ECC Improved Verifiable Second-Price Auction by Smart Contract

PoChu, Hsu[1,a)] Atsuko, Miyaji[1,b)]

Abstract: A second-price auction is an auction method in which all the bidders submit their bids simultaneously. The highest bidder then purchases the good at the second highest bid price. This scheme does satisfies the requirements of maintaining the bid price secret from the manager while also ensuring financial fairness for the legitimate parties. In this paper, we propose an efficient and secure second-price auction protocol that guarantees financial fairness through the introduction of smart contracts, does not leak any information to the manager, and verifies the correctness of the auction under a malicious model. We also used elliptic curve cryptography to reduce the gas usage in smart contract. This makes 81% improvements on average compared with discrete logarithm implementation.

Keywords: second-price auction, vickrey auction, smart contract, protocol, elliptic curve cryptography

1. Introduction

Second-price auction is an auction in which all bidders submit their bids simultaneously. The bidder who proposed the highest price can purchases the good at the second highest price. In reality, the manager will know every bidder's bid. To construct a scheme in which no participant, even a trusted third party (TTP), knows each bidder's bid, a typical method is to divide the functions of TTP into two independent entities as Second-price sealed-bid auction [7]. In their scheme, all participants do not know any information about the bid price, except the winning price and the winning bidder. However, this scheme is based on the p-th root problem, which causes high time-complexity and requires huge memory usages by public parameters and bid values. Thus, this scheme is unsuitable for blockchain, as the computation power, storage, and memory are all critical resources on blockchain. On the other hand, the M+1st price auction [7] is based on the ElGamal encryption scheme, which is an improvement in terms of time complexity and memory usages. Nevertheless, there's an entity called the manager can obtain the bid price of all the bidders.

To overcome these problems, we introduce the notion of double encryption, which divides the TTP into two independent entities. By applying smart contracts, we propose an efficient and secure second-price auction protocol that guarantees financial fairness and verifies the correctness of the auction. Also, we used Elliptic Curve Cryptography (ECC) to reduce the computation resources usage in smart contract. Which makes 81% improvements on average compared with discrete logarithm implementation.

In this paper, we introduce the related studies on smart contract auction protocols in Section 2 and explain the cryptography preliminaries in Section 3; We propose an efficient and secure second-price auction protocol in Section 4 and describe the implementation and optimization in Section 5. We compare our work with the related works in Section 6. Finally, we conclude our work in Section 7.

2. Related work

2.1 Hawk

Hawk executes Smart contracts while keeping secrets [6] when an bidder and a manager exchange data in the fol-

[1] Osaka University
[a)] hsu@cy2sec.comm.eng.osaka-u.ac.jp
[b)] miyaji@comm.eng.osaka-u.ac.jp

lowing way. 1. The bidders and the manager deposit to the smart contract as a stake when the protocol starts; 2. Zero-knowledge proof (zk-SNARK) for data; 3. Smart contract verifies the zero-knowledge proof and the time limit in each phase. In the Hawk protocol, since the bid amount of all bidders is leaked to the manager by design, it is necessary to assume that the manager does not leak any information inappropriately or collude with any party. In other words, the manager has to be a trusted third party. The Hawk protocol satisfies the following requirements.

On-chain privacy As long as the contractual parties do not intentionally disclose the information, the privacy of the transaction is kept as a secret, and it is disclosed only with the Hawk manager.

Bid privacy Each party does not see others' bids before committing to their own. All parties' bids are independent of others' bids.

TTP-based posterior privacy As long as the manager does not disclose information, users' bids are kept private from another (and from the public) even after the auction.

Financial fairness If bidders or auction managers aborts from the protocol, then the aborting party would be financially penalized while the remaining parties would receive compensation.

Hawk provides the second price auction. The bidder first sends his bid commitment to the Hawk contract (Freeze); then, the Hawk contract verifies the proof and places the proof on the blockchain. The proof indicates the bidder's bid commitment but does not state the bid itself. Next, the bidder encrypts his bid amount with the Manager's public key and sends it to the smart contract for verification and entry (Compute). The Manager decrypts their bidding and calculates the result of the auction. Then, the manager determines the winner as well as the distribution of money and its commitment, and he sends it to the smart contract with zero-knowledge proof. Finally, the smart contract verifies the proof and redistributes the bidding (Finalize).

2.2 Verifiable Sealed-Bid Auction protocol

Another verifiable sealed-bid auction protocol [4] was proposed by using a smart contract and Pedersen commitment scheme [8]. In this protocol, there are bidders and managers just as in Hawk. The manager and a bidder pay a deposit to the smart contract, and the bidder simultaneously sends a bid commitment to the smart contract. Next, the bidder encrypts the value to open the commitment with the manager's public key, and transmit to the smart contract. The manager determines the winning bid and winner, sends the commitment of the winning bid to the smart contract, and proves to the smart contract that the winning bid is higher than any other bid. This scheme achieves bid-privacy, TTP-based posterior privacy, and financial fairness in the same way as the protocol reported in [6]. In addition to these advantages, it satisfies the following conditions:

Bid Binding Once the bidding phase is closed, bidders cannot change their commitments.

