Lattice Based Coin Flipping Protocol

1 Preliminaries

1.1 Group, Ring, and Field

Definition 1. A *group* is a set G together with an operation $*$ such that the following properties hold:

1. $*$ is *associative*: for any $a, b, c \in G$
$$a * (b * c) = (a * b) * c$$
2. There exists an *identity* element $e \in G$ such that for all $x \in G$,
$$x * e = e * x = x$$
3. For each element $x \in G$ there exists an *inverse element* $a^{-1} \in G$ such that
$$x * x^{-1} = x^{-1} * x = e$$
4. If the group satisfies: For all $x, y \in G$,
$$x * y = y * x$$
then the group is called *abelian*.

Definition 2. A *ring* $(R, +, \cdot)$ is a set R together with two operations $+$ which is called addition and $\cdot$ which is called multiplication such that:

1. R is an abelian group with respect to addition.
2. $\cdot$ is associative.
3. The *distributive law* holds: for all $a, b, c \in R$ we have
$$a \cdot (b + c) = a \cdot b + a \cdot c$$
and
$$(b + c) \cdot a = b \cdot a + c \cdot a$$
4. A ring is said to be *commutative* if: For all $a, b \in R$ we have
$$a \cdot b = b \cdot a$$

Definition 3. The ring formed by the polynomials with coefficients in ring R is called the *polynomial ring* over R and denoted by $R[X]$.

Definition 4. A commutative ring in which the non-zero elements form a group with respect to multiplication is called a *field*.

Definition 5. A subset J of a ring R is called *ideal* provided J is a subring of R and for all $a \in J$ and $r \in R$ we have $ar \in J$ and $ra \in J$

Definition 6.

1.2 Lattice

A lattice in $\mathbb{R}^n$ is a subgroup of the additive group $\mathbb{R}^n$ which is isomorphic to the additive group $\mathbb{Z}^n$ and spans the real vector space $\mathbb{R}^n$. In other words, linear combination of any basis vectors of $\mathbb{R}^n$ with integer coefficients forms a lattice. Several lattice problems are conjectured to be hard and therefore lattices are very useful in coding theory and cryptography.

Definition 7. Given n linearly independent vectors $b_1, b_2, \ldots, b_n \in R^m$. A *lattice* Λ generated by them is defined as

$$\Lambda = \mathcal{L}(b_1, b_2, \ldots, b_n) = \left\{ \sum_{i=1}^{n} x_i b_i | x_i \in \mathbb{Z} \right\}$$

where n is the rank of the lattice and m is the dimension of the lattice. The lattice is called full-ranked lattice if $n = m$.

Definition 8. Let Λ be a lattice of rank n. For $i \in \{1, \ldots, n\}$ we define the i-th *successive minima* as

$$\lambda_i = inf\{r | dim(span(\Lambda \cap \bar{B}(0, r))) \geq i\}$$

where $\bar{B}(0, r) = \{x \in \mathbb{R}^m | \, \|x\| < r\}$ is the closed ball of radius r around 0.

1.2.1 Several Lattice Problems

1.2.1.1 Closest Vector Problem (CVP)

Search CVP	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ and a target vector $t \in \mathbb{Z}^m$, find $x \in \mathbb{Z}^m$ that minimizes $\|\mathcal{B} - t\|$
Optimization CVP	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ and a target vector $t \in \mathbb{Z}^m$, find $dist(t, \mathcal{L}(\mathcal{B}))$
Decisional CVP	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$, a target vector $t \in \mathbb{Z}^m$ and $r \in \mathbb{Q}$, determine whether $dist(t, \mathcal{L}(\mathcal{B})) \leq r$

The decisional variant of CVP is known to be in **NP**. Since there is a reduction from subset-sum problem which is known to be **NP-complete**, decisional CVP is also **NP-complete**. A solution to the search variant implies solution to the optimization variant and a solution to the optimization variant gives solution to the decisional variant, hence Decisional SVP $\leq$ Optimization SVP $\leq$ Search SVP.

Search CVP$_\gamma$	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ and a target vector $t \in \mathbb{Z}^m$, find a vector $v \in \mathcal{L}(\mathcal{B})$ such that $\|v - t\| \leq \gamma \cdot dist(t, \lambda_1(\mathcal{L}(\mathcal{B})))$
Optimization CVP$_\gamma$	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ and a target vector $t \in \mathbb{Z}^m$, find d such that $d \leq dist(t, \lambda_1(\mathcal{L}(\mathcal{B}) \leq \gamma \cdot d$
GapCVP$_\gamma$	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$, a target vector $t \in \mathbb{Z}^m$ and $r \in \mathbb{Q}$, YES = $\lambda_1(\mathcal{L}(\mathcal{B})) \leq r$ NO = $\lambda_1(\mathcal{L}(\mathcal{B})) > \gamma \cdot r$

1.2.1.2 Shortest Vector Problem (SVP)

Search SVP	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ find a vector $v \in \mathcal{L}(\mathcal{B})$, such that $\|v\| = \lambda_1(\mathcal{L}(\mathcal{B}))$
Optimization SVP	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ find $\lambda_1(\mathcal{L}(\mathcal{B}))$
Decisional SVP	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ and $r \in \mathbb{Q}$, determine whether $\lambda_1(\mathcal{L}(\mathcal{B})) \leq r$

Search SVP$_\gamma$	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ find a vector $v \in \mathcal{L}(\mathcal{B})$, such that $v \neq 0$ and $\|v\| \leq \gamma \cdot \lambda_1(\mathcal{L}(\mathcal{B}))$
Optimization SVP$_\gamma$	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ find find d such that $d \leq \lambda_1(\mathcal{L}(\mathcal{B})) \leq \gamma \cdot d$
GapSVP$_\gamma$	Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$ and $r \in \mathbb{Q}$, YES = $\lambda_1(\mathcal{L}(\mathcal{B})) \leq r$ NO = $\lambda_1(\mathcal{L}(\mathcal{B})) > \gamma \cdot r$

It is easy to see that GapSVP$_\gamma \leq$ Optimization SVP$_\gamma \leq$ Search SVP$_\gamma$, however it is not known if Search SVP$_\gamma$ $\leq$ Optimization SVP$_\gamma$. The exact version of SVP is proven to be **NP-hard** in the l_∞-norm [11], whereas in the other norms it is shown to be **NP-hard** for randomized reductions [1][15].

