Article:

Bag, Samiran, Hao, Feng, Shahandashti, Siamak F. orcid.org/0000-0002-5284-6847 et al. (1 more author) (2019) SEAL: Sealed-Bid Auction without Auctioneers. Information Forensics and Security, IEEE Transactions on. 2042–2052. ISSN 1556-6013

https://doi.org/10.1109/TIFS.2019.2955793

SEAL: Sealed-Bid Auction Without Auctioneers

Samiran Bag, Feng Hao, Siamak F. Shahandashti, and Indranil G. Ray

***Abstract*—We propose the first auctioneer-free sealed-bid auction protocol with a linear computation and communication complexity $O(c)$, c being the bit length of the bid price. Our protocol, called Self-Enforcing Auction Lot (SEAL), operates in a decentralized setting, where bidders jointly compute the maximum bid while preserving the privacy of losing bids. In our protocol, we do not require any secret channels between participants. All operations are publicly verifiable; everyone including third-party observers is able to verify the integrity of the auction outcome. Upon learning the highest bid, the winner comes forward with a proof to prove that she is the real winner. Based on the proof, everyone is able to check if there is only one winner or there is a tie. While our main protocol works with the first-price sealed-bid, it can be easily extended to support the second-price sealed-bid (also known as the Vickrey auction), revealing only the winner and the second highest bid, while keeping the highest bid and all other bids secret. To the best of our knowledge, this work establishes to date the best computation and communication complexity for sealed-bid auction schemes without involving any auctioneer.**

I. Introduction

Auction is a method of allocating scarce resources based on competition. Goods or services are sold by offering them to bidding, and the winner is the bidder who has the highest bid. From the allocation of bandwidth spectrum to the sales of antiques, painting and rare collectibles, auction has become a prevailing practice in our society. It is also commonly used by governments, e.g., long-term securities are sold in weekly auctions conducted by the U.S. Treasury to finance the borrowing needs of the government [1].

In general, there are two types of auctions: open cry and sealed bid. In an open-cry auction, the price may be ascending from a reserve price until there is only one bidder left, or descending from a high price until the first bidder comes forward willing to pay at that price. The former is often known as "English auction" while the later "Dutch auction". In a sealed-bid auction, each bidder hands over a sealed envelope containing their secret bid to an auctioneer. The auctioneer opens all envelopes and declares the highest bidder as the winner, while keeping losing bids secret. In the first-price sealed-bid, the winner pays for the highest bid, but in the second-price sealed-bid, the winner only needs to pay the second highest bid. The second-price sealed-bid is also called the Vickrey auction (named after William Vickrey).

There has been extensive research in auction theory to show the inherent relations between various auction schemes [1]. In particular, the Dutch auction is shown to be strategically equivalent to the first-price sealed-bid auction based on game theory. The English auction is equivalent to the Vickrey auction under the assumption that bidders evaluate the value of the item in private. The Vickrey auction is extremely important in auction theory, as it is *strategy-proof*. In other words, when the evaluated values are all private, the best strategy for bidders is to bid their true evaluation. William Vickrey first developed the theory for this auction scheme and received a Nobel prize in 1996. The scheme is thereafter named after him in recognition of his ground-breaking contributions.

Unfortunately, the Vickrey auction is rarely used in practice [2]. This is largely due to two reasons, both of which are related to the distrust on the auctioneer. First, a dishonest auctioneer might surreptitiously substitute the second highest price to one slightly below the top price to increase the revenue. Second, a dishonest auctioneer might disclose the losing bids to other parties, since the true evaluation of the auction item is considered a commercial secret.

Since the early work in this field by Franklin and Reiter in 1996 [3], many sealed-bid e-auction schemes have been proposed [2], [4]–[10]. However, the past schemes generally assume the role of an auctioneer (as in traditional sealed-bid auctions). To mitigate the trust problem about the auctioneer, researchers propose to apply threshold cryptography or multi-party computation (MPC) techniques [10], [11] to distribute the trust from a single auctioneer to two or several. However, one can never rule out the possibility that the auctioneers may collude all together to compromise the privacy of the bids [7].

In this paper, we propose to address the trust issue about the auctioneer by removing the need for any auctioneer completely from an auction system. We consider a totally decentralized setting with public verifiability. In this setting, the auction is run by the bidders themselves without involving any auctioneer, and all operations are publicly verifiable without any secret channels. Clearly, generic MPC techniques [12] that require pairwise secret channels are unsuitable for our purpose. Our setting is similar to the "bidder-resolved auction" proposed by Brandt [2], [7], [13], [14]), but Brandt's schemes also involve a seller, who actively takes part in the protocol and is trusted not to collude with bidders. Furthermore, Brandt's schemes incur an exponential system complexity $O(2^c)$ for computation and bandwidth usage, c being the bit length of the bid price. No sealed-bid auction scheme in the past has achieved the linear system complexity $O(c)$ in a decentralized setting.

Our contributions in this paper are summarized below.

- We propose the first decentralized sealed-bid auction protocol with a linear computation and communication complexity $O(c)$, c being the bit length of the bid price. Our protocol is publicly verifiable without involving any auctioneer or requiring any secret channels between bidders.

S. Bag, F. Hao and I.G. Ray are with the Department of Computer Science, University of Warwick, United Kingdom. E-mails: {samiran.bag, feng.hao, indranil.ghosh-ray}@warwick.ac.uk. S.F. Shahandashti is with the Department of Computer Science, University of York, United Kingdom. Email: siamak.shahandashti@york.ac.uk. This work is supported by the ERC starting grant, No. 306994, and Royal Society grant, ICA/R1/180226.

- We present security proofs to prove the correctness, and privacy aspects of the protocol, as well as performing analysis on the system complexity.
- We show how our first-prize sealed-bid auction scheme can be extended to support second-prize sealed-bid auctions (Vickrey auction) while maintaining the linear system complexity.

II. Preliminaries

A. Communication model

Sealed-bid auction can be considered a special instance of a secure multiparty computation (MPC) problem, in which participants jointly compute the highest bid without revealing losing bids. In a typical MPC setting, the communication model assumes that there exist pairwise secret and authenticated channels between participants, in addition to an authenticated public channel [8], [10], [15]. In the real-world example of applying MPC to the Danish sugar beets auction, three appointed auctioneers set up pairs of public and private key, which were used to establish secure point-to-point channels between the auctioneers [9]. The existence of secret channels is not considered a problem in [9], since by design all farmers are required to trust auctioneers – more specifically, trusting that at least 2 out of the 3 auctioneers are honest.

In a decentralized setting without any auctioneer, we consider it desirable to remove any secret channels, so all operations are publicly verifiable. This is especially important for third-party observers (e.g., the seller) who are not involved in the protocol, but still want to verify the integrity of the auction process. Therefore, in our model, we only assume an authenticated public channel available to all participants. Such a channel can be realized using physical means or a public bulletin board, as described in [6], [16], [17].

B. A Modified Anonymous Veto Protocol

A basic building block in our auction protocol is a primitive that securely computes the logical-OR of binary inputs without revealing each individual bit. We choose to modify Hao-Zieliński's Anonymous Veto network (AV-net) protocol [18] for our purpose. This protocol is chosen for its optimal efficiency in terms of the number of rounds, the computation and the communication bandwidth. However, we need to modify the original scheme to make it applicable to our system.

The modified AV-net protocol works as follows. Assume a group of n voters who wish to find out if there is one voter who would like to veto a motion. In other words, they wish to securely compute the logical-OR function of a number of input bits, each bit coming from a separate entity. Let G be a group of p elements in which the Decisional Diffie-Hellman problem is assumed to be intractable. Let g be a random generator of G. All computations in G are modular operations with respect to a prime modulus q, but we omit the $\bmod q$ notation for simplicity. Each voter V_i holds a secret bit $v_i \in \{0,1\}$, and they compute the logical-OR $\bigvee_{i=1}^{n} v_i$ in two rounds. For any two integers a and b, where $a < b$, we denote by $[a,b]$, the set: $\{a, a+1, \ldots, b\}$.

Round I: Each voter $V_i : i \in [1,n]$ chooses two random elements $(x_i, r_i) \in \mathbb{Z}_p^2$ and computes $X_i = g^{x_i}, R_i = g^{r_i}$. V_i posts (X_i, R_i) on the public bulletin board, together with non-interactive zero-knowledge (NIZK) proofs [19], [20] to prove the knowledge of x_i and r_i, respectively, using Schnorr's signature [18].

Round II: Each voter $V_i : i \in [1,n]$ computes $Y_i = \prod_{j=1}^{i-1} X_j / \prod_{j=i+1}^{n} X_j$. V_i also computes an encrypted ballot b_i and posts it on the bulletin board together with a NIZK proof to show b_i is well-formed.

$$b_i = \begin{cases} Y_i^{x_i} & \text{if } v_i = 0 \\ R_i^{x_i} & \text{if } v_i = 1 \end{cases}$$

The NIZK proof in the second round is to prove the following statement: $(b_i = Y_i^{x_i}) \vee (b_i = R_i^{x_i})$. Note that the first term is equivalent to proving $\{X_i, Y_i, b_i = Y_i^{x_i}\}$ is a DDH tuple and the second term equivalent to proving $\{X_i, R_i, b_i = R_i^{x_i}\}$ is a DDH tuple. Thus, the NIZK proof on the well-formedness of b_i is a disjunctive proof of two sub-statements. More details on the zero-knowledge proofs can be found in the Appendix.

After the second round, everyone can compute $B = \prod_{i=1}^{n} b_i$ after fetching the ballots $b_i : i \in [1,n]$ from the bulletin board. It is easy to see that if every input is 0, $B = 1$; otherwise, if at least one input is 1, $B \neq 1$ with overwhelming probability (i.e., B is a random element in G). Hence, the logical-OR of all input bits has been securely computed by all participants without revealing the value of each individual bit. The key element in this protocol is the fact that $\prod_i Y_i^{x_i} = 1$, i.e., the random factors x_i are all canceled out. For a proof of this fact, the reader is referred to [18].

Our modified veto protocol differs from Hao-Zieliński's AV-net protocol [18] in that we use two random variables, x_i and r_i, while AV-net only uses one. This requires more computation from each participant, but the (asymptotic) computation and communication complexity remains the same as AV-net. In AV-net, the '1' vote (or the 'veto' vote) is encoded by raising a pre-defined generator to the power of a random variable, while in the modified veto protocol, the '1' vote is encoded by raising a random generator to the power of a pre-defined exponent (namely, x_i). This modification allows us to effectively integrate the veto protocol into the e-auction scheme as some zero-knowledge proofs will require proving the equality of the exponents. This should become clear after we explain the details of the a-auction scheme in the next section. In the rest of the paper, we will simply refer to the modified anonymous veto protocol as the "veto protocol".

C. Intuition Behind the Scheme

First, we explain the high-level intuition of the protocol. Assume there are n bidders denoted as V_i, $i \in [1,n]$. The binary representation of each bid contains c bits.[1] For any bid price $p_i, i \in [1,n]$, let $p_{i1}||p_{i2}||\ldots||p_{ic}$ be the binary representation of p_i, with $||$ denoting concatenation.

[1] The bid may fall in a sub-range of $[0, 2^c - 1]$. We do not consider this detail since we are mainly concerned with the system complexity of the protocol.

The protocol has two phases. In the first phase, every bidder V_i commits their bid p_i on a public bulletin board. In the second phase, all bidders jointly compute the maximum bid bit-by-bit, starting from the most significant bit.

As an example, we start from the most significant bit position on the left, as shown in Figure 1. We denote this starting position as $j = 1$. We use the modified anonymous veto protocol as a basic building block to compute the logical OR of the input bits without revealing individual bits. Now every bidder V_i uses a bit d_{ij} ($j = 1$) as the input to this veto protocol, with a NIZK proof to prove that $d_{ij} = p_{ij}$ without revealing the committed bit. All bidders are able to compute the logical OR of all d_{ij} bits for the first bit position ($j = 1$).

$$T_j = d_{1j} \vee d_{2j} \vee \ldots \vee d_{nj} \tag{1}$$

At each iteration for $j = 1, 2, \ldots$, when the logical OR result T_j is 1, we call this bit position a *deciding* position; otherwise, we call it a *non-deciding* position. We use $\overleftarrow{j}$ to denote the deciding position that immediately precedes the jth bit position ($\overleftarrow{j}$ will always return a valid bit position in the context of our protocol after the first *deciding* position is past).

