Audit Report

Random Earth Smart Contracts

v1.0

December 10, 2021

Table of Contents

License

Disclaimer

This audit has been performed by

Oak Security

https://oaksecurity.io/
info@oaksecurity.io

Introduction

Purpose of this Report

Oak Security has been engaged by Lido Finance to perform a security audit of stLuna

The objectives of the audit are as follows:

1. Determine the correct functioning of the protocol, in accordance with the project specification.
2. Determine possible vulnerabilities, which could be exploited by an attacker.
3. Determine smart contract bugs, which might lead to unexpected behaviour.
4. Analyze whether best practices have been applied during development.
5. Make recommendations to improve code safety and readability.

This report represents a summary of the findings.

As with any code audit, there is a limit to which vulnerabilities can be found, and unexpected execution paths may still be possible. The author of this report does not guarantee complete coverage (see disclaimer).

Codebase Submitted for the Audit

The audit has been performed on the following GitHub repository:

https://github.com/RandomEarthTeam/contracts

Commit hash: `36a04924d299efd817070a07ce950cd63f64b836`

Methodology

The audit has been performed in the following steps:

1. Gaining an understanding of the code base's intended purpose by reading the available documentation.
2. Automated source code and dependency analysis.
3. Manual line by line analysis of the source code for security vulnerabilities and use of best practice guidelines, including but not limited to:
    a. Race condition analysis
    b. Under-/overflow issues
    c. Key management vulnerabilities
4. Report preparation

Functionality Overview

The submitted code implements the smart contracts for an NFT marketplace, including auction and distribution contracts.

How to read this Report

This report classifies the issues found into the following severity categories:

Severity	Description
Critical	A serious and exploitable vulnerability that can lead to loss of funds, unrecoverable locked funds, or catastrophic denial of service.
Major	A vulnerability or bug that can affect the correct functioning of the system, lead to incorrect states or denial of service.
Minor	A violation of common best practices or incorrect usage of primitives, which may not currently have a major impact on security, but may do so in the future or introduce inefficiencies.
Informational	Comments and recommendations of design decisions or potential optimizations, that are not relevant to security. Their application may improve aspects, such as user experience or readability, but is not strictly necessary. This category may also include opinionated recommendations that the project team might not share.

The status of an issue can be one of the following: **Pending,** **Acknowledged** or **Resolved**. Informational notes do not have a status, since we consider them optional recommendations.

Note, that audits are an important step to improve the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of the system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**. We include a table with these criteria below.

Note, that high complexity or low test coverage does not necessarily equate to a higher risk, although certain bugs are more easily detected in unit testing than a security audit and vice versa.

Summary of Findings

No	Description	Severity	Status
1	Wrong reservation assignment could lead to incorrect receiver for NFT	Critical	Resolved
2	Missing check for maker's ability to cover the maker fee leads to incorrect validation query result	Major	Resolved
3	NFT ownership is not cleared when NFTs are withdrawn, which leads to inconsistent state	Major	Resolved
4	Setting royalties in settlement contract fails for CW1155 token	Major	Resolved
5	Submit execution plan will fail due to message sent to wrong contract	Major	Resolved
6	Missing check for ability to execute an order leads to misleading validation query result	Minor	Acknowledged
7	Querier contract's order update returns a misleading taker asset filled value	Minor	Acknowledged
8	Closing an English auction leaves the maker order open	Minor	Acknowledged
9	Treating Luna as a special case for tax calculation may lead to problems in the case of protocol updates	Minor	Acknowledged
10	Approve and transfer pattern implies a bad user experience and might introduce security issues	Minor	Acknowledged
11	Relayer fees are skipped if either taker or maker fee is zero	Minor	Resolved
12	Missing input validation of order infos and fill amounts in order execution might confuse users	Minor	Resolved
13	Overflow checks not enabled for release profile in all packages	Minor	Acknowledged
14	Only the auction contract's owner can close English auctions	Informational	Acknowledged
15	Settlement contract's configuration change of nft only flag could lead to open orders that cannot be cancelled anymore	Informational	Acknowledged

16	Use Addr type instead of String	Informational	Acknowledged

Code Quality Criteria

Criteria	Status	Comment
Code complexity	Medium	-
Code readability and clarity	Medium-High	-
Level of Documentation	Medium-High	-
Test Coverage	Low	The codebase submitted contains a minimal amount of integration tests, and only one single unit test. The test coverage reported by `cargo tarpaulin` is 1.57%. We strongly recommend increasing test coverage, especially to cover edge cases that are hard to spot in a manual audit. Examples are the matching of orders in the English auction contract and authorization of order execution in the settlement contract.