In [12] we can see that GapSVP$_\gamma \leq$ GapCVP$_\gamma$ by Cook reduction, however whether there exists a deterministic Karp reduction from GapSVP$_\gamma$ to GapCVP$_\gamma$ is still an open question.

1.2.1.3 Shortest Independent Vectors Problem (SIVP)

Given a lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}$, find n independent vectors $v_1, v_2, \ldots, v_n$ such that $\|v_i\| \leq \lambda_i(\mathcal{L}(\mathcal{B}))$.

1.2.1.4 Bounded Distance Decoding (BDD)

Given lattice basis $\mathcal{B} \in \mathbb{Z}^{m \times n}, t \in \mathbb{R}^m$, and $d \in \mathbb{R}$ where $d < \lambda_1(\mathcal{L}(\mathcal{B}))/2$ such that $dist(t, \mathcal{L}(\mathcal{B})) \leq d$. Find the unique $v \in \mathcal{L}(\mathcal{B})$ closest to t. This is equivalent to finding $e \in t + \mathcal{L}(\mathcal{B})$ such that $\|e\| \leq d$.

1.2.1.5 Short Integer Solution (SIS)

Given $\mathbf{A} \in \mathbb{Z}^{m \times n}$ consists of m uniformly random column vecors $a_i \in \mathbb{Z}_q^n$. Find a nonzero vector $x \in \mathbb{Z}^m$ such that:

a. $\|x\| \leq \beta$

b. $\mathbf{A}x = 0 \in \mathbb{Z}_q^n$

where $\sqrt{n \log q} \leq \beta$ and $m \geq n \log q$.

1.2.1.6 Learning With Errors (LWE)

Search LWE: Find a secret $s \in \mathbb{Z}_q^n$ given a sequence of 'approximate' random inner products

$$\begin{aligned} b_1 &= \langle s, a_1 \rangle + e_1 \\ b_2 &= \langle s, a_2 \rangle + e_2 \\ &\vdots \\ b_m &= \langle s, a_m \rangle + e_m \end{aligned}$$

where $a_1, a_2, \ldots, a_m$ are random vectors in $\mathbb{Z}_q^n$ and $e_1, e_2, \ldots, e_m$ are randomly chosen from Gaussian distribution χ over $\mathbb{Z}$ with width $\alpha q > \sqrt{n}$.

Decision LWE : Distinguish $(\mathbf{A}, b^T = s^T\mathbf{A} + e^T)$ from uniform $(\mathbf{A}, b^T)$.

There are several reasons for believing that LWE problem is hard: (1) The best known algorithms to solve this problem runs in exponential time, (2) It is a generalization of LPN problem which is believed to be hard, (3) LWE is hard if we assume the hardness of some other lattice problems such as quantum hardness of GapSVP$_\gamma$ and SIVP$_t$ [19]. Peikert [18] also proved similar results for classical hardness assumption of GapSVP$_{\zeta,\gamma}$. The search and decision variants of LWE are known to be equivalent as shown by Regev [19].

We know that LWE $\leq$ SIS from this observation: If we find a short vector z such that $\mathbf{A}z = 0$, then we can find $b^T z = s^T A + e^T z$ from $b^T z = s^T A + e^T z$. If (A, b^T) is from LWE distribution then $b^T z$ is short, otherwise $b^T z$ is rather well spread.

1.2.2 Coin Flipping Protocol

Coin flipping protocol was first introduced by Blum [6] that involves two players flipping a coin over the telephone. Alice and Bob have just divorced, live in different cities, and want to decide who get the car. They both flip a coin and communicate the outcome on the telephone. Bob would not like to tell Alice the side of coin he gets when he is not sure that Alice is not lying about her outcome. To solve this, Alice and Bob should agree on a completely secure 1-1 one-way function f. Alice choose a random integer x and then compute $f(x)$. She then sends $f(x)$ to Bob who cannot compute x from $f(x)$. Bob tells Alice whether he thinks x is even or odd. At this point, Alice knows whether Bob is correct or not. To convince Bob, she sends x to Bob. Bob wins if his guess is correct and Alice wins otherwise. By following this protocol, it is guaranteed that the result of coin flipping done by Alice is random and that Alice cannot cheat by declaring a different result than the side she obtained.

A more general protocol where more players participate in this protocol require a public bulletin board to post the messages and verify the result. However, if half or more players are corrupted, adversary can bias the output or abort the protocol anytime. Benny et al. [4] introduced a protocol that guarantees the output through threshold verifiable secret sharing (VSS) if the majority of players are honest. However, this protocol requires interactions between the dealer and the players which challenges scalability. One solution is to use publicly verifiable secret sharing scheme (PVSS) which allows anybody to verify the validity of shares and reconstruct secrets without requiring any interaction between dealers and players.

1.2.3 Shamir's Secret Sharing Scheme

Secret sharing is a method that allows a dealer to distribute shares of secret to a number of players where some subset of authorized players can reconstruct the secret together. It was first introduced by Blakley [5] and Shamir [20], motivated by the problem of secure information storage. Currently, there are many applications of secret sharing schemes including: secure multiparty computations, threshold cryptography, access control, attribute-based encryption, and generalized oblivious transfer.

A secret sharing scheme should meet the following requirements:

- **Correctness**: Secret can be reconstructed by any authorized set of players
- **Perfect Privacy**: Every unauthorized set of players cannot learn anything about the secret from the shares

Suppose a secret s will be shared among n players. A (k, n)-threshold secret sharing scheme allows us to divide s into n pieces $s_1, s_2, \ldots s_n$ such that:

1. Knowledge of at least k of s_i makes it easy to compute s.
2. Knowledge of $k-1$ or less s_i cannot determine s.

Shamir's secret sharing scheme is based on polynomial interpolation. For (k, n)-threshold scheme, we pick a random $k-1$ degree polynomial $p(x) = a_0 + a_1 x + \ldots + a_{k-1}x^{k-1}$ where $a_0 = s$. The share s_i is obtained by computing $s_i = p(i)$.

1.2.4 Publicly Verifiable Secret Sharing (PVSS)

PVSS scheme which was introduced by Stadler[] allow anyone to verify the validity of the shares from public information published in the public ledger. In most PVSS schemes, the verification procedure involves interactive proofs of knowledge. These proofs are made noninteractive by the Fiat-Shamir technique.

