t-Private and t-Secure Auctions

Markus Hinkelmann, Andreas Jakoby, and Peer Stechert

[1]Institut für Theoretische Informatik,
Universität zu Lübeck, Germany
[2]Fachgruppe Didaktik der Informatik und E-Learning,
Universität Siegen, Germany

{hinkelma, jakoby}@tcs.uni-luebeck.de
stechert@die.informatik.uni-siegen.de

SIIM-TR-A-08-01
Report Series of the Institutes for Computer Science and Mathematics,
Universität zu Lübeck

May 14, 2008

Abstract

In most of the used auction systems the values of bids are known to the auctioneer. This allows him to manipulate the outcome of the auction. Hence, one is interested in hiding these values. Some cryptographically secure protocols for electronic auctions have been presented in the last decade. Our work extends these protocols in several ways. Based on garbled circuits, i.e., encrypted circuits, we present protocols for sealed-bid auctions that fulfill the following requirements:

1. Protocols are information-theoretically t-private for *honest but curious parties*.
2. The number of bits that can be learned by *malicious adversaries* is bounded by the output length of the auction.
3. The computational requirements for participating parties are very low: only random bit choices and bitwise computation of the `XOR`-function are necessary.

Note that one can distinguish between the protocol that generates a garbled circuit for an auction and the protocol to evaluate the auction. In this paper we address both problems. We will present a t-private protocol for the construction of a garbled circuit that reaches the lower bound of $2t + 1$ parties, and a more randomness efficient protocol for $(t + 1)^2$ parties. Finally, we address the problem of bid changes in an auction.

Keywords Multi-party private and secure computation, garbled circuits, private auctions

1 Introduction

Traditional auctions involve an auctioneer and numerous bidders who want to sell or buy an item, respectively. In a secure electronic auction system a bid has to remain hidden for all other bidders and for the auctioneer if it is not a part of the output. In particular, the auctioneer has to be prevented from learning the bids.
There exist numerous different auction types and models: The *English* or ascending auction is the most common auction type. Here, in each round a bidder can increase his previous bid. At the end, the highest bidder takes the item and pays the price he has bid. Hence, the auctioneer learns the bids of all bidders. In a *Dutch* or descending auction, the potential price decreases over time. The first person who places a bid takes the item and has to pay the current price. Note that privacy of other bidders' bids remains preserved. In a *sealed-bid* auction, every player turns in a secret bid. The auctioneer opens all bids and computes the winner. Again, the auctioneer learns the bids of all bidders. Dealing with sealed-bids we distinguish between *first-price* and *second-price* auctions. In a first-price auction, the highest bidder takes the item and pays the price of his bid, i.e., the highest bid. In a second-price auction also called *Vickrey* auction the highest bidder takes the item and pays the second highest bid (see e.g., [10]).
In this paper, we present protocols for sealed-bid auctions, which keep the privacy of the bidders' bids, even if a collusion of parties tries to attack the protocol. Our work relies on two concepts. First, it relies on the cryptographically privacy preserving auction scheme of Naor et al. [24], using the garbled circuit (GC) construction, i.e., encrypted Boolean circuits. Second, it relies on the concept of perfect privacy, i.e., on the concept of privacy in the information-theoretic sense [2, 7]. Ishai and Kushilevitz showed how to use the model of garbled circuits to compute a function in a perfect private way [17]. We will generalize both concepts:

1. We present *protocols to construct garbled circuits t-privately*: The construction does not leak more information to any collusion of up to t parties, than the garbled circuit itself.

2. We generalize the concept of garbled circuits such that the *garbled circuits will be t-private* according to the inputs of the circuit. This implies that any collusion of up to t parties does not get more information about the inputs of the circuit than the collusion can deduce from result of the circuit.

1.1 Related work concerning Electronic Auction Systems

In 1996, Franklin and Reiter [12] proposed the first auction scheme that has a privacy contribution with regard to the bidders, whereas the auctioneer is allowed to learn all bids. They also demand for a deposit of digital cash for each player to ensure non-repudiation in their cryptographic auction scheme.
Sadeghi, Schunter, and Steinbrecher [28] presented a combinatorial auction, that sells multiple interdependent items by proceeding multiple rounds. They consider the German UMTS auction as an example, where every winner had to win at least

two licenses. In this model, public key cryptography is used to obtain verifiable and private bidding. The possibility to repudiate a bid is mentioned, whereby the involved bidder has to pay a fine.
Kikuchi, Harkavy, and Tygar [20] introduced a model based on Shamir's secret sharing scheme [26]. In each round, a bid-vector of k prices is distributed to the players who can mark all prices they want to bid. Kikuchi et al. [15] also described sealed-bid auctions via a verifiable secret sharing scheme [8, 2]. Their approach deals with n bidders and m auctioneers and uses an error correcting code to share the secrets. Accountability is reached with public key encryption.
Kurosawa and Ogata [22] dealt with a bit-slice auction circuit. Traditional techniques to evaluate an auction compare the incoming bids one-by-one, but their approach alternatively compares the bids bit-by-bit, beginning with the most significant one. Additionally, they combine this approach with the mix and match method of Jakobsson and Juels [18] that uses a homomorphic encryption scheme.
Omote and Miyaji [25] proposed a second-price sealed-bid auction that satisfies public verifiability of the auction without revealing the highest bid. To achieve this, they use a cryptographic primitive and two auction managers.
Brandt [4, 5] introduced a cryptographic auction protocol without a special trusted party. Full privacy is achieved by distributing shares of every bid via Shamir's secret sharing scheme.
Cachin [6] presented a scheme based on homomorphic encryption. It involves an auctioneer who wants to receive at least a certain price A for his item, and a bidder who wants to pay at most B. To answer the question, whether the deal takes place, an oblivious third party is introduced, that neither learns anything about both values nor about the result, i.e., whether $A > B$.
Naor et al. [24] presented an electronic auction system based on garbled circuits. Using pseudo-random generators these circuits allow to define a cryptographically secure and verifiable electronic auction system. To construct the garbled circuits the authors introduce a new party called auction issuer. They assume that the auction issuer does not collude with the auctioneer or a bidder. Juels and Szydlo [19] relied directly on the garbled circuit construction of Naor et al. [24]. The proxy oblivious transfer of Naor et al. has the disadvantage that the correct input of a bidder is not verifiable by other parties. Juels and Szydlo introduce a verifiable proxy oblivious transfer resulting in less computation for the bidders.

1.2 Related work concerning Private Computation

Private computation was introduced by Yao [30]. He considered the problem under cryptographic assumptions. Private Computation with unconditional, i.e., information-theoretically, security has been introduced by Ben-Or et al. [2] and Chaum et al. [7]. Kushilevitz et al. [23] proved that the class of Boolean functions, that have linear size circuit, is exactly the class of functions, that can privately be computed using a constant number of random bits.
Franklin and Yung [13] investigated the role of the connectivity of the underlying network in private computations. Bläser et al. [3] completely characterized the class

of privately computable Boolean functions, if the underlying network is connected but not 2-connected. In particular, no non-degenerate function can be computed privately if the network consists of three or more blocks. On networks with two blocks only a small class of functions can be computed privately. This result has been generalized for non-Boolean functions by Beimel [1]. He has shown that only functions with a restricted type of communication matrix can be computed on a non-2-connected network. One can easily verify that most types of auctions do not fulfill these restrictions of the function. Hence, the underlying network has to be 2-connected.
Chaum et al. [7] proved that any Boolean function can be computed privately, if at most one third of the participating parties are dishonest. For this model, Ben-Or et al. [2] proved that any n-ary Boolean function can be computed $\lfloor \frac{n-1}{2} \rfloor$-private, i.e., at most $\lfloor \frac{n-1}{2} \rfloor$ players collude. Chor and Kushilevitz [9] showed that if a function can be computed at least $\frac{n}{2}$-privately, then it can be computed n-privately as well.
Randomizing polynomials are introduced by Ishai and Kushilevitz [16] as a new algorithmic approach for round-efficient private computations with low error probability. In [17] they have shown that all functions can privately be computed in a constant number of rounds and 0 error probability. Damgård and Ishai presented a constant-round protocol for general multiparty computation. It uses randomized polynomials, black-box pseudo-random generators, and Shamir shares [11]. The protocol uses multiplications and additions over a finite field $F = \mathrm{GF}(2^k)$ with security parameter k for construction and evaluation of the polynomials. Damgård and Ishai distinguish between input players, output players, and servers, i.e., parties which construct the randomized polynomials. Hence, this protocol can be used to implement auctions. It is cryptographically secure against an malicious adversary corrupting $t < n/3$ servers.

1.3 Our Contributions

For private electronic auction systems one has to find a concept and protocols for evaluating auctions preserving the privacy of the bidders and the bids.
Our Concept: We generalize the concept of garbled circuits in order to evaluate a circuit t-privately, i.e., no collusion of up to t parties can deduce any knowledge about the inputs of the circuit, that cannot be deduced from the result of the circuit and the input of the colluding parties. Focusing on electronic auction systems fulfills the following requirements:

- In contrast to previous electronic auction systems our system is information-theoretically t-private.
- Some protocols presented in the literature may fail with small probability. Garbled circuits allow to evaluate an auction in a perfectly correct way.

In most papers cited above verification is only investigated regarding the bidders and their secret inputs. In our electronic auction system it is possible that all parties t-privately verify the correctness of the auction evaluation. Furthermore, our auction scheme guarantees that the total number of bits of information revealed by

a collusion of up to t malicious parties is bounded by the number of output-bits of the auction, even if the malicious parties try to stay undetected .

The Protocols:

1. We present information-theoretically t-private protocols to construct a garbled circuit $\Gamma(C)$ for a given circuit C. The first protocol requires $(t+1)^2$ parties and uses a number of random bits that is polynomial in $|\Gamma(C)|$ and t. The second protocol is based on the first protocol and reaches the lower bound of $2t+1$ parties. In return the second protocol uses an exponential number of additional random bits in t.

2. We present two information-theoretically t-private protocols for the bidding process.

In our protocols there are four kinds of parties: (i) auctioneer who evaluates the auction, (ii) bidders, (iii) auction issuers who determine the encryption of the auction, and (iv) slaves who perform the encryption. If the parties have access to a source of random bits, the computation performed by auction issuers, bidders and slaves can be implemented by circuits of depth $\mathcal{O}(\log t)$, that consists of binary `XOR`-gates and one level of gates to multiplex one bit out of two. Thus, the computational requirements are very low.
Introducing some minor changes to these protocols, one can use them to translate intermediate data of an evaluation process of a garbled circuit into intermediate data of the evaluation process of another garbled circuit of the same circuit. Based on this observation our protocols allow a dynamic sealed-bid auction, i.e., bidders can change their bids at will or they can enter an auction that has already started without further bidders involved in this procedure.
All protocols presented in this paper are information-theoretically t-private. In return the garbled circuit requires a large number of random bits in the encryption of the underlying circuit. But, the protocols for generating the garbled circuits might be useful for generating garbled circuits that are cryptographically t-private. As one can see in [24, 11] these encryptions are often more efficient than information-theoretically private encryptions. Hence, this paper will be a first step towards an efficient implementation of t-private electronic auction systems with very low requirements for hardware. Our approach is based on [29].
This work is organized as follows: In Section 2 we present some basic definitions and notations of private computation. We will redraw the privacy preserving cryptographic auction of Naor et al. Furthermore, we discuss randomizing polynomials that can be used as an evaluation technique for garbled circuits. Section 3 presents information-theoretically t-private, static auction protocols. We dedicate Section 4 to the proofs of t-privacy. In Section 5 we analyze the security of our protocol against active attacks. Section 6 deals with dynamic aspects of the electronic auction system. The concluding Section 7 summarizes the results.

