

MixBytes()

Resolv Staking Security Audit Report

MAY 27, 2025

Table of Contents

| | |
|---|-----------|
| 1. Introduction | 2 |
| 1.1 Disclaimer | 2 |
| 1.2 Executive Summary | 2 |
| 1.3 Project Overview | 2 |
| 1.4 Security Assessment Methodology | 5 |
| 1.5 Risk Classification | 7 |
| 1.6 Summary of Findings | 8 |
| 2. Findings Report | 9 |
| 2.1 Critical | 9 |
| 2.2 High | 9 |
| H-1 checkpoint() resets totalEffectiveSupply | 9 |
| 2.3 Medium | 11 |
| 2.4 Low | 11 |
| L-1 No restriction on emergency withdrawal of RESOLV staking tokens | 11 |
| L-2 Incomplete reward balance check | 12 |
| 3. About MixBytes | 13 |

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

Resolv Staking allows users to stake RESOLV tokens and earn rewards over time. The protocol features a weighted average holding period (WAHP) mechanism that provides additional staking incentives based on the duration of staking, along with a withdrawal cooldown system to prevent rapid liquidity outflows.

The audit was conducted by 3 auditors over a period of 3 days, employing automatic and manual code review methods.

Key focus areas during the audit included effective balance calculation, reward distribution mechanics, withdrawal processes, and general access control. The scope excluded external dependencies and focused on the core staking and airdrop functionality.

The code is secure; a previously discovered vulnerability affecting withdrawals has been promptly fixed.

Key notes and recommendations:

- **No reward token removal.** The `ResolvStaking` contract allows adding reward tokens but lacks functionality to remove or disable them.
- **Incentivization of frequent user interaction with the protocol.** The boost is based on WAHP (Weighted Average Holding Period) and updates only when the user interacts with the protocol. If the user is inactive, the boost stays the same. This is a deliberate design choice to encourage regular engagement.

1.3 Project Overview

Summary

| Title | Description |
|--------------|----------------|
| Client Name | Resolv |
| Project Name | Resolv Staking |

| | |
|----------|-------------------------|
| Title | Description |
| Type | Solidity |
| Platform | EVM |
| Timeline | 08.04.2025 – 14.05.2025 |

Scope of Audit

| File | Link |
|--|--|
| contracts/staking/ResolvStakingSilo.sol | ResolvStakingSilo.sol |
| contracts/staking/ResolvStaking.sol | ResolvStaking.sol |
| contracts/staking/libraries/ ResolvStakingCheckpoints.sol | ResolvStakingCheckpoints.sol |
| contracts/staking/libraries/ ResolvStakingErrors.sol | ResolvStakingErrors.sol |
| contracts/staking/libraries/ ResolvStakingEvents.sol | ResolvStakingEvents.sol |
| contracts/staking/libraries/ ResolvStakingStructs.sol | ResolvStakingStructs.sol |
| contracts/airdrop/ StakedTokenDistributor.sol | StakedTokenDistributor.sol |

Versions Log

| Date | Commit Hash | Note |
|------------|--|----------------------|
| 08.04.2025 | 771e82ae7e487d1740a85d80ce96a6f764471419 | Commit for the audit |
| 11.04.2025 | 8eac98d46a46f25303d4832f605e270081bc8322 | Commit for re-audit |
| 29.04.2025 | f1d45d1835c9f61e279a50bae6953f58280dc343 | Commit for re-audit |

| Date | Commit Hash | Note |
|------------|--|---------------------|
| 14.05.2025 | 49969c0e55cdf40236c8530d6b4cbbfa4bf780c9 | Commit for re-audit |

Mainnet Deployments

| File | Address | Blockchain |
|---------------------------------|---------------------|------------|
| TransparentUpgradeableProxy.sol | 0xFE4BCE...BE2E5E23 | Ethereum |
| ResolveStaking.sol | 0x1d2d1e...cfd827e6 | Ethereum |
| ProxyAdmin.sol | 0x1400e0...27f85b4D | Ethereum |
| ResolveStakingCheckpoints.sol | 0x253C6e...4d8F2784 | Ethereum |
| ResolveStakingSilo.sol | 0x502f9F...0eC88D4f | Ethereum |
| ResolveStakingHelpers.sol | 0x948AdE...86f17970 | Ethereum |

