decurity

Smart Contract Security Audit Report

1inch

1. Contents

2. General Information

This report contains information about the results of the security audit of the 1inch (hereafter referred to as “Customer”) Fusion smart contracts, conducted by Decurity in the period from 03/18/2024 to 03/22/2024.

2.1. Introduction

Tasks solved during the work are:

- Review the protocol design and the usage of 3rd party dependencies,
- Audit the contracts implementation,
- Develop the recommendations and suggestions to improve the security of the contracts.

2.2. Scope of Work

The audit scope included the following repositories:

- https://github.com/1inch/limit-order-protocol/pull/306/files (commit 2f87361ed9ce77435998a83064f612b3baeaa058, re-test 5e0be56b9e49caad7a1d0121eb9bfb8fd45830bb),
- https://github.com/1inch/limit-order-settlement/compare/2.0.0-prerelease-2...master (commit d6f86ed124b38e2af114cd821a6bf3beed94193b).

2.3. Threat Model

The assessment presumes the actions of an intruder who might have the capabilities of any role (an external user, token owner, token service owner, or a contract). The risks of centralization were not taken into account at the Customer's request.

The main possible threat actors are:

- Users (takers and makers),
- Protocol owners and relayers,

- Token contracts, etc.

2.4. Weakness Scoring

An expert evaluation scores the findings in this report, and the impact of each vulnerability is calculated based on its ease of exploitation (based on the industry practice and our experience) and severity (for the considered threats).

2.5. Disclaimer

Due to the intrinsic nature of the software and vulnerabilities and the changing threat landscape, it cannot be generally guaranteed that a certain security property of a program holds.

Therefore, this report is provided “as is” and is not a guarantee that the analyzed system does not contain any other security weaknesses or vulnerabilities. Furthermore, this report is not an endorsement of the Customer’s project, nor is it an investment advice.

That being said, Decurity exercises the best effort to perform its contractual obligations and follow the industry methodologies to discover as many weaknesses as possible and maximize the audit coverage using limited resources.

3. Summary

As a result of this work, we haven't discovered any exploitable security issues.

3.1. Suggestions

The table below contains the discovered issues, their risk level, and their status as of April 02, 2024.

Table. Discovered weaknesses

Issue	Contract	Risk Level	Status
Tokens can be sweeped from FeeTaker	limit-order-protocol/contracts/extensions/FeeTaker.sol	**Low**	Fixed
Calculations can be done in unchecked block	limit-order-settlement/contracts/extensions/ExtensionLib.sol	**Info**	Acknowledged

4. General Recommendations

This section contains general recommendations on how to improve the overall security level.

The Findings section contains technical recommendations for each discovered issue.

4.1. Security Process Improvement

The following is a brief long-term action plan to mitigate further weaknesses and bring the product security to a higher level:

- Keep the whitepaper and documentation updated to make it consistent with the implementation and the intended use cases of the system,
- Perform regular audits for all the new contracts and updates,
- Ensure the secure off-chain storage and processing of the credentials (e.g. the privileged private keys),
- Launch a public bug bounty campaign for the contracts.

5. Findings

5.1. Tokens can be sweeped from FeeTaker

Risk Level: Low

Status: Fixed in the commit d118e534.

Contracts:

- limit-order-protocol/contracts/extensions/FeeTaker.sol

Description:

A malicious resolver may transfer out any tokens that are owned by the `FeeTaker` contract. Normally, the `FeeTaker` contract is not expected to have tokens or tokens allowances, however there is still a possibility, e.g. airdrops or accidental transfers.

Remediation:

Consider adding the `onlyLimitOrderProtocol` modifier to block unauthorized calls and implement a rescue admin function.

5.2. Calculations can be done in `unchecked` block

Risk Level: Info

Status: Acknowledged

Contracts:

- limit-order-settlement/contracts/extensions/ExtensionLib.sol

Description:

In `ExtensionLib` contract it's not necessary to ensure that `extraData.length` is greater or equal to 1, so unchecked block can be used there

```
limit-order-settlement/contracts/extensions/ExtensionLib.sol:
  19:     function resolverFeeEnabled(bytes calldata extraData) internal pure
returns (bool) {
  20:         return extraData[extraData.length - 1] & _RESOLVER_FEE_FLAG ==
_RESOLVER_FEE_FLAG;
```

```
    }
    function integratorFeeEnabled(bytes calldata extraData) internal pure returns (bool) {
        return extraData[extraData.length - 1] & _INTEGRATOR_FEE_FLAG == _INTEGRATOR_FEE_FLAG;
    }
    function resolversCount(bytes calldata extraData) internal pure returns (uint256) {
        return uint8(extraData[extraData.length - 1]) >> _WHITELIST_SHIFT;
    }
```

Remediation:

Consider adding an unchecked block for gas savings.

6. Appendix

6.1. About us

The Decurity team consists of experienced hackers who have been doing application security assessments and penetration testing for over a decade.

During the recent years, we've gained expertise in the blockchain field and have conducted numerous audits for both centralized and decentralized projects: exchanges, protocols, and blockchain nodes.

Our efforts have helped to protect hundreds of millions of dollars and make web3 a safer place.