Redacted Money - Mixer Contracts - Audit Report

Prepared for Redacted Money, 9 September 2022

Table of Contents

Introduction

SCV was engaged by Redacted Money 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/redactedLabs/mixer-contract
- Code Freeze: *dc3fe574c5066cf8f3be804b9ca57af20d7e22ff*

SCV notes that, cryptography attacks were not part of the scope and were not comprehensive tested. The following libraries references were taken into consideration while performing the security audit.

- https://github.com/scipr-lab/libsnark
- https://github.com/matter-labs/awesome-zero-knowledge-proofs

Remediations were applied into the following branch:

- https://github.com/redactedLabs/mixer-contract/tree/audit
- Code Freeze *e20fed834a99c7045a0a45802f02fc7415eb2067*

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 Redacted Money. 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	Protocol documentation not provided
Code Coverage Test	**Not Sufficient**	3-4	Lacking testing coverage
Code Readability	**Sufficient**	6-7	N/A
Code Complexity	**Sufficient**	5-6	N/A

Vulnerabilities Summary

	Title and Summary	Risk	Status
1	Lack of end-to-end test coverage	**Low**	**Remediated**
2	Lack of validation during contract instantiation and update can lead to misconfigurations	**Low**	**Remediated**
3	execute_withdraw does not validate proof is not empty which will cause panic	**Low**	**Remediated**
4	Canonical address transformations are inefficient	**Informational**	**Remediated**
5	Consider removing unused code to improve readability and maintainability	**Informational**	**Remediated**
6	must_pay function can simplify fund checking logic	**Informational**	**Remediated**

Detailed Vulnerabilities

1. Lack of end-to-end test coverage

Likelihood	Impact	Risk
Unlikely	Low	**Low**

Description

The contracts in the scope of this audit lacked full end-to-testing which may help in testing true contract flows rather than simple tests for isolated function execution. It is best practice to provide end-to-end testing flows.

Recommendations

We recommend implementing end to end tests before bringing the code to production.

2. Lack of validation during contract instantiation and update can lead to misconfigurations

Likelihood	Impact	Risk
Possible	Low	**Low**

Description

When performing contract instantiation and updating the configuration, there are no validations being performed on the message parameters that ensure the address supplied are valid for.

This is also true for `denomination` which could use additional validation to ensure it falls within the expected values. `denomination` should be checked to ensure it is not zero.

Affected code lines and variables:

- `contracts/redacted/src/contract.rs:40,47,51`
  - `msg.admin_addr`,
  - `msg.fee_addr`,
  - `msg.hashier_addr`(contract instantiation phase)
- `contracts/redacted/src/contract.rs:99,103`
  - `admin_addr`,
  - `fee_addr` (update config phase)
- `contracts/redacted/src/contract.rs:50`
  - `denomination` (contract instantiation phase)

Recommendations

Consider validating the `Addr` string before saving it into storage for the addresses mentioned above, and validating that denomination is greater than zero.

3. execute_withdraw does not validate proof is not empty which will cause panic

Likelihood	Impact	Risk
Unlikely	Low	**Low**

Description

The execute_withdraw function in `mixer-contract/contracts/redacted/src/contract.rs:309` does not validate that the proof provided by the caller is not empty. If an empty proof is provided, it will silently pass into the `verify_proof` function which will cause a panic. It is best practice to avoid panics and return an appropriate error if the proof field is an empty string.

Recommendations

We recommend validating that proof is not empty before passing it to `verify_proof` and returning an informative error.

4. Canonical address transformations are inefficient

Likelihood	Impact	Risk
Unlikely	Informational	**Informational**

Description

While previously recommended as a best practice, usage of canonical addresses for storage is no longer encouraged. Canonical addresses are no longer stored in a canonical format, so the transformation adds overhead without much benefit. Additionally, the codebase is more complicated with address transformations.

Recommendations

We recommend removing any transformation from human to canonical addresses and using the Addr type for validated addresses instead.

5. Consider removing unused code to improve readability and maintainability

Likelihood	Impact	Risk
Rare	Informational	**Informational**

Description

There are many instances in the contract where unused code blocks are commented and not removed. This impacts the readability of the whole codebase.

Affected code lines:

- `contracts/redacted/src/contract.rs`
  - 346-347
  - 355-360
  - 446-470
  - 479-482
  - 489-491
  - 512-517
- `contracts/redacted/src/verify.rs`
  - 12-17
  - 24-29

Recommendations

Consider removing unused code from the codebase.

6. must_pay function can simplify fund checking logic

Likelihood	Impact	Risk
Unlikely	Informational	**Informational**

Description

The `execute_deposit` function in `mixer-contract/contracts/redacted/src/contract.rs:125-139` uses validations to ensure the funds are sent as expected with the deposit message. This usage is valid as it is currently implemented, but the common way of handling this validation now is through the `cw_utils` `must_pay` function.

Recommendations

We recommend using the must_pay function to simplify the fund checking logic, and then simply checking to ensure that the returned amount from the function is equal to the expected sum of the denomination and platform_fee.

https://docs.rs/cw-utils/latest/src/cw_utils/payment.rs.html#32-39

Document control

Document changes

Version	Date	Name	Changes
0.1	2022-09-06	Vinicius Marino	Initial report
0.2	2022-09-07	Vinicius Marino	Team communication and Pre-Release
1.0	2022-09-09	Vinicius Marino	Revisions and Document 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.

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:

Likelihood / Impact	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