Attacking Privacy in a Fully Private Auction Protocol

Jannik Dreier, Jean-Guillaume Dumas, Pascal Lafourcade

Attacking Privacy in a Fully Private Auction Protocol

Jannik Dreier*‡ Jean-Guillaume Dumas†‡ Pascal Lafourcade*‡

October 25, 2012

Abstract

Auctions have a long history, having been recorded as early as 500 B.C. With the rise of Internet, electronic auctions have been a great success and are increasingly used. Many cryptographic protocols have been proposed to address the various security requirements of these electronic transactions, in particular to ensure privacy. In 2006 Brandt [1] developed a protocol that computes the winner using homomorphic operations on a distributed ElGamal encryption of the bids. He claimed that it ensures full privacy of the bidders, i.e. that no information apart from the winner and the winning price is leaked. We show that this protocol – when using interactive zero-knowledge proofs – is vulnerable to attacks by dishonest bidders. Such bidders can manipulate the publicly available data in a way that allows the seller to deduce all participants' bids. Additionally, even if non-interactive zero-knowledge proofs are used, we show that the protocol is vulnerable to a different attack, which allows to recover one targeted bidder's bid.

1 Introduction

Auctions are a simple method to sell goods and services. Typically a *seller* offers a good or a service, and the *bidders* make offers. Depending on the type of auction, the offers might be sent using sealed envelopes which are opened simultaneously to determine the winner (the "sealed-bid" auction), or an *auctioneer* could announce prices decreasingly until one bidder is willing to pay the announced price (the "dutch auction"). Additionally there might be several rounds, or offers might be announced publicly directly (the "English" or "shout-out" auction). The winner usually is the bidder submitting the highest bid, but in some cases he might only have to pay the second highest offer as a price (the "second-price"- or "Vickrey"-Auction). In general a bidder wants to win the auction at the lowest possible price, and the seller wants to sell his good at the highest possible price. For more information on different auction methods

*Université de Grenoble. Laboratoire Verimag, umr CNRS 5104, Centre quation - 2, avenue de Vignate F38610 Gières, France

†Université de Grenoble. Laboratoire J. Kuntzmann, umr CNRS 5224, 51, rue des Mathématiques, BP 53X, F38041 Grenoble, France

‡{Jannik.Dreier,Jean-Guillaume.Dumas,Pascal.Lafourcade}@imag.fr

see [8]. To address this huge variety of possible auction settings and to achieve different security and efficiency properties numerous protocols have been developed, e.g. [1, 10, 3, 9, 12, 11, 7, 4, 14, 13, 16, 15, 20, 19, 17, 18].

One of the key requirements of electronic auction (e-Auction) protocols is privacy, i.e. that the bids of losing bidders remain private. In 2006 Brandt [1] proposed a first-price sealed-bid auction protocol and claimed that it is fully private, i.e. that it leaks no information apart from the winner, the winning bid, and what can be deduced from these two facts (for example that the other bids were inferior).

Our Contributions. The protocol is based on an algorithm that computes the winner using bids encoded as bit vectors. In this paper we show that the implementation using the homomorphic property of a distributed Elgamal encryption proposed in the original paper suffers from a weakness that can in some cases be exploited by dishonest participants. In fact, we prove that any two different inputs (i.e. different bids) result in different outcome values, which are only hidden using random values. We show how a dishonest participant can remove this random noise, if interactive zero-knowledge proofs are used. The seller can then efficiently compute the bids of all bidders, hence completely breaking privacy. Next, we also show that even if this attack is prevented via non-interactive proofs, the protocol remains vulnerable to attacks on the privacy of a single bidder. This is due to a lack of authentication.

Outline. In the next section, we recall the protocol by Brandt. Then, in Section 3, we present our attacks in several steps. We first study the protocol using interactive zero-knowledge proofs and without noise. Then we show how a dishonest participant can remove the noise. Finally, as the messages are not authenticated, we show that a malicious attacker in control of the network can recover any bidder's bid, even if non-interactive zero-knowledge proofs.

2 The Protocol

The protocol of Brandt [1] was designed to ensure full privacy in a completely distributed way. It exploits the homomorphic properties of a distributed El-Gamal Encryption scheme [5] for a secure multi-party computation of the winner. We first give a high level description of the protocol and then present details on the main cryptographic primitives it uses.

2.1 Informal Description

The participating n bidders and the seller communicate essentially using broadcast messages. The latter can for example be implemented using a bulletin board, i.e. an append-only memory accessible to everybody. The bids are encoded as k-bit-vectors where each entry corresponds to a price. If the bidder a wants to bid the price b_a, all entries will be 1, except the entry b_a which will be Y (a public constant). Each entry of the vector is then encrypted separately using a n-out-of-n-encryption scheme set up by all bidders. The bidders use multiplications of the encrypted bids (exploiting

the homomorphic property) to compute values v_{aj}. Each one of this values is 1 if the bidder a wins at price j, and is a random number otherwise. The decryption of the final values takes place in a distributed way to ensure that nobody can access intermediate values.

In a nutshell, the protocol realizes the following steps:

1. First, the distributed key is generated: each bidder chooses his part of the secret key and publishes the corresponding part of the public key.
2. Each bidder then computes the joint public key, encrypts his bid-vector entry-wise using this key and publishes the result.
3. Then the auction function is computed for every bidder using the homomorphic properties of the encryption scheme, see next paragraph.
4. The outcome of this computation ($n^2 \cdot k$ encrypted values) are published on the bulletin board, and each bidder partly decrypts each value using his secret key.
5. These shares are sent to the seller, who can combine them to obtain the result. The seller also publishes part of the shares so that each bidder can verify his winning or loosing situation.

