Bidder Scalable M+1st-Price Auction with Public Verifiability

Po-Chu Hsu
Graduate School of Engineering
Osaka University, Japan
hsu@cy2sec.comm.eng.osaka-u.ac.jp

Atsuko Miyaji
Graduate School of Engineering
Osaka University, Japan
Japan Advanced Institute of Science and Technology
miyaji@comm.eng.osaka-u.ac.jp

***Abstract*—$M+1$st-price auction, also called Vickrey auction, is a type of sealed-bid auction to sell M identical goods. B bidders secretly choose a price from P bidding points as their bid. The top M bidders can buy the goods at the $M+1$st bidding price. A trusted manager is commonly used to compare these sealed-bids. In our research, trusted manager and trusted mix servers used by mix and match are removed. Instead of cooperating all managers or bidders to find out the winning bidders, winning bidders prove that they are a winner by themself. By further adopt a greedy strategy on searching the $M+1$st-price, the time complexity of each bidder can be reduced to $O(P)$, which is the same as most previous researches. Thus, we construct a scheme that removed the manager without increasing bidders' time complexity. The implementation shows that the gas usage reduced 87% from a manager architecture in a 3 bidder and 6 bidding price setting. The cost to participate in this auction is $12,000,000P$ gas or $600P$ US dollars at this moment, which is enough practical.**

***Index Terms*—M+1st-price auction, blockchain, smart contract, privacy**

I. Introduction

$M+1$st-price auction, also called Vickrey auction, is a type of sealed-bid auction. Bidders submit written bids without knowing other bidders' bid. A simplified version of Vickery auction is a second-price auction. In a second-price auction, the highest bidder can get the good with the second highest price. This was designed to encourage bidders to bid what they truely wanted to pay. The $M+1$st-price auction is used when a good can be divided to M equivilant parts or there are M identical goods. The top M bidders can get goods with the $M+1$st-price. All information except the identity of the top M bidders and the $M+1$st-price should be secret. These are called bids' secrecy and bidders' anonimity. Neverless, the final result, that is, the top M bidders can $M+1$st-price need to be publicky verified. There are many known $M+1$st-price auction protocols. One of the most classical design was given by Abe et al. [1]. They solve this problem elegantly by a semi-homomorphic encryption scheme. Mistunaga et al. [2], [3] follow up this research and use the binary format of the bidding price to decrease the time complexity from P to $\log P$. However, they also require a stronger, full-homomorphic encryption scheme to achieve secrecy and anonimity while keeping the public verifiability. We focus on two important features of trusted manager and public verifiability. All of these researches require a mix server by using mix and match to achieve public verifiability.

A trusted manager is commonly used [1]–[4]. However, the trusted manager knows all secrets, and it is impossible to prove a side channel between the manager and a bidder doesn't exists. i.e. manager might collude with bidder. Another approach to reduce a power of trusted manager is to separate a trusted manager into two managers [5]. To construct a scheme without manager, Smart contract is a very plausible choice. Smart contract is a program running on top of Blockchain. It can be considered as a bulletin board with some computation power and a monetary system. After the smart contract appeared, Hawk [6], as an universal smart contract framework that provides secure computations, gave an auction example. Even though all computations are public verifiable, the trusted manager in Hawk can still know all secrets. In 2018, Verifiable Sealed-Bid Auction [4] was proposed to better fit the nature of smart contract protocol. It aggregates the monetary system to provide financial fairness. However, their scheme also relies on a trusted manager to hide secrets. Furthermore, their scheme does not achieve a feature of public verifiability since it requires interactive proofs. The posterior bid secrecy and bidder anonimity does not hold since a commitment scheme will be opened at the end of the auction. Recently, the first scheme without trusted managers was proposed [7]. They removed trusted manager from Abe [1]'s construction.

Another important issue is public verifiability while keeping bids secrecy and bidders' anonymity. To achieve public verifiability, Mix and match is commonly used in auction protocols [1], [3]–[5], which can publicly verify whether a plaintext m corresponding to a ciphertext $\mathsf{Enc}(m)$ is in a given set $\{m_1, m_2, ...\}$ without revealing m. However, mix and match is not only expensive seen in the mix net [8], [9] (require multiple trusted mix servers), but also slows down the entire protocol since the settlement time of blockchain is not negligible.

One of our contribution can totally increase scalability

	Other bidders	$M + 1$st bidder	Top M bidders
Identity	**Secret**	**Secret**	Public
Price	**Secret**	Public	**Secret**

TABLE I: Public and secret information in $M + 1$st-price auction

even without trusted manager by improving mix and match. In previous research [1], [3]–[5], manager need to use mix and match on M ciphertexts and decrypt some ciphertexts. As a result, the computation cost and communication cost are $O(TBPM)$ ($O(BPM)$/TTP). To remove manager, [7] ask bidders to act as manager. Thus, the cost becomes $O(B^2PM)$ ($O(BPM)$/bidder) since T managers are replaced by B bidders. B^2 is a huge cost. To solve this problem, we proposed a decentralized design. The decision of $M+1$st price and top M winning bidders are not made by a manager [1], [3]–[5], or a group of bidders [7]. In our design, bidders interact with smart contract independently to prove herself/himself as a winner. As a result, the costs are reduced from $O(TBPM)$ ($O(BPM)$/TTP) or $O(B^2PM)$ ($O(BPM)$/bidder) to $O(BP)$ ($O(P)$/bidder). Thus, the time complexity of our scheme is independent to number of bidders. No matter how many bidders join the auction, the cost for each bidder are same.

By analyzing the $M+1$st-price auction, a greedy strategy is used. Instead of mix and match M ciphertexts (require T mix server), we only need to use zero-knowledge equality proof to test one ciphertext. Thus, our design is $O(TM)$ better than previous researches [1], [4], [5].

