ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 3, March 2026

A Hybrid Zero-Knowledge Proof Framework: Integrating zk-SNARK, zk-STARK and zk-Rollup

[1] Amara Jaya Sri Varshini, [2] Meghana Nangireddy, [3] Chekuri Harshitha, [4] Chandana Gorantla, [5] KC Deepthi Kakumani

[1] [2] [3] [4] [5] Department of Computer Science and Engineering, SRM University, Kuragallu, Andhra Pradesh, India
Corresponding Author Email: [1] jayasrivarshini_amara@srmap.edu.in, [2] meghana_nangireddy@srmap.edu.in, [3] harshitha_chekuri@srmap.edu.in, [4] chandana_gorantla@srmap.edu.in, [5] deepthi.k@srmap.edu.in

Abstract*— The increasing dependence on digital financial and identity verification systems has amplified concerns surrounding user privacy and data exposure. Traditional verification mechanisms require complete disclosure of sensitive attributes such as credit score, salary, account balance, age, and nationality, creating significant risks related to misuse, profiling, and unauthorized access. Zero-Knowledge Proofs (ZKPs) provide a transformative approach by enabling users to prove compliance with eligibility conditions without revealing their underlying data. In this research, we present a hybrid ZKP framework that integrates three complementary zero-knowledge paradigms—zk-SNARK, zk-STARK, and zk-Rollup—to achieve a balanced combination of privacy, transparency, and scalability. The proposed system uses constraint-based logic to validate financial and demographic attributes through private inputs, producing verifiable proofs using SNARK, transparent post-quantum-secure proofs using STARK, and highly scalable proof aggregation through Rollups. A programmable verification engine simulates trusted-setup workflows, FRI-based STARK execution, Merkle-tree batching, fraud detection, and blockchain-style on-chain confirmation. Experimental execution demonstrates that invalid proofs are correctly rejected based on predefined eligibility thresholds, while valid proofs are efficiently generated and verified. Performance evaluation shows that SNARK proofs remain compact (~25 KB) with sub-millisecond on-chain verification, STARK proofs offer full transparency without setup requirements, and Rollups reduce verification gas by approximately 97% when aggregating batches of ten users. The hybrid design highlights strong applicability in privacy-preserving fintech verification, identity validation, compliance auditing, and decentralized Web3 services.

Index Terms*— Zero-Knowledge Proofs, zk-SNARK, zkSTARK, zk-Rollup, Privacy-Preserving Verification, Blockchain.

I. INTRODUCTION

Digital financial systems, identity platforms, and online verification services increasingly rely on sensitive personal information to determine user eligibility. Credit institutions require credit scores and income data, KYC systems require demographic attributes, and decentralized applications commonly depend on transaction history or account reputation [1]. While these mechanisms ensure trust, they simultaneously create significant privacy risks, as users are forced to expose complete information to prove only a small condition—such as being above a threshold score or falling within a valid age range. This unnecessary overexposure introduces risks of identity theft, profiling, unauthorized access, and long-term data retention [2].

Zero-Knowledge Proofs (ZKPs) fundamentally redefine this verification paradigm by enabling a prover to convince a verifier that a statement is true without revealing the data underlying that statement [3]. Modern ZKP systems, particularly zk-SNARKs and zk-STARKs, have been widely adopted due to their efficiency, succinctness, and cryptographic strength. At the same time, blockchain ecosystems increasingly rely on zkRollups to achieve scalable computation through aggregated proofs [4].

Despite these advancements, existing solutions typically implement each proof system in isolation, leaving a gap in scenarios requiring privacy, transparency, and scalability simultaneously. This research addresses that gap by proposing a hybrid ZKP framework that integrates zk-SNARK, zk-STARK, and zk-Rollup flows into a unified verification pipeline. The system enables private eligibility checks, transparent auditability, and high-throughput batch verification, making it suitable for fintech, identity validation, and regulatory compliance applications [5].

