A Non-Interactive Shuffle with Pairing Based Verifiability*

Jens Groth[1]** and Steve Lu[2]***

[1] University College London
j.groth@ucl.ac.uk
[2] University of California, Los Angeles
stevelu@math.ucla.edu

Abstract. A shuffle is a permutation and re-encryption of a set of ciphertexts. Shuffles are for instance used in mix-nets for anonymous broadcast and voting. One way to make a shuffle verifiable is to give a zero-knowledge proof of correctness. All currently known practical zero-knowledge proofs for correctness of a shuffle rely on interaction. We give the first efficient non-interactive zero-knowledge proof for correctness of a shuffle.

Keywords: Shuffle, mix-net, non-interactive zero-knowledge, bilinear group.

1 Introduction

A shuffle is a permutation and re-encryption of a set of ciphertexts. Shuffles are used for instance in mix-nets [Cha81], which in turn are used in protocols for anonymous broadcast and electronic voting. In a typical construction of a mix-net, the users encrypt messages that they want to publish anonymously. They send the encrypted messages to a set of mix-net servers that will anonymize the messages. The first server permutes and re-encrypts the incoming set of messages, i.e., it carries out a shuffle. The next server takes the output from the first server and shuffles these ciphertexts. The protocol continues like this until all servers have permuted and re-encrypted the ciphertexts. After the mixing is complete, the mix-servers may now perform a threshold decryption operation to get out the permuted set of messages. The idea is that if just one mix-server is honest, the messages will be randomly permuted and because of the re-encryption step nobody will know the permutation. The messages therefore appear in random order and cannot be traced back to the senders.

The mix-net protocol we just described is not secure if one of the mix-servers is dishonest. A dishonest mix-server could for instance discard some of the ciphertexts and inject new ciphertexts of its own choosing. It is therefore desirable to make the shuffle verifiable. An obvious way to make the mix-net verifiable is to ask each mix-server to

* Work initiated while participating in Securing Cyberspace: Applications and Foundations of Cryptography and Computer Security, Institute of Pure and Applied Mathematics, UCLA, 2006.

** Work done while at UCLA supported by NSF ITR/Cybertrust grant No. 0456717.

*** Supported by NSF Cybertrust grant No. 0430254.

provide a zero-knowledge proof of its shuffle being correct. The zero-knowledge proof guarantees that the shuffle is correct, yet reveals nothing about the permutation or the re-encryption and therefore preserves the privacy of the mix-net.

Much research has already been done on making shuffles verifiable by providing interactive proofs of correctness [SK95, Abe99, AH01, Nef01, FS01, Gro03, NSNK06, NSNK05, Fur05, Wik05, GL07]. The proofs in these papers are all interactive and rely on the verifier choosing random challenges. Using the Fiat-Shamir heuristic, where the verifier's challenges are computed through the use of a cryptographic hash-function, it is possible to make these proofs non-interactive. As a heuristic argument for the security of these non-interactive proofs, one can prove them secure in the random oracle model [BR93], where the cryptographic hash-function is viewed as a random oracle that outputs a random string. However, Goldwasser and Kalai [GK03] demonstrate that the Fiat-Shamir heuristic sometimes yields insecure non-interactive proofs. Other works casting doubt on the Fiat-Shamir heuristic are [CGH98, Nie02, BBP04, CGH04].

It is still an open problem to construct efficient non-interactive zero-knowledge (NIZK) proofs or arguments for the correctness of a shuffle that do not rely on the random oracle model in the security proof. Such NIZK arguments can be used to reduce the round-complexity of protocols relying on verifiable shuffles. Moreover, interactive zero-knowledge proofs are usually deniable [Pas03]; a transcript of an interactive proof can only convince somebody who knows that the challenges were chosen correctly. NIZK arguments on the other hand are transferable. They consist of a single message that can be distributed and convince anybody that the shuffle is correct.

Obviously, one can apply general NIZK proof techniques to demonstrate the correctness of a shuffle. However, reducing the shuffle proof to a general NP statement and applying a general NIZK to it is very inefficient. Using NIZK techniques developed by Groth, Ostrovsky and Sahai [GOS06b, GOS06a, Gro06, GS07] one can get better performance. Some existing interactive zero-knowledge arguments for correctness of a shuffle naturally fit this framework. For example, it is possible to achieve non-interactive shuffle proofs of size $O(n \log n)$ group elements for a shuffle of n ciphertexts by using Abe and Hoshino's scheme [AH01]. This kind of efficiency still falls short of what can be achieved using interactive techniques and the interactive proofs or arguments that grow linearly in the size of the shuffle do not seem easy to make non-interactive using the techniques of Groth, Ostrovsky and Sahai.

OUR CONTRIBUTION. We offer the first (efficient) non-interactive zero-knowledge argument for correctness of a shuffle. The NIZK argument is in the common reference string model and has perfect zero-knowledge. The security proof of our scheme does not rely on the random oracle model. Instead we make use of recently developed techniques for making non-interactive witness-indistinguishable proofs for bilinear groups by Groth and Sahai [GS07], which draws on earlier work by Groth, Ostrovsky and Sahai [GOS06b, GOS06a, Gro06].

The NIZK argument we suggest is for the correctness of a shuffle of BBS ciphertexts. This cryptosystem, suggested by Boneh, Boyen and Shacham [BBS04], has ciphertexts that consist of 3 group elements for each group element that they encrypt. We consider statements consisting of n input ciphertexts and n output ciphertexts and the claim that the output ciphertexts are a shuffle of the input ciphertexts. Our NIZK arguments consist of $15n$ group elements, which is reasonable in comparison with the statement size, which is $6n$ group elements.

2 Preliminaries and Notation

In this paper, we work over prime order bilinear groups. In other words, we assume there is probabilistic polynomial time algorithm $\mathcal{G}$ that takes a security parameter k as input and outputs (p, G, G_T, e, g), where:

1. p is a prime
2. G and G_T are cyclic groups of order p
3. g is a random generator of G
4. $e : G \times G \to G_T$ is a map with the following properties
   - Bilinearity: $e(g^a, g^b) = e(g, g)^{ab}$ for all $a, b \in \mathbb{Z}_p$
   - Non-degeneracy: $e(g, g)$ generates G_T
5. Group operations and the bilinear map are efficiently computable and group membership is efficiently decidable.

We will for notational simplicity assume that group membership always is checked when appropriate without writing this explicitly.

2.1 BBS Encryption

The BBS cryptosystem was introduced by Boneh, Boyen and Shacham [BBS04]. We work in a bilinear group (p, G, G_T, e, g). The public key is of the form $(f = g^x, h = g^y)$. The secret key is $(x, y) \in (\mathbb{Z}_p^*)^2$. To encrypt $m \in G$, we choose random $s, t \in \mathbb{Z}_p$ and let the ciphertext be

$$(u, v, w) := (f^s, h^t, g^{s+t}m).$$

To decrypt a ciphertext $(u, v, w) \in G^3$, we compute

$$m = u^{-1/x}v^{-1/y}w.$$

The BBS cryptosystem is semantically secure under chosen plaintext attack if the Decisional Linear Problem is hard in the bilinear group. We refer to Section 3.1 for a formal definition of this assumption.