2.2 Mathematical Description

We do not detail all proofs here except the one used in step 3 "Outcome computation". Indeed, our attack exploits the algebraic properties of the latter. We consider that $i, h \in \{1, \ldots, n\}$, $j, bid_a \in \{1, \ldots, k\}$ (where bid_a is the bid chosen by the bidder with index a), $Y \in \mathbb{G}_q \setminus \{1\}$. More precisely, the n bidders execute the following five steps of the protocol [1]:

1. **Key Generation**
   Each bidder a, whose bidding price is bid_a among k offers:
   - chooses a secret $x_a \in \mathbb{Z}_q$
   - choose randomly m_{ij}^a and $r_{aj} \in \mathbb{Z}_q$ for each i and j.
   - publishes $y_a = g^{x_a}$ and proves the knowledge of y_a's discrete logarithm.
   - then computes $y = \prod_{i=1}^{n} y_i$.
2. **Bid Encryption**
   Each bidder a
   - sets $b_{aj} = \begin{cases} Y & \text{if } j = bid_a \\ 1 & \text{otherwise} \end{cases}$
   - publishes $\alpha_{aj} = b_{aj} \cdot y^{r_{aj}}$ and $\beta_{aj} = g^{r_{aj}}$ for each j.

- proves that for all j, $\log_g(\beta_{aj})$ equals $\log_y(\alpha_{aj})$ or $\log_y\left(\frac{\alpha_{aj}}{Y}\right)$, and that $\log_y\left(\frac{\prod_{j=1}^{k}\alpha_{aj}}{Y}\right) = \log_g\left(\prod_{j=1}^{k}\beta_{aj}\right)$.

3. **Outcome Computation**

   - Each bidder a computes and publishes for all i and j:

$$\gamma_{ij}^{a} = \left(\left(\prod_{h=1}^{n}\prod_{d=j+1}^{k}\alpha_{hd}\right)\cdot\left(\prod_{d=1}^{j-1}\alpha_{id}\right)\cdot\left(\prod_{h=1}^{i-1}\alpha_{hj}\right)\right)^{m_{ij}^{a}}$$

$$\delta_{ij}^{a} = \left(\left(\prod_{h=1}^{n}\prod_{d=j+1}^{k}\beta_{hd}\right)\cdot\left(\prod_{d=1}^{j-1}\beta_{id}\right)\cdot\left(\prod_{h=1}^{i-1}\beta_{hj}\right)\right)^{m_{ij}^{a}}$$

     and proves its correctness.

4. **Outcome Decryption**

   - Each bidder a sends $\phi_{ij}^{a} = (\prod_{h=1}^{n}\delta_{ij}^{h})^{x_a}$ for each i and j to the seller and proves its correctness. After having received all values, the seller publishes ϕ_{ij}^{h} for all i, j, and $h \neq i$.

5. **Winner determination**

   - Everybody can now compute $v_{aj} = \frac{\prod_{i=1}^{n}\gamma_{aj}^{i}}{\prod_{i=1}^{n}\phi_{aj}^{i}}$ for each j.
   - If $v_{aw} = 1$ for some w, then the bidder a wins the auction at price p_w.

2.3 Proof of equality of two discrete logs

In the original paper [1] the following interactive proof [2] is proposed to prove the correctness of γ_{ij}^{a} and δ_{ij}^{a} in step 3 of the protocol: Peggy and Victor know v, w, g_1 and g_2, and Peggy wants to prove that she knows x such that $v = g_1^x$ and $w = g_2^x$; here $g_1 = \gamma_{ij}^{a}$, $g_2 = \delta_{ij}^{a}$ and $x = m_{ij}^{a}$:

1. Peggy chooses z at random and sends $\lambda = g_1^z$ and $\mu = g_2^z$ to Victor.
2. Victor chooses a challenge c at random and sends it to Peggy.
3. Peggy computes $r = (z + c \cdot x) \mod q$ and sends it to Victor.
4. Victor tests if $g_1^r = \lambda \cdot v^c$ and $g_2^r = \mu \cdot w^c$.

This interactive protocol can be converted into a non-interactive one using the Fiat-Shamir heuristic [6].

3 The Attacks

We present two kinds of attacks. The first one uses some algebraic properties of the computations performed during the protocol execution. Then we discuss other attacks based on the lack of authentication of the protocol.

3.1 Attacking the fully private computations

For our main attack we analyze the computations done in step 3 of the protocol. Consider the following example with three bidders and three possible prices. Then the first bidder computes

$$\begin{array}{lccccccccccc}
\gamma^1_{11} = (& (\alpha_{12}\cdot\alpha_{13}\cdot & \alpha_{22}\cdot\alpha_{23}\cdot & \alpha_{32}\cdot\alpha_{33}) & \cdot & (1) & \cdot & (1) &)^{m^1_{11}} \\
\gamma^1_{12} = (& (\alpha_{13}\cdot & \alpha_{23}\cdot & \alpha_{33}) & \cdot & (\alpha_{11}) & \cdot & (1) &)^{m^1_{12}} \\
\gamma^1_{13} = (& & & (1) & \cdot & (\alpha_{11}\cdot\alpha_{12}) & \cdot & (1) &)^{m^1_{13}} \\
\gamma^1_{21} = (& (\alpha_{12}\cdot\alpha_{13}\cdot & \alpha_{22}\cdot\alpha_{23}\cdot & \alpha_{32}\cdot\alpha_{33}) & \cdot & (1) & \cdot & (\alpha_{11}) &)^{m^1_{21}} \\
\gamma^1_{22} = (& (\alpha_{13}\cdot & \alpha_{23}\cdot & \alpha_{33}) & \cdot & (\alpha_{21}) & \cdot & (\alpha_{12}) &)^{m^1_{22}} \\
\gamma^1_{23} = (& & & (1) & \cdot & (\alpha_{21}\cdot\alpha_{22}) & \cdot & (\alpha_{13}) &)^{m^1_{23}} \\
\gamma^1_{31} = (& (\alpha_{12}\cdot\alpha_{13}\cdot & \alpha_{22}\cdot\alpha_{23}\cdot & \alpha_{32}\cdot\alpha_{33}) & \cdot & (1) & \cdot & (\alpha_{11}\cdot\alpha_{21}) &)^{m^1_{31}} \\
\gamma^1_{32} = (& (\alpha_{13}\cdot & \alpha_{23}\cdot & \alpha_{33}) & \cdot & (\alpha_{31}) & \cdot & (\alpha_{12}\cdot\alpha_{22}) &)^{m^1_{32}} \\
\gamma^1_{33} = (& & & (1) & \cdot & (\alpha_{31}\cdot\alpha_{32}) & \cdot & (\alpha_{13}\cdot\alpha_{23}) &)^{m^1_{33}}
\end{array}$$