Public verifiable correctness The auction contract verifies the correctness of the manager's work to determine the winner.

Non-Interactivity Bidders do not participate with the underlying protocol of the auction contract.

3. Preliminaries

Definition 1 (DDH assumption) Let t be a security parameter. A decisional Diffie-Hellman (DDH) parameter generator $\mathcal{IG}$ is a probabilistic polynomial time (PPT) algorithm that takes an input 1^k and outputs the description of a finite field $\mathbb{F}_p$ and a basepoint $g \in \mathbb{F}_p$ with the prime order q. We say that $\mathcal{IG}$ satisfies the *DDH assumption* if $|p_1 - p_2|$ is negligible (in K) for all PPT algorithms A, where $p_1 = PR[(\mathbb{F}_p, g) \leftarrow \mathcal{IG}(1^K); y_1 = g^{x_1}, y_2 = g^{x_2} \leftarrow \mathbb{F}_p : A(\mathbb{F}_p, g, y_1, y_2, g^{x_1 x_2}) = 0]$ and $p_2 = Pr[(\mathbb{F}_p, g) \leftarrow \mathcal{IG}(1^K); y_1 = g^{x_1}, y_2 = g^{x_2}, z \leftarrow \mathbb{F}_p : A(\mathbb{F}_p, g, y_1, y_2, z) = 0]$.

Definition 2 (ECC based ElGamal scheme) Let p and q be large primes. Let $\langle G \rangle$ denotes a prime subgroup of $GF(p)$ generated by G whose order is q. Given a message $M \in GF(p)$, we define ElGamal [3] encryption as $E_Y(M)$, where Y is the public key. Given a ciphertext C, decryption is defined as $D_x(C)$, where x is the private key.

Definition 3 (Proof of knowledge of a scalar) Given $G \in GF(p)$ and $x \in \mathbb{Z}_q^*$, define $ZKP[x \,|\, Y = xG]$ as the non-interactive zero-knowledge proof of the scalar.

Definition 4 (Proof of equality of two scalar) Given $G_1, G_2 \in GF(p)$ and $x_1, x_2 \in \mathbb{Z}_q^*$, define $ZKP[x_1 \equiv x_2 \pmod q \,\big|\, (Y_1 = x_1 G_1, Y_2 = x_2 G_2)]$ as the

non-interactive zero-knowledge proof of the equality of two scalar defined as in Definition 3.

Algorithm 1 (Binary search) Binary search [2] is an algorithm used to search on an ordered list. It can reduce the time complexity to $O(log(N))$, where N is the number of elements.

Input Given an ordered list A of N elements and a compare function cmp.

Output The index j of the target element $A[j]$. If there are no element satisfies the condition, output the largest i where $cmp(i) = 1$.

We use the symbol $BiSearch[A, cmp] = i$ for it.

Algorithm 2 (Mix and match) Mix and match [5] is a technique used to examine whether the decryption of a ciphertext $D(C)$ belongs to a set of plaintexts. The process is publicly verifiable without revealing the permutation.

Input Ciphertext C and the set of plaintexts $S = \{M_1, M_2, ..., M_n\}$.

Output Output true if $D(C) \in S$. Otherwise, output false.

We use symbol $MixMatch[C, S]$ to represent it.

4. Our scheme

To construct a scheme without any such entity as the so called TTP, who knows the secret information of the bidding values of each bidder, a typical method is to divide the functions of the entity into two entities as in [7, 1]. The protocol of [7] can realize two separate and independent entities; thus, no entity can obtain the bidding prices of any bidder, except the winning price and bidder. However, this scheme has a high computational complexity and requires a huge memory size for public parameters including bidding values, as it is based on the p-th root problem. As the size of the public parameters influences the size of data in blockchain, it should be reduced. In addition, their scheme requires each bidder to use each different bidding price. Thus, the memory size used for bidding points depends on the number of bidders. To overcome the above problems, we introduce the notion of *double encryption*. Double encryption satisfies two-entity independent encryption and decryption operations, which can divide the functions of TTP into two independent entities. In this section, we first propose the notion of the double encryption scheme together with the necessary features and a concrete example of double encryption. Then, we present our scheme.

4.1 Double Encryption

In this section, we describe the definition of the double encryption.

Definition 5 (Double encryption) Let the public and private key pairs of the public key cryptography of the entity be $(Y_1, x_1), (Y_2, x_2)$, respectively. The ciphertext of plaintext using the public key Y_i is $\mathsf{Enc}_{Y_i}(M)$, and the decryption is $\mathsf{Dec}_{Y_i}(\mathsf{Enc}_{Y_i}(M))$. When public key cryptography satisfies the following, the public key cryptosystem is called a double encryption cryptosystem.

