Proofs of Restricted Shuffles

Björn Terelius and Douglas Wikström

CSC KTH Stockholm, Sweden
{terelius,dog}@csc.kth.se

Abstract. A proof of a shuffle is a zero-knowledge proof that one list of ciphertexts is a permutation and re-encryption of another list of ciphertexts. We call a shuffle restricted if the permutation is chosen from a public subset of all permutations. In this paper, we introduce a general technique for constructing proofs of shuffles which restrict the permutation to a group that is characterized by a public polynomial. This generalizes previous work by Reiter and Wang [22], and de Hoogh et al. [7].

Our approach also gives a new efficient proof of an unrestricted shuffle that we think is conceptually simpler and allow a simpler analysis than all previous proofs of shuffles.

Keywords: cryptographic protocols, election schemes, mix-nets, proof of a shuffle.

1 Introduction

Mix-Nets. Suppose that N senders $S_1, \ldots, S_N$ each wish to send a message, but remain anonymous within the group of senders, e.g., the senders could be voters in an election. Chaum [5] introduced the notion of a mix-net (or anonymous channel) to solve this problem. A mix-net is a cryptographic protocol executed by k mix-servers $M_1, \ldots, M_k$, where k typically is much smaller than N. All provably secure mix-nets proposed in the literature take as input a list L_0 of ciphertexts and order the mix-servers in a chain. Each mix-server M_j in the chain takes as input the output L_{j-1} of the previous mix-server in the chain. It processes each ciphertext in L_{j-1} by decrypting and/or re-encrypting it, and then forms its output L_j as the processed ciphertexts in random order. This operation is called a *shuffle*. If the ciphertexts are not decrypted during processing, the mix-servers then perform a joint verifiable decryption of the final list L_k. In some applications, the final output of the mix-net may be a list of the ciphertexts themselves, in which case no joint decryption takes place.

In Chaum's original construction a generic cryptosystem is used. A sender encrypts its message m_i as $\mathsf{E}_{pk_1}(\mathsf{E}_{pk_2}(\cdots \mathsf{E}_{pk_k}(m_i)\cdots))$, where pk_j is the public key of the jth mix-server, and the mix-servers use standard decryption when processing the ciphertexts. Park et al. [18] observed that if a homomorphic cryptosystem is used, the increase in the size of the submitted ciphertext can be avoided. Another, perhaps more important, consequence of using a homomorphic cryptosystem is that it simplifies the construction of a zero-knowledge proof

D.J. Bernstein and T. Lange (Eds.): AFRICACRYPT 2010, LNCS 6055, pp. 100–113, 2010.

that a mix-server shuffles its input correctly, a so called *proof of a shuffle*. We give a detailed account of previous work on such protocols later, but first we conclude the discussion on mix-nets.

Although the mix-servers use the homomorphic properties of the cryptosystem constructively, Pfitzmann [20] points out that a non-malleable cryptosystem must be used in the submission phase. There are several ways to achieve this in a provably secure way.

The security of a mix-net as a whole was first formalized by Abe and Imai [1] in a standalone model, but they did not provide a construction that satisfied their definition. The first definition of security in the UC framework [4] and the first mix-net provably secure as a whole was given by Wikström [25]. Later work [26] gave simpler and more efficient constructions.

Previous Work On Proofs of Shuffles. As far as we know the first proof of a shuffle appears in Sako and Kilian [24] based on a previous protocol by Park et al. [18]. They give a cut-and-choose based zero-knowledge proof of a shuffle with soundness 1/2. Its soundness can be increased using repetition, but this becomes computationally expensive if the number of ciphertexts is large. The first efficient proofs of shuffles were given independently by Neff [17] and Furukawa and Sako [11]. Both approaches were subsequently optimized and generalized, in particular by Groth [13] and Furukawa [9] respectively. Wikström [26] presented a proof of a shuffle based on an idea distinct from both that of Neff and that of Furukawa and Sako.

These shuffles have all been optimized and/or generalized in various ways, e.g., for proving re-encryption and partial decryption shuffling [10], for shuffling hybrid ciphertexts [12], or to reduce communication complexity [15].

A different approach to mix-nets was introduced by Adida and Wikström[3,2]. They investigate to what extent a shuffle can be pre-computed in such a way that the shuffling can be done in public and the online processing of the mix-servers can be reduced to joint decryption. The construction in [2] is conceptually appealing, but inefficient, whereas the construction in [3] is very efficient as long as the number of senders is relatively small.

In a recent paper, Wikström [27] shows that any proof of a shuffle can be split in an offline part and an extremely efficient online part. The offline part is executed before, or at the same time, that senders submit their inputs and consists of a commitment scheme for permutations and a zero-knowledge proof of knowledge of correctly opening such a commitment. The online part is a commitment-consistent proof of a shuffle, where the prover shows that it correctly uses the committed permutation to process the input. This drastically reduces the online complexity of provably secure mix-nets.

Motivated by insider attacks, Reiter and Wang [22] construct a proof of a rotation (they use the term "fragile mixing"). A rotation is a shuffle, where the permutation used is restricted to a random rotation of the ciphertexts. Their idea is to reduce the incentive of insiders to reveal any input/output-correspondence, since this would reveal the complete permutation used, which is assumed to be associated with a cost for the insider. Recently, a more efficient proof of a rotation

is given by de Hoogh et al. [7]. In fact, they give two protocols: a general protocol for any homomorphic cryptosystem and rotation, and a more efficient solution which requires some mild assumptions on the homomorphic cryptosystem used.

De Hoogh et al. lists several possible applications of proofs of rotations beyond the classical application of proofs of shuffles for mix-nets, e.g., secure integer comparison [21], secure function evaluation [16], and submission schemes in electronic election schemes [23].

1.1 Our Contribution

We introduce a novel technique for restricting the class of permutations in a proof of a shuffle of N ciphertexts by showing that π is chosen such that $F(x_{\pi(1)}, \ldots, x_{\pi(N)}) = F(x_1, \ldots, x_N)$ for some public polynomial F. In particular, we can prove that the permutation is contained in the automorphism group of a (directed or undirected) graph on N elements.

A concrete general proof of rotation with efficiency comparable to that of de Hoogh et al. [7] is trivially derived from our technique, but several other natural concrete examples are derived just as easily, e.g. the list of ciphertexts may be viewed as a complete binary tree and the set of permutations restricted to isomorphisms of the tree.