The second and third bidder do the same computations, but using different random values m^a_{ij}. Since each α_{ij} is either the encryption of 1 or Y, for example the value γ^1_{22} will be an encryption of 1 only if

- nobody submitted a higher bid (the first block) and
- bidder 2 did not bid a lower bid (the second block) and
- no bidder with a lower index submitted the same bid (the third block).

If we ignore the exponentiation by m^a_{ij}, each γ^a_{ij} is the encryption of the product of several b_{ij}'s. Each b_{ij} can be either 1 or Y, hence $(\gamma^a_{ij})^{-m^a_{ij}}$ will be the encryption of a value $Y^{l_{ij}}$, where $0 \le l_{ij} \le n$. The lower bound of l_{ij} is trivial, the upper bound follows from the observation that each α_{ij} will be used at most once, and that each bidder will encrypt Y at most once.

Assume for now that we know all l_{ij}. We show next that this is sufficient to obtain all bids. Consider the function f which takes as input the following vector[1]

$$b = \log_Y \left((\ b_{11}, \ \ldots, \ b_{1k}, \quad b_{21}, \ \ldots, \ b_{2k}, \quad \ldots, \quad b_{n1}, \ \ldots, \ b_{nk}\ )^T \right)$$

and returns the values l_{ij}. The input vector is thus a vector of all bid-vectors, where 1 is replaced by 0 and Y by 1. Consider our above example with three bidders and three possible prices, then we have

$$b = \log_Y \left((\ b_{11}, \ b_{12}, \ b_{13}, \quad b_{21}, \ b_{22}, \ b_{23}, \quad b_{31}, \ b_{32}, \ b_{33}\ )^T \right).$$

[1] By abuse of notation we write $log_s\,(x_1, \ \ldots, \ x_n)$ for $(log_s(x_1), \ \ldots, \ log_s(x_n))$.

A particular instance where bidder 1 and 3 submit price 1, and bidder 2 submits price 2 would then look as follows

$$b = \big(\ 1,\quad 0,\quad 0,\qquad 0,\quad 1,\quad 0,\qquad 1,\quad 0,\quad 0\ \big)^T$$

Hence only the factors α_{11}, α_{22} and α_{31} are encryptions of Y, all other α's are encryptions of 1. By simply counting how often the factors α_{11}, α_{22} and α_{31} show up in each equation as described above, we can compute the following result of $f(b)$:

$$\big(\ 1,\quad 1,\quad 1,\qquad 2,\quad 0,\quad 1,\qquad 2,\quad 1,\quad 1\ \big)^T$$

Note that since we chose the input of f to be a bit-vector, we have to simply count the ones (which correspond to Y's) in particular positions in b, where the positions are determined by the factors inside γ_{ij}^a. Hence we can express f as a matrix, i.e. $f(b) = M \cdot b$ for the following matrix M:

$$f(b) = M \cdot b = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{1} \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \mathbf{1} \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 2 \\ \mathbf{0} \\ 1 \\ 2 \\ 1 \\ 1 \end{pmatrix}$$

To see how the matrix M is constructed, consider for example

$$(\gamma_{22}^a)^{-m_{22}^a} = (\alpha_{13} \cdot \alpha_{23} \cdot \alpha_{33}) \cdot (\alpha_{21}) \cdot (\alpha_{12})$$

which corresponds to the **second row** in the second vertical block:

- α_{12} and α_{13}; hence the two ones at position 2 and 3 in the first horizontal block
- α_{21} and α_{23}; hence the two ones at position 1 and 3 in the second horizontal block
- α_{33}; hence the one at position 3 in the third horizontal block

More generally, we can see that we each 3×3 block consists of potentially three parts:

- An upper triangular matrix representing all bigger bids.
- On the diagonal we add a lower triangular matrix representing a lower bid by the same bidder,
- In the lower left half we add an identity matrix representing a bid at the current price by a bidder with a lower index.

This corresponds exactly to the structure of the products inside each γ_{ij}^a. It is also equivalent to formula (1) in Section 4.1.1 of the original paper [1] without the random vector R_k^*. In the following we prove that the function f is injective. We then discuss how this function can be efficiently inverted (i.e. how to compute the bids when knowing all l_{ij}'s).

3.1.1 Preliminaries.

Let I_k be the $k \times k$ identity matrix.

Let L_k be a lower $k \times k$ triangular matrix with zeroes on the diagonal, ones in the lower part and zeroes elsewhere. Let U_k be an upper $k \times k$ triangular matrix with zeroes on the diagonal, ones in the upper part, and zeroes elsewhere. In Figure 1 we give a representation of these matrices. By abuse of notation we use I, L and U to denote respectively I_k, L_k and U_k.

$$I_k = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \quad L_k = \begin{bmatrix} 0 & & & & 0 \\ 1 & 0 & & & \\ 1 & 1 & \ddots & & \\ \vdots & \vdots & \ddots & \ddots & \\ 1 & 1 & \dots & 1 & 0 \end{bmatrix} \quad U_k = \begin{bmatrix} 0 & 1 & 1 & \dots & 1 \\ & 0 & 1 & \dots & 1 \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & 1 \\ 0 & & & & 0 \end{bmatrix}$$

Figure 1: I_k, L_k and U_k

For a $k \times k$-matrix M_k we define $(M_k)^r = M \cdots M$ (r times) and $(M_k)^0 = I_k$. Let $(e_1, \dots, e_k)$ be the canonical basis.

