Practical secrecy-preserving, verifiably correct and trustworthy auctions.

David C. Parkes, Michael O. Rabin, Stuart M. Shieber, and Christopher A. Thorpe. Practical secrecy-preserving, verifiably correct and trustworthy auctions. In *Proceedings of the Eighth International Conference on Electronic Commerce (ICEC '06)*, pages 70–81, Fredericton, New Brunswick, Canada, 14-16 August 2006.

Practical Secrecy-Preserving, Verifiably Correct and Trustworthy Auctions

D. C. Parkes, M. O. Rabin, S. M. Shieber, and C. A. Thorpe
Harvard University
Division of Engineering and Applied Sciences
Cambridge MA 02138
{parkes,rabin,shieber,cat}@deas.harvard.edu

ABSTRACT

We present a practical system for conducting sealed-bid auctions that preserves the secrecy of the bids while providing for verifiable correctness and trustworthiness of the auction. The auctioneer must accept all bids submitted and follow the published rules of the auction. No party receives any useful information about bids before the auction closes and no bidder is able to change or repudiate her[1] bid. Our solution uses Paillier's homomorphic encryption scheme [25] for zero knowledge proofs of correctness. Only minimal cryptographic technology is required of bidders; instead of employing complex interactive protocols or multi-party computation, the single auctioneer computes optimal auction results and publishes proofs of the results' correctness. Any party can check these proofs of correctness via publicly verifiable computations on encrypted bids. The system is illustrated through application to first-price, uniform-price and second-price auctions, including multi-item auctions. Our empirical results demonstrate the practicality of our method: auctions with hundreds of bidders are within reach of a single PC, while a modest distributed computing network can accommodate auctions with thousands of bids.

1. INTRODUCTION

In recent years, auctions and electronic marketplaces have been used to facilitate trillions of dollars in trade in the world economy [10]. Individual events, for instance, the procurement of truckload services by Proctor and Gamble, approach $1 billion in transaction value [32]. The eBay marketplace reported a record $44.3 billion volume in the 2005 calendar year, representing a 30% increase over 2004. Governments world-wide use auctions to allocate property rights, including high profile auctions for wireless spectrum [20] and licenses for new cars [2]. Previously used for rare goods, or for time-sensitive goods (e.g., flowers, fish), auctions can now be harnessed for all kinds of commercial transactions [21]. In a typical week in February, 2006, the U.S. treasury sells more than $25 billion in three-month treasury bills.[2] Most recently, sponsored search auctions have driven upwards of $1 billion in revenue to Google in a single quarter.

Despite this success, there is increasing evidence that fraud is an issue that can plague electronic auctions.[3] Indeed, a number of authors have argued that the reason that theoretically appealing auctions such as Vickrey auctions are rare in practice is because of the problem of fraud and untrustworthy auctioneers [31, 19]. Two kinds of manipulations come to mind. The first is an auctioneer that deviates from the rules of an auction. This problem can be alleviated at a cost in privacy by the public revelation and verification of all bids. Another more subtle and harder to prevent problem can occur when an auctioneer is in collusion with some bidders, perhaps conveying useful information about bids received during the bidding process.

We have developed a practical protocol for sealed bid auctions that prevents such manipulations. An important factor in its practicality is having a clearly understandable and convincing solution accessible to knowledgable people who are nevertheless not experts on the intricacies of cryptography and general zero knowledge proofs. To that end, we have carefully examined the role of all parties in a sealed-bid auction and formalized their role in a cryptographically sound protocol. We consider who among them needs to know what, and when; based on that, we have constructed a protocol whose primary aim is not complete security, but rather practicality. We touch on the real-world issues that arise in the actual implementation of such a system. Our protocol provides clear proofs of correctness that reveal minimal knowledge to the parties involved, yet is easily implemented and requires no special technology on the part of the bidders.

We assume only commodity computing resources and a public key infrastructure under which the auctioneer, seller, bidders, and notaries all possess public/private key pairs for digital signatures. The auctioneer holds a private key for bid encryptions and publishes an appropriately certified public key. Bids are encrypted by bidders using this public key, although encrypted bids are kept secret from the auctioneer until an auction closes. The cryptographic methods of *homomorphic encryption* [25] are used in providing verifiable correctness and trustworthiness.

[1] For clarity of reference, we use "she", "her", etc. to refer to the bidders and verifiers, and "he", "his", etc. to refer to the auctioneer (generally a prover), and other parties to the auction.

[2] Sold in Vickrey auctions. See `http://www.publicdebt.treas.gov`

[3] The number of fraud complaints submitted to the FBI's Internet Crime Complaint Center rose from 16,775 in 2001 to 103,959 in 2004, and the percentage from auction fraud in particular rose from 42.8% to 71.2% over the same period.

ICEC '06, August 14–16, 2006. Fredericton, Canada.

We thus present a framework for auctions that is both *practical* and *secrecy preserving*, while providing for *verifiably correct* and *trustworthy* auctions. We focus on two aspects of practicality. First, the auction must clear in reasonable time and with reasonable communication requirements, even for a large number of bidders. Second, the computational architecture must be consistent with practical business models. To achieve this we focus on *proofs of correctness* rather than secure computation. Unlike previous solutions, e.g., Naor et al. [22], we require neither the existence of multiple auctioneers nor that the auctioneers or bidders collaborate to conduct the auction. We believe that a model involving a single auctioneer that is solely responsible for conducting the auction and independent verification of the auction by third parties is more realistic from a business perpective.

We preserve secrecy by keeping bid information secret from everyone except the auctioneer, and keeping bid information secret even from the auctioneer until the auction closes. The only information revealed to the public is that implied by the outcome of the auction, that is, that implied by the identity of the winner or the payment made by the winner. A secrecy-preserving verification protocol allows anyone, including bidders and third parties, to verify that the auction was correct: the auctioneer correctly determined the winner(s) and associated payment(s) according to published rules. Most importantly, trustworthiness is supported by carefully ensuring that all bidders must follow through with information about bids of pre-committed value and quantity after an auction closes, and the auctioneer must accept and respect all bids in determining the outcome.

In addition to a seller, multiple bidders, and an auctioneer, our model assumes the following commercial entities: *Notaries* ensure the security of an auction by acting as witnesses. Notaries witness bid submission in order to protect a bidder against an auctioneer who tries to ignore her bids. They may also be used to enforce nonrepudiation of bids after the auction is closed. *Delayed private key revelation services* are used to prevent a bidder from refusing to respect commitments she has made during the auction protocol.

Our auction process ensures verifiable correctness and trustworthiness. Although an auctioneer learns the values of bids submitted after an auction closes, an auctioneer is not able to use this information to change the outcome in the auction or provide an advantage to any bidder. Thus, we prevent a "bad apple" within the auctioneer's organization from being able to profitably leak information during the course of an auction. On the other hand, and at a considerable gain in simplicity, we have deliberately chosen *not* to protect against an auctioneer revealing bid values and quantities after an auction has closed and the outcome has been announced. Our design does not provide any algorithmic enforcement for this additional privacy protection. Algorithmic and software methods are available for solving this problem. But in our view they are too cumbersome and hard to understand to find wide business applicability.[4]

In solving what we view as the first-order problem of trustworthiness during the execution of an auction, we choose to push these secondary considerations into the realm of contractual obligations and the auctioneer's business reputation. An additional benefit is that this architecture will also allow for extensions to combinatorial auctions for which fully-private computational techniques cannot scale.

Parenthetically, we observe that *complete secrecy* by the auctioneer can be provided, in cases where it is deemed absolutely essential, by appeal to hardware solutions. *Trusted servers*, with specially designed hardware and software audited by third parties for correctness, and installed in physically secure locations with ongoing monitoring and auditing, can prevent the leaking of information with high assurance [33]. In fact, with such deliberately opaque servers it is of the utmost import that an auction participant can independently verify the correctness of the outcome of an auction and be assured that there is no fraud. Thus, such technological methods to eliminate secrecy leaks are very well complemented by our methods for verifiable correctness.

To demonstrate the scalability of our technology, we have conducted preliminary timing tests (Section 5). We show that for acceptable strength of the cryptographic security key, single or multi-item auctions with 100 bidders can be prepared in around two hours of computation and verified in less than half an hour, all on a standard (2.8 GHz Pentium 4) PC. We also show that the computations scale linearly with the number of bidders. Because our method is easily parallelizable, it is possible to accommodate even tens of thousands of bidders in at most a day of computation on a 64-node network.

1.1 Related Work

Much of the previous work on the use of cryptography for conducting verifiably correct and trustworthy auctions has focused on the goal of *complete privacy* [16, 22, 13]; see Brandt [5] for a recent discussion. This is typically achieved through assuming two or more trusted third parties, either with numerous auctioneers [13] or with asymmetric models in which the commerical entity of an *auction issuer* is assumed in addition to the auctioneer [22, 18]. Some protocols achieve this property through bidder-resolved multi-party computation [5]. In comparison, we settle for verifiable correctness and trustworthiness in combination with complete privacy to all parties except the auctioneer; see also [11]. As discussed above, the auctioneer cannot learn any information about bids until the auction has closed. In return we achieve a non-interactive[5] protocol that is especially simple from a bidder's perspective. For trusted third parties we require only notaries, who provide a lightweight "witness" service and are independent business entities that already exist in practice [34]. In the same spirit, whereas previous architectures use cryptography for anonymity, we adopt business entities (e.g., notaries as proxy bidders) for this purpose. Note that achieving information-theoretic guarantees on privacy is impossible in most Vickrey auctions [6]. A single-item Vickrey auction, for example, necessarily reveals the exact second place bid to the winner.

