Secure Second Price Auctions with a Rational Auctioneer

Boaz Catane and Amir Herzberg
Department of Computer Science, Bar Ilan University, Ramat Gan, 52900, Israel
{*cataneb, herzbea*}*@cs.biu.ac.il*

Keywords: Auctions, Cryptographic Auction Schemes, Cryptographic Protocols, Vickrey Auctions

Abstract: We present novel security requirements for second price auctions and a simple, efficient and practical protocol that provably maintains these requirements. Novel requirements are needed because commonly used requirements, such as the indistinguishability-based secrecy requirement of encryption schemes presented by (Goldwasser and Micali, 1982), do not fit properly in the second price auctions context. Additionally, the presented protocol uses a trustworthy supervisor that checks if the auctioneer deviated from the protocol and fines him accordingly. By making sure the expected utility of the auctioneer when deviating from the protocol is lower than his expected utility when abiding by the protocol we ascertain that a *rational* auctioneer will abide by the protocol. This allows the supervisor to optimize by performing (computationally-intensive) inspections of the auctioneer with only low probability.

1 INTRODUCTION

Various types of auctions are used worldwide (English, Dutch, sealed bid, etc.), each with its own advantages and disadvantages. Special interest has been given to *second*[1] *price sealed bid auctions* (also known as *Vickrey auctions* (Vickrey, 1961)), because of their efficiency and simplicity: each bidder sends only a single sealed bid, and the winning bidder (sender of the maximal bid) pays the amount of the second (or k^{th}) highest bid. The optimal strategy for players is to bid their true valuation of the goods, hence these auctions complete efficiently and securely (assuming rational bidders).

In sealed bid auctions, the auctioneer advertises the auction details, receives sealed bids and declares the winner and the price she[2] has to pay for the goods (the clearing price). Many works on such auctions focus on the actions taken by the bidders and regard the auctioneer as part of the auction mechanism, assuming he abides by the protocol. Because in such sealed bid auctions only the auctioneer sees the bids and declares the auction's outcome, a 'real world' auctioneer may misbehave, e.g. output false results or insert a fictitious bid just below the highest honest bid to deceitfully raise the clearing price. Hence, a mechanism that carries out an auction correctly even in the face of a misbehaving auctioneer is required.

We present a simple and efficient protocol ensuring that a rational auctioneer will not misbehave, i.e., his expected utility from outputting the wrong outcome or dishonestly affecting the outcome will be lower than his expected utility when outputting the auction's true outcome. The protocol is based on a trusted *supervisor*, that (randomly) inspects the outcome reported by the auctioneer; we show that infrequent random inspections are sufficient to ensure that a rational auctioneer will operate correctly. This is significant since the supervisor - in our protocol and other protocols - is an external entity trusted by both bidders and auctioneer, which implies significant processing costs.

1.1 Auctions: Entities and Interests

So far, there were only few real-world applications of secure auction protocols, in spite of the many works and protocols published. One reason for this may be that existing proposals are rather complex, conceptually and computationally. Our goal is to design a simple and practical protocol for secure auctions.

A secure auction protocol would protect the interests of all parties: *bidders, auctioneer* and *owner*. Since these interests may be conflicting, we assume

[1] For simplicity, we focus on second price auctions, but our results apply also to k^{th} price auctions.

[2] For clarity, we use "she", "her", etc. to refer to the bidders, and "he", "his", etc. for other entities.

the parties agree on an additional *trusted party*, whom we refer to as the *supervisor*. The roles and interests of the parties are as follows:

Owner The entity owning goods to be auctioned. The owner wants to receive a maximal price for the goods, and in particular to receive the value of the second-highest bid (minus the fee paid to the auctioneer, if exists).

Bidders Parties that are interested in buying the auctioned goods. They send sealed bids to the auctioneer in order to win the auction. The winning bidder is a bidder that bid the highest bid. The winner pays no more than her bid and receives the goods. Bidder interests include confidentiality of the submitted bids (from other bidders and from the auctioneer, at least while the auction takes place), and integrity of the auction's result (i.e. that the correct (highest bidding) bidder wins and that she is charged the correct amount offered in the second-highest bid).

Auctioneer Manages the auction by receiving bids and outputting the winning bidder and the clearing price. The auctioneer (usually) is a proxy for the *owner* of the goods. Note that the auctioneer may also be interested in purchasing the goods, and may participate in the auction as a bidder by inserting his own bids.

Supervisor An entity that the bidders and owner trust which is used to ensure that the auctioneer operates correctly and does not try to cheat. The auctioneer also trusts the supervisor to operate correctly, e.g. to follow the protocol and allow the auction to complete. In practice, in order for all parties to trust the supervisor he may be implemented using tamper-resistant hardware running attested-software, or using a set of multiple machines operated by different entities for redundancy; both cases imply significant computational and communication costs. To decrease his (amortized) work, the supervisor only checks the auctioneer randomly; we show that this suffices (for a rational auctioneer).

1.2 Contribution

This paper's contributions are:

1. Presentation of novel security requirements appropriate for second price auction schemes. These requirements include *validity, rational correctness, secrecy* and *non-malleability*. The commonly used encryption related security requirements, such as the indistinguishability-based secrecy (Goldwasser and Micali, 1982), do not fit properly in the auctions context since an adversary may legitimately learn some information about bids (see Section 1.4).
2. Presentation of a simple and practical auction scheme that provably maintains the aforementioned requirements. This is achieved using a trusted supervisor that randomly validates the auctioneer's behavior, which enables the protocol to be very efficient.

1.3 Goals and Protocol Overview

Our protocol's goals are threefold: Firstly, keep bids secret from other bidders and from auctioneer until the end of the bidding phase. If bids are not secret then a correct clearing price cannot be guaranteed. A malicious bidder might insert a bid that is only slightly lower than the highest honest bid, resulting in a fictitiously high clearing price. This goal is achieved by using cryptographic tools such as encryption.

Secondly, minimize the trust in the auctioneer. In sealed bid auctions the auctioneer declares the auction's outcome after he alone receives all bids, which are kept secret from all other entities. Such an entity can easily manipulate the auction's results by adding or removing bids or by declaring false outcome. Minimizing the trust in the auctioneer is needed if bidders are to participate in such an auction. This is attained by the presented protocol by modeling the auctioneer as a rational adversary and having a supervisor fine him in case a false outcome is detected. The fine is high enough such that the auctioneer's expected utility when abiding by the protocol is higher than when cheating. Thus, a rational auctioneer will not cheat. The supervisor, as opposed to the auctioneer, is trusted either because he has less motivation to deviate from the protocol (since the bidders do not send him money at any stage of the protocol) or because his computation is done using secure and costly means (e.g. secure hardware or secure multiparty computation).

Thirdly, have a practical and efficient protocol. This goal, which is of utmost importance in our scheme because of the use of costly means to implement a trusted supervisor, is achieved by using the supervisor occasionally, i.e. only with some (small) probability.

Figure 1 shows the high level overview of the presented protocol. In it, the auctioneer publishes the auction details (e.g. item to be auctioned, deadline for bid commitment submission, etc.), and bidders send timed-commitments of their bids to the auctioneer. The commitments are timed such that the auctioneer is able to reveal the value of the bids but not before

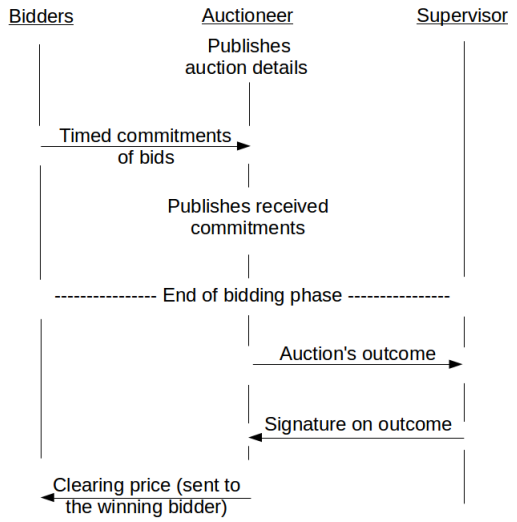

Figure 1: Overview of the presented protocol: Auctioneer publishes auction details. Bidders send timed-commitments of bids to Auctioneer, and Auctioneer publishes them. After the bidding phase Auctioneer sends the computed auction's outcome to Supervisor. After Supervisor signs on the outcome Auctioneer sends clearing price to the winning bidder.

the end of the bidding phase. The auctioneer publishes the received commitments on a bulletin board (e.g. the auctioneer's website). After the end of the bidding phase the auctioneer computes the auction's outcome (i.e. winning bidder and clearing price). The supervisor (possibly) verifies auction's outcome. If he detects that the auctioneer tried to cheat he fines the auctioneer. Otherwise, the supervisor signs the outcome and sends his signature to the auctioneer. The auctioneer then sends the supervisor-signed outcome to the winning bidder.

Loosely speaking, the protocol ensures it will not be beneficial for the auctioneer to cheat, i.e. deviate from the protocol. Auctioneer's cheating may include influencing the auction's outcome in a variety of ways: First, he can ignore bid commitments. Second, the auctioneer can declare arbitrary false results (wrong winning bidder or wrong clearing price). Third, the auctioneer can use knowledge about received bids to add or remove specific bids. This may influence the auction's outcome in different ways:

1. After opening the commitments and seeing all bids the auctioneer might insert a bid that is only slightly lower than the highest honest bid. This will make the winning bidder pay a fictitiously high clearing price.
2. The auctioneer might insert in advance many different bids. After seeing the honest bids he might remove bids that he inserted and that are higher than the highest honest bid but leave the rest, again making the winning bidder pay a fictitiously high clearing price.

The protocol ensures that deviation from the protocol would not be the auctioneer's rational move.

Notice that some actions taken by the auctioneer to influence the auction's outcome are legitimate, and do not result in a fine; in our formal modeling of utilities these actions are legitimate and do not harm other parties, although arguably in a practical deployment some of these actions may be deemed inappropriate. Firstly, the auctioneer may insert (himself or by a collaborating bidder) a commitment for a bid, and in case his bid is the highest bid he wins the auction. This effectively introduces a minimal price for the goods (i.e. inserting such a bid ensures that the goods will not be sold for a price lower than the auctioneer's bid). This is a legitimate strategy and we do not try to prevent it. Secondly, in case of a tie (e.g. a few bidders bid the highest bid) the auctioneer may choose a winner arbitrarily. Such actions are not considered a fraud since no bidder is harmed: The winning bidder pays an amount equal to her bid and receives the goods which she values as equal to the paid price. Likewise, other bidders do not pay or receive anything and so no damage is inflicted upon them. In reality, there may be some objection to this modeling, e.g., since bidders may have preferred to bid elsewhere, but this is not captured by the usual modeling of utilities and will require further research.