- Double encryption: We define the encryption using the public key Y_2 for the ciphertext encrypted with public key Y_1 for a plaintext M through the following operation.

$$\mathsf{Enc}_{Y_1 \cdot Y_2}(M) = \mathsf{Enc}_{Y_2}(\mathsf{Enc}_{Y_1}(M))$$

- Partial decryption: We define the decryption performed by each entity for the ciphertext $\mathsf{Enc}_{Y_1 \cdot Y_2}(M)$ encrypted with the public keys Y_1, Y_2 for the plaintext M through the following operation.

$$\mathsf{Dec}_{Y_1}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M)) = \mathsf{Enc}_{Y_2}(M)$$
$$\mathsf{Dec}_{Y_2}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M)) = \mathsf{Enc}_{Y_1}(M)$$

- Full decryption: We define the decryption performed by both entities for the ciphertext $\mathsf{Enc}_{Y_1 \cdot Y_2}(M)$ encrypted with the public keys Y_1, Y_2 for the plaintext M through the following operation.

$$\mathsf{Dec}_{Y_1}(\mathsf{Dec}_{Y_2}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M))) = M$$

Definition 6 (Commutative encryption) For a given double encryption $\mathsf{Enc}_{Y_1 \cdot Y_2}(M)$, if

$$\mathsf{Dec}_{Y_1}(\mathsf{Dec}_{Y_2}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M))) = \mathsf{Dec}_{Y_2}(\mathsf{Dec}_{Y_1}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M))) = M$$

is satisfied, then the encryption is commutative encryption.

Definition 7 (Bi-homomorphic) When a double encryption satisfies the following properties, we say that the encryption is satisfies bi-homomorphic.

- $\mathsf{Enc}_{Y_1 \cdot Y_2}(M) + \mathsf{Enc}_{Y_1}(\tilde{M}) = \mathsf{Enc}_{Y_1 \cdot Y_2}(M + \tilde{M})$
- $\mathsf{Enc}_{Y_1 \cdot Y_2}(M) + \mathsf{Enc}_{Y_2}(\tilde{M}) = \mathsf{Enc}_{Y_1 \cdot Y_2}(M + \tilde{M})$
- $\mathsf{Enc}_{Y_1 \cdot Y_2}(M) + \mathsf{Enc}_{Y_1 \cdot Y_2}(\tilde{M}) = \mathsf{Enc}_{Y_1 \cdot Y_2}(M + \tilde{M})$
- $\omega \mathsf{Enc}_{Y_1 \cdot Y_2}(M) = \mathsf{Enc}_{Y_1 \cdot Y_2}(\omega M)$

We propose the double encryption based on the ElGamal encryption as follows. We also show that it satisfies bi-homomorphism.

- ElGamal double encryption: The encryption using the public keys Y_2 and Y_1 is as follows.

$$\mathsf{Enc}_{Y_1 \cdot Y_2}(M) = (U_1, U_2, C), where$$
$$U_1 = r_1 G, U_2 = r_2 G \text{ and } C = M + r_1 Y_1 + r_2 Y_2.$$

- Partial decryption: The partial decryption for a ciphertext $\mathsf{Enc}_{Y_1 \cdot Y_2}(M)$ by each entity Y_1, Y_2 is as follows.

$$\mathsf{Dec}_{Y_1}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M)) = (U_1, U_2, C - s_1 U_1) = (U_2, M + r_2 Y_2)$$
$$= \mathsf{Enc}_{Y_2}(M), \text{ or}$$
$$\mathsf{Dec}_{Y_2}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M)) = (U_1, U_2, C - s_2 U_2) = (U_1, M + r_1 Y_1)$$
$$= \mathsf{Enc}_{Y_1}(M), \text{ respectively.}$$

- Full decryption: The decryption performed by both entities for the ElGamal ciphertext $\mathsf{Enc}_{Y_1 \cdot Y_2}(M)$ encrypted with the public keys Y_1, Y_2 for the plaintext M is defined as follows.

$$\mathsf{Dec}_{Y_1}(\mathsf{Dec}_{Y_2}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M))) = \mathsf{Dec}_{Y_1}(\mathsf{Enc}_{Y_1}(M)) = M$$

Thus, our double encryption scheme satisfies partial decryption, full decryption, and commutative encryption.