Let $\mathcal{P} = P_1, \ldots, P_n$ be a set of n participants. Basically, PVSS scheme consists of the following subprotocols:

1. **Setup**: Dealer D publishes the public parameters. Also all participants publish their public key and withholds the corresponding secret key.
2. **Distribution**: For secret s, D creates shares $s_1, \ldots, s_n$ and computes $\hat{s}_i = Enc(s_i, pk_i)$ for each participant $P_i \in \mathcal{P}$. D also publishes $PROOF_D$ to convince the verifier that the published values are the correct encryption of the shares.

3. **Verification**:Verifier V verifies *non-interactively* that the published information is consistent and that every authorized subset of (honest) participants will recover the same secret. If verification fails, the protocol is aborted.
4. **Reconstruction**: Every participant P_i in a qualified subset $A \subset \mathcal{P}$ decrypts $\hat{s}_i$ by using their secret key sk_i. Then, all participants in A exchange their shares s_i together with $PROOF_i$. Every participant in A locally reconstructs the secret from a subset of t correct shares.

1.2.4.1 Properties of PVSS

The properties required for a PVSS scheme are defined as follows:

- **Correctness**: If the dealer and the participants are honest, every qualified subset of participants can reconstruct the secret s in the reconstruction phase.
- **Verifiability**: If a dishonest dealer passes the verification subprotocol, then there exists a unique value s such that the honest participants in any qualified subset with at least t honest participants recover s as the secret.
- **Security**: For an honest dealer, the adversary cannot learn any information about the secret, even after the execution of the reconstruction subprotocol by all honest participants.

1.2.4.2 Computational Secrecy

Since the encrypted shares in PVSS are sent by public channels, an unbounded adversary can decrypt them and compute the secrets. Therefore, unconditional secrecy is not possible in PVSS schemes. The best thing we can do to ensure the security of the scheme is by the notion of compuational secrecy where we are dealing with a computationally bounded adversary.

The following is a formalization of the intuitive notion of semantic security for a PVSS scheme.

Definition 9. (**Indistinguishability of secrets (IND-1)**) A (t, n)-threshold PVSS scheme is *IND1-secret* if any probabilistic polynomial time algorithm $\mathcal{A}$ has a negligible advantage in the following game against a challenger C. During the game, $\mathcal{A}$ can corrupt up to $t-1$ participants and receives their secret keys.

1. C runs the Setup subprotocol and sends the public parameters to $\mathcal{A}$ along with the public keys of uncorrupted participants. C stores the secret keys of those participants.
2. $\mathcal{A}$ sends the public keys of already corrupted participants.
3. C picks two random secrets x_0, x_1 and a random bit $b \in 0, 1$. Then he runs the Distribution subprotocol for secret x_0 and sends all the resulting information to $\mathcal{A}$ along with x_b.
4. C runs the Reconstruction subprotocol for the set of all uncorrupted participants and sends all the messages exchanged via public channels to $\mathcal{A}$. No new corruptions are allowed from this point.
5. $\mathcal{A}$ outputs a guess bit b.

1.2.5 Reed-Solomon Code

We define a $[n, k, d]$ code C to be a linear error correcting code over Z_q of length n, dimension k and minimum distance d. The dual code $C^{\perp}$ of code C is the vector space which consists of all vectors $c^{\perp} \in Z_q^n$ such that $\langle c^{\perp}, c \rangle = 0$ for all $c \in C$.

Reed-Solomon code transforms a message $m = m_1 m_2 \dots m_k \in F^k$ into a codeword $C(m) \in F^n$ by evaluating $p_m(x) = \sum_{i=1}^{k} m_i x^{i-1}$ at n different values of $x_1, x_2, \dots x_n \in F$. Hence, we have

$$C(m) = (p_m(x_1), p_m(x_2), \dots, p_m(x_n))$$

For all m in the message space M, we have $C(m)$ belongs to the Reed Solomon code $RS_k(M)$. This code has parameters $[n, k, n-k+1]$ and a generator matrix of the form

$$G = \begin{pmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \\ x_1^2 & x_2^2 & \dots & x_n^2 \\ \vdots & \vdots & & \vdots \\ x_1^{k-1} & x_2^{k-1} & & x_n^{k-1} \end{pmatrix}$$

such that $C(m) = mG$.

Suppose $s_1, s_2, \dots, s_n$ are shares of secret s obtained from Shamir's secret sharing scheme where $s_i = s + a_1 i + \dots + a_{k-1} i^{k-1}$. If we set $x_i = i$ in G, then $\gamma = (s_1, s_2, \dots, s_n)$ is a codeword of a message $A = (s, a_1, \dots, a_{k-1})$, in other words $\gamma \in RS_k(A)$.

1.2.6 Zero Knowledge Protocol

Zero knowledge proof protocols allow one party to convince another party that they know a certain secret x without revealing any information concerning this value. The party who knows the value x is called the *prover* while the other party is called *verifier*. The protocol must satisfy the following properties:

- **Completeness**: the honest verifier will be convinced by an honest prover if the statement is true.
- **Soundness**: if the statement is false, there is no dishonest prover who can convince the honest verifier except with some negligible probability.
- **Zero-knowledge**: if the statement is true, there is no honest verifier who knows anything about the secret except the fact that the statement is true.

1.2.6.1 Σ-protocol

Zero knowledge proof can be inefficient in some cases when we have to repeat the protocol many times to satisfy one of the properties. One way to obtain an efficient zero knowledge is to use Σ-protocol [9].

Definition 10. A protocol P is said to be a Σ-protocol for relation R if:

- P is in a 3-move form and we have completeness: On input x and private input w to Prover where $(x, w) \in R$, the Verifier always accepts.
- From any x and any pair of accepting conversations on input $x, (a, e, z), (a, e', z')$ where $e \neq e'$, one can efficiently compute w such that $(x, w) \in R$. This is called *special soundness property*.
- There exists a polynomial-time simulator M which on input x and a random e outputs an accepting conversation of the form (a, e, z) with the same probability distribution as conversations between the honest Prover and Verifier on input x. This is called *special honest-verifier property*.