1.4 Related Work

Auctions are a well studied subject; few of the many good references to this area include (Engelbrecht-Wiggans, 1980; Milgrom and Weber, 1982; Klemperer, 2004). Special interest is given to second price auctions (also known as Vickrey auctions) (Vickrey, 1961), in which the bidders' optimal strategy is to bid their true value of the auctioned goods.

Much work on the use of cryptography for conducting secure auctions has focused on the goal of *complete privacy*, where no one (including the auctioneer) learns information about the bids even after the auction has ended (e.g. (Harkavy et al., 1998; Naor et al., 1999)); unfortunately, these solutions have high computational requirements. Our protocol is much more efficient and hence more practical, although allowing the auctioneer and supervisor to learn the values of the bids (but only after the end of the bidding phase).

Some protocols achieve complete privacy by bidder-resolved multi-party computation (Brandt,

2006). However, in many cases privacy is achieved by using third parties, either through numerous auctioneers or by asymmetric models in which in addition to the auctioneer the entity of a *supervisor* (or *auction issuer*) is assumed (Naor et al., 1999; Lipmaa et al., 2003). Parkes et al. (Parkes et al., 2008) settle for verifiable correctness and trustworthiness in combination with complete secrecy to all parties except the auctioneer. However, their scheme demands considerable time for preparation and verification of auctions.

In contrast to the aforementioned work, we define and require novel security requirements to capture the notions of correctness, secrecy, and non-malleability in the auctions context. Past works did not formally define these requirements or even required all of them, e.g. non-malleability, although using malleable encryption for preserving bid secrecy can be disastrous, as shown by Dolev et al. (Dolev et al., 1991). In addition, close observation reveals that the indistinguishability based secrecy definition (Goldwasser and Micali, 1982) commonly used for encryption schemes is not fit for use in auctions, since an attacker may distinguish between encrypted messages and win the cryptographic experiment by using legitimately learned information about encrypted bids (i.e. the identity of the winning bidder or the clearing price). Thus, we settle for complete secrecy for all bidders (but not the auctioneer or supervisor) and non-malleability of bids, and analytically prove correctness. This is achieved by using a supervisor and assuming the auctioneer will deviate from the protocol if and only if it maximizes his utility function.

In the following protocol we use timed commitments. The general notion of timed cryptography (e.g. an encrypted message that can only be decrypted after a predetermined amount of time has passed) was first introduced by (Rivest et al., 1996). Timed-commitments, a commitment scheme in which there is an optional forced opening phase enabling the receiver to recover (with effort) the committed value without the help of the sender, were later presented by (Boneh and Naor, 2000). Although their scheme is sound, it has considerable computational overhead. We note that our scheme can be adapted to use their cryptographic primitive, but for efficiency reasons we implement timed-commitments using a Time Lapse Cryptography (TLC) Service that is similar to the one presented in (Rabin and Thorpe, 2006). This TLC service provides a commitment scheme which, in addition to the general hiding and binding properties of commitments, ensures that the committed value can be revealed at an exact time in the future even without the help of the committing party.

A privacy and efficiency comparison of various protocols is presented in Table 1. For each protocol we count the (average) number of modular exponentiations computed by each entity.

2 PRELIMINARIES

2.1 Model

In the *supervised auction* model there are n bidders participating in an online sealed bid second price auction with an untrusted auctioneer. Each bidder ψ has a private valuation v_ψ for the goods being auctioned, and each sends her bid b_ψ to the auctioneer. Note that in a second price auction bidding one's true valuation is the optimal strategy (i.e. $b_\psi = v_\psi$). The auctioneer should output (ψ_{win}, p) where ψ_{win} is the winning bidder (i.e. a bidder that bid the highest bid) and p is the amount she has to pay (i.e. the clearing price, which is equal to the second highest bid). If it maximizes his utility function, the auctioneer might output different values than the true values of (ψ_{win}, p). In order to prevent him from deviating from the protocol and outputting false results a trusted third party, a supervisor, checks the auction's outcome, settles disputes, and fines the auctioneer (if needed).

2.2 Time Lapse Cryptography Service

In the presented protocol we use timed-commitments for hiding the bid values until the end of the bidding phase. Implementation of timed-commitments can be done using cryptographic methods, as presented by (Boneh and Naor, 2000). For efficiency reasons, our protocol uses a Time Lapse Cryptography (TLC) Service resembling the one presented by Rabin and Thorpe (Rabin and Thorpe, 2006). The TLC service provides a cryptographic timed-commitment protocol that enables the use of commitments with the classical hiding and binding properties. In addition, it prevents bidders from refusing to open committed bids and also prevents the auctioneer from dropping received commitments after he published them, claiming not to have been able to open the committed bids. The supervisor, acting as the TLC service provider, publishes a public key of a non-malleable encryption scheme before the auction begins, and sends the corresponding private key only when no new bids can be sent (i.e. after the bid submission deadline).

Whenever timed commitments are used in the protocol it is to say that a bidder encrypts her bid using the supervisor-generated public encryption key. This

Table 1: Comparison of various secure auction schemes. Legend: n - Number of bidders; α - Probability that a third party verifies auctioneer's output; l - Maximum number of bits needed to represent a single bid; k - A constant used in (Parkes et al., 2008).

Protocol	**Bids Privacy Kept**	**No. of Modular Exponentiations**		
		Single Bidder	Auctioneer	Third Party
Boneh and Naor (Boneh and Naor, 2000)	None	1	0	N/A
Our protocol	From other bidders	2	$2n+1$	$2+\alpha(n-1)$
Parkes et al. (Parkes et al., 2008)	From other bidders	$n+5$	$kn+3$ (typically $k>5400$)	n
Lipmaa et al. (Lipmaa et al., 2003)	From bidders and Aucioneer. Third party learns bid statistics	$O(l)$	$O(2^l)$	$O(l)$
Naor et al. (Naor et al., 1999)	From all entities	$O(l)$	$O(nl)$	$O(nl)$
Juels and Szydlo (Juels and Szydlo, 2003)	From all entities	2 (+ $O(l)$ modular multiplications)	$O(nl)$	$O(nl)$

encrypted bid is to be opened later by the auctioneer after receiving the corresponding decryption key.

2.3 Notation

If A is a probabilistic polynomial time (p.p.t) algorithm that runs on input x, then $A(x)$ denotes the random variable corresponding to the output of A on input x and uniformly random coins. In addition, we denote computational indistinguishability (Goldwasser and Micali, 1984) of ensembles A and B by $A \stackrel{c}{\approx} B$.

We will need to discuss vectors of values. A vector is denoted in bold font, as in $\mathbf{x}$. We denote by $|\mathbf{x}|$ the number of components in $\mathbf{x}$, and by $\mathbf{x}.i$ the i-th component, so that $\mathbf{x} = (\mathbf{x}.1, \ldots, \mathbf{x}.|\mathbf{x}|)$. We extend set membership notation to vectors, writing e.g. $x \in \mathbf{x}$ to mean that x is in the set $\{\mathbf{x}.i : 1 \leq i \leq |\mathbf{x}|\}$. We also extend the notation for algorithms with variables as input to accept also vectors as input with the understanding that operations are performed componentwise. Thus if A is an algorithm then $\mathbf{x} \leftarrow \mathrm{A}(\mathbf{y})$ is shorthand for the following: for $1 \leq i \leq |\mathbf{y}|$ do $\mathbf{x}.i \leftarrow \mathrm{A}(\mathbf{y}.i)$.

We will consider relations of arity d where d will be polynomial in the security parameter k. Rather than writing $R(x_1, \ldots, x_d)$ we write $R(x, \mathbf{x})$ meaning that the first argument is special and the rest are bunched into a vector $\mathbf{x}$ with $|\mathbf{x}| = d-1$.

Regarding the auctioneer's utility we use the following notations:

U denotes the auctioneer's utility when abiding by the protocol. This utility comprises the salary he receives for functioning as an auctioneer plus any rightfully won auction gains (in case he bid and won the auction).

U_σ^+ denotes, for a given cheating (i.e. deviating from the protocol) strategy σ, the auctioneer's utility when playing σ and not being caught.

U_σ^- denotes, for a given cheating strategy σ, the auctioneer's utility when playing σ and being caught.

α denotes the probability that the auctioneer's cheating will be caught (i.e. the probability that the supervisor will check the auction's outcome).

Note that the auctioneer's expected utility when not abiding by the protocol and playing some cheating strategy σ is

$$(1-\alpha)U_\sigma^+ + \alpha U_\sigma^- \tag{1}$$

Hence, abiding by the protocol would maximize his expected utility iff for every cheating strategy σ the following holds:

$$U \geq (1-\alpha)U_\sigma^+ + \alpha U_\sigma^- \tag{2}$$

That is, a rational auctioneer would not cheat if his expected utility when abiding by the protocol is greater than his expected utility when deviating from it.

