# 1INCH AGGREGATION ROUTER V5 SECURITY AUDIT REPORT

November 10, 2022

MixBytes()

# TABLE OF CONTENTS

# 1. INTRODUCTION

## 1.1 Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only. The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of Customer. If you are not the intended recipient(s) of this document, please note that any disclosure, copying or dissemination of its content is strictly forbidden.

## 1.2 Security Assessment Methodology

A group of auditors are involved in the work on the audit. The security engineers check the provided source code independently of each other in accordance with the methodology described below:

### 1. Project architecture review:

- Project documentation review.
- General code review.
- Reverse research and study of the project architecture on the source code alone.

Stage goals
- Build an independent view of the project's architecture.
- Identifying logical flaws.

### 2. Checking the code in accordance with the vulnerabilities checklist:

- Manual code check for vulnerabilities listed on the Contractor's internal checklist. The Contractor's checklist is constantly updated based on the analysis of hacks, research, and audit of the cients' codes.
- Code check with the use of static analyzers (i.e Slither, Mythril, etc).

Stage goal

Eliminate typical vulnerabilities (e.g. reentrancy, gas limit, flash loan attacks etc.).

### 3. Checking the code for compliance with the desired security model:

- Detailed study of the project documentation.
- Examination of contracts tests.
- Examination of comments in code.
- Comparison of the desired model obtained during the study with the reversed view obtained during the blind audit.
- Exploits PoC development with the use of such programs as Brownie and Hardhat.

Stage goal

Detect inconsistencies with the desired model.

### 4. Consolidation of the auditors' interim reports into one:

- Cross check: each auditor reviews the reports of the others.
- Discussion of the issues found by the auditors.
- Issuance of an interim audit report.

Stage goals

- Double-check all the found issues to make sure they are relevant and the determined threat level is correct.
- Provide the Customer with an interim report.

### 5. Bug fixing & re-audit:

- The Customer either fixes the issues or provides comments on the issues found by the auditors. Feedback from the Customer must be received on every issue/bug so that the Contractor can assign them a status (either "fixed" or "acknowledged").
- Upon completion of the bug fixing, the auditors double-check each fix and assign it a specific status, providing a proof link to the fix.
- A re-audited report is issued.

Stage goals

- Verify the fixed code version with all the recommendations and its statuses.
- Provide the Customer with a re-audited report.

### 6. Final code verification and issuance of a public audit report:

- The Customer deploys the re-audited source code on the mainnet.
- The Contractor verifies the deployed code with the re-audited version and checks them for compliance.
- If the versions of the code match, the Contractor issues a public audit report.

Stage goals

- Verify the fixed code version with all the recommendations and its statuses.
- Provide the Customer with a re-audited report.

## Finding Severity breakdown

All vulnerabilities discovered during the audit are classified based on their potential severity and have the following classification:

| Severity | Description |
|---|---|
| Critical | Bugs leading to assets theft, fund access locking, or any other loss funds to be transferred to any party. |
| High | Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement. |
| Medium | Bugs that can break the intended contract logic or expose it to DoS attacks, but do not cause direct loss funds. |
| Low | Other non-essential issues and recommendations reported to/ acknowledged by the team. |

Based on the feedback received from the Customer regarding the list of findings discovered by the Contractor, they are assigned the following statuses:

| Status | Description |
| --- | --- |
| Fixed | Recommended fixes have been made to the project code and no longer affect its security. |
| Acknowledged | The Customer is aware of the finding. Recommendations for the finding are planned to be resolved in the future. |

## 1.3 Project Overview

1Inch is a DeFi aggregator and a decentralized exchange with smart routing. The core protocol connects a large number of decentralized and centralized platforms in order to minimize price slippage and find the optimal trade for the users.

## 1.4 Project Dashboard

### Project Summary

| Title | Description |
| --- | --- |
| Client | 1Inch |
| Project name | Aggregation Router V5 |
| Timeline | 18-07-2022 - 27-09-2022 |
| Number of Auditors | 4 |

## Project Log

### Main repository (1inch-contract)

| Date | Commit Hash | Note |
|---|---|---|
| 18-07-2022 | 8aa5ec4b4871b1d63bb045ddb78aaf7c5dc84dfa | Initial commit |
| 26-07-2022 | 5e682c5227d9428b395041c93db0fb657b741afc | Final commit |
| 22-09-2022 | bc00c75f2c99d62e1a206aa2f81c408caba4b370 | Pre-release commit |

### 1st dependent repository (limit-order-protocol)

| Date | Commit Hash | Note |
|---|---|---|
| 18-07-2022 | d8437885744543e3f057e84e1b0a05c4c211c553 | Initial commit |
| 26-07-2022 | fdc93648cb665a3bfac469584fd6e1d1d8f07d05 | Final commit |
| 21-09-2022 | 171c5d7bbb280d9f754404828051f2a47fb726df | Pre-release commit |

### 2nd dependent repository (solidity-utils)

| Date | Commit Hash | Note |
|---|---|---|
| 18-07-2022 | eec6b523860af5215a8dd196fe3aff3a4d252fc9 | Initial commit |
| 26-07-2022 | 71d6fe457cc420eeba55f612e902e66920faa64c | Final commit |
| 20-09-2022 | c35dc32fd91ee01f961df13ab7c30faf40be8b89 | Pre-release commit |