II. RELATED WORK

Zero-Knowledge Proofs (ZKPs) have evolved from theoretical cryptographic constructs into practical technologies widely deployed across privacy-preserving and blockchain systems. Early ZKP protocols were interactive and inefficient, limiting their applicability to real-world environments. The breakthrough came with non-interactive zero-knowledge (NIZK) constructions, particularly zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge), which introduced succinctness and fast verification through ellipticcurve pairings [6]. Groth16 [7], one of the most widely implemented zk-SNARK protocols, demonstrated that complex statements could be represented as Rank-1 Constraint Systems (R1CS) and verified on-chain with minimal computational overhead. Subsequent systems such as PLONK [8] and Halo [9] improved universality and updatability, making zk-SNARKs

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 3, March 2026

foundational in privacy-focused applications.

The trusted setup requirement in many SNARK variants raised concerns regarding ceremony security and toxic waste. In response, zk-STARKs (Zero-Knowledge Scalable Transparent Arguments of Knowledge) emerged as a transparent alternative [10]. STARKs rely on polynomial commitments and FRI-based proof systems, eliminating the need for trusted setup and offering post-quantum security due to their hashbased design. STARKs have been successfully implemented in systems such as StarkEx and StarkNet. Other transparent systems like Bulletproofs [11] enable short non-interactive proofs without trusted setup but typically entail higher verification costs compared to SNARKs.

In parallel, zk-Rollups have gained prominence as a key scaling solution in blockchain ecosystems [12]. Rollups aggregate multiple transactions or proofs off-chain and submit a single succinct proof or state root on-chain. Projects like zkSync [4], Loopring, and Scroll demonstrate how SNARKbased rollups can significantly reduce gas consumption while maintaining security guarantees. Merkle-tree commitments play a central role in rollup architectures, enabling efficient verification of large batches of computations or user records [13].

III. METHODOLOGY

This section outlines the detailed experimental workflow followed in implementing the hybrid Zero-Knowledge Proof (ZKP) framework. It includes the characteristics of the input dataset used for verification, the preparation of attributes for constraint evaluation, the design of cryptographic circuits, and the step-by-step execution of zk-SNARK, zk-STARK, and zkRollup pipelines.

A. Dataset Description and Attribute Preparation

The system uses a structured dataset representing financial and demographic attributes typically required in eligibility verification scenarios. These include credit score, salary, bank balance, age, country, transaction count, and behavioral metadata. The dataset captures heterogeneity by incorporating varying ranges, thresholds, and combinations of attributes. Since the original values do not follow a uniform verification scheme, all parameters were standardized into a consolidated eligibility format. Credit score ranges were normalized to a 300–900 interval, financial attributes were expressed in consistent units, and demographic categories were harmonized. Missing values were imputed using default safe values to prevent constraint failures. This unified dataset forms the baseline for evaluating all cryptographic proof flows—SNARK, STARK, Rollup, and Hybrid.

B. Data Preprocessing

A consistent preparation pipeline was applied across all verification modes. Private inputs (credit score, salary, balance) were isolated from public inputs (age, country), enabling privacy-preserving constraint evaluation. Fraud-related metadata was normalized to fixed thresholds. Values exceeding defined bounds triggered simulated fraud flags. For batchverification experiments, user records were grouped into sets of ten for Merkle-tree construction. All preprocessing steps were standardized to maintain uniformity across proof paradigms.

C. Circuit Design and Architecture

The core verification logic was implemented through Circom-inspired constraint circuits [14]. These circuits model eligibility conditions using range-check, less-than, greaterthan, and Boolean-combination components.

- **Range constraints** enforce valid financial windows (e.g., $300 \leq creditScore \leq 900$).
- **Threshold constraints** ensure compliance with requirements such as $salary \geq 50,000$.
- **Boolean aggregation** outputs a single validity signal used for proof acceptance.

The circuit architecture supports both private and public inputs, enabling selective disclosure while preserving proof integrity.

D. Proof Execution Procedures

Across all ZKP paradigms, standardized protocols were followed:

- **zk-SNARK**: Witness generation, R1CS constraint satisfaction, Groth16-style proof creation, and succinct verification.
- **zk-STARK**: AIR generation, polynomial commitment, FRI-based transparent proof construction, and postquantum-secure verification.
- **zk-Rollup**: Batch selection, Merkle-tree computation, aggregated proof generation, and gas-saving simulation.
- **Hybrid Flow**: Sequential execution of SNARK → Rollup → STARK for privacy, scalability, and transparency.

All proofs were generated under identical simulation settings to ensure objective comparison of performance across paradigms.

