PAPER *Special Section on Data Engineering and Information Management*

A Sealed-Bid Auction with Fund Binding: Preventing Maximum Bidding Price Leakage**

Kota CHIN[†a)], ***Nonmember***, **Keita EMURA**[††*], **Shingo SATO**[†††], ***and*** **Kazumasa OMOTE**[†], ***Members***

SUMMARY In an open-bid auction, a bidder can know the budgets of other bidders. Thus, a sealed-bid auction that hides bidding prices is desirable. However, in previous sealed-bid auction protocols, it has been difficult to provide a "fund binding" property, which would guarantee that a bidder has funds more than or equal to the bidding price and that the funds are forcibly withdrawn when the bidder wins. Thus, such protocols are vulnerable to a false bidding. As a solution, many protocols employ a simple deposit method in which each bidder sends a deposit to a smart contract, which is greater than or equal to the bidding price, before the bidding phase. However, this deposit reveals the maximum bidding price, and it is preferable to hide this information. In this paper, we propose a sealed-bid auction protocol that provides a fund binding property. Our protocol not only hides the bidding price and a maximum bidding price, but also provides a fund binding property, simultaneously. For hiding the maximum bidding price, we pay attention to the fact that usual Ethereum transactions and transactions for sending funds to a one-time address have the same transaction structure, and it seems that they are indistinguishable. We discuss how much bidding transactions are hidden. We also employ DECO (Zhang et al., CCS 2020) that proves the validity of the data to a verifier in which the data are taken from a source without showing the data itself. Finally, we give our implementation which shows transaction fees required and compare it to a sealed-bid auction protocol employing the simple deposit method.

key words: *blockchain, sealed-bid auction, price hiding, fund binding*

1. Introduction

1.1 Background

Non-Fungible Tokens (NFTs) have received much attention recently, as digital assets such as artworks have been traded on blockchains, and open-bid auctions based on smart contracts have been frequently held. For example, the NFT of Kabosu (the name of the female Shiba Inu dog that became the Internet meme "Doge") was one of the most expensive NFT of all, with a bidding price of 1696.9 ETH (approximately 400 million USD using the Ether price on June 11, 2021)[***]. Thus, open-bid auctions based on smart contracts have been used for transactions of relatively expensive assets and have acquired a certain degree of reliability. They also treated relatively cheap assets; for example, the median price of NFTs traded during May 2023 was approximately 300 USD[****]. As another merit of employing smart contracts, it has a high affinity with NFTs (since they are also based on a blockchain-related technology) and it is easy to directly provide the NFT to the winner of the auction. We remark that any data preserved on the blockchain are open. Thus we need to carefully consider security and privacy when we design a protocol employing a smart contract because anyone obtains data used by the smart contract.

In an open-bid auction, there are two types, English auction and Dutch auction. In an English auction, bidders compete to raise the bidding price until there is only one bidder left. In a Dutch auction, the price is decreased from the initial amount until the first bidder appears. In a sealed-bid auction, the second-price sealed-bid is called the Vickrey auction where the winner pays the second highest bid. Krishna [3] showed that, based on game theory, the English auction is equivalent to the Vickrey auction when bidders evaluate the value of the item in private. Moreover, the Dutch auction is strategically equivalent to the first-price sealed-bid auction (see a nice summarization given by Bag et al. [4]). Since smart contracts are recently employed, to the best of our knowledge, no game theoretic estimation against an auction based on a smart contract has been shown. However, as a fact, anyone can view the balance of addresses used during the bidding process if smart contracts are employed, and can guess the budgets of other bidders. Thus, the final selling price may be lower because bidders do not bid a price higher than the budgets of other bidders. This could be a detriment of the seller and auctioneer. Therefore, it seems desirable to guarantee that no bidder can know the balance of addresses of other bidders in advance when a smart contract is employed.

1.2 Previous Work

Many sealed-bid auction protocols employing smart contracts have been proposed. However, this type of protocol

Manuscript received June 13, 2023.
Manuscript revised November 12, 2023.
Manuscript publicized February 6, 2024.

[†]The authors are with University of Tsukuba, Tsukuba-shi, 305–8573 Japan.
[††]The author is with the National Institute of Information and Communications Technology, Koganei-shi, 184–8795 Japan.
[†††]The author is with Yokohama National University, Yokohama-shi, 240–8501 Japan.
[*]Presently, the author is with Kanazawa University.

[**]An extended abstract appeared at Computer Security Symposium 2021 [1] and IEEE Blockchain 2022 [2]. This is the full version.

a) E-mail: s2120540@u.tsukuba.ac.jp
DOI: 10.1587/transinf.2023DAP0002

[***]Transaction detail is available at https://etherscan.io/tx/0x8668bb338f7cf9896db75c00e8bef18cc549d04b2dcaf1cee01dc0e1522e7e87

[****]Calculation detail is available at (https://dune.com/queries/351118)

necessitates ensuring a bidder has funds more than or equal to the bidding price (called "fund binding"), but because the bidding prices are hidden, this is difficult to do. Such protocols are vulnerable to a false bidding, meaning bidders indicate bidding prices, but do not have the money to be paid. As a countermeasure, Galal and Youssef [5] and Li et al. [6] considered using a small deposit, wherein if bidding prices are false, then the deposit is automatically confiscated. However, if some bidders do not care about such a penalty because of a small deposit, then their proposals do not prevent bidders from submitting a false bidding. Another countermeasure against submitting a false bidding is called "simple deposit method" where each bidder is required to send a deposit more than or equal to the bidding price. The simple deposit method has been widely employed, e.g., [7]–[18]. Although each bidder can hide their actual bidding price by submitting a deposit more than the actual bidding price, the maximum bidding price is leaked because deposit information is publicly available on smart contracts. Thus, the final selling price may be lower because bidders do not bid a price higher than the highest deposit. Thus, it is preferable to hide the maximum bidding price, so items being auctioned sell closer to their actual value. In the Ma et al.'s sealed-bid auction protocol [19], a commitment on a deposit is computed before the bidding phase. Thus, at first sight, their protocol provides fund binding without revealing the bidding price. However, it is not guaranteed that a bidder who computes a commitment on a bidding price has funds more than or equal to the bidding price. Thus, it is difficult to provide a fund binding property in sealed-bid auction protocols.

