MixBytes()

Lido EasyTrack Security Audit Report

MAY 13, 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

The contracts within the scope include EasyTrack EVM factories for the MEV Boost Relay Allowed List contract. The AddMEVBoostRelays contract generates calldata for the EvmScriptExecutor, which adds new relays while ensuring there are no duplicates and that all limits are respected. The EditMEVBoostRelays contract generates calldata to modify existing relays, and the RemoveMEVBoostRelays contract creates calldata to remove specified relays after validation. Utility functions in the MEVBoostRelaysInputUtils library assist with decoding and validating relay input data.

The audit required a total effort of 2 team-days.

Before the audit, the same team conducted a review of the EVM Script Factories for the MEV Boost Allowed List specification and highlighted a few code improvements, which were applied prior to the start of this audit. As a result, no new issues were discovered.

During the audit, we thoroughly examined the following potential attack vectors:

- It is not possible to provide **crafted** relay or URI **byte data** while bypassing the checks implemented in the factory contracts. Providing correctly formatted but still erroneous data will result in the motion execution being reverted.
- The MEVBoostRelaysInputUtils library is used correctly to ensure the **correctness of the provided data**.
- The relay editing process is correctly implemented and consists of **two sequential actions**: deleting and then adding the relayer. The relayer URI is used as the key for lookup, allowing the correct relayer to be found and its fields to be updated.
- The **method signatures** called on MEVBoostRelayAllowedList.vy are correct.
- There should be no issues interacting with the contract in Vyper, as its additional runtime checks on certain data types are supported by the MEVBoostRelaysInputUtils library, which ensures **type correctness**.
- There are **no functional limitations** resulting from transferring the MEVBoostRelayAllowedList.vy manager role from the Relay Maintenance Committee to the EvmScriptExecutor contract.
- Creating **two conflicting motions** (e.g., removing and then editing the same relayer) will pass all validation checks but will not lead to data corruption or denial of service, as the conflicting motion will revert during execution.
- The calldata passed to the EVMScriptCreator is **correctly encoded** and aligns with the MEVBoostRelayAllowedList.vy contract interface.

Overall, the code quality within the audit scope appears to be robust and well-structured. The implementation of validation checks ensures that inputs are properly managed. The contracts exhibit a strong adherence to best practices in smart contract design and development.

1.3 Project Overview

Summary

Title	Description
Client Name	Lido
Project Name	EasyTrack
Type	Solidity
Platform	EVM
Timeline	24.03.2025 - 26.03.2025

Scope of Audit

File	Link
contracts/EVMScriptFactories/AddMEVBoostRelays.sol	AddMEVBoostRelays.sol
contracts/EVMScriptFactories/EditMEVBoostRelays.sol	EditMEVBoostRelays.sol
contracts/EVMScriptFactories/RemoveMEVBoostRelays.sol	RemoveMEVBoostRelays.sol
contracts/libraries/MEVBoostRelaysInputUtils.sol	MEVBoostRelaysInputUtils.sol

Versions Log

Date	Commit Hash	Note
24.03.2025	96606b69371268e5e3db3175bd1004d55c033a31	Initial Commit

Mainnet Deployments

File	Address	Blockchain
AddMEVBoostRelays.sol	0x00A3D6...FFd7bE60	Ethereum Mainnet
RemoveMEVBoostRelays.sol	0x9721c0...269c0306	Ethereum Mainnet
EditMEVBoostRelays.sol	0x6b7863...CC025535	Ethereum 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	0

2. Findings Report

2.1 Critical

Not Found

2.2 High

Not Found

2.3 Medium

Not Found

2.4 Low

Not Found

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