Audit Report

ICS721

v1.0

January 27, 2023

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 Public Awesome to perform a security audit of ICS721.

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 behavior.
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/public-awesome/ics721

Commit hash: `52852e07446c91983a91495e85103370f49814a7`

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

This audit covers a CosmWasm implementation of the ICS721 specification. It allows NFTs to be moved between IBC-compatible blockchains.

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**.

Note that audits are an important step to improving 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 in a security audit and vice versa.

Code Quality Criteria

The auditor team assesses the codebase’s code quality criteria as follows:

Criteria	Status	Comment
Code complexity	**Medium**	-
Code readability and clarity	**Medium-High**	-
Level of documentation	**High**	Detailed documentation was provided including a security posture review. In addition, the code comments were descriptive and accurately reflected the functionality.
Test coverage	**High**	The contracts included CosmWasm unit testing, Cosmos SDK unit testing, and end-to-end adversarial IBC testing.

Summary of Findings

No	Description	Severity	Status
1	Instantiator may introduce misconfiguration by providing non-compliant `CW721_CODE_ID`	Minor	Acknowledged
2	Instantiator may introduce misconfiguration by providing non-compliant `proxy code_id`	Minor	Acknowledged
3	Remove duplicate sender validation	Informational	Resolved
4	IBC timeout is not validated	Informational	Acknowledged
5	Nondescript storage keys may make storage state difficult to maintain	Informational	Acknowledged
6	`NonFungibleTokenPacketData` diverges from ICS721 standard	Informational	Resolved
7	Spelling errors found in codebase	Informational	Partially Resolved

Detailed Findings

1. Instantiator may introduce misconfiguration by providing non-compliant `CW721_CODE_ID`

Severity: Minor

The `instantiate` function in `contracts/cw-ics721-bridge/src/contract.rs:26` does check to ensure the `code_id` specified belongs to a `cw721_base` contract. This means that a `code_id` of a contract that is not compliant with the cw721 standard may be set, which would introduce unexpected errors. The contract does not determine if the `CW721_CODE_ID` is compliant until the `execute_do_instantiate_and_mint` is invoked when an IBC packet is received. There is no functionality to change this once the error is introduced.

We classify this issue as informational since only the instantiator can cause it.

Recommendation

We recommend introducing validation to ensure that the `CW721_CODE_ID` provided during the instantiation is a compliant `cw721_base` contract. The `cw721_base` contract does not include any queries that identify it as the correct contract. One method of performing this validation could be checking to ensure that the provided `code_id` supplied a `mint` execute entry point.

Status: Acknowledged

The client stated that due to current limitations in CosmWasm this issue is not resolvable. The client created a test that handles a non-spec-compliant instantiation and the contract is shown to handle this scenario.

2. Instantiator may introduce misconfiguration by providing non-compliant proxy `code_id`

Severity: Minor

The `instantiate` function in `contracts/cw-ics721-bridge/src/contract.rs:26` does not check to ensure that the proxy `code_id` specified is a compliant `cw721-proxy` contract. The functionality for the `INSTANTIATE_PROXY_REPLY_ID` reply is generic contract instantiation logic which does not confirm that the contract is compliant. While it is unlikely that the instantiator of the contract would unknowingly introduce a misconfiguration by providing an invalid proxy, the impact of this would be that NFTs could be sent from other chains to this bridge and debt vouchers could be minted, but they would be unredeemable if the proxy contract did not support the correct interface to allow for NFTs to be sent back to their origin chain. This could be corrected by a migration to a compliant proxy contract.

Recommendation

We recommend introducing validation to ensure that the proxy `code_id` specified is a compliant `cw721-proxy`.

Status: Acknowledged

The client stated that due to current limitations in CosmWasm this issue is not resolvable.

3. Remove duplicate sender validation

Severity: Informational

The `execute_callback` function performs a sender validation in `contracts/cw-ics721-bridge/src/contract.rs:78`. This validation is duplicated in each callback message function.

The following functions contain unnecessary duplicate validation functions:

- `contracts/cw-ics721-bridge/src/contract.rs:245`
- `contracts/cw-ics721-bridge/src/contract.rs:272`
- `contracts/cw-ics721-bridge/src/contract.rs:318`
- `contracts/cw-ics721-bridge/src/contract.rs:387`

Recommendation

We recommend removing these duplicate sender validations.

Status: Resolved

4. IBC timeout is not validated

Severity: Informational

In the `do_receive_nft` function in `contracts/cw-ics721-bridge/src/contract.rs:219`, `msg.timeout` is a variable expected to be passed from the front end. However, there is no validation performed on it. The client could supply any arbitrary timeout, ranging from unreasonably small to extremely large, or not passing any at all.

Recommendation

We recommend setting a reasonable default timeout in case the front end does not send any to prevent the function from potentially running indefinitely if the packet cannot be sent. Also, we recommend setting an upper and lower boundary timeout value to ensure the contract behaves in a reasonable manner.

Status: Acknowledged

The client states that they intend to implement timeout validation in the frontend, since arbitrary timeouts in the contract may be useful in the future.

5. Nondescript storage keys may make storage state difficult to maintain

Severity: Informational

Multiple constants defined in `contracts/cw-ics721-bridge/src/state.rs:7-27` use a single letter, as a storage key. This contradicts best programming practices as the keys are not descriptive, and could lead to naming conflicts in the future. Additionally, the developer might find it hard in the future if they introduce new states or need to group similar values together, as new states can only use the next character in the sequence.

Recommendation

We recommend using more descriptive keys.

Status: Acknowledged

6. `NonFungibleTokenPacketData` diverges from ICS721 standard

Severity: Informational

The ICS721 standard defines the `NonFungibleTokenPacketData` type. It specifies the class ID, class URI, token IDs, token URIs, token data, sender address, and receiver address. In `contracts/cw-ics721-bridge/src/contract.rs:206`, the `do_receive_nft` specifies `ibc_message` without the required `tokenData` field.

Recommendation

We recommend including the `tokenData` field in `NonFungibleTokenPacketData` or adding an explanation why the implementation diverges from the ICS721 standard.

Status: Resolved

7. Spelling errors found in codebase

Severity: Informational

The codebase contains several spelling errors that are listed below:

- `contracts/cw-ics721-bridge/src/contract.rs:342` - `"ICS771"`
- `contracts/cw-ics721-bridge/src/error.rs:49` - `"TokenInfoLenMissmatch"`
- `contracts/cw-ics721-bridge/src/ibc.rs:25` - `"succeded"`

Recommendation

We recommend correcting these spelling errors.

Status: Partially Resolved

The first spelling error has been corrected, but the following spelling errors have not been corrected.