2.4 Syntax of Auction Schemes

An *auction scheme* $\mathcal{AUC}$ = (KG, Bidder, Decode, Auctioneer, Supervisor, Winner) consists of six polynomial-time algorithms:

- The probabilistic key generation algorithm *KG* takes as input an entity $\psi \in \{1,\ldots,n\} \cup \{\text{A}, \text{S}\}$ and a sting 1^k, where $k \in \mathbb{N}$ is the security parameter, and returns a (*Public key, Private key*) pair.
- The probabilistic bidding algorithm *Bidder* takes the following as input: As local input, Bidder receives a (*Public key, Private key*) pair and a numeric bid value *bid*. Bidder also receives public information (that may consist of the auctioneer's public key, details about the planned auction, etc.). Additional information (such as the supervisor's public key) is received off-band. Bidder then outputs a vector ***message*** that consists of ***message***.*encoded_bid* and ***message***.*id* such that ***message***.*encoded_bid* encodes *bid* while ***message***.*id* contains additional information (bidder's identification information, commitment to pay for the item in case the bidder won the auction, etc.). Generating ***message***.*encoded_bid* may require *bid* and a secret key. In addition, the encoding string may hide the numeric value of *bid* (i.e. may be the output of an encryption process). The encoding is reversible, namely *bid* can be retrieved from ***message***.*encoded_bid* by the Decode algorithm (see below).
- The deterministic decoding algorithm *Decode* takes as input a key *Dec* and a string *encoded_bid* and outputs the numeric value *bid* that was used by Bidder to generate *encoded_bid*, or $\perp$ if no such *bid* exists. Formally, Decode outputs *bid* such that for any *auction_details*, bidder's public and secret keys (pk, sk), numeric bid value *bid*, auctioneer's public key A.*pk*, and supervisor's public key S.*pk* and key pair (*Enc*, *Dec*), if (***message***.*encoded_bid*, ***message***.*id*) $\leftarrow$ Bidder(*auction_details*, S.*pk*, A.*pk*, *pk*, *sk*, *Enc*, *bid*) then *bid* $\leftarrow$ Decode(*Dec*, ***message***.*encoded_bid*), otherwise $\perp \leftarrow$ Decode(*Dec*, ***message***.*encoded_bid*).

  We note that the Decode algorithm is mainly for defining and proving the security requirements (see Sections 2.5 and 4), and does not necessarily need to be implemented in a real auction.
- The probabilistic auctioneering algorithm *Auctioneer* takes as local input a (*Public key*, *Private key*) pair. Furthermore, Auctioneer receives public keys (such as the bidders' or supervisor's), and may receive additional public or private information (e.g. encoded bids) and a stage $stg \in \{$'init', 'receive', 'outcome'$\}$. If $stg =$ 'init' the algorithm outputs details about a planned auction. If $stg =$ 'receive' the algorithm outputs state information to be used later. If $stg =$ 'outcome' the algorithm outputs an auction's outcome (i.e. the winning bidder and the clearing price).
- The probabilistic auction supervising algorithm *Supervisor* takes as input a (*Public key, Private key*) pair, public information (that may consist of the auctioneer's public keys), additional input from the auctioneer (such as auction details or outcome), a probability α, and a stage $stg \in \{$'sign', 'generate', 'verify'$\}$. If $stg =$ 'sign' the algorithm outputs a signature on the auction details input. If $stg =$ 'generate' the algorithm outputs a (*Public key, Private key*) pair of a non-malleable CPA-secure encryption scheme. If $stg =$ 'verify' it outputs either the string 'Verified outcome' or a proof that the auctioneer cheated and an amount *fine* the auctioneer needs to pay.
- The deterministic winner finding algorithm *Winner* takes as input ***bids***, a vector of bidders' bids and ***messages***, a vector containing outputs of the Bidder algorithm (namely, a vector of (*encoded_bid*, *id*) pairs) such that bidder *i*'s bid is ***bids***.*i*, and her encoded bid and identity information is ***messages.i***.*encoded_bid* and ***messages.i***.*id*, respectively. The algorithm outputs the winning bidder ψ_{winner} according to the auction rules (i.e. ψ_{winner} is the bidder that bid the highest bid. In case more than one bidder bid the highest bid the algorithm outputs the bidder whose encoded bid has lexicographic precedence).

2.5 Requirements

An auction scheme is required to be *secure*, as defined below.

Definition 1. *An auction scheme is* secure *iff it is* valid*,* correct for a rational auctioneer*,* preserves secrecy*, and* non-malleable *(as detailed below).*

2.5.1 Validity

An auction scheme is said to be *valid* if, when the supervisor checks the auction's outcome, in case the auctioneer deviated from the protocol he will be caught with overwhelming probability. Formally, for any auction scheme $\mathcal{AUC} =$ (KG, Bidder, Decode, Auctioneer, Supervisor, Winner), adversary Adv and $k \in \mathbb{N}$ we associate Experiment 1. In it, Adv has four stages: *init, details, encode* and *cheat*. In the *init* stage Adv is given a unary string 1^k and outputs $n \in \mathbb{N}$, the number of participating bidders. In the *details* stage, after a public and private key pair was issued to each entity (the bidders, the auctioneer and the supervisor),

Adv receives the secret keys of all bidders[3] along with all public keys. He then outputs an *auction_details* string that defines the auction. Possible details may be the item to be auctioned, maximum allowed bid, etc. Later, in the *encode* stage, Adv is invoked on behalf of the bidders and is given the public key *Enc* generated by the supervisor. He outputs a vector ***messages*** as the output of all bidders which is comprised of n (bid encodings, identification string) pairs. Adv is invoked again, this time as the auctioneer, in the *cheat* stage. He is given the key *Dec* and outputs the encoded bid of the winning bidder *winning_encoded_bid* and a clearing price p as the auction's outcome. The Supervisor is then given his private and public keys along with the auction details, all encoded bids, the declared outcome (*winning_encoded_bid*, p), and 1 as the probability to validate this auction, and outputs a *verification* string. If Adv deviated from the protocol (i.e. *winning_encoded_bid* is not the encoded bid of the winning bidder or p is not the second highest bid) while *verification* = 'Verified outcome' then the experiment outputs 1, Otherwise it outputs 0. Note that in addition to the above, in the *details* and *encode* stages Adv outputs state information to be used later.

The advantage of adversary Adv in breaking the validity of $\mathcal{AUC}$ is denoted by

$$\mathbf{Advantage}^{\text{Validity}}_{\mathcal{AUC},\,\text{Adv}}(k) = \Pr[\mathbf{Exp}^{\text{Validity}}_{\mathcal{AUC},\,\text{Adv}}(k) = 1] \quad (3)$$

Definition 2. *An auction scheme $\mathcal{AUC}$ is* valid *if for any polynomial-time adversary Adv and $k \in \mathbb{N}$ the function* $\mathbf{Advantage}^{\text{Validity}}_{\mathcal{AUC},\,\text{Adv}}(k)$ *is negligible.*

2.5.2 Rational Correctness

An auction scheme is *correct for a rational auctioneer* if a rational auctioneer will not deviate from the protocol, i.e. his expected utility when deviating from the protocol is not greater than when abiding by it. Formally, for every auctioneer's strategy σ Equation 2 should hold (See Section 2.3).

2.5.3 Secrecy

An auction scheme *preserves secrecy* if no colluding set of bidders is able to learn any nonessential information about other bids even after the auction is over. Consider two adversarially-chosen bidders (ψ_0 and ψ_1) which are assigned two adversarially-chosen bids (bid_0 and bid_1). A colluding group of (other) bidders should not be able to tell which of ψ_0 and ψ_1 bade either of the two bids with probability significantly better than guessing.

[3] Although it may be sufficient to give Adv the secret keys of *colluding* bidders only, we simplify this experiment, as well as Experiments 2 and 3, by giving him the secret keys of *all* bidders, as done in other works (e.g. (Bellare et al., 2003)).

Experiment 1 $\text{Exp}^{\text{Validity}}_{\mathcal{AUC},\,\text{Adv}}(k)$

$n \leftarrow \text{Adv}(\text{'init'}, 1^k)$
for each $\psi \in \{1,\ldots,n\} \cup \{\text{A}, \text{S}\}$ **do** $(\psi.\text{pk}, \psi.\text{sk}) \leftarrow \text{KG}(\psi, 1^k)$
$(auction\_details,\ St_1) \leftarrow \text{Adv}(\text{'details'},\ \text{S}.pk,\ \text{A}.pk,\ \text{A}.sk,\ 1.pk, 1.sk, \ldots, n.pk, n.sk)$
$(Enc, Dec) \leftarrow \mathcal{AUC}.\text{Supervisor}(\text{'generate'},\ auction\_details,\ \text{S}.pk,\ \text{S}.sk,\ \text{A}.pk,\ 1^k)$
$(\boldsymbol{messages},\ St_2) \leftarrow \text{Adv}(\text{'encode'},\ St_1,\ Enc) \triangleright |\boldsymbol{messages}| = n$
$(winning\_encoded\_bid,\ p) \leftarrow \text{Adv}(\text{'cheat'},\ St_2,\ Dec)$
$verification \leftarrow \mathcal{AUC}.\text{Supervisor}(\text{'verify'},\ auction\_details,\ \text{S}.pk,\ \text{S}.sk,\ \boldsymbol{messages.1}.encoded\_bid,\ \ldots,\ \boldsymbol{messages.n}.encoded\_bid,\ Dec,\ winning\_encoded\_bid,\ p,\ 1)$
for each $i \in \{1,\ldots,n\}$ **do**
    $\boldsymbol{bids}.i \leftarrow \mathcal{AUC}.\text{Decode}(Dec,\ \boldsymbol{messages.i}.encoded\_bid)$
$\psi_{\text{winner}} \leftarrow \mathcal{AUC}.\text{Winner}(\boldsymbol{bids}, \boldsymbol{messages})$
if $\big(winning\_encoded\_bid \neq \boldsymbol{messages}.\psi_{\text{winner}}.encoded\_bid \vee p \neq \text{second\_highest}(\boldsymbol{bids})\big) \wedge verification = \text{'Verified outcome'}$ **then return** 1
else return 0

Formally, for any auction scheme $\mathcal{AUC}$ = (KG, Bidder, Decode, Auctioneer, Supervisor, Winner), adversary Adv, $k \in \mathbb{N}$ and bit b we associate Experiment 2. In it, Adv has five stages: *init, details, choose, encode* and *guess*. In the *init* stage Adv outputs n, the number of participating bidders, s.t. $n > 1$. In the *details* stage, after each entity (the bidders, the auctioneer and the supervisor) received a public and private key pair, Adv receives the secret keys of all bidders and all public keys. He then outputs state information St_1 and an *auction_details* string that defines the auction. Possible details may be the item to be auctioned, maximum allowed bid, and so on. Later, in the *choose* stage, Adv is given the public key *Enc* generated by the supervisor and outputs state information St_2 along with two special bidders $\psi_0, \psi_1 \in \{1,\ldots,n\}$ and two bids bid_0, bid_1 for these bidders. For each special bidder ψ_i the following takes place: the Bidder algorithm is invoked with bidder ψ_i's keys and bid $bid_{b \oplus i}$. Additionally, the Auctioneer algorithm receives Bidder's

output and saves state information in *state*. Later, for each non-special bidder i Adv is invoked in stage *encode*. Adv receives St_2, i, and the encoded bids of the two special bidders and outputs a vector ***message*** for that bidder. The Auctioneer algorithm then receives the vector ***message*** and outputs *state*. After all messages were received by Auctioneer he is invoked in stage *outcome* with *state* and *Dec* as input and outputs the auction's outcome, namely the winning bidder ψ_{winnder} and the clearing price p. In the *guess* stage, if the winning bidder is one of the two special bidders then Adv receives state information St_2 only. Otherwise, if the winning bidder is non-special then Adv additionally receives the winning bidder's name and the clearing price. In either case Adv then outputs his guess, a bit b'. If $b' = b$ then the experiment's output is 1, otherwise it returns 0.

