MixBytes()

# BarterDAO InchFusion Security Audit Report

SEPTEMBER 05, 2025

# Table of Contents

# 1. Introduction

## 1.1 Disclaimer

The audit makes no statements or warranties regarding the utility, safety, or security of the code, the suitability of the business model, investment advice, endorsement of the platform or its products, the regulatory regime for the business model, or any other claims about the fitness of the contracts for a particular purpose or their bug-free status.

## 1.2 Executive Summary

BarterDAO is a DeFi swap execution protocol that provides a unified interface for executing trades across multiple decentralized exchanges and liquidity sources. The protocol integrates with 1inch Fusion, Paraswap, UniswapX, and various DEX aggregators to offer optimal swap execution with MEV protection.

In this audit, we reviewed the **InchFusionBarterResolver** contract, which serves as a taker interaction resolver for 1inch orders, enabling complex swap operations with fee handling and balance validation.

This interim audit was conducted over 1 day by a team of experienced auditors using manual code review methodology. The audit focused on the core InchFusionBarterResolver contract and its integration with the broader swap execution infrastructure.

The audit examined potential attack vectors, including reentrancy attacks, access control bypasses, token balance manipulation, fee extraction vulnerabilities, MEV exploitation, and integration risks with external protocols. Particular focus was placed on the taker interaction mechanism and the handling of arbitrary executor calls, especially regarding the possibility of a contract-level denial-of-service (DoS).

The code demonstrates good security practices with proper access controls, error handling, and integration patterns.

**Key notes and recommendations:**

1. Slippage protection is implemented on the SwapFacade side.
2. Exact amounts in orders will be calculated off-chain to account for the logic of leaving 1 wei on contracts.
3. Users are expected not to leave funds on InchFusionBarterResolver or give personal permits to InchFusionBarterResolver, as bots would immediately drain them otherwise.
4. `call` to `block.coinbase` can trigger reentrancy. However, negative slippage is not possible because SwapFacade enforces `minReturn` checks. Theft of positive slippage is possible even without reentrancy, since the `block.coinbase` address can already execute MEV against the caller.

# 1.3 Project Overview

## Summary

| Title | Description |
|---|---|
| **Client Name** | BarterDAO |
| **Project Name** | InchFusionBarterResolver |
| **Type** | Solidity |
| **Platform** | EVM |
| **Timeline** | 02.09.2025–03.09.2025 |

## Scope of Audit

| File | Link |
|---|---|
| **contracts/InchFusionBarterResolver.sol** | InchFusionBarterResolver.sol |

## Versions Log

| Date | Commit Hash | Note |
|---|---|---|
| **02.09.2025** | f653d58132124854db42d2bd93d0c6b91da2c398 | Initial Commit |
| **03.09.2025** | bbe529891fc02ccf360e8a17b089269b415ca6dc | Commit for re-audit |

## Mainnet Deployments

| File | Address | Blockchain |
|---|---|---|
| **InchFusionBarterResolver.sol** | 0x6d1Ea9...6ebe1325 | Ethereum Mainnet |

# 1.4 Security Assessment Methodology

## Project Flow

| Stage | Scope of Work |
| --- | --- |
| Interim audit | **Project Architecture Review:**<br><br>· Review project documentation<br>· Conduct a general code review<br>· Perform reverse engineering to analyze the project's architecture based solely on the source code<br>· Develop an independent perspective on the project's architecture<br>· Identify any logical flaws in the design<br><br>OBJECTIVE: UNDERSTAND THE OVERALL STRUCTURE OF THE PROJECT AND IDENTIFY POTENTIAL SECURITY RISKS. |
| | **Code Review with a Hacker Mindset:**<br><br>· Each team member independently conducts a manual code review, focusing on identifying unique vulnerabilities.<br>· Perform collaborative audits (pair auditing) of the most complex code sections, supervised by the Team Lead.<br>· Develop Proof-of-Concepts (PoCs) and conduct fuzzing tests using tools like Foundry, Hardhat, and BOA to uncover intricate logical flaws.<br>· Review test cases and in-code comments to identify potential weaknesses.<br><br>OBJECTIVE: IDENTIFY AND ELIMINATE THE MAJORITY OF VULNERABILITIES, INCLUDING THOSE UNIQUE TO THE INDUSTRY. |
| | **Code Review with a Nerd Mindset:**<br><br>· Conduct a manual code review using an internally maintained checklist, regularly updated with insights from past hacks, research, and client audits.<br>· Utilize static analysis tools (e.g., Slither, Mythril) and vulnerability databases (e.g., Solodit) to uncover potential undetected attack vectors.<br><br>OBJECTIVE: ENSURE COMPREHENSIVE COVERAGE OF ALL KNOWN ATTACK VECTORS DURING THE REVIEW PROCESS. |