- Double commutative encryption: The order of ElGamal encryption encrypted with the public keys Y_1, Y_2 for the plain text M can be changed.

$$\mathsf{Dec}_{Y_1}(\mathsf{Dec}_{Y_2}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M))) = M = \mathsf{Dec}_{Y_2}(\mathsf{Dec}_{Y_1}(\mathsf{Enc}_{Y_1 \cdot Y_2}(M)))$$

- Bi-homomorphic: For two given ciphertexts $\mathsf{Enc}_{Y_1 \cdot Y_2}(M) = (U_1, U_2, C)$ and $\mathsf{Enc}_{Y_1}(\tilde{M}) = (\tilde{U_1}, \tilde{C})$ with $U_1 = r_1 G$, $U_2 = r_2 G$, $\tilde{U_1} = \tilde{r_1} G$, $C = M + r_1 Y_1 + r_2 Y_2$, and $\tilde{C} = \tilde{M} + \tilde{r_1} Y_1$, the addition of the ciphertexts is as follows.

$$\mathsf{Enc}_{Y_1 \cdot Y_2}(M) + \mathsf{Enc}_{Y_1}(\tilde{M}) = (U_1, U_2, C) + (\tilde{U_1}, \tilde{C})$$
$$= (U_1 + \tilde{U_1}, U_2, C + \tilde{C}) = \mathsf{Enc}_{Y_1 \cdot Y_2}(M + \tilde{M})$$

For two given ciphertexts $\mathsf{Enc}_{Y_1 \cdot Y_2}(M) = (U_1, U_2, C)$ and $\mathsf{Enc}_{Y_2}(\tilde{M}) = (\tilde{U_2}, \tilde{C})$ with $U_1 = r_1 G$, $U_2 = r_2 G$, $\tilde{U_2} = \tilde{r_2} G$, $C = M + r_1 Y_1 + r_2 Y_2$, $\tilde{C} = \tilde{M} + \tilde{r_2} Y_2$, the addition of the ciphertexts is as follows.

$$\mathsf{Enc}_{Y_1 \cdot Y_2}(M) + \mathsf{Enc}_{Y_2}(\tilde{M}) = (U_1, U_2, C) + (\tilde{U_2}, \tilde{C})$$
$$= (U_1, U_2 + \tilde{U_2}, C\tilde{C}) = \mathsf{Enc}_{Y_1 \cdot Y_2}(M + \tilde{M})$$

For two given ciphertexts $\mathsf{Enc}_{Y_1 \cdot Y_2}(M) = (U_1, U_2, C)$ and $\mathsf{Enc}_{Y_1 \cdot Y_2}(\tilde{M}) = (\tilde{U_1}, \tilde{U_2}, \tilde{C})$ with $U_1 = r_1 G$, $U_2 = r_2 G$, $C = M + r_1 Y_1 + r_2 Y_2$, $\tilde{U_1} = \tilde{r_1} G$, $\tilde{U_2} = \tilde{r_2} G$, $\tilde{C} = \tilde{M} + \tilde{r_1} Y_1 + \tilde{r_2} Y_2$, the addition of the ciphertexts is as follows.

$$\mathsf{Enc}_{Y_1 \cdot Y_2}(M) + \mathsf{Enc}_{Y_1 \cdot Y_2}(\tilde{M}) = (U_1, U_2, C) + (\tilde{U_1}, \tilde{U_2}, \tilde{C})$$
$$= (U_1 + \tilde{U_1}, U_2 + \tilde{U_2}, C + \tilde{C}) = \mathsf{Enc}_{Y_1 \cdot Y_2}(M + \tilde{M})$$

For a given ciphertext $\mathsf{Enc}_{Y_1 \cdot Y_2}(M) = (U_1, U_2, C)$ with $U_1 = r_1 G$, $U_2 = r_2 G$, $C = M + r_1 Y_1 + r_2 Y_2$, the scalar multiplication of the ciphertext is as follows.

$$\omega \mathsf{Enc}_{Y_1 \cdot Y_2}(M) = (\omega U_1, \omega U_2, \omega C)$$
$$= (r_1 \omega G, r_2 \omega G, \omega M + r_1 \omega Y_1 + r_2 \omega Y_2) = \mathsf{Enc}_{Y_1 \cdot Y_2}(\omega M)$$

Thus, we show the following theorem.

Theorem 1 The ElGamal encryption satisfies the double commutative encryption with bi-homomorphism.

Theorem 2 (Verifiable partial decryption) In double encryption, a prover can prove that a decrypted ciphertext $\mathsf{Enc}_{Y_1} = (U_1, C_1)$ is the partial decryption of $\mathsf{Enc}_{Y_1 \cdot Y_2}(M) = (U_1, U_2, C)$ without revealing x_2 by showing the following signiture of knowladge (SPK) , where $U_1 = r_1 G$, $U_2 = r_2 G$, $C_1 = M + r_1 Y_1$, $C = M + r_1 Y_1 + r_2 Y_2$.

$$SPK[(\alpha) : C - C_1 \equiv \alpha U_2 \wedge Y_2 \equiv \alpha G](M)$$

4.2 Features of our protocol

We propose an efficient second-price auction protocol which achieves following properties.

Bid privacy: All the bids except the second highest bid should be private.

Anonymity of the second highest bid: No one can identify the bidder who places the second highest bid.

