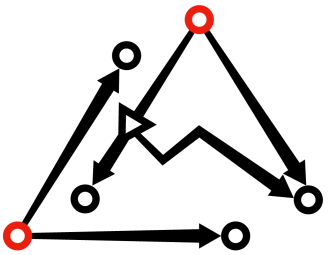

Least Authority

PRIVACY MATTERS

Splitcoin - Manual Seed Phrase Encryption

Whitepaper Review Summary Report

# Splitcoin Inc.

Whitepaper Review Summary Version: 31 July 2023

# Table of Contents

# Overview

## Background

Splitcoin, Inc. has requested that Least Authority provide consulting services in order to review and offer feedback on the Splitcoin Manual Seed Phrase Encryption Whitepaper, which serves as the fundamental guide for the Splitcoin project. Splitcoin is a manual encryption system for seed phrases. Its primary purpose is to serve as a seed phrase management solution for non-custodial cryptocurrency wallet users.

## Project Dates

- **August 3, 2022 - September 7, 2022**: Whitepaper Review *(Completed)*
- **September 9, 2022**: Delivery of Initial Whitepaper Review Report *(Completed)*
- **July 31, 2023**: Delivery of Final Whitepaper Review Report *(Completed)*

## Review Team

- Alicia Blackett, Security Researcher and Engineer
- Nikos Ilikas, Security Researcher and Engineer
- DK, Security Researcher and Engineer

## Whitepaper

The following whitepaper is considered in scope for the review:

- SplitcoinWhitePaper_8.8.22_CONFIDENTIAL.pdf *(shared with Least Authority via email on 24 August 2022)*

However, a complete audit of the code associated with Splitcoin's Manual Seed Phrase Encryption is considered out of scope.

## Supporting Documentation

The following documentation was available to the review team:

- CreateVault_NoPassword_CONFIDENTIAL.png *(shared with Least Authority via email on 24 August 2022)*
- SplitcoinWhitePaperRevision_5.31.2023_CONFIDENTIAL.pdf (*shared with Least Authority via email on 31 May 2023)*
- Splitcoin Videos *(shared with Least Authority via email on 27 July 2022)*
- App Video: Create Vault *(shared with Least Authority via email on 31 May 2023)*
- App Video: Open Vault *(shared with Least Authority via email on 31 May 2023)*

In addition, this Whitepaper Review Summary Report references the following documents and links:

- K.M. Martin, Everyday Cryptography Fundamental Principles and Applications, University Press, Oxford, 2017.
- Zeroization: https://csrc.nist.gov/glossary/term/zeroization
- V1: Architecture, Design, and Threat Modeling Requirements: https://mobile-security.gitbook.io/masvs/security-requirements/0x06-v1-architecture_design_and_threat_modelling_requireme

## Areas of Concern

Our investigation focused on the following areas:

- Coverage and completeness: A general review for design comprehensiveness;
- Logical and realistic technical design details: A review for basic technical details that would likely result in a rational implementation;
- Potential security issues: A security review of the system's design with a focus on data confidentiality and system integrity;
- Mechanisms and incentives: An analysis of the token functions within the larger system and how various parties interact with it; and
- Anything else as identified during the initial analysis phase.

## Project Phases

The Whitepaper Review consisted of:

- A high-level review of the Splitcoin Manual Seed Phrase Encryption design documentation by the Least Authority team;
- An overview presentation of Splitcoin Manual Seed Phrase Encryption by the Splitcoin technical team, along with time for Q&A;
- Analysis and preparation of a *Whitepaper Review Summary Report* by the Least Authority team, including feedback and open questions; and
- A follow-up discussion with Splitcoin and Least Authority, if desired.

# Findings

## General Comments

Our team performed a whitepaper review of the Splitcoin Manual Seed Phrase Encryption (Splitcoin), a manual encryption system for seed phrases. Splitcoin is mainly used as a seed phrase management tool for non-custodial cryptocurrency wallet users. The purpose of this review was to assess the security of the design of Splitcoin as described in the whitepaper, and to identify potential security vulnerabilities within that design. Our team identified recommendations and suggestions to improve the overall security of the system.

The Splitcoin whitepaper provides a high-level overview of Splitcoin's intended functionality. The reasoning behind the design choices in Splitcoin's use of cryptographic primitives is described, including details on how these primitives are used to form their cryptographic scheme. The whitepaper describes cryptanalysis attacks, where much emphasis is placed on man-in-the-middle (MitM) attacks, while there are also differential attacks, which are relevant in this case. We recommend adding additional links and references in the whitepaper to add clarity for readers (Suggestion 4).

The security of the ciphertext has been taken into consideration. However, this is not necessarily the case in the context of the whole system. Due to the early stages of the project, our team did not review the code for the system in its entirety. Our recommendations and feedback are based on known good practices and examine the partial code in the whitepaper (Suggestion 6) and (Suggestion 8).

