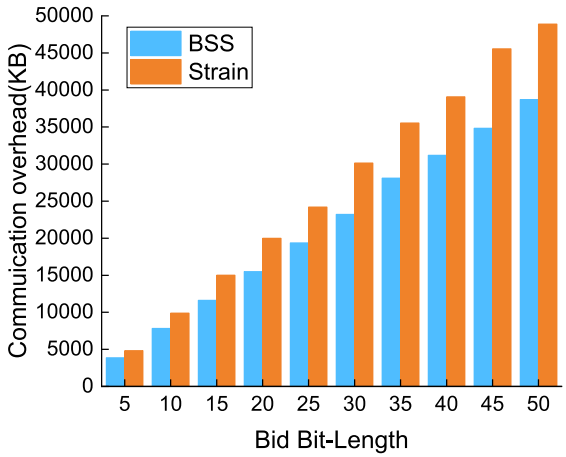
We designed two sets of experiments to compare the execution efficiency of BSS and Strain. In the first set of experiments, we fix the bid ciphertext length l to 10 b. experiments set the initial value of the number of bids to $n = 2$ and increase it in steps of 2. The execution times of BSS and Strain for different values of n are shown in Table III. In another set of experiments, we fix the value of n to be 10 and increase l by an initial value of 5. Each increase is in steps of 5. The execution times of BSS and Strain for different values of l are shown in Table IV. Finally, we obtained a comparison curve of the execution efficiency of the BSS and Strain schemes based on the experimental measurements in Fig. 6.

We can make three observations from Fig. 6(a). First, the execution times for both BSS and Strain show a continuous upward trend as the number of bids increases. It is easy to see from Figs. 4 and 5 that as the number of bids V increases, both auction protocols need to perform more comparisons of encrypted bids and execution becomes less efficient. Although auctions to achieve privacy protection come at the cost of reduced efficiency, we believe that latency in seconds is acceptable. Another observation shows that the slope of the execution efficiency curve for BSS is significantly lower than that of the Strain protocol. As the value of n increases from 2 to 26, the execution efficiency of the Strain protocol decreases by 212%, while the rate of decrease for BSS is only 11%. BSS performs fewer rounds of comparisons but achieves the same level of privacy protection as Strain. Therefore, BSS is more efficient when n takes on the same value and the improvement in efficiency becomes more pronounced as n increases.

Finally, we also observed an interesting conclusion. When the value of n is 6, the execution efficiency of both BSS



(a) Communication cost for different number of bidders($l = 10$ bits)



(b) Communication cost for different bid ciphertext lengths (n=10)

Fig. 7. Comparison of the communication cost of BSS and Strain.

and Strain scenarios is almost identical. When $\mathrm{n} < 6$, stram execution is instead more efficient than BSS. By a reasonable analysis we find that the time complexity of the function exF is higher than F in the single round bid comparison as in Figs. 4 and 5. Therefore, in the case of $\mathrm{n} < 6$, the execution efficiency of BSS is not dominant. However, because BSS effectively reduces the number of rounds compared, there is a clear advantage in execution efficiency for $\mathrm{n} > 6$, and this advantage gradually increases as n increases.

By observing Fig. 6(b), we can draw the following conclusions. First, the execution efficiency of both BSS and Strain decreases to varying degrees as l increases, and there is an almost linear positive correlation between execution time and l. The length of the bid directly affects the execution time of the homomorphic encryption algorithm used for bidding. Thus, the execution efficiency of the bid decreases as l grows. In addition, the experimental results show that the execution time difference between BSS and Strain is gradually expanding when the value of l is the same. When l is 5 bits, the difference in time overhead is 2.556s, and when l is increased to 50 bits, the difference has reached 25.326s. Thus, the efficiency of the BSS has an advantage for the same value of l, and this advantage becomes greater.

