Electron Commer Res (2010) 10: 291–311
DOI 10.1007/s10660-010-9057-x

On server trust in private proxy auctions

Giovanni Di Crescenzo · Javier Herranz · Germán Sáez

Published online: 9 October 2010

Abstract We investigate *proxy auctions*, an auction model which is proving very successful for on-line businesses (e.g., http://www.ebay.com), where a trusted server manages bids from clients by continuously updating the current price of the item and the currently winning bid as well as keeping private the winning client's maximum bid.

We propose techniques for reducing the trust in the server by defining and achieving a security property, called *server integrity*. Informally, this property protects clients from a novel and large class of attacks from a corrupted server by allowing them to verify the correctness of updates to the current price and the currently winning bid. Our new auction scheme achieves server integrity and satisfies two important properties that are not enjoyed by previous work in the literature: it has minimal interaction, and only requires a single trusted server. The main ingredients of our scheme are two minimal-round implementations of zero-knowledge proofs for proving lower bounds on encrypted values: one based on discrete logarithms that is more efficient but uses the random oracle assumption, and another based on quadratic residuosity that only uses standard intractability assumptions but is less efficient.

G. Di Crescenzo: Part of this work done while visiting UPC, Spain.
J. Herranz and G. Sáez: Work partially supported by Spanish *MICINN Ministry*, project TSI2006-02731.

G. Di Crescenzo (✉)
Telcordia Technologies, Piscataway, NJ, USA
e-mail: giovanni@research.telcordia.com

J. Herranz · G. Sáez
Dept. Matemàtica Aplicada IV, Universitat Politècnica de Catalunya C. Jordi Girona, 1-3, Mòdul C3, Campus Nord, 08034 Barcelona, Spain

J. Herranz
e-mail: jherranz@ma4.upc.edu

G. Sáez
e-mail: german@ma4.upc.edu

Keywords Auctions · Privacy · Trust · Cryptography

1 Introduction

The overwhelming expansion of the Internet is today being accompanied by a large increase of financial activities and transactions that are conducted on-line. An example of notable success is represented by on-line auctions. A few minutes navigation on the Internet allows to realize the existence of several sites offering easy to implement auctions as a way for anybody to sell items of any kind to the best bidder. Different are the types of auction that are being offered by these businesses, but one in particular is becoming very popular: proxy auctions.

Typically, in an auction a server is managing the selling of some item and receiving bids from clients, eventually choosing one of these bids and the item's price according to some method. There are various types of auctions, varying in features, such as how the sale management is implemented (e.g., a human, a proxy computer server, or a paper sheet, can act like a server); how the bids are received (e.g., single or multiple, open or sealed bids that may be based or not on previous bids); and how the item's price is selected (e.g., English-style or Vickrey-style). For instance, in the most popular type of non-computerized auction, a human manages the item sale, multiple bids are openly offered and can depend on previous ones, and the price is determined in English-style fashion, meaning that it is set equal to the highest bid. Alternatively, in the currently most popular type of on-line auction, also called *proxy auction*, a server manages the item sale; a client can submit closed bids specifying the maximum price that he is willing to pay; the current price of the item is updated because of the various bids; the client offering the highest maximum price before the end of the auction eventually wins; and the price is determined in Vickrey-style fashion [35], meaning that it is set equal to a minimal increment amount above the second highest bid. More precisely, the system acts as an electronic proxy that repeatedly places bids for the clients (up to the bidder's specified maximum price) to keep them ahead of other bidders. Therefore, a bidder can only be outbid if someone else enters a greater maximum price.

On one hand, proxy auctions seem very attractive, especially for Internet users, since they can submit their maximum bid and then not care about how the auction goes until the end of the auction itself. Instead, in open-bid English-style auctions, users need to carefully listen how the auction goes and repeatedly submit bids to outbid other bidders.

On the other hand, proxy auctions put a significant amount of trust in servers. If servers are fully trusted then the auction winner and price are going to be fairly decided. However, if this is not the case, then both can be compromised. In the interest of maximizing the final selling price and therefore its commission fee, a server might both claim that one particular bidder outbid another one without this being the case, or decide to update the item price by an amount larger than what he is supposed to.

In this paper we present techniques for preventing these types of undesired behavior from corrupted servers in proxy auctions.

Our model, definition and results. We consider a model with several clients who intend to purchase an item by participating in a proxy auction, and a server, taking

care of various auction management operations, including setting starting and final date of the auction, notifying clients of bidding transactions, and updating current price and currently winning bid.

We present a first formal definition of some basic security properties that one would expect in this type of auction. In particular, we define security against clients preventing any other client to win the auction, and privacy against clients trying to obtain some information about either the current auction winner's or the final winner's offered maximum price. Both these properties are already achieved by many businesses on the Internet (e.g. [1]).

Most importantly, we focus on other security requirements not achieved by businesses on the Internet, such as server-integrity, also called security against the server. We formally define this property and propose a simple and efficient auction scheme that achieves this property without compromising the above mentioned privacy property. We do not even compromise the main efficiency property of these schemes: that is, their round complexity. In fact, we believe one important property of our auction scheme is that it preserves the number of messages of the original proxy auction scheme. Specifically, bidding requires a single message from client to server and updating current item value and currently winning bid also requires a single message from server to client, which is minimal. Another important property of our scheme is that it uses a *single server* to guarantee the correctness of the scheme, rather than many servers that cannot collude, as done previously in many works. Our investigation focused on enhancing the security of auction systems that are used in many businesses on the Internet. This should be contrasted with essentially all papers on auctions in the cryptographic literature that instead focus on designing elegant protocols with many interesting security properties but that unfortunately remain very far from protocols used in practice.

Crucial tools in the design of our auction scheme are two new techniques for 'lower-bound proof systems' (i.e., proof systems for proving that an encrypted value is larger than some known value) with efficient round complexity.

Our first technique assumes that bids are encrypted using known encryption schemes based on the intractability of computing discrete logarithms, and consists of interactive perfect zero-knowledge proof systems that requires only 3 messages (or can require a single message if we use the random oracle assumption), and time and communication complexity only linear (and with a low multiplicative constant) in the binary expansion m of the bid, despite the fact that the boolean circuit to express the lower bound has size quadratic in m. The previous best schemes in the literature [4, 7, 25] focus on minimizing the asymptotic dependencies of the communication complexity on m, sometimes achieving communication complexity constant as a function of m, but for large constants, and sometimes at the expense of the time complexity. We note that for our application of a private and server-secure proxy auction scheme, typically $m \leq 6$ suffices, and thus our scheme is preferable.

Our second technique assumes that bids are encrypted using known encryption schemes based on the intractability of computing quadratic residuosity modulo Blum integers, and consists of non-interactive perfect zero-knowledge proof systems (see, e.g. [5, 10, 11]). Here, we provide the first 1-message lower-bound proof system in the common random string model under standard intractability assumptions (i.e., without

using random oracles). This construction is less efficient than the previous one, but we pay special attention in designing it so that it has improved efficiency with respect to solutions that could be obtained by directly using general non-interactive zero-knowledge for $\mathcal{NP}$-complete languages [5, 18] or general composition results for languages having non-interactive perfect zero-knowledge proofs [11].

Related results. Several investigations have been done in the cryptographic literature on auctions (see, e.g., [6, 14, 22, 28, 30, 32, 33]), mostly dealing with sealed-bid auctions (i.e., auctions where each client submits a sealed bid, and all bids are simultaneously opened at the end of the auction). Some papers (see, e.g., [2, 3, 24, 26, 28]) dealt with Vickrey auctions, but all considered multi-server models to guarantee protocol correctness and required many rounds of interaction. Having even two servers has been recognized by the auction designer themselves (e.g., [26]) or in follow-up papers (e.g., [23, 28]) as a source of easy attacks to the auction scheme.

The problem of computing lower bounds (or range) proofs for encrypted data has been already studied, for instance, in [4, 7, 25, 27, 34]. However these and other previous papers provide either interactive solutions or non-interactive solutions that are proved to be secure under the random oracle assumption. Instead, in this paper we provide the first non-interactive construction that can be proved secure under conventional cryptographic assumptions (i.e., the hardness of deciding quadratic residuosity modulo Blum integers). Very recently, another non-interactive lower bound proof was given in [36] under the subgroup decision problem assumption.

