COIN FLIPPING PROTOCOL

1 The Project

We would like to construct a coin flipping protocol for an honest majority with some additional properties: Guaranteed Output Delivery (GOD) and quantum resistant. 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. The encryption and commitment schemes that will be considered are lattice-based which are believed to be quantum-safe. The schemes should be homomorphic to allow us to do computation on encrypted messages.

2 The Protocol

Suppose there are M parties participate in the protocol. In general the protocol is divided into three phases based on[2]:

1. **Commit**:
   - 1.1 Each party P_j picks a secret s^j randomly and computes M shares by Shamir's secret sharing scheme with threshold $t = \frac{M}{2}$.
   - 1.2 The encrypted shares are published on the public ledger along with the commitment of each share for verification.
   - 1.3 P_j also publishes the commitment $Com(s^j, r_j)$ of the secret s^j.
2. **Reveal**:
   - 2.1 All parties verify two things :
     - (a) The shares are valid (ZKP on the commitment and encryption of shares).
     - (b) The shares are consistent with the secret.
   - 2.2 Parties who fail the verification are not allowed to participate.
   - 2.3 Each party publishes the commitments and the valid shares that they received. Once t parties have posted their commitments and valid shares on the ledger, P_j opens its commitment $Open(s^j, r_j)$.
3. **Reconstruction**: For every party P_a that does not publish $Open(s^a, r_a)$ in the Reveal phase, party P_j reconstruct the secret s^a from the valid shares published on the public ledger.

3 Shamir's secret sharing

The aim of the scheme is to divide a secret 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.

This scheme is called (k,n) threshold scheme.

Shamir's secret sharing scheme is based on polynomial interpolation. For (k,n) threshold scheme, we pick a random $k-1$ degree polynomial $p(x) = a_0 + a_1x + \ldots + a_{k-1}x^{k-1}$ where $a_0 = s$.The share s_i is obtained by computing $s_i = p(i)$.

3.1 Reed Solomon code

The codeword of a message $m = m_1 m_2 \dots m_n \in F^k$ is obtained 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

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

We can see that $\gamma = (s_1, s_2, \dots s_n) \in RS_k(\Gamma)$ where $s_1, s_2, \dots, s_n$ are shares of the secret s obtained from the Shamir's secret sharing scheme by setting $x_i = i$.

4 Encryption scheme

We will use BGV scheme introduced in[3] which security relies on the hardness of General Learning With Errors (GLWE) problem. We only consider the Ring Learning With Errors based scheme for our construction. The scheme works as follows:

- **Setup**$(1^\lambda, 1^\mu)$: Choose μ-bit modulus q and parameters $N = N(\lambda, \mu), n = n(\lambda, \mu), k = k(\lambda, \mu)$ 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 $params = (q, N, n, k, \chi)$

- **SecretKeyGen**$(params)$: Sample $\mathbf{u} \leftarrow \chi^n$. Output

  $$sk = \mathbf{s} \leftarrow (1, \mathbf{u}[1], \dots, \mathbf{u}[n])^T \in R_q^{n+1}$$

- **PublicKeyGen**$(params)$: Generate a matrix $\mathbf{B} \leftarrow R_q^{k \times n}$ and a vector $\mathbf{e} \leftarrow \chi^k$. Set $\mathbf{b} := \mathbf{B}\mathbf{u} + 2\mathbf{e}$. Set $pk = \mathbf{A} = (-\mathbf{B}, \mathbf{b})$.

- **Enc**$(params, pk, m_0)$: To encode message $m_0 \in R_2$ set $\mathbf{m} = (0, \dots, 0, m_0)^T \in R_q^{n+1}$, sample $r \leftarrow R_2^k$ and compute

  $$\mathbf{c} := \mathbf{m} + \mathbf{A}^T \mathbf{r} \in R_q^{n+1}$$

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

  $$\left[[\langle \mathbf{c}, \mathbf{s} \rangle]_q\right]_2 = \left[\left[\left\langle \mathbf{m} + \mathbf{A}^T \mathbf{r}, \mathbf{s} \right\rangle\right]_q\right]_2 = \left[\left[\left(\mathbf{m}^T + \mathbf{r}^T \mathbf{A}\right) \cdot \mathbf{s}\right]_q\right]_2 = \left[\left[m_0 + 2\mathbf{r}^T \mathbf{e}\right]_q\right]_2 = m_0$$

5 Commitment scheme

The commitment scheme is based on[1] where the hiding property is based on the hardness on Module-SIS problem and the binding property relies on the hardness of Module-LWE problem.

5.1 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.

Definition 5.1 *The challenge space $\mathscr{C}$ for the zero knowledge proof protocol is defined as*

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

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

5.2 The Lattice Problems

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 5.2 *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 $\mathscr{A}$ has advantage ε in solving the $DKS^\infty_{n,k,\beta}$ problem if*

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

Definition 5.3 *The $SKS^2_{n,k,\beta}$ problem asks to find a short vector $\mathbf{y}$ satisfying $\left[\mathbf{I_n}\ \mathbf{A}'\right] \cdot \mathbf{y} = 0^n$ when given a random $\mathbf{A}'$. We say that an algorithm $\mathscr{A}$ has advantage ε in solving $SKS^2_{n,k,\beta}$ problem if*

$$Pr\left[\|y_i\| \leq \beta \wedge \left[\mathbf{I_n}\ \mathbf{A}'\right] \cdot \mathbf{y} = 0^n | \mathbf{A}' \xleftarrow{\$} R_q^{n\times(k-n)}; 0 \neq \mathbf{y} = [y_1 \ldots y_k]^T \leftarrow \mathscr{A}(\mathbf{A}') \right] \geq \varepsilon$$

5.3 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

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

- **Commit**: To commit to $\mathbf{x} \in R_q^l$, we choose a random polynomial vector $\mathbf{r} \xleftarrow{\$} S_\beta^k$ 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 $f \in \overline{\mathscr{C}}$. 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 $\sigma = 11 \cdot \kappa \cdot \beta \cdot \sqrt{kN}$.