Furthermore, the basic principle behind our technique can be used in a natural way to construct a novel efficient proof of an *unrestricted* shuffle with an exceptionally simple analysis. Given the large and unwieldy literature on how to construct efficient proofs of shuffles we think this conceptual simplification is of independent interest.

1.2 Informal Description of Our Technique

We briefly describe our results and the technique we use, but before we do so we recall the definition of Pedersen's perfectly hiding commitment scheme [19], or more precisely a well known generalization thereof. The commitment parameters consist of $N+1$ randomly chosen generators $g, g_1, \ldots, g_N$ in a group G_q of prime order q in which the discrete logarithm assumption holds. To commit to an array $(e_1, \ldots, e_N) \in \mathbb{Z}_q^N$, the committer forms $g^s \prod_{i=1}^N g_i^{e_i}$. Below we use the fact that sigma proofs can be used to efficiently prove any polynomial relation between the values $e_1, \ldots, e_N$.

A New Proof of a Shuffle. We describe how to prove that a matrix $M \in \mathbb{Z}_q^{N \times N}$ over a finite field $\mathbb{Z}_q$ hidden in a Pedersen commitment is a permutation matrix. Let $\langle \cdot, \cdot \rangle$ denote the standard inner product in $\mathbb{Z}_q^N$ and let $\overline{x} = (x_1, \ldots, x_N)$ be a list of variables. If M does not have exactly one non-zero element in each column and each row, then $\prod_{i=1}^N \langle \overline{m}_i, \overline{x} \rangle \neq \prod_{i=1}^N x_i$ where $\overline{m}_i$ denotes the ith row of M. This can be tested efficiently by Schwarz-Zippel's lemma, which we recall states that if $f(x_1, \ldots, x_N)$ is a non-zero polynomial of degree d and we pick a point $\overline{e}$ from $\mathbb{Z}_q^N$ randomly, then the probability that $f(\overline{e}) = 0$ is at most d/q.

To prove that M is a permutation matrix, given that it has exactly one non-zero element in each row and column, is of course trivial; simply show that the elements of each row sum to one.

We observe that proving both these properties can be done efficiently using the matrix commitment scheme used in [11,27]. The commitment parameters of this scheme consists of independently chosen generators $g, g_1, \ldots, g_N$ of a group G_q of prime order q. To commit to a matrix M, the committer computes $(a_1, \ldots, a_N) = (g^{s_1} \prod_{i=1}^{N} g_i^{m_{i,1}}, \ldots, g^{s_N} \prod_{i=1}^{N} g_i^{m_{i,N}})$. which allows publicly computing a commitment of all $\langle \overline{m}_i, \overline{e} \rangle$ as $\prod_{j=1}^{N} a_j^{e_j} = g^{\langle s, \overline{e} \rangle} \prod_{j=1}^{N} g_i^{\langle \overline{m}_i, \overline{e} \rangle}$. The prover may then use standard sigma proofs to show that $\prod_{i=1}^{N} \langle \overline{m}_i, \overline{e} \rangle = \prod_{i=1}^{N} e_i$. To show that the sum of each row is one it suffices to prove that $\prod_{j=1}^{N} a_j / \prod_{j=1}^{N} g_j$ is on the form g^s for some s.

At this point we may invoke the commitment-consistent proof of a shuffle in [27] to extend the above to a complete proof of an unrestricted shuffle, but we also present a more direct construction.

Restricting the Set of Permutations. In the interest of describing the techniques, we consider how to restrict the set of permutations to the automorphism group of an undirected graph $\mathscr{G}$. Let the graph have vertices $V = \{1, 2, 3, \ldots, N\}$ and edges $E \subseteq V \times V$ and encode the graph as a polynomial $F_{\mathscr{G}}(x_1, \ldots, x_N) = \sum_{(i,j) \in E} x_i x_j$ in $\mathbb{Z}_q[x_1, \ldots, x_N]$. Observe that π is contained in the automorphism group of $\mathscr{G}$ if and only if $F_{\mathscr{G}}(x_{\pi(1)}, \ldots, x_{\pi(N)}) = F_{\mathscr{G}}(x_1, \ldots, x_N)$.

Suppose that a prover has shown that a committed matrix $M \in \mathbb{Z}_q^{N \times N}$ is a permutation matrix corresponding to a permutation π. If π is not contained in the automorphism group of $\mathscr{G}$, it follows from Schwartz-Zippel's lemma that $\Pr\left[F_{\mathscr{G}}(e_{\pi(1)}, \ldots, e_{\pi(N)}) = F_{\mathscr{G}}(e_1, \ldots, e_N)\right]$ is exponentially small, when $\overline{e} \in \mathbb{Z}_q$ is randomly chosen by the verifier. Since $M\overline{e} = (e_{\pi(1)}, \ldots, e_{\pi(N)})$ the verifier can easily compute a commitment of the permuted random exponents as $\prod_{j=1}^{N} a_j^{e_j} = g^{\langle \overline{s}, \overline{e} \rangle} \prod_{j=1}^{N} g_i^{e_{\pi(i)}}$. The prover can then use a standard sigma proof to prove the equality, for example by proving correct computation of each gate in the arithmetic circuit for the polynomial.

Note that it is not important that the polynomial comes from a graph. Hence, we can apply the same technique to prove that π satisfies $F(e_{\pi(1)}, \ldots, e_{\pi(N)}) = F(e_1, \ldots, e_N)$ for any public polynomial F.

2 Notation and Tools

We use n as the security parameter and let q denote an n-bit prime integer. The field with q elements is denoted by $\mathbb{Z}_q$. Our protocols are employed in a group G_q of prime order q with standard generator g, in which the discrete logarithm problem is assumed to be hard. We use bars over vectors to distinguish them from scalars. The angle bracket $\langle \overline{a}, \overline{b} \rangle$ denotes the inner product $\sum_{i=1}^{N} a_i b_i$ of two vectors $\overline{a}, \overline{b} \in \mathbb{Z}_q^N$. For a list $u = (u_1, \ldots, u_N) \in G_q^N$ and a vector $\overline{e} \in \mathbb{Z}_q^N$ we abuse notation and write $u^{\overline{e}} = \prod_{i=1}^{N} u_i^{e_i}$. Throughout, we use $S \subseteq \mathbb{Z}_q$ to denote the set from which the components of our random vectors are chosen.

If $\mathscr{R}_1$ and $\mathscr{R}_2$ are relations we denote by $\mathscr{R}_1 \vee \mathscr{R}_2$ the relation consisting of pairs $((x_1, x_2), w)$ such that $(x_1, w) \in \mathscr{R}_1$ or $(x_2, w) \in \mathscr{R}_2$.

