New Year's Resolutions

Angold J. Wang
WeidS Lab, Hong Kong
Thursday 6th February, 2025
awang@weids.dev

***Abstract*—This proposal presents a refined research direction focused on benchmarking zk-Rollups by evaluating different zk-SNARK implementations using the `gnark` library. Specifically, we compare Groth16 and Plonk protocols across several elliptic curves (BN254, BLS12-381, etc.) on a lightweight Hyperledger Fabric testbed. Our study delivers a detailed performance analysis—including latency, throughput, proof generation and verification costs, and resource utilization—while also situating these findings within a comparative framework alongside other layer 2 solutions like Plasma and Optimistic Rollups. Additionally, the proposal includes comprehensive background material that demystifies zk-SNARKs for newcomers and provides a side-by-side comparison of key Layer-2 solutions, offering guidance for optimal protocol selection in blockchain scaling.**

***Index Terms*—Blockchain, Benchmark, ZK-SNARKs, ZK-Rollups, Layer-2.**

I. Background and Motivation

A. Layer-2 Scaling Solutions

Several L2 protocols have been proposed to improve the scalability of blockchain networks:

- **Plasma:** Leverages child chains to offload transactions but suffers from delayed finality due to challenge and exit periods.
- **Optimistic Rollups:** Assume transaction validity optimistically, relying on fraud proofs & challenge windows.
- **zk-Rollups:** Utilize zero-knowledge proofs to provide succinct and secure state transitions on the mainchain. They offer stronger security guarantees but at a computational cost due to proof generation.

B. Zero-Knowledge Proofs and zk-SNARKs

Zero-knowledge proofs, particularly zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge), have become central to ensuring the security and efficiency of zk-Rollups. Recent developments include:

- **Groth16:** A widely adopted zk-SNARK protocol known for its succinct proofs.
- **Plonk:** A newer protocol that offers universal and updatable structured reference strings.

Different elliptic curves (e.g., BN254, BLS12-381, etc.) impact the performance and security properties of these protocols, motivating a systematic benchmarking study.

C. Motivation for the Proposal

Preliminary studies indicate that while zk-Rollups offer significant security benefits, the computational overhead—especially in proof generation—can be a bottleneck. By benchmarking different zk-SNARK implementations under uniform conditions, we aim to:

1) Provide a comprehensive performance comparison (latency, throughput, and resource utilization).
2) Offer practical guidance for selecting the most suitable zk-SNARK approach and elliptic curve for specific blockchain applications.
3) Place these insights within the broader landscape of L2 scaling solutions by also examining Plasma and Optimistic Rollups.

II. Objectives and Expected Contributions

A. Research Objectives

- **Benchmarking zk-Rollups:** Evaluate the performance of zk-Rollups by implementing different zk-SNARK protocols (Groth16 and Plonk) using various elliptic curves.
- **Performance Metrics:** Measure key metrics including latency, throughput, proof generation time, verification time, and resource utilization.
- **Comparative Analysis:** Contextualize the performance of zk-Rollups by comparing them with Plasma and Optimistic Rollups on parameters such as gas consumption and overheads on the Layer-1 blockchain.
- **Educational Component:** Provide a comprehensive introduction to zk-SNARKs for newcomers, detailing their underlying mechanisms and applications.

B. Expected Contributions

- A detailed performance benchmark of zk-Rollups implementations under various configurations.
- Insights into the optimal selection of zk-SNARK protocols and elliptic curves for blockchain scaling.
- A comprehensive framework for comparing Layer-2 scaling solutions, aiding practitioners in making informed protocol choices.
- Educational material that demystifies zk-SNARKs for new researchers and developers.

III. Methodology

A. Experimental Testbed

We will deploy a lightweight, customizable Layer-1 blockchain network using Hyperledger Fabric, which allows controlled experimentation and reproducibility. This testbed will serve as the foundation for all benchmarking tests.

B. Benchmarking Metrics

Key performance indicators will include:

1) **Latency and Throughput:** Measure transaction confirmation times and transaction processing rates.
2) **Proof Generation and Verification Costs:** Quantify the computational overhead for generating and verifying zk-SNARK proofs.
3) **Resource Utilization:** Monitor CPU, memory, and network usage during benchmarking tests.
4) **Gas Consumption and Overheads:** Evaluate the cost implications on the Layer-1 blockchain when committing zk-Rollup state transitions.

C. Comparative Analysis

After establishing the performance benchmarks for zk-Rollups, we will conduct a comparative analysis with other Layer-2 solutions (Plasma and Optimistic Rollups). This comparison will help identify:

- The trade-offs between efficiency and security.
- The ideal contexts or use cases for each protocol.