Experiment 2 $\text{Exp}^{\text{Secrecy-b}}_{\mathcal{AUC},\,\text{Adv}}(k)$

1: $n \leftarrow \text{Adv}(\text{'init'}, 1^k)$ ▷ $n > 1$
2: **for each** $\psi \in \{1,\ldots,n\} \cup \{\text{A}, \text{S}\}$ **do** $(\psi.\text{pk}, \psi.\text{sk}) \leftarrow \text{KG}(\psi, 1^k)$
3: $(\textit{auction\_details}, St_1) \leftarrow \text{Adv}(\text{'details'}, \text{S}.pk, \text{A}.pk, 1.pk, 1.sk, \ldots, n.pk, n.sk)$
4: $(Enc, Dec) \leftarrow \mathcal{AUC}.\text{Supervisor}(\text{'generate'}, \textit{auction\_details}, \text{S}.pk, \text{S}.sk, \text{A}.pk, 1^k)$
5: $(bid_0, bid_1, \psi_0, \psi_1, St_2) \leftarrow \text{Adv}(\text{'choose'}, St_1, Enc)$ ▷ $|bid_0| = |bid_1|$, $\psi_0, \psi_1 \in \{1,\ldots,n\}$, $\psi_0 < \psi_1$
6: $state \leftarrow \perp$
7: **for each** $i \in \{0,1\}$ **do**
8: $\quad \boldsymbol{messages}.\psi_i \leftarrow \mathcal{AUC}.\text{Bidder}(\textit{auction\_details}, \text{S}.pk, \text{A}.pk, \psi_i.pk, \psi_i.sk, Enc, bid_{b\oplus i})$
9: $\quad state \leftarrow \mathcal{AUC}.\text{Auctioneer}(\text{'receive'}, \textit{auction\_details}, \text{S}.pk, \text{A}.pk, \text{A}.sk, 1.pk, \ldots, n.pk, \boldsymbol{messages}.\psi_i, state)$
10: **for each** $i \in \{1,\ldots,n\} \setminus \{\psi_0, \psi_1\}$ **do**
11: $\quad (\boldsymbol{messages.i}, St_2) \leftarrow \text{Adv}(\text{'encode'}, St_2, \boldsymbol{messages}.\psi_0.\textit{encoded\_bid}, \boldsymbol{messages}.\psi_1.\textit{encoded\_bid}, i)$
12: $\quad state \leftarrow \mathcal{AUC}.\text{Auctioneer}(\text{'receive'}, \textit{auction\_details}, \text{S}.pk, \text{A}.pk, \text{A}.sk, 1.pk, \ldots, n.pk, \boldsymbol{messages}.\psi_i, state)$
13: $(\psi_{\text{winner}}, p) \leftarrow \mathcal{AUC}.\text{Auctioneer}(\text{'outcome'}, state, Dec)$
14: **if** $\psi_{\text{winner}} \in \{\psi_0, \psi_1\}$ **then** $b' \leftarrow \text{Adv}(\text{'guess'}, St_2)$
15: **else** $b' \leftarrow \text{Adv}(\text{'guess'}, St_2, \psi_{\text{winner}}, p)$
16: **if** $b' = b$ **then return** 1
17: **else return** 0

The advantage of adversary Adv in breaking the secrecy of $\mathcal{AUC}$ is denoted by

$$\textbf{Advantage}^{\text{Secrecy-b}}_{\mathcal{AUC},\,\text{Adv}}(k) = \left|\Pr[\textbf{Exp}^{\text{Secrecy-b}}_{\mathcal{AUC},\,\text{Adv}}(k) = 1] - \Pr[\textbf{Exp}^{\text{Secrecy-b}}_{\mathcal{AUC},\,\text{Adv}}(k) = 0]\right| \tag{4}$$

Definition 3. *An auction scheme $\mathcal{AUC}$* preserves secrecy *if for any polynomial-time adversary Adv, $k \in \mathbb{N}$ and bit b the function* $\textbf{Advantage}^{\textit{Secrecy-b}}_{\mathcal{AUC},\,\textit{Adv}}(k)$ *is negligible.*

2.5.4 Non-Malleability

Informally, non-malleability requires that an attacker, after receiving an encoding of some bid, cannot modify it into an encoding of a different bid whose "meaningfully related" to the original bid. This requirement ensures that a set of colluding bidders and the auctioneer cannot insert bids whose values depend on bids of honest bidders. Following the work of Pass et al. (Pass et al., 2006), we present this non-malleability requirement using an indistinguishability based experiment.

Formally, for any auction scheme $\mathcal{AUC} = (\text{KG}, \text{Bidder}, \text{Decode}, \text{Auctioneer}, \text{Supervisor}, \text{Winner})$, adversary Adv and $k, l \in \mathbb{N}$ we associate Experiment 3. In it, Adv has four stages: *init, details, choose* and *guess*. In the *init* stage Adv outputs the number of bidders that will participate in the auction. A public and secret key pair is then generated and given to each entity. In the *details* stage Adv receives the supervisor's public key along with the public and secret keys of the auctioneer and all bidders (to capture the possibility of an adversary colluding with both the auctioneer and bidders). He then outputs the auction details of his choice and state information St_1. The supervisor is then invoked and is given the auction details, his public and private keys and the auctioneer's public key, and outputs a newly generated key pair (Enc, Dec). Afterwards, in the *choose* stage, Adv is given Enc and St_1 and is required to output a bidder ψ, two bids bid_0 and bid_1 and state information St_2. The Bidder algorithm is then invoked with *auction_details*, the supervisor's and auctioneer's public keys, bidder ψ's public and private keys, Enc and bid_b as input and outputs a ($encoded\_bid$, id) pair. In the *guess* stage Adv receives St_2 along with bidder ψ's encoded bid and outputs a vector of length l containing encoded strings. The experiment outputs a corresponding vector containing, for each encoding c_i, the symbol $\perp$ if c_i is identical to the challenge *encoded_bid*, or a decoding of c_i (using the Decode algorithm) otherwise. Adv is *successful* if the vector returned by the experiment is computationally distinguishable when $b = 0$ compared to when $b = 1$.

Experiment 3 $\text{Exp}^{\text{Non-Mal-b}}_{\mathcal{AUC},\,\text{Adv}}(k,l)$

$n \leftarrow \text{Adv}(\text{'init'}, 1^k)$
for each $\psi \in \{1,\ldots,n\} \cup \{\text{A}, \text{S}\}$ **do** $(\psi.\text{pk}, \psi.\text{sk}) \leftarrow \text{KG}(\psi, 1^k)$
$(auction\_details,\ St_1) \leftarrow \text{Adv}(\text{'details'}, \text{S}.pk, \text{A}.pk, \text{A}.sk, 1.pk, 1.sk, \ldots, n.pk, n.sk)$
$(Enc,\ Dec) \leftarrow \mathcal{AUC}.\text{Supervisor}(\text{'generate'}, auction\_details, \text{S}.pk, \text{S}.sk, \text{A}.pk, 1^k)$
$(\psi,\ bid_0,\ bid_1,\ St_2) \leftarrow \text{Adv}(\text{'choose'}, Enc, St_1)$ ▷ $|\text{bid}_0| = |\text{bid}_1|$
$(encoded\_bid,\ id) \leftarrow \mathcal{AUC}.\text{Bidder}(auction\_details, \text{S}.pk, \text{A}.pk, \psi.pk, \psi.sk, Enc, bid_b)$
$(c_1, \ldots, c_l) \leftarrow \text{Adv}(\text{'guess'}, St_2, encoded\_bid)$
return $(d_1, \ldots, d_l)$ where $d_i = \begin{cases} \perp & \text{if } c_i = encoded\_bid \\ \mathcal{AUC}.\text{Decode}(c_i) & \text{otherwise} \end{cases}$

Definition 4. *Let $\mathcal{AUC}$ be an auction scheme and let the random variable $Exp^{Non\text{-}Mal\text{-}b}_{\mathcal{AUC},\,Adv}(k,l)$ where $b \in \{0,1\}$, Adv is an adversary algorithm and $k,l \in \mathbb{N}$ denote the result of the Experiment 3. $\mathcal{AUC}$ is* non-malleable *if for any p.p.t algorithm Adv and for any polynomial $p(k)$, the following two ensembles are computationally indistinguishable:*

$$\left\{Exp^{Non\text{-}Mal\text{-}0}_{\mathcal{AUC},\,Adv}(k,p(k))\right\}_{k\in\mathbb{N}} \overset{c}{\approx} \left\{Exp^{Non\text{-}Mal\text{-}1}_{\mathcal{AUC},\,Adv}(k,p(k))\right\}_{k\in\mathbb{N}} \quad (5)$$

3 THE PROTOCOL

3.1 Assumptions

In the presented protocol we assume the following:

- All entities have signature key pairs. Public keys are known to all.
- All entities have synchronized clocks.
- The auctioneer has a certified bulletin board (such as a website) to post public information.
- Communication delays for all messages are at most Δ.

In addition to the above, the protocol uses an encryption scheme that is non-malleable with respect to chosen-plaintext attacks. We use the indistinguishability-based definition of such non-malleability, as presented by Pass et al. (Pass et al., 2006). To facilitate readability of the protocol and forthcoming proofs we bring the definition below.

3.1.1 Indistinguishability-based Non-Malleability Definition

In their paper, given an adversary $\text{Adv} = (\text{Adv}_1, \text{Adv}_2)$ and $k,l \in \mathbb{N}$, Pass et al. present an indistinguishability based definition of non-malleability for an encryption scheme Π under a chosen-plaintext attack, as shown in Definition 5. Note that we have changed some variable names for the reader's convenience.

