INTENTABLE

Powered by KIROBO

Intentable's Smart Transaction technology powered by the Kirobo FCT Platform

FCT Smart Wallet (FCT Runner) Security Analysis

by Pessimistic

This report is public

October 2, 2024

Abstract

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

Disclaimer

The audit does not give any warranties on the security of the code. A single audit cannot be considered enough. We always recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. Besides, a security audit is not investment advice.

Summary

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

The report has designations like C01, M05, L10, N08. The letter represents severity, and the number represents the issue number.

This is a condensed version of the report; a more detailed one can be found at the following links:

- Smart Wallet report. The fixed issues: C01, M01, M02, M03, M04, L01-L07, N02.
- Report #1. The fixed issues:
    - M01, L03, L06, L07, which were relevant to the current submodule;
    - M04 issue, which was addressed and was relevant to the whole project.
- Report #2. The fixed issues:
    - M01, L16, N02, which were relevant to the current submodule;
    - L04, which was relevant for the whole project.
- Report #3. The fixed issues:
    - L01, L02, N01, which were relevant to the current submodule;
    - M01, which was addressed and was relevant to the whole project.

The report includes L01 (L08 in the Smart Wallet report) and the following notes with the `commented` status: N01 and N02 (N05 and N06 in the Report #2), and N03 (N02 in the Report #3). All the tests passed. The code coverage is sufficient.

After the last recheck, which was fully described in Report #3 (see the Codebase update #8) and copied to the Project description, the codebase was updated again. The number of tests and the code coverage increased. We did not find any new issues.

During the small update, the developers fixed L01, which was copied from the Report #3, on commit 78c90bb9bf2b38c50b03ef3d89f0d125e2bc3d6a.

According to our recommendations, the developers split the long function in the **FCT_BatchMultiSig** contract, improving the code readability. However, it is still important to note that the project and its architecture are complex, though we realize that it is difficult or impossible to implement simply.

It is crucial to study the four reports above to get a full picture of the security of smart contracts.

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 Intentable project on a private GitHub repository, commit cf31b27dd9e44e016ced8e3060a487e7e1e2d5cc.

It is a condensed version of the report. More detailed information about codebase updates can be found at the following links:

- Smart Wallet report;
- Report #1;
- Report #2;
- Report #3.

The scope included:

- Activator.sol;
- ActivatorBase.sol;
- Backupable.sol;
- Factory.sol;
- FactoryProxy.sol;
- Heritable.sol;
- Proxy.sol;
- ProxyLatest.sol;
- SmartOracle.sol;
- SmartWallet.sol;
- SmartWalletCore.sol;
- Storage.sol;
- StorageBase.sol;
- Interface.sol;
- IBackupable.sol;
- ICreator.sol;
- IFactory.sol;
- IHeritable.sol;
- IOracle.sol;
- ITokenEconomy.sol;
- IWallet.sol.

The documentation for the project included https://kirobo.gitbook.io/fct-developers-guide/.

The total LOC of the audited scope is 2570.

All 592 tests passed. The code coverage of the project was 84.64%.

Codebase update #1

For the recheck, we were provided with Intentable project on a private GitHub repository, commit 9ba7299653676b06cadfcbfac2d1f9fff66d333c.

The developers provided the test results and the code coverage. All 630 tests passed, the code coverage of the scope was 96.75%, and the code coverage of the FCT platforms was 92.93%.

Audit process

We started to check the FCT platform around two years ago and created four extensive reports. The developers asked us to split these reports into seven submodules. Each submodule has:

- The last common commit for all reports;
- The same results of the tests and the code coverage for the whole project;
- The individual scope;
- Links to the corresponding previous reports;
- The list of fixed issues, e.g., M01, M04, L03 (see the description in the Summary), etc.;
- The descriptions of unfixed or commented issues that are still actual;
- The findings relevant to the whole project.

Issue descriptions copied from previous reports can have different contract names and lines, as they were checked again and updated due to the last commit in the Project description.

