Hawk and Aucitas: e-auction schemes from the Helios and Civitas e-voting schemes

Adam McCarthy[1], Ben Smyth[1], and Elizabeth A. Quaglia[2]

[1] INRIA Paris-Rocquencourt, France
[2] ENS, Paris, France

Abstract. The cryptographic foundations of e-auction and e-voting schemes are similar, for instance, seminal works in both domains have applied mixnets, homomorphic encryption, and trapdoor bit-commitments. However, these developments have appeared independently and the two research communities are disjoint. In this paper, we demonstrate a relation between e-auction and e-voting: we present Hawk and Aucitas, two e-auction schemes derived from the Helios and Civitas e-voting schemes. Our results make progress towards the unification of the e-auction and e-voting domains.

Keywords. Aucitas, auction, bid secrecy, Civitas, collusion resistance, Hawk, Helios, price flexibility, privacy, sealed-bid, verifiability, voting.

1 Introduction

An *e-auction* is a process for the trade of goods and services from *sellers* to *bidders* (or *buyers*), with the aid of an *auctioneer*. We study *sealed-bid auctions*, which are defined as follows. First, each bidder submits a *bid* which encapsulates the *price* that the bidder is willing to pay. Secondly, the bids are *opened* to derive the *winning price*. Finally, the *winner* is *revealed*. The winning price and winner are derived in accordance with the auction's policy, for example, in *first-price sealed-bid auctions* the winning price is the highest price bid and the winner is the bidder who bid at the winning price. We shall focus on *Mth price sealed-bid auctions*, which generalise first-price sealed-bid auctions to sell M identical items at the highest price that M bidders are mutually willing to pay. For instance, in the case $M = 6$, six identical items will be sold at the sixth highest price that is bid, because six bidders are mutually willing to pay this price.

An *election* is a decision-making process by which *voters* choose a *representative* from some *candidates*. We study *secret ballot elections*, which are defined as follows. First, each voter submits a *ballot* which encapsulates the voter's chosen candidate (i.e., the voter's *vote*). Secondly, all ballots are *tallied* to derive the *distribution of votes*. Finally, the representative is derived in accordance with the election's policy, e.g., in *first-past-the-post elections* the representative is the candidate with the most votes. In this paper, we shall demonstrate that it is possible to derive e-auction schemes from e-voting schemes.

⋆ See [16] for the long version of this paper.

Constructing e-auction schemes from e-voting schemes. Our translation from an e-voting scheme to an e-auction scheme assumes that prices can be represented as candidates, for example, an e-auction with a *starting price* of 10, *price increments* of 5 and a *price ceiling*[1] of 30 can be represented by the following five candidates: 10, 15, 20, 25 and 30 (we refer to these values as *biddable prices*). In this setting, an e-auction proceeds as follows. First, to bid for a particular price, bidders "vote" for the candidate that represents the price that the bidder is willing to pay, for example, a bid at price 20 is captured by a "vote" for the third candidate. Secondly, the bids are "tallied" to determine the distribution of "votes" and the winning price is derived from this distribution: the winning price is the largest price in $(10, 15, 20, 25, 30)$ for which at least M bidders "voted" at or above. Finally, we link the winning price to winning bidders. This final step distinguishes our e-auction scheme from the underlying e-voting scheme and we shall see that this can be achieved in the context of secret ballot elections.

1.1 Security properties

Bidders should be able to bid in auctions without fear of repercussions. This property is known as *privacy* and *bid secrecy* has emerged as a *de facto* standard privacy requirement.

- **Bid secrecy**: A losing bidder cannot be linked to a price.

We are also interested in *collusion resistance* (to help prevent *bid rigging* [19] by conspiring bidders).

- **Collusion resistance**: A losing bidder cannot collaborate with a conspirator to gain information which can be used to prove how they bid.

Verifiability allows bidders and observers to verify that bids have been recorded and tallied correctly without trusting the system running the e-auction. The concept is intended to avoid situations whereby systems are trusted and, subsequently, discovered to be untrustworthy, thus bringing auctions into disrepute. We distinguish the following three aspects of verifiability.

- **Outcome verifiability**: A bidder can check that their bid is included in the e-auction and anyone can check that the winning price is valid.
- **Eligibility verifiability**: Anyone can check that all bids were submitted by registered bidders.
- **Non-repudiation**: Anyone can check the winners' identities.

We are also interested in the following functional requirement, which avoids restricting the bidding amount.

- **Price flexibility**: Bidders can submit any price.

[1] A price ceiling – that is, an upper bound on the price that may be offered by bidders – is common in e-auctions.

2 Cryptographic preliminaries

We adopt standard notation for the application of probabilistic algorithms A, namely, $A(x_1, \ldots, x_n; r)$ is the result of running A on input $x_1, \ldots, x_n$ and coins r. Moreover, $A(x_1, \ldots, x_n)$ denotes $A(x_1, \ldots, x_n; r)$, where r is chosen at random. We write $x \leftarrow \alpha$ for the assignment of α to x. Vectors are denoted using boldface, for example, $\mathbf{x}$. We write $|\mathbf{x}|$ to denote the length of a vector $\mathbf{x}$ and $\mathbf{x}[i]$ for the ith component of the vector, where $\mathbf{x} = (\mathbf{x}[1], \ldots, \mathbf{x}[|\mathbf{x}|])$. We extend set membership notation to vectors: we write $x \in \mathbf{x}$ (respectively, $x \notin \mathbf{x}$) if x is an element (respectively, x is not an element) of the set $\{\mathbf{x}[i] : 1 \leq i \leq |\mathbf{x}|\}$.