2.2 Shuffling BBS Ciphertexts

The BBS cryptosystem is homomorphic in the sense that entrywise multiplication of two ciphertexts yields an encryption of the product of the plaintexts. We have:

$$(f^s, h^t, g^{s+t}m) \cdot (f^S, h^T, g^{S+T}M) = (f^{s+S}, h^{t+T}, g^{s+S+t+T}mM).$$

It is easy to make a random shuffle of BBS ciphertexts. Given n input ciphertexts, we permute them randomly and then re-encrypt them by multiplying them with random encryptions of 1. Multiplication with encryptions of 1 preserves the plaintexts by the homomorphic property, but the plaintexts now appear in permuted order. If the Decisional Linear Assumption holds, the BBS cryptosystem is semantically secure and thus the permutation is hidden. For notational purposes, we will let $\{x_i\}$ denote $\{x_i\}_{i=1}^n$.

Definition 1. *A* shuffle *of n BBS ciphertexts $\{(u_i, v_i, w_i)\}$ is a list of output ciphertexts $\{(U_i, V_i, W_i)\}$ such that there exists some permutation $\pi \in S_n$ and randomizers $\{(S_i, T_i)\}$ so:*

$$(\forall i) \qquad U_i = u_{\pi(i)} f^{S_i} \quad \wedge \quad V_i = v_{\pi(i)} h^{T_i} \quad \wedge \quad W_i = w_{\pi(i)} g^{S_i + T_i}.$$

2.3 Non-interactive Zero-Knowledge Arguments

We will construct non-interactive zero-knowledge (NIZK) arguments for correctness of a shuffle of n BBS ciphertexts. Informally, such an argument will demonstrate that the shuffle is correct, but will not reveal anything else, in particular the permutation will remain secret. We will now define NIZK arguments with perfect completeness, perfect zero-knowledge and R_{co}-soundness. The notion of co-soundness in NIZK arguments for NP-languages was introduced in the full paper of [GOS06b, GOS06a]. Since it is quite new we will give some further intuition after the formal definitions.

An NIZK argument for R with R_{co}-soundness consists of six probabilistic polynomial time algorithms: a setup algorithm $\mathcal{G}$, a CRS generation algorithm K, a prover P, a verifier V and simulators (S_1, S_2). The setup algorithm $\mathcal{G}$ outputs some initial information gk. The CRS generation algorithm produces a common reference string σ corresponding to the setup. The prover takes as input (gk, σ, x, w) and produces a proof ψ. The verifier takes as input (gk, σ, x, ψ) and outputs 1 if the proof is acceptable and 0 if the proof is rejected. The simulator S_1 takes as input gk and outputs a simulated common reference string σ as well as a simulation trapdoor τ. S_2 takes as input gk, σ, τ, x and simulates a proof ψ.

Definition 2. *We call $(\mathcal{G}, K, P, V, S_1, S_2)$ an NIZK argument for R with R_{co}-soundness if for all non-uniform adversaries $\mathcal{A}$ we have completeness, soundness and zero-knowledge as described below.*

Perfect completeness:

$$\Pr\Big[gk \leftarrow \mathcal{G}(1^k)\ ;\ \sigma \leftarrow K(gk)\ ;\ (x, w) \leftarrow \mathcal{A}(gk, \sigma)\ ;$$
$$\psi \leftarrow P(gk, \sigma, x, w) :\ (gk, x, w) \notin R\ \vee\ V(gk, \sigma, x, \psi) = 1\Big] = 1.$$

Computational R_{co}-soundness:

$$\Pr\Big[gk \leftarrow \mathcal{G}(1^k)\ ;\ \sigma \leftarrow K(gk)\ ;\ (x, \psi, w_{\mathrm{co}}) \leftarrow \mathcal{A}(gk, \sigma) :$$
$$V(gk, \sigma, x, \psi) = 1\ \wedge\ (gk, x, w_{\mathrm{co}}) \in R_{\mathrm{co}}\Big] \approx 0.$$

Perfect zero-knowledge:

$$\Pr\Big[gk \leftarrow \mathcal{G}(1^k)\ ;\ \sigma \leftarrow K(gk)\ ;\ (\mathrm{St}, x, w) \leftarrow \mathcal{A}(gk, \sigma)\ ;$$
$$\psi \leftarrow P(gk, \sigma, x, w) : (gk, x, w) \in R\ \wedge\ \mathcal{A}(\mathrm{St}, \psi) = 1\Big]$$
$$= \Pr\Big[gk \leftarrow \mathcal{G}(1^k)\ ;\ (\sigma, \tau) \leftarrow S_1(gk)\ ;\ (\mathrm{St}, x, w) \leftarrow \mathcal{A}(gk, \sigma)\ ;$$
$$\psi \leftarrow S_2(gk, \sigma, \tau, x) : (gk, x, w) \in R\ \wedge\ \mathcal{A}(\mathrm{St}, \psi) = 1\Big].$$

We remark that if R ignores gk then R defines a language in NP. The definition given here generalizes the notion of NIZK arguments by allowing R to depend on a setup. The setup we have in mind in this paper, is to let gk be a description of a bilinear group. Given gk describing a bilinear group, the relation R defines a *group-dependent* language L. It is common in the cryptographic literature to assume an appropriate finite group or bilinear group has already been chosen and build protocols in this setting, so it is natural to consider NIZK arguments for setup-dependent languages as we do here.

Our definition also differs in the definition of soundness, where we let R_{co} be a relation that specifies what it means to break soundness. Informally, computational R_{co}-soundness can be interpreted as it being infeasible for the adversary to prove $x \in L$ if it knows $x \in L_{\mathrm{co}}$. We remark that the standard definition of soundness is a special type of R_{co}-soundness. If R ignores gk and R_{co} ignores gk, w_{co} and contains all $x \notin L$, then the definition given above corresponds to saying that it is infeasible to construct a valid proof for $x \notin L$.

Let us explain further, why it is worthwhile to consider R_{co}-soundness in the context of non-interactive arguments with perfect zero-knowledge instead of just using the standard definition of soundness. The problem with the standard definition appears when the adversary produces a statement x and a valid NIZK argument without knowing whether $x \in L$ or $x \notin L$. In these cases it may not be possible to reduce the adversary's output to a breach of some underlying (polynomial) cryptographic hardness assumption. Abe and Fehr [AF07] give a more formal argument for this. They consider NIZK arguments with direct black-box reductions to a cryptographic hardness assumption and show that only languages in P/poly can have direct black-box NIZK arguments with perfect zero-knowledge. Since all known constructions of NIZK arguments rely on direct black-box reductions this indicates that the "natural" definition of soundness is not the right definition of soundness for perfect NIZK arguments. We note that for NIZK *proofs* there is no such problem since they are not perfect zero-knowledge except for trivial languages; and in the case of interactive arguments with perfect zero-knowledge this problem does not appear either because the security proofs rely on rewinding techniques which make it possible to extract a witness for the statement being proven.

The generalization to R_{co}-soundness makes it possible to get around the problem we described above. The adversary only breaks R_{co}-soundness when it knows a witness w_{co} for $x \in L_{\mathrm{co}}$. By choosing R_{co} the right way, this witness can make it possible to reduce a successful R_{co}-soundness attack to a breach of a standard polynomial cryptographic complexity assumption.

At this point, one may wonder whether it is natural to consider a soundness definition where we require the adversary to supply some w_{co}. It turns out that many cryptographic schemes assume a setup where such a w_{co} is given automatically. One example is shuffling that we consider in this paper: when setting up a mix-net using a homomorphic threshold cryptosystem, the threshold decryption keys can be used to decrypt the ciphertexts and check whether indeed they do constitute a shuffle or not.