Definition 5 (from (Pass et al., 2006))**.** *Let Π = (KG, Encrypt, Decrypt) be an encryption scheme and let the random variable $Exp^{Non\text{-}Mal\text{-}b}_{\Pi,\,Adv}(k,l)$ where $b \in \{0,1\}$, $Adv = (Adv_1, Adv_2)$ and $k,l \in \mathbb{N}$ denote the result of Experiment 4. Π is non-malleable under a chosen-plaintext attack if for every p.p.t. algorithm $Adv = (Adv_1, Adv_2)$ and every polynomial $p(k)$, the following two ensembles are computationally indistinguishable:*

$$\left\{Exp^{Non\text{-}Mal\text{-}0}_{\Pi,\,Adv}(k,p(k))\right\}_{k\in\mathbb{N}} \overset{c}{\approx} \left\{Exp^{Non\text{-}Mal\text{-}1}_{\Pi,\,Adv}(k,p(k))\right\}_{k\in\mathbb{N}} \quad (6)$$

Experiment 4 $\text{Exp}^{\text{Non-Mal-b}}_{\mathcal{AUC},\,\text{Adv}}(k,l)$

$(Enc, Dec) \leftarrow \text{KG}(1^k)$
$(bid_0,\ bid_1,\ state) \leftarrow \text{Adv}_1(Enc,\ 1^k)$ ▷ $|bid_0| = |bid_1|$
$y \leftarrow \text{Encrypt}_{Enc}(bid_b)$
$(c_1, \ldots, c_l) \leftarrow \text{Adv}_2(state, y)$
return $(d_1, \ldots, d_l)$ where $d_i = \begin{cases} \perp & \text{if } c_i = y \\ \text{Decrypt}_{Dec}(c_i) & \text{otherwise} \end{cases}$

3.2 The Protocol

Below are details of the presented protocol, as depicted in Figure 2.

1. The auctioneer sends auction details to the supervisor and asks him to participate in the auction. The auctioneer may pay the supervisor for his services. Auction details include: details of the auctioned goods; maximum allowed bid b_{maxAl}; legal commitment by the auctioneer to pay a fine of value *fine* in case the supervisor presents proof that the auctioneer deviated from the protocol; bid commitment submission deadline t_{sd}; maximum number of bidders n_{max}; auction ending time t_{end} (see Section 3.2.1 for detailed computation of the auction ending time).

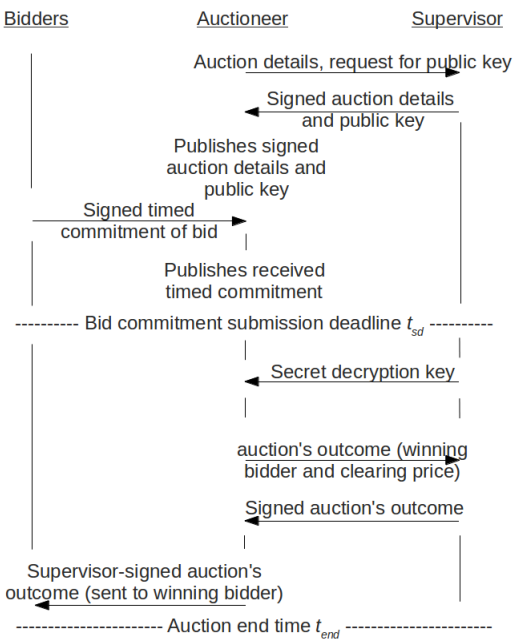

Figure 2: Outline of the protocol. Auctioneer asks Supervisor to participate in the auction and publishes auction details. Bidders send timed-commitments of bids to Auctioneer. Supervisor helps Auctioneer open bids and compute auction's outcome, and then Supervisor (possibly verifies and) signs on the outcome. Auctioneer sends clearing price to winning bidder.

2. If the supervisor accepts the auction details he arbitrarily sets α such that

$$\alpha \geq \frac{b_{maxAl}}{fine + b_{maxAl}} \tag{7}$$

and sends the auctioneer a public key of a CPA-secure non-malleable encryption scheme, along with the supervisor's signature on the key and the auction details.

3. The auctioneer publishes the signed auction details and public key.

4. Each bidder i sends the auctioneer an encryption of her bid. The bidder also sends a legal commitment to pay for the goods (a price not higher than her committed bid) in case the auctioneer presents, no later than the auction ending time t_{end}, a supervisor-signed statement saying that this bidder won the auction and needs to pay such and such as clearing price. The message is signed by the bidder's private signing key.

5. After receiving each message the auctioneer verifies both that the bidder's signature is valid and that an identical encryption was not published earlier on the bulletin board (otherwise the auctioneer ignores the message). He then publishes the encrypted bid on the bulletin board.

   In case a bidder detects that her encrypted bid was not published she will resend it to the supervisor, who in turn, after verifying the bidder's signature and that such an encrypted bid was not published already, will forward it to the auctioneer. If the auctioneer ignores encrypted bids forwarded to him by the supervisor then the supervisor will stop participating in the auction.

6. After time $t_{sd} + 3\Delta$, the supervisor saves all published bid commitments (to be used later in case he would verify the auction's outcome), and sends the secret decryption key to the auctioneer[4].

7. The auctioneer decrypts all bids and computes (i', p') where i' is the bidder with the highest bid and p' is the amount she has to pay (i.e. the second highest bid). He sends (i', p') to the supervisor. In case of a tie the auctioneer chooses a winner arbitrarily out of the set of highest bidders (e.g. he chooses the highest bidder with lexicographical precedence.).

8. With probability α, the supervisor validates the auction's outcome: He decrypts all encrypted bids he previously saved and verifies that the winning bidder and clearing price are indeed (i', p'). If the supervisor detects that the auctioneer cheated he will fine him. In case the supervisor did not detect cheating (either the auctioneer did not deviate from the protocol or the supervisor did not verify the auction's outcome) he sends the auctioneer his signature on (i', p').

9. The auctioneer sends the supervisor-signed values (i', p') to the winning bidder.

3.2.1 Auction Ending Time

Note that the auctioneer can compute in advance an upper bound for the auction's ending time t_{end}. The ending time for the auction is no later than the time to send a bid to the auctioneer in the worst case[5], which is $t_{sd} + 4\Delta$, plus time to decrypt n_{max} encryptions and compute the auction's outcome *twice* (in case the supervisor verifies the auctioneer's computed outcome),

[4] According to Section 3.2.1 and Footnote 5, the last moment in which the supervisor might receive legitimate encrypted bids is $t_{sd} + 3\Delta$.

[5] e.g. a bidder sends a bid at the last moment possible, the auctioneer does not publish it, and the bidder resends it to the auctioneer via the supervisor. So the last message is received by the auctioneer at time $t_{sd} + 4\Delta$.

plus time to send 3 more messages (the auctioneer sends outcome to the supervisor, supervisor replies with signature, auctioneer sends outcome to winning bidder). This computed bound is t_{end}.

4 SECURITY ANALYSIS

Theorem 1. *The presented scheme is secure.*

Proof. Security of the scheme follows from Lemmas 1, 2, 3 and 5 asserting that the scheme is *valid*, *correct for a rational auctioneer*, *preserves secrecy*, and *non-malleable*, respectively. □

4.1 Validity

Lemma 1. *The presented scheme is valid.*

Proof. We prove that the probability that the auctioneer will not be caught by the supervisor when deviating from the protocol if the supervisor validates the auction's outcome is negligible. This is done by dividing the protocol into three phases, analyzing deviation at each phase separately, and proving that the auctioneer cannot deviate in any phase without being caught with overwhelming probability.

The protocol can be divided into the following phases:

1. The preliminary phase: up until the auctioneer publishes the auction details.
2. The submission phase: from the end of the preliminary phase until the auctioneer stops receiving new submissions (time $t_{sd} + 4\Delta$).
3. The outcome revelation phase: from time $t_{sd} + 4\Delta$ until the end of the auction (t_{end}).

In the preliminary phase, the auctioneer sends the supervisor auction details of his choice, and the supervisor signs these details. Publishing supervisor-signed false auction details requires the auctioneer to forge the supervisor's digital signature, in which he has only negligible probability to succeed.

In the submission phase the auctioneer cannot cheat without being caught with overwhelming probability. Firstly, he cannot ignore received messages (i.e. not publish them on the bulletin board) because then the ignored bidders will resend their messages to the supervisor, who will verify that the auctioneer publishes them (otherwise the supervisor will not sign the auction's outcome and the auction will be aborted). Secondly, because of the hiding property of the used encryption scheme, the auctioneer cannot learn information about the bids from the received messages. This prevents him from sending messages with bid values according to values of already received bids (e.g. send an (encryption for a) bid that is slightly lower than the highest honest bid in order to have the winning bidder pay a fictitiously high clearing price). We note, however, that he can still insert bids independently of other bids, since the auctioneer can participate in the auction as a bidder too. Thirdly, one might argue that the auctioneer may try to influence the outcome of the auction by inserting fictitious encryptions for a range of bids. Then, after receiving the decryption key and decrypting all messages, the auctioneer ignores the fictitious bids that are higher than any honest bid but leaves the rest. This way the honest winning bidder will still have to pay a fictitiously high clearing price. But such a cheating will be caught by the supervisor since the supervisor himself decrypts all published bids and ensures that they were all taken into account when determining the auction's outcome.

The auctioneer cannot deviate from the protocol in the outcome revelation phase with non-negligible probability either. After receiving the decryption key and decrypting the bid commitments, if the auctioneer sends the supervisor erroneous outcome (i.e. a wrong winning bidder i or a wrong clearing price p) the supervisor will notice it when he independently calculates the outcome. Furthermore, after sending the correct outcome to the supervisor and receiving his signature on it the auctioneer cannot send a bidder a supervisor-signed erroneous outcome without forging the supervisor's signature, in which he has only negligible probability to succeed. □

4.2 Rational Correctness

Lemma 2. *The presented scheme is correct for a rational auctioneer.*

Proof. Correctness of the protocol follows from the assumption that the auctioneer is rational and the fine that he pays when caught cheating is high enough. If his expected utility when abiding by the protocol is higher than his expected utility when deviating from it he will not deviate and the auction's outcome will be correct. As shown in Lemma 1, the protocol is valid, i.e. in case the supervisor checks the auction he will detect any misbehavior on behalf of the auctioneer. In addition, according to Step 2 of the protocol, the supervisor sets the probability α for validating the outcome such that Equation 7 holds. As shown below, this ensures that the auctioneer's expected utility when deviating from the protocol is non-positive.