In addition to providing business realism (also see Lipmaa et al. [18] for a critique of previously published methods), we choose to adopt standard methods from homomorphic encryption combined with test sets and eschew more complex cryptographic ideas

[4] We also note that even when bid values stay concealed from the auctioneer at great process complexity cost, a determined adversary can try to spy and obtain information on a rival's bid using corrupt insiders. Thus, an absolute guarantee of secrecy is not attainable in real life in any case.

[5] Interactive cryptographic auction protocols require the active participation of bidders throughout the auction process in order to obtain the auction results, generally via multi-party computation or related methods. Non-interactive protocols such as ours require no such bidder participation; submission of bids is the only required bidder activity.

such as secure multi-party computation, obfuscation of circuits, and oblivious transfer. As Bradford et al. [4] argue, many such complex protocols requiring the participation of bidders suffer from "protocol completion incentive problems", in which bidders who know they have lost or change their minds can disrupt the protocol and prevent the completion of an auction. We intentionally avoid such problems by having a single trusted party compute the outcome.

We share with Lipmaa et al. [18] (see also [1, 3, 5, 35, 36]) the use of homomorphic encryption, but seek a simpler solution through the use of a single auctioneer in place of the two server model adopted in their work. In their protocol, the seller and an auction authority, who are trusted not to collude, work interactively to generate zero-knowledge proofs of correctness. This results in stronger privacy properties at the cost of this additional process complexity.

Our approach can be extended to secrecy-preserving multi-item auctions (presented here) and combinatorial auctions (reserved for future work). Specifically, our trusted auctioneer can apply fast algorithms to the combinatorial optimization problem in determining winners. The auctioneer must simply construct a *proof* that the outcome is correct and need not involve multiple parties in *computing* the outcome. Earlier work on multi-item auctions either assumes distributed trust [14, 36, 1], or adopts multi-party computation techniques [5], and the current state of the art for secure combinatorial auctions is still not very scalable [37, 35]. One practical issue, addressed in previous work but not here, is that of *noncoercibility* [7, 34] of an auction. Noncoercibility prevents a bidder from being able to credibly claim to a third party that it bid in a particular way after the close of an auction. Auctions with this property are resistant to bidding rings, which depend on bidders proving that their bid was submitted according to the rules of the bidding ring.

2. PRELIMINARIES

2.1 Desired Auction Properties

Based on the analysis in the introduction, we list desiderata for our auction process.

- Non-repudiation by bidders: Once a bidder submits a bid, her bid is provably unalterable. Moreover, a bidder is committed to finally revealing her bid.
- Non-repudiation by auctioneer: The auctioneer's exclusion of a properly submitted bid can be conclusively proven and thus becomes legally actionable.
- Trustworthiness: The auctioneer cannot know the bids until after the close of the bid submission phase. Thus the auctioneer cannot collude with bidders by sharing others' bids during the auction.
- Verifiable correctness: The public and bidders receive a proof of which bids won, and (if applicable) a proof of the correctness of their own payments. The auction protocol enforces correctness; an auctioneer will not be able to present valid proofs for invalid winners or incorrect payments.
- Privacy: The bids are hidden to everyone until all bids are committed. At the close of the auction, only the auctioneer knows any private information. He may keep the outcome private, notifying only winners, or make it public by revealing some or all of the bids, items won by whom, and payments. Revelation of these values does not reveal other private information not implied by the values themselves.

In achieving these properties we make standard cryptographic assumptions. For our homomorphic encryption, we make Paillier's "Composite Residuosity Assumption" (CRA) [25].[6] CRA implies that if the public key n is difficult to factor, then it is also difficult to compute the n^{th} root of a number $x = r^n \pmod{n^2}$. This assumption is related to the widely accepted RSA assumption underlying the security of RSA encryption and is believed to be of similar strength.[7] We further assume that the cryptographic hash function used for commitments preserves the security of the encrypted bids. See Section 2.4.2 for a detailed description of such hash functions. Because the security of our encryption is related to the computational complexity of solving these cryptographic problems, longer cryptographic keys and more complex hash algorithms can be adopted over time as computational hardware gets more powerful. This will maintain the same level of realized security at comparable computational running time.

2.2 Real-World Components

As usual, our auction system comprises an auctioneer AU, bidders $B = \{B_1, \ldots, B_k\}$, and a seller. Bidders can also be *proxies* to provide anonymity. In addition, we assume a universally accessible, tamper resistant clock (such as provided by the United States NIST time servers) and the following two components:

2.2.1 Certified Bulletin Board

The auctioneer maintains a certified bulletin board. This can be a publicly known website maintained and updated by the auctioneer. The auctioneer uses the bulletin board to post all public information about the auction, including the initial auction announcement as well as (encrypted) information about bids that have been submitted and proofs that can be used to verify all publicly available information about the outcome. All posts to the the bulletin board will carry appropriate digital signatures.

2.2.2 Notaries

Notaries $N = \{N_1, \ldots, N_m\}$ are reputable agents, such as law firms, accountants, or firms specializing in providing a *witness* for bidders. When preparing to participate in an auction, a bidder may select a set of notaries of her choosing from some set of notaries possibly authorized by the auctioneer.[8] In using a notary, whenever a bidder sends bid information to the auctioneer she also sends the information to any notaries she has selected. These notaries also submit this information to the auctioneer, and act as witnesses in the case that a bidder notices that an auctioneer does not post correct information to the bulletin board. We require that a majority of the notaries is uncorruptible. Note that our process is structured so that no information about the actual bids is revealed to the notaries, and their only role is to serve as witnesses in case of a dispute between a bidder and the auctioneer.

2.3 Overall Flow and Main Steps of Auction

Schematically, the auction process will proceed in three main stages. In the first stage, the auctioneer posts the auction announcement on the bulletin board. The announcement, to be detailed later

[6] A number $x = r^n \pmod{n^2}$ is known as an n^{th} *residue* mod n^2. Because n is a composite number — the product of two primes — x is a composite residue.

[7] In the RSA problem, if x is an e^{th} residue mod n, that is, $x = m^e \pmod{n}$, x is secure if n's only two prime factors are unknown and n is hard to factor.

[8] Whether or not the use of a notary is mandatory depends on details of how non-repudiation is achieved. See the discussion in Section 3.1.

on, includes a deadline time T for submitting bids. In the second stage, the bidders commit to bids but post bid information in a form that is hidden even to the auctioneer. Notaries are engaged in this stage. In the final stage, the bidders must follow through and reveal the encryptions of their bids to the auctioneer and the public. The auctioneer and other bidders verify that these encrypted bids are consistent with the posted commitments. The auctioneer then decrypts the bids in secret, and performs computation to determine the optimal outcome of the auction. The auctioneer then posts public proofs that the selection of the winner(s) and their payments was done according to the auction rules. After the last posting, any party can verify the correctness of the outcome.

2.4 Cryptographic Tools

Our system relies on mathematically sound and widely accepted cryptographic tools. We describe the tools we employ in our result, referring to other publications for established results and providing proofs for new uses of existing tools. We will sometimes refer to a "prover" $\mathcal{P}$ and a "verifier" $\mathcal{V}$ when discussing secrecy-preserving proofs of mathematical facts relating to our auctions.

2.4.1 Public Key Infrastructure

We assume cryptographically sound methods of establishing and exchanging public keys used for all the cryptographic tools we employ. Specifically, the auctioneer requires a public/private key pair for Paillier encryption [25]. Public keys are used for encryption and private keys for decryption. In addition, the auctioneer, notaries, and all bidders require public/private key pairs for digital signatures. The public keys of all parties must be mutually known and certified correct. We notate digital signatures as follows: AU can sign message x, generating $\mathbf{Sign}_{AU}(x)$. A bidder B_i's signature of x is denoted $\mathbf{Sign}_i(x)$.

2.4.2 Commitments

Cryptographic commitments enable a party to commit to a particular value (such as a bid or number of items desired) without revealing that value until later, and prevent the party from claiming their value was anything other than the original value committed to. Auction participants will, when so required, commit to data D by applying a *hash function* H to data D then digitally signing that hash value $H(D)$. The hash function is required to be *perfectly concealing* of all information about D, as well as *collision resistant* in the sense that it is computationally intractable to find two different data values x and y such that $H(x) = H(y)$. In practice, we can employ a widely used hash function such as Whirlpool [28] or a member of the SHA-2 [23] family, which are assumed to have the required properties.

2.4.3 Sources of Randomness

Cryptographic key generation and probabilistic encryption require a good source of random data. We postulate bidders' and notaries' ability to create enough highly random data to create strong key pairs and encrypt or sign a small number of values. We further postulate that the auctioneer has a source of random data sufficient to encrypt large numbers of integers used in the secrecy-preserving proofs described below. Such a source might be hardware that extracts randomness from radio static or quantum noise in diodes. Such "hardware randomness generators" are already in widespread use in applied cryptography.

2.4.4 Delayed private key revelation

Let $\mathbf{Bid}_i$ denote the bid value of bidder B_i. We need to guard against a bidder B_i, possibly in collusion with the auctioneer, refusing to open her commitment and reveal her encrypted bid $E(\mathbf{Bid}_i)$. One approach to provide non-repudiation employs a *delayed private key revelation service*, *DPrKR*. Such a service will at regular intervals (say every minute) post a new cryptographic public key followed by posting the associated private decryption key after a fixed period of time (say an hour later).[9]

Before time T of the close of the auction, each bidder B_i encrypts $Z = E_{DR}(E(\mathbf{Bid}_i))$ (where the bid is first encrypted with the public key of the auctioneer), and posts $\mathbf{Sign}_i(Z)$ on the bulletin board using a *DPrKR* public encryption key *DR* whose decryption key will be released after time $T+1$. After time $T+1$, the decryption key *DDR* associated with *DR* will be posted by *DPrKR*. This method will be used to allow the auctioneer (and everybody else) to decrypt $Z = E_{DR}(E(\mathbf{Bid}_i))$ using *DDR* after time $T+1$ if the bidder herself refuses, guaranteeing the auctioneer alone access to $\mathbf{Bid}_i$. If so desired, several independent *DPrKR* services can be used for combining encryptions.

