Mix and Match: Secure Function Evaluation via Ciphertexts (Extended Abstract)

Markus Jakobsson[1] and Ari Juels[2]

[1] Information Sciences Research Center
Bell Labs
Murray Hill, New Jersey 07974
markusj@research.bell-labs.com
[2] RSA Laboratories
RSA Security Inc.
Bedford, MA 01730, USA
ajuels@rsasecurity.com

Abstract. We introduce a novel approach to general secure multiparty computation that avoids the intensive use of verifiable secret sharing characterizing nearly all previous protocols in the literature. Instead, our scheme involves manipulation of ciphertexts for which the underlying private key is shared by participants in the computation. The benefits of this protocol include a high degree of conceptual and structural simplicity, low message complexity, and substantial flexibility with respect to input and output value formats. We refer to this new approach as *mix and match*.

While the atomic operations in mix and match are logical operations, rather than full field operations as in previous approaches, the techniques we introduce are nonetheless highly practical for computations involving intensive bitwise manipulation. One application for which mix and match is particularly well suited is that of sealed-bid auctions. Thus, as another contribution in this paper, we present a practical, mix-and-match-based auction protocol that is fully private and non-interactive and may be readily adapted to a wide range of auction strategies.

Key words: auction, general secure multiplayer computation, millionaires' problem, secure function evaluation

1 Introduction

Consider the following scenario. Alice and Bob have respective fortunes A and B. They wish to determine who is richer, i.e., whether $A > B$, but do not wish to reveal any additional information about their fortunes. This task is known as the *millionaires' problem* [44]. It is a special instance of the more general setting in which Alice and Bob, or indeed a larger number of players, wish to compute

T. Okamoto (Ed.): ASIACRYPT 2000, LNCS 1976, pp. 162–177, 2000.

the output of a function f on secret inputs without revealing any additional information.

Alice and Bob can take one of several approaches. They might confide their fortunes to a trusted third party charged with the task of determining honestly whether $A > B$ and not leaking any information to either party. Alternatively, they might construct a piece of trusted hardware for the same purpose. It has been known for some time, however, that Alice and Bob can in fact *simulate* a trusted party or device in such a way as to enable secure computation of general functions [44,27]. Computation of this sort involving two or more players is known as *secure multiparty computation*. For general functions f it is known as *general secure multiparty computation* or *secure function evaluation*.

All of the current approaches to general secure multiparty computation rely on the simulation of a circuit C_f for the function f of interest. This circuit is typically viewed as being composed of gates implementing two operators, such as $+$ (modular addition) and $\times$ (modular multiplication), that together allow for the realization of an arbitrary computable function. In nearly every protocol in the literature with robustness against active adversaries, the lynchpin is a cryptographic primitive known as *verifiable secret sharing* (VSS), introduced in [14]. Players distribute their inputs to C_f by dealing shares to other players through a VSS protocol. At any stage in the computation, concealed values are held distributively. To simulate a $+$ gate, players perform local addition of their shares. To simulate a $\times$ gate, they perform an interactive protocol involving multiplication of pairs of shares held by different players.

In this paper, we investigate a different approach to secure function evaluation. Rather than employing multi-player sharing of individual inputs or intermediate computational results, we consider a representation of these values as ciphertexts. We concentrate in particular in this paper on use of the El Gamal cryptosystem [24], although use of other semantically secure cryptosystems, such as Cramer-Shoup [18], is possible. Distribution of trust among the players in our scheme relies on sharing of a single, underlying private decryption key. Players perform the operations required by the computation using well established techniques for distributed manipulation of El Gamal ciphertexts.

A brief sketch of our approach is as follows. Having agreed upon a function f and a circuit representation C_f, the players provide El Gamal ciphertexts of their input bits. Gates in C_f are each represented by a boolean function, such as AND or NOT (although others are possible). For each gate, the players construct a logical table corresponding to the function computed by the gate, the entries in this table consisting of El Gamal ciphertexts. In an initial blinding phase, the players use a primitive known as a *mix network* to blind and perform row-wise permutation of these tables in a distributed fashion. The basis of the subsequent computation phase, which we refer to as *matching*, is a primitive called a *plaintext equality test* ($\mathcal{PET}$). The $\mathcal{PET}$ primitive enables players to determine in a distributed fashion whether two given ciphertexts represent the same plaintext. Players evaluate the circuit C_f iteratively, using $\mathcal{PET}$ to perform table lookups. For each gate, they compare ciphertext input values to ciphertext values in the

corresponding blinded logical table. When the correct row in the table is found, the players obtain an output ciphertext from the third column. Due to the use of blinded permutation, they do not learn the plaintext corresponding to the output value. The output ciphertext is used as input to the next gate (table). We refer to this approach as *mix-and-match* computation.

1.1 Previous Work

The idea of performing secure computation by means of blinded table lookups was essentially the basis for the original proposal of Yao [44], whose two-player scheme was predicated on the hardness of factoring. Goldreich, Micali, and Wigderson [27] generalized the basis of Yao's scheme to use of any one-way trapdoor permutation. The idea behind both approaches in their two-party instantiations is as follows. Alice constructs a circuit C_f using boolean gates represented as randomly permuted, blinded logical tables. Inputs to a gate (table) are randomly generated tags representing different bit values. Each set of tags, representing a given set of inputs to a table, serves as a decryption key for a particular row of the table, and thus a particular output tag for the gate. By means of a 1-2 oblivious transfer protocol, Alice blindly transfers to Bob the tags representing his input values for the circuit, and also sends Bob her table representation of C_f. For each gate in C_f, Bob uses the input tags to decrypt output tags representing the corresponding gate output. He is thereby able to evaluate C_f without further interaction with Alice. See [27] for further details.