The features are listed below:

- **No TTP:** No trusted manager or trusted mix servers are used in our scheme.
- **Bid secrecy:** The price of top M bids are secret. All other bids are secrets except the $M+1$st-price.
- **Bid anonymity:** The identity of the $M+1$st-bidder, and all other bidders except the top M bidders.
- **Posterior secrecy and anonimity:** Bid secrecy and bid anonimity holds after the auction ends.
- **Public verifiability:** Every messages sent by bidder are attached with a non-interactive zero-knowledge proof.
- **Financial fairness:** Malicious parties' stake will be sent to honest party as compensation.

In this paper, we explain cryptographic preliminaries in Section II, and propose an efficient and secure $M+1$st-price auction protocol in Section III, describe the implementation and optimization in Section IV, and compare our scheme with the previous researches in Section V. Finally, we conclude our scheme in Section VI.

II. 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 $\epsilon = |p_1 - p_2|$ is negligible (in K) for all PPT algorithms A, where $p_1 = PR[(\mathbb{F}_p, g) \leftarrow \mathcal{IG}(1^K); y_1 = g^{x_1}, y_2 = g^{x_2} \leftarrow \mathbb{F}_p : A(\mathbb{F}_p, g, y_1, y_2, g^{x_1x_2}) = 0]$ 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 (ElGamal encryption [10]): Let p and q be large primes. Let $\langle g \rangle$ denotes a prime subgroup of $\mathbb{Z}_p^*$ generated by g whose order is q. Given a message $m \in \mathbb{Z}_p^*$, we define ElGamal [10] encryption as $\mathsf{Enc}_y(m) = (g^r, m \cdot y^r)$, where y is the public key and $r \in \mathbb{Z}_q^*$. Given a ciphertext $c = (g^r, m \cdot y^r)$, decryption is defined as $\mathsf{Dec}_x(c) = m$, where x is the private key.

Definition 3 (ElGamal based multi-party encryption [7], [11]): Assume there are n encryptors with private key public key pairs $(x_i, y_i), i \in \{1, ..., n\}$ accordingly. Given a message m. The ElGamal ciphertext of m is defined as $\mathsf{Enc}_Y(m) = \mathsf{Enc}_{y_1}(...\mathsf{Enc}_{y_n}(m)) = c$, where $R = \sum_{i=1}^n r_i$ and $Y = \prod_{i=1}^n y_i$. The decryption is defined as $\mathsf{Dec}_X(c) = \mathsf{Dec}_{x_n}(...\mathsf{Dec}_{x_1}(m)) = m$, where $X = \{x_1, ..., x_n\}$.

Theorem 1: ElGamal based multi-entity decryption is commutative [7], [11]. Given two ciphertext $\mathsf{Enc}_{y_2}(\mathsf{Enc}_{y_1}(m)) = c_1$ and $\mathsf{Enc}_{y_1}(\mathsf{Enc}_{y_2}(m)) = c_2$. The decryption can be performed in any order. $\mathsf{Dec}_{x_1}(\mathsf{Dec}_{x_2}(c_1)) = \mathsf{Dec}_{x_2}(\mathsf{Dec}_{x_1}(c_1)) = \mathsf{Dec}_{x_1}(\mathsf{Dec}_{x_2}(c_2)) = \mathsf{Dec}_{x_2}(\mathsf{Dec}_{x_1}(c_2)) = m$.

Theorem 2 (Plaintext equivalence proof of ElGamal ciphertext [12]): Given an ElGamal ciphertext (g^r, my^r) and a plaintext m', the encryptor can prove the value of $m = m'$ without revealing r. This type of proof is based on the proof of equality of two discrete logarithms. i.e. g^r and my^r/m' share the same discrete logarithm r.

Theorem 3 (OR of the Two plaintext equivalence proof of ElGamal ciphertext [12]):

Given an ElGamal ciphertext (g^r, my^r) and plaintexts m' and m'', the value of $m = m'$ or $m = m''$ can be proved without revealing m and r. It is a variation of the plaintext equivalence proof. Helios [12] gave an example of how to construct this proof.

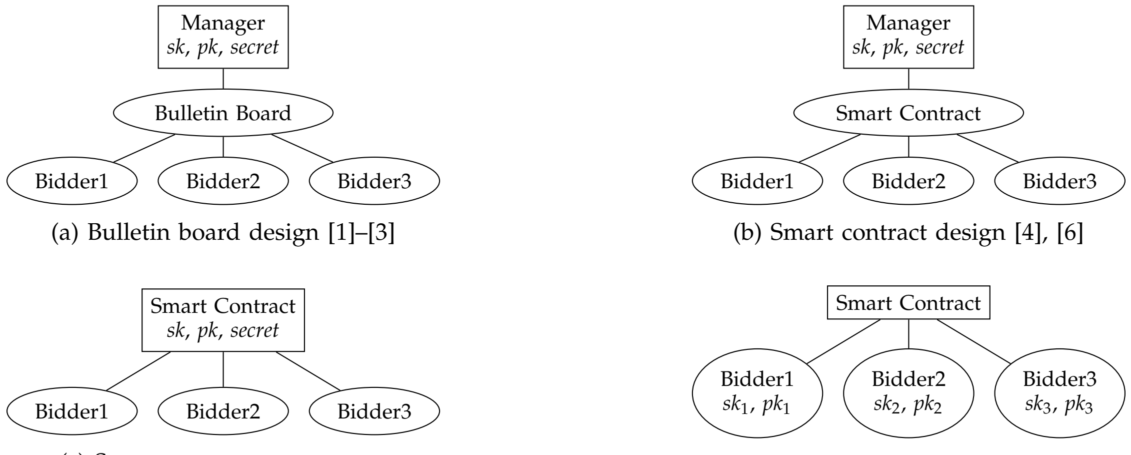
Theorem 4 (Proof of knowledge of a discrete logarithm): Given $g \in \mathbb{Z}_p$ and $x \in \mathbb{Z}_q^*$, the knowledge of g^x's discrete logarithm x can be proved without revealing x.