1.3 Our Contribution

In this paper, we propose a sealed-bid auction protocol providing the fund binding property. Our protocol simultaneously provides the following:

- Price Hiding: It guarantees that nobody can know bidding prices of other bidders until the revealing phase. Especially, the maximum bidding price is also hidden.
- Fund Binding: It guarantees that a bidder has funds more than or equal to the bidding price, and the funds are forcibly withdrawn when the bidder wins.

We emphasize that sealed-bid auction protocols with the simple deposit method reveal the maximum bidding price although they provide fund binding. Our goal in this work is to provide price hiding and fund binding simultaneously.

Our Technique. To achieve our goal, we pay attention to the fact that an address of the smart contract can be calculated before deploying the contract, and bidders issue a one-time address by themselves. Then, each bidder sends a transaction to own one-time address. Here, we assume that a usual Ethereum transaction and a bidding transaction to a one-time address are indistinguishable, which is a reasonable assumption and allows us to hide the maximum bidding price. For example, let the bidding phase be from May 25 to 31, 2023 (one week). Then there are 109,516 possible one-time addresses. We discuss how much bidding transactions are hidden in Sect. 5.1. We remark that, by observing transactions during the bidding phase, the maximum bidding price could be guessed because it is less than or equal to the trading price during this phase. Therefore, we also assume that the actual maximum bidding price is not revealed from transactions during the bidding phase. We discuss whether this assumption is reasonable by observing actual transactions in Sect. 5.2.

In addition to employing one-time addresses, each bidder is required to prove that enough balance is preserved on the one-time address without revealing the balance and the address. We employ DECO (DECentralized Oracle) [20] that allows a bidder to prove the statement above in a zero-knowledge manner. In particular, it allows bidders to prove that the balance is preserved on a one-time address.

We implement our protocol and compare it to the simple deposit method. We show that the two protocols require almost similar fee to run the protocol (the additional fee was 5.28 USD calculated by the price on June 1, 2023).

Out of Scope for Our Work. Our protocol hides bidding prices during the bidding phase. Each bidder sends a trapdoor (a decommitment of the underlying commitment scheme), and then the auction smart contract reveals all bidding prices and decides the winner. Although some sealed-bid auction protocols consider how to decide the winner without revealing other bidding prices, we do not consider to hide it after revealing since our goal in this work is to provide price hiding and fund binding simultaneously during the bidding phase. However, from the viewpoint of privacy, one merit of such a sealed-bid auction is that losing bids are kept secret when the auctioneer declares the winner and the winning bid only. This privacy-preserving manner is effective to protect revealing unnecessary information to run an auction. We expect that our protocol can be adopted/combined with other sealed-bid auction protocol, and price hiding during the revealing phase is left as a future work.

In our protocol, bidders open their bid commitments in an undefined order. So, one may think that bidders do not want to open the commitments if another bidder has already opened a higher bid. Actually, the information about how much they bid in this auction might be useful to other bidders in subsequent related auctions. For example, Naor et al. [21] hide all unsuccessful bids for this reason. In our protocol, even such bidders have an incentive to send decommitments since they have no way to withdraw the funds of the one-time addresses. Thus, we assume that all bidders send their decommitments honestly, and do not consider a case when bidders do not send their decommitments.

1.4 Concurrent and Independent Work

1.4.1 FAST

In addition to sealed-bid auction protocols providing small or simple deposit methods [5]–[18], recently, David et al. [22] proposed FAST (Fair Auctions via Secret Transactions) with

the same motivation as ours. That is, the deposit reveals information about the bid, and thus it should be hidden. They introduced secret deposits based on confidential transaction [23]. They defined deposit committee members (which are different from bidders) and a trapdoor, which can be used to reveal the value of deposits, that is distributed among m deposit committee members using a proactive secret sharing scheme [24]. There are ℓ rounds in total where ℓ is the bit length of the bids. For each round, the anonymous veto protocol [25] is run, and if a party is found to be cheating, a smart contract automatically redistributes cheaters' deposits among the honest parties, which creates the incentive for parties to behave honestly. The main difference from our protocol is complexity because FAST requires each bidder to communicate with other bidders. That is $O(\ell n)$ computations are required where n is the number of bidders. Conversely, our protocol does not require any interaction between bidders (a bidder is required to communicate with the oracle for running DECO, and later the bidder communicates with the auction smart contract only). Moreover, FAST needs to assume that the number of corrupted deposit committee members is less than $m/2-2$, whereas we do not require such a limitation. From our perspective, FAST employs a cryptographic approach to hide and bind the deposit whereas we employ another approach where a transaction of the usual transfer of Ethereum and a transaction for sending a bidding price to a one-time address are indistinguishable. This drastically reduces the complexity of the protocol for hiding and binding deposits.

1.4.2 Anonymous Vickrey Auctions on Chain

The anonymous vickrey auction called vickrey.xyz has been launched [26], [27]. As in our system, the auction leverages uninitialized CREATE2 addresses. Their main motivation besides sealed-bid is to provide anonymity, or private participation because merely knowing who is participating allows you to collude off-chain to lower the final price. Bidders send money to an uninitialized CREATE2 address instead of sending shielded money to a contract. Both our system and vickrey.xyz assume that other users' usual transactions and transactions to the one-time address are indistinguishable, i.e., assuming that they just look like EOA (account) transfers (See Sect. 5.1). The main difference is that our system employs DECO to prove that enough balance is preserved on the one-time address without revealing the balance and the address. This prevents bidders to change their bidding amount after seeing other bidding amount in the revealing phase. On the other hand, the vickrey.xyz auction employs the snapshotted blockhash which encodes the root of the Ethereum storage trie at that point in time. Each bidder sends a Merkle Patricia tree proof in Ethereum of the balance of their CREATE2 address during the snapshotted blockhash.

2. Preliminaries

2.1 Ethereum and Smart Contracts

Ethereum [28] is a platform for decentralized applications based on blockchain technology. It allows users to create applications that run on the Ethereum Virtual Machine (EVM), called smart contracts. The results of smart contract execution are shared and agreed upon among all Ethereum nodes, making it difficult to tamper.

ETH is a native token on Ethereum and is used to pay transaction fees. The transaction fee is calculated by multiplying the amount of used gas by gas price. The amount of gas used by the transaction varies depending on the operations. Gas price also fluctuates according to the demand of the Ethereum network. A web service called Etherscan[†] allows users to check the balance of their addresses and transaction information.

2.2 Oracles

Smart contracts cannot directly retrieve data outside the blockchain, such as ETH price, stock prices, weather, or election results. Therefore, if a smart contract requires the use of these data, then someone needs to input them into the smart contract from outside. An entity that inputs data from outside the blockchain to a smart contract is called an oracle.