An example of Σ-protocol suggested by Schnorr is as follows:

1. Let p be a prime, q a prime divisor of $p-1$, and g is an element of order q in $\mathbb{Z}_p^*$. Prover chooses r at random in $\mathbb{Z}_q$ sends $a = g^x \bmod p$ to Verifier.
2. Verifier chooses a challenge e at random in $\mathbb{Z}_{2^t}$ where $2^t < q$ and sends it to Prover.
3. Prover sends $z = r + ew \bmod q$ to Verifier. Verifier then checks that p, q are prime, $g^z = ah^e \bmod p$, and g, h have order q, and accepts iff everything is correct.

2 SCRAPE

Cascudo and David [8] introduced a coin flipping protocol for an honest majority that improves the number of exponentiations to $O(n)$ from $O(nt)$ which is obtained from the fact that sharing a secret by using Shamir

secret sharing is equivalent to encoding the secret with Reed-Solomon code. The protocol which is called SCRAPE (Scalable Randomness Attested by Public Entities), is using a publicly verifiable secret sharing scheme (PVSS) that can be instantiated by the Random Oracle Model (ROM) under Decisional Diffie Hellman (DDH) assumption or by the plain model under Decisional Bilinear Square (DBS) assumption.

The SCRAPE protocol runs through three phases below:

1. **Commit**: For $1 \leq j \leq n$, P_j does the following:
   - Execute Distribution phase of the PVSS sub protocol as the Dealer with threshold $t = \frac{n}{2}$
   - Publish the encrypted share $\hat{s}_1^j, \hat{s}_1^j, \ldots, \hat{s}_n^j$ along with the verification information $PROOF_D^j$ on the public ledger
   - Publish a commitment $Com(s^j, r_j)$ with fresh randomness $r_j \in \mathbb{Z}_q$
2. **Reveal**: For every set of encrypted shares $\hat{s}_1^j, \hat{s}_1^j, \ldots, \hat{s}_n^j$ and the verification information $PROOF_D^j$:
   - All participants run the Verification phase of the PVSS sub protocol
   - Let C be the set of all participants who published commitments and valid shares. Once $\frac{n}{2}$ participants have posted their commitments and valid shares, P_j posts $Open(s^j, r_j)$, for $j \in C$
3. **Recovery**: For every particpant $P_a \in C$ that does not publish $Open(s^a; r_a)$ in the Reveal phase:
   - P_j runs the Reconstruction phase of the PVSS protocol posting $\tilde{s}_a^j$ and $PROOF_a^j$ for $1 \leq j \leq n$
   - Once $\frac{n}{2}$ valid decrypted shares are published, every participant reconstructs the secret h^{s^a}

The final randomness is computed as $\rho = \prod_{j \in C} h^{s^j}$

Both the commitment and encryption schemes of this protocol rely on DDH or DBS assumptions which we know to be unsecured against quantum attacks. Therefore, for our protocol we will use lattice based encryption and commitment schemes.

3 Encryption Scheme

Brakerski et al. [7] introduced a fully homomorphic encryption scheme which relies on the hardness of General Learning With Errors (GLWE) problem that represents Learning With Errors (LWE) and Ring Learning With Errors (RLWE) problems.

3.1 The Scheme

For our protocol we will use the RLWE-based scheme. We define the polynomial ring $R = \mathbb{Z}[X]/\langle X^N + 1 \rangle$, where N is a power of 2. The scheme works as follows:

- **Setup**($1^\lambda, 1^\mu$): Choose μ-bit modulus q and parameters $N = N(\lambda, \mu), k = k(\lambda, \mu)$ and $\chi = \chi(\lambda, \mu)$ such that the scheme achieves 2^λ security against known attacks. Let $R = \mathbb{Z}[x]/(x^N + 1)$ and let $params = (q, N, k, \chi)$
- **SecretKeyGen**($params$): Sample $u \leftarrow \chi \subset R_q$. Output

$$\mathbf{sk} = \begin{bmatrix} u \\ 1 \end{bmatrix} \in R_q^2$$

- **PublicKeyGen**($params$): Generate a row vector $\mathbf{B_1} \leftarrow R_q^{k \times 1}$ and a vector $\mathbf{e} \leftarrow \chi^k$. Set $\mathbf{B_2} := -(\mathbf{B_1}u + 2\mathbf{e}^T)$. Set $\mathbf{pk} = \begin{bmatrix} \mathbf{B_1} \\ \mathbf{B_2} \end{bmatrix}$.

- **Enc**($params, pk, m_0$): To encode message $m_0 \in R_2$ set $\mathbf{m} = \begin{bmatrix} 0 \\ m_0 \end{bmatrix} \in R_q^2$, sample $r \leftarrow R_2^k$ and compute

$$\begin{aligned} \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} &= \mathbf{pk} \cdot \mathbf{r} + \mathbf{m} \\ &= \begin{bmatrix} \mathbf{B_1} \\ \mathbf{B_2} \end{bmatrix} \cdot \mathbf{r} + \begin{bmatrix} 0 \\ m_0 \end{bmatrix} \end{aligned}$$

- **Dec**($params, sk, c$): Output $m = [[\langle \mathbf{sk}, \mathbf{v} \rangle]_q]_2$. We can see that the decryption works correctly from:

$$\begin{aligned} \Big[[\langle \mathbf{sk}, \mathbf{c} \rangle]_q\Big]_2 &= \left[\left[\begin{bmatrix} u & 1 \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}\right]_q\right]_2 \\ &= \Big[[uv_1 + v_2]_q\Big]_2 \\ &= \Big[[u \cdot \mathbf{B_1} \cdot \mathbf{r} + \mathbf{B_2} \cdot \mathbf{r} + m_0]_q\Big]_2 \\ &= \Big[\big[u \cdot \mathbf{B_1} \cdot \mathbf{r} - (\mathbf{B_1} u + 2e^T) \cdot \mathbf{r} + m_0\big]_q\Big]_2 \\ &= \Big[\big[m_0 - 2e^T \cdot \mathbf{r}\big]_q\Big]_2 \\ &= m_0 \end{aligned}$$

3.2 Security Assumption

The security of the scheme relies on the ring learning with errors (RLWE) problem which was introduced by Lyubaskevsky, Peikert and Regev [].