Theorem 5 (Proof of equality of two discrete logarithms): Given $g_1, g_2 \in \mathbb{Z}_p$ and $x_1, x_2 \in \mathbb{Z}_q^*$, the knowledge of g^x's discrete logarithm x can be proved without revealing x. This type of proof is a variation of the proof of knowledge of a discrete logarithm.

Note that all proofs mentioned above can be non-interactive by using Fiat-Shamir heuristic [13].

III. Our Protocol

Our protocol consists of a seller, who sells M goods and sets price range for M; and bidders, who bid a



(a) Bulletin board design [1]–[3]

(b) Smart contract design [4], [6]

(c) Smart contract as manager.

(d) Bidder as manager.

Fig. 1: (a) and (b): Traditional design. (d): our proposal. Note that (c) doesn't work since smart contract does not have secret storage.

	$\alpha = count(b_i = j_{M+1} \forall i)$	$\beta = count(b_i > j_{M+1} \forall i)$
Case A	$\alpha \geq 1$	$\beta = M$
Case B	$\alpha \geq 1$	$\beta < M$
Case C	$\alpha \geq 1$	$\beta > M$

TABLE II: Possible cases in an $M+1$st-price auction. j_{M+1} is the $M+1$st-price.

price to the good. The M highest bidders can buy the goods by the $M+1$st-price. Remark that no other entity except seller and bidders are required in our protocol. In this section, we show our protocol. An overview of our protocol are as follows.

0) **Smart contract deployment**
1) **Bidder initialization:** Bidders who want to join must submit their stake and their public key used for encryption to the smart contract.
2) **Bidder submits the bids:** Bidders decide their bidding price, encrypt it by all other bidders' public key and submit it to the smart contract.
3) **$M+1$st-price decision preparation:** Bidders randomize the ciphertexts in the smart contract.
4) **$M+1$st-price decision:** This phase will be performed multiple times until the $M+1$th-price is found.
5) **Winner decision:** The top M winning bidders must submit a zk proof to the smart contract to prove that they win the auction.
6) **Payment:** The top M winning bidders pay the seller the $M+1$st-price for the goods.

A. Greedy strategy to the $M+1$st price:

The main goal of the $M+1$st-price auction is to find the boundary between the Mst-price and the $M+1$st-price. In table II, assume α is the number of bidders whose bid b_i is same as j_{M+1}, and β is the number of bidders whose bid b_i is larger than j_{M+1}. A $M+1$st-price auction can be separated into three cases:

- **Case A (valid):** There are exactly M bidders whose bid is larger than the $M+1$st-price. In this case, the top M bidders pay $p_{j_{M+1}}$ for the goods.
- **Case B (invalid):** The Mst-price is same as the $M+1$st-price. $M-\beta$ amount of bidders must be picked from the top α bidders as winner. In our research, we consider this case as invalid. The protocol aborts without leaking any information of α and β. The seller can increase the set of possible bidding prices and restart the auction. In previous research [1], [3], they consider the top α bidders as winner and leak the number of α.
- **Case C (invalid):** This case will never happen.

B. Zero-knowledge Equality Proof of Aggregated Ciphertexts (ZKEP-AC):

- Input: 1. Target z^M. 2. A product of ciphertexts $c = \prod_{i=1}^{B} a_i$, where $a_1, ..., a_B$ are ciphertexts generated by bidder $\mathsf{B}_1, ..., \mathsf{B}_B$ accordingly.
- Output: True if $\mathsf{Dec}(c) = z^M$. Otherwise, return False.

Let $c = \prod_{i=1}^{B} a_i = \prod_{i=1}^{B}(g^{r_i}, z^{t_i} Y^{r_i}) = (g^{r_1+...+r_B}, z^{t_1+...+t_B} Y^{r_1+...+r_B}) = (g^R, z^T Y^R)$. Traditionally, an equality proof proves the knowledge of randomness R. However, c is a product of ciphertexts generated by all bidders. A way to prove this is to aggregate the proof of knowledge of randomness r_i generated by each bidder B_i. Another way is to aggregate the proof of knowledge of secret key x_i. However, it is either difficult to verify each bidder's proof before it is aggregated or it can break the secrecy of a_i. Therefore, we proposed a three step approach.

1) SC divides $c = (u, v)$ by z^M:

$$c' = (u, v/z^M)$$

2) For all bidders B_i, get $c' = (u', v')$ from SC, power by a random number $w_i \leftarrow \mathbb{Z}_q$ to hide the plaintext of c if it is not z^M. The bidder B_i then send $c''_i = (u'^{w_i}, v'^{w_i})$ back to SC. A same discrete log zero-knowledge proof is attached to prove the power w_i are same. This step is asynchronous. Each bidder B_i can generate their own c''_i without waiting for other bidders.
3) SC aggregates all $c''_i = (u''_i, v''_i)$, $i = 1, ..., B$ as C.

$$C = \left(\prod_{i=1}^{B} u''_i, \prod_{i=1}^{B} v''_i\right) = \left(g^{\sum_{i=1}^{B} r_i \sum_{i=1}^{B} w_i}, z^{((\sum_{i=1}^{B} t_i)-M)\sum_{i=1}^{B} w_i} Y^{\sum_{i=1}^{B} r_i \sum_{i=1}^{B} w_i}\right)$$

4) Decrypt $C = (U, V)$:
   a) All bidders B_i send U^{x_i} to SC with a same discrete log zero-knowledge proof that x_i is the same secret key as $y_i = g^{x_i}$.
   b) SC calculates

$$V \cdot U^{-x_1} \cdots U^{-x_B} = z^{((\sum_{i=1}^{B} t_i)-M)\sum_{i=1}^{B} w_i} = \sigma$$

5) Return True if $\sigma = 0$. Otherwise, return False.