The supervisor set α such that Equation 7 holds.

Therefore:

$$\begin{aligned} \alpha \cdot \mathit{fine} + \alpha \cdot b_{maxAl} &\geq b_{maxAl} \\ \alpha \cdot \mathit{fine} &\geq (1-\alpha) b_{maxAl} \\ 0 &\geq (1-\alpha) b_{maxAl} - \alpha \cdot \mathit{fine} \end{aligned} \tag{8}$$

Now, since bids are not higher than the maximum allowed bid b_{maxAl}, the auctioneer's utility U_σ^+ when playing some cheating strategy σ and not being caught is not higher than b_{maxAl} (i.e. in any case the auctioneer will not receive payments higher than b_{maxAl}). Additionally, when the auctioneer is caught cheating he pays *fine*, so $\mathit{fine} = -U_\sigma^-$. Thus:

$$\begin{aligned} (1-\alpha) b_{maxAl} - \alpha \cdot \mathit{fine} &\geq (1-\alpha) U_\sigma^+ + \alpha U_\sigma^- \\ (1-\alpha) U_\sigma^+ + \alpha U_\sigma^- &= \text{Expected utility} \end{aligned} \tag{9}$$

From Equations 8 and 9 we get that the auctioneer's expected utility when playing a cheating strategy σ is non-positive. Hence, for every non-negative utility U for abiding by the protocol, a rational auctioneer will not cheat.

□

4.3 Secrecy

Lemma 3. *The presented scheme preserves secrecy.*

Secrecy of bids in the face of colluding bidders is shown by reduction: If an adversary Adv exists that has a non-negligible advantage in the secrecy experiment (Experiment 2) then an adversary Adv' can be constructed that has a non-negligible advantage in the non-malleability experiment of the underlying CPA-secure encryption scheme.

Proof. The proof is depicted as follows: Below we motivate and present a weaker definition for non-malleability. In Section 4.3.2 we use the weaker definition to show a reduction proving the lemma. Section 4.3.3 discusses some preliminaries needed for analyzing the reduction, and the analysis is in Section 4.3.4.

We start the proof by noting that Pass et al. (Pass et al., 2006) show that the (new) non-malleability definition for encryption schemes presented in Section 3.1.1 is *stronger* than older definitions of non-malleability, such as the one presented by Bellare and Sahai (Bellare and Sahai, 1999) and depicted in Section 4.3.1, i.e. an encryption scheme that is non-malleable according to the new definition is also non-malleable according to the old definition. Therefore, in order to show that an encryption scheme is *malleable* according to the new definition it is sufficient to show that the scheme is *malleable* according to the old definition. Indeed, for this reduction we use the weaker non-malleability definition for a CPA-secure encryption scheme presented by Bellare and Sahai. For the reader's convenience we present the definition below.

4.3.1 Weaker Definition of Non-Malleability

In their paper, Bellare and Sahai (Bellare and Sahai, 1999) define non-malleability of an encryption scheme with respect to chosen-plaintext attacks by Experiment 5 presented below (note that we changed some variable names to fit the auctions context). In the experiment, an adversary Adv works in two stages: Adv_1 and Adv_2. Algorithm Adv_1 receives a public key *Enc* and outputs bid_0, bid_1 (and state information) such that $|bid_0| = |bid_1|$. After one of the plaintexts, bid_b, is randomly chosen and encrypted the resulting ciphertext y is given to Adv_2 in order to guess which plaintext was chosen. Algorithm Adv_2 is itself comprised of two algorithms, $\text{Adv}_{2.\text{Q}}$ and $\text{Adv}_{2.\text{G}}$. The former, $\text{Adv}_{2.\text{Q}}$, receives the two plaintexts, the challenge ciphertext y and state information and outputs a query vector $\boldsymbol{c}$ of ciphertexts and new state information. The latter, $\text{Adv}_{2.\text{G}}$, receives the new state information and Decrypt($\boldsymbol{c}$), namely the output of a decryption oracle when given $\boldsymbol{c}$, and outputs *guess*. Adv wins if $\mathit{guess} = b$.

Definition 6. **(from (Bellare and Sahai, 1999))** *Let Π = (KGen, Encrypt, Decrypt) be an encryption scheme and let Adv be an adversary. For $k \in \mathbb{N}$ let*

$$\begin{aligned} \boldsymbol{Advantage}^{\textit{ind-pca0}}_{\Pi,\,\textit{Adv}}(k) = \\ 2 \cdot Pr[Exp^{\textit{ind-pca0}}_{\Pi,\,\textit{Adv}}(k) = 1] - 1 \end{aligned} \tag{10}$$

where $Exp^{\textit{ind-pca0}}_{\Pi,\,\textit{Adv}}(k)$ is defined in Experiment 5. We say that Π is secure in the sense of IND-PCA0 if Adv being polynomial-time implies that $\boldsymbol{Advantage}^{\textit{ind-pca0}}_{\Pi,\,\textit{Adv}}(k)$ is negligible.

Experiment 5 $\text{Exp}^{\text{ind-pca0}}_{\Pi,\,\text{Adv}}(k)$ (from (Bellare and Sahai, 1999))

1: $(\mathit{Enc}, \mathit{Dec}) \leftarrow \text{KG}(1^k)$
2: $(bid_0,\ bid_1,\ s_1) \leftarrow \text{Adv}_1(\mathit{Enc}, 1^k)$ ▷ $|bid_0| = |bid_1|$
3: $b \xleftarrow{r} \{0,1\}$
4: $y \leftarrow \text{Encrypt}_{\mathit{Enc}}(bid_b)$
5: $(\boldsymbol{c}, s_2) \leftarrow \text{Adv}_{2.\text{Q}}(bid_0, bid_1, s_1, y)$
6: $\boldsymbol{d} \leftarrow \text{Decrypt}_{\mathit{Dec}}(\boldsymbol{c})$
7: $\mathit{guess} \leftarrow \text{Adv}_{2.\text{G}}(\boldsymbol{d}, s_2)$
8: **if** $y \notin \boldsymbol{c} \wedge (\mathit{guess} = b)$ **then return** 1
9: **else return** 0

4.3.2 Reduction

Given an adversary Adv that has a non-negligible advantage in the secrecy experiment (Experiment 2), an adversary Adv' = (Adv_1, $\text{Adv}_{2.\text{Q}}$, $\text{Adv}_{2.\text{G}}$) can be constructed that has a non-negligible advantage in the non-malleability experiment (Experiment 5), as shown in Algorithms 6, 7 and 8.

In Algorithm 6, given a public key *Enc* of an encryption scheme and a unary string of k '1's Adv_1 simulates an auction scheme for Adv: In the *init* stage Adv is given the unary string and outputs n, the number of participating bidders. In the *details* stage, after Adv_1 generates key pairs for all bidders and for the auctioneer and supervisor, Adv receives the appropriate keys and outputs *auction_details*. Later, in the *choose* stage, Adv receives the key *Enc* and outputs two special bidders ψ_0, ψ_1 and two bids bid_0, bid_1. Adv_1 then saves relevant data as state information s_1 and returns the two bid values bid_0, bid_1 as two plaintexts.

Upon receiving bid_0, bid_1 and s_1 as well as the challenge ciphertext y, the $\text{Adv}_{2.\text{Q}}$ algorithm executes as depicted in Algorithm 7: The algorithm creates a ***messages*** vector that will hold bidders' messages. Special bidder ψ_0's encoded bid is set as y and his *id* string as $\perp$. $\text{Adv}_{2.\text{Q}}$ then randomly guesses which plaintext was *not* encrypted by Experiment 5 and sets $\boldsymbol{messages}.\psi_1$ to be the output of the Bidder algorithm when given this bid and key *Enc* as input. $\text{Adv}_{2.\text{Q}}$ then repeatedly invokes Adv in stage *encode* with the encoded bids of the two special bidders as input in order to receive messages for all other bidders. The query vector $\boldsymbol{c}$ is then set to contain the encoded bids of all non-special bidders and $\perp$ as the encoded bids of the special bidders. This is done both to preserve correct indexing of encoded bids and to ensure the query vector does not contain the challenge ciphertext $y = \boldsymbol{messages}.\psi_1.encoded\_bid$. After saving relevant data as state information s_2 Adv' outputs the query vector $\boldsymbol{c}$.

After vector $\boldsymbol{d}$ of bid values of non-special bidders is received $\text{Adv}_{2.\text{G}}$ runs according to Algorithm 8: It sets the ***bids*** vector to contain bid values of all bidders. $\text{Adv}_{2.\text{G}}$ then checks whether the winning bidder is a special bidder using the IsWinnerSpecial() subroutine (Algorithm 9). Note that this subroutine is indifferent to whether special bidder ψ_0 bid bid_0 and ψ_1 bid bid_1 or vice versa, so $\text{Adv}_{2.\text{G}}$ does not need to know this information. If the winner is a special bidder $\text{Adv}_{2.\text{G}}$ invokes Adv with stage *guess* and St_3 as input. If the winner is not a special bidder then $\text{Adv}_{2.\text{G}}$ finds the winning bidder (among the non-special bidders) and the clearing price, and invokes Adv with stage *guess* and state information, the winning bidder, and the clearing price as input. $\text{Adv}_{2.\text{G}}$ then outputs a guess bit *guess* identical to the output of Adv.

Subroutine IsWinnerSpecial() (Algorithm 9) works as follows: It receives the two special bidders ψ_0, ψ_1 along with bid values and messages (containing encoded bids) of all bidders. It then finds the set of bidders that bid the highest bid. Within this set the bidder whose encoded bid has lexicographic precedence is set as the winner. If the winner is one of the two special bidders then the algorithm returns True, otherwise it returns False.

