MixBytes()

Enso ERC-4337 Integration Security Audit Report

AUGUST 18, 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

Enso is an intent-based development layer that unifies on-chain data and smart contracts so developers can call high-level actions instead of writing bespoke integrations. The shortcuts-client-contracts repository contains wallet contracts that execute Enso shortcuts, i.e. predefined DeFi workflows; they use the ERC-4337 account-abstraction model. Core components include AbstractMultiSend for atomic batched calls without delegatecall, AbstractEnsoShortcuts for executing command sequences, a Withdrawable mix-in for owners to reclaim tokens, and the EnsoReceiver wallet that validates signatures, executes shortcuts and multisends, and supports safe fallback in case of failure. A factory deploys deterministic clones and a SignaturePaymaster verifies off-chain signatures and manages deposits to pay gas fees.

In this audit, we paid special attention to attacks related to Enso Shortcuts' batched-execution flow and its ERC-4337 account-abstraction wallet:

- Delegate-call injection – verified that delegatecall is explicitly blocked; no bypass paths detected;
- Partial-failure fund loss – confirmed full-transaction reverts on any sub-call failure, preserving atomicity;
- Signature forgery / signer spoofing – entryPoint increments nonces; signature validation is correct; only whitelisted signers accepted;

We also went through our detailed checklist, covering other aspects such as business logic, common ERC-20 issues, interactions with external contracts, integer overflows, reentrancy attacks, access control, type-cast pitfalls, rounding errors and other potential issues.

No significant vulnerabilities were found.

Key notes:

- `EnsoReceiver` and `SignaturePaymaster` don't increment `nonce` or check `validAfter`/`validUntil` because these are handled by the entryPoint (e.g., entryPoint v0.8 at 0x4337084d9e255ff0702461cf8895ce9e3b5ff108 on Ethereum mainnet). However, SignaturePaymaster allows setting a custom entryPoint, which might skip these checks.
- `EnsoReceiver.safeExecute(token, amount, call)` refunds the specified `amount` of tokens to the contract owner if `call` fails. If `amount` is less than the actual balance, the entire transaction reverts.
- Switching from one entryPoint to another may leave stake and deposit on the old one, but it's not an issue since it is possible to revert to the old entryPoint and withdraw via

unlockStake/withdrawStake/withdrawTo.
- SignaturePaymaster and EnsoReceiver may have different entryPoint during migration. If they're out of sync, transactions will revert.

1.3 Project Overview

Summary

Title	Description
Client Name	Enso Finance
Project Name	ERC-4337 Integration
Type	Solidity
Platform	EVM
Timeline	01.08.2025–18.08.2025

Scope of Audit

File	Link
src/AbstractMultiSend.sol	AbstractMultiSend.sol
src/AbstractEnsoShortcuts.sol	AbstractEnsoShortcuts.sol
src/utils/Withdrawable.sol	Withdrawable.sol
src/delegate/EnsoReceiver.sol	EnsoReceiver.sol
src/factory/ERC4337CloneFactory.sol	ERC4337CloneFactory.sol
src/paymaster/SignaturePaymaster.sol	SignaturePaymaster.sol

Versions Log

Date	Commit Hash	Note
01.08.2025	389c8a02b19974f3807ed3caca3a23931677db27	Initial Commit

Date	Commit Hash	Note
08.08.2025	f5b5952a6921d83857feda2182157a562614a722	Commit for reaudit
18.08.2025	715a1ff9ae8afb169255f5af23d8c0ca1b805980	Commit for reaudit