Matrix Commitments. We use a variation of Pedersen commitments [19] in a group G_q, to commit to a matrix over $\mathbb{Z}_q$. The commitment parameter ck needed to commit to an $N \times 1$ matrix consists of a description of the group G_q with its standard generator g and randomly chosen group elements $g_1, \ldots, g_N \in G_q$. An $N \times 1$-matrix M over $\mathbb{Z}_q$ is committed to by computing $a = \mathcal{C}_{ck}(M, s) = g^s \prod_{i=1}^{N} g_i^{m_i}$, where $s \in \mathbb{Z}_q$ is chosen randomly. We abuse notation and omit the commitment parameter when it is clear from the context. An $N \times N$-matrix M is committed to column-wise, i.e., $\mathcal{C}(M, \overline{s}) = \left(\mathcal{C}\left((m_{i,1})_{i=1}^{N}, s_1\right), \ldots, \mathcal{C}\left((m_{i,N})_{i=1}^{N}, s_N\right)\right)$, where in this case $\overline{s}$ is chosen randomly in $\mathbb{Z}_q^N$. By committing to a matrix M column-wise we get the useful identity

$$\mathcal{C}(M, \overline{s})^{\overline{e}} = \prod_{j=1}^{N} g^{s_j e_j} \prod_{i=1}^{N} g_i^{m_{i,j} e_j} = g^{\langle \overline{s}, \overline{e} \rangle} \prod_{i=1}^{N} g_i^{\sum_{j=1}^{N} m_{i,j} e_j} = \mathcal{C}(M\overline{e}, \langle \overline{s}, \overline{e} \rangle) \quad .$$

This feature plays a central role in our approach. It is easy to see that the commitment scheme is perfectly hiding. The binding property is known to hold under the discrete logarithm assumption in G_q, see [11] for a proof.

We construct protocols that are sound under the assumption that the binding property of the above protocol is not violated. We define $\mathscr{R}_{com}$ to be the relation consisting of pairs $((ck, a), (M, \overline{s}, M', \overline{s}'))$ such that $a = \mathcal{C}_{ck}(M, \overline{s}) = \mathcal{C}_{ck}(M', \overline{s}')$, i.e., finding a witness corresponds to violating the binding property of the commitment scheme.

Σ-proofs. Recall that a sigma proof is a three-message protocol that is both special sound and special honest verifier zero-knowledge [6]. The first property means that the view of the verifier can be simulated for a given challenge, and the second property means that a witness can be computed from any pair of accepting transcripts with identical first messages and distinct challenges. It is well known that if a prover $\mathcal{P}^*$ convinces the verifier with probability δ, there exists an extractor running in expected time $\mathcal{O}\big(T/(\delta - \epsilon)\big)$ for some polynomial T and some negligible knowledge error ϵ. Given a statement x, we denote the execution of a sigma proof of knowledge of a witness w such that $(x, w) \in \mathscr{R}$ by $\Sigma\text{-}\mathsf{proof}\left[w \,|(x, w) \in \mathscr{R}\right]$.

We need to prove knowledge of how to open commitments such that the committed values satisfy a public polynomial relation, i.e. to construct a sigma proof

$$\Sigma\text{-}\mathsf{proof}\left[\overline{e} \in \mathbb{Z}_q^N, s \in \mathbb{Z}_q \,\middle|\, a = \mathcal{C}(\overline{e}, s) \wedge f(\overline{e}) = e'\right]$$

given a commitment $a = \mathcal{C}(\overline{e}, s)$, a polynomial $f \in \mathbb{Z}_q[x_1, \ldots, x_N]$, and a value $e' \in \mathbb{Z}_q$. We remind the reader how this can be done.

The parties agree on an arithmetic circuit over $\mathbb{Z}_q$ that evaluates the polynomial f. The prover constructs new commitments a_i to each individual value e_i

hidden in a and proves that it knows how to open all commitments consistently with a proof of the form

$$\Sigma\text{-proof}\left[\overline{e} \in \mathbb{Z}_q^N, s, s_1, \ldots, s_N \in \mathbb{Z}_q \,\middle|\, a = \mathcal{C}\left(\overline{e}, s\right) \wedge \forall_{i=1}^{N}(a_i = \mathcal{C}\left(e_i, s_i\right))\right] \quad .$$

The resulting commitments $a_1, \ldots, a_N$ are considered the input of the arithmetic circuit. For each summation gate, the two input commitments of the gate are multiplied to form the output of the gate. For each product gate with input commitments $a_1 = \mathcal{C}\left(e_1, s_1\right)$ and $a_2 = \mathcal{C}\left(e_2, s_2\right)$, the prover forms a new commitment $a_3 = \mathcal{C}\left(e_3, s_3\right)$ and proves that $e_3 = e_1 e_2$ with a sigma protocol

$$\Sigma\text{-proof}\left[e_2, s_2, s_3' \in \mathbb{Z}_q \,\middle|\, a_3 = g^{s_3'} a_1^{e_2} \wedge a_2 = \mathcal{C}\left(e_2, s_2\right)\right] \quad .$$

Finally, the output a of the entire circuit is shown to be a commitment of e' using a protocol of the form

$$\Sigma\text{-proof}\left[s \in \mathbb{Z}_q \,\middle|\, a/g_1^{e'} = g^s\right] \quad .$$

Special soundness and special honest verifier zero-knowledge allow us to execute all these protocols in parallel using a single challenge from the verifier, thus forming a new sigma protocol. Together with the binding property of the commitment scheme, this implies that the prover knows $\overline{e} \in \mathbb{Z}_q^N$ and $s \in \mathbb{Z}_q$ such that $\mathcal{C}\left(\overline{e}, s\right) = a \wedge f(\overline{e}) = e'$.

We remark that it is sometimes possible to do better than the general technique above. In particular when proving that a shuffle is a rotation, one has to prove that a polynomial of the form $\sum_{i=1}^{N} x_i y_i$ has a certain value. This can be done efficiently by evaluating the polynomial as an inner product between the vectors $\overline{x}$ and $\overline{y}$ using a linear algebra protocol from [14].

Polynomial Equivalence Testing. We use the Schwartz-Zippel lemma to analyze the soundness of our protocols. The lemma gives an efficient, probabilistic test of whether a polynomial is identically zero.