C. Auction Protocol:

Smart contract deployment:

The following parameters are deployed in the smart contract (SC) to start the auction.

- Cryptographic parameters: Large prime p, a base-point g with prime order q, and a auction base $z \leftarrow \mathbb{Z}_p \backslash \{0, 1\}$.
- Seller initialization:
  – Bidding price list $\{p_1, ..., p_P\}$ of P bidding points.
  – Timeouts for each phases $T_1, ..., T_6$: Failed to submit things to smart contract within a given time period will be treat as a violation to the protocol and be financially penalized.
  – Seller's stake $d_\S$: Seller submit $d_\S$ amount of ether as stake to start the auction.
  – Bidders' stake requirement d_B: Bidders are required to submit d_B amount of ether as stake to join the auction.

Phase 1. Bidder initialization:

Bidders join the auction within time T_1 as follows. We assume that there are more than $M + 1$ bidders.

- Each bidder $\mathsf{B}_i, i \in \{1, ..., B\}$ submit following messages to the SC:
  – Public key $y_i = g^{x_i}$ for the ElGamal encryption, where $x_i \leftarrow \mathbb{Z}_q$ is a randomly chosen secret key.
  – The proof of $y_i = g^{x_i}$: This proof can be constructed by using proof of knowledge of y_i's discrete logarithm x_i in Section II.
  – d_B amount of ether as stake.
- After this phase end, an aggregated public key $Y = \prod_{i=1}^{B} y_i$ can be calculated by SC.

Phase 2. Bidder submit their bids:

All bidders must submit their bids to SC within time T_2 as follows:

- Each bidder $\mathsf{B}_i, i \in \{1, ..., B\}$ submit following messages to the SC:
  – Encrypted bidding vector $V_i = (V_{i1}, ..., V_{iP})$:

$$V_{ij} = \begin{cases} \mathsf{Enc}_Y(z^1) & \text{if } j = b_i \\ \mathsf{Enc}_Y(z^0) & \text{if } j \neq b_i \end{cases}$$

The bidder B_i first chooses a bidding point $b_i \in \{1, ..., P\}$ from the price list $\{p_1, ..., p_P\}$, and generates the bidding vector V_i. The elements in V_i should contain exactly one $\mathsf{Enc}_Y(z^1)$ and $P - 1$ amount of $\mathsf{Enc}_Y(z^0)$. An equivilant statement is

$$\left(V_{ij} \in \{\mathsf{Enc}_Y(z^0), \mathsf{Enc}_Y(z^1)\} \forall j\right) \wedge \left(\prod_{j=1}^{P} V_{ij} = \mathsf{Enc}_Y(z^1)\right)$$

  – The proof of $V_{ij} \in \{\mathsf{Enc}_Y(z^0), \mathsf{Enc}_Y(z^1)\} \forall j$: This proof can be constructed by using an OR proof of $\mathsf{Dec}(V_{ij}) \in \{z^0, z^1\}$ in Section II.
  – The proof of $\prod_{j=1}^{P} V_{ij} = \mathsf{Enc}_Y(z^1)$: This proof can be constructed by using a plaintext equality proof of plaintext z^1 in Section II.

Phase 3. $M + 1$st-price decision preparation:

In this phase, SC first calculates a $c = (c_1, ..., c_P)$ array. Then performs ZKEP-AC on all $c_1, ..., c_P$. The smallest j where $\mathsf{Dec}(c_{j+1}) = z^M$ is the $M + 1$st bidding point. Step 1 and 2 of ZKEP-AC are performed on all ciphertexts $c_1, ..., c_P$ within time T_3. The protocol is as follows:

1) SC calculates array $a_i = (a_{i1}, ..., a_{iP})$ for all bidder B_i:

$$a_{ij} = \prod_{k=j}^{P} V_{ik} = \mathsf{Enc}_Y(z^{\sum_{k=j}^{P} t_{ik}})$$

For all j, the value of a_{ij} equals to $\mathsf{Enc}_Y(z^1)$ if the bid b_i is larger than or equals to j. Otherwise, $a_{ij} = \mathsf{Enc}_Y(z^0)$. A better way to reduce the time complexity from $O(P^2)$ to $O(P)$ is to compute $a_{ij} = a_{i(j+1)} \cdot V_{ij}$.

2) SC calculates array $c = (c_1, ..., c_P)$:

$$c_j = \prod_{i=1}^{B} a_{ij} = \mathsf{Enc}_Y(z^{\sum_{i=1}^{B} \sum_{j=1}^{P} t_{ij}})$$

The value of c_j is the number of bidders whose bid b_i is larger than or equals to j. Figure 2 provides an example of array a and array c.

3) By all bidders' cooperation, SC performs ZKEP-AC protocol from $j = 1$ to $j = P$ to find out the smallest j where $\mathsf{Dec}(c_{j+1}) = z^M$. This j is the $M + 1$st bidding point.