2.4.5 Verifiable and Confidential Comparisons

Paillier's encryption scheme [25] enables integer values to be encrypted in such a way that it is possible to perform arithmetic operations on those values using only the encrypted data.[10] We present a technical exposition in Appendix A for the interested reader.

Paillier's is a *homomorphic encryption* system, in which the result of an operation applied to two ciphertexts is a valid encryption of an operation (possibly the same one) applied to their plaintexts.[11] In cryptography, a *plaintext* is the original form of a message, in our case the integer representing a bid or quantity; a *ciphertext* is the encryption of a plaintext. Homomorphic encryption schemes enable computation over the hidden values without revealing either the values themselves or the results of the computation without proper decryption. Paillier's system employs a public/private key pair, n and ϕ respectively. The private key n is the product of two large prime numbers p and q, and its size is determined by the security requirements of the application. A 1024-bit public encryption key is widely considered sufficient for security until 2010 [12]. Paillier encryption is also a "probabilistic encryption" scheme. Encryptions are performed with a random "help value" r that is used to achieve *semantic security*: given two plaintexts and two encryptions of them, one cannot tell which ciphertext corresponds to which plaintext without being able to decrypt them. Semantic security is critical for our test set mechanism to preserve the secrecy of the bids.

The encryption of a message x will typically be denoted $E(x,r)$, where the public key n is implicit and the help value r is made explicit. In discussion below, the help value r will sometimes be omitted to simplify notation where it is implicit or irrelevant, for example, $C = E(x)$.

[9] Rivest et al. [30] also provide a method for cryptography with forced time release where the user sends x to a time-released cryptography service which sends $E_s(x)$ using a secret key s to be released later on. Thus, in their solution the service knows the secret.

[10] Paillier proves the security of his scheme under the "composite residuosity assumption", the assumption of the computational infeasability of solving a number theoretic problem similar in difficulty to the assumptions underlying the security of other cryptosystems such as RSA [29], ElGamal [9], and Rabin [27] encryption.

[11] More formally, in a homomorphic encryption scheme, there exist operations $\oplus$ and $\otimes$ such that given ciphertexts $C_1 = E(x_1)$ and $C_2 = E(x_2)$, $C_1 \otimes C_2 = E(x_1 \oplus x_2)$. Paillier's encryption scheme is doubly homomorphic. See Appendix A.

We present here a summary of the properties of, and extensions to, Paillier's scheme we use in this paper. First, given only the encryption $E(x_1)$ and either another encryption $E(x_2)$ or a constant k, anyone can compute the encryptions $E(x_1 + x_2)$, $E(x_1 + k)$, and $E(x_1 \cdot k)$ *without learning anything about* x_1*,* x_2*, or* n*.* Second, based on these properties and the following Range-of-Values tests, we can also prove a full set of *inequality operations* for two encrypted values $E(x_1)$ and $E(x_2)$, e.g., $x_1 = x_2$, $x_1 > x_2$, etc., again, without revealing anything about x_1 or x_2. It is also possible to compare encrypted bids to constants in a similar way. We employ the notation $E(x) \trianglelefteq E(y)$ to mean "$x \leq y$ can be proven using encrypted values $E(x)$ and $E(y)$" and the similar notation $\trianglerighteq$ ($\geq$), $\triangleleft$ ($<$), and $\triangleright$ ($>$). The verification of these comparisons is detailed in Appendix A.4.

2.4.6 Verifiable, Confidential Range-of-Values Tests

Given ciphertext $C = E(x,r)$ we need to prove that $x < 2^t$ for some t such that $2^t < n/2$. That is, we we want to be able to verify that a bid $\mathbf{Bid}_i$ is smaller than some agreed upon bound 2^t, without revealing any information about $\mathbf{Bid}_i$. The value of t determines the number of bits of resolution available to bidders in seelcting their bids. For our purposes it suffices to take $t = 34$, so that if bids are in units of one thousand dollars, for example, then bids are limited to at most $16 trillion.

This primitive is essential for proving inequalities. Because some of our mathematical operations are over the integers modulo n ($\mathbb{Z}_n$), a small negative number is the same as a large positive number, and vice versa. For example, $13 \equiv -2 \pmod{15}$. To prove for two values a and b that $a \geq b$, we first show that $a, b < n/2$ and then that $a - b < n/2$. This works because if a and b are less than $n/2$ and a is greater than b, then clearly $a - b < n/2$, and if a is *less* than b, then $a - b$ will "wrap around" modulo n and must be a large number, that is, $a < b \rightarrow a - b \pmod{n} > n/2$. The formal details of this are found in Appendix A.4.

We perform the test as follows:

Definition 1. *A valid* test set TS *for the assertion "*$C = E(x,r)$ *is an encryption of a number* $x < 2^t < n/2$*" is a set of* $2t$ *encryptions:*

$$TS = \{G_1 = E(u_1,s_1), \ldots, G_{2t} = E(u_{2t},s_{2t})\} \tag{1}$$

where each of the powers of 2*:* $1, 2, \ldots, 2^{t-1}$ *appears among the* u_i *exactly once and the remaining* t *values* u_j *are all* 0*.*

By use of a test set TS, the prover $\mathcal{P}$ can prove that $x < 2^t < n$ as follows:

Range Protocol. Let $x = 2^{t_1} + \ldots + 2^{t_\ell}$ be the representation of m, a sum of distinct powers of 2. AU selects from TS the encryptions $G_{j_1}, \ldots, G_{j_\ell}$ of $2^{t_1}, \ldots, 2^{t_\ell}$, and further $t - \ell$ encryptions $G_{j_{\ell+1}}, \ldots, G_{j_t}$ of 0. Note that:

$$\left(E(x,r)^{-1} \cdot G_{j_1} \cdot \ldots \cdot G_{j_t}\right) \pmod{n^2} = E(0,s) \tag{2}$$

is an encryption of 0 with help value $s = \left(r^{-1} \cdot s_{j_1} \cdot \ldots \cdot s_{j_t}\right) \pmod{n}$ if and only if indeed $x = 2^{t_1} + \ldots + 2^{t_\ell}$ and the G_{j_h} were chosen as stated. Now since AU has the decryption key ϕ and thus knows the help value r, then he can hand over to $\mathcal{V}$ the set $\{G_{j_1}, \ldots, G_{j_t}\}$ and the above help value s. $\mathcal{V}$ can now verify on her own that (2) holds and deduce that $x < 2^t < n/2$. □

The above protocol reveals nothing to $\mathcal{V}$ beyond $x < 2^t < n/2$, because TS is a set, in actual implementation a randomly permuted array of the elements in question. Consequently $\mathcal{V}$ has no information about *which* encryptions of powers of 2 are included in $\{G_{j_1}, \ldots, G_{j_t}\}$ Furthermore, the inclusions of $t - \ell$ encryptions of 0 hides even the number of non-zero bits in the binary representation of m. Finally, the inclusion of random factors $s_{j_1}, \ldots, s_{j_t}$ in the computation of the help value s completely masks any information about the help value r in the encryption $E(x,r)$. Consequently no information about x is revealed.

There is, however, a problem with the above protocol in that $\mathcal{V}$ does not know that AU has presented her with a true test set. This is overcome as follows. For ease of understanding we first describe an interactive verification protocol, then modify it for non-interactive use. The idea is to use a "cut and choose" procedure in which the prover commits to a number of test sets and allows the verifier to choose and inspect multiple test sets and make sure that they are each valid. Finally, the remaining test sets are all used to complete the proof. An early, possibly the first, use of this idea was presented by Rabin [26].

Tamper Proof Interactive Verification of $x < 2^t < n/2$**.** First, the prover $\mathcal{P}$ creates $2v$, say for $v = 20$, test sets $TS_1, \ldots, TS_{2v}$, and presents those to $\mathcal{V}$ claiming that they are all valid. Verifier $\mathcal{V}$ randomly selects v test sets $TS_{i_1}, \ldots, TS_{i_v}$ and requests that $\mathcal{P}$ reveal all the encryptions by revealing all the corresponding help values. $\mathcal{V}$ verifies all the encryptions and checks that every TS_{i_h} is valid. If any verification fails, the process is aborted. Otherwise, there now remain v unexamined test sets, call them $TS_{j_1}, \ldots, TS_{j_v}$. $\mathcal{P}$ now completes v repetitions of the above **Range Protocol**, and establishes that $x < 2^t < n/2$ by use of each of the above remaining v test sets. If all verifications succeed then $\mathcal{V}$ accepts that indeed $x < 2^t < n/2$.

The only way that $\mathcal{P}$ can cheat is if all the above remaining v test sets are invalid, which requires that initially the $2v$ test sets comprised v proper test sets and v improper ones and, furthermore, when examining the test sets, $\mathcal{V}$ randomly chose all the v proper ones. The probability of such an unfortuitous choice is $\binom{2v}{v}^{-1}$. In our example of $v = 20$, that probability is, by Sterling's Theorem, about $\sqrt{\frac{20\pi}{2^{40}}} < \frac{8}{10^{12}}$. Thus, we have a zero-knowledge protocol for $\mathcal{V}$ to verify interactively with AU that $x < 2^t < n/2$, when given a ciphertext $E(x,r)$ such that the inequality actually holds.

Tamper Proof Non-Interactive Verification of $x < 2^t < n/2$**.** We prefer to adopt the following non-interactive method to establish the validity of test sets in our scheme. In what follows, we adopt the auctioneer AU as the prover. Suppose that there are (as in Section 3.2) $2k$ range-of-values tests to perform. On closing the auction but before receiving information about bids, AU posts $4kv$ test sets on the bulletin board. (For expository convenience, we proceed below with our assumption of $v = 20$.)