Algorithm 6 $\text{Adv}_1(Enc, 1^k)$

$n \leftarrow \text{Adv}(\text{'init'}, 1^k)$
for each $\psi \in \{1,\ldots,n\} \cup \{\text{A,S}\}$ **do** $(\psi.\text{pk}, \psi.\text{sk}) \leftarrow \text{KG}(\psi, 1^k)$
$(auction\_details,\ St_1) \leftarrow \text{Adv}(\text{'details'},\ \text{S}.pk,\ \text{A}.pk,\ 1.pk, 1.sk, \ldots, n.pk, n.sk)$
$(bid_0, bid_1, \psi_0, \psi_1, St_2) \leftarrow \text{Adv}(\text{'choose'}, St_1, Enc)$
$s_1 \leftarrow (\psi_0, \psi_1, \text{S}.pk, \text{A}.pk, \psi_1.pk, \psi_1.sk, Enc, St_2)$
return (bid_0, bid_1, s_1)

Algorithm 7 $\text{Adv}_{2.\text{Q}}(bid_0, bid_1, s_1, y)$

$(\psi_0, \psi_1, \text{S}.pk, \text{A}.pk, \psi_1.pk, \psi_1.sk, Enc, St_2) \leftarrow s_1$
$\boldsymbol{messages}.\psi_0 \leftarrow (y, \perp)$
$bit \xleftarrow{r} \{0,1\}$
$\boldsymbol{messages}.\psi_1 \leftarrow \mathcal{AUC}.\text{Bidder}(auction\_details, \text{S}.pk, \text{A}.pk, \psi_1.pk, \psi_1.sk, Enc, bid_{bit})$
for each $i \in \{1,\ldots,n\} \setminus \{\psi_0, \psi_1\}$ **do**
    $(\boldsymbol{messages}.i,\ St_2) \leftarrow \text{Adv}(\text{'encode'}, St_2, \boldsymbol{messages}.\psi_0.encoded\_bid, \boldsymbol{messages}.\psi_1.encoded\_bid, i)$
    $\boldsymbol{c}.i \leftarrow \boldsymbol{messages}.i.encoded\_bid$
$\boldsymbol{c}.\psi_0 \leftarrow \perp;\ \boldsymbol{c}.\psi_1 \leftarrow \perp$
$s_2 \leftarrow (St_2, \psi_0, \psi_1, bid_0, bid_1, \boldsymbol{messages})$
return $(\boldsymbol{c}, s_2)$

4.3.3 Preliminaries

We begin the analysis by observing that the query vector $\boldsymbol{c}$ sent to the non-malleability oracle does not contain y (which is significant in step 8 of Experiment 5). Firstly, bidder ψ_0's encoded bid is set to $\perp$ instead of y (see Algorithm 7, Step 8). Secondly, Adv can be assumed to not output encoded bids for non-special bidders that are identical to y. This is because the auctioneer ignores bid encodings that are identical to encodings that were already published on the bulletin board. Since no new information is gained by

Algorithm 8 $\text{Adv}_{2.\text{G}}(\boldsymbol{d}, s_2)$

$(St_3, \psi_0, \psi_1, bid_0, bid_1, \boldsymbol{messages}) \leftarrow s_2$
$\boldsymbol{bids} \leftarrow \boldsymbol{d}$
$\boldsymbol{bids}.\psi_0 \leftarrow bid_0$; $\boldsymbol{bids}.\psi_1 \leftarrow bid_1$
if IsWinnerSpecial(ψ_0, ψ_1, ***bids***, ***messages***) **then**
    guess = Adv('guess', St_3)
else
    $\psi_{\text{winner}}^{\text{non-spcl}} \leftarrow \mathcal{AUC}.\text{Winner}(\boldsymbol{bids}, \boldsymbol{messages})$
    $p \leftarrow \text{second\_highest}(\boldsymbol{d} \cup \{bid_0, bid_1\})$
    guess = Adv('guess', St_3, $\psi_{\text{winner}}^{\text{non-spcl}}$, p)
return *guess*

Algorithm 9 $\text{IsWinnerSpecial}(\psi_0, \psi_1, \boldsymbol{bids}, \boldsymbol{messages})$

$\boldsymbol{highest\_bidders} \leftarrow \arg\max_i(\boldsymbol{bids}.i)$
$\psi_{\text{winner}} \leftarrow$ First(Order(***messages***$.i.$*encoded_bid* **for all** $i \in$ ***highest_bidders***))
if $\psi_{\text{winner}} \in \{\psi_0, \psi_1\}$ **then**
    return True
else return False

Adv when outputting encodings for non-special bidders that are identical to encodings of special bidders (because the new non-special encodings are immediately ignored) it can be assumed, w.l.o.g., that Adv does not output such encodings. As a side note, we observe that the probability that honest bidders output encodings that are identical to other bidders' encodings when using a probabilistic encryption scheme is negligible. Therefore, with overwhelming probability having an auctioneer that ignores such identical encodings does not affect the auction.

In addition, we claim that Adv is indifferent in stage *encode* to whether it is given as input encodings of bid_0 and bid_1 or encodings of any other bids. This is shown in Lemma 4 below by proving that any adversary (and in particular, our Adv) that has stages *init, details, choose* and *encode* as defined in the secrecy experiment cannot tell with probability significantly higher than guessing whether it was given in stage *encode* encodings of bid_0 and bid_1 or not.

Lemma 4. *For every adversary* $\widehat{\text{Adv}}$ *with stages* init, details, choose *and* encode *as defined in the secrecy experiment (Experiment 2), the probability for it to distinguish if it was given in stage* encode *input containing encodings of* bid_0 *and* bid_1 *or input containing encodings of different bids is no more than negligibly higher than guessing.*

Proof. For every adversary $\widehat{\text{Adv}}$ that has probability β to distinguish if its input in stage *encode* includes encodings of bid_0 and bid_1 or not, i.e. for every $\widehat{\text{Adv}}$ that in stage *encode* has probability β to output correctly a bit b such that $b = 1$ if the input contains encodings of bid_0 and bid_1 and $b = 0$ otherwise, an adversary $\widehat{\text{Adv'}} = (\widehat{\text{Adv'}}_1, \widehat{\text{Adv'}}_2)$ can be constructed that has probability β to win the indistinguishability experiment of the underlying CPA-secure encryption scheme.

We note that if $\widehat{\text{Adv}}$ in stage *init* sets $n = 2$ then $\widehat{\text{Adv}}$ is never called with stage *encode* and the lemma is voidly true. We therefore continue the proof assuming $n > 2$.

Given an encryption key *Enc*, $\widehat{\text{Adv'}}_1$ runs similarly to $\text{Adv}_1(Enc, 1^k)$ (see Algorithm 6): it simulates an auction scheme for $\widehat{\text{Adv}}$ and outputs the same two plaintexts bid_0 and bid_1 as $\widehat{\text{Adv}}$. $\widehat{\text{Adv'}}_1$ saves the variable n in the state information variable s_1 in addition to all information that $\text{Adv}_1(Enc, 1^k)$ saves in s_1.

When $\widehat{\text{Adv'}}_2$ receives the challenge ciphertext y it runs as shown in Algorithm 10: It invokes the Bidder algorithm with key *Enc* and bid bid_0 to receive *encoded_bid*$_0$ and *id*. It then invokes $\widehat{\text{Adv}}$ in stage *encode* with ('encode', St_2, y, *encoded_bid*$_0$, i) as input (using a randomly chosen i). $\widehat{\text{Adv'}}$ receives a guess bit b and outputs it as its own guess bit.

The winning probability of both $\widehat{\text{Adv}}$ and $\widehat{\text{Adv'}}$ is β because their output bit is identical. Since the underlying encryption scheme is CPA-secure, β is negligibly higher (at most) than a random guess. □

Algorithm 10 $\widehat{\text{Adv'}}_2(bid_0, bid_1, s_1, y)$

(ψ_0, ψ_1, S.*pk*, A.*pk*, ψ_1.*pk*, ψ_1.*sk*, *Enc*, St_2, n) $\leftarrow s_1$
(*encoded_bid*$_0$, *id*) $\leftarrow$ $\mathcal{AUC}$.Bidder(*auction_details*, S.*pk*, A.*pk*, ψ_1.*pk*, ψ_1.*sk*, *Enc*, bid_0)
$i \xleftarrow{r} \{1, \ldots, n\} \setminus \{\psi_0, \psi_1\}$
$b \leftarrow \widehat{\text{Adv}}$('encode', St_2, y, *encoded_bid*$_0$, i)
return b

4.3.4 Reduction Analysis

When Adv' simulates an auction scheme for Adv he randomly chooses which one of bid_0, bid_1 will be encrypted as ***messages***$.\psi_1.$*encoded_bid* (See steps 3 and 4 in Algorithm 7). Therefore, Adv' correctly simulates an auction scheme and gives Adv encodings of both special bids with probability $\frac{1}{2}$. In such a case, since the output bit *guess* of Adv' is identical to the output bit b' of Adv, the advantage of Adv' in winning the cryptographic experiment is non-negligible, as the advantage of Adv.

With probability $\frac{1}{2}$ Adv' does not simulate an auction scheme correctly, i.e. does not supply Adv with correct bid encodings. In step 6 of algorithm 7 Adv receives either two encodings of bid_0 or two encodings of bid_1 (and not one encoding of each). Lemma 4 ensures that Adv is oblivious to the values the encodings encode, i.e. the output of Adv is indifferent to whether y and $\boldsymbol{messages}.\psi_1.encoded\_bid$ encode the same bid or not. In either case, Adv' will continue and query the decryption oracle using vector $\boldsymbol{c}$ and receive the bids of all non-special bidders. Adv' will then know if the winning bidder is one of the two special bidders and what is the clearing price, ensuring he will invoke Adv with correct input in the *guess* stage (see Algorithm 8). Thus, the case in which Adv' gave Adv wrong y and $\boldsymbol{messages}.\psi_1.encoded\_bid$ such that they both encode bid_0 is indistinguishable (regarding the information Adv sees) to the case in which Adv' gave Adv wrong y and $\boldsymbol{messages}.\psi_1.encoded\_bid$ such that they both encode bid_1. Consequently, the probability that Adv guesses correctly in the former case is equal to the probability that Adv guesses correctly in the latter, i.e. Adv (and Adv', whose guess is identical) has a winning probability of $\frac{1}{2}$ when the auction scheme was not simulated correctly.

In summary, with probability $\frac{1}{2}$ the advantage Adv' has in the non-malleability experiment (Experiment 5) is non-negligible, and with probability $\frac{1}{2}$ it is zero. Therefore, his overall advantage in the experiment is non-negligible and the scheme is malleable. As discussed earlier, this implies that the underlying encryption scheme is malleable also according to Definition 5, contradicting the initial assumption. □

4.4 Non-Malleability

Lemma 5. *The presented scheme is non-malleable.*

Non-malleability is shown by reduction: If an adversary Adv exists that can *maul* an encoded bid into another one that is related to the original bid then an adversary Adv' exists that can maul a ciphertext, generated by encrypting a plaintext using the auction's underlying encryption scheme, into a different ciphertext that is related to the original plaintext. For the proof we use the definition of non-malleable encryption schemes presented by Pass et al. (Pass et al., 2006). For the reader's convenience we bring it below.

