A verifiable shuffle for the GSW cryptosystem

Martin Strand

Department of Mathematical Sciences, NTNU
martin.strand@ntnu.no

Abstract

This paper provides the first verifiable shuffle specifically for fully homomorphic schemes. A verifiable shuffle is a way to ensure that if a node receives and sends encrypted lists, the content will be the same, even though no adversary can trace individual list items through the node. Shuffles are useful in e-voting, traffic routing and other applications.

We build our shuffle on the ideas and techniques of Groth's 2010 shuffle, but make necessary modifications for a less ideal setting where the randomness and ciphertexts admit no group structure.

The protocol relies heavily on the properties of the so-called gadget matrices, so we have included a detailed introduction to these.

Keywords: verifiable shuffle, fully homomorphic encryption, post-quantum

1 Introduction

A verifiable shuffle is used to prove that two sets of ciphertexts will decrypt to the same values, but without revealing how the sets relate. Such shuffles are well-known for group homomorphic schemes, and are still being developed and improved. Today, shuffles are particularly useful in e-voting and mixnets, in order to make it hard to correlate the input and the output of a node.

Fully homomorphic encryption has also been suggested as a useful primitive for both e-voting and private network routing. Complex voting systems in particular can take advantage of the features of fully homomorphic encryption, but the FHE toolbox is still missing a number of useful protocols. Recent development have brought shuffling for FHE within reach.

Our result starts from Groth's 2010 shuffle [10], which uses an idea from Neff [14]. A polynomial $(X - x_1)(X - x_2) \cdots (X - x_n)$ is obviously unchanged when the roots are permuted. One can then ask the prover to evaluate the polynomial at random points. The probability of two nonidentical polynomials evaluating to the same value at a random point is negligible. While later development have resulted in even more efficient protocols, Groth's 2010 approach has the advantage of simplicity and that there are few compromises: the

protocol satisfies a standard soundness condition and it is honest-verifier zero knowledge. We return to the details in Section 2.

The polynomial mentioned above is hidden in a subprotocol which is used to prove correctness of a shuffle of known content. The subprotocol is completely independent of the encryption scheme, and uses only a homomorphic commitment scheme. It is important to note that the commitment scheme need only be homomorphic with respect to a single operation. We return to the selection of such schemes. The protocol is then completed by binding the secret data – which we want to prove the claim for – to the known content for which one can prove the relation.

Groth's protocol depends crucially on the fact that some group homomorphic schemes are homomorphic both with respect to the message and the randomness. For instance, the product of two ElGamal ciphertexts with messages m_1 and m_2 and randomness r_1 and r_2, will be a new ciphertext encrypting $m_1 m_2$ using $r_1 + r_2$ as randomness. Generally, a necessary requirement for the original shuffle is that the equation

$$\mathsf{Enc}(m_0 \oplus_{\mathcal{M}} m_1; r_0 \oplus_{\mathcal{R}} r_1) = \mathsf{Enc}(m_0; r_0) \oplus_{\mathcal{C}} \mathsf{Enc}(m_1; r_1).$$

holds, where $\oplus_{\mathcal{M}}$, $\oplus_{\mathcal{R}}$ and $\oplus_{\mathcal{C}}$ are the algebraic operations used in the message, randomness and ciphertext groups respectively. Note that $r_1 \oplus r_2$ is an equally likely randomness as either r_1 or r_2.

The noise-based homomorphic schemes do not satisfy the above requirements, since the ciphertext spaces usually are far from being groups at all. The reason is the noise management; even sufficiently many additions will eventually make the ciphertext decrypt to the wrong value, there need not be an identity element, and associativity may not hold, especially for multiplication in combination with noise management techniques. In fact, the Gentry-Sahai-Waters scheme even exploits this property to minimise the noise growth [9].

Furthermore, the homomorphic property does in general not hold concurrently for the messages and the randomisers. It can, however, be possible to compute the noise after an operation for certain simple cases. We show that the abelian group requirement is not necessary, so that a variant of the original protocol is secure also for a noise-based homomorphic scheme.

The final issue is to take advantage of the quantum security of the lattice based encryption schemes, and then make the protocol future-proof. The secrecy requirements for verifiable shuffles is long-term, while soundness is only short-term. This allows us to achieve security against a potential future quantum adversary using a perfectly hiding commitment scheme, since the computational binding property is only necessary until the proof has been verified.

However, using a lattice based commitment scheme by Baum et al. [3], we can also clear the protocol completely of classic cryptography.

1.1 A naive approach

Recall the polynomial $(X-x_1)(X-x_2)\cdots(X-x_n)$, where the roots $x_1, x_2, \ldots, x_n$ are the secret data to be shuffled. Assume we have two sets of ciphertexts, say

$\{E_i\}$ and $\{e_i\}$, and some secret permutation π such that $\mathsf{Dec}(E_i) = \mathsf{Dec}(e_{\pi(i)})$. The straightforward approach to shuffling using fully homomorphic encryption is to compare the two polynomials

$$P_1(X) = (X - e_1)(X - e_2)\cdots(X - e_n)$$
$$P_2(X) = (X - E_1)(X - E_2)\cdots(X - E_n)$$

by requiring the prover to demonstrate that the ciphertext $P_2(e_i)$ for one or more i given by the verifier decrypts to 0. Such proofs exist [3,5], given that the prover has decryption capabilities. Also, it would be straightforward to verify this protocol using multilinear maps [7] with their zero-test abilities. However, at the time of writing, all multilinear map candidates are broken for this application [1]. Additionally, this computation would require a very *deep* circuit, i.e. high degree polynomials, which with today's FHE techniques is forbiddingly expensive.

