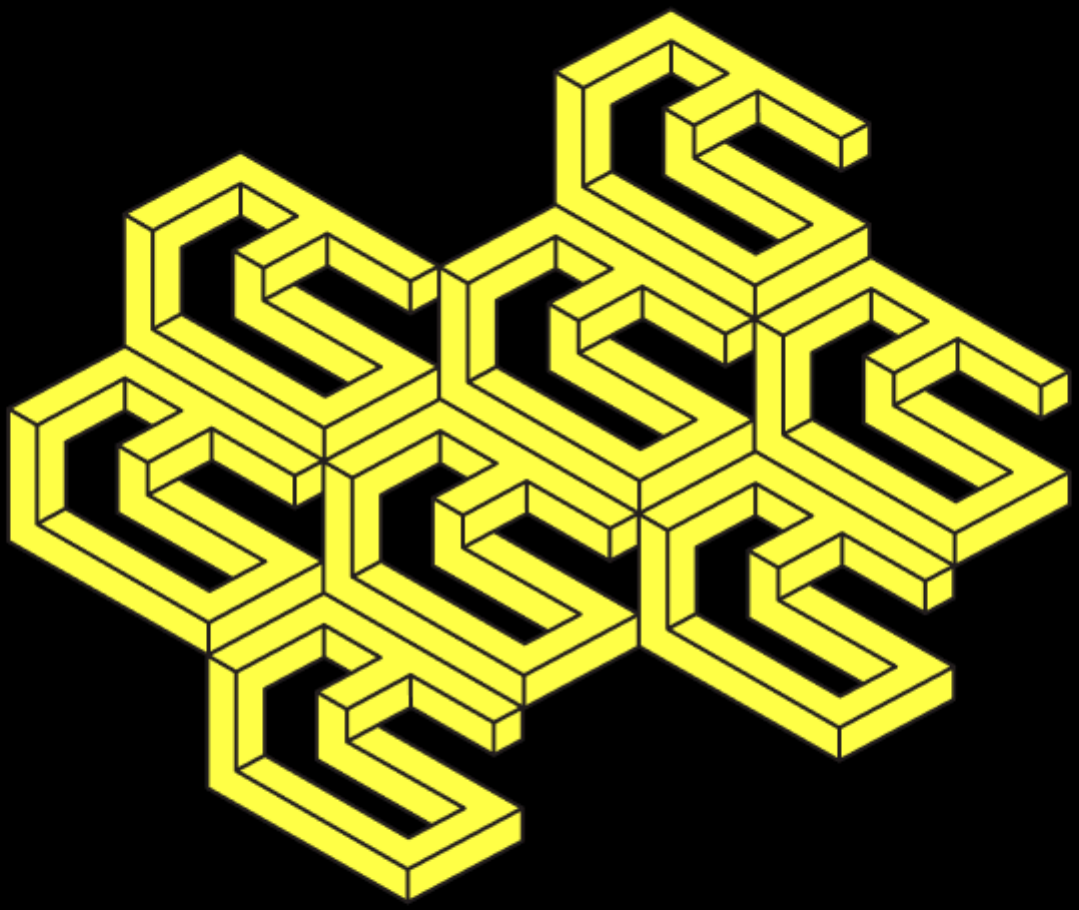

Exactly Protocol Staking Smart Contracts Review | August 2024

by ChainSafe Systems | August 2024

Table of contents

1. Introduction

Date	Auditor(s)
August 2024	Oleksii Matiiasevych, Anderson Lee

Exactly Protocol requested **ChainSafe Systems** to perform a review of the contracts update diff used for Staking Model. The contracts in scope can be identified as the following git diff:
`8ae3561e18d08930a28f85f136fba8372876c383 original`
`0872c11bc7fc6bcdbe630f072c688a448beacbe0 update`

After the initial review, **Exactly Protocol** team applied a number of updates which can be identified by the following git commit hash:
`f2073c0022a3ecb88b8b16acc10e1e797972898c`

Additional verification was performed after that.

Defining Severity

Each finding is assigned a severity level.

Note	Notes are informational in nature. They are typically suggestions around best practices or readability. Code maintainers should use their own judgment as to whether to address such issues.
Optimization	Optimizations are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.
Minor	Minor issues are objective in nature but are not security vulnerabilities. These should be addressed unless there is a clear reason not to.
Major	Major issues are security vulnerabilities that may not be directly exploitable or may require certain conditions in order to be exploited. All major issues should be addressed.
Critical	Critical issues are directly exploitable security vulnerabilities that need to be fixed.

Referencing updated code

Resolved	The finding has been acknowledged and the team has since updated the code.
Rejected	The team dismissed this finding and no changes will be made.

Disclaimer

The review makes no statements or warranties about the utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about the fitness of the contracts for any specific purpose, or their bug free status.

2. Executive Summary

There are **no** known compiler bugs for the specified compiler version (0.8.23), that might affect the contracts' logic.

There were **0 critical**, **0 major**, **2 minors**, 5 informational/optimization issues identified in the initial version of the contracts. The issues found in the contracts were not present in the final version of the contracts. They are described below for historical purposes. Besides that, we also confirmed that the final code contained fixes for findings by other researchers.

We are looking forward to future engagements with the Exactly Protocol team.

3. Line-by-line review

contracts/Market.sol

L653 Minor Resolved

The `clearBadDebt()` function does not use `unassignedEarnings` in bad debt clearing, which could result in a need to repeatedly call the `clearBadDebt()` function for the same borrower to clear more debt. Consider also increasing accumulator local variable.

L652 Note Resolved

The `clearBadDebt()` function has a semantically incorrect condition checking if the debt at maturity is repaid in full. The condition should be: `fixedPools[maturity].borrowed == 0`.

contracts/StakedEXA.sol

L195, L202, L390, L394, L431 Minor Resolved

The usage of `reward.transfer()` across the contract would not work on ERC20 tokens that do not return bool. Consider using `SafeERC20.safeTransfer()` instead.

L104 Note Resolved

the `initilize()` function could utilize `setMarket()` function instead of directly changing market storage variable for consistency.

L138 Optimization Resolved

The `_update()` function reads `rewardsTokens.length` value from storage multiple times.

L178 Note Acknowledged

The `claimWithdraw()` function description is partially incorrect as it claims the rewards not only when withdrawing but when depositing as well if the refTime already passed.

L461 Note Resolved

The `setMarket()` function allows zero address market to be used.