Kujira orca-queue Report

Prepared for Kujira, 9 March 2022

Table of Contents

Document control

Document changes

Version	Date	Name	Changes
0.1	2022-03-09	Vinicius Marino	Initial report
0.2	2022-03-09	Vinicius Marino	Team communication and Pre-Release
1.0	2022-03-09	Vinicius Marino	Report Release

Document contributors

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

Introduction

SCV was engaged by Kujira 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.

`orca-queue` contract is an implementation based of Anchor's Queue public contract with some custom logic developed by Kujira.

Scope

SCV performed the security assessment on the following Kujira assets.

- https://github.com/Team-Kujira/orca-queue/commit/5586e528e14145598b5312033a748b6b1bdefdb2

Kujira uses a `liquidity-stability-pool` curve logic in the `orca-queue` contract that was excluded from scope as any other directly financial related attacks. The whitepaper related to the implementation can be found in the following link below:

- Scalable_Reward_Distribution_with_Compounding_Stakes.pdf

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 Kujira. 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;
- Analysis each line of the code base and inspect application perimeter;
- Review underlying infrastructure technologies and supply chain security posture;

Code Criteria and Test Coverage

SCV is using a scale from **0** to **10** that represents how `SUFFICIENT(6-10)` or `NOT SUFFICIENT(0-5)` each code criteria was assessed:

Criteria	Status	Scale Range	Notes
Provided Documentation	**Not Sufficient**	1-2	N\A
Code Coverage Test	**Sufficient**	8-9	N\A
Code Readability	**Sufficient**	7-8	N\A
Code Complexity	**Sufficient**	7-8	N\A

Vulnerabilities Summary

	Title and Summary	Risk	Status
1	Automatically activate bids might lead to pool manipulation	**Low**	**Acknowledged**
2	Lack of validations on UpdateConfig parameters may lead to inconsistent state	**Low**	**Acknowledged**
3	Unnecessary use of Canonical address transformations adds complexity	**Informational**	**Remediated**

Detailed Vulnerabilities

Vulnerability 1: Automatically activate bids might lead to pool manipulation

Likelihood	Impact	Risk
Rare	Low	**Low**

Notes

```
Implementation is intended by design.
```

Description

In extreme market conditions, during the bid execution logic, a bidder can activate any number of bids without having to wait the pre-determined time to activated it. This condition triggers when the total amount of bids available is below `bid_threshold` defined in config. As a result, it opens an opportunity to bots to manipulate the pool when conditions are reached.

```
let available_bids: Uint256 = TOTAL_BIDS.load(deps.storage).
    unwrap_or_default();
     if available_bids < config.bid_threshold {
         bid.activate(&mut bid_pool);
         bid_pool.save(deps.storage)?;
         let total_bids = available_bids + amount;
         TOTAL_BIDS.save(deps.storage, &total_bids)?;
```

Recommendations

It might be worthy tracking submit bids coming from the same wallet address or defined a delay between submits requests to be activated.

Vulnerability 2: Lack of validations on UpdateConfig parameters may lead to inconsistent state

Likelihood	Impact	Risk
Rare	Low	**Low**

Description

The exposed interface for `update_config` permits the contract owner to modify parameters that may result in an inconsistent state. An example, would be the `waiting_period` parameter.

Recommendations

Enforce validation on all parameters on state. Alternately, implement a migration of outstanding bids logic when `update_config` is executed.

Vulnerability 3: Unnecessary use of Canonical address transformations adds complexity

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Description

Canoncial address transformations are no longer encouraged to use due its lack of efficiency in most cases. In fact, Canoncial transformations no longer saves addresses in canoncial format into the storage. The use of the Canoncial transformation has no directly security implications, although it adds an unnecessary complexity to the code base due transformations back and forth, which could be optimized and simplified.

Recommendations

As a good practices, it's recommended to use Add type for addresses input validations and removing all unnecessary references to Canoncial transformations.

More information can be found in the URL link below:

- https://docs.cosmwasm.com/docs/0.16/architecture/addresses/

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.

COPYRIGHT OF THIS REPORT REMAINS WITH THE AUTHOR.

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

SMART CONTRACT VERIFICATION