2 Notation and Preliminaries

In this section we refer to the literature and define our notations. In section 2.3, garbled circuits are presented using a compact definition. Throughout this work, we will use $\oplus$ to denote the (bit-wise) `XOR`-operation on Boolean values and binary strings, as well as on vectors and matrices of binary strings.
Let $x = x[1]x[2]\dots x[n] \in \{0,1\}^n$ be a binary string of length n. For a set $J = \{j_1, \dots, j_k\}$ where $1 \leq j_1 < j_2 < \dots < j_k \leq n$ let $x[J] = x[j_1]\dots x[j_k]$. We extend this notation to arbitrary sequences. Let $C = (c_1, \dots, c_n)$. Then $C[J]$ denotes the subsequence $(c_{j_1}, \dots, c_{j_k})$. The operator $\circ$ denotes the concatenation of strings. For $i, j \in \mathbb{N}$, let $[i, j] := \{i, \dots, j\}$. All logarithms in this paper are to the base 2.
An n-input l-output circuit $\mathcal{C}_n^l$ is described by a DAG $G = (V, E)$. V contains $2n$ nodes of in-degree 0, the inputs $x_1, \bar{x}_1, \dots, x_n, \bar{x}_n$. The remaining nodes have in-degree 2 and are called gates. The gates $g_1, \dots, g_m$ are labeled by either `AND` or `OR`. G has l nodes with out-degree 0 called output-gates. For convenience we write $\mathcal{C}$ instead of $\mathcal{C}_n^1$ and we use the variables x_i synonymously to the corresponding input values. If the input of a circuit is fixed by the context, we denote the value of a gate g by b_g. The value b_e of a wire e is determined by the value of the gate g the wire comes from resp., by the input x_i if e is an input wire.

2.1 Privacy and Security

In this work we consider both, the model of honest but curious parties, and the model of malicious parties. We do not want any party to learn more about the inputs of other parties, than it can deduce from the result of the function and its own input. We assume that every party has access to a random tape. Parties can send messages to other parties via point-to-point communication using secure channels. Computing a function f on the input $x = (x_1, \dots, x_n)$ we assume that x_i is the private input of party P_i.

Definition 1 (t-Privacy) *For a subset of parties $U \subseteq V$ let C_U resp., R_U denote the random variables of the communication string resp., of the random string seen by parties of U. Let c_U be a particular communication string. An n-party protocol $\mathcal{P}$ for computing a function $f : \Sigma^n \to \Sigma$ is private with respect to the parties of U, if for every pair of input vectors x, y with $f(x) = f(y)$ and $x[U] = y[U]$, for every sequence of random strings r_U provided to the parties of U,*

$$Pr[C_U = c_U | R_U = r_U, x] \;=\; Pr[C_U = c_U | R_U = r_U, y]$$

where all parties follow protocol $\mathcal{P}$. A protocol is t-private if it is private with respect to any subset U of at most t parties. A protocol is called private if it is 1-private.

In other words, a protocol is called t-private, $1 \leq t \leq n-1$, if no collusion of t parties learns anything about the distributed secret that cannot be deduced from the result and the input of the members of the collusion, while honestly participating in the protocol. From [3, 1] we can conclude that most auction protocols cannot be run

privately on non-2-connected networks. Therefore, we demand a communication network to be at least 2-connected.
Alternatively, one can define t-privacy by using conditional entropy. Shannon [27] introduced the entropy function

$$\mathrm{H}(X) = -\sum_x \Pr[X = x] \cdot \log(\Pr[X = x]).$$

He proved that the functions of the class $K \cdot H(X)$ with $K > 0$ are the only functions that satisfy his requirements for a measure of information generated by a dicrete information source. The entropy can also be interpreted as a measure of uncertainty about X. If $H(X) = 0$ for a discrete random variable X, then we have full information about the outcome of the random experiment related to X. The conditional mutual information

$$\mathrm{H}(X \,|\, Y) = -\sum_y \sum_x \Pr[Y = y] \Pr[X = x \,|\, Y = y] \log(\Pr[X = x \,|\, Y = y])$$

describes the uncertainty about X if we have knowledge about Y. The mutual information

$$\mathrm{I}(X \,;\, Y \,|\, Z) = \mathrm{H}(X \,|\, Z) - \mathrm{H}(X \,|\, Y, Z)$$

measures the information gain about X knowing Z if we additionaly have knowledge about Y. We cannot deduce information about X from Y if and only if $\mathrm{I}(X \,;\, Y) = 0$. We define the *leakage* of information of a protocol with respect to U as the minimum value τ such that it holds that

$$\mathrm{I}(X \,; C_U \,| f(X), X[U], R_U) \;\leq\; \tau \;.$$

A protocol is t-private if its leakage is 0 with respect to every subset U of at most t parties. We call a protocol $\mathcal{P}$ (t, τ)-*secure*, if for every subset U of at most t parties and for every protocol $\mathcal{P}'$, where the parties of U follow $\mathcal{P}'$ and the honest parties in $V \setminus U$ follow the protocol $\mathcal{P}$, the leakage is bounded by τ. Note that malicious parties may run an arbitrary attack.

2.2 The NPS Protocol

In this section, we discuss the cryptographic electronic auction system of Naor et al. [24], that is based on Yao's garbled circuits and pseudo-random generators [31, 14]. Garbled circuits are encrypted versions of standard Boolean circuits where the Boolean values are replaced by encrypted values. Knowing the encrypted input, one can evaluate the circuit gate-by-gate. Then one can produce the garbled output without learning anything about the original input and intermediary results. To achieve privacy of an auction Naor et al. [24] introduce a third party: the *auction issuer AI*. This party generates the garbled circuit and they assume that the auction issuer does not conspire with any other party. To send the encrypted bids to the auctioneer the bidders use a protocol realizing the *proxy oblivious transfer*. For one bit bids the proxy oblivious transfer is defined as follows:

Each bidder has his private value $x_i \in \{0,1\}$ and the auction issuer provides two encryptions W^0, W^1 for the bit. The goal is to send W^{x_i} to the auctioneer such that neither the bidder learns W^0 or W^1 nor the auction issuer learns x_i. Furthermore, the auctioneer does not learn anything about x_i or W^{1-x_i}. Using the garbled circuit, the auctioneer can evaluate the auction. We will refer to this protocol as the `NPS`-protocol.
The `NPS`-protocol provides 1-privacy, only. That means, a party cannot break the secrets of the other parties. But if the auction issuer and the auctioneer collude, both will gain knowledge about all bids.

2.3 Garbled Circuits

In this section, we shortly discuss garbled circuits. A garbled circuit consists of a circuit of encrypted gates. To encrypt a gate g we `XOR` the truth table of g by a randomly generated table W_g and permute their entries. To compute the result of the encrypted gate we add to the result of predecessor gates the corresponding rows resp., columns of some random tables such that `XOR`-ing these random values allows us to compute exactly one entry of W_g. To keep track of the permutation of the truth table of g, we `XOR` the results of the predecessor gates by a random bit. Note that this allows us to encrypt the result of the predecessor gates, and simultaneously to permute the truth table of g.
Let us now continue with a more formal description of garbled circuits. We use random values of two types: For every wire e of a circuit C there exists a vector of random strings $W_e = (W_e^0, W_e^1)$. For every gate g and every input bit x_i there exists a random bit r_g resp., r_i. The length of the strings W_e^b is defined recursively from top to bottom: For every $b \in \{0,1\}$, every wire e and every gate g let $|W_e^b| = 0$ if e is an output wire, and $|W_e^b| = 2(1 + \sum_{\text{output wire } e' \text{ of } g} |W_{e'}^0|)$ if e is an input wire of gate g.
Each string $W_e^b = W_e^{b,0} \circ W_e^{b,1}$ consists of two equally sized halves. In order to encrypt a Boolean value $x_i \in \{0,1\}$ on an input wire e, we use a random bit r_i and compute the polynomial $W_e^{x_i} \circ (x_i \oplus r_i)$. Note that $W_e^{x_i}$ can be written as $(x_i \cdot W_e^1) \oplus ((1 - x_i) \cdot W_e^0)$.
Let g be a gate with input wires e_1, e_2, output wires $o_1, \ldots, o_k$ and gate function $g(b_1, b_2)$ where $b_1, b_2 \in \{0,1\}$. We encrypt the values b_{g_i} by $c_{g_i} = b_{g_i} \oplus r_{g_i}$ where g_i denotes the predecessor of g with output wire e_i. Hence, we use one additional random bit r_g for every gate g. Each random bit r_{g_i} induces a permutation of the rows (resp., columns) of the garbled table in the succeeding gates. For each of the four possible values $c_{g_1}, c_{g_2} \in \{0,1\}$ of the preceding gates of g, we define a corresponding polynomial on strings of length $1 + \sum_{1 \le l \le k} |W_{o_l}^0|$. This polynomial defines the entry of the garbled table indexed by (c_{g_1}, c_{g_2}). Informally, these expressions encrypt the values of the (permuted) truth table of g. The garbled table is given by

$$Q_g = \begin{pmatrix} Q_g^{0,0} & Q_g^{0,1} \\ Q_g^{1,0} & Q_g^{1,1} \end{pmatrix}$$

with

$$Q_g^{a,b} \;=\; W_{e_1}^{a\oplus r_{g_1},b} \oplus W_{e_2}^{b\oplus r_{g_2},a} \oplus (W_g^{g(a\oplus r_{g_1},b\oplus r_{g_2})} \circ (r_g \oplus g(a\oplus r_{g_1}, b\oplus r_{g_2})))$$

and

$$W_g^b = W_{o_1}^b \circ \ldots \circ W_{o_k}^b$$

where g_1 denotes the left and g_2 the right predecessor of g.
In every entry of the table, we store a string $W_g^{g(a\oplus r_{g_1},b\oplus r_{g_2})}$ that is used as a key in the table for the succeeding gate. To decrypt a table we use only one half of W_{g_1}, resp., W_{g_2}. Thus, the length of these strings grows exponentially in the depth of the corresponding circuit. But if the circuit has logarithmic depth the total length of the strings in the garbled circuits is only polynomial in these size of the circuit.
Knowing the *encryption* $W_{e_1}^{b_{g_1}}, W_{e_2}^{b_{g_2}}$ and c_{g_1}, c_{g_2} of the input of an gate g, one can `XOR` each of the suitable halves of $W_{e_1}^{b_{g_1}}, W_{e_2}^{b_{g_2}}$ with the garbled table entry $Q_g^{c_{g_1},c_{g_2}}$ and gets the encryption of b_g and b_{o_i} for all output wires o_i of g without learning anything about the hidden Boolean value. That is,

$$\begin{aligned} &Q_g^{c_{g_1},c_{g_2}} \oplus W_{e_1}^{b_{g_1},c_{g_2}} \oplus W_{e_2}^{b_{g_2},c_{g_1}} \\ &\quad= \; W_g^{g(c_{g_1}\oplus r_{g_1},c_{g_2}\oplus r_{g_2})} \circ (g(c_{g_1}\oplus r_{g_1}, c_{g_2}\oplus r_{g_2}) \oplus r_g). \end{aligned} \tag{1}$$