An *asymmetric encryption scheme* is a tuple of algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ satisfying the standard correctness property (see the long version [16, Definition 1] of this paper for a formal definition). We say an encryption scheme is *homomorphic* if there exists binary operators $\oplus$, $\otimes$ and $\odot$ such that for all $(pk, sk, \mathfrak{m}) \leftarrow \mathsf{Gen}(1^k)$, messages $m_1, m_2 \in \mathfrak{m}$ and coins r_1 and r_2, we have $\mathsf{Enc}(pk, m_1; r_1) \otimes \mathsf{Enc}(pk, m_2; r_2) = \mathsf{Enc}(pk, m_1 \odot m_2; r_1 \oplus r_2)$. The scheme is *additive homomorphic* if $\odot$ is the addition operator or *multiplicative homomorphic* if $\odot$ is the multiplication operator.

An interactive proof system is a two party protocol between a prover and a verifier on some common input, which allows a claim of membership to be evaluated. Formally, we capture such proof systems as *sigma protocols* (see the long version [16, Definition 2] of this paper for a formal definition). A sigma protocol for an $\mathcal{NP}$ language $\mathcal{L}_R$, where $\mathcal{L}_R = \{s \mid \exists\ w \text{ such that } (s, w) \in R\}$, is a tuple of algorithms $(\mathsf{Comm}, \mathsf{Chal}, \mathsf{Resp}, \mathsf{Verify})$ satisfying *special soundness* and *special honest-verifier zero-knowledge* (see [5] for details), in addition to the standard completeness property. Our e-auction schemes are dependent upon the sigma protocols given in Definition 1.

Definition 1. *Given an asymmetric encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *and a sigma protocol* Σ *for the language* $\mathcal{L}_R$*, we say* Σ*:*

- proves correct key construction *if* $((1^k, pk', \mathfrak{m}'), (sk', r)) \in R \Leftrightarrow (pk', sk', \mathfrak{m}') = \mathsf{Gen}(1^k; r)$
- proves plaintext knowledge in $\mathfrak{M}$ *if* $\mathfrak{M} \subseteq \mathfrak{m}$ *and* $((pk, c), (m, r)) \in R \Leftrightarrow c = \mathsf{Enc}(pk, m; r) \wedge m \in \mathfrak{M}$
- proves correct ciphertext construction *if* $((pk, c_1, \ldots, c_\ell), (m_1, r_1, \ldots, m_\ell, r_\ell)) \in R \Leftrightarrow \bigwedge_{1 \leq i \leq \ell} c_i = \mathsf{Enc}(pk, m_i; r_i)$
- *is a* plaintext equality test *(PET) if* $((pk, c, c', i), sk) \in R \wedge i \in \{0, 1\} \Leftrightarrow ((i = 0 \wedge \mathsf{Dec}(pk, sk, c) \neq \mathsf{Dec}(pk, sk, c')) \vee (i = 1 \wedge \mathsf{Dec}(pk, sk, c) = \mathsf{Dec}(pk, sk, c'))) \wedge \mathsf{Dec}(pk, sk, c) \neq \perp$
- proves decryption *if* $((pk, c, m), sk) \in R \Leftrightarrow m = \mathsf{Dec}(pk, sk, c)$

where $(pk, sk, \mathfrak{m}) \leftarrow \mathsf{Gen}(1^k)$.

We can derive *proofs of knowledge* from sigma protocols using the *Fiat-Shamir heuristic* [9], which replaces the verifier's challenge with a hash of the prover's commitment, optionally concatenated with the prover's statement [5] and a message.

Definition 2 (Fiat-Shamir transformation). *Given a sigma protocol* $\Sigma = (\mathsf{Comm}_\Sigma, \mathsf{Chal}_\Sigma, \mathsf{Resp}_\Sigma, \mathsf{Verify}_\Sigma)$ *and a hash function* $\mathcal{H}$*, the Fiat-Shamir transformation* $\mathsf{FS}(\Sigma, \mathcal{H}) = (\mathsf{Prove}, \mathsf{Verify})$*, where* Prove *and* Verify *are the algorithms defined as follows:*

- *The* proof algorithm Prove *takes a statement* s*, witness* w*, and (optionally) message* m *as input. The algorithm proceeds as follows. First, compute* $(\mathsf{comm}, t) \leftarrow \mathsf{Comm}_\Sigma(s, w)$*. Secondly, derive* chal *as follows: if* m *is defined, then* $\mathsf{chal} \leftarrow \mathcal{H}(s, \mathsf{comm}, m)$*, otherwise,* $\mathsf{chal} \leftarrow \mathcal{H}(s, \mathsf{comm})$*. Thirdly, compute* $\mathsf{resp} \leftarrow \mathsf{Resp}_\Sigma(\mathsf{chal}, t)$*. Finally, output* $\sigma = (\mathsf{comm}, \mathsf{resp})$*.*
- *The* verification algorithm Verify *takes a statement* s*, candidate proof* $(\mathsf{comm}, \mathsf{resp})$ *and (optionally) message* m *as input and outputs* $\mathsf{Verify}_\Sigma(s, (\mathsf{comm}, \mathsf{chal}, \mathsf{resp}))$*, where* chal *is derived as follows: if* m *is defined, then* $\mathsf{chal} \leftarrow \mathcal{H}(s, \mathsf{comm}, m)$*, otherwise,* $\mathsf{chal} \leftarrow \mathcal{H}(s, \mathsf{comm})$*.*

3 Syntax for e-auction schemes

Based upon Bernhard *et al.* [4, 5, 18], we formalise *e-auction schemes* as a tuple of algorithms (Setup, BB, Open, Reveal) which are executed by an auctioneer and bidders as follows. (We consider a single auctioneer for simplicity and note that schemes can be generalised to several auctioneers to distribute trust, if necessary.) The Setup algorithm is run by the auctioneer to initialise a key pair and bulletin board. The Bid algorithm is used by bidders to generate their bids and the BB algorithm is used by the auctioneer to process bids, in particular, the algorithm adds correctly formed bids to the bulletin board. Once all of the bids have been collected, the auctioneer runs Open to find the winning price, which is announced by the auctioneer. Finally, the Reveal algorithm is used to identify winners; the Reveal algorithm uses private data $\mathbf{s}$ to reveal the winners, for example, $\mathbf{s}$ could be a private key which is used to decrypt bids. We define the inputs and outputs of our algorithms below:

- $\mathsf{Setup}(1^k) \rightarrow (pk, sk, \mathfrak{bb}, \textit{aux-pk})$.The *setup algorithm* Setup takes the security parameter 1^k as input and outputs a public key pk, private key sk, bulletin board $\mathfrak{bb}$ and auxiliary data *aux-pk*, where $\mathfrak{bb}$ is a set.
- $\mathsf{Bid}(pk, \textit{aux-pk}, \mathbf{P}, p) \rightarrow b$. The *bid algorithm* Bid takes as input a public key pk, auxiliary data *aux-pk*, vector of biddable prices $\mathbf{P}$ and price p, where $1 \leq p \leq |\mathbf{P}|$. It outputs a bid b such that $b = \perp$ upon failure.
- $\mathsf{BB}(pk, \mathbf{P}, \mathfrak{bb}, b) \rightarrow \mathfrak{bb}'$. The *bulletin board algorithm* BB takes as input a public key pk, vector of biddable prices $\mathbf{P}$, bulletin board $\mathfrak{bb}$ and bid b, where $\mathfrak{bb}$ is a set. It outputs $\mathfrak{bb} \cup \{b\}$ if successful or $\mathfrak{bb}$ to denote failure.
- $\mathsf{Open}(pk, sk, \mathbf{P}, \mathfrak{bb}, M) \rightarrow (p, \textit{aux-open})$. The *opening algorithm* Open takes as input a public key pk, private key sk, vector of biddable prices $\mathbf{P}$, bulletin board $\mathfrak{bb}$ and parameter M denoting the number of items to be sold, where $\mathfrak{bb}$ is a set and $M > 0$. It outputs the winning price p and auxiliary data *aux-open* such that $p = 0$ if no winning price is found and $p = \perp$ upon failure.

$\mathsf{Reveal}(pk, \mathbf{s}, \textit{aux-pk}, \mathbf{P}, \mathbb{bb}, M, p, \textit{aux-open}) \to (w, \textit{aux-reveal})$. The *reveal algorithm* Reveal takes as input a public key pk, private data $\mathbf{s}$, auxiliary data *aux-pk*, a vector of biddable prices $\mathbf{P}$, bulletin board $\mathbb{bb}$, parameter M denoting the number of items to be sold, winning price p and auxiliary data *aux-open*, where $M > 0$ and $1 \leq p \leq |\mathbf{P}|$. It outputs a vector of winners w and auxiliary data *aux-reveal* such that $w = \perp$ upon failure.

Our definition assumes that a vector of biddable prices $\mathbf{P}$ has been published and a bid for price $\mathbf{P}[p]$ is identified by price index p, where $\mathbf{P}[1] < \cdots < \mathbf{P}[|\mathbf{P}|]$ and $1 \leq p \leq |\mathbf{P}|$. For ease of understanding, we sometimes refer to p as a price.

4 Hawk: An e-auction scheme based on Helios

Hawk is an e-auction scheme derived from the Helios e-voting scheme [3]. An auction is created by naming an auctioneer. The auctioneer generates a key pair and a proof of correct construction. The auctioneer publishes the public key, proof, biddable prices, and number of items to be sold. The bidding phase proceeds as follows.

Bidding. The bidder creates a bid by encrypting her price with the auctioneer's public key and proving that the ciphertext contains a biddable price. The bidder sends her bid to the auctioneer. The auctioneer authenticates the bidder, checks that she is eligible to bid, and verifies the bidder's proof; if these checks succeed, then the auctioneer publishes the bid on the bulletin board.

After some predefined deadline, the opening and revealing phases commence.

Opening. The auctioneer homomorphically combines the bids, decrypts the homomorphic combination, proves that decryption was performed correctly, and announces the winning price.

Revealing. The auctioneer identifies bids for prices greater than or equal to the winning price, decrypts these bids, and proves that decryption was performed correctly.

Intuitively, every phase of the auction is verifiable. Bidders can check that their bid appears on the bulletin board and, by verifying bidders' proofs, observers are assured that bids represent valid prices. Moreover, anyone can check that the homomorphic combination of bids and decryption were correctly computed. Furthermore, anyone can verify that the decrypted bids contain prices greater than or equal to the winning price. It follows that outcome verifiability is satisfied. In addition, our scheme satisfies bid secrecy, since bids for prices less than the winning price are not decrypted, and also provides non-repudiation, assuming that the auctioneer authenticates the relation between bidders and bids. (An informal security analysis appears in the long version [16, §4.4] of this paper.)

4.1 Cryptographic construction

We derive Hawk (Auction Scheme 1) from our informal description using an additively homomorphic encryption scheme satisfying IND-CPA, proofs of correct key construction, proofs of plaintext knowledge, and proofs of decryption. The Setup algorithm generates the auctioneer's key pair, proves correct key construction, and initialises the bulletin board. The Bid algorithm outputs ciphertexts $c_1, \ldots, c_{|\mathbf{P}|}$, such that ciphertext c_p contains plaintext 1 and the remaining ciphertexts contain plaintext 0, where $\mathbf{P}[p]$ is the price that the bidder is willing to pay. The algorithm also outputs proofs $\sigma_1, ..., \sigma_{|\mathbf{P}|}$ so that this can be verified. Moreover, it outputs a proof $\sigma_{|\mathbf{P}|+1}$ that the bidder bid for at most one price. The BB algorithm adds correctly formed ballots to the bulletin board. The Open algorithm homomorphically combines ciphertexts representing bids at the highest price and decrypts the homomorphic combination, the algorithm repeats this process for ciphertexts at lower prices, until the sum of the decrypted ciphertexts is equal to or greater than the number of items to be sold, i.e., M. The Reveal algorithm homomorphically combines a bidder's ciphertexts at or above the winning price, and decrypts the homomorphic combination. The bidder is a winner if the decryption reveals plaintext 1. In the long version [16] of this paper we demonstrate an execution of Hawk and implement[2] a variant which provides a stronger notion of privacy.