1.4 Security Assessment Methodology

Project Flow

| Stage | Scope of Work |
|---------------|--|
| Interim audit | Project Architecture Review: <ul style="list-style-type: none">• 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 <p>OBJECTIVE: UNDERSTAND THE OVERALL STRUCTURE OF THE PROJECT AND IDENTIFY POTENTIAL SECURITY RISKS.</p> |
| | Code Review with a Hacker Mindset: <ul style="list-style-type: none">• 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. <p>OBJECTIVE: IDENTIFY AND ELIMINATE THE MAJORITY OF VULNERABILITIES, INCLUDING THOSE UNIQUE TO THE INDUSTRY.</p> |
| | Code Review with a Nerd Mindset: <ul style="list-style-type: none">• 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. <p>OBJECTIVE: ENSURE COMPREHENSIVE COVERAGE OF ALL KNOWN ATTACK VECTORS DURING THE REVIEW PROCESS.</p> |

| Stage | Scope of Work |
|-------------|---|
| | <p>Consolidation of Auditors' Reports:</p> <ul style="list-style-type: none"> • Cross-check findings among auditors • Discuss identified issues • Issue an interim audit report for client review <p>OBJECTIVE: COMBINE INTERIM REPORTS FROM ALL AUDITORS INTO A SINGLE COMPREHENSIVE DOCUMENT.</p> |
| Re-audit | <p>Bug Fixing & Re-Audit:</p> <ul style="list-style-type: none"> • 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 <p>OBJECTIVE: VALIDATE THE FIXES AND REASSESS THE CODE TO ENSURE ALL VULNERABILITIES ARE RESOLVED AND NO NEW VULNERABILITIES ARE ADDED.</p> |
| Final audit | <p>Final Code Verification & Public Audit Report:</p> <ul style="list-style-type: none"> • 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 <p>OBJECTIVE: PERFORM A FINAL REVIEW AND ISSUE A PUBLIC REPORT DOCUMENTING THE AUDIT.</p> |

1.5 Risk Classification

Severity Level Matrix

| Severity | Impact: High | Impact: Medium | Impact: Low |
|--------------------|--------------|----------------|-------------|
| Likelihood: High | Critical | High | Medium |
| Likelihood: Medium | High | Medium | Low |
| Likelihood: 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 | 1 |
| Medium | 0 |
| Low | 2 |

Findings Statuses

| ID | Finding | Severity | Status |
|-----|--|----------|--------------|
| H-1 | <code>checkpoint()</code> resets <code>totalEffectiveSupply</code> | High | Fixed |
| L-1 | No restriction on emergency withdrawal of <code>RESOLV</code> staking tokens | Low | Acknowledged |
| L-2 | Incomplete reward balance check | Low | Fixed |

2. Findings Report

2.1 Critical

Not Found

2.2 High

| | | | |
|----------|--|--------|-------------------|
| H-1 | checkpoint() resets totalEffectiveSupply | | |
| Severity | High | Status | Fixed in 8eac98d4 |

Description

When `ResolvStaking.checkpoint()` is called with the zero address as the user, the variable `totalEffectiveSupply` is reset to zero because the function `ResolvStakingCheckpoints.updateEffectiveBalance()` returns zero for the zero address.

• [ResolvStakingCheckpoints.sol#L128](#)

A hacker can pass the zero address to the `checkpoint()` function via `ResolvStaking.updateCheckpoint()`. Resetting `totalEffectiveSupply` to zero would prevent users from being able to withdraw funds from the contract due to an underflow at the following line:

```
newTotalEffectiveSupply =  
    // (  
    _params.totalEffectiveSupply // =0  
    - oldEffectiveBalance // >0  
    // )  
    + newEffectiveBalance;
```

[ResolvStakingCheckpoints.sol#L163](#)

Recommendation

1. In `ResolvStakingCheckpoints.updateEffectiveBalance()`, return the current `totalEffectiveSupply` when the user address is zero.
2. Change the order of operations when calculating `newTotalEffectiveSupply`, adding first, subtracting last:

```
newTotalEffectiveSupply =  
    (_params.totalEffectiveSupply +  
    newEffectiveBalance)  
    - oldEffectiveBalance;
```

Client's Commentary:

56f2fe95

2.3 Medium

Not Found

2.4 Low

| | | | |
|----------|---|--------|--------------|
| L-1 | No restriction on emergency withdrawal of RESOLV staking tokens | | |
| Severity | Low | Status | Acknowledged |

Description

[ResolvStaking](#) and [ResolvStakingSilo](#) have a function `emergencyWithdrawERC20()`, which allows the admin to withdraw [RESOLV](#) tokens from the contract.

- [ResolvStaking.sol#L212](#)
- [ResolvStakingSilo.sol#L33](#)

The ability to withdraw [RESOLV](#) tokens (the main staking asset) via this function may present centralization concerns, as it could allow administrators to remove staked user funds.

Recommendation

We recommend restricting the emergency withdrawal function from transferring out the main staking asset.

Client's Commentary:

Acknowledged

| | | | |
|----------|---------------------------------|--------|-------------------|
| L-2 | Incomplete reward balance check | | |
| Severity | Low | Status | Fixed in 8eac98d4 |

Description

The condition

```
reward.token.balanceOf(address(this)) > reward.rewardRate * duration
```

fails when the balance is exactly sufficient. Since the multiplication could be precise, `>=` should be used.

• [ResolvStaking.sol#L172](#)

Recommendation

We recommend replacing `>` with `>=` to allow exact-match balances.

Client's Commentary:

[8eac98d4](#)

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>