Each expression $W_{o_i}^h$ for a Boolean variable $h \in \{0,1\}$ can be represented by $(h \cdot W_{o_i}^1) \oplus ((1-h)\cdot W_{o_i}^0)$. The output gates g of the circuit have the values $r_g \oplus b_g$.
This construction is closely related to the construction of Ishai and Kushilevitz [17]. They presented a construction of garbled circuits based on Boolean formulas. Our construction above is an extension of this approach to Boolean circuits. The proof of correctness and privacy follows analogously to the case of Boolean formulas.
We will now present an alternative way to generate these garbled tables. This strategy will be used in our protocol to generate garbled tables t-privately. For $r, r_1, r_2 \in \{0,1\}$ and binary strings $W_{e_1}^0, W_{e_1}^1, W_{e_2}^0, W_{e_2}^1, W_e^0, W_e^1, A^{0,0}, A^{0,1}, A^{1,0}, A^{1,1}$ of equal length we define

$$(W_{e_1}^0, W_{e_1}^1) \otimes (W_{e_2}^0, W_{e_2}^1) \;=\; \begin{pmatrix} W_{e_1}^{0,0} \oplus W_{e_2}^{0,0} & W_{e_1}^{0,1} \oplus W_{e_2}^{1,0} \\ W_{e_1}^{1,0} \oplus W_{e_2}^{0,1} & W_{e_1}^{1,1} \oplus W_{e_2}^{1,1} \end{pmatrix}$$

$$\Pi_r(W_e^0, W_e^1) \;=\; \begin{cases} (W_e^0, W_e^1) & \text{for } r = 0 \\ (W_e^1, W_e^0) & \text{for } r = 1 \end{cases}$$

$$\Pi_{r_1,r_2} \begin{pmatrix} A^{0,0} & A^{0,1} \\ A^{1,0} & A^{1,1} \end{pmatrix} \;=\; \begin{pmatrix} A^{r_1,r_2} & A^{r_1,\bar{r}_2} \\ A^{\bar{r}_1,r_2} & A^{\bar{r}_1,\bar{r}_2} \end{pmatrix} .$$

For every gate g and every wire e let $\widehat{W}_e = (W_e^0, W_e^1)$ and

$$\widehat{Q}_g \;=\; \begin{pmatrix} \widehat{Q}_g^{0,0} & \widehat{Q}_g^{0,1} \\ \widehat{Q}_g^{1,0} & \widehat{Q}_g^{1,1} \end{pmatrix} \;\text{ with }\; \widehat{Q}_g^{a,b} \;=\; W_g^{g(a,b)} \circ (r_g \oplus g(a,b)) \; .$$

Then, we have

$$Q_g \;=\; (\Pi_{r_1}(\widehat{W}_{e_1}) \otimes \Pi_{r_2}(\widehat{W}_{e_2})) \oplus \Pi_{r_1,r_2}(\widehat{Q}_g) \; .$$

For a given circuit $\mathcal{C}$ a garbled circuit is denoted by $\Gamma_{\mathcal{C}}$. If $\mathcal{C}$ is known from the context we will omit the index.

3 t-Private Garbled Circuit Construction

In this section we will present two protocols for generating a garbled auction circuit Γ t-privately. We say that a protocol *generates a garbled circuit t-privately* if it generates a garbled circuit Γ, and while constructing Γ no collusion U of up to t parties can deduce any information about the chosen random values of Γ that cannot be deduced from Γ directly.
Basically our protocol works as follows: For each gate of a circuit we independently generate $t+1$ collections of the random bits used in the construction of the corresponding garbled table — each random collection is generated by one separate auction issuer. Instead of computing the garbled auction circuit by the auction issuers we proceed as follows: we introduce a field of $(t+1)^2$ slaves that is divided into $t+1$ columns, where the i-th column is used to permute all truth tables according to the random permutation chosen by the i-th auction issuer. The truth tables and the corresponding W-strings of each auction issuer and of each gate move through all columns of the field. At the end the tables of all auction issuers are permuted by the same permutation. `XOR`-ing all corresponding tables gives us the desired t-private garbled circuit.
Our protocols are designed such that every party can simultaneously take part in the construction of the garbled circuit, bidding and evaluation. Hence, every party can simultaneously be the auctioneer, one auction issuer, one slave, as well as an arbitrary number of bidders, without gaining additional knowledge about the bids of the other bidders. Hence, if it is not necessary to distinguish between the different entities we call them just parties.
In the following, we use the term *a party P generates a ℓ-share $w_1, \ldots, w_\ell$ of w* to denote that party P generates ℓ random strings $w_1, \ldots, w_\ell$ such that $w = w_1 \oplus \ldots \oplus w_\ell$.
Since most proofs of our theorems and lemmata are quite long and technical, we will give the proofs in a separate section. In the following we will present our main protocols.

3.1 Constructing Garbled Circuits

In the first strategy we will use a set of $t+1$ auction issuers AI_k with $k \in [1, t+1]$ and a $(t+1) \times (t+1)$-array of $(t+1)^2$ sub-workers $S_{i,j}$ with $i, j \in [1, t+1]$ called slaves. Every auction issuer generates an independent garbled circuit Γ_k for a globally given auction circuit C. The slaves are used to combine these garbled circuits. More precisely:

- AI_k generates two random strings $W^0_{k,e}, W^1_{k,e}$ for every wire e,
- a random bit $r_{k,g}$ for every gate g, and a random bit $r_{k,i}$ for every input variable x_i. For easier notion, we associate to every wire e the random bit $r_{k,g}$ of the gate g resp., $r_{k,i}$ of the input variable x_i where the wire is coming from. Let $r_{k,e}$ denote this random bit.

After generating the random values, all AI_k put these random values together into the *ungarbled* tables $\widehat{Q}_{k,g}$ of every gate g and the ungarbled vectors $\widehat{W}_{k,e}$ of every wire e of C:

$$\widehat{W}_{k,e} \;=\; (W^0_{k,e}, W^1_{k,e}) \qquad \text{and} \qquad \widehat{Q}_{k,g} \;=\; \begin{pmatrix} \widehat{Q}^{0,0}_{k,g} & \widehat{Q}^{0,1}_{k,g} \\ \widehat{Q}^{1,0}_{k,g} & \widehat{Q}^{1,1}_{k,g} \end{pmatrix} .$$

For technical reasons the auction issuers AI_k choose the table entries as follows

$$\widehat{Q}^{a,b}_{k,g} \;=\; \begin{cases} W^{g(a,b)}_{1,g} \circ (r_{1,g} \oplus g(a,b)) & \text{for } k = 1 \\ W^{g(a,b)}_{k,g} \circ r_{k,g} & \text{for } k > 1 \end{cases}$$

where

$$W^b_{k,g} \;=\; W^b_{k,o_1} \circ \ldots \circ W^b_{k,o_m}$$

where $o_1, \ldots, o_m$ denote the output wires of gate g.
Our goal is to construct Γ and thus all tables

$$Q_g \;=\; \left(\Pi_{r_{e_1}}(\oplus_{k=1}^{t+1}\widehat{W}_{k,e_1}) \;\otimes\; \Pi_{r_{e_2}}(\oplus_{k=1}^{t+1}\widehat{W}_{k,e_2})\right) \;\oplus\; \Pi_{r_{e_1},r_{e_2}}(\oplus_{k=1}^{t+1}\widehat{Q}_{k,g})$$

t-privately where $r_{e_1} = \oplus_{k=1}^{t+1} r_{k,e_1}$ and $r_{e_2} = \oplus_{k=1}^{t+1} r_{k,e_2}$.
Before we present a protocol for constructing Γ we will show that Γ does not allow to reveal information about the private choices of an action issuer.
Let R_i be the random string of AI_i used in Γ, i.e., R_i consists of all $\widehat{W}_{i,e}$ and $r_{i,e}$. Let $R_{\neg i} = \langle R_j | i \neq j \rangle$ be the collection of random strings of all AI_j different from AI_i. Finally, let $|\Gamma|$ denote the length of a binary description of Γ.

Lemma 2 *For every shape Γ of the garbled circuit, every content $R_{\neg i}$ of the random values chosen by $AI_j \neq AI_i$, and every content $r_{i,e}$ of the permutation bits chosen by AI_i there exist $\mu = 2^{|\Gamma|}$ different values of the content of R_i such that $R_i \cup R_{\neg i}$ defines the same shape Γ of the garbled circuit.*

Proof: Since all values $r_{i,e}$ are fixed, we have to count the different values of the random strings $W^0_{i,e}, W^1_{i,e}$ for all wires e that defines the same shape Γ of the garbled circuit.
For each output gate o the strings $W^0_{i,o}, W^1_{i,o}$ are empty. Thus, $\widehat{Q}^{a,b}_{k,o}$ consists of $(r_{1,g} \oplus o(a,b))$ for $k = 1$ and $r_{k,o}$ for $k > 1$. Since for all wires e and all j the values of $r_{j,e}$ are given, all values r_e are given, too. Hence, the different choices of the content of $\widehat{W}_{i,e_1}, \widehat{W}_{i,e_2}$ to construct a given table Q_o can be reduced to the different choices of the content of $\widehat{W}_{i,e_1}, \widehat{W}_{i,e_2}$ to construct a given table

$$\widetilde{Q}_o = \Pi_{r_{e_1}}(\widehat{W}_{i,e_1}) \otimes \Pi_{r_{e_2}}(\widehat{W}_{i,e_2}) \; .$$

Thus, for every choice of the content of $\widehat{W}_{i,e_1}$ there exists exactly one choice of the content of $\widehat{W}_{i,e_2}$ such that $\widetilde{Q}_o$ has the desired shape, i.e., in total there are $2^{|\widehat{W}_{i,e_1}|}$ different choices. To count all possible contents of all random variables $W^0_{i,e}, W^1_{i,e}$

in R_i we assume in the following, that the values of $W^0_{i,e_1}, W^1_{i,e_1}, W^0_{i,e_2}, W^1_{i,e_2}$ of the input wires of o are given.
Inductively, for all previous choices, there exists at least one gate g such that all values $\widehat{W}_{i,e}, \widehat{W}_{i,e}$ of the output wires of g are determined. Hence, $\Pi_{r_{e_1},r_{e_2}}(\oplus_{k=1}^{t+1})\widehat{Q}_{k,g})$ is known, where e_1, e_2 denote the input wires of g.
Since all permutation bits r_{e_1}, r_{e_2} are given, for every choice of the content of $\widehat{W}_{i,e_1}$ there exists exactly one possibility content of $\widehat{W}_{i,e_1}$, such that the garbled table of g has the shape as given in Γ. Hence, for each previous choice there are $2^{|\widehat{W}_{i,e_1}|}$ choices for the random bits of gate g made by AI_i such that the garbled table of g has the desired shape.
Thus, there are $\prod_e 2^{|\widehat{W}_{i,e}|} = 2^{|\Gamma|}$ possible choices for the random values in R_i such that $R_i \cup R_{\neg i}$ result in a garbled circuit of the desired shape Γ. □
The first protocol to generate a garbled circuit performs the following steps for every wire e and every gate g:

1. *Distributing Phase*

   (a) Every AI_k generates a $t+1$-share

   $$r_{k,1,e}, \ldots, r_{k,t+1,e}$$

   of every random bit $r_{k,e}$ and sends $r_{k,i,e}$ to all slaves $S_{h,i}$ with $h \in [1, t+1]$, i.e., to all slaves in column i of the array.
   After receiving $r_{k,i,e}$ from every auction issuer every slave $S_{h,i}$ of the ith column computes

   $$s_{i,e} = \oplus_{k=1}^{t+1} r_{k,i,e} \ .$$

   (b) Every AI_k generates $t+1$-shares

   $$\widehat{W}^1_{k,0,e}, \ldots, \widehat{W}^{t+1}_{k,0,e} \qquad \text{and} \qquad \widehat{Q}^1_{k,0,g}, \ldots, \widehat{Q}^{t+1}_{k,0,g}$$

   of the random values $\widehat{W}_{k,e}$ and $\widehat{Q}_{k,g}$. Afterwards, he sends $\widehat{W}^i_{k,0,e}$ and $\widehat{Q}^i_{k,0,g}$ to the slave $S_{i,1}$.