A comparison of Helios and Hawk. In terms of functionality, the new contribution of Hawk is the introduction of its reveal algorithm, which can be used to link a price to a bidder, given the auctioneer's private key. In addition, we improve efficiency: Hawk's opening algorithm modifies Helios's tallying algorithm, in particular, Hawk only decrypts homomorphic combinations of ciphertexts until the sum of the decrypted ciphertexts is equal to or greater than the number of items to be sold, whereas Helios decrypts all homomorphic combinations of ciphertexts.

5 Aucitas: An e-auction scheme based on Civitas

Aucitas is an e-auction scheme derived from the Civitas e-voting scheme [7], which extends the e-voting scheme by Juels, Catalano & Jakobsson [13]. An auction is created by naming an auctioneer and registrar. The auctioneer generates a key pair and a proof of correct key construction. The auctioneer publishes the public key, proof, biddable prices, and number of items to be sold. The registration phase proceeds as follows.

Registration. For each eligible bidder, the registrar constructs a (private) credential, sends the credential to the bidder, and derives the public credential by encrypting the credential with the auctioneer's public key.

[2] Our implementation is available from the following URL: `http://bensmyth.com/publications/2014-Hawk-and-Aucitas-auction-schemes/`.

Auction Scheme 1 Hawk

Suppose $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is an additively homomorphic asymmetric encryption scheme satisfying IND-CPA, Σ_1 proves correct key construction, Σ_2 proves plaintext knowledge in $\{0,1\}$ and Σ_3 proves decryption, where Π's message space is $\{0,1\}^*$. Further suppose $\mathcal{H}$ is a hash function and let $\mathsf{FS}(\Sigma_1, \mathcal{H}) = (\mathsf{ProveKey}, \mathsf{VerKey})$, $\mathsf{FS}(\Sigma_2, \mathcal{H}) = (\mathsf{ProveCiph}, \mathsf{VerCiph})$, and $\mathsf{FS}(\Sigma_3, \mathcal{H}) = (\mathsf{ProveDec}, \mathsf{VerDec})$. We define *Hawk* as $\Gamma(\Pi, \Sigma_1, \Sigma_2, \Sigma_3, \mathcal{H}) = (\mathsf{Setup}, \mathsf{Bid}, \mathsf{BB}, \mathsf{Open}, \mathsf{Reveal})$.

$\mathsf{Setup}(1^k)$. Select coins r, compute $(pk, sk, \mathfrak{m}) \leftarrow \mathsf{Gen}(1^k; r)$; $\rho \leftarrow \mathsf{ProveKey}((1^k, pk, \mathfrak{m}), (sk, r))$; $\mathbf{aux\text{-}pk} \leftarrow (1^k, \mathfrak{m}, \rho)$; $\mathfrak{bb} \leftarrow \emptyset$ and output $(pk, sk, \mathfrak{bb}, \mathbf{aux\text{-}pk})$. .

$\mathsf{Bid}(pk, \mathbf{aux\text{-}pk}, \mathbf{P}, p)$. Parse $\mathbf{aux\text{-}pk}$ as $(1^k, \mathfrak{m}, \rho)$, outputting $\perp$ if parsing fails or $\mathsf{VerKey}((1^k, pk, \mathfrak{m}), \rho) \neq \top$. Select coins $r_1, \ldots, r_{|\mathbf{P}|}$ and compute:

for $1 \leq i \leq |\mathbf{P}|$ **do**
  if $i = p$ **then** $m_i \leftarrow 1$ **else** $m_i \leftarrow 0$
  $c_i \leftarrow \mathsf{Enc}(pk, m_i; r_i)$; $\sigma_i \leftarrow \mathsf{ProveCiph}((pk, c_i), (m_i, r_i), i)$
$c \leftarrow c_1 \otimes \cdots \otimes c_{|\mathbf{P}|}$; $m \leftarrow m_1 \odot \cdots \odot m_{|\mathbf{P}|}$; $r \leftarrow r_1 \oplus \cdots \oplus r_{|\mathbf{P}|}$;
$\sigma_{|\mathbf{P}|+1} \leftarrow \mathsf{ProveCiph}((pk, c), (m, r), |\mathbf{P}| + 1)$

Output the bid $b = (c_1, \ldots, c_{|\mathbf{P}|}, \sigma_1, \ldots, \sigma_{|\mathbf{P}|+1})$.

$\mathsf{BB}(pk, \mathbf{P}, \mathfrak{bb}, b)$. Parse b as a vector $(c_1, \ldots, c_{|\mathbf{P}|}, \sigma_1, \ldots, \sigma_{|\mathbf{P}|+1})$. If parsing succeeds and $\bigwedge_{i=1}^{|\mathbf{P}|+1} \mathsf{VerCiph}((pk, c_i), \sigma_i, i) = \top$, where $c_{|\mathbf{P}|+1} \leftarrow c_1 \otimes \cdots \otimes c_{|\mathbf{P}|}$, then output $\mathfrak{bb} \cup \{b\}$, otherwise, output $\mathfrak{bb}$.

$\mathsf{Open}(pk, sk, \mathbf{P}, \mathfrak{bb}, M)$. Parse $\mathfrak{bb} = \{b_1, \ldots, b_n\}$ as a set of vectors of length $2 \cdot |\mathbf{P}| + 1$, outputting $(\perp, \perp)$ if parsing fails. Initialise index $p \leftarrow |\mathbf{P}| + 1$ and vector $\mathbf{aux\text{-}open} \leftarrow (\perp, \ldots, \perp)$ of length $|\mathbf{P}|$, and compute:

do
  $p \leftarrow p - 1$;
  $c \leftarrow b_1[p] \otimes \cdots \otimes b_n[p]$;
  $m \leftarrow \mathsf{Dec}(pk, sk, c)$; $\mathbf{aux\text{-}open}[p] \leftarrow \mathsf{ProveDec}((pk, c, m), sk)$;
  $M \leftarrow M - m$
while $M > 0 \wedge p > 0$;
if $M > 0$ **then** $p \leftarrow 0$

Output p and auxiliary data $\mathbf{aux\text{-}open}$.