1.2 Related work

Independently, Costa, Martínez and Morillo [6] have published a shuffle for lattice based schemes. Their shuffle is based on an idea of Wikström using *permutation matrices*. Unfortunately, one cannot guarantee the secrecy of the shuffle due to lack of circuit privacy for their re-encryption procedure. They observe that the RLWE scheme is additively homomorphic, and suggest to re-encrypt by adding an encryption of 0. This idea is sound for group homomorphic schemes since the randomness is near-uniformly distributed over the group. With noise-based schemes, the randomness is typically a Gaussian, so decryption and a following analysis of the noise term can reveal extra information about the ciphertext. As a consequence, the permutation can leak from the ciphertexts regardless of the properties of the zero-knowledge protocol for which the authors provide a proof.

1.3 Our contribution

Our main contribution is the first adaptation of a verifiable shuffle specifically for a FHE cryptosystem under the assumption of equal noise levels in the input ciphertexts. The efficiency is mostly affected by the inherent limitations of the cryptosystem. The assumption can if necessary be met using bootstrapping.

A second contribution is a detailed exposition of the properties of the gadget matrix, and it is our hope that it can be useful for others who need to work with the details of the GSW ciphertexts.

1.4 Outline

We have introduced the main ideas here in Section 1. The upcoming section will in turn describe the concepts we need to build our protocol, such as *gadget matrices*, the GSW cryptosystem, commitment schemes, zero-knowledge protocols, and Groth's original shuffle. Then, in Section 3, we describe our modifications

and present the shuffle in full, including a proof of it being a zero-knowledge argument.

2 Preliminaries

This section introduces the concepts and technical terms needed in this paper to successfully use the Groth shuffle on GSW ciphertexts. Before we discuss the cryptosystem, we look at gadget matrices in detail.

Following the the description of GSW, we will discuss commitment schemes, zero-knowledge proofs and Groth's verifiable shuffle protocol.

2.1 Gadget matrices

Much of the notation will follow Alperin-Sheriff and Peikert [2]. Assume we work in a field $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$, and let $\ell = \lceil \log_2 q \rceil$. One can then define the gadget vector $\vec{g} \in \mathbb{Z}_q^\ell$ as

$$\begin{pmatrix} 1 \\ 2 \\ 4 \\ \vdots \\ 2^{\ell-1} \end{pmatrix}$$

For any $a \in \mathbb{Z}_q$, it is clear that there exist many vectors $\vec{x} \in \mathbb{Z}_q^\ell$ such that $\langle \vec{g}, \vec{x} \rangle = a$. In particular, the vector $\vec{x}$ can be the binary decomposition of a, with all entries 0 or 1. The binary decomposition is the output of the function or algorithm denoted $\vec{g}^{-1}_{\text{det}}$.

Sometimes, we want a random preimage of $\vec{g}$ rather than the binary decomposition. Let $X = \{\vec{x} \mid \langle \vec{x}, \vec{g} \rangle = 0\}$, the set of all preimages of 0. Let $\vec{g}^{-1}_{\text{rand}}$ denote an algorithm that computes $\vec{g}^{-1}_{\text{det}}$ and samples a value $\vec{x}$ from X, typically from a Gaussian distribution with a prescribed radius, and outputs the sum $\vec{g}^{-1}_{\text{det}} + \vec{x}$.

From now on, we will use subscripts when it is necessary to distinguish between the two variants of the algorithm. If no subscript is given, then the discussion applies equally to both.

Next, we can expand the whole process to handle an n-dimensional vector $\vec{a}$ rather than a single value. Define the sparse matrix

$$G = \begin{pmatrix} 1 & \dots & 2^{\ell-1} & & & & & & \\ & & & 1 & \dots & 2^{\ell-1} & & & \\ & & & & & & \ddots & & \\ & & & & & & & 1 & \dots & 2^{\ell-1} \end{pmatrix} \in \mathbb{Z}_q^{n \times n\ell}.$$

The matrix G is known as the *gadget matrix*, and the literature often express it in shorthand as $\vec{g}^T \otimes I_n$.

The map from $\mathbb{Z}_q^{n\ell}$ to $\mathbb{Z}_q^n$ induced by the matrix G is not invertible, but it is easy to find preimages. As with g, the binary decomposition of each coordinate is a preimage. In line with the literature, let G_{det}^{-1} denote this function. It is not linear, since the sum of two binary decompositions need not be all binary again. The output of G^{-1} is a right-inverse for the map G, and we can extend it to $\mathbb{Z}_q^{n\times m}$ by applying G^{-1} column-wise to some $n \times m$ matrix A. As with $\vec{g}^{\,-1}$, we sometimes want random samples, and denote the resulting sampling algorithm by G_{rand}^{-1}. We use the notation $X \leftarrow G_{\text{rand}}^{-1}(A)$ when we want to indicate that we sample from some distribution imposed on the algorithm.

The following properties are straightforward to derive from the above construction.

Lemma 1. *Assume all operations are modulo some q, and let $A \in \mathbb{Z}_q^{n\times m}$ and $\lambda \in \mathbb{Z}_q$ be some scalar. Then,*

1. $G \cdot G^{-1}(I) = I = I_n \in \mathbb{Z}_q^{n\times n}$
2. $G \cdot G^{-1}(A) = A \in \mathbb{Z}_q^{n\times m}$
3. *In particular,* $G \cdot G^{-1}(\lambda G) = \lambda G$

We get a particularly nice structure when applying the G^{-1} algorithm on multiples of G.

