Security Assessment & Formal Verification Final Report

Fragmetric v1

October 2024

Prepared for Fragmetric

Table of content

Project Summary

Project Scope

Project Name	Repository (link)	Commit Hash	Platform
Fragmetric v1	/fragmetric-labs/fragmetric-contracts	`0b3eeff` and `51b9c90`	Solana

Project Overview

This document describes the specification and verification of Fragmetric using the Certora Prover and manual code review findings. The work was undertaken from Oct 7, 2024 to Oct 21, 2024

The following contract list is included in our scope: all files in /program/restaking/src are considered as in scope for this audit.

The Certora team performed a manual audit of all the Solana contracts. During this process, the team discovered bugs in the smart contracts code, as listed on the following pages.

The Certora Prover demonstrated that the implementation of the Solana contracts above is correct with respect to the formal rules written by the Certora team. Please note that a few more formal rules are not included in this report, as they were proven with an unreleased version of the Certora Prover. Once those rules are proven on a released version of the Certora Prover, we will add them to the next version of this document.

Protocol Overview

Fragmetric’s mission is to build a secure, transparent, and highly efficient restaking infrastructure that empowers users and supports the stability of the Solana's restaking ecosystem.

Fragmetric offers the option to deposit supported Liquid Staking Tokens (LST) or SOL to receive the Liquid Restaking Token (LRT) fragSOL. Fragmetric will restake deposited funds in various restaking protocols. It aims to get an optimal ratio between the different LSTs and restaking protocols to give high rewards and minimized risk.

Fragmetric has implemented a reward pool system where fragSOL holders can receive rewards points based on the amount of tokens they hold over time. By accumulating reward points, users can receive extra rewards on top of the restasking yield.

Protocol Status

At the time of this audit only part of all intended functionality is implemented.
The Fragmetric team intends to deploy in different phases. The reviewed version is called “phase 1” and only supports deposits of SOL and supported tokens, reward pool accounting and limited withdrawal functionality.

Upcoming scheduled features will include

- Staking Protocol Integrations
- Restaking protocol integration
- Enabling of fragSOL withdrawal and transfer

Certora has recommended against this and believes code should not be deployed before all basic functionality, including withdrawals and transfers, is implemented.

Findings Summary

The table below summarizes the findings of the review, including type and severity details.

Severity	Discovered	Confirmed	Fixed
Critical	1	1	-
High	1	1	-
Medium	1	1	1
Low	3	3	1
Informational	3		
Total	9		

Severity Matrix

Impact	High	Medium	High	Critical
	Medium	Low	Medium	High
	Low	Low	Low	Medium
		Low	Medium	High
			Likelihood	

Detailed Findings

ID	Title	Severity	Status
C-01	Withdrawals are not working for deposited tokens	Critical	Not yet fixed
H-01	fragSOL token cannot be transferred	High	Not yet fixed
M-01	Closed reward pool blocks withdrawals and deposits	Medium	Fixed
L-01	Claiming of rewards for users not implemented	Low	Not yet fixed
L-02	Possible to DOS deposits with repeated deposits/withdrawals	Low	Not yet fixed
L-03	Not possible to create new base reward pool after old one is closed	Low	Fixed

Critical Severity Issues

C-01 Withdrawals not working for deposited tokens

Severity: **Critical**	Impact: **High**	Likelihood: **High**
Files: user_fund_context.rs	Status: Not yet fixed	

Description: Users can make deposits in SOL and supported LST tokens. Withdrawals are only possible in SOL. The withdrawal process does not have functionality to convert or unstaked LST tokens for SOL.
This means that if users deposit LST tokens, there will not be enough SOL liquidity in the protocol to withdraw all deposited funds and tokens will be locked in the protocol.

Recommendations: Change/Implement withdrawal functionality to always allow withdrawal of all deposited funds.

Customer's response: Restaking ecosystem is building at a rapid pace. We are developing a procedure to integrate with the Jito restaking protocol and staking pool programs to circulate all assets in a fund through SOL, LST and VRT to fulfill investment and withdrawal obligations. This feature will be included in the next release and will enable withdrawals and transfers.