Currently, Chainlink [29] is the protocol with the largest share among the oracles. Chainlink is a decentralized oracle network that aggregates data obtained by multiple oracle nodes to prevent oracle fraud. In addition, because Chainlink nodes are financially penalized if they deviate from the Chainlink protocol, they are trustworthy in terms of their behavior according to the protocol. That is, in a cryptographic manner, oracles are modeled as honest-but curious entities where they follow the protocol procedure but they may try to extract information through the execution of the protocol.

2.3 DECO

In this section, we introduce DECO [20] which allows a user to prove the possession of data in a zero-knowledge manner wherein the data come from a particular website via TLS (Transport Layer Security). DECO contains three entities: a data source (e.g., a particular website) $\mathcal{S}$, a prover $\mathcal{P}$, and a verifier $\mathcal{V}$. To access $\mathcal{S}$, some secret value is required such as salt. Then, $\mathcal{P}$ proves that the data come from $\mathcal{S}$ without showing the data, and $\mathcal{V}$ checks the validity of the proof. The flow of DECO is explained as follows. $\mathcal{S}$, $\mathcal{P}$, and $\mathcal{V}$ run a three-party handshake protocol. Then, $\mathcal{S}$ obtains an encryption key and a key for MAC (Message Authentication Code), denoted by k_{MAC}, as in the TLS protocol. That is, DECO does not require any server-side modifications or cooperation, and $\mathcal{S}$ follows the unmodified TLS protocol. Via the

[†]https://etherscan.io/

three-party handshake protocol, $\mathcal{S}$ and $\mathcal{P}$ share the encryption key, and $\mathcal{P}$ and $\mathcal{V}$ share k_{MAC} in a secret sharing manner ($\mathcal{P}$ obtains k_P and $\mathcal{V}$ obtains k_V where $k_{\mathsf{MAC}} = k_P + k_V$). In this setting, with the help of $\mathcal{V}$, $\mathcal{P}$ can make a query sent to $\mathcal{S}$ via TLS. More concretely, $\mathcal{P}$ and $\mathcal{V}$ run a MPC (Multi-Party Computation) that takes k_P and k_V as input, respectively. We remark that $\mathcal{V}$ cannot observe the content of the encrypted query because $\mathcal{V}$ does not have the encryption key. $\mathcal{P}$ generates a commitment on the query and the response from $\mathcal{S}$, and sends the commitment to $\mathcal{V}$. Then, $\mathcal{V}$ sends k_V to $\mathcal{P}$, and $\mathcal{P}$ recovers k_{MAC}, and checks the MAC on the response from $\mathcal{S}$. Finally, $\mathcal{P}$ generates a zero-knowledge proof for the query and sends it to $\mathcal{V}$.

The DECO website[†] states that *Chainlink* [29] *plans to perform an initial PoC of DECO, with a focus on decentralized finance applications such as Mixicles*. Then, a user would be able to prove the validity of the data containing their own personal information to a Chainlink node. In the proposed scheme, we employ DECO to prove that a bidder has funds that are equal to the bidding price to an oracle. Then the bidder is regarded as $\mathcal{P}$, the oracle is regarded as $\mathcal{V}$, and Etherscan is regarded as $\mathcal{S}$ in our protocol.

2.4 Hash-Based Commitment Scheme

In this section we introduce a hash-based commitment scheme. A commitment scheme $(\mathsf{Commit}, \mathsf{ComOpen})$ is defined as follows. The Commit algorithm takes as input a message M, and outputs a commitment com and a decommitment dec. The $\mathsf{ComOpen}$ algorithm takes as input com, dec, and M, and outputs 0 (reject) or 1 (accept). A commitment scheme is required to provide hiding where no adversary can obtain information of M from com, and is required to provide binding where no adversary can produce dec and dec' where $\mathsf{ComOpen}(\mathsf{com}, \mathsf{dec}, M) = 1$, $\mathsf{ComOpen}(\mathsf{com}, \mathsf{dec}', M') = 1$, and $M \neq M'$ holds.

Next, we select the commitment scheme to be employed in our implementation. Because the smart contract runs the $\mathsf{ComOpen}$ algorithm, it is desirable to select an efficient scheme to reduce the gas price. Therefore, we select a hash-based commitment scheme because it is more efficient than a scheme based on an algebraic structure such as the Pedersen commitment scheme [30]. Let $\mathsf{Hash} : \{0,1\}^* \to \{0,1\}^k$ where k is a security parameter. In the Commit algorithm, choose a random $R \xleftarrow{\$} \{0,1\}^k$, compute $\mathsf{com} = \mathsf{Hash}(M||R)$, and output com and $\mathsf{dec} = R$. The $\mathsf{ComOpen}$ algorithm outputs 1 if $\mathsf{com} = \mathsf{Hash}(M||\mathsf{dec})$ holds, and 0, otherwise. This hash-based scheme provides both hiding and binding if Hash is modeled as a random oracle [31], [32].

The commitment scheme provides either statistical hiding or statistical binding according to the range size [31], [32]. Specifically, for $\mathsf{Hash} : \{0,1\}^{2n} \to \{0,1\}^{\ell(n)}$, the scheme provides statistical hiding if $\ell(n) = n/8$ (Lemma 9 in [31] and Lemma 17 in [32]), and statistical binding if $\ell(n) = 4n$ (Lemma 8 in [31]), respectively. In our implementation, we employ SHA256 as the underlying hash function because SHA256 has been prepared in a smart contract environment as a pre-compiled function, and can be efficiently run with a low gas usage. For our usage, the input length is 768 bits (one-time address [256 bits]+a bidding price [256 bits]+a random R [256 bits]). Thus, the hash function is defined as $\mathsf{Hash} : \{0,1\}^{768} \to \{0,1\}^{256}$. Because our usage does not match the above cases, the commitment scheme provides computational hiding and computational binding.