The first *deciding* position is called a *junction* (see Figure 1). This is where $T_j = 1$ ($j = 1, 2, \ldots$) for the first time. After this junction, bidders still use the veto protocol to compute the logical OR of the input bits, but the rule for specifying the input bit is different. Instead of using a bit that must be the same as the committed bit, each bidder uses the bit $d_{ij} = p_{ij} \wedge d_{i\overleftarrow{j}}$, where $\overleftarrow{j}$ denotes the previous *deciding* position before j. In addition, each bidder provides a zero-knowledge proof to prove the following statement.

$$\left(d_{ij} = p_{ij} \text{ AND } d_{i\overleftarrow{j}} = 1\right) \text{ OR } \left(d_{ij} = 0 \text{ AND } d_{i\overleftarrow{j}} = 0\right) \tag{2}$$

The zero-knowledge proof statement in Equation 2 is the key to our auction protocol. Essentially, it enforces the following behavior: at each *deciding* bit position, if the bidder V_i's committed bit in the corresponding position is 0, it means V_i has lost, hence in all subsequent bit positions, V_i is enforced to use 0 as the input to the veto protocol. However, the bidders do not need to reveal whether they have already lost or they are still in the race. Losing bidders simply follow through the rest auction procedure so no information about their bids will be revealed. The guarantee of the privacy for the losing bids provides an incentive for losing bidders to follow through the rest of the process. In practice, bidders often pay a deposit and get it refunded after showing that they have honestly followed the auction rules. This gives another incentive for all bidders to complete the whole auction protocol. At the end of the auction, every bidder, as well as any observer, can compute the maximum bid, but without learning losing bids. In the next section, we will explain details of building a secure sealed-bid auction scheme based on this intuition.

III. Our main protocol

A. Requirements

Our main protocol is called Self-Enforcing Auction Lot[2]. We design the protocol to fulfill the following requirements.

1) *Public verifiability.* All operations in the auction process are publicly verifiable. No secret channels are required between participants. Only an authenticated public channel is available to all participants (which can be realized by using a public bulletin board as in [8], [16], [17]).
2) *Correctness.* The protocol is guaranteed to output the highest bid with proofs that everyone is able to verify.
3) *Losing-bid privacy.* While the protocol outputs the highest bid, the privacy of losing bids should be preserved, as we formally define below.

In the generic definition of input privacy in MPC, each participant is limited to learn nothing more than their own input and the output of the function [12]. Based on this, we define *inclusive-privacy* for sealed-bid auction, in which participants perform a secure computation of a *max* function. We call it "inclusive" as the function includes inputs p_i from all participants, $i \in [1, n]$.

Definition 1 (Inclusive-privacy): In an auction protocol that satisfies *inclusive-privacy*, each bidder V_i learns nothing more than their own input and the output of the function $f_{\max}(p_1, \ldots, p_n)$.

We slightly modify the above definition by excluding the bidder's own input from the inputs of the function and introduce *exclusive-privacy* as defined below. Here we consider a more general case where a set of bidders may collude together. Let C be a set of colluding bidders and H be the rest of the bidders, i.e. $C \cup H = [1, n]$. Let θ be the size of H and $h_i \in H$, $i \in [1, \theta]$. At minimum, C contains only one bidder, but in the general case, it may contain any number of bidders.

Definition 2 (Exclusive-privacy): In an auction protocol that satisfies *exclusive-privacy*, C learns nothing more than their own input and the output of the function $f_{\max}(p_{h_1}, \ldots, p_{h_\theta})$.

The generic definition of privacy (namely, inclusive-privacy) is commonly used in the past MPC literature, whereas we consider the latter definition of privacy, i.e. exclusive-privacy in the publicly verifiable setting. Protocols that use general MPC techniques have been shown to be able to satisfy the strong notion of inclusive-privacy, but this relies on several important assumptions [12]. First of all, the privacy of inputs depends on pairwise secret channels between participants, which however do not exist in our setting. Second, normally the majority of the participants are assumed to be honest. In our case, we consider the vast majority of participants (up to $n - 2$) may be dishonest; in the extreme case, as long as there is at least another honest bidder in H, the bidder's privacy should still be preserved (if all bidders except one are dishonest, the privacy guarantee for the remaining bid would not be meaningful since dishonest bidders can trivially find out the bid by supplying zeros as their own inputs). In such a

[2] In the auction terminology, a "lot" is an item or set of items for sale.

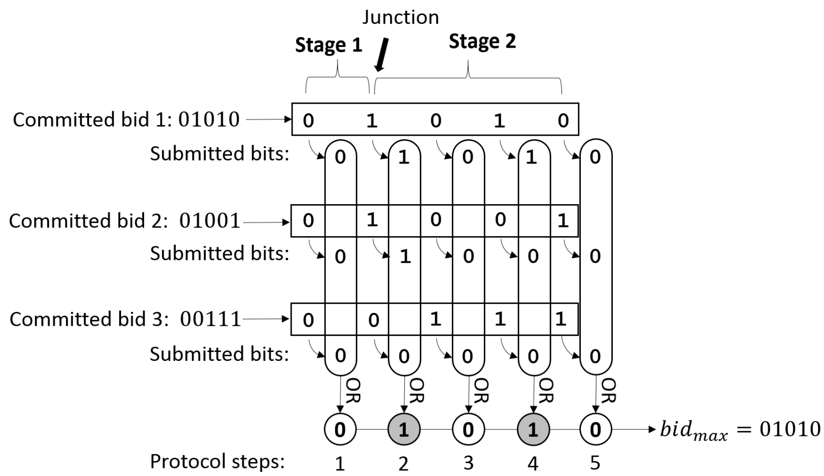


Fig. 1: Example of an execution of our protocol, starting with three bids: 10, 9, and 7, and calculating the maximum bid, i.e., 10, in five steps.

setting, general MPC protocols cannot assure any privacy of inputs at all.

Therefore, although *exclusive-privacy* is a slightly weaker notion of privacy, our protocol guarantees a much stronger assurance of privacy in practice than general MPC protocols in a setting where no pairwise secret channels exist and a vast majority of participants may be compromised. We should also highlight that even with an ideal MPC protocol (say based on a simulated trusted third party) that provides *inclusive-privacy*, colluding bidders can always find out the maximum of the inputs of the non-colluding set by supplying 0s as their inputs. Hence, the difference between the two notions is subtle. As we will explain in Section III-C, allowing a bidder to find out the maximum of the other bids can prove crucially useful in the context of an auction system, especially in constructing an efficient Vickrey auction scheme and in detecting a tie. In the following section, we will present a decentralized first-price sealed-bid auction protocol that guarantees exclusive-privacy, and in the meanwhile achieves a linear system complexity with respect to the bit length of the bid.

B. Protocol

Let $G = \langle g \rangle$ be the same group mentioned in Section II-B. Our protocol has c iterations, c being the number of bits in the binary representation of bid-prices. The bid price by V_i is expressed in the binary form as: $p_i = p_{i1}||p_{i2}||\dots||p_{ic}$, where $p_{ij} \in \{0, 1\}$, with p_{i1} being the most significant bit and p_{ic} the least significant bit.

*a) **Commit**:* In the Commit phase, bidders commit their bids to the public bulletin board. Each bidder V_i computes c commitments, each one for a bit of p_{ij}, for $j \in [1, c]$. In order to do this, V_i selects random $\alpha_{ij}, \beta_{ij} \in_R \mathbb{Z}_p, j \in [1, c]$ and computes c individual commitments for c distinct bits of p_{ij} as $\chi_i = \{\varepsilon_{ij} : j \in [1, c]\}$, where :

$$\varepsilon_{ij} = \langle g^{\alpha_{ij}\beta_{ij}} g^{p_{ij}}, g^{\alpha_{ij}}, g^{\beta_{ij}} \rangle, j \in [1, c]$$

If the Decisional Diffie Hellman assumption holds true in G, $(g^{\alpha_{ij}}, g^{\beta_{ij}}, g^{\alpha_{ij}\beta_{ij}}) \overset{c}{\approx} (g^{\alpha_{ij}}, g^{\beta_{ij}}, g^{\alpha_{ij}\beta_{ij}} g)$ [21], where $\overset{c}{\approx}$ denotes computational indistinguishability. Hence, the commitment would not reveal the value of the committed bit p_{ij}, for all $i \in [1, n], j \in [1, c]$. Each bidder V_i posts a NIZK proof of well formedness of each committed bit in the form of ε_{ij}. The NIZK proof shows p_{ij} is either 0 or 1 without revealing which one. The construction of this one-of-two NIZK proof can be found in [22] and it is also elaborated in the Appendix.

In the second phase, bidders jointly compute the maximum bid without revealing other bids. This proceeds in two stages as indicated in Figure 1.

*b) **Stage 1**:* Bidders start from the most significant bit position $j = 1$, and move to the less significant bit positions bit-by-bit until they reach a junction where the logical-OR computation at that bit position is 1 for the first time (see an example of the junction in Figure 1). In the jth bit position, bidders apply the two-round anonymous veto protocol described in Section II-B with private binary inputs d_{ij}, for $i \in [1, n]$, with a zero-knowledge proof to prove that the input bit is the same as the committed one, i.e., $d_{ij} = p_{ij}$.

Round 1: Each bidder $V_i, i \in [1, n]$ selects two private keys $x_{ij}, r_{ij} \in \mathbb{Z}_p$, stores the private keys and publishes the corresponding public keys $Pub_{ij} = (X_{ij}, R_{ij}) = (g^{x_{ij}}, g^{r_{ij}})$ on the bulletin board. The bidder V_i also publishes NIZK proofs of knowledge of $x_{ij} = \log_g X_{ij}$ and $r_{ij} = \log_g R_{ij}$ (See the Appendix for the construction of this NIZK proof).

Round 2: Each bidder $V_i, i \in [1, n]$ computes a cryptogram

$$b_{ij} = \begin{cases} Y_{ij}^{x_{ij}} = g^{x_{ij}y_{ij}} & \text{if } p_{ij} = 0 \text{ (0-cryptogram);} \\ R_{ij}^{x_{ij}} = g^{x_{ij}r_{ij}} & \text{if } p_{ij} = 1 \text{ (1-cryptogram).} \end{cases}$$

where $Y_{ij} = g^{y_{ij}} = \prod_{k=1}^{i-1} g^{x_{kj}} / \prod_{k=i+1}^{n} g^{x_{kj}} = g^{\sum_{k=1}^{i-1} x_{kj} - \sum_{k=i+1}^{n} x_{kj}}$. V_i also computes a proof π_{ij} of well-formedness of b_{ij}. V_i posts b_{ij} and π_{ij} on the bulletin board.

Here, b_{ij} is an encrypted ciphertext of d_{ij}, and π_{ij} is the NIZK proof that serves to prove $d_{ij} = p_{ij}$ without revealing the committed bit. Effectively, π_{ij} is a proof of the following logical statement.

$$(d_{ij} = 0 \wedge p_{ij} = 0) \vee (d_{ij} = 1 \wedge p_{ij} = 1)$$

The public parameters are $X_{ij} = g^{x_{ij}}, Y_{ij} = g^{y_{ij}}, R_{ij} = g^{r_{ij}}$. Let us assume $\varepsilon_{ij} = \langle c_{ij}, A_{ij}, B_{ij}\rangle$. If $d_{ij} = 0$, (X_{ij}, Y_{ij}, b_{ij}) and (A_{ij}, B_{ij}, c_{ij}) are two DDH tuples. Also if $d_{ij} = 1$, (X_{ij}, R_{ij}, b_{ij}) and $(A_{ij}, B_{ij}, (c_{ij}/g))$ are two DDH tuples as well. So we have to prove a statement that is a logical-OR of a pair logical-AND sub-statements such that each of the two sub-statements is logical-AND of two DDH tuples. Thus, the above statement is equivalent to the following statement (see [23], [24] and the appendix for the construction of this type of NIZK proof):

$$((b_{ij} = g^{x_{ij}y_{ij}} \wedge X_{ij} = g^{x_{ij}} \wedge Y_{ij} = g^{y_{ij}}) \wedge (c_{ij} = g^{\alpha_{ij}\beta_{ij}} \wedge A_{ij} = g^{\alpha_{ij}} \wedge B_{ij} = g^{\beta_{ij}})) \vee ((b_{ij} = g^{x_{ij}r_{ij}} \wedge X_{ij} = g^{x_{ij}} \wedge R_{ij} = g^{r_{ij}}) \wedge (c_{ij} = g^{\alpha_{ij}\beta_{ij}} g \wedge A_{ij} = g^{\alpha_{ij}} \wedge B_{ij} = g^{\beta_{ij}}))$$

After the two rounds, all the bidders verify that the NIZK proofs are correct, and then compute the logical-OR of the input bits d_{ij} for the jth position: $T_j = \bigvee_{i=1}^{n} d_{ij}$. This logical-OR computation is realized by multiplying b_{ij}, $i \in [1, n]$ and comparing the result against 1 (see the veto protocol in Section II-B). If $T_j = 0$, all bidders remain in Stage 1, and move on to compute the logical-OR of the next bit position. However, if $T_j = 1$, this means the junction is reached, and all bidders move to Stage 2.