Definition 11. For security parameter λ, let $f(x) = x^d + 1$ where $d = d(\lambda)$ is a power of 2. Let $R = \mathbb{Z}[X]/(f(X))$ and let $R_q = R/qR$ where $q = q(\lambda) \geq 2$ is an integer. Let $q = q(\lambda)$ be a distribution over R. The $RLWE_{d,q,\chi}$ problem is to distinguish the following distributions:

1. $(a_i, b_i) \xleftarrow{\$} R_q^2$
2. $(a_i, b_i = a_i \cdot + e_i)$ where $a_i, s \xleftarrow{\$} R_q$ and $e_i \xleftarrow{\$} \chi$

The following theorem shows that Shortest Vector Problem (SVP) can be reduced to RLWE problem.

Theorem 1. For any d that is a power of 2, ring $R = \mathbb{Z}[X]/(X^d + 1)$, prime $q = 1 \bmod d$, and $B = \omega(\sqrt{d \log q})$, there exists distribution χ that outputs element of R of length at most B with high probability such that if there is an efficient algorithm that solves $RLWE_{d,q,\chi}$ then there exists an efficient quantum algorithm for solving $d^{\omega(1)} \cdot (q/B)$-approximate worst-case SVP for ideal lattices over R.

4 Commitment Scheme

Commitment schemes allow us to commit to a certain value and keep it hidden to other people with the intention to reveal the committed value at a later time. The schemes are designed such that we cannot alter the value after the commit phase, which is called the *binding property* of the protocol. In addition, the scheme must also have *hiding property* which means any adversaries will not be able figure out the message that is being committed. If adversaries cannot change their mind after committing even with infinite computing power we call the scheme is *statistically binding*. If adversaries have very large computing resources and their chances to change their mind are very small then the scheme is *computationally binding*. A scheme is called *statistically hiding* if any infinitely powerful adversaries cannot reveal any information about the committed

value from the commitment. Furthermore, we have a *computationally hiding* scheme if a polynomially bounded adversary has a hard time guessing the committed value.

In general, a commitment scheme consists of the following three algorithms :

- **Keygen** : a probabilistic polynomial time (PPT)-algorithm that outputs public parameter PP.
- **Commit** : a PPT-algorithm that on input PP and a message x, will output a random number r and $c = Com(x, r)$. The value c is called the commitment.
- **Open** : a PPT-algorithm that on input PP, a message x, and values c, r will output a bit b.

Since factorization-based cryptographic schemes are not considered secure by quantum attack, researchers are considering other alternatives including lattice based cryptography. Earlier work by Kawachi et al. [14] constructs a string commitment scheme based on SIS assumption. This protocol requires the message space to be restricted to vectors of small norm, otherwise the binding property is lost. Another work by Jain et al. [13] based the hiding property on LPN and additionally they use zero knowledge proof (ZKP) to prove general relations on bit strings. In [22], the scheme is based on Ring-LWE and they build Σ-protocol from it. The main drawback of these schemes is the ZKP has a non-negligible soundness error, hence they require many iterations to have full security.

In order to tackle this problem, Benhamouda et al. 2015 [3] propose a more efficient commitment scheme and ZKP which allows us to commit to a vector and the commitment is only a constant factor larger than the message. In addition, the soundness error is negligible for a single iteration. This protocol is based on ring-LWE for the binding and hiding property.

We will introduce a scheme by Baum et al. [2] which is an upgrade from previous schemes where the commitment can be done to vectors over polynomial rings. The binding property is based on module-LWE whereas the hiding property is based on module-SIS.

4.1 The Setting

Let q be a prime and $r \in \mathbb{N}^+$. Set $N = 2^s$ and define the rings $R = \mathbb{Z}[X]/\langle X^N + 1\rangle$, $R_q = \mathbb{Z}_q[X]/\langle X^N + 1\rangle$. We also define the set S_β as set of all elements $x \in R$ with l_∞-norm at most β. The following Lemma will show that for some prime q, all short elements both in l_∞ and l_2 are invertible.

Lemma 2. Let $N \geq d > 1$ be powers of 2 and $q \equiv 2d \pmod{4d}$ be a prime. Then $X^N + 1$ factors into d irreducible polynomials $X^{N/d} - r_j \pmod{q}$ and any $y \in R_q \setminus \{0\}$ that satisfies

$$\|y\|_\infty < \frac{1}{\sqrt{d}} \cdot q^{\frac{1}{d}} \quad \text{or} \quad \|y\|_2 < q^{\frac{1}{d}}$$

is invertible in R_q

The challenge space for the zero knowledge proof in the opening of the commitment scheme is defined as follow.

Definition 12. The challenge space $\mathcal{C}$ for the zero knowledge proof protocol is defined as

$$\mathcal{C} = \{c \in R_q | \, \|c\|_\infty = 1, \|c\|_1 = \kappa\}$$

The set $\overline{\mathcal{C}}$ is defined as the set of differences $\mathcal{C} - \mathcal{C}$ excluding 0.

If we would like the size of $\mathcal{C}$ to be 2^λ, then we need to set κ such that $\binom{N}{2} \cdot 2^\kappa > 2^\lambda$.

4.2 The Scheme

The scheme consists of the following algorithms:

- **KeyGen**: The public parameters are $\mathbf{A}_1 \in R_q^{n \times k}$ and $\mathbf{A}_2 \in R_q^{l \times k}$ defined as

$$\mathbf{A}_1 = [\mathbf{I}_n \;\; \mathbf{A}_1^{'}], \;\; \text{where } \mathbf{A}_1^{'} \leftarrow_U R_q^{n \times (k-n)}$$
$$\mathbf{A}_2 = [\mathbf{0}^{\mathbf{l} \times \mathbf{n}} \;\; \mathbf{I}_l \;\; \mathbf{A}_2^{'}], \;\; \text{where } \mathbf{A}_2^{'} \leftarrow_U R_q^{l \times (k-n-l)}$$

- **Commit**: To commit to $\mathbf{x} \in R_q^l$, we choose a random polynomial vector $\mathbf{r} \leftarrow_U S_\beta^k$ and output the commitment

$$Com(x; r) = \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{bmatrix} \cdot \mathbf{r} + \begin{bmatrix} 0^n \\ \mathbf{x} \end{bmatrix}$$

- **Open**: A valid opening of a commitment $\begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix}$ is a 3-tuple consisting of an $\mathbf{x} \in R_q^l$, $\mathbf{z} \in R_q^k$, and $f \in \overline{\mathcal{C}}$. The verifier checks that

$$f \cdot \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{bmatrix} \cdot \mathbf{z} + f \cdot \begin{bmatrix} 0^n \\ \mathbf{x} \end{bmatrix}$$

and that for all i, $\|z_i\|_2 \leq 4\sigma\sqrt{N}$ where $\sigma = 11 \cdot \kappa \cdot \beta \cdot \sqrt{kN}$.