High Severity Issues

H-01 fragSOL token cannot be transferred		
Severity: **High**	Impact: **Medium**	Likelihood: **High**
Files: user_receipt_token_transfer_context.rs	Status: Not yet fixed	

Description: On depositing funds, a user receives fragSOL Liquid Restaking Tokens (LRT). These tokens are currently non transferable, severely limiting their use case and making them illiquid.
Transfer of the tokens is disabled via the token transfer hook.

Recommendations: Enable token transfers

Customer's response: At the beginning of the launch, we believed that fragSOL's lack of liquidity in De-Fi could lead to de-pegging issues, regardless of the actual fund's underlying asset reserve. After securing sufficient liquidity with the Phase1 launch, we are planning to integrate the reward system into De-Fi, including DEX and lending protocols, and enable token transfers in the next release.

Medium Severity Issues

M-01 Closed reward pool blocks withdrawals and deposits

Severity: **Medium**	Impact: **High**	Likelihood: **Low**
Files: modules/reward/update.rs	Status: Fixed in `d71f29e`	

Description: Reward pools are used to accumulate reward points for fragSOL holders. When depositing or withdrawing funds, the rewards are updated.
When a reward pool is closed by the fund manager, it still tries to update rewards on that pool when a user tries to deposit or withdraw.
In the `RewardPool.update` function a `RewardPoolClosedError` is thrown when it tries to update rewards for a closed pool. As a result all deposit and transfer requests will fail.

Customer's response: Fragmetric has confirmed the issue and applied a fix.
This is resolved in commit `d71f29e`

Fix Review: The logic for closed reward pools has been rewritten. Closing a pool is now only possible for holder specific pools and reward updates can still be processed for closed pools, but now use the `pool.closed_at` slot instead of the `current_slot` for reward calculation.

Low Severity Issues

L-01 Claiming of rewards for users not implemented		
Severity: **Low**	Impact: **Low**	Likelihood: **Medium**
Files: user_reward_context.rs	Status: Not yet fixed	Comment: General likelihood is considered high, but given the current status of code, we downgraded the likelihood to medium.

Description: fragSOL holders receive reward points based on their holdings. These reward points can be used to emit rewards proportionally to the accumulated amount of reward points. To receive those rewards, they need to be claimed by the user. The `claim_rewards` function is currently not implemented, making it impossible for users to claim rewards.

Recommendations: Implement user rewards claim functionality

Customer's response: Restaking protocols are not yet finalized, but are being developed at a rapid pace. In our Phase 1, prior to restaking protocol integration, we completed building a trustless reward system to transparently measure users' engagement with the Fragmetric platform. Once the implementation of rewards in the Jito re-staking protocol is clarified, this built rewards system will complete the implementation of automated reward distribution and claim functionality.

L-02 Possible to DOS deposits with repeated deposits/withdrawals		
Severity: **Low**	Impact: **Low**	Likelihood: **Low**
Files: user_fund_context	Status: Not yet fixed	

Description: For all depositable assets, the amount that can be deposited is capped at `capacity_amount`. This limit is checked against the accumulated deposit amount, which is the total sum of all deposits for that asset and does not decrease on withdrawals.
This makes it possible to repeatedly deposit and withdraw until the capacity_amount is reached, causing a Denial of Service for further deposits.

Exploit Scenario: Repeatedly deposit and withdraw until accumulated deposit amount reaches capacity amount.

Recommendations: There is no immediate action needed. Since there is a withdrawal fee being charged on withdrawals, it will cost the attacker a significant amount to be able to perform this DOS with no benefit. The fund manager can also easily increase the capacity amount to resolve it.

Customer's response: Acknowledged.

L-03 Not possible to create new base reward pool after old one is closed

Severity: **Low**	Impact: **Low**	Likelihood: **Low**
Files: fund_manager_reward_context.rs	Status: Fixed in `d71f29e`	

Description: A fund manager has the option to add and close reward pools. There is a distinction between “holder specific” reward pools and general “non holder specific” reward pools.
When adding a pool, it is checked that there can be only 1 pool per `holder/custom_contribution_accrual_rate_enabled` combination. So there can only be two “non holder specific pools”. One with `custom_contribution_accrual_rate_enabled == true` and one with `custom_contribution_accrual_rate_enabled == false`. Both of these will be created on deployment.
If one of those pools is closed, it is not possible to later create a new “non holder specific” pool with the same `custom_contribution_accrual_rate_enabled` flag because the closed pools are included in the check for duplicate pools.