Lemma 1 (Schwartz-Zippel). *Let $f \in \mathbb{Z}_q[x_1, \ldots, x_N]$ be a non-zero multivariate polynomial of total degree $d \geq 0$ over $\mathbb{Z}_q$, let $S \subseteq \mathbb{Z}_q$, and let $e_1, \ldots, e_N$ be chosen randomly from S. Then*

$$\Pr\left[f(e_1, \ldots, e_N) = 0\right] \leq \frac{d}{|S|} \quad .$$

3 Proof of Knowledge of Permutation Matrix

We show how to prove knowledge of how to open a Pedersen commitment of a matrix such that the matrix is a permutation matrix. Wikström [27] constructs a commitment-consistent proof of a shuffle for any shuffle-friendly map, based on the same permutation commitment we use here. Thus, it is trivial to construct

a proof of a shuffle by combining the protocol below with the online protocol in [27].

Our protocol is based on a simple probabilistic test that accepts a non-permutation matrix with negligible probability.

Theorem 1 (Permutation Matrix). *Let $M = (m_{i,j})$ be an $N \times N$-matrix over $\mathbb{Z}_q$ and $\overline{x} = (x_1, \ldots, x_N)$ a vector of N independent variables. Then M is a permutation matrix if and only if $\prod_{i=1}^{N} \langle \overline{m}_i, \overline{x} \rangle = \prod_{i=1}^{N} x_i$ and $M\overline{1} = \overline{1}$.*

Proof. Consider the polynomial $f(\overline{x}) = \prod_{i=1}^{N} \langle \overline{m}_i, \overline{x} \rangle$ in the multivariate polynomial ring $R = \mathbb{Z}_q[x_1, \ldots, x_N]$, where $\overline{m}_i$ is the ith row of M. It is clear that $M\overline{1} = \overline{1}$ and $f(\overline{x}) = \prod_{i=1}^{N} x_i$ if M is a permutation matrix. Conversely, suppose that $M\overline{1} = \overline{1}$ and $f(\overline{x}) = \prod_{i=1}^{N} x_i$. If any row $\overline{m}_i$ were the zero vector, then f would be the zero polynomial. If all rows of M were non-zero, but some row $\overline{m}_i$ contained more than one non-zero element, then f would contain a factor of the form $\sum_{j \in J} m_{i,j} x_j$ with $|J| \geq 2$ and $m_{i,j} \neq 0$ for $j \in J$. Since R is a unique factorization domain, this contradicts the assumption that $f(\overline{x}) = \prod_{i=1}^{N} x_i$. If the jth column contained more than one non-zero element, then $\deg_{x_j} f \geq 2$, again contradicting $f = \prod_{j=1}^{N} x_j$. Thus there is exactly one non-zero element in each row and column and since $M\overline{1} = \overline{1}$, the non-zero element must equal one.

Protocol 1 (Permutation Matrix).
Common Input: Matrix commitment $a \in G_q^N$ and commitment parameters $g, g_1, \ldots, g_N \in G_q$.
Private Input: Permutation matrix $M \in \mathbb{Z}_q^{N \times N}$ and randomness $\overline{s} \in \mathbb{Z}_q^N$ such that $a = \mathcal{C}(M, \overline{s})$.

1. $\mathcal{V}$ chooses $\overline{e} \in S^N \subseteq \mathbb{Z}_q^N$ randomly and hands $\overline{e}$ to $\mathcal{P}$.
2. $\mathcal{P}$ defines $t = \langle \overline{1}, \overline{s} \rangle$ and $k = \langle \overline{s}, \overline{e} \rangle$. Then $\mathcal{V}$ outputs the result of

$$\Sigma\text{-proof}\left[\begin{matrix} t, k \in \mathbb{Z}_q \\ \overline{e}' \in \mathbb{Z}_q^N \end{matrix} \middle| \mathcal{C}(\overline{1}, t) = a^{\overline{1}} \wedge \mathcal{C}(\overline{e}', k) = a^{\overline{e}} \wedge \prod_{i=1}^{N} e'_i = \prod_{i=1}^{N} e_i \right] .$$

We remark that $\mathcal{V}$ could instead hand a random seed to $\mathcal{P}$ and define $\overline{e}$ as the output of a PRG invoked on the seed. This reduces the amount of randomness needed by the verifier, which is important in applications where the role of $\mathcal{V}$ is played by a multiparty coin-flipping protocol. Since this trick is well known (cf. [27]) and complicates the exposition, we do not detail it here.

Proposition 1. *Protocol 1 is a perfectly complete, 4-message honest verifier zero-knowledge proof of knowledge of the relation $\mathscr{R}_\pi \vee \mathscr{R}_{com}$, where the relation $\mathscr{R}_\pi$ consists of pairs $((ck, a), (M, \overline{s}))$ such that M is a permutation matrix and $a = \mathcal{C}_{ck}(M, \overline{s})$.*

Under the discrete logarithm assumption, it is infeasible to find a witness of the relation $\mathscr{R}_{com}$ for the commitment parameter $(g, g_1, \ldots, g_N)$. Thus, for a randomly chosen commitment parameter, the proposition may be interpreted as a proof of knowledge of the relation $\mathscr{R}_\pi$.

3.1 Proof of Proposition 1

The completeness and zero-knowledge properties follows from the completeness and zero-knowledge properties of the sigma protocol and the hiding property of the commitment scheme. What remains is to show that the protocol is a proof of knowledge by creating an extractor. We do this by extracting witnesses $(\overline{e}', t, k)$ from the sigma proof for N linearly independent vectors $\overline{e}$ and use them to recover the matrix M. Finally, we show that if M is not a permutation matrix, then we are able to extract a witness of the commitment relation $\mathscr{R}_{com}$.

Three-Message Protocol. We first make a conceptual change that allows us to view our four-round prover as a particular three-round prover of a standard sigma-protocol. Given a prover $\mathcal{P}^*$, we denote by $\mathcal{P}_+$ the interactive machine that chooses $\overline{e} \in \mathbb{Z}_q^N$ randomly itself instead of letting $\mathcal{V}$ choose it, and then simulates $\mathcal{P}^*$. We denote by $\mathcal{V}_+$ the corresponding verifier that accepts $\overline{e}$ as part of the first message in the sigma proof.

