Greco

Fast Zero-Knowledge Proofs for Valid FHE RLWE Ciphertexts Formation

Enrico Bottazzi

PSE - Ethereum Foundation

27-6-24 FHE.org

Outline

Secret Voting Application

- The tally is computed by summing up the ciphertexts encoding the votes (either 1 or 0)
- Valid encrypted votes are of the form $E(0)$ and $E(1)$.
- A malicious voter could send an invalid encrypted vote such as $E(145127835)$, which can mess up the whole election.

Problem

- *Any* FHE-based application will be required to check the correctness of the ciphertexts
- Only exceptions are applications in which the party performing the encryption is the only one affected by the result of the homomorphic computation

Correctness

Users must be able to prove:

- the ciphertext they submitted is a valid Ring-Learning with Errors (RLWE) ciphertext
- the plaintext message they encrypted meets certain properties

Correctness

Users must be able to prove:

- the ciphertext they submitted is a valid Ring-Learning with Errors (RLWE) ciphertext Greco ZKP
- the plaintext message they encrypted meets certain properties App specific ZKP

LWE

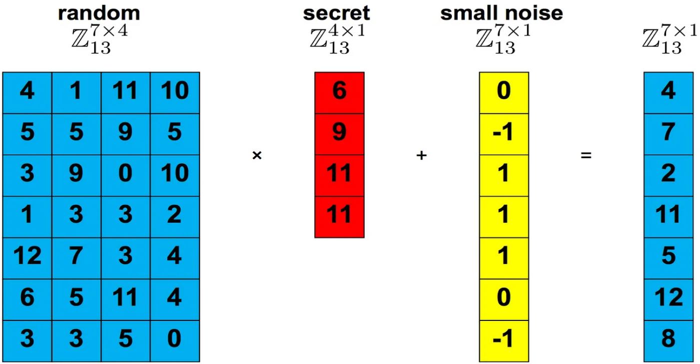

Figure 1: Source: Prof Bill Buchanan OBE FRSE - Learning With Errors and Ring Learning With Errors

$$\vec{A} \cdot \vec{s} + \vec{E} = \vec{B}$$

RLWE

$\mathbb{Z}_{13}[x]/\langle x^4 + 1 \rangle$

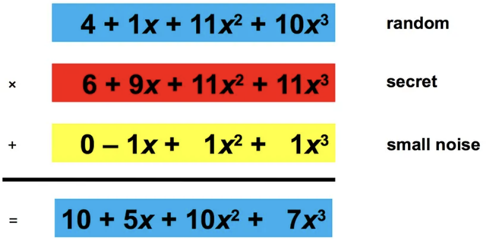

Figure 2: Source: Prof Bill Buchanan OBE FRSE - Learning With Errors and Ring Learning With Errors

$$A \cdot s + E = B$$

BFV

BFV [Bra12][FV12] is a leveled FHE scheme based on the RLWE problem

$$Ct = (Ct_0, Ct_1) = ([A \cdot s + E + K]_Q, -A)$$

- Q be the ciphertext modulus and t be the plaintext modulus where $Q >> t$
- R_Q be the polynomial ring $\frac{Z_Q[X]}{X^N+1}$, with N being a power of two.
- $A \leftarrow R_Q, s \leftarrow \chi_{key}, E \leftarrow \chi_{error}$
- $K = \lceil \frac{Q[M]_t}{t} \rfloor$ [KPZ21]

Chinese Remainder Theorem (CRT)

- Set $Q = \prod q_i$ where the q_i factors are pairwise coprime.
- Using this technique, an integer $x \in \mathbb{Z}_Q$ can be represented by its CRT components $\{x_i = x \mod q_i \in \mathbb{Z}_{q_i}\}_i$, and operations on x in Z_Q can be implemented by applying the same operations to each CRT component x_i in Z_{q_i}.

BFV in CRT Setting [Baj+17]

$$Ct_i = (Ct_{0,i}, Ct_{1,i}) = ([A_i \cdot s + E + K_{0,i}K_1]_{q_i}, -A_i)$$

- i indicates the i-th CRT decomposition of the ciphertext Ct in the basis q_i
- Operations in R_q are implemented directly in CRT representation.
- If we choose Q and t such that they are co-prime one can calculate K directly in CRT [KPZ21]
$$K = -t^{-1}[QM]_t \mod q_i$$
- We will denote the scalar $K_{0,i} = -t^{-1} \mod q_i$ and the polynomial $K_1 = [QM]_t$

zk-SNARKs

Informally, a *proof* for a relation $\mathcal{R}$ is a protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ by which $\mathcal{P}$ convinces $\mathcal{V}$ that $\exists w : \mathcal{R}(x, w) = 1$, where x is a called an *instance*, and w a *witness* for x.

- Setup$(1^\lambda, \mathcal{R}) \to$ pp: setup public parameters for $\mathcal{R}$.
- Prove(pp, x, w) $\to \pi/\perp$: if $(x, w) \in \mathcal{R}$, output a proof π, otherwise $\perp$.
- Verify(pp, x, π) $\to \{0, 1\}$: check a proof.

Introduction

Greco allows users to prove the validity of a FHE Ring-Learning with Errors (RLWE) ciphertext. Here we focus on BFV Secret Key Encryption in the CRT setting.