| Stage | Scope of Work |
| --- | --- |
| | **Consolidation of Auditors' Reports:**<br><br>· Cross-check findings among auditors<br>· Discuss identified issues<br>· Issue an interim audit report for client review<br><br>OBJECTIVE: COMBINE INTERIM REPORTS FROM ALL AUDITORS INTO A SINGLE COMPREHENSIVE DOCUMENT. |
| Re-audit | **Bug Fixing & Re-Audit:**<br><br>· The client addresses the identified issues and provides feedback<br>· Auditors verify the fixes and update their statuses with supporting evidence<br>· A re-audit report is generated and shared with the client<br><br>OBJECTIVE: VALIDATE THE FIXES AND REASSESS THE CODE TO ENSURE ALL VULNERABILITIES ARE RESOLVED AND NO NEW VULNERABILITIES ARE ADDED. |
| Final audit | **Final Code Verification & Public Audit Report:**<br><br>· Verify the final code version against recommendations and their statuses<br>· Check deployed contracts for correct initialization parameters<br>· Confirm that the deployed code matches the audited version<br>· Issue a public audit report, published on our official GitHub repository<br>· Announce the successful audit on our official X account<br><br>OBJECTIVE: PERFORM A FINAL REVIEW AND ISSUE A PUBLIC REPORT DOCUMENTING THE AUDIT. |

# 1.5 Risk Classification

## Severity Level Matrix

| Severity | Impact: High | Impact: Medium | Impact: Low |
|---|---|---|---|
| Likehood: High | Critical | High | Medium |
| Likehood: Medium | High | Medium | Low |
| Likehood: Low | Medium | Low | Low |

## Impact

- **High** – Theft from 0.5% OR partial/full blocking of funds (>0.5%) on the contract without the possibility of withdrawal OR loss of user funds (>1%) who interacted with the protocol.
- **Medium** – Contract lock that can only be fixed through a contract upgrade OR one-time theft of rewards or an amount up to 0.5% of the protocol's TVL OR funds lock with the possibility of withdrawal by an admin.
- **Low** – One-time contract lock that can be fixed by the administrator without a contract upgrade.

## Likelihood

- **High** – The event has a 50–60% probability of occurring within a year and can be triggered by any actor (e.g., due to a likely market condition that the actor cannot influence).
- **Medium** – An unlikely event (10–20% probability of occurring) that can be triggered by a trusted actor.
- **Low** – A highly unlikely event that can only be triggered by the owner.

## Action Required

- **Critical** – Must be fixed as soon as possible.
- **High** – Strongly advised to be fixed to minimize potential risks.
- **Medium** – Recommended to be fixed to enhance security and stability.
- **Low** – Recommended to be fixed to improve overall robustness and effectiveness.

## Finding Status

- **Fixed** – The recommended fixes have been implemented in the project code and no longer impact its security.
- **Partially Fixed** – The recommended fixes have been partially implemented, reducing the impact of the finding, but it has not been fully resolved.
- **Acknowledged** – The recommended fixes have not yet been implemented, and the finding remains unresolved or does not require code changes.