In addition, as shown in Fig. 7, we design experiments to analyse the communication overhead of Strain and BSS. The communication overhead of the privacy auction protocol comes from the number of bytes occupied by the bid ciphertexts transmitted by the bidders to the auctioneer. A larger number of bytes in the bid means a larger communication overhead is incurred. In the Strain and BSS schemes, the information to be transmitted by the bidder consists of the bid ciphertext and the hash digest as a commitment. The number of bytes of hash value output is the same in both schemes. That is, the experiment can compare the communication overhead of Strain and BSS schemes by measuring the bid ciphertext.

As shown in Fig. 7(a), we fix the bid length to 10 bits, take $n = 5$ as the initial value, and increase the number of bidders with a step size of 5. The experimental data show that the communication overhead of both schemes increases as n increases, but for all values of n, the communication overhead of our proposed BSS scheme is always lower than that of Strain.

As shown in Fig. 7(b), we fix the number of bidders to 10 and measure the effect of different bid lengths on the communication overhead. As the bid length l increases, the communication overhead of both schemes tends to grow. However, for any value of l, the communication overhead of BSS is lower than that of Strain, and this gap becomes more and more obvious as l increases.

VIII. Conclusion

In this paper, a homomorphic encryption(HE)-based bid comparison circuit is first designed to compute the winning bid in a sealed-bid auction. This circuit enables bidders to verify the correctness of the computation without requiring any bids. Then, a blockchain-based sealed-bid scheme (BSS) is proposed by using the HE-based bid comparison circuit, commitment and zero knowledge proof. Following by that, the security of BSS is formally proven. Specifically, neither the bid leakage nor the winner change can occur overwhelmingly in the auction. Finally, the performance of BSS is analysed to quantitatively estimate the efficiency and practicality of BSS.

References

[1] S. Balseiro, C. Kroer, and R. Kumar, "Contextual standard auctions with budgets: Revenue equivalence and efficiency guarantees," in *Proc. 23rd ACM Conf. Econ. Comput.*, 2021, Art. no. 476.

[2] D. Ivanov and A. Nesterov, "Identifying bid leakage in procurement auctions: Machine learning approach," in *Proc. ACM Conf. Econ. Comput.*, 2019, pp. 69–70.

[3] H. R. Varian, "Economic mechanism design for computerized agents," in *Proc. USENIX Workshop Electron. Commerce*, 1995, pp. 13–21.

[4] M. K. Franklin and M. K. Reiter, "The design and implementation of a secure auction service," *IEEE Trans. Softw. Eng.*, vol. 22, no. 5, pp. 302–312, May 1996.

[5] J. A. Montenegro, M. J. Fischer, J. Lopez, and R. Peralta, "Secure sealed-bid online auctions using discreet cryptographic proofs," *Math. Comput. Modelling*, vol. 57, no. 11/12, pp. 2583–2595, 2013.

[6] E.-O. Blass and F. Kerschbaum, "Strain: A secure auction for blockchains," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2018, pp. 87–110.

[7] J. Ma, B. Qi, and K. Lv, "Fully private auctions for the highest bid," in *Proc. ACM Turing Celebration Conf.-China*, 2019, pp. 1–6.

[8] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *Proc. IEEE Symp. Secur. Privacy*, 2016, pp. 839–858.

[9] F. Benhamouda, S. Halevi, and T. Halevi, "Supporting private data on hyperledger fabric with secure multiparty computation," *IBM J. Res. Dev.*, vol. 63, no. 2/3, pp. 3–1, 2019.

[10] H. S. Galal and A. M. Youssef, "Trustee: Full privacy preserving vickrey auction on top of ethereum," in *Proc. Cryptography Data Secur.: FC 2019 Int. Workshops, VOTING WTSC*, St. Kitts, St. Kitts and Nevis, Island, Feb. 18–22, 2019.

[11] S. Bag, F. Hao, S. F. Shahandashti, and I. G. Ray, "SEAL: Sealed-bid auction without auctioneers," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 2042–2052, 2020, doi: 10.1109/TIFS.2019.2955793.

[12] H. Li and W. Xue, "A blockchain-based sealed-bid e-auction scheme with smart contract and zero-knowledge proof," *Secur. Commun. Netw.*, vol. 2021, pp. 5523394:1–5523394:10, 2021, doi: 10.1155/2021/5523394.