Lemma 2. *Assume all operations are modulo some q, and let $\lambda \in \mathbb{Z}_q$ be some scalar with binary decomposition $\sum_{i=0}^{\ell-1} \lambda_i 2^i$. Then*

$$G_{\text{det}}^{-1}(\lambda G) = \begin{pmatrix} \Lambda & & & \\ & \Lambda & & \\ & & \ddots & \\ & & & \Lambda \end{pmatrix} \in \mathbb{Z}_q^{n\ell\times n\ell}$$

where

$$\Lambda = \begin{pmatrix} \lambda_0 & \lambda_{\ell-1} & \cdots & \lambda_1 \\ \lambda_1 & \lambda_0 & \cdots & \lambda_2 \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{\ell-1} & \lambda_{\ell-2} & \cdots & \lambda_0 \end{pmatrix} \in \mathbb{Z}_q^{\ell\times\ell}$$

In particular, $G_{\text{det}}^{-1}(G)$ is the $n\ell \times n\ell$ identity matrix.

The pattern comes from the fact that multiplying by 2 corresponds with one-step shifts in the binary expression of a number.

One can view $G_{\text{det}}^{-1}(\lambda G)$ as a representation of λ, and consider all representations as equivalent (modulo the kernel of G). Then the G^{-1} algorithm is homomorphic on equivalence classes, which is crucial for the GSW cryptosystem. Recall that linear mappings can be represented by matrices. Consider the mapping $G : \mathbb{Z}_q^{n\ell} \to \mathbb{Z}_q^n$ given by $\vec{x} \mapsto G\vec{x}$. This mapping have several right-inverses $H : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n\ell}$ such that

$$(G \circ H)(\vec{x}) = GH\vec{x} = \vec{x},$$

so $G \circ H = \mathrm{id}_{\mathbb{Z}_q^n}$. Fix G^{-1} as one specific such right-inverse. Then, for all H,

$$G^{-1}(\vec{x}) - H(\vec{x}) \in \ker G.$$

As explained above, we can expand the map G to $\mathbb{Z}_q^{n\ell \times m} \to \mathbb{Z}_q^{n \times m}$, and we can expand the right-inverses as well. By the above relation, for each H we then get

$$G^{-1}(A) = HA + B_A, \qquad G(B_A) = 0,$$

and so for scalars $a, b \in \mathbb{Z}_q$, we have

$$\begin{aligned} G^{-1}(aG)G^{-1}(bG) &= (aHG + B_a)(bHG + B_b) \\ &= ab(HGHG + b^{-1}HGB_b + a^{-1}B_aHG + (ab)^{-1}B_aB_b) \\ &= ab(HIG + b^{-1}H \cdot 0 + a^{-1}B_aHG + (ab)^{-1}B_aB_b) \\ &= abHG + B' \qquad (\text{with } GB' = 0) \\ &= G^{-1}(abG) + B' - B_{ab}. \end{aligned}$$

As a consequence, $G^{-1}(aG)G^{-1}(bG)$ and $G^{-1}(abG)$ can be said to encode the same scalar ab, but with a difference which lies in the kernel of G. A similar computation holds for the sum $G^{-1}(aG) + G^{-1}(bG)$.

We can illustrate this with a toy example. Let $q = 7, \ell = 3$ and $n = 3$. Then

$$G = \begin{pmatrix} 1 & 2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 4 \end{pmatrix}.$$

Consider $G_{\det}^{-1}(5G)$ and $G_{\det}^{-1}(3G)$, which will have blocks $\left[\begin{smallmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{smallmatrix}\right]$ and $\left[\begin{smallmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{smallmatrix}\right]$. Both their sum and product will be $\left[\begin{smallmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{smallmatrix}\right]$ which contains a 2, something we cannot avoid since we are computing modulo 7. However, a new encoding of $5 \cdot 3 \equiv 5 + 3 \equiv 1 \pmod 7$ is just the identity matrix.

This is an effect one has to take into account when computing. Still, it is certainly so that the different matrices represent the same value. In particular,

$$\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix},$$

and note that

$$\begin{pmatrix} 1 & 2 & 4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \equiv \begin{pmatrix} 0 & 0 & 0 \end{pmatrix} \pmod 7.$$

In other words, $(1, 1, 1)$ is in the kernel of G.

2.2 The GSW cryptosystem and circuit privacy

The 2013 cryptosystem by Gentry, Sahai and Waters (GSW) [9] is based on hiding the message as an eigenvalue of the ciphertext. The private key is an *approximate eigenvector*. For simplicity, we will use the symmetric formulation by Alperin-Sheriff and Peikert [2], and at the end explain how to make the scheme public key. Let n be an integer, let q be a modulus and $\ell = \lceil \log_2 q \rceil$. Finally, let χ be a subgaussian distribution (Gaussian with very small tails) over $\mathbb{Z}$.

Key generation Let $\vec{\bar{s}} \leftarrow \chi^{n-1}$ coordinate-wise, and output $\vec{s} = (\vec{\bar{s}}, 1)$ as the private key.

Encryption To encrypt a message $\mu \in \{0, 1\}$, choose a random matrix $\bar{C}$ from $\mathbb{Z}_q^{(n-1)\times m}$, where $m = n\ell$, an error vector $\vec{e} \leftarrow \chi^m$ and set $\vec{b}^T = \vec{e}^T - \vec{\bar{s}}^T \bar{C} \pmod q$. Let

$$C = \begin{pmatrix} \bar{C} \\ \vec{b}^T \end{pmatrix} + \mu G.$$

Decryption Given $\vec{s}$ and C, let $\vec{c}$ be the penultimate column of C, and output 0 if $\langle \vec{s}, \vec{c} \rangle \pmod q$ is closer to 0 than 2^{l-2}. Otherwise, output 1.[1]

Addition Add the matrices C_1 and C_2.

Multiplication Define $C_1 \odot C_2$ as $C_1 \cdot G^{-1}(C_2)$.