### Feedback and Recommendations

In addition to our comprehensive review of the design of the system as described in the whitepaper, our team investigated specific security issues and concerns highlighted by the Splitcoin team:

- There does not appear to be a feasible attack, which would allow the ciphertext to be decrypted, if one or more of the coins were missing. Most attacks that occur within NFCs are when the tag itself is malicious (for example, if the tag tries to trick the user into visiting a malicious website). The most feasible attack for the Splitcoin use case is denial-of-service (DoS), where a malicious user overwrites or erases data. The use of NFC tags with built-in anti-collision reduces the risk of data corruption. However, the user can also unintentionally alter the data on the coins, rendering them useless (Suggestion 3). Malicious software cannot trick users into reading altered data, due to the authentication of `AES-256_GCM`.
- Although the memory has a data retention period of 50 years, this does not necessarily mean that all components will last for 50 years. The antennae within the tag and the logic that controls memory access are not always built from the same process and may not be as resilient. In addition, they can suffer from different malfunctions other than memory. One concern that the Splitcoin team has, is that the main purpose of the tags given by the manufacturer is for supply chain management. More research should be carried out into the longevity of all components.
- For the use case of Splitcoin, the use of the password means that only encrypted data is sent and stored on the coin. However, this process does not implement end-to-end encryption in a traditional sense, as both ends would need to be able to perform encryption and decryption.

### Additional Areas of Improvement

Although our team did not conduct a complete code review, we did examine the Splitcoin Create Vault process and the Splitcoin Open Vault process at the system level, including the physical implementation of secure data transfer when using NFC technology. Due to the lack of code on the mobile phone implementation in the whitepaper, our review of the design was limited. We recommend that the Splitcoin team conduct a follow-up audit of the mobile phone implementation to examine the coded implementation in addition to the system design, and to perform and document a threat modeling analysis once the system design is finalized (Suggestion 7).

# Feedback & Recommendations

| SUGGESTIONS | STATUS |
|---|---|
| Suggestion 1: Remove the Use of Encryption in No Password Mode | Resolved |
| Suggestion 2: Allow User to Recover the Password in the Application | Resolved |
| Suggestion 3: Reduce Access to Commands That Can Alter Data | Resolved |
| Suggestion 4: Add Links/References | Resolved |
| Suggestion 5: Increase the Minimum Character Requirements for the Password | Resolved |
| Suggestion 6: Design and Implement Zeroization Mechanism | Resolved |
| Suggestion 7: Perform Threat Modeling to Accurately Describe Mobile Application Security Aspects | Resolved |
| Suggestion 8: Do Not Use String Type to Store Sensitive Data | Resolved |

| | |
|---|---|
| Suggestion 9: Make Some of the Code Open Source to Support the Longevity of the Memory | Partially Resolved |

## Suggestion 1: Remove the Use of Encryption in No Password Mode

**Observation**

The security of the passwordless mode relies solely on an attacker not having access to all coins. Encrypting the plaintext on the coins and adding the encryption key to the coin unencrypted does not add more security than separating the coins.

**Recommendation**

We recommend that the Splitcoin team either use this mode as a plain storage mode without encryption, or remove this mode completely. Otherwise, the user may be misled into believing that encryption here adds extra security.

**Status**

After discussion with the Splitcoin team, it became clear that the premise for this suggestion is incorrect, and recent design changes render it unnecessary. As such, we consider this suggestion resolved.

**Verification**

Resolved.

## Suggestion 2: Allow User to Recover the Password in the Application

**Observation**

The application allows users to enter a password in the password-protected mode, and this is used as an input to encryption. If the user forgets their password, they are no longer able to access the seed phrase. Most users of a mobile application are accustomed to being able to recover or reset their passwords.

**Recommendation**

Given that the password is set in the application and is fundamental to recovering the seed phrase, we recommend allowing the user to recover it within the application.

**Status**

The Splitcoin team has stated that they improved key management in the current design, but prefer to not use their own mechanisms to restore the password.

**Verification**

Resolved.

## Suggestion 3: Reduce Access to Commands That Can Alter Data

**Observation**

The `read write` tool of NXP NFC allows the user to erase user memory or overwrite it. These commands will result in the loss of the seed phrase.

#### Recommendation

We recommend that the `read write` tool be integrated in such a way that it does not allow users to access destructive commands. We further recommend that the lock mechanism for user memory be made compulsory to avoid overwriting data needed to recover the seed phrase.

#### Status

The Splitcoin team has made the lock mechanism compulsory on `write` commands to prevent the overwriting of tags and the altering of data.

#### Verification

Resolved.

## Suggestion 4: Add Links/References

#### Observation

The whitepaper cites claims from articles, but links to those specific articles are not attached to the whitepaper for reference.

#### Location

Page 9, at the bottom: "*AES-256-GCM is semantically secure … It is the strongest encryption standard in the world. It is used by governments, financial institutions, and armed forces.*"

Page 10: "*AES-256 is even believed to be quantum-resistant, unlike… (not working)*"

Page14: BIP39

Page 15: `RNGCryptoServiceProvider`, pepper

#### Recommendation

We recommend adding links to help support the claims written in the paper and provide reassurance to readers.