## Project Scope

The audit covered the following files:

| File name | Link |
| --- | --- |
| AggregationRouterV5.sol | AggregationRouterV5.sol |
| ClipperRouter.sol | ClipperRouter.sol |
| GenericRouter.sol | GenericRouter.sol |
| UnoswapRouter.sol | UnoswapRouter.sol |
| UnoswapV3Router.sol | UnoswapV3Router.sol |
| IClipperExchangeInterface.sol | IClipperExchangeInterface.sol |
| IAggregationExecutor.sol | IAggregationExecutor.sol |
| IUniswapV3Pool.sol | IUniswapV3Pool.sol |
| IUniswapV3SwapCallback.sol | IUniswapV3SwapCallback.sol |
| Errors.sol | Errors.sol |
| OrderMixin.sol | OrderMixin.sol |
| OrderRFQMixin.sol | OrderRFQMixin.sol |
| OrderLib.sol | OrderLib.sol |
| OrderRFQLib.sol | OrderRFQLib.sol |
| AmountCalculator.sol | AmountCalculator.sol |
| NonceManager.sol | NonceManager.sol |
| PredicateHelper.sol | PredicateHelper.sol |
| IOrderMixin.sol | IOrderMixin.sol |

| File name | Link |
| --- | --- |
| NotificationReceiver.sol | NotificationReceiver.sol |
| ArgumentsDecoder.sol | ArgumentsDecoder.sol |
| Callib.sol | Callib.sol |
| Errors.sol | Errors.sol |
| EthReceiver.sol | EthReceiver.sol |
| StringUtil.sol | StringUtil.sol |
| UniERC20.sol | UniERC20.sol |
| SafeERC20.sol | SafeERC20.sol |
| ECDSA.sol | ECDSA.sol |
| RevertReasonForwarder.sol | RevertReasonForwarder.sol |
| IWETH.sol | IWETH.sol |
| IDaiLikePermit.sol | IDaiLikePermit.sol |

## 1.5 Summary of findings

| Severity | # of Findings |
| --- | --- |
| Critical | 0 |
| High | 0 |
| Medium | 0 |
| Low | 4 |

| ID | Name | Severity | Status |
|---|---|---|---|
| L-1 | Extra inheritance | Low | Fixed |
| L-2 | Spelling mistakes | Low | Fixed |
| L-3 | Null check | Low | Fixed |
| L-4 | Reduce gas cost | Low | Fixed |

## 1.6 Conclusion

During the audit process, 4 low severity issues have been found and fixed by the Client.

| File name | Contract deployed on mainnet |
|---|---|
| AggregationRouterV5 | 0x1111111254EEB25477B68fb85Ed929f73A960582 |

# 2. FINDINGS REPORT

## 2.1 Critical

Not Found

## 2.2 High

Not Found

## 2.3 Medium

Not Found

## 2.4 Low

| | |
|---|---|
| L-1 | Extra inheritance |
| **File** | |
| **Severity** | Low |
| **Status** | Fixed in 340fff3d |

**Description**

The `OrderMixin` contract inherits the `NonceManager` contract, and the `PredicateHelper` contract inherits the `NonceManager` contract. You can remove the `NonceManager` contract from the `OrderMixin` inheritance chain because the `OrderMixin` contract inherits `PredicateHelper` and does not use the `NonceManager` contract code.

### Recommendation

We recommend removing the `NonceManager` contract from inheritance for the `OrderMixin` contract.

| L-2 | Spelling mistakes |
| --- | --- |
| **Files** | UniERC20.sol#L62<br>OrderRFQMixin.sol#L98 |
| **Severity** | Low |
| **Status** | Fixed in afdd1394, 2d8931d4 |

### Description

Some texts have spelling mistakes:

1. `SYBMOL` -> `SYMBOL` (bad input param)
   UniERC20.sol#L62
2. `signuture` -> `signature`
   OrderRFQMixin.sol#L98

### Recommendation

We recommend correcting them.

| L-3 | Null check |
| --- | --- |
| **File** | AggregationRouterV5.sol#L22 |
| **Severity** | Low |
| **Status** | Fixed in 5e682c52 |

### Description

Some parameters have no null checks:
AggregationRouterV5.sol#L22

**Recommendation**

We recommend adding a null check for `clipperExchange`.

| L-4 | Reduce gas cost |
| --- | --- |
| **File** | UniERC20.sol#L99 |
| **Severity** | Low |
| **Status** | Fixed in c3b06194 |

**Description**

Reduce gas cost for some functions:
UniERC20.sol#L99

**Recommendation**

We recommend adding `unchecked block` for `len++`.

## 2.5 Appendix

### 1 Monitoring Recommendation

The project contains smart contracts that require active monitoring. For these purposes, it is recommended to proceed with developing new monitoring events based on Forta (https://forta.org) with which you can track the following exemplary incidents:

- Anomalies of AggregationRouter's components;
- Call `destroy()` and other privileged methods;
- Unexpected addresses in routers.

# 3. ABOUT MIXBYTES

MixBytes is a team of blockchain developers, auditors and analysts keen on decentralized systems. We build opensource solutions, smart contracts and blockchain protocols, perform security audits, work on benchmarking and software testing solutions, do research and tech consultancy.

## Contacts


https://github.com/mixbytes/audits_public


https://mixbytes.io/


hello@mixbytes.io


https://twitter.com/mixbytes