$\mathsf{Reveal}(pk, sk, \mathbf{aux\text{-}pk}, \mathbf{P}, \mathfrak{bb}, M, p, \mathbf{aux\text{-}open})$. Parse $\mathfrak{bb} = \{b_1, \ldots, b_n\}$ as a set of vectors of length $2 \cdot |\mathbf{P}| + 1$, outputting $(\perp, \perp)$ if parsing fails. Initialise a set $w \leftarrow \emptyset$, vector $\mathbf{aux\text{-}reveal} \leftarrow (\perp, \ldots, \perp)$ of length n and integer $j \leftarrow 1$, and compute:

do
  $c \leftarrow b_j[p] \otimes \cdots \otimes b_j[|\mathbf{P}|]$;
  $m \leftarrow \mathsf{Dec}(pk, sk, c)$; $\mathbf{aux\text{-}reveal}[j] \leftarrow \mathsf{ProveDec}((pk, c, m), sk)$;
  if $m = 1$ **then** $w \leftarrow w \cup \{b_j\}$
  $j \leftarrow j + 1$
while $M > |w| \wedge j \leq n$;

Output $(w, \mathbf{aux\text{-}reveal})$.

The registrar authentically publishes the public credentials $\mathbf{L}$ and the bidding phase proceeds as follows.

Bidding. The bidder produces two ciphertexts under the auctioneer's public key: the first contains her price and the second contains her credential. In

addition, the bidder proves plaintext knowledge of both ciphertexts. The bidder sends the bid – namely, the ciphertexts and proof – to the auctioneer. The auctioneer verifies the bidder's proof and if verification succeeds, then the auctioneer publishes the bid on the bulletin board.

After some predefined deadline, the opening and revealing phases commence.

Opening. The auctioneer proceeds as follows.

- *Eliminating duplicates:* The auctioneer performs pairwise plaintext equality tests on the ciphertexts containing credentials and discards any bids for which a test holds, i.e., bids using the same credential are discarded.
- *Mixing:* The auctioneer mixes the ciphertexts in the bids (i.e., the ciphertexts containing prices and the ciphertexts containing credentials), using the same secret permutation for both mixes, hence, the mix preserves the relation between encrypted prices and credentials. Let $\mathbf{C_1}$ and $\mathbf{C_2}$ be the outputs of these mixes. The auctioneer also mixes the public credentials published by the registrar and assigns the output to $\mathbf{C_3}$.
- *Checking credentials:* The auctioneer discards ciphertexts $\mathbf{C_1}[i]$ from $\mathbf{C_1}$ if there is no ciphertext c in $\mathbf{C_3}$ such that a PET holds for c and $\mathbf{C_2}[i]$, that is, bids cast using ineligible credentials are discarded.
- *Decrypting:* The auctioneer decrypts the remaining encrypted prices in $\mathbf{C_1}$ and proves that decryption was performed correctly.

The auctioneer identifies the winning price from the decrypted prices.

Revealing. The auctioneer identifies ciphertexts $\mathbf{C_1}[i]$ containing prices greater than or equal to the winning price, and performs PETs between $\mathbf{C_2}[i]$ and $\mathbf{L}$ to reveal the identities of winning bidders.

Intuitively, every phase of the auction is verifiable and, hence, outcome and eligibility verifiability, and non-repudiation are derived from the individual, universal and eligibility verifiability properties of Civitas. Moreover, we shall define biddable prices from a starting price of 1 using price increments of 1 and a price ceiling equal to the size of the encryption scheme's message space, hence we have price flexibility. Furthermore, we derive collusion resistance from the coercion resistance property of Civitas.

5.1 Cryptographic construction

For our cryptographic construction of Aucitas, we extend the syntax for e-auctions schemes to include a registration algorithm, hence, an e-auction scheme is a tuple of algorithms $(\mathsf{Setup}, \mathsf{Register}, \mathsf{Bid}, \mathsf{BB}, \mathsf{Open}, \mathsf{Reveal})$ such that $\mathsf{Register}(pk, \textit{aux-pk}) \rightarrow (d, pd)$, where pk is the auctioneer's public key, $\textit{aux-pk}$ is auxiliary data, d is a (private) credential, and pd is a public credential. Moreover, we modify the input parameters of Bid, Open and Reveal, namely, $\mathsf{Bid}(d, pk, \textit{aux-pk}, \mathbf{P}, p) \rightarrow b$, $\mathsf{Open}(pk, sk, \textit{aux-pk}, \mathbf{P}, \mathfrak{bb}, M, \mathbf{L}) \rightarrow (p, \textit{aux-open})$ and $\mathsf{Reveal}(pk, sk, \textit{aux-pk}, \mathbf{P}, \mathfrak{bb}, p, \textit{aux-open}, \mathbf{L}) \rightarrow (\mathbf{L}', \textit{aux-reveal})$, where d is a bidder's credential, $\mathbf{L}$ and $\mathbf{L}'$ are vectors of public credentials, and the remaining

inputs and outputs are as per Section 3. We define a mixnet as $\mathsf{Mix}(\mathbf{c}) \rightarrow (\mathbf{c}', \rho)$ such that $\mathbf{c}'$ contains a permutation of the ciphertexts in $\mathbf{c}$ after re-encryption and ρ is a proof that the mix has been performed correctly. For brevity, we omit a formal definition and refer the reader to Jakobsson, Juels & Rivest [12].

We present Aucitas in Auction Scheme 2. The Setup algorithm generates the auctioneer's key pair using an asymmetric encryption scheme, proves that the key has been correctly constructed, and initialises the bulletin board. The scheme is price flexible using biddable prices $\mathbf{P} = (1, 2, \ldots, |\mathfrak{m}|)$, where $\mathfrak{m}$ is the encryption scheme's message space. The Register algorithm generates bidders' credentials and we assume that the auctioneer provides the bidder with a credential d corresponding to a public credential $\mathsf{Enc}(pk, d)$; this assumption can be dropped using designated verifier proofs, for example. The specification of the Bid, BB, Open and Reveal algorithms follow from our informal description. We demonstrate an execution of Aucitas in the long version [16, Figure 3] of this paper.

