Smart Contract Security Audit Report

1inch Crosschain Update Audit

1. Contents

2. General Information

This report contains information about the results of the security audit of the 1inch (hereafter referred to as "Customer") smart contracts, conducted by Decurity in the period from 2025-08-06 to 2025-08-11.

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 contracts in the following repository: https://github.com/1inch/cross-chain-swap. Initial review was done for the commit d0a59ab2c4b6be5c9769d5775769681873fcf162.

2.3. Threat Model

The assessment presumes actions of an intruder who might have capabilities of any role (an external user, token owner, token service owner, a contract). The centralization risks have not been considered upon the request of the Customer.

The main possible threat actors are:

- Maker,
- Taker,
- Protocol Owner.

2.4. Weakness Scoring

An expert evaluation scores the findings in this report, an 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 best effort to perform their contractual obligations and follow the industry methodologies to discover as many weaknesses as possible and maximize the audit coverage using the limited resources.

3. Summary

As a result of this work, we have discovered a single informational issue.

3.1. Suggestions

The table below contains the discovered issues, their risk level, and their status as of August 23, 2025.

Table. Discovered weaknesses

Issue	Contract	Risk Level	Status
Resolvers may abuse rescue function	contracts/EscrowSrc.sol, contracts/EscrowDst.sol	Info	Acknowledged

4. General Recommendations

This section contains general recommendations on how to improve 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. Resolvers may abuse rescue function

Risk Level: Info

Status: Acknowledged

Client's comment: On our side, the following off-chain operational measures have been implemented:

- All resolvers undergo KYC/KYB and operate on an allowlist.
- Access to AccessTokens/fees is granted only to approved resolvers.
- Any dishonest behavior (including attempts to profit from another party's mistake) leads to immediate access revocation and blacklisting.

Economics and abuse scenarios:

- A resolver has no rational incentive to deposit more than the order amount and deposit: it brings no benefit and only increases their own risk.
- Attempts by third-party resolvers to "monetize" someone else's mistake without collusion with the maker are limited (a multiple excess of deposit and tokens, assuming the resolver "makes mistakes" constantly) and are practically economically meaningless (only within the amount of the deposit); any coordinated actions with the maker are classified as abuse and trigger enforcement measures on our part as the regulator.

Contracts:

- contracts/EscrowSrc.sol
- contracts/EscrowDst.sol

Description:

If enough funds (i.e., the required tokens and ETH) are accidentally sent to either the source or destination escrow, resolvers may, for example, invoke the withdraw function twice to retrieve those funds. However, these funds are not part of an order and should be recovered via the rescue functionality

after the RESCUE_DELAY. If the public withdrawal/cancellation time arrives, this can be done by any resolver, not just the original taker, as intended by the rescue function.

Remediation:

Consider disallowing invocation of the escrow functions besides rescue after the order has been processed.

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.