Chaum, Damgård, and van de Graaf [12] extend the notion of blinded table mixing and lookup to a multiparty scenario. In their scheme, each player in turn blinds the logical table for a given gate. The basis of this scheme is a homomorphic commitment scheme that enables one player to alter the commitment of another without knowing the correct decommitment.[1] Players provide cut-and-choose proofs of correct behavior. The security of the scheme is unconditional for one player, and for the others is based on the quadradic residuosity problem.

The Chaum *et al.* scheme is not robust against an active adversary, in the sense that such an adversary may corrupt the computation irretrievably or force it to halt. To achieve robustness, the authors recommend incorporation of VSS to enable reconstruction of the commitments in their scheme. Similarly, since the introduction of secure multiparty computation in [27], such protocols have generally employed VSS as a means of enforcing robustness for the computation on each gate. Ben-Or, Goldwasser, and Widgerson [4] and Chaum, Crépeaud, and Damgård [11] introduced the first protocols enabling security against an active adversary in the non-cryptographic model, that is, one in which players are assumed to have unbounded computing power, but cannot eavesdrop on honest players. Their approach has loosely formed the basis of the majority of subsequent work, even some of the most recent and efficient constructions such as [15,26].

[1] Manipulation of homomorphic commitments by n players here in fact yields a kind of (n, n)-VSS protocol in this scheme.

Other work related to our own is that of Franklin and Haber [22]. As we do here, they propose a secure multiparty computation method dependent on manipulation of ciphertexts. In their scheme, the underlying encryption scheme is a special variant of El Gamal. The Franklin and Haber system, however, is only secure against passive adversaries.

While drawing on table-based approaches to secure function evaluation, and particularly on the frameworks presented in [12,22], mix and match offers robustness without the use of VSS on input values or sharing of intermediate values. The mix-and-match approach consequently achieves several benefits unavailable in conventional VSS-based approaches:

1. Mix and match is conceptually very simple.
2. The message complexity for mix and match is quite low. In the random oracle model, the broadcast message complexity is $\mathcal{O}(nN)$ group elements, where n is the number of players and N, the number of gates in the circuit.
3. Sharing in mix and match occurs only at the level of a decryption key, and not for input or intermediate computational values.

This last property means that mix and match has the advantage of natural flexibility in terms of both input and output formats and player participation. For example, players contributing inputs need not even know what players or how many will be performing the computation, and need not themselves participate. Outputs may be made to take the form of ciphertexts under an arbitrary key, with no additional protocol overhead. We note that a similar property emerges in independent work by Cramer, Damgård, and Nielsen [16].

A drawback to mix and match is the fact that the atomic computational unit is the boolean formula, rather than the field operations employed in secure multiparty computation schemes based on [4]. For many functions, therefore, such as threshold signature computation [25], it is probably substantially less efficient than, e.g., [30]. For functions involving intensive bitwise manipulations, however, mix and match is highly competitive. A good example is the millionaires' problem, or natural multi-player extensions such as auction protocols. To highlight this strength, we present a flexible non-interactive auction protocol in this paper based on mix and match. This protocol, as we show, has several practical advantages over state-of-the-art proposals for non-interactive secure auctions.

1.2 Organization

We present model details and definitions in section 2. We describe our mix-and-match scheme in section 3. In section 4, we briefly discuss the literature on auction protocols and outline a non-interactive, fully private auction protocol based on mix and match.

2 Model and Building Blocks

In elaborating mix and match, we consider the cryptographic model of secure multiparty computation. This involves n players, $P_1, P_2, \ldots, P_n$, who are as-

sumed to share an authenticated broadcast channel, and an *adversary* with resources polynomially bounded in all security parameters. We consider an adversary who may corrupt up to $t < n/2$ of these players in an active fashion, i.e., the adversary gains access to their private information, and may govern their behavior in an arbitrary fashion. We assume that the adversary is *static*, that is, she must choose in advance which players she wishes to corrupt. Our results can be extended straightforwardly to more complex adversarial structures. We can achieve security in the mix-and-match protocol reducible to the *Decision Diffie-Hellman* (DDH) assumption (see, e.g., [35]) on the group $\mathcal{G}$ over which the computation takes place. To achieve the best possible efficiency and simplicity here, however, we additionally invoke the random oracle model (see, e.g., [3]). The asymptotic costs presented in this paper assume malicious adversarial behavior.

2.1 Building Blocks

El Gamal cryptosystem: We employ the El Gamal cryptosystem [24] as the basis for our constructions. Encryption in the El Gamal cipher takes place over a group[2] $\mathcal{G}_q$ of prime order q.

Let g be a generator of $\mathcal{G}_q$. This generator is typically regarded as a system parameter, as it may correspond to multiple key pairs. A private encryption key consists of an integer $x \in_U Z_q$, where $\in_U$ denotes uniform random selection. The corresponding public key is defined to be $y = g^x$. To encrypt a message $m \in \mathcal{G}_q$ under public key y, we select $a \in_U Z_q$, and compute the ciphertext $(\alpha, \beta) = (my^a, g^a)$. To decrypt this ciphertext using the private key x, we compute $\alpha/\beta^x = my^a/(g^a)^x = m$.

The El Gamal cryptosystem is *semantically secure* [28] under the Decision Diffie-Hellman (DDH) assumption over $\mathcal{G}_q$. Informally, this means that an attacker who selects message pair (m_0, m_1) is unable to distinguish between encryptions of these two messages with probability significantly greater than 1/2, i.e., than a random guess. See [43] for details.