Detailed Findings

1. Wrong reservation assignment could lead to incorrect receiver for NFT

Severity: Critical

In `contracts/contracts/stardust-nft-distribution/src/contract.rs:222`, the sender's address is assigned to the reservation. Since the contract owner and operator are able to create reservations on behalf of users, this might lead to the contract owner or operator receiving the NFT, rather than the reservation owner, even though the reservation owner paid for the NFT. This could lead to lost NFTs, for example, if the owner or operator is a smart contract such as a governance contract that cannot transfer NFTs.

Recommendation

We recommend replacing `info.sender` with `reservation_owner` in `contracts/contracts/stardust-nft-distribution/src/contract.rs:222`.

Status: Resolved

2. Missing check for maker's ability to cover the maker fee leads to incorrect validation query result

Severity: Major

The `query_validate` function in `contracts/contracts/stardust-settlement/src/queries.rs:9` does not validate whether the maker has enough stardust balance to cover the maker fee charged in `contracts/contracts/stardust-settlement/src/order.rs:117` when checking if an order is sufficiently capitalized. This will lead to undercapitalized orders being returned as valid, which is incorrect.

Recommendation

We recommend including a check for the maker fee to the validation function.

Status: Resolved

3. NFT ownership is not cleared when NFTs are withdrawn, which leads to inconsistent state

Severity: Major

The settlement contract does not clear the stored `NFT_OWNER` for an NFT when a user sends the `Withdraw` message. NFT ownership is only updated on deposit in the `deposit_balance` function in `contracts/contracts/stardust-settlement/src/state.rs:70`, but not cleared in the `withdraw_balance` function in `contracts/contracts/stardust-settlement/src/state.rs:80`. That leads to an inconsistent state.

Recommendation

We recommend clearing the NFT owner on user withdrawal.

Status: Resolved

4. Setting royalties in settlement contract fails for CW1155 tokens

Severity: Major

As part of the settlement contract's `set_royalty` function in `contracts/contracts/stardust-settlement/src/royalties.rs:22`, the minter of the token is queried. That query uses the CW721 contract's `Minter` query, which does not exist for CW1155 tokens. Consequently, royalty fees can only be set for CW721 tokens, not for CW1155 tokens.

Recommendation

We recommend adding support to set royalties for CW1155 tokens as well.

Status: Resolved

5. Submit execution plan will fail due to message sent to wrong contract

Severity: Major

In `contracts/contracts/stardust-settlement/src/execution.rs:131`, the `ValidateAndTransferExecutorBalance` message is sent to the executor contract, but the executor contract does not accept that message.

Recommendation

We recommend sending the `ValidateAndTransferExecutorBalance` message to the settlement contract instead by replacing `cfg.executor_address.clone()` with `env.contract.address`.

Status: Resolved

6. Missing check for ability to execute an order leads to misleading validation query result

Severity: Minor

The `query_validate` function in `contracts/contracts/stardust-settlement/src/queries.rs:10` calls the `validate_into_info` function with `None` as the last argument. That leads to a skip of validation whether a particular account can actually execute the order. This implies a misleading query result since a successfully validated order might fail validation during execution.

Recommendation

We recommend adding an address to the query that will be used in the `query_validate` function.

Status: Acknowledged

The Random Earth team states that there is currently no plan to support direct orders.

7. Querier contract's order update returns a misleading taker asset filled value

Severity: Minor

In the querier contract's OrderUpdate logic in `contracts/contracts/stardust-querier/src/contract.rs:132`, `taker_asset_filled` is set to a value of `0` for orders that are undercapitalized, filled or cancelled. A `0` fill value is inconsistent.

Recommendation

We recommend returning an `Option` instead with a `None` value for orders that are undercapitalized, filled or cancelled.

Status: Acknowledged

The Random Earth team states: "[T]he querier contract is only supposed to be used by [our] own backend, and is not meant for general use, so the value is irrelevant."

8. Closing an English auction leaves the maker order open

Severity: Minor

In the `close_english_auction` function in `contracts/contracts/stardust-auction/src/contract.rs:132`, the maker order is not closed when closing the English auction. That leaves dangling maker orders, leading to an inconsistent state.

Recommendation

We recommend closing the maker order when the English auction is closed.

Status: Acknowledged

The Random Earth team states that the maker order cannot be executed anymore once the English auction gets executed, so it is not a problem if it stays open.