Prior to closing, each bidder, the seller (if desired), and the auctioneer are also asked to commit to a random string of length M bits, which will be revealed after the auction closes and after the auctioneer commits to test sets. Given strings S_i from each bidder, S_S from the seller, and S_{Au} from the auctioneer, the strings are XORed together to generate $X = S_1 \oplus S_2 \oplus \cdots \oplus S_k \oplus S_S \oplus S_{AU}$. Note that even if only one of the participants chooses his string randomly and independently, then X is a truly random string.

The $80k$ test sets posted on the Bulletin Board are then segmented into $2k$ groups of 40 test sets each, i.e., the first 40 test sets, the next 40 test sets, etc. The random bit-string X is then used, in combination with a fixed rule available to all participants and posted at the start of the auction to the bulletin board, to select 20 test sets from each group. This random selection replaces the random selection by the verifier $\mathcal{V}$ employed in the interactive proof and allows the proof to work without interaction. In Appendix B we offer an accelerated version of this non-interactive verification, that we refer to as *bulk verification*. Bulk verification verifies all the test sets used in the auction *en masse* and economizes on the number of random sets that must be checked.

Damgård et al. [8] and Lipmaa et al. [18] present other solutions for proving an encrypted value is within a particular range.

3. SINGLE-ITEM AUCTIONS

Given the above cryptographic tools, we can formulate a single-item auction succinctly. We assume that the bidders $B_1, \ldots, B_k$ are known entities with publicly known digital signatures $\mathbf{Sign}_i$. We further assume that the winner and her payment depend only on the ordering of the bids and that the payment is one of the bids. This class of auctions include first-price and second-price auctions, and also allows for auctions with reservation prices by a simple extension in which the seller also submits a bid [15].

3.1 Protocol

Step 1. *AU* posts the following information on the bulletin board: the terms of the auction specifying the item, the mechanism for selection of the winner, the deadline T, an identifier *ID* of the auction, and a Paillier encryption key n. *AU* knows the corresponding decryption key ϕ. The auctioneer also posts information about the notaries that are to be used for the auction. He posts the cryptographic hash function H to be used by all participants in constructing their commitments. Finally, the auctioneer defines the method that will be used for extracting a random permutation of test set indices from a random string to be used when proving the correctness of the auction.[12]

We emphasize that all of the above data D_{AU} is posted on the bulletin board, accompanied by *AU*'s signature $\mathbf{Sign}_{AU}(D_{AU})$.

Step 2. Every B_i chooses a bid $\mathbf{Bid}_i$. She encrypts it as $C_i = E(\mathbf{Bid}_i, r_i)$ using the public key n and a randomly chosen help value r_i. In order to create efficient test sets to prove bid sizes, we restrict the size of the bid so that $\mathbf{Bid}_i < 2^t < n/2$ for small t, say, $t = 34$. Every B_i also generates a random bit string S_i of length M which is used in the proof. Bidder B_i then commits to C_i and S_i by using the hash function, to form a single commitment string $\mathbf{Com}_i = [H(C_i), H(S_i), ID]$, which also includes the identifier *ID* of the auction. Finally, the bidder signs this commitment, and sends $\mathbf{Sign}_i(\mathbf{Com}_i)$ to *AU* and her notaries, if used, before time T. *AU* returns a signed receipt $R_i = \mathbf{Sign}_{AU}([\mathbf{Com}_i, ID, T])$.

Note that hiding of the encrypted bids and of the random strings by use of the hash function H prevents anyone from gaining any knowledge of the data prior to time T. In particular, neither the notaries nor the auctioneer have any meaningful information.

Step 3. At time T, the *AU* posts all the received commitments $\mathbf{Com}_1, \ldots, \mathbf{Com}_k$ on the bulletin board, as well as a random bit string S_{AU} of length M. *AU* also creates a number of test sets $TS_1, TS_2, \ldots, TS_K$, where K is a multiple of k, e.g., $K = 80k$. He signs and posts the test sets on the bulletin board.

Step 4. Between time T and $T+1$ any Bidder B_i who has a receipt R_i for a bid which is not posted, can appeal her non-inclusion, resorting to her notaries if she has used them.

Step 5. After time $T+1$, every B_i sends to *AU* her encrypted bid $C_i = E(\mathbf{Bid}_i, r_i)$ as well as her random string S_i. After time $T+1$, *AU* posts the encrypted bids, $C_1, \ldots, C_k$, and the random strings, $S_1, \ldots, S_k$, on the bulletin board. Every Bidder B_i can verify, for any bidder B_j, that the posted value $\mathbf{Com}_j$ corresponds to the ciphertext C_j and the random data string S_j. In case of discrepancies she protests. This check can be performed simply by computing $H(C_j)$, $H(S_j)$, and checking the digital signature $\mathbf{Sign}_j(H(C_j), H(S_j), ID)$.

To discourage *AU* from decrypting and observing some bids after time T and sending instructions to a favored bidder (for instance, instructing the bidder not to unlock her bid), we summarize two solutions. First, bidders who get such a warning and consequently refuse to unlock their bids before time $T+1$ could be obligated to pay a large *fine* to a disinterested third party, such as one of the notaries in the auction. Thus, with this view the notaries not only act on behalf of a bidder in providing a witness to ensure that their own bids are respected by the auctioneer, but notaries also act on behalf of a bidder in ensuring that other bidders must follow through and reveal bids to which they had earlier committed.[13]

Our preferred method (due to its simplicity) is to use delayed decryption key revelation services, *DPrKR*. For this, bidders must submit encryptions of their encrypted bids $E_{DR}(C_i)$ before time $T+1$ to be decrypted at time $T+1$. *AU* posts these on the bulletin board before time $T+1$, and at time $T+1$ both *AU* and verifiers can open them simultaneously to recover the encrypted bids C_i. Incidentally, a completely trustworthy *DPrKR* service could even be used from the beginning of the auction, obviating the need for cryptographic commitments.

Step 6. Using the decryption key ϕ, *AU* recovers the bids $\mathbf{Bid}_1, \ldots, \mathbf{Bid}_k$. The auctioneer then computes the winner of the auction and the payment according to the auction rules. The auctioneer posts the winner's identity, B_i and then information to define the payment to be made by the winner to the bulletin board. This information about payment can be posted in an encrypted form if the payment is to be kept private from nonwinning bidders. Finally, and most importantly, the auctioneer also posts information that will enable any party to verify that the correct result was implemented. These include proofs of the correctness of the winner and payment, and proofs of the validity of each bid.

3.2 Verification

We now show how any verifier $\mathcal{V}$ (including any of the bidders) can verify on her own that the winner and payment of the auction were determined according to the rules of the auction. This will be done

[12] We recall the random strings S_i XORed together to yield the auction random data X. *AU* must specify here the method used to extract a permutation of test sets from X before *AU* sees X so that everyone knows *AU* is revealing a truly random selection of test sets.

[13] If fines are used to enforce nonrepudiation, then notaries must be mandatory for all bidders, and fines must go to the notaries, or to another designated party *other than* the auctioneer or seller. This is because if the auctioneer were to receive the proceeds of such a fine, such a fine would be meaningless as a deterrent to the auctioneer's collusion with a bidder.

in a "zero knowledge" fashion, that is, without revealing anything about the value of any bid except that implied by the outcome of the auction. In addition, the auctioneer can choose how much of the outcome is revealed. For example, the proof can validate that an encrypted payment was correctly determined but without revealing any information about the value of the payment.

The class of single-item auctions under consideration (including first-price and second-price auctions) has the property that the winner and payment depend only on the *ordering of the bids*. In the case of a second price (or *Vickrey*) auction, the item is sold to the highest bidder but for the second highest price. This auction has useful theoretical properties: it is a dominant strategy for bidders to report their true willingness to pay, the auction is efficient, and Vickrey auctions with reservation prices[14] are revenue maximizing in symmetric environments when the auctioneer has the same prior information about the value of each bidder before the auction [15]. In the case of a first-price auction, the item is sold to the highest bidder for the highest price.

Take as an example the Vickrey auction and assume, without loss of generality, that AU announces that B_1 is the winning bidder, which is tantamount to the following set of claims:

$$\{\mathbf{Bid}_1 > \mathbf{Bid}_2;\ \mathbf{Bid}_2 \geq \mathbf{Bid}_3;\ \ldots;\mathbf{Bid}_2 \geq \mathbf{Bid}_k\} \tag{3}$$

Note that the encrypted values

$$\{C_1,\ldots,C_k\} = \{E(\mathbf{Bid}_1,r_1),\ldots,E(\mathbf{Bid}_k,r_k)\}, \tag{4}$$

were posted in Step 5 of the protocol. To prove the claims, it suffices to show that each C_i is an encryption of a valid bid $0 \leq \mathbf{Bid}_i < 2^t < n/2$ for all i, and that

$$\{C_1 \rhd C_2, C_2 \unrhd C_3, \ldots, C_2 \unrhd C_k\} \tag{5}$$

Verifier $\mathcal{V}$ verifies these $2k-1$ claims in a zero knowledge fashion using the tools described above, which enables verification of the winner, item allocation, and payment as described in the next paragraphs.

Recall that the auctioneer had posted $2k$ groups of 40 test sets in Step 3. He creates proofs for each of the first k claims using k of these groups of 40 test sets, one for each claim. He reveals all encryptions for the subgroup of 20 test sets determined by the random string X and the random method posted in Step 1 of the auction. With each of the 20 other test sets AU performs the computation described in Section 2.4.6 (**Range Protocol**) and posts it on the bulletin board. $\mathcal{V}$ can verify that all the revealed test sets are valid, that their indices were chosen correctly, and that the k posted computations are of the form (2). This verifies the first k claims. In addition, AU posts proofs for the $k-1$ claims that $\mathbf{Bid}_1 > \mathbf{Bid}_2$ and $\mathbf{Bid}_2 \geq \mathbf{Bid}_i, 2 < i \leq k$ by using $k-1$ groups of 40 additional test sets for each inequality using the methods described in Section A.4.