The cryptosystem is usually only defined for a binary plaintext space, but the definition can be modified even up to the large space $\mathbb{Z}_q$ by modifying the decryption algorithm to extract bits from more columns than the penultimate, and then building the message from the bits. However, this has a strong negative impact on the noise behaviour. When two ciphertexts encrypting binary messages are multiplied, the noise grows far less than with previous FHE cryptosystems. The growth is a function of the encrypted value of the first ciphertext, so larger plaintext spaces can potentially also give worse noise problems. Nonetheless, we will have to assume a large message space for our application, in the order of 160–180 bits, in order to facilitate the scalar multiplications we will perform. This fact is the main drawback of our work.

The original GSW scheme does not achieve *circuit privacy*. Informally, this property guarantees that nobody are able to deduce which circuit output a given ciphertext. Gentry [8] defined the notion by requiring that an encryption of an evaluation of a circuit should be indistinguishable from an encryption of evaluation of the circuit on encrypted data. In other words, evaluate-then-encrypt should be the same as encrypt-then-evaluate. Bourse et al. [4] provide a simulation based definition – capturing mostly the same intuition – and prove that the GSW cryptosystem is circuit private if the multiplication algorithm is slightly modified and all input ciphertexts have low noise from the same

[1] See Alperin-Sheriff and Peikert [2] for a justification of this algorithm.

distribution. The definitions only differ in that Bourse et al. allow the length of the circuit to leak.

Alperin-Sheriff and Peikert [2] proposed to use the G^{-1}_{rand} algorithm instead of G^{-1}_{det} for performance reasons. Bourse et al. go one step further, and also add a matrix which is 0 everywhere except for the bottom row, which constitutes *gaussian shift* on the ciphertext,

$$C_1 \odot C_2 = C_1 G^{-1}_{\text{rand}}(C_2) + \begin{pmatrix} 0 \\ \vec{y}^T \end{pmatrix},$$

where C_1 and C_2 are ciphertexts and $\vec{y}$ is a vector drawn from χ^m. In particular, one can scale C_1 by α by letting $C_2 = \alpha G$. Also note that $\begin{pmatrix} 0 \\ \vec{y}^T \end{pmatrix}$ is a valid encryption of 0. We will use this fact in the upcoming protocol.

Finally, we note that the GSW scheme can be made public-key by publishing the above $\begin{pmatrix} \bar{C} \\ b^T \end{pmatrix}$ as, say, $\hat{A}$ and define encryption as

$$C \leftarrow \hat{A}R + \mu G,$$

where R is a random matrix with entries in $\{-1, 0, 1\}$.

2.3 Commitment schemes

A commitment scheme is an important tool in protocols. The concept allows a player to make a binding promise to use certain values, but without revealing them at the time of the promise. The commitment can later be verified when the committer reveals the opening information. Formally, a commitment scheme consists of three algorithms:

KeyGen On input 1^ℓ, output a public key pk

Commit On input (pk, m, r), return c.

Verify On input (pk, m, r, c), return accept if c is a valid commitment to m, otherwise reject.

We say that (m, r) is an opening of c. The key material will normally be omitted to simplify notation.

Any public key cryptosystem can be turned into a commitment scheme which is unconditionally binding. Pedersen commitments [15] is an example of a scheme that is unconditionally hiding and where binding depends on the discrete logarithm problem being hard. A particularly nice property about the Pedersen scheme is that it is homomorphic; we have

$$\mathsf{Commit}(m_1, r_1) \cdot \mathsf{Commit}(m_2, r_2) = \mathsf{Commit}(m_1 + m_2, r_1 + r_2).$$

A commitment scheme must satisfy two security properties. The scheme must be *hiding*, which means that a commitment to some message m_1 is indistinguishable from a commitment to some other message m_2. Next, it must be

binding, which means that it is hard to find two openings for distinct messages for a single commitment. At most one of these properties may hold unconditionally, but both may hold only computationally.

Lately, Baum et al. [3] proposed a new additively homomorphic commitment scheme based on the Ring-SIS problem. This is conjectured to be safe also against quantum computers. Recall from the introduction that also classical commitment schemes that are unconditionally hiding and computationally binding will remain secure and usable until the adversary has quantum computers readily available, since the binding property is only needed during the protocol execution to provide soundness.

2.4 Zero-knowledge protocols

Zero-knowledge protocols capture the intuition of being able to convince someone else about the validity of some claim, but without revealing any other information.

Definition 1. Let R be a relation, and let $(x, w) \in R$. An *honest-verifier zero-knowledge protocol* $(\mathcal{P}, \mathcal{V})$ for R is a two-party game between a prover $\mathcal{P}$ on input (x, w) and a verifier $\mathcal{V}$ on input x, satisfying

Completeness Whenever $(x, w) \in R$, $\mathcal{V}$ accepts.

Soundness If $(x, w) \notin R$, then for any $\mathcal{P}^*$, $\mathcal{V}$ will only accept with negligible probability.

Honest-verifier zero knowledge (HVZK) There exists a simulator $\mathcal{M}$ running in expected polynomial time on input x such that the output is indistinguishable from the transcripts of $(\mathcal{P}, \mathcal{V})$ run with (x, w) as input to $\mathcal{P}$.

The w is called a *witness* for the relation.

If the prover is only given bounded computationally resources, the protocol is usually called an *argument*, otherwise we call it a *proof*. The zero-knowledge property can be varied by requiring the indistinguishability to hold computationally, statistically or unconditionally.

The simulator can choose all messages arbitrarily. A proof or argument is *special* honest-verifier zero knowledge (SHVZK) if the output of the simulator is indistinguishable from a real transcript if it has to use truly random messages as simulated challenges from the verifier (as opposed to being able to choose such challenges arbitrarily).

