Intro

The previous post referred to the main idea of the private Uniswap or general private/secret smart contract execution.

This idea cannot be the denial of the post "Why you can't build a private Uniswap with ZKPs," but this can be the effective mitigation with some extension.

The previous post, titled "A zkRollup with no transaction history data," was not appropriate to explain the effective secrecy since the scaling topic and the privacy topic were mixed and gave a vague explanation about the secrecy limitation.

Approach

1) First, you adopt the "Option 2" in zkRollup in this post.

A zkRollup with no transaction history data to enable secret smart contract execution with calldata efficiency

There are 2 two options of using txcalldata to restore the full states.

Option 1 is recording all of transaction history data to txcalldata.

Option 2 is recording the diff of the final state as a result of transactions in the block (batch).

In option 2, millions of transactions with the same result of no transactions use 0 gas for the txcalldata use, since there is nothing to record in the txcalldata. The soundness of the Merkle root transition is guaranteed by zkp.

Then you can make a batch of zkRollup without tx history data.

Tx history data will be part of the private input of a zkp circuit (like Groth16 or Plonk) and prove that final state diffs are the correct result of that hidden tx history data.

The remarkable thing is that the final state diffs are the result of many transactions.

Here you find the result is mixed, and hard to distinguish them IF THE USER BALANCES / STATES ARE HIDDEN.

2) Second, zkp and the "user state" model hide the user's balance.

If you define the data model that every asset is described as a leaf of the small Merkle Tree for each user, each user can prove her assets and these relevant transitions with zkp without revealing their balance/assets themselves.

The Merkle proof of the inclusion of the assets bind to the last Merkle root, and the Merkle proof of the inclusion of the changed balance/assets bind to the next Merkle root, can be in the private input of the zkp circuit, and the relevant change of global states can be the public input.

Here you find that the user balances/states are hidden if you combine the first step (result mixing) and this second step.

This is what zkp can do as was researched in the works by Barry Whitehat.

Why you can't build a private uniswap with ZKPs

ZKPs allow you to prove the state of some data that you know. They do not let you prove about things that you do not know.

However, as he said, zkp cannot hide the changes of global states that reveal users' activities.

Why you can't build a private uniswap with ZKPs

So anyone who is able to update the system must have this state info in order to create the zkp that they updated correctly. If they have the state info they can monitor as the state changes. If they can monitor as the state changes they can see what others are doing.

So with ZKPs you end up building private things using only user specific state. So everything is like an atomic swap. If there is global state then this breaks privacy as it needs to be shared for others to make proofs about this state.

If you are the operator, you can know each content of trades in the batch you aggregate. (The important thing here is that other operators cannot know that from the batch.)

In this situation, only the operator and a transactor know what the transactor is doing since all tx histories banish into the private input, and the result is mixed.

So, at the same time, the operators themselves have the "mixing-level" privacy for their transactions because only the operator and a transactor know the activity.

3) Third, combining transactions makes “weak secrecy.”

If the operator is the last person who has secrets of the others, let’s make all transactors operators.

The final operator (is the usual operator) combines all batches all small operators (just the users) make.

The transactor has or generates several dummy accounts and integrates the dummy transactions (or cheap transactions from those) to the main transaction as a batch like at the first step. Then the final operator(the usual operator) only knows that one of the addresses with dummies sent a Uniswap trade transaction.

Such a batch, which is actually a transaction, can be combined with others by applying recursive zkp repeatedly. The proof can be the private input to the next proof.

Finally, the usual operator finds a Uniswap transaction mixed with many activities among many addresses.