SMART CONTRACTS REVIEW

zokyo

November 7th 2023 | v. 1.0

Security Audit Score

PASS

Zokyo Security has concluded that this smart contract passed a security audit.

ZOKYO AUDIT SCORING GALAXY GAMES

1. Severity of Issues:
    - Critical: Direct, immediate risks to funds or the integrity of the contract. Typically, these would have a very high weight.
    - High: Important issues that can compromise the contract in certain scenarios.
    - Medium: Issues that might not pose immediate threats but represent significant deviations from best practices.
    - Low: Smaller issues that might not pose security risks but are still noteworthy.
    - Informational: Generally, observations or suggestions that don't point to vulnerabilities but can be improvements or best practices.
2. Test Coverage: The percentage of the codebase that's covered by tests. High test coverage often suggests thorough testing practices and can increase the score.
3. Code Quality: This is more subjective, but contracts that follow best practices, are well-commented, and show good organization might receive higher scores.
4. Documentation: Comprehensive and clear documentation might improve the score, as it shows thoroughness.
5. Consistency: Consistency in coding patterns, naming, etc., can also factor into the score.
6. Response to Identified Issues: Some audits might consider how quickly and effectively the team responds to identified issues.

SCORING CALCULATION:

Let's assume each issue has a weight:
- Critical: -30 points
- High: -20 points
- Medium: -10 points
- Low: -5 points
- Informational: -1 point

Starting with a perfect score of 100:
- 0 Critical issues: 0 points deducted
- 0 High issues: 0 points deducted
- 0 Medium issues: 0 points deducted
- 1 Low issue: 1 resolved = 0 points deducted
- 4 Informational issues: 4 resolved = 0 points deducted

Thus, the score is 100

TECHNICAL SUMMARY

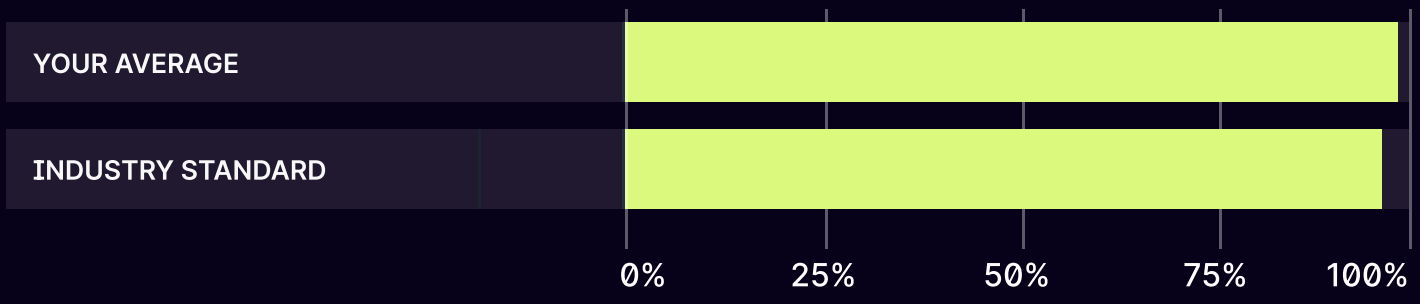
This document outlines the overall security of the Galaxy Games smart contract/s evaluated by the Zokyo Security team.

The scope of this audit was to analyze and document the Galaxy Games smart contract/s codebase for quality, security, and correctness.

Contract Status

There were 0 critical issues found during the review. (See Complete Analysis)

Testable Code



100% of the code is testable, which is above the industry standard of 95%.

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract/s but rather limited to an assessment of the logic and implementation. In order to ensure a secure contract that can withstand the Ethereum network's fast-paced and rapidly changing environment, we recommend that the Galaxy Games team put in place a bug bounty program to encourage further active analysis of the smart contract/s.

Table of Contents

AUDITING STRATEGY AND TECHNIQUES APPLIED

The source code of the smart contract was taken from the Galaxy Games repository:
Repo: https://github.com/Blockchain-Army/gaga

Last commit - a0948b2b411e95284d5c7d918604fabe47e0eaad

Within the scope of this audit, the team of auditors reviewed the following contract(s):

- GalaxyGames.sol

During the audit, Zokyo Security ensured that the contract:

- Implements and adheres to the existing standards appropriately and effectively;
- The documentation and code comments match the logic and behavior;
- Distributes tokens in a manner that matches calculations;
- Follows best practices, efficiently using resources without unnecessary waste;
- Uses methods safe from reentrance attacks;
- Is not affected by the most recent vulnerabilities;
- Meets best practices in code readability, etc.

Zokyo Security has followed best practices and industry-standard techniques to verify the implementation of Galaxy Games smart contract/s. To do so, the code was reviewed line by line by our smart contract developers, who documented even minor issues as they were discovered. Part of this work includes writing a test suite using the Hardhat testing framework. In summary, our strategies consist largely of manual collaboration between multiple team members at each stage of the review:

01	Due diligence in assessing the overall code quality of the codebase.
02	Cross-comparison with other, similar smart contract/s by industry leaders.
03	Testing contract/s logic against common and uncommon attack vectors.
04	Thorough manual review of the codebase line by line.

Executive Summary

The Zokyo team has performed a security audit of the provided codebase. The contracts submitted for auditing are well-crafted and organized. Detailed findings from the audit process are outlined in the "Complete Analysis" section.

Galaxy Games is a cross-chain utility token designed for the gaming ecosystem, implementing advanced security standards and controlled supply management. It’s built on LayerZero’s OFT standard, it enables seamless cross-chain operations while maintaining robust security measures and enhanced pool protection mechanisms.