We started auditing the FCT platforms in 2022. During this time, we made three FCT reports and one report for the Smart Wallet part.

Each report is a continuation of the previous one. We made the split in the process to avoid one infinitely extensive report. The chronology of the reports:

- Smart Wallet report - finished on August 15, 2022;
- Report #1 - finished on November 17, 2022;
- Report #2 - finished on July 26, 2023, and last updated on January 16, 2024;
- Report #3 - finished on June 26, 2024.

See the previous reports for a more detailed description of the issues and process. The following rechecks related to a specific module will only appear in the corresponding report.

We made recheck #1 for this scope on June 26-27, 2024. It was the first recheck in this report but not the first one in the entire audit process (see the previous reports). The developers provided the test results and the code coverage, as we could not run them. This update included code refactoring and fixes for adding the new functionality. We did not discover new issues.

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.

The audit showed no issues of medium severity.

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. Unused variable (fixed)

The `MAX_FEES` value is never used in the **Activator** contract.

The issue has been fixed on commit 78c90bb9bf2b38c50b03ef3d89f0d125e2bc3d6a.

Notes

N01. Unsafe delegatecall using (commented)

The using of `delegatecall` is potentially unsafe since the `target` contract may call `selfdestruct`.

In the `fctCall` function of the **RecoveryWallet** contract, `delegatecall` can be called by `ENS_LIBRARY` type target only. And targets of this type can be added through `FCT_ENS.setLocalEns` function by `LOCAL_ENS_ROLE`.

So, this `delegatecall` is protected by the project role. However, if admin's private keys become compromised, any targets can be added.

> *Comment from the developers:* *In the function* `ensToAddress` *the second param can be an address or 0, for extra security the user can choose to add the lib address, and so, if the ENS of the lib was changed the FCT will not work.*

> *Pessimistic's comment:*
> *During the rechecks, the **RecoveryWallet** contract was renamed **SmartWallet**.*

N02. Overpowered roles (commented)

The project has the `LOCAL_ENS_ROLE` role in the **FCT_ENS** contract, which can add new targets. This role has an impact on the `delegatecall` in the **RecoveryWallet** contract (see the Unsafe delegatecall using issue).

Thus, if the admin's private keys are compromised, there may be scenarios that could lead to undesirable consequences for the project and its users.

> *Comment from the developers:* *Now we are using the same key for all, but in production we plan to add multiSig for that.*

> *Pessimistic's comment:*
> *During the rechecks, the **RecoveryWallet** contract was renamed **SmartWallet**.*

N03. The creator is equal to zero address (commented)

The `creator` method always returns `address(0)` in the **StorageBase** contract. It has the comment "needed to pass compilation". However, this method is used in the contracts that are inherited from the **StorageBase** contract. Consider fixing it or making it `virtual` to allow overriding.

> *The `StorageBase.creator` reverts now. We have not checked the lll code of the minimal proxy (it is out of scope).*

> *<u>Comment from the developers:</u>*
>
> *`creator` function always returns the factory address. When the factory creates a new wallet (**SmartWallet** for example) it creates a special proxy, written in lll, that hijacks the creator function call and always returns the factory address.*
>
> *This makes the **SmartWallet** immune to bad updates, such as implementing creator function or overriding owner address or proxy's target data storage.*
>
> *The factory handles owner and version changes and has functions to recover the owner or version in such cases. The creator function is needed to pass compilation, so it can be used on from a solidity code (although it is hijacked by the lll code).*
>
> *It was updated to always revert instead of returning `address(0)` in order to make sure it is never being called directly.*

This analysis was performed by Pessimistic:

Daria Korepanova, Senior Security Engineer
Yhtyyar Sahatov, Security Engineer
Evgeny Bokarev, Junior Security Engineer
Konstantin Zherebtsov, Business Development Lead
Irina Vikhareva, Project Manager
Alexander Seleznev, CEO

October 2, 2024