In our paper, the setup algorithm will be $\mathcal{G}$ that outputs a description of a bilinear group. The relation R will consist of statements that contain a public key for the BBS cryptosystem using the bilinear group and a shuffle of n ciphertexts. The witness will be the permutation used in the shuffle as well as the randomness used for re-randomizing the ciphertexts. In other words:

$$R = \Big\{ \Big((p, G, G_T, e, g) \ , \ (f, h, \{(u_i, v_i, w_i)\}, \{(U_i, V_i, W_i)\}) \ , \ (\pi, \{(S_i, T_i)\}) \Big) \ \Big| \ \pi \in S_n \ \wedge \ \forall i : U_i = u_{\pi(i)} f^{S_i} \ \wedge \ V_i = v_{\pi(i)} h^{T_i} \ \wedge \ W_i = w_{\pi(i)} g^{S_i + T_i} \Big\}.$$

The relation $R_{\rm co}$ will consist of non-shuffles. The witness $w_{\rm co}$ will be the decryption key, which makes it easy to decrypt and check that there is no permutation matching the input plaintexts with the output plaintexts. As we remarked above, NIZK arguments for correctness of a shuffle are usually deployed in a context where such a decryption key can be found. It is for instance common in mix-nets that the mix-servers have a threshold secret sharing of the decryption key for the cryptosystem used in the shuffle. NIZK arguments with $R_{\rm co}$-soundness for correctness of a shuffle therefore give us exactly the guarantee we need for the shuffle being correct.

$$R_{\rm co} = \Big\{ \Big((p, G, G_T, e, g) \ , \ (f, h, \{(u_i, v_i, w_i)\}, \{(U_i, V_i, W_i)\}) \ , \ (x, y) \Big) \ \Big| \ x, y \in \mathbb{Z}_p^* \ \wedge \ f = g^x \ \wedge \ h = g^y \ \wedge \ \forall \pi \in S_n \exists i : W_i U_i^{-1/x} V_i^{-1/y} \neq w_{\pi(i)} u_{\pi(i)}^{-1/x} v_{\pi(i)}^{-1/y} \Big\}.$$

2.4 Non-interactive Witness-Indistinguishable Proofs for Bilinear Groups

We will employ the non-interactive proof techniques of Groth and Sahai [GS07]. They allow a prover to give short proofs for the existence of group elements which satisfy a list of so-called pairing product equations. With their techniques, one can prove that there exists $x_1, \ldots, x_n \in G$ and $\phi_1, \ldots, \phi_n \in \mathbb{Z}_p$ such that they simultaneously satisfy a set of pairing product equations, for instance $\prod_{i=1}^n e(a_i, x_i) = 1$ and $\prod_{i=1}^n x_i^{\phi_i} = 1$. One instantiation of their scheme works over bilinear groups where the Decisional Linear Assumption holds.

Their scheme has the following properties. It has a key generation algorithm that outputs a common reference string consisting of 8 group elements. These 8 group elements specify the public key for two commitment schemes: one for group elements in G and one for exponents in $\mathbb{Z}_p$. In their proof, the prover commits to the witness by committing to the group elements $x_1, \ldots, x_n \in G$ and the exponents $\phi_1, \ldots, \phi_n \in \mathbb{Z}_p$. After that the prover makes non-interactive proofs that the committed elements satisfy all the pairing product equations.

There are two ways of setting up the commitment schemes. One can choose the common reference string such that both commitment schemes are perfectly binding, in which case the proof has perfect completeness and perfect soundness. With a perfect binding key, the commitments to group elements are BBS ciphertexts, so we can decrypt the commitments to learn $x_1, \ldots, x_n$.

Another way to choose the common reference string is to have perfectly hiding commitment schemes. In this case, we can set up the commitment to the exponents $\phi_1, \ldots, \phi_n$ as a perfect trapdoor commitment scheme. We can create a commitment and two different openings to respectively 0 and 1 for instance. When we have perfectly hiding keys in the common reference string, the non-interactive proof has perfect completeness and perfect witness-indistinguishability. In other words, an adversary that sees a proof for a statement for which two or more witnesses exist, gets no information whatsoever as to whether one witness or the other was used in the non-interactive proof.

We write $(\sigma_{\text{binding}}, \xi_{\text{extraction}}) \leftarrow K_{\text{binding}}(p, G, G_T, e, g)$, when creating a perfectly binding common reference string with extraction key $\xi_{\text{extraction}}$ for the commitments to group elements in G. We write $(\sigma_{\text{hiding}}, \tau_{\text{trapdoor}}) \leftarrow K_{\text{hiding}}(p, G, G_T, e, g)$ when creating a perfect hiding common reference string with trapdoor τ_{trapdoor} for the commitments to exponents in $\mathbb{Z}_p$. Perfect binding common reference strings and perfect hiding common reference strings are computationally indistinguishable if the Decisional Linear Assumption holds for the bilinear group we are working over.

3 Cryptographic Assumptions

The security of our NIZK argument for correctness of a shuffle will be based on three assumptions: the Decisional Linear Assumption, the Permutation Pairing Assumption and the Simultaneous Pairing Assumption. The BBS cryptosystem and the non-interactive proofs of Groth and Sahai rely on the Decisional Linear Assumption. The other two assumptions are needed for the NIZK argument for correctness of a shuffle. We will now formally define these assumptions and for the two new assumptions give heuristic reasons for believing them by showing that they hold in the generic group model.

3.1 Decisional Linear Assumption

We first recap the Decisional Linear Problem introduced by Boneh, Boyen and Shacham [BBS04]: Given $gk = (p, G, G_T, e, g)$ and $f, h, g, f^s, h^t, g^z \in G$, decide if $z = s + t$.

Definition 3. *The Decisional Linear Assumption holds for $\mathcal{G}$ if for all non-uniform polynomial time adversaries $\mathcal{A}$ we have:*

$$\begin{aligned}
&\Pr\Big[gk := (p, G, G_T, e, g) \leftarrow \mathcal{G}(1^k)\ ;\ f, h \xleftarrow{R} G\ ;\\
&\qquad s, t \xleftarrow{R} \mathbb{Z}_p\ :\ \mathcal{A}(gk, f, h, f^s, h^t, g^{s+t}) = 1\Big]\\
\approx\ &\Pr\Big[gk := (p, G, G_T, e, g) \leftarrow \mathcal{G}(1^k)\ ;\ f, h \xleftarrow{R} G\ ;\\
&\qquad s, t, z \xleftarrow{R} \mathbb{Z}_p\ :\ \mathcal{A}(gk, f, h, f^s, h^t, g^{z}) = 1\Big].
\end{aligned}$$

3.2 Permutation Pairing Assumption

The Permutation Pairing Problem is: Given (p, G, G_T, e, g) and $g_1 := g^{x_1}, \ldots, g_n := g^{x_n}, \gamma_1 := g^{x_1^2}, \ldots, \gamma_n := g^{x_n^2}$ for random $x_1, \ldots, x_n \in \mathbb{Z}_p$ find elements $a_1, \ldots, a_n, b_1, \ldots, b_n \in G$ such that the following holds:

$$\prod_{i=1}^{n} a_i = \prod_{i=1}^{n} g_i$$

$$\prod_{i=1}^{n} b_i = \prod_{i=1}^{n} \gamma_i$$
$$e(a_i, a_i) = e(g, b_i) \text{ for } i = 1 \ldots n$$
$$\{a_i\} \quad \text{is not a permutation of } \{g_i\}$$

Note that if $\{a_i\}$ is a permutation of $\{g_i\}$, then by the third equation $\{b_i\}$ is $\{\gamma_i\}$ permuted in the same way.