Mainnet Deployments

File	Address	Blockchain
EnsoReceiver.sol	0xdde9d7E5665954C77fAA50a1476B65C8e6FdECCd	Arbitrum Mainnet
ERC4337CloneFactory.sol	0x1a59347d28f64091079fa04a2cbd03da63dff154	Arbitrum Mainnet
SignaturePaymaster.sol	0xfa66d86a5Efc7632070b1F0b1C639C69a7E7D8C5	Arbitrum Mainnet
EnsoReceiver.sol	0xdde9d7E5665954C77fAA50a1476B65C8e6FdECCd	Base Mainnet
ERC4337CloneFactory.sol	0x1a59347d28f64091079fa04a2cbd03da63dff154	Base Mainnet
SignaturePaymaster.sol	0xfa66d86a5Efc7632070b1F0b1C639C69a7E7D8C5	Base Mainnet
EnsoReceiver.sol	0xdde9d7E5665954C77fAA50a1476B65C8e6FdECCd	BSC Mainnet
ERC4337CloneFactory.sol	0x1a59347d28f64091079fa04a2cbd03da63dff154	BSC Mainnet
SignaturePaymaster.sol	0xfa66d86a5Efc7632070b1F0b1C639C69a7E7D8C5	BSC Mainnet
EnsoReceiver.sol	0xdde9d7E5665954C77fAA50a1476B65C8e6FdECCd	Ethereum Mainnet
ERC4337CloneFactory.sol	0x1a59347d28f64091079fa04a2cbd03da63dff154	Ethereum Mainnet
SignaturePaymaster.sol	0xfa66d86a5Efc7632070b1F0b1C639C69a7E7D8C5	Ethereum Mainnet

File	Address	Blockchain
EnsoReceiver.sol	0xdde9d7E5665954C77fAA50a1476B65C8e6FdECCd	Optimism Mainnet
ERC4337CloneFactory.sol	0x1a59347d28f64091079fa04a2cbd03da63dff154	Optimism Mainnet
SignaturePaymaster.sol	0xfa66d86a5Efc7632070b1F0b1C639C69a7E7D8C5	Optimism Mainnet
EnsoReceiver.sol	0xdde9d7E5665954C77fAA50a1476B65C8e6FdECCd	Polygon Mainnet
ERC4337CloneFactory.sol	0x1a59347d28f64091079fa04a2cbd03da63dff154	Polygon Mainnet
SignaturePaymaster.sol	0xfa66d86a5Efc7632070b1F0b1C639C69a7E7D8C5	Polygon Mainnet

1.4 Security Assessment Methodology

Project Flow

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

Stage	Scope of Work
	Consolidation of Auditors' Reports: · Cross-check findings among auditors · Discuss identified issues · Issue an interim audit report for client review OBJECTIVE: COMBINE INTERIM REPORTS FROM ALL AUDITORS INTO A SINGLE COMPREHENSIVE DOCUMENT.
Re-audit	**Bug Fixing & Re-Audit:** · The client addresses the identified issues and provides feedback · Auditors verify the fixes and update their statuses with supporting evidence · A re-audit report is generated and shared with the client 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:** · Verify the final code version against recommendations and their statuses · Check deployed contracts for correct initialization parameters · Confirm that the deployed code matches the audited version · Issue a public audit report, published on our official GitHub repository · Announce the successful audit on our official X account 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	2

Findings Statuses

ID	Finding	Severity	Status
L-1	No Zero Address Check	Low	Fixed
L-2	ERC4337CloneFactory Front-Running	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	No Zero Address Check		
Severity	Low	**Status**	Fixed in f5b5952a

Description

`SignaturePaymaster.validatePaymasterUserOp()` uses `ECDSA.tryRecover()` without revert and assumes that a failed recovery returns `address(0)`, which is not considered a valid signer. However, the admin can call `setSigner(address(0), true)`, allowing any signature to pass validation.

SignaturePaymaster.sol#L169

Recommendation

We recommend restricting `address(0)` from being added to `validSigners` to prevent bypassing signature validation.

Client's Commentary:

fixed: f5b5952a6921d83857feda2182157a562614a722

L-2	ERC4337CloneFactory Front-Running		
Severity	Low	**Status**	Fixed in f5b5952a

Description

`ERC4337CloneFactory.deploy()` and `delegateDeploy()` can be front-run, causing the user's contract to be deployed prematurely and their original deployment transaction to revert.

Recommendation

We recommend either documenting this as expected behavior or checking the target address codehash and skipping deployment without reverting if it's non-zero.

Client's Commentary:

fixed: f5b5952a6921d83857feda2182157a562614a722

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