AUDITMOS

Secure your space

STARTERRA

audit report

August 21, 2021

TABLE OF CONTENTS

contact@auditmos.com

AUDITMOS

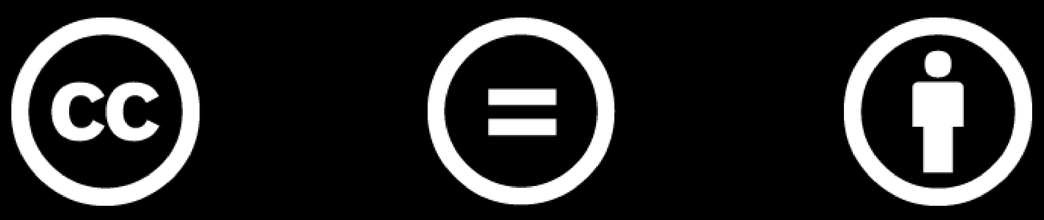
Secure your space

STARTERRA

audit report

August 21, 2021

LICENSE



contact@auditmos.com

AUDITMOS

Secure your space

STARTERRA

audit report

August 21, 2021

DISCLAIMER

contact@auditmos.com

AUDITMOS

Secure your space

STARTERRA

audit report

August 21, 2021

APPROACH AND METHODOLOGY

PURPOSE

1. Determine the correct operation of the protocol, according to the design specification.
2. Identify possible vulnerabilities that could be exploited by an attacker.
3. Detect errors in the smart contract that could lead to unexpected behavior.
4. Analyze whether best practices were followed during development.
5. Make recommendations to improve security and code readability.

CODEBASE

Repository	https://github.com/starterra/app-smart-contracts/
Branch	develop
Commit hash	63107b7c00764621dd49c36d098b2bdb97191089

METHODOLOGY

1. Reading the available documentation and understanding the code.
2. Doing automated code analysis and reviewing dependencies.
3. Checking manually source code line by line for security vulnerabilities.
4. Following guildlines and recommendations.
5. Preparing this report.

AUDITMOS

Secure your space

STARTERRA

audit report

August 21, 2021

DESCRIPTION

Issues Categories:

Severity	Description
CRITICAL	vulnerability that can lead to loss of funds, failure to recover blocked funds, or catastrophic denial of service.
HIGH	vulnerability that can lead to incorrect contract state or unpredictable operation of the contract.
MEDIUM	failure to adhere to best practices, incorrect usage of primitives, without major impact on security.
LOW	recommendations or potential optimizations which can lead to better user experience or readability.

Each issue can be in the following state:

State	Description
PENDING	still waiting for resolving
ACKNOWLEDGED	know but not planned to resolve for some reasons
RESOLVED	fixed and deployed

contact@auditmos.com

AUDITMOS

Secure your space

STARTERRA

audit report

August 21, 2021

FINDINGS

Finding	Severity	Status
#1 - limit the number of vesting contracts to iterate in vesting_gateway contract	MEDIUM	RESOLVED
#2 - limit the number of pending unbonds in staking contract	MEDIUM	RESOLVED
#3 - use struct instead of vector	LOW	RESOLVED
#4 - use unauthorized pattern to all handle methods	LOW	RESOLVED
#5 - cover with test all private methods and check if unauthorized exception is thrown	LOW	RESOLVED
#6 - improve tests code coverage	LOW	RESOLVED

contact@auditmos.com

AUDITMOS

Secure your space

STARTERRA

audit report

August 21, 2021

#1 - LIMIT THE NUMBER OF VESTING CONTRACTS TO ITERATE IN VESTING_GATEWAY CONTRACT

Severity	Status
MEDIUM	RESOLVED

The current contract implementation does not limit in any way the number of contracts among which the vesting_gateway contract looks for the given address and returns which vesting contract contains it.

This can lead to an overrun of the amount of gas that has been provided to execute that contract, especially when those contracts are a significant number.