Let $(\alpha_0\alpha_1, \beta_0\beta_1) = (\alpha_0, \beta_0) \otimes (\alpha_1, \beta_1)$. Another useful property of the El Gamal cryptosystem is the fact that it possesses a *homomorphism* under the operator $\otimes$. In particular, observe that if (α_0, β_0) and (α_1, β_1) represent ciphertexts corresponding to plaintexts m_0 and m_1 respectively, then $(\alpha_0, \beta_0) \otimes (\alpha_1, \beta_1)$ represents an encryption of the plaintext $m_0 m_1$. A consequence of this homomorphic property is that it is possible, using knowledge of the public key alone, to derive a random *re-encryption* (α', β') of a given ciphertext (α, β). This is accomplished by computing $(\alpha', \beta') = (\alpha, \beta) \otimes (\gamma, \delta)$, where (γ, δ) represents an encryption of the plaintext value 1. It is possible to prove quite efficiently in zero-knowledge that (α', β') represents a valid re-encryption of (α, β) using, e.g.,

[2] Most commonly, we let $p = 2q + 1$, and we let $\mathcal{G}_q$ be the set of quadratic residues in Z_p^*. In this setting, plaintexts not in $\mathcal{G}_q$ can be mapped onto $\mathcal{G}_q$ by appropriate forcing of the LeGendre symbol, e.g., through multiplication by a predetermined non-residue.

a variant of the Schnorr proof of knowledge protocol [17,42]. This proof may also be made non-interactive. See [6] for an overview. We let $(\alpha, \beta) \equiv (\alpha', \beta')$ denote equivalence of underlying plaintexts for (α, β) and (α', β'), and $(\alpha, \beta) \not\equiv (\alpha', \beta')$ denote non-equivalence.

The plaintext 0 has only a degenerate ciphertext in the El Gamal cryptosystem. We can represent it by some other plaintext in our protocol.

Distributed decryption for El Gamal: A final useful property of the El Gamal cipher is that it may easily be converted into a threshold protocol. With use of a *distributed key generation* protocol such as that in [8] or [9], players generate a private key x that is held according to a t-out-of-n sharing scheme for some $t < n/2$. In particular, each player P_i obtains a private share x_i consisting of the evaluation of a $(t-1)$-degree polynomial on, e.g., the value i over an appropriate field. To decrypt a ciphertext (α, β), each player P_i publishes the value $\beta_i = \beta^{x_i}$, along with a ZK proof of correct exponentiation, that is, a ZK proof of knowledge of z such that $\log_g y_i = \log_\beta \beta_i$ using, e.g., an appropriate variant on the protocol described in [13]. Players then compute $\beta^x = \prod_{i=1}^{t} \beta_{a_i}^{\lambda_{a_i}}$ on t correct shares $\{\beta_{a_i}\}_{i=1}^{t}$, where λ_{a_i} is the LaGrange coefficient for the a_i^{th} share. Assuming use of non-interactive proofs, with security consequently relying on the random oracle model, the broadcast round complexity of the protocol is $\mathcal{O}(1)$, the message complexity is $\mathcal{O}(n)$ group elements, and the per-player computational costs are $\mathcal{O}(n)$ exponentiations. We do not provide further details, but instead refer the reader to, e.g., [25], which describes a threshold DSS scheme with similar properties.

Proof of knowledge of El Gamal plaintext: Given knowledge of the encryption exponent a of an El Gamal ciphertext $(\alpha, \beta) = (my^a, g^a)$, a player can prove knowledge of m in zero knowledge. This may be accomplished simply by means of an (honest-verifier) zero-knowledge proof of knowledge of a such that $\beta = g^a$ using, e.g., a variant of the Schnorr identification protocol [42] with a challenge carefully generated jointly by all servers. Soundness may then be based on the discrete log problem. The proof of knowledge may also be replaced with a non-interactive protocol through use of Fiat-Shamir techniques [20]. In this case, the ciphertext and accompanying proof may be regarded as a *plaintext-aware* encryption, and security depends additionally on use of the random oracle model. In this case, the broadcast round complexity of this protocol is $\mathcal{O}(1)$, the message complexity is $\mathcal{O}(1)$ group elements, and the per-player computational costs are $\mathcal{O}(1)$ exponentiations.

Mix network ($\mathcal{MN}$): The second key tool in our construction is known as a *mix network*. This primitive for privacy was introduced by Chaum [10], and has recently received considerable attention, both in terms of implementation improvements [1,32,33,34,37] and a wide variety of ideas for applications, of which some examples may be found in [23,31,38,41]. Intuitively, a mix network is a multi-party protocol that takes as input a list of ciphertext items and from this produces a new, random list of ciphertext items such that there is a one-to-one

correspondence between the underlying plaintexts of input and output items. In other words, the underlying output plaintexts represent a random permutation of the underlying input plaintexts. The security of a mix network is characterized by the infeasibility for an adversary of determining which output items correspond to which input items.

While there are many flavors of mix network, the type we employ is based on the El Gamal cipher, and works as follows. An El Gamal public key y is published for use by all players, who are assumed to share an authenticated broadcast channel (or, equivalently, a bulletin board). Some subset of n players known as *mix servers* share the corresponding private key according to a (t, n)-threshold scheme as described above. Input to the mix network consists of a sequence $(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_k, \beta_k)$ of El Gamal ciphertexts posted by the players to the bulletin board.[3] The mix servers perform a sequence of distributed operations on these inputs. The output of the mix network is a random permutation and re-encryption of the inputs, namely a sequence $(\alpha'_{\sigma(1)}, \beta'_{\sigma(1)}), (\alpha'_{\sigma(2)}, \beta'_{\sigma(2)}), \ldots, (\alpha'_{\sigma(k)}, \beta'_{\sigma(k)})$ where (α'_i, β'_i) represents a random re-encryption of (α_i, β_i), and σ is a random permutation on k elements.

