AUDIT REPORT

TerraForm Labs
Alliance NFT Collection

Prepared by SCV-Security

On 18th November 2023

Table of Contents

Introduction

SCV has been engaged by TerraForm Labs to conduct a comprehensive security review with the goal of identifying potential security threats and vulnerabilities within the codebase. The purpose of this audit is to evaluate the security posture of the codebase and provide actionable recommendations to mitigate any identified risks. This report presents an overview of the findings from our security audit, outlining areas of concern and proposing effective measures to enhance the codebase's security.

Scope Functionality

This audit evaluates the Alliance NFT Collection contracts, designed to incentivize Game Of Alliance testers through rewards. The contracts facilitate staking a specified token quantity in the Alliance module, enabling a portion of Luna's inflation to be directed towards the NFT collection.

Submitted Codebase

locked-astroport-vault	
Repository	https://github.com/SCV-Security/alliance-nft-collection
Commit	`c7718480412f7df5390794c219ab580a307dc8a5`
Branch	`main`

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 TerraForm Labs. Testing includes, but is not limited to, the following:

- Understanding the application and its functionality 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

This section provides an evaluation of specific criteria aspects as described below:

- **Documentation:** Evaluating the presence and comprehensiveness of publicly available or provided explanatory information, diagram flowcharts, comments, and supporting documents to enhance code understanding.
- **Coverage:** Evaluating whether the code adequately addresses all necessary cases and scenarios, ensuring that the intended functionality or requirements are sufficiently covered.
- **Readability:** Assessing how easily the code can be understood and maintained, considering factors such as code structure, naming conventions, and overall organisation.
- **Complexity:** Evaluating the complexity of the code, including factors such as, number of lines, conditional statements, and nested structures.

The status of each criteria is categorised as either **SUFFICIENT** or **NOT-SUFFICIENT** based on the audit assessment. This categorisation provides insights to identify areas that may require further attention and improvement.

Criteria	Status	Notes
Documentation	**SUFFICIENT**	N/A
Coverage	**SUFFICIENT**	Testing coverage is considered sufficient, although there is room for improvement as the current coverage only extends to 61.79% coverage, 325/526 out of the code.
Readability	**SUFFICIENT**	The codebase had good readability overall and utilised many Rust and CosmWasm best practices.
Complexity	**SUFFICIENT**	N/A

Findings Summary

Summary Title	Risk Impact	Status
Undelegation ignores collected rewards	SEVERE	PENDING
Contract does not implement withdraw message	SEVERE	PENDING
CLAIM_REWARD_ERROR_REPLY_ID not handled	MODERATE	PENDING
Parent NFT messages may allow for unexpected actions	MODERATE	PENDING
Undelegation does not track undelegation request count	MODERATE	PENDING
reply_on_instantiate builds event but does not emit it	INFO	PENDING
Misleading error in is_minting_period	INFO	PENDING
Instantiate parameters lacking validation	INFO	PENDING
Implement a two-step ownership transfer	INFO	PENDING
Update config not implemented	INFO	PENDING

Findings Technical Details

1. Undelegation ignores collected rewards

RISK IMPACT: SEVERE	STATUS: PENDING

Description

The `try_alliance_undelegate` function in `contracts/alliance-nft-collection/src/contract/execute.rs:177` prepares undelegation messages to be dispatched to the Alliance module. When the Alliance module prepares to send the undelegation requests to the target chain, it first executes the `ClaimDelegationRewards` function. This function will claim delegation rewards and send them back to the contract.

Currently the contract only handles rewards with the `update_reward_callback` function, which updates the `REWARD_BALANCE` with the newly received rewards. But this rewards update will not occur during the unbonding process because the contract is not set to receive rewards during that function execution.

This issue is also present for the `try_alliance_redelegate` function of the alliance-nft-collection contract in `contracts/alliance-nft-collection/src/contract/execute.rs:198`.

Recommendation

We recommend handling the rewards received by the alliance module during the execution of the `try_alliance_undelegate` and `try_alliance_redelegate` functions.

2. Contract does not implement withdraw message

RISK IMPACT: SEVERE	STATUS: PENDING

Description

The `alliance-nft-collection` contract doesn't allow the owner to withdraw the unstaked alliance tokens once the number of NFTs is zero resulting in a loss of staking rewards and alliance tokens.

Recommendation

We recommend implementing a withdrawal message to facilitate the withdrawal of tokens from the `alliance-nft-collection` contract.

3. CLAIM_REWARD_ERROR_REPLY_ID not handled

RISK IMPACT: MODERATE	STATUS: PENDING

Description

The `try_alliance_claim_rewards` in `contracts/alliance-nft-collection/src/contract/execute.rs:91` a submessage is specified with a `reply_on_error` type, but there is no logic in the reply handler to properly let the error pass as stated in the comments . This means that if an error is encountered it will not be ignored as the comments state

Recommendation

We recommend implementing a reply handler for the `CLAIM_REWARD_ERROR_REPLY_ID` reply that passes the error state successfully.

4. Parent NFT messages may allow for unexpected actions

RISK IMPACT: MODERATE	STATUS: PENDING

Description

The `alliance-nft-collection` contract fully exposes the parent CW-721 contract's execute messages. This is likely intended to allow for transfers and normal NFT operations. But fully exposing all messages is problematic because it may allow for unexpected actions that can circumvent the contract's state changes. For example, a message may be sent directly to the parent that burns the NFT or the owner can mint NFTs that are outside of the reward collections logic. This action could be successfully performed but the contract would have an inconsistent state as `NUM_ACTIVE_NFTS` was not updated.