4.2.1 Proof for Opening a Commitment

To prove a valid opening of a commitment, we use a zero-knowledge proof protocol Π_{open} which is a 3-move Σ-protocol defined as follows:

$\underline{\Pi_{open}}$

Public Instance-Specific Information: $\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{bmatrix}$

Prover's Information: $r \in S_\beta^k$

Commitment: $\mathbf{c} = \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} = Com(x; r)$

Prover		Verifier
$y \leftarrow_r \mathcal{N}_\sigma^k$ $t := \mathbf{A} \cdot y$		
	$\xrightarrow{t}$	
		$d \leftarrow_r \mathcal{C}$
	$\xleftarrow{d}$	
$z = y + d \cdot r$ Abort with probability $1 - min\left(1, \frac{\mathcal{N}_\sigma^k(z)}{M \cdot \mathcal{N}_{dr,\sigma}^k(z)}\right)$		
	$\xrightarrow{z}$	Write $z = [z_1, \ldots, z_k]^T$ Accept iff $\forall i, \|z_i\|_2 \leq 2\sigma\sqrt{N}$ and $\mathbf{A}_1 \cdot z = t + d \cdot \mathbf{c}_1$

Given a commitment c and a pair of transcripts for $\Pi_{open}(t, d, z), (t, d', z')$ where $d \neq d'$, we can extract a valid opening $(x, r = z - z', f = d - d')$ of c.

4.3 Security Assumptions

The problems used as the hardness assumption of the commitment scheme are Module-SIS problem in the l_2-norm (SKS^2) and Module-LWE problem in the l_∞-norm (DKS^∞).

Definition 13. The $DKS^{\infty}_{n,k,\beta}$ problem asks to distinguish the distribution $[\mathbf{I_n}\ \mathbf{A'}] \cdot \mathbf{y}$ for a short $\mathbf{y}$, from the uniform distribution when given $\mathbf{A'}$. We say that an algorithm $\mathcal{A}$ has advantage ϵ in solving the $DKS^{\infty}_{n,k,\beta}$ problem if

$$\Big| Pr\left[b = 1 | \mathbf{A'} \leftarrow_r R_q^{n\times(k-n)}; \mathbf{y} \leftarrow_r S_\beta^k; b \leftarrow \mathcal{A}\left(\mathbf{A}', [\mathbf{I_n}\ \mathbf{A}'] \cdot \mathbf{y}\right)\right] - Pr\left[b = 1 | \mathbf{A'} \leftarrow_r R_q^{n\times(k-n)}; \mathbf{u} \leftarrow_r R_q^n; b \leftarrow \mathcal{A}\left(\mathbf{A}', u\right)\right] \geq \epsilon$$

Definition 14. The $SKS^2_{n,k,\beta}$ problem asks to find a short vector $\mathbf{y}$ satisfying $[\mathbf{I_n}\ \mathbf{A}'] \cdot \mathbf{y} = 0^n$ when given a random $\mathbf{A}'$. We say that an algorithm $\mathcal{A}$ has advantage ϵ in solving $SKS^2_{n,k,\beta}$ problem if

$$Pr\big[\ \|y_i\| \leq \beta \wedge [\mathbf{I_n}\ \mathbf{A}'] \cdot \mathbf{y} = 0^n | \mathbf{A}' \leftarrow_r R_q^{n\times(k-n)}; 0 \neq \mathbf{y} = [y_1 \dots y_k]^T \leftarrow \mathcal{A}(\mathbf{A}')\big] \geq \epsilon$$

The following lemmas show that the hiding property of the scheme is based on DKS^{∞} problem and the binding property is based on SKS^2 problem.

Lemma 3. For any $x, x \in R^l$, if there exists an algorithm $\mathcal{A}$ that has advantage ϵ in breaking the hiding property of the commitment scheme, then there exists another algorithm $\mathcal{A}'$ that runs in the same time and has advantage ϵ in solving the $DKS^{\infty}_{n+l,k,\beta}$ problem.

Lemma 4. If there is an algorithm $\mathcal{A}$ that can break the binding property of the commitment scheme with probability ϵ, then there is an algorithm $\mathcal{A}'$ that can solve the $SKS^2_{n,k,16\sigma\sqrt{\kappa N}}$ problem with advantage ϵ.

5 Our Protocol

We would like to construct a lattice based coin flipping protocol for an honest majority with Guaranteed Output Delivery (GOD). In order to achieve the guaranteed output delivery property, we will use a publicly verifiable secret sharing scheme (PVSS) that allows any set of participants with a certain threshold to reconstruct the secret.

5.1 Π_{PVSS}

The protocol Π_{PVSS} is a PVSS scheme that will be used in our coin flipping protocol. This protocol is run between a dealer D and players $P_1, P_2, \dots, P_M$. Let q be prime and $r \in \mathbb{N}^+$. We set $R = \mathbb{Z}[X]/\left\langle X^N + 1\right\rangle$ and $R_q = \mathbb{Z}_q[X]/\left\langle X^N + 1\right\rangle$. Protocol Π_{PVSS} proceeds as follows:

1. **Setup**: Party P_i will do the following:
   - 1.1 Run SecretKeyGen, and PublicKeyGen of encryption scheme to obtain $\mathbf{sk} = \begin{bmatrix} u \\ 1 \end{bmatrix} \in R_q^2$ and $\mathbf{pk} = \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \end{bmatrix} \in R_q^{2\times d}$.
   - 1.2 Run KeyGen of commitment scheme to obtain $\mathbf{A}_1 \in R_q^{n\times k}$ and $\mathbf{A}_2 \in R_q^{l\times k}$. Set $\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{bmatrix} \in R_q^{(n+l)\times k}$.
   - 1.3 Post $\mathbf{pk}$ and $\mathbf{A}$ on the public ledger.
2. **Distribution**: The dealer D will do the following:
   - 1.1 Sample a secret $s \in \mathbb{Z}_q$ and coefficients $a_1, \dots, a_t \in \mathbb{Z}_q$ uniformly at random for $t = \frac{M}{2}$ and compute M shares of the secret $s_i = p(i) = s + a_i i + \dots + a_{t-1} i^{t-1}$ for all $i = 1, \dots, M$.
   - 1.2 Suppose $\bar{s}_i = \alpha_{N-1}\alpha_{N-2}\dots\alpha_0$ is binary representation of s_i. Set $\hat{s}_i = \sum_{j=0}^{N-1} \alpha_j x^j$. Now we have $\hat{s}_i \in R_2$ for all $i = 1, \dots, M$.
   - 1.3 Sample $r_1, \dots, r_M \xleftarrow{\$} R_2^d$. Encrypt each share s_i with public key pk_i for all $i = 1, \dots, M$. We have

$$v_i = \text{Enc}(pk_i, s_i) = pk_i \cdot r_i + \begin{bmatrix} 0 \\ s_i \end{bmatrix} = \begin{bmatrix} -B_i \\ b_i \end{bmatrix} \cdot r_i + \begin{bmatrix} 0 \\ s_i \end{bmatrix} \in R_q^2$$

1.4 Choose uniformly random $\rho_i \xleftarrow{\$} R_q^k$ for all $i = 1, \ldots, M$. Compute the commitment of each share

$$c_i = \text{Com}(s_i, \rho_i) = A \cdot \rho_i + \begin{bmatrix} 0 \\ s_i \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \cdot \rho_i + \begin{bmatrix} 0 \\ s_i \end{bmatrix} \in R_q^2$$

where . Also compute the commitment of the secret $c = \text{Com}(s, \rho) = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \cdot \rho + \begin{bmatrix} 0 \\ s \end{bmatrix}$, where $\rho \xleftarrow{\$} R_q^k$.

1.5 Compute $PROOF_1(A_1, A_2, -B, b, v_i, c_i)$ for all $i = 1, \ldots, M$ to prove that the commitment and the encryption encode the same message.

1.6 Publish v_i, c, c_i, and $PROOF_1$ for all $i = 1, \ldots, M$.

3. **Verification**: The Verifier V will do the following:

1.1 Check if $PROOF_1$ provided by D is valid. If the check fails, abort the protocol.

1.2 If the proof is valid, sample a random codeword $g^{\perp} = (g_1, \ldots, g_M) \in R_q^M$ from the dual code $C^{\perp}$ corresponding to the instance of Shamir's (n, t)-threshold secret sharing scheme used by D and run $PROOF_2(A, c_1, \ldots, c_M, g^{\perp})$ with D as Prover to verify that the shares are indeed the valid shares of the secret.

4. **Reconstruction**: If a set of t or more parties Q wishes to reconstruct the secret, each party $P_i \in Q$ publishes their decrypted share s_i. Once every party in Q publishes their decrypted shares, they reconstruct the secret s by Lagrange interpolation.

5.2 $\Pi_{lattice}$

We assume that the Setup phase of Π_{PVSS} protocol has been executed. Protocol $\Pi_{lattice}$ works as follows :

1. **Commit**: For $1 \leq j \leq M$, P_j executes the Distribution phase of Π_{PVSS} protocol as the Dealer with threshold $t = \frac{M}{2}$.

2. **Reveal**: For every set of encrypted shares $v_1^j, \ldots, v_2^j$ and $PROOF_1^j$ belong to P_j, all parties run the Verification phase of the Π_{PVSS} sub protocol. Let $\mathcal{C}$ be the set of all participants who published commitments and valid shares. Once t participants have posted their commitments and valid shares, P_j posts Open(s^j, ρ^j), for $j \in \mathcal{C}$

3. **Recovery**: For every party $P_a \in \mathcal{C}$ that does not publish $Open(s^a, \rho^a)$ in the **Reveal** phase, each party P_j runs the Reconstruction phase of the Π_{PVSS}. .

5.2.1 $PROOF_1$ (encryptions and commitments encode the same message)

$$\underline{PROOF_1(A_1, A_2, B_1, B_2, v, c)}$$

Public Instance-Specific Information: A_1, A_2, B_1, B_2
Prover's Information: r, ρ
Commitment: $\mathbf{c} = \begin{bmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \cdot \rho + \begin{bmatrix} 0 \\ s \end{bmatrix}$
Encryption: $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \cdot r + \begin{bmatrix} 0 \\ s \end{bmatrix}$

Prover		Verifier
$y, y' \xleftarrow{\$} \mathcal{N}_\sigma^k$ $x := A_1 \cdot y$ $x' := B_1 \cdot y'$ $h := A_2 \cdot y - B_2 \cdot y'$		
	$\xrightarrow{x, x', h}$	
		$d \xleftarrow{\$} \mathcal{C}$
	$\xleftarrow{d}$	
$z = y + d \cdot \rho$ $z' = y' + d \cdot r$		
	$\xrightarrow{z, z'}$	Accept iff $A_1 \cdot z = x + d \cdot \mathbf{c}_1$, $B_1 \cdot z' = x' + d \cdot \mathbf{v}_1$ and $A_2 \cdot z - B_2 \cdot z' = h + d \cdot (c_2 - v_2)$

5.2.2 $PROOF_2$ (shares are consistent with secret)

Observe that $w = (s_1, \ldots, s_M) \in RS_t(M)$. Hence $\langle g^\perp, w \rangle = \sum_{i=1}^M g_i s_i = 0$.

$$\underline{PROOF_2(\mathbf{A}, \mathbf{c}_1, \ldots, \mathbf{c}_M, g)}$$

Public Instance-Specific Information: $\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{bmatrix}, g$

Prover's Information: $\rho_1, \ldots, \rho_M \in S_\beta^k$

Commitments: $\mathbf{c}_i = \begin{bmatrix} \mathbf{c}_1^i \\ \mathbf{c}_1^i \end{bmatrix} = Com(x_i; r_i)$ for all $i = 1, \ldots, M$

