Angel Protocol
Audit Report

Prepared for Angel Protocol, 30th November 2022

Table of Contents

Introduction

SCV was engaged by Angel Protocol to assist in identifying security threats and vulnerabilities that have the potential to affect their security posture. Additionally, SCV will assist the team in understanding the risks and identifying potential mitigations.

Scope

SCV performed the security assessment on the following codebase:

- https://github.com/AngelProtocolFinance/angelprotocol-smart-contracts/tree/RC-v1.9/
- Code Freeze: *14af18fdb346b7374c7158092e698391bc84e18f*

Remediations were performed and reviewed by SCV in the following hash:

- Code Freeze: *23fc2d2b24527d605bca36664483c835f49ca371*

Methodologies

SCV performs a combination of automated and manual security testing based on the scope of testing. The testing performed is based on the extensive experience and knowledge of the auditor to provide the greatest coverage and value to Angel Protocol. Testing includes, but is not limited to, the following:

- Understanding the application and its code base purpose;
- Deploying SCV in-house tooling to automate dependency analysis and static code review;
- Analyse each line of the code base and inspect application security perimeter;
- Review underlying infrastructure technologies and supply chain security posture;

Code Criteria and Test Coverage

SCV used a scale from 0 to 10 that represents how *SUFFICIENT* or *NOT SUFFICIENT* each code criteria was during the assessment:

Criteria	Status	Notes
Provided Documentation	**SUFFICIENT**	While the vault contracts did not provide detailed documentation or specification, the code comments were detailed enough.
Code Coverage Test	**SUFFICIENT**	Astroport - 59.89% coverage, 749/1253 lines covered. Loop - 61.57% coverage, 782/1270 lines covered.
Code Readability	**SUFFICIENT**	The codebase had good readability and utilised many Rust and CosmWasm best practices.
Code Complexity	**SUFFICIENT**	N/A

Vulnerabilities Summary

#	Summary Title	Risk Impact	Status
1	Incorrect vault token calculation	**Critical**	**Resolved**
2	Static mint amount may cause vault tokens to unexpectedly exceed maximum capacity	**Medium**	**Resolved**
3	restake_claim_reward does not validate that returned amount is not zero	**Low**	**Resolved**
4	Contracts should use two-step ownership transfer	**Low**	**Resolved**
5	Instantiation lacking parameter validation	**Low**	**Resolved**
6	Replace Magic Numbers	**Informational**	**Resolved**
7	Updatable LP token may introduce unintended consequences	**Informational**	**Resolved**

Detailed Vulnerabilities

1 – Incorrect vault token calculation

Risk Impact: Critical - **Status**: Resolved

Description

The current `vt_2_lp_rate` calculation for determining the vault token per LP token redemption rate is incorrectly implemented. This will result in the incorrect amount of vault tokens being issued during the `stake_lp_token` and `reinvest_to_locked_receive` functions.

This issue exists because the `vt_2_lp_rate` rate is determined by dividing `state.total_shares` by `increased_total_lp`. `Increased_total_lp` is the sum of `state.total_lp_amount` and `lp_amount`. `Increased_total_lp` is where the issue exists; this value effectively dilutes the denominator with the newly added `lp_amount` that should not be included.

The following simulation illustrates the current vault token calculation compared to the recommended vault token calculation. The initial deposit is performed and `1000000` (one vault token) is minted for 5 added LP. Then when the second deposit occurs with 10 LP tokens, an amount which is double the initial deposit amount and represents a 2/3 share of the vault, the current formula calculates a mint amount of `666666` when the true share should be `2000000`.

```
----------------------
         ADDED LP: 5
        VT MINTED: 1000000
        TOTAL SHARES: 1000000 | EXPECTED: 1000000
        TOTAL LP: 5
```

```
----------------------
        ADDED LP: 10
        VT MINTED: 666666 | EXPECTED: 2000000
        TOTAL SHARES: 1666666 | EXPECTED: 3000000
        TOTAL LP: 15
----------------------
        ADDED LP: 5
        VT MINTED: 416666 | EXPECTED: 1000000
        TOTAL SHARES: 2083332 | EXPECTED: 4000000
        TOTAL LP: 20
----------------------
        ADDED LP: 432
        VT MINTED: 1991149 | EXPECTED: 86400000
        TOTAL SHARES: 4074481 | EXPECTED: 90400000
        TOTAL LP: 452
----------------------
        ADDED LP: 1
        VT MINTED: 8994 | EXPECTED: 200000
        TOTAL SHARES: 4083475 | EXPECTED: 90600000
        TOTAL LP: 453
----------------------
(*1 Vault Token = 1000000)
```

This issue is found in the following code lines:

- `contracts/vaults/astroport/src/executers.rs:693`
- `contracts/vaults/astroport/src/executers.rs:921`
- `contracts/vaults/astroport/src/executers.rs:957`
- `contracts/vaults/loop/src/executers.rs:701`
- `contracts/vaults/loop/src/executers.rs:923`
- `contracts/vaults/loop/src/executers.rs:959`

Recommendations

We recommend adjusting the `vt_2_lp_rate` so that instead of using `Increased_total_lp` in the denominator, it uses `state.total_lp_amount`.

2 – Static mint amount may cause vault tokens to unexpectedly exceed maximum capacity

Risk Impact: Medium - **Status**: Resolved

Description

The `vt_mint_amount` determines the amount of vault tokens that are minted per LP token, but for the first invocation when `state.total_shares` is zero the value is simply set to a static value of one vault token (`1000000`) . The functions where this exists do not implement a check to ensure that the first deposit is of sufficient size because `vt_2_lp_rate` is based off of this initial LP deposit. It is best practice to enforce a minimum value for the first deposit.