Intuitively, collusion resistance is satisfied if a bidder can convince a conspirator that they behaved as instructed, when they actually behaved differently. In Aucitas, this condition is satisfied as follows: given an instruction, a bidder generates a fake credential and follows the instruction using the fake credential. For instance, if the bidder is instructed to bid for a particular price, then the bidder constructs a bid for the price using the fake credential. It follows from the description of Aucitas that this bid will be removed during credential checking, however, the adversary will be unable to detect this, assuming at least one bidder bids at the adversary's price. We acknowledge that price flexibility and collusion resistance are conflicting properties – allowing bidders to submit any price decreases the probability that at least one bidder bids the price instructed by an adversary – and we can balance the degree of price flexibility and collusion resistance by restricting the prices.

6 Related work

Magkos, Alexandris & Chrissikopoulos [15] and Her, Imamot & Sakurai [10] also study the relation between e-auction and e-voting schemes. Magkos, Alexandris & Chrissikopoulos remark that e-voting and e-auction schemes have a similar structure and share similar security properties. Her, Imamot & Sakurai contrast privacy properties of e-voting and e-auctions, and compare the use of homomorphic encryption and mixnets between domains. Our work is distinguished from these earlier works, since we *demonstrate* a relation between e-auction and e-voting schemes.

Lipmaa, Asokan & Niemi [14] propose an e-auction scheme, based upon homomorphic encryption, which is similar to the e-voting scheme proposed by Damgård, Jurik & Nielsen [8] (although the similarities are not explicitly discussed) and Hawk. In essence, their scheme is defined as follows: 1) encrypted bids are sent to the seller during the bidding phase, 2) these encrypted bids are homomorphically combined by the seller in the opening phase and the homo-

Auction Scheme 2 Aucitas

Suppose $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is a homomorphic asymmetric encryption scheme satisfying IND-CPA, Σ_1 proves correct key construction, Σ_2 proves correct ciphertext construction, Σ_3 proves decryption, Σ_4 is a PET, and $\mathcal{H}$ is a hash function. Let $\mathsf{FS}(\Sigma_1, \mathcal{H}) = (\mathsf{ProveKey}, \mathsf{VerKey})$, $\mathsf{FS}(\Sigma_2, \mathcal{H}) = (\mathsf{ProveBind}, \mathsf{VerBind})$, $\mathsf{FS}(\Sigma_3, \mathcal{H}) = (\mathsf{ProveDec}, \mathsf{VerDec})$, and $\mathsf{FS}(\Sigma_4, \mathcal{H}) = (\mathsf{ProvePET}, \mathsf{VerPET})$. We define *Aucitas* below.

$\mathsf{Setup}(1^k)$. Select coins r, compute $(pk, sk, \mathfrak{m}) \leftarrow \mathsf{Gen}(1^k; r); \rho \leftarrow \mathsf{ProveKey}((1^k, pk, \mathfrak{m}), (sk, r)); \mathfrak{bb} \leftarrow \emptyset; \mathbf{aux\text{-}pk} \leftarrow (1^k, \mathfrak{m}, \rho)$ and output $(pk, sk, \mathfrak{bb}, \mathbf{aux\text{-}pk})$.

$\mathsf{Register}(pk, \mathbf{aux\text{-}pk})$. Parse $\mathbf{aux\text{-}pk}$ as $(1^k, \mathfrak{m}, \rho)$, outputting $(\perp, \perp)$ if parsing fails. Assign a random element from $\mathfrak{m}$ to d and compute $pd \leftarrow \mathsf{Enc}(pk, d)$ and output (d, pd).

$\mathsf{Bid}(d, pk, \mathbf{aux\text{-}pk}, \mathbf{P}, p)$. Parse $\mathbf{aux\text{-}pk}$ as $(1^k, \mathfrak{m}, \rho)$, outputting $\perp$ if parsing fails or $\mathsf{VerKey}((1^k, \mathfrak{m}, \rho), \rho) \neq \top$. Suppose $\mathfrak{m} = \{m_1, \ldots, m_{|\mathfrak{m}|}\}$ such that $m_1 < \cdots < m_{|\mathfrak{m}|}$. Select coins r_1 and r_2, compute $c_1 \leftarrow \mathsf{Enc}(pk, m_p; r_1); c_2 \leftarrow \mathsf{Enc}(pk, d; r_2); \sigma \leftarrow \mathsf{ProveBind}((pk, c_1, c_2), (m_p, r_1, d, r_2)); b \leftarrow (c_1, c_2, \sigma)$ and output bid b.

$\mathsf{BB}(pk, \mathbf{P}, \mathfrak{bb}, b)$. Parse b as (c_1, c_2, σ). If parsing succeeds and $\mathsf{VerBind}((pk, c_1, c_2), \sigma) = \top$, then output $\mathfrak{bb} \cup \{b\}$, otherwise, output $\mathfrak{bb}$.

$\mathsf{Open}(pk, sk, \mathbf{aux\text{-}pk}, \mathbf{P}, \mathfrak{bb}, M, \mathbf{L})$. Parse $\mathbf{aux\text{-}pk}$ as $(1^k, \mathfrak{m}, \rho)$ and $\mathfrak{bb} = \{b_1, \ldots, b_n\}$ as a set of vectors of length 3, outputting $(\perp, \perp)$ if parsing fails. Proceed as follows.

