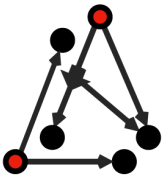
Security Audit Final Report:
Overwinter+Sapling Specification

Zcash

Report Version: 17 August 2018



Least Authority
Freedom Matters

Table of Contents

Background

Purpose

Least Authority performed a security review of the *Zcash Protocol Specification: Version 2018.0-beta-20 [Overwinter+Sapling]*, at the request of the Zcash Company.

Timing

The audit was performed from May 29 to June 13, 2018, by Ramakrishnan Muthukrishnan, James Prestwich, and Jean-Paul Calderone. The initial report was issued on June 13, 2018. This updated report was issued on August 17, 2018 after the verification phase. The report issuance was delayed to allow for further review or discussion of the specification after the Zcash team delayed the Sapling release plans. However, no substantial changes were made to the report during this delay.

Scope

In preparation for the release of Sapling's changes to the Zcash cryptography and consensus protocols the Least Authority team performed this audit.

The *Zcash Protocol Specification: Version 2018.0-beta-20 [Overwinter+Sapling]*, we reviewed included the following Zcash Improvement Proposals (ZIPs):

- https://github.com/zcash/zips/blob/master/protocol/sapling.pdf

During this review, the team focused on the following concerns:

- All issues in scope for Magic Bean/Overwinter,
- Performance problems,
- Consensus failures and
- Game-theoretic issues.

Findings

Although Zcash has only been around for a short time, it is clear that the contributing team has put significant effort into this protocol specification and the subsequent Sapling updates. In this review, we were unable to identify any significant issues, only two minor issues, worth noting in this report. We have included comments and suggestions on the specification because we think they offer insight into the readability and interpretation of the specification.

Issues

Issue A: Spec is inconsistent regarding fOverwintered

Synopsis

According to the specification, fOverwintered MUST be set from Overwinter onward, but MUST NOT be set pre-Sapling. This is likely a typographical error.

Impact

Negligible

Technical Detail

Section 7.1 lists the following conflicting requirements for transactions:

• [Pre-Sapling] The fOverwintered flag MUST NOT be set.

• [Overwinter onward] The fOverwintered flag MUST be set.

Overwinter is pre-Sapling, and thus according to the specification fOverwintered both MUST and MUST NOT be set for Overwinter transactions.

Remediation

Update the spec to read as follows:

• [Sprout] The fOverwintered flag MUST NOT be set.

• [Overwinter onward] The fOverwintered flag MUST be set.

Status

Resolved.

Verification

Verified. Change in Version 2018.0-beta-21 [Overwinter+Sapling] in *Consensus Rules* on page 77.

Issue B: Transparent address encoding is compressed

Synopsis

According to the specification, the P2PKH address has uncompressed ECDSA key encoding, however, the reference code uses compressed key encoding.

Impact

Negligible. Potentially confusing to wallet implementers who do not share code with reference Zcash implementation.

Technical Detail

Section 5.6.1 specifies how P2PKH address is encoded. It says

"20 bytes specifying a public key hash, which is a RIPEMD-160 hash [RIPEMD160] of a SHA-256 hash [NIST2015] of **an uncompressed** ECDSA key encoding."

However, the Zcash reference implementation seem to use "compressed" ECDSA key serialization[1].

Remediation

Update the spec to read as follows:

20 bytes specifying a public key hash, which is a RIPEMD-160 hash [RIPEMD160] of a SHA-256 hash [NIST2015] of **a compressed** ECDSA key encoding."

Status

Resolved.

[1] https://github.com/zcash/zcash/blob/master/src/wallet/wallet.cpp#L131

Verification

Verified. Change in Version 2018.0-beta-21 [Overwinter+Sapling].

Suggestions

Suggestion A: Include more explanations for design decisions

Summary

There are many cryptography concepts being deployed in Zcash that understanding the specification requires deep knowledge of cryptographic principles and engineering implementations. The use of zero knowledge proofs, alone, is a challenging new advancement and in Zcash is combined with the use of pairing crypto, commitment schemes, and methods for resistance against various kinds of replay and forgery attacks. It may be helpful to include explanations of why Zcash needs to use these various algorithms to help anyone reading the specification to better assess the use.

Status

No changes made. Checked in Version 2018.0-beta-21 [Overwinter+Sapling].

Suggestion B: Include more guidance on zk-SNARKS

Summary

More guidance on the use of zk-SNARKS in Zcash would be helpful. Posts like "Quadratic Arithmetic Programs: from Zero to Hero[2]" by Vitalik Buterin are helpful for providing context and explanation for the specification of Zcash.

Status

Partially updated. The *Appendix A* in Version 2018.0-beta-21 [Overwinter+Sapling] has some newly added information, which is very helpful for the reader. However, it is incomplete with a number of TODOs with certain sections not written, yet.

Comments

Comment 1: Section 5.8 "TODO"

Summary

In this version of the specification, Section 5.8 Sapling zk-SNARK Parameters (page 70) notes that "These parameters were obtained by a multi-party computation described: TODO." We did not find this concerning, but can be revisited when the section has been completed.

Status

No change required.