*c) **Stage 2:*** Stage 2 is almost the same as Stage 1, except that d_{ij} is defined differently. Instead of using a bit that must be the same as the committed one, every bidder now uses $d_{ij} = p_{ij} \wedge d_{i\overleftarrow{j}}$ where $d_{i\overleftarrow{j}}$ is the bit that the bidder used in the previous *deciding* bit position. For the completeness, we describe the full details of Stage 2 below. Assume this stage starts from the jth position, and iterates towards the lest significant bit position until $j = c$.

Round 1: Each bidder $V_i, i \in [1, n]$ selects two private keys $x_{ij}, r_{ij} \in \mathbb{Z}_p$, stores the private keys and publishes the corresponding public keys $Pub_{ij} = (X_{ij}, R_{ij}) = (g^{x_{ij}}, g^{r_{ij}})$ on the bulletin board. The bidder V_i also publishes NIZK proofs of knowledge of $x_{ij} = \log_g X_{ij}$ and $r_{ij} = \log_g R_{ij}$.

Round 2: each bidder $V_i, i \in [1, n]$ chooses a bit $d_{ij} = p_{ij} \wedge d_{i\overleftarrow{j}}$ and computes a cryptogram of d_{ij}.

$$b_{ij} = \begin{cases} Y_{ij}^{x_{ij}} = g^{x_{ij}y_{ij}} & \text{if } p_{ij} \wedge d_{i\overleftarrow{j}} = 0;\ (\text{0-cryptogram}) \\ \\ R_{ij}^{x_{ij}} = g^{x_{ij}r_{ij}} & \text{if } p_{ij} \wedge d_{i\overleftarrow{j}} = 1.\ (\text{1-cryptogram}) \end{cases}$$

where $Y_{ij} = g^{y_{ij}} = \prod_{k=1}^{i-1} g^{x_{kj}} / \prod_{k=i+1}^{n} g^{x_{kj}} = g^{\sum_{k=1}^{i-1} x_{kj} - \sum_{k=i+1}^{n} x_{kj}}$. V_i also computes a proof π_{ij} of well-formedness of b_{ij}. V_i posts b_{ij} and π_{ij} on the bulletin board.

The NIZK proof π_{ij} is a proof of the following logical statement:

$$(d_{ij} = 1 \wedge (p_{ij} \wedge d_{i\overleftarrow{j}}) = 1) \vee (d_{ij} = 0 \wedge (p_{ij} \wedge d_{i\overleftarrow{j}}) = 0)$$

This is equivalent to proving the following logical statement. $(d_{ij} = 1 \wedge p_{ij} = 1 \wedge d_{i\overleftarrow{j}} = 1) \vee (d_{ij} = 0 \wedge p_{ij} = 0 \wedge d_{i\overleftarrow{j}} = 1) \vee (d_{ij} = 0 \wedge d_{i\overleftarrow{j}} = 0)$.

Let us assume $\varepsilon_{ij} = \langle c_{ij}, A_{ij}, B_{ij}\rangle$. We express the above logical statement in terms of the ciphertexts.

$$\begin{aligned} &(b_{ij} = g^{x_{ij}r_{ij}} \wedge c_{ij} = g^{\alpha_{ij}\beta_{ij}} \cdot g \wedge b_{i\overleftarrow{j}} = g^{x_{i\overleftarrow{j}} r_{i\overleftarrow{j}}}) \\ \vee\ &(b_{ij} = g^{x_{ij}y_{ij}} \wedge c_{ij} = g^{\alpha_{ij}\beta_{ij}} \wedge b_{i\overleftarrow{j}} = g^{x_{i\overleftarrow{j}} r_{i\overleftarrow{j}}}) \\ \vee\ &(b_{ij} = g^{x_{ij}y_{ij}} \wedge b_{i\overleftarrow{j}} = g^{x_{i\overleftarrow{j}} y_{i\overleftarrow{j}}}) \end{aligned} \tag{3}$$

The public parameters are $X_{ij} = g^{x_{ij}}, Y_{ij} = g^{y_{ij}}, R_{ij} = g^{r_{ij}}, A_{ij} = g^{\alpha_{ij}}$ and $B_{ij} = g^{\beta_{ij}}, X_{i\overleftarrow{j}} = g^{x_{i\overleftarrow{j}}}, Y_{i\overleftarrow{j}} = g^{y_{i\overleftarrow{j}}}, R_{i\overleftarrow{j}} = g^{r_{i\overleftarrow{j}}}$. Notice that if $d_{ij} = 1, p_{ij} = 1$ and $d_{i\overleftarrow{j}} = 1$, (X_{ij}, R_{ij}, b_{ij}), $(A_{ij}, B_{ij}, c_{ij}/g)$ and $(X_{i\overleftarrow{j}}, R_{i\overleftarrow{j}}, b_{i\overleftarrow{j}})$ are three DDH tuples. Also if $d_{ij} = 0, p_{ij} = 0$ and $d_{i\overleftarrow{j}} = 1$, (X_{ij}, Y_{ij}, b_{ij}), (A_{ij}, B_{ij}, c_{ij}) and $(X_{i\overleftarrow{j}}, R_{i\overleftarrow{j}}, b_{i\overleftarrow{j}})$ are three DDH tuples as well. Lastly, if $d_{ij} = 0$ and $d_{i\overleftarrow{j}} = 0$, (X_{ij}, Y_{ij}, b_{ij}) and $(X_{i\overleftarrow{j}}, Y_{i\overleftarrow{j}}, b_{i\overleftarrow{j}})$ are two DDH tuples. So we have to prove a statement that is a logical-OR of three logical-AND sub-statements. In each of the three sub-statements, we essentially prove the items form valid DDH tuples such that each of the three sub-statements is a logical-AND of two/three statements in the form of DDH tuples. The construction of a NIZK proof for the above statement can be found in [23], [24]. It is also described in the Appendix.

After the second round, all bidders, as well as anyone else with a read access to the bulletin board, can check all NIZK proofs, and compute $T_j = \bigvee_{i=1}^{n} d_{ij}$ based on the veto protocol described in Section II-B. The bidders follow the same procedure to iterate through the rest of bit positions. The logical-OR output from each of the c bit positions constitutes the binary representation of the highest bid, i.e. the highest bid is $T_1||T_2||\cdots||T_c$ in binary format. Once, the highest bid is computed, the winning bidder V_w can come forward and prove that she is indeed the real winner either by opening her commitments $\{\varepsilon_{wj} : j \in [1, c]\}$ or by revealing the randomness $x_{w\kappa} = \log_g X_{w\kappa}$ used in the last *deciding* bit position allowing everyone else to decipher the cryptogram submitted by her in the iteration corresponding to the last *deciding* bit position. It is easy to see that only the winner(s) would submit 1-cryptogram(s) in the iteration corresponding to the last *deciding* bit position. Based on x_{wk}, everyone is able to verify if there is only one winner of if there is a tie.

C. Extension to Vickrey auction

Our main protocol is described for the first-price sealed-bid auction. A straightforward way to extend it to support second-price sealed-bid (i.e., Vickrey auction) works as follows. The protocol is first run to identify the highest bid and the winner, and then run the second time with the winner excluded to compute the second-highest bid. The bidder who commits the second-highest bid remains anonymous. The winner pays the second-highest bid in the end. (Imagine an MPC protocol on the *max* function that satisfies the generic *inclusive privacy* in Definition 1, this would naturally be the method of extension to support the second-price sealed-bid auction.) However, the highest bid will be revealed, which is not strictly necessary, and may cause some privacy concerns.

We describe a more efficient and privacy-preserving method to support the Vickrey auction. In this method, the bit iterations will only need to be run once and the highest bid remains secret. This method is only possible because it builds on the *exclusive privacy* in Definition 2. As we will show in the proof of Theorem 2, at each jth bit iteration, every bidder V_k learns nothing more than $\bigvee_{i\in[1,n]\setminus k} d_{ij}$. Therefore, at each bit iteration, the bidder who remains a winner can learn if she is the only winner in the race. Thus, if the bidder finds that she has submitted the sole one bit in that bit iteration and thus has become a confirmed winner, she declares herself as the winner and steps aside to let other bidders continue. Those losing bidders reset the output of that winning iteration to be 0 and make it a *non deciding* iteration. Losing bidders then continue executing the rest of the steps as specified in the main protocol. This would reveal the next highest bid, while hiding the $c-j$ least significant bits of the highest bid.

IV. Analysis of the SEAL Protocol

In this section, we present security proofs to prove the correctness and the privacy aspects of our protocol. We will focus on the main protocol for the first-prize sealed bid. The same proofs apply to the second-prize sealed bid straightforwardly.

A. Correctness

The following theorem proves that our scheme is correct and it yields the highest bid price of the winner.

Theorem 1: The e-auction scheme discussed above is correct.

Proof: Let us assume the c bits output by the protocol are $last[1 \to c]$. We need to show that $last[j] = p_{wj}$ for all $j \in [1, c]$, where p_w is the winning bid of the winning bidder V_w. Let m be the last iteration of Stage 1. Hence, according to the protocol $\bigvee_{i=1}^{n} p_{im} = 1$ and for all $j \in [1, m-1], last[j] = \bigvee_{i=1}^{n} p_{ij} = 0$. This essentially means that $last[j] = p_{wj}$ for $j \in [1, m]$.

Now, after iteration m the algorithm shifts from Stage 1 to Stage 2. In Stage 2, all cryptograms generated by a bidder V_i correspond to the logical-AND of the bit at the current position (d_{ij}) and the value of the bit $d_{i\overleftarrow{j}}$ submitted in the most recent *deciding* iteration $\overleftarrow{j}$ such that $last[\overleftarrow{j}] = 1$ and for all $t \in [\overleftarrow{j}+1, j-1]$, $last[t] = 0$. Thus, we need to prove that in any iteration j in Stage 2 if $\bigvee_{i=1}^{n} d_{ij} = 1$, then $p_{wj} = 1$ and if $\bigvee_{i=1}^{n} d_{ij} = 0$, then $p_{wj} = 0$. We prove it by method of induction. Without loss of generality, we may assume that $last[t] = p_{wt}, \forall t \in [1, j-1]$. So, we need to prove that $last[j] = p_{wj}$. Now, $last[j] = \bigvee_{i=1}^{n} d_{ij} = \bigvee_{i=1}^{n} p_{ij} \wedge d_{i\overleftarrow{j}} = p_{wj} \wedge d_{w\overleftarrow{j}} \vee \left(\bigvee_{i\in[1,n]\setminus\{w\}} p_{ij} \wedge d_{i\overleftarrow{j}}\right)$. But, according to our assumption, $d_{w\overleftarrow{j}} = 1$. Thus, $last[j] = \bigvee_{i=1}^{n} d_{ij} = \bigvee_{i=1}^{n} p_{ij} \wedge d_{i\overleftarrow{j}} = p_{wj} \vee \left(\bigvee_{i\in[1,n]\setminus\{w\}} p_{ij} \wedge d_{i\overleftarrow{j}}\right)$. If $p_{wj} = 1$, $last[j] = p_{wj}$ trivially holds. We shall have to prove that $(p_{wj} = 0) \implies \left(\bigvee_{i\in[1,n]\setminus\{w\}} p_{ij} \wedge d_{i\overleftarrow{j}}\right) = 0$. If the above statement does not hold then there will be at least one bidder $V_\eta, \eta \in [1, n] \setminus \{w\}$, such that $p_{\eta j} = d_{\eta\overleftarrow{j}} = p_{\eta\overleftarrow{j}} = 1$. This essentially means that $p_{\eta t} = last[t] = p_{wt}$ for all $t \in [1, j-1]$ and $p_{\eta j} = 1, p_{wj} = 0$. This means that the bid price p_η of V_η is higher than that of V_w. This is a contradiction as according to our assumption V_w is the highest bidder and hence $p_w \geq p_\eta$. So, we conclude that the scheme is correct and it yields the maximum bid price, that is $last[j] = p_{wj}, \forall j \in [1, c]$. □

B. Privacy of losing bids

In this section, we prove that our protocol satisfies the *exclusive privacy* requirement in Definition 2. More specifically, when the colluding set do not contain the winner, they will learn nothing more than their own inputs and the highest bid; in this case, the definitions of *inclusive privacy* and *exclusive privacy* are equivalent. When the colluding set contain the winner, they will learn no more than the highest bid of the non-colluding set under the Decision Diffie-Hellman (DDH) assumption [21]; they learn the highest bid of the non-colluding set only in the worse case that the winner is decided at the last bit iteration (Theorem 2).

