Proof-of-Useful Work Consensus Protocol for Computing zk-SNARK Proofs

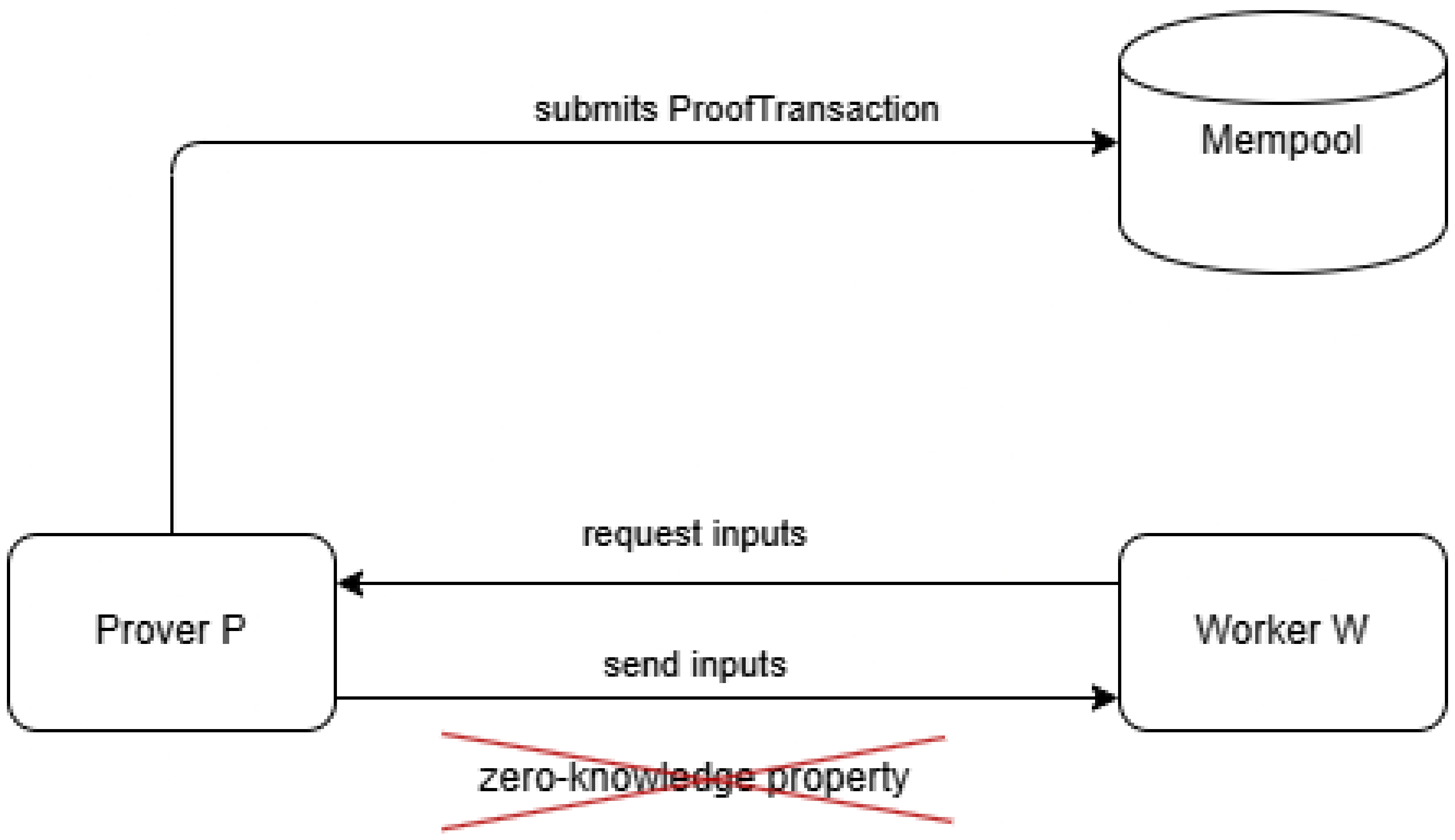
Richard Gazdík xgazdi05@stud.fit.vut.cz
Supervisor: Ing. Martin Perešíni
Consultant: Ing. Samuel Olekšák

Abstract

The increasing demand for efficient and meaningful blockchain consensus protocols has highlighted the limitations of traditional Proof-of-Work (PoW), which consumes substantial computational resources without providing valuable outputs. This paper presents an alternative approach based on Proof-of-Useful-Work (PoUW) for computing zk-SNARK proofs while ensuring that the secret witness remains confidential. Building upon previous research, we design and implement a blockchain-based zero-knowledge proof marketplace that extends the PoUW framework to support private inputs. By integrating secure outsourcing mechanisms and cryptographic protections, the system enables the generation and verification of zk-SNARK proofs without revealing sensitive data. The proposed solution successfully incorporates privacy-preserving outsourcing into the decentralized environment with minimal overhead.