- Eliminating duplicates: Let $\mathbf{aux\text{-}dupl}$ be a vector of length n and $\mathbf{BB}$ be the empty vector. For each $1 \leq i \leq n$, if there exists σ and $j \in \{1, \ldots, i-1, i+1, \ldots, n\}$ such that $\sigma \leftarrow \mathsf{ProvePET}((pk, b_i[2], b_j[2], 1), sk)$ and $\mathsf{VerPET}((pk, b_i[2], b_j[2], 1), \sigma) = \top$, then assign $\mathbf{aux\text{-}dupl}[i] \leftarrow \sigma$, otherwise, compute $\sigma_j \leftarrow \mathsf{ProvePET}((pk, b_i[2], b_j[2], 0), sk)$ for each $j \in \{1, \ldots, i-1, i+1, \ldots, n\}$ and assign $\mathbf{aux\text{-}dupl}[i] \leftarrow (\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_n); \mathbf{BB} \leftarrow \mathbf{BB} \parallel (b_i)$, where $\mathbf{BB} \parallel (b_i)$ denotes the concatenation of vectors $\mathbf{BB}$ and (b_i), i.e., $\mathbf{BB} \parallel (b_i) = (\mathbf{BB}[1], \ldots, \mathbf{BB}[|\mathbf{BB}|], b_i)$.
- Mixing: Suppose $\mathbf{BB} = (b'_1, \ldots, b'_\ell)$, select coins r, and compute $(\mathbf{C_1}, \mathbf{aux\text{-}mix_1}) \leftarrow \mathsf{Mix}((b'_1[1], \ldots, b'_\ell[1]); r); (\mathbf{C_2}, \mathbf{aux\text{-}mix_2}) \leftarrow \mathsf{Mix}((b'_1[2], \ldots, b'_\ell[2]); r); (\mathbf{C_3}, \mathbf{aux\text{-}mix_3}) \leftarrow \mathsf{Mix}(\mathbf{L})$.
- Checking credentials: Let $\mathbf{aux\text{-}cred}$ be a vector of length $|\mathbf{C_2}|$. For each $1 \leq i \leq |\mathbf{C_2}|$, if there exists σ and $c \in \mathbf{C_3}$ such that $\sigma \leftarrow \mathsf{ProvePET}((pk, \mathbf{C_2}[i], c, 1), sk)$ and $\mathsf{VerPET}((pk, \mathbf{C_2}[i], c, 1), \sigma) = \top$, then assign $\mathbf{aux\text{-}cred}[i] \leftarrow \sigma$, otherwise, compute $\sigma_j \leftarrow \mathsf{ProvePET}((pk, \mathbf{C_2}[i], \mathbf{C_3}[j], 0), sk)$ for each $j \in \{1, \ldots, |\mathbf{C_3}|\}$ and assign $\mathbf{aux\text{-}cred}[i] \leftarrow (\sigma_1, \ldots, \sigma_{|\mathbf{C_3}|})$.
- Decrypting: Let *aux-dec* be the empty set. For each $1 \leq i \leq |\mathbf{C_1}|$ such that $|\mathbf{aux\text{-}cred}[i]| = 1$ assign $aux\text{-}dec \leftarrow aux\text{-}dec \cup \{((\mathbf{C_1}[i], \mathbf{C_2}[i]), \sigma, m)\}$, where $m \leftarrow \mathsf{Dec}(pk, sk, \mathbf{C_1}[i])$ and $\sigma \leftarrow \mathsf{ProveDec}((pk, \mathbf{C_1}[i], m), sk)$.

If $|aux\text{-}dec| < M$, then output $(0, \perp)$. Otherwise, output $(p, \mathbf{aux\text{-}open})$, where $p \in \{1, \ldots, |\mathfrak{m}|\}$ is the largest integer such that M integers in the set $\{m \mid (b, \sigma, m) \in aux\text{-}dec\}$ are greater than or equal to m_p, and $\mathbf{aux\text{-}open} \leftarrow (\mathbf{aux\text{-}dupl}, \mathbf{aux\text{-}mix_1}, \mathbf{aux\text{-}mix_2}, \mathbf{aux\text{-}mix_3}, \mathbf{aux\text{-}cred}, aux\text{-}dec)$.

$\mathsf{Reveal}(pk, sk, \mathbf{aux\text{-}pk}, \mathbf{P}, \mathfrak{bb}, M, p, \mathbf{aux\text{-}open}, \mathbf{L})$. Let $aux\text{-}dec \leftarrow \mathbf{aux\text{-}open}[6]$. Parse $\mathbf{aux\text{-}pk}$ as $(1^k, \mathfrak{m}, \rho)$ and *aux-dec* as a set of vectors of length 3, outputting $(\perp, \perp)$ if parsing fails. Suppose $\mathfrak{m} = \{m_1, \ldots, m_{|\mathfrak{m}|}\}$ such that $m_1 < \cdots < m_{|\mathfrak{m}|}$. If there exist M distinct triples $(b_1, \sigma_1, m'_1), \ldots, (b_M, \sigma_M, m'_M) \in aux\text{-}dec$ and ciphertexts $c_1, \ldots, c_M \in \mathbf{L}$ such that for each $1 \leq i \leq M$ we have $\mathsf{VerPET}((pk, b_i[2], c_i, 1), \tau_i) = \top \wedge m'_i \geq m_p$, where $\tau_i \leftarrow \mathsf{ProvePET}((pk, b_i[2], c_i, 1), sk)$, then output $((c_1, \ldots, c_M), (\tau_1, \ldots, \tau_M))$, otherwise, output $(\perp, \perp)$.

morphic combination is decrypted by the auctioneer, and 3) bidders demonstrate to sellers that they are winning bidders during the reveal phase. Their scheme satisfies bid secrecy under the assumption that either the seller or auctioneer is trusted; by comparision, Hawk assumes that the auctioneer is trusted. This suggests that Hawk requires a stronger trust assumption, however, as we have discussed (Section 3), we can mitigate against the possibility that the auctioneer is dishonest by distributing trust amongst several auctioneers and, hence, the trust assumptions of Hawk and the scheme by Lipmaa, Asokan & Niemi are similar in the case that the seller is also an auctioneer. In addition, Lipmaa, Asokan & Niemi claim that their e-auction scheme could be used to construct an e-voting scheme [14, §9]; by comparision, we focus on the inverse, i.e., the construction of e-auction schemes from e-voting schemes.