Assumption 1 (DDH Assumption [21]): Given g, g^a, g^b and a challenge $\Omega \in \{g^{ab}, R\}$, where $R \xleftarrow{\$} G$, it is computationally hard to find whether $\Omega = g^{ab}$ or $\Omega = R$.

Lemma 1: Let G be a group in which the DDH assumption holds. Given g, g^a, g^b, g^c and a challenge $\Omega \in \{g^{ab}, g^{bc}\}$, it is computationally hard to find whether $\Omega = g^{ab}$ or $\Omega = g^{bc}$.

Proof: Let R be a random element in G. Based on the DDH assumption, $(g, g^a, g^b, g^c, g^{ab}) \overset{c}{\approx} (g, g^a, g^b, g^c, R)$. Similarly, $(g, g^a, g^b, g^c, g^{ac}) \overset{c}{\approx} (g, g^a, g^b, g^c, R)$. Since computational indistinguishability is an equivalence relation, the same is also a transitive relation. Thus, we have $(g, g^a, g^b, g^c, g^{ab}) \overset{c}{\approx} (g, g^a, g^b, g^c, g^{ac})$. □

Lemma 2: Based on the DDH assumption, given $g, S_x = \{g^{x_i} : i \in [1, n]\}, S_y = \{Y_i : i \in [1, n]\}, Y_i = g^{y_i} = \prod_{j=1}^{i-1} g^{x_j} / \prod_{j=i+1}^{n} g^{x_j}, i \in [1, n], w, t \in [1, n], w \neq t, R = \{R_i = g^{r_i} : i \in [1, n] \setminus \{w, t\}\}$, g^{r_w}, g^{r_t}, $\phi \subseteq \{x_i : i \in [1, n] \setminus \{w, t\}\}$ and a challenge $\Omega \in \{A, B\}$, it is computationally hard to find if $\Omega = A$ or $\Omega = B$, where:

$$\begin{aligned} A =&(g, \phi, g^{x_1 z_1}, g^{x_2 z_2}, \ldots, g^{x_{w-1} z_{w-1}}, g^{x_w r_w}, g^{x_{w+1} z_{w+1}}, \ldots, \\ &g^{x_t y_t}, \ldots, g^{x_n z_n}) \\ B =&(g, \phi, g^{x_1 z_1}, g^{x_2 z_2}, \ldots, g^{x_{w-1} z_{w-1}}, g^{x_w r_w}, g^{x_{w+1} z_{w+1}}, \ldots, \\ &g^{x_t r_t}, \ldots, g^{x_n z_n}), \end{aligned}$$

where z_i is either y_i or r_i for $i \in [1, n] \setminus \{w, t\}$, and ϕ is chosen by the attacker.

Proof: We show that if there exists an adversary $\mathcal{A}'$, against the statement of Lemma 2, she can be used to construct another adversary $\mathcal{A}$ against Assumption 1. $\mathcal{A}$ works as follows:

$\mathcal{A}$ receives as input g, g^a, g^b, g^c and a challenge $\Omega \in \{g^{ab}, g^{ac}\}$. Then she lets $\mathcal{A}'$ select random $x_1, x_2, \ldots, x_{w-1}, x_{w+1}, \ldots, x_{t-1}, x_{t+1}, \ldots, x_n, r \in_R \mathbb{Z}_p$. Then she sets $X_i = g^{x_i}$ for $i \in [1, n] \setminus \{w, t\}$ and $X_w = g^{x_w} = g^b, X_t = g^{x_t} = g^a$, that is she implicitly sets $x_w = b$ and $x_t = a$. She also computes

$$Y_i = \prod_{j=1}^{i-1} X_j / \prod_{j=i+1}^{n} X_j, \forall i \in [1, n] \setminus \{w, t\}$$

She sets $\omega_i \in_R \{Y_i^{x_i}, R_i^{x_i}\}$, $\forall i \in [1,n] \setminus \{w,t\}$, and

$$\omega_w =(X_w)^{r_w},$$
$$\omega_t = \prod_{i=1}^{w-1}(X_t)^{x_i} \prod_{i=w+1}^{t-1}(X_t)^{x_i} / \prod_{i=t+1}^{n}(X_t)^{x_i}\Omega.$$

If $\Omega = g^{ab}$, then

$$\begin{aligned}\omega_t &= \prod_{i=1}^{w-1}(X_t)^{x_i} * \prod_{i=w+1}^{t-1}(X_t)^{x_i} / \prod_{i=t+1}^{n}(X_t)^{x_i} * g^{ab}\\ &= \prod_{i=1}^{t-1}(X_t)^{x_i} / \prod_{i=t+1}^{n}(X_t)^{x_i}.\end{aligned}$$

Let, $\omega = (g, \phi, \omega_1, \omega_2, \ldots, \omega_n)$. Hence, if $\Omega = g^{ab}$,

$$\begin{aligned}\omega =&(g, \phi, g^{x_1 z_1}, g^{x_2 z_2}, \ldots, g^{x_{w-1} z_{w-1}}, g^{x_w r_w}, g^{x_{w+1} z_{w+1}}, \ldots,\\ &g^{x_t y_t}, \ldots, g^{x_n z_n})\\ =&A,\end{aligned}$$

where $x_w = b$ and $x_t = a$. Alternatively, if $\Omega = g^{ac}$, then

$$\begin{aligned}\omega_t &= \prod_{i=1}^{w-1}(X_t)^{x_i} * \prod_{i=w+1}^{t-1}(X_t)^{x_i} / \prod_{i=t+1}^{n}(X_t)^{x_i} * g^{ac}\\ &=(g^{x_t})^{c+\sum_{i=1}^{w-1} x_i + \sum_{i=w+1}^{t-1} x_i - \sum_{i=t+1}^{n} x_i}\\ &=g^{x_t r_t},\end{aligned}$$

where $r_t = c + \sum_{i=1}^{w-1} x_i + \sum_{i=w+1}^{t-1} x_i - \sum_{i=t+1}^{n} x_i$ or $g^{r_t} = g^c * g^{\sum_{i=1}^{w-1} x_i + \sum_{i=w+1}^{t-1} x_i - \sum_{i=t+1}^{n} x_i}$. Thus, if $\Omega = g^{ac}$,

$$\begin{aligned}\omega =&(g, \phi, g^{x_1 z_1}, g^{x_2 z_2}, \ldots, g^{x_{w-1} z_{w-1}}, g^{x_w r_w}, g^{x_{w+1} z_{w+1}}, \ldots,\\ &g^{x_t r_t}, \ldots, g^{x_n z_n})\\ =&B.\end{aligned}$$

Now, $\mathcal{A}$ can send $S_x = \{X_i : i \in [1,n]\}$, g^{r_w}, g^{r_t} and ω to $\mathcal{A}'$. $\mathcal{A}'$ will identify ω as either A or B. If $\omega = A$, then $\Omega = g^{ab}$. Else if $\omega = B$, then $\Omega = g^{ac}$. Thus, with the help of $\mathcal{A}'$, we can construct an adversary $\mathcal{A}$ that can break Assumption 1. Hence, the lemma holds.

□

Lemma 3: Let C be a set of colluding bidders and H be the set of honest bidders. $C \cup H = [1,n]$. Let $\theta = |H|$. Let us assume $d_{h_i j}$ is the bit corresponding to the cryptogram submitted in iteration j by V_{h_i} for $h_i \in H, i \in [1,\theta]$ and $d_{c_i j}$ is the bit corresponding to the cryptogram submitted in iteration j by V_{c_i} for $c_i \in C, i \in [1, n-\theta]$. The colluding bidders learn nothing more than $\bigvee_{i=1}^{\theta} d_{h_i j}$.

Proof: In an iteration $j \in [1,c]$, if $K_j = \bigvee_{i=1}^{\theta} d_{h_i j} = 0$, the colluding bidders will obviously learn that $d_{h_i j} = 0$ for all $h_i \in H, i \in [1,\theta]$. We shall have to prove that when $K_j = \bigvee_{i=1}^{\theta} d_{h_i j} = 1$, the colluding bidders will not learn any other information. In order for proving this fact it is sufficient to show that

1) the colluding bidders will not be able to distinguish between the two cases where $K_j = \bigvee_{i=1}^{\theta} d_{h_i j} = 1$, but the number of bidders who submitted 1-cryptogram is different.
2) if two honest bidders who submitted different bits exchange their inputs, then this cannot be detected by the adversary.

We choose two scenarios in which a particular honest bidder submits different cryptograms: i.e in one scenario the bidder submits a 0-cryptogram and in the other one she submits a 1-cryptogram. We show that if the value of K_j is 1 in both the scenarios, then the two scenarios will be indistinguishable to the adversary (colluding bidders). Once we prove these results, they could be easily extended to show that the statement of the above lemma holds. Let us assume that the public keys used by the colluding bidders are $(X_{c_i}, R_{c_i}) = (g^{x_{c_i}}, g^{r_{c_i}}), c_i \in C, i \in [1,|C|]$. Similarly, the public keys of the honest bidders will be $(X_{h_i}, R_{h_i}), h_i \in H, i \in [1,\theta]$. The cryptograms of the colluding bidders will be $g^{x_{c_i} z_{c_i}}$, where $z_{c_i} \in \{y_{c_i}, r_{c_i}\}$. Let, $\phi = \{x_{c_i} : i \in [1,|C|]\}$. Now, let us assume that one honest bidder V_{h_w} has submitted a 1-cryptogram in the form: $g^{x_{h_w} r_{h_w}}$. As such we need to show that the colluding bidders will not be able to find whether or not there is another bidder $V_{h_t}, h_t \in H, t \in [1,\theta]$ who also submitted a 1-cryptogram. If V_{h_t} submitted a 1-cryptogram, then her cryptogram will be this: $b_{h_t} = g^{x_{h_w} r_{h_w}}$, and if she submitted a 0 cryptogram, her cryptogram should look like this: $b'_{h_t} = g^{x_{h_t} y_{h_t}}$, where the notations bear usual meanings as they do in the paper. Now, according to Lemma 2, no adversary can distinguish between A and B, where

$$\begin{aligned}A =&(\phi, b_1, b_2, \ldots, b_{h_w}, \ldots, b_{h_t}, \ldots, b_n)\\ B =&(\phi, b_1, b_2, \ldots, b_{h_w}, \ldots, b'_{h_t}, \ldots, b_n).\end{aligned}$$

Hence, the colluding bidders will not be able to distinguish between two cases where the value of $K_j = \bigvee_{i=1}^{\theta} d_{h_i j}$ is 1, but the number of bidders $V_{h_i}, h_i \in H$ who submitted 1-cryptogram in iteration j is different.