Strong posterior privacy: As long as the managers do not collude with each other, the bidders' bids are kept private even after the auction.

Bid binding

Robustness: Malicious behaviors in each phase can be found and compensated in time.

Public verifiability

Financial fairness: Bidders and managers may abort or violate the protocol to avoid payment or affect the redistribution of wealth. These malicious parties will be financially penalized, whereas the others will receive compensations.

4.2.0.1 The following notations are used in the protocol:

- SC: smart contract.
- M_1, M_2: managers.
- B: the number of bidders.
- B_i: bidder, $i \in \{1, ..., B\}$.
- P: the number of bidding prices.
- p_j: bidding price, $j \in \{1, ..., P\}$.

Protocol 1 (Full decryption procedure)

Given a ciphertext $\mathsf{Enc}_{y_1 \cdot y_2}(M) = (U_1, U_2, C)$ stored in SC.

(1) For all $i \in \{1, 2\}$, A_i sends $(x_i U_i, \pi_i)$ to SC, where the proof π_i is $ZKP[x_i \equiv x_i \,|\, (x_i U_i, Y_i = x_i G)]$.

(2) SC validates the proofs π_1, π_2 and calculates $M = C - (x_1 U_1) - (x_2 U_2)$.

We use the symbol $\mathsf{DEC}_{\mathsf{M}_1 \cdot \mathsf{M}_2}[M]$ to represent the full decryption protocol.

4.3 Auction protocol

To simplify the statement, we assume that every message sent to SC is verified by SC. If any participant violates the protocol, SC aborts the protocol and financially penalizes the violating participant.

4.3.1 Phase 1. Manager initialization:

Assume that the address of $\mathsf{M}_1, \mathsf{M}_2$ and cryptographic parameters such as p, q, G are set during the deployment of SC. The public parameter $Z = kG$, where $k \leq 2, k \in Z_q$ and bidding price list $\{p_j\}$ are also set to SC during the deployment.

For each manager M, deposit d_{M} amount of ether to SC as stake and submit its public key $Y = xG$ for the double encryption. A proof of knowledge π of x must also be attached. The protocol is as follows.

(1) For each M_i, send (Y_i, π_i) and d_{M_i} amount of ether to SC as a stake.

$$(Y_i, \pi_i) = ZKP[x_i \,|\, x_i G]$$

Y_i is the public key and $x_i \in \mathbb{Z}_q$ is the secret key.

(2) The initialization ends after SC receives all the necessary information or timeout T_1 reached.

4.3.1.1 Success condition:

SC receives valid public keys from all managers.

4.3.2 Phase 2. Bidder joins the auction:

In this phase, the bidder chooses the bid price $b_i \in \{1, ..., P\}$ and uses double encryption to calculate the bidding vector V_i, which is composed of P ciphertexts. The ciphertext is in the form of $\mathsf{Enc}_{Y_1 \cdot Y_2}(tZ)$, where $t = 1$ for the bidding price the bidder wants to choose, and $t = 0$ otherwise. The protocol is as follows.

(1) The bidder B_i sends (V_i) and d_{B_i} amount of ether to SC as stake. V_i is the bidding vector.

$$V_i = \left\{\mathsf{Enc}_{Y_1 \cdot Y_2}(t_{ij} Z)\right\}_j, \text{ where } t_{ij} = \begin{cases} 0 & \text{if } j \neq b_i \\ 1 & \text{if } j = b_i \end{cases}$$

(2) SC stops receiving new bids after T_2.

4.3.2.1 Success condition:

The timeout T_2 has passed and more than 2 bidders have joined.

4.3.3 Phase 3. Bid verification:

To validate the bids, SC, M_1, and M_2 perform the following protocols to ensure each bidder only encrypts one Z and each encrypted tZ is either $0Z$ or $1Z$ without revealing tZ itself. The protocol is as follows.

(1) SC computes U_i, the product of the bidding vector V_i for all $i \in \{1, ..., B\}$.

$$U_i = \sum_{j=1}^{P} \mathsf{Enc}_{Y_1 \cdot Y_2}(t_{ij} Z)$$

To ensure the sum of $\{t_{ij}\}_j$ equals to 1.

(2) Apply the full decryption procedure described in Definition 1 on U_i.

$$V_i = \mathsf{DEC}_{\mathsf{M}_1 \cdot \mathsf{M}_2}(U_i) = \sum_{j=1}^{P} t_{ij} Z$$

(3) SC verifies all the proofs. The sum of $\{t_{ij}\}_j$ equals to 1 if $V_i = \sum_{j=1}^{P} t_{ij} Z$ equals to Z.