This ordering of bids is used to verifier the winner as the bidder with identity corresponding to submitted bid $E(\mathbf{Bid}_1)$, and the item is allocated to this bidder. In a Vickrey auction, the payment to be made by the winner is $\mathbf{Bid}_2$ and this can be proved by sending a verifier $\mathcal{V}$ the random help value r_2 from B_2's encrypted bid $C_2 = E(\mathbf{Bid}_2, r_2)$. $\mathcal{V}$ can then verify the correctness of its payment by re-encrypting $\mathbf{Bid}_2$ with r_2 and checking the result is C_2.

4. MULTI-ITEM AUCTIONS

Consider now auctions for multiple identical items. In these auctions, the auctioneer has some number l of available identical items for sale. Real-life examples include large lots of refurbished items on eBay, or U.S. Treasury bills. As before, we will be able to implement a general class of auctions that includes the first-price, uniform-price, and second-price (generalized Vickrey) auctions. We choose to illustrate the framework for *divisible* bids, in which bidders are willing to accept any number of items up to a maximal limit and bid a price per item. We also assume that no winning bids are equal. However, there is nothing about the framework that is limited in this way, and a treatment of tied bids and extensions to "all-or-nothing" bids and "bid curves" will be described in future work.

4.1 Protocol

Step 1. AU posts the auction information on the bulletin board as in Section 3.1. In addition, AU posts the total number of items available, l, and the maximum allocation to any one bidder (if any), $l_{\max}$.

Step 2. Each participating bidder B_i prepares two integer values $(\mathbf{Bid}_i, \mathbf{Qty}_i)$ for each bid she wishes to submit to the auction, where $\mathbf{Bid}_i$ is the amount that she will pay per item and $\mathbf{Qty}_i$ is the maximum number of items desired by B_i. As above, B_i also generates a random bit string S_i and sends it to AU. B_i then encrypts $\mathbf{Bid}_i$ and $\mathbf{Qty}_i$, using AU's public Paillier key n, as $E(\mathbf{Bid}_i)$ and $E(\mathbf{Qty}_i)$ and commits by sending AU and her notaries, if used, commitments

$$\mathbf{Com}_i = [H(E(\mathbf{Bid}_i)), H(E(\mathbf{Qty}_i)), H(S_i), ID], \tag{6}$$

and digital signature $\mathbf{Sign}_i(\mathbf{Com}_i)$. AU issues a receipt for these commitments and publishes them on the bulletin board in accordance with our standard protocol.

Step 3. As above, at time T AU posts received commitments, his random string S_{AU}, and test sets on the bulletin board. The number of test sets will depend on the type of the auction and the payment calculation; these numbers are detailed in Section 5.

Step 4. As above, bidders have between time T and $T+1$ to appeal non-inclusion, which may involve resorting to the commitments sent to any notaries.

Step 5. As above, bidders reveal their encrypted bids and quantities $E(\mathbf{Bid}_i)$ and $E(\mathbf{Qty}_i)$, as well as their strings S_i, between time T and $T+1$, which AU publishes on the bulletin board. All bidders can check that the revealed values correspond with earlier commitments.

Step 6. AU privately recovers bids $\mathbf{Bid}_i$ and quantities $\mathbf{Qty}_i$ using private key ϕ, and uses the information to compute the correct outcome of the auction. An important notion in a multi-item auction is that of the *threshold bid index*, α. This is defined such that bidders $B_\alpha, \ldots, B_k$ do not receive any items. The sum of the quantities associated with winning bids $\mathbf{Bid}_1, \ldots, \mathbf{Bid}_{\alpha-1}$ is greater than or equal to the number of available items l, and this is not true for a smaller threshold index. Thus all bidders B_i, such that $i < \alpha$, are

[14] In a Vickrey auction with a reservation price, in addition to bids $\mathbf{Bid}_1, \ldots, \mathbf{Bid}_k$ there is a price rp_s from the seller. This is handled just as a bid within the auction. The item is sold to the highest bidder if the maximal bid is at least rp_i but goes unsold otherwise. (Think of this as "selling back to the seller".) When sold, the payment is the maximal value of the second highest bid and the reservation price. Note that because the seller must commit to her reservation price just like any other bidder there is no danger of shill bidding.

winners. The threshold winner $\alpha-1$ may receive some subset of her total demand. Formally, threshold index α is defined so that:

$$\left[\sum_{i=1}^{\alpha-2}\mathbf{Qty}_i < l\right] \wedge \left[\sum_{i=1}^{\alpha-1}\mathbf{Qty}_i \geq l\right] \tag{7}$$

Note that we have assumed here that there are enough bidders to cover all of the supply. This can be handled without loss of generality, by also introducing a single dummy bid at zero price for all supply, l. In addition to determining α, and thus the winners in the auction, *AU* also posts proofs of the identity of the winner(s) and their allocations on the bulletin board, as well as proofs of the validity of each bidder's bid and quantity. He also computes proofs of correctness of each winner B_i's payment. If public verification of payments is required, *AU* posts these correctness proofs on the bulletin board, along with the random help values needed to decrypt the payments. If the payments are to remain secret, he publishes the proofs on the bulletin board but sends the random help values privately to each winner.

4.2 Verification

The verification step in a multi-item auction is more complex than for the single item auction, but relies largely on the same cryptographic primitives used in the simpler single-item case. Each verification can be done in a zero knowledge fashion, revealing no information beyond that implied by the outcome of the auction.[15]

As before, *AU* first publicly proves the minimum *bid-ordering information*, that all winning bids are strictly greater than the threshold bid $\mathbf{Bid}_\alpha$, i.e., $\mathbf{Bid}_i > \mathbf{Bid}_{\alpha-1}$ for all $i < \alpha-1$ and $\mathbf{Bid}_{\alpha-1} > \mathbf{Bid}_j$ for all $j \geq \alpha$. This reveals only minimum public information about the value of the bids; the same information that is implied by the outcome. *AU* will also prove that the bid values are valid and without wraparound. (See Section 2.4.6 for an explanation of wraparound.)

In addition, *AU* must also prove that the *quantities* of the items were encrypted correctly, i.e., without wraparound. We assume that $l < 2^t < n/2$ for number of available items l and test set size parameter t. *AU* first proves that no bidder has submitted a quantity greater than a specified maximum allowed allocation $l_{\max} \leq l$. To do this, *AU* first encrypts $E(l,1)$ and $E(l_{\max},1)$; a random help value 1 is used so that anyone can verify those encryptions. *AU* then proves $E(\mathbf{Qty}_i) \trianglelefteq E(l_{\max},1)$ for all $1 \leq i \leq k$. Next, *AU* can use encryptions of various sums of quantities to prove the correctness of the threshold bid index α. Paillier's homomorphic encryption system allows for a zero-knowledge proof that a ciphertext represents the encrypted value of the sum of two encrypted values; in particular, $\prod_{i=1}^{\alpha-2} E(\mathbf{Qty}_i) = E(\sum_{i=1}^{\alpha-2}\mathbf{Qty}_i)$. Given this, *AU* can establish Eq. 7 over the encrypted quantities:

$$\left[E(\sum_{i=1}^{\alpha-2}\mathbf{Qty}_i) \lhd E(l)\right] \wedge \left[E(\sum_{i=1}^{\alpha-1}\mathbf{Qty}_i) \trianglerighteq E(l)\right] \tag{8}$$

4.2.1 Payment

In a *first-price* auction, the auctioneer can prove a payment to a third party by revealing the random help value used to encrypt winner B_1's bid, and optionally the value $\mathbf{Bid}_1$ itself. A verifier can simply check that the bid value corresponds with the encrypted value submitted by the bidder. Similarly, in a *uniform-price* auction, whereby every bidder pays the bid price of the losing threshold bidder $B_{\alpha-1}$, then *AU* can provide a public proof by revealing $\mathbf{Bid}_{\alpha-1}$ via the help value used by $B_{\alpha-1}$. The uniform price auction is an approximation to a Vickrey auction in this setting.[16]

We turn our attention to proving the correctness of prices in a generalized Vickrey auction, or *Vickrey-Clarke-Groves* (VCG) mechanism for this multi-item setting [15]. In a VCG mechanism the number of items are allocated according to the price bid but the actual payment for each winner depends on others' bids. The Vickrey payment for bidder B_i is defined as:

$$p_{\text{vcg},i} = \mathbf{Qty}_i^* \cdot \mathbf{Bid}_i - [V(B) - V(B_{-i})], \tag{9}$$

where $V(B)$ is the total revenue in the auction with all bidders, $V(B_{-i})$ is the total revenue in the marginal economy with bidder B_i removed, and $\mathbf{Qty}_i^*$ denotes the quantity allocated to bidder i in the auction. This has a simple interpretation: a bidder's payment is determined as *the greatest amount other (displaced) bidders would have paid for the same items had B_i not been participating in the auction.*

