PUBLIC

Code Assessment

of the Sulu Extensions XII Smart Contracts

December 13, 2023

Produced for

by

Contents

1 Executive Summary

Dear Mona and Sean,

Thank you for trusting us to help Avantgarde Finance with this security audit. Our executive summary provides an overview of subjects covered in our audit of the latest reviewed contracts of Sulu Extensions XII according to Scope to support you in forming an opinion on their security risks.

Avantgarde Finance implements external positions for staking with StakeWise v3.

The most critical subjects covered in our audit are asset solvency, functional correctness, front-running, and accurate fund valuation. The security of all the aforementioned subjects is good. However, some aspects could be improved. Namely, the delayed fund valuation may be problematic, see Slashing Can Be Avoided. Further, functional correctness could be improved, see StakeWise Deposit May Revert.

The general subjects covered are code complexity, upgradeability, unit testing, and documentation.

In summary, we find that the codebase provides a good but improvable level of security.

It is important to note that security audits are time-boxed and cannot uncover all vulnerabilities. They complement but don't replace other vital measures to secure a project.

The following sections will give an overview of the system, our methodology, the issues uncovered and how they have been addressed. We are happy to receive questions and feedback to improve our service.

Sincerely yours,

ChainSecurity

1.1 Overview of the Findings

Below we provide a brief numerical overview of the findings and how they have been addressed.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	1
• Risk Accepted	1
Low-Severity Findings	2
• Code Corrected	1
• Risk Accepted	1

2 Assessment Overview

In this section, we briefly describe the overall structure and scope of the engagement, including the code commit which is referenced throughout this report.

2.1 Scope

The assessment was performed on the source code files inside the Sulu Extensions XII repository based on the documentation files. The table below indicates the code versions relevant to this report and when they were received.

V	Date	Commit Hash	Note
1	10 July 2023	86d45e025f520f2e1ebf7810df1fe8b978601d01	Initial Version
2	5 December 2023	70678cbb87592e8ac7875511712e4f75a81bda42	After intermediate report

For the solidity smart contracts, the compiler version `0.8.19` was chosen.

The contracts in scope are:

```
contracts/external-interfaces/IStakeWiseV3EthVault.sol
contracts/external-interfaces/IStakeWiseV3VaultsRegistry.sol
contracts/persistent/external-positions/stakewise-v3-staking/StakeWiseV3StakingPositionLibBase1.sol
contracts/release/extensions/external-position-manager/external-positions/stakewise-v3-staking/IStakeWiseV3StakingPosition.sol
contracts/release/extensions/external-position-manager/external-positions/stakewise-v3-staking/StakeWiseV3StakingPositionDataDecoder.sol
contracts/release/extensions/external-position-manager/external-positions/stakewise-v3-staking/StakeWiseV3StakingPositionLib.sol
contracts/release/extensions/external-position-manager/external-positions/stakewise-v3-staking/StakeWiseV3StakingPositionParser.sol
```

2.1.1 Excluded from scope

Everything not mentioned above is out-of-scope.

Further, the external systems are out-of-scope and are assumed to work correctly. We assume that users and managers are made aware of the potential pitfalls of using the external position.

2.2 System Overview

This system overview describes the initially received version (Version 1) of the contracts as defined in the Assessment Overview.

Furthermore, in the findings section, we have added a version icon to each of the findings to increase the readability of the report.

Avantgarde Finance offers a StakeWise v3 external position.

StakeWise v3 is a liquid staking protocol where users can create vaults for their liquid staking derivative. These vault tokens can then be unified to osETH through a lending mechanism that takes the vault tokens as collateral. Avantgarde Finance implements logic to stake (mint vault tokens) and to redeem them (there is currently no interaction with osETH).

The external position implements the following actions:

- `Stake`: Stake to a given vault and receive vault tokens. Note that it is not required to deposit 32 ETH (or a multiple of it).

- `Redeem`: Burn and withdraw immediately (skipping the queue). Only possible under the condition that sufficient unstaked funds are available in the Stakewise vault.
- `EnterExitQueue`: Signal an exit. Eventually, the funds will be unstaked. Note that vault tokens are burned immediately and a position ticket representing the exit is generated.
- `ClaimExitedAssets`: Claim the exited assets. Potentially, not all assets will be claimed and yet another position ticket will be received for the assets not exited yet.

`getManagedAssets()` returns the value of the vault shares plus the value of the position ticket by using the ERC-4626 `convertToAssets()` that StakeWise v3 vaults implement. Since no debt is taken, `getDebtAssets()` return empty arrays.

Please see Slashing Can Be Avoided for further consideration.

2.2.1 Roles and Trust Model

Please refer to the main audit report and the extension audit reports for a general trust model of Sulu.

In general, we assume Enzyme only interacts with normal ERC-20 tokens that do not have multiple entry points, callbacks, fees-on-transfer, or other special behaviours.

Fund owners and asset managers are generally fully trusted for a fund. However, their powers can be limited through the fund's settings. The funds' settings/policies are assumed to be set up correctly for the intended configuration/usage.

The managers are expected to regularly claim the fees and to pause the position if under-/overvaluation of the fund occurs.

Governance is fully trusted and expected to not only behave honestly but also to fully understand the systems they are interacting which includes choosing appropriate parameters.

All external systems are expected to be non-malicious and work correctly as documented. StakeWise v3 code base is private at the time of writing. However, we received the code and documentation. Given that it is still in development at the time of writing, its semantics could still change and lead to breaking changes.