There are a number of variants on this basic primitive. For example, inputs to the mix network may be plaintexts, rather than ciphertexts; alternatively, the converse is possible. Note that these are really just special cases of what is described here: a plaintext is a degenerate form of ciphertext with encryption factor 0. Another important variant that we employ here is a mix network in which the input consists of a two-dimensional matrix of ciphertexts $\{(\alpha_{i,j}, \beta_{i,j})\}$ for $1 \leq i \leq k$ and $1 \leq j \leq v$. The output consists of a random, blinded permutation of the rows of this matrix, with no alteration of the order of underlying plaintexts within rows. In other words the output is $\{(\alpha'_{\sigma(i),j}, \beta'_{\sigma(i),j})\}$ for a random permutation σ on k elements, where $(\alpha'_{i,j}, \beta'_{i,j})$ represents a re-encryption of $(\alpha_{i,j}, \beta_{i,j})$. We do not provide details on this extension of the basic mix network primitive, but simply note that it may be implemented with overhead linear in the number v of input columns.

We can base the mix network $\mathcal{MN}$ for the mix-and-match protocol on any of several constructions proposed in the literature. For small input sizes, the construction of Abe [2] or the similar construction of Jakobsson and Juels [33] is most efficient. Given that these schemes are publicly verifiable and possess easily provable security properties, we adopt either construction where we must make explicit reference to the properties of the mix network in mix and match.

The mix networks described in [2] and [34] have the following properties. They are secure, that is, both private and robust, against a static, active adversary that corrupts $t < n/2$ mix servers. Let k be the number of input elements. With underlying interactive zero-knowledge proof protocols, i.e., those involving challenges carefully generated jointly by the servers, privacy for this protocol

[3] To prevent attacks involving one player posting a re-encryption of the ciphertext of another player, it is sometimes necessary for ciphertexts to be encrypted in a manner that is *plaintext aware*. This may be accomplished through, e.g., a ZK proof of knowledge of the discrete log of β for a ciphertext (α, β) (see [32,43]).

may be reduced to the DDH assumption, and protocol robustness to the discrete log problem. By using non-interactive proof protocols under the Fiat-Shamir heuristic [20], we introduce additional dependence on the random oracle model for security. In this latter case, the asymptotic broadcast round complexity is $\mathcal{O}(n)$. The message complexity is $\mathcal{O}(nk \log k)$, while the total computational cost per server is $\mathcal{O}(nk \log k)$ exponentiations.

Distributed plaintext equality test ($\mathcal{PET}$): Let (α, β) and (α', β') be El Gamal ciphertexts with respective underlying plaintexts m_1 and m_2. In the $\mathcal{PET}$ protocol, players jointly determine whether $m_1 = m_2$, i.e., whether $(\alpha, \beta) \equiv (\alpha', \beta')$.

Consider the ciphertext $(\epsilon, \zeta) = (\alpha/\alpha', \beta/\beta')$. If $(\alpha, \beta) \equiv (\alpha', \beta')$, then (ϵ, ζ) represents an encryption of the plaintext integer 1; otherwise, it represents an encryption of the quotient m_1/m_2. The idea behind the $\mathcal{PET}$ protocol, therefore, is to have the parties blind and then decrypt (ϵ, ζ) in such a way that the resulting output is 1 if $(\alpha, \beta) \equiv (\alpha', \beta')$, and a random integer otherwise. Player P_i blinds (ϵ, ζ) by raising each element in the pair to a random exponent $z_i \in Z_q$. This form of blinding leaves the plaintext intact if it is equal to 1 and randomizes it otherwise. The players then combine their blinded shares and perform a distributed decryption on the resulting ciphertext. The protocol is as follows:

1. Each player P_i selects $z_i \in_U Z_q$. She publishes a Pedersen commitment [39] $C_i = g^{z_i} h^{r_i}$ to z_i, where h is a generator such that $\log_g h$ is unknown to any coalition of servers and $r_i \in_U Z_q$ is selected by P_i.
2. Each player computes $(\epsilon_i, \zeta_i) = (\epsilon^{z_i}, \zeta^{z_i})$ and broadcasts it.
3. Each player P_i proves to the other players that (ϵ_i, ζ_i) is well formed relative to the commitment C_i. In particular, she provides a zero-knowledge proof of knowledge of a pair $(z_i, r_i) \in Z_q^2$ such that $C_i = g^{z_i} h^{r_i}$ and $\epsilon_i = \epsilon^{z_i}$ and $\zeta_i = \zeta^{z_i}$. This may be accomplished efficiently using appropriate variants on protocols elaborated in [13,17].
4. The players jointly decrypt $(\gamma, \delta) = (\prod_{i=1}^{n} \epsilon_i, \prod_{i=1}^{n} \zeta_i)$.
5. If the resulting plaintext is 1, then the players conclude that $(\alpha, \beta) \equiv (\alpha', \beta')$. Otherwise, they conclude that $(\alpha, \beta) \not\equiv (\alpha', \beta')$.

If any player is found to be deviating from the protocol, that player is excluded from further participation.

Like the other building blocks presented here, $\mathcal{PET}$ is minimal knowledge under the DDH assumption. By "minimal knowledge", we mean that players learn nothing beyond whether or not $(\alpha, \beta) \equiv (\alpha', \beta')$. To be more precise, even given malicious adversarial behavior (such as refusal of servers to participate in step 3), the distribution of protocol transcripts may be simulated by any entity that knows whether or not $(\alpha, \beta) \equiv (\alpha', \beta')$. The simulated transcript is indistinguishable from a correct one under the DDH assumption. Under the discrete log problem, it is infeasible for any adversary controlling $t < n/2$ servers to cause a server to deviate from the protocol without detection. Assuming non-interactive proof protocols with security in the random oracle model, the protocol

may be executed with $\mathcal{O}(1)$ broadcast rounds, with a message complexity of $\mathcal{O}(n)$ group elements. The computational costs per player are $\mathcal{O}(n)$ exponentiations.[4]

The $\mathcal{PECT}$ algorithm is the tool that enables us to perform comparisons between encrypted values input to gates and encrypted values in blinded tables. In other words, it is the basic tool for lookups in blinded tables.

