MixBytes()

Mellow Finance Security Audit Report

JUNE 09, 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 code presented in the scope implements a modular, upgradeable vault architecture based on the ERC4626 standard, with components designed for compatibility, migration, and access control. DVV contract serves as the main vault contract, extending MellowVaultCompat to support seamless migration from legacy storage. ERC4626Vault implements the core tokenized vault logic in compliance with the ERC4626 standard and MellowVaultCompat enables backward compatibility by managing dual storage layouts and supporting gradual state migration.

The audit was conducted over 2 days by 3 auditors.

During the audit, the following attack vectors were reviewed:

1. Gradual Migration via Dual Storage Logic. The MellowVaultCompat contract supports both legacy and new ERC20 storage layouts by dynamically resolving balances, allowances, and total supply across both. This enables a non-disruptive migration process where user data is automatically transitioned as they interact with the contract.

2. Storage Slot Offset for Backward Compatibility. All storage slots used in the previous contract version are explicitly protected from clashing using the _reserved padding defined in ERC4626Vault. This design ensures that legacy storage layout is not unintentionally overwritten during upgrades, preserving data integrity.

3. Allowance Handling & type(uint256).max**.** The _spendAllowance function deliberately skips allowance migration if the allowance is set to type(uint256).max (infinite approval). However, the infinite allowance is still respected correctly. If a user later reduces the allowance, the migration occurs, and the new value is used for subsequent transfers, preserving correctness and consistency.

4. Gas-Limited Migration in MigratorDVV**.** The MigratorDVV.migrateDVV() may exceed gas limits when processing a high number of withdrawals. In such case, the protocol allows for manual pre-processing of withdrawals to mitigate failures and support large-scale transitions.

5. Role Revocation on Deposit Pause. In VaultControl, calling pauseDeposits() sets the deposit pause flag and immediately revokes the PAUSE_DEPOSITS_ROLE from the caller. Even if the contract is already paused, subsequent calls from other contracts or users will still trigger role revocation. This design might cause some confusion but is minor in impact and likely intentional to enforce one-time use of that authority.

6. Correct ERC20 Behavior. The current implementation ensures that standard ERC20 behavior is upheld throughout the migration process. All key methods like transfer, approve, and transferFrom account for both storage layers, so user interactions remain reliable regardless of migration state.

7. Proper Use of Initializers. All initializer functions from the upgradeable contracts are explicitly and correctly invoked, following OpenZeppelin's upgradeable pattern. As a result, the contracts are securely initialized, and _disableInitializers() is called in the DVV constructor to prevent any possibility of re-initialization or unauthorized upgrade attacks.

The currently deployed implementation of the DVstETH contract was out of scope for this audit.

The code is well-structured and follows best practices for upgradeable contracts; no issues were found during the audit.

1.3 Project Overview

Summary

Title	Description
Client Name	Mellow Finance
Project Name	Simple-LRT Vault
Type	Solidity
Platform	EVM
Timeline	29.05.2025 - 02.06.2025

Scope of Audit

File	Link
src/vaults/VaultControl.sol	VaultControl.sol
src/vaults/DVV.sol	DVV.sol
src/vaults/MellowVaultCompat.sol	MellowVaultCompat.sol
src/vaults/ERC4626Vault.sol	ERC4626Vault.sol
src/vaults/VaultControlStorage.sol	VaultControlStorage.sol

File	Link
src/utils/MigratorDVV.sol	MigratorDVV.sol

Versions Log

Date	Commit Hash	Note
29.05.2025	2ee6ce9b4c54ad94bcebc12f976ee4af1a47afbe	Initial Commit

Mainnet Deployments

File	Address	Blockchain
DVV.sol	0x000000...962d93a1	Ethereum
MigratorDVV.sol	0x000000...5D239319	Ethereum

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