We require a proof to establish the correctness of this payment. Let $\mathbf{Qty}_j^{-i}$ denote the quantity awarded to bidder B_j in the marginal auction without bidder B_i. For a non-marginal winner, i.e., $i < \alpha-1$, her VCG payment is:

$$\begin{aligned}\mathbf{Qty}_i^* \cdot \mathbf{Bid}_i - \left[\mathbf{Qty}_i^* \cdot \mathbf{Bid}_i + \sum_{j\neq i, j\leq\alpha-1}\mathbf{Qty}_j^* \cdot \mathbf{Bid}_j\right] \\ + \sum_{j\neq i, j\leq\beta_i-1}\mathbf{Qty}_j^{-i} \cdot \mathbf{Bid}_j = \left[\sum_{\alpha-1<j\leq\beta_i-1}\mathbf{Qty}_j^{-i} \cdot \mathbf{Bid}_j\right] \\ + [\mathbf{Qty}_{\alpha-1}^{-i} \cdot \mathbf{Bid}_{\alpha-1} - \mathbf{Qty}_{\alpha-1}^{*} \cdot \mathbf{Bid}_{\alpha-1}]\end{aligned} \tag{10}$$

For the marginal winner, $i = \alpha-1$, her VCG payment is:

$$\begin{aligned}\mathbf{Qty}_i^* \cdot \mathbf{Bid}_i - [\mathbf{Qty}_i^* \cdot \mathbf{Bid}_i + \sum_{j\neq i, j<\alpha-1}\mathbf{Qty}_j^* \cdot \mathbf{Bid}_j] \\ + \sum_{j\neq i, j\leq\beta_i-1}\mathbf{Qty}_j^{-i} \cdot \mathbf{Bid}_j = \sum_{\alpha-1<j\leq\beta_i-1}\mathbf{Qty}_j^{-i} \cdot \mathbf{Bid}_j\end{aligned} \tag{11}$$

Thus, the VCG payment by bidder B_i is a linear combination of the product of the bid price and allocated quantity to bidders displaced by bidder B_i from the winning allocation. In the case of a non-marginal bidder, this computation also accounts for the effect on the allocation to bidder $\alpha-1$.

Consider the following verifiable proof structure for the term $\sum_{\alpha-1<j\leq\beta_i-1}\mathbf{Qty}_j^{-i} \cdot \mathbf{Bid}_j$ that is common to both kinds of winners:

Step 1. In generating the proof, *AU* must first establish a bid ordering for the marginal auction without B_i, i.e., prove that β_i is the correct threshold bid index by showing $\mathbf{Bid}_j > \mathbf{Bid}_{\beta_i-1}$ for

[15] In the method presented, the verifier $\mathcal{V}$ learns the *number* of bids required to compute a Vickrey payment in the marginal economy $\mathbf{E}(B_{-i})$. We can get around this through using multiple "thresholds" and zero allocations; we reserve a full discussion of this detail for future work.

[16] It generates the same payment as in the Vickrey auction to winning bidders $i < \alpha-1$, as long as the threshold bidder has enough spare demand to cover the allocated capacity of any winner. The payment by the threshold winner $B_{\alpha-1}$ is always larger than in the Vickrey scheme, though.

$j \neq i, j < \beta_i - 1$ and $\mathbf{Bid}_{\beta_i - 1} > \mathbf{Bid}_j$ for $j \geq \beta_i$, this can be done as in the main auction. Second, AU must prove that bidder $\beta_i - 1$ is the threshold winner in this auction, by proving the analogue to Eq. 7. Third, AU must publish encrypted values $\mathbf{Pay}_j = \mathbf{Qty}_j \cdot \mathbf{Bid}_j$ for all $j > \alpha_i, j < \beta_i - 1$ (and similarly for the new marginal bidder, $\mathbf{Pay}_{\beta_i - 1} = \mathbf{Qty}^{-i}_{\beta_i - 1} \cdot \mathbf{Bid}_{\beta_i - 1}$), and prove the correctness of all of these ciphertexts. This requires proofs of *correct multiplication*, as described in Appendix A. The proof of $\mathbf{Pay}_{\beta_i - 1}$ in turn requires a proof of the quantity allocated $\mathbf{Qty}^{-i}_{\beta_i - 1}$ to this bidder, via a proof that a published ciphertext is the encrypted value of $l - \sum_{j \neq i, j < \beta_i - 1} \mathbf{Qty}_j$. Fourth, AU must publish the encrypted value of the sum of these payments and a proof of its correctness.

Step 2. A verifier $\mathcal{V}$ can independently compute the encrypted Vickrey payment as above and check the correctness of the proof.

Step 3. AU reveals the random help value in the resulting encrypted Vickrey payment to $\mathcal{V}$, who decrypts using that value and verifies it is correct by re-encryption.

The verifier $\mathcal{V}$ now knows that B_i's Vickrey payment is correct knowing nothing more about any bidder's bid value than can be derived from the definition of Vickrey payments.[17]

The additional term, $[\mathbf{Qty}^{-i}_{\alpha-1} \cdot \mathbf{Bid}_{\alpha-1} - \mathbf{Qty}^{*}_{\alpha-1} \cdot \mathbf{Bid}_{\alpha-1}]$ can be determined in the case that bidder i is the threshold winner and $i = \alpha - 1$ in an analogous fashion. Encrypted values of the allocation quantities received by bidder i in the main auction and in the marginal auction, i.e., $\mathbf{Qty}^{*}_{\alpha-1}$ and $\mathbf{Qty}^{-i}_{\alpha-1}$ can be established via subtraction from total items l of the total allocation to other bidders. Then, a ciphertext for the difference, $\mathbf{Qty}^{-i}_{\alpha-1} - \mathbf{Qty}^{*}_{\alpha-1}$, and then the product $(\mathbf{Qty}^{-i}_{\alpha-1} - \mathbf{Qty}^{*}_{\alpha-1})\mathbf{Bid}_{\alpha-1}$ can be published and proved.

5. EMPIRICAL RESULTS

We implemented Paillier encryption and test set verification in C++ using the LiDIA number theory package [17] on a commodity Linux workstation with a Pentium 4 2.8 GHz processor.

The greatest computational cost in our protocol is the construction and verification of test sets, and in particular the exponentiation of random help values (r^n) required to encrypt or (verifiably) decrypt a value. This preparation cost dominates all other computation; for example, to sort one million random 64-bit bids takes less than one second on our system. In a single-item auction, the auctioneer can prepare for an auction of 100 bidders in about two hours, and each verifier can independently verify the auctioneer's proofs of correctness in less than half an hour. Both preparation and verification scale linearly and are easily parallelized. Thus, with modest distributed computation, even a multi-item auction with ten thousand bidders can be prepared in a few hours and verified in reasonable time.

We present data for both 1024- and 2048-bit symmetric public encryption keys, which are considered safe until 2010 and 2030, respectively [12]. Because the lifetime of a security key is based on the difficulty of breaking it on available computing power, we claim that, for the most part, an auction with "5-year" security at any point in time will take about the same amount of time as it does today, as improvements in computing power for breaking keys are likely to be comparable to those in encryption.[18]

Table 1 shows the time it takes to compute various cryptographic operations on our test machine. We observe that the time required to prepare or verify a test set is essentially that required by the encryption and decryption. All test sets represent 2^{34} discrete values.

Table 1: Time to perform basic operations

Operation	**Time** (s.) (1024-bit)	**Time** (s.) (2048-bit)
Computation of r^n	0.045	0.287
Encryption	0.045	0.287
Decryption with r	0.045	0.287
Decryption with ϕ	0.014	0.089
Decryption with r^n	0.000	0.001
Constructing a TS	3.01	19.32
Verifying a TS	3.00	19.30
Proving $0 \leq x < 2^t$ given TS	0.001	0.001
Verifying proof of $0 \leq x < 2^t$	0.070	0.41

For a single item auction of k bidders, the auctioneer must produce k proofs of valid bids (i.e. $\mathbf{Bid}_i < 2^t$ for small t; we use 34), and $k-1$ proofs of comparisons to prove the ordering of the outcome. Using the bulk verification method suggested in Appendix B, such an auction requires $10 \cdot (2k-1)$ test sets, plus 25% for the test sets that will be revealed to prove the test sets are valid. This gives us an upper bound of $25k$ test sets required to conduct a trustworthy single-item auction.

For a multi-item auction with payments based on one bid (e.g. first-price or second-price), we need only add to the above k proofs $\mathbf{Qty}_i < 2^t$, k comparisons $\mathbf{Qty}_i < l_{\max}$, and 2 comparisons to prove Equation 7. This means we need about double the number of test sets, $4k+1$, to conduct such a multi-item auction; about $50k$ test sets are needed for trustworthiness. We list the time taken to prepare these test sets and correctness proofs in Table 2.

Table 2: Time to prepare and verify auctions

Operation	**Number of Bids**		
	100	1000	10000
		Single-item Auctions	
Preparation (1024-bit)	2.1 hr	21 hr	8.7 days
Verification (1024-bit)	25 min	4.2 hr	42 hr
Preparation (2048-bit)	13.4 hr	5.6 days	56 days
Verification (2048-bit)	2.7 hr	27 hr	11 days
		Multi-item Auctions	
Preparation (1024-bit)	4.2 hr	42 hr	17.5 days
Verification (1024-bit)	52 min	8.7 hr	3.6 days
Preparation (2048-bit)	27 hr	11.2 days	112 days
Verification (2048-bit)	5.4 hr	54 hr	22 days

For verified VCG payments in multi-item auctions (Section 4.2.1), we also require proofs of multiplications for at most $2k+1$ products, namely, $\leq k$ proofs of the products $\mathbf{Qty}_i \cdot \mathbf{Bid}_i$ and $k+1$ proofs of the products of the partial allocation to the threshold bidder for the main economy $\mathbf{E}(B)$ and up to k marginal economies (that is,

[17] See Footnote 15.

