



Powered by **KIROBO** 

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

# FCT Engines Security Analysis

by Pessimistic

This report is public

October 2, 2024

Abstract .....	2
Disclaimer .....	2
Summary .....	2
General recommendations .....	3
Project overview .....	4
Project description .....	4
Codebase update #1 .....	4
Audit process .....	5
Manual analysis .....	6
Critical issues .....	6
Medium severity issues .....	6
Low severity issues .....	6
Notes .....	7
N01. Weak multisig (commented) .....	7
N02. Very long functions (commented) .....	7

# 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 [FCT Engines submodule of 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:

- [Report #1](#). The fixed issues:
  - M01, 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:
  - C02, M05, M09, M14, M15, L07, L08, N02, which were relevant to the current submodule;
  - L04, which was relevant for the whole project.
- [Report #3](#). The fixed issues:
  - M02, N03, which were relevant to the current submodule;
  - M01, which was addressed and was relevant to the whole project.

The report includes the following notes with the `commented` status: N01 (N04 in the [Report #1](#)) and N02 (N04 in the [Report #2](#)). 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.

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 three 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:

- [Report #1](#);
- [Report #2](#);
- [Report #3](#).

The scope included:

- FCT\_BatchMultiSig.sol;
- FCT\_Constants.sol;
- FCT\_Helpers.sol.

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

The total LOC of the audited scope is 1666.

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 [9ba7299653676b06cadfcfbfac2d1f9fff66d333c](#).

The developers provided the test results and the code coverage. All 630 tests passed, the code coverage of the scope was 94.74%, 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.

**The audit showed no issues of low severity.**

## Notes

### N01. Weak multisig (commented)

The optional external signers feature of the FCT implementation in the **FCT\_BatchMultiSig** contract does not include signer uniqueness at lines 343–363. So the user can use multiple signatures from one signer to pass the multisig check.

This behavior is an intentional design choice to save gas.

*Comment from the developers: wanted behavior. See [documentation](#).*

### N02. Very long functions (commented)

Consider splitting the `abiToEIP712` (at lines 791-930) and `batchMultiSigCall` (at lines 243-724) functions of the **FCT\_BatchMultiSig** contract into several smaller functions to improve code structure and readability.

*Comment from the developers: The method is long due to technical issues of "stack too deep" and gas optimizations. Even so, it was built with independent parts that could be considered as functions. Each part was documented with the `@notice` comment.*

*The developers split the `batchMultiSigCall` function into the smaller ones.*



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