Recommendation

We recommend assessing only the necessary messages that can be passed to the parent contract but do not cause any inconsistent states within the contract.

5. Undelegation does not track undelegation request count

RISK IMPACT: MODERATE	STATUS: PENDING

Description

The `try_alliance_undelegate` function in `contracts/alliance-nft-collection/src/contract/execute.rs:164` does not track the quantity of unbonding requests. This may cause a situation where the contract cannot make any additional unbonding requests because its maximum of 7 unbonding requests are already submitted.

For example, if the contract issues 7 small unbonding requests over a 2 day period and then the next week a situation arises where contract needs to issue a high priority unbonding request it will fail until the full unbonding period is reached for the first request because a single delegator can only have 7 concurrent requests.

We classify this as a moderate severity because only the owner can issue unbonding requests.

Recommendation

We recommend batching the unbonding requests with a period that is the unbonding period divided by 7. This way the unbonding requests are dispatched at set intervals.

6. `reply_on_instantiate` builds event but does not emit it

RISK IMPACT: INFO	STATUS: PENDING

Description

The `instantiate` function in `contracts/alliance-nft-minter/src/contract/instantiate.rs:65` constructs an event to detail the relevant reply information but this event is not emitted in the functions response.

Recommendation

We recommend adding the constructed event to the response.

7. Misleading error in `is_minting_period`

RISK IMPACT: INFO	STATUS: PENDING

Description

The `is_minting_period` function in `packages/alliance-nft-packages/src/state.rs:69` returns a misleading error. If the `current_time` is less than the `start_time`, the function will return the `MintTimeCompleted` error. This error may mislead users.

Recommendation

We recommend defining a specific error message for this case.

8. Instantiate parameters lacking validation

RISK IMPACT: INFO	STATUS: PENDING

Description

The instantiate function in `contracts/alliance-nft-collection/src/contract/instantiate.rs:32` is lacking validations for the following parameters:

- `alliance-nft-minter/src/contract/instantiate.rs:36` `msg.dao_treasury_address` is not validated as address
- `alliance-nft-minter/src/contract/instantiate.rs:39` `msg.mint_start_time` is not checked to be less then `msg.mint_end_time`

This can lead to misconfigurations in the contract during instantiations.

Recommendation

We recommend validating the parameters mentioned above in the `instantiate` function.

9. Implement a two-step ownership transfer

RISK IMPACT: INFO	STATUS: PENDING

Description

It is a good practice to implement a two-step ownership transfer. The two-step process provides additional security and control during the migration.

Using a two-step ownership transfer mechanism helps provide a window of opportunity for the current owner to cancel the transfer if they did not intend to initiate it or if there were any unintended actions.

Recommendation

We recommend modifying the code to implement two-step ownership transfer.

10. Update config not implemented

RISK IMPACT: INFO	STATUS: PENDING

Description

The current design of the contracts does not include an implementation for an update config message. Consequently, this design choice restricts the owner's ability to modify any of the contract configurations once it has been deployed and is operational.

Recommendation

We recommend incorporating an `update_config` function into the contracts designs. This addition will enable the contract owner to modify configurations post-deployment, ensuring flexibility and adaptability in response to evolving requirements or unforeseen scenarios.

Document Control

Version	Date	Notes
-	13th November 2023	Security audit commencement date.
0.1	18th November 2023	Initial report with identified findings delivered.
0.5		Fixes remediations implemented and reviewed.
1.0		Audit completed, final report delivered.

Appendices

A. Appendix – Risk assessment methodology

SCV-Security employs a risk assessment methodology to evaluate vulnerabilities and identified issues. This approach involves the analysis of both the LIKELIHOOD of a security incident occurring and the potential IMPACT if such an incident were to happen. For each vulnerability, SCV-Security calculates a risk level on a scale of 5 to 1, where 5 denotes the highest likelihood or impact. Consequently, an overall risk level is derived from combining these two factors, resulting in a value from 10 to 1, with 10 signifying the most elevated level of security risk

Risk Level	Range
CRITICAL	10
SEVERE	From 9 to 8
MODERATE	From 7 to 6
LOW	From 5 to 4
INFORMATIONAL	From 3 to 1

LIKELIHOOD and **IMPACT** would be individually assessed based on the below:

Rate	LIKELIHOOD	IMPACT
5	**Extremely Likely**	Could result in severe and irreparable consequences.
4	**Likely**	May lead to substantial impact or loss.
3	**Possible**	Could cause partial impact or loss on a wide scale.
2	**Unlikely**	Might cause temporary disruptions or losses.
1	**Rare**	Could have minimal or negligible impact.

B. Appendix – Report Disclaimer

This report should not be regarded as an "endorsement" or "disapproval" of any specific project or team. These reports do not indicate the economics or value of any "product" or "asset" created by a team or project that engages SCV-Security for a security review. The audit report does not make any statements or warranties about the code's utility, safety, suitability of the business model, regulatory compliance of the business model, or any other claims regarding the fitness of the implementation for its purpose or its bug-free status. The audit documentation is intended for discussion purposes only. The content of this audit report is provided "as is," without representations and warranties of any kind, and SCV-Security disclaims any liability for damages arising from or in connection with this audit report. Copyright of this report remains with SCV-Security.

THANK YOU FOR CHOOSING

SCV SECURITY

scv.services

contact@scv.services