3 The Mix and Match Protocol

We are now ready to describe in detail our main protocol, the mix-and-match scheme. Recall that players must agree in advance on a representation of the target function f as a circuit C_f. Let us suppose that this circuit consists of N gates, denoted by $G_1, G_2, \ldots, G_N$. We may assume, without loss of generality, that the numbering of gates is such that every gate G_{i+1} has circuit depth at least that of G_i. Thus, evaluation of gate values may proceed in order of index number. For simplicity of presentation, we assume that all gates G_i are binary, i.e., each gates has two inputs and one output, all of which are bit values. We also assume that the function f is binary, i.e., the output is a single bit. We let gate G_N be the output gate for f. We later give a brief description of how to extend the described scheme to non-binary gates and functions f quite straightforwardly.

Let us denote the sequence of input bits of player i by $B_i = \{b_1, b_2, \ldots, b_k\}$. Thus, the aim of the protocol is for players to compute $f(B_1, B_2, \ldots, B_n)$ without revealing any additional information about any of $B_1, B_2, \ldots, B_n$. Let us denote the lookup table corresponding to gate G_i by T_i. As we assume that gates are binary, table T_i contains three columns and four rows; the first two columns represent input bit values, and the third, the corresponding output bit. Table 1, for example, depicts the logical table corresponding to an AND gate. This is, of course, just a standard truth table.

$left$	$right$	$output$
0	0	0
0	1	0
1	0	0
1	1	1

Figure 1. Logical table representing an AND gate

We let $T_i[u, v]$ represent the value in row u and column v of table T_i. We denote by $\overline{T}_i$ the blinded, permuted table yielded by application of $\mathcal{MN}$ to T_i.

[4] Pedersen commitments are included here for technical reasons, namely to aid in security proofs for the protocol. Under the random oracle assumption on hash function h, we can have each player P_i instead publish a commitment $C_i = h(\epsilon_i, \zeta_i)$, and subsequently have all players decommit. This reduces protocol costs by a significant constant factor.

We present our mix-and-match protocol in terms of four steps:

1. **Input protocol:** Each player contributing input to f broadcasts El Gamal encryptions of her input bits B_i under the public key y. (Note that as the integer 0 has only a degenerate ciphertext in the El Gamal cryptosystem, it is convenient to represent a '0' bit by the plaintext g^{-1} and a '1' bit by the plaintext g.) Players prove knowledge of the associated plaintexts.
2. **Mixing:** Players apply the mix network $\mathcal{MN}$ to the tables $\{T_i\}$ under public key y. Each player in turn mixes all tables. The output is the set of blinded tables $\overline{T}_1, \overline{T}_2, \ldots, \overline{T}_N$.
3. **Matching:** For gates $G_1, G_2, \ldots, G_N$ in order, the players do the following. Let l_i (left) and r_i (right) be the ciphertext input values to gate G_i. The players use $\mathcal{PET}$ in order to compare the pair $(l_i,\ r_i)$ with each row u in $\overline{T}_i$ until a match is found. For each row u, the players check whether $\mathcal{PET}(l_i, \overline{T}_i[u,1]) = 1$ and $\mathcal{PET}(r_i, \overline{T}_i[u,2]) = 1$. If both checks hold, then the players determine that the encrypted output value o_i of gate G_i is $\overline{T}_i[u,3]$. The players do this for $u = 1, 2, 3$, and then 4 until a match is found.
4. **Output:** After evaluating the last gate, G_N, the players obtain o_N, a ciphertext encrypting $f(B_1, B_2, \ldots, B_n)$. They jointly decrypt this ciphertext value to reveal the output of the function f.

If a player has provided an invalid input ciphertext (α, β), i.e., a ciphertext whose plaintext does not represent a bit, then the matching step as applied to that ciphertext will fail. In other words, no matching row will be found. This will reveal the invalidity of the input to participating players. An alternative strategy to identify invalid inputs is for players to provide validity proofs along with their inputs. Let (α, β) be a ciphertext input in which a '0' bit is represented by the plaintext value g^{-1}, and a '1' bit is represented by the plaintext value g. Such a proof then takes the form of a proof of knowledge of $(z \mid \alpha/g = y^z, \beta = g^z)$ or $(z' \mid \alpha g = y^{z'}, \beta = g^{z'})$. See [6,17] for descriptions of how to construct disjunctive and conjunctive proofs of knowledge efficiently.

If players determine that a player P_i participating in the function evaluation protocol has cheated or failed, they expel him from the protocol, according to standard practice in the literature for threshold security protocols. They then rewind and recompute as necessary. Due to space limitations, we do not prove security results, but simply state that our mix-and-match construction meets the security requirements formalized by Canetti [7] for secure multiplayer protocols. The interactive variant does this in a computational sense, i.e., there is an *ideal process adversary* capable of producing a simulation indistinguishable from a real one under the DDH assumption. The non-interactive variant depends additionally on use of the random oracle model.