(4) Apply the mix and match *MixMatch* described in Definition 2 to verify whether the plaintext of $\mathsf{Enc}_{Y_1, Y_2}(t_{ij})$ is in the set $\{0, 1\}$ for all $i \in \{1, ..., B\}$ and $j \in \{1, ..., P\}$. In this stage, M_2 is the person who performs the r power randomization.

$$MixMatch\left[\mathsf{Enc}_{Y_1, Y_2}(t_{ij}), \{0, 1\}\right]$$

(5) SC stops receiving new messages after T_3. Assume that B_h bidders have passed the verification and those who have not passed are penetrated and removed from the bidder array B.

4.3.3.1 Success conditions:

The following conditions must all satisfied before timeout T_3.

(1) The sum of each bidder's bid is well decrypted.

(2) The decrypted message equals to Z.

(3) The mix and match is valid.

4.3.4 Phase 4. Second-highest-bid decision:

In this phase, we find the second highest bid by aggregating the bidding vector by bidding price. This makes the bidding vector becomes a decremented array. Therefore, we can use binary search to speedup the search when applying $MixMatch[C_j, \{0,1\}]$ to find the largest j where $C_j \notin \{0,1\}$. In this stage, M_2 is the person who performs the r power randomization. The protocol is as follows.

(1) SC computes

(a) A_{ij} for all $i \in \{1, ..., B_h\}$.

By this transformation, $A_{ij} = 1$ if there exists $t_{ij} = 1$ for all $j \in \{j, ..., P\}$.

$$A_{ij} = \sum_{k=j}^{P} \mathsf{Enc}_{Y_1 \cdot Y_2}(t_{ik}Z) = \mathsf{Enc}_{Y_1 \cdot Y_2}(\sum_{k=j}^{P} t_{ik}Z)$$

According to this calculation, a bid whose value is less than the bid price b_i has a ciphertext of Z.

(b) C_j for all $j \in \{1, ..., P\}$.

$$C_j = \sum_{i=1}^{B_h} \mathsf{Enc}_{Y_1 \cdot Y_2}(A_{ij}) = \sum_{i=1}^{B_h} \mathsf{Enc}_{Y_1 \cdot Y_2}(\sum_{k=j}^{P} t_{ik}Z)$$

$$= \mathsf{Enc}_{Y_1 \cdot Y_2}(\sum_{i=1}^{B_h} \sum_{k=j}^{P} t_{ik}Z)$$

The smart contract uses the homomorphic property of double encryption to calculate the product of each bid for a specific bidding price.

(2) Use the binary search described in Definition 1 to $\{C_j\}, j \in \{1, ..., P\}$ to find the largest j where $C_j \notin \{0,1\}$. Let $S_j = MixMatch\left[C_j, \{0,1\}\right]$. Define the compare function cmp as

$$j_{win} = BiSearch[\{C_j\}, cmp]$$

$$cmp(j) = \begin{cases} 1 & \text{if } S_j \text{ is not a valid proof} \\ -1 & \text{if } S_j \text{ is a valid proof} \end{cases}$$

where j_{win} is the second highest bidding price. If $j_{win} = B_h$, there are more than one bidder's bid equals to P. In this case, the managers should adjust P and re-run the protocol.

Note that the ZKP in this step is necessary to hide the highest bid price.

(3) SC stops receiving new messages after T_4.

4.3.4.1 Success condition:

In $BiSearch$ procedure, $L + 1 \equiv R$.

4.3.5 Phase 5. Winner decision:

In this phase, we simply decrypt all A_{ij} for $j = j_{win}$. The bidder who holds the plaintext Z is the winner of the auction. The protocol is as follows.

(1) Apply the full decryption procedure described in Definition 1 on $\{A_{i,j_{win}+1}\}_i$.

$$\Delta_i = \mathsf{DEC}_{\mathsf{M}_1 \cdot \mathsf{M}_2}(\{A_{i,j_{win}+1}\}_i) = \sum_{k=j}^{P} t_{ik}Z$$

Let i_{win} be the i for which $\Delta_i = Z$. The bidder $\mathsf{B}_{i_{win}}$ wins the auction.

(2) SC stops receiving new messages after T_5.

4.3.5.1 Success condition:

The $j_{win} + 1$ bid of bidder $\mathsf{B}_{i_{win}}$ is decrypted and the message equals Z.

4.3.6 Phase 6. Payment:

The bidder who wins the auction sends $d_{p_{j_{win}}}$ amount of ether to the seller through SC. The protocol is as follows.

(1) The bidder $\mathsf{B}_{i_{win}}$ sends $d_{p_{j_{win}}}$ amount of ether to SC.

(2) SC sends $d_{p_{j_{win}}}$ amount of ether to the seller and refunds the initial deposits to all the honest parties within T_6.

4.3.6.1 Success condition:

The deposited stakes of all the managers and bidders are all refunded.

4.4 Features and Security

In this section, we discuss the features and security of our auction. Our scheme satisfies the security requirements unless the managers collude with each other.