[13] X. Jia, X. Song, and M. Sohail, "Effective consensus-based distributed auction scheme for secure data sharing in Internet of Things," *Symmetry*, vol. 14, no. 8, 2022, Art. no. 1664.

[14] M. Usama, O. Rehman, I. Memon, and S. Rizvi, "An efficient construction of key-dependent substitution box based on chaotic sine map," *Int. J. Distrib. Sensor Netw.*, vol. 15, no. 12, 2019, Art. no. 1550147719895957.

[15] Y. Chen, X. Tian, Q. Wang, J. Jiang, M. Li, and Q. Zhang, "SAFE: A general secure and fair auction framework for wireless markets with privacy preservation," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 3, pp. 2038–2053, May/Jun. 2022.

[16] R. Zhu, H. Liu, L. Liu, X. Liu, W. Hu, and B. Yuan, "A blockchain-based two-stage secure spectrum intelligent sensing and sharing auction mechanism," *IEEE Trans. Ind. Informat.*, vol. 18, no. 4, pp. 2773–2783, Apr. 2022.

[17] L. Luo, J. Feng, H. Yu, and G. Sun, "Blockchain-enabled two-way auction mechanism for electricity trading in internet of electric vehicles," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8105–8118, Jun. 2022.

[18] X. Ma, D. Xu, and K. Wolter, "Blockchain-enabled feedback-based combinatorial double auction for cloud markets," *Future Gener. Comput. Syst.*, vol. 127, pp. 225–239, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X21003551

[19] Z. Shi, H. Zhou, C. de Laat, and Z. Zhao, "A Bayesian game-enhanced auction model for federated cloud services using blockchain," *Future Gener. Comput. Syst.*, vol. 136, pp. 49–66, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X22001881

[20] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Comput. Surv.*, vol. 51, no. 4, pp. 1–35, 2018.

[21] D. Benarroch, Z. Brakerski, and T. Lepoint, "Fhe over the integers: Decomposed and batched in the post-quantum regime," in *Proc. IACR Int. Workshop Public Key Cryptogr.*, 2017, pp. 271–301.

[22] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2010, pp. 24–43.

[23] J. H. Cheon et al., "Batch fully homomorphic encryption over the integers," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2013, pp. 315–335.

[24] J. H. Cheon and D. Stehlé, "Fully homomophic encryption over the integers revisited," in *Proc. Annu. Int. Conf. Theory Appl. Cryptographic Techn.*, 2015, pp. 513–536.

[25] M. Fischlin, "A cost-effective pay-per-multiplication comparison method for millionaires," in *Proc. Cryptographers' Track RSA Conf.*, 2001, pp. 457–471.

[26] I. Damgård, "On $\sum$-protocols," Lecture Notes, University of Aarhus, Department for Computer Science, 2002, Art. no. 84.

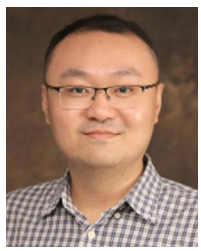
[27] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. 14th Annu. ACM Symp. Theory Comput.*, 2008, pp. 197–206.

[28] V. Buterin et al., "Ethereum white paper," *GitHub Repository*, vol. 1, pp. 22–23, 2013.

[29] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Proc. Conf. Theory Appl. Cryptographic Techn.*, 1986, pp. 186–194.

Zijian Zhang (Senior Member, IEEE) is an Associate Professor in the School of Cyberspace Science and Technology, and School of Southeast Institute of Information Technology with the Beijing Institute of Technology. His research interests include cryptographic protocol design and user behaviour analysis, and blockchain.

Xin Lu received the BE and MS degree from North China Electric Power University, Beijing, China, in 2017 and 2020. He is currently working toward the PhD degree in the School of Cyberspace Science and Technology, Beijing Institute of Technology. His current research focuses on privacy-preserving and trusted computing.