9. Treating Luna as a special case for tax calculation may lead to problems with Terra protocol updates

Severity: Minor

In `contracts/packages/stardust_protocol/src/asset.rs:35` Luna is treated as a special case for tax calculations, with a hard-coded zero value. However, this might lead to inconsistencies if Terra changes Luna tax policy in a future protocol update. In such a case, the contract would pay the tax, leading to any user funds being spent.

Recommendation

We recommend treating Luna the same as other native tokens and querying the tax rate from Terra.

Status: Acknowledged

The Random Earth team states: "This is the same logic as TerraSwap, and it is extremely unlikely that Terra would add a tax to LUNA and brick most of the dApps on their blockchain."

10. Approve and transfer pattern implies a bad user experience and might introduce security issues

Severity: Minor

In several places in the codebase, the protocol relies on the `approve` and `transfer` pattern. It is generally better to use CW20 receive hooks, as they provide a better user experience (they do not require a separate approval and do not require revocation), are more gas efficient, and are usually more secure since approvals have often a disproportionate size and no expiry.

Recommendation

We recommend the usage of CW20 receive hooks.

Status: Acknowledged

11. Relayer fees are skipped if either taker or maker fee is zero

Severity: Minor

The if statement in `contracts/contracts/stardust-settlement/src/order.rs:94` contains the logical or `||` operator instead of the logical and `&&` operator. This leads to no fees are being charged if either the maker or the taker fee is zero

Recommendation

We recommend correcting the logical or to the logical and operator.

Status: Resolved

12. Missing input validation of order infos and fill amounts in order execution might confuse users

Severity: Minor

The `execute_orders` function in `contracts/contracts/stardust-settlement/src/order.rs:149` does not validate whether two arrays `order_infos` and `fill_amounts` are of equal length. If they are not, any excess elements will be ignored. That might be unexpected for users.

Recommendation

We recommend asserting equal length of the arrays to improve usability.

Status: Resolved

13. Overflow checks not enabled for release profile in all packages

Severity: Minor

While set in the workspace level `contracts/Cargo.toml`, some packages do not explicitly enable overflow checks. This is not a security concern in the current version of the contracts since the checks are enabled from the workspace to all packages, but future refactoring might leave some packages vulnerable to over- or underflows. Packages that currently don't have overflow checks enabled are:

- `contracts/contracts/stardust-settlement/Cargo.toml`
- `contracts/contracts/stardust-auction/Cargo.toml`
- `contracts/contracts/stardust-executor/Cargo.toml`
- `contracts/contracts/stardust-nft-distribution/Cargo.toml`
- `contracts/contracts/stardust-querier/Cargo.toml`
- `contracts/contracts/stardust-registry/Cargo.toml`

Recommendation

We recommend enabling overflow checks explicitly in every `Cargo.toml` file.

Status: Acknowledged

14.Only the auction contract's owner can close English auctions

Severity: Informational

Closing an English auction is restricted to the auction contract's owner in `contracts/contracts/stardust-auction/src/contract.rs:42`. That might be misleading for users.

Recommendation

We recommend allowing the auction's maker to close the auction as well.

Status: Acknowledged

The Random Earth team states that the auction contract is currently not used.

15. Settlement contract's configuration change of nft only flag could lead to open orders that cannot be cancelled anymore

Severity: Informational

Setting the `nft_only` field from `false` to `true` using the settlement contract's `update_config` function in `contracts/contracts/stardust-settlement/src/admin.rs:24` will lead to the `validate_into_info` function to fail in `contracts/contracts/stardust-settlement/src/order_utils.rs:125` for any token/token orders. That failure applies to existing open token/token orders as well, which can no longer be executed or cancelled.

Recommendation

Disallow such a configuration change or allow the change to only be made when there are no existing token/token orders pending.

Status: Acknowledged

The Random Earth team states that setting the `nft_only` field from `false` to `true` is equivalent to cancelling all open non-NFT orders. The team also states that the intention is just to go from `true` to `false`, and not the other way around.

16. Use Addr type instead of String

Severity: Informational

Throughout the codebase, the `String` data type is used for storing the addresses instead of `Addr`. It is best practice to validate any addresses passed as strings and use the validated `Addr` type in memory/storage. Storing unvalidated addresses might cause errors when those addresses are used at a later time. `Addr` also provides clearer error messages if invalid addresses are passed.

Recommendation

Consider using `Addr` instead of `String`.

Status: Acknowledged