Hipo Finance

Audit Report

Mon Nov 27 2023

contact@scalebit.xyz https://twitter.com/scalebit_

ScaleBit

Hipo Finance Audit Report

1 Executive Summary

1.1 Project Information

Description	A decentralized open-source liquid staking protocol on the TON blockchain
Type	DeFi
Auditors	ScaleBit
Timeline	Wed Oct 11 2023 – Thu Nov 02 2023
Languages	Func
Platform	Ton
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/HipoFinance/contract
Commits	277944e3c2216ce1ab193ebd5c091414ce0b7a8c

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
STD	contracts/imports/stdlib.fc	1b3c9e98489d61be2c6c99b7c480d2181a50affd
UTI	contracts/imports/utils.fc	a228c4e6af68e7123d25b30fc475dd03b4318f86
CON	contracts/imports/constants.fc	df212a619bcfa4944ff68dae494e5b154c158f26
WAL	contracts/wallet.fc	a41aca0d126b0bb1620c923414535be2305c453c
TRE	contracts/treasury.fc	2333feaad0f8e653bb7da0d69aca7a1fe5668d5e
LOA	contracts/loan.fc	e35c5fc8453a6f1d894d136405296faed4d5fdda

1.3 Issue Statistic

Item	Count	Fixed	Acknowledged
Total	4	0	0
Informational	2	0	0
Minor	1	0	0
Medium	0	0	0
Major	1	0	0
Critical	0	0	0

2 Summary

This report has been commissioned by Hipo Finance to identify any potential issues and vulnerabilities in the source code of the Hipo Finance smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 4 issues of varying severity, listed below.

ID	Title	Severity	Status
TRE-1	Centralization Risk	Major	Pending
TRE-2	Insufficient Documentation Comments	Informational	Pending
TRE-3	Code Optimization	Informational	Pending
UTI-1	Redundant Specifier	Minor	Pending

3 Participant Process

Here are the relevant actors with their respective abilities within the Hipo Finance Smart Contract:

Staker

- Staker can deposit `TON` and receive `hTON` through `deposit_coins()`.
- Staker can stake their `TON` into the validation node through `stake_coins()`.
- Staker can send their `hTON` to anyone through `send_tokens()`.
- Staker can burn their `hTON` and withdraw `TON` through `unstake_tokens()`.
- Staker can withdraw their `TON` from the validation node through `withdraw_tokens()`.

Validator

- Validator can requests a loan through `request_loan()`.
- Validator can participate in election through `participate_in_election()`.
- Validator can prompt the treasury that a round has passed through `vset_changed()`.
- Validator can ask the treasury to retrieve the loaned amount plus reward for each given loan and distribute rewards through `finish_participation()`.

4 Findings

TRE-1 Centralization Risk

Severity: Major

Status: Pending

Code Location:

contracts/treasury.fc

Descriptions:

Most resource storage and modification is currently under the control of a single project administrator account and this creates a significant centralization risk which could have negative consequences. Such as the `halter` and `governor` can set the status of the contract through `set_stopped()` and withdraw surplus through `withdraw_surplus()`, and set rounds imbalance through `set_rounds_imbalance()` and so on.

Suggestion:

It is recommended to take some measures to mitigate centralization risk, such as implementing an interface that allows for multiple administrators or a multi-signature account to manage resources.

TRE-2 Insufficient Documentation Comments

Severity: Informational

Status: Pending

Code Location:

contracts/treasury.fc

Descriptions:

Some functions in the smart contract lack documentation comments. The absence of documentation comments in a smart contract may have a negative impact on code readability and maintainability. Documentation comments are an essential development practice that can assist other developers in understanding the intent, functionality, and usage of the code.

Suggestion:

It is recommended to add documentation comments for some important functions in the smart contract.

TRE-3 Code Optimization

Severity: Informational

Status: Pending

Code Location:

contracts/treasury.fc#183,184

Descriptions:

In the `deposit_coins` function, the variable `storage_fee` is defined as 0, but in the subsequent code this variable is involved in the calculation, and we think it is unnecessary.

Suggestion:

It is recommended to remove meaningless variable definitions and calculations to save gas consumption.

UTI-1 Redundant Specifier

Severity: Minor

Status: Pending

Code Location:

contracts/imports/utils.fc#44,52,62,68,79,103,117,252,261,290,326,352,357,385,417,422,427,451,510

contracts/treasury.fc#418

Descriptions:

The impure specifier means that the function can have some side effects which can't be ignored. For example, modify the contract storage, send messages, or throw an exception. But functions of the return value type in `utils.fc` and `treasury.fc` do not have any of the above occurring, so it is not necessary to add impure specifiers to the function.

Suggestion:

It is recommended to remove impure specifier from these return value type functions.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.