Phase 4. $M + 1$st-bid decision:

In this phase, step 4 of ZKEP-AC is performed on $C = (C_1, ..., C_P)$. i.e. decrypt $C_1, ..., C_P$. However, since the smallest $k > j$ where $c_k \neq \mathsf{Enc}_Y(z^M)$ is the Mst bidding

$$\begin{array}{lrlcccccll}
 & & & p_1 & p_2 & p_3 & p_4 & p_5 & & \\
b_1 = 2, & V_1 & = [& \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^0) &], & \\
b_2 = 4, & V_2 & = [& \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^0) &], & \\
b_3 = 3, & V_3 & = [& \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^0) &], & \\
 & & & & & & & & & \\
 & a_1 & = [& \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^1), & \boxed{\mathsf{Enc}_Y(z^0)}, & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^0) &] & \\
 & a_2 & = [& \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^1), & \boxed{\mathsf{Enc}_Y(z^1)}, & \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^0) &] & \text{win} \\
 & a_3 & = [& \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^1), & \boxed{\mathsf{Enc}_Y(z^1)}, & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^0) &] & \text{win} \\
 & & & & & & & & & \\
 & c & = [& \mathsf{Enc}_Y(z^3), & \mathsf{Enc}_Y(z^3), & \mathsf{Enc}_Y(z^2), & \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^0) &] & \\
\text{ZKEP-AC} & \#1\ c' & = [& \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^1), & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^{-1}), & \mathsf{Enc}_Y(z^{-2}) &] & \\
\text{ZKEP-AC} & \#2\text{-}3\ C & = [& \mathsf{Enc}_Y(z^{W_1}), & \mathsf{Enc}_Y(z^{W_2}), & \mathsf{Enc}_Y(z^0), & \mathsf{Enc}_Y(z^{-W_4}), & \mathsf{Enc}_Y(z^{-2W_5}) &] & \\
 & & & & & & & & & \\
\text{ZKEP-AC} & \#4 & = [& \mathit{False}, & \mathit{False}, & \mathit{True}, & \text{-}, & \text{-} &] & \\
 & & & \rightarrow & j_{M+1st} & \text{-} & \text{-} & \text{-} & &
\end{array}$$

Fig. 2: Example of $M = 2$ with 3 bidders and 5 bidding prices. $W_1, ..., W_5$ are the randomness introduced by ZKEP-AC. The proof of $C = \mathsf{Enc}_Y(z^0)$ is performed from left to right, and stops at $j_{M+1st} = 2$. Bidders who can provide a valid zk proof of $a_{i3} = \mathsf{Enc}_Y(z^1)$ (boxed ciphertexts) can win the auction. The bidder B_2 and B_3 win the auction and pay $p_{j_{M+1st}} = p_2$ for the goods.

point, the decryption should be performed one by one from left to right. It is important to stop when we find the smallest j where $c_{j+1} = \mathsf{Enc}_Y(z^M)$ to keep k secret. The protocol is as follows:

- All bidders B_i submits following messages to the SC:
  1) The decryption message $U_j^{x_i}$ of ciphertext $C_j = (U_j, V_j)$ for all $j = 1, ..., P$: After all $U_j^{x_i}$ are collected, the ciphertext can be decrypted by $V_j \prod_{i=1}^{B} U_j^{-x_i}$.
  2) The proof that $U_j^{x_i}$ is calculated by the secret key x_i for all $j = 1, ..., P$: This proof can be constructed by using the same discrete logarithm proof of $U_j^{x_i}$ and $y_i = g^{x_i}$ in Section II.

Phase 5. Winner decision:

Let the j found in the previous phase be j_{M+1st}. In this phase, All bidders B_i whose bid b_i is larger than j_{M+1st} can submit a proof that $a_{i,j_{M+1st}+1} = \mathsf{Enc}_Y(z^1)$ and win the auction. This behavior will not leak b_i since $a_{i,j_{M+1st}+1} = \prod_{j=j_{M+1st}+1}^{P} V_{ij}$. All elements in $V_{i,j_{M+1st}+1}, ..., V_{iP}$ can be $\mathsf{Enc}_Y(z^1)$. The protocol is as follows:

- Bidders B_i whose bid $b_i > j_{M+1st}$ can submit following messages to the SC:
  1) The proof of $a_{i,j_{M+1st}+1} = \mathsf{Enc}_Y(z^1)$: This proof can be constructed by using a plaintext equality proof of plaintext z^1 in Section II.

Phase 6. Payment:

The bidders who win the auction send $p_{j_{M+1st}}$ amount of ether to the seller through SC. The protocol is as follows:

- All winning bidders pay $p_{j_{M+1st}}$ amount of ether to SC.
- The seller use winning bidders' public keys to encrypt the goods individually with a proof and sends them to SC.
- SC sends bidders' payment to the seller.

D. Features and Security

The design of this protocol provides following properties:

- **Scalability:** In our protocol, each bidder's computation or communication, even the SC execution are only related to the bidder itself. Therefore, the computations, communications, and storage used by the bidder and SC are scalable on the number of bidder B.
- **Bid Binding:** According to our implementation, the functions in SC will not allow bidders to change their bidding point after the bid submission phase is closed.
- **Bid secrecy:** The bidding vectors $V_1, ..., V_B$ are encrypted by all bidders' public keys $Y = y_1 \cdots y_B$. Without all bidders' help, the bid is kept as a secret. The decryption of $C_1, ..., C_P$ in phase 4 is randomized in phase 3. An adversary cannot get any information if the plaintext is not z^M. The secrecy of top M bids is protected in phase 5. If $a_{i,j_{M+1st}+1}$ is a encrypted z^1, all bids from $a_{i,j_{M+1st}+1}$ to a_{iP} can contain z^1.
- **Bidder anonymity:** Since $c_1, ..., c_P$ are products of ciphertexts generated by all bidders. i.e. $c_j = \prod_{i=1}^{B} a_{ij}$. The proof of $\mathsf{Dec}(c_j) = z^M$ will not leak $a_{ij}, i = 1, ..., B$. Thus, the identity of the $M + 1$st bidder is still a secret.
- **Posterior secrecy and anonymity:** The bidding points $b_i, i = 1, ..., B$ and bidding vectors $V_i, i = 1, ..., B$

are still secrets even after the auction. Thus, posterior secrecy and anonymity still holds.