Basic Extractor. We augment the common input with a list of linearly independent vectors $\overline{e}_1, \ldots, \overline{e}_l \in \mathbb{Z}_q^N$ where $l < N$, let $\mathcal{P}_\perp$ be identical to $\mathcal{P}_+$, and define $\mathcal{V}_\perp$ to be identical to $\mathcal{V}_+$ except that it only accepts if $\overline{e}$ is linearly independent of these. If $\mathcal{P}^*$ convinces $\mathcal{V}$ with probability δ, then $\mathcal{P}_\perp$ clearly convinces $\mathcal{V}_\perp$ with probability at least $\delta - \frac{1}{|S|}$, since the probability that $\overline{e}$ is linearly dependent of $\overline{e}_1, \ldots, \overline{e}_l \in \mathbb{Z}_q^N$ is bounded by $\frac{1}{|S|}$.

It is well known that the sigma proof has an extractor $\mathcal{E}_\perp$ running $\mathcal{P}_\perp$ as a black-box that given linearly independent $\overline{e}_1, \ldots, \overline{e}_l \in \mathbb{Z}_q^N$ extracts an $\overline{e}$ that is linearly independent of the former vectors and a corresponding witness $(\overline{e}', t, k)$ of the sigma proof. Furthermore, $\mathcal{E}_\perp$ runs in expected time $T/(\delta - \frac{1}{|S|} - \epsilon)$ for some polynomial $T(n)$ in the security parameter n, where ϵ is the knowledge error of the sigma proof. Denote by $\mathcal{E}_N$ the extractor that computes witnesses $(\overline{e}_l, t_l, \overline{e}'_l, k_l) = \mathcal{E}_\perp(\overline{e}_1, \ldots, \overline{e}_{l-1}, a, g, g_1, \ldots, g_N)$ for $l = 1, \ldots, N$. Then $\mathcal{E}_N$ runs in expected time $\mathcal{O}\big(NT/(\delta - \frac{1}{|S|} - \epsilon)\big)$.

Computation of Committed Matrix. From linear independence follows that there exists $\alpha_{l,j} \in \mathbb{Z}_q$ such that $\sum_{j=1}^N \alpha_{l,j}\overline{e}_j$ is the lth standard unit vector in $\mathbb{Z}_q^N$. We conclude that:

$$a_l = \prod_{j=1}^N a^{\alpha_{l,j}\overline{e}_j} = \prod_{j=1}^N \mathcal{C}\left(\overline{e}'_j, k_j\right)^{\alpha_{l,j}} = \mathcal{C}\left(\sum_{j=1}^N \alpha_{l,j}\overline{e}'_j, \sum_{j=1}^N \alpha_{l,j}k_j\right) \quad .$$

Thus, we have $a = \mathcal{C}(M, \overline{s})$, where $\sum_{j=1}^N \alpha_{l,j}\overline{e}'_j$ is the lth column of a matrix $M \in \mathbb{Z}_q^{N\times N}$ and $\overline{s} = \left(\sum_{j=1}^N \alpha_{1,j}k_j, \ldots, \sum_{j=1}^N \alpha_{N,j}k_j\right) \in \mathbb{Z}_q^N$ is the random vector used to commit.

Product Inequality Extractor. We expect that the matrix M is a permutation matrix, but if this is not the case we must argue that we can find a non-trivial representation of 1 in G_q. We augment the original input with a non-permutation

matrix M which we assume satisfy $M\overline{1} = \overline{1}$. We will see that had $M\overline{1} \neq \overline{1}$, we would have been able to extract a witness of $\mathscr{R}_{com}$. Let $\mathcal{P}_\pi$ be identical to $\mathcal{P}_+$, and define $\mathcal{V}_\pi$ to be identical to $\mathcal{V}_+$ except that it only accepts if

$$\prod_{i=1}^{N} \langle \overline{m}_i, \overline{e} \rangle \neq \prod_{i=1}^{N} e_i \ . \tag{1}$$

From Theorem 1 and Schwartz-Zippel's lemma follows that the probability that the additional requirement is not satisfied is at most $N/|S|$. Thus, if $\mathcal{P}^*$ convinces $\mathcal{V}$ with probability δ, then $\mathcal{P}_\pi$ convinces $\mathcal{V}_\pi$ with probability at least $\delta - N/|S|$. Again, a standard argument implies that there exists an extractor $\mathcal{E}_\pi$ with black-box access to $\mathcal{P}^*$ running in time $\mathcal{O}\big(T'/(\delta - \frac{N}{|S|} - \epsilon)\big)$, which extracts $(\overline{e}, t, \overline{e}', k)$ such that $\mathcal{C}(\overline{e}', k) = a^{\overline{e}}$ and Equation (1) is satisfied.

Main Extractor. Denote by $\mathcal{E}$ the extractor that proceeds as follows:

1. It invokes $\mathcal{E}_N$ to find a matrix M and randomness $\overline{s}$ such that $a = \mathcal{C}(M, \overline{s})$. If M is a permutation matrix, then $\mathcal{E}$ has found the witness $(M, \overline{s})$ of relation $\mathscr{R}_\pi$.
2. If M does not satisfy $M\overline{1} = \overline{1}$, then set $\overline{e}'' = M\overline{1}$ and note that

   $$\overline{e}'' \neq \overline{1} \quad \text{and} \quad \mathcal{C}\left(\overline{1}, t_1\right) = a^{\overline{1}} = \mathcal{C}\left(\overline{e}'', \langle \overline{s}, \overline{1} \rangle\right) \ .$$

   Then $\mathcal{E}$ has found the witness $(a^{\overline{1}}, \overline{1}, t_1, \overline{e}'', \langle \overline{s}, \overline{1} \rangle)$ of the commitment relation $\mathscr{R}_{com}$.
3. If M satisfies $M\overline{1} = \overline{1}$, but is not a permutation matrix, then $\mathcal{E}$ invokes $\mathcal{E}_\pi$ with the additional input M to find $(\overline{e}, t, \overline{e}', k)$ such that $\mathcal{C}(\overline{e}', k) = a^{\overline{e}}$ and Equation (1) holds. Define $\overline{e}'' = M\overline{e}$ and note that

   $$\overline{e}'' \neq \overline{e}' \quad \text{and} \quad \mathcal{C}(\overline{e}', k) = a^{\overline{e}} = \mathcal{C}(\overline{e}'', \langle \overline{s}, \overline{e} \rangle) \ .$$

   The former holds, since $\prod_{i=1}^{N} e'_i = \prod_{i=1}^{N} e_i \neq \prod_{i=1}^{N} e''_i$. Then $\mathcal{E}$ has found the witness $(a^{\overline{e}}, \overline{e}', k, \overline{e}'', \langle \overline{s}, \overline{e} \rangle)$ of the commitment relation $\mathscr{R}_{com}$.