RECOMMENDATION

Limit the number of vesting contracts that the vesting_gateway contract searches

PROOF OF SOURCE

/vesting-gateway/src/contract.rs#L196

contact@auditmos.com

STARTERRA

audit report

August 21, 2021

#2 - LIMIT THE NUMBER OF PENDING UNBONDS IN STAKING CONTRACT

Severity	Status
MEDIUM	RESOLVED

The current implementation of the contract in no way limits the number of requests to withdraw funds from a staking contract.

This could lead to this being exploited in an undesirable way by a malicious actor who, due to the low transaction fees on the Terra blockchain, will want to cause a denial of service in the staking contract by repeatedly requesting to withdraw a small number of tokens. However this will affect only single user who does it. That is why we decided to classified is as medium not high.

RECOMMENDATION

Limit the number of pending unbonds in the staking contract and let the user know how many of these allowed are currently in use.

PROOF OF SOURCE

/staking/src/contract.rs#L500

/staking/src/contract.rs#L534

STARTERRA

audit report

August 21, 2021

#3 - USE STRUCT INSTEAD OF VECTOR

Severity	Status
LOW	RESOLVED

The current implementation of the `assert_distribution_schedule` function is not very readable due to the use of the `Vec<T>` data type instead of `Struct`, this is not a security bug but the correct use of the appropriate data types for their intended purpose leads to optimal code according to best practices in this area.

RECOMMENDATION

To make smart contract code more readable it is recommended to use struct data type instead of vector for complex data type.

PROOF OF SOURCE

/staking/src/contract.rs#L40

STARTERRA

audit report

August 21, 2021

Severity	Status
LOW	RESOLVED

#4 - USE UNAUTHORIZED PATTERN TO ALL HANDLE METHODS

The current implementation of contracts contains two different methods to check if a message called within a handle function is called by the contract owner, leads to less readable code and more difficult to manage.

RECOMMENDATION

It is worth to use KISS(Keep It Simple Stupid) and DRY(Do Not Repeat Yourself) principles while developing software. These principles allow to create less code, not to repeat implementation and easier testing.

PROOF OF SOURCE

/vesting-gateway/src/contract.rs#L43

contact@auditmos.com

STARTERRA

audit report

August 21, 2021

#5 - COVER WITH TEST ALL PRIVATE METHODS AND CHECK IF UNAUTHORIZED EXCEPTION IS THROWN

Severity	Status
LOW	RESOLVED

In the current contract implementation not all functions with restricted access to contract owner are covered by tests. It might lead to some unpredictable behaviour during this function call by some address which is not owner address.

RECOMMENDATION

It is highly recommended to test all of the functions which should have restriceted access and/or should be called only by contract owner.

PROOF OF SOURCE

/vesting-gateway/src/testing

/vesting-toll-bridge/src/testing

/vesting/src/testing

STARTERRA

audit report

August 21, 2021

#6 - IMPROVE TESTS CODE COVERAGE

Severity	Status
LOW	RESOLVED

Test Coverage is an important indicator of software quality and an essential part of software maintenance. It helps in evaluating the effectiveness of testing by providing data on different coverage items. It is a useful tool for finding untested parts of a code base. Test coverage is also called code coverage in certain cases.

Test coverage can help in monitoring the quality of testing and assist in directing the test generators to create test cases that cover areas that have not been tested. It helps in determining a quantitative measure of Test coverage, which is an indirect measure of quality and identifies redundant test cases that do not increase coverage.

RECOMMENDATION

It is highly recommended to test all of the functions and have high ratio of test coverage. It is recommended to use code coverage reporting tool for the Cargo build system for example cargo-tarpaulin.

PROOF OF SOURCE

vesting-gateway/src/testing

/vesting-toll-bridge/src/testing

/vesting/src/testing

/staking/src/testing.rs

auditmos.com

Secure your space

contact@auditmos.com