For example, if the initial ratio is set to a very high vault token to LP token rate, the `TOKEN_INFO` maximum capacity could be reached very quickly which would block future vault token minting if the maximum capacity is set for future vaults.

This issue is found in the following code lines:

- `contracts/vaults/astroport/src/executers.rs:690`
- `contracts/vaults/astroport/src/executers.rs:918`
- `contracts/vaults/astroport/src/executers.rs:954`
- `contracts/vaults/loop/src/executers.rs:698`
- `contracts/vaults/loop/src/executers.rs:920`
- `contracts/vaults/loop/src/executers.rs:956`

Recommendations

We recommend implementing a minimum value validation when the `state.total_shares` is zero to enforce a sufficient initial deposit amount.

3 – restake_claim_reward does not validate that returned amount is not zero

Risk Impact: Low - **Status**: Resolved

Description

The `restake_claim_reward` function in `contracts/vaults/loop/src/executers.rs:238` and `contracts/vaults/astroport/src/executers.rs:250` does not check to ensure that the `reward_amount` returned from subtracting the previous balance from the current reward token balance is not zero. If this amount is zero the contract should return an error. While it is likely that an error would be returned further in the execution, it is best practice to return a specific error at this point noting that the reward token claim amount is zero, rather than relying on an external contract or function to error on the zero amount. This is especially important when considering this design for future vaults.

Recommendations

We recommend returning an error in the vault's `restake_claim_reward` function if the calculated `reward_amount` is zero.

4 – Contracts should use two–step ownership transfer

Risk Impact: Low - **Status**: Resolved

Description

The current ownership transfer for each of the contracts is executed in one step, which imposes a risk that if the new owner is incorrect, then the admin privileges of the contract are effectively transferred and lost. A two-step ownership transfer is best practice because it requires the new admin to accept ownership before the transfer and config changes occur.

- `contracts/vaults/astroport/src/executers.rs:46`
- `contracts/vaults/junoswap/src/executers.rs:41`

Recommendations

We recommend implementing a two-step ownership transfer where the current owner proposes a new owner address, and then that new owner address must call the contract to accept ownership within a finite time frame. This also applies when updating other privileged addresses.

5 – Instantiation lacking parameter validation

Risk Impact: Low - **Status**: Resolved

Description

The `instantiate` function in `contracts/vaults/astroport/src/contract.rs:55-56` does not properly validate that decimal values in `ap_tax_rate` and `interest_distribution` are within the expected range of values. This may allow the admin to introduce a misconfiguration.

Recommendations

We recommend validating `ap_tax_rate` and `interest_distribution` to ensure that their values are within an acceptable range.

6 – Replace Magic Numbers

Risk Impact: Informational - **Status**: Resolved

Description

Throughout the codebase, magic numbers are used. Magic numbers are hard-coded numbers without context. The use of magic numbers reduces the readability and maintainability of the codebase

Instances of magic numbers are listed below:

- `contracts/vaults/astroport/src/executers.rs:690, 918, 945`
- `contracts/vaults/loop/src/executers.rs:698, 920, 956`

Recommendations

We recommend replacing the magic numbers mentioned above with a constant that is descriptive of its value and use case.

7 – Updatable LP token may introduce unintended consequences

Risk Impact: Informational - **Status**: Resolved

Description

The vaults contain a `lp_pair_contract` address in their config values. This value is updatable in each vault's `update_config` function. This is used when performing the initial vault setup operations, but the function does not prevent the owner address from overwriting an existing LP token pair contract address stored in the vault contracts config. If this value were to be updated to a different address it may cause inconsistencies and unintended consequences.

Recommendations

We recommend first checking that the `config.lp_pair_contract` is unset before updating the value. If the value is Some, then return an error unless there is a specific use case that can be documented to validate the need to update this parameter.

Document control

Version	Date	Approved by	Changes
0.1	21/11/2022	Vinicius Marino	Document Pre-Release
0.2	29/11/2022	SCV Auditors Team	Remediation Revisions
1.0	30/11/2022	Vinicius Marino	Document Release

Appendices

A. Appendix – Risk assessment methodology

A qualitative risk assessment is performed on each vulnerability to determine the impact and likelihood of each.

Risk rate will be calculated on a scale. As per criteria Likelihood vs Impact table below:

	Rare	Unlikely	Possible	Likely
Critical	**Medium**	**Severe**	**Critical**	**Critical**
Severe	**Low**	**Medium**	**Severe**	**Severe**
Moderate	**Low**	**Medium**	**Medium**	**Severe**
Low	**Low**	**Low**	**Low**	**Medium**
Informational	**Informational**	**Informational**	**Informational**	**Informational**

LIKELIHOOD

- Likely: likely a security incident will occur;
- Possible: It is possible a security incident can occur;
- Unlikely: Low probability a security incident will occur;
- Rare: In rare situations, a security incident can occur;

IMPACT

- Critical: May cause a significant and critical impact;
- Severe: May cause a severe impact;
- Moderate: May cause a moderated impact;
- Low: May cause low or none impact;
- Informational: May cause very low impact or none.

B. Appendix – Report Disclaimer

The content of this audit report is provided “As is”, without representations and warranties of any kind.

The author and their employer disclaim any liability for damage arising out of, or in connection with, this audit report.

Copyright of this report remains with the author.