Note that the the expected running time of the extractor $\mathcal{E}$ is bounded by $\mathcal{O}\big((NT + T')/(\delta - \frac{N}{|S|} - \epsilon)\big)$ as required and that it always finds a witness of either $\mathscr{R}_\pi$ or $\mathscr{R}_{com}$. □

4 Proof of Knowledge of Restricted Permutation Matrix

We now detail how one can restrict π to the subset S_F of permutations that satisfies $F(\overline{x}'_1, \ldots, \overline{x}'_d) = F(\overline{x}_1, \ldots, \overline{x}_d)$ for a multivariate polynomial $F(\overline{x}_1, \ldots, \overline{x}_d)$ in $\mathbb{Z}_q[\overline{x}_1, \ldots, \overline{x}_d]$ where $\overline{x}'_i = (x_{i,\pi(1)}, \ldots, x_{i,\pi(N)})$. We remark that $d = 1$ in many instances, in which case the polynomial F will just depend on a single list of N variables.

The protocol below is an extension of Protocol 1. Thus, to simplify the exposition we denote by $\mathsf{P}_\pi(a, t, \overline{e}, \overline{e}', k)$ the predicate used to define the sigma proof of Protocol 1, i.e., $\mathcal{C}\left(\overline{1}, t\right) = a^{\overline{1}} \wedge \mathcal{C}(\overline{e}', k) = a^{\overline{e}} \wedge \prod_{i=1}^{N} e'_i = \prod_{i=1}^{N} e_i$.

Protocol 2 (Restricted Permutation Matrix).
COMMON INPUT: Matrix commitment $a \in G_q^N$, commitment parameters $g, g_1, \ldots, g_N \in G_q$, and a polynomial invariant F.
PRIVATE INPUT: Permutation matrix $M \in \mathbb{Z}_q^{N \times N}$ for a permutation $\pi \in S_F$ and randomness $\overline{s} \in \mathbb{Z}_q^N$ such that $a = \mathcal{C}(M, \overline{s})$.

1. $\mathcal{V}$ chooses $\overline{e}_1, \ldots, \overline{e}_d \in S^N \subseteq \mathbb{Z}_q^N$ randomly and hands $\overline{e}_1, \ldots, \overline{e}_d$ to $\mathcal{P}$.
2. $\mathcal{P}$ defines $t = \langle \overline{1}, \overline{s} \rangle$ and $k_\iota = \langle \overline{s}, \overline{e}_\iota \rangle$ for $\iota = 1, \ldots, d$. Then $\mathcal{V}$ outputs the result of

$$\Sigma\text{-proof}\left[\begin{array}{c|c} \overline{e}'_1 \in \mathbb{Z}_q^N, t, k_1 \in \mathbb{Z}_q & \mathsf{P}_\pi(a, t, \overline{e}_1, \overline{e}'_1, k_1) = 1 \\ \overline{e}'_2, \ldots, \overline{e}'_d \in \mathbb{Z}_q^N & \bigwedge_{j=1}^d \mathcal{C}\left(\overline{e}'_j, k_j\right) = a^{\overline{e}_j} \\ k_2, \ldots, k_d \in \mathbb{Z}_q & \bigwedge F(\overline{e}'_1, \ldots, \overline{e}'_d) = F(\overline{e}_1, \ldots, \overline{e}_d) \end{array}\right] .$$

Proposition 2. *Protocol 2 is a perfectly complete 4-message honest verifier zero-knowledge proof of knowledge of the relation $\mathscr{R}_{\mathscr{G}} \vee \mathscr{R}_{com}$, where the relation $\mathscr{R}_{\mathscr{G}}$ consists of pairs $((ck, a, F), (M, \overline{s}))$ such that M is a permutation matrix of $\pi \in S_F$ and $a = \mathcal{C}_{ck}(M, \overline{s})$.*

The proof, given in the full version, is a slight modification of the proof of Proposition 1.

4.1 Encoding Graphs as Polynomials

So far, we have not discussed where the polynomials would come from. In this section we describe how a graph can be encoded as a polynomial which is invariant under automorphisms of the graph.

The edge set E of an undirected graph $\mathscr{G}$ with N vertices can be encoded by the polynomial $F_{\mathscr{G}}(\overline{x}) = \sum_{(i,j)\in E} x_i x_j$ where $\overline{x}$ is a list of N independent variables. This encoding is generalized in the natural way to a hypergraph with edge set E by defining $F_{\mathscr{G}}(\overline{x}) = \sum_{e\in E} \prod_{i\in e} x_i$. Both encodings allow multiple edges and self-loops.

Notice that the encoding above does not preserve information about the direction of the edges. For directed graphs, we instead introduce new variables $y_1, \ldots, y_N$ and use the polynomial $F_{\mathscr{G}}(\overline{x}, \overline{y}) = \sum_{(i,j)\in E} x_i y_j$, where x_i and y_i represent the origin and destination of a directed edge from and to vertex i respectively. For example, the cyclic group C_N of rotations of N elements arise as the automorphism group of the directed cyclic graph on N vertices. This graph can be encoded by the polynomial $F_{\mathscr{G}}(\overline{x}, \overline{y}) = \sum_{(i,j)\in E} x_i y_j$ so we can use a proof of a restricted shuffle to show that one list of ciphertexts is a rotation of another.

This trick of adding more variables can be generalized to encode the order of the vertices in the edges of a hypergraph.

Theorem 2. *Let $F_{\mathscr{G}}(\overline{x}_1, \ldots, \overline{x}_d)$ be the encoding polynomial of a (directed or undirected) graph or hypergraph $\mathscr{G}$. A permutation π is an automorphism of $\mathscr{G}$ if and only if*

$$F_{\mathscr{G}}(\overline{x}_1, \ldots, \overline{x}_d) = F_{\mathscr{G}}(\overline{x}'_1, \ldots, \overline{x}'_d) \ ,$$

where $\overline{x}'_i = (x_{i,\pi(1)}, \ldots, x_{i,\pi(N)})$.

Proof. Recall that an automorphism is a permutation of the vertices which maps edges to edges. Since $F_{\mathscr{G}}$ is an encoding of the edge set and the edge sets are equal if and only if the permutation is an automorphism, it follows that the encoding polynomials are equal if and only if the permutation is an automorphism. □

5 Proofs of Restricted Shuffles

We immediately get an 8-message proof of a restricted shuffle for any shuffle-friendly map and any homomorphic cryptosystem by combining our result with the protocol in [27]. Here we give a 5-message proof of a restricted shuffle for the important special case where each element in the groups of ciphertexts and randomness have prime order q, e.g. El Gamal [8].