Lemma 1. *Matrices L_k and U_k have the following properties, for $0 < j \leq k$ and $r \geq 0$:*

$$(U_k)^r \cdot e_j = \sum_{s=1}^{j-r} e_s \text{ and } (L_k)^r \cdot e_j = \sum_{s=j+r}^{k} e_s$$

Lemma 2. *Matrices L_k and U_k are nilpotent, i.e. $(U_k)^k = 0$ and $(L_k)^k = 0$.*

This lemma follows immediately from Lemma 1 by computing $(U_k)^k \cdot I_k$ and $(L_k)^k \cdot I_k$.

Lemma 3. *If $\sum_{i=1}^{k} x_i = 1$ then we have $L_k \cdot x = \mathbf{1} - (I_k + U_k) \cdot x$, where $\mathbf{1} = (1, \dots, 1)^T$.*

Proof. First note that since $\sum_{i=1}^{k} x_i = 1$,

$$L_k \cdot x = \begin{bmatrix} 0 & & & & 0 \\ 1 & 0 & & & \\ 1 & 1 & \ddots & & \\ \vdots & \vdots & \ddots & \ddots & \\ 1 & 1 & \dots & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_k \end{bmatrix} = \begin{bmatrix} 0 \\ x_1 \\ x_1 + x_2 \\ \vdots \\ \sum_{i=1}^{k-1} x_i \end{bmatrix} = \begin{bmatrix} 1 - \sum_{i=1}^{k} x_i \\ 1 - \sum_{i=2}^{k} x_i \\ 1 - \sum_{i=3}^{k} x_i \\ \vdots \\ 1 - x_k \end{bmatrix}$$

On the other hand, we have also:

$$\mathbf{1} - (I_k + U_k) \cdot x = \mathbf{1} - \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ & 1 & 1 & \dots & 1 \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & 1 \\ 0 & & & & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \vdots \\ x_k \end{bmatrix} = \begin{bmatrix} 1 - \sum_{i=1}^{k} x_i \\ 1 - \sum_{i=2}^{k} x_i \\ 1 - \sum_{i=3}^{k} x_i \\ \vdots \\ 1 - x_k \end{bmatrix}$$

□

Lemma 4.

$$e_1^T \cdot U^{k-t-1} \cdot z = z_{k-t-1} + e_1^T \cdot U^{k-t} \cdot z$$

The proof follows immediately from the fact that $e_1^T \cdot U^{k-x} = (\underbrace{0, \dots, 0}_{k-x}, \underbrace{1, \dots, 1}_{x})$.

As an immediate consequence we obtain the following corollary.

Corollary 1.

$$e_1^T \cdot U^{k-t} \cdot z = z_{k-t} + e_1^T \cdot U^{k-t+1} \cdot z$$

Lemma 5. *For $z = e_i - e_j$, we have that $(L_k + U_k) \cdot z = -z$.*

Proof. If $i = j$, then $z = 0$ and the results is true. Suppose w.l.o.g. that $i > j$ (otherwise we just prove the result for $-z$). Then

$$U_k \cdot (e_i - e_j) = \sum_{s=1}^{i-1} e_s - \sum_{s=1}^{j-1} e_s = \sum_{s=j}^{i-1} e_s$$

Similarly

$$L_k \cdot (e_i - e_j) = \sum_{s=i+1}^{k} e_s - \sum_{s=j+1}^{k} e_s = \sum_{s=j+1}^{i} -e_s$$

Therefore

$$(L_k + U_k) \cdot (e_i - e_j) = \sum_{s=j}^{i-1} e_s - \sum_{s=j+1}^{i} e_s = e_j - e_i = -z$$

□

3.1.2 How to recover the bids when knowing the l_{ij}'s.

As discussed above, we can represent the function f as a matrix multiplication. Let M be the following square matrix of size $nk \times nk$:

$$M = \begin{bmatrix} (U+L) & U & \dots & \dots & U \\ (U+I) & (U+L) & U & \dots & U \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ (U+I) & \dots & (U+I) & (U+L) & U \\ (U+I) & \dots & \dots & (U+I) & (U+L) \end{bmatrix}$$

Then

$$f(b) = M \cdot b$$

The function takes as input a vector composed of n vectors, each of k bits. It returns the nk values l_{ij}, $1 \leq i \leq n$ and $1 \leq j \leq k$. As explained above, the structure of the matrix is defined by the formula that computes γ_{ij}^a, which consists essentially of three factors: Firstly we multiply all α_{ij} which encode bigger bids (represented by the matrix U), then we multiply all α_{ij} which encode smaller bids by the same bidder (represented by adding the matrix L on the diagonal), and finally we multiply by all α_{ij} which encode the same bid by bidders with a smaller index (represented by adding the matrix I on the lower triangle of M). In our encoding there will be a "1" in the vector for each Y in the protocol, hence f will count how many Ys are multiplied when computing γ_{ij}^a. Using this representation we can prove the following theorem.

Theorem 1. *f is injective, i.e. for two different correct bid vectors*

$$u = \begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix} \text{ and } v = \begin{pmatrix} v_1 \\ \vdots \\ v_k \end{pmatrix}$$

with $u \neq v$ we have $M \cdot u \neq M \cdot v$.