Observe that permutations trivially satisfy the first three conditions and not the fourth, but one could imagine some particular choice of the $\{a_i\}$ and $\{b_i\}$ would satisfy all four conditions. The *Permutation Pairing Assumption* holds if finding such a clever choice is computationally infeasible.

Definition 4. *The Permutation Pairing Assumption holds if for all non-uniform polynomial time adversaries $\mathcal{A}$ we have:*

$$\Pr\Big[gk := (p, G, G_T, e, g) \leftarrow \mathcal{G}(1^k) \; ; \; x_1, \ldots, x_n \xleftarrow{R} \mathbb{Z}_p \; ; \\ \{g_i\} := \{g^{x_i}\} \; ; \; \{\gamma_i\} := \{g^{x_i^2}\} \; ; \; (\{a_i\}, \{b_i\}) \leftarrow \mathcal{A}(gk, \{g_i\}, \{\gamma_i\}) \; : \\ \prod_{i=1}^{n} a_i g_i^{-1} = 1 \; \wedge \; \prod_{i=1}^{n} b_i \gamma_i^{-1} = 1 \; \wedge \; (\forall i) \; e(a_i, a_i) = e(g, b_i) \; \wedge \\ \{a_i\} \textit{ is not a permutation of } \{g_i\}\Big] \approx 0$$

3.3 Simultaneous Pairing Assumption

The Simultaneous Pairing Problem is: Given (p, G, G_T, e, g) and $g_1 := g^{x_1}, \ldots, g_n := g^{x_n}, \gamma_1 := g^{x_1^2}, \ldots, \gamma_n := g^{x_n^2}$ for random $x_1, \ldots, x_n \in \mathbb{Z}_p$ find a non-trivial set of elements $\mu_1, \ldots, \mu_n \in G$ such that the following holds:

$$\prod_{i=1}^{n} e(\mu_i, g_i) = 1 \qquad \wedge \qquad \prod_{i=1}^{n} e(\mu_i, \gamma_i) = 1.$$

The intuition behind this problem is that it may be hard to find a set of non-trivial elements to simultaneously satisfy two pairing products of "independent" sets of elements. The *Simultaneous Pairing Assumption* holds if this problem is hard.

Definition 5. *The Simultaneous Pairing Assumption holds if for all non-uniform polynomial time adversaries $\mathcal{A}$ we have:*

$$\Pr\Big[gk := (p, G, G_T, e, g) \leftarrow \mathcal{G}(1^k) \; ; \; x_1, \ldots, x_n \xleftarrow{R} \mathbb{Z}_p \; ; \; \{g_i\} := \{g^{x_i}\} \; ; \\ \{\gamma_i\} := \{g^{x_i^2}\} \; ; \; \{\mu_i\} \leftarrow \mathcal{A}(gk, \{g_i\}, \{\gamma_i\}) \; : \\ \prod_{i=1}^{n} e(\mu_i, g_i) = 1 \; \wedge \; \prod_{i=1}^{n} e(\mu_i, \gamma_i) = 1 \; \wedge \; \exists i : \mu_i \neq 1\Big] \approx 0$$

3.4 Our Assumptions in the Generic Group Model

We will provide heuristic evidence for our new assumptions by showing that they hold in the generic group model [Sho97]. In this model the adversary is restricted to using only generic bilinear group operations and evaluating equality of group elements.

We accomplish this restriction of the adversary by using a model of the bilinear group where we encode the group elements (or equivalently we encode their discrete logarithms) as unique random strings and letting the adversary see only this representation of the group elements. We then provide the adversary with a bilinear group operation oracle such that it can still perform group operations.

Let us give a few more details. We start by picking a random bilinear group $(p, G, G_T, e, g) \leftarrow \mathcal{G}(1^k)$, which the adversary gets as input. We also pick random bijections $[\cdot] : \mathbb{Z}_p \to G$ and $[[\cdot]] : \mathbb{Z}_p \to G_T$. We give the adversary access to an oracle that operates as follows:

- On input $(\mathbf{exp}, a)$ return $[a]$.
- On input $(\mathbf{mult}, [a], [b])$ return $[a + b]$.
- On input $(\mathbf{mult}, [[a]], [[b]])$ return $[[a + b]]$.
- On input $(\mathbf{map}, [a], [b])$ return $[[ab]]$.

This oracle corresponds to the effect exponentiations, group operations and using the bilinear map have on the discrete logarithms of group elements. Please note that other operations such as inversion of a group element for instance can be easily computed using these group operations since the group order p is known to the adversary.

Theorem 1. *The Permutation Pairing Assumption holds in the generic group model.*

Proof. Let us first formulate the Permutation Pairing Assumption in the generic group model. We generate $(p, G, G_T, e, g) \leftarrow \mathcal{G}(1^k)$. We pick $[\cdot] : \mathbb{Z}_p \to G$ and $[[\cdot]] : \mathbb{Z}_p \to G_T$ as random bijective functions. We pick $x_1, \ldots, x_n \leftarrow \mathbb{Z}_p$. We now give the adversary $\mathcal{A}$ the following input: $(p, G, G_T, e, g, \{[x_i]\}, \{[x_i^2]\})$ and access to the bilinear group operation oracle. $\mathcal{A}$ is computationally unbounded but can only make a polynomial number of queries to the bilinear group operation oracle. The challenge for $\mathcal{A}$ is to find $\{([a_i], [b_i])\}$ so:

$$\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} x_i \quad \wedge \quad \sum_{i=1}^{n} b_i = \sum_{i=1}^{n} x_i^2 \quad \wedge \quad \forall i : \; a_i^2 = b_i \quad \wedge \quad \forall \pi \exists i : \; a_i \neq x_{\pi(i)}.$$

In the generic group model we can without loss of generality assume the adversary computes $[a_i], [b_i]$ via repeated calls to the group operation oracle. This means we have

$$a_i = \sum_{j=1}^{n} x_j a_{ij} + \sum_{j=1}^{n} x_j^2 \alpha_{ij} + r_i \qquad , \qquad b_i = \sum_{j=1}^{n} x_j b_{ij} + \sum_{j=1}^{n} x_j^2 \beta_{ij} + s_i$$

for values $\{a_{ij}\}, \{\alpha_{ij}\}, \{r_i\}, \{b_{ij}\}, \{\beta_{ij}\}, \{s_i\}$ that can be deduced from the calls to the group operation oracle.

Consider now the first conditions on the adversary being successful:

$$\sum_{i=1}^{n} a_i - \sum_{i=1}^{n} x_i = 0 \quad \wedge \quad \sum_{i=1}^{n} b_i - \sum_{i=1}^{n} x_i^2 = 0 \quad \wedge \quad \forall i: \; a_i^2 = b_i.$$

These are polynomials over unknowns $x_1, \ldots, x_n$ that are randomly chosen. The adversary only has indirect access to them by using the bilinear group operation oracle. The adversary can choose two strategies for satisfying the equations. It can pick the values $a_{ij}, \alpha_{ij}, r_i, b_{ij}, \beta_{ij}, s_i$ so the polynomials are identical zero in $\mathbb{Z}_p[x_1, \ldots, x_n]$ or it can hope to be lucky that the polynomials evaluate to zero on the random choice of $x_1, \ldots, x_n \leftarrow \mathbb{Z}_p$. The Schwartz-Sippel theorem tells us that a guess according to the latter strategy has only negligible probability of being successful. Since the adversary can access the bilinear group operation oracle only a polynomial number of times, it can only verify a polynomial number of guesses, so the latter strategy has negligible success probability.