[18] Of course, if efficient algorithms to solve the composite residuosity problem or factor large composites are discovered, our claim does not hold.

excluding bidder B_i) $\mathbf{E}(B_{-i})$. Each proof of a product requires 4 exponentiations for creating the MTS ("multiplication test set") and 6 exponentiations to verify it. To achieve a reasonably small probability of error, we need to repeat the multiplication proof 80 times ($\frac{3}{4}^{80} \approx 10^{-10}$). Thus each proof requires 320 exponentiations to create and 480 to verify. Table 3 shows time required, again on a P4 2.8 GHz processor, to verify Vickrey payments in the worst case for various sizes of multi-item auctions. These computations are required *in addition* to the above computations for verifying prices and quantities.

Table 3: Verification of Vickrey payments for multi-item auctions

Operation	**Number of Bids**		
	100	1000	10000
Preparation (1024-bit)	48 min	8 hr	3.3 days
Verification (1024-bit)	72 min	12 hr	5 days
Preparation (2048-bit)	5.1 hr	51 hr	21 days
Verification (2048-bit)	7.7 hr	77 hr	32 days

6. CONCLUSIONS AND FUTURE WORK

We have presented a new protocol for sealed-bid auctions that guarantees trust and preserves a high level of secrecy, yet is practical enough to run efficiently on commodity hardware and be accepted in the business community. Because we focus on proofs of correctness and secrecy during the auction, an auctioneer can still compute optimal results efficiently and publish efficiently verifiable proofs of those results. Our protocol rests on sound cryptographic foundations, and lends itself to straightforward extensions to further types of auctions, including support for all-or-nothing bids, bid curves, and full combinatorial auctions; we intend to pursue these extensions in later work, in addition to completing a full description of tiebreaking and bulk verification of test sets. We believe that our practical test-set model will extend to other areas of privacy, including electronic transactions, trading systems, privacy-preserving open outcry markets, and zero-knowledge public verification of private data.

7. ACKNOWLEDGMENTS

This work was supported in part by an Alfred P. Sloan Foundation Award to Parkes and NSF grant CCR-0205423 to Rabin. The authors thank Michael Hamburg, Alex Healy, and Adam Juda for helpful comments and information.

8. REFERENCES

[1] M. Abe and K. Suzuki. M+1-st price auction using homomorphic encryption. In *Proc. Public Key Cryptography*, 2002.

[2] V. Afualo and J. McMillan. Auctions of rights to public property. In P. Newman, editor, *The New Palgrave Dictionary of Economics and the Law*. Palgrave Macmillan, 2004.

[3] O. Baudron and J. Stern. Non-interactive private auctions. In *Proc. Financial Cryptography*, 2001.

[4] P. G. Bradford, S. Park, and M. H. Rothkopf. Protocol completion incentive problems in cryptographic Vickrey auctions. Technical Report RRR 3-2004, Rutgers Center for Operations Research (RUTCOR), 2004.

[5] F. Brandt. How to obtain full privacy in auctions. Technical report, Carnegie Mellon University, 2005.

[6] F. Brandt and T. Sandholm. (Im)possibility of unconditionally privacy-preserving auctions. In *Proc. 3rd Int. Conf. on Autonomous Agents and Multi-Agent Systems*, pages 810–817, 2004.

[7] M. Burmester, E. Magkos, and V. Chrissikopoulos. Uncoercible e-bidding games. *Electronic Commerce Research*, 4:113–125, 2004.

[8] I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In *Proc. Public Key Cryptography 01*, 2001.

[9] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Information Theory*, IT-31(4):469–472, 1985.

[10] W. J. Elmaghraby. Pricing and auctions in emarketplaces. In *Handbook of Quantitative Supply Chain Analysis: Modeling in the E-Business Era*. Kluwer Academic Publishers, Norwell, MA, 2004.

[11] M. K. Franklin and M. K. Reiter. The design and implementation of a secure auction server. *IEEE Transactions on Software Engineering*, 22(5):302–312, 1986.

[12] D. Giry and P. Bulens. Cryptographic key length recommendation. `http://www.keylength.com`, 2006.

[13] M. Harkavy, J. D. Tygar, and H. Kikuchi. Electronic auctions with private bids. In *Proc. Third USENIX Workshop on Electronic Commerce*, pages 61–74, 1998.

[14] H. Kikuchi. (m+1)st price auction protocol. In *Proc. Financial Cryptography*, 2001.

[15] V. Krishna. *Auction Theory*. Academic Press, 2002.

[16] M. Kumar and S. I. Feldman. Internet auctions. In *Proc. 3rd USENIX Workshop on Electronic Commerce*, 1998.

[17] LiDIA-Group. LiDIA — a library for computational number theory. *TU Darmstadt*, 2001.

[18] H. Lipmaa, N. Asokan, and V. Niemi. Secure Vickrey auctions without threshold trust. In *Proc. 6th International Conference on Financial Cryptography (FC 2002)*, pages 87–101, 2002.

[19] D. Lucking-Reiley. Vickrey auctions in practice: From nineteenth century philately to twenty-first century e-commerce. *J. of Economic Perspectives*, 14:182–192, 2000.

[20] J. McMillan. Selling spectrum rights. *Journal of Economic Perspectives*, 8:145–162, 1994.

[21] P. Milgrom. *Putting Auction Theory to Work*. Cambridge University Press, 2004.

[22] M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *Proc. First ACM Conf. on Elec. Commerce*, pages 129–139, 1999.

[23] National Institute of Standards and Technology. Secure hash signature standard. `http://csrc.nist.gov/publications/fips/fips180-2/fips180-2withchangenotice.pdf`, 2002. FIPS PUB 180-2.

[24] P. Paillier. *Cryptographie à Clé Publique Basée sur la Résiduosité de Degré Composite*. PhD thesis, École Nationale Supérieure des Télécommunications, 1999.

[25] P. Paillier. Public-key cryptosystems based on composite residuosity classes. In *Proc. EUROCRYPT '99*, pages 223–239, 1999.

[26] M. O. Rabin. Digitalized signatures. In *Foundations of Secure Computing*, pages 155–166. Academic Press, New York, 1978.

[27] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[28] V. Rijmen and P. S. L. M. Barreto. The Whirlpool hash function. `http://paginas.terra.com.br/informatica/paulobarreto/WhirlpoolPage.html`.

[29] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. Technical Report MIT/LCS/TM-82, MIT Laboratory for Computer Science, 1977.

[30] R. L. Rivest, A. Shamir, and D. A. Wagner. Time-lock puzzles and timed release crypto. Technical Report MIT/LCS/TR-684, MIT, 1996.

[31] M. H. Rothkopf, T. J. Teisberg, and E. P. Kahn. Why are Vickrey auctions rare? *Journal of Political Economy*, 98:94–109, 1990.

[32] T. Sandholm, D. Levine, M. Concordia, P. Martyn, R. Hughes, J. Jacobes, and D. Begg. Changing the game in strategic sourcing at Procter & Gamble: Expressive competition enabled by optimization. *Interfaces*, 36(1):55–68, 2006.

[33] S. W. Smith. *Trusted Computing Platforms: Design and Applications*. Springer, New York, 2005.

[34] S. G. Stubblebine and P. F. Syverson. Fair on-line auctions without special trusted parties. In *Proc. of Financial Cryptography*, 1999.

[35] K. Suzuki and M. Yokoo. Secure generalized Vickrey auction using homomorphic encryption. In *Proc. Financial Cryptography*, 2003.

[36] B. L. Xiaofeng Chen, Kwangjo Kim. Receipt-free electronic auction schemes using homomorphic encryption. In *ICISC*, 2003.

[37] M. Yokoo and K. Suzuki. Secure multi-agent dynamic programming based on homomorphic encryption and its application to combinatorial auctions. In *Proc. First Int. Conf. on Autonomous Agents and Multiagent Systems*, 2002.

APPENDIX

A. PAILLIER ENCRYPTION

A.1 Public/Private Keys

Paillier encryption uses an encryption key $n = p \cdot q$, where p and q are large primes. The decryption key is based on the factorization of n, $\phi = \varphi(n) = (p-1)\cdot(q-1)$. $\varphi(n)$ is Euler's totient function, the number of integers relatively prime to n.

A.2 Encryption

To encrypt a plaintext x, first compute a random value r from the range $[1, n-1]$ such that $\mathbf{gcd}(r,n) = 1$, then recall that $(1+n)^x \equiv (1+xn) \pmod{n^2}$ and encrypt as

$$E(x,r) = (1+xn)\cdot r^n \pmod{n^2} \qquad (12)$$

A.3 Decryption

To decrypt $C = E(x)$, given decryption key $\phi = (p-1)(q-1)$, observe that $r^{n\cdot\phi} \equiv 1 \pmod{n^2}$ by Euler's Totient Theorem, and

$$\begin{aligned} C^{\phi} &= (1+n)^{x\cdot\phi} r^{n\cdot\phi} \pmod{n^2} \\ &= 1 + x\phi n \pmod{n^2} \\ x &= \frac{(C^{\phi}-1)/\phi \mod n^2}{n} \end{aligned} \qquad (13)$$

We did not use this method when obtaining our results in Section 5. Instead, we used a more efficient algorithm involving precomputation and Chinese remaindering, as described in Paillier's Ph.D. thesis [24].

A.3.1 Decryption with random help value r

It is also possible for some $\mathcal{P}$ who knows the r used to encrypt $C = E(x,r)$ to show $\mathcal{V}$ that x is the unique decryption of C by revealing r. $\mathcal{P}$ may know r either by having encrypted all the values used to compute C or by computing it via the decryption key ϕ. To recover x, $\mathcal{V}$ computes

$$x = \frac{(C\cdot r^{-n} \mod n^2) - 1}{n} \qquad (14)$$

A.3.2 Uniqueness of Encryptions

Paillier encryption constitutes a bijection from $(\mathbb{Z}_n \times \mathbb{Z}_n^*) \rightarrow \mathbb{Z}_{n^2}^*$ [25].[19] Thus any integer in $\mathbb{Z}_{n^2}^*$ represents a single valid encryption of an integer $x \in \mathbb{Z}_n$ with random help value $r \in \mathbb{Z}_n^*$. Consequently, if $C = E(x,r)$, $C \neq E(x',r')$ for any $x' \in \mathbb{Z}_n$ and $r' \neq r$.