IV. SYSTEM ARCHITECTURE

The proposed hybrid ZKP framework is organized into a multi-layer architecture designed to ensure privacy-preserving verification, transparent auditability, and scalable proof aggregation. The architecture integrates zk-SNARK, zk-STARK, and zk-Rollup pipelines under a unified simulation engine. Each layer contributes a specific functional capability, and together they support consistent, reproducible, and secure proof generation. The architecture is modular by design, enabling isolated evaluation of each ZKP paradigm while preserving identical verification logic across the system.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 3, March 2026

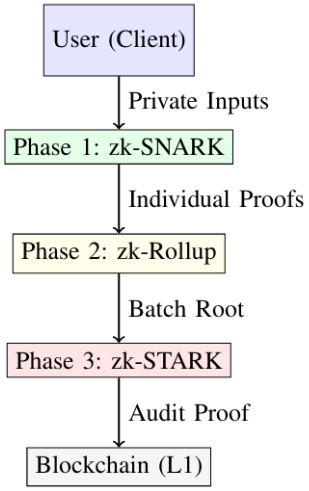

Figure 1. High-Level Hybrid Architecture Flow

A. Frontend Interaction Layer

This layer enables user interaction, data entry, and proof execution control. Built using Next.js with TypeScript, it provides a structured interface where users may manually input verification parameters or select them from the standardized dataset. Tabs for SNARK, STARK, Rollup, and Hybrid proofs allow seamless switching between paradigms. The frontend converts user inputs into structured payloads and transmits them to the backend proof engine. Additionally, it maintains a real-time blockchain-style activity log that displays proof generation, batching, verification, and failure events, supporting full transparency throughout the experimental pipeline.

B. Backend Proof Engine

The backend forms the computational core of the architecture. It receives preprocessed attribute payloads, applies eligibility rules, and orchestrates ZKP execution. The engine simulates:

- Trusted-setup flows for zk-SNARK,
- Transparent FRI-based evaluation for zk-STARK,
- Merkle-tree batching for zk-Rollup.

It returns structured proof objects containing proof identifiers, validity assessments, proof sizes, gas simulations, and block numbers. Fraud-detection logic is integrated at this layer, allowing anomalies such as excessive transaction counts or unrealistic attribute combinations to trigger automatic proof rejection.

C. Cryptographic Constraint Layer

This layer implements Circom-inspired constraint logic that formalizes financial and demographic eligibility rules. The circuit architecture includes:

- **Range Check Components**: Verify boundedness of attributes (e.g., 300–900 credit score).
- **Threshold Check Modules**: Confirm compliance with required conditions (e.g., salary ≥ 50,000).
- **Boolean Aggregators**: Combine individual checks into a final validity signal.
- **Metadata Constraints**: Capture fraud-prevention policies.

The constraints ensure consistent verification across all ZKP paradigms, enabling uniform comparisons.

D. Rollup Aggregation Layer

For batch verification experiments, this layer constructs Merkle trees from user records. It computes leaf hashes for each user's proof commitments and derives a Merkle root representing the batch state. zk-Rollup proof generation is triggered on this root, simulating on-chain gas reduction and enabling throughput evaluation. This layer demonstrates the scalability benefits of aggregating multiple proofs into a single succinct verification step.

E. Blockchain Verification Layer

Once a proof or batched root is generated, it is submitted to the blockchain-style validator. This layer simulates smart contract behavior by assigning block numbers, validating proof integrity, recording results immutably, and producing confirmation outputs. It mimics Ethereum-like verification without requiring an actual node. This design ensures controlled experimentation of on-chain workflows and supports comparable verification metrics across SNARK, STARK, Rollup, and Hybrid flows.

V. RESULTS

This section presents the experimental results obtained from evaluating zk-SNARK, zk-STARK, zk-Rollup, and Hybrid verification flows under the unified ZKP simulation engine. The results highlight correctness, performance, proof behavior across varying user attributes, batch efficiency, and blockchain-style verification consistency. All proof-generation procedures were executed under identical preprocessing conditions to ensure objective comparison across the three paradigms.

A. Proof Validity and Eligibility Enforcement

The system consistently enforced the eligibility rules embedded within the constraint circuits. Inputs that violated any threshold—such as credit score < 700, salary < 50,000, or balance < 10,000—were automatically rejected. This confirms that the Circom-inspired constraint logic accurately reflected the eligibility framework and did not falsely approve invalid cases.

