electronics

MDPI

Article

Anonymous Sealed-Bid Auction on Ethereum †

Gaurav Sharma [1,*], Denis Verstraeten [1], Vishal Saraswat [2], Jean-Michel Dricot [1] and Olivier Markowitch [1]

[1] Département d'Informatique, Université libre de Bruxelles, 1050 Brussels, Belgium; denis.verstraeten@ulb.be (D.V.); jean-michel.dricot@ulb.be (J.-M.D.); olivier.markowitch@ulb.be (O.M.)
[2] Engineering Product Security Solutions(RBEI/ESY), Robert Bosch Engineering and Business Solutions Pvt. Ltd., Bangalore 560068, India; vishal.saraswat@gmail.com
* Correspondence: gaurav.sharma@ulb.be
† An extended abstract of this contribution has already been accepted by Blockchains and Smart Contracts workshop.

Abstract: In a competitive market, online auction systems enable optimal trading of digital products and services. Bidders can participate in existing blockchain-based auctions while protecting the confidentiality of their bids in a decentralized, transparent, secure, and auditable manner. However, in a competitive market, parties would prefer not to disclose their interests to competitors, and to remain anonymous during auctions. In this paper, we firstly analyze the specific requirements for blockchain-based anonymous fair auctions. We present a formal model tailored to study auction systems that facilitate anonymity, as well as a generic protocol for achieving bid confidentiality and bidder anonymity using existing cryptographic primitives such as designated verifier ring signature. We demonstrate that it is secure using the security model we presented. Towards the end, we demonstrate through extensive simulation results on Ethereum blockchain that the proposed protocol is practical and has minimal associated overhead. Furthermore, we discuss the complexity and vulnerabilities that a blockchain environment might introduce during implementation.

Keywords: blockchain; auction; anonymity; designated identity verifier ring signature; ethereum

Citation: Sharma, G.; Verstraeten, D.; Saraswat, V.; Dricot, J.-M; Markowitch, O. Anonymous Sealed-Bid Auction on Ethereum. *Electronics* **2021**, *10*, 2340. https://doi.org/10.3390/electronics10192340

Academic Editor: Martin Reisslein

Received: 18 August 2021
Accepted: 19 September 2021
Published: 24 September 2021

1. Introduction

A distributed ledger of transactions providing a record of consistency and immutability is well admired in the domain of cryptocurrencies. The decentralization inherent to blockchains currently offers several properties including transparency and integrity. These interesting features inspired many other areas such as insurance, cross-border banking, secure medical data, voting and e-auctions. The adoption of salient characteristics is immediate in these applications, however each of them demands several other particular features. For instance, e-voting [1] and e-auctions [2] might need privacy besides transparency and integrity. However, privacy in blockchain is still an unconventional asset, intuitively challenging because it seems against its default transparent nature. In blockchain transactions, privacy is achieved with confidentiality and anonymity techniques.

A well designed auction system having a thorough understanding of bidding behaviour can allocate resources in accordance with an expected outcome. There are different types of auctions determined by the particular environments and settings in which they take place. The most popular type of auction is the sealed-bid auction. Each bidder hands over a sealed envelope containing their secret bid to the auctioneer. There are two variations in this category, namely first-price and second-price auctions based on what the winning bidder pays eventually. In first-price, the auctioneer opens all envelopes and declares the highest bidder as the winner, while keeping losing bids secret. For second-price, the winner pays the second highest bid. The second-price sealed-bid is also called the Vickrey auction (named after William Vickrey).

The auction process has to be managed by an auctioneer, who is the designated party responsible for dispatching the auction resources. Ideally, this party is trusted and

should not collude with any of the bidders to outcome a biased decision [3]. However, reducing the trust on the auctioneer is an innovative feature in line with the philosophy of blockchains, this is why our contribution aims to expand the abilities of such blockchain-based auction systems.

1.1. Related Work

The anonymity in sealed-bid auctions is a well investigated area of research. However, these solutions can not directly be endorsed to apply on blockchain-based auctions. In the view of permissioned and permissionless settings, the essential requisites can be different and therefore a tailored solution must be adapted. For instance, the anonymity in Hyperledger Fabric [4]—a permissioned blockchain—can be achieved with Idemix [5], and hence anonymity techniques such as ring signatures, which we introduce in what follows, or group signatures may not be needed in such Fabric-based applications. The Idemix technology implements anonymous credential scheme that relies on a blind signature and efficient Zero-Knowledge Proof (ZKP) of signature possession. Such credential systems need a trusted third party to issue these credentials, for example the membership service provider (MSP) in Hyperledger Fabric. Indeed the anonymity or confidentiality support from the underlying blockchain platform can provide a less complicated solution to some complex use cases. To reduce complexity in open blockchains, some emerging solutions rely on a trusted third party (TTP) as a registration authority ensuring that the participants in the application are already registered and therefore malicious participants can be traced back. One example here is ZebraLancer [6], wherein a decentralized crowdsourcing of human knowledge is implemented on top of Ethereum. The authors implicitly consider a registration authority validating each participant's unique identity and issuing certificates to them. The common prefix-linkable anonymous authentication scheme provides anonymity to participants as long as they honestly follow the protocol. Such solutions are limited to easy traceability by the registration authority and collusion attack. The malicious auctioneer can easily collude with some bidder and make them win by sharing bids before the auction's conclusion. Furthermore, Ethereum can easily reveal anonymity while depositing/paying fees for rewards. Removing the trust from the auctioneer is indeed appreciable, however, the overhead seems to be high enough not to be acceptable. Ref. [7] presented an auction demonstration with only three parties: one seller and two buyers. A helper server assists as a trusted party with access to all the secrets and facilitates communication among chain codes. The time to execute a transaction is around 0.3 s which is expected to grow exponentially with an increasing number of participants. Moreover, this auction does not consider the anonymity of participants.

However, it seems more challenging to deliver a solution for anonymous auction on permissionless blockchain such as Ethereum. Ref. [8] proposed Hawk, a combination of the privacy of Zcash with the programmability of Ethereum. The privacy preserving Smart Contracts have been employed to handle transactional privacy on Ethereum blockchain. They focused on presenting a framework which can simultaneously support several applications such as sealed-bid auction, rock paper scissor game, crowdfunding application and swap financial instrument. The main limitation of Hawk is that it relies on a manager trusted explicitly not to leak secret inputs to Smart Contract. A recent improvement presented as zkHawk [9] by Banerjee et al. replaces the trusted manager with an Multi-Party Computation (MPC) protocol wherein the secret inputs are hidden from the blockchain as well as from other participants. To mitigate the overhead incurred with MPC protocol, the authors propose to run MPC off-chain as an interactive protocol. The major limitation here is that all participating bidders have to be online during the whole process to run MPC. A private and verifiable smart contract approach [10], introduced as a combination of secure MPC and proof-carrying code focusing primarily on correctness, privacy and verifiability guarantees for smart contracts on blockchains. The approach is well applicable to several applications such as private and verifiable crowdfundings, investment funds and double auctions for decentralized exchanges.

Strain [11] presented a protocol to build a blockchain-based sealed-bid auction, preserving bid privacy against malicious parties. A bulletin board is used to publish the winning bid which is determined by comparing them by pairs. Two different ZKPs ensure that the participants used the original bids under commitment and that the auctioneer declared the winner without manipulation. The malicious participant is punished by opening their commitment as their private key is partly shared among all participants through a distributed key generation process.

Galal and Youssef [12] presented a smart contract for verifiable first-price sealed-bid auction on the Ethereum blockchain. Bidders submit their bids to a smart contract using Pedersen commitment [13]. The commitments are secretly revealed to the auctioneer via a Public Key Encryption (PKE) scheme. After declaring the winner, for each losing bid, the auctioneer has to engage into a set of interactive commit-challenge-verify protocols to prove that the winning bid is greater than the losing bid and therefore, the complexity of interaction depends on the number of bidders. Later, Galal and Youssef [3] improved this protocol and presented a Smart Contract with succinctly verifiable ZKP which enables single proof verification for the whole auction process. Similar to Zcash, they used MPC among auctioneer and bidders to derive Common Reference String (CRS) during Zero-Knowledge Succinct Non-interactive ARgument of Knowledge (zk-SNARK) set up.

Some interesting approaches such as Anon-Pass [14] and SEAL [15] can be further tailored for blockchain implementation. Anon-Pass is an anonymous subscription service focusing on trade-off between unlinkability vs. re-authentication epoch. SEAL is an auctioneer-free first-price sealed-bid auction protocol. It securely computes the logical-OR of binary inputs without revealing each individual bit. No secret channel is required among participants. All operations are publicly verifiable and everyone including third party observers is able to verify the integrity of the auction outcome. Finally, a party comes forward with a winning proof.

Following the above discussion, it can be easily inferred that existing research is primarily focused on bids confidentiality and enabling fair auction using ZKP. However, the bidder's privacy is still missing. The same can be followed from Table 1. The use case *auction in a fair market* that we present in Section 1.2 requires additional cryptographic primitives in order to be successfully achieved.

Table 1. Summary of Existing Auction Protocols on Blockchain.