Let us now see what happens if the adversary follows the first strategy. The first equation gives us:

$$\sum_{i=1}^{n} \left(\sum_{j=1}^{n} x_j a_{ij} + \sum_{j=1}^{n} x_j^2 \alpha_{ij} + r_i \right) - \sum_{i=1}^{n} x_i = 0.$$

Viewed as a multivariate polynomial equation over vairables $x_1, \ldots, x_n$ we must have for all j, $\sum_{i=1}^{n} a_{ij} = 1$ and $\sum_{i=1}^{n} \alpha_{ij} = 0$ and $\sum_{i=1}^{n} r_i = 0$.

Next, if $\prod_{i=1}^{n} b_i = \sum_{i=1}^{n} x_i^2$ then it must be the case that

$$\sum_{i=1}^{n} \left(\sum_{j=1}^{n} x_j b_{ij} + \sum_{j=1}^{n} x_j^2 \beta_{ij} + s_i \right) - \sum_{i=1}^{n} x_i^2 = 0.$$

When viewed as a polynomial in $x_1, \ldots, x_n$, we see that we must have for all j, $\sum_{i=1}^{n} b_{ij} = 0$ and $\sum_{i=1}^{n} \beta_{ij} = 1$ and $\sum_{i=1}^{n} s_i = 0$.

Finally, if $(\forall i)\ a_i^2 = b_i$ then it must be the case that

$$\begin{aligned}
&\sum_{j=1}^{n}\sum_{k=1}^{n} x_j x_k a_{ij} a_{ik} + \sum_{j=1}^{n}\sum_{k=1}^{n} x_j^2 x_k^2 \alpha_{ij} \alpha_{ik} + r_i^2 \\
&\quad + 2\sum_{j=1}^{n}\sum_{k=1}^{n} x_j x_k^2 a_{ij} \alpha_{ik} + 2\sum_{j=1}^{n} x_j a_{ij} r_i + 2\sum_{j=1}^{n} x_j^2 \alpha_{ij} r_i \\
&\quad = \sum_{j=1}^{n} x_j b_{ij} + \sum_{j=1}^{n} x_j^2 \beta_{ij} + s_i
\end{aligned}$$

Once again by viewing this as a polynomial equation, for all i we must have that $a_{ij}\alpha_{ik} = 0$. Also $a_{ij}a_{ik} = 0$ when $j \neq k$, $r_i^2 = s_i$, $b_{ij} = 2a_{ij}r_i$, $\beta_{ij} = a_{ij}^2 + 2\alpha_{ij}r_i$.

We now consider what the matrix $A = (a_{ij})$ must be. Each row A has at most one non-zero entry by the fact that $a_{ij}a_{ik} = 0$ when $j \neq k$. Also, each column must sum to

1 by $\sum_{i=1}^{n} a_{ij} = 1$. These two facts combined implies A to have exactly one 1 in each column and each row, thus A is a permutation matrix. Since permutation matrices are invertible, from the equations $\sum_{i=1}^{n} a_{ij}\alpha_{ik} = \sum_{i=1}^{n} 0 = 0$, $\sum_{i=1}^{n} a_{ij} r_i = \frac{1}{2}\sum_{i=1}^{n} b_{ij} = 0$, we obtain that $\alpha_{ik} = 0$ and $r_i = 0$. Therefore, the $\{a_i\}$ are a permutation of the $\{x_i\}$. □

Theorem 2. *The Simultaneous Pairing Assumption holds in the generic group model.*

Proof. Let us first formulate the Simulatenous Pairing Assumption in the generic group model. We generate $(p, G, G_T, e, g) \leftarrow \mathcal{G}(1^k)$. We pick $[\cdot] : \mathbb{Z}_p \to G$ and $[[\cdot]] : \mathbb{Z}_p \to G_T$ as random bijective functions. We pick $x_1, \ldots, x_n \leftarrow \mathbb{Z}_p$. We now give the adversary $\mathcal{A}$ the following input: $(p, G, G_T, e, g, \{[x_i]\}, \{[x_i^2]\})$ and access to the bilinear group operation oracle. $\mathcal{A}$ is computationally unbounded but can only make a polymomial number of queries to the bilinear group operation oracle. The challenge for $\mathcal{A}$ is to find non-trivial $\{[mu_i]\}$ so $\sum_{i=1}^{n} \mu_i x_i = 0$ and $\sum_{i=1}^{n} \mu_i x_i^2 = 0$. The Simultaneous Pairing Assumption in the generic model says that any adversary $\mathcal{A}$ has negligible probability of succeeding in this game.

Without loss of generality we can think of $\mathcal{A}$ as being restricted to computing $\{[\mu_i]\}$ using the bilinear group operation oracle only. This means it chooses

$$\mu_i = \sum_{j=1}^{n} x_j a_{ij} + \sum_{j=1}^{n} x_j^2 \alpha_{ij} + r_i$$

for known a_{ij}, α_{ij} and r_i.

A successful adversary chooses these values so both of these equations are satisfied:

$$\sum_{i=1}^{n} \left(\sum_{j=1}^{n} x_j a_{ij} + \sum_{j=1}^{n} x_j^2 \alpha_{ij} + r_i \right) x_i = 0$$
$$\sum_{i=1}^{n} \left(\sum_{j=1}^{n} x_j a_{ij} + \sum_{j=1}^{n} x_j^2 \alpha_{ij} + r_i \right) x_i^2 = 0$$

We can view them as multi-variate polynomials in $x_1, \ldots, x_n$ which are chosen at random. The adversary never sees $x_1, \ldots, x_n$, it only has indirect access to them through the group operation oracle. There are two strategies the adversary can use: It can select a_{ij}, α_{ij}, r_i so the two polynomials have zero-coefficients or it can hope to be lucky that the random choice of $x_1, \ldots, x_n$ actually evaluates zero. The Schwartz-Sippel theorem tells us that a guess has negligible chance of being correct when $x_1, \ldots, x_n$ are chosen at random from $\mathbb{Z}_p$. Since the adversary can access the bilinear group operations oracle only a polynomial number of times, it can only verify the correctness of a polynomial number of guesses. The latter strategy therefore has negligible success-probability.

Let us now consider the former strategy, where the adversary chooses the coefficients of the polynomials in $\mathbb{Z}_p[x_1, \ldots, x_n]$ so they are the zero-polynomials. Looking at the coefficients for the first polynomial we see that we must have $r_i = 0$ and $\alpha_{ij} = 0$. Looking at the coefficients of the second polynomial we see that $a_{ij} = 0$. The adversary can therefore only find the trivial solution $\mu_1 = \ldots = \mu_n = 0$. □

4 NIZK Argument for Correctness of a Shuffle

We will now present an NIZK argument for correctness of a shuffle of BBS ciphertexts. The common reference string contains $2n$ elements $\{g_i := g^{x_i}\}$ and $\{\gamma_i := g^{x_i^2}\}$ for random $x_1, \dots, x_n \in \mathbb{Z}_p$. The statement contains a public key (f, h) and a set of n input ciphertexts $\{(u_i, v_i, w_i)\}$ and a set of output ciphertexts $\{(U_i, V_i, W_i)\}$ that may be a shuffle of the input ciphertexts.