- **Robustness:** Even if some malicious bidders violate the protocol, the auction can still continue if there are $M + 1$ honest bidders. The robustness of each phase is as follows:
  - Phase 1. Bidder initialization: The seller can reduce M if there are no enough bidders.
  - Phase 2. Bidder submit their bids: As long as there are $M + 1$ bidders submit their bid, the auction can continue by skipping those malicious bidders' bid.
  - Phase 3. $M + 1$st-price decision preparation: As long as there are sufficient bidders help the randomization in ZKEP-AC, the auction can continue.
  - Phase 4. $M + 1$st-bid decision: The auction can either continue by skipping absence bidders if a threshold encryption is used or rollback to phase 2.
  - Phase 5. Winner decision: All winning bidders should claim their rights in this phase. However, the auction can continue if some winning bidders gave up their rights. The seller can still ask all bidders' help to decrypt all $a_{i,j_{M+1st}+1}$, $i = 1, ..., B$ to find out all winning bidders if needed.
- **Public verifiability:** All messages sent to SC are attached with a public verifiable non-interactive proof, which can be verified by smart contract immediately. Therefore, the correctness of the protocol is public verifiable.
  - Phase 1. Bidder initialization: A public key $y_i = g^{x_i}$ is submitted by each bidder B_i to SC, $i = 1, ..., B$. The proof of knowledge of x_i is public verifiable. Thus, the correctness of the public key is public verifiable.
  - Phase 2. Bidder submit their bids: A valid bidding vector $V_i = (V_{i1}, ..., V_{iP})$ consists of $P - 1$ amount of $\mathsf{Enc}_Y(z^0)$ and one $\mathsf{Enc}_Y(z^1)$.
    * For all $j = 1, ..., P$, a OR proof of $\mathsf{Dec}(V_{ij}) \in \{z^0, z^1\}$ is submitted to SC to prevent malicious bidders from submitting V_{ij} other than $\mathsf{Enc}_Y(z^0)$ and $\mathsf{Enc}_Y(z^1)$.
    * A equality proof of $\mathsf{Dec}(\prod_{j=1}^{P} V_{ij}) = z^1$ is used to prevent malicious bidders from chosing multiple bidding points.

    Thus, by using $P + 1$ non-interactive zero-knowledge proofs, the bidding vectors are public verifiable.
  - Phase 3. $M + 1$st-price decision preparation: A same discrete log proof is used in the randomization of ZKEP-AC. Therefore, the randomization process is public verifiable.
  - Phase 4. $M + 1$st-bid decision: The proof of same discrete logrithm that $u_{ij}^{x_i}$ and public key $y_i = g^{x_i}$ has same discrete logrithm x_i is public verifiable. Thus, the decryption step in ZKEP-AC is public verifiable.
  - Phase 5. Winner decision: A public verifiable equality proof of $a_{i,j_{M+1st}+1} = \mathsf{Enc}_Y(z^1)$ is used to prove a bidder wins the auction. Thus, the winner decision is public verifiable.
- **Correctness:** The correctness of each phase is as follows:
  - Phase 1. Bidder initialization: All public keys $y_i = g^{x_i}, i = 1, ..., B$ are publicly verified. Thus, the correctness holds.
  - Phase 2. Bidder submit their bids: public verifiable proofs of $\mathsf{Dec}(V_{ij}) \in \{z^0, z^1\}$ $j = 1, ..., P$ and $\mathsf{Dec}(\prod_{j=1}^{P} V_{ij}) = z^1$ are verified by SC. Thus, the correctness of the bidding vector V_i holds. V_i contains exactly $P-1$ amount of $\mathsf{Enc}_Y(z^0)$ and one $\mathsf{Enc}_Y(z^1)$.
  - Phase 3 and 4. $M+1$st-bid decision: According to the rule of $M+1$st auction, the Mst bidding price must be different from the $M+1$st bidding price. Thus, the smallest j where $\mathsf{Dec}(c_{j+1}) = z^M$ is the $M+1$st bidding point. The correctness holds.
  - Phase 5. Winner decision: Only bidders who can prove $a_{i,j_{M+1st}+1} = \mathsf{Enc}_Y(z^1)$ can win the auction. $a_{i,j_{M+1st}+1} = \mathsf{Enc}_Y(z^1)$ if and only if $b_i \le j_{M+1st} + 1$. Thus, the correctness holds.
- **Financial fairness:** The seller and all bidders are asked to deposit some amount of ether in the smart contract when they join the auction. If they perform any malicious behavior, smart contract can send their stake to others as compensation.

IV. Implementation and Optimization

There are two main parts to this protocol, smart contract [1] and web3 client [2]. The gas usages are estimated by ganache-cli, a successor of ethereumjs-testrpc. Figure 3 shows the gas usage by using 1024-, 2048- and 3072-bit primes. To reach 3072-bit security level, the cost is tremendous and unacceptable. By adopting the elliptic curve cryptography (ECC), the cost reduced up to 80% for 256-bit ECC compared with 3072-bit DLP. Both of them have same security level. On average, the gas usage is $12,000,000P$ for each bidder, where P is the length of the price list. If the gas price is 50 Gwei, and the ethereum is $1,000$ USD/ether, the cost of an auction is $600P$ USD for each bidder.