To guarantee that a zero-knowledge protocol can be used as a subprotocol in a larger context, Lindell [11] introduced the notion of witness-extended emulation, which loosely speaking requires that there exists a machine that on basis of sufficiently many rounds of the protocol (reusing the prover's commitments) is able to both output a valid witness for the relation as well as a valid simulated transcript. Our use of this property is limited to noting that the shuffle

of known content satisfies it in order for us to be able to use it for our protocol, so we refer to the original source for details.

2.5 Groth's shuffle

In 2010, following up on Neff's idea [14] of proving the validity of a shuffle by using the fact that a polynomial $\prod(X - x_i)$ is stable under permutation of the roots x_i, Groth presented an efficient, yet conceptually simple shuffle [10]. The idea is two-fold. First, one uses the polynomial idea to prove that some c is a commitment to a permutation of messages $m_1, \ldots, m_n$. The values are known by the verifier, but the permutation remains hidden. Next, one binds the secret data to the known data, and proves that the same permutation is still used.

It is important to note that the shuffle of known content is independent of the encryption scheme employed in the main protocol, and only requires a group homomorphic commitment scheme. Later, we can therefore reuse the shuffle of known content completely, and rely on the following properties [10, Theorem 1].

- The shuffle satisfies special honest-verifier zero knowledge with witness-extended emulation.
- If the commitment scheme is statistically hiding we get statistical HVZK.
- If the commitment scheme is unconditionally binding we get unconditional soundness.

Recall that the property of witness-extended emulation guarantees that we can use the SKC protocol as a building block of the full shuffle.

While the SKC protocol only uses the commitment scheme, the outer protocol depends heavily on the encryption scheme, in particular rerandomisation and cancellation of original randomness. Both of these features are less straightforward in FHE schemes than in classical group homomorphic schemes, and call for some modifications to the original protocol.

Remark 1. Groth introduces two security parameters, ℓ_e and ℓ_s, subject to the conditions that

- ℓ_e must be sufficiently large to make it hard to break soundness, i.e. it must be hard to predict a challenge of length ℓ_e,
- For any a sampled from the uniform distribution on $[0, 2^{\ell_e}] \cap \mathbb{Z}$, d and $a + d$, must be statistically indistinguishable whenever d is sampled from the uniform distribution on $[0, 2^{\ell_e+\ell_s}] \cap \mathbb{Z}$, and
- If the commitment space has message space $\mathbb{Z}_q^n$, then $2^{\ell_e+\ell_s} \leq q$.

The second bullet point is to avoid leakage of information whenever $a + d < 2^{\ell_e}$ or $2^{\ell_e+\ell_s} \leq a + d$. Notice that we can achieve the same result with smaller parameters if we employ rejection sampling [12, 13]. The probability of $2^{\ell_e} \leq a + d \leq 2^{\ell_e+\ell_s}$ is approximately $1 - \frac{1}{2^{\ell_s}} - \frac{1}{2^{\ell_s+\ell_e}}$.

The third bullet point is to avoid overflow that would require modular reductions. However, Groth notes that "[w]hen the cryptosystem has a message space where $m^q = 1$ for all messages, this requirement can be waived".

Concretely, Groth suggests $\ell_e = \ell_s = 80$ for the interactive variant, and $\ell_e = 160$ and $\ell_s = 20$ if the protocol is made non-interactive using the Fiat-Shamir heuristic and rejection sampling. We will keep the same parameters for our protocol.

We reached the above probability by considering the uniform distributions on $X = [0, 2^{\ell_e}] \cap \mathbb{Z}$ and $Y = [0, 2^{\ell_e+\ell_s}] \cap \mathbb{Z}$, with probability density functions (PDF) $m_1(X = k) = \frac{1}{2^{\ell_e}}$ and $m_2(Y = k) = \frac{1}{2^{\ell_e+\ell_s}}$. The PDF of $Z = X + Y$ is then the convolution $m_3(Z = j) = \sum_{k=-\infty}^{\infty} m_1(k)m_2(j-k)$. For each j up to $2^{\ell_e} - 1$, there are $j + 1$ combinations, so we get $m_3(j) = \frac{j+1}{2^{2\ell_e+\ell_s}}$ for the first part. For j between 2^{ℓ}_e and $2^{\ell_e+\ell_s}$, there are constantly $2^{\ell_e} + 1$ combinations, and it decreases by 1 for each j in the tail that follows, down to $j = 2^{2\ell_e+\ell_s}$, which is the greatest value we can sample. From this, we can compute the cumulative density function F_3 at the two points $j = 2^{\ell_e}$ and $j = 2^{\ell_e+\ell_s}$, which is approximately $\frac{1}{2^{\ell_s}}$ and $1 - \frac{1}{2^{\ell_e+\ell_s}}$, respectively.

3 Verifiable shuffle for GSW

Now we can combine the tools and ideas above to get a verifiable shuffle for GSW ciphertexts. Let n denote the number of ciphertexts, and recall that ℓ_e and ℓ_s denote security parameters for the zero-knowledge protocol. We now briefly describe the changes that must be made to Groth's shuffle.

The first part of a shuffle is to permute and rerandomise the ciphertexts. Given an ElGamal ciphertext $(a = g^r, b = \mu h^r)$, a new ciphertext will typically look like $(ag^{r'}, bh^{r'})$, and one can easily prove that it is hard to find the correct correspondence between the old and new set as long as r' is random. The fundamental reason is that the randomness of ElGamal forms a group, and that any rerandomisation is indistinguishable from a fresh encryption.

This is not the case for FHE in general and GSW in particular. The randomness is not bounded, and the Eval algorithm will result in a new ciphertext with randomness being a function of both the messages and the randomness of the inputs. We need to employ Bourse et al.'s technique for circuit privacy. Let the old and new ciphertexts be denoted by $\{e_i\}$ and $\{E_i\}$, and the permutation by π.