[2] https://medium.com/@VitalikButerin/quadratic-arithmetic-programs-from-zero-to-hero-f6d558cea649

Comment 2: Inclusion of Appendices

Summary

The Appendices are helpful, especially the A.3.1.1 appendix on algebraically expressing boolean constraints.

Status

No change required.

Comment 3: Interpretation of “x”

Summary

In Section A.3.1 it only seems reasonable to interpret "×" as scalar multiplication but in section 2 "×" is defined as cartesian product and "·" is defined as multiplication (over "integers, rationals, or finite field elements"). If this is incorrect, it should be discussed.

Status

Updated. The Zcash team made a typographical distinction between constraint multiplication and cartesian product. We agree this clarification makes the reading unambiguous, now.

Comment 4: Clarification of `KA.Base` and `KA.Public`

Summary

In Section 4.1.4 (page 18), the type signature of `KA.DerivePublic` says `KA.DerivePublic : KA.Private × KA.Public → KA.Public`. Although it looks like `KA.Base` is just a type synonym for `KA.Public`, it could be read as `KA.DerivePublic : KA.Private × KA.Base → KA.Public`.

Status

No change. The Zcash team advised that this is incorrect (KA.Base is a specific base value used in Sprout, not the type of possible bases).

Comment 5: Clarification of selection constraints

Summary

In Section A.3.1.3, we interpreted it as *b?x:y = z* implies *if b is true then x binds to z else y binds to z*. This interpretation makes more sense than *if b is true then the value of x else y binds to z*. However, it is unclear what the construct is in a context where the two options could be compared against each other.

Status

Updated. The Zcash team added parentheses. We agree this change provides clarification.

Comment 6: Clarification of certain acronyms

Summary

Acronyms like *ock* (Outgoing Cipher Key) are not documented in the specification. We recommend including such information to support the specification being a standalone document. This would allow it

to better serve those who are trying to create alternate implementation of the Zcash protocol that would interoperate with the official Zcash software.

Status

Partially updated. The Zcash team has made some edits. OVK (outgoing viewing key) is now explained, but OCK (outgoing cipher key) remains unexplained. We recommend that this also be defined.

Recommendations

We recommend that the two issues found be addressed, the suggestions be considered and the comments be taken into account.

Additionally, we recommend that we discuss the possibility of Least Authority facilitating further review of the specification by additional people. Although the Zcash Company is on a set schedule for the Sapling release, we still think it is worthwhile to have further review of the specification. If this is of interest, Least Authority can seek out additional reviewers with particular skills, including any reviewers referred by the Zcash team.

Appendix 1: Audit Process

Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protects users. The following is the methodology we used in our security audit process.

Audit Preparation

Prior to auditing the Sapling specification, we reviewed the Magic Bean implementation and Overwinter specification. This gave our team a foundational appreciation and understanding of Zcash's project vision and status. We installed and used the Zcash code to operate a full node and interact with the network. While we do this, we brainstorm threat models and attack surfaces. We hypothesize what vulnerabilities may be present, creating Issue entries, and for each we follow the following Issue Investigation and Remediation process.

Audit Activities

We started this audit by reading the specification version *2018.0-beta-20* from the beginning with a special emphasis on the Sapling related changes.

In reviewing the specification, we looked for any potential issues with design logic, cryptographic errors, and weak assumptions. We also analyzed for areas where different approaches could possibly reduce the risk of future issues.

In general, the Sprout to Sapling changes indicated in green typeface were given a lot of scrutiny. The new cryptographic algorithms and functions were looked at by looking up the referenced literature to see what functions it fulfils and whether the proposed implementation poses any issues.

In particular, we spent some extra time understanding the notations and the diagrams in *Section 3.1: Payment Addresses and Keys* to understand how the various derived keys fit together and to understand what roles they play in ensuring the privacy of users.

The consensus rules changes, encoding of transactions and various descriptions in Sapling were analyzed because they changed significantly compared to the pre-Sapling consensus rules.

Since the zk-SNARK is at the heart of the privacy properties of ZCash, we spent some time to understand zk-SNARK theory of operation by reading the *Appendix A.1: Quadratic Constraint Programs* and other blog posts on the zcash website.

In our investigations, we also assessed the status of the supporting documentation to see if there were any areas where additional documentation, explanation or diagrams could speed up future audits. Although our primary focus is on the in-scope specification we examined potential dependency behavior as it was relevant to a particular line of investigation. This included reviewing related public literature on the topics to ensure a broader understanding of the proposed implementation compared to industry standards.

Additionally, we reviewed the GitHub issues in the Zcash repository to understand how the Jubjub curve parameters were arrived at. We found the subsequent discussions around Jubjub to show that it is based on a clearly communicated scientific basis, therefore not leading to any concerns about the security of Zcash users

Documenting Results

We followed a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through to a successful remediation. Whenever a potential issue was discovered, we reported our findings to the Zcash team even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions by creating an Issue entry for it in this document, then confirming the issue through further analysis and discussions.

Suggested Changes

We strived to suggest actionable and immediate mitigations that can be made to the specification, both minor and overall changes. We expected our recommendations to be scrutinized by the Zcash team to ensure the is an ongoing collaborative process after we deliver our report, and before these details are made public.