The current paper is a revised and improved version of our previous extended abstract in [16].

2 Definition of secure proxy auctions

Setup: parties, items, connectivity. The parties involved in a secure proxy auction are a *server*, denoted as S, and the *clients*, denoted as $C_1, C_2, \ldots, C_n$. The server is managing the auctioning of a *single* item (for simplicity) and the clients are allowed to send to the server bids they would like to pay in order to buy the item. Server must be connected with all clients by means of authenticated channels; but connection between any two clients is not necessary for the auction to properly function.

Basic auction mechanics. At some *starting date sd* the server S announces on a public site (e.g., a website associated with the server) the auctioning of the item, and also defines a *starting price sp*, a *deadline date dd*, and a *minimal increment* Δ. The time of the auction goes then between *sd* and *dd*, where the following happens. First of all, the item is associated with *current price cp* and *currently winning bid cwb*. While at time *sd* it holds that $cwb = cp = sp$, we note that during the auction, both the current price and the currently winning bid may be modified. Moreover, the current price will always be publicly known. Right after time *sd*, each client C_i can send to the server a message bid_i specifying the *maximum price* mp_i that C_i is willing to pay for the item. If $mp_i \geq cp + \Delta$ then the message bid_i is considered *valid* by the server, who checks if $mp_i \geq cwb + \Delta$. If so, then client C_i will hold the currently winning bid; that is, the value of cp is updated to $cwb + \Delta$, and cwb is set equal to

mp_i. Otherwise, client C_i is sent a message by S saying that he has been outbid by another client, the current price cp is updated to $mp_i + \Delta$ and the value cwb remains unchanged. At deadline time dd, the client who submitted cwb is declared winner of the auction and is supposed to buy the item at price cp.

We remark that each client would typically go through a *registration phase* with the server in order to be able to take part in the auction, possibly involving some exchange of personal information. This phase can happen after the starting date of any particular action.

Protocols. A first protocol involved in the auction scheme is run in the registration phase between the server and a particular client. We will assume that this phase heavily depends on the specific application and that it can be realized in a quite straightforward way, and therefore we will only consider the auction phase from now on. This phase includes two protocols: the *bidding protocol* and the *price update protocol*. Although technically not necessary, for sake of efficiency, we will insist that in our solutions both protocols have minimal round complexity, in that a single message is sent by one party to the other. In particular, in the bidding protocol a client sends a single message to the server specifying her maximum price. In the price-update protocol, the server replies to the client's bid (for simplicity, we assume that this reply is publicly posted), and possibly updates the current price and the currently winning bid, where the reply contains a *bid transcript* certifying the client's bid and the server's price or winning bid updates.

Requirements. Let m, n denote positive integers and k be a security parameter. We will denote by sd the starting date, by dd the deadline date, by $\Delta \in \{0,1\}^m$ the minimal increment, by $cp \in \{0,1\}^m$ the current price, and by $cwb \in \{0,1\}^m$ the currently winning bid. We also let ℓ denote the index in $\{1, \ldots, n\}$ such that mp_ℓ is the maximum price submitted by any client during the auction. (Here, the values n, ℓ are not fixed at the starting date, but only after the deadline date.) If we denote by Π_b the bidding protocol and by Π_{pu} the price update protocol, an execution of a proxy auction scheme, denoted as $\Pi \equiv (\Pi_b, \Pi_{pu})$, can be written as a sequence $(\Gamma_1, \ldots, \Gamma_q)$, where each Γ_i is an ordered execution of protocol Π_b and protocol Π_{pu} between S and any one among clients $C_1, \ldots, C_n$.

Given these definitions, we require a *secure proxy auction scheme* $\Pi \equiv (\Pi_b, \Pi_{pu})$ to satisfy the following requirements:

Correctness. If S and all clients $C_1, \ldots, C_n$ honestly run all executions of protocols Π_b and Π_{pu}, then the probability that at the end of the auction scheme the client C_ℓ is declared winner is equal to 1.

Security against clients. If S honestly runs all executions of protocols Π_b and Π_{pu}, then for all algorithms $C'_1, \ldots, C'_{\ell-1}, C'_{\ell+1}, \ldots, C'_n$, the probability that at the end of the auction scheme Π the client C_ℓ is not declared winner is exponentially small (in k).

Privacy against clients. Let mp_i be the maximum price submitted using protocol Π_b from client C_i, at some time t_1 when the current price is cp, and let $t_2 > t_1$ be a time (if any) when the current price is updated to cp' as a result of a new bid with

maximum price $\leq mp_i$, and let $t_3 > t_2 > t_1$ be a time (if any) when the current price is updated to cp'' as a result of a new bid with maximum price $> mp_i$.
(Current winner privacy:) Assume that the client C_i is outbid. For all probabilistic polynomial time algorithms $C'_1, \ldots, C'_{i-1}, C'_{i+1}, \ldots, C'_n$, trying to guess the value of mp_i in the time interval $[t_1, t_2]$ (resp., $[t_2, t_3]$), the probability that they succeed better than randomly choosing among all Δ increments of interval $[cp, 2^m - 1]$ (resp., $[cp', 2^m - 1]$), is negligible.
(Final winner privacy:) Assume that client C_i is not outbid and becomes the winner of the auction. For all probabilistic polynomial time algorithms $C'_1, \ldots, C'_{i-1}, C'_{i+1}, \ldots, C'_n$, trying to guess the value of mp_i at any time after time t_2, the probability that they succeed better than randomly choosing among all Δ increments in interval $[cp, 2^m - 1]$, is negligible.

Security against the server. Assume client C_i and server S' run an execution of the bidding protocol Π_b and an execution of price update protocol Π_{pu}. Furthermore, assume that C_i submits a maximum price $mp_i \geq cp + \Delta$ during the execution of Π_b and that S' publicly posts a bid transcript at the end of the execution of Π_{pu}. For any such server S', there exists a probabilistic polynomial time algorithm J (for *judge*) such that the following holds:
(Update of currently winning bidder:) If $mp_i \geq cwb + \Delta$ and this execution of Π_b, Π_{pu} does not result in the current price and currently winning bid to be updated to $cwb + \Delta$ and mp_i, respectively, then the probability that algorithm J, on input C_i's view during the execution of protocols Π_b, Π_{pu}, does not return 1 is exponentially small in k.
(Update of current price:) If $mp_i < cwb + \Delta$ and this execution of Π_b, Π_{pu} does not result in the current price to be updated to $mp_i + \Delta$, then the probability that algorithm J, on input C_i's view during the execution of protocols Π_b, Π_{pu}, does not return 1 is exponentially small in k.

Remark. The above definition captures a novel and large class of somewhat 'innocent-looking' (and therefore, more dangerous) attacks from a corrupted server: for example, incorrect updates of the currently winning bidder (e.g., a server might try to favor one client over another, even though the latter client's bid was higher), and incorrect update of the current price (e.g., a server might try to claim that the second highest bid was higher than it really was so to increase the current price). We do not consider but plan to study in the future investigations more 'risky' (and thus, arguably, less likely) attacks from a server, such as denial of service to a particular client, or coalitions server-client that will favor one client over another during the auction.

3 Proving lower bounds on encrypted values

A major ingredient of our auction protocol is a type of proof system that will be crucial in guaranteeing our desired server integrity properties. Informally speaking, we would like to design proof systems that will be used by the server of a proxy auction to prove that all updates during the execution of the auction are being performed

according to the prescribed protocol. Such a proof would have to be executed upon any execution of the update protocol and should be verifiable by all auction members (that is, not only by the bidder that caused a particular update protocol, or not only by bidders for the same item). As we will clarify in Sect. 4, all such proofs can be considered as proofs of lower bounds on encrypted values. In the rest of this section, we formally define the statement to be proved, recall various types of interactive proof systems from the cryptography literature, including known proofs of lower bounds on encrypted values, and present 2 new solutions for such proof systems with desirable efficiency properties.

