Response to the November 2020 Trail of Bits Audit

In November 2020, Trail of Bits conducted a security review of Tezori / Galleon and ConseilJS. After considering the resulting security recommendations, Cryptonomic made judicious choices about the fixes to be made. In this document, we are memorializing our rationale for our users' benefit.

1. Remote code execution via openExternal [Fixed]

Fixed

2. Unencrypted secrets in memory [Not fixed]

There seems to be no reasonable way to do this in the Javascript family of languages as there is no direct memory access. As a result, secrets such as keys decrypted in memory must wait to be garbage collected. While this is not ideal, we feel the possibility of an exploit is low. One long term possibility we have in mind is to write a module in Rust or C++ to handle in-memory secrets, transpile it into an NPM package and use it in the Galleon codebase.

3. Insecure private key in-memory encryption [Partially fixed]

The ConseilJS SoftSigner needs to have the raw key material to sign operations. The default for initializing a SoftSigner instance provides for storage of the key in encrypted form. This is the way Galleon uses it. However for cases where this is not necessary we continue to allow storage of raw key material for developers who need it.

4. Access to raw private key and signing without additional authentication [Partially fixed]

This was not flagged as being remotely exploitable by compromising Galleon. We expect our users to follow basic security precautions like setting a screen lock timeout.

5. Signing valid operation hashes via UI dialog [Not fixed]

Unclear how this can be validated to prevent *all* attacks. We display the payload being signed and expect the user to exercise due caution when performing operations. This functionality already signs arbitrary payloads for audit purposes for example where the audit firm controls the text we sign.

6. Wallet file permissions [Partially fixed]

t’s not clear what benefit would be gained by checking permissions of a file that would be fed into a JSON parser to extract text

7. Ignored exceptions [Not fixed]

Informational issues were not addressed.

8. Users can be tricked to blindly sign transactions [Fixed]

Fixed

9. Client-side request forgery through dApp authentication [Not fixed]

Fixed. This feature was removed.

10. User interface bugs [Not fixed]

Informational issues were not addressed. Galleon is being actively worked on and we address issues that affect usability.

11. Wallet’s password not cleared from a dialog box [Not fixed]

Although this was not fixed at the time of the audit, it is now being fixed.

12. Operations can be injected by a Tezos node before signing [Fixed]

Fixed

13. Entrypoint not validated, possible injection of data to sign [Not fixed]

This code is part of the encoding library. It is meant to be general purpose and to be able to encode correctly any valid value, in this case, name of an entry point. There is no reasonable way to validate if the entry point being provided is incorrect or that it’s somehow malicious.

14. URL components not encoded [Not fixed]

Informational issues were not addressed.

15. Arguments to contract’s parameters not encoded [Not fixed]

Informational issues were not addressed.

16. JSONPath argument is not escaped [Not fixed]

Informational issues were not fixed. In this particular case, JSONPath arguments are not passed as parameters either from the user or the network. The values are built into the application and cannot be changed during the normal course of operation.

17. Discrepancies between master branch and latest release [Fixed]

Fixed.