Let us assume $b_{h_e} = g^{x_{h_e} r_{h_e}}$ and $b'_{h_e} = g^{x_{h_e} y_{h_e}}$ for $e \in \{w,t\}$. Now, observe that

$$\begin{aligned}B =&(\phi, b_1, b_2, \ldots, b_{h_w}, \ldots, b'_{h_t}, \ldots, b_n)\\ \overset{c}{\approx}&(\phi, b_1, b_2, \ldots, b_{h_w}, \ldots, b_{h_t}, \ldots, b_n)\\ \overset{c}{\approx}&(\phi, b_1, b_2, \ldots, b'_{h_w}, \ldots, b_{h_t}, \ldots, b_n).\end{aligned}$$

Hence, the colluding adversary will not be able to distinguish between two cases where a pair of honest bidders exchange the value of their submitted bits whose values are complement to each other. This way anyone can prove that as long as at least one bidder submits a 1-cryptogram in any iteration, the colluding bidders will not be able to distinguish between the set of all possible cryptograms corresponding to all possible values of the bits submitted by honest bidders. What the colluding bidders learn is the logical-OR of all bits submitted by all honest bidders which is given by $K_j = \bigvee_{i=1}^{\theta} d_{h_i j}$, $j \in [1,c]$.

□

Theorem 2: The proposed e-auction scheme satisfies the *exclusive-privacy* in Definition 2.

Proof: Based on Lemma 3, at each bit iteration the colluding set learn nothing more than $K_j = \bigvee_{i=1}^{\theta} d_{h_i j}$, $j \in [1,c]$. Here $d_{h_i j}$ is the bit submitted by V_{h_i} at the jth iteration; it is equal to the actual bid value $b_{h_i j}$ only if the bidder V_{h_i} remains in the race (she will have to submit 0 if she has lost in the race as enforced by the ZKP in Eq. 2). Assume the winner is decided at the βth iteration, $1 \leq \beta \leq c$. If the colluding set do not contain the winner, the bit value K_j that they learn is the same as the jth most significant bit in the highest bid. In other words, they learn nothing more than the highest bid of all bidders. However, if the colluding set contain the winner, they can learn $K_1||K_2||\ldots||K_\beta$, which are the β most significant bits of the maximum bid of the non-colluding set H. The colluding set will learn the maximum bid of the non-colluding set in the worse case when $\beta = c$ (namely, the winner is only decided in the last bit iteration). Hence, the result. □

V. Efficiency of the Scheme

First, we discuss the computation overhead of our protocol. Since exponentiation is the costliest operation in our scheme, we measure the computation cost in terms of the number of exponentiations performed by a single entity. During the setup phase, each bidder generates commitments to each of the c bits of its bid-price. Each commitment requires 3 exponentiations. Hence, the total number of exponentiations required is $3c$. In order for generating NIZK proof of well-formedness of the commitments, the bidder needs to do $8c$ exponentiations. In every iteration of Stage 1, a bidder generates a pair of keys, publishes the public key and a cryptogram which encrypts a single bit. Computation of the public key requires 2 exponentiations. Also, the computation of the cryptogram requires 1 exponentiation. The zero knowledge proofs in Stage 1 require 14 exponentiations. Thus for one each iteration of Stage 1 a bidder needs to do 17 exponentiations. If there are τ iterations of Stage 1, then each bidder will need to perform 17τ exponentiations. Similar to Stage 1, in Stage 2, each bidder will need to perform 3 exponentiations in order to generate the keys and the cryptogram in an iteration. Again, computation of the NIZK proofs requires at most 30 exponentiations in every iteration of Stage 2. Hence, each bidder needs to perform 33 exponentiations per iteration of Stage 2. Since, the sum of all iterations of Stage 1 and Stage 2 is c, the total number of iterations of Stage 2 is $c - \tau$. Thus, each bidders performs $33(c - \tau)$ exponentiations in all the $c - \tau$ iterations of Stage 2. Therefore, the total number of exponentiations done by a single bidder during all the 3 phases (Setup, Stage 1 and Stage 2) is $44c - 16\tau$. The verifier needs to perform $48c - 16\tau$ exponentiations in order to check all the NIZK proofs generated by a single bidder during the auction process. Table I shows a break-down of the computation overhead on a bidder and a verifier.

During the setup phase a bidder generates c commitments corresponding to c bits of the bid-price. Each commitment is a 3-tuple. The bidder also generates NIZK proof of well-formedness of the c commitments. Each of such proof consists of 14 elements. Thus during the setup phase each bidder generates information of size $17c$. During one iteration of Stage 1, a bidder publishes a key of size 2, a cryptogram of size 1 and three NIZK proofs of total size 20. Hence, for τ iterations of Stage 1, the total bandwidth consumed is 23τ. During one iteration of Stage 2, a bidder generates a key of size 2, a cryptogram of size 1, three NIZK proofs of total size equal to 33. So, for $c-\tau$ iterations, the total space complexity turns out to be $36(c-\tau)$. In aggregate, each bidder generates a total of $53c-13\tau$ units of data. The verifier needs to download all the data generated by n distinct bidders for examining the authenticity of the NIZK proofs. Hence, the communication overhead on the verifier is n times that of a common bidder. Table II shows a break-down of the communication overhead on a bidder and a verifier.

Based on Table I and II, we can conclude that with respect to the bit length c of the bid price, our protocol incurs a linear complexity $O(c)$ for both the computational load and the bandwidth usage. The computation per bidder remains roughly unchanged with the number of participants n, although the verification cost increases linearly with more participants. Assuming that every bidder is also a verifier, responsible for checking every other bidder's ZKPs, our protocol incurs a linear computational and communication complexity $O(n)$, with respect to the number of participants n. These are the best possible system complexities that one may hope for. In practice, the verification of the ZKPs may be centrally performed by the public bulletin board before the data is published and every observer is able to check at any time. This can alleviate each bidder's task in verifying the ZKPs.

We have implemented the SEAL auction scheme using Java on a Linux platform. The experiment was done on an Asus A Series Core i3 laptop (2.10 GHz with 4 GB RAM). We have plotted four graphs on the basis of the data obtained from this experiment. Figure 2a depicts the average time needed to generate the parameters in one iteration of Stage 1 and Stage 2 for different values of c, i.e., the bit-length of the maximum bid-price. Here we have fixed the number of bidders n at 10. Figure 2b shows the average time needed to generate the parameters in one iteration of Stage 1 and Stage 2 for different values of n, the number of bidders. Here, we have fixed the value of c at 10. Figure 2c depicts the average time needed to generate the c commitments by one bidder for different values of c with $n = 10$. Figure 2d depicts the average time needed to generate the commitments by one bidder for different values of n with $c = 10$.

It can be observed from Figure 2a and 2b that the time required to finish one iteration in Stage 1 and Stage 2 remains almost the same irrespective of the number of bidders or the value of c. In other words, if there are a iterations of Stage 1 and $c-a$ iterations of Stage 2, then the total time to complete all the c iterations will be $a \cdot t_1 + (c - a) \cdot t_2$, where t_1 and t_2 are the time required to complete exactly one iteration at Stage 1 and 2 respectively. So, the time to execute all the c iterations will be upper-bounded by the the time to complete c iterations of Stage 2. Figure 2c shows that the time taken to generate all the commitments for all the c bits of one bid-price also increases linearly with the increase of the value of c when the number of bidders is constant. Additionally, Figure 2d shows that the time to compute the commitments for all the c bits of the bid-price of one bidder does not depend on the number of bidders. We therefore conclude that the overall time required to finish the auction protocol per bidder (including the time to generate commitments and the time to execute c

	Setup		Stage 1			Stage 2			
Entity	**Commitment**	**ZKP**	**Key**	**Cryptogram**	**ZKP**	**Key**	**Cryptogram**	**ZKP**	**Total**
Bidder	$3c$	$8c$	2τ	τ	14τ	$2(c-\tau)$	$(c-\tau)$	$30(c-\tau)$	$44c-16\tau$
Verifier	-	$12nc$	-	-	$20n\tau$	-	-	$36n(c-\tau)$	$48nc-16n\tau$

TABLE I: The number of exponentiations in computational load. Here, c is the length of the binary representation of a bid-price. n is the total number of bidders and τ is the number of iterations of Stage 1.

	Setup		Stage 1			Stage 2			
Entity	**Commitment**	**ZKP**	**Key**	**Cryptogram**	**ZKP**	**Key**	**Cryptogram**	**ZKP**	**Total**
Bidder	$3c$	$14c$	2τ	τ	20τ	$2(c-\tau)$	$(c-\tau)$	$33(c-\tau)$	$53c-13\tau$
Verifier	$3nc$	$14nc$	$2n\tau$	$n\tau$	$20n\tau$	$2n(c-\tau)$	$n(c-\tau)$	$33n(c-\tau)$	$53nc-13n\tau$

TABLE II: Communication bandwidth (in total number of G and Z_p elements). Here, c is the length of the binary representation of a bid-price. n is the total number of bidders and τ is the number of iterations of Stage 1.

iterations of the main protocol) is linear in the bit length c of the highest bid-price irrespective of the number of bidders. This corroborates the theoretical analysis in Table I. Obviously, the communication bandwidth per bidder scales linearly with the bit length of the bid price as well given the constant size of the message sent at each bit iteration for both Stage 1 and 2 (see Table II).

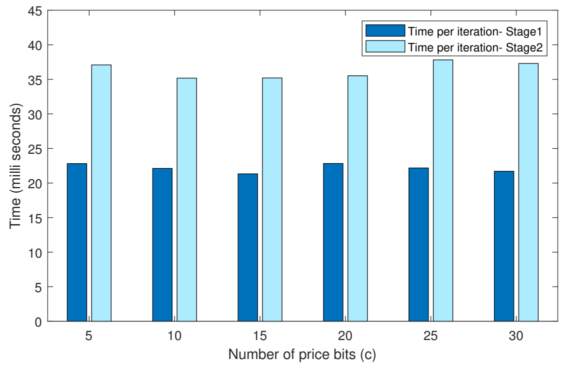
VI. Related Work

Sealed-bid e-auction can be regarded as an instance of secure multiparty computation (MPC) on a *max* function, where participants jointly compute the maximum of a set of inputs while preserving the privacy of other input values. Generic MPC techniques generally require pairwise secret channels between the participants, in addition to an authenticated public channel [12]. However, pairwise secret channels are difficult to realize among bidders who are mutually distrustful. In addition, generic MPC techniques suffer from various efficiency issues [14]. For these reasons, although inspiring in theory, they are not directly applicable to build a decentralized sealed-bid e-auction scheme. In the following, we will review main e-auction schemes in the literature.

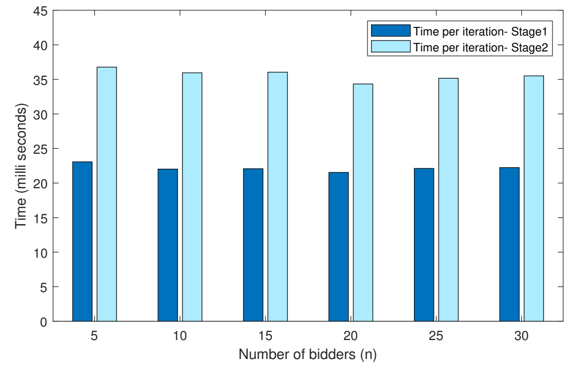
Since the early work by Franklin and Reiter in 1996 [3], researchers have proposed many sealed-bid e-auction schemes [2]–[4], [6]–[10], [16], [17], [25], [26]. Most of these schemes involve the role of an "auctioneer", which mirrors a similar role in traditional auctions. Hence, the mainstream research in this field focuses on applying cryptography to distribute trust on the "auctioneer". In general, there are two main approaches.

The first approach is to apply threshold cryptography, or MPC techniques [10] to distribute the trust from a single auctioneer to several auctioneers. Franklin et al. presented a second-price sealed-bid auction scheme in [3]. In this scheme, a number of servers play the role of the auctioneer, and they apply Shamir's secret sharing to split each bid among themselves so no single server sees all bids. However, if a sufficient number of servers collude, the secrecy of all bids is lost. Sako proposed a similar scheme based on threshold cryptography to let auctioneers jointly decrypt submitted bids [4]. Kurosawa and Ogata proposed a bit-slice approach for auctioneers to determine the highest bid, bit by bit, assuming the majority of the auctioneers are honest [17]. Their system involves m bidders and n auctioneers. The auctioneers apply secure multi-party computation on a bit-slice circuit, and decrypt the result at each bit position using verifiable threshold decryption. The threshold is set such that compromising the decryption requires compromising at least the majority of the auctioneers. The number of rounds required for threshold decryption is $O(n \cdot c)$, c being the bit length of the bid and n being the number of participants in the decryption process. The Kurosawa-Ogata's method, like many other works [3], [9]–[11], performs encryption and decryption of bids as two separate phases. By comparison, in our protocol, the encryption and decryption operations are more integrated within the constant 2-round Boolean-OR computation (veto protocol) at each bit iteration, which results in $O(c)$ rounds in total regardless of the number of participants. Furthermore, our protocol is free from any auctioneers. Bogetoft et al. proposed to apply secure MPC to sealed-bid auction [9], [10]. Their solution was used in Denmark for auction sales on sugar beets. In Bogetoft et al.'s solution, the role of the auctioneer is played by three parties. It is assumed that at least 2 parties must be honest. Cartlidge et al. proposed an e-auction scheme for dark pools/markets where all bids are encrypted under a global public key and the decryption is performed by auctioneers using MPC [11]. They presented two implementations based on the SCALE-MAMBA library: the first uses the SPDZ protocol [11] to implement the role of "auctioneer" as two servers assuming at least one is trustworthy; the second uses Shamir's secret sharing to implement the auctioneers as three servers assuming at least one of them is trustworthy. In all these auctioneer-based protocols, if auctioneers collude all together, they can trivially break the privacy of sealed bids.