	Crypto Primitive	**Trusted 3rd Party**	**Bid Privacy**	**Anonymity**	**Blockchain**
Strain [11]	two-party comparison protocol, ZKP	Semi-trusted	√	Optional	NIS [$]
Galal and Youssef [12]	commitment, zk-SNARK [¶], PKE [§]	Semi-trusted	√	×	Ethereum
Galal and Youssef [3]	commitment, zk-SNARK [¶], PKE [§]	Semi-trusted	√	×	Ethereum
Hawk [8]	Generic Compiler	Semi-trusted	√	×	NIS [$]
Benhamouda [7]	secure MPC	Semi-trusted	√	×	HyperLedger Fabric
Our Work	ring Signature, commitment, PKE [§], zk-SNARK [¶]	Semi-trusted	√	√	Ethereum

[§] Public Key Encryption, [¶] Zero-Knowledge Succinct Non-interactive ARgument of Knowledge, [$] Not Implemented Specifically.

1.2. Auction in a Fair Market

We assume a situation where all the sellers or merchants want to offer their products to consumers at competitive prices. In such a competitive environment, merchants do not even want to make their interest public. Even if a merchant is the winner, he might not be interested to disclose it. Only the consumer, auctioneer and merchant are aware about

this procurement or purchase. Therefore, it would be a desirable feature for sellers and merchants to be able to offer a sealed-bid auction, without disclosing their identity. Without loss of generality, the auction type we choose here is first price sealed-bid auction. To summarise, *auction in a fair market* implies that the bidders are able to keep their identities secret and not to disclose their bids, and for the overall auction process to be able to be publicly verified.

Motivation and Contribution. The limitations of traditional auctions are centralization, lack of transparency, no interoperability and malicious behaviour by auctioneer or bidders. A blockchain-based auction can address these issues, however adding anonymity to this feature list is quite challenging. The existing blockchain-based auction solutions are either in permissioned setting or do not satisfy the anonymity of the bidders. We introduce a new protocol allowing to run an auction in a fair market. Our construction presents the advantage of being general and of using only existing cryptographic building blocks. The confidentiality and anonymity properties are achieved by using Designated Verifier Ring Signature (DVRS). Moreover, it is possible to leverage the transparency and auditability of blockchain platforms in order to make the auction process publicly verifiable, which is an interesting feature in the event of a dispute among parties.

Precisely, our sealed-bid auction protocol enables the following properties:

- The bidders do not want to disclose their bids (confidentiality of bids).
- The bidders do not want to disclose their identity during the whole process (anonymity of bidding parties).
- The auctioneer is minimally trusted to assist in the bidding process but not to affect its correct execution.
- Bids, once committed, can not be retracted or changed (bid binding).
- The auction process is publicly verifiable.

2. Preliminaries

In this section, we recall some standard definitions of commitment schemes PKE, Ring Signature (RS) and ZKP.

Definition 1. *A function* $f : \mathbb{N} \mapsto \mathbb{R}^+$ *is* negligible *if* $\forall$ *p belonging to the set of positive polynomials,* $\exists\ N \in \mathbb{N}$ *s.t.* $\forall\ n > N$, $f(n) < \frac{1}{p(n)}$ [16]. *A negligible function f is denoted* $f(n) = \mathsf{negl}(n)$.

Definition 2. *Let* $\mathcal{S}$ *be a set.* $s \xleftarrow{\$} \mathcal{S}$ *means that an element is randomly selected from a uniform distribution over* $\mathcal{S}$ *and assigned to s.*

Definition 3. *Let* $\mathcal{A}$ *be an adversary and let* $\mathcal{O}$ *be some oracle.* $\mathcal{A}^{\mathcal{O}(\cdot)}$ *means that* $\mathcal{A}$ *is given access to oracle* $\mathcal{O}$.

Definition 4. *Let* s_1 *and* s_2 *be two strings. Their concatenation is denoted as* $s_1||s_2$.

2.1. Commitment Scheme

A non-interactive commitment scheme is a set of algorithms

$$\mathsf{Comm} = (\mathtt{Setup}, \mathtt{Commit}, \mathtt{Open})$$

such that:

- $\mathtt{Setup}(1^k)$ outputs the public parameters of the system for a given security parameter k
- for any message m belonging to the message space, `Commit` outputs a commitment/opening pair
$$(c, d) \leftarrow \mathtt{Commit}(m)$$
- $\mathtt{Open}(c, d) \rightarrow m$ if c is a valid commitment, otherwise $\perp$

- Correctness: for any m belonging to the message space,

$$\Pr[\texttt{Open}(\texttt{Commit}(m)) = m] = 1$$

The commitment scheme should possess the following two properties:

Hiding. For any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\left| \Pr \left[\begin{array}{l} (s_0, s_1, \mathsf{st}) \leftarrow \mathcal{A}_1(1^k), \\ b \xleftarrow{\$} \{0,1\}, \\ (c,d) \leftarrow \texttt{Commit}(s_b), \\ b' \leftarrow \mathcal{A}_2(c, \mathsf{st}) \end{array} : b' = b \right] - \frac{1}{2} \right| = \mathsf{negl}(k)$$

where st is a state allowing stateful communication between $\mathcal{A}_1$ and $\mathcal{A}_2$.

Binding. For two distinct strings s and s' and any probabilistic polynomial time (PPT) adversary $\mathcal{A}$,

$$\Pr \left[\begin{array}{l} (c, s, d, s', d') \leftarrow \mathcal{A}(1^k), \\ vs. \leftarrow \texttt{Open}(c,d), \\ v' \leftarrow \texttt{Open}(c,d') \end{array} : v' = vs. \right] = \mathsf{negl}(k)$$

2.2. Ring Signature

The Ring Signature technique was introduced by Rivest et al. in 2001 [17] and allows a party to anonymously sign a message. A ring is a set of possible signers, with each of them being associated with a public key PK_i, and the corresponding secret key SK_i (for the i^{th} user). When a *verifier* checks the signature, they can only conclude that a member of the ring endorsed it, without being able to determine the identity of the signer. A RS scheme should be unforgeable and anonymous, and consists of a triple of PPT algorithms

$$\mathsf{RS} = (\texttt{RGen}, \texttt{RSig}, \texttt{RVer})$$

defined as follows:

1. Key Generation: $(\mathsf{PK}, \mathsf{SK}_s, s) \leftarrow \texttt{RGen}(1^k)$, where k denotes the security parameter, $\mathsf{PK} = [\mathsf{PK}_1, \ldots, \mathsf{PK}_n]$ is a list of n public keys, which includes the public key of the signer and $n-1$ decoy public keys, SK_s is the secret key of the signer and s is the index of the signer's public key PK_s in PK.
2. Signature: $\sigma \leftarrow \texttt{RSig}(\mathsf{PK}, \mathsf{SK}_s, m)$, where σ is the signature, and m the message to be signed.
3. Verification: $b \leftarrow \texttt{RVer}(\mathsf{PK}, m, \sigma)$, where outcome $b \in \{0,1\}$ indicates the validity of the signature.

Correctness. A RS is said to be correct if for any valid message m belonging to the message space:

$$\Pr\Big[(\mathsf{PK}, \mathsf{SK}_s, s) \leftarrow \texttt{RGen}(1^k) : \texttt{RVer}(\mathsf{PK}, m, \texttt{RSig}(\mathsf{PK}, \mathsf{SK}_s, m)) = 1\Big] = 1$$

Anonymity. A RS satisfies anonymity if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\left| \Pr \left[\begin{array}{l} (\mathsf{PK}, \mathsf{SK}_s, s) \leftarrow \texttt{RGen}(1^k), \\ (m, s_0, s_1, \mathsf{st}) \leftarrow \mathcal{A}_1^{\texttt{RSig}(\mathsf{PK}, \mathsf{SK}_s, \cdot)}(\mathsf{PK}), \\ b \xleftarrow{\$} \{0,1\}, \\ \sigma \leftarrow \texttt{RSig}(\mathsf{PK}, \mathsf{SK}_{s_b}, m), \\ b' \leftarrow \mathcal{A}_2^{\texttt{RSig}(\mathsf{PK}, \mathsf{SK}_s, \cdot)}(\mathsf{st}, \sigma) \end{array} : b' = b \right] - \frac{1}{2} \right| = \mathsf{negl}(k)$$

where st is a state allowing stateful communication between $\mathcal{A}_1$ and $\mathcal{A}_2$.

Unforgeability. A RS is unforgeable if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\Pr\left[\begin{array}{l}(\mathsf{PK}, \mathsf{SK}_s, s) \leftarrow \mathtt{RGen}(1^k),\\ (m, \sigma) \leftarrow \mathcal{A}^{\mathtt{RSig}(\mathsf{PK}, \mathsf{SK}_s, \cdot)}(\mathsf{PK}), \quad : b = 1\\ b \leftarrow \mathtt{RVer}(\mathsf{PK}, m, \sigma)\end{array}\right] = \mathsf{negl}(k)$$

where the adversary $\mathcal{A}$ has not been allowed to query the signing oracle $\mathtt{RSig}(\mathsf{PK}, \mathsf{SK}_s, \cdot)$ with message m.

2.3. Public Key Encryption

A PKE scheme consists of a triple of PPT algorithms

$$\mathsf{PKE} = (\mathtt{PKEGen}, \mathtt{Enc}, \mathtt{Dec})$$

defined as follows:

1. Key Generation: $(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathtt{PKEGen}(1^k)$, with k the security parameter, and PK and SK the public and secret key, respectively.
2. Encryption: $c \leftarrow \mathtt{Enc}(\mathsf{PK}, m)$, with c the ciphertext and m the message.
3. Decryption: $m \leftarrow \mathtt{Dec}(\mathsf{SK}, c)$, with $m = \perp$ if the decryption fails.

