1inch Audit

Limit Order Protocol

August 2022

By CoinFabrik

Introduction

CoinFabrik was asked to audit the contracts for 1inch's Limit Order Protocol project. First we will provide a summary of our discoveries, and then we will show the details of our findings.

Scope

The audited files are from the git repository located at `github.com/1inch/limit-order-protocol`. The audit is based on the commit `3f1a4357b1965b8084c8d8a827031f6b05bd8c7f`.

The audited files are:

- `LimitOrderProtocol.sol`: Contract which inherits from `OrderMixin` and `OrderRFQMixin` contract in order to provide a single interface for order filling.
- `OrderLib.sol`: Library which defines the Order struct and its methods.
- `OrderMixin.sol`: Function implementations for filling orders from the OrderLib library.
- `OrderRFQLib.sol`: Library which defines the `OrderRFQ` struct and its methods.
- `OrderRFQMixin.sol`: Function implementations for filling orders from the `OrderRFQLib` library.
- `libraries/ArgumentsDecoder.sol`: Library for argument deserialization.
- `libraries/Errors.sol`: InvalidaMsgValue error definition.
- `libraries/Callib.sol`: Library which implements a function for making static calls expecting a uint256 as output.
- `helpers/AmountCalculator.sol`: Helper contract to calculate swap taker/maker amount.
- `helpers/NonceManager.sol`: Helper for managing nonce of each `tx-sender`.
- `helpers/PredicateHelper.sol`: Helper contract for executing boolean functions on arbitrary target call results.

The scope of the audit is limited to those files. No other files in this repository were audited. Its dependencies are assumed to work according to their documentation. Also, no tests were reviewed for this audit.

Analyses

Without being limited to them, the audit process included the following analyses:

- Arithmetic errors
- Outdated version of Solidity compiler
- Race conditions
- Reentrancy attacks
- Misuse of block timestamps
- Denial of service attacks
- Excessive gas usage
- Missing or misused function qualifiers
- Needlessly complex code and contract interactions
- Poor or nonexistent error handling
- Insufficient validation of the input parameters
- Incorrect handling of cryptographic signatures
- Centralization and upgradeability

Summary of Findings

We found a minor issue. Also, an enhancement was proposed.

The issue was acknowledged.

Security Issues

ID	Title	Severity	Status
MI-01	ERC20 Transfer Output Unchecked	Minor	Acknowledged

Security Issues Found

Severity Classification

Security risks are classified as follows:

- **Critical:** These are issues that we manage to exploit. They compromise the system seriously. They must be fixed **immediately**.

- **Medium:** These are potentially exploitable issues. Even though we did not manage to exploit them or their impact is not clear, they might represent a security risk in the near future. We suggest fixing them **as soon as possible**.
- **Minor:** These issues represent problems that are relatively small or difficult to take advantage of, but can be exploited in combination with other issues. These kinds of issues do not block deployments in production environments. They should be taken into account and be fixed **when possible**.

Issues Status

An issue detected by this audit can have four distinct statuses:

- **Unresolved**: The issue has not been resolved.
- **Acknowledged**: The issue remains in the code, but is a result of an intentional decision.
- **Resolved**: Adjusted program implementation to eliminate the risk.
- **Partially resolved**: Adjusted program implementation to eliminate part of the risk. The other part remains in the code, but is a result of an intentional decision.
- **Mitigated**: Implemented actions to minimize the impact or likelihood of the risk

Critical Severity Issues

No issues found.

Medium Severity Issues

No issues found.

Minor Severity Issues

MI-01 ERC20 Transfer Output Unchecked

Location:

- `contracts/OrderRFQMixin.sol:230,242`
- `contracts/OrderMixin.sol:272`

ERC20 `transfer()` and `transferFrom()` methods return a boolean value. Most of the implementations always return true and revert if the transfer fails. However, there are some implementations which fail silently and return false.

In this case, WETH transfers results are not checked. This would not be an issue unless an implementation different from the one deployed at `0xC02aaA39b223FE8D0A0e5C4F27eAD9083C756Cc2` on Ethereum mainnet is used by the protocol. However, since any address can be provided, a silently-failing implementation could be assigned.

Recommendation

The address provided for the _WETH variable in both contracts should be verified and documented.

However, adding checks for transfer results is the safest approach, considering the previous recommendation requires deployer discretion.

Status

Acknowledged. As the development team knows the token implementation in advance, they intentionally skipped the output validation for WETH, as there is no case when it will return false instead of reverting.

Enhancements

These items do not represent a security risk. They are best practices that we suggest implementing.

Table

ID	Title	Status
EN-01	Remove Unnecessary Inheritance	Not implemented

Details

EN-01 Remove Unnecessary Inheritance

Location:

- `contracts/OrderMixin.sol:22`

`OrderMixin` contract inherits from NonceManager, but none of its functions are used by the contract.

Recommendation

Remove `NonceManager` from the inheritance.

Status
Not implemented.

Other Considerations

The considerations stated in this section are not right or wrong. We do not suggest any action to fix them. But we consider that they may be of interest for other stakeholders of the project, including users of the audited contracts, owners or project investors.

Centralization

There are no centralized mechanisms implemented in the contracts included in the scope.

Upgrades

The protocol implemented is not upgradeable, it is immutable.

Changelog

- 2022-08-23 – Initial report based on commit `3f1a4357b1965b8084c8d8a827031f6b05bd8c7f`.
- 2022-08-30 – Final report based on development team's feedback.

Disclaimer: This audit report is not a security warranty, investment advice, or an approval of the 1inch project since CoinFabrik has not reviewed its platform. Moreover, it does not provide a smart contract code faultlessness guarantee.