3 Limitations and use of report

Security assessments cannot uncover all existing vulnerabilities; even an assessment in which no vulnerabilities are found is not a guarantee of a secure system. However, code assessments enable the discovery of vulnerabilities that were overlooked during development and areas where additional security measures are necessary. In most cases, applications are either fully protected against a certain type of attack, or they are completely unprotected against it. Some of the issues may affect the entire application, while some lack protection only in certain areas. This is why we carry out a source code assessment aimed at determining all locations that need to be fixed. Within the customer-determined time frame, ChainSecurity has performed an assessment in order to discover as many vulnerabilities as possible.

The focus of our assessment was limited to the code parts defined in the engagement letter. We assessed whether the project follows the provided specifications. These assessments are based on the provided threat model and trust assumptions. We draw attention to the fact that due to inherent limitations in any software development process and software product, an inherent risk exists that even major failures or malfunctions can remain undetected. Further uncertainties exist in any software product or application used during the development, which itself cannot be free from any error or failures. These preconditions can have an impact on the system's code and/or functions and/or operation. We did not assess the underlying third-party infrastructure which adds further inherent risks as we rely on the correct execution of the included third-party technology stack itself. Report readers should also take into account that over the life cycle of any software, changes to the product itself or to the environment in which it is operated can have an impact leading to operational behaviors other than those initially determined in the business specification.

4 Terminology

For the purpose of this assessment, we adopt the following terminology. To classify the severity of our findings, we determine the likelihood and impact (according to the CVSS risk rating methodology).

- *Likelihood* represents the likelihood of a finding to be triggered or exploited in practice
- *Impact* specifies the technical and business-related consequences of a finding
- *Severity* is derived based on the likelihood and the impact

We categorize the findings into four distinct categories, depending on their severity. These severities are derived from the likelihood and the impact using the following table, following a standard risk assessment procedure.

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

As seen in the table above, findings that have both a high likelihood and a high impact are classified as critical. Intuitively, such findings are likely to be triggered and cause significant disruption. Overall, the severity correlates with the associated risk. However, every finding's risk should always be closely checked, regardless of severity.

5 Findings

In this section, we describe any open findings. Findings that have been resolved have been moved to the Resolved Findings section. The findings are split into these different categories:

- Design: Architectural shortcomings and design inefficiencies

Below we provide a numerical overview of the identified findings, split up by their severity.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	1

- Slashing Can Be Avoided Risk Accepted

Low-Severity Findings	1

- StakeWise Deposit May Revert Risk Accepted

5.1 Slashing Can Be Avoided

Design Medium Version 1 Risk Accepted

CS-SUL12-002

The smart contract layer does not immediately know when a slashing event on the Consensus Layer has happened. Offchain, however, it is easy to immediately know when a validator has been slashed.

A user of an enzyme vault that uses one of the staking external positions could monitor for slashings and immediately withdraw from the vault when such an event happens. By doing this, they will be able to redeem their assets (up to the available liquidity) at the pre-slashing price.

Once the slashing is accounted for in the vault (after about 12 hours in Stakewise), the slashing loss of the users that withdrew previously will instead be taken by those users that are still deposited in the vault.

Also, note that the same behaviour is present in Stakewise's vaults. There, any user can withdraw immediately up to the available liquidity and also dodge slashing. However, it is expected that the available liquidity in Stakewise vaults should never be more than 32 ETH, as otherwise, it would have been possible to stake them with an additional validator.

Risk accepted:

Avantgarde Finance acknowledged the issue and replied:

```
Since consensus layer slashing is not readable directly from the execution layer,
slashing can only be made known by posting to the execution layer, and there will
always be an opportunity to front-run posting (the same goes for Chainlink aggregators).

Fund managers must be aware of this risk and take any necessary precautions to mitigate
the risk where needed, e.g., via policies and/or queued redemptions.
```

5.2 StakeWise Deposit May Revert

Design | Low | Version 1 | Risk Accepted

CS-SUL12-003

The following is an excerpt from the StakeWise documentation:

> When keeper.canHarvest(<vault address>) returns false, the user can stake ETH to the vault without a state update. Otherwise, the updateStateAndDeposit function must be used.

The enzyme external position only uses `deposit()`, not `updateStateAndDeposit()`. If a state update is required, the manager currently needs to call `updateState()` from a separate address and then `deposit()` through the vault. Otherwise, `deposit()` reverts.

The same issue also applies to `redeem()`.

Risk accepted:

Avantgarde Finance replied:

```
For now, managers can wait until a non-harvestable (i.e., depositable) moment.
At a later time, we may update this or include a link/button to call `updateState()`
directly.
```

6 Resolved Findings

Here, we list findings that have been resolved during the course of the engagement. Their categories are explained in the Findings section.

Below we provide a numerical overview of the identified findings, split up by their severity.

Critical-Severity Findings	0
High-Severity Findings	0
Medium-Severity Findings	0
Low-Severity Findings	1

- StakeWise V3 Position Ticket Valuation Code Corrected

6.1 StakeWise V3 Position Ticket Valuation

Design Low Version 1 Code Corrected

CS-SUL12-001

The StakeWise position's value is the sum of the value of the vault tokens held and the value of the position tickets.

A position ticket's withdrawal value is determined in Stakewise's `calculateExitedAssets` function. This function does not use the current share price, it uses a checkpointed price which is set when the update happens that lets the contract know that a withdrawal is completed.

The external position uses the current exchange rate instead of the fixed exchange rate at the relevant state update.

As a result, a fund could be under- or overvalued.

Code corrected:

The code has been changed. Tickets not exited yet are evaluated as previously. However, exited tickets are evaluated based on `calculateExitedAssets()` which returns the claimed amount.