Correctness. A PKE scheme is said to be correct if for any valid message m belonging to the message space,

$$\Pr\Big[(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathtt{PKEGen}(1^k) : \mathtt{Dec}(\mathsf{SK}, \mathtt{Enc}(\mathsf{PK}, m)) = m\Big] = 1$$

Ciphertext Indistinguishability. A PKE scheme satisfies ciphertext indistinguishability if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\Pr\left[\begin{array}{l}(\mathsf{PK}, \mathsf{SK}) \leftarrow \mathtt{PKEGen}(1^k),\\ (m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}_1^{\mathtt{Dec}(\mathsf{SK}, \cdot)}(\mathsf{PK}),\\ b \xleftarrow{\$} \{0,1\}, \qquad\qquad : b' = b\\ c \leftarrow \mathtt{Enc}(\mathsf{PK}, m_b),\\ b' \leftarrow \mathcal{A}_2^{\mathtt{Dec}(\mathsf{SK}, \cdot)}(\mathsf{st}, c)\end{array}\right] = \mathsf{negl}(k)$$

where st is a state allowing stateful communication between $\mathcal{A}_1$ and $\mathcal{A}_2$ and where the adversary $\mathcal{A}_2$ is not allowed to query c from the decryption oracle $\mathtt{Dec}(\mathsf{SK}, \cdot)$.

2.4. Zero-Knowledge Proof

The ZKP system relies on the construction presented by Galal and Youssef in [3] which itself is an application of the zk-SNARK introduced by Ben-Sasson et al. in [18]. Let us recall a few definitions and properties about this proof system. A ZKP system is a triple of algorithms

$$\mathsf{ZKP} = (\mathtt{ZKGen}, \mathtt{ZKProve}, \mathtt{ZKVer})$$

defined as follows:

1. Key Generation: $\mathsf{CRS} \leftarrow \mathtt{ZKGen}\Big(1^k, \mathcal{L}\Big)$, with CRS the common reference string, k the security parameter, and $\mathcal{L}$ the language description.
2. Proof Generation: $\pi \leftarrow \mathtt{ZKProve}(\mathsf{CRS}, s, w)$, with s the statement whose membership to $\mathcal{L}$ has to be proved, w the corresponding witness, and π the proof.
3. Proof Verification: $b \leftarrow \mathtt{ZKVer}(\mathsf{CRS}, \pi, s)$, with $b \in \{0,1\}$ the single bit indicating whether the verification succeeded.

The zk-SNARK described in [18] features the following properties:

Perfect Completeness. An honest prover with a valid witness is always able to convince a honest verifier. Mathematically, let s be a claim belonging to the language $\mathcal{L}$ with w the corresponding witness:

$$\Pr\left[\begin{array}{l}\mathsf{CRS} \leftarrow \mathtt{ZKGen}\left(1^k, \mathcal{L}\right), \\ \pi \leftarrow \mathtt{ZKProve}(\mathsf{CRS}, s, w)\end{array} : \mathtt{ZKVer}(\mathsf{CRS}, \pi, s) = 1\right] = 1$$

Computational Soundness. The probability that a PPT adversary can convince an honest verifier that a false statement is true is negligible. Mathematically, let $\mathcal{A}$ be a PPT adversary and s be a claim which does not belong to the language $\mathcal{L}$:

$$\Pr\left[\begin{array}{l}\mathsf{CRS} \leftarrow \mathtt{ZKGen}\left(1^k, \mathcal{L}\right), \\ \pi \leftarrow \mathcal{A}^{\mathtt{ZKProve}(\mathsf{CRS}, s, \cdot)}(\mathsf{CRS}, s)\end{array} : \mathtt{ZKVer}(\mathsf{CRS}, \pi, s) = 1\right] = \mathsf{negl}(k)$$

Computational Zero-Knowledge. A PPT adversary is not able to extract any information about the witness from the proof. Slightly more formally, let $\mathcal{S}$ be a simulator knowing the witness w to a statement s in $\mathcal{L}$, and let $\overline{\mathcal{S}}$ be a simulator not knowing the witness w. For any $\mathcal{S}$, there exists an $\overline{\mathcal{S}}$ such that their views are computationally indistinguishable.

Succinctness. The proof system should run in polynomial size and time. Mathematically, $|\pi| = \mathsf{poly}_1(k)$ and $\mathsf{Time}(\mathtt{ZKVer}(\mathsf{CRS}, \pi, s)) = \mathcal{O}(|s| \cdot \mathsf{poly}_2(k))$, with $|\cdot|$ being the bit-length operator, poly_i with $\mathsf{i} \in \{1, 2\}$ belonging to the set of polynomial functions and $\mathsf{Time}(\cdot)$ being the asymptotic time operator.

3. Formal Model and Security Definitions

In this section, the mathematical definition of our Anonymous Fair Auction protocol is introduced, as well as the security properties which it has to fulfill. An Anonymous Bidding System (ABS) is a tuple of PPT algorithms

$$\mathsf{ABS} = (\textsc{KeyGen}, \textsc{Bid}, \textsc{BidOp}, \textsc{IdOp})$$

defined as follows:

- Key Generation: $(\mathsf{PK}, \mathsf{SK}_s, s, v) \leftarrow \textsc{KeyGen}(1^k)$, with k the security parameter, $\mathsf{PK} = [\mathsf{PK}_1, \ldots, \mathsf{PK}_n]$ the ring members' public key, SK_s the signer's private key, s and v the indices of the signer's and verifier's public key in PK, respectively.
- Bid submission: $(c, \mathsf{sig}, \tau_1, \tau_2) \leftarrow \textsc{Bid}(\mathsf{PK}, \mathsf{SK}_s, \mathsf{PK}_v, x)$, with x the bid value, PK_v the verifier's public key, c the commitment to the bid, sig the modified signature to the bid, τ_1 the bid opening token and τ_2 the identity opening token.
- Bid opening: $b \leftarrow \textsc{BidOp}(\mathsf{PK}, \mathsf{SK}_v, c, \mathsf{sig}, \tau_1)$, with SK_v the verifier's private key and $b \in \{0, 1\}$ a single bit indicating whether the bid opening succeeded.
- Identity opening: $b \leftarrow \textsc{IdOp}(\mathsf{SK}_v, \mathsf{sig}, \tau_2)$, with PK_s the signer's public key and $b \in \{0, 1\}$ a single bit indicating whether the identity opening succeeded.

Our ABS has to feature the following security properties:

Correctness. An ABS is said to be correct if for any valid bid x, the following two properties hold:

$$\Pr\left[\begin{array}{l}(\mathsf{PK}, \mathsf{SK}_s, s, v) \leftarrow \textsc{KeyGen}(1^k), \\ (c, \mathsf{sig}, \tau_1, \tau_2) \leftarrow \textsc{Bid}(\mathsf{PK}, \mathsf{SK}_s, \mathsf{PK}_v, x)\end{array} : \textsc{BidOp}(\mathsf{PK}, \mathsf{SK}_v, c, \mathsf{sig}, \tau_1) = 1\right] = 1$$

$$\Pr\left[\begin{array}{l}(\mathsf{PK}, \mathsf{SK}_s, s, v) \leftarrow \textsc{KeyGen}(1^k), \\ (c, \mathsf{sig}, \tau_1, \tau_2) \leftarrow \textsc{Bid}(\mathsf{PK}, \mathsf{SK}_s, \mathsf{PK}_v, x)\end{array} : \textsc{IdOp}(\mathsf{SK}_v, \mathsf{sig}, \tau_2) = 1\right] = 1$$

Unforgeability. An ABS satisfies unforgeability if for any PPT adversary $\mathcal{A}$,

$$\Pr\left[\begin{array}{l}(\mathsf{PK},\mathsf{SK}_s,s,v)\leftarrow \textsc{KeyGen}(1^k),\\(x,c,\mathsf{sig},\tau_1)\leftarrow \mathcal{A}^{\textsc{Bid}(\mathsf{PK},\mathsf{SK}_s,\mathsf{PK}_v,\cdot)}(\mathsf{PK}),\\b\leftarrow \textsc{BidOp}(\mathsf{PK},\mathsf{SK}_v,c,\mathsf{sig},\tau_1)\end{array}\;:\; b=1\right]=\mathsf{negl}(k)$$

where the adversary $\mathcal{A}$ is not allowed to query the bid submission oracle $\textsc{Bid}(\mathsf{PK},\mathsf{SK}_s,\mathsf{PK}_v,\cdot)$ with bid x.

An ABS satisfies *strong* unforgeability if it satisfies unforgeability with the adversary $\mathcal{A}$ given access to the bidding oracle $\textsc{Bid}(\mathsf{PK},\mathsf{SK}_s,\mathsf{PK}_v,\cdot)$ for bid x as well, but without the ability to obtain $(c,\mathsf{sig},\tau_1,\tau_2)$ as a response.