IV. Demystifying zk-SNARKs

zk-SNARKs enable a prover to convince a verifier that a computation was performed correctly without revealing any details of the computation or the witness. This is achieved by combining three main ideas: encoding computations as Quadratic Arithmetic Programs, leveraging elliptic curve trapdoor functions under the knowledge-of-exponent assumption, and performing succinct verification via bilinear pairings.

A. From Circuits to Quadratic Arithmetic Programs (QAPs)

A circuit (or more generally, any NP statement) can be expressed in terms of a set of constraints. In zk-SNARKs the idea is to encode these constraints as a Quadratic Arithmetic Program (QAP): Let $\mathbf{s} = (s_1, s_2, \ldots, s_n)$ be the witness vector, and let $\{A_i(x)\}$, $\{B_i(x)\}$, and $\{C_i(x)\}$ be families of polynomials encoding the circuit's constraints. The QAP can be expressed as:

$$P(x) = \left(\sum_{j=1}^{n} s_j A_j(x)\right)\left(\sum_{j=1}^{n} s_j B_j(x)\right) - \left(\sum_{j=1}^{n} s_j C_j(x)\right).$$

For a valid witness, the polynomial $P(x)$ vanishes on a predetermined set of points. Equivalently, one can write:

$$P(x) = H(x) \cdot Z(x),$$

where $Z(x)$ is the *vanishing polynomial* (with roots at the chosen evaluation points) and $H(x)$ is the quotient polynomial. This formulation mathematically certifies that all circuit constraints are satisfied, by leveraging the Schwartz-Zippel lemma, if an unpredictable t can satisfy this formula, we can say all constraints are satisfied in high probability.

However, directly revealing the witness $\mathbf{s}$ and verifying $P(t)$ are inefficient (linear in circuit size) and non-private. To overcome these issues, zk-SNARKs employ elliptic curves in following two key ways:

B. EC Trapdoor Commitments and the KoE Assumption

A *Structured Reference String (SRS)* is generated in a trusted setup phase. A secret value t and auxiliary *toxic waste* parameters (e.g., k_a, k_b, k_c) are chosen and then deleted. The SRS includes, for each index i, group elements such as:

$$\{\, G \cdot A_i(t), \quad G \cdot A_i(t) \cdot k_a \,\},$$

and analogously for $B_i(t)$ and $C_i(t)$, as well as powers of t (e.g., $G \cdot t$, $G \cdot t^2$, $\ldots$) for $H(t)$. The hardness of the discrete logarithm problem in elliptic curve groups guarantees that:

- It is computationally infeasible to recover secret t or toxic parameters k_A, k_B and k_c from the SRS Group.
- Under the *knowledge-of-exponent assumption*, if the prover ourputs a pair (P, Q) satisfying $Q = k_a \cdot P$, then P must be a valid linear combination of each $G \cdot A_i(t)$ in the SRS elements, which means the prover's commitments (e.g., $G \cdot A(t)$) are guaranteed to be valid linear combinations derived from the circuit.

C. Succinct Verification via Bilinear Pairings

Bilinear pairings

$$e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$$

are maps defined on elliptic curve groups that satisfy:

$$e(x \cdot G,\, y \cdot G) = e(G, G)^{xy}.$$

This property enables the verifier to check multiplicative relationships between committed values succinctly. For instance, to confirm that the commitment to $A(t)$ is properly scaled by k_a, the verifier can check:

$$e\big(G \cdot A(t) \cdot k_a,\, G\big) \stackrel{?}{=} e\big(G \cdot A(t),\, G \cdot k_a\big).$$

Similarly, the final verification of the QAP relation

$$\left(\sum_j s_j A_j(t)\right) \cdot \left(\sum_j s_j B_j(t)\right) - \left(\sum_j s_j C_j(t)\right) = H(t) \cdot Z(t)$$

is performed with a pairing equation such as:

$$\frac{e\Big(G \cdot \sum_j s_j A_j(t),\, G \cdot \sum_j s_j B_j(t)\Big)}{e\Big(G \cdot \sum_j s_j C_j(t),\, G\Big)} \stackrel{?}{=} e\Big(G \cdot H(t),\, G \cdot Z(t)\Big).$$

This single equation, independent of the circuit size, ensures that the committed polynomials satisfy the QAP relation.

V. Conclusion

The proposed research aims to fill a critical gap in the evaluation of blockchain Layer-2 scaling solutions by focusing on a detailed performance benchmarking of zk-Rollups. By leveraging the `gnark` library and a controlled Hyperledger Fabric environment, we expect to provide actionable insights that guide both theoretical research and practical implementations. We believe this work will be a valuable resource for the blockchain community, offering clarity on the trade-offs inherent in zk-SNARK-based scaling approaches and helping to drive future innovations in secure and scalable blockchain architectures.