Kinto SuperBridge Hooks Security Analysis

by Pessimistic

This report is public

June 6, 2024

Abstract

In this report, we consider the security of smart contracts of Kinto SuperBridge Hooks project. Our task is to find and describe security issues in the smart contracts of the platform.

Disclaimer

Summary

In this report, we considered the security of Kinto SuperBridge Hooks smart contracts. We described the audit process in the section below.

The audit showed several issues of medium severity: Project roles, Overpowered role. Also, one low-severity issue was found.

After the initial audit, the codebase was updated.

The developers fixed the Overpowered role issue of medium severity, commented on the Project roles issue of medium severity, and fixed the low severity issue.

The number of tests decreased by 1.

General recommendations

We recommend implementing CI to run tests, calculate code coverage, and analyze code with linters and security tools.

Project overview

Project description

For the audit, we were provided with Kinto SuperBridge Hooks project on a public GitHub repository, commit c1e437dbaa0b6475be570c5aa208e58606a628a1.

The scope of the audit included the following contracts:

- **contracts/hooks/KintoHook.sol**;
- **contracts/hooks/LimitHook.sol**;
- **contracts/hooks/SenderHook.sol**.

and their dependencies:

- **contracts/hooks/plugins/ConnectorPoolPlugin.sol**;
- **contracts/hooks/plugins/LimitPlugin.sol**;
- **contracts/hooks/HookBase.sol**;
- **contracts/utils/Gauge.sol**;
- **contracts/utils/RescueBase.sol**;
- **contracts/libraries/RescueFundsLib.sol**;
- **contracts/common/Errors.sol**;
- **contracts/common/Constants.sol**;
- **contracts/interfaces/IHook.sol**;
- **contracts/interfaces/IController.sol**.

The documentation for the project included SuperBridge description and the SUPERBRIDGE_README.md.

All 131 tests pass successfully. The code coverage is 78.87%.

The total LOC of audited sources is 647.

Codebase update

After the initial audit, the codebase was updated again, and we were provided with commit 0a75a5943b234de5185bbe3abfe796b1b6ea4747.

The developers fixed or commented on all the issues.

All 130 tests passed successfully. The code coverage was 78.87%.

Audit process

We started the audit on May 31 and finished on June 5, 2024.

We checked the SuperBridge Kinto hook, which is the fork of the SuperBridge, and inspected the materials provided for the audit. The audit included only the code with hooks functionality.

During the work, we stayed in touch with the developers and discussed confusing or suspicious parts of the code.

We manually analyzed all the contracts within the scope of the audit and checked their logic. Among other, we verified the following properties of the contracts:

- Whether it is not possible to change the bridge data;
- How the admin roles can influence the project;
- Whether it is not possible to bypass the checks and get tokens not on the Kinto wallet;
- Whether the Kinto wallet checks are enough.

We scanned the project with the following tools:

- Static analyzer Slither;
- Our plugin Slitherin with an extended set of rules;
- Semgrep rules for smart contracts. We also sent the results to the developers in the text file.

We ran tests and calculated the code coverage.

We combined in a private report all the verified issues we found during the manual audit or discovered by automated tools.

We made the recheck on June 6, 2024. We checked whether the previous issues were fixed. Also, we re-ran the tests and re-calculated the code coverage. Finally, we updated the report.

Manual analysis

The contracts were completely manually analyzed, their logic was checked. Besides, the results of the automated analysis were manually verified. All the confirmed issues are described below.

Critical issues

Critical issues seriously endanger project security. They can lead to loss of funds or other catastrophic consequences. The contracts should not be deployed before these issues are fixed.

The audit showed no critical issues.

Medium severity issues

Medium severity issues can influence project operation in the current implementation. Bugs, loss of potential income, and other non-critical failures fall into this category, as well as potential problems related to incorrect system management. We highly recommend addressing them.

M01. **Project roles** (commented)

In the current implementation, the system depends heavily on the following admin roles:

- The vault or controller owner can change the connector's status (`validConnectors`) in the `Base.updateConnectorStatus` function. The valid connectors can call the `receiveInbound` in the **Vault** or **Controller** contracts with changed data, for instance, an address of the receiver;
- The owner can change a hook address in the **Vault** or **Controller** contracts and avoid additional Kinto checks;
- The `LIMIT_UPDATER_ROLE` can change the bridge limits in the **KintoHook** or **SenderHook** contracts;
- The `owner` can change connector pool Ids and locked amounts of a pool through the `ConnectorPoolPlugin.updateConnectorPoolId` and `ConnectorPoolPlugin.updatePoolLockedAmounts` contracts;
- The `RESCUE_ROLE` can withdraw any token or ETH from the **Vault** or **Controller**. They can also withdraw any token from the hook contracts, but it is assumed that they do not store tokens;
- The `owner` can grant any new role.

> *Comment from the developers: We would not say this is an issue but more a design decision that the Socket team has made when building their contracts. We are aware of the power of each role and of the owner. This is why we have transferred ownership to a Safe wallet for all these contracts.*

M02. **Overpowered role** (fixed)

In the current implementation, funds can be sent to the **BridgerL2** contract (see L01 issue). To receive tokens from this contract, a Kinto wallet should claim them. However, this is possible only if the owner of the **BridgerL2** contract calls the `writeL2Deposit` function and writes users' balances to the storage. For example, this case can be used to withdraw tokens without KYC.

The issue has been fixed and is not present in the latest version of the code.

Low severity issues

Low severity issues do not directly affect project operation. However, they might lead to various problems in future versions of the code. We recommend fixing them or explaining why the team has chosen a particular option.

L01. Misleading code (fixed)

According to the `KintoHook.dstPreHookCall` function, after bridging to Kinto, tokens can be sent to a Kinto wallet or to the **BridgerL2** contract. However, the developers said they no longer send funds to the **BridgerL2** contract.

Consider removing this code or changing it and NatSpecs to an up-to-date version.

The issue has been fixed and is not present in the latest version of the code.

This analysis was performed by [Pessimistic](https://pessimistic.io):

Daria Korepanova, Senior Security Engineer
Oleg Bobrov, Security Engineer
Konstantin Zherebtsov, Business Development Lead
Irina Vikhareva, Project Manager
Alexander Seleznev, CEO

June 6, 2024