Signer Anonymity. An ABS satisfies anonymity if for any PPT adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$,

$$\left|\Pr\left[\begin{array}{l}(\mathsf{PK},\mathsf{SK}_s,s,v)\leftarrow \mathtt{Gen}(1^k),\\(x,s_0,s_1,\mathsf{st})\leftarrow \mathcal{A}_1^{\textsc{Bid}(\mathsf{PK},\mathsf{SK}_s,\mathsf{PK}_v,\cdot)}(\mathsf{PK}),\\b\xleftarrow{\$}\{0,1\},\\(c,\mathsf{sig},\tau_1,\tau_2)\leftarrow \textsc{Bid}(\mathsf{PK},\mathsf{SK}_{s_b},\mathsf{PK}_v,x),\\b'\leftarrow \mathcal{A}_2^{\textsc{Bid}(\mathsf{PK},\mathsf{SK}_s,\mathsf{PK}_v,\cdot)}(c,\mathsf{sig},\tau_1,\mathsf{st})\end{array}\;:\; b'=b\right]-\frac{1}{2}\right|=\mathsf{negl}(k)$$

where st is a state allowing stateful communication between $\mathcal{A}_1$ and $\mathcal{A}_2$. If the attacker has an additional access to the list of secret keys, and if signer anonymity is preserved, then the scheme has signer anonymity *against full key exposure*.

Unpretendability. An ABS satisfies unpretendability if for any PPT adversary $\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)$,

$$\Pr\left[\begin{array}{l}(\mathsf{PK},\mathsf{SK}_s,s,v)\leftarrow \mathtt{Gen}(1^k),\\(x,s_1,v_1,\mathsf{st})\leftarrow \mathcal{A}_1^{\textsc{Bid}(\mathsf{PK},\mathsf{SK}_s,\mathsf{PK}_v,\cdot)}(\mathsf{PK}),\\(c,\mathsf{sig},\tau_1,\tau_{2,1})\leftarrow \textsc{Bid}(\mathsf{PK},\mathsf{SK}_{s_1},\mathsf{PK}_{v_1},x)\\(s_2,v_2,\tau_{2,2})\leftarrow \mathcal{A}_2^{\textsc{Bid}(\mathsf{PK},\mathsf{SK}_s,\mathsf{PK}_v,\cdot)}(c,\mathsf{sig},\tau_1,\tau_{2,1},\mathsf{st})\\b\leftarrow \textsc{IdOp}(\mathsf{SK}_{v_2},\mathsf{sig},\tau_{2,2})\end{array}\;:\; b=1\right]=\mathsf{negl}(k)$$

where the adversary $\mathcal{A}$ is not allowed to choose $s_1=s_2$.

4. Proposed Scheme

Our proposed construction is built upon a DVRS introduced by Saraswat and Pandey [19]. The resulting scheme provides correctness, unforgeability, unpretendability and signer anonymity. In this context, the auctioneer is the designated verifier trusted to open the bids. However, a collusion between malicious bidders and verifier should not affect the outcome.

There are separate time frames for bid submission (T_1), bid opening (T_2) and winner declaration (T_3) such that $T_1 < T_2 < T_3$. During the first time interval $[0, T_1]$, the bidders submit ring signed commitments of their bids and of their identities to the auctioneer. In the course of second time interval $[T_1, T_2]$, the bidders reveal their bids by encrypting them using the public key of the auctioneer (known by anyone). In the last interval $[T_2, T_3]$, the winning bidder reveals their identity to the auctioneer.

In order to successfully realize our construction, we rely on the following assumptions:

- The auctioneer is minimally trusted and assumed not to disclose the bidder's input
- Each participant and auctioneer deposit an amount of cryptocurrency in the Smart Contract. After the auction is concluded, the deposits get refunded to honest participants. This provides an economic incentive to strictly follow the protocol.
- We assume the existence of a blockchain providing *anonymity* and *confidentiality* on the transactions. The first property is required to ensure that the anonymity of the overall scheme is not broken by merely looking at the transaction metadata, such as the fields *from* and *to* in the context of Ethereum. Without the second property, it

would be possible to gain information about the value of the winning bid by looking at currency exchanges on the network. This assumption, which might seem unrealistic at first glance, is not so costly since there already exists protocols providing such features, e.g., Zether [20] for Ethereum.

4.1. Our Generic Construction

Our construction is built using four algorithms. The first one, represented in Algorithm 1, is the key generation algorithm, KEYGEN, which takes care of generating the keys required to form the rings. It can be used for bidders and verifier. The same keys are used for both signature and encryption.

Algorithm 1 Key generation algorithm.

function KEYGEN(1^k)
  $(\mathsf{PK}, \mathsf{SK}_s, s) \leftarrow \mathtt{RGen}(1^k)$
  $v \leftarrow \mathtt{getAuctioneerIndex}(\mathsf{PK})$
  return $(\mathsf{PK}, \mathsf{SK}_s, s, v)$
end function

The second one is the bid submission algorithm BID depicted in Algorithm 2. This algorithm outputs the commitment to the bid c, the modified signature $\mathsf{sig} = \sigma||c_1||c_2$, the bid opening token $\tau_1 = C_1||d_1$ and the identity opening token $\tau_2 = C_2||d_2$. The modified signature sig is used to obtain a proof that all the required information about the bid value and the identity of the bidder are embedded and that they cannot be changed at later time. The bidder needs to send the bid and identity opening tokens for the verifier to be able to retrieve the bid value and the identity, respectively. Two different rings of parties are used in this algorithm. The first one is arbitrarily constructed by the bidder to maintain their anonymity. The second one is used to make sure that the verifier cannot convince anyone that the bidder actually generated a given bid, since the verifier is also included in the ring, which is precisely the designated verifier approach presented by Rivest et al. as an ad hoc application of their RS scheme introduced in their foundational paper [17]. For the sake of generality, two different RS schemes can be used, `RSig` and `R2Sig` in this case.

Algorithm 2 Bid submission algorithm.

function BID($\mathsf{PK}, \mathsf{SK}_s, \mathsf{PK}_v, x$)
  $(c, d) \leftarrow \mathtt{Commit}(x)$
  $\sigma \leftarrow \mathtt{RSig}(\mathsf{PK}, \mathsf{SK}_s, c)$
  $\Sigma \leftarrow \mathtt{RSig}(\mathsf{PK}, \mathsf{SK}_s, c||\sigma)$
  $m_1 \leftarrow c||\sigma||\Sigma||d$
  $C_1 \leftarrow \mathtt{Enc}(\mathsf{PK}_v, m_1)$
  $(c_1, d_1) \leftarrow \mathtt{Commit}(C_1)$
  $\delta \leftarrow \mathtt{R2Sig}([\mathsf{PK}_s, \mathsf{PK}_v], \mathsf{SK}_s, c||\sigma||\Sigma)$
  $m_2 \leftarrow c||\mathsf{PK}_s||\mathsf{PK}_v||\sigma||\Sigma||\delta$
  $C_2 \leftarrow \mathtt{Enc}(\mathsf{PK}_v, m_2)$
  $(c_2, d_2) \leftarrow \mathtt{Commit}(C_2)$
  return $(c, \sigma||c_1||c_2, C_1||d_1, C_2||d_2)$
end function

The third one, depicted in Algorithm 3, is the bid opening algorithm, BIDOP, which is used to verify that the bid is valid, and to open its value. The bid opening token generated by the bidder is required. If the signatures and the commitments are successfully verified, the verifier locally stores the bid x and the corresponding opening value d.

Finally, the last one is the identity opening algorithm, IDOP, depicted in Algorithm 4, which allows the verifier to check that the identity of the winning bidder is valid and to obtain their public key. The corresponding identity opening token is needed. The RS is

verified using the ring made of the public keys of the winning bidder and the verifier only. If the identity is successfully verified, the winning bidder's public key is locally stored by the verifier.

Algorithm 3 Bid opening algorithm.

```
1: function BIDOP(PK, SK_v, c, sig, τ_1)
2:     b ← 0
3:     σ, c_1, c_2 ← parse(sig)
4:     if RVer(PK, c, σ) then
5:         C_1, d_1 ← parse(τ_1)
6:         if C_1 = Open(c_1, d_1) then
7:             m_1 ← Dec(SK_v, C_1)
8:             c̃, σ̃, Σ, d ← parse(m_1)
9:             if c̃ = c ∧ σ̃ = σ then
10:                if RVer(PK, c||σ, Σ) then
11:                    x ← Open(c, d)
12:                    store(x)
13:                    store(d)
14:                    b ← 1
15:                end if
16:            end if
17:        end if
18:    end if
19:    return b
20: end function
```

Algorithm 4 Identity opening algorithm.

```
1: function IDOP(SK_v, sig, τ_2)
2:     b ← 0
3:     σ, c_1, c_2 ← parse(sig)
4:     C_2, d_2 ← parse(τ_2)
5:     if C_2 = Open(c_2, d_2) then
6:         m_2 ← Dec(SK_v, C_2)
7:         c, PK_s, PK_v, σ̃, Σ, δ ← parse(m_2)
8:         if c = getWinningCommitment() then
9:             if R2Ver([PK_s, PK_v], c||σ||Σ, δ) then
10:                store(PK_s)
11:                b ← 1
12:            end if
13:        end if
14:    end if
15:    return b
16: end function
```

4.2. Anonymous Fair Auction Protocol

The protocol implemented in our construction is presented in Algorithm 5. It is designed for a fixed number of N bidders. The first step is concerned with the generation of the keys for the auctioneer and the bidders. Regarding the keys and the rings, some notations need to be introduced:

- PK_v: Verifier's public key.
- PK^i: i^{th} bidder's ring of public keys.
- PK^i_j: Public key at j^{th} position in i^{th} bidder's ring of public keys.
- PK_i: i^{th} bidder's public key.

Algorithm 5 Anonymous Fair Trading protocol.