2. *Permuting Phase*
   After receiving the values $\widehat{W}^k_{i,j-1,g}$ and $\widehat{Q}^k_{i,j-1,g}$ for every $i \in [1, t+1]$ slave $S_{k,j}$ computes for every gate g and every wire e

   $$\widehat{W}_{k,j,e} = \Pi_{s_{j,e}}(\oplus_{i=1}^{t+1}\widehat{W}^k_{i,j-1,e}) \quad \text{and} \quad \widehat{Q}_{k,j,g} = \Pi_{s_{j,e_1},s_{j,e_2}}(\oplus_{i=1}^{t+1}\widehat{Q}^k_{i,j-1,g})$$

   where e_1, e_2 are the input wires of g.
   If $j < t+1$, $S_{k,j}$ generates $t+1$-shares

   $$\widehat{W}^1_{k,j,e}, \ldots, \widehat{W}^{t+1}_{k,j,e} \qquad \text{and} \qquad \widehat{Q}^1_{k,j,g}, \ldots, \widehat{Q}^{t+1}_{k,j,g}$$

   of $\widehat{W}_{k,j,e}$ and $\widehat{Q}_{k,j,g}$ and sends $\widehat{W}^i_{k,j,e}$ and $\widehat{Q}^i_{k,j,g}$ to slave $S_{i,j+1}$.

3. *Combination Phase*

(a) The slaves $S_{k,t+1}$ in the last column compute for every gate g

$$Q_{k,g} \;=\; (\widehat{W}_{k,t+1,e_1} \otimes \widehat{W}_{k,t+1,e_2}) \oplus \widehat{Q}_{k,t+1,g}$$

where e_1, e_2 are the input wires of g and send $Q_{k,g}$ to the auctioneer A.

(b) Finally, A computes

$$Q_g \;=\; \oplus_{k=1}^{t+1} Q_{k,g} \;.$$

We call this protocol the *field-protocol*. The field-protocol can be simulated on a complete network of $2t+1$ parties.

Theorem 3 *The garbled circuit Γ of an auction circuit can be constructed t-privately by using $t+1$ auction issuers, $(t+1)^2$ slaves, and $O(|\Gamma|t^3)$ random bits where $|\Gamma|$ denotes the length of the binary representation of Γ.*

3.2 A Slave-Efficient Protocol

We will now present a t-private protocol that generates the garbled circuit Γ by using $t+1$ auction issuers and t slaves. Note that this number of parties matches the number of parties that are needed to compute a function t-privately if the function has an embedded OR-function [21], and therefore, it is optimal in the number of parties.

To present our protocol we will proceed in two steps: In a first step we will present a protocol for constructing a garbled circuit, where every slave receives messages only from one corresponding slave. This protocol uses an exponential number of slaves. In a second step we will show that this protocol can be simulated by $2t+1$ parties. The following protocol is closely related to the protocol discussed in the last section. In the first step the $t+1$ auction issuers AI_k generate the tables $\widehat{Q}_{k,g}$, the vectors $\widehat{W}_{k,e}$, and the random bits $r_{k,e}$ as in the protocol above.

Our protocol starts by generating shares of

$$\Pi_{\oplus_{k=1}^{t+1} r_{k,e_1}, \oplus_{k=1}^{t+1} r_{k,e_1}}(\widehat{Q}_{k,g}) \qquad \text{and} \qquad \Pi_{\oplus_{k=1}^{t+1} r_{k,e}}(\widehat{W}_{k,e})$$

t-privately. This will be done by using $t+1$ *virtual* $t+1$-ary complete trees $T_1, \ldots, T_{t+1}$ of depth t. Later, these virtual trees will represent ways of communication within a complete network of $2t+1$ nodes. The nodes of a virtual tree will be mapped to the complete network.

We denote the nodes of T_k by S_k^σ, where $\sigma \in [1, t+1]^{\leq t}$ denotes the path from the root of T_k to S_k^σ. That means, $S_k^\lambda = AI_k$ (λ denotes the empty string) gives the root of T_k, and for every internal node S_k^σ its direct successors are denoted by $S_k^{\sigma,1}, \ldots, S_k^{\sigma,t+1}$. We will now describe the *tree-protocol* that runs on the trees T_k. Afterwards we will show how to implement this protocol by using only $2t+1$ parties without losing t-privacy. The tree-protocol runs as follows:

1. *Distributing Phase*
   All AI_i generate for all random bits $r_{i,e}$ and every node S_k^u of T_k a random bit $r_{i,k,e}^u$ such that

$$r_{i,e} = \oplus_{u \text{ is a prefix of } \sigma} r_{i,k,e}^u$$

for every leaf S_k^σ. Then, the AI_i sends $r_{i,k,e}^u$ to S_k^u and every node S_k^u computes

$$s_{k,e}^u = \oplus_{i=1}^{t+1} r_{i,k,e}^u \ .$$

2. *Permuting Phase*
   Now, the tables $\widehat{Q}_{k,g}$ and the vectors $\widehat{W}_{k,e}$ are shared and manipulated on the trees as follows: Let $\widehat{Q}_{k,g}^u$, $\widehat{W}_{k,e}^u$ denote the values received by S_k^u. In the case that $u = \lambda$ we choose $\widehat{Q}_{k,g}^u = \widehat{Q}_{k,g}$ and $\widehat{W}_{k,e}^u = \widehat{W}_{k,e}$. For $|u| < t$ S_k^u computes $t+1$-shares $\widehat{Q}_{k,g}^{u,1}, \ldots, \widehat{Q}_{k,g}^{u,t+1}$ and $\widehat{W}_{k,e}^{u,1}, \ldots, \widehat{W}_{k,e}^{u,t+1}$ of $\Pi_{s_{k,e_1}^u, s_{k,e_2}^u}(\widehat{Q}_{k,g}^u)$ and $\Pi_{s_{k,e}^u}(\widehat{W}_{k,e}^u)$. Finally, S_k^u sends $\widehat{Q}_{k,g}^{u,i}$ and $\widehat{W}_{k,e}^{u,i}$ to $S_k^{u,i}$.

3. *Combination Phase*

   (a) For $|\sigma| = t$ S_k^σ computes a $t+1$-share $Q_{k,g}^{\sigma,1}, \ldots, Q_{k,g}^{\sigma,t+1}$ of

   $$(\Pi_{s_{k,e_1}^\sigma}(\widehat{W}_{k,e_1}^\sigma) \otimes \Pi_{s_{k,e_2}^\sigma}(\widehat{W}_{k,e_2}^\sigma)) \oplus \Pi_{s_{k,e_1}^\sigma, s_{k,e_2}^\sigma}(\widehat{Q}_{k,g}^\sigma)$$

   and sends $Q_{k,g}^{\sigma,i}$ to AI_i.

   (b) AI_i computes

   $$Q_g^i = \oplus_{\sigma \in [1,t+1]^t} \oplus_{k \in [1,t+1]} Q_{k,g}^{\sigma,i}$$

   and sends the resulting tables to A.

   (c) Finally, A computes the garbled table

   $$Q_g = \oplus_{i \in [1,t+1]} Q_g^i \ .$$

Note that we can run the tree-protocol for all the $t+1$ virtual trees on $\frac{(t+1)^{t+1}-1}{t}$ nodes.

Lemma 4 *The tree-protocol gives a t-private construction of the garbled circuit Γ using $\frac{(t+1)^{t+1}-1}{t}$ parties.*

In the following we will show that we can simulate the tree-protocol by a network of $2t+1$ parties such that our protocol is still t-private. We will call the resulting protocol the *slave-efficient-protocol*. To distinguish between the parties used in the tree-protocol and the parties used in the slave-efficient-protocol we call the former *virtual parties* and the latter *real parties*. In the simulation we will guarantee the following properties:

1. no real party simulates two virtual parties on the same root-leaf path, i.e., for every path $\sigma \in [1, t+1]^{\leq t}$ and all $u \neq u'$ that are prefixes of σ, it holds that S_k^u and $S_k^{u'}$ are simulated by different parties,

2. no real party simulates two virtual parties that are direct successor of the same virtual party in a virtual tree, for all paths $\sigma \in [1, t+1]^{\leq t-1}$ and all $q \neq q' \in [1, t+1]$, $S_k^{\sigma,q}$ and $S_k^{\sigma,q'}$ are simulated by different parties, and

3. the roots of the virtual trees are simulated by different real parties.

The privacy proof of the slave-efficient-protocol will be based on these properties. Recall that there are $t+1$ trees. For $k \in [1, t+1]$ the auction issuer $AI_k = S_k^\lambda$ is the root of the $t+1$-ary complete tree T_k.
Inductively, we show that $2t+1$ parties are sufficient to realize a mapping from the virtual parties to the real parties that fulfills the properties required above.

1. As mentioned above, the auction issuers AI_k, $k \in [1, t+1]$, simulating the roots S_k^λ of the trees T_k. Hence, we need $t+1$ parties for these simulation steps.

2. Assume that we can simulate the subtrees T_k^ℓ of all T_k with nodes S_k^u with $u \in \{1, \ldots, t+1\}^{\leq \ell}$ by $t+1+\ell$ real parties such that our properties above are fulfilled. Let us now consider the virtual parties $S_k^{u',i}$ with $u' \in \{1, \ldots, t+1\}^\ell$ and $i, k \in \{1, \ldots, t+1\}$. Since $S_k^{u'}$ hat ℓ predecessors, our simulation uses $\ell+1$ real parties to simulate the computation of the virtual paths on the path from the root S_k^λ to $S_k^{u'}$. Thus, we can use the remaining t parties of the already used $t+1+\ell$ real parties to simulate the computation of t direct successors of $S_k^{u'}$. Hence, we need one additional new real party to simulate the computation of the remaining direct successor of $S_k^{u'}$. Note that this leads to a protocol, where $t+1+\ell+1$ real parties are used for the simulation of the virtual nodes of the subtrees $T_k^{\ell+1}$ of all T_k with nodes S_k^u with $u \in \{1, \ldots, t+1\}^{\leq \ell+1}$.

3. Since the virtual trees T_k have depth t, our simulation uses $2t+1$ real parties. Furthermore, from the construction above it follows, that the simulation fulfills the three properties from above.

From the three properties of our simulation it follows that:

Theorem 5 *There is a t-private protocol to build Γ for $2t+1$ parties using $\mathcal{O}((t+1)^{t+3}|\Gamma|)$ random bits.*

Recall that the parameter t does not limit the number of users of the protocol. In fact t denotes the number of untrusted parties within in the subset of parties that generate the garbled circuit.

3.3 t-Private Bidding

Let $x_{i,j}$ denote the jth bit of the bid x_i of bidder B_i and let e be a wire of the circuit C connected to $x_{i,j}$. Our protocol should give the auctioneer A access to $W_e^{x_{i,j}} \circ (r_e \oplus x_{i,j})$ in a t-private way. To simplify our analysis we assume that A publishes Γ and $W_e^{x_{i,j}} \circ (r_e \oplus x_{i,j})$. For the field-protocol we proceed as follows:

1. B_i computes a $t+1$-share $x_{i,j}^1, \ldots, x_{i,j}^{t+1}$ of $x_{i,j}$ and sends $x_{i,j}^k$ to AI_k.

2. AI_k computes a $t+1$-share $c_{k,e}^1, \ldots, c_{k,e}^{t+1}$ of $x_{i,j}^k \oplus r_{k,e}$ and sends $c_{k,e}^i$ to $S_{1,i}$.

3. $S_{1,i}$ sends $c_e^i = \oplus_{k=1}^{t+1} c_{k,e}^i$ to all slaves in the last row $S_{j,t+1}$.

