Poseidon in Filecoin. Final report. 3 June 2020

Review by ADBK Consulting
Mikhail Vladimirov and Dmitry Khovratovich

Introduction

We were asked by Protocol Labs to review the implementation and the usage of hash function Poseidon (https://eprint.iacr.org/2019/458.pdf). We consulted the Filecoin whitepaper (https://filecoin.io/filecoin.pdf), the privacy preserving proof of storage protocol (supplied privately by Protocol Labs), and the code, where we reviewed the Neptune library (https://github.com/filecoin-project/neptune/tree/f08cc42b8f3894f25cef920d9732d63a1c8cbd22) (the implementation of Poseidon) and rust-fil-proofs library (https://github.com/filecoin-project/rust-fil-proofs/tree/tnet2) (the PoS protocol and its circuits).

Our findings

We carefully studied the protocol and the implementation, and prepared a separate report (https://hackmd.io/41SV4vsfQseM1EN8rgdCmQ). In particular, we checked that:

- The Poseidon permutation parameters (round number, internal constants) are selected properly according to the authors' recommendations.
- The optimized version of the Poseidon permutation, where a more sparse but equivalent matrix is used, is produced correctly.
- The circuit for the Poseidon hash is implemented correctly and corresponds to the regular implementation.

We also discovered the following issues:

- Some hash functions based on the Poseidon permutation are not designed correctly. We recommended a small but sufficient fix that takes the message length into account.
- There is a very wide version of Poseidon for long message hashing, which is correct and secure but probably too slow.
- There is some redundancy in padding.
- Round constants for the Poseidon permutation use a bit order that differs from the reference implementation.
- Several equivalent implementations of Poseidon are not used.

Resolved issues

We checked that all security-critical issues are resolved in the latest commits available:

- Constant generation is fixed (https://github.com/filecoin-project/neptune/pull/23/commits/c8b12c55101c46e05482291912face12300962c3).
- Unused code is refactored into separate files (https://github.com/filecoin-project/neptune/pull/23/commits/08b5164a718d97f94faf1fb3ae694d370c9d28f1).
- The incorrect hash functions are no longer used.

As reported by Filecoin designers, the version of Poseidon with width 9, which allows Merkle trees of arity 8, results in faster Merkle proofs and tree construction in the Filecoin environment. Therefore a concern about too wide permutations is no longer relevant.

Other recommendations

We were also asked to provide a set of parameters for the Poseidon permutation which would provide a bigger security margin against potential attack. We suggested that a 25% increase in the number of rounds should prevent most of attacks, so that even if there is one, the complexity is almost certainly above 2^{64}. We consulted with several other Poseidon authors in private communication, and got their confirmation that it is a sound choice.