Financial fairness: This protocol guarantees financial fairness in each step. Participants who violate the protocol will be detected. Manager initialization: The manager who generates an invalid key will be detected if he cannot provide a valid proof. Bidder joins the auction: A malicious bidder who generates illegal bids (use different Z or use $t_{ij} \notin \{0,1\}$) will be detected by the manager during the bid verification phase. Bid verification: If M_2 is malicious, it will be detected by the verifiable mix-and-match and decryption. If both a bidder and M_2 are malicious, M_1 can use the verifiable decryption to prove bidder and M_2 are malicious. Second-highest-bid decision phase and winner decision phase: As in

$$\{b_i\} = \begin{bmatrix}1\\5\\3\end{bmatrix}, \quad \{t_{ij}\} = \begin{bmatrix}1 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 1\\0 & 0 & 1 & 0 & 0\end{bmatrix}, \quad \{A_{ij}\} = \begin{bmatrix}1Z & 0Z & 0Z & 0Z & 0Z\\1Z & 1Z & 1Z & 1Z & 1Z\\1Z & 1Z & 1Z & 0Z & 0Z\end{bmatrix}, \quad \{C_j\} = [3,2,2,1,1],$$

$$j_{win} = 3, \quad i_{win} = 2, \quad \mathsf{B}_{i_{win}} = \mathsf{B}_2$$

Fig. 1 Example of $\{b_i\}$, $\{t_{ij}\}$, $\{A_{ij}\}$, $\{C_j\}$, j_{win}, i_{win} and $\mathsf{B}_{i_{win}}$ for $N = 3$ and $P = 5$. The bidder B_2 wins and pays p_3 to the seller.

the previous phase, verifiable mix-and-match and decryption can detect any malicious behaviors.

Secrecy of bids: No party knows the bids, including the highest bid, unless the managers collude with each other. The bidders' bid are encrypted by the public key of M_1 and M_2. Thus, neither M_1 nor M_2 knows the bid price. This property is secured by the ElGamal encryption (under the DDH assumption).

Anonymity of the second-highest-bid: All the bids are encrypted by double encryption using M_1 and M_2. Neither M_1 nor M_2 can identify the bidder who offered the second-highest-bid, as the bid is encrypted by the public key of another manager. The only information the managers can obtain is the sum of the bids. Therefore, the identity of the bidder who placed the second highest bid is kept secret.

Public verifiability: In each phase, every calculation result based on a secret must be submitted to SC with a valid proof. Therefore, the correctness of the protocol is publicly verifiable.

Posterior privacy: As mentioned previously, secrecy and anonymity are ensured even after the auction, as long as the managers do not collude with each others.

Robustness: If a bidder submits an invalid bid, M_1 and M_2 can fully decrypt the bid using verifiable decryption. The fraud behavior is thereby detected. If both a bidder and M_2 are malicious, the fraud behavior will be detected during verifiable decryption by M_1. If a bidder and M_1 are malicious, the manager cannot forge a valid proof due to the soundness of the zero-knowledge proof. Consequently, a bidder cannot proceed in an auction with an invalid bid unless the managers collude with each other.

5. Implementation and Optimization

In this section, we introduce our implementation, optimization, and benchmarks. Systematically, there are two parts in this protocol, smart contract and client. Our smart contract [*1] is implemented by solidity 0.6.10. The experimental ABIEncoderV2 [*2] supports struct in struct, array of struct and array of struct in struct, which make codes more clear and more readable. We also used the "solidity-BigNumber" library [*3] for big number computation, which is the only efficient open-source library we can found. In the client part, we use Python and web3.py [*4] for better support of big number computation and cryptographic libraries.

The computational costs of discrete logarithm problem (DLP) based algorithms are usually significantly affected by the key size. Figure 2 shows the gas usage by using 1024-, 2048- and 3072-bit DLP. A common way to solve this problem is to use elliptic curve cryptography (ECC). Our ECC contract used the "elliptic-curve-solidity"[*5] library, which is well tested and provides common NIST series curves such as secp256r1(P256). As an Ethereum virtual machine always uses 256-bit integers, the gas consumption should not have significant differences between P192, P224, and P256. Table 1 lists the gas usages of both DLP 3072 and ECC P256 and the corresponding gas reduction percentage between them. The ECC version can save 75% to 90% gas in each stage and save 81% gas on average.

6. Comparison

In this section, we compared the performance of our scheme with those of the M+1st [1], Second-price [7], and Sealed-Bid [4] scheme in terms of memory usage, time complexity, and interactions. Our scheme does not use TTP in contrast to the other schemes. Our memory usage is as good as that of the M+1st scheme and better than that of the Second-price scheme. The time complexity of our manager is as good as that of the M+1st and Second-price schemes.