$\mathcal{P}$ can attempt to cheat by providing a different random help value r'. Using r' instead of r in (14) will yield a different but invalid "decryption" x'. $\mathcal{V}$ must therefore verify the provided value r' is consistent with the known encryption C. This is done by re-encrypting the derived value x' as $C' = E(x',r')$ and rejecting r' unless $C' = C$.

A.4 Mathematical Operations on Encrypted Values

The following definitions apply to any values encrypted as above, such as bids, deposit amounts, or desired quantities. These properties are due to the homomorphic properties of Paillier's encryption scheme [25]. In these definitions we refer to a prover $\mathcal{P}$ who has the decryption key or all random help values for encrypted data, (generally the auctioneer), and a verifier $\mathcal{V}$ who does not.

Addition. Addition of two encrypted values:

$$E(x)\cdot E(y) = E(x+y) \pmod{n^2}$$

Adding a constant k to an encrypted value x is easily done by encrypting k with the random help value 1 and multiplying the two encryptions.

$$E(x)\cdot(1+kn) = E(x+k) \pmod{n^2}$$

Multiplication or division by a constant. Division is only possible when k is invertible mod n^2.[20]

$$\begin{aligned} (E(x))^k &= E(x\cdot k) \pmod{n^2} \\ (E(x))^{1/k} &= E(x/k) \pmod{n^2} \end{aligned}$$

[19] $\mathbb{Z}_n$: the set of integers $[0,n)$
$\mathbb{Z}_n^*$: the subset of $\mathbb{Z}_n$ relatively prime to n

[20] This is no impediment, as finding a noninvertible k is tantamount to breaking the security key.

Negation. Implied by multiplication by a constant.

$$(E(x))^{-1} = E(-x) \pmod{n^2}$$

Comparison to a constant k. $\mathcal{P}$ can prove any encryption $C = E(k,r)$ is an encryption of k by revealing the help value r used to encrypt C. $\mathcal{V}$ then verifies that $(1+nk)r^n = C \pmod{n^2}$, because

$$E(k,r) = (1+n)^k \cdot r^n \pmod{n^2} \tag{15}$$

This is of particular interest when $k = 0$. We remark that no encryption of a value other than zero is an n^{th} residue mod n^2.[21]

Equality comparison. Given two ciphertexts $C_1 = E(x_1,r_1)$ and $C_2 = E(x_2,r_2)$, $\mathcal{P}$ can prove $x_1 = x_2$ without revealing any additional information—most importantly, the value of x_1 or x_2. Both $\mathcal{P}$ and $\mathcal{V}$ compute $C' = C_1 \cdot C_2^{-1} \pmod{n^2} = E(x_1 - x_2, r_1/r_2) = E(0, r_1/r_2)$. $\mathcal{P}$ then proves C' is an encryption of zero as above by revealing r_1/r_2.

Inequality comparison. Given two ciphertexts $C_x = E(x)$ and $C_y = E(y)$, $\mathcal{P}$ can show $x > y$ and $x \geq y$. Because our values x and y are integers mod n^2, we can prove $x > y$ by showing $x \geq y+1$, provided $y \neq n-1$. Due to the homomorphic properties of Paillier encryption, $E(x+1) = E(x) \cdot (n+1) \pmod{n^2}$, and so adding 1 to a value in its encrypted form is trivial. Thus, all ordering comparisons can be reduced to the ability to prove $x \geq y$. We first specify that x and y must be in the range $[0, 2^t)$ for $2^t < n/2$. This can be proven as in Section 2.4.6. Then, to prove $x \geq y$, both $\mathcal{P}$ and $\mathcal{V}$ calculate $E(x-y) = E(x) \cdot E(y)^{-1} \pmod{n^2}$, and $\mathcal{P}$ proves $0 \leq (x-y) < 2^t < n/2$ from $E(x-y)$. If in fact $x < y$, then $(x-y)$ will wrap around mod n^2 so that $(x-y) \geq n/2$ and no such proof is possible. This principle is also described in Section 2.4.6.

Proof of multiplication of two values. Because Paillier encryption does not enable the secrecy-preserving multiplication of two encrypted values as it does addition, we require a method that allows a prover $\mathcal{P}$ with three plaintexts u, v, and w such that $uv = w \pmod{n}$ to prove this fact to a verifier $\mathcal{V}$ who has Paillier encryptions $E(u)$, $E(v)$, and $E(w)$, respectively. Dåmgard et al. [8] propose another solution to this; the solution we present is in the spirit of our other cryptographic primitives.

DEFINITION 2. *A* Multiplication Test Set *(MTS) for $E(u,r)$, $E(v,s)$, and $E(w,t)$ is a set of 8 elements:*

$$\{E(u_1,r_1), E(u_2,r_2), E(v_1,s_1), E(v_2,s_2),$$
$$E(w_{i,j}) = E(u_i v_j, p_{i,j}) \mid i,j \in \{1,2\}\}$$

where $u = u_1 + u_2 \pmod{n}$ and $v = v_1 + v_2 \pmod{n}$.

In each MTS, u_1 and v_1 are chosen uniformly at random from $\mathbb{Z}_n$; u_2 and v_2 are correspondingly defined, as above, so that $u = u_1 + u_2 \pmod{n}$ and likewise for v.

Clearly, if given encryptions as in MTS and

$$w_{1,1} + w_{1,2} + w_{2,1} + w_{2,2} = w \pmod{n} \tag{16}$$

then in fact $uv = w \pmod{n}$. But for $\mathcal{P}$ to prove and for $\mathcal{V}$ to verify all the relationships included in the MTS entails revealing u_1, u_2, v_1, and v_2, consequently revealing u and v. Thus we adopt for an interactive proof the following challenge and partial revelation proof. $\mathcal{P}$ constructs and sends MTS. $\mathcal{V}$ randomly chooses a challenge pair (i,j), say, $(1,2)$, and sends it to $\mathcal{P}$. In this case, $\mathcal{P}$ reveals r_1, s_2, and $p_{1,2}$. This allows $\mathcal{V}$ to decrypt $E(u_1)$, $E(v_2)$, and $E(w_{1,2})$, and directly verify that $u_1 \cdot v_2 \equiv w_{1,2} \pmod{n}$. $\mathcal{P}$ further reveals:

$$R = r_1 \cdot r_2 \cdot r^{-1} \pmod{n}$$
$$S = s_1 \cdot s_2 \cdot s^{-1} \pmod{n}$$
$$p = p_{1,1} \cdot p_{1,2} \cdot p_{2,1} \cdot p_{2,2} \cdot t^{-1} \pmod{n}$$

$\mathcal{V}$ by use of R verifies $E(u_1) \cdot E(u_2) \cdot E(u)^{-1} \pmod{n^2} = E(0,R)$, i.e., verifies $u = u_1 + u_2 \pmod{n}$ and similarly $v = v_1 + v_2 \pmod{n}$ via S. Finally, $\mathcal{V}$ verifies $E(w_{1,1}) \cdot E(w_{1,2}) \cdot E(w_{2,1}) \cdot E(w_{2,2}) \cdot t^{-1} \pmod{n^2} = E(0,p)$, thereby verifying that (16) holds.

A moment's thought reveals that if MTS was not proper then the probability of $\mathcal{V}$ uncovering this by the random choice of (i,j) is at least $\frac{1}{4}$. Thus the probability of $\mathcal{P}$ meeting the challenge when $uv \neq w \pmod{n}$ is at most $\frac{3}{4}$. This implies that if m MTS's are used and $\mathcal{P}$ meets all m random challenges then the probability of $\mathcal{P}$ cheating is smaller than $(\frac{3}{4})^m$. In practice, the auctioneer will act as $\mathcal{P}$ and verify the multiplications required to prove the validity of multi-item auction allocations by repeating these zero-knowledge proofs until the desired likelihood of error is achieved.

B. BULK VERIFICATION OF TEST SETS

We have already shown how AU can use a test set to prove both that for any encrypted bids $E(\mathbf{Bid}_1)$ and $E(\mathbf{Bid}_2)$, $\{\mathbf{Bid}_1, \mathbf{Bid}_2\} \leq 2^t$ and $\mathbf{Bid}_1 > \mathbf{Bid}_2$, provided $2^t < n/2$. We also provided a non-interactive proof to allow the validity of test-sets to be established. Here, we improve the computational speed of this "cut and choose" approach for multiple range-of-value proofs by allowing anyone to verify *en masse* a whole collection of test sets, to then be used in proofs of range and ordering of values. Recall that in single-item auctions with k bidders, AU will verify that k bids are in range, and then perform $k-1$ comparisons to prove the correctness of the auction. These auctions require $2k-1$ range-of-value proofs.

Adopting numbers that are appropriate for an auction with 100 bidders and moderate security requirements, we assume for illustration that the auctioneer employs 10 test sets per proof and first creates and posts 2500 (claimed) test sets. For bulk verification we select and reveal 500 test sets uniformly at random in a collection of 2500. The probability that all 500 will be correct and 200 (or more) of the remaining 2000 are incorrect is $< 7 \times 10^{-19}$. We can then prove correctness of each bid or comparison with probability of error $< 10^{-10}$ by drawing 10 of the remaining 2000 test sets uniformly at random and proving correctness on each of them. We can achieve a truly random ordering of the 2500 test sets using the random data string X as in the main description of our method.

[21] To say that x is an n^{th} *residue* $\pmod{m}$ means that there exists some value g such that $x = g^n \pmod{m}$. See also Footnote 6.