3.1 Lower-bounds proofs: definitions and preliminaries

We denote a public-key encryption scheme as a triple (KG, E, D) of (probabilistic) algorithms, formally described as follows: KG is the key-generation algorithm that, on input a security parameter (in unary) 1^k, returns a public key pk and a secret key sk; E is the encryption algorithm that, on input public key pk and a message d, returns a ciphertext c; and D is the decryption algorithm that, on input public key pk, secret key sk and a ciphertext c, returns a message d or a decryption failure symbol $\perp$. Given any public-key encryption scheme (KG, E, D), we define language

$$\mathrm{GT}_{(KG,E,D)} = \{(1^m; pk; c; t) \text{ s.t. } |D(sk, pk, c)| = m,\ |t| = m,\ D(sk, pk, c) > t\},$$

briefly denoted as GT, and investigate interactive proof systems for language GT.

Recall that an interactive proof system [21] for a language L is a 2-party protocol between two efficient interacting algorithms, called the prover P and the verifier V. On input a string x, P tries to convince V that $x \in L$, by exchanging messages, possibly computed using random bits, so that at the end of the communication, V returns: *accept* or *reject*, to communicate its decision about P's claim that $x \in L$. Formally speaking, an *interactive proof system* for language L is a pair (P, V) of efficient interacting algorithms that satisfies the following requirements (where $k_x = |x|$ denotes the length of the input x common to P and V). If $x \in L$ then the probability that V does not return *accept* at the end of the protocol is negligible in k_x (this is the *completeness* requirement). If $x \notin L$ then the probability that V returns *accept* at the end of the protocol is negligible in k_x (this is the *soundness* requirement). *Non-interactive proof systems* of [5] are similarly defined, but here, the prover and verifier share a public random string, called the *reference string*, that can be used to compute their messages. Thus, the probability in the completeness and soundness requirements is also over the random choice of the reference string.

In the rest of this section we investigate various types of interactive and non-interactive proof systems for language GT, also called *lower-bound proof systems*, with specific security and efficiency properties, which we now discuss.

Security properties. Zero-knowledge proof systems [21] are proof systems with the additional security property that no information is revealed to a possibly cheating verifier in addition to the fact that $x \in L$. Honest-verifier zero-knowledge proofs are a variant of zero-knowledge proof where this additional security property is only required to hold against verifiers that follow their protocol correctly. We note that

the study of zero-knowledge proof systems is a very active research sub-area in the cryptography and computational complexity research areas (see, e.g., [19]) in both the interactive and non-interactive model. In this paper we will design some lower-bound proof systems with zero-knowledge and honest-verifier zero-knowledge properties. Typically, designing a proof system with stronger security requirements comes with harder to achieve efficiency requirements.

Efficiency properties. Following the literature, we will consider the following efficiency metrics, associated to a proof system (P, V), where an execution of the proof system takes as input an k_x-bit string x, and the metrics are measured as a function of k_x: the *round complexity*, that is, the number of messages exchanged by P and V; the *communication complexity*, that is, the sum of the lengths of all messages exchanged by P and V; and the *time complexity*, that is, the maximum running time of P and V.

Although a natural goal is that of minimizing all three efficiency metrics, we will especially focus on decreasing the round and communication complexity, which seems the most sensitive metric in a typical execution of a proxy auction (as later explained in more detail). In particular, for $\tilde{n} = 1, 2$, we will target *$\tilde{n}$-message* lower-bound proof systems, consisting of a message sent by V (empty in the case $\tilde{n} = 1$) followed by a message sent by P, as these proof systems will not increase the round complexity of a proxy auction protocol (as later explained in more detail).

Rewriting language GT. For any $t \in \{0,1\}^m$, and for any ciphertext c computed using the encryption algorithm E on input public key pk and an m-bit message d, we denote as $t_1 \circ \cdots \circ t_m$ the binary representation of t, and as $d_1 \circ \cdots \circ d_m$ the binary representation of $d = D(pk, sk, c)$, assuming that the latter is different from $\perp$. In other words, $t = t_1 2^{m-1} + t_2 2^{m-2} + \cdots + t_{m-1} 2 + t_m$, and $d = d_1 2^{m-1} + d_2 2^{m-2} + \cdots + d_{m-1} 2 + d_m$. Also, let $\vee, \wedge, \oplus$ denote the logical boolean OR, AND, XOR operators, respectively. Then the statement $D(sk, pk, c) > t$ in the definition of language GT is equivalent to the following one:

$$\Phi_0(d,t) = \bigvee_{j=1}^{m} \left[(d_j = 1) \wedge (t_j = 0) \wedge \left(\bigwedge_{i=1}^{j-1} \overline{d_i \oplus t_i} \right) \right].$$

In both of our constructions of lower bound proof systems, for different reasons, it will be convenient to use a rewriting of the above formula Φ_0. Specifically, we note that the negation $\overline{\Phi_0}$ of Φ_0, which represents the statement $D(sk, pk, c) \leq t$, is equivalent to

$$\left(\bigwedge_{i=1}^{m} \overline{d_i \oplus t_i} \right) \vee \bigvee_{j=1}^{m} \left[(d_j = 0) \wedge (t_j = 1) \wedge \left(\bigwedge_{i=1}^{j-1} \overline{d_i \oplus t_i} \right) \right].$$

Therefore, if we define $t_{m+1} = 1$ and $d_{m+1} = 0$, we have that an alternative way of writing Φ_0 is

$$\Phi_1(d,t) = \overline{\overline{\Phi_0}} = \bigwedge_{j=1}^{m+1} \left[\overline{t}_j \vee d_j \vee \left(\bigvee_{i=1}^{j-1} t_j \oplus d_j \right) \right].$$

Note that both formulae Φ_0, Φ_1 have size quadratic in m.

Equality proofs. Our auction scheme will also need *equality proof systems*, denoting proof systems for the language

$$\mathrm{EQ}_{(KG,E,D)} = \{(1^m; pk; c; t) \text{ s.t. } |D(sk, pk, c)| = m,\ |t| = m,\ D(sk, pk, c) = t\},$$

briefly denoted as EQ, where (KG, E, D) denotes an arbitrary public-key encryption scheme. In our constructions we will use encryption schemes for which it is simple to exhibit 1-message equality proof systems.

3.2 Lower-bound and equality proofs under the random oracle assumption

We define an encryption scheme and a lower-bound proof system for the associated language GT, based on an arbitrary public-key encryption scheme, an arbitrary pseudo-random generator, and the hardness of the discrete logarithm problem. The resulting lower-bound proof system requires 3 messages, but can be transformed into a 1-message lower-bound proof system using the random oracle assumption. The communication and time complexity of the proof system is only linear in the value m. Specifically, we obtain the following

Theorem 1 *Assuming the existence of both secure public-key encryption schemes and secure pseudo-random generators, and assuming the hardness of the discrete logarithm problem, there exists (constructively) a 3-message, honest-verifier zero-knowledge, lower-bound proof system with communication and time complexity linear in m, where m is the length of the first input to the proof system. By further making the random oracle assumption, this system can be transformed into a 1-message zero-knowledge lower-bound proof system.*

The rest of this subsection is devoted to proving Theorem 1.

One main idea in our construction is that of designing an encryption scheme (based on the commitment scheme from [29]) having the following special homomorphic property: the product of two ciphertexts each encrypting a single bit results in an encryption of the OR of the encrypted bits. This, together with the rewriting of language GT by using formula Φ_1, and using the efficient proof of knowledge from [31], allows us to construct a 3-message lower-bound proof system associated with the above encryption scheme, having time and communication complexity only linear in m (despite the size of Φ_0, Φ_1 being quadratic in m). Rewriting formula Φ_0 in its equivalent form Φ_1 is crucial to be able to apply the homomorphic properties of our encryption scheme.

We start the proof of Theorem 1 by recalling basic definitions of the tools and assumptions used and then present our construction.

Tools and assumptions. We use one arbitrary public-key encryption scheme (aKG, aE, aD), where aKG is the key generation algorithm, aE is the encryption algorithm and aD is the decryption algorithm.

We also use an arbitrary pseudo-random generator (see, e.g., [19]). Recall that a pseudo-random generator G is a function family, where each element g_w maps a w-bit input, called the *seed*, to a $p(w)$-bit output, for some given arbitrary polynomial p,

such that if the w-bit seed is uniformly distributed, then the output is computationally indistinguishable from the uniform distribution over $p(w)$ bits.