Recall the definition of shuffle-friendly maps from [27], where we use $\mathcal{C}_{pk}$ and $\mathcal{R}_{pk}$ to denote the groups of ciphertexts and randomness for a public key pk.

Definition 1. *A map ϕ_{pk} is shuffle-friendly for a public key $pk \in \mathcal{PK}$ of a homomorphic cryptosystem if it defines a homomorphic map $\phi_{pk} : \mathcal{C}_{pk} \times \mathcal{R}_{pk} \to \mathcal{C}_{pk}$.*

For example, the shuffle-friendly map[1] of a shuffle where the ciphertexts are re-encrypted and permuted is defined by $\phi_{pk}(c, r) = c \cdot \mathsf{E}_{pk}(1, r)$. All the known shuffles of homomorphic ciphertexts or lists of ciphertexts can be expressed similarly (see [27] for more examples).We let $\mathsf{P}_F(a, t, \{\overline{e}_\iota, k_\iota, \overline{e}'_\iota\}_{\iota=1}^d)$ denote the predicate $\mathsf{P}_\pi(a, t, \overline{e}_1, \overline{e}'_1, k_1) \wedge F(\overline{e}'_1, \ldots, \overline{e}'_d) = F(\overline{e}_1, \ldots, \overline{e}_d) \wedge \mathcal{C}(\overline{e}'_j, k_j) = a^{\overline{e}_j}$ for all j, i.e., the predicate that was used to define the sigma proof in Protocol 2.

Protocol 3 (Proof of Restricted Shuffle).
Common Input: Commitment parameters $g, g_1, \ldots, g_N \in G_q$, a polynomial F, public key pk, and ciphertexts $c_1, \ldots, c_N, c'_1, \ldots, c'_N \in \mathcal{C}_{pk}$.
Private Input: A permutation $\pi \in S_F$ and randomness $\overline{r} \in \mathcal{R}_{pk}^N$ such that $c'_i = \phi_{pk}\left(c_{\pi(i)}, r_{\pi(i)}\right)$.

1. Let $M \in \mathbb{Z}_q^{N \times N}$ be the permutation matrix representing π. $\mathcal{P}$ chooses $\overline{s} \in \mathbb{Z}_q^N$ randomly, computes $a = \mathcal{C}(M, \overline{s})$, and hands a to $\mathcal{V}$.
2. $\mathcal{V}$ chooses $\overline{e}_1, \ldots, \overline{e}_d \in S^N \subseteq \mathbb{Z}_q^N$ randomly and hands $\overline{e}_1, \ldots, \overline{e}_d$ to $\mathcal{P}$.
3. $\mathcal{P}$ defines $\overline{e}'_\iota = M\overline{e}_\iota$, $t = \langle \overline{1}, \overline{s} \rangle$, $k_\iota = \langle \overline{s}, \overline{e}_\iota \rangle$ for $\iota = 1, \ldots, d$, and $u = \langle \overline{r}, \overline{e}_1 \rangle$. Then $\mathcal{V}$ outputs the result of

$$\Sigma\text{-proof}\left[\begin{array}{c} \{\overline{e}'_\iota \in \mathbb{Z}_q^N, k_\iota \in \mathbb{Z}_q\}_{\iota=1}^d \\ t \in \mathbb{Z}_q, u \in \mathcal{R}_{pk} \end{array} \middle| \begin{array}{c} \mathsf{P}_F(a, t, \{\overline{e}_\iota, k_\iota, \overline{e}'_\iota\}_{\iota=1}^d) = 1 \\ \prod_{i=1}^N (c'_i)^{e'_{1,i}} = \phi_{pk}\left(\prod_{i=1}^N c_i^{e_{1,i}}, u\right) \end{array} \right] \ .$$

[1] We remark that nothing prevents proving a shuffle of other objects than ciphertexts, i.e., any groups $\mathcal{C}_{pk}$ and $\mathcal{R}_{pk}$ of prime order q, and any homomorphic map ϕ_{pk} defined by some public parameter pk can be used.

In the full version we give a concrete instantiation of the above sigma proof for an unrestricted shuffle which has efficiency comparable to some of the most efficient proofs of a shuffle in the literature. We also explain how a shuffle of a complete binary tree can be derived with essentially no additional computational cost.

Proposition 3. *Protocol 3 is a perfectly complete 5-message honest verifier zero-knowledge proof of knowledge of the relation* $\mathscr{R}_{\phi_{pk}} \vee \mathscr{R}_{com}$*, where* $\mathscr{R}_{\phi_{pk}}$ *consists of pairs* $((ck, a, F, pk, \overline{c}, \overline{c}'), (M, \overline{s}, \overline{r}))$ *such that* M *is a permutation matrix of* $\pi \in S_F$*,* $a = \mathcal{C}_{ck}(M, \overline{s})$ *and* $c'_i = \phi_{pk}\left(c_{\pi(i)}, r_{\pi(i)}\right)$*.*

A proof of the proposition is given in the full version.

6 Variations and Generalizations

There are many natural variations and generalizations of our approach. Below we briefly mention some of these.

An alternative encoding of a graph is found by switching the roles of multiplication and addition in the encoding polynomial, e.g., the encoding polynomial of an undirected graph $\mathscr{G}$ could be defined as $F_{\mathscr{G}}(x_1, \ldots, x_N) = \prod_{(i,j)\in E}(x_i + x_j)$. Direction of edges can also be represented in an alternative way using powers to distinguish the ends of each edge, e.g, given a directed graph $\mathscr{G}$ the encoding polynomial could be defined by $F_{\mathscr{G}}(x_1, \ldots, x_N) = \sum_{(i,j)\in E} x_i x_j^2$. We can combine these ideas, turning exponentiation into multiplication by a scalar, and get the encoding polynomial $F_{\mathscr{G}}(x_1, \ldots, x_N) = \prod_{(i,j)\in E}(x_i + 2x_j + 1)$ for our directed graph. The additive constant 1 is needed to fix the scalar multiple of each factor since factorization in the ring $\mathbb{Z}_q[x_1, \ldots, x_N]$ is only unique up to units. The same ideas can be extended to hypergraphs and oriented hypergraphs, i.e. hypergraphs where the edges are ordered tuples rather than sets of vertices.