To inspect how many gas is needed from removing the manager and how much gas can we save from removing mix and match, a manager scheme (based on AS [1]) and a no manager scheme (based on HM [7]) was implemented. Table III shows the gas usage of different phases. In phase 2 and phase 3, as the most costly part

[1]https://github.com/tonypottera24/m-1st_auction_dlp_sol
[2]https://github.com/tonypottera24/m-1st_auction_dlp_py

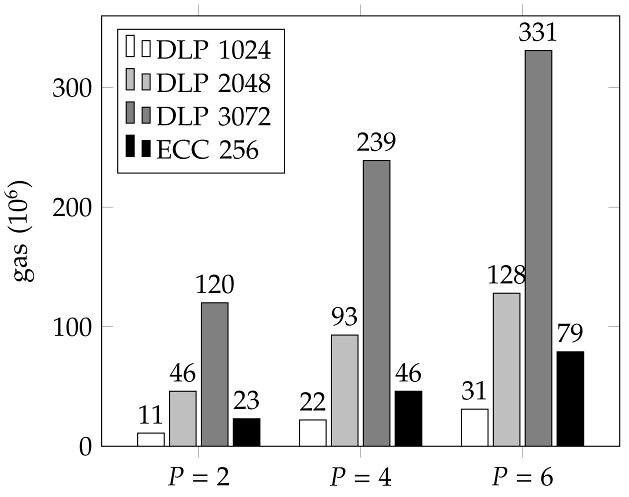


Fig. 3: Gas usages of 1024-, 2048-, 3072-bit DLP and 256-bit ECC

TABLE III: The gas usage of [1], [7], and our scheme by using ECC 256 (3 bidders and 6 bidding prices).

	AS [1]	HM [7]		Our scheme		
Manager	Yes	No		No		
Mix & match	Yes	Yes		No		
Role / cmp	Manager	Bidder	AS	Bidder	HM	AS
1.	1	1	−0%	1	−0%	−0%
2.	5	3	−40%	54	−85%	−90%
3.	**520**	**350**	−33%			
4.	**79**	**52**	**−34%**	21	−60%	−73%
5.	9	8	−11%	3	−63%	−67%
6.	4	2	−50%	0.3	−85%	−93%
Overall	618	416	−33%	79	−81%	-87%

of an auction protocol, are improved up to 90% after we use zero-knowledge proof to replace mix and match.

V. Comparison

The detailed comparisons of each phases are as follows:

1) **Phase 1. Bidder initialization:** In previous research [1]–[3], [14], the bids are encrypted only by managers public key. The bid secrecy, bidder anonymity, etc are relied on the trusted manager. On the other hand, in our scheme, the ciphertexts are encrypted by all bidders' public key. Without all bidders' collaboration, no one can break the bid secrecy and bidder anonimity, etc.
2) **Phase 2. Bidder submit their bids:** A valid bidding vector $V_i = (V_{i1}, ..., V_{iP})$ should contain exactly $P - 1$ amounts of $\mathsf{Enc}_Y(z^0)$ and one $\mathsf{Enc}_Y(z^1)$. In previous research [1]–[3], [14], bidding vector verification contains two parts. 1. $V_{i1} \in \{\mathsf{Enc}_Y(z^0), \mathsf{Enc}_Y(z^1)\}$; 2. $\sum_{j=1}^{P} V_{ij} = \mathsf{Enc}_Y(z^1)$. The first part is accomplished by mix and match [15]. This requires T (trusted) mix servers to perform mix (secure shuffle [16]–[18]) and match (zk equality proof). The second part is accomplished by asking trusted manager to decrypt $\sum_{j=1}^{P} V_{ij}$.
   In our scheme, instead of asking trusted manegers to verify the bidding vector V_i, bidder B_i submit non-interactive zero-knowledge proofs to prove their bidding vector V_i is valid. SC can verify these proofs immediately without any other bidders' help. Compared with previous works, this reduces the local computation cost and communication cost from $O(BP)$ to $O(P)$.
3) **Phase 3 and 4. $M + 1$st-bid decision:** In previous research [1]–[3], [14], the $M + 1$st bidding point j is defined as $\mathsf{Dec}(c_j) \notin \{z^0, ..., z^M\}$ but $\mathsf{Dec}(c_{j+1}) \in \{z^0, ..., z^M\}$. The test is accomplished by using mix and match over T mix servers. This method can find j without revealing array c. However, if $c_{j+1} \neq z^M$, i.e. some bidders bid the same bidding price as the $M + 1$st bidder, there are less than M bidders wins the auction. Previous research will still determine these bidders as winner. This violates the rule of the $M + 1$st auction. There should be exactly M winner wins the auction by paying the $M + 1$st price.
   In our scheme, we used a greedy strategy. Instead of using mix and match to test M values, we only need to use zero-knowledge equality proof to test if $\mathsf{Dec}(c_j) = z^M$. This reduced the time complexity by T (no mix server) and M (only test z^M). This test may repeat up to P times, and each time cost $O(1)$ for each bidder. Therefore, the time complexity of this phase is $O(P)$ for each bidder. Compared with previous works, this reduces the local computation cost and communication cost from $O(BPM)$ to $O(P)$.
4) **Phase 5. Winner decision:** In previous research [1], trusted manager(s) decrypts all $a_{i,j_{M+1st}+1}$, $i = 1, ..., B$ to find out the winning bidders. In our scheme, each winning bidder B_i can provide a proof that $\mathsf{Dec}(a_{i,j_{M+1st}+1}) = z^1$ to prove that he is a winner. Therefore, the time complexity is only $O(1)$ for each bidder. Compared with previous works, this reduces the local computation cost and communication cost from $O(B)$ to $O(1)$.
5) **Phase 6. Payment:** Different from previous research, the goods in our design are encrypted by winners' public key and sent to SC. Seller also gets the ether from SC for the goods.