Problem Description

Outsourcing zk-SNARK proof generation with a private witness typically requires the prover to send sensitive inputs directly to the worker, as shown in the diagram.
This breaks the zero-knowledge property and introduces interactivity between the prover and the worker, defeating the core privacy guarantees of zk-SNARKs.



Randomized Witness Encryption

In order to securely outsource zk-SNARK proof generation without revealing the secret witness *w*, we introduce a lightweight encryption technique based on randomization [1]. Instead of encrypting the witness using heavyweight cryptographic primitives, we apply a simple and efficient additive masking scheme.
Before submitting a proof generation task, the prover generates a random vector *r* of the same size as the secret witness *w*. The encrypted witness $\hat{w}$ is then computed as:

$$\boxed{\hat{w} = w + r}$$

where the addition is performed element-wise over the finite field used by the zk-SNARK system.
This transformation ensures that the worker node, which retrieves and processes $\hat{w}$ cannot recover the original secret witness *w* without knowledge of the random mask *r*. From the worker's perspective, $\hat{w}$ appears indistinguishable from random data.
To allow correct proof generation over the randomized witness, the original computation circuit must be adjusted. We define a modified circuit $\hat{F}$ as:

$$\boxed{\hat{F} := (E_2 \times 1) \circ F \circ (D_1 \times 1)}$$

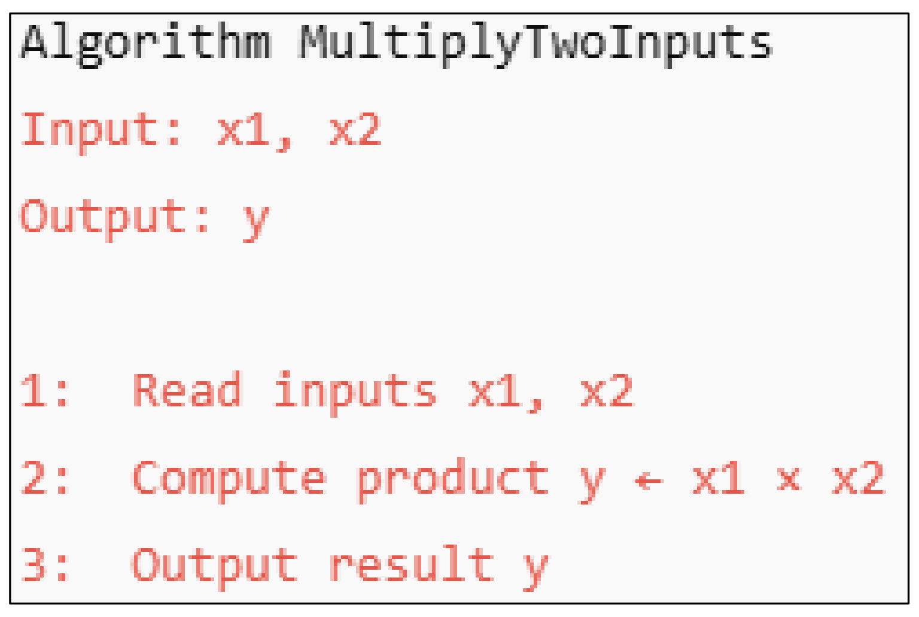
where:

- D_1 is the decryption function corresponding to the prover's randomization, specifically $D_1(x) = x - r$,
- E_2 is an optional re-randomization function applied to outputs if necessary,
- and 1 represents the identity mapping over public inputs and outputs.

This modification ensures that the zk-SNARK proof generation over $\hat{w}$ yields a valid proof for the original computation without revealing *w*.
After the proof is generated and verified, the original witness *w* remains private, as the random vector *r* is never revealed or transmitted to any other party.
The randomized witness encryption method provides several advantages:

- It is computationally lightweight, requiring only basic arithmetic operations.
- It preserves the zero-knowledge property without complex cryptographic assumptions.
- It seamlessly fits into the existing zk-SNARK proving workflow with minimal changes.