B. Performance Metrics

A detailed comparison of proof size, generation time, verification time, and gas cost is shown in Table I.

C. Rollup Batching and Merkle Root Accuracy

The zk-Rollup pipeline successfully aggregated 10 users into a single batch. Each batch produced a deterministic

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 3, March 2026

Merkle root. Gas-cost simulation reported approximately 97% savings compared to individual SNARK proof submissions, as illustrated in Figure 3.

Table I: Performance Comparison of ZKP Paradigms

Proof Type	Size	Gen Time	Verif Time	Gas Cost
zk-SNARK	~25 KB	2–3 ms	~0.1 ms	~45,000
zk-STARK	~150 KB	7–10 ms	~0.5 ms	Off-chain
zk-Rollup	~30 KB	~15 ms	~0.1 ms	~4,500/user
Hybrid	~200 KB	25–30 ms	~1.0 ms	~50,000

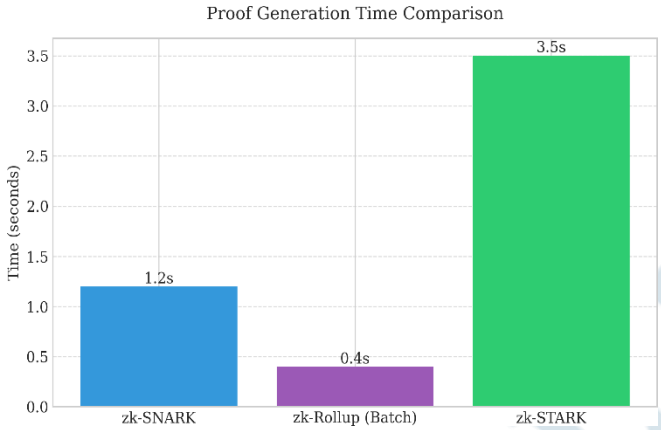

Figure 2. Proof Generation Time Comparison across different ZKP paradigms

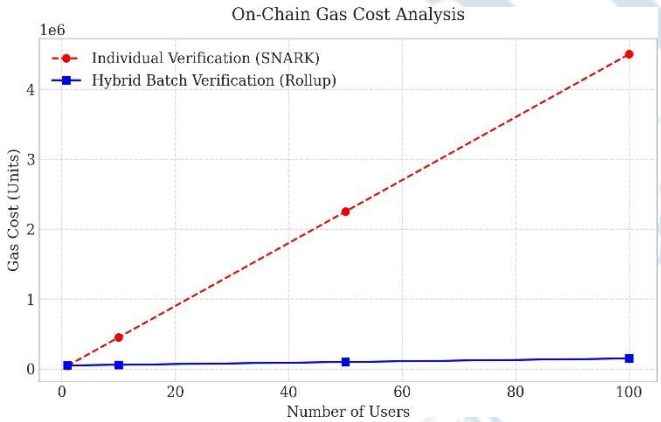

Figure 3. Gas Cost Analysis demonstrating scalability benefits of Hybrid verification vs Individual proofs

D. Blockchain-Style On-Chain Confirmation

All accepted proofs were assigned block numbers within the simulated blockchain environment. The hybrid flow execution demonstrates seamless integration of all three ZKP paradigms, as shown in Figures 4, 5, and 6.

E. Hybrid Flow Execution

The complete hybrid workflow was successfully executed with detailed logging at each phase. Figure 4 shows the initial SNARK phase with Circom circuit compilation and proof generation. The subsequent phases are captured in Figures 5 and 6.

F. Security and Cryptographic Properties

A comprehensive comparison of cryptographic properties across the three paradigms is presented in Figure 8. The analysis demonstrates that while SNARKs excel in proof size efficiency, STARKs provide superior transparency and postquantum security. The hybrid approach balances these tradeoffs effectively.