$(\mathsf{PK}^v, \mathsf{SK}_v, s, v) \leftarrow \textsc{KeyGen}(1^k)$ ▷ For auctioneer, $s = v$.
for $i \in \{1, \ldots, N\}$ **do**
  $(\mathsf{PK}^i, \mathsf{SK}_i, s_i, v_i) \leftarrow \textsc{KeyGen}(1^k)$ ▷ Generating keys for i^{th} bidder.
end for
for $i \in \{1, \ldots, N\}$ **do** ▷ Time T_1: bid submission.
  $x_i \leftarrow \texttt{chooseBid}()$
  $(c_i, \mathsf{sig}_i, \tau_{1,i}, \tau_{2,i}) \leftarrow \textsc{Bid}(\mathsf{PK}^i, \mathsf{SK}_i, \mathsf{PK}^i_{v_i}, x_i)$
  $\texttt{bidderPostToBlockchain}(c_i, \mathsf{sig}_i, \mathsf{PK}^i)$
end for
for $i \in \{1, \ldots, N\}$ **do** ▷ Time T_2: bid opening and verification.
  $\texttt{bidderPostToBlockchain}(\tau_{1,i})$
end for
for $i \in \{1, \ldots, N\}$ **do**
  $(c_i, \mathsf{sig}_i, \tau_{1,i}) \leftarrow \texttt{auctioneerFetchFromBlockchain}()$
  if $\textsc{BidOp}(\mathsf{PK}^i, \mathsf{SK}_v, c_i, \mathsf{sig}_i, \tau_{1,i})$ **then** ▷ Bid x_i and decommitment d_i stored locally at auctioneer.
    continue
  end if
end for
$c \leftarrow \texttt{auctioneerFetchFromBlockchain}()$ ▷ Array of size N with the commitments of all the bidders.
$(c_w, \text{success}) \leftarrow \textsc{VerifiableAuction}(x, c, d)$ ▷ c_w is the commitment to the winning bid.
$\pi \leftarrow \texttt{ZKProve}(\mathsf{CRS}, (c, c_w), (x, d))$
$\texttt{auctioneerPostToBlockchain}(c_w, \pi)$
for $i \in \{1, \ldots, N\}$ **do** ▷ Time T_3: opening of winning bidder's identity and verification.
  $c_w \leftarrow \texttt{bidderFetchFromBlockchain}()$
  if $c_i = c_w$ **then** ▷ i^{th} bidder is the winner.
    $\texttt{bidderPostToBlockchain}(\tau_{2,i})$
  end if
end for
$(\mathsf{sig}_w, \tau_{2,w}) \leftarrow \texttt{auctioneerGetchFromBlockchain}()$
if $\textsc{IdOp}(\mathsf{SK}_v, \mathsf{sig}_w, \tau_{2,w})$ **then** ▷ Identity of winning bidder successfully opened and verified.
  continue
end if

Since all the bidders need to share the same auctioneer, the following condition must hold:

$$\mathsf{PK}_v \subseteq \cap_{i=1}^{N} \mathsf{PK}^i$$

At time T_1, each bidder chooses a bid and executes the BID algorithm to submit the commitment to their bid and the corresponding signature to the blockchain, and to locally store the opening tokens.

At time T_2, the bidders submit their respective bid opening token to the blockchain. The auctioneer fetches the bid opening tokens, verifies the bids and stores their value. Then, the auctioneer computes the winning bid using the VERIFIABLEAUCTION algorithm and sends the commitment to the winning bid and the zk-SNARK to the blockchain.

Finally, at time T_3, the winning bidder submits their identity opening token. The auctioneer verifies the identity and stores the winner's public key.

During the whole auction process, the interactions between the auctioneer, the bidders and the Smart Contract can be listed as follows:

- $\texttt{Tx}_1$: The auctioneer triggers the auction Smart Contract.

- $\texttt{Tx}_2$: The commitment c and the modified signature sig output by BID are submitted by each bidder to the Smart Contract.
- $\texttt{Tx}_3$: The bid opening token τ_1 is submitted by each bidder to the Smart Contract so that the auctioneer can run the algorithm BIDOP.
- $\texttt{Tx}_4$: The auctioneer submits the return value of Algorithm 6 c_w as well as the corresponding zk-SNARK proof π.
- $\texttt{Tx}_5$: The identity opening token τ_2 is submitted by the winning bidder to the Smart Contract so that the auctioneer can run algorithm IDOP.

Algorithm 6 Returns the commitment to the winning bid.

function VERIFIABLEAUCTION(x, c, d)
  $max \leftarrow 0$
  $c_w \leftarrow \texttt{null}$
  success $\leftarrow 1$
  for $i \in \{1, \ldots, N\}$ **do** ▷ N is the constant number of bidders.
    if $x_i \neq \texttt{Open}(c_i, d_i)$ **then**
      success $\leftarrow 0$
      return c_w, success
    end if
    if $x_i > max$ **then**
      $max \leftarrow x_i$
      $c_w \leftarrow c_i$
    end if
  end for
  return c_w, success
end function

4.3. Zero-Knowledge Proof

As indicated in Section 2.4, the ZKP system used in our scheme is a slightly modified version of the one proposed in [3] and can be observed in Algorithm 6. In [3], they designed their algorithm to handle a Vickrey auction, which implies that they have to return both the highest and the second highest bids. In our approach, we want to implement a first-price sealed-bid auction. Moreover, we do not want to disclose the value of the winning bid, hence the fact that our algorithm only returns the *commitment* to this bid. We also assume that all the bids are different.

Practically, as it is suggested in [3], it is required to translate the algorithm into an Arithmetic Circuit. To do so, an Arithmetic Circuit Generator, as the one designed by Ben-Sasson et al. in [18], has to be used. More formally, the setup and usage of the ZKP system requires the following operations:

1. The auctioneer has to generate a proof (or an Argument of Knowledge, which is the probabilistic counterpart, as it case for a zk-SNARK) and therefore needs a language description of the algorithm:

$$\mathcal{L} = \texttt{Lang}(\texttt{verifiableAuction})$$

   With $\texttt{Lang} : \texttt{Func} \mapsto \Sigma^*$ a mapping between the space of algorithms Func to the space of language descriptions over an alphabet Σ. Intuitively, this mapping can be seen as a compiler and is a black-box representation of the Arithmetic Circuit Generator described in [18].
2. Having defined the language, it is possible to generate the CRS, which finalizes the initial setup of the ZKP system:

$$\text{CRS} \leftarrow \texttt{ZKGen}\left(1^k, \mathcal{L}\right)$$

These first two steps are only required when the proof system is setup for the first time, and the CRS stays valid as long as N, the maximum number of bids, remains the same. CRS is stored on the blockchain.

3. When the bidders have revealed their bids, the auctioneer executes Algorithm 6 and determines c_w, which is the commitment to the highest bid stored on the blockchain, and is able to compute the proof π. In this situation, the statement, which is publicly known, is the $(N+1)$ tuple $(c; c_w)$, with c the N tuple made of the commitments to the bids. The witness, known only to the auctioneer, is (x, d), which is a $2N$ tuple made of the value of the bids as well as their opening value. This ensures that the auction is executed correctly without leaking any information about the value of the bids. The proof π is also stored on the blockchain.

$$\pi \leftarrow \texttt{ZKProve}(\mathsf{CRS}, (c; c_w), (x, d))$$

4. The Smart Contract verifies the auction thanks to the proof π and to $(c; c_w)$. Since the CRS, π and $(c; c_w)$ are stored on the blockchain, one can leverage the auditability inherent to this data structure to be convinced that the auction was executed correctly in a transparent, trustless configuration.

$$b \leftarrow \texttt{ZKVer}(\mathsf{CRS}, \pi, (c; c_w))$$

5. Security Analysis

Anonymous Bidding System should be analyzed in two aspects- the security of our proposed cryptographic scheme and its integration in the blockchain platform. Here, we claim that the auctioneer will declare the correct auction winner. However, the validity of our claim depends upon some presumptions: (i) we assume that the blockchain is an ideal public ledger; (ii) the underlying components such as ring signature, public key encryption, commitment scheme and ZKP satisfy their cryptographic properties as mentioned in Section 2. The immutable and transparent nature of blockchain guarantees smooth integration of our generic scheme, to successfully hold an anonymous blockchain-based auction.

An adversary can break the *anonymity of bidders* only if during the interaction, a bidder's blockchain address is identifiable to auctioneer. The unconditional anonymity provided by ring signatures in time interval T_1 and T_2 prohibits auctioneer to identify the bidder until winning bidder opens their identity. On the other hand, the auctioneer's identity or blockchain address is public and known to all bidders in advance. Maintaining *bid privacy* seems challenging in a transparent blockchain environment. We achieve it by using a commitment scheme which is hiding and binding, and submitting this commitment value to smart contract, publicly on blockchain. Moreover, the auctioneer can not access these bid values until the bid submission is closed.

A malicious auctioneer can pose threats in following ways: (i) auctioneer can claim that the ciphertext submitted by bidder did not open to the bid commitment; (ii) auctioneer claims the opening value provided by bidder did not compute the corresponding commitment; (iii) auctioneer can collude with a bidder by sharing bid values of other bidders and assist them to win; (iv) auctioneer can announce a favoured bidder as winner. The first and second threats can be mitigated easily by utilizing blockchain's transparency feature. The ciphertext and commitment values are submitted as transactions on immutable blockchain by bidders, and therefore the auctioneer cannot cheat in this case. To protect against third threat of collusion with bidder, we introduced the concept of time intervals, the bid opening is initiated only after bid submission is closed. The ZKP protects against the last threat by submitting a public proof on blockchain where anyone can verify it.