Extensions: As explained above, it is easy to extend the mix-and-match protocol to non-binary gates G_i. For G_i to take as input a j-tuple of values, we construct T_i such that columns $1, 2, \ldots, j$ contain input values. (We increase the number of rows correspondingly to 2^j.) On evaluating G_i, we compare the input

tuple with the first j columns of a given row in $\overline{T}_i$. For G_i to yield multiple output values, it suffices to have the output column in T_i carry multiple values. Input or output values can be made non-binary simply by formulating the entries in T_i appropriately. Additionally, the circuit for f can have multiple output gates, requiring simply that players perform multiple decryptions in the final step of the mix-and-match protocol.

The set of players providing inputs to f in step 1 may be distinct or arbitrarily overlapping with the set of players performing the secure computation. We see this principle at work in our auction protocol in section 4. Similarly, since the set of players performing the mixing operation need have knowledge only of y and not x, this set may be disjoint from the set of players performing the matching.

Remarks:

- The multiplicative homomorphism property of the El Gamal cipher allows for secure multiplication of plaintext values at the cost of a single modular multiplication. This observation may often be exploited to reduce the cost of the protocol. For example, if the left and right input bits to an AND gate are ciphertexts l_i and r_i respectively, each with plaintext value g^{-1} (representing a '0' bit) or g (representing a '1' bit), then the product $l_i r_i$ will have one of three corresponding plaintexts, g^{-2}, 1, or g^2. We can thus condense the AND gate to include only three rows.
- One means of reducing the number of gates is to use the El Gamal variant proposed by Franklin and Haber [22] for use in secure multiparty computation. This is an El Gamal cryptosystem in which -1 and 1 are both valid plaintexts. In consequence of the multiplicative homomorphism of El Gamal, it is possible for players to compute XOR non-interactively with this scheme.
- It may easily be seen that the transcripts of all players' proofs constitute a publicly verifiable proof of the correctness of the computation.

3.1 Performance

The full protocol in the random oracle model, i.e., with non-interactive proofs, may be achieved in $\mathcal{O}(n + d)$ broadcast rounds, where d is the depth of the circuit C_f. As players invoke $\mathcal{MN}$ once per gate and $\mathcal{PET}$ a constant number of times per gate, the overall message complexity is $\mathcal{O}(nN)$ group elements, while the computational complexity per player is likewise $\mathcal{O}(nN)$ exponentiations. As noted above, mix and match can be implemented with use of interactive proofs using, e.g., techniques in [16], thereby eliminating the random oracle assumption as a security requirement at the expense of slightly higher protocol costs. Asymptotic costs in the non-interactive mix-and-match protocol are on a par with the best contemporaneous results, such as [16,30]. It is important to note, though, that these latter two results achieve full field operations for each gate. (The result in [16] is in the computational model with an assumed broadcast channel, and tolerates an adversary in control of any minority coalition. The scheme in [30] is in the private channels model with security for $t < n/3$.)

4 Auctions

We now show how to apply mix and match to the construction of an auction protocol. We consider two, possibly overlapping sets of participants, m bidders, denoted by $A_1, A_2, \ldots, A_m$, and n servers, denoted by $P_1, P_2, \ldots, P_n$. All participants are assumed to share an authenticated broadcast channel. We achieve the following properties in our proposed scheme:

1. *Non-interactivity:* Bidders submit bids in a non-interactive fashion. That is, they broadcast their bids to the servers, but need not participate subsequently in the auction protocol except to learn the outcome.
2. *Auction adaptability:* Our auction protocol is readily adaptable with little overhead to a range of auction types, such as highest-price auctions and Vickrey auctions, as well as to related non-auction procedures, such as polling.
3. *Full privacy:* The only information revealed at the conclusion of the auction is that essential to public execution of resulting transactions. In a highest-price auction, for example, only the winning bid and the identity of the winning bidder are revealed.
4. *Robustness:* An adversary consisting of a coalition of bidders or a minority coalition of servers cannot disrupt the auction protocol or undermine the privacy guarantees. (The servers are simply players as described in the mix-and-match scheme above, so that the computation achieves the same robustness and security characteristics.)
5. *Multiple servers:* Our auction protocol accommodates an arbitrary number of servers and a range of trust models on these servers.
6. *Public verifiability:* The proof transcripts of the auction servers are publicly verifiable. That is, any player (and indeed, any external entity) can verify the correctness of the auction execution without trust in the auction servers.

The principle drawback of our scheme is the intensive communication it requires among servers. Given that this may occur in a manner that is offline from the perspective of bidders, however, it does not pose a serious practical limitation.

In principle, it is possible to achieve the above set of properties with use of any general secure multiparty computation technique and any threshold public-key cryptosystem. The most difficult property to achieve in practice is that of non-interactivity. One method is as follows. Given an appropriate circuit representation for the computation, bidders submit ciphertext bids which the the servers decompose into shares using their private shares of the ciphertext key. This approach, however, is rather inefficient, as the circuit must be very large.

Another, more efficient approach to building a non-interactive protocol is for a bidder to make the sharing implicit in her bid. The bidder submits verifiable secret sharings of the component bits of her bid, along with ciphertexts of the shares. These ciphertexts may be encrypted under the keys of the individual servers, or under a shared key. In the latter case, the ciphertexts may be *directively decrypted*, i.e., decrypted for a unique recipient. This approach, while

fairly practical, is still cumbersome. The bidder must, at a minimum, know how many servers are participating, and, to achieve a practical scheme, must submit nk ciphertexts, where k is the number of bits composing her bid. Additionally, as noted above, we believe that mix and match is quite competitive with other secure function evaluation protocols for applications, like auctions, involving intensive bitwise manipulation.