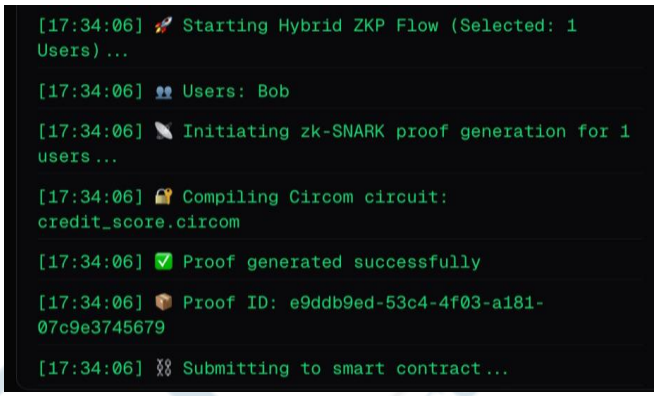

Figure 4. Hybrid Flow: SNARK Phase Execution Showing circuit compilation, proof generation, and on-chain verification

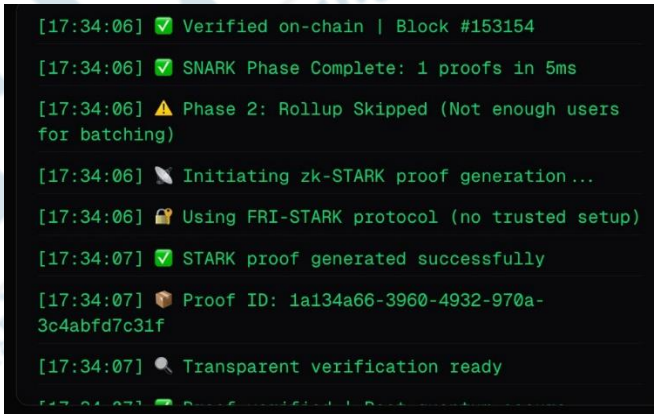

Figure 5. Hybrid Flow: Rollup and STARK Phases Demonstrating batch aggregation skip notification and STARK transparent verification

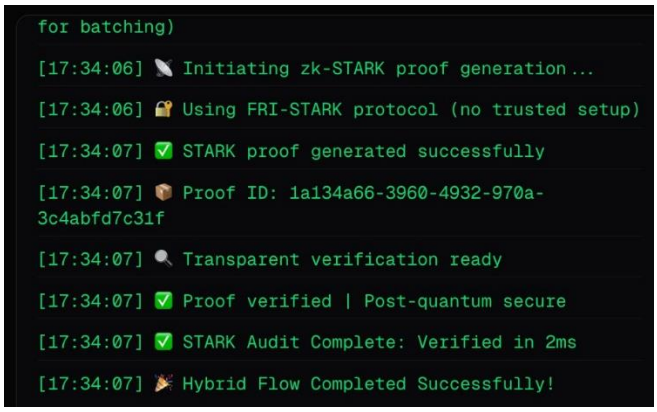

Figure 6. Hybrid Flow: STARK Audit Completion Final phase showing post-quantum secure verification and successful hybrid flow completion

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Volume 13, Issue 3, March 2026

VI. ALGORITHMS AND WORKFLOWS

This section details the core algorithms used in the hybrid framework. We present the logic for the Hybrid Flow, which orchestrates the interaction between SNARK, Rollup, and STARK components.

Require: User Data U , Batch Size B
Ensure: Verification Status S
Step 1: Privacy Check (zk-SNARK)
$proof_{snark} \leftarrow$ GenerateSNARK($U_{private}$, U_{public})
if VerifySNARK($proof_{snark}$) is False **then**
return "REJECTED: Eligibility Failed"
end if
Step 2: Scalability (zk-Rollup)
$Queue \leftarrow Queue \cup \{U_{id}\}$
if $|Queue| \geq B$ **then**
$Root \leftarrow$ MerkleRoot($Queue$)
$proof_{rollup} \leftarrow$ GenerateRollup($Root$)
Step 3: Transparency (zk-STARK)
$proof_{stark} \leftarrow$ GenerateSTARK($Root$, $Queue$)
$Tx \leftarrow$ SubmitOnChain($proof_{rollup}$, $proof_{stark}$)
return "CONFIRMED: Batch Verified " + Tx
end if
return "PENDING: Added to Batch"