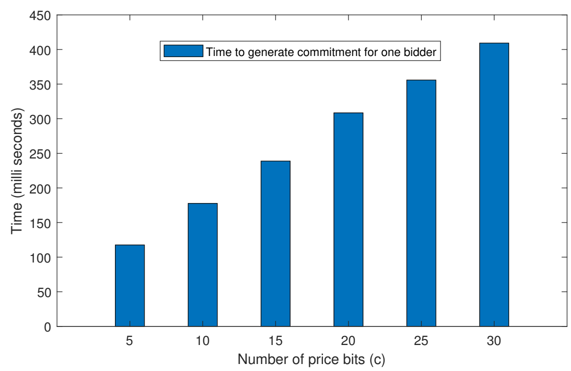
The second approach is to introduce more trusted third parties in addition to auctioneers. Naor et al. presented a second-price sealed auction scheme in [5]. This scheme uses two different auction servers who communicate using an oblivious transfer protocol. One server takes the role as an auctioneer and the other as an auction issuer. The two servers are assumed not to collude. However, the original Naor et al.'s scheme has a weakness in which one of the two servers can cheat to modify bids without detection. This weakness was later addressed by Juels and Szydlo in [8], but the revised scheme



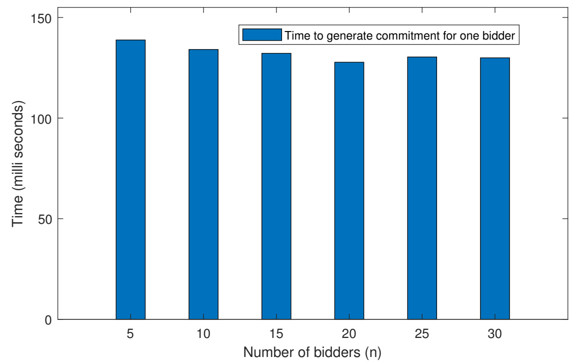
(a) Computation time needed by the bidder application per iteration for 10 bidders.



(b) Average computation time needed by the bidder application per iteration for $c = 10$.



(c) Computation time needed by the bidder application in the Commit phase for 10 bidders



(d) Computation time needed by the bidder application in the Commit phase for a fixed $c = 10$

Fig. 2: Time required by a bidder at different stages.

still requires the two servers must not collude. In [26], Abe and Suzuki proposed an (M + 1)-st price sealed bid auction using homomorphic encryption and the mix and match technique. The scheme involves an auctioneer and a trusted authority, who are assumed not to collude. In [27], Montengero et al. propose a sealed-bid online auction scheme that employs an auctioneer and a randomness server. The randomness server is trusted to provide the randomness for the bidders in the protocol and not to collude with the auctioneer. In [25], Lipmaa et al. proposed Vickrey auction schemes that involve a seller and an auction authority. The seller and the authority authority are assumed not to collude. In [28], Galal and Youssef proposed a Vickrey auction system by using the Intel SGX enclave as a trusted hardware device to compute the winner. Here, the SGX enclave essentially plays the role as an auctioneer. Similar solutions based on trusted computing are presented in [29].

Brandt was among the first to argue that neither of the above approaches is desirable due to the involvement of trusted auctioneers or third parties. He proposed the notion of "bidder-resolved auction" and a concrete auctioneer-free solution in [7] by applying secret sharing techniques. Follow-up works by Brandt, with improvement in efficiency, are published in [2], [13], [14]. In all these "bidder-resolved auction" schemes, a seller is actively involved in the protocol and is assumed not to collude with bidders. However, as pointed out by Dreier et al. [30], if the seller colludes with a subset of bidders in Brandt's bidder-resolved auctions schemes, they can learn the bids of other participants. Another major limitation in Brandt's schemes is that they incur exponential computational and communication complexities $O(2^c)$, where c is the bit length of the bid price. Motivated by Brandt's initial 2002 scheme [7], Wu et al. remove the seller and propose a decentralized sealed-bid auction scheme based on a general socialist millionaire protocol. However, a critical downside of their system is that the computational load and the bandwidth usage per bidder is exponential $O(2^c)$ with respect to the bit length of the bid price, which makes their scheme less than practical.

VII. Conclusion

In this paper, we propose a publicly verifiable sealed bid auction scheme that does not require any auctioneer and has a linear system complexity in terms of computation and communication with respect to the bit length of the bid. The bidders execute the protocol themselves and compute the highest bid while preserving the privacy of losing bids. Furthermore, our protocol does not require any secret channels and all operations are publicly verifiable. Overall, our work removes the dependence on any auctioneer and drastically reduces the system complexity associated with existing state-of-the-art auction schemes. This brings secure e-auction much closer to practice.

References

[1] V. Krishna, *Auction theory*. Academic press, 2009.

[2] F. Brandt, "How to obtain full privacy in auctions," *International Journal of Information Security*, vol. 5, no. 4, pp. 201–216, 2006.

[3] M. K. Franklin and M. K. Reiter, "The design and implementation of a secure auction service," *IEEE Transactions on Software Engineering*, vol. 22, no. 5, pp. 302–312, 1996.

[4] K. Sako, "An auction protocol which hides bids of losers," in *International Workshop on Public Key Cryptography*. Springer, 2000, pp. 422–432.

[5] M. Naor, B. Pinkas, and R. Sumner, "Privacy preserving auctions and mechanism design," *EC*, vol. 99, pp. 129–139, 1999.

[6] W. Qianhong, W. Changjie, C. Xiaofeng, and W. Yumin, "Publicly verifiable auctions with minimal leakage," in *Computer Software and Applications Conference, 2004. COMPSAC 2004. Proceedings of the 28th Annual International*. IEEE, 2004, pp. 384–389.

[7] F. Brandt, "Secure and private auctions without auctioneers," *Technical Report FKI-245–02. Institut fur Informatick, Technishce Universitat Munchen*, 2002.

[8] A. Juels and M. Szydlo, "A two-server, sealed-bid auction protocol," in *International Conference on Financial Cryptography*. Springer, 2002, pp. 72–86.

[9] P. Bogetoft, I. Damgård, T. P. Jakobsen, K. Nielsen, J. Pagter, and T. Toft, "A practical implementation of secure auctions based on multiparty integer computation," in *Financial Cryptography*, vol. 4107. Springer, 2006, pp. 142–147.

[10] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. P. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter *et al.*, "Secure multiparty computation goes live." in *Financial Cryptography*, vol. 5628. Springer, 2009, pp. 325–343.

[11] J. Cartlidge, N. P. Smart, and Y. Talibi Alaoui, "Mpc joins the dark side," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*. ACM, 2019, pp. 148–159.

[12] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. ACM, 1987, pp. 218–229.

[13] F. Brandt, "Fully private auctions in a constant number of rounds," in *Computer Aided Verification*. Springer, 2003, pp. 223–238.

[14] F. Brandt and T. Sandholm, "Efficient privacy-preserving protocols for multi-unit auctions," *Lecture notes in computer science*, vol. 3570, p. 298, 2005.

[15] R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin, "Efficient multiparty computations secure against an adaptive adversary," in *Eurocrypt*, vol. 99. Springer, 1999, pp. 311–326.

[16] M. Jakobsson and A. Juels, "Mix and match: Secure function evaluation via ciphertexts," *ASIACRYPT 2000*, pp. 162–177, 2000.

[17] K. Kurosawa and W. Ogata, "Bit-slice auction circuit," *ESORICS 2002*, pp. 24–38, 2002.

[18] F. Hao and P. Zieliński, "A 2-round anonymous veto protocol," in *International Workshop on Security Protocols*. Springer, 2006, pp. 202–211.

[19] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on computing*, vol. 18, no. 1, pp. 186–208, 1989.

[20] M. Blum, P. Feldman, and S. Micali, "Non-interactive zero-knowledge and its applications," in *Proceedings of the twentieth annual ACM symposium on Theory of computing*. ACM, 1988, pp. 103–112.

[21] D. Boneh, "The decision diffie-hellman problem," in *International Algorithmic Number Theory Symposium*. Springer, 1998, pp. 48–63.

[22] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *Annual International Cryptology Conference*. Springer, 1994, pp. 174–187.

[23] J. Camenisch and M. Stadler, "Proof systems for general statements about discrete logarithms," ETH Zurich, Technical Report No. 260, Tech. Rep., 1997.

[24] S. A. Brands, *Rethinking public key infrastructures and digital certificates: building in privacy*. Mit Press, 2000.

[25] H. Lipmaa, N. Asokan, and V. Niemi, "Secure vickrey auctions without threshold trust," in *International Conference on Financial Cryptography*. Springer, 2002, pp. 87–101.

[26] M. Abe and K. Suzuki, "M+ 1-st price auction using homomorphic encryption," in *Public Key Cryptography*, vol. 2274. Springer, 2002, pp. 115–124.

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

[28] H. S. Galal and A. M. Youssef, "Verifiable sealed-bid auction on the ethereum blockchain," in *2018 Financial Cryptography and Data Security Workshops on Trusted Smart Contracts*, 2018, pp. 265–278.

[29] R. Ankele, K. A. Küçük, A. Martin, A. Simpson, and A. Paverd, "Applying the trustworthy remote entity to privacy-preserving multiparty computation: Requirements and criteria for large-scale applications," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE, 2016, pp. 414–422.

[30] J. Dreier, J.-G. Dumas, and P. Lafourcade, "Brandt's fully private auction protocol revisited." *AFRICACRYPT*, vol. 13, pp. 88–106, 2013.

[31] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 1986, pp. 186–194.

[32] S. F. Shahandashti and F. Hao, "DRE-ip: A verifiable e-voting scheme without tallying authorities," in *ESORICS (2)*, ser. LNCS, vol. 9879. Springer, 2016, pp. 223–240, full version: IACR ePrint Report No. 2016/670.

Appendix

NIZK proof systems used in this paper

In this section we show how the various non-interactive zero knowledge proofs [19], [20], [24] essential for the implementation of our proposed auction scheme can be constructed. Following [18], we propose to include the unique user identity (i.e., the index i of each participant) into the hash function when generating the challenge using the Fiat-Shamir transformation [31]. The inclusion of the user identity is to prevent the replay of a ZKP by a different party (say back to the sender).

We have provided construction of four different NIZK proofs. The first proof is for showing the well-formedness of each commitment posted by the bidders during the setup phase. This proof allows a bidder to show that the commitments provided by her indeed correspond to bits, rather than anything else. The second proof shows that the bidder knows the part of the secret key that will allow her to compute the cryptogram. This NIZK proof is provided by the bidder during the key generation sub-phase of Stage 1 and Stage 2. The last two NIZK proofs are generated by bidders during each iteration of Stage 1 and Stage 2 and they show the well-formedness of the cryptograms issued by the bidders. Examples of first two types of NIZK proofs can be found in [22], [24], [32]. We provide the construction of NIZK proofs for cryptogram well-formedness in the following sections. Construction of this type of NIZK proofs can also be found in [23].