In consequence of the difficulties involved in deploying standard general secure function evaluation techniques, a number of secure protocols have been proposed in the literature that are specially tailored for auctions. One of the earliest of these is the scheme of Franklin and Reiter [21]. This scheme is not fully private, in the sense that it only ensures the confidentiality of bids until the end of the protocol (although the authors mention a fully private variant). Some more recent schemes include those of Harkavy, Tygar, and Kikuchi [29], Cachin [5], Sako [40], Di Crescenzo [19], and Naor, Pinkas, and Sumner [36]. The Harkavy *et al.* scheme is fully privacy preserving, but involves intensive bidder involvement [29], and is not easily adaptable to different auction types or to related protocols. The scheme of Cachin involves two servers, and requires some communication among bidders. At the end of the protocol, a list of bidders is obtained, but not the bid amounts. The scheme of Di Crescenzo [19] requires no communication between bidders, and has low round complexity, but involves the participation of only a single server. The scheme of Sako [40] works on a different principle from these others, involving opening of bids in what is effectively a privacy-preserving Dutch-style auction. While efficient for small auctions, it involves costs linear in the range of possible bids, and does not allow for extension to second-price and other auction types. The scheme of Naor *et al.* [36] is the first to seek to achieve the first four auction properties enumerated above.[5]

4.1 A Mix-and-Match Auction Protocol

We now present our auction protocol, achieving all six of the properties enumerated above. We describe an architecture for executing highest-bid auctions, although variants such as Vickrey auctions may be achieved through simple modifications imposing minimal additional overhead. Let $B_i = b_i^{(k)}, b_i^{(k-1)}, \ldots, b_i^{(1)}$ be a bitwise representation of the bid B_i of bidder A_i. Let $E[b]$ represent the El Gamal encryption of a given bit b. To avoid cumbersome details, we use somewhat loose notation here, and also do not consider the association of user identities with bids. Also for the sake of simplicity, we assume that there are no ties between bids. The protocol is as follows.

1. Each bidder A_i submits her bid consisting of El Gamal ciphertexts $E[b_i^{(k)}]$, $E[b_i^{(k-1)}], \ldots, E[b_i^{(1)}]$ along with proofs of knowledge of the associated plaintexts. Let $E[B_i]$ denote the k-tuple of ciphertexts representing the submitted bid of A_i. (This submission $E[B_i]$ may be also digitally signed by A_i.)

[5] A security flaw which we do not have space to describe here, however, allows one of the servers to tamper with bids in this protocol.

2. For each pair of encrypted bids, players do the following:
   - Servers apply a mix-and-match-based millionaires' problem protocol to pairs of encrypted bids $(E[B_i], E[B_j])$. Let $E[w]$ denote the ciphertext outcome of the comparison on the bid pair $(\overline{E}_i, \overline{E}_j)$. Here $w = 1$ if bid i is higher and $w = -1$ if bid j is higher.[6]
   - Servers construct a two-row table T in which the first row contains the pair $(-1, E[B_i])$, and the second row contains the pair $(1, E[B_j])$. Thus each row contains $k + 1$ columns. Players mix T to obtain blinded table $\overline{T}$. Thus $\overline{T}$ consists of rows $(E[-1], \overline{E}[B_i])$ and $(E[1], \overline{E}[B_j])$ in a random order, where $\overline{E}[B_i]$ and $\overline{E}[B_j]$ denote re-encryptions of ciphertexts $E[B_i]$ and $E[B_j]$.
   - Servers match w against the first column entries of $\overline{T}$. When they find a match, they output the ciphertext bid in the corresponding row. This will be $\overline{E}[B_i]$ if $w = -1$ and $\overline{E}[B_j]$ if $w = 1$.
3. Servers repeat the previous step following a tennis tournament format until only the ciphertext $E[B_t]$ of a winning bid remains.
4. Servers jointly decrypt the winning bid $E[B_t]$.

Many variants of this basic scheme are possible. For example, to handle ties, players might execute a sorting algorithm based on pairwise comparisons, rather than a tennis tournament. In this case, we must enforce a secret, random tie-breaking mechanism for comparisons between equal bids. Once an ordered list is obtained, it suffices to compare bids from highest to lowest until all highest bids are identified. We leave further details to the reader.

Assuming use of the publicly verifiable mix network proposed in [2,34] and correct server behavior, it may easily be seen that the asymptotic computational cost of the protocol described above is $\mathcal{O}(knm)$ exponentiations per server, while the message complexity is also $\mathcal{O}(knm)$. With the use of fast workstations, a crude estimate suggests that for $m = 100, n = 5$, and $k = 20$, i.e., for 100 bidders, 5 servers, and bids ranging values ranging from 1 to just over 1,000,000, an auction may be conducted in under 3 minutes on fast workstations.

Acknowledgments

The authors wish to extend their thanks to Ivan Damgård, Moti Yung, and the anonymous reviewers of this paper.

References

1. M. Abe. Universally verifiable mix-net with verification work independent of the number of mix-servers. In K. Nyberg, editor, *EUROCRYPT '98*, pages 437–447. Springer-Verlag, 1998. LNCS no. 1403.
2. M. Abe. A mix-network on permutation networks. In K.Y. Lam, C. Xing, and E. Okamoto, editors, *ASIACRYPT '99*, pages 258–273, 1999. LNCS no. 1716.

[6] The plaintext -1 may be encoded as other integer, as needed.