Ideally, the shuffling circuit should have all old ciphertexts and the permutation as input, such that all old ciphertexts contribute to each new ciphertext,

$$E_i = \sum_{j=1}^{n} e_{\pi(i)} G^{-1}_{\text{rand}}(\delta_{\pi(i),j} G) + \begin{pmatrix} 0 \\ \vec{y}_i^T \end{pmatrix}$$

where $\delta_{a,b}$ is 1 if $a = b$ and 0 otherwise, and $\vec{y}$ is some vector chosen by the circuit privacy algorithm [4].

However, this is causing problems for achieving the completeness property of the protocol, so we have opted for a simpler version. For each i, sample $X_i \leftarrow G^{-1}_{\text{rand}}(G)$, and let

$$E_i = e_{\pi(i)} X_i + \begin{pmatrix} 0 \\ \vec{y}_i^T \end{pmatrix}$$

which is sufficient under the condition that all $\{e_i\}$ have the same noise level and equal-length decryptions. The order is not coincidental. The shuffling circuit is essentially included in X_i, and this order of multiplication hides it [4]. If necessary, enforce the noise-level condition by bootstrapping the ciphertexts before shuffling them. Bootstrapping is an deterministic operation which only requires the public key. Note that one can only measure the noise by using the decryption key.[2]

The original protocol was expressed using multiplications. Since we are using the additive structure of the GSW scheme, we switch from multiplications and exponentiations to additions and scalar multiplications. This is in itself a favourable move, as the efficiency of the original protocol was measured in exponentiations, while additions and scalar multiplications are almost for free in FHE schemes.

Finally, one of the verifications step in the original protocol involved creating a ciphertext using randomness provided by the prover. Since we lack the nice structure on the randomness in the GSW cryptosystem, we need to provide complete ciphertexts instead of just randomness. This requires us to convince the verifier that the ciphertext is "innocent", in the sense that it doesn't encrypt a value that allows the prover to cheat. Fortunately, we can observe that the ciphertext in question will be all zeros except for the bottom row, which guarantees that it can only encrypt 0.

The complete protocol follows.

Precomputation Start with fresh ciphertexts $\{e_i\}$ with equal noise levels. Bootstrap each ciphertext to achieve near-freshness if necessary. Shuffle using a random permutation π and re-encrypt to get new ciphertexts $\{E_i\}$.

Common input Fresh ciphertexts $\{e_i\}$ and shuffled ciphertexts $\{E_i\}$.

Private input to $\mathcal{P}$ Permutation π, matrices $X_i \leftarrow G^{-1}_{\text{rand}}(G)$ and vectors $\vec{y}$ such that for each i,

$$E_i = e_{\pi(i)} X_i + \begin{pmatrix} 0 \\ \vec{y}_i^T \end{pmatrix}$$

[2] An anonymous reviewer pointed out that it is important to ensure that a malicious mix server cannot mark the ciphertexts, typically by using randomness of different size, resulting in more noise. This may lead to a DoS attack unless one employ bootstrapping, but should not compromise secrecy since only the decryption service can measure noise.

Protocol

$\mathcal{P}1$ Select randomness r and r_d for the commitment scheme, and select n random values d_i of length $\ell_e + \ell_s$.

Let

$$c \leftarrow \mathsf{Commit}(\pi(1), \ldots, \pi(n); r)$$
$$c_d \leftarrow \mathsf{Commit}(-d_1, \ldots, -d_n; r_d).$$

Set $D_i \leftarrow G_{\text{rand}}^{-1}(d_i G)$, $\vec{y}_d \leftarrow \chi_{\mathbb{Z}^m}$ and $E_d \leftarrow \sum_{i=1}^n E_i D_i + \begin{pmatrix} 0 \\ \vec{y}_d^T \end{pmatrix}$

Send c, c_d and E_d to the verifier.

$\mathcal{V}1$ Return a set of random numbers $\{t_i\}$ of length ℓ_e.

$\mathcal{P}2$ Set $f_i \leftarrow t_{\pi(i)} + d_i$, compute X'_i such that

$$X'_{\pi(i)} = G_{\text{det}}^{-1}(t_{\pi(i)} G) - X_i G_{\text{det}}^{-1}(f_i G) + X_i D_i,$$

and set $Z = \sum_{i=1}^n \left(\begin{pmatrix} 0 \\ \vec{y}_i^T \end{pmatrix} G_{\text{det}}^{-1}(f_i G) - \begin{pmatrix} 0 \\ \vec{y}_i^T \end{pmatrix} D_i \right) + \begin{pmatrix} 0 \\ \vec{y}_d^T \end{pmatrix}$. Cancel if not $2^{\ell_e} \leq f_i \leq 2^{\ell_e + \ell_s}$ for all i.

Send $\{f_i\}, \{X'_i\}, Z$ to the verifier.

$\mathcal{P}$–$\mathcal{V}$ Run the shuffle of known content to prove that

$$c^\lambda c_d \mathsf{Commit}(f_1, \ldots, f_n) = \mathsf{Commit}(\lambda\pi(1) + t_{\pi(1)}, \ldots, \lambda\pi(n) + t_{\pi(n)}),$$

where λ is a challenge from the verifier.

$\mathcal{V}2$ Verify the following

- The elements c and c_d are in the commitment space
- For all i, $2^{\ell_e} \leq f_i \leq 2^{\ell_e + \ell_s}$
- $GX'_i = 0$ for all i
- The shuffle of known content
- The matrix Z is of the form $\begin{pmatrix} 0 \\ \vec{y}^T \end{pmatrix}$
- $\sum_{i=1}^n E_i G_{\text{det}}^{-1}(f_i G) - \sum_{i=1}^n e_i (G_{\text{det}}^{-1}(t_i) + X'_i) - E_d = Z$

Theorem 1. *Assume that $\{e_i\}$ is a set of fresh ciphertexts. Then the above protocol is a special honest-verifier zero-knowledge argument for correctness of a shuffle of fully homomorphic ciphertexts. If the commitment scheme is statistically binding, then the scheme is an SHVZK proof of a shuffle.*

Proof. Completeness

As shown in Remark 1, the probability of $\mathcal{P}$ aborting is $\frac{1}{2^{\ell_s}} + \frac{1}{2^{\ell_e+\ell_s}}$, which can be made arbitrarily small with a suitable choice of ℓ_s.