Task: design a zkSNARK to prove the following relation:

$$Ct_0 = [A \cdot s + E + K]_Q$$

Challenge - Non Native Arithmetic

- Witness values inside the circuit are elements of prime field $\bmod p$.
  - In KZG-based SNARKs p is 254 bits
- The coefficients of Ct_0 are defined in Z_Q. All the polynomial operations are performed modulo the ring R_Q
  - Q can range from 27 to 881 bits [Alb+22]

Solution - CRT Decomposition

- Instead of working with Ct_0, work with their CRT decomposed $Ct_{0,i}$
  - If Q is 881 bits we can decompose using CRT into $k = 15$ components $Ct_{0,i}$ where q_i is at max 59 bits
- Coefficients of $Ct_{0,i}$ can be represented in Z_p
- Operations on Ct_0 are safely implemented on its k CRT components $Ct_{0,i}$

Solution - Precompute Auxiliary Polynomials

Operation to prove $A_i \cdot s + E + K_{0,i}K_1 = Ct_{0,i} \mod R_{q_i}$

$$\hat{Ct}_{0,i} = A_i \cdot s + E + K_{0,i}K_1$$
$$\hat{Ct}_{0,i} = Ct_{0,i} \mod R_{q_i}$$
$$\hat{Ct}_{0,i} = Ct_{0,i} - R_{2,i}(X^N + 1) \mod Z_{q_i}$$
$$\hat{Ct}_{0,i} = Ct_{0,i} - R_{2,i}(X^N + 1) - R_{1,i}q_i$$

Since $q_i << p$, the equation stays unchanged in Z_p:

$$\hat{Ct}_{0,i} = Ct_{0,i} - R_{2,i}(X^N + 1) - R_{1,i}q_i \mod Z_p$$
$$Ct_{0,i} = A_i \cdot s + E + K_{0,i}K_1 + R_{2,i}(X^N + 1) + R_{1,i}q_i \mod Z_p$$

Solution - Precompute Auxiliary Polynomials

To prove that $Ct_{0,i}$ is correctly formed, it is needed to prove that the equation above holds. This can be rewritten as:

$$Ct_{0,i} = \begin{bmatrix} A_i & 1 & K_{0,i} & (X^N + 1) & q_i \end{bmatrix} \times \begin{bmatrix} s \\ E \\ K_1 \\ R_{2,i} \\ R_{1,i} \end{bmatrix}$$

or

$$Ct_{0,i} = U_i \times S_i$$

Solution - Precompute Auxiliary Polynomials

To prove that $Ct_{0,i}$ is correctly formed, it is needed to prove that the equation above holds. This can be rewritten as:

$$Ct_{0,i} = \begin{bmatrix} A_i & 1 & K_{0,i} & (X^N + 1) & q_i \end{bmatrix} \times \begin{bmatrix} s \\ E \\ K_1 \\ R_{2,i} \\ R_{1,i} \end{bmatrix}$$

or

$$Ct_{0,i} = U_i \times S_i$$

Challenge - Large Degree Polynomial Multiplication

- Many large degree polynomial multiplications involved in the previous operation
- Considering two polynomials f and g of degree n, performing the polynomial multiplications $fg = h$ using the direct method would generate:
  - $(n+1)^2$ multiplication
  - n^2 addition

Solution - Challenge-based Large Degree Polynomial Multiplication

- Evaluate the polynomials f, g, and h at a random point γ
- Enforce $f(\gamma) * g(\gamma) = h(\gamma)$ which would be true if $fg = h$ according to Schwartz-Zippel lemma.
  - n multiplication and n addition to evaluate $f(\gamma)$
  - n multiplication and n addition to evaluate $g(\gamma)$
  - $2n$ multiplication and $2n$ addition to evaluate $f(\gamma)$

Achievement

- Complexity of performing polynomial multiplication is reduced from $O(n^2)$ to $O(n)$.
- The constraint is then reduced to proving that

$$\begin{bmatrix} A_i(\gamma) & 1 & K_{0,i} & (\gamma^N + 1) & q_i \end{bmatrix} \times \begin{bmatrix} s(\gamma) \\ E(\gamma) \\ K_1(\gamma) \\ R_{2,i}(\gamma) \\ R_{1,i}(\gamma) \end{bmatrix} = Ct_{0,i}(\gamma)$$

or

$$U_i(\gamma) \times S_i(\gamma) = Ct_{0,i}(\gamma) \tag{1}$$

Proving Strategy

During **phase one** of Proof Generation:

1. Fill the witness table with the secret polynomials of S_i
2. Extract the commitment of the witness so far and hash it to generate the challenge γ (Fiat-Shamir Heuristic)

During **phase two** of Proof Generation:

1. Prove that the coefficients of the polynomials of S_i are in the expected range
2. Evaluate the secret polynomials of S_i at γ, the public polynomials of U_i at γ and the ciphertext $Ct_{0,i}(\gamma)$
3. Prove that (1) holds

Benchmarks

n	$\log q_i$	k	Proof Gen Time	Proof Ver Time
1024	27	1	685.51ms	3.66ms
2048	53	1	1.39s	3.74ms
4096	55	2	3.47s	5.02 ms
8192	55	4	8.98s	4.18ms
16384	54	8	29.43s	6.97ms
32768	59	15	102.15s	14.06ms

Table 1: Greco performance benchmarks for different security parameters.

Run M2 Macbook Pro with 12 cores and 32GB of RAM. Implementation in Halo2-lib. Plonk + KZG Commitments zk-SNARKs

Sections Omitted from the Presentation

- Calculating $R_{2,i}$ and $R_{1,i}$
- Strategies to prove the correct formation of k ciphertexts
- Public Key Encryption Extension
- Composability with Application-Specific Logic

Takeaways

- *Almost any* FHE-based application will be required to check the correctness of the ciphertexts using zk-SNARKs
- The main strategies employed to efficiently perform RLWE inside a zk-SNARK are:
  - Leverage CRT for native coefficient representation
  - Move reduction "outside" the circuit leveraging auxiliary polynomial
  - Challenge-based polynomial multiplication

Improvements

- Faster (or more FHE-friendly) zk Protocol. Leverage parallelization across different CRT moduli?
- Support for further encodings and FHE schemes

Thank You

Any Questions?