## 1.6 Summary of Findings

### Findings Count

| Severity | Count |
| --- | --- |
| Critical | 0 |
| High | 0 |
| Medium | 0 |
| Low | 3 |

### Findings Statuses

| ID | Finding | Severity | Status |
| --- | --- | --- | --- |
| L-1 | Implement sweep() for ETH | Low | Fixed |
| L-2 | Balance validation should use delta instead of absolute balance | Low | Fixed |
| L-3 | Remove unnecessary ETH handling logic for maker asset | Low | Fixed |

# 2. Findings Report

## 2.1 Critical

Not Found

## 2.2 High

Not Found

## 2.3 Medium

Not Found

## 2.4 Low

| | | | |
|---|---|---|---|
| L-1 | Implement sweep() for ETH | | |
| Severity | Low | Status | Fixed in bbe52989 |

**Description**

The InchFusionBarterResolver.sol#L125-L133 function in the InchFusionBarterResolver contract only handles ERC20 tokens but does not provide functionality to sweep native ETH that may accumulate in the contract.

**Recommendation**

We recommend implementing ETH sweeping functionality.

| L-2 | Balance validation should use delta instead of absolute balance | | |
|---|---|---|---|
| **Severity** | Low | **Status** | Fixed in bbe52989 |

**Description**

In the InchFusionBarterResolver.sol#L93-L123 function, the balance validation uses the absolute balance of the contract (IERC20(takerAsset.get()).balanceOf(address(this))) instead of calculating the delta between the balance before and after the swap execution.
This approach assumes that the contract starts with zero balance of the target token, which may not always be true. If the contract has any pre-existing balance of the target token, the validation will incorrectly pass even if the swap didn't produce the expected amount.

**Recommendation**

We recommend implementing balance validation using delta calculation (i. e. balanceAfter - balanceBefore).

| L-3 | Remove unnecessary ETH handling logic for maker asset | | |
|---|---|---|---|
| **Severity** | Low | **Status** | Fixed in bbe52989 |

**Description**

In the InchFusionBarterResolver.sol#L93-L123 function, there's logic that checks if the maker asset is ETH and skips token transfer:

```
if (makerAsset.get() != address(0) &&
    makerAsset.get() != address(0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE)
   ) {
    IERC20(makerAsset.get()).safeTransfer(address(executor), makingAmount);
}
```

This logic appears to be leftover code from UniswapX solver implementation and is not needed in the context of 1inch Fusion orders.

**Recommendation**

We recommend removing unnecessary logic.

# 3. About MixBytes

MixBytes is a leading provider of smart contract audit and research services, helping blockchain projects enhance security and reliability. Since its inception, MixBytes has been committed to safeguarding the Web3 ecosystem by delivering rigorous security assessments and cutting-edge research tailored to DeFi projects.

Our team comprises highly skilled engineers, security experts, and blockchain researchers with deep expertise in formal verification, smart contract auditing, and protocol research. With proven experience in Web3, MixBytes combines in-depth technical knowledge with a proactive security-first approach.

## Why MixBytes

- **Proven Track Record:** Trusted by top-tier blockchain projects like Lido, Aave, Curve, and others, MixBytes has successfully audited and secured billions in digital assets.
- **Technical Expertise:** Our auditors and researchers hold advanced degrees in cryptography, cybersecurity, and distributed systems.
- **Innovative Research:** Our team actively contributes to blockchain security research, sharing knowledge with the community.

## Our Services

- **Smart Contract Audits:** A meticulous security assessment of DeFi protocols to prevent vulnerabilities before deployment.
- **Blockchain Research:** In-depth technical research and security modeling for Web3 projects.
- **Custom Security Solutions:** Tailored security frameworks for complex decentralized applications and blockchain ecosystems.

MixBytes is dedicated to securing the future of blockchain technology by delivering unparalleled security expertise and research-driven solutions. Whether you are launching a DeFi protocol or developing an innovative dApp, we are your trusted security partner.

## Contact Information


 https://mixbytes.io/

 https://github.com/mixbytes/audits_public

 hello@mixbytes.io

 https://x.com/mixbytes