A malicious bidder can cheat the system by: (i) submitting arbitrary ciphertext; (ii) not providing correct opening values; (iii) not responding in time interval T_3. The first two threats are straightforward to mitigate as all the data submitted by bidders is in the form of transactions to the smart contract, and therefore can be verified. Our proposed solution

does not cover this third threat and we leave it as an open problem. However, the solution suggested by Strain [11] can partially fill this gap by allowing the other participants to open the commitments by an honest majority, in the case of a dispute. The auction process is *publicly verifiable* as its correctness can be verified using ZKP.

Below, we analyze the security of our proposed cryptographic scheme. Here we prove that the security of proposed cryptographic scheme directly relies on the individual security of the different building blocks.

Theorem 1. *Assume* $\mathsf{RS} = (\mathtt{RGen}, \mathtt{RSig}, \mathtt{RVer})$ *be an anonymous and unforgeable ring signature,* $\mathsf{PKE} = (\mathtt{PKEGen}, \mathtt{Enc}, \mathtt{Dec})$ *be a semantically secure encryption scheme and* $\mathsf{Comm} = (\mathtt{Setup}, \mathtt{Commit}, \mathtt{Open})$ *be a hiding and binding commitment scheme, then our construction*

$$\mathsf{ABS} = (\textsc{KeyGen}, \textsc{Bid}, \textsc{BidOp}, \textsc{IdOp})$$

relying on these components is correct, signer anonymous, unforgeable and unpretendable.

Proof. In order to prove this, we need to successfully deduce the security requirements of our construction from the corresponding security requirements of the underlying components.

Correctness. The correctness of ABS is implied by successful opening of bids and uncovering the true identity of the winner. For a submitted bid $Z_i = (c_i, \mathsf{sig}_i)$, with modified signature $\mathsf{sig}_i = \sigma_i||c_{1,i}||c_{2,i}$, bid opening token $\tau_{1,i} = C_{1,i}||d_{1,i}$ and ring PK^i, the correctness of the bid opening algorithm relies on the correctness of RS, PKE and Comm. Therefore, the following conditions must hold:

- $\mathtt{RVer}(\mathsf{PK}^i, c_i, \sigma_i) = 1$
- $\mathtt{Open}(c_{1,i}, d_{1,i}) = C_{1,i}$
- $\mathtt{Dec}(\mathsf{SK}_v, C_{1,i}) = m_{1,i}$ where $m_{1,i} = c_i||\sigma_i||\Sigma_i||d_i$
- $\mathtt{RVer}(\mathsf{PK}^i, c_i||\sigma_i, \Sigma_i) = 1$

Similarly, for successful identity verification $\textsc{IdOp}(\mathsf{SK}_v, \mathsf{sig}_i, \tau_{2,i}) = 1$ where $\tau_{2,i} = C_{2,i}||d_{2,i}$ is the identity opening token introduced in our proposed construction, the following must hold:

- $\mathtt{Open}(c_{2,i}, d_{2,i}) = C_{2,i}$
- $\mathtt{Dec}(\mathsf{SK}_v, C_{2,i}) = m_{2,i}$ where $m_{2,i} = c_i||\mathsf{PK}_w||\mathsf{PK}_v||\sigma_i||\Sigma_i||\delta_i$ and where w represents the winner's index.
- $\mathtt{R_2Ver}([\mathsf{PK}_w, \mathsf{PK}_v], c_i||\sigma_i||\Sigma_i, \delta_i) = 1$

Unforgeability. Let us assume that $\mathcal{A}$ is the adversary attempting a forgery on ABS. Here we construct a new adversary $\mathcal{B}$ which has same polynomial time complexity as $\mathcal{A}$, attacking the unforgeability of RS with advantage

$$\mathsf{Adv}^{\mathsf{UF-CMA}}_{\mathsf{ABS},\mathcal{A}}(k) \leq \mathsf{Adv}^{\mathsf{UF-CMA}}_{\mathsf{RS},\mathcal{B}}(k)$$

The adversary $\mathcal{B}$ has access to the signing oracles $\mathtt{RSig}(\cdot, \mathsf{SK}_s, \cdot)$ and $\mathtt{R_2Sig}(\cdot, \mathsf{SK}_s, \cdot)$ and answers the signing queries coming from $\mathcal{A}$ as follows:

- For any bid x_i, $\mathcal{B}$ requests its own signing oracle RSig and outputs σ_i. Further, $\mathcal{B}$ computes the RS Σ_i on message $c_i||\sigma_i$, with $(c_i, d_i) \leftarrow \mathtt{Commit}(x_i)$.
- Computes encryption $C_{1,i} \leftarrow \mathtt{Enc}(\mathsf{PK}_v, m_{1,i})$ where message $m_{1,i} = c_i||\sigma_i||\Sigma_i||d_i$.
- Computes commitment $(c_{1,i}, d_{1,i}) \leftarrow \mathtt{Commit}(C_{1,i})$.
- Computes a RS δ_i on message $c_i||\sigma_i||\Sigma_i$ using $\mathtt{R_2Sig}(\cdot, \mathsf{SK}_s, \cdot)$ oracle.
- Computes encryption $C_{2,i} \leftarrow \mathtt{Enc}(\mathsf{PK}_v, m_{2,i})$ where message $m_{2,i} = c_i||\mathsf{PK}_i||\mathsf{PK}_v||\sigma_i||\Sigma_i||\delta_i$.
- Computes commitment $(c_{2,i}, d_{2,i}) \leftarrow \mathtt{Commit}(C_{2,i})$.
- Returns $Z_i = (c_i, \mathsf{sig}_i)$ and $\tau_{1,i}$ to adversary $\mathcal{A}$, with $\mathsf{sig}_i = \sigma_i||c_{1,i}||c_{2,i}$ and $\tau_{1,i} = C_{1,i}||d_{1,i}$.

According to the description of ABS, the above interaction between $\mathcal{A}$ and $\mathcal{B}$ is perfectly simulated for $\mathcal{A}$ by $\mathcal{B}$. Following the definition of BIDOP, whenever adversary $\mathcal{A}$ outputs a forged $Z_i, \tau_{1,i}$, it means that $\mathcal{B}$ successfully forged some intermediate RS σ_i, Σ_i or δ_i. Therefore, a forgery by $\mathcal{A}$ in ABS implies forgery of RS by $\mathcal{B}$.

Unpretendability. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be the adversary breaking the unpretendability of ABS and having access to the bidding oracle $\text{BID}(\mathsf{PK}, \mathsf{SK}_s, \mathsf{PK}_v, \cdot)$. Here we construct a new adversary $\mathcal{B}$ which has the same polynomial time complexity as $\mathcal{A}$, attacking the binding property of commitment scheme Comm satisfying

$$\mathsf{Adv}^{\mathsf{UNP-FKE}}_{\mathsf{ABS},\mathcal{A}}(k) \leq \mathsf{Adv}^{\mathsf{Bind}}_{\mathsf{Comm},\mathcal{B}}(k)$$

The challenger runs $(\mathsf{PK}, \mathsf{SK}_i, i, v) \leftarrow \text{KEYGEN}(1^k)$ and provides it to adversary $\mathcal{B}$. $\mathcal{B}$ further forwards this tuple to adversary $\mathcal{A}$. $\mathcal{B}$ answers the commitment queries of $\mathcal{A}$ by using its commitment oracle $\texttt{Commit}(\cdot)$. The unpretendability game runs as follows:

- $\mathcal{B}$ obtains from $\mathcal{A}_1$: $(x, s_1, v_1, \mathsf{st}) \leftarrow \mathcal{A}_1^{\text{BID}(\mathsf{PK},\mathsf{SK}_i,\mathsf{PK}_{v,\cdot})}(\mathsf{PK})$
- $\mathcal{B}$ computes: $(c, \mathsf{sig}, \tau_1, \tau_{2,1}) \leftarrow \text{BID}(\mathsf{PK}, \mathsf{SK}_{s_1}, \mathsf{PK}_{v_1}, x)$, with $\mathsf{sig} = \sigma||c_1||c_2$, $\tau_1 = C_1||d_1$ and $\tau_{2,1} = C_{2,1}||d_{2,1}$
- $\mathcal{B}$ obtains from $\mathcal{A}_2$: $(s_2, v_2, \tau_{2,2}) \leftarrow \mathcal{A}_2^{\text{BID}(\mathsf{PK},\mathsf{SK}_i,\mathsf{PK}_{v,\cdot})}(c, \mathsf{sig}, \tau_1, \tau_{2,1}, \mathsf{st})$, with $\tau_{2,2} = C_{2,2}||d_{2,2}$

where $\tau_{2,j}$ is the identity opening token generated for sender at index s_j in the ring PK by adversary $\mathcal{A}_j$, $j \in \{1, 2\}$. Now the adversary $\mathcal{B}$ halts with a commitment tuple:

$$(c_2, C_{2,0}, d_{2,0}, C_{2,1}, d_{2,1})$$

Here the sender's index $s_1 \neq s_2$ and we claim that unpretendability game for $\mathcal{A}$ is perfectly simulated by $\mathcal{B}$. Furthermore, the successful verification

$$\text{IDOP}(\mathsf{SK}_{v_2}, \mathsf{sig}, \tau_{2,2}) = 1$$

implies the corresponding commitments must hold, i.e., $\texttt{Open}(c_2, d_{2,2}) = C_{2,2}$. On the other hand, since $(c_2, d_{2,1}) = \texttt{Commit}(C_{2,1})$ by definition, we have $\texttt{Open}(c_2, d_{2,1}) = C_{2,1}$. Since the signer index $s_1 \neq s_2$, $C_{2,1} \neq C_{2,2}$ and therefore $\mathcal{B}$ has clearly attacked the binding property of Comm. Therefore, whenever the adversary $\mathcal{A}$ successfully breaks the unpretendability of ABS, $\mathcal{B}$ also breaks the binding property of Comm.

