A quick preface before the summary, from the Zucoins Team.

Prior to receiving this report, the Zucoins Team received a full report breakdown detailing the exact implementation analysis, spanning over 20 pages.

The key focus of the audit was to review the cryptographic implementation and provide useful feedback for our development team.

We greatly appreciated the communication, collaboration and feedback with UL during this period.

All of the five recommendations were reviewed by our team and the system was promptly updated in accordance with the recommendations.

In particular, further clarification is needed for the original nonce implementation change that was advised by UL's cryptographic analysis review process, of which several of their technical staff, including two cryptographers, reviewed components of the SplitChain system in its early demonstration state at the time.

Our implementation of nonces was somewhat untraditional but logical. In several cryptographic resources we utilised, in addition to our own calculations, it was determined that it would likely be extremely difficult to guess the nonce using the original implementation method. This is because our implementation placed the nonce alongside the private key that the user of the system keeps safety protected. The nonce is never stored alongside the circulated encrypted data, as it normally does in common implementations of the utilised encryption method.

While nonces in the utilised cryptographic method are generally advised to be generated anew on each encryption cycle, this is partly due to the nonces being stored alongside the encrypted data, that is often publicly circulated with the encrypted data itself, e.g. as a packaged file containing both elements. As a generated nonce shouldn't be reused, this explains why this recommendation is often adhered to.

Our reason for this alternative implementation was to isolate the circulating encrypted data and keep anything other than the encrypted data itself, placed alongside the private key, where it would more likely be kept safe by the user. In our view, while unlikely, any additional information to accompany the encrypted data, could in future be discovered as an attack vector for the encryption algorithm—even if such a thing is yet to be discovered at this point. We did this to reduce any potential attack surface of the encrypted data that is shared amongst the network of peers. This is because there is, of course, a mathematical relationship between the key(s) used to the encrypt the data, the encrypted data and the nonce itself.

In the full audit report and in email correspondences with UL's cryptographic analysis team at the time of the audit, they considered that while this may be valid reasoning, however for compliance, they recommended reverting the implementation to the most commonly established practise of generating a new nonce on each encryption process that occurs. Meaning, the nonce would then be stored with the publicly circulating encrypted data itself, instead of alongside the data containing the private key that is expected to be protected by the user (otherwise, the process of backing up an offline user wallet would become more frequent, amongst other negative user experience effects we observed).

Since our UL audit, we promptly made all recommended changes to the system in the weeks following the audit's completion and have utilised their recommendations in our internal company processes.

—Zucoins Team

Identity Management & Security

UL International (France) SA

Z.I Athélia IV

174, Avenue du Jujubier

13703 La Ciotat Cedex, France

La Ciotat, 24 October 2019

At the request of Zukaz, UL has performed a security evaluation lasting 18 days. The target of evaluation was the client side JavaScript implementation of the SplitChain solution and comment on the cryptography used. The final report has been delivered the 29 August 2019.

Below some extract of the Security Assessment :

“SplitChain is a solution designed by Zukaz to ensure the security of the cryptocurrency Zucoins. The JavaScript based wallet code is designed to be integrated into web applications. SplitChain functionality includes creation of a new wallet, transfer of value from one wallet to another and wallet management functions.

Cryptographic primitives, including key generation, encryption and decryption are handled by the open source library, libsodium.

UL was asked to study the client side JavaScript implementation of the SplitChain solution and comment on the cryptography used. This report presents our findings based on materials provided by the Zukaz team.

The objective of the present Security Assessment is to identify and report on security vulnerabilities to allow the client to improve the security protection of their solution. The client understands that security is a continually growing and changing field and that this Security Assessment does not mean to guarantee that the client’s product is secure from every form of attack.” There is no such thing as 100% security evaluation coverage as there is always something else that can be tried. ”This Security Assessment provides a snapshot of the current security problems of the system, and it is limited in terms of time and personnel. Therefore, they cannot provide a 100% guarantee that every attack vector has been included, and that the system will stay secure over time.”

“During the review the supplied documentation and source code was understood and analyzed. Details of the implementation were assessed against commonly accepted key management principles, industry standard cryptographic algorithms and known vulnerabilities in cryptographic systems.”

As a result, the recommendations following the security assessment are :

- "Add code to zeroise secret or private key material when no longer required. If possible, avoid creating any copies of secret or private key material.
- Minimize the access to the secret and private key material from functions or code which do not need access.
- Create a new nonce whenever the wallet is being encrypted.
- Separate the nonce storage from the wallet encryption key storage.
- Consider whether it is possible to move secret and private key operations to a more secure environment, e.g. a hardware token, protected security zone in the device (e.g. Apple Secure Enclave, Android KeyStore, Trusted Platform Modules) or isolated software environment (e.g. Virtual Machine, Docker)."

And the summary of the security assessment is :

"Review of the supplied source code and specifications confirmed that industry standard algorithms and implementations are used for cryptography. Cryptographic APIs were found to be used correctly, except for reuse of the nonces used during wallet encryption.

The biggest risk to the cryptography of the solution is hackers stealing the secret and private keys from the client whilst the code is executing. Industry best practice is to store and process plaintext cryptographic keys in secure cryptographic devices or physically and logically secured environments[1]. Processing keys in plaintext using JavaScript in a web browser has potential security issues, however these are outside the scope of this project.

The communications protocol, data format and synchronization mechanism were out of scope of this study and no comment is made on the suitability of the protocol or system.

Yours Faithfully,

[illegible]

Fabrice Heiser

UL Mobile Security Service Line Manager

[1] ISO 11568, FIPS 140-2, ANSI X9