3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS '93*, pages 62–73. ACM, 1993.
4. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computations. In *STOC '88*, pages 1–10. ACM, 1988.
5. C. Cachin. Efficient private bidding and auctions with an oblivious third party. In G. Tsudik, editor, *ACM CCS '99*, pages 120–127. ACM, 1999.
6. J. Camenisch and M. Michels. Proving that a number is the product of two safe primes. In J. Stern, editor, *EUROCRYPT '99*, pages 107–122. Springer-Verlag, 1999. LNCS no. 1592.
7. R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.
8. R. Canetti, R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Adaptive security for threshold cryptosystems. In M. Weiner, editor, *CRYPTO '99*, pages 98 – 115. Springer-Verlag, 1999. LNCS no. 1166.
9. R. Canetti, R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. The (in)security of distributed key generation in dlog-based cryptosystems. In J. Stern, editor, *EUROCRYPT '99*, pages 295–310. Springer-Verlag, 1999. LNCS no. 1592.
10. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.
11. D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *STOC '88*, pages 11–19. ACM, 1988.
12. D. Chaum, I. Damgård, and J. van de Graaf. Multiparty computations ensuring privacy of each party's input and correctness of the result. In C. Pomerance, editor, *CRYPTO '87*, pages 87–119. Springer-Verlag, 1987. LNCS no. 293.
13. D. Chaum and T.P. Pedersen. Wallet databases with observers. In E.F. Brickell, editor, *CRYPTO '92*, pages 89–105. Springer-Verlag, 1992. LNCS no. 740.
14. B. Chor, S. Goldwasser, S. Micali, and B. Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *FOCS '85*, pages 383–395. IEEE Computer Society, 1985.
15. R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin. Efficient multiparty computations secure against an adaptive adversary. In J. Stern, editor, *EUROCRYPT '99*, pages 311–326. Springer-Verlag, 1999. LNCS no. 1592.
16. R. Cramer, I. Damgård, and J.B. Nielsen. Multiparty computation from threshold homomorphic encryption, 2000. IACR ePrint archive manuscript.
17. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Y.G. Desmedt, editor, *CRYPTO '94*, pages 174–187. Springer-Verlag, 1994. LNCS no. 839.
18. R. Cramer and V. Shoup. A practical public-key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *CRYPTO '98*, pages 13–25. Springer-Verlag, 1998. LNCS no. 1462.
19. G. Di Crescenzo. Private selective payment protocols. In P. Syverson, editor, *Financial Cryptography '00*, 2000. To appear.
20. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In J. L. Massey, editor, *EUROCRYPT '86*, pages 186–194. Springer-Verlag, 1986. LNCS no. 263.
21. M. Franklin and M. Reiter. The design and implementation of a secure auction server. *IEEE Transactions on Software Engineering*, 22(5):302–312, 1996.
22. M.K. Franklin and S. Haber. Joint encryption and message-efficient secure computation. *Journal of Cryptology*, 9(4):217–232, 1996.

23. A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale elections. In J. Seberry and Y. Zheng, editors, *AUSCRYPT '92*, pages 244–251. Springer-Verlag, 1992. LNCS no. 718.
24. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.
25. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Robust threshold DSS signatures. In U. Maurer, editor, *EUROCRYPT '96*, pages 354–371. Springer-Verlag, 1996. LNCS no. 1070.
26. R. Gennaro, M. Rabin, and T. Rabin. Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. In *PODC '98*, pages 101–111. ACM, 1998.
27. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC '87*, pages 218–229. ACM, 1987.
28. S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comp. Sys. Sci*, 28(1):270–299, 1984.
29. M. Harkavy, J.D. Tygar, and H. Kikuchi. Electronic auctions with private bids. In *3rd USENIX Workshop on Electronic Commerce*, pages 61–73, 1999.
30. M. Hirt, U. Maurer, and B. Przydatek. Efficient secure multi-party computation. In T. Okamoto, editor, *ASIACRYPT '00*, 2000. To appear.
31. P. Horster, M. Michels, and H. Petersen. Some remarks on a receipt free and universally verifiable mix-type voting scheme. In K. Kim and T. Matsumoto, editors, *ASIACRYPT '96*, pages 125–132. Springer-Verlag, 1996. LNCS no. 1163.
32. M. Jakobsson. A practical mix. In K. Nyberg, editor, *EUROCRYPT '98*, pages 448–461. Springer-Verlag, 1998. LNCS no. 1403.
33. M. Jakobsson. Flash mixing. In *PODC '99*, pages 83–89. ACM, 1999.
34. M. Jakobsson and A. Juels. Millimix: Mixing in small batches, June 1999. DIMACS Technical Report 99-33.
35. A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
36. M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *1st ACM Conf. on Electronic Commerce*, pages 129–139. ACM, 1999.
37. W. Ogata, K. Kurosawa, K. Sako, and K. Takatani. Fault tolerant anonymous channel. In *ICICS '97*, pages 440–444. Springer-Verlag, 1997. LNCS no. 1334.
38. C. Park, K. Itoh, and K. Kurosawa. All/nothing election scheme and anonymous channel. In T. Helleseth, editor, *EUROCRYPT '93*. Springer-Verlag, 1993. LNCS no. 921.
39. T. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *CRYPTO '91*, pages 129–140. Springer-Verlag, 1991. LNCS no. 576.
40. K. Sako. An auction protocol which hides bids of losers. In H. Imai and Y. Zheng, editors, *PKC '00*, pages 422–432. Springer-Verlag, 2000. LNCS no. 1751.
41. K. Sako and J. Kilian. Receipt-free mix-type voting scheme - a practical solution to the implementation of a voting booth. In L.C. Guillou and J.-J. Quisquater, editors, *EUROCRYPT '95*. Springer-Verlag, 1995. LNCS no. 921.
42. C.P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4:161–174, 1991.
43. Y. Tsiounis and M. Yung. On the security of ElGamal-based encryption. In H. Imai and Y. Zheng, editors, *PKC '98*, pages 117–134. Springer-Verlag, 1998. LNCS no. 1431.
44. A.C. Yao. Protocols for secure computations (extended abstract). In *FOCS '82*, pages 160–164. IEEE Computer Society, 1982.