We need to check two of the verification equations, the rest is straightforward. Note that X_i comes from the G^{-1} algorithm and encodes a 1. By the discussion in Section 2.1, one can see that $X'_{\pi(i)}$ must encode $-f_i + t_{\pi(i)} + d_i = 0$ for all i, hence $GX'_i = 0$, all i.

Next, we verify that $\sum_{i=1}^{n} E_i G_{\text{det}}^{-1}(f_i G) - \sum_{i=1}^{n} e_i (G_{\text{det}}^{-1}(t_i) + X'_i) - E_d = Z$. This is a tedious, but uncomplicated computation:

$$
\begin{aligned}
&\sum_{i=1}^{n} E_i G_{\text{det}}^{-1}(f_i G) - \sum_{i=1}^{n} e_i (G_{\text{det}}^{-1}(t_i) + X'_i) - E_d \\
&= \sum_{i=1}^{n} E_i G_{\text{det}}^{-1}(f_i G) - \sum_{i=1}^{n} e_{\pi(i)} (G_{\text{det}}^{-1}(t_{\pi(i)}) + X'_{\pi(i)}) - \sum_{i=1}^{n} E_i D_i + \begin{pmatrix} 0 \\ y_d^T \end{pmatrix} \\
&= \sum_{i=1}^{n} \left(e_{\pi(i)} X_i + \begin{pmatrix} 0 \\ y_i^T \end{pmatrix} \right) G_{\text{det}}^{-1}(f_i G) - \sum_{i=1}^{n} e_{\pi(i)} (G_{\text{det}}^{-1}(t_{\pi(i)}) \\
&\quad - X_i G_{\text{det}}^{-1}(f_i G) + G_{\text{det}}^{-1}(t_\pi(i)) + X_i D_i) \\
&\quad - \sum_{i=1}^{n} \left(e_{\pi(i)} X_i + \begin{pmatrix} 0 \\ y_i^T \end{pmatrix} \right) D_i + \begin{pmatrix} 0 \\ y_d^T \end{pmatrix} \\
&= \sum_{i=1}^{n} e_{\pi(i)} \left(X_i (G_{\text{det}}^{-1}(f_i G) - G_{\text{det}}^{-1}(f_i G) + D_i - D_i) - G_{\text{det}}^{-1}(t_{\pi(i)}) \right. \\
&\quad \left. + G_{\text{det}}^{-1}(t_{\pi(i)}) \right) + \sum_{i=1}^{n} \left(\begin{pmatrix} 0 \\ y_i^T \end{pmatrix} G_{\text{det}}^{-1}(f_i G) - \begin{pmatrix} 0 \\ y_i^T \end{pmatrix} D_i \right) + \begin{pmatrix} 0 \\ y_d^T \end{pmatrix} \\
&= Z
\end{aligned}
$$

Soundness

We need to prove that there exists a permutation π, such that $\mathsf{Dec}(e_{\pi(i)}) = \mathsf{Dec}(E_i)$ for all $1 \leq i \leq n$. We can extract the permutation using rewinding, but we will not extract the matrices X_i used to rerandomise the ciphertexts (although we can prove that they must exist, and have been generated in an honest way).

Run the protocol $(\mathcal{P}^*, \mathcal{V})$ until the prover outputs a transcript. Due to the rejection sampling, the prover may try several times. If the verifier would reject the transcript, we output $\perp$. Following the exact same argument as in Groth's original proof, we can extract π and $\{-d_i\}$ using two valid transcripts [10, p. 562].

Because of the commitment we now know that $f_i = t_{\pi(i)} + d_i$, and since $GX'_i = 0$, we know that $\mathsf{Dec}(X'_i) = 0$. Also, we know that $\mathsf{Dec}(Z) = 0$. Recall that we scale a ciphertext C by computing $CG^{-1}(\lambda G)$, hence if we apply the

decryption function to

$$\sum_{i=1}^{n} E_i G_{\text{det}}^{-1}(f_i G) - \sum_{i=1}^{n} e_i (G_{\text{det}}^{-1}(t_i) + X_i') - E_d = Z,$$

we get

$$\begin{aligned}
&\sum_{i=1}^{n} f_i \mathsf{Dec}(E_i) - \sum_{i=1}^{n} (t_i + 0)\mathsf{Dec}(e_i) - \mathsf{Dec}(E_d) \\
&= \sum_{i=1}^{n} t_i \mathsf{Dec}(E_{\pi^{-1}(i)}) + \sum_{i=1}^{n} d_i \mathsf{Dec}(E_i) - \sum_{i=1}^{n} t_i \mathsf{Dec}(e_i) - \mathsf{Dec}(E_d) \\
&= \sum_{i=1}^{n} t_i (\mathsf{Dec}(E_{\pi^{-1}(i)}) - \mathsf{Dec}(e_i)) + \sum_{i=1}^{n} d_i \mathsf{Dec}(E_i) - \mathsf{Dec}(E_d) \\
&= \mathsf{Dec}(Z) = 0.
\end{aligned}$$

Since only one sum depends on $\{t_i\}$, both sums must be 0 individually. Furthermore, since each t_i is unpredictable, each summand must be 0. Hence, $\mathsf{Dec}(E_{\pi^{-1}(i)}) = \mathsf{Dec}(e_i)$, which we wanted to prove.