Let G be a group of order q, for some large prime $q > 2^k$, let $\langle G \rangle$ denote its (short) description, let $\mathbb{Z}_q = \{0, \ldots, q-1\}$, and assume that G is a multiplicative group.

Let p, q be primes such that $p = 2q + 1$ and let G_q denote the only subgroup of $Z_p^\star$ of order q. We note that it can be efficiently decided whether an integer a is in G_q, by checking that $a^q \equiv 1 \bmod p$. Moreover, any element of G_q different from 1 generates such a subgroup. For any $y, g \in G_q$, if $g \neq 1$ the *discrete logarithm of y in base g* is the element $x \in \mathbb{Z}_q$ such that $g^x = y \bmod p$. The *discrete logarithm assumption* says that for any polynomial-time algorithm A, the probability that A, on input (p, q, g, y), returns the discrete logarithm x of y, in base g, where g generates $\mathbb{Z}_p$, is negligible in k.

The random oracle assumption postulates the existence of a (black-box) function H that maps a given input to a uniformly distributed output. It facilitates the design of cryptographic protocols in many ways, including by reducing a protocol's round complexity, as first suggested in [17], and done here as well. Random oracles do not exist in real life (see [9], for example) and thus, although constructions based on this assumption are often used in practice, the associated security properties should be relied upon with appropriate caution.

Our 3-message construction. Our public-key encryption scheme (KG, E, D) is defined as follows. On input 1^k, algorithm KG runs algorithm aKG on input 1^k and obtains a pair (apk, ask). Then KG generates k-bit primes p, q such that $p = 2q + 1$ and a generator g for group G_q defined above, randomly chooses an integer $x \in \mathbb{Z}_q$ and computes $h = g^x \bmod p$. Finally, KG sets $pk = (apk, p, q, g, h, G)$ and $sk = ask$ and returns (pk, sk).

On input public key pk and m-bit message d, algorithm E does the following. First, it writes d in its binary expansion $d_1 \circ \cdots \circ d_m$. Then it randomly chooses two seeds s_1, s_2 for pseudo-random generator G and computes two mk-bit pseudo-random strings $(a_1, \ldots, a_m) = G(s_1)$, and $(b_1, \ldots, b_m) = G(s_2)$, where $a_i, b_i \in \mathbb{Z}_q$, for $i = 1, \ldots, m$. The pseudo-random strings are used to compute commitment values $z_i = g^{a_i} h^{d_i \cdot b_i} \bmod p$, for $i = 1, \ldots, m$, and the seeds s_1, s_2 are encrypted as $c_0 = aE(apk, (s_1, s_2))$. Finally, algorithm E returns ciphertext $c = (c_0, z_1, \ldots, z_m)$.

On input public key pk, secret key sk and ciphertext $c = (c_0, z_1, \ldots, z_m)$, algorithm D does the following. First, it decrypts c_0 by computing $(s_1, s_2) = aD(apk, ask, c_0)$ and uses the pseudo-random generator G to compute the two mk-bit pseudo-random strings $(a_1, \ldots, a_m) = G(s_1)$, and $(b_1, \ldots, b_m) = G(s_2)$. Then, for $i = 1, \ldots, m$ algorithm D sets $d_i = 1$ if $z_i = g^{a_i} h^{b_i} \bmod p$, or $d_i = 0$ if $z_i = g^{a_i} \bmod p$. Finally, D returns plaintext $d = d_1 \circ \cdots \circ d_m$.

Given the above cryptosystem (KG, E, D), we can define language

$$\mathrm{GT}_{(KG,E,D)} = \{(1^m; pk; c; t) \text{ s.t. } |D(sk, pk, c)| = m,\ |t| = m,\ D(sk, pk, c) > t\},$$

briefly referred as GT, where $pk = (apk, p, q, g, h, G)$, $c = (c_0, z_1, \ldots, z_m)$, and $sk = ask$. Because the conditions $|D(sk, pk, c)| = m$ and $|t| = m$ are satisfied for the above scheme (KG, E, D), to construct a lower bound proof system for GT, we only need to construct a proof system for proving statement $D(sk, pk, c) > t$. Our approach to

do that is to consider a boolean circuit, on input $d_1, \ldots, d_m, t_1, \ldots, t_m$, describing this statement. Specifically, we use the circuit $\Phi_1(d, t)$ defined in Sect. 3.1, and give a proof system that this circuit is satisfied when $d = D(sk, pk, c)$ is only known to the prover and is not revealed to the verifier. We show how to construct the latter proof bottom-up: starting by a proof system for an atomic statement '$d_i = 1$' to more composed statements, until we obtain a proof system for statement '$\Phi_1(d, t) = 1$'.

Proving statement '$d_i = 0$'. First of all, we note that P, who knows the plaintext d, does not know the discrete logarithm of h in base g, and thus, by construction of commitment key z_i, it holds that when $d_i = 0$ (resp., $d_1 = 1$) P knows (resp. does not know) the discrete logarithm a_i of z_i modulo g. Thus, to construct a proof system that '$d_i = 0$', we can use the 3-message proof of knowledge of the discrete logarithm of z_i modulo p in base g from [31]. Briefly speaking, this proof system goes as follows: the prover randomly chooses $r \in \mathbb{Z}_q$, computes $R = g^r \bmod p$ and sends R to the verifier; the verifier replies by sending a random challenge $\tilde{c} \in \mathbb{Z}_q$ to the prover; finally the prover sends $s = r + a_i \cdot \tilde{c} \bmod q$ to the verifier, who returns: *accept* if $g^s = R y^{\tilde{c}} \bmod p$ or *reject* otherwise. This proof system satisfies completeness, soundness under the discrete logarithm assumption (as a prover correctly answering two challenges $\tilde{c}, \tilde{c}'$ from the verifier can be used to compute the discrete logarithm of h) and honest-verifier zero-knowledge (as a simulator knowing the verifier's challenge $\tilde{c}$ can perfectly simulate both messages R, s from the prover by randomly choosing $s \in \mathbb{Z}_q$ and computing $R = g^s y^{-\tilde{c}} \bmod q$).

Proving statement '$d_i = 1$'. Using an analogous argument, we have now that when $d_i = 1$ (resp., $d_1 = 0$) P knows (resp. does not know) the discrete logarithm of $z_i h^{-b_i}$ modulo g. Thus, to construct a proof system that '$d_i = 1$', we can use the same 3-message proof of knowledge of the discrete logarithm of $z_i h^{-b_i}$ in base g from [31], if we additionally require that the prover sends seed s_2 in his first message, which can be used by the verifier to compute b_i.

Proving statement '$\vee_{j=1}^{i-1}((d_j = 1) \oplus (t_j = 1))$'. We note that all bits t_j are known to the verifier, and thus each statement '$(d_j = 1) \oplus (t_j = 1)$' is either of the form '$d_j = 0$' (when $t_j = 1$) or of the form '$d_j = 1$' (when $t_j = 0$). For $b = 0, 1$, let $Set_b \subseteq \{1, \ldots, i-1\}$ be the set of indices j such that $t_j = b$. We note that when the statement '$\vee_{j=1}^{i-1}((d_j = 1) \oplus (t_j = 1))$' is true (resp., false), then P knows (resp. does not know) the discrete logarithm of $(\prod_{j=1}^{i-1} z_j) \cdot h^{-u}$, where $u = \sum_{j \in Set_0} b_j$. Thus, to construct a proof system for this statement, we can use the 3-message proof of knowledge of the discrete logarithm of $(\prod_{j=1}^{i-1} z_j) \cdot h^{-u}$ in base g from [31], if we additionally require that the prover sends s_2 in his first message, which can be used by the verifier to compute all b_j values when $j \in Set_0$. As for the previous two proof systems, this proof system satisfies completeness, soundness under the discrete logarithm assumption and honest-verifier zero-knowledge.