Prover		Verifier
$y_1, \ldots, y_M \xleftarrow{\$} \mathcal{N}_\sigma^k$ $X := \mathbf{A}_1 \cdot [y_1\ y_2 \ldots y_M]$ $h = \mathbf{A}_2 \cdot \left(\sum_{i=1}^M g_i y_i \right)$	$\xrightarrow{X, h}$	
		$d \xleftarrow{\$} \mathcal{C}$
	$\xleftarrow{d}$	
$Z = [y_1 + d \cdot \rho_1\ , \ldots, y_M + d \cdot \rho_M]$		
	$\xrightarrow{Z}$	Write $Z = [z_1\ z_2 \ldots z_M]$. Accept iff $\mathbf{A}_1 \cdot Z = X + d \cdot [\mathbf{c}_1^1\ \mathbf{c}_2^1 \ldots \mathbf{c}_M^1]$ and $\sum_{i=1}^M g_i \cdot \mathbf{A}_2 \cdot z_i = h + d \cdot \left(\sum_{i=1}^M g_i c_i^2 \right)$

5.3 Properties of PVSS

5.3.1 Correctness

If the dealer and the participants act honestly then they will pass the Verification and Reconstruction phase of the Π_{PVSS} protocol. Since there are $M > t$ participants, we can choose a subset of t participants such that the secret will be recovered.

5.3.2 Verifiability

Lemma 5. From any pair $\{X = (x, x', h), d, Z = (z, z')\}$ and $\{X = (x, x', h), \bar{d}, \bar{Z} = (\bar{z}, \bar{z}')\}$ accepting transcripts of $PROOF1$, where $d \neq \bar{d}$, one can prove that both encryption and commitment scheme encode the same secret s.

Proof. □

5.4 Secrecy

6 *Discussion*

We would like to prove the special soundness property of $PROOF1$.For two valid transcripts $\{X = (x, x', h1), d_1, Z_1 = (z_1, z_1')\}$ and $\{X = (x, x', h), d_2, Z_2 = (z_2, z_2')\}$, we should extract the same secret s from both the commitment and the encryption.

Suppose $f = d_1 - d_2$ and $\bar{z} = z_1 - z_2$.

References

[1] Ajtai, M. 1998, 'Shortest vector problem in L2 is NP-hard for randomized reductions', *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, pp. 10-9.

[2] Baum, C., Damgård, I., Lyubashevsky, V., Oechsner, S. & Peikert, C. 2018, 'More efficient commitments from structured lattice assumptions', *Lecture Notes in Computer Science*, vol. 11035, pp. 368-85.

[3] Benhamouda, F., Krenn, S., Lyubashevsky, V. & Pietrzak, K. 2015, 'Efficient zero-knowledge proofs for commtiments from learning with errors over rings', *Computer Security - ESORICS*, pp. 305-25.

[4] Benny, C., Goldwasser, S., Silvio, M. & Baruch, A. 1985, 'Verifiable secret sharing and achieving simultaneity in the presence of faults', *Annual Symposium on Foundations of Computer Science*, pp. 383-95.

[5] Blakley, G.R. 1979, 'Safeguarding cryptographic keys', *Proc. of the 1979 AFIPS National Computer Conference*, vol. 48, pp. 313-7.

[6] Blum, M. 1983, 'Coin flipping by telephone a protocol for solving impossible problems', *ACM SIGACT News*, vol. 15, no. 1, pp. 23-7.

[7] Brakerski, Z., Gentry, C. & Vaikuntanathan, V. 2014, '(Leveled) Fully homomorphic encryption wihtout bootstrapping', *ACM Transactions on Computation Theory (TOCT)*, vol. 6, no. 3, pp. 1-36.

[8] Cascudo, I. & David, B. 2017, 'Scrape: Scalable randomness attested by public entities', *Applied Cryptography and Network Security*, pp. 537–56.

[9] Damgard, I. 2002, 'On Σ-protocols', *Lecture Notes, University of Aarhus, Department for Computer Science*.

[10] Elgamal, T. 1985, 'A public key cryptosystem and a signature based on discrete logarithms', *IEEE Transactions on Information Theory*, vol. 31, no. 4, pp. 469-72.

[11] Emde Boas, P.V. 1981, 'Another NP-complete problem and the complexity of computing short vectors', *Technical report, University of Amsterdam, Department of Mathematics.*

[12] Goldreich, O., Micciancio, D., Safra, S. & Seifert, J.P. 1999, 'Approximating shortest lattice vectors is not harder than approximating closest lattice vectors', *Information Processing Letters*, vol. 71, no. 2, pp. 55-61.

[13] Jain, A., Krenn, S., Pietrzak, K. & Tentes, A. 2012, 'Commitments and efficient zero-knowledge proofs from learning parity with noise', *Lecture Notes in Computer Science*, vol. 7658, pp. 663-80.

[14] Kawachi, A., Tanaka, K. & Xagawa, K. 2008, 'Concurrently secure identification schemes based on the worst-case hardness of lattice problems', *Lecture Notes in Computer Science*, vol. 5350, pp. 372-89.

[15] Khot, S. 2005, 'Hardness of approximating the shortest vector problem in lattices', *Journal of the ACM (JACM)*, vol. 52, no. 5, pp. 789-808.

[16] McEliece, R.J. & Sarwate, D.V. 1981, 'On sharing secrets and reed solomon codes.', *Communications of the ACM*, vol. 24, no. 9, pp. 583-4.

[17] Paillier, P. 1999, 'Public-key cryptosystems based on composite degree residuosity classes', *Lecture Notes in Computer Science*, vol. 1592, pp. 223-38.

[18] Peikert, C. 2009, 'Public-key cryptosystems from the worst-case shortest vector problem', *Proceedings of the 41st ACM Symposium on Theory of Computing (STOC)*, pp. 333-42.

[19] Regev, O. 2009, 'On lattices, learning with errors, random linear codes, and cryptography', *Journal of the ACM (JACM)*, vol. 56, no. 6, pp. 1-40.

[20] Shamir, A. 1979, 'How to share a secret', *Communications of the ACM*, vol. 22, pp. 612-3.

[21] Shor, P.W. 1994, 'Algorithms for quantum computation: discret logarithms and factoring', *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124-34.

[22] Xie, X., Xue, R. & Wang, M. 2013, 'Zero knowledge proofs from ring-LWE', *Lecture Notes in Computer Science*, vol. 8257, pp. 57-73.