The first part of the NIZK argument consists of setting up pairing product equations that can only be satisfied if indeed we are dealing with a shuffle. The prover will use a set of variables $\{a_i\}$ and $\{b_i\}$ in these pairing product equations. She will set up a Permutation Pairing Problem over these variables to guarantee that $\{(a_i, b_i)\}$ are a permutation of $\{(g_i, \gamma_i)\}$.

Assume now that $\{(a_i, b_i)\}$ are a permutation of $\{(g_i, \gamma_i)\}$. Let $\{m_i\}$ be the plaintexts of $\{(u_i, v_i, w_i)\}$ and $\{M_i\}$ be the plaintexts of $\{(U_i, V_i, W_i)\}$. The prover also sets up equations such that $\prod_{i=1}^n e(a_i, M_i) = \prod_{i=1}^n e(g_i, m_i)$ and $\prod_{i=1}^n e(b_i, M_i) = \prod_{i=1}^n e(\gamma_i, m_i)$. Since $\{(a_i, b_i)\}$ are a permutation of $\{(g_i, \gamma_i)\}$, then there exists a permutation $\pi \in S_n$ so

$$\prod_{i=1}^n e(g_i, M_{\pi^{-1}(i)} m_i^{-1}) = 1 \quad \wedge \quad \prod_{i=1}^n e(\gamma_i, M_{\pi^{-1}(i)} m_i^{-1}) = 1.$$

This is a Simultaneous Pairing Problem, and assuming the hardness of this problem we will have $M_{\pi^{-1}(i)} = m_i$ for all i.

To give further intuition of the construction, consider a naïve protocol where the prover sends the permutation directly to the verifier. Denote $a_i := g_{\pi(i)}$ and $b_i := \gamma_{\pi(i)}$. With $U_i = u_{\pi(i)} f^{S_i}, V_i = v_{\pi(i)} h^{T_i}, W_i = w_{\pi(i)} g^{S_i + T_i}$ we have:

$$\begin{aligned}
\prod_{i=1}^n e(a_i, u_{\pi(i)} f^{S_i}) &= e(\prod_{i=1}^n a_i^{S_i}, f) \prod_{i=1}^n e(g_{\pi(i)}, u_{\pi(i)}) = e(c_u, f) \prod_{i=1}^n e(g_i, u_i) \\
\prod_{i=1}^n e(a_i, v_{\pi(i)} h^{T_i}) &= e(\prod_{i=1}^n a_i^{S_i}, h) \prod_{i=1}^n e(g_{\pi(i)}, v_{\pi(i)}) = e(c_v, h) \prod_{i=1}^n e(g_i, v_i) \\
\prod_{i=1}^n e(a_i, w_{\pi(i)} g^{S_i + T_i}) &= e(\prod_{i=1}^n a_i^{S_i}, g) \prod_{i=1}^n e(g_{\pi(i)}, w_{\pi(i)}) = e(c_w, g) \prod_{i=1}^n e(g_i, w_i),
\end{aligned}$$

where $c_u = \prod_{i=1}^n a_i^{S_i}, c_v = \prod_{i=1}^n a_i^{T_i}$ and $c_w = \prod_{i=1}^n a_i^{S_i + T_i}$. By construction, $c_w = c_u c_v$. In addition, we may look at the equations by pairing the $\{b_i\}$ with the U_i, V_i, and W_i. From this we obtain another three equations, and we define new elements $c'_u = \prod_{i=1}^n b_i^{S_i}$, $c'_v = \prod_{i=1}^n b_i^{T_i}$, $c'_w = c'_u c'_v$. In total we have six equations:

$$\begin{array}{ll}
\prod_{i=1}^n e(a_i, U_i) = e(c_u, f) \prod_{i=1}^n e(g_i, u_i) & \prod_{i=1}^n e(b_i, U_i) = e(c'_u, f) \prod_{i=1}^n e(\gamma_i, u_i) \\
\prod_{i=1}^n e(a_i, V_i) = e(c_v, h) \prod_{i=1}^n e(g_i, v_i) & \prod_{i=1}^n e(b_i, V_i) = e(c'_v, h) \prod_{i=1}^n e(\gamma_i, v_i) \\
\prod_{i=1}^n e(a_i, W_i) = e(c_u c_v, g) \prod_{i=1}^n e(g_i, w_i) & \prod_{i=1}^n e(b_i, W_i) = e(c'_u c'_v, g) \prod_{i=1}^n e(\gamma_i, w_i)
\end{array}$$

A naïve non-interactive argument would be to let the prover sends $\pi, c_u, c_v, c'_u, c'_v$ to the verifier. The verifier can check the six above equations himself for the verification step.

The naive protocol described is complete by observation. We also have the following lemma:

Lemma 1. *The naïve protocol is R_{co}-sound.*

Proof. The idea behind R_{co}-soundness is to look at the underlying messages. If a dishonest prover were to convince a verifier with a non-shuffle as well as produce a witness (decryption key) $w_{\text{co}} = (x, y)$, we can "decrypt" the equations checked by the verifier. Namely, if we let $m_i = u_i^{-1/x} v_i^{-1/y} w_i$ and $M_i = U_i^{-1/x} V_i^{-1/y} W_i$, then by applying the same algebraic manipulations to the equations, we obtain:

$$\begin{aligned}&\Big[\prod_{i=1}^{n} e(a_i, U_i)\Big]^{-1/x} \cdot \Big[\prod_{i=1}^{n} e(a_i, V_i)\Big]^{-1/y} \cdot \Big[\prod_{i=1}^{n} e(a_i, W_i)\Big]\\ = &\Big[e(c_u, f)\prod_{i=1}^{n} e(g_i, u_i)\Big]^{-1/x} \cdot \Big[e(c_v, h)\prod_{i=1}^{n} e(g_i, v_i)\Big]^{-1/y} \cdot \Big[e(c_u c_v, g)\prod_{i=1}^{n} e(g_i, w_i)\Big].\end{aligned}$$

This gives us $\prod_{i=1}^{n} e(a_i, M_i) = e(c_u^{-1}, g) e(c_v^{-1}, g) e(c_u c_v, g) \prod_{i=1}^{n} e(g_i, m_i) = \prod_{i=1}^{n} e(g_i, m_i)$.

In a similar way we can show that $\prod_{i=1}^{n} e(b_i, M_i) = \prod_{i=1}^{n} e(\gamma_i, m_i)$. Observe that the equations may be rearranged to be $\prod_{i=1}^{n} e(\mu_i, g_i) = 1$ and $\prod_{i=1}^{n} e(\mu_i, \gamma_i) = 1$ where $\mu_i = m_i / M_{\pi^{-1}(i)}$. By the Simultaneous Pairing Assumption, it it is infeasible for the prover to find non-trivial μ_i satisfying these two equations and thus we reach a contradiction. □

The downfall of the naïve protocol is that it completely reveals the permutation. In the actual NIZK argument, we will instead argue that there exist elements $\{a_i\}$ and $\{b_i\}$ that satisfy the equations above rather than revealing them directly. We accomplish this by making a GS proof for the set of pairing product equations given earlier. Our NIZK argument is described in Figure 1.