[†]https://www.deco.works/

2.5 Zero-Knowledge Proofs

Zero-knowledge proofs/arguments for a committed value are required in DECO. In this section, we show the feasibility of the zero-knowledge argument for the hash-based commitment scheme, that is, a proof of M and R satisfying $\mathsf{com} = \mathsf{Hash}(M||R)$, especially when SHA256 is chosen as the underlying hash function. First, we generate an arithmetic circuit C to compute SHA256 that can be obtained by the circuit generator [33] with a C program to compute SHA256. Next, we can produce a desirable argument system by employing zk-SNARK (zero-knowledge Succinct Non-interactive ARgument of Knowledge) for arithmetic circuits [34]. Ben-Sasson et al. [35] provided a hand-optimized arithmetic circuit for the SHA256 compression function and then reduced the number of gates compared to that generated by the circuit generator [33]. Although we may also be able to reduce the number of gates by providing a hand-optimized arithmetic circuit for SHA256, here we only show the feasibility because zero-knowledge proofs/arguments are run in an off-chain and we focus on how to reduce the gas price for running smart contracts.

3. Proposed Sealed-Bid Auction with Fund Binding

In this section, we introduce the proposed protocol. Figure 1 shows a sequence diagram of the proposed protocol. We employ a smart contract as an auctioneer that collects bid amounts in a hidden manner and decides the winner of the auction. We refer to the smart contract as the auction contract. The proposed protocol does not require a deposit because it reveals the maximum bidding price; thus, bidders do not directly send their own bidding price to the auction contract. The bidder sends funds to a fund-binding contract deployed from the auction contract. Here, Ethereum offers CREATE2 opcode for deploying smart contracts. With the CREATE2 opcode, the bidder can compute the address of a fund-binding contract before its deployment, based on some information (See Sect. 3.2.1), though usually the address of a smart contract is generated when the contract is deployed. We call the address of a fund-binding contract a one-time address. In the proposed protocol, one-time addresses are used before deploying the fund-binding contract. We assume that other users' usual transactions and transactions to the one-time address are indistinguishable. Then, a one-time address and its balance will be kept secret. Next, the bidder enters

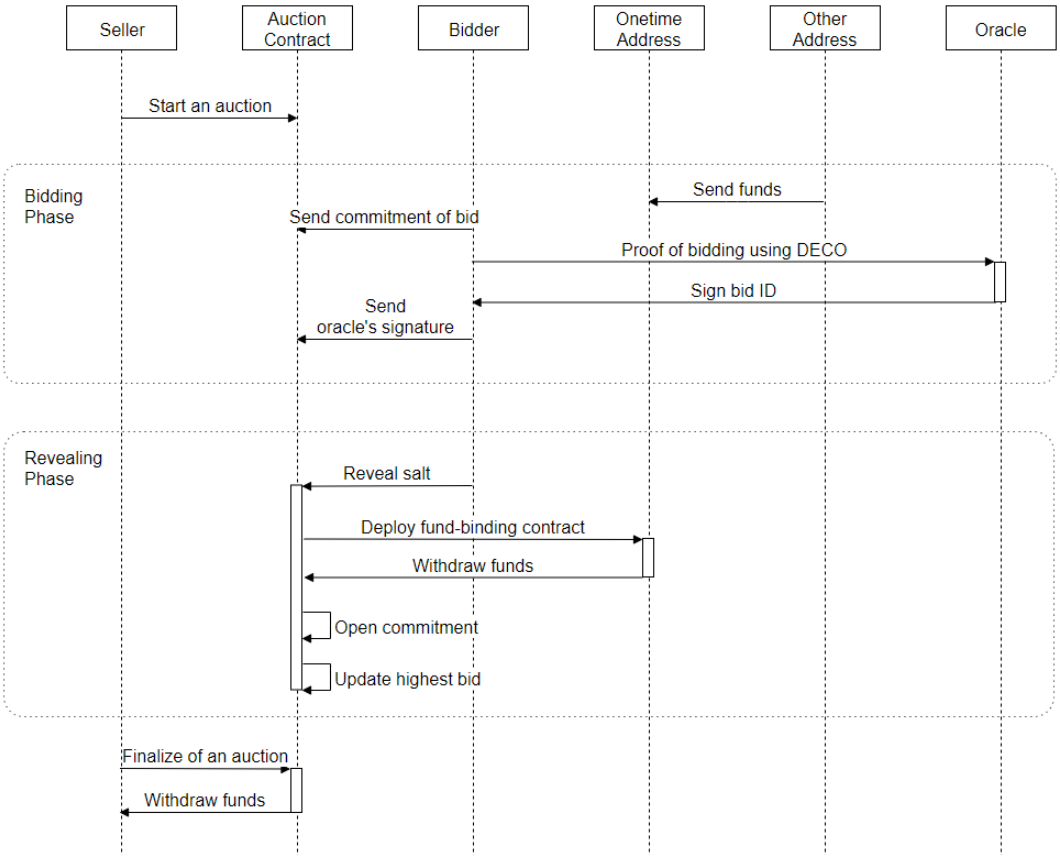

Fig. 1 The Sequence Diagram of the Proposed Protocol

a commitment of the bidding price into the auction contract, and uses DECO [20] to prove that the balance of the one-time address is the same as the bidding price. We emphasize that DECO is run in an off-chain among Etherscan, a bidder, and an oracle node, and the auction contract just checks the validity of signatures sent from the oracle. Finally, the auction contract deploys fund-binding contracts whose addresses are bidder's one-time addresses. Then, the auction contract withdraws the funds from one-time addresses, and determines the winning bidder by the opening result of commitments. As mentioned before, we assume that all bidders send their decommitments honestly because they have no way to withdraw the funds of the one-time addresses.

3.1 Starting an Auction

A seller who wants to start an auction sends auction information such as the duration of the bidding phase and the revealing phase to the smart contract. In this paper, we refer to this smart contract as an auction contract. At this time, a unique number is assigned to each auction managed by the auction contract as an "auction ID". In our implementation, the auction smart contract incremented an "auction ID" by 1 sequentially as each auction started. Since the auction contract starts each auction serially, we can assume that the ID is unique.

3.2 Bidding Phase

A bidder, who wants to bid on the auction, performs the following steps during the bidding phase.

3.2.1 Issuing One-Time Address

A one-time address is a disposable address that is issued for a certain purpose. For example, in our proposed protocol, a one-time address is used to bind the funds. If there is a trusted third party, the third party can issue a one-time address. However, assuming a trusted party is a much stronger assumption, and it would be better to avoid introducing it in cryptographic protocols as much as possible. To issue a one-time address without any trusted third party, we observe that an address of a smart contract can be calculated in advance, that is, before the smart contract is deployed.[†] Specifically, we can precompute the address from the following arguments using CREATE2 opcode before deploying fund-binding contacts.