4. All $S_{k,t+1}$ compute $c_e = \oplus_{i=1}^{t+1} c_e^i$ and send $\widehat{W}_{k,t+1,e}^{c_e} \circ c_e$ to A. Here, $\widehat{W}_{k,t+1,e}^{c_e}$ denotes the c_eth half of the tuple $\widehat{W}_{k,t+1,e}$.

5. Finally, A computes $W_e^{x_{i,j}} \circ c_e = (\oplus_{k=1}^{t+1} \widehat{W}_{k,t+1,e}^{c_e}) \circ c_e$.

After the bidding the auctioneer can easily evaluate the garbled circuit and therefore determine the value of this circuit. This can be done gate by gate using the formula given in Equation (1). Note that this formula gives us the encrypted value of the corresponding gate, and the keys to compute the encrypted values of the direct successors of the gate.
Since the evaluation of the garbled circuit on the encrypted inputs gives an encrypted output, the result of the auction remains hidden. To decrypted the output all auction issuers AI_i have to publish their random bits $r_{i,g}$ for every output gate g.

Theorem 6 *There exists a t-private protocol for an electronic auction system with $t+1$ auction issuers and $(t+1)^2$ slaves using $O(t^3|\Gamma|)$ random bits.*

The protocol for the bidding phase for our slave efficient protocol works analogously to the bidding protocol presented above.

1. B_i computes a $t+1$-share $x_{i,j}^1, \ldots, x_{i,j}^{t+1}$ of $x_{i,j}$ and sends $x_{i,j}^k$ to AI_k.

2. AI_k computes a $t+1$-share $c_{k,e}^1, \ldots, c_{k,e}^{t+1}$ of $x_{i,j}^k \oplus r_{k,e}$ and sends $c_{k,e}^i$ to AI_i.

3. AI_i sends $c_e^i = \oplus_{k=1}^{t+1} c_{k,e}^i$ to all leaves S_k^σ with $\sigma \in \{1, \ldots, t+1\}^t$ of all virtual trees T_k.

4. Every leaf S_k^σ compute $c_e = \oplus_{i=1}^{t+1} c_e^i$ and send $\widehat{W}_{k,t+1,e}^{\sigma,c_e} \circ c_e$ to A.

5. Finally, A computes $W_e^{x_{i,j}} \circ c_e = (\oplus_{k=1}^{t+1} \oplus_{\sigma \in \{1,\ldots,t+1\}^t} \widehat{W}_{k,t+1,e}^{\sigma,c_e}) \circ c_e$.

To evaluate the garbled circuit we can again use the formula given in Equation (1).

Theorem 7 *There exists a t-private protocol for an electronic auction system with $t+1$ auction issuers and t slaves using $O((t+1)^{t+3}|\Gamma|)$ random bits.*

4 Proofs of Privacy of our Protocols

In this section we will prove our theorems presented in the previous section. The proofs of t-privacy are based on some structural properties:

Definition 8 *Let U be a collusion of at most t parties. There exists a* trusted skeleton *of the field-protocol according to U, if there exists $i, j \in \{1, \ldots, t+1\}$ such that the auction issuer AI_i, all slaves in column $S_{1,i}, \ldots, S_{t+1,i}$, and all slaves in row $S_{j,1}, \ldots, S_{j,t+1}$ are honest.*

Lemma 9 *For every collusion of at most t parties there exists a trusted skeleton for the field-protocol.*

Proof: By the pigeon hole principle it follows that there exists $i \in \{1, \ldots, t+1\}$ such that the auction issuer AI_i and all slaves in column $S_{1,i}, \ldots, S_{t+1,i}$ are honest, i.e., they are not members of the collusion. Analogously, there exists $j \in \{1, \ldots, t+1\}$ such that all slaves in row $S_{j,1}, \ldots, S_{j,t+1}$ are honest. □

Definition 10 *Let U be a collusion of at most t parties. Then, a* trusted skeleton (TS) *of the field-protocol consists of the honest auction issuer AI_i, the honest slaves in $S_{\cdot,i}$ and $S_{j,\cdot}$, and all honest bidders. The* extended collusion (EC) *consists of the auctioneer and the remaining auction issuers, slaves, and bidders that are not in trusted skeleton. The sets*

$$[\mathrm{TS}] \subseteq \mathrm{TS} \times \mathrm{TS}, \qquad [\mathrm{EC}] \subseteq \mathrm{EC} \times \mathrm{EC},$$
$$[\chi] \subseteq (\mathrm{TS} \times \mathrm{EC} \setminus \{A\}) \cup (\mathrm{EC} \setminus \{A\} \times \mathrm{TS}),$$
$$[\mathrm{SA}] = \{(S_{j,t+1}, A)\}, \qquad \textit{and} \qquad [\mathrm{AIA}] = \{(AI_i, A), (A, AI_i)\}$$

consist of all (sender, receiver)-pairs from the corresponding domains that exchange messages in the field-protocol.

[TS] describes the communication channels between all members in the trusted skeleton, [EC] between all members in the extended collusion, $[\chi]$ between members of trusted skeleton and extended collusion without the auctioneer, [SA] between the honest slave in the last column and the auctioneer, and [AIA] between the honest auction issuer AI_i and the auctioneer A.
Due to arguments of symmetry, we can permute the numbering of rows and auction issuer such that row i does not belong to the collusion.
We will now define the trusted skeleton of the slave-efficient protocol.

Definition 11 *Let U be a collusion of at most t parties. There exists a* trusted skeleton *of the slave-efficient protocol according to U, if*

- *there exists $i \in \{1, \ldots, t+1\}$ such that the root of T_i is simulated by the honest real party,*
- *for every $k \in \{1, \ldots, t+1\}$ and every root-leaf path in the virtual tree T_k there exists a virtual party on this path that is simulated by the honest real party and*
- *for every virtual tree T_i and every node u of T_i that is simulated by the honest real party there exists a u-leaf path in T_i such that every virtual party on this path will be simulated by the honest real party.*

Lemma 12 *For every collusion of at most t parties there exists a trusted skeleton for the slave-efficient-protocol.*

Proof: Recall, that our simulation follows the following properties:

1. no real party simulates two virtual parties on the same root-leaf path, i.e., for every path $\sigma \in [1, t+1]^{\leq t}$ and all $u \neq u'$ that are prefixes of σ it holds that S_k^u and $S_k^{u'}$ are simulated by different parties,

2. no real party simulates two virtual parties that are direct successor of the same virtual party in a virtual tree, for all paths $\sigma \in [1, t+1]^{\leq t-1}$ and all $q \neq q' \in [1, t+1]$, $S_k^{\sigma,q}$ and $S_k^{\sigma,q'}$ are simulated by different parties, and

3. the roots of the virtual trees are simulated by different real parties.

Hence, by the pigeon hole principle it follows, that for every virtual non-leaf node of every virtual tree there exists a direct successor that is simulated by an honest real party. Moreover, since all roots are simulated by different real parties, there exists at least one root $S_i^{\lambda} = AI_i$ that is simulated by an honest real party. From the former observation it follows, that there exists a root-leaf path in the virtual tree T_i where every virtual party on this path will be simulated by the honest real party. Analogously, it follows that for every virtual tree T_i and every node u of T_i that is simulated by the honest real party, there exists a path from u to a leaf of T_i such that every virtual party on this path will be simulated by the honest real party. Since there are $t+1$ virtual parties on each root-leaf-path of every virtual tree and these virtual parties are simulated by $t+1$ real parties, at least one of these real parties is honest. □

Definition 13 *Let U be a collusion of at most t parties. Then, a* trusted skeleton (TS) *of the slave-efficient protocol consists of all honest bidders and of $t+1$ honest parties (auction issuers and slaves), such that*

- *there exists $i \in \{1, \ldots, t+1\}$ such that there exists a root-leaf path in the virtual tree T_i where every virtual party on this path will be simulated by a party in* TS *and*

- *for every $k \in \{1, \ldots, t+1\}$ and every root-leaf path in the virtual tree T_k there exists a virtual party on this path that simulated by a party in* TS.

The extended collusion (EC) *consists of the auctioneer and the remaining auction issuers, slaves, and bidders that are not in trusted skeleton. The sets*

$$\begin{gathered}[\mathrm{TS}] \subseteq \mathrm{TS} \times \mathrm{TS}, \qquad [\mathrm{EC}] \subseteq \mathrm{EC} \times \mathrm{EC}, \\ [\chi] \subseteq (\mathrm{TS} \times \mathrm{EC} \setminus \{A\}) \cup (\mathrm{EC} \setminus \{A\} \times \mathrm{TS}), \\ \textit{and} \qquad [\mathrm{AIA}] = \{(\mathrm{TS}, A), (A, \mathrm{TS})\}\end{gathered}$$

consist of all (sender, receiver)-pairs from the corresponding domains that exchange messages in the field-protocol.

4.1 t-Privacy of the Auction Protocols

Our privacy proofs are based on the following idea: We analyze the random variable C describing the communication between the trusted skeleton and the extended collusion. Assume that an adversary controlling the extended collusion can generate a random variable S, that is indistinguishable from C only by knowing the garbled circuit and the random bits chosen by the parties of the extended collusion. Then,

the adversary cannot deduce any knowledge from the C and Γ that cannot be deduced from Γ.
Let R_i be a discrete random variable describing the sequence of the random variables $\widehat{W}_{i,e}$ and $r_{i,e}$ for all wires e of the auction issuer AI_i of the trusted skeleton. Analogously, let $R_{\neg i}$ describe the sequence of the random variables $\widehat{W}_{i,e}$ and $r_{i,e}$ for all wires e of the auction issuer of the extended collusion. Furthermore, we interpret Γ as a discrete random variable for the generated garbled circuit. Let R_{EC} resp., R_{TS} be the discrete random variables describing all other random bits uniformly chosen by the extended collusion resp., the trusted skeleton. Let U be a uniformly distributed discrete random variable over binary strings of an adequate length. Let $C(R_i, R_{\neg i}, R_{TS}, R_{EC}, \Gamma)$ be a binary description of the communication between the extended collusion and the trusted skeleton generated by a run of the field-protocol. The random choices of the parties are given by $R_i, R_{\neg i}, R_{TS}, R_{EC}$ and the generated garbled circuit is given by Γ. Finally, let $\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma)$ be a binary description of a communication string generated by a simulator that works as follows:

- The simulator performs the same operations as the auction issuers and slaves of the extended collusion. During this simulation the random choices of the parties in the extended collusion are taken from $R_{\neg i}$ and R_{EC}.
- We divide the strings given by U into disjoint blocks of adequate length. Any message sent through $[\chi]$ from a party of the trusted skeleton to a party of the extended collusion is given by a separate block of U, i.e., these messages are independent and uniformly chosen.
- Any message sent through $[\chi]$ from a party P of the extended collusion to a party of the trusted skeleton is computed by the simulation described above, i.e., we simulate the operations performed by P on the messages received by P in the simulation and on the random choices of P given by $R_{\neg i}$ and R_{EC}.
- All messages sent through [SA] are deterministically determined by Γ, R_i and the previous choices for the communication through the channels of $[\chi]$.