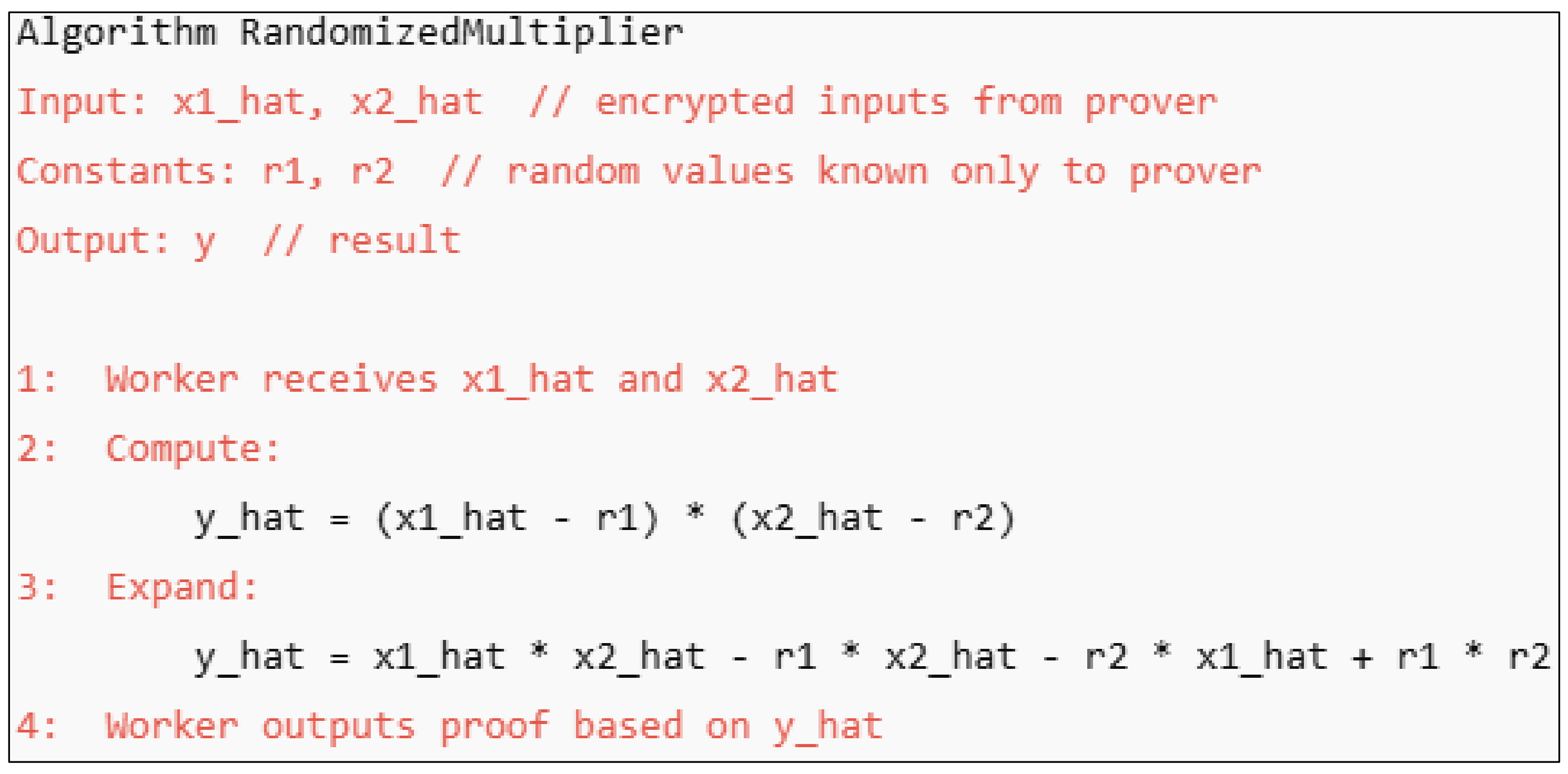
The pseudocode in **Figure 1** illustrates the original circuit design, where operations are performed directly on the plaintext inputs without any witness protection.
In contrast, **Figure 2** shows the adapted circuit design used in our approach, where the computations are modified to operate over randomized (encrypted) inputs, preserving the secrecy of the witness during the proof generation process.



```
Algorithm MultiplyTwoInputs
Input: x1, x2
Output: y

Read inputs x1, x2
Compute product y ← x1 × x2
Output result y
```

Figure 1: Standard Circuit Design



```
Algorithm RandomizedMultiplier
Input: x1_hat, x2_hat  // encrypted inputs from prover
Constants: r1, r2  // random values known only to prover
Output: y  // result

Worker receives x1_hat and x2_hat
Compute:
    y_hat = (x1_hat - r1) * (x2_hat - r2)
Expand:
    y_hat = x1_hat * x2_hat - r1 * x2_hat - r2 * x1_hat + r1 * r2
Worker outputs proof based on y_hat
```

Figure 2: Encrypted Circuit Design

Blockchain Workflow

The Blockchain system consists of **Trusted Setup Nodes**, **Provers**, and **Workers**.
Trusted Setup Nodes (elected by PoS) collaboratively generate the trusted setup parameters using MPC and destroy sensitive files afterwards.
Provers (see Figure 1) create randomness, encrypt their witness, and submit a ProofTransaction to the mempool.
Workers (see Figure 2) retrieve ProofTransactions, generate zk-SNARK proofs using the trusted setup, and include the proofs in new blocks.
This decentralized workflow ensures privacy-preserving proof generation while embedding useful computation directly into the consensus protocol.