It is easy to generalize the protocol to proving that $f(x_{\pi(1)}, ...x_{\pi(N)}) = g(x_1, ...x_N)$ for an arbitrary function g. In the body of the paper, we dealt with the important special case where $f = g$, but by choosing g different from f, it is possible to prove that the permutation belong to a set that is not necessarily a group. For example, one can prove that the permutation is odd by choosing

$$f(x_1, ...x_N) = \prod_{i<j}(x_i - x_j) \quad \text{and} \quad g(x_1, ...x_N) = -\prod_{i<j}(x_i - x_j) \ .$$

However, it will generally not be possible to create a chain of mix-servers unless the permutation is restricted to a set that is closed under composition.

For utmost generality, one can modify the protocol to prove that $f(x_1, ...x_N, x_{\pi(1)}, ...x_{\pi(N)}) = 0$ or even $f(x_1, ...x_N, x_{\pi(1)}, ...x_{\pi(N)}) \neq 0$ for any function f that can be computed verifiably. Given a commitment $y = \mathcal{C}(b, s)$, the prover can demonstrate that $b \neq 0$ by computing $t = 1/b$, $z = g^{s'} y^t$ and running a sigma proof of the form

$$\Sigma\text{-proof}\left[r, t, s' \,\middle|\, z = g^{s'} y^t \wedge z/g_1 = g^r\right] \ .$$

As an example, one can prove that a permutation is a derangement, i.e. that $\pi(i) \neq i$ for all i by verifying $\prod_{i=1}^{N} (x_{\pi(i)} - x_i) \neq 0$.

In our exposition we assume for clarity that q is prime, but this is not essential. Composite q can be used, but this requires a more delicate analysis to handle the possibility of non-invertible elements and zero-divisors in the ring $\mathbb{Z}_q$, e.g., the random vectors are no longer vectors, but elements in a module. Even the case where q is unknown can be handled using an approach similar to that of [27].

Acknowledgements

We thank Johan Håstad for helpful discussions.

References

1. Abe, M., Imai, H.: Flaws in some robust optimistic mix-nets. In: Safavi-Naini, R., Seberry, J. (eds.) ACISP 2003. LNCS, vol. 2727, pp. 39–50. Springer, Heidelberg (2003)
2. Adida, B., Wikström, D.: How to shuffle in public. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 555–574. Springer, Heidelberg (2007)
3. Adida, B., Wikström, D.: Offline/online mixing. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 484–495. Springer, Heidelberg (2007)
4. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: 42nd IEEE Symposium on Foundations of Computer Science (FOCS), pp. 136–145. IEEE Computer Society Press, Los Alamitos (2001); Full version at Cryptology ePrint Archive, Report 2000/067 (October 2001), `http://eprint.iacr.org`
5. Chaum, D.: Untraceable electronic mail, return addresses and digital pseudo-nyms. Communications of the ACM 24(2), 84–88 (1981)
6. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 103–118. Springer, Heidelberg (1997)
7. de Hoogh, S., Schoenmakers, B., Skoric, B., Villegas, J.: Verifiable rotation of homomorphic encryptions. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 393–410. Springer, Heidelberg (2009)
8. El Gamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory 31(4), 469–472 (1985)
9. Furukawa, J.: Efficient and verifiable shuffling and shuffle-decryption. IEICE Transactions 88-A(1), 172–188 (2005)
10. Furukawa, J., Miyauchi, H., Mori, K., Obana, S., Sako, K.: An implementation of a universally verifiable electronic voting scheme based on shuffling. In: Blaze, M. (ed.) FC 2002. LNCS, vol. 2357, pp. 16–30. Springer, Heidelberg (2003)
11. Furukawa, J., Sako, K.: An efficient scheme for proving a shuffle. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 368–387. Springer, Heidelberg (2001)
12. Furukawa, J., Sako, K.: An efficient publicly verifiable mix-net for long inputs. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, pp. 111–125. Springer, Heidelberg (2006)

13. Groth, J.: A verifiable secret shuffle of homomorphic encryptions. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 145–160. Springer, Heidelberg (2002)
14. Groth, J.: Linear algebra with sub-linear zero-knowledge arguments. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 192–208. Springer, Heidelberg (2009)
15. Groth, J., Ishai, Y.: Sub-linear zero-knowledge argument for correctness of a shuffle. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 379–396. Springer, Heidelberg (2008)
16. Jakobsson, M., Juels, A.: Mix and match: Secure function evaluation via ciphertexts. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 162–177. Springer, Heidelberg (2000)
17. Neff, A.: A verifiable secret shuffle and its application to e-voting. In: 8th ACM Conference on Computer and Communications Security (CCS), pp. 116–125. ACM Press, New York (2001)
18. Park, C., Itoh, K., Kurosawa, K.: Efficient anonymous channel and all/nothing election scheme. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 248–259. Springer, Heidelberg (1994)
19. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992)
20. Pfitzmann, B.: Breaking an efficient anonymous channel. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 332–340. Springer, Heidelberg (1995)
21. Reistad, T.I., Toft, T.: Secret sharing comparison by transformation and rotation. In: Desmedt, Y. (ed.) ICITS 2007. LNCS, vol. 4883, pp. 169–180. Springer, Heidelberg (2009)
22. Reiter, M.K., Wang, X.: Fragile mixing. In: 11th ACM Conference on Computer and Communications Security (CCS), pp. 227–235. ACM Press, New York (2004)
23. Ryan, P.Y.A., Schneider, S.A.: Prêt à voter with re-encryption mixes. In: Gollmann, D., Meier, J., Sabelfeld, A. (eds.) ESORICS 2006. LNCS, vol. 4189, pp. 313–326. Springer, Heidelberg (2006)
24. Sako, K., Killian, J.: Reciept-free mix-type voting scheme. In: Guillou, L.C., Quisquater, J.-J. (eds.) EUROCRYPT 1995. LNCS, vol. 921, pp. 393–403. Springer, Heidelberg (1995)
25. Wikström, D.: A universally composable mix-net. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 317–335. Springer, Heidelberg (2004)
26. Wikström, D.: A sender verifiable mix-net and a new proof of a shuffle. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 273–292. Springer, Heidelberg (2005)
27. Wikström, D.: A commitment-consistent proof of a shuffle. In: Boyd, C., González Nieto, J. (eds.) ACISP 2009. LNCS, vol. 5594, pp. 407–421. Springer, Heidelberg (2009)