Lemma 14 *For all given garbled circuits γ, all given strings $r_i, r_{ec}, r_{\neg i}$, and all given communication strings c it holds that*

$$\begin{aligned} & \Pr(\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma) = c \mid \Gamma = \gamma,\ R_{\neg i} = r_{\neg i},\ R_i = r_i, R_{EC} = r_{ec}) \\ = \ & \Pr(C(R_i, R_{\neg i}, R_{TS}, R_{EC}, \Gamma) = c \mid \Gamma = \gamma,\ R_{\neg i} = r_{\neg i},\ R_i = r_i,\ R_{EC} = r_{ec}). \end{aligned}$$

Proof: We fix any message sent through $[\chi]$ from a party in the trusted skeleton to a party of the extended collusion. Recall that this message must be an element of a $(t+1)$-share such that at least one element sh of this $(t+1)$-share is only sent to a member of the trusted skeleton. sh is determined by R_{TS}. The t elements of the $(t+1)$-share sent from a party in the trusted skeleton to a party of the extended collusion are are independently and uniformly chosen. That means, a certain message sent from a party in the trusted skeleton to a party of the extended collusion is chosen with probability $1/2^{|sh|}$.

In both settings – using the simulator resp., using the real protocol – all messages sent from a party of the extended collusion to a party of the trusted skeleton through $[\chi]$ are determined by the messages sent from a party of the trusted skeleton to a party of the extended collusion through $[\chi]$ and by U resp., R_{EC}. Note that both U and R_{EC} are independently and uniformly distributed.
The simulator and the protocol compute the messages from all auction issuers AI_j of the extended collusion deterministically using $r_{\neg i}$ and r_{ec}. Since all strings are uniformly distributed those messages are equally distributed.
Furthermore, in both settings the messages sent from the slaves $S_{\cdot,t+1}$ in the last column are fully determined by Γ, R_{EC}, and the messages previously received by the slaves $S_{\cdot,t+1}$. As seen above these messages only depend on U resp., R_{EC}. Thus, the messages sent from the slaves $S_{\cdot,t+1}$ to the auctioneer A are uniformly distributed in both settings.
Hence,

$$\begin{aligned} & \Pr(\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma) = c \mid \Gamma = \gamma,\ R_{\neg i} = r_{\neg i},\ R_i = r_i, R_{EC} = r_{ec}) \\ = \ & \Pr(C(R_i, R_{\neg i}, R_{TS}, R_{EC}, \Gamma) = c \mid \Gamma = \gamma,\ R_{\neg i} = r_{\neg i},\ R_i = r_i,\ R_{EC} = r_{ec}) \ . \end{aligned}$$

□

Proof of Theorem 3: For the t-privacy we show that

$$\mathrm{I}(R_i; C \mid \Gamma, R_{EC}, R_{\neg i}) = \mathrm{I}(C; R_i \mid \Gamma, R_{EC}, R_{\neg i}) = 0 \ .$$

That means, the communication string seen by the extended collusion (here described by C for short) gives no additional knowledge about the content of R_i chosen by AI_i for the garbled circuit, if an adversary controlling the extended collusion knows the generated garbled circuit Γ and the random values chosen by the parties of the extended collusion.

$$\begin{aligned} \mathrm{I}(C; R_i \mid \Gamma, R_{EC}, R_{\neg i}) \ = \ & \mathrm{H}(C(R_i, R_{\neg i}, R_{TS}, R_{EC}, \Gamma) \mid \Gamma, R_{EC}, R_{\neg i}) \\ & - \mathrm{H}(C(R_i, R_{\neg i}, R_{TS}, R_{EC}, \Gamma) \mid R_i, \Gamma, R_{EC}, R_{\neg i}). \end{aligned}$$

Using Lemma 14 it holds that

$$\begin{aligned} & \mathrm{H}(C(R_i, R_{\neg i}, R_{TS}, R_{EC}, \Gamma) \mid R_i, \Gamma, R_{EC}, R_{\neg i}) \\ = \ & \mathrm{H}(\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma) \mid R_i, \Gamma, R_{EC}, R_{\neg i}) \end{aligned}$$

and

$$\begin{aligned} & \mathrm{H}(C(R_i, R_{\neg i}, R_{TS}, R_{EC}, \Gamma) \mid \Gamma, R_{EC}, R_{\neg i}) \\ = \ & \mathrm{H}(\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma) \mid \Gamma, R_{EC}, R_{\neg i}) \ . \end{aligned}$$

Since U and R_i are uniformly distributed and independent from each other, it holds that

$$\begin{aligned} & \mathrm{H}(\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma) \mid R_i, \Gamma, R_{EC}, R_{\neg i}) \\ = \ & \mathrm{H}(\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma) \mid \Gamma, R_{EC}, R_{\neg i}) \ . \end{aligned}$$

Hence,

$$\begin{aligned} \mathrm{I}(C; R_i \mid \Gamma, R_{EC}, R_{\neg i}) \ = \ & \mathrm{H}(\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma) \mid \Gamma, R_{EC}, R_{\neg i}) \\ & - \mathrm{H}(\mathrm{Sim}(U, R_{\neg i}, R_{EC}, \Gamma) \mid R_i, \Gamma, R_{EC}, R_{\neg i}) \ = \ 0 \ . \end{aligned}$$

Since R_{EC} and $R_{\neg i}$ are independently chosen from R_i they do not provide any information about R_i. Thus, the adversary controlling the extended collusion can only deduce information about R_i from the garbled circuit Γ (resp., from the shares of Γ that are sent from $S_{i,t+1}$ to the auctioneer). By construction Γ (resp., the shares of Γ from $S_{i,t+1}$) is a garbled circuit and gives no information about the values chosen from AI_i. As no further communication takes place, the construction of Γ is t-private.
Finally, let us denote that the number of participating parties and the number of used random bits directly follows from the protocol. □

Proof of Lemma 4: Since each tree T_k is a $t+1$-ary tree of depth t, T_k consists of $\frac{(t+1)^{t+1}-1}{t}$ nodes. Thus, we can set up each tree such that each party controls at least one node in T_k.
Let U be a collusion of at most t parties. We say there exists a *trusted skeleton* (TS) of the tree-protocol according to U, if

- there exists $i \in \{1, \ldots, t+1\}$ such that the root of T_i is not in U.
- for every $k \in \{1, \ldots, t+1\}$ and every root-leaf path in T_k there exists a honest party on this path.
- for every tree T_i and every honest node u of T_i there exists a u-leaf path in T_i such that every party on this path is honest. This also follows from the fact that each internal node must have at least one honest successor.

It is easy to verify that a trusted skeleton of the tree-protocol according to a collusion of at most t parties always exists.
Let V denote a trusted skeleton then we call the remaining parties the *extended collusion (EC)*. We denote the set of communication channels between the parties of the trusted skeleton and the parties of the extended collusion except of the auctioneer as $[\chi]$.
Similar to the proof of Theorem 3 we can introduce a simulator that uniformly chooses random strings for the messages sent through the channels of $[\chi]$. Furthermore, the messages sent to the auctioneer A deterministically depend on Γ, the random bits of the extended collusion, and the messages sent through $[\chi]$.
Since for each $(t+1)$-share that is computed in the tree-protocol at least one element is send to a member of the trusted skeleton, we can prove a result analogously to Lemma 14 for the tree-protocol. Thus, the communication does not provide any information about the choices for $r_{i,e}, \widehat{W}_{i,e}, \widehat{Q}_{i,g}$ for all wires e and gates g. □

Proof of Theorem 5: Let us consider a trusted skeleton for the slave-efficient protocol. If we take the inverse of the mapping of the virtual parties of the tree-protocol to the real parties of the slave-efficient protocol as defined in Section 3.2, one can see that we get a trusted skeleton for the tree-protocol as defined in Lemma 4. Since an adversary that controls the extended collusion of the tree-protocol receives the same messages as an adversary that controls the corresponding extended collusion of the slave-efficient protocol, both adversaries can deduce the same information. For Lemma 4 we can deduce that our protocol gives a t-private construction of Γ by using $2t+1$ parties. □

4.2 t-Private Bidding

Proof of Theorem 6: In the bidding process there exist (at least) one auction issuer AI_i and two slaves $S_{1,i}, S_{j,t+1}$ that are members of the trusted skeleton of the field-protocol. We fix Γ and all random strings of the extended collusion. Using the simulator argument, we can see that all communication seen by the extended collusion does neither give any information about the random strings of the trusted skeleton nor of the honest bidders: In steps (1) and (2) the bidders, resp., the auction issuers, generate $t+1$-shares. One element of each $t+1$-share is sent to a member of the trusted skeleton. Thus, the other shares are only random strings (see Lemma 14). In steps (3) and (4) c_e is computed and $\widehat{W}^{c_e}_{k,t+1,e} \circ c_e$ is published.
Assuming that the adversary knows Γ and $\widehat{W}^{c_e}_{k,t+1,e} \circ c_e$ for all input bits $x_{i,j}$ of the bids, we can show a lemma simular to Lemma 2:

Lemma 15 *For every shape Γ of the garbled circuit, every value of the strings $\widehat{W}^{c_e}_{k,t+1,e} \circ c_e$ of the input bits of the circuit, every content $R_{\neg i}$ of the random values chosen by $AI_j \neq AI_i$, and every value of the input bits $x_{i,j}$ of the bids of the bidders of the trusted skeleton, there exist $\mu = 2^{|\Gamma|/4}$ different values of the content of R_i such that $R_i \cup R_{\neg i}$ defines the same shape Γ of the garbled circuit and the same values $\widehat{W}^{c_e}_{k,t+1,e} \circ c_e$ of the encrypted input bits $x_{i,j}$.*

Proof: The proof follows analogously to the proof of Lemma 2. It is based on counting the number of different values for the random variables R_i of the auction issuer in the trusted skeleton, such that $R_i \cup R_{\neg i}$ defines the same shape Γ of the garbled circuit and the same values $\widehat{W}^{c_e}_{k,t+1,e} \circ c_e$ of the encrypted input bits $x_{i,j}$.
In a first step we evaluate the garbled circuit. One can see that this evaluation determines exactly one of the four entries of each garbled table. Since, our goal is to find a fixed but arbitrary garbled table, the value of the determined entry is fixed. Since the corresponding value of $W^{g(a,b)}_{i,g}$ and $r_{i,g}$ are chosen by the trusted auction issuer, this entry can be chosen to have the desired shape. In the following we use value of $W^{1-g(a,b)}_{i,g}$ to show that the garbled tables of the direct successors of g can be generated to have the desired shape.
Let us continue now with the possible shapes of the remaining three entries of the garbled table of g. Since we have to choose $W^{1-g(a,b)}_{i,g}$ such that all possible shapes of the garbled tables of the direct successors of g are admissible, we can assume that this value is fixed in the construction of the garbled table of g by the shape of the garbled tables of its direct successors. Note that two of the remaining three entries of the garbled table of g are only encrypted by one half of the so far undetermined strings W^{1-a}_{i,e_1} and W^{1-b}_{i,e_2}. Hence these values are fixed by the desired shape of the garbled table of g. For the last entry it follows that for every value the remaining half of W^{1-a}_{i,e_1} we can choose one value of the remaining half of W^{1-b}_{i,e_2} such that we get the desired shape of the garbled table of g. Summarizing there are $2^{|Q_g|/4}$ different values for the random variable in R_i that we can use to get the desired shape of Q_g. Taking the product over all gates gives us the number of different values for the random variable in R_i such that $R_i \cup R_{\neg i}$ defines the same shape Γ of the garbled

circuit and the same values $\widehat{W}^{c_e}_{k,t+1,e} \circ c_e$ of the encrypted input bits $x_{i,j}$. It holds that

$$\prod_g 2^{|Q_g|/4} \quad = \quad 2^{\sum_g |Q_g|/4} \quad = \quad 2^{|\Gamma|/4} \ .$$