#### Status

The Splitcoin team has added references to the whitepaper.

#### Verification

Resolved.

## Suggestion 5: Increase the Minimum Character Requirements for the Password

#### Observation

Passwords of 8 characters length, which is the minimum requirement (among one symbol and one capital letter requirements), are small, notably in such applications.

#### Recommendation

We recommend changing the minimum requirement to 12 characters (along with symbol and capital letter requirements).

#### Status

The Splitcoin team has increased the minimum requirement to 20 characters.

#### Verification

Resolved.

## Suggestion 6: Design and Implement Zeroization Mechanism

#### Observation

Zeroization of sensitive data stored in memory is not described in the whitepaper and is likely not implemented in the system.

#### Recommendation

We recommend that appropriate zeroization of sensitive fields be integrated into the design and implementation of the system.

#### Status

The Splitcoin team has implemented a zeroization mechanism.

#### Verification

Resolved.

## Suggestion 7: Perform Threat Modeling to Accurately Describe Mobile Application Security Aspects

#### Observation

The system uses iOS and Android mobile applications. Concurrently, the whitepaper does not mention security threats and mechanisms addressed on this layer, such as whether the mobile application stores secrets (which should not be a function), how cryptographically secure random numbers are generated, and how secrets are zeroized.

#### Recommendation

We recommend that appropriate threat modeling be performed to identify security vulnerabilities that could impact the implementation and that mitigations and remediations for those vulnerabilities be considered and integrated into the design. This exercise and its results should be well-documented and included in the project documentation. At minimum, the following basic security requirements must be addressed:

- No sensitive data should be written to application logs;
- The keyboard cache should be disabled on text inputs that process sensitive data;
- The application should remove sensitive data from views when moved to the background;
- The application should not hold sensitive data in memory longer than necessary, and memory should be cleared explicitly after use;
- The application should prevent usage of custom third-party keyboards whenever sensitive data is entered; and
- The application should protect itself against screen overlay attacks.

#### Status

The Splitcoin team has added a description of their threat modeling in section 4.4 of their whitepaper.

#### Verification

Resolved.

## Suggestion 8: Do Not Use String Type to Store Sensitive Data

#### Observation

`String` type is used to store cryptographic keys and passwords. Using type `string` means that the seed phrase is captured from the user and stored in plaintext in memory before being encrypted. The general `string` type is also immutable and therefore cannot be zeroed out using normal methods. In addition, with each copy of the string, a new variable is created in memory for the new value.

#### Location

Screenshots of the whitepaper on pages 12 and 16.

#### Recommendation

We recommend using special types or zeroization mechanisms to store sensitive data. We further recommend that the Splitcoin team investigate the use of secure `string` types and implement methods that protect the seed phrase from capture to encryption, which include ensuring that:

- The seed phrase is not captured in plaintext;
- The seed phrase is not persisted through events such as pausing and resuming;
- The seed phrase is not immutable or has a zeroing method; and
- The seed phrase is not stored in general memory but protected memory.

#### Status

The Splitcoin team has implemented a zeroization mechanism using the recommended methods.

#### Verification

Resolved.

## Suggestion 9: Make Some of the Code Open Source to Support the Longevity of the Memory

#### Observation

Although the data retention of the memory block is 50 years, there is no guarantee that other components will last that long or that third-party `read write` tools will be supported for this period of time.

#### Recommendation

In the event that Splitcoin ceases to exist, and as a countermeasure to device failure, we recommend that Splitcoin make a portion of the code open source. It should be enough code to allow a user to retrieve their seed phrase, if any of the above conditions occur.

#### Status

The Splitcoin team has considered the recommendation and implemented it partially. The minimalistic open-source tool will be implemented and shared in the future.

#### Verification

Partially Resolved.

# About Least Authority

We believe that people have a fundamental right to privacy and that the use of secure solutions enables people to more freely use the Internet and other connected technologies. We provide security consulting services to help others make their solutions more resistant to unauthorized access to data and unintended manipulation of the system. We support teams from the design phase through the production launch and after.

The Least Authority team has skills for reviewing code in multiple Languages, such as C, C++, Python, Haskell, Rust, Node.js, Solidity, Go, JavaScript, ZoKrates, and circom, for common security vulnerabilities and specific attack vectors. The team has reviewed implementations of cryptographic protocols and distributed system architecture in cryptocurrency, blockchains, payments, smart contracts, and zero-knowledge protocols. Additionally, the team can utilize various tools to scan code and networks and build custom tools as necessary.

Least Authority was formed in 2011 to create and further empower freedom-compatible technologies. We moved the company to Berlin in 2016 and continue to expand our efforts. We are an international team that believes we can have a significant impact on the world by being transparent and open about the work we do.

For more information about our security consulting, please visit https://leastauthority.com/security-consulting/.