Signer Anonymity. The ABS signer anonymity relies upon the signer anonymity of the RS, the semantic security of the encryption scheme PKE and the hiding property of the commitment scheme Comm. The final message $Z = (c, \mathsf{sig})$, with $\mathsf{sig} = \sigma||c_1||c_2$, includes a RS σ and three commitments c, c_1 and c_2. The underlying RS σ guarantees anonymity and therefore a non-negligible advantage for the adversary can only come from the commitment scheme.

Until the time interval T_3 starts and identity token τ_2 is released, the designated verifier is a potential adversary as well and we ensure the signer anonymity is preserved for that period.

Let $\mathcal{E}$ be the adversary attacking the anonymity of RS with zero advantage and $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be the adversary breaking the anonymity of ABS and having access to the bidding oracle $\text{BID}(\mathsf{PK}, \mathsf{SK}_i, \mathsf{PK}_v, \cdot)$. Here we construct a new adversary $\mathcal{B}$ which has same polynomial time complexity as $\mathcal{A}$, attacking the hiding property of commitment scheme Comm with advantage

$$\mathsf{Adv}^{\mathsf{ANON-FKE}}_{\mathsf{ABS},\mathcal{A}}(k) \leq \mathsf{Adv}^{\mathsf{Hide}}_{\mathsf{Comm},\mathcal{B}}(k)$$

The challenger runs $(\mathsf{PK}, \mathsf{SK}_i, i, v) \leftarrow \textsc{KeyGen}(1^k)$ and provides it to adversary $\mathcal{B}$. $\mathcal{B}$ further forwards this tuple to adversary $\mathcal{A}$. The adversary $\mathcal{B}$ answers $\mathcal{A}$'s commitment queries by calling the commitment oracle $\texttt{Commit}(\cdot)$ in its own challenge. The signer anonymity game runs as follows:

- $\mathcal{A}$ provides to $\mathcal{B}$: $(x, s_0, s_1, \mathsf{st}) \leftarrow \mathcal{A}_1^{\textsc{Bid}(\mathsf{PK}, \mathsf{SK}_i, \mathsf{PK}_{v'}, \cdot)}(\mathsf{PK})$
- For a bid x and two signer indexes s_i, $\mathcal{B}$ requests its own signing oracle $\texttt{RSig}$ and outputs two signatures σ_i
- $\mathcal{B}$ further computes Σ_i on message $c||\sigma_i$
- Compute encryption $C_{1,i} \leftarrow \texttt{Enc}(\mathsf{PK}_v, m_{1,i})$ where message $m_{1,i} = c||\sigma_i||\Sigma_i||d$
- $\mathcal{B}$ further computes δ_i on message $c||\sigma_i||\Sigma_i$ and outputs the ciphertext as $C_{2,i}$ on message $m_{2,i} = c||\mathsf{PK}_{s_i}||\mathsf{PK}_v||\sigma||\Sigma||\delta$

Now, for two different signer identities, $\mathcal{B}$ forwards these two ciphertexts $C_{1,i}$ and $C_{2,i}$ to $\mathcal{B}$'s challenger. The challenger chooses a bit $b \xleftarrow{\$} \{0,1\}$, computes $(c_{1,b}, d_{1,b})$ and $(c_{2,b}, d_{2,b})$ using $\texttt{Commit}$ oracle and returns to $\mathcal{B}$. Now, $\mathcal{B}$ chooses a bit $b_1 \xleftarrow{\$} \{0,1\}$ and provides $(c, \mathsf{sig}_{b_1}, \tau_{1,b_1}, \tau_{2,b_2})$ as challenge bid return to adversary $\mathcal{A}$. The adversary $\mathcal{A}$ guesses a random bit $b_2 \in \{0,1\}$ and returns to $\mathcal{B}$ which $\mathcal{B}$ forwards as its own guess to challenger. Here, $\mathcal{B}$ perfectly simulates the anonymity game for $\mathcal{A}$ and therefore

$$\mathsf{Adv}^{\mathsf{ANON-FKE}}_{\mathsf{ABS},\mathcal{A}}(k) = \mathsf{Adv}^{\mathsf{Hide}}_{\mathsf{Comm},\mathcal{B}}(k)$$

Here, note that it is unlikely to guess whether $c_{1,b}$ is the commitment for $C_{1,i}$ depending upon b's choice by challenger. The adversary $\mathcal{A}$ returns $b_2 = b_1$ with non-negligible advantage if $c_{1,b}$ is the commitment for C_{1,b_1} otherwise returns a random b_2. As $\mathcal{B}$ returns b_2 as their response, which is actually received from $\mathcal{A}$, therefore, $\mathcal{B}$'s advantage is equal to the advantage of $\mathcal{A}$. So, before the release of identity verification token, we have

$$\mathsf{Adv}^{\mathsf{ANON-FKE}}_{\mathsf{ABS},\mathcal{A}}(k) \leq \mathsf{Adv}^{\mathsf{ANON-FKE}}_{\mathsf{RS},\mathcal{E}}(k) + \mathsf{Adv}^{\mathsf{Hide}}_{\mathsf{Comm},\mathcal{B}}(k)$$

Once the time T_2 is over and identity token $C_{2,i}$ is released to auctioneer, the PKE scheme semantic security can only protect the signer's anonymity. However, we assume the PKE guarantees security against any such attack and therefore, we have

$$\mathsf{Adv}^{\mathsf{ANON-FKE}}_{\mathsf{ABS},\mathcal{A}}(k) \leq \mathsf{Adv}^{\mathsf{ANON-FKE}}_{\mathsf{RS},\mathcal{E}}(k) + \mathsf{Adv}^{\mathsf{SEC}}_{\mathsf{PKE},\mathcal{B}}(k)$$

□

6. Performance Analysis and Blockchain Related Issues

In this section, we evaluate the performance of the protocol presented above. A proof of concept has been designed using the Ethereum platform and can be found on a GitHub repository (https://github.com/lepilotedef22/anonymous-sealed-bid-auction, accessed on 18 August 2021). In order to run the protocol on a full blockchain network, the software *Ganache* (https://www.trufflesuite.com/ganache, accessed on 15 November 2020) has been used. It allows to simulate an arbitrary Ethereum network. At the time of the writing of this paper, 30 November 2020, the exchange value of the ether is 589 US dollars and the gas cost is 20 GWei. The gas and US dollar costs of the functions used during the protocol can be found in Table 2.

Table 2. Gas cost and US dollar cost of the Smart Contract functions called during the execution of the protocol, as of 30 November 2020.

Function	Gas	USD
Deployment	1,861,811	21.35
startAuction	87,570	1
placeBid	194,757	2.23
openBid	66,072	0.76
announceWinningCommitment	65,978	0.76
openIdentity	66,322	0.76
withdrawDeposit	34,058	0.39
punishBidder	33,564	0.38

The auction contract code is written in Solidity language and then compiled to EVM byte code and replicated to Ethereum nodes. The smart contract deployment is one of the most expensive tasks (1,861,811 gas units) but since this should be executed only once and can then be reused arbitrarily many times, this is an acceptable overhead. The cost of the function placeBid grows linearly with the number of bidders. This is due to the fact that it takes as an input a ring signature, as well as the ring of users itself, which are both proportional to the number of bidders. This is a practical limitation because if a number of bidders above 15 is reached, it will lead to a gas limit exception. However, a simple solution would be to only store a hash of this data on chain and rely on other off-chain resources, such as for example InterPlanetary File System (IPFS) or Swarm, to actually access the data. More sophisticated solutions will be investigated in future work.

At the current stage of development, the ZKP is still a missing feature. However, it is expected that the overhead linked to this part of the protocol should be close to that of the auction protocol presented by Galal and Youssef [3], since our VERIFIABLEAUCTION algorithm is similar to the one they present. Our new scheme exhibits a major improvement compared to theirs since it enables the bidders to maintain their anonymity by using a DVRS approach. This feature requires a maximum of three interactions with the Smart Contract (bid commitment, bid opening and identity opening for the winning bidder), whereas only two are needed in Galal and Youssef's protocol.

While comparing our proposed work with [12], keeping the same number of ten bidders and same blockchain platform, we demonstrate that the Smart Contract deployment cost of our protocol saves 40%, with respect to deployment cost of 3131261 gas units in [12]. Furthermore, other functions such as openBid, announceWinningCommitment and withdrawDeposit consume less gas while placeBid is marginally more expensive in our case. Therefore we conclude that our proposed protocol is more efficient with additional features.

In Figure 1, the auction time as function of the number of bidders taking part to the auction is displayed. This time grows monotonically with the number of bidders. This can be explained by the fact that the rings formed in order to perform the ring signature that ensures anonymity is built by selecting a random number of parties among the bidders. Therefore, the more bidders, the larger the rings on average, and the longer the computations involving them, as indicated above.