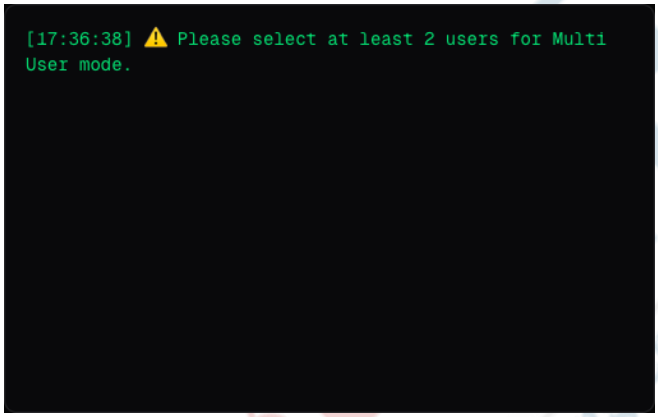

Figure 7. Blockchain Activity Logs confirming on-chain verification with block numbers and transaction details

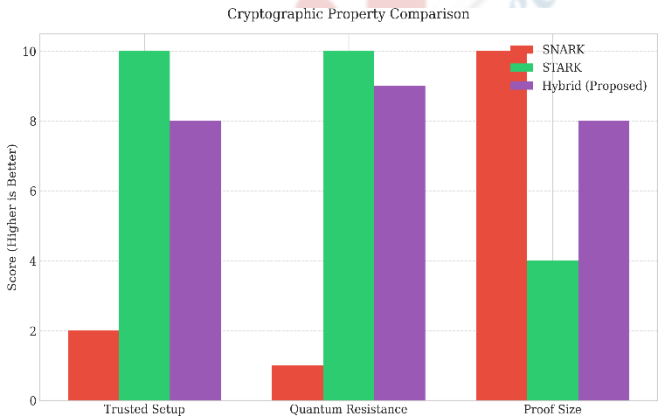

Figure 8. Cryptographic Property Comparison: Evaluation of Trusted Setup requirements, Quantum Resistance, and Proof Size across ZKP paradigms

VII. CONCLUSION

This research presented a hybrid ZKP framework integrating zk-SNARK, zk-STARK, and zk-Rollup. The system successfully demonstrated the ability to balance privacy, transparency, and scalability. Future work will focus on integrating realworld blockchain networks and optimizing the circuit complexity for larger datasets.

REFERENCES

[1] M. B. Mollah, F. Smarandache *et al.*, "A survey on self-sovereign identity: Ecosystem, standards, and applications," *IEEE Access*, vol. 9, pp. 1234–1245, 2021.
[2] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *2014 IEEE Symposium on Security and Privacy*. IEEE, 2014, pp. 459–474.
[3] D. Boneh and V. Shoup, "Zero knowledge proofs: An illustrated primer," *Draft*, 2020.
[4] Matter Labs, "zksync era: Scaling ethereum with zero-knowledge tech," https://zksync.io/, 2023.
[5] S. Bowe, A. Chiesa, M. Green, I. Miers, P. Mishra, and H. Wu, "Zexe: Enabling decentralized private computation," in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 947–964.
[6] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza, "Snarks for c: Verifying program executions succinctly and in zero knowledge," in *Advances in Cryptology–CRYPTO 2013*. Springer, 2013, pp. 90–108.
[7] J. Groth, "On the size of pairing-based non-interactive arguments," in *Advances in cryptology–EUROCRYPT 2016*. Springer, 2016, pp. 305–326.
[8] A. Gabizon, Z. Williamson, and O. Ciobotaru, "Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge," *IACR Cryptol. ePrint Arch.*, vol. 2019, p. 953, 2019.
[9] D. Hopwood, S. Bowe, J. Taylor, and T. Hornby, "Halo: Recursive proof composition without a trusted setup," https://eprint.iacr.org/2019/1021, 2020.
[10] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable, transparent, and post-quantum secure computational integrity," *IACR Cryptol. ePrint Arch.*, vol. 2018, p. 46, 2018.
[11] B. Bu¨nz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: Short proofs for confidential transactions and more," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 721–736.
[12] V. Buterin *et al.*, "Ethereum: A next-generation smart contract and decentralized application platform," in *White Paper*, 2014.
[13] J. Thaler, "Time-optimal interactive proofs for circuit evaluation," *Advances in Cryptology–CRYPTO 2013*, pp. 71–89, 2013.
[14] Iden3, "Circom: A robust and scalable language for building zeroknowledge circuits," https://docs.circom.io/, 2020.