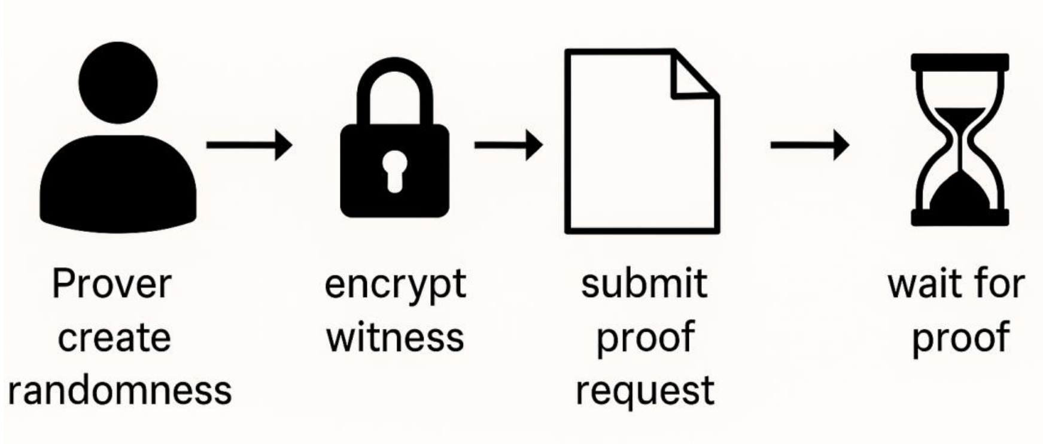


Figure 3: Prower workflow.

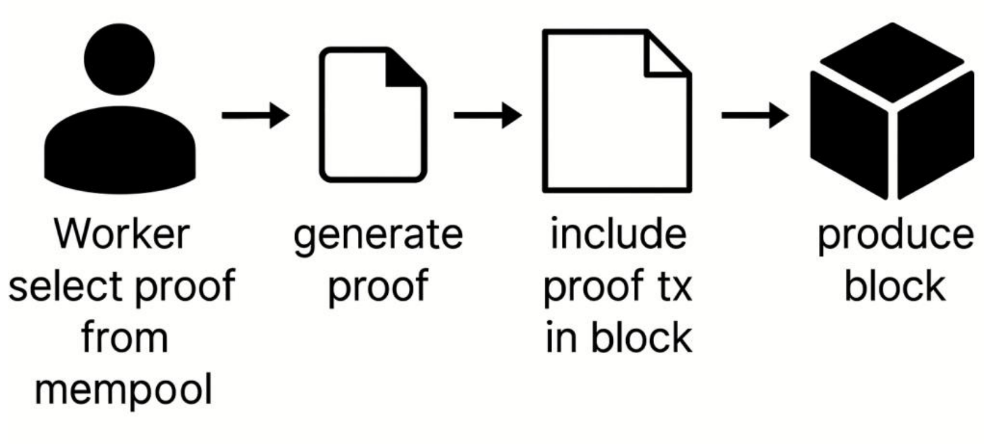


Figure 4: Worker workflow.

Results

Following the marketplace design proposed by Ing. Samuel Olekšák [2], we implemented a Proof-of-Useful-Work (PoUW) consensus protocol for zk-SNARK proof generation while preserving zero-knowledge and non-interactiveness from the prover's side.
Instead of heavy cryptographic constructions, we use lightweight witness randomization and circuit adaptation, adding only negligible overhead compared to standard zk-SNARK workflows.
Alternative technologies like FHE, TEE, and MPC introduce significant overhead and design complexity, making them less practical for consensus integration.
Our solution shows that zk-SNARK-based useful work can be securely and efficiently embedded into blockchain consensus without compromising scalability or security.

References

[1] Nakamura, M., Miyamae, T., Morinaga, M. *A Privacy-preserving Outsourcing Scheme for Zero-knowledge Proof Generation.* Journal of Information Processing, Vol. 30, pp. 151–154, 2022. DOI: 10.2197/ipsjjip.30.151.
[2] Olekšák, S. *The Analysis of Cryptographic Techniques for Offloading Computations and Storage in Blockchains.* Master's Thesis, Brno University of Technology, Faculty of Information Technology, 2024.