Abe & Suzuki [1] propose an e-auction scheme based upon homomorphic encryption. Their scheme satisfies bid secrecy and a complimentary privacy property: with the exception of the winning price, prices are not revealed (this property helps protect bidding strategies, for example). The scheme is similar to Hawk until the opening phase, but differs thereafter, using Jakobsson & Juels's *mix and match* technique [11] to find the winning price, for instance. By contrast, Hawk is conceptually simpler.

Peng *et al.* [17] propose an e-auction schemes based upon mixnets, however, unlike Aucitas, they focus on bid secrecy rather than collusion resistance. Abe & Suzuki [2] introduce an e-auction scheme using trapdoor bit-commitments and Chen, Lee & Kim [6] introduce a scheme using mixnets; these two schemes satisfy collusion resistance. However, Abe & Suzuki assume the existence of a *bidding booth*, where the bidder must bid and cannot communicate with a conspirator, and Chen, Lee & Kim assume the seller is trusted. By comparision, Aucitas achieves collusion resistance without such assumptions.

Acknowledgements. We are particularly grateful to Florian Kerschbaum and the anonymous reviewers who read earlier versions of this paper and provided useful guidance. This work has been partly supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC project *CRYSP* (259639), the ANR-09-VERS-016 BEST project, and Campus France.

References

1. Abe, M., Suzuki, K.: M + 1-st price auction using homomorphic encryption. In: PKC'02: 5th International Workshop on Practice and Theory in Public Key Cryptography. Volume 2274 of LNCS., Springer (2002) 115–124
2. Abe, M., Suzuki, K.: Receipt-free sealed-bid auction. In: Information Security. Volume 2433 of LNCS. Springer (2002) 191–199
3. Adida, B., Marneffe, O., Pereira, O., Quisquater, J.: Electing a University President Using Open-Audit Voting: Analysis of Real-World Use of Helios. In: EVT/WOTE'09: Electronic Voting Technology Workshop/Workshop on Trustworthy Elections, USENIX Association (2009)

4. Bernhard, D., Cortier, V., Pereira, O., Smyth, B., Warinschi, B.: Adapting Helios for provable ballot privacy. In: ESORICS'11: 16th European Symposium on Research in Computer Security. Volume 6879 of LNCS., Springer (2011) 335–354
5. Bernhard, D., Pereira, O., Warinschi, B.: How Not to Prove Yourself: Pitfalls of the Fiat-Shamir Heuristic and Applications to Helios. In: ASIACRYPT'12: 18th International Conference on the Theory and Application of Cryptology and Information Security. Volume 7658 of LNCS., Springer (2012) 626–643
6. Chen, X., Lee, B., Kim, K.: Receipt-Free Electronic Auction Schemes Using Homomorphic Encryption. In: ICISC'03: 6th International Conference on Information Security and Cryptology. Volume 2971 of LNCS., Springer (2003) 259–273
7. Clarkson, M.R., Chong, S., Myers, A.C.: Civitas: Toward a Secure Voting System. In: S&P'08: 29th Security and Privacy Symposium, IEEE Computer Society (2008) 354–368
8. Damgård, I., Jurik, M., Nielsen, J.B.: A Generalization of Paillier's Public-Key System with Applications to Electronic Voting. International Journal of Information Security **9**(6) (2010) 371–385
9. Fiat, A., Shamir, A.: How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In: CRYPTO'86: 6th International Cryptology Conference. Volume 263 of LNCS., Springer (1987) 186–194
10. Her, Y.S., Imamoto, K., Sakurai, K.: Analysis and comparison of cryptographic techniques in e-voting and e-auction. Technical Report 10(2), Information Science and Electrical Engineering, Kyushu University (September 2005)
11. Jakobsson, M., Juels, A.: Mix and match: Secure function evaluation via ciphertexts. In: Advances in CryptologyASIACRYPT 2000. Springer (2000) 162–177
12. Jakobsson, M., Juels, A., Rivest, R.L.: Making Mix Nets Robust for Electronic Voting by Randomized Partial Checking. In: 11th USENIX Security Symposium. (2002) 339–353
13. Juels, A., Catalano, D., Jakobsson, M.: Coercion-Resistant Electronic Elections. In Chaum, D., Jakobsson, M., Rivest, R.L., Ryan, P.Y., eds.: Towards Trustworthy Elections: New Directions in Electronic Voting. Volume 6000 of LNCS. Springer (2010) 37–63
14. Lipmaa, H., Asokan, N., Niemi, V.: Secure Vickrey Auctions without Threshold Trust. In: FC'02: 6th International Conference on Financial Cryptography and Data Security. Volume 2357 of LNCS., Springer (2002) 87–101
15. Magkos, E., Alexandris, N., Chrissikopoulos, V.: A Common Security Model for Conducting e-Auctions and e-Elections. CSCC'02: 6th WSEAS International Multiconference on Circuits, Systems, Communications and Computers `http://www.wseas.us/e-library/conferences/crete2002/papers/444-766.pdf` (2002)
16. McCarthy, A., Smyth, B., Quaglia, E.A.: Hawk and Aucitas: e-auction schemes from the Helios and Civitas e-voting schemes. `http://bensmyth.com/publications/2014-Hawk-and-Aucitas-auction-schemes/` (2014)
17. Peng, K., Boyd, C., Dawson, E., Viswanathan, K.: Efficient implementation of relative bid privacy in sealed-bid auction. In: Information Security Applications. Volume 2908 of LNCS. Springer (2004) 244–256
18. Smyth, B., Bernhard, D.: Ballot secrecy and ballot independence coincide. In: ESORICS'13: 18th European Symposium on Research in Computer Security. Volume 8134 of LNCS., Springer (2013) 463–480
19. Zhou, X., Zheng, H.: Breaking bidder collusion in large-scale spectrum auctions. In: MobiHoc'10: 11th ACM international symposium on Mobile ad hoc networking and computing, ACM Press (2010) 121–130