Alternative Fully-Validating Zcash Node in Rust

Zcash Foundation Grant Proposal (2017 Q4)

Jason Davies <jason@jasondavies.com>

Motivation and overview

The primary aim is to implement an alternative fully-validating Zcash node in Rust for network robustness: suppose some strange implementation error is found in the Zcash reference client; alternative implementations that do not have this error would help keep the network running if the error was to be exploited, and would help detection of any error in the case where the network stays up but the main chain was determined to be invalid by the alternative implementation.

In addition, using Rust, a memory-safe language, improves security by eliminating particular types of bugs. Adding wallet and proving functionality in the future would make this a fully-functioning client.

Technical approach

I propose porting an existing Rust-based Bitcoin client, parity-bitcoin to support full validation of the Zcash blockchain. I have already written a Rust-based verifier for Zcash zk-SNARK proofs, so it should be easy to include this.

The work can be split into two parts:

1. Modify parity-bitcoin to parse and validate Zcash blocks (along with unshielded transactions); all changes to the consensus protocol relative to Bitcoin need to be supported e.g. difficulty algorithm changes, mining reward schedule. I anticipate this will be around 50% of the work.
2. Validate shielded transactions using zcash-sprout-verifier. This requires implementing various Zcash-specific things such as the incremental merkle tree etc. and should be around 50% of the work.

Background and qualifications

MA (Cantab) CompSci, Univ. of Cambridge, contributor to various open-source projects including Apache CouchDB and D3.js.

I've been following Zcash since its inception and have made some minor contributions so far:

- zcash-sprout-verifier - verifier for Zcash zk-SNARK proofs in Rust.
- zcash-vanity - vanity z-addr generator in Rust and OpenCL.

Evaluation plan

Milestones should reflect the points in the technical approach:

1. Support for parsing Zcash blocks and validation of unshielded transactions. After this milestone is completed, it should be possible to run the client and connect to the Zcash mainnet, and download and validate the full Zcash blockchain (unshielded transactions only).
2. Validation of shielded transactions. After this milestone is completed, it should be possible to run the client, connect to the Zcash mainnet, download and validate all blocks and transactions.

Security considerations

Since the intention is to use this for validation only, security considerations are not as important per se as they do not impact wallets directly, and other

nodes on the network will validate any transactions that this implementation will relay. In the future, the code would need to be extensively reviewed if wallet/proving functionality was to be added.

Schedule

- Parsing and validation of Zcash blocks and unshielded transactions: estimated end of December 2017.
- Validation of shielded transactions: estimated end of January 2018.

Budget and justification

I estimate around $15k would be sufficient to support the development work for a minimal working implementation. Note that I originally submitted a version of this proposal that focused on optimisations of zk-SNARK verification, but after further input I feel it is preferable to prioritise getting a fully-working alternative implementation rather than optimisation work, which can happen later.

Note also that I have submitted a separate proposal for implementing an XCAT web tool, but I do not anticipate this will impact the above schedule much if both are accepted.