Proving statement '$\Phi_1(d, t) = 1$'. We can write $\Phi_1(d, t)$ as $\wedge_{i=1}^{m+1} T_i$, where $T_i =$ '$((t_i = 0) \vee (d_i = 1) \vee_{j=1}^{i-1}((d_j = 1) \oplus (t_j = 1)))$'. Since all bits t_j are known to the verifier, this can be further simplified as $\Phi_1(d, t) = \wedge_{i:t_i=1} U_i$, where $U_i =$ '$((d_i = 1) \vee_{j=1}^{i-1}((d_j = 1) \oplus (t_j = 1)))$'. Note that a proof system for each statement U_i can be obtained by the 'OR-composition' technique from [8] (see [13] for a revised and more

accurate statement of their results) and [12] on the previously discussed protocol to prove statement '$d_i = 1$' and the previously discussed protocol to prove statement '$\vee_{j=1}^{i-1}((d_j = 1) \oplus (t_j = 1))$'. This composition results in a 3-message proof system for U_i where the verifier sends a single challenge, satisfying completeness, soundness (under the discrete logarithm assumption), and honest-verifier zero-knowledge. All these properties are directly implied from the results in [8, 12, 13] about the OR-composition technique.

Moreover, we note that proving statement $\Phi_1(d, t)$ only requires proving up to $m + 1$ U_i statements. This can be done using the 'AND-composition' technique from [8, 12] on the just discussed protocol to prove statement U_i. This composition results in a 3-message proof system for proving statement $\Phi_1(d, t)$ where the verifier sends a single challenge, satisfying completeness, soundness (under the discrete logarithm assumption), and honest-verifier zero-knowledge. All these properties are directly implied from the results in [8, 12, 13] about the AND-composition technique.

Finally, we note that the resulting protocol is a 3-message, honest-verifier zero-knowledge lower-bound proof system for language GT.

The 1-message construction. These proof systems can be made non-interactive using the Fiat-Shamir heuristic [17] of computing the verifier's challenge as the output of a hash on the input and the proof system's first message. Under the random-oracle assumption, the resulting protocol is a 1-message protocol maintaining the completeness and soundness properties of the corresponding 3-message protocol, and further satisfying zero-knowledge property.

Time and communication complexity. We stress that the described protocol has time and communication complexity comparable to at most m executions of the protocol from [31] (despite the formula for $\Phi_1(d, t)$ having size quadratic in m).

Equality proof system. We note that a 1-message equality proof system for the above proof system (KG, E, D) is obtained as follows. The prover simply sends both seeds s_1, s_2 to the verifier. The latter computes all values a_i, b_i, for $i = 1, \ldots, m$, and returns *accept* if and only if $z_i = g^{a_i} h^{t_i \cdot b_i} \bmod p$, for $i = 1, \ldots, m$.

3.3 Lower-bound and equality proofs under standard intractability assumptions

We define an encryption scheme and a lower-bound proof system for the associated language GT, based on an arbitrary public-key encryption scheme and the intractability of quadratic residuosity modulo Blum-Williams (BW) integers. The resulting lower-bound proof system requires 1 message in the non-interactive model (i.e., in the presence of a common random string), where it is zero-knowledge. The communication and time complexity of the proof system are, respectively, quadratic and cubic in the value m, defined as the length of the first input to the proof system. Specifically, we obtain the following

Theorem 2 *Assuming the existence of public-key encryption schemes, and the hardness of deciding quadratic residuosity modulo BW integers, there exists, constructively, a 1-message zero-knowledge lower-bound proof system in the non-interactive model with communication complexity quadratic in m and time complexity cubic in m.*

We note that this solution avoids using random oracles, but is not as efficient in terms of communication and time complexity. In practice, however, an auction protocol does not require the greatest time efficiency from a lower-bound proof system since the verification of such proofs can be performed off-line, i.e., at the end of the auction.

The rest of this subsection is devoted to proving Theorem 2. We note that every language in $\mathcal{NP}$ has a non-interactive *computational* zero-knowledge proof system under the hardness of deciding quadratic residuosity [5] or even more general complexity assumptions [18]. However, a lower-bound proof system based on these proof systems would have a much less efficient communication and time complexity than the one claimed in the theorem. Naturally, we consider non-interactive *perfect* zero-knowledge proof systems which are, however, only known for a relatively small class of languages, including formula compositions over quadratic residuosity statements [10, 11, 15]. Using some of these schemes, we can achieve a scheme with the performance claimed in the theorem. Rewriting formula Φ_0 in its equivalent form Φ_1 is crucial to be able to apply the known results from [15].

We start the proof of Theorem 2 by recalling basic definitions of the tools and assumptions used and then present our construction.

Tools and assumptions. For each integer x, we say that $z \in \mathbb{Z}_x^*$ is a *quadratic residue modulo* x if there exists an integer $r \in \mathbb{Z}_x^*$ such that $r^2 \equiv z \bmod x$. If such r does not exist, we say that z is a *quadratic non residue modulo* x. The quadratic residuosity predicate of an integer $y \in \mathbb{Z}_x^*$ can be defined as $\mathbb{Q}_x(y) = 0$ if y is a quadratic residue modulo x and 1 otherwise. A Blum-Williams (BW) integer is an integer x that is product of two primes $\equiv 3 \bmod 4$ and enjoys the following properties: the set $\mathbb{Z}_x^*$ can be partitioned into 2 equal-size sets $\mathbb{Z}_x^{+1}, \mathbb{Z}_x^{-1}$, where $\mathbb{Z}_x^c$ denoted the subset of integers from $\mathbb{Z}_x^*$ having Jacobi symbol c; the set $\mathbb{Z}_x^{+1}$ can be partitioned into 2 equal-size sets QR_x, QNR_x, denoting the sets of integers from $\mathbb{Z}_x^{+1}$ that are quadratic residues and quadratic non residues modulo x, respectively; the integer -1 is in QNR_x and the product of two elements in QNR_x is in QR_x. The problem of deciding quadratic residuosity modulo BW integers is the problem of returning $\mathbb{Q}_x(y)$ for a given $y \in \mathbb{Z}_x^{+1}$, and the assumption that this problem is hard, called the *quadratic residuosity assumption*, has been used as a base for cryptographic protocols (starting with [20]).

A 1-message construction in the non-interactive model. Our public-key encryption scheme (KG, E, D) is defined as follows. On input 1^k, algorithm KG generates k-bit primes $p, q \equiv 3 \bmod 4$, computes $x = pq$, sets $pk = x$, $sk = (p, q)$ and returns (pk, sk).

On input public key pk and m-bit message d, algorithm E does the following. First, it writes d in its binary expansion $d_1 \circ \cdots \circ d_m$. Then, for $i = 1, \ldots, m$, it randomly chooses $r_i \in \mathbb{Z}_x^*$ and computes $z_i = (-1)^{d_i} r_i^2 \bmod x$. Finally, algorithm E returns ciphertext $c = (z_1, \ldots, z_m)$.

On input public key pk, secret key sk and ciphertext $c = (z_1, \ldots, z_m)$, algorithm D does the following. For $i = 1, \ldots, m$ algorithm D sets $d_i = 1$ if z_i is a quadratic non-residue modulo x, or $d_i = 0$ if z_i is a quadratic residue modulo x. In other words, $d_i = \mathbb{Q}_x(z_i)$. Finally, D returns plaintext $d = d_1 \circ \cdots \circ d_m$.

Given the above cryptosystem (KG, E, D), we can define language

$$\mathrm{GT}_{(KG,E,D)} = \{(1^m; pk; c; t) \text{ s.t. } |D(sk, pk, c)| = m,\ |t| = m,\ D(sk, pk, c) > t\},$$

briefly referred as GT, where $pk = x$, $c = (z_1, \ldots, z_m)$, and $sk = (p, q)$. Because the conditions $|D(sk, pk, c)| = m$ and $|t| = m$ are satisfied for the above scheme (KG, E, D), to construct a lower bound proof system for GT, we only need to construct a proof system for proving statement $D(sk, pk, c) > t$. As before, we use the circuit $\Phi_1(d, t)$ defined in the Sect. 3.1, and give a proof system that this circuit is satisfied when $d = D(sk, pk, c)$ is only known to the prover and is not revealed to the verifier.

