zkFold: Zero-Knowledge Prover Backend Close-out Report

Summary

zkFold and Maestro have completed the implementation of ZK Prover Backend product. Our solution enables developers to spin up prover servers to assist ZK-powered applications. A prover server provides a convenient API endpoint to generate a ZK proof given the circuit’s input. The circuit is stored locally on the prover server in order to avoid big data transfers between the client and the server.

ZKP Protocol

We use the protocol that we call *zkFold's PlonkUp ZKP* as our zero-knowledge proof protocol. To generate statements verifiable via this protocol, we have developed a ZK framework (in Haskell) called *Symbolic* as an output of another Project Catalyst proposal. The currently available workflow is as follows:

1. Compile a circuit by implementing the statement you want to prove as a Symbolic function.
2. Compile a prover server equipped with this circuit.
3. Generate the circuit input.
4. Send the circuit input to the prover.
5. The prover returns the public input and proof, corresponding to that circuit input.
6. Repeat steps 3-5 for every instance of the statement that needs to be verified.

Notes:

- A large collection of examples in Symbolic can be found here.
- If you want to verify the those proofs on Cardano, you might want to check out our ZKP verification smart contracts from this repository.

Server setup

Below, we provide a setup guide for spinning up a prover server on Ubuntu 22.04. The steps may differ slightly on a different Linux distribution.

1. Install the necessary tools and libraries by following this guide from CoinCachew.
2. Clone the `main` branch of the zkfold-prover-api repository. Go to the root folder.
3. Use `make build` command to build the server.
4. Launch the server with `make run` command.

By default, the server runs on port 8080.

Server API

The server provides the endpoint `prove` to generate ZK proofs for the clients. This endpoint expects a *witness* in a binary format. The witness type is defined here.

Example query

```
curl -X POST -H "Content-Type: application/json" -d '"MA=="' localhost:808
0/prove
```

ZK Prover via Maestro

For hosting a ZK Prover API server for your use case, contact Maestro at info@gomaestro.org.

Future development

The following quality-of-life features are outside the scope of our Project Catalyst proposal. However, zkFold plans to implement them in the upcoming months.

- Prover optimization with better Rust cryptographic primitives;
- Reading circuits from files;
- API command for installing new circuits into the prover server.

Closing remarks

For any questions, suggestions, or collaboration opportunities, don't hesitate to get in touch with us at info@zkfold.io.