Theorem 3. *The protocol in Figure 1 is a non-interactive perfectly complete, computationally R_{co}-sound, perfect zero-knowledge argument of a correct shuffle of BBS ciphertexts under the Decisional Linear Assumption, Permutation Pairing Assumption, and Simultaneous Pairing Assumption.*

Proof. As we see in the protocol, the prover can generate the witness for the GS proof herself. Perfect completeness follows from the perfect completeness of the GS proofs.

We will now prove that we have perfect zero-knowledge. The simulator $S = (S_1, S_2)$ will generate a transcript as described in Figure 2. By construction, the common reference strings are generated in the same way. The only difference between a real proof and a simulated proof is the witness given to the GS proof. By the perfect witness-indistinguishability of the GS proof, real proofs and simulated proofs are perfectly indistinguishable.

It remains to prove that we have computational R_{co}-soundness. The adversary is trying to output a public key (f, h) and a non-shuffle of n input ciphertexts and n output ciphertexts, a convincing NIZK argument ψ of it being a shuffle, and a decryption key

Setup: Generate a bilinear group $gk := (p, G, G_T, e, g) \leftarrow \mathcal{G}(1^k)$.
Common reference string: Generate a perfectly hiding common reference string $(\sigma_{\text{hiding}}, \tau_{\text{trapdoor}}) \leftarrow K_{\text{hiding}}(p, G, G_T, e, g)$ to get perfectly witness-indistinguishable GS proofs. Pick random $x_1, \ldots, x_n \leftarrow \mathbb{Z}_p$ and compute $\forall i : g_i := g^{x_i}, \gamma_i := g^{x_i^2}$.
The common reference string is $\sigma := (\sigma_{\text{hiding}}, \{g_i\}, \{\gamma_i\})$.
Shuffle statement: Public key (f, h) for the BBS cryptosystem. Input ciphertexts $\{(u_i, v_i, w_i)\}$ and output ciphertexts $\{(U_i, V_i, W_i)\}$.
Prover's input: Permutation $\pi \in S_n$ and randomizers $\{(S_i, T_i)\}$ so $U_i = u_{\pi(i)} f^{S_i}$, $V_i = v_{\pi(i)} h^{T_i}$ and $W_i = w_{\pi(i)} g^{S_i+T_i}$ for all i.
Proof: The prover sets up the following pairing product equations:

$$\phi = 1 \bmod p, \quad d_u^\phi = 1, \quad d_v^\phi = 1, \quad d_w^\phi = 1, \quad (d'_u)^\phi = 1, \quad (d'_v)^\phi = 1, \quad (d'_w)^\phi = 1,$$

$$\prod_{i=1}^n a_i^\phi g_i^{-\phi} = 1, \qquad \prod_{i=1}^n b_i^\phi \gamma_i^{-\phi} = 1, \qquad (\forall i)\; e(a_i, a_i) = e(g, b_i)$$

$$e(d_u, g) \prod e(a_i, U_i) = e(c_u, f) \prod e(g_i, u_i) \qquad e(d'_u, g) \prod e(b_i, U_i) = e(c'_u, f) \prod e(\gamma_i, u_i)$$
$$e(d_v, g) \prod e(a_i, V_i) = e(c_v, h) \prod e(g_i, v_i) \qquad e(d'_v, g) \prod e(b_i, V_i) = e(c'_v, h) \prod e(\gamma_i, v_i)$$
$$e(d_w, g) \prod e(a_i, W_i) = e(c_u c_v, g) \prod e(g_i, w_i)$$
$$e(d'_w, g) \prod e(b_i, W_i) = e(c'_u c'_v, g) \prod e(\gamma_i, w_i)$$

A witness for satisfiability of the equations can be computed as:

$$\phi := 1, \qquad c_u := \prod_{i=1}^n a_i^{S_i}, \qquad c_v := \prod_{i=1}^n a_i^{T_i}, \qquad c'_u := \prod_{i=1}^n b_i^{S_i}, \qquad c'_v := \prod_{i=1}^n b_i^{T_i},$$

$$\forall i : \; a_i := g_{\pi(i)}, \qquad b_i := \gamma_{\pi(i)},$$

and setting the remaining variables to 1. The prover generates a GS proof ψ that there exists an exponent $\phi \in \mathbb{Z}_p$ and group elements $\{a_i\}, \{b_i\}, c_u, c_v, c'_u, c'_v, d_u, d_v, d_w, d'_u, d'_v, d'_w$ that satisfy the equations.
Verification: The verifier accepts the non-interactive argument if and only if the GS proof ψ is valid.

Fig. 1. NIZK Argument for Correct Shuffle of BBS Ciphertexts

(x, y). The relation R_{co} is a polynomial time decidable relation that tests that (x, y) is the decryption key for (f, h) and that indeed we do have a non-shuffle.

We will change the way we construct the common reference string for the NIZK argument. Instead of generating $\sigma = (\sigma_{\text{hiding}}, \{g_i\}, \{\gamma_i\})$ as in the scheme, we return $\sigma := (\sigma_{\text{binding}}, \{g_i\}, \{\gamma_i\})$ where $(\sigma_{\text{binding}}, \xi_{\text{extraction}}) \leftarrow K_{\text{binding}}(p, G, G_T, e, g)$. By the Decisional Linear Assumption, perfect binding and perfect hiding common reference strings for the GS proofs are computationally indistinguishable, so the adversary's success probability only changes negligibly.

The commitment with trivial randomness is now a perfectly binding commitment to the exponent $\phi = 1$. The GS proof is a perfect proof of knowledge of variables $c_u, c_v, c'_u, c'_v, d_u, d_v, d_w, d'_u, d'_v, d'_w, \{a_i\}, \{b_i\}$ satisfying the equations, which can be extracted using $\xi_{\text{extraction}}$. Since $\phi = 1$, the equations demonstrate that $d_u = d_v = d_w = d'_u = d'_v = d'_w = 1$. The elements $\{a_i\}, \{b_i\}$ satisfy a Permutation Pairing problem and the hardness of this problem tells us that with overwhelming probability they are a permutation of $\{(g_i, \gamma_i)\}$. Lemma 1 now gives us that there is negligible probability of $c_u, c_v, c'_u, c'_v, \{a_i\}, \{b_i\}$ satisfying the equations and at the same time the input and output ciphertexts not being a shuffle. $\square$

Simulated common reference string: The simulator S_1 runs the common reference string generation protocol. It sets $\tau := (\tau_{\text{trapdoor}}, x_1, \dots, x_n)$ and outputs (σ, τ).

Shuffle statement: Public key (f, h) for the BBS cryptosystem. Input ciphertexts $\{(u_i, v_i, w_i)\}$ and output ciphertexts $\{(U_i, V_i, W_i)\}$.

Simulator's input: The simulator S_2 receives the shuffle statement and (σ, τ).

Simulated proof: Create a trapdoor commitment with double opening to $\phi = 0$ and $\phi = 1$. Compute

$$d_u := \prod_{i=1}^{n} u_i^{x_i}, \qquad d_v := \prod_{i=1}^{n} v_i^{x_i}, \qquad d_w := \prod_{i=1}^{n} w_i^{x_i},$$

$$d'_u := \prod_{i=1}^{n} u_i^{x_i^2}, \qquad d'_v := \prod_{i=1}^{n} v_i^{x_i^2}, \qquad d'_w := \prod_{i=1}^{n} w_i^{x_i^2}.$$

Set the remaining variables to 1 and create a perfect witness indistinguishable GS proof ψ that there exists an exponent $\phi \in \mathbb{Z}_p$ and group elements $\{a_i\}, \{b_i\}, c_u, c_v, c'_u, c'_v, d_u, d_v, d_w, d'_u, d'_v, d'_w$ that satisfy the required equations.