Recall that we can write $\Phi_1(d, t)$ as $\wedge_{i:t_i=1} U_i$, where $U_i =$ '$((d_i = 1) \vee_{j=1}^{i-1} ((d_j = 1) \oplus (t_j = 1)))$'. First of all, we note that each of the substatements '$(d_j = 1) \oplus (t_j = 1)$' can be proved in non-interactive perfect zero-knowledge, as follows. Since $d_j = \mathbb{Q}_x(z_j)$ and the value of t_j is known, to prove that '$(\mathbb{Q}_x(z_j) = 1) \oplus (t_j = 1)$', it is enough to prove a single quadratic residuosity statement; that is, '$\mathbb{Q}_x((-1)^{t_j} z_j \bmod x) = 1$'. Then, each formula U_i can be seen as an OR of i quadratic non residuosity statement, which we know how to prove in non-interactive perfect zero-knowledge [15], and the AND of all formulae U_i can therefore be proved in non-interactive perfect zero-knowledge [5], in the presence of a common random string. If we let (A, B) be the non-interactive perfect zero-knowledge proof system for proving the OR of two or more quadratic non-residuosity statements (modulo a BW integer) given in [15], then, given $x, z_1, \ldots, z_m, t_1, \ldots, t_m$, our final protocol can be seen as the sequential and independent composition of proof system (A, B) on input $(x; z_1, (-1)^{1-t_1})$; and then, for $i = 2, \ldots, m+1$, of proof system (A, B) on input $(x; z_i, (-1)^{1-t_i}, (-1)^{t_1} z_1, \ldots, (-1)^{t_{i-1}} z_{i-1})$.

Equality proof system. We note that a 1-message equality proof system for the above scheme (KG, E, D) is obtained as follows. The prover sends $r_1, \ldots, r_m$ to the verifier. The latter returns *accept* if and only if $z_i = (-1)^{t_i} r_i^2 \bmod x$, for $i = 1, \ldots, m$.

4 A proxy auction scheme with server-integrity

In this section we present a 1-server proxy auction scheme which enjoys the server-integrity property. Using any of the two variants of the lower-bound proof systems in Sect. 3, we obtain a different variant of a 1-server proxy auction scheme. One variant admits more efficient proofs of server integrity, but assumes the existence of random oracles. Another variant admits less efficient proofs of server integrity but is based on a standard cryptographic hardness assumption.

First of all we describe a proxy auction scheme based on an arbitrary lower-bound proof system. Then we show that the described scheme satisfies the requirements of the definition in Sect. 2. The basic ideas underlying the proxy auction scheme consist of the server proving, using 1-message zero-knowledge proofs, that he is honestly running the price update protocol. In particular, there are two types of updates the server could be doing when receiving a new bid: updating the winner name and the item price. While updating the item price, the server considers the previously winning bid and the new bid, and reveals the decryption of the smaller one. This directly sets the current item price equal to the revealed bid plus a minimal increment. However, this also directly declares the currently winning client as either the previously winning client or the client who sent the latest bid. In order to prove that this decision was

made correctly, the server will have to prove that one encryption of a bid is larger than the revealed smaller bid. All these proofs are required to be zero-knowledge so to preserve the privacy of the currently winning maximum price as it could be the auction winner's maximum price. Moreover, they are required to be realizable in at most 2 messages so to preserve the round-complexity of the auction scheme. The realization of these ideas will make crucial use of the 1-message zero-knowledge lower-bound and equality proof systems that we have constructed in Sect. 3. We divide the formal description into two phases: a setup phase and an auction phase.

Setup phase. The server S chooses a security parameter 1^k, a price parameter 1^m, a minimal increment $\Delta \in \{0,1\}^m$, a starting date sd, a deadline date dd, some item data id and a starting price sp. Then he sets current price $cp = sp$, currently winning bid $cwb = cp$. The server generates pair $(pk, sk) = KG(1^k)$ and randomly chooses a tuple $\vec{y} = (y_1, \ldots, y_m)$ as a valid encryption of cp using algorithm E and public key pk. Thus, the currently winning encrypted bid $cweb$ is set to $\vec{y} = (y_1, \ldots, y_m)$. Finally the server posts on the public site the tuple $(1^k, 1^m, sd, dd, id, cp, \Delta, pk, (y_1, \ldots, y_m))$, and keeps (sk, cwb) secret.

Auction phase. At any time between sd and dd a client C_i can decide to register to participate in the auction of item id. The details of this step are inessential for the rest of the scheme. Once registered, client C_i can run the bidding protocol Π_b to bid some maximum price $mp_i \in \{0,1\}^m$ she would be interested in paying for id. Protocol Π_b goes as follows:

1. C_i writes mp_i in binary, as $d_1 \circ \cdots \circ d_m$.
2. C_i uses E to compute a ciphertext eb as encrypted bid of $d_1 \circ \cdots \circ d_m$.
3. C_i generates the first message mes_1 for the lower-bound proof.
4. C_i sends (eb, mes_1) to S, in a time-stamped and authenticated way.

Digital signatures can be used to send the bid message in an authenticated way. This avoids the possibility that a dishonest server claims that some client has sent a bid (for example, a very high one) that she has not really sent.

Upon receiving (eb, mes_1) from a client C_i, the server S runs protocol Π_{pu} to eventually update the current price cp, the currently winning bid cwb and the currently winning encrypted bid $cweb$. Protocol Π_{pu} goes as follows:

1. Using sk, server S computes the decryption $(d_1, \ldots, d_m)$ of the encrypted bid eb.
2. Let mp_i be the integer whose binary expression is $d_1 \circ \cdots \circ d_m$.
3. If $mp_i \leq cp$, then
   - S computes a 1-message equality proof π_{eb} that $(1^m, pk, eb, t) \in EQ$, for $t = mp_i$;
   - S posts on a public site a bid transcript containing signed transcripts of the received bid (eb, mes_1), the equality proof π_{eb}, and an "invalid" message.
4. If $mp_i \geq cwb + \Delta$ then:
   - S computes a 1-message equality proof π_{cweb} that $(1^m, pk, cweb, t) \in EQ$, for $t = cwb$ (thus, proving that the currently winning encrypted bid $cweb$ is an encryption of cwb);

- S computes the second message of the lower-bound proof system to prove that $(1^m, pk, eb, t) \in GT$, for $t = cwb + \Delta$;
- S updates the current winning encrypted bid $cweb = eb$, the current price $cp = cwb + \Delta$ and the current winning bid $cwb = mp_i$;
- S posts on a public site the bid transcript containing signed transcripts of the received bid (eb, mes_1), the equality proof π_{cweb}, the lower-bound proof, the updated currently winning encrypted bid $\vec{y}$ and the current price cp.

Else ($cp + \Delta \leq mp_i < cwb + \Delta$):

- S computes a 1-message equality proof π_{eb} that $(1^m, pk, eb, t) \in EQ$, for $t = mp_i$ (thus, proving that the received encrypted bid eb is an encryption of mp_i);
- S computes the second message of the lower-bound proof system to prove that $(1^m, pk, cweb, t) \in GT$, for $t = mp_i - \Delta$;
- S updates the current price $cp = mp_i + \Delta$;
- S posts on a public site the bid transcript containing signed transcripts of the received bid (eb, mes_1), the equality proof π_{eb}, the lower-bound proof, and the current price cp.

Remark on public verifiability. We have designed the scheme so that the server's honest behavior is *publicly verifiable*; that is, all parties can verify the correctness of the server's updates of both the current price and the currently winning encrypted bid. If this property is not required, minor modifications are needed; for instance, rather than posting all proofs, the server will send proofs only to the involved client in each bidding operation.

4.1 Properties of the scheme

Theorem 3 *Assuming the existence of the* 1*-message zero-knowledge lower-bound proof system and equality proof system, the* 1*-server secure proxy auction scheme explained above satisfies the following properties*:

1. *the scheme preserves the round complexity of the proxy auction scheme*;
2. *the "privacy against clients" property holds, thanks to the zero-knowledge property of the lower-bound proof system used.*
3. *the "security against clients" property holds unconditionally*;
4. *the "security against server" property holds, thanks to the soundness property of both types of proof systems used.*

Proof First of all we stress that the scheme preserves the round-complexity of the proxy auction scheme. The bidding protocol only consists of a single message from a client to the server (which is anyway required by the class of proxy auction protocols that we consider, as the client needs to send his bid to the server). The price update protocol only consists of a single message from the server to a client (which is anyway required by the proxy auction protocol, as the server needs to reply to the client's bid).