- A salt
- A bytecode of the fund-binding contact

[†]This fact has been utilized in protocols such as Argent [36] on Ethereum.

- An address of the auction contract

We remark that, in our system, first the auction contract is deployed and the source code is published on platforms like Etherscan for providing transparency. The auction contract will deploy the fund-binding contract to withdraw the bidding prices and thus the auction contract incorporates the source code of the fund-binding contract. Thus, when one compiles the source code, then one obtains the bytecode of the fund-binding contract (in addition to the bytecode of the auction contract). In this paper, the salt is defined as a concatenation of the auction ID, the bidder's address, which is used to deposit to the one-time address, and a random value chosen by the bidder. Thus, the bidder is the only one who can know the one-time address until the salt is revealed. Note that only an auction contract can deploy a fund-binding contract to a one-time address and can withdraw funds from a one-time address.

3.2.2 Bidding

A bidder sends funds to the one-time address issued. A bidder is required to send the bidding amount to a one-time address from an address that the bidder does not use to send bidding information to the auction contract. If the bidder uses the same address, it helps to distinguish whether an address is related to the auction, and our assumption does not hold.

3.2.3 Proving of Bidding Using DECO

Each bidder is required to prove that the bidding price is preserved at a one-time address to provide the fund-binding property. In the proposed protocol, the bidder proves to the oracle node that the balance of the one-time address displayed by Etherscan is equal to the bidding price using DECO[†]. Let Etherscan be $\mathcal{S}$, the bidder be $\mathcal{P}$, the oracle be $\mathcal{V}$, and $(\mathsf{vk}_O, \mathsf{sigk}_O)$ be the oracle's verification key and signing key. The secret information for the bidder's access to Etherscan is the one-time address θ_p. First, the bidder sets P as the bidding price, calculates $(\mathsf{com}_P, \mathsf{dec}_P) = \mathsf{Commit}(P)$, and sends com_P to the auction contract. The auction contract issues a unique number as the bid ID that corresponds to the commitment. Next, the Etherscan, a bidder, and the oracle execute DECO. The bidder accesses the page of Etherscan that describes the balance value of the one-time address, and obtains the balance value P^*. Note that the oracle does not know which page in the Etherscan corresponds to the bidder because the query is hidden. This means that the oracle cannot learn the one-time address of the bidder. The bidder then provides the oracle with a zero-knowledge proof of knowledge that P^* was obtained from θ_p, plus $\mathsf{ZK\text{-}PoK}\{P^* : \mathsf{com}_P = \mathsf{Commit}(P^*)\}$. This shows that the committed value P stored in the auction contract before the execution of DECO is P^*. If the proof is valid, the oracle generates a signature on the bid ID using sigk_O as a credential. Finally, the bidder sends the oracle's signature to the auction contract, and if the signature is valid, the auction contract accepts the bidding of com_P specified by the bid ID.

[†]We modified an application of binary options using DECO [20].

3.3 Revealing Phase

In the revealing phase, the bidder reveals the bidding price according to the following procedure.

3.3.1 Revealing Salt and Decommitment

The bidder sends the auction ID, the bidding price, the salt, and the decommitment to the auction contract. Based on the salt, the auction contract deploys a fund-binding contract whose address has been generated by the bidder using CREATE2 opcode. We remark that, a fund-binding contract is allowed to transfer funds to the auction contract in our implementation, and thus each bidder cannot withdraw the balance without the auction contract. After the funds is transferred from the fund-binding contract to the auction contract, the auction contract runs the ComOpen algorithm (described in Sect. 2.4) against commitments whose bid ID have been verified. The auction contract obtains a bidding price, and if it is greater than the current highest bid, then the auction contract updates the current highest bid. Otherwise, if it is less than the current highest bid, then the auction contact refunds the funds. We assume that there is one highest bid and do not consider the case that there are many highest bids because it depends on a rule of the auction. We remark that the auction contract only holds the funds of the highest bidder in the first-price auction. However, our system can handle the second-price auction easily where the auction contract preserves the second-highest bid when it updates the current highest bid.

Since funds of the one-time contract have been verified using DECO at the bidding phase, funds will never be less than the committed amount. One may wonder the freshness of the data source. Actually, a bidder can send ETH to the one-time address even after proving of the bidding price using DECO. Even then, we assume that the auction contract selects the committed bidding price. On the other hand, if the auction contract employs the funds withdrawn from the one-time contract, then this rule gives the bidder a room for increasing the funds after the bidding phase.

We remark that a bidder may bid at multiple prices and open only the lowest bid that wins the auction during the revealing phase. Since deploying the fund-binding contract is the only way to withdraw funds from a one-time address, only the auction contract can withdraw the funds. This means that any funds linked to unrevealed bids stay locked, and the bidder needs to open all bids to withdraw their own funds. Thus the highest bid of the multiple bids is selected as the bidder's bid. Thus, there is no benefit that bidders send multiple bids in our system.

3.4 Finalization of an Auction

The seller finalizes the auction by using the auction contract's finalize function. The auction contract determines the winning bidder on the highest bids in the revealing phase, and the seller receives the funds. As a special case, if the auctioned item is an NFT, the auction contract directly transfers the NFT to the winning bidder right after the auction contract withdraws the bidding price[†].

3.5 Security Discussion

We assume that all parties follow the protocol description (i.e., semi-honest parties), and we do not introduce any trusted third party in our protocol. Then, our protocol provides price hiding if the underlying commitment scheme provides hiding, the DECO protocol provides zero-knowledge for hiding balances, and our two assumptions hold: (1) bidding transactions to one-time addresses and usual transactions are indistinguishable, and (2) the maximum bidding price is not revealed for transactions during the bidding phase. We discuss how much these assumptions are reasonable in Sects. 5.1 and 5.2.

Moreover, our protocol provides fund binding if the underlying commitment scheme provides binding, the underlying signature scheme (run by the oracle) is existentially unforgeable under chosen-message attack (EUF-CMA), and the DECO protocol provides soundness for proving balances. Here, soundness means that no proof for false statements (e.g., a balance is less than and not equal to the bidding price) is accepted.

4. Implementation

4.1 Gas Prices for Running the Proposed Protocol

In this section, we discuss the fees of operating auctions. As mentioned in Sect. 2.1, we need to pay a fee called "gas" to operate a smart contract. The gas and USD-denominated fees for bidders' processing in the auction of the proposed protocol are shown in Table 1. This paper assumes that the gas price is 39 gwei (median gas price on June 1, 2023[††]) and 1 ETH is 1900 USD (Approximate price on June 1, 2023[†††]). Normally, a bidder will perform each operation shown in Table 1 one by one at a time, so the bidder will consume about 19.65 USD worth of ETH in a series of operations. The amount of gas consumed varies depending on the length of the salt. However it is considered within the margin of error due to the high volatility of gas price and ETH prices.