Note that we can apply the decryption function without actually being able to compute it for unknown ciphertexts.

Special honest-verifier zero knowledge

Let π_0 and π_1 be two permutations, and let $\mathcal{C}_0$ and $\mathcal{C}_1$ be the corresponding shuffle circuits. By circuit privacy, the adversary cannot decide whether $\mathcal{C}_0$ or $\mathcal{C}_1$ was used to generate $\{E_i\}$ from $\{e_i\}$. Hence, the precomputation step does not leak any information.

To prove that the shuffle itself is HVZK given the challenges, we construct a simulator whose output will be indistinguishable from a real protocol transcript. We provide the simulator through a hybrid argument.

Sim I Simulate the shuffle of known content, and select c and c_d as random commitments.

It follows from the properties of the shuffle of known content that Sim I is indistinguishable from a real transcript.

Sim II Construct a random Z from the same distribution as the original. The distribution is hard to give explicitly, but does not depend on secret data, so it can be simulated by choosing the fundamental terms of the sum independently, and adding. Likewise, choose $\{X_i'\}$ by choosing $\{(\bar{X}_i, \bar{d}_i)\}$ under the same distributions as the prover would, and some permutation $\bar{\pi}$. Compute $\{X_i'\}$ by the equation in $\mathcal{P}2$, such that $GX_i' = 0$ for all i. Choose $\{f_i\}$ from the sum of the uniform distributions over $[0, 2^{\ell_e}] \cap \mathbb{Z}$ and $[0, 2^{\ell_e+\ell_s}] \cap \mathbb{Z}$ under the constraint that $2^{\ell_e} \leq f_i \leq 2^{\ell_e+\ell_s}$. Then choose E_d to fit.

The simulated values for Z, $\{X'_i\}$ and $\{f_i\}$ have the same distribution since they are computed from the same formulas as the original, but using new (but identically distributed) random values instead of X_i and π. Then E_d becomes a valid ciphertext by the homomorphic property of GSW. Circuit privacy makes a simulated E_d indistinguishable from a real E_d, and the IND-CPA property of the cryptosystem will finally provide computational SHVZK. □

4 Further work

We have presented a verifiable shuffle for fully homomorphic schemes. Shuffling techniques have evolved further since the protocol we have chosen to forge from, and we believe it would be interesting to see adaptions of newer shuffles.

Furthermore, Groth's original shuffle can be used with a large family of group homomorphic encryption schemes. The result in this paper can only use the GSW scheme, due to the existence of the efficient and computationally simple circuit privacy technique. However, one should pick one's scheme based on what the application needs, so the shuffling primitive should be available for more schemes. This requires more research on techniques for circuit privacy.

Finally, it would be interesting to see an implementation of verifiable shuffling for FHE schemes, coupled with a real-life application. Only then will one be able to see if the parameters and the runtime of the proof will be acceptable. For instance, we predict that the scheme will be unsuitable for applications with many shuffles, such as onion routing. However, for elections, where one can spend minutes or even hours on the process, this protocol may already be mature.

Acknowledgements The author wishes to thank Jens Groth for his useful comments to an early version of this manuscript, as well as to the anonymous reviewers of Voting'18.

References

[1] Martin Albrecht and Alex Davidson. Are graded encoding scheme broken yet? `http://malb.io/are-graded-encoding-schemes-broken-yet.html`, 2017. Accessed 2017-08-30.

[2] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology - CRYPTO 2014*, volume 8616 of *Lecture Notes in Computer Science*, pages 297–314. Springer, 2014.

[3] Carsten Baum, Ivan Damgård, Sabine Oechsner, and Chris Peikert. Efficient commitments and zero-knowledge protocols from ring-sis with applications to lattice-based threshold cryptosystems. Cryptology ePrint Archive, Report 2016/997, 2016. `http://eprint.iacr.org/2016/997`.

[4] Florian Bourse, Rafaël Del Pino, Michele Minelli, and Hoeteck Wee. FHE circuit privacy almost for free. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016*, volume 9815 of *Lecture Notes in Computer Science*, pages 62–89. Springer, 2016.

[5] Christopher Carr, Anamaria Costache, Gareth T. Davies, Kristian Gjøsteen, and Martin Strand. Zero-knowledge proof of decryption for FHE ciphertexts. Cryptology ePrint Archive, Report 2018/026, 2018. `https://eprint.iacr.org/2018/026`.

[6] Núria Costa, Ramiro Martínez, and Paz Morillo. Proof of a shuffle for lattice-based cryptography (full version). Cryptology ePrint Archive, Report 2017/900, 2017. `http://eprint.iacr.org/2017/900`.

[7] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2013.

[8] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. `crypto.stanford.edu/craig`.

[9] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92. Springer, 2013.

[10] Jens Groth. A verifiable secret shuffle of homomorphic encryptions. *J. Cryptology*, 23(4):546–579, 2010.

[11] Yehuda Lindell. Parallel coin-tossing and constant-round secure two-party computation. *J. Cryptology*, 16(3):143–184, 2003.

[12] Vadim Lyubashevsky. Lattice-based identification schemes secure under active attacks. In Ronald Cramer, editor, *Public Key Cryptography - PKC 2008*, volume 4939 of *Lecture Notes in Computer Science*, pages 162–179. Springer, 2008.

[13] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *Advances in Cryptology - ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616. Springer, 2009.

[14] C. Andrew Neff. A verifiable secret shuffle and its application to e-voting. In Michael K. Reiter and Pierangela Samarati, editors, *CCS 2001, Proceedings of the 8th ACM Conference on Computer and Communications Security*, pages 116–125. ACM, 2001.

[15] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology - CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.