Well-formedness of Commitments: Each commitment of our scheme is of the form $\varepsilon = \langle g^{\alpha\beta} g^{v}, g^{\alpha}, g^{\beta}\rangle$, where v is the committed bit. The bidder has to provide c commitments, each for exactly one of the c bits in the binary representation of the bid-price of that bidder. The construction of the NIZK proof of well-formedness of the commitment is as follows:
First, given g^{α} and g^{β}, the bidder needs to prove knowledge of $\alpha = \log_g(g^{\alpha})$ and $\beta = \log_g(g^{\beta})$ using Schnorr's signature [18] (See next subsection on well-formedness of public keys). Then the statement that the prover (bidder) needs to show is that ε is well formed for $v \in \{0, 1\}$, that is:

$$\begin{aligned}\sigma \;\equiv\; & (\ \phi = g^{\alpha\beta} \ \wedge\ A = g^{\alpha} \ \wedge\ B \equiv g^{\beta}\ ) \\ & \vee\ (\ \phi = g^{\alpha\beta} g \ \wedge\ A = g^{\alpha} \ \wedge\ B = g^{\beta}\ ).\end{aligned}$$

Note that only one of the statement can be true. Let us assume that the first statement is correct, that is $(\phi = g^{\alpha\beta} \wedge A = g^{\alpha} \wedge B = g^{\beta})$. So, the prover needs to provide a real proof for this statement and a simulated proof for the other statement $(\phi = g^{\alpha\beta}g \wedge A = g^{\alpha} \wedge B = g^{\beta})$. The prover selects random $r_1 \in_R \mathbb{Z}_p$ and computes following commitments:

$$\varepsilon_{11} = g^{r_1},\ \varepsilon_{12} = (g^{\beta})^{r_1}.$$

The prover then chooses random $ch_2, \rho_2 \in_R \mathbb{Z}_p$ and computes commitments:

$$\varepsilon_{21} = g^{\rho_2}(g^{\alpha})^{ch_2},\ \varepsilon_{22} = (g^{\beta})^{\rho_2}(\phi/g)^{ch_2}.$$

Let, ch be the grand challenge of the NIZK protocol. Now, the bidder computes a response $\rho_1 = r_1 - \alpha \cdot ch_1$, where $ch_1 = ch - ch_2$. The verification equations are as below:

1) $g^{\rho_1} \stackrel{?}{=} \frac{\varepsilon_{11}}{(g^{\alpha})^{ch_1}}$
2) $(g^{\beta})^{\rho_1} \stackrel{?}{=} \frac{\varepsilon_{12}}{(\phi)^{ch_1}}$
3) $g^{\rho_2} \stackrel{?}{=} \frac{\varepsilon_{21}}{(g^{\alpha})^{ch_2}}$
4) $(g^{\beta})^{\rho_2} \stackrel{?}{=} \frac{\varepsilon_{22}}{(\phi/g)^{ch_2}}$

If all 4 relations hold then the proof is accepted. The proof consists of 4 commitments, 2 challenges and 2 responses, making the space complexity equal to 8. The prover needs to do 6 exponentiations for generating the proof. The verifier needs to do 8 exponentiations for verifying them. Again, in order for showing knowledge of α and β, the prover needs to do 2 exponentiation and the size of the two proofs will be 6 in total.

Well-formedness of public keys: In each iteration of Stage 1 as well as Stage 2, a bidder selects a random secret key of the form $(x, r) \in \mathbb{Z}_p^2$ and publishes the corresponding public key $(X, R) = (g^x, g^r)$. The public key comes along with an NIZK proof that proofs that given a public key (X, R), the prover knows the discrete logarithm of X with respect to g. We show how the prover (bidder) can provide NIZK proof showing the knowledge of the discrete logarithm of X. The prover generates random $\bar{r} \in \mathbb{Z}_p$ and computes a commitment $\varepsilon = g^{\bar{r}}$. Let ch be the random challenge of the NIZK proof obtained through feeding the commitment and all other available argument into a random oracle. The prover generates a response $\rho = \bar{r} - ch \cdot x$. The verification equation is: $g^{\rho} \stackrel{?}{=} \frac{\varepsilon}{X^{ch}}$. The prover needs to do just one exponentiation for generating the proof that contains one challenge, one commitment and one response, 3 parameters in total. The verifier needs to do 2 exponentiations to verify the proof. Similarly a NIZK proof can be constructed for proving the knowledge of $r = \log_g R$.

NIZK proof of Stage 1: Here, we discuss the construction of NIZK proofs of well-formedness of cryptograms mentioned in Stage 1 of the e-auction scheme.

In iteration j, each bidder V_i constructs a NIZK proof π_{ij} of well-formedness of b_{ij}. The proof π_{ij} proves the following statement:

$$\begin{aligned}\sigma \ \equiv \Big(& \ (b_{ij} = g^{x_{ij}y_{ij}} \wedge X_{ij} = g^{x_{ij}} \wedge Y_{ij} = g^{y_{ij}}) \\ & \wedge (c_{ij} = g^{\alpha_{ij}\beta_{ij}} \wedge A_{ij} = g^{\alpha_{ij}} \wedge B_{ij} = g^{\beta_{ij}}) \Big) \\ \vee \Big(& \ (b_{ij} = g^{x_{ij}r_{ij}} \wedge X_{ij} = g^{x_{ij}} \wedge R_{ij} = g^{r_{ij}}) \\ & \wedge (c_{ij} = g^{\alpha_{ij}\beta_{ij}}g \wedge A_{ij} = g^{\alpha_{ij}} \wedge B_{ij} = g^{\beta_{ij}}) \Big).\end{aligned}$$

For ease of writing, we denote b_{ij} as B, x_{ij} as x, y_{ij} as y, c_{ij} as C, α_{ij} as α, β_{ij} as β, X_{ij} as X, Y_{ij} as Y, r_{ij} as r, A_{ij} as $\bar{A}$, B_{ij} as $\bar{B}$ and R_{ij} as R. Hence, σ can be rewritten as

$$\begin{aligned}\sigma \ \equiv \Big(& \ (B = g^{xy}) \wedge (X = g^{x}) \wedge (Y = g^{y}) \\ & \wedge (c = g^{\alpha\beta}) \wedge (\bar{A} = g^{\alpha}) \wedge (\bar{B} = g^{\beta}) \Big) \\ \vee \Big(& \ (B = g^{xr}) \wedge (X = g^{x}) \wedge (R = g^{r}) \\ & \wedge (c = g^{\alpha\beta}g) \wedge (\bar{A} = g^{\alpha}) \wedge (\bar{B} = g^{\beta}) \Big).\end{aligned}$$

This is a one-out-of-two statement. Hence, only one of the two constituent sub-statements is true. That is, either $(B = g^{xy}) \wedge (X = g^{x}) \wedge (Y = g^{y}) \wedge (c = g^{\alpha\beta}) \wedge (\bar{A} = g^{\alpha}) \wedge (\bar{B} = g^{\beta}))$ is true or $((B = g^{xr}) \wedge (c = g^{\alpha\beta}g) \wedge (\bar{A} = g^{\alpha}) \wedge (\bar{B} = g^{\beta}))$ is true but not both. We show how a NIZK proof can be constructed when the first sub-statement is true, i.e. if $(B = g^{xy}) \wedge (X = g^{x}) \wedge (Y = g^{y}) \wedge (c = g^{\alpha\beta}) \wedge (\bar{A} = g^{\alpha}) \wedge (\bar{B} = g^{\beta}))$ is true. The bidder selects random $r_{11}, r_{12} \in \mathbb{Z}_p$ and computes these commitments:

$$\varepsilon_{11} = g^{r_{11}}, \varepsilon_{12} = g^{r_{12}}, \varepsilon_{13} = (Y)^{r_{11}}, \varepsilon_{14} = (\bar{B})^{r_{12}}$$

The bidder also selects random $\rho_{21}, \rho_{22} \in \mathbb{Z}_p$ and $ch_2 \in \mathbb{Z}_p$ and computes $\varepsilon_{21} = g^{\rho_{21}}(X)^{ch_2}$, $\varepsilon_{22} = g^{\rho_{22}}(\bar{A})^{ch_2}$, $\varepsilon_{23} = (R)^{\rho_{21}}(B)^{ch_2}$, $\varepsilon_{24} = (\bar{B})^{\rho_{22}}(C/g)^{ch_2}$. Let, the grand challenge of NIZK proof be ch. The bidder computes $ch_1 = ch - ch_2$ and two responses $\rho_{11} = r_{11} - x \cdot ch_1, \rho_{12} = r_{12} - \alpha \cdot ch_1$.
The verification equations are as below:

1) $g^{\rho_{11}} \stackrel{?}{=} \frac{\varepsilon_{11}}{(X)^{ch_1}}$
2) $g^{\rho_{12}} \stackrel{?}{=} \frac{\varepsilon_{12}}{(\bar{A})^{ch_1}}$
3) $(Y)^{\rho_{11}} \stackrel{?}{=} \frac{\varepsilon_{13}}{B^{ch_1}}$
4) $(\bar{B})^{\rho_{12}} \stackrel{?}{=} \frac{\varepsilon_{14}}{C^{ch_1}}$
5) $g^{\rho_{21}} \stackrel{?}{=} \frac{\varepsilon_{21}}{(X)^{ch_2}}$
6) $g^{\rho_{22}} \stackrel{?}{=} \frac{\varepsilon_{22}}{(\bar{A})^{ch_2}}$
7) $(R)^{\rho_{21}} \stackrel{?}{=} \frac{\varepsilon_{23}}{B^{ch_2}}$
8) $(\bar{B})^{\rho_{22}} \stackrel{?}{=} \frac{\varepsilon_{24}}{(c/g)^{ch_2}}$

If the above 8 relations hold, the NIZK proof is authentic. A bidder needs to do 12 exponentiations for computing the above NIZK proof arguments. The proof itself consists of 8 commitments, two challenges and 4 responses. Hence, the space complexity of the proof is 14. Moreover, a verifier needs to do 16 exponentiations for checking all the arguments of this NIZK proof. Similarly a NIZK proof can be constructed if the second statement

$((B = g^{xr}) \wedge (c = g^{\alpha\beta} g) \wedge (\bar{A} = g^{\alpha}) \wedge (\bar{B} = g^{\beta}))$ is true. Here, we skip the construction due to space constraint.

NIZK proof of Stage 2: Now, we discuss the construction of the NIZK proof π'_{ij} of well-formedness of the cryptogram b_{ij} of Stage 2.

The logical statement for which a NIZK proof is to be constructed is the following:

$$\begin{aligned}\sigma \equiv \Big(& \big((b_{ij} = g^{x_{ij} r_{ij}}) \wedge (X_{ij} = g^{x_{ij}}) \wedge (R_{ij} = g^{r_{ij}}) \big) \\ & \wedge \big((b_{ij'} = g^{x_{ij'} r_{ij'}}) \wedge (X_{ij'} = g^{x_{ij'}}) \wedge (R_{ij'} = g^{r_{ij'}}) \big) \\ & \wedge \big((c_{ij} = g^{\alpha_{ij}\beta_{ij}} g) \wedge (A_{ij} = g^{\alpha_{ij}}) \wedge (B_{ij} = g^{\beta_{ij}}) \big) \Big) \\ \vee \Big(& \big((b_{ij} = g^{x_{ij} y_{ij}}) \wedge (X_{ij} = g^{x_{ij}}) \wedge (Y_{ij} = g^{y_{ij}}) \big) \\ & \wedge \big((b_{ij'} = g^{x_{ij'} r_{ij'}}) \wedge (X_{ij'} = g^{x_{ij'}}) \wedge (R_{ij'} = g^{r_{ij'}}) \big) \\ & \wedge \big((c_{ij} = g^{\alpha_{ij}\beta_{ij}}) \wedge (A_{ij} = g^{\alpha_{ij}}) \wedge (B_{ij} = g^{\beta_{ij}}) \big) \Big) \\ \vee \Big(& \big((b_{ij} = g^{x_{ij} y_{ij}}) \wedge (X_{ij} = g^{x_{ij}}) \wedge (Y_{ij} = g^{y_{ij}}) \big) \\ & \wedge \big((b_{ij'} = g^{x_{ij'} y_{ij'}}) \wedge (X_{ij'} = g^{x_{ij'}}) \wedge (Y_{ij'} = g^{y_{ij'}}) \big) \Big)\end{aligned}$$