[†]Unlike to NFT, if a smart contact does not treat an auctioned item, then there is room for argument on whether exchange between the funds and the auctioned item is honestly run. Thus, it seems reasonable to employ fair exchange protocols, especially smart contract-based protocols [37]–[40] in addition to our protocol.

[††]https://dune.xyz/queries/4294/11099

[†††]https://www.coingecko.com/en/coins/ethereum

Table 1 Cost of Gas for Bidder's Operation (Proposed Protocol)

Operation	Used Gas	Fee
Sending funds	21,000	1.56 USD
Committing bid	68,903	5.10 USD
Proving bid	52,755	3.90 USD
Revealing bid	122,546	9.08 USD

Table 2 Cost of Gas for Seller's Operation (Proposed Protocol)

Operation	Used Gas	Fee
Starting auction	166,510	12.33 USD
Finalization auction	40,312	2.99 USD

Table 3 Cost of Gas for Bidder's Operation (Simple Deposit Method)

Operation	Used Gas	Fee
Committing bid	110,928	8.22 USD
Revealing bid	83,119	6.16 USD

Table 4 Cost of Gas for Bidder's Operation (Open-Bid)

Operation	Used gas	Fee
Bidding	71,137	5.27 USD

In addition, the fee for the series of operations is independent of the number of bidders.

Next, the gas and fees for the seller's operation are shown in Table 2. The gas is approximately 15.32 USD, although it varies depending on the length of the information entered into the smart contract at the start of the auction. For example, assuming an NFT auction, in addition to the gas shown in Table 2, additional gas required to send the NFT is also necessary.

4.2 Comparison

First, we consider the simple deposit method. As described in Sect. 1.1, this method is problematic in that the maximum bidding price is exposed. Therefore, we can examine the fee for hiding the maximum bidding price by comparing the proposed protocol with the simple deposit method. Because the fee depends on the amount of data stored in the smart contract and the amount of computation, we implemented the simple deposit method on a smart contract implemented in the proposed protocol without changing the structure of the smart contract as much as possible. Table 3 shows the amount of used gas and fee for operations performed by the bidder.
As described in Sect. 4.1, the fee for the proposed protocol is about 19.65 USD (265,204 gas). Thus, the increased fee for hiding the maximum bidding price by the proposed method is about 14.37 USD (194,047 gas), and is about 37% more expensive than that of the simple deposit method. Note that we use 1.56 USD as a normal transfer of Ethereum costs in this paper.

Next, we compare the fees for the operation of the open-bid auction. As in the deposit method, we implemented the open-bid smart contract without changing the smart contract structure implemented in the proposed protocol as much as possible. Table 4 shows the gas used and fee for operations

performed by the bidder. The fee for bidding is approximately 5.27 USD, which is lower than the fee for the sealed-bid auction. However, multiple bids are generally placed in open-bid auctions. As a result, the sealed bid auction, which requires only one operation, may lower the fees.

5. Analyses of Assumptions

5.1 How Much Bidding Transactions are Hidden

In this section, we consider how much bidding transactions are hidden. Because the one-time address is issued by a 256-bit salt which is known only by the bidder, it is difficult to guess the one-time address by others. In addition, a bidding transaction to a one-time address and usual transactions in Ethereum are indistinguishable because a one-time contract has not been deployed yet. If malicious bidders want to find the bidding transaction, they can infer the one-time address from the following properties.

1. Addresses that do not transfer ETH to other addresses during the bidding phase.
2. Addresses that received ETH for the first time during the bidding phase.

The first property is clear from the fact that the one-time contract has not been deployed yet in the bidding phase, and only the one-time contract, which is deployed in the revealing phase, can withdraw the funds of the one-time address. The second property is also clear from the fact that the one-time address has not been used yet before the bidding phase. If there are only a few candidate addresses, malicious bidders may be able to guess the bidding price. To extract addresses that satisfy the two conditions, we used Google Cloud BigQuery, which has access to all data of Ethereum transactions. The number of addresses with two conditions is shown in Table 5.

We consider cases of three days (May 29 to 31, 2023), one week (May 25 to 31, 2023), and two weeks (May 18 to 31, 2023) as the bidding phase period. Assuming that the bidding price in the auction is 0.1 to 0.5 ETH, Table 5 shows that there are 4,458 possible addresses in the three days case, 10,229 addresses in the one week case, and 18,668 addresses in the two weeks case. Because the median price of the NFTs is approximately $300 (0.16 ETH as of June 1, 2023), as described in Sect. 1.1, this assumption is reasonable. Figure 2 shows the number of addresses with balance

Table 5 The Number of Addresses Satisfying the Two Conditions

	Three Days	One Week	Two Weeks
0~0.1ETH	35,722	89,067	173,348
0.1~0.5ETH	4,458	10,229	18,668
0.5~1ETH	1,286	2,944	5,368
1~10ETH	2,361	5,558	10,505
10~50ETH	573	1,193	2,196
50~100ETH	94	198	347
100~	168	329	558

