COSD Token

AUDIT REPORT

Version 1.0.0

Serial No. 20220623000420 23

Presented by Fairyproof

June 23, 2022

01. Introduction

This document includes the results of the audit performed by the Fairyproof team on the COSD Token issuance and NFT issuance functions.

Audit Start Time:

June 22, 2022

Audit End Time:

June 23, 2022

Audited Source Files' Onchain Address:

- https://bscscan.com/address/0x07af6b7e729e152f406fe1982d9b3360b2332941#code
- https://bscscan.com/address/0xdad1f1b7b2483eeb1e08fc535d026f1400d5c83f#code

The goal of this audit is to review COSD's solidity implementation for its token issuance and NFT issuance functions, study potential security vulnerabilities, its general design and architecture, and uncover bugs that could compromise the software in production.

We make observations on specific areas of the code that present concrete problems, as well as general observations that traverse the entire codebase horizontally, which could improve its quality as a whole.

This audit only applies to the specified code, software or any materials supplied by the COSD team for specified versions. Whenever the code, software, materials, settings, environment etc is changed, the comments of this audit will no longer apply.

— Disclaimer

Note that as of the date of publishing, the contents of this report reflect the current understanding of known security patterns and state of the art regarding system security. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk.

The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. If the audited source files are smart contract files, risks or issues introduced by using data feeds from offchain sources are not extended by this review either.

Given the size of the project, the findings detailed here are not to be considered exhaustive, and further testing and audit is recommended after the issues covered are fixed.

To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services.

FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

— Methodology

The above files' code was studied in detail in order to acquire a clear impression of how the its specifications were implemented. The codebase was then subject to deep analysis and scrutiny, resulting in a series of observations. The problems and their potential solutions are discussed in this document and, whenever possible, we identify common sources for such problems and comment on them as well.

The Fairyproof auditing process follows a routine series of steps:

1. Code review that includes the following
   i. Review of the specifications, sources, and instructions provided to Fairyproof to make sure we understand the size, scope, and functionality of the project's source code.
   ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
   iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Fairyproof describe.
2. Testing and automated analysis that includes the following:
   i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run the test cases.
   ii. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the source code to improve maintainability, security, and control based on the established industry and academic practices, recommendations, and research.

— Structure of the document

This report contains a list of issues and comments on all the above source files. Each issue is assigned a severity level based on the potential impact of the issue and recommendations to fix it, if applicable. For ease of navigation, an index by topic and another by severity are both provided at the beginning of the report.

— Documentation

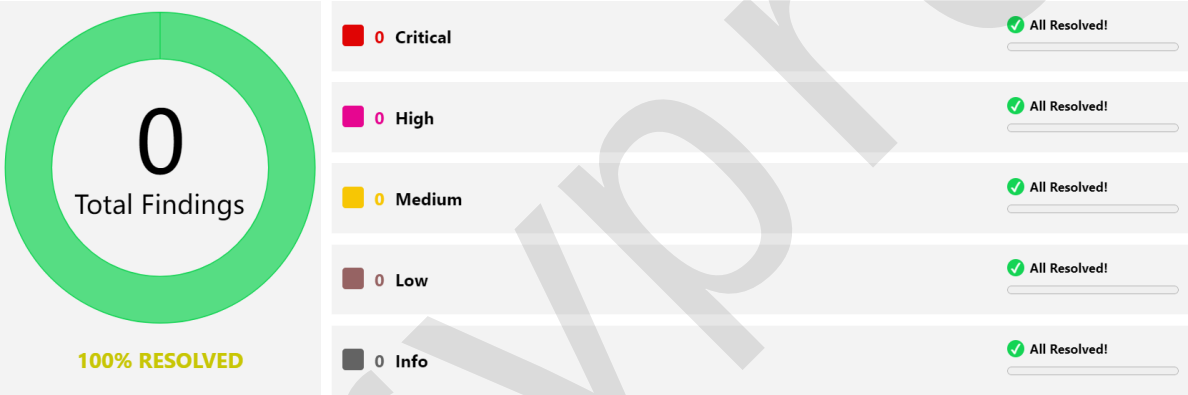
For this audit, we used the following sources of truth about how the token issuance and NFT issuance functions should work:

Contract Source Code

This was considered the specification, and when discrepancies arose with the actual code behavior, we consulted with the COSD team or reported an issue.

— Comments from Auditor

Serial Number	Auditor	Audit Time	Result
2022062300042023	Fairyproof Security Team	2022.06.22 - 2022.06.23	Passed



Summary:

The Fairyproof security team used its auto analysis tools and manual work to audit the project. During the audit, no risks were uncovered.

02. About Fairyproof

Fairyproof is a leading technology firm in the blockchain industry, providing consulting and security audits for organizations. Fairyproof has developed industry security standards for designing and deploying blockchain applications.