We continue now by showing that the above scheme satisfies the requirements as in the definition in Sect. 2.

Correctness. This property is easy to verify. In particular, note that the server, given the secret key sk, can always compute the maximum price sent by any client in their encrypted bid, and therefore properly update the currently winning bid to the maximum price obtained from any client. We also recall that our two encryption schemes defined in Sect. 3 admit a lower-bound proof system and then note that they admit a 1-message equality proof system for proving the value of an encrypted bid: in both cases, indeed, the decryption algorithm can compute witnesses that the decryption of a certain value is valid (specifically, the seeds s_1, s_2 in the case of the scheme in Sect. 3.2, and the square roots r_i of the values $(-1)^{t_i} z_i \bmod x$, for $i = 1, \ldots, m$, for the scheme in Sect. 3.3).

Security against clients. This property is almost automatically derived from the scheme, since possibly dishonest players have no way to prevent a honest player who bids the maximum price to actually win the auction. This is achieved thanks to the fact that the server is honestly following her instructions.

Privacy against clients. We need to prove the two parts of this requirement: the current winner privacy and the final winner privacy. We start with the latter, as the former is an extension of it.

To show that final winner privacy is satisfied, we argue that a bidder who is never outbid by any other client can keep her maximum price "sufficiently" private (that is, private among all possible values of a bid that are greater than the final price of the auction item, that is required to be public by the auction rule.) It is clear from the development of the auction that the bidder's maximum price must be larger than the previously winning bid (or otherwise this client would have been outbid). However, we now show that no other information is revealed to the other clients about the value of this maximum price. First of all we note that each bidder only sends an encryption of her maximum value, without never revealing it in clear (and so is the server doing since she is never required to do so and she is assumed to be honest). Furthermore, the only other steps in the scheme that depend on this value are those from the server who must prove that this value is larger than some other maximum price that is submitted (specifically, the 'else' part of step 4 in protocol Π_{pu}). Since this proof is zero-knowledge, it does not allow an efficient algorithm to compute the bidder's maximum value better than without access to the lower-bound proof. However, with no access to this proof, the only information related to this value is its encryption and thus the only successful strategy to guess some information about the value of this maximum price is to break its encryption, which is infeasible by the security property of encryption.

The proof that the current winner privacy requirement is satisfied is essentially the same, but holds, as required from the definition in Sect. 2, only in more restricted time and item price intervals, due to the item price changes caused by future bids. Specifically, there are two main cases for this.

In the first case, a current winner C_i had bid a maximum price mp_i at time t_1 when the item was priced cp, and at a later time t_2 another client bids a maximum price $mp'_i < mp_i$. Then after time t_2, the item price changes to $cp' = mp'_i + \Delta$, and C_i remains the current winner. As both facts are publicly revealed by the server, this implies that $mp_i > cp' - \Delta$. However, in the time interval $[t_1, t_2]$, the same reasoning

as for the final winner privacy can be applied, and the price mp_i is, from the point of view of all other clients, still undetermined within the interval $[cp, 2^m - 1]$, except with negligible probability.

In the second case, the current winner C_i is outbid at a later time t_3 when another client bids a maximum price $mp'_i > mp_i$. Then after time t_3, the item price changes to $cp'' = mp_i + \Delta$, and C_i is not the current winner any more. As both facts are publicly revealed by the server, this implies that $mp_i = cp'' - \Delta$ and thus no more privacy is left about the value mp_i. However, in the time interval $[t_2, t_3]$, the same reasoning as for the final winner privacy can be applied, and the price mp_i is, from the point of view of all other clients, still undetermined within the interval $[cp', 2^m - 1]$, except with negligible probability.

Security against server. We assume client C_i and server S' run an execution of the bidding protocol Π_b, where C_i submits a maximum price $mp_i \geq cp + \Delta$, and an execution of price update protocol Π_{pu}, at the end of which S' publicly posts a bid transcript. We need to prove the two parts of this requirement: the update of the currently winning bidder and the update of the current price. In both parts, we show that any such polynomial-time algorithm S' cannot convince another polynomial-time algorithm J unless S' followed the protocol, where J only needs to verify the proofs and any new settings of value cp in the bid transcript posted by the server S'.

In the first case, the update of the currently winning bidder, we assume that $mp_i \geq cwb + \Delta$ and that, towards contradiction, this execution of Π_b does not result in the current price and currently winning bid to be updated to $cwb + \Delta$ and mp_i, respectively. This can happen if in its bid transcript, the server S' either (a) claims that $mp_i \leq cp$ or (b) claims that $mp_i \leq cwb$. However, in case (a), S' has to show an equality proof that mp_i is equal to some value $\leq cp$, and, because of the soundness property, such a proof would be convincing only with probability exponentially small in k. Similarly, in case (b), S' has to show both an equality proof that mp_i is equal to some value less than cwb, and a lower-bound proof that the bid encrypted by $cweb$ is $> mp_i - \Delta$. Here, the soundness of the lower-bound proof implies that, unless with probability exponentially small in k, S' at best manages to obtain a convincing lower-bound proof for a value of mp_i less than cwb, in which case the soundness property of the equality proof implies that S' produces a convincing lower-bound proof only with probability exponentially small in k.

In the second case, the update of the current price, we assume that $cp + \Delta \leq mp_i < cwb + \Delta$ and that, towards contradiction, this execution of Π_b does not result in the current price to be updated to $cp + \Delta$. This can happen if in its bid transcript, the server S' either (a) claims that $mp_i \leq cp$ or (b) claims that $mp_i \geq cwb + \Delta$. Here, case (a) follows from the soundness property of the equality proof, as already noted above. Moreover, in case (b), S' has to show both an equality proof that $cweb$ encrypts a value cwb, and a lower-bound proof that $mp_i \geq cwb + \Delta$. By the soundness of both proofs, this results in two convincing proofs only with probability exponentially small in k. □

5 Conclusions and future work

We uncovered a novel class of attacks that a server in a proxy auction can mount against its clients to increase bids or favor one player over another towards winning the auction. To address these attacks, we proposed techniques based on lower-bound proof systems to reduce the amount of trust that needs to be put on servers for the correct functioning of a private proxy auctions.

More attacks are possible from proxy auction servers, including attacks that are plausible in real-life auctions, such as coalitions of servers and clients trying to convince another, possibly fake, client to take part into the auction and put a higher bid on an item. Note that if the honest client is bidding without considering the current item price or the number of participants to the auction (as the auction itself does demand), then these attacks are of no advantage to the auction server. On the other hand, in real life clients may not respect this rule and thus the study of these attacks, left as future work, may be of interest.

Further attacks in proxy auctions include those to the privacy of clients, as all bids but the winner's are revealed to all auction participants or followers, due to the functionality of the proxy auction that we considered. One could use additional cryptographic techniques (for instance, keeping all bids and current price hidden via encryption) to enforce such privacy, but then these techniques would modify the proxy auction functionality, thus not necessarily capturing real-life auctions such as [1].

