St4k3h0us3 - steak-contracts Audit Report

Prepared for St4k3h0us3, 7 April 2022

Table of Contents

Introduction

SCV was engaged by St4k3h0us3 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/st4k3h0us3/steak-contracts

- Git sha1 hash: *bcb34d59f8677a49518413c1b244bb9e436a8237*

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 St4k3h0us3. 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(6-10)** or **NOT SUFFICIENT(0-5)** each code criteria was during the assessment:

Criteria	Status	Scale Range	Notes
Provided Documentation	**Not Sufficient**	3-4	N\A
Code Coverage Test	**Not Sufficient**	2-3	N\A
Code Readability	**Sufficient**	7-8	N\A
Code Complexity	**Sufficient**	6-7	N\A

Conclusion

The identified vulnerabilities pose a *low* to *none* level of technical risk to the St4k3h0us3. The provided codebase implements the *liquid staking derivative* and has been implemented securely.

Vulnerabilities Summary

	Title and Summary	Risk	Status
1	Contract might run out-of-gas due unbonding requests	**Low**	**Acknowledged**
2	Slippage tolerance (max_spread) is not enforced on swaps	**Low**	**Remediated**
3	Validator slashing events can cause unbalance during unbonding period	**Low**	**Acknowledged**
4	English word typos found in the codebase	**Informational**	**Remediated**
5	Ensure native assets is not Zero	**Informational**	**Remediated**

Detailed Vulnerabilities

Vulnerability 1: Contract might run out-of-gas due unbonding requests

Likelihood	Impact	Risk
Possible	Low	**Low**

Notes

It an edge case that will only happen for thousands of unclaimed unbonding requests, which is quite unlikely to happen.

Description

The `withdraw_unbonded()` function logic at the `stake-hub/src/execute.rs#431` can caused the contract to run out of gas depending on the size of unbonding request it's processing.

Recommendations

It's an edge case that might never happen although might be worthy to re-think this logic to avoid running into this edge case.

Vulnerability 2: Slippage tolerance (max_spread) is not enforced on swaps

Likelihood	Impact	Risk
Rare	Low	**Low**

Notes

Slippage is enforced by the `minimum_received` parameter of the *zap()* function that should be specified by the frontend.

Description

When performing `Zap()` swaps on `steak-zapper/src/contract.rs#84` there is no checks to ensure the swap spread is appropriate and that won't result in a huge slippage spread gap and consequentially impact funds if the pair pool is unbalanced.

In *Astroport*, the spread is calculated as the difference between the ask amount (using the constant pool price) before and after the swap operation. Once `max_spread` is set, it will be compared against the actual swap spread. In case the swap spread exceeds the provided max limit, the swap will fail.

Note that the spread is calculated before commission deduction in order to properly represent the pool's ratio change.

Recommendations

As a suggestion, ensure the slippage tolerance stays within *0.1%* to *1%*.

Vulnerability 3: Validator slashing events can cause unbalance during unbonding period

Likelihood	Impact	Risk
Rare	Low	**Low**

Notes

A slashing protection mechanism will be introduced in a future release.

Description

During slashing events, a validators loses part of their stake as a punishment which impacts the total amount of Lunas to be received from the contract to the user.

The contract will return an error when executing *withdraw_unbonded()* since the total amount is less from the expected amount due slashing.

The current implementation requires that, any loss caused by slashing or validator misbehave would need to be manually/programmatically refunded to the contract which might not be too practical.

Recommendations

Slashing does not happen often, although a new implementation logic might be required to address such events rather than rely in the manual process.

Vulnerability 4: English word typos found in the codebase

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Description

The codebase contains typos in comments that can affect overall code quality and makes it more difficult to understand.

- `minimum_recieved` at `steak-zapper/src/state#14` should be `minimum_received`;
- `minimum_recieved` at `steak-zapper/src/state#24` should be `minimum_received`;
- `state.minimum_recieved` at `steak-zapper/src/contract.rs#80` should be `state.minimum_received`;
- `state.minimum_recieved` at `steak-zapper/src/contract.rs#217` should be `state.minimum_received`;
- `state.minimum_recieved` at `steak-zapper/src/contract.rs#225` should be `state.minimum_received`;

Recommendations

Rename words to the correct values.

Vulnerability 5: Ensure native assets is not Zero

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Description

In the `steak-zapper/src/helper.rs#18` there is no checks to ensure the received Terra demon amount is not equal to *0*.

Recommendations

It's recommended to add this check to ensure received funds is not Zero.

Document control

Document changes

Version	Date	Name	Changes
0.1	2022-04-06	Vinicius Marino	Initial report
0.2	2022-04-06	Vinicius Marino	Team communication and Pre-Release
1.0	2022-04-07	Vinicius Marino	Final report release

Document contributors

Name	Role	Email address
Vinicius Marino	Security Specialist	vini@scv.services

Appendices

Appendix A: 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.

Appendix B: 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:

Impact \ Likelihood	Rare	Unlikely	Possible	Likely
Critical	Medium	High	Critical	Critical
Severe	Low	Medium	High	High
Moderate	Low	Medium	Medium	High
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.


SCV