This document is the audit of Tornado circuit set performed by ABDK Consulting.

1. Introduction

We’ve been asked to review the circom circuits in a public repo. We referred to the circom internal documentation and Tornado spec for the intended behaviour. No major issues were found. We found a few minor issues (non-optimal templates mainly) many of them fixed in the commits 719, f8c, 071, 9ef.

2. Tornado Templates: Remaining Minor Issue

Withdraw()

- pathIndex should be explicitly checked to contain only zeros and ones. Now checked implicitly inside Merkle tree template. In general, if template contains constraints for ensuring that input values are valid (are bits in this case), then such template is sub-optimal, because in case caller already ensured input validity (i.e. inputs obtained from another template that is guaranteed to return only bits), then input validation constraints will be redundant. In your case, Selector template does check that pathIndex is a bit. This is not redundant only because neither MerkleTree nor Withdraw templates do not guarantee path indices to be bits, but embedding this knowledge into Selector breaks architecture layering, because low level Selector template now knows about internal implementation details of higher-level templates such as MerkleTreeChecker and Withdraw. Moving bit checks from Selector to Withdraw would fix this problem.

Authors' comment: It is only ever used in the merkle tree, and is checked there. Not sure what an explicit check outside will accomplish.

3. Tornado Templates: Fixed Issues

HashLeftRight()

- The n_rounds parameter should be made constant as it is determined by the security level and as it is actually hardcoded in circomlib. MiMC Sponge from circomlib does not support more than 220 rounds.
MerkleTreeChecker()

- Private modifier is not needed as the template is not assigned to main()
- pathIndex should be probably pathIndices
- This could be put into previous loop like this:
  - selectors[i].inputElement <= (i == 0) ? leaf : hashers[i-1].hash;
  - There are N selectors and N hashers, and configuring them all in a single loop of N iterations, so i-th selector and i-th hasher are configured at i-th iteration, would make code easier to read, even if on lower level some expressions will become more complicated.

Selector()

- This code shares a lot with circomlib multiplexors (Multiplexor2, Mux3), and probably should use some of them. Alternatively, see the comment below.
- Most signals can be turned into variables.

Withdraw()

- The rounds parameter should be made constant equal to 220 as currently the circomlib code matches the solidity one only for 220 rounds.
- Squaring inputs seems redundant. Verifier treats all public signals in the same way, so incorrect value for either of them will invalidate the proof, regardless of whether the signal participates in any constraint or not.