Handling an auction on *blockchain* itself raises special challenges. In an conventional online auction with a centralized auctioneer the confidentiality, integrity and authentication can be achieved with standard security protocols such as SSL/TLS. Furthermore, there are several means to achieve privacy as well such as ToR, VPN and proxies. However these safeguards are not easy to leverage in an blockchain environment.

In a blockchain auction scenario, the bidding participants disclose their identity (fully/partially) and even the bid amount is also visible to others, even to non-participants. The naive solution is to adopt our anonymous auction protocol on the top of an existing privacy preserving blockchain. However, there can still be some hidden practical implementa-

tion challenges as well. For example, a peculiarity which is linked to our proposed protocol is the necessity to publish the list of possible signers, since our scheme relies on RS.

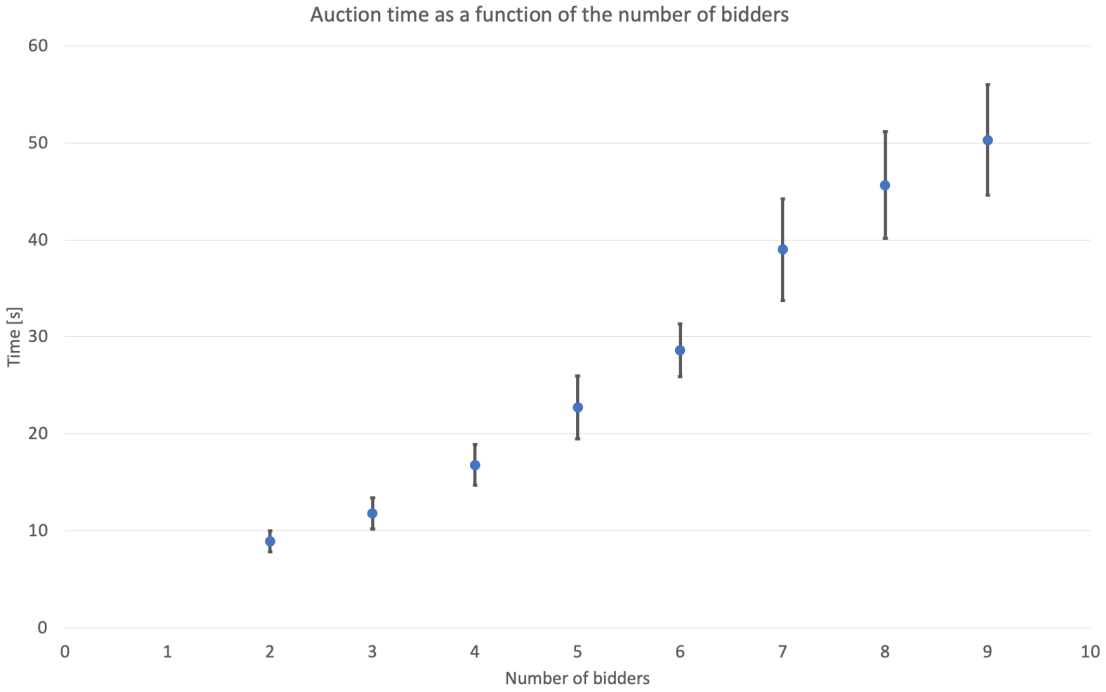

Figure 1. Auction time as a function of the number of bidders. For each number of bidders, a sample of ten tests was simulated. The dots represent the average and the bar the standard deviation.

We strengthen our solution by adopting several blockchain oriented measures, listed below:

- In order to protect the wining bid privacy, we store the commitment of bid on the blockchain;
- The deposit guarantee provides economic incentive to strictly follow the protocol;
- Store the zero-knowledge proof for all the losing bidders, verifiable by anyone.

Instead of relying on a expensive privacy preserving blockchain, the bidders can use pseudonyms to decouple their fixed key pair with freshly generated key pairs (pseudonyms). However, this gives rise to another issue—how to pay the transaction fee for interaction with the Smart Contract. The fund transfer from the bidder's conventional Ethereum account to newly generated key pair will clearly establish a connection between these accounts. Indeed some solutions exist such as (i) registration of fresh key pairs using blind signature by auctioneer and (ii) ZKP to enable privacy preserving authentication, and to hide the fund trail, the auctioneer transfers some funds to bidders for interacting with the Smart Contract [11]. However, these solutions are also not very economical to adapt and require further substantial research.

7. Conclusions and Future Work

We have introduced a generic protocol for anonymous fair trade using only standard cryptographic building blocks. The confidentiality and anonymity properties are achieved using DVRS and the transparency and auditability of blockchain platforms is leveraged in order to make the auction process publicly verifiable. In this paper, we assume the existence of a blockchain facilitating anonymous and confidential transactions. We also analyzed the efficiency of using such cryptographic primitives on Ethereum blockchain and investigate the complexity and vulnerabilities that a blockchain environment might introduce during implementation. In a future work, we intend to further analyze the performance of the designed system on an existing privacy preserving blockchain.

Author Contributions: Conceptualization, G.S.; Supervision, J.-M.D. and O.M.; Writing—original draft, D.V.; Writing, review and editing, V.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dossogne, J.; Markowitch, O. E-voting: Individual verifiability of public boards made more achievable. *WICSITB2010* **2010**, 5–10.
2. Sharma, G.; Verstraeten, D.; Saraswat, V.; Dricot, J.M.; Markowitch, O. Anonymous Fair Auction on Blockchain. In Proceedings of the 2021 11th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Paris, France, 19–21 April 2021; pp. 1–5.
3. Galal, H.S.; Youssef, A.M. Succinctly Verifiable Sealed-Bid Auction Smart Contract. In *Data Privacy Management, Cryptocurrencies and Blockchain Technology*; Springer: Berlin/Heidelberg, Germany, 2018; Volume 11025, pp. 3–19.
4. Hyperledger Fabric. Available online: https://www.hyperledger.org/use/fabric (accessed on 18 August 2021).
5. Hyperledger Fabric Idemix. Available online: https://hyperledger-fabric.readthedocs.io/en/release-2.2/idemix.html (accessed on 18 August 2021).
6. Lu, Y.; Tang, Q.; Wang, G. Zebralancer: Decentralized crowdsourcing of human knowledge atop open blockchain. *arXiv* **2018**, arXiv:1803.01256.
7. Benhamouda, F.; Halevi, S.; Halevi, T. Supporting private data on Hyperledger Fabric with secure multiparty computation. *IBM J. Res. Dev.* **2019**, *63*, 3:1–3:8.
8. Kosba, A.E.; Miller, A.; Shi, E.; Wen, Z.; Papamanthou, C. Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts. In Proceedings of the IEEE Symposium on Security and Privacy, San Jose, CA, USA, 22–26 May 2016; pp. 839–858.
9. Banerjee, A.; Clear, M.; Tewari, H. zkHawk: Practical Private Smart Contracts from MPC-based Hawk. *arXiv* **2021**, arXiv:2104.09180.
10. Sánchez, D.C. Raziel: Private and verifiable smart contracts on blockchains. *arXiv* **2018**, arXiv:1807.09484.
11. Blass, E.O.; Kerschbaum, F. Strain: A Secure Auction for Blockchains. In *ESORICS*; Springer: Berlin/Heidelberg, Germany, 2018; Volume 11098, pp. 87–110.
12. Galal, H.S.; Youssef, A.M. Verifiable Sealed-Bid Auction on the Ethereum Blockchain. In *Financial Cryptography*; Springer: Berlin/Heidelberg, Germany, 2018; Volume 10958, pp. 265–278.
13. Pedersen, T.P. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In *CRYPTO*; Springer: Berlin/Heidelberg, Germany, 1991; Volume 576, pp. 129–140.
14. Lee, M.Z.; Dunn, A.M.; Waters, B.; Witchel, E.; Katz, J. Anon-pass: Practical anonymous subscriptions. In Proceedings of the 2013 IEEE symposium on security and privacy, San Francisco, CA, USA, 23–24 May 2013; pp. 319–333.
15. Bag, S.; Hao, F.; Shahandashti, S.F.; Ray, I.G. SEAL: Sealed-Bid Auction Without Auctioneers. *IEEE Trans. Inf. Forensics Secur.* **2020**, *15*, 2042–2052.
16. Katz, J.; Lindell, Y. *Introduction to Modern Cryptography*; CRC Press/Taylor & Francis: New York, NY, USA, 2015.
17. Rivest, R.L.; Shamir, A.; Tauman, Y. How to Leak a Secret. In *ASIACRYPT*; Springer: Berlin/Heidelberg, Germany, 2001; Volume 2248, pp. 552–565.
18. BBen-Sasson, E.; Chiesa, A.; Tromer, E.; Virza, M. Succinct Non-Interactive Zero Knowledge for a von Neumann Architecture. In Proceedings of the USENIX Security Symposium, San Diego, CA, USA, 20–22 August 2014; pp. 781–796.
19. Saraswat, V.; Pandey, S.K. How to Leak a Secret and Reap the Rewards too. In *LATINCRYPT*; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8895, pp. 348–367.
20. Bünz, A.B.; Agrawal, S.; Zamani, M.; Boneh, D. Zether: Towards Privacy in a Smart Contract World. In *Financial Cryptography*; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12059, pp. 423–443.