The above statement is a one-out-of-3 logical statement. If a NIZK proof for the above statement can be constructed each bidder will be able to show the well-formednss of her cryptogram without revealing whether the cryptogram is an encryption of 0 or 1. Again, the NIZK statement will not reveal the state of the W_i variable for a bidder V_i. Nonetheless, the NIZK proof will establish the fact that the cryptogram provided by V_i is the correct encryption of the logical-AND of p_{ij} and the value of W_i as of iteration j. We denote b_{ij} as B_i, $b_{ij'}$ as B_j and c_{ij} as C_i for ease of writing. We also write R_{ij} as R_i, $R_{ij'}$ as R_j, r_{ij} as r_i, $r_{ij'}$ as r_j, $X_{ij'}$ as X_j, X_{ij} as X_i, $Y_{ij'}$ as Y_j, Y_{ij} as Y_i, $x_{ij'}$ as x_j, x_{ij} as x_i, $y_{ij'}$ as y_j, y_{ij} as y_i, Y_{ij} as Y_i, $Y_{ij'}$ as Y_j, y_{ij} as y_i, $y_{ij'}$ as y_j, A_{ij} as A, B_{ij} as B, α_{ij} as α_i and β_{ij} as β_i. Hence, we can rewrite the above statement as following:

$\sigma \equiv (((B_i = g^{x_i r_i}) \wedge (X_i = g^{x_i}) \wedge (R_i = g^{r_i})) \wedge ((B_j = g^{x_j r_j}) \wedge (X_j = g^{x_j}) \wedge (R_j = g^{r_j})) \wedge ((C_i = g^{\alpha_i\beta_i} g) \wedge (A = g^{\alpha_i}) \wedge (B = g^{\beta_i})))$
$\vee (((B_i = g^{x_i y_i}) \wedge (X_i = g^{x_i}) \wedge (Y_i = g^{y_i})) \wedge ((B_j = g^{x_j r_j}) \wedge (X_j = g^{x_j}) \wedge (R_j = g^{r_j})) \wedge ((C_i = g^{\alpha_i\beta_i}) \wedge (A = g^{\alpha_i}) \wedge (B = g^{\beta_i})))$
$\vee (((B_i = g^{x_i y_i}) \wedge (X_i = g^{x_i}) \wedge (Y_i = g^{y_i})) \wedge ((B_j = g^{x_j y_j}) \wedge (X_j = g^{x_j}) \wedge (Y_j = g^{y_j})))$

The above statement is a 1-out-of-3 statement. So, exactly one of the three constituent sub-statements has to be true. We show how this could be done for each of the three cases below.

a) **Case 1::** : the first statement $(((B_i = g^{x_i r_i}) \wedge (X_i = g^{x_i}) \wedge (R_i = g^{r_i})) \wedge ((B_j = g^{x_j r_j}) \wedge (X_j = g^{x_j}) \wedge (R_j = g^{r_j})) \wedge ((C_i = g^{\alpha_i\beta_i} g) \wedge (A = g^{\alpha_i}) \wedge (B = g^{\beta_i})))$ is true.

Generate $r_{11}, r_{12}, r_{13} \in_R \mathbb{Z}_p$ and compute commitments

$$\varepsilon_{11} = g^{r_{11}}, \varepsilon_{12} = g^{r_{12}}, \varepsilon_{13} = g^{r_{13}}$$

and

$$\varepsilon'_{11} = (R_i)^{r_{11}}, \varepsilon'_{12} = (R_j)^{r_{12}}, \varepsilon'_{13} = (B)^{r_{13}}$$

Then select $ch_2, \rho_{21}, \rho_{22}, \rho_{23} \in_R \mathbb{Z}_p$ and compute

$$\varepsilon_{21} = g^{\rho_{21}}(X_i)^{ch_2}, \varepsilon_{22} = g^{\rho_{22}}(X_j)^{ch_2}, \varepsilon_{23} = g^{\rho_{23}}(A)^{ch_2}$$

and $\varepsilon'_{21} = (Y_i)^{\rho_{21}}(B_i)^{ch_2}, \varepsilon'_{22} = (R_j)^{\rho_{22}}(B_j)^{ch_2}, \varepsilon'_{23} = (B)^{\rho_{23}}(C_i)^{ch_2}$. Also select $ch_3, \rho_{31}, \rho_{32}, \rho_{33} \in_R \mathbb{Z}_p$ and compute

$$\varepsilon_{31} = g^{\rho_{31}}(X_i)^{ch_3}, \varepsilon_{32} = g^{\rho_{32}}(X_j)^{ch_3}$$

and

$$\varepsilon'_{31} = (Y_i)^{\rho_{31}}(B_i)^{ch_3}, \varepsilon'_{32} = (Y_j)^{\rho_{32}}(B_j)^{ch_3}$$

Let, the grand challenge be ch. Compute $ch_1 = ch - ch_2 - ch_3$. Now, compute three responses

$$\rho_{11} = r_{11} - x_i * ch_1, \rho_{12} = r_{12} - x_j \cdot ch_1, \rho_{13} = r_{13} - \alpha_i \cdot ch_1$$

Publish all the commitments, challenges and the responses.

The verification equations are as below:

1) $g^{\rho_{11}} \stackrel{?}{=} \frac{\varepsilon_{11}}{(X_i)^{ch_1}}$
2) $g^{\rho_{12}} \stackrel{?}{=} \frac{\varepsilon_{12}}{(X_j)^{ch_1}}$
3) $g^{\rho_{13}} \stackrel{?}{=} \frac{\varepsilon_{13}}{(A)^{ch_1}}$
4) $(R_i)^{\rho_{11}} \stackrel{?}{=} \frac{\varepsilon'_{11}}{(B_i)^{ch_1}}$
5) $(R_j)^{\rho_{12}} \stackrel{?}{=} \frac{\varepsilon'_{12}}{(B_j)^{ch_1}}$
6) $(B)^{\rho_{13}} \stackrel{?}{=} \frac{\varepsilon'_{13}}{(C_i/g)^{ch_1}}$
7) $g^{\rho_{21}} \stackrel{?}{=} \frac{\varepsilon_{21}}{(X_i)^{ch_2}}$
8) $g^{\rho_{22}} \stackrel{?}{=} \frac{\varepsilon_{22}}{(X_j)^{ch_2}}$
9) $g^{\rho_{23}} \stackrel{?}{=} \frac{\varepsilon_{23}}{(A)^{ch_2}}$
10) $(Y_i)^{\rho_{21}} \stackrel{?}{=} \frac{\varepsilon'_{21}}{(B_i)^{ch_2}}$
11) $(R_j)^{\rho_{22}} \stackrel{?}{=} \frac{\varepsilon'_{22}}{(B_j)^{ch_2}}$
12) $(B)^{\rho_{23}} \stackrel{?}{=} \frac{\varepsilon'_{23}}{(C_i)^{ch_2}}$
13) $g^{\rho_{31}} \stackrel{?}{=} \frac{\varepsilon_{31}}{(X_i)^{ch_3}}$
14) $g^{\rho_{32}} \stackrel{?}{=} \frac{\varepsilon_{32}}{(X_j)^{ch_3}}$
15) $(Y_i)^{\rho_{31}} \stackrel{?}{=} \frac{\varepsilon'_{31}}{(B_i)^{ch_3}}$
16) $(Y_j)^{\rho_{32}} \stackrel{?}{=} \frac{\varepsilon'_{32}}{(B_j)^{ch_3}}$

If the above 16 equations hold, the proof is correct.

b) **Case 2::** The second statement $(((B_i = g^{x_i y_i}) \wedge (X_i = g^{x_i}) \wedge (Y_i = g^{y_i})) \wedge ((B_j = g^{x_j r_j}) \wedge (X_j = g^{x_j}) \wedge (R_j = g^{r_j})) \wedge ((C_i = g^{\alpha_i\beta_i}) \wedge (A = g^{\alpha_i}) \wedge (B = g^{\beta_i})))$ is true. Generate $r_{21}, r_{22}, r_{23} \in_R \mathbb{Z}_p$ and compute commitments

$$\varepsilon_{21} = g^{r_{21}}, \varepsilon_{22} = g^{r_{22}}, \varepsilon_{23} = g^{r_{23}}$$

and

$$\varepsilon'_{21} = (Y_i)^{r_{21}}, \varepsilon'_{22} = (R_j)^{r_{22}}, \varepsilon'_{23} = (B)^{r_{23}}$$

Then select $ch_1, \rho_{11}, \rho_{12}, \rho_{13} \in_R \mathbb{Z}_p$ and compute

$$\varepsilon_{11} = g^{\rho_{11}}(X_i)^{ch_1}, \varepsilon_{12} = g^{\rho_{12}}(X_j)^{ch_1}, \varepsilon_{13} = g^{\rho_{13}}(A)^{ch_1}$$

and

$$\varepsilon'_{11} = (R_i)^{\rho_{11}}(B_i)^{ch_1}, \varepsilon'_{12} = (R_j)^{\rho_{12}}(B_j)^{ch_1}, \varepsilon'_{13} = (B)^{\rho_{13}}(C_i/g)^{ch_1}$$

Also select $ch_3, \rho_{31}, \rho_{32}, \rho_{33} \in_R \mathbb{Z}_p$ and compute

$$\varepsilon_{31} = g^{\rho_{31}}(X_i)^{ch_3}, \varepsilon_{32} = g^{\rho_{32}}(X_j)^{ch_3}$$

and

$$\varepsilon'_{31} = (Y_i)^{\rho_{31}}(B_i)^{ch_3}, \varepsilon'_{32} = (Y_j)^{\rho_{32}}(B_j)^{ch_3}$$

Let, the grand challenge be ch. Compute $ch_2 = ch - ch_1 - ch_3$. Now, compute three responses

$$\rho_{21} = r_{21} - x_i \cdot ch_2, \rho_{22} = r_{22} - x_j \cdot ch_2, \rho_{23} = r_{23} - \alpha_i \cdot ch_2$$

Publish all the commitments, challenges and the responses.

The verification equations are as that of Case 1.

c) **Case 3::** The third statement $(((B_i = g^{x_i y_i}) \wedge (X_i = g^{x_i}) \wedge (Y_i = g^{y_i})) \wedge ((B_j = g^{x_j y_j}) \wedge (X_j = g^{x_j}) \wedge (Y_j = g^{y_j})))$ is true. Generate $r_{31}, r_{32} \in_R \mathbb{Z}_p$ and compute commitments

$$\varepsilon_{31} = g^{r_{31}}, \varepsilon_{32} = g^{r_{32}}$$

and

$$\varepsilon'_{31} = (Y_i)^{r_{31}}, \varepsilon'_{32} = (Y_j)^{r_{32}}$$

Then select $ch_1, \rho_{11}, \rho_{12}, \rho_{13} \in_R \mathbb{Z}_p$ and compute

$$\varepsilon_{11} = g^{\rho_{11}}(X_i)^{ch_1}, \varepsilon_{12} = g^{\rho_{12}}(X_j)^{ch_1}, \varepsilon_{13} = g^{\rho_{13}}(A)^{ch_1}$$

and

$$\varepsilon'_{11} = (R_i)^{\rho_{11}}(B_i)^{ch_1}, \varepsilon'_{12} = (R_j)^{\rho_{12}}(B_j)^{ch_1}, \varepsilon'_{13} = (B)^{\rho_{13}}(C_i/g)^{ch_1}$$

Also select $ch_2, \rho_{21}, \rho_{22}, \rho_{23} \in_R \mathbb{Z}_p$ and compute

$$\varepsilon_{21} = g^{\rho_{21}}(X_i)^{ch_2}, \varepsilon_{22} = g^{\rho_{22}}(X_j)^{ch_2}, \varepsilon_{23} = g^{\rho_{23}}(A)^{ch_2}$$

and

$$\varepsilon'_{21} = (Y_i)^{\rho_{21}}(B_i)^{ch_2}, \varepsilon'_{22} = (R_j)^{\rho_{22}}(B_j)^{ch_2}, \varepsilon'_{23} = (B)^{\rho_{23}}(C_i)^{ch_2}$$

Let, the grand challenge be ch. Compute $ch_3 = ch - ch_1 - ch_2$. Now, compute two responses as follows

$$\rho_{31} = r_{31} - x_i \cdot ch_3, \rho_{32} = r_{32} - x_j \cdot ch_3$$

Publish all the commitments, challenges and the responses.

The verification equations are as that of Case 1.

Overall it requires at most 28 exponentiations for computing the above zero knowledge proof. Also the space complexity of the proof is 27. The verifier needs to perform 32 exponentiations for verifying all the arguments.