STRUCTURE AND ORGANIZATION OF THE DOCUMENT

For the ease of navigation, the following sections are arranged from the most to the least critical ones. Issues are tagged as “Resolved” or “Unresolved” or “Acknowledged” depending on whether they have been fixed or addressed. Acknowledged means that the issue was sent to the Galaxy Games team and the Galaxy Games team is aware of it, but they have chosen to not solve it. The issues that are tagged as “Verified” contain unclear or suspicious functionality that either needs explanation from the Client or remains disregarded by the Client. Furthermore, the severity of each issue is written as assessed by the risk of exploitation or other unexpected or otherwise unsafe behavior:

Critical

The issue affects the contract in such a way that funds may be lost, allocated incorrectly, or otherwise result in a significant loss.

High

The issue affects the ability of the contract to compile or operate in a significant way.

Medium

The issue affects the ability of the contract to operate in a way that doesn’t significantly hinder its behavior.

Low

The issue has minimal impact on the contract’s ability to operate.

Informational

The issue has no impact on the contract’s ability to operate.

COMPLETE ANALYSIS

FINDINGS SUMMARY

#	Title	Risk	Status
1	Lack of a restriction on the recoverERC20() function in the pause	Low	Resolved
2	Lack of return value check	Informational	Resolved
3	Floating pragma	Informational	Resolved
4	Duplicated zero address check	Informational	Resolved
5	Unnecessary use of nonReentrant modifier	Informational	Resolved

LOW-1 | RESOLVED

Lack of a restriction on the recoverERC20() function in the pause status

Regarding the documentation, the recoverERC20() function is not supposed to be called while paused.
However, the function is missing a whenNotPaused modifier and it makes the function to be called in the pause status.

Recommendation:
Add a check to make sure that the return value is true and if not, revert the function.

INFORMATIONAL-1 | RESOLVED

Lack of return value check

The recoverERC20() function is missing a check for the return value of the token's transfer.

Recommendation:
Add a check to make sure that the return value is true and if not, revert the function.

INFORMATIONAL-2 | RESOLVED

Floating pragma

The contract uses pragma solidity ^0.8.27.
This may result in the contracts being deployed using the wrong pragma version, which is different from the one they were tested with.

Recommendation:
It is recommended to lock the pragma version.

INFORMATIONAL-3 | RESOLVED

Duplicated zero address check

The constructor() has a zero address check for the _delegate variable passed.
However, _delegate value is already checked in the both Ownable and OAppCore contracts which the GalaxyGames contract inherits from.
(https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/access/Ownable.sol#L40
https://github.com/LayerZero-Labs/devtools/blob/main/packages/oapp-evm/contracts/oapp/OAppCore.sol#L29)

Recommendation:
It is recommended to remove the zero address check for the _delegate variable passed.

INFORMATIONAL-4 | RESOLVED

Unnecessary use of nonReentrant modifier

The burn() and recoverERC20() functions have the nonReentrant modifiers but the functions could not have reentrancy attacks because the functions can only be called by the owner and _burn() function doesn't have an implementation to cause the reetrancy.
Unnecessary use of modifiers will result in unnecessary gas consumption.

Recommendation:
Remove the nonReentrant modifier in the functions and delete the ReentrancyGuard import in the contract.

	GalaxyGames.sol
Re-entrancy	Pass
Access Management Hierarchy	Pass
Arithmetic Over/Under Flows	Pass
Unexpected Ether	Pass
Delegatecall	Pass
Default Public Visibility	Pass
Hidden Malicious Code	Pass
Entropy Illusion (Lack of Randomness)	Pass
External Contract Referencing	Pass
Short Address/ Parameter Attack	Pass
Unchecked CALL Return Values	Pass
Race Conditions / Front Running	Pass
General Denial Of Service (DOS)	Pass
Uninitialized Storage Pointers	Pass
Floating Points and Precision	Pass
Tx.Origin Authentication	Pass
Signatures Replay	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass

CODE COVERAGE AND TEST RESULTS FOR ALL FILES

Tests written by Zokyo Security

As a part of our work assisting Galaxy Games in verifying the correctness of their contract/s code, our team was responsible for writing integration tests using the Hardhat testing framework.
The tests were based on the functionality of the code, as well as a review of the Galaxy Games contract/s requirements for details about issuance amounts and how the system handles these.

```
GalaxyGames
  Deployment
✓ Should set the right owner (97ms)
✓ Could not pass zero addresses (65ms)
  Burn
✓ Owner should burn
✓ Owner should not burn until the total supply is getting less than minimum
✓ Only owner could burn
✓ Only owner could burn when not paused
✓ Owner could burn non zero amount
  Recover
✓ Owner should recover
✓ Only owner could recover
✓ Owner could not recover native token
✓ Owner could recover non zero amount
✓ Owner could recover only the amount less than the balance
✓ Owner could not recover non-standard token
  Transfer
✓ Could not transfer to the token contract
  Pause
✓ Owner should pause/unpause
✓ Only owner could pause

  16 passing (341ms)
```

The resulting code coverage (i.e., the ratio of tests-to-code) is as follows:

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	%UNCOVERED LINES
GalaxyGames.sol	100	100	100	100	
All Files	**100**	**100**	**100**	**100**	

We are grateful for the opportunity to work with the Galaxy Games team.

The statements made in this document should not be interpreted as an investment or legal advice, nor should its authors be held accountable for the decisions made based on them.

Zokyo Security recommends the Galaxy Games team put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

zokyo