Proof. Let u and v be two correct bid vectors such that $u \neq v$. We want to prove that $M \cdot u \neq M \cdot v$. We make a proof by contradiction, hence we assume that $M \cdot u = M \cdot v$ which is equivalent to $M \cdot (u - v) = 0$. Because u and v are two correct bid vectors they only contain elements of the canonical basis $(e_1, \ldots, e_k)$, then

$$u = \begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix} = \begin{pmatrix} e_{i_1} \\ \vdots \\ e_{i_k} \end{pmatrix} \text{ and } v = \begin{pmatrix} v_1 \\ \vdots \\ v_k \end{pmatrix} = \begin{pmatrix} e_{j_1} \\ \vdots \\ e_{j_k} \end{pmatrix},$$

where e_{i_k} and e_{j_k} are elements of the canonical basis. We denote $u - v$ by z, consequently

$$\begin{pmatrix} z_1 \\ \vdots \\ z_k \end{pmatrix} = \begin{pmatrix} e_{i_1} - e_{j_1} \\ \vdots \\ e_{i_k} - e_{j_k} \end{pmatrix}$$

Knowing that $M \cdot z = 0$, we prove by induction on a that for all a the following property $P(a)$ holds:

$$P(a) : \forall l, 0 < l \leq a, U^{k-l} \cdot z = 0$$

This proves in particular that $U^0 \cdot z_l = 0$, i.e. $z = 0$ which contradicts out hypothesis.

- Case $a = 1$: We prove by induction for all $b \geq 1$ that the property $Q(b)$ holds, where:

  $$Q(b) : \forall m, 0 < m \leq b, U^{k-1} \cdot z_m = 0$$

  which gives us that $U^{k-1} \cdot z = 0$.

– Base case $b = 1$: We start by looking at the multiplication of the first row of M with z. We obtain:

$$(L + U) \cdot z_1 + U \cdot (z_2 + \ldots + z_k) = 0$$

We can multiply each side by U^{k-1}, and use Lemma 5 to obtain:

$$U^{k-1} \cdot [-z_1 + U \cdot (z_2 + \ldots + z_k)] = 0$$

Since U is nilpotent, according to Lemma 2 the latter gives $-U^{k-1} \cdot z_1 = 0$. Hence we know $Q(1) : U^{k-1} \cdot z_1 = 0$, i.e. that the last entry of z_1 is 0.

– Inductive step $b+1$: Assume $Q(b)$. Consider now the multiplication of the $(b + 1)$-th row of the matrix M:

$$(U+I) \cdot z_1 + \ldots + (U+I) \cdot z_b + (L+U) \cdot z_{b+1} + U \cdot (z_{b+2} + \ldots + z_k) = 0$$

Then by multiplying by U^{k-1} and using Lemma 5 we obtain

$$U^{k-1} \cdot [(U+I) \cdot z_1 + \ldots + (U+I) \cdot z_b - z_{b+1} + U \cdot (z_{b+2} + \ldots + z_k)] = 0$$

Since U is nilpotent according to Lemma 2 we have

$$U^{k-1} \cdot z_1 + \ldots + U^{k-1} \cdot z_b - U^{k-1} \cdot z_{b+1} = 0$$

Using the fact that for all $m < b$ we have $U^{k-1} \cdot z_m = 0$, the latter gives $-U^{k-1} \cdot z_{b+1} = 0$.

- Inductive step $a + 1$: Assume $P(a)$. By induction on $b \geq 1$ we will show that $Q'(b)$ holds, where

$$Q'(b) : \forall m, 0 < m \leq b, U^{k-(a+1)} \cdot z_m = 0$$

which gives us that $U^{k-(a+1)} \cdot z = 0$, i.e. $P(a + 1)$.

– Base case $b = 1$: Consider the multiplication of the first row with $U^{k-(a+1)}$:

$$U^{k-(a+1)} \cdot [(L + U) \cdot z_1 + U \cdot (z_2 + \ldots + z_k)] = 0$$

which can be rewritten as

$$-U^{k-(a+1)} \cdot z_1 + U^{k-a} \cdot (z_2 + \ldots + z_k)] = 0$$

Using $U^{k-a} \cdot z_l = 0$ for all l, we can conclude that

$$-U^{k-(a+1)} \cdot z_1 = 0$$

i.e. $Q'(1)$ holds.

– Inductive step $b+1$: Assume $Q'(b)$. Consider now the $(b+1)$-th row of the matrix M:

$$(U+I)\cdot z_1+\ldots+(U+I)\cdot z_b+(L+U)\cdot z_{b+1}+U\cdot(z_{b+2}+\ldots+z_k)=0$$

Then by multiplying by $U^{k-(a+1)}$ and using Lemma 5 we obtain

$$U^{k-(a+1)}\cdot[(U+I)\cdot z_1+\ldots+(U+I)\cdot z_b+-z_{b+1}+U\cdot(z_{b+2}+\ldots+z_k)]=0$$

Using $U^{k-a}\cdot z_l=0$ for all l, we can conclude that

$$U^{k-(a+1)}\cdot I\cdot z_1+\ldots+U^{k-(a+1)}\cdot I\cdot z_b-U^{k-(a+1)}\cdot z_{b+1}=0$$

Using the fact that for all $m<b$ we have $U^{k-(a+1)}\cdot z_m=0$, we can conclude that

$$-U^{k-(a+1)}\cdot z_{b+1}=0$$

i.e. $Q'(b+1)$ holds.

□ □

This theorem shows that if there is a constellation of bids that led to certain values l_{ij}, this constellation is unique. Hence we are able to inverse f on valid outputs. We will now show that this can be efficiently done.