Fig. 2. Simulated Argument for Correct Shuffle of BBS Ciphertexts

SIZE OF THE NIZK ARGUMENT. To commit to $\phi = 1$ we can use trivial randomness, so the commitment to ϕ does not have to be included in the proof – the verifier can compute it himself. There are $2n + 10$ variables in G and it takes 3 group elements for each commitment, so the commitments contribute a total of $6n + 30$ group elements towards the proof size.

The first 6 equalities cost 9 group elements each for a total of 54 group elements. The next two multi-exponentiation equations cost 9 group elements each for a total of 18 group elements. We then have n pairing product equations of the form $e(a_i, a_i) = e(g, b_i)$ which cost a total of $9n$ group elements. Finally, we have 6 pairing product equations, where one side of the pairings is publicly known and one side is committed. They each cost 3 group elements for a total of 18 group elements.

The total size of the proof is $15n + 120$ group elements. The size of the common reference string is $2n + 8$ group elements.[3]

We remark that the cost of shuffling multiple sets of ciphertexts with the same permutation may be amortized to a constant number of group elements. The first set of ciphertexts costs $15n + 120$ group elements. But we only need to commit to a_i, b_i and

[3] One could wish for a common reference string that has only a constant number of group elements, but currently even all known 3-move zero-knowledge arguments have common reference strings of size $\Omega(n)$.

prove $e(a_i, a_i) = e(g, b_i)$ once. Regardless of n, the subsequent shuffles under the same permutation only cost 120 group elements each.

5 Remark on Shuffling BGN Ciphertexts

Another homomorphic cryptosystem over bilinear groups was introduced by Boneh, Goh and Nissim [BGN05]. This cryptosystem is based on the Subgroup Decision Assumption over composite order bilinear groups. The ciphertexts consist of one group element each, so with n input ciphertexts and n outputs ciphertexts, the shuffle statement contains $2n$ group elements and another group elements to describe the public key. The techniques we have presented in this paper can also be used to shuffle BGN ciphertexts. Assuming the Subgroup Decision Assumption holds and assuming suitable variants of the Permutation Pairing and the Simultaneous Pairing Assumptions hold, we can make an NIZK argument for correctness of a shuffle consisting of $3n + O(1)$ group elements. Since the Subgroup Decision Assumption only holds when factoring the group order is hard, the group elements in this scheme are quite large though.

While this scheme may have applications, we note that there is one subtle issue that one must be careful about. The GS proofs can be instantiated with bilinear groups of composite order where the Subgroup Decision Problem is hard, but they are only secure if the factorization of the composite group is unknown. The decryption key for the cryptosystem is the factorization of the group order. The R_{co}-soundness of the scheme therefore only holds as long as the adversary does not know the decryption key for the cryptosystem. The NIZK argument is therefore not R_{co}-sound as defined in this paper, albeit it will satisfy a suitably weakened R_{co}-soundness definition.

References

[Abe99] Masayuki Abe. Mix-networks on permutation networks. In *proceedings of ASIACRYPT '99*, pages 258–273, 1999.

[AF07] Masayuki Abe and Serge Fehr. Perfect nizk with adaptive soundness. In *proceedings of TCC '07, LNCS series, volume 4392*, pages 118–136, 2007.

[AH01] Masayuki Abe and Fumitaka Hoshino. Remarks on mix-network based on permutation networks. In *proceedings of PKC '01, LNCS series, volume 1992*, pages 317–324, 2001.

[BBP04] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An uninstantiable random-oracle-model scheme for a hybrid encryption problem. In *proceedings of EUROCRYPT '04, LNCS series, volume 3027*, pages 171–188, 2004. Full paper available at `http://eprint.iacr.org/2003/077`.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *proceedings of CRYPTO '04, LNCS series, volume 3152*, pages 41–55, 2004.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF formulas on ciphertexts. In *proceedings of TCC '05, LNCS series, volume 3378*, pages 325–341, 2005.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS '93*, pages 62–73, 1993.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. In *proceedings of STOC '98*, pages 209–218, 1998.

[CGH04] Ran Canetti, Oded Goldreich, and Shai Halevi. On the random-oracle methodology as applied to length-restricted signature schemes. In *proceedings of TCC '04, LNCS series, volume 2951*, pages 40–57, 2004.

[Cha81] David Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[FS01] Jun Furukawa and Kazue Sako. An efficient scheme for proving a shuffle. In *proceedings of CRYPTO '01, LNCS series, volume 2139*, pages 368–387, 2001.

[Fur05] Jun Furukawa. Efficient and verifiable shuffling and shuffle-decryption. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 88-A(1):172–188, 2005.

[GK03] Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the Fiat-Shamir paradigm. In *proceedings of FOCS '03*, pages 102–113, 2003. Full paper available at `http://eprint.iacr.org/2003/034`.

[GL07] Jens Groth and Steve Lu. Verifiable shuffle of large size ciphertexts. In *proceedings of Practice and Theory in Public Key Cryptography - PKC '07, LNCS 4450*, pages 377–392, 2007.

[GOS06a] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for nizk. In *proceedings of CRYPTO '06, LNCS series, volume 4117*, pages 97–111, 2006.

[GOS06b] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero-knowledge for NP. In *proceedings of EUROCRYPT '06, LNCS series, volume 4004*, pages 339–358, 2006.

[Gro03] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. In *proceedings of PKC '03, LNCS series, volume 2567*, pages 145–160, 2003.

[Gro06] Jens Groth. Simulation-sound nizk proofs for a practical language and constant size group signatures. In *proceedings of ASIACRYPT '06, LNCS series*, 2006. Full paper available at `http://www.brics.dk/~jg/NIZKGroupSignFull.pdf`.

[GS07] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. Cryptology ePrint Archive, Report 2007/155, 2007. Available at `http://eprint.iacr.org/2007/155`.

[Nef01] C. Andrew Neff. A verifiable secret shuffle and its application to e-voting. In *proceedings of ACM CCS '01*, pages 116–125, 2001.

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *proceedings of CRYPTO '02, LNCS series, volume 2442*, pages 111–126, 2002.

[NSNK05] Lan Nguyen, Reihaneh Safavi-Naini, and Kaoru Kurosawa. A provably secure and effcient verifiable shuffle based on a variant of the paillier cryptosystem. *Journal of Universal Computer Science*, 11(6):986–1010, 2005.

[NSNK06] Lan Nguyen, Reihaneh Safavi-Naini, and Kaoru Kurosawa. Verifiable shuffles: a formal model and a paillier-based three-round construction with provable security. *International Journal of Informations Security*, 5(4):241–255, 2006.

[Pas03] Rafael Pass. On deniability in the common reference string and random oracle model. In *proceedings of CRYPTO '03, LNCS series, volume 2729*, pages 316–337, 2003.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *proceedings of EUROCRYPT '97, LNCS series, volume 1233*, pages 256–266, 1997.

[SK95] Kazue Sako and Joe Kilian. Receipt-free mix-type voting scheme - a practical solution to the implementation of a voting booth. In *proceedings of EUROCRYPT '95, LNCS series, volume 921*, pages 393–403, 1995.

[Wik05] Douglas Wikström. A sender verifiable mix-net and a new proof of a shuffle. In *proceedings of ASIACRYPT '05, LNCS series, volume 3788*, pages 273–292, 2005.