03. Major functions of audited code

The audited code mainly implements the following functions:

1 Issuance of ERC-777 Token

- Token Standard: ERC-777
- Token Name: Chess Of Stars Dividents
- Token Symbol: COSD
- Decimals: 18
- Total Supply: 1,000,000,000
- Burn on Transaction: No
- Additional Mintage: No

Note: this token is compatible with ERC-20. The ERC1820 Registry at 0x1820a4B7618BdE71Dce8cdc73aAB6C95905faD24 can be used to implement a callback function that will called before and after a transfer happens.

This token itself doesn't have any re-entrancy issues of risks. But when another application interacts with this ERC-777 token, it is necessary to implement measures in that application to prevent re-entrancy attacks that might be introduced by this token.

2 Issuance of NFT

- NFT Standard: ERC-721
- NFT Name: Chess Of Stars
- NFT Symbol: COS
- Max Supply: Uncapped

This implementation directly uses `openzeppelin`'s ERC-721 implementation. In this implementation, two functions can be used to transfer an NFT: : `transferFrom` and `safeTransferFrom`. For the latter function, if the receiver is a contract, the contract's `onERC721Received` function will be called to verify whether or not an NFT is received.

Therefore the implementer needs to be aware of the following two points:

- If the receiver of an NFT is a contract, that contract's `onERC721Received` function needs to be implemented.
- When this NFT interacts with another application, since the `onERC721Received` function is a callback function it may introduce re-entrancy issues or risks to that application.

In this NFT contract, there are two admins: `DEFAULT_ADMIN_ROLE` and `GAME_SERVER_ROLE`. Both can mint or burn NFTs.

`DEFAULT_ADMIN_ROLE` can set `GAME_SERVER_ROLE`.

Both of the admins' private keys need to be safely and securely managed.

04. Coverage of issues

The issues that the Fairyproof team covered when conducting the audit include but are not limited to the following ones:

- Re-entrancy Attack
- Replay Attack
- Reordering Attack
- Miner's Advantage
- Rollback Attack
- DDoS Attack
- Transaction Ordering Attack
- Race Condition
- Access Control
- Integer Overflow/Underflow
- Timestamp Attack
- Gas Consumption
- Inappropriate Callback Function
- Function Visibility
- Implementation Vulnerability
- Uninitialized Storage Pointer
- Arithmetic Precision
- Tx.origin
- Fake Deposit
- Shadow Variable
- Design Vulnerability
- Token Issuance
- Admin Rights
- Inappropriate Proxy Design
- Inappropriate Use of Slots
- Asset Security
- Contract Upgrade/Migration
- Code Improvement
- Misc

05. Severity level reference

Every issue in this report was assigned a severity level from the following:

Critical severity issues need to be fixed as soon as possible.

High severity issues will probably bring problems and should be fixed.

Medium severity issues could potentially bring problems and should eventually be fixed.

Low severity issues are minor details and warnings that can remain unfixed but would be better fixed at some point in the future.

Informational is not an issue or risk but a suggestion for code improvement.

06. Major areas that need attention

Based on the provided source code the Fairyproof team focused on the possible issues and risks related to the following functions or areas.

- Function Implementation

We checked whether or not the functions were correctly implemented.

We didn't find issues or risks in these functions or areas at the time of writing.

- Integer Overflow/Underflow

We checked all the code sections, which had arithmetic operations and might introduce integer overflow or underflow if no safe libraries were used. All of them used safe libraries.

We didn't find issues or risks in these functions or areas at the time of writing.

- Access Control

We checked each of the functions that could modify a state, especially those functions that could only be accessed by "owner".

We didn't find issues or risks in these functions or areas at the time of writing.

- Token Issuance/Exchange

We checked whether or not the contract files could mint tokens at will.

We didn't find issues or risks in these functions or areas at the time of writing.

- State Update

We checked some key state variables which should only be set at initialization.

We didn't find issues or risks in these functions or areas at the time of writing.

- Asset Security

We checked whether or not all the functions that transfer assets were safely hanlded.

We didn't find issues or risks in these functions or areas at the time of writing.

- Contract Migration/Upgrade

We checked whether or not the contract files would introduce issues or risks associated with contract migration/upgrade.

We didn't find issues or risks in these functions or areas at the time of writing.

- Miscellaneous

We didn't find issues or risks in other functions or areas at the time of writing.

07. List of issues by severity

- N/A

08. Issue descriptions

- N/A

09. Recommendations to enhance the overall security

We list some recommendations in this section. They are not mandatory but will enhance the overall security of the system if they are adopted.

- Consider safely and securely managing the private keys of the two admins that have access control to the NFT contract's mint and burn functions and transferring them to a multi-sig wallet or DAO when necessary.
- When another application interacts with the ERC-777 token implemented in this contract, consider implementing necessary measures in that application to prevent re-entrancy attacks that might be introduced by this ERC-777 token.

FAIRYPROOF

https://medium.com/@FairyproofT

https://twitter.com/FairyproofT

https://www.linkedin.com/company/fairyproof-tech

https://t.me/Fairyproof_tech

Reddit: https://www.reddit.com/user/FairyproofTech