Customer’s response: Fragmetric has confirmed the issue and applied a fix.
This is resolved in commit `d71f29e`

Fix Review: The logic for closed reward pools has been rewritten. Closing a pool is now only possible for holder specific pools and reward updates can still be processed for closed pools, but now use the `pool.closed_at` slot instead of the `current_slot` for reward calculation.

Informational Severity Issues

I-01. Incorrect function name is_contribution_accrual_rate_valid{}

Description: in modules/reward/update.rs there is a function `is_contribution_accrual_rate_valid()` which returns false when the rate is valid and true when invalid, which makes the behavior not match the function name.

Recommendation: Change function name to `is_contribution_accrual_rate_invalid()`

Customer's response: Acknowledged.

I-02. Incorrect function naming mock_transfer_hook_from_xx functions

Description: in instructions/user_fund_contest.rs there are three mock_transfer_hook_from_xx functions which update reward points after a fund transfer.
The names of those functions do not match the usage:

On deposit, tokens are minted (**from null to user**) and it uses
`mock_transfer_hook_from_fund_to_user`

On request withdrawal tokens go **from user to fund** and it uses
`mock_transfer_hook_from_user_to_null`

On cancel withdrawal funds go **from fund to user**, and it uses
`mock_transfer_hook_from_null_to_user`

Recommendation: Change function name to correspond with usage:
`Mock_transfer_hook_from_fund_to_user` **->** `mock_transfer_hook_from_null_to_user`
`Mock_transfer_hook_from_user_to_null` **->** `mock_transfer_hook_from_user_to_fund`
`Mock_transfer_hook_from_null_to_user` **->** `mock_transfer_hook_from_fund_to_user`

Customer's response: Acknowledged.

I-03. Using spot prices for price sources

Description: In the implementation for the price sources, the price is determined by looking at the current stakes value and total supply of the LST. Using spot prices can be a source for price manipulation, opening the system to several attack vectors.
For the Solana staking program and Marinade pools, we could not identify ways to manipulate the read amounts to manipulate the price.

Recommendation: It is recommended to use price oracles instead of spot prices, preferably with time weighted average or other forms of protection to avoid price manipulations.

Customer's response: Acknowledged.

Formal Verification

Note that the current list of properties is incomplete and will be extended in the next version of this document.

Verification Notations

Formally Verified	The rule is verified for every state of the contract(s), under the assumptions of the scope/requirements in the rule.
Formally Verified After Fix	The rule was violated due to an issue in the code and was successfully verified after fixing the issue
Violated	A counter-example exists that violates one of the assertions of the rule.

Formal Verification Properties

UserFundContext

Module General Assumptions

- `Loop iterations: Any loop was unrolled at most 3 times (iterations)`
- `CP-invocations are summarized to mock the expected behavior`

Module Properties

P-01. Integrity of UserFundContext	
Status: Verified	

Rule Name	Status	Description
rule_integrity_of_deposit_sol	Verified	*This rule verifies that sol deposited by the user is transferred to the fund's account and the appropriate amounts of fragSOL are minted to the user, thereby increasing*
rule_integrity_of_withdraw_request	Verified	*This rule verifies that a withdrawal request with the correct amounts is opened and the withdrawal amount is burned from the user and locked in the fund for further processing.*

Disclaimer

Even though we hope this information is helpful, we provide no warranty of any kind, explicit or implied. The contents of this report should not be construed as a complete guarantee that the contract is secure in all dimensions. In no event shall Certora or any of its employees be liable for any claim, damages, or other liability, whether in an action of contract, tort, or otherwise, arising from, out of, or in connection with the results reported here.

About Certora

Certora is a Web3 security company that provides industry-leading formal verification tools and smart contract audits. Certora's flagship security product, Certora Prover, is a unique SaaS product that automatically locates even the most rare & hard-to-find bugs on your smart contracts or mathematically proves their absence. The Certora Prover plugs into your standard deployment pipeline. It is helpful for smart contract developers and security researchers during auditing and bug bounties.

Certora also provides services such as auditing, formal verification projects, and incident response.