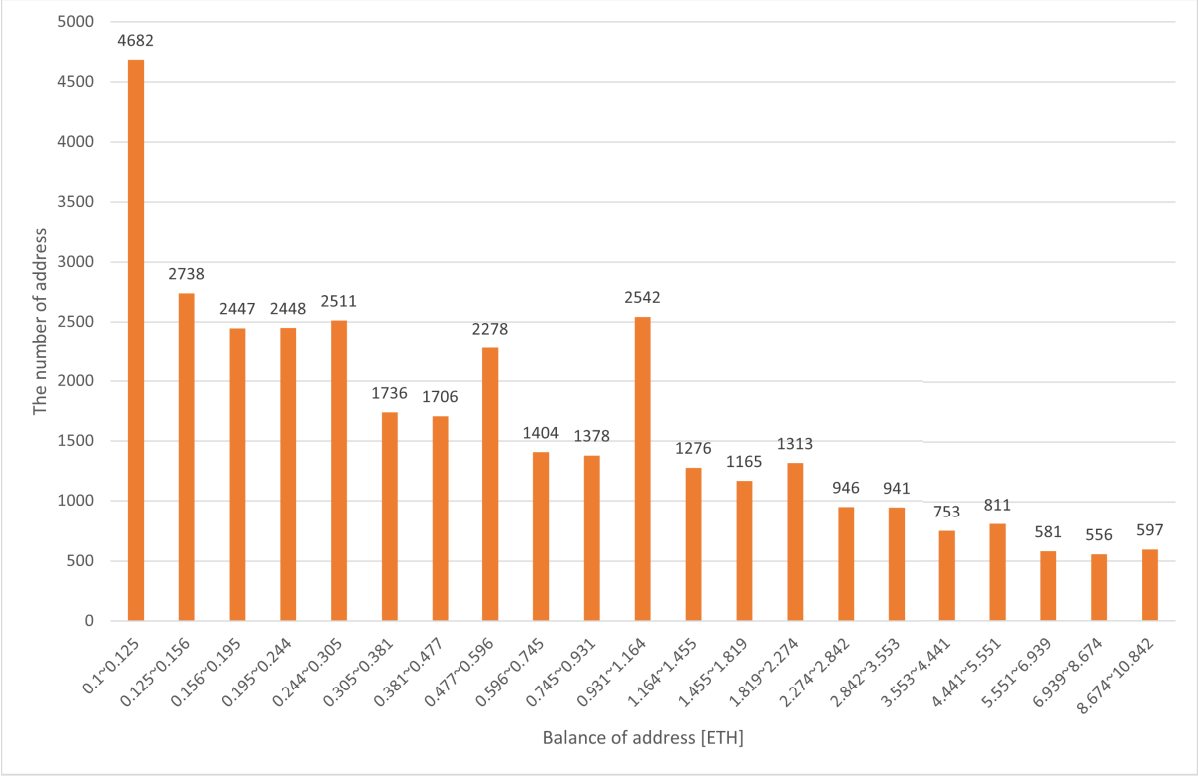

Fig. 2 The Number of Addresses Meeting the Two Conditions (0.1 ETH to 10 ETH)

between 0.1 ETH and 10 ETH during two weeks (May 18 to 31, 2023). We remark that in many auctions, bidders know the approximate market price of an auctioned item, and can expect the range of bidding price, i.e., it does not become much higher/lower than the market price. For example, when an item is sold 0.125 ETH market price, then it seems reasonable that the winning bid is between 0.1 ETH and 0.15 ETH. Thus, we described the width of the graph in 1.25x intervals (between 0.1 and 0.125, between 0.125 and 0.156, and so on) that seems appropriately treat the above situation. Because all these addresses can be regarded as one-time addresses, it is clear that a longer bidding phase period is effective in hiding more one-time addresses.

5.2 How Much the Maximum Bidding Price is Hidden

In the proposed protocol, the maximum bidding price is less than or equal to the maximum balance of all addresses. However, many transactions for a relatively high ETH are transferred every day on Ethereum. For example, when we assume the bidding phase to be two weeks (May 18 to 31, 2023), the maximum balance of all addresses is 29,565 ETH. It is potentially difficult to determine which transaction is for the maximum bidding price. On the other hand, in the case of a sealed-bid auction with the simple deposit method, the maximum bidding price is revealed.

6. Conclusion

In this paper, we proposed a sealed-bid action protocol providing the fund binding property. We introduced transactions for one-time addresses and employed DECO protocol. We implemented our protocol and compared the protocol with the simple deposit method. Although we have discussed how much bidding transactions are hidden, there is a room for argument on this point. For example, bidders may be able to estimate a rough maximum bidding price from transactions that increased after the bidding phase. Alternatively, some statistical analyses may distinguish or identify transactions for one-time addresses or reveal the maximum bidding price. Thus, the statistical analysis of these transactions could be conducted in a future work. We expect that our blockchain-oriented technique could be a stepping stone for hiding transactions.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers JP21K11897 and JP22H03588.

References

[1] K. Chin, K. Emura, K. Omote, and S. Sato, "A sealed bid auction with binding that can prevent leakage of budget information (in Japanese)," Computer Security Symposium, pp.4D1–3, October 26, 2021.
[2] K. Chin, K. Emura, K. Omote, and S. Sato, "A sealed-bid auction with fund binding: Preventing maximum bidding price leakage," Blockchain, pp.398–405, IEEE, 2022.
[3] V. Krishna, Auction Theory, Academic Press, 2002.
[4] S. Bag, F. Hao, S.F. Shahandashti, and I.G. Ray, "SEAL: sealed-bid auction without auctioneers," IEEE Trans. Information Forensics and Security, vol.15, pp.2042–2052, 2020.
[5] H.S. Galal and A.M. Youssef, "Trustee: Full privacy preserving vickrey auction on top of ethereum," Financial Cryptography and Data Security, pp.190–207, 2019.
[6] H. Li and W. Xue, "A blockchain-based sealed-bid e-auction scheme with smart contract and zero-knowledge proof," Security and Communication Networks, vol.2021, pp.5523394:1–5523394:10, 2021.
[7] S. Wu, Y. Chen, Q. Wang, M. Li, C. Wang, and X. Luo, "CReam: A smart contract enabled collusion-resistant e-auction," IEEE Trans. Inf. Forensics Secur., vol.14, no.7, pp.1687–1701, 2019.
[8] B. Chen, X. Li, T. Xiang, and P. Wang, "SBRAC: Blockchain-based sealed-bid auction with bidding price privacy and public verifiability," J. Information Security and Applications, vol.65, p.103082, 2022.
[9] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts," IEEE S&P, pp.839–858, 2016.
[10] M. Król, A. Sonnino, A.G. Tasiopoulos, I. Psaras, and E. Rivière, "PASTRAMI: privacy-preserving, auditable, scalable & trustworthy auctions for multiple items," ACM Middleware, pp.296–310, 2020.
[11] M. Kadadha, R. Mizouni, S. Singh, H. Otrok, and A. Ouali, "*ABCrowd*: An *A*uction mechanism on *B*lockchain for spatial *Crowd*sourcing," IEEE Access, vol.8, pp.12745–12757, 2020.
[12] G. Sharma, D. Verstraeten, V. Saraswat, J.-M. Dricot, and O. Markowitch, "Anonymous fair auction on blockchain," IFIP NTMS, pp.1–5, IEEE, 2021.
[13] I. Vakilinia, S. Badsha, and S. Sengupta, "Crowdfunding the insurance of a cyber-product using blockchain," IEEE UEMCON, pp.964–970, 2018.
[14] A. Sonnino, M. Król, A.G. Tasiopoulos, and I. Psaras, "AStERISK: Auction-based shared economy resolution system for blockchain," CoRR, vol.abs/1901.07824, 2019.
[15] B. Al-Sada, N. Lasla, and M. Abdallah, "Secure scalable blockchain for sealed-bid auction in energy trading," IEEE ICBC, pp.1–3, 2021.
[16] G. Sharma, D. Verstraeten, V. Saraswat, J.-M. Dricot, and O. Markowitch, "Anonymous sealed-bid auction on Ethereum," Electronics, vol.10, no.19, 2021.
[17] P.-C. Hsu and A. Miyaji, "Verifiable M+1st-price auction without manager," IEEE DSC, pp.1–8, 2021.
[18] T. Constantinides and J. Cartlidge, "Block auction: A general blockchain protocol for privacy-preserving and verifiable periodic double auctions," IEEE Blockchain, pp.513–520, 2021.
[19] J. Ma, B. Qi, and K. Lv, "Fully private auctions for the highest bid," ACM TUR-C, pp.64:1–64:6, 2019.
[20] F. Zhang, D. Maram, H. Malvai, S. Goldfeder, and A. Juels, "DECO: liberating web data using decentralized oracles for TLS," ACM CCS, pp.1919–1938, 2020.
[21] M. Naor, B. Pinkas, and R. Sumner, "Privacy preserving auctions and mechanism design," ACM EC, pp.129–139, 1999.
[22] B. David, L. Gentile, and M. Pourpouneh, "FAST: Fair auctions via secret transactions," ACNS, pp.727–747, 2022.
[23] G. Maxwell, "Confidential transactions," 2016. https://web.archive.org/web/20200502151159/https://people.xiph.org/~greg/confidential_values.txt.
[24] I. Cascudo and B. David, "ALBATROSS: publicly attestable batched randomness based on secret sharing," ASIACRYPT, pp.311–341, 2020.
[25] F. Hao and P. Zieliński, "A 2-round anonymous veto protocol," Security Protocols, pp.202–211, 2006.
[26] ETHGrobal, "Anonymous vickrey auctions on chain," 2023. https://ethglobal.com/showcase/anonymous-vickrey-auctions-on-chain-igh5e.
[27] A. Gupta, "Cheap, anonymous vickrey auctions on-chain," Nov 15,