Meng Li (Senior Member, IEEE) received the PhD degree in computer science and technology from the School of Computer Science and Technology, Beijing Institute of Technology (BIT), China, in 2019. He is an Associate Professor and dean assistant with the School of Computer Science and Information Engineering, Hefei University of Technology (HFUT), China. He is also a post-doc researcher with the Department of Mathematics and HIT Center, University of Padua, Italy, where he is with the Security and PRIvacy Through Zeal (SPRITZ) research group led by Prof. Mauro Conti (IEEE Fellow). He was sponsored by ERCIM 'Alain Bensoussan' Fellowship Programme (from October 2020 to March 2021) to conduct Post-Doc research supervised by Prof. Fabio Martinelli at CNR, Italy. He was sponsored by China Scholarship Council (CSC) (from September 2017 to August 2018) for joint Ph.D. study supervised by Prof. Xiaodong Lin (IEEE Fellow) in the Broadband Communications Research (BBCR) Lab, University of Waterloo and Wilfrid Laurier University, Canada. His research interests include security, privacy, fairness, applied cryptography, cloud computing, edge computing, blockchain, and vehicular networks. In this area, he has published 70 papers in international peer-reviewed transactions, journals and conferences, including *IEEE Transactions on Information Forensics and Security*, *IEEE Transactions on Dependable and Secure Computing*, *IEEE/ACM Transactions on Networking*, *IEEE Transactions on Knowledge and Data Engineering*, *ACM Transactions on Database Systems*, *IEEE Transactions on Services Computing*, TSG, *IEEE Transactions on Industrial Informatics*, *IEEE Transactions on Vehicular Technology*, *IEEE Transactions on Network and Service Management*, *IEEE Transactions on Network Science and Engineering*, *IEEE Transactions on Green Communications and Networking*, *IEEE Communications Surveys and Tutorials*, MobiCom, ICICS, SecureComm, TrustCom, and IPCCC. He is an associate editor for IEEE TIFS, IEEE TNSM, and IEEE IoTJ.

Jincheng An is currently an assistant research fellow with Qi An Xin Technology Group Inc. He is an expert in cybersecurity and technology standardizing documents drafting. His research interests include cyber resilence theory and evaluation, data security detection.

Yang Yu received the BE and MS degrees in computer science and technology from the Beijing Institute of Technology, in 2019 and 2022, respectively. His research interests include applied cryptography, information security, privacy preservation, and blockchain technology.

Hao Yin received received the BE and MS degrees in information security and software engineering from the University of Science and Technology Beijing, in 2015 and 2018, respectively, and the PhD degree in the School of Computer Science and Technology from Beijing Institute of Technology. His research interests include applied cryptography, information security, privacy preservation, blockchain technology, and distributed consensus.

Liehuang Zhu (Senior Member, IEEE) is a professor in the School of Computer Science and Technology, Beijing Institute of Technology. He is selected into the Program for New Century Excellent Talents in University from Ministry of Education, P. R. China. His research interests include cryptographic algorithms and secure protocols, Internet of Things security, cloud computing security, Big Data privacy, mobile and Internet security, and trusted computing.

Yong Liu is currently a research fellow with Qi An Xin Technology Group Inc. He is an expert in cybersecurity strategy design. His research interests include cyber resilence theory and evaluation, data security detection, and block chain security evaluation.

Jiamou Liu received the PhD degree in computer science from the University of Auckland, Auckland, New Zealand, in 2010. He is a senior lecturer with the School of Computer Science, University of Auckland. He was a senior lecturer with the Auckland University of Technology, Auckland, from 2011 to 2015 and a researcher with the Department of Computer Science, Leipzig University, Leipzig, Germany, from 2009 to 2010. His current research interests include social network analysis, multiagent systems, and algorithms.

Bakh Khoussainov received the PhD degree in mathematics from the Algebra and Logic Department, Novosibirsk University, USSR. He is currently a professor with the Computer Science Department, the University of Auckland, New Zealand. His research interests include computable algebraic systems and model theory, automata and automatic structures, games on finite graphs and complexity, abstract data types and algebraic specifications, computably enumerable reals and randomness. He is also an editor of the Journal for Symbolic Logic.