[*1] https://github.com/tonypottera24/m_second_price_auction_sol
[*2] https://solidity.readthedocs.io/en/v0.6.10/layout-of-source-files.html#abiencoderv2
[*3] https://github.com/zcoinofficial/solidity-BigNumber
[*4] https://github.com/ethereum/web3.py
[*5] https://github.com/witnet/elliptic-curve-solidity

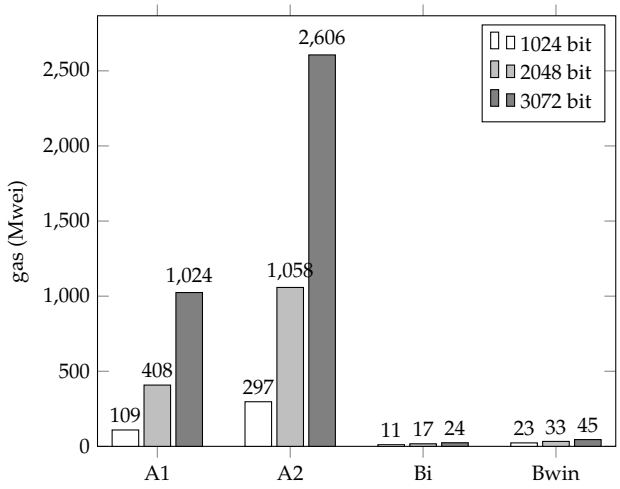

Fig. 2 Gas usages of 1024-, 2048- and 3072-bit DLP (3 bidders and 5 bidding prices)

Table 1 Gas usages of DLP 3072 and ECC P256 (Mwei) (3 bidders and 5 bidding prices)

	DLP 3072			ECC 256		
Phase	M_1	M_2	B_i	M_1	M_2	B_i
1	12	13	-	2 (83%)	1 (92%)	-
2	-	-	24	-	-	4 (83%)
3.1	73	73	-	9 (88%)	9 (88%)	-
3.2.1	-	1,149	-	-	286 (75%)	-
3.2.2	750	797	-	149 (80%)	158 (80%)	-
4.1	-	385	-	-	61 (84%)	-
4.2	106	106	-	12 (89%)	12 (89%)	-
5	83	82	-	10 (88%)	10 (88%)	-
6	-	-	21	-	-	2 (90%)
Sum	1,025	2,606	24 45	182 (82%)	537 (79%)	4 (83%) 6 (87%)

Table 2 Comparison of previous works and our scheme (P: the number of bidding prices, B: the number of bidders, M: the number of challenges and responses) ct: ciphertexts, enc: encryptions, pf: proofs, cm: commitments, h: hash

	TTP	Memory	Manager	Bidder	Interaction
M+1st	Yes	BP ct BP pf	$O(BP)$	P enc $P+1$ pf	$O(\log P)$
Second-price	No	$2BP$ ct $2BP$ h BP pf	$O(BP)$	$2P$ enc	$O(\log P)$
Verifiable Sealed-Bid	Yes	BM cm	$O(BM)$	1 cm 1 enc	$O(B)$
Our scheme	No	BP ct BP pf	$O(BP)$	$2P$ enc	$O(\log P)$

The time complexity of our bidder is as good as Second-price. Last but not least, our round of interaction is as good as that of the M+1st and Second-price scheme and better than that of the Verifiable Sealed-Bid scheme.

7. Conclusion

We proposed an efficient and secure second-price auction protocol that guarantees financial fairness through the introduction of smart contracts, does not leak any information to the manager, and verifies the correctness of the auction.

Acknowledgments This work is partially supported by enPiT(Education Network for Practical Information Technologies) at MEXT, and Innovation Platform for Society 5.0 at MEXT.

References

[1] Masayuki Abe and Koutarou Suzuki. "M + 1-st Price Auction Using Homomorphic Encryption". In: *Public Key Cryptography*. Springer, 2002, pp. 115–124. ISBN: 978-3-540-45664-3.

[2] Jon Louis Bentley. "Multidimensional binary search trees used for associative searching". In: *Communications of the ACM* 18.9 (1975), pp. 509–517.

[3] Taher ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms". In: *IEEE transactions on information theory* 31.4 (1985), pp. 469–472.

[4] Hisham Galal. "Verifiable Sealed-Bid Auction on the Ethereum Blockchain". In: Mar. 2018.

[5] Markus Jakobsson and Ari Juels. "Mix and match: Secure function evaluation via ciphertexts". In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2000, pp. 162–177.

[6] A. Kosba et al. "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts". In: *2016 IEEE Symposium on Security and Privacy (SP)*. 2016, pp. 839–858.

[7] Kazumasa Omote and Atsuko Miyaji. "A Second-price Sealed-bid Auction with the Discriminant of the p_0-th Root". In: *Financial Cryptography, 6th International Conference, FC 2002*. Vol. 2357. Springer, 2002, pp. 57–71.

[8] Torben Pedersen and Bent Petersen. "Explaining gradually increasing resource commitment to a foreign market". In: *International Business Review* 7.5 (1998), pp. 483–501.