2022. https://blog.aayushg.com/posts/vickrey/.
[28] V. Buterin, "A next-generation smart contract and decentralized application platform," 2015.
[29] B. Lorenz, C. Cachin, B. Chan, A. Coventry, S. Ellis, A. Juels, F. Koushanfar, A. Miller, B. Magauran, D. Moroz, S. Nazarov, A. Topliceanu, F. Tramer, and F. Zhang, "Chainlink 2.0: Next steps in the evolution of decentralized oracle networks," 2021. https://research.chain.link/whitepaper-v2.pdf.
[30] T.P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," CRYPTO, pp.129–140, 1991.
[31] R. Pass, "Alternative variants of zero-knowledge proofs," tech. rep., 2004.
[32] D. Unruh, "Computationally binding quantum commitments," EUROCRYPT, pp.497–527, 2016.
[33] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," IEEE S&P, pp.238–252, 2013.
[34] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "Succinct non-interactive zero knowledge for a von neumann architecture," USENIX Security Symposium, pp.781–796, 2014.
[35] E. Ben-Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," IEEE Symposium on Security and Privacy, pp.459–474, 2014.
[36] argentlabs, "argent-contracts," 2021. https://github.com/argentlabs/argent-contracts.
[37] S. Avizheh, P. Haffey, and R. Safavi-Naini, "Privacy-preserving FairSwap: Fairness and privacy interplay," Proceedings on Privacy Enhancing Technologies, vol.2022, no.1, pp.417–439, 2022.
[38] L. Eckey, S. Faust, and B. Schlosser, "OptiSwap: Fast optimistic fair exchange," ACM ASIACCS, pp.543–557, 2020.
[39] S. Dziembowski, L. Eckey, and S. Faust, "FairSwap: How to fairly exchange digital goods," ACM CCS, pp.967–984, 2018.
[40] C. Lin, D. He, X. Huang, and K.-K.R. Choo, "OBFP: optimized blockchain-based fair payment for outsourcing computations in cloud computing," IEEE Trans. Inf. Forensics Secur., vol.16, pp.3241–3253, 2021.

Kota Chin received his B.E. degrees in information science from University of Tsukuba, Japan, in 2021. He is currently enrolled in a master's degree program at the University of Tsukuba.

Keita Emura received M.E. degrees from Kanazawa University in 2004. He was with Fujitsu Hokuriku Systems Ltd., from 2004 to 2006. He received the Ph.D. degree in information science from the Japan Advanced Institute of Science and Technology (JAIST) in 2010, where he was with the Center for Highly Dependable Embedded Systems Technology as a Post-Doctoral Researcher in 2010-2012. From 2012 to 2023, he worked at the National Institute of Information and Communications Technology (NICT). He has been an Associate Professor at Kanazawa University since 2023. His research interests include public-key cryptography and information security. He was a recipient of the SCIS Innovation Paper Award from IEICE in 2012, the CSS Best Paper Award from IPSJ in 2016, the IPSJ Yamashita SIG Research Award in 2017, and the Best Paper Award from ProvSec 2022. He is a member of IEICE, IPSJ, and IACR.

Shingo Sato received the B.E., M.E., and Ph.D. degrees in information science from Yokohama National University, Japan, in 2015, 2017, and 2020, respectively. Currently, he is a specially appointed assistant professor at Institute of Advanced Sciences, Yokohama National University, Japan. His research interests include cryptography and information security.

Kazumasa Omote received his Ph.D. degree in information science from the Japan Advanced Institute of Science and Technology (JAIST), in 2002. He worked at Fujitsu Laboratories Ltd. during 2002-2008 and was engaged in research and development for network security. He was a Research Assistant Professor during 2008-2011 at JAIST, an Associate Professor during 2011-2016 at JAIST, and an Associate Professor during 2016-2022 at the University of Tsukuba. He has been a Professor with the University of Tsukuba, since 2022. His research interests include applied cryptography, network security, and blockchain security. He was the General Co-Chair of ACNS 2021 International Conference. He received the WISTP 2019 Best Paper Award.