Compared with previous research, our design is more scalable since the complexity of each bidder is $O(P)$, not $O(BP)$. The cost for each bidder will not increase when the number of bidders increased. Table IV shows that we do not use trusted manager and trusted mix server. The (posterior) *bid secrecy*, *bidder anonymity*, and *robustness* are also not based on TTPs. Also, the usage of smart contract and zero-knowledge proof ensures the *public*

TABLE IV: Comparison of previous researches and our scheme (TM: Trusted Manager, B: Bidder)

	Trusted Manager	Trusted Mix server	(Posterior) Bid Secrecy & Bidder Anonymity	Robustness	Public Verifiability	Financial Fairness	Scalability
AS [1]	Yes	Yes	Based on TM	Based on TM	No	No	No
OM [5]	Yes	Yes	Based on TM	Based on TM	No	No	No
MMO [3]	Yes	Yes	Based on TM	Based on TM	No	No	Scalable on P
GY [4]	Yes	Yes	Based on TM	Based on TM	Interactive	Yes	No
HM [7]	No	No	Yes	Yes	Yes	Yes	No
Our scheme	No	No	Yes	Yes	Yes	Yes	Scalable on B and M

TABLE V: Comparison of the complexity with previous works. (T: the number of trusted managers and mix servers, B: the number of bidders, P: the number of bidding prices, M: number of goods)

	Manager		Bidder		Smart Contract		
	Local computation	Communication costs	Local computation	Communication costs	Computation	Communication costs	Storage used
AS [1]	$O(BPM)$	$O(TBPM)$	$O(P)$	$O(P)$	-	-	-
OM [5]	$O(BPM)$	$O(TBPM)$	$O(P)$	$O(P)$	-	-	
MMO [3]	$O(B\log PM)$	$O(TB\log PM)$	$O(\log P)$	$O(\log P)$	-	-	-
GY [4]	$O(BPM)$	$O(BPM)$	$O(P)$	$O(P)$	$O(TBPM)$	$O(TBPM)$	$O(TBPM)$
HM [7]	-	-	$O(BPM)$	$O(BPM)$	$O(B^2PM)$	$O(B^2PM)$	$O(BPM)$
Our scheme	-	-	$O(P)$	$O(P)$	$O(BP)$	$O(BP)$	$O(BP)$

verifiability and *financial fairness*. Compared with Hsu's scheme [7], we removed the Mix servers and improved the scalability.

Table V compared the local computation cost, communication cost, and space used by manager, bidder, and smart contract. In terms of communication costs, since AS [1], OM [5], and MMO [3] didn't use smart contract, all T mix servers' messages are sent back to the manager. The communication cost of manager is T times of the local computation cost. GY [4], HM [7], and our scheme used smart contract as a message center, so it is T times in GY, and B times in HM and our scheme.

Since our scheme does not use trusted mix servers, our scheme is T times better than previous researches. The greedy strategy also reduced the complexity by M. Thus, the local computation cost and communication cost is only $O(P)$ for each bidder. As a scheme without a trusted manager, this is a great improvement compared to Hsu's scheme. Mitsunaga [3] did great work to compute bids in binary format to reduce P to $\log P$. However, their construction is based trusted manager, mix and match, and full-homomorphic BGN encryption. Their local computation costs and communication costs are also not scalable on B.

VI. Conclusion

In our research, by utilizing smart contracts, this protocol reached the ultimate goal of decentralized apps (DApps): Decentralized: no TTP or manager is used. Scalable: the time and space complexity for each bidder is not related to the number of bidders. Robustness, the auction do not necessarily need to restart if there are some malicious bidders at the first time.

Acknowledgement

This work is partially supported by enPiT(Education Network for Practical Information Technologies) at MEXT, Innovation Platform for Society 5.0 at MEXT, and JSPS KAKENHI Grant Number JP21H03443.

References

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

[2] T. Mistunaga, Y. Manabe, and T. Okamoto, "A secure m+ 1st price auction protocol based on bit slice circuits," in *International Workshop on Security*. Springer, 2011, pp. 51–64.

[3] T. Mitsunaga, Y. Manabe, and T. Okamoto, "A secure m+ 1st price auction protocol based on bit slice circuits," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 99, no. 8, pp. 1591–1599, 2016.

[4] H. S. Galal and A. M. Youssef, "Verifiable sealed-bid auction on the ethereum blockchain," in *International Conference on Financial Cryptography and Data Security*. Springer, 2018, pp. 265–278.

[5] K. Omote and A. Miyaji, "A second-price sealed-bid auction with verifiable discriminant of p 0-th root," in *International Conference on Financial Cryptography*. Springer, 2002, pp. 57–71.

[6] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," in *2016 IEEE symposium on security and privacy (SP)*. IEEE, 2016, pp. 839–858.

[7] P. Hsu and A. Miyaji, "Verifiable m+lst-price auction without manager," in *IEEE Conference on Dependable and Secure Computing, DSC 2021, Aizuwakamatsu, Japan, January 30 - February 2, 2021*. IEEE, 2021, pp. 1–8.

[8] M. Abe, "Universally verifiable mix-net with verification work independent of the number of mix-servers," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1998, pp. 437–447.

[9] D. Wikström, "A universally composable mix-net," in *Theory of Cryptography Conference*. Springer, 2004, pp. 317–335.

[10] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE transactions on information theory*, vol. 31, no. 4, pp. 469–472, 1985.

[11] K. Huang and R. Tso, "A commutative encryption scheme based on elgamal encryption," in *2012 International Conference on Information Security and Intelligent Control*. IEEE, 2012, pp. 156–159.

[12] D. Bernhard and B. Warinschi, "Cryptographic voting—a gentle introduction," in *Foundations of Security Analysis and Design VII*. Springer, 2013, pp. 167–211.

[13] 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.

[14] K. Kurosawa and W. Ogata, "Bit-slice auction circuit," in *European Symposium on Research in Computer Security*. Springer, 2002, pp. 24–38.

[15] M. Jakobsson and A. 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.

[16] C. A. Neff, "A verifiable secret shuffle and its application to e-voting," in *Proceedings of the 8th ACM conference on Computer and Communications Security*, 2001, pp. 116–125.

[17] J. Furukawa and K. Sako, "An efficient scheme for proving a shuffle," in *Annual International Cryptology Conference*. Springer, 2001, pp. 368–387.

[18] J. Groth and S. Lu, "A non-interactive shuffle with pairing based verifiability," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2007, pp. 51–67.