Proof. If an auction scheme $\mathcal{AUC}$, $k,l \in \mathbb{N}$, and an adversary Adv exist such that the output of the non-malleability experiment (Experiment 3) is computationally distinguishable when $b = 0$ compared to when $b = 1$ then an adversary Adv' = (Adv'$_1$, Adv'$_2$) exists such that the output of the non-malleability experiment of the underlying encryption scheme Π (Experiment 4) is computationally distinguishable. Adv' would run according to Algorithms 11 and 12, as detailed below.

Algorithm 11 Adv'$_1(Enc, 1^k)$

1: $n \leftarrow$ Adv('init', 1^k)
2: **for each** $\psi \in \{1,\ldots,n\} \cup \{\text{A,S}\}$ **do** $(\psi.\text{pk}, \psi.\text{sk}) \leftarrow \text{KG}(\psi, 1^k)$
3: ($auction\_details$, St_1) $\leftarrow$ Adv('details', S.pk, A.pk, $1.pk, 1.sk, \ldots, n.pk, n.sk$)
4: $(\psi, bid_0, bid_1, St_2) \leftarrow$ Adv('choose', St_1, Enc)
5: **return** (bid_0, bid_1, St_2)

Algorithm 12 Adv'$_2(St_2, y)$

1: $(c_1, \ldots, c_l) \leftarrow$ Adv('guess', St_2, y)
2: **return** $(c_1, \ldots, c_l)$

Given an encryption key Enc algorithm Adv'$_1$ simulates an auction scheme for Adv as detailed in Algorithm 11: Adv is invoked in stage *init* and outputs the number of participating bidders n. Keys are then generated to all entities by the KG algorithm. Adv receives relevant keys and outputs *auction_details* and state information St_1. Adv then receives the public encryption key Enc and outputs a bidder ψ, two bids bid_0, bid_1 and state information St_2. Adv'$_1$ outputs the two bids along with the state information.

When Adv'$_2$ receives state information and the challenge ciphertext y it invokes Adv in stage *guess* with St_2 and y as input. Adv'$_2$ then outputs the same vector of encryptions as Adv does.

If the output of Adv is computationally indistinguishable when $b = 0$ compared to when $b = 1$ then so is the output of Adv', since they have identical outputs. Therefore, having an adversary that breaks the auction's non-malleability implies that the underlying encryption scheme is non-malleable.

□

4.5 Adversarial Bidders

Bidders may slightly abuse the protocol and send their bids somewhat later than the submission deadline t_{sd}. By sending her bid directly to the supervisor at time $t_{sd} + 2\Delta$, a bidder may commit to a bid at time which is 2Δ later than the submission deadline. Such a submission tardiness may give only negligible utility in

some scenarios, but might be significant in others. In cases where such abuse should be avoided, the following may be added to the protocol: If a bid is to be submitted after time $t_{sd} - 2\Delta$ (but before t_{sd}), then the bidder must simultaneously send a commitment of the bid to the auctioneer and a doubly committed bid to the supervisor (using a different, bidder generated key for the second commitment). Note that this doubly committed bid will be received by the supervisor not later than $t_{sd} + \Delta$. If a bidder sees that her bid was not published by the auctioneer she will resend the committed bid to the supervisor. The supervisor will verify that the received commitment is indeed what was committed to by the bidder and received earlier (i.e. up until $t_{sd} + \Delta$), and will forward the commitment to the auctioneer. This way the bidder must commit to a bid no later than t_{sd}. Using doubly committed bids ensures that the supervisor does not learn nonessential information about the committed bids it received.

Auctioneer's security against non-paying bidders is achieved by using signed commitments. Bidders are legally committed to pay, or else they may be sued. Alternatively, the auctioneer may require a signed statement from a trusted third party that a bidder has the amount of money she is bidding.

5 CONCLUSIONS

In this paper we presented novel security definitions for the validity, rational correctness, secrecy, and non-malleability of second price auction schemes. In addition, a simple and efficient scheme is presented in which the security requirements hold. This is done using a trusted supervisor which randomly validates the auction's outcome.

One may wonder, in case such a trusted supervisor exists, why not let this trusted entity run the auction instead of the auctioneer. One answer is that an entity that validates the outcome but which does not receive payments from the bidders (such as the supervisor) has less incentive to cheat, as opposed to the auctioneer More importantly, in case there is a need for a highly trusted supervisor, the supervisor program may be run using secure means such as special hardware or secure multiparty computation. Employing such costly means for the supervisor may introduce substantial overhead. To ensure the protocol's efficient and practicality, such a costly supervisor may choose to participate only in auctions where α is low enough, guaranteeing both that the auction is secure and that the supervisor has a low amount of expected computation. Notice, however, that requiring α to be small may induce a high fine, since according to Equation 7 the lower bound of *fine* is:

$$fine \geq \frac{(1-\alpha)b_{maxAl}}{\alpha} \tag{11}$$

Still, we believe there is enough freedom in tuning the supervisor's workload to ensure the scheme is practical.

6 FURTHER RESEARCH

There are many issues for further research, both in adding attributes to the presented protocol and in using the techniques displayed in this paper for creating protocols for other scenarios.

It is possible to keep bidders' privacy by hiding their identity from the auctioneer. This can be done by schemes such as group signatures, which allow the auctioneer to authenticate bids as being sent from authorized bidders without personally identifying the senders. In case of dispute or misbehavior, an identity called the group manager can break the anonymity and identify senders of specific bids. For details, see (Chaum and Van Heyst, 1991; Bellare et al., 2003; Bellare et al., 2005). Note that group signature schemes hide the identity of the bidders but do not hide their bids. Second price auctions that hide the bids from the auctioneer are the focus of papers such as (Naor et al., 1999).

Techniques displayed in this paper can be used for creating protocols for other scenarios as well. For example, one may consider the brokerage scenario in which many put and call options are presented simultaneously. A server receives these options and matches them according to a predefined matching algorithm. Such scenarios are presented in various papers (Malone et al., 1987; Bichler and Segev, 1999; Resnick et al., 1995). A possible research subject would be to find ways to reduce trust in the matchmaker by modeling him as a rational player and creating an appropriate and efficient protocol for this scenario, as displayed in this paper.

ACKNOWLEDGEMENTS

We thank Alon Rosen, Yehuda Lindell, and Benny Pinkas for their comments. Additionally, we thank the following organizations for financially supporting this research: The Israeli Ministry of Science and Technology, and the RSA division of EMC corporation.

REFERENCES

Bellare, M., Micciancio, D., and Warinschi, B. (2003). Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT*, pages 614–629.

Bellare, M. and Sahai, A. (1999). Non-malleable encryption: Equivalence between two notions, and an indistinguishability-based characterization. In *Advances in cryptology-CRYPTO99*, pages 78–78. Springer. `http://cseweb.ucsd.edu/~mihir/papers/nm.pdf`.

Bellare, M., Shi, H., and Zhang, C. (2005). Foundations of group signatures: The case of dynamic groups. *Topics in Cryptology–CT-RSA 2005*, pages 136–153.

Bichler, M. and Segev, A. (1999). A brokerage framework for internet commerce. *Distributed and Parallel Databases*, 7(2):133–148.

Boneh, D. and Naor, M. (2000). Timed commitments. In *CRYPTO*, pages 236–254.

Brandt, F. (2006). How to obtain full privacy in auctions. *International Journal of Information Security*, 5(4):201–216.

Chaum, D. and Van Heyst, E. (1991). Group signatures. In *Proceedings of the 10th annual international conference on Theory and application of cryptographic techniques*, pages 257–265. Springer-Verlag.

Dolev, D., Dwork, C., and Naor, M. (1991). Nonmalleable cryptography. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 542–552. ACM.

Engelbrecht-Wiggans, R. (1980). Auctions and bidding models: A survey. *Management Science*, pages 119–142.

Goldwasser, S. and Micali, S. (1982). Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, pages 365–377. ACM New York, NY, USA.

Goldwasser, S. and Micali, S. (1984). Probabilistic encryption. *Journal of computer and system sciences*, 28(2):270–299.

Harkavy, M., Tygar, J., and Kikuchi, H. (1998). Electronic auctions with private bids. In *Proceedings of the 3rd USENIX Workshop on Electronic Commerce*, volume 31.

Juels, A. and Szydlo, M. (2003). A two-server, sealed-bid auction protocol. In *Financial Cryptography*, pages 72–86. Springer.

Klemperer, P. (2004). Auctions: theory and practice.

Lipmaa, H., Asokan, N., and Niemi, V. (2003). Secure vickrey auctions without threshold trust. In *Financial Cryptography*, pages 87–101. Springer.

Malone, T., Yates, J., and Benjamin, R. (1987). Electronic markets and electronic hierarchies. *Communications of the ACM*, 30(6):484–497.

Milgrom, P. and Weber, R. (1982). A theory of auctions and competitive bidding. *Econometrica: Journal of the Econometric Society*, pages 1089–1122.

Naor, M., Pinkas, B., and Sumner, R. (1999). Privacy preserving auctions and mechanism design. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 129–139. ACM.

Parkes, D., Rabin, M., Shieber, S., and Thorpe, C. (2008). Practical secrecy-preserving, verifiably correct and trustworthy auctions. *Electronic Commerce Research and Applications*, 7(3):294–312.

Pass, R., Vaikuntanathan, V., et al. (2006). Construction of a non-malleable encryption scheme from any semantically secure one. In *Advances in Cryptology-CRYPTO 2006*, pages 271–289. Springer.

Rabin, M. and Thorpe, C. (2006). Time-lapse cryptography.

Resnick, P., Zeckhauser, R., and Avery, C. (1995). Roles for electronic brokers. In *Toward a Competitive Telecommunication Industry: Selected Papers from the 1994 Telecommunications Policy Research Conference. Mahwah, New Jersey: Lawrence Erlbaum Associates*, pages 289–306.

Rivest, R., Shamir, A., and Wagner, D. (1996). Time-lock puzzles and timed-release crypto.

Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *The Journal of finance*, 16(1):8–37.