3.1.3 An efficient algorithm.

Our aim is solve the following linear system: $M\cdot x=l$. We will use the same steps we used for the proof of injectivity to solve this system efficiently. First note that

$$M\cdot x=l\Rightarrow diag(U^{k-t-1})\cdot M\cdot x=diag(U^{k-t-1})\cdot l$$

where $diag(U^{k-t-1})$ is a $nk\times nk$ block diagonal matrix containing only diagonal blocks of the same matrix U^{k-t-1}. We consider the r-th block of size k of the latter equality. We have $x_r=(x_{r,1},x_{r,2},\ldots,x_{r,k})$. When multiplying by e_1^T we obtain the first line of this block. The r-th block of $M\cdot x$ is

$$\begin{aligned}(U+I)\cdot x_1+\ldots+(U+I)\cdot x_{r-1}+(L+U)\cdot x_r+U\cdot x_{r+1}+\ldots+U\cdot x_k\\ =U\cdot(\sum_{i=1}^{k}x_i)+(\sum_{i=1}^{r-1}x_i)+L\cdot x_r\end{aligned}$$

and the r-th block of l is l_r. Hence

$$e_1^T\cdot\left[U^{k-t}\cdot\left(\sum_{i=1}^{k}x_i\right)+U^{k-t-1}\cdot\left(\sum_{i=1}^{r-1}x_i\right)+U^{k-t-1}\cdot L\cdot x_r\right]=e_1^T\cdot U^{k-t-1}\cdot l_r$$

Using Lemma 3, we have

$$e_1^T \cdot \left[U^{k-t} \cdot \left(\sum_{i=1}^{k} x_i \right) + U^{k-t-1} \cdot \left(\sum_{i=1}^{r-1} x_i \right) + U^{k-t-1} \cdot (\mathbf{1} - (I_n + U_n)\, x_r) \right] = e_1^T \cdot U^{k-t-1} \cdot l_r$$

We remark that $e_1^T \cdot U^{k-t-1} \cdot \mathbf{1} = t + 1$, then we get

$$e_1^T \cdot \left[U^{k-t} \cdot \left(\sum_{i=1, i \neq r}^{k} x_i \right) + U^{k-t-1} \cdot \left(\sum_{i=1}^{r-1} x_i \right) - U^{k-t-1} \cdot x_r \right] = e_1^T \cdot U^{k-t-1} \cdot l_r - (t+1)$$

Using Lemma 4, we have

$$e_1^T \cdot \left[U^{k-t} \cdot \left(\left(\sum_{i=1}^{k} x_i \right) - 2 \cdot x_r \right) + U^{k-t-1} \cdot \left(\sum_{i=1}^{r-1} x_i \right) \right] + (t+1) - e_1^T \cdot U^{k-t-1} \cdot l_r = x_{r,k-t-1} \quad (1)$$

Using several times Corollary 1 we have:

- $e_1^T \cdot U^{k-t} \cdot \left(\left(\sum_{i=1}^{k} x_i \right) - 2 \cdot x_r \right) = e_1^T \cdot U^{k-t+1} \cdot \left(\left(\sum_{i=1}^{k} x_i \right) - 2 \cdot x_r \right) + e_{k-t}^T \cdot \left(\left(\sum_{i=1}^{k} x_i \right) - 2 \cdot x_r \right)$
- $e_1^T \cdot U^{k-t-1} \cdot \left(\sum_{i=1}^{r-1} x_i \right) = e_1^T \cdot U^{k-t} \cdot \left(\sum_{i=1}^{r-1} x_i \right) + e_{k-t-1}^T \cdot \left(\sum_{i=1}^{r-1} x_i \right)$
- $e_1^T \cdot U^{k-t-1} \cdot l_r = e_1^T \cdot U^{k-t} \cdot l_r + l_{r,k-t-1}$

By a changing t to $t-1$ in Equation (1) we get

$$e_1^T \cdot \left[U^{k-t+1} \cdot \left(\left(\sum_{i=1}^{k} x_i \right) - 2 \cdot x_r \right) + U^{k-t} \cdot \left(\sum_{i=1}^{r-1} x_i \right) \right] + t - e_1^T \cdot U^{k-t} \cdot l_r = x_{r,k-t}$$

Then regrouping the applications of Corollary 1 and the latter formula within Equation (1), we obtain:

$$x_{r,k-t} + e_{k-t}^T \cdot \left(\left(\sum_{i=1}^{k} x_i \right) - 2 \cdot x_r \right) + e_{k-t-1} \cdot \left(\sum_{i=1}^{r-1} x_i \right) + 1 + l_{r,k-t-1} = x_{r,k-t-1}$$

This gives us a formula to compute the values of $x_{i,j}$, starting with the last element of the first block $x_{1,k}$. Then we can compute the last elements of all other blocks $x_{2,k}, \ldots, x_{n,k}$, and then the second to last elements $x_{1,k-1}, \ldots, x_{n,k-1}$ and so on.

Complexity Analysis. To obtain all values, we have to apply the above formula for each $t \leq n$ and $r \leq k$, hence we have:

$$\sum_{t=1}^{n}\sum_{r=1}^{k}(k+r) = n \cdot \left(k^2 + \frac{k(k+1)}{2}\right) = \frac{3}{2}nk^2 + \frac{1}{2}nk \in \mathcal{O}\left(nk^2\right)$$

This is efficient enough to be computed on a standard PC for realistic values of n (the number of bidders) and k (the number of possible bids). Those could be less than a hundred bidders with a thousand different prices, thus requiring about the order of only a hundred million arithmetic operations.

3.1.4 How to obtain the l_{ij}'s.

In the previous section we showed that knowing the l_{ij}'s allows us the efficiently break the privacy of all bidders. Here is now how to obtain the l_{ij}'s.

The seller will learn all

$$v_{ij} = \left(Y^{l_{ij}}\right)^{(\sum_{h=1}^{n} m_{ij}^{h})}$$

at the end of the protocol. Since the m_{ij}^h are randomly chosen, this will be a random value if $l_{ij} \neq 0$. However a malicious bidder ("Mallory", of index a) can cancel out the m_{ij}^h as follows: In step 3 of the protocol each bidder will compute his γ_{ij}^a and δ_{ij}^a. Mallory waits until all other bidders have published their values, and then computes his values γ_{ij}^a and δ_{ij}^a as:

$$\gamma_{ij}^a = \left(\left(\prod_{h=1}^{n}\prod_{d=j+1}^{k}\alpha_{hd}\right)\cdot\left(\prod_{d=1}^{j-1}\alpha_{id}\right)\cdot\left(\prod_{h=1}^{i-1}\alpha_{hj}\right)\right)\cdot\left(\prod_{k\neq a}\gamma_{ij}^k\right)^{-1}$$

$$\delta_{ij}^a = \left(\left(\prod_{h=1}^{n}\prod_{d=j+1}^{k}\beta_{hd}\right)\cdot\left(\prod_{d=1}^{j-1}\beta_{id}\right)\cdot\left(\prod_{h=1}^{i-1}\beta_{hj}\right)\right)\cdot\left(\prod_{k\neq a}\delta_{ij}^k\right)^{-1}$$