Table 1. Parameter settings for commitment scheme

Parameter	Optimal	Stat. Hiding	Stat. Binding
q	$\approx 2^{32}$	$\approx 2^{35}$	$\approx 2^{71}$
N	1024	512	1024
l	1	1	1
n	1	3	6
k	3	18	9
κ	36	44	36
β	1	128	1
σ	≈ 27000	≈ 594700	≈ 46000

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

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

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.

6 DISCUSSION

6.1 The Protocol

Let M be the number of parties in the protocol and $R = \mathbb{Z}[X]/(X^N + 1)$. Protocol $\Pi_{lattice}$ works as follows :

1. **Commit**:

   1.1 Party P picks random matrices $A_1 \in R_q^{n\times k}$, $A_2 \in R_q^{l\times k}$ and set $A = (A_1, A_2)^T$ for the commitment scheme then run Setup, SecretKeyGen, and PublicKeyGen in the encryption scheme to obtain $params, sk = (1, u)$ and $pk = (-B, b)$. P picks a secret $s \in R_q$ randomly then chooses cofficients $a_1, \ldots, a_t$ randomly for $t = M/2$ and compute $s_i = p(i) = s + a_1 i + \ldots + a_t i^t$ for all $i = 1, \ldots, M$.

   1.2 P encrypts each share s_i with public key pk_i for all $i = 1, \ldots, M$. We have $v_i = \text{Enc}(params, pk, s_i) = \begin{bmatrix} -B_i \\ b_i \end{bmatrix} \cdot r_i + \begin{bmatrix} 0 \\ s_i \end{bmatrix}$ for all $i = 1, \ldots, M$. P also computes the commitment $c = \text{Com}(s, \rho) = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \cdot \rho + \begin{bmatrix} 0 \\ s \end{bmatrix}$ and the commitments of each share. *There are 2 possibilities to compute commitments of each share:*

      i. $c_i = \text{Com}(s_i, \rho_i) = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \cdot \rho_i + \begin{bmatrix} 0 \\ s_i \end{bmatrix}$ for all $i = 1, \ldots M$

      ii. Let $\bar{s} = (s_1, \ldots s_M)^T$. $\bar{c} = \text{Com}(\bar{s}, \rho) = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \cdot \rho + \begin{bmatrix} 0 \\ \bar{s}. \end{bmatrix}$

   P then 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 encryption encode the same message.

   1.3 P publishes v_i, c, c_i, and $PROOF = (v_1, \ldots, v_M, \gamma, c_1, \ldots, c_M)$.

2. **Reveal**:

   2.1 All parties verify $PROOF_1$, choose random codeword $g^\perp = (g_1, \ldots, g_M) \in R_q^M$ from the dual code corresponding to g to the instance of Shamir's (n,t)-threshold secret sharing used by P and run $PROOF_2(A_1, A_2, c_1, \ldots, c_M, g^\perp)$ with P as Prover.

   2.2 Parties who fail the verification are not allowed to participate.

   2.3 Each P publishes all v_i and c_i they received in the **Commit** phase. Once t parties have posted v_i and c_i on the ledger, P opens its commitment $Open(s, \rho)$

3. **Reconstruction**: For every party Q that does not publish $Open(s^Q, \rho^Q)$ in the **Reveal** phase, each party P_i publishes s_i^Q and c_i^Q. Everyone can then verify and reconstruct secret s^Q.

6.2 $PROOF_1$ (encryptions and commitments encode the same message)

$$\underline{PROOF_1(A_1, A_2, B_1, B_2, r, \rho)}$$

6.3 $PROOF_2$ (shares are consistent with secret)

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

$$\underline{PROOF_2(A_1, c, g)}$$

Prover's Information: $\rho \in S_\beta^k$, shares $s_0 = (s_1, \ldots, s_M)$
Commitment of shares: $c_0 = (c_1, \ldots, c_M)$

$$\begin{array}{lcl}
\underline{\text{Prover}} & & \underline{\text{Verifier}} \\
y \leftarrow_r \mathcal{N}_\sigma^k & & \\
t := \mathbf{A_1} \cdot y & & \\
 & \xrightarrow{t} & \\
 & & d \leftarrow_r \mathscr{C} \\
 & \xleftarrow{d} & \\
z = y + d \cdot r & & \\
\text{Abort with probability} & & \\
1 - min\left(1, \frac{\mathcal{N}_\sigma^k(z)}{M \cdot \mathcal{N}^k_{dr,\sigma}(z)}\right) & & \\
 & \xrightarrow{z} & \text{Write } z = [z_1, \ldots, z_M]^T \\
 & & \text{Accept iff } \forall i, \|z_i\|_2 \leq 2\sigma\sqrt{N} \text{ and} \\
 & & \mathbf{A}_1 \cdot z = t + d \cdot \mathbf{c}_1
\end{array}$$

References

1. Carsten Baum, Ivan Dåmgard, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. *More efficient commitments from structured lattice assumptions*. Lecture Notes in Computer Science, 11035:368-85, 2018.
2. Ignacio Cascudo and Bernardo David. *Scrape: Scalable randomness attested by public entities*. Applied Cryptography and Network Security, 2017.
3. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, ITCS, pages 309–325. ACM, 2012.