□

From the Lemma above we can conclude that we cannot deduce any additional knowledge about the unknown input bits of the bidders in the trusted skeleton from the garbled circuit Γ and the encrypted input bits $\widehat{W}^{c_e}_{k,t+1,e} \circ c_e$.
Since the communication in the bidding process is independent from the communication in the field-protocol if the garbled circuit is given, the bidding and the evaluation processes do not provide any further information about the input bids of the trusted skeleton.
The correctness of the bidding process follows from the fact that

$$\oplus_{k=1}^{t+1} \widehat{W}^{c_e}_{k,e} = \Pi_{\oplus_{k=1}^{t+1} r_{k,e}}(W_e^{x_{i,j}}) \quad \text{where} \quad c_e = x_{i,j} \oplus \oplus_{k=1}^{t+1} r_{k,e} \ .$$

Finally, for all output gates o the permutation bits $r_{i,o}$ are sent to the auctioneer at the end of the auction. Then, the auctioneer is able to reveal the result of the auction. Thus, the auction system provides only the same information to the extended collusion as the collusion can deduce from the result of the auction. □
For the tree-protocol and the slave-efficient-protocol the bidding process runs analogously to our protocol for $(t+1)^2$ slaves, if we replace the slaves $S_{1,i}$ and $S_{i,t+1}$ by AI_i.
Proof of Theorem 7: As in the bidding process of the field-protocol at least one auction issuer AI_i and one leaf S_j^σ belongs to the trusted skeleton of the slave-efficient protocol. Furthermore, there are only $2t+1$ parties that are involved in the bidding process of the field-protocol. Due to the t-privacy slave-efficient protocol, this claim follows analogously to the proof of Theorem 6. □

5 Security against Active Attacks

If a node is malicious (Byzantine), then it does not need to follow the instructions of the protocol. It can arbitrarily drop, add or change messages. We define an active collusion as a collusion of t malicious nodes that are under control of a single active adversary. Thus, the active adversary knows all information gathered by the colluding nodes and he can arbitrarily instruct these nodes.
One can think of different aims the adversary tries to achieve when attacking our field protocol:

1. The adversary may try to change the outcome of the auction systematically.
2. The adversary is destructive, i.e., he tries to sabotage the execution or changes the result of the computation at random.

To make our field protocol resistant against these types of attacks we can construct the garbled circuit redundantly.

3. The aim of the third kind of adversary is to collect as much of information about the inputs as possible by manipulating messages.

In the following we focus on an upper bound for the leakage for the field-protocol for every kind of adversaries. We will show that the following theorem holds even in the case that the adversary controls all parties in the extended collusion (Definition 10).

Theorem 16 *Let l be the number of output bits of C. Then, an active adversary does not get more than l bits of information about the input bits of the bidders of the trusted skeleton, i.e., the protocol is (t, l)-secure.*

Assume that an adversary does not want to change the result of an auction, but tries to gain some additional information. Then, the theorem above gives us even more:

Corollary 17 *An active adversary that controls at most t parties without changing the output of an auction does not get any information about the bids of the honest bidders that cannot be deduced by a passive adversary controlling the same parties.*

Consequently, if the adversaries do not want to change the output of the auction our field protocol is secure against active adversaries.

5.1 Proof of Theorem 16

To prove Theorem 16 we will investigate the communication between a trusted skeleton of the parties in the network and all remaining parties. To simplify our proof, we will make the following changes of the distributing phase at the field-protocol.

> For every h, i, e, after a slave $S_{h,i}$ has computed $s_{i,e}$ he sends $s_{h,i}$ to all slaves $S_{j,i}$ in column i. After he has received all values $s_{i,e}$ from the slaves in column i, he verifies that all of these values are equal.

Note that such a verification is not necessary if we consider only passive adversaries. Analyzing our field-protocols for generating a garbled circuit and for the bidding, one can see that one can simulate an active adversary by using only one party that deviates from the presented protocols. All remaining deviations from the protocols can be simulated by `XOR`-ing some strings with the resulting garbled circuit or with the strings sent by the single party that deviates from the presented protocols. Note that a malicious bidder can be simulated by an honest bidder by changing its bid. Summarizing we get:

Fact 18 *Assume that all slaves $S_{j,i}$ of row i are members of the trusted skeleton. Then, every attack of an active adversary can be simulated by a passive adversary using the same extended collusion where only $S_{i-1,i-1}$, resp., AI_2 if $i = 1$, deviates from the given protocols.*

In the following we will assume that all parties except of the party $S_{i-1,i-1}$, resp., AI_2, emphasized in Fact 18 follow the protocol. We will call this party the *malicious party*. Note that all remaining parties expect the malicious party to send a fixed number of bits that are assumed to be shares of the permuted vectors

$$\widehat{W}_e = (W_e^0, W_e^1)$$

and of the permuted tables

$$\widehat{Q}_g \;=\; \begin{pmatrix} \widehat{Q}_g^{0,0} & \widehat{Q}_g^{0,1} \\ \widehat{Q}_g^{1,0} & \widehat{Q}_g^{1,1} \end{pmatrix} \text{ with } \widehat{Q}_g^{a,b} \;=\; W_g^{g(a,b)} \circ (r_g \oplus g(a,b)) \ .$$

The attack of the malicious party can be interpreted as an `XOR` of these values with some binary strings

$$\overline{W}_e = (\overline{W}_e^0, \overline{W}_e^1) \quad \text{and} \quad \overline{Q}_g \;=\; \begin{pmatrix} \overline{Q}_g^{0,0} & \overline{Q}_g^{0,1} \\ \overline{Q}_g^{1,0} & \overline{Q}_g^{1,1} \end{pmatrix} \ .$$

On the other hand, if we assume that one of the auction issuers of the extended collusion generates the vector $\widehat{W}_e \oplus \overline{W}_e$ instead of $\widehat{W}_e$, we can also assume to have the same communication between the extended collusion and the trusted skeleton. So we can assume that $\overline{W}_e^0, \overline{W}_e^1 \in 0^*$. Therefore, in the following we assume that the malicious party only modifies the shares of the permuted tables $\widehat{Q}_g$.
We divide the communication between the extended collusion and the trusted skeleton into three parts:

- the messages sent in the construction of the garbled circuit and the bidding by a party that is not a slave of the last column (through $[\chi]$), i.e., all messages except of the messages sent by $S_{i,t+1} \in \text{TS}$,
- the messages sent in for the construction of the garbled circuit and the bidding by $S_{i,t+1} \in \text{TS}$ (through [SA]), and
- the messages sent by the action issuer $AI_i \in \text{TS}$ to decrypt the result of the garbled circuit (through [AIA]), i.e., the random bits $r_{i,o}$ for all output gates o.

We will denote these parts of the communication by C_1, C_2, and C_3. Note that C_1 only consists of independent randomly chosen binary strings. Hence, it is not possible that any adversary can deduce any information from C_1. For a circuit of ℓ output gates C_3 consists of ℓ bits, hence, an adversary can deduce at most ℓ bits of information from these ℓ bits. In the remaining part of these section we will show that an adversary cannot deduce any information from C_1 and C_2.
After finishing C_1 and C_2 the adversary receives

$$\begin{aligned} & (\Pi_{\oplus_k r_{k,e_1}}(\oplus_k \widehat{W}_{k,e_1}) \otimes \Pi_{\oplus_k r_{k,e_2}}(\oplus_k \widehat{W}_{k,e_2})) \oplus \Pi_{\oplus_k r_{k,e_1}, \oplus_k r_{k,e_1}}((\oplus_k \widehat{Q}_{k,g}) \oplus \overline{Q}_g) \\ = \;& (\Pi_{\oplus_k r_{k,e_1}}(\oplus_k \widehat{W}_{k,e_1}) \otimes \Pi_{\oplus_k r_{k,e_2}}(\oplus_k \widehat{W}_{k,e_2})) \oplus \Pi_{\oplus_k r_{k,e_1}, \oplus_k r_{k,e_1}}(\oplus_k \widehat{Q}_{k,g}) \\ & \qquad \oplus \Pi_{\oplus_k r_{k,e_1}, \oplus_k r_{k,e_1}}(\overline{Q}_g) \end{aligned}$$

and

$$W_e^{x_{i,j}} \circ c_e \;=\; (\oplus_{k=1}^{t+1} \widehat{W}_{k,t+1,e}^{c_e}) \circ c_e$$

for every input wire e with input bit $x_{i,j}$. From the construction of the garbled circuit it follows that every combination of binary strings of adequate length may occur for

$$(\Pi_{\oplus_k r_{k,e_1}}(\oplus_k \widehat{W}_{k,e_1}) \otimes \Pi_{\oplus_k r_{k,e_2}}(\oplus_k \widehat{W}_{k,e_2})) \oplus \Pi_{\oplus_k r_{k,e_1},\oplus_k r_{k,e_1}}(\oplus_k \widehat{Q}_{k,g})$$

and for

$$(\oplus_{k=1}^{t+1} \widehat{W}_{k,t+1,e}^{c_e}) \circ c_e$$

for every input $x_1, x_2, \ldots$ with the same probability. As we have seen in Section 4 this even holds if the content of the random values of the parties in the extended collusion is given. Thus, even every combination of binary strings of adequate length may occur for

$$(\Pi_{\oplus_k r_{k,e_1}}(\oplus_k \widehat{W}_{k,e_1}) \otimes \Pi_{\oplus_k r_{k,e_2}}(\oplus_k \widehat{W}_{k,e_2})) \oplus \Pi_{\oplus_k r_{k,e_1},\oplus_k r_{k,e_1}}(\oplus_k \widehat{Q}_{k,g})$$

and for

$$(\oplus_{k=1}^{t+1} \widehat{W}_{k,t+1,e}^{c_e}) \circ c_e$$

for every input $x_1, x_2, \ldots$ with the same probability. It even holds if the content of the random values of the parties in the extended collusion and if $\Pi_{\oplus_k r_{k,e_1},\oplus_k r_{k,e_1}}(\oplus_k \widehat{Q}_{k,g})$ are given. Hence, the adversary cannot deduce any information from C_1 and C_2. Since C_3 consists of ℓ bits Theorem 16 follows directly.

6 Dynamic *t*-Private Auctions

We call an electronic auction system dynamic if it allows a bidder to change his bid and if bidders can join a running auction. Introducing dynamism into a electronic auction system may lead to two scenarios:

1. A bidder gets feedback whether his bid changes the result of the auction. Hence, he gets additional information about the bids of the other bidders.

2. A bidder does not get any feedback. Then, dynamism can be reduced to the static case where only the last bid of each bidder is used.

Talking about a dynamic secure electronic auction system, we assume that the system works as follows:

1. The system generates an algorithm that allows to evaluate the auction without revealing any information about the bids. Our system will compute an encrypted result.

2. The bidders can deposit their bids in such a way that no party gets any information about the values of the bids (at this moment).

3. If a bidder changes his bid all other bidders can stay inactive.

4. If a bidder is paying some fee, he will get the information whether he is the winner of the auction at this moment.

5. A bidder can withdraw or change his bid such that no party gets any information about the value of his bid (at this moment).

Note that combining the last two requirements results in some leakage (one bit) of information about the bids of the remaining bidders. But, one can assume that the fee makes this bit of information very expensive.
Let us first investigate whether we can recycle a garbled table of a gate for evaluating a circuit on two different inputs. In general, we have to give a negative answer. For a Boolean circuit C and two binary inputs $x, x' \in \{0,1\}^n$ of C with $C(x) = C(x')$ let $\Delta(x, x')$ be the set of input variables with different values in x and x'. Furthermore, for a subset of input variables X let $\Lambda_C(x, X)$ be the set of gates g such that for some x, x' with $C(x) = C(x')$ and $\Delta(x, x') \subseteq X$ the value of g is different on x and x'.