The first part is a correct encryption of $Y^{l_{ij}}$, with $m_{ij}^a = 1$ for all i and j. The second part is the inverse of the product of all the other bidders γ_{ij}^k and δ_{ij}^k, and thus it will eliminate the random exponents. Hence after decryption the seller obtains $v_{ij} = Y^{l_{ij}}$, where $l_{ij} < n$ for a small n. He can compute l_{ij} by simply (pre-)computing all possible values Y^r and testing for equality. This allows the seller to obtain the necessary values and then to use the resolution algorithm to obtain each bidder's bid. Note that although we changed the intermediate values, the output still gives the correct result (i.e. winning bid). Therefore, the attack might even be unnoticed by the other participants. Note also that choosing a different Y_i per bidder does not prevent the attack, since all the Y_i need to be public in order to prove the correctness of the bid in step 2 of the protocol.

However the protocol requires Mallory to prove that γ_{ij}^a and δ_{ij}^a have the same exponent. This is obviously the case, but Mallory does not know the exact value of

this exponent. Thus it is impossible for him to execute the proposed zero-knowledge-proof-protocol directly. Yet, if interactive proofs are used, he is able to fake this proof as follows.

First, note that we can rewrite γ_{ij}^a and δ_{ij}^a as:

$$v = \gamma_{ij}^a = \underbrace{\left(\left(\prod_{h=1}^{n}\prod_{d=j+1}^{k}\alpha_{hd}\right)\cdot\left(\prod_{d=1}^{j-1}\alpha_{id}\right)\cdot\left(\prod_{h=1}^{i-1}\alpha_{hj}\right)\right)}_{g_1}^{1-\left(\sum_{k\neq a} m_{ij}^k\right)}$$

$$w = \delta_{ij}^a = \underbrace{\left(\left(\prod_{h=1}^{n}\prod_{d=j+1}^{k}\beta_{hd}\right)\cdot\left(\prod_{d=1}^{j-1}\beta_{id}\right)\cdot\left(\prod_{h=1}^{i-1}\beta_{hj}\right)\right)}_{g_2}^{1-\left(\sum_{k\neq a} m_{ij}^k\right)}$$

When Mallory is asked by Victor for a proof of correctness of his values, he starts by asking all other bidders for proofs. Each of them answers with values $\lambda_o = g_1^{z_o}$ and $\mu_o = g_2^{z_o}$. Mallory can then answer Victor with values $\lambda = \prod_o \lambda_o^{-1}$ and $\mu = \prod_o \mu_o^{-1}$. Victor then sends a challenge c, which Mallory simply forwards to the other bidders. They answer with $r_o = z_o + c \cdot m_{ij}^o$, and Mallory sends $r = c - \sum_o r_o$ to Victor, who can check that $g_1^r = \lambda \cdot v^c$ and $g_2^r = \mu \cdot w^c$. If the other bidders did their proofs correctly, then Mallory's proof will appear valid to Victor:

$$\lambda \cdot v^c = \prod_o \lambda_o^{-1} \cdot \left(g_1^{1-\left(\sum_o m_{ij}^o\right)}\right)^c = \prod_o g_1^{-z_o} \cdot g_1^{c-c\left(\sum_o m_{ij}^o\right)} = g_1^{c-\sum_o\left(z_o + c m_{ij}^o\right)}$$

$$\mu \cdot w^c = \prod_o \mu_o^{-1} \cdot \left(g_2^{1-\left(\sum_o m_{ij}^o\right)}\right)^c = \prod_o g_2^{-z_o} \cdot g_2^{c-c\left(\sum_o m_{ij}^o\right)} = g_2^{c-\sum_o\left(z_o + c m_{ij}^o\right)}$$

Hence in the case of interactive zero-knowledge proofs Mallory is able to modify the values γ_{ij}^a and δ_{ij}^a as necessary, and even prove the correctness using the bidders. Hence the modifications may stay undetected, and the seller will be able to break privacy.

Putting everything together, the attack works as follows:

1. The bidders set up the keys as described in the protocol.
2. They encrypt and publish their bids.
3. They compute γ_{ij}^h and δ_{ij}^h and publish them.
4. Mallory, who is a bidder himself, waits until all other bidders have published their values. He then computes his values as defined above, and publishes them.
5. If he is asked for a proof, he can proceed as explained above.
6. The bidders (including Mallory) jointly decrypt the values.
7. The seller obtains all $Y^{l_{ij}}$'s. He can then compute the l_{ij}'s by testing at most n possibilities.

8. Once he has all values, he can invert the function f as explained above.

9. He obtains all bidders bids.

Again, note that for all honest bidders, this execution will look normal, so they might not even notice that an attack took place.

3.2 Attacking the general protocol architecture

The previous attack only works in the case of interactive proofs. If we switch to non-interactive proofs, Mallory cannot ask the other bidders for proofs using a challenge of his choice.

However even with non-interactive zero-knowledge proofs the protocol is still vulnerable to attacks on a targeted bidder's privacy. For example, we can exploit the fact that no message is authenticated. A malicious attacker in control of the network can hence impersonate any bidder of his choice, as well as the seller. In particular, if he wants to know Alice's bid, he can proceed as follows:

1. Mallory impersonates all other bidders. He starts by creating keys on their behalf and publishes the values y_i and the corresponding proofs for all of them.

2. Alice also creates her secret keyshare and publishes y_a together with a proof.

3. Alice and Mallory compute the public key y.

4. Alice encrypts her bid and publishes her α_{aj} and β_{bj} together with the proofs.

5. Mallory publishes $\alpha_{ij} = \alpha_{aj}$ and $\beta_{ij} = \beta_{aj}$ for all other bidders i and also copies Alice's proofs.