References

1. Http://www.ebay.com/.
2. Abe, M., & Suzuki, K. (2002). $M + 1$-st price auction using homomorphic encryption. In *LNCS: Vol. 2274. Proc. of public key cryptography'02* (pp. 115–224). Berlin: Springer.
3. Bogetoft, P., Damgard, I., Jakobsen, T., Nielsen, K., Pagter, J., & Toft, T. (2006). A practical implementation of secure auctions based on multiparty integer computation. In *LNCS: Vol. 4107. Proc. of financial cryptography'06* (pp. 142–147). Berlin: Springer.
4. Boudot, F. (2000). Efficient proofs that a committed number lies in an interval. In *LNCS: Vol. 1807. Proc. of Eurocrypt'00* (pp. 431–444). Berlin: Springer.
5. Blum, M., De Santis, A., Micali, S., & Persiano, G. (1991). Non-interactive zero-knowledge. *SIAM Journal of Computing*, *20*(6), 1084–1118.
6. Cachin, C. (1999). Efficient private bidding and auctions with an oblivious third party. In: *Proc. of ACM conference CCS'99* (pp. 120–127).
7. Camenisch, J., Chaabouni, R., & Shelat, A. (2008). Efficient protocols for set membership and range proofs. In *LNCS: Vol. 5350. Proc. of Asiacrypt'08* (pp. 234–252). Berlin: Springer.
8. Cramer, R., Damgård, I., & Schoenmakers, B. (1994). Proofs of partial knowledge and simplified design of witness hiding protocols. In *LNCS: Vol. 839. Proc. of Crypto'94* (pp. 174–187). Berlin: Springer.
9. Canetti, R., Goldreich, O., & Halevi, S. (1998). The random oracle methodology, revisited. In: *Proc. of ACM symposium on theory of computing'98* (pp. 209–218).
10. De Santis, A., Di Crescenzo, G., & Persiano, G. (1994). The knowledge complexity of quadratic residuosity languages. *Theoretical Computer Science*, *132*, 291–317.
11. De Santis, A., Di Crescenzo, G., & Persiano, G. (2004). On NC1 Boolean circuit composition of non-interactive perfect zero-knowledge. In *LNCS: Vol. 3153. Proc. of mathematical foundations of computer science'04* (pp. 356–367). Berlin: Springer.
12. De Santis, A., Di Crescenzo, G., Persiano, G., & Yung, M. (2008). On monotone formula composition of perfect zero-knowledge languages. *SIAM Journal on Computing*, *38*(4), 1300–1329.

13. Di Crescenzo, G. (2005). You can prove so many things in zero-knowledge. In *LNCS: Vol. 3822. Proc. of CISC'05* (pp. 10–27). Berlin: Springer.
14. Di Crescenzo, G. (2000). Private selective payment protocols. In *LNCS: Vol. 1962. Proc. of Financial Cryptography'00* (pp. 72–89). Berlin: Springer.
15. Di Crescenzo, G. (1995). Recycling random bits in composed perfect zero-knowledge. In *LNCS: Vol. 921. Proc. of Eurocrypt'95* (pp. 367–381). Berlin: Springer.
16. Di Crescenzo, G., Herranz, J., & Sáez, G. (2004). Reducing server trust in private proxy auctions. In *LNCS: Vol. 3184. Proc. of TrusBus'04* (pp. 80–89). Berlin: Springer.
17. Fiat, A., & Shamir, A. (1986). How to prove yourself: practical solutions to identification and signature problems. In *LNCS: Vol. 263. Proc. of Crypto'86* (pp. 186–194). Berlin: Springer.
18. Feige, U., Lapidot, D., & Shamir, A. (1999). Multiple non-interactive zero knowledge proofs under general assumptions. *SIAM Journal on Computing*, *29*(1), 1–28.
19. Goldreich, O. (2004). *Foundations of cryptography: basic applications*. Cambridge: Cambridge University Press.
20. Goldwasser, S., & Micali, S. (1984). Probabilistic Encryption. *Journal of Computer and System Sciences*, *28*(2), 270–299.
21. Goldwasser, S., Micali, S., & Rackoff, C. (1989). The knowledge complexity of interactive proof-systems. *SIAM Journal on Computing*, *18*(1), 186–208.
22. Harkavy, M., Tygar, D., & Kikuchi, H. (1998). Electronic auctions with private bids. In: *Proc. of 3rd USENIX workshop on electronic commerce* (pp. 61–74).
23. Juels, A., & Szydlo, M. (2003). A Two-server, sealed-bid auction protocol. In *LNCS: Vol. 2357. Proc. of financial cryptography'03* (pp. 72–86). Berlin: Springer.
24. Kikuchi, H. (2001). (M+1)st-price auction protocol. In *LNCS: Vol. 2339. Proc. of financial cryptography'01* (pp. 351–363). Berlin: Springer.
25. Lipmaa, H. (2003). On diophantine complexity and statistical zero-knowledge arguments. In *LNCS: Vol. 2894. Proc. of Asiacrypt'03* (pp. 398–415). Berlin: Springer.
26. Lipmaa, H., Asokan, N., & Niemi, V. (2002). Secure Vickrey auctions without threshold trust. In *LNCS: Vol. 2357. Proc. of financial cryptography'02* (pp. 87–101). Berlin: Springer.
27. Mao, W. (1998). Guaranteed correct sharing of integer factorization with off-line share- holders. In *LNCS: Vol. 1431. Proc. of public-key cryptography'98* (pp. 60–71). Berlin: Springer.
28. Naor, M., Pinkas, B., & Sumner, R. (1999). Privacy preserving auctions and mechanism design. In: *Proc. of the ACM Conference on Electronic Commerce* (pp. 129–139).
29. Pedersen, T. P. (1991). A threshold cryptosystem without a trusted party. In *LNCS: Vol. 547. Proc. of Eurocrypt'91* (pp. 522–526). Berlin: Springer.
30. Sako, K. (2000). An auction protocol which hides bids of losers. In *LNCS: Vol. 1751. Proc. of public key cryptography'00* (pp. 422–432). Berlin: Springer.
31. Schnorr, C. P. (1990). Efficient identification and signatures for smart cards. In *LNCS: Vol. 435. Proc. of Crypto'89* (pp. 239–252). Berlin: Springer.
32. Sakurai, K., & Miyazaki, S. (1999). A bulletin-board based digital auction scheme with bidding down strategy. In: *Proc. of CrypTEC'99* (pp. 180–187).
33. Stubblebine, S., & Syverson, P. (1999). Fair on-line auctions without special trusted parties. In *LNCS: Vol. 1648. Proc. of financial cryptography'99* (pp. 230–240). Berlin: Springer.
34. Teranishi, I., & Sako, K. (2006). K-times anonymous authentication with a constant proving cost. In *LNCS: Vol. 3958. Proc. of public key cryptography'06* (pp. 525–542). Berlin: Springer.
35. Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance*, *16*(1), 8–37.
36. Yuen, T. H., Huang, Q., Mu, Y., Susilo, W., Wong, D. S., & Yang, G. (2009). Efficient non-interactive range proof. In *LNCS: Vol. 5609. Proc. of Cocoon'09* (pp. 138–147). Berlin: Springer.

Giovanni Di Crescenzo is a senior scientist at Telcordia Technologies with 15+ year research experience. He received a Ph.D. in Computer Science and Engineering from the University of California San Diego, USA, (with a thesis on cryptography) and a Ph.D. in Applied Mathematics and Computer Science from the University of Naples, Italy (with a thesis on zero-knowledge proofs). His main research activity has been in various areas of Mathematics and Computer Science, including Cryptography, Computer/Network/Information Security, Computational Complexity and Algorithms, where he has produced 100+ scientific publications in major, refereed conferences and journals, including 1 book, 1 book chapter, 3 proceedings of conferences or workshops for which he was a technical program chair. He was awarded

more than 10 among prizes and patents, including the O-1 United States VISA for aliens of extraordinary ability in the field of science (given to Nobel Prize winners or individuals "showing sufficient evidence of being one of the small percentage who have arisen to the very top of the field of endeavor"). He has given more than 20 invited talks, regularly referees papers for the major conferences and journals in his areas of expertise, and has been involved in more than 15 research projects funded by government agencies or commercial institutions in the area of security, telecommunication, telematics, and entertainment.

Javier Herranz obtained his Ph.D. in Applied Mathematics in 2005, in the Technical University of Catalonia (UPC, Barcelona, Spain). After that, he spent 9 months in the École Polytechnique (France), 9 months in the Centrum voor Wiskunde en Informatica (CWI, The Netherlands) and 2 years in IIIA-CSIC (Bellaterra, Spain), as a post-doctoral researcher. Currently he enjoys a Ramón y Cajal grant by the Spanish Ministry of Education and Sciences, for post-doctoral research in the group MAK (Dept. Applied Mathematics IV, UPC). His research interests are related to cryptography and privacy of databases.

Germán Sáez completed his first degree in mathematics at the University of Barcelona (UB), Spain, in 1987. He is now an associate professor in the Department of Applied Mathematics IV at the Technical University of Catalonia (UPC), Barcelona, where he was awarded his Ph.D. degree in mathematics in 1998. His research interests include secret sharing schemes and digital signatures in distributed scenarios. He belongs to UPC's Research Group on Mathematics Applied to Cryptography (MAK). and served as the general cochair of Eurocrypt'07.