Observation 19 *Let Γ and Γ' be two garbled circuits and X be a subset of input variables that may change their values. For every input pair $x, x' \in \{0,1\}^n$ with $C(x) = C(x')$ and $\Delta(x, x') \subseteq X$ the random strings W_g^0, W_g^1, r_g of all $g \in \Lambda_C(x, X)$ have to be chosen independently in Γ and Γ' to preserve privacy when evaluating Γ on x and Γ' on x'.*

If the random strings W_g^0, W_g^1, r_g are not chosen independently in Γ and in Γ' for all gates $g \in \Lambda_C(x, X)$, then one can observe whether the value of g changes from x to x'.
Let B_i be a bidder that has announced to change his bid, i.e., $X = \{x_{i,1}, x_{i,2}, \ldots\}$. Our goal is to find a protocol that allows all bidders $B_j \neq B_i$ to stay inactive. Let Γ be the garbled circuits used on the old bids, and let Γ' be the garbled circuits used on the changed bids. From our observation above one can conclude that all random values W_g^0, W_g^1, r_g in Γ and in Γ' with $g \in \bigcup_{x \in \{0,1\}^n} \Lambda_C(x, X)$ have to be chosen independently. Hence, we can run our protocols for all gates $g \in \Lambda_C(X) = \bigcup_{x \in \{0,1\}^n} \Lambda_C(x, X)$ and all wires leaving these gates. Since the values of the gates $g' \notin \Lambda_C(X)$ do not change their values for every input x and x' with $\Delta(x, x') \in X$ we can recycle the random values $W_{g'}^0, W_{g'}^1, r_{g'}$ from Γ in Γ'. Note that for these garbled circuits Γ and Γ' the intermediary strings $W_e^{b_e} \circ (b_e \oplus r_e)$ on the wires that are leaving gates $g' \notin \Lambda_C(X)$ are the same for every pair of input x and x' with $\Delta(x, x') \in X$. Hence, we only have to reevaluate the gates in $\Lambda_C(X)$. This can be done by running a new bidding process for B_i and running a new evaluation process of Γ' by the auctioneer A.

Theorem 20 *There exists a t-private protocol for a dynamic electronic auction system for $(t+1)^2$ or more parties using $O(dt^3|\Gamma|)$ random bits, and a t-private protocol for $2t+1$ or more parties using $O(d(t+1)^{t+3}|\Gamma|)$ random bits where d denotes the number of bid-changes.*

Proof: Assume that there are d bid changes that occur successively. In the following we fix the bid change. Let X_ℓ be the set of input gates that may change their values in the ℓth bid change. Since for every input x and x' with $\Delta(x, x') \in X_\ell$ the values of the gates $g' \notin \Lambda_C(x, X_\ell)$ do not change their values, we can recycle the random values $W^0_{g'}, W^1_{g'}, r_{g'}$ from Γ in the modified circuit Γ'. For all gates $g \in \Lambda_C(x, X_\ell)$ new garbled tables have to be constructed t–privately with respect to the recently generated strings W'_e and permutation bits $r'_{k,e}$. To reevaluate the circuit the bidders transfer their changed bids using the t–private bidding protocol. The tree-protocol and the slave-efficient-protocol can be modified in a similar way. Note that those bidders who do not change their bids remain inactive whenever a bid changes. The t-privacy of the protocols for bidding and for the construction of the modified garbled circuits (see Theorem 3, 5, 6, and 7) implies the t-privacy of the protocols for bid-changes.
We have to reconstruct $O(\sum_{\ell=1}^{d} |\Lambda(x, X_\ell)|)$ gates. In the worst case this results in $O(dt^3|\Gamma|)$ random bits for the field-protocol, respectively $O(d(t+1)^{t+3}|\Gamma|)$ random bits for the tree-protocol and for the slave-efficient-protocol. □
We can use a similar protocol to extend a garbled circuit for an auction from n to $n+1$ bidders, if the comparison of the bids can be described by an associative operator with a neutral element. In this case, we generate in the beginning a garbled circuit for the auction system with at least one dummy bidder that gives a neutral bid. If a new bidder joins the auction, we can simply add a subcircuit to the existing one for the new bidder and a new dummy bidder that replaces the an old dummy bidder with his neutral bid. To generate a resulting garbled circuit, one can run a protocol simular to our protocol for changing a bid of an existing bidder.
Note that if changing bids goes along with a change of the circuit, e.g., the new bid may require more bits than the old one, then our strategy above fails. In this case we have to construct a completely new garbled circuit Γ'. To translate the encrypted bids of the passive bidders $B_j \neq B_i$ from the previous encryption of Γ we can run a protocol similar to the bidding protocol. Then, for an input gate g of B_j the auctioneer (additionally) performs the role of the bidder. Furthermore, the auctioneer uses $r_g \oplus b_g$ instead of the input $x_{i,j}$. The bits $r_{k,g}$ of the auction issuers AI_k are replaced by $r_{k,g} \oplus r'_{k,g}$. So, we can translate the string $W^{b_g}_g \circ (b_g \oplus r_g)$ of Γ to the string $W'^{b_g}_g \circ (b_g \oplus r'_g)$ of Γ'. This strategy can also be used to extend a given garbled circuit from n to $n+1$ bidders.
The t-privacy of the protocols for bidding and for the construction of the modified garbled circuits (see Theorem 3, 5, 6, and 7) implies the t-privacy of the protocols for bid-changes and for extending a given garbled circuit. This implies:

Theorem 21 *There exists a t-private protocol for a dynamic electronic auction system for $(t+1)^2$ or more parties using $O(dt^3H)$ random bits and a t-private protocol for $(2t+1)$ or more parties using $O(d(t+1)^{t+3}H)$ random bits where d is the number of bid-changes and H is the size of the final garbled circuit.*

7 Conclusions

In this paper we present a t-private protocol for $2t + 1$ parties and a more randomness efficient t-private protocol for $O(t^2)$ parties for sealed bid auctions. Our protocols are based on garbled circuits for evaluating the auction, and on strategies for constructing such a circuit in a distributed way. Usually, the bidder-to-auctioneer connection is the bottleneck of an electronic auction system, e.g., the bidder has a limited bandwidth or he is not always available. Thus, we postulate that bidders have to be involved as infrequently as possible. In our system a bidder must only be online to bid and can stay passive during the remaining time. This feature also holds for dynamic auctions where bids are changed or bidders join a running auction. Finally, we analyze the information gain of Byzantine attackers on our electronic auction system. We present a strategy for introducing dynamism into such an electronic auction system.

References

[1] A. Beimel, *On Private Computation in Incomplete Networks*, 12th SIROCCO, pp. 18–33, 2005.

[2] M. Ben-Or, S. Goldwasser, A. Wigderson, *Completeness theorems for non-cryptographic fault-tolerant distributed computation*, 20th STOC, pp. 1–10, 1988.

[3] M. Bläser, A. Jakoby, M. Liśkiewicz, B. Siebert, *Private Computation – k-Connected versus 1-Connected Networks*, 22nd CRYPTO, pp. 194–209, 2002.

[4] F. Brandt, *Secure and Private Auctions without Auctioneers*, Technical Report FKI-245-02, Institut für Informatik, Technische Universität München, 2002.

[5] F. Brandt, *Fully Private Auctions in a Constant Number of Rounds*, 7th Annual Conference on Financial Cryptography (FC), pp. 223–238, 2003.

[6] C. Cachin, *Efficient Private Bidding and Auctions with an Oblivious Third Party*, 6th ACM Conference on Computer and Communications Security, pp. 120–127, 1999.

[7] D. Chaum, C. Crépeau, I. Damgård, *Multiparty unconditionally secure protocols*, 20th STOC, pp. 11–19, 1988.

[8] B. Chor, S. Goldwasser, S. Micali, B. Awerbuch, *Verifiable secret sharing and achieving simultaneity in the presence of faults*, 26th FOCS, pp. 383–395, 1985.

[9] B. Chor, E. Kushilevitz, *A zero-one law for boolean privacy*, SIAM J. Disc. Math., 4(1):36–47, 1991.

[10] K. Chui, R. Zwick, *Auction on the Internet - A Preliminary Study*, Manuscript, Technical report, Hong Kong University of Science and Technology, Department of Marketing, 1999.

[11] I. Damgård, Y. Ishai, *Constant-Round Multiparty Computation Using a Black-Box Pseudorandom Generator*, 25th CRYPTO, pp. 378–394, 2005.

[12] M. Franklin, M. Reiter, *The Design and Implementation of a Secure Auction Service*, IEEE Transactions on Software Engineering 22(5), pp. 302–312, 1996.

[13] M. Franklin, M. Yung, *Secure hypergraphs: Privacy from partial broadcast*, 27th STOC, pp. 36–44, 1995.

[14] O. Goldreich, S. Micali, A. Widgerson, *How to play any mental game or a completeness theorem for protocols with honest majority*, 19th STOC, pp. 218–229, 1987.

[15] M. Harkavy, H. Kikuchi, J. Tygar, *Electronic Auctions with Private Bids*, 3rd USENIX Workshop on Electronic Commerce, pp. 61–74, 1998.

[16] Y. Ishai, E. Kushilevitz, *Randomizing Polynomials: A New Representation with Application to Round-Efficient Secure Computation*, 41st FOCS, pp. 294–304, 2000.

[17] Y. Ishai, E. Kushilevitz, *Perfect Constant-Round Secure Computation via Perfect Randomizing Polynomials*, 29th ICALP, pp. 244–256, 2002.

[18] M. Jakobsson, A. Juels, *Mix and Match: Secure Function evaluation via Ciphertexts*, 6th ASIACRYPT, pp. 162–177, 2000.

[19] A. Juels, M. Szydlo, *A Two-Server, Sealed-Bid Auction Protocol*, 6th Annual Conference on Financial Cryptography (FC), pp. 72–86, 2002.

[20] H. Kikuchi, M. Harkavy, J. Tygar, *Multi-round Anonymous Auction Protocols*, 1st IEEE Workshop on Dependable and Real-Time E-Commerce Systems, pp. 62–69, 1998.

[21] E. Kushilevitz, S. Micali, R. Ostrovsky, *Reducibility and Completeness In Multi-Party Private Computations*, 35th FOCS, pp. 478–489, 1994.

[22] K. Kurosawa, W. Ogata, *Bit-Slice Auction Circuit*, 7th ESORICS, pp. 24–38, 2002.

[23] E. Kushilevitz, R. Ostrovsky, A. Rosén, *Characterizing linear size circuits in terms of privacy*, JCSS, 58(1):129–136, 1999.

[24] M. Naor, B. Pinkas, R. Sumner, *Privacy Preserving Auctions and Mechanism Design*, 1st ACM Conference on Electronic Commerce, pp. 129–139, 1999.

[25] K. Omote, A. Miyaji, *A Second-price Sealed-bid Auction with Verifiable Discriminant of p_0-th Root*, 6th Financial Cryptography Conference (FC), pp. 57–71, 2002.

[26] A. Shamir, *How to Share a Secret*, Communic. of the ACM 22(11), pp. 612–613, 1979.

[27] *A Mathematical Theory of Communication*, The Bell System Technical Journal, Vol. 27, pp. 379-423, 1948

[28] A. Sadeghi, M. Schunter, S. Steinbrecher, *Private Auctions with Multiple Rounds and Multiple Items*, 13th IEEE DEXA, pp. 423–427, 2002.

[29] P. Stechert, *Dynamic Private Auctions*, Diplomarbeit, Institut für Theoretische Infomatik, Universität zu Lübeck, Germany, January 2005.

[30] A. C. Yao, *Protocols for secure computations*, 23rd FOCS, pp. 160–164, 1982.

[31] A. C. Yao, *How to generate and exchange secrets*, 27th FOCS, pp. 162–167, 1986.