6. Alice and Mallory execute the computations described in the protocol and publish γ_{ij}^a and δ_{ij}^a.

7. They compute ϕ_{ij}^a and send it to the seller.

8. The seller publishes the ϕ_{ij}^a and computes the v_{aj}.

Since all submitted bids are equal, the seller (which might also be impersonated by Mallory) will obtain Alice's bid as the winning price, hence it is not private any more. This attack essentially simulates a whole instance of the protocol to make Alice indirectly reveal a bid that was intended for another, probably real auction.

More generally, the lack of authentication allows Mallory to impersonate *all* of the bidders if he can control the network. Then Mallory can choose the bids of all the bidders, and therefore select the winner at a chosen price, breaking fairness.

4 Conclusion and Future Work

In this paper we analyzed the protocol by Brandt [1] and showed that the underlying computations have a weakness. This weakness can be exploited by malicious bidders if interactive zero-knowledge proofs are used. When non-interactive zero-knowledge proofs are used, we also described a different attack which exploits the lack of authentication in the protocol.

This shows that properties such as authentication are necessary to achieve other properties that might appear to be unrelated at first sight (like for instance privacy). It also makes it clear that there is a difference between computing the winner in a fully private way, and ensuring privacy for the bidders: if we use modified inputs, we can break privacy even if the computations themselves are secure. Additionally our analysis highlights that interactive and non-interactive proofs have different properties, and that this is an important choice in protocol design. All in all, the results underline that designing protocols is a complex task where the devil hides in the details.

References

[1] Felix Brandt. How to obtain full privacy in auctions. *International Journal of Information Security*, 5:201–216, 2006.

[2] David Chaum and Torben P. Pedersen. Wallet databases with observers. In *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '92, pages 89–105, London, UK, UK, 1993. Springer-Verlag.

[3] Brian Curtis, Josef Pieprzyk, and Jan Seruga. An efficient eAuction protocol. In *ARES*, pages 417–421. IEEE Computer Society, 2007.

[4] George Danezis. An anonymous auction protocol using "money escrow" (transcript of discussion). In Bruce Christianson, Bruno Crispo, and Michael Roe, editors, *Security Protocols Workshop*, volume 2133 of *Lecture Notes in Computer Science*, pages 223–233. Springer, 2000.

[5] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 10–18, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[6] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

[7] Ari Juels and Michael Szydlo. A two-server, sealed-bid auction protocol. In Matt Blaze, editor, *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 72–86. Springer, 2002.

[8] Vijay Krishna. *Auction Theory*. Academic Press, 2002.

[9] Toru Nakanishi, Daisuke Yamamoto, and Yuji Sugiyama. Sealed-bid auctions with efficient bids. In Jong In Lim and Dong Hoon Lee, editors, *ICISC*, volume 2971 of *Lecture Notes in Computer Science*, pages 230–244. Springer, 2003.

[10] Moni Naor, Benny Pinkas, and Reuban Sumner. Privacy preserving auctions and mechanism design. In *ACM Conference on Electronic Commerce*, pages 129–139, 1999.

[11] Kazumasa Omote and Atsuko Miyaji. A practical English auction with one-time registration. In Vijay Varadharajan and Yi Mu, editors, *ACISP*, volume 2119 of *Lecture Notes in Computer Science*, pages 221–234. Springer, 2001.

[12] Kazumasa Omote and Atsuko Miyaji. A second-price sealed-bid auction with the discriminant of the p_0-th root. In Matt Blaze, editor, *Financial Cryptography*, volume 2357 of *Lecture Notes in Computer Science*, pages 57–71. Springer, 2002.

[13] Kun Peng, Colin Boyd, and Ed Dawson. A multiplicative homomorphic sealed-bid auction based on Goldwasser-Micali encryption. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *ISC*, volume 3650 of *Lecture Notes in Computer Science*, pages 374–388. Springer, 2005.

[14] Kun Peng, Colin Boyd, and Ed Dawson. Sealed-bid micro auctions. In Simone Fischer-Hübner, Kai Rannenberg, Louise Yngström, and Stefan Lindskog, editors, *SEC*, volume 201 of *IFIP*, pages 246–257. Springer, 2006.

[15] Kun Peng, Colin Boyd, Ed Dawson, and Kapali Viswanathan. Non-interactive auction scheme with strong privacy. In Pil Joong Lee and Chae Hoon Lim, editors, *ICISC*, volume 2587 of *Lecture Notes in Computer Science*, pages 407–420. Springer, 2002.

[16] Kun Peng, Colin Boyd, Ed Dawson, and Kapali Viswanathan. Robust, privacy protecting and publicly verifiable sealed-bid auction. In Robert H. Deng, Sihan Qing, Feng Bao, and Jianying Zhou, editors, *ICICS*, volume 2513 of *Lecture Notes in Computer Science*, pages 147–159. Springer, 2002.

[17] Ahmad-Reza Sadeghi, Matthias Schunter, and Sandra Steinbrecher. Private auctions with multiple rounds and multiple items. In *DEXA Workshops*, pages 423–427. IEEE Computer Society, 2002.

[18] Kazue Sako. An auction protocol which hides bids of losers. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 422–432. Springer, 2000.

[19] Frank Stajano and Ross J. Anderson. The cocaine auction protocol: On the power of anonymous broadcast. In Andreas Pfitzmann, editor, *Information Hiding*, volume 1768 of *Lecture Notes in Computer Science*, pages 434–447. Springer, 1999.

[20] Kapali Viswanathan, Colin Boyd, and Ed Dawson. A three phased schema for sealed bid auction system design. In Ed Dawson, Andrew Clark, and Colin Boyd, editors, *ACISP*, volume 1841 of *Lecture Notes in Computer Science*, pages 412–426. Springer, 2000.