



LayerK

SMART CONTRACTS REVIEW

 zokyo

November 8th 2024 | v. 1.0

Security Audit Score

PASS

Zokyo Security has concluded that
these smart contracts passed a
security audit.



ZOKYO AUDIT SCORING LAYERK

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.

HYPOTHETICAL 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 issues: 1 acknowledged = -2 points deducted
- 2 Informational issues: 2 acknowledged = -1 points deducted

Thus, $100 - 2 \cdot -1 = 97$

TECHNICAL SUMMARY

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

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

Contract Status



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

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 Arbitrum network's fast-paced and rapidly changing environment, we recommend that the LayerK 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	5
Executive Summary	7
Structure and Organization of the Document	8
Complete Analysis	9

AUDITING STRATEGY AND TECHNIQUES APPLIED

The source code of the smart contract was taken from the LayerK repository:

Repo: <https://github.com/LayerK/LayerK-Blockchain>

Last commit - a9bf86d7dc6257d71737e933faf018cb21ec24fc

<https://arbiscan.io/address/0x5c9b04a46641ab50fe9ca27429071fc4aecb4648#code>

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

- Arbitrum Nitro with contracts + token bridge

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 LayerK 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. 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.

03

Thorough manual review of the codebase line by line.

02

Cross-comparison with other, similar smart contract/s by industry leaders.

Executive Summary

The security audit of LayerK's Arbitrum Nitro code and Token Bridge contract revealed no critical vulnerabilities. A low severity finding regarding accidental ownership renunciation and some informational findings related to code readability and optimization were identified.

Overall, the codebase appears secure with minor recommendations for improvement.



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 LayerK team and the LayerK 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	Renounce Ownership	Low	Acknowledged
2	Too many digits	Informational	Acknowledged
3	Unused functions	Informational	Acknowledged

Renounce Ownership

Where: LayerK.sol

The renounceOwnership function can be called accidentally by the admin leading to the immediate renouncement of ownership to zero address after which any onlyOwner functions will not be callable which can be risky.

Recommendation:

It is advised that the Owner cannot call renounceOwnership without first transferring ownership to a different address. Additionally, if a multi-signature wallet is utilized, executing the renounceOwnership method for two or more users should be confirmed. Alternatively, the Renounce Ownership functionality can be disabled by overriding it.

Too many digits

Where: LayerK.sol

Literals with many digits are difficult to read and review. Use scientific notation or suffixes to make the code more readable.

Recommendation:

use a `gwei` suffix i.e.: `_mint(msg.sender , 1gwei * 10** decimals());`

Unused functions

Where LayerK.sol

Included libraries contain functions that aren't used in the code

Recommendation:

to make a contract smaller and use less gas, consider including only required functions in the code

Arbitrum Nitro with contracts + token bridge	
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	Pass
Return Values	
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

We are grateful for the opportunity to work with the LayerK 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 LayerK team put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

