QuillAudits

# Audit Report
# April, 2022

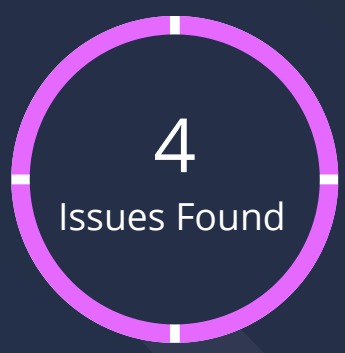
For

DRUNK ROBOTS

# Table of Content

# Executive Summary

| | |
|---|---|
| **Project Name** | Metal Token.sol by Drunk Robots |
| **Overview** | It is used to purchase NFTs and in-game items. It is used to participate in the Farm function It is used for prizes at major tournaments |
| **Timeline** | April 13, 2022 - April 15, 2022 |
| **Method** | Manual Review, Functional Testing, Automated Testing, etc. |
| **Scope of Audit** | The scope of this audit was to analyse Metal smart contract's codebase for quality, security, and correctness. |
| **Contract Address** | 0x200C234721b5e549c3693CCc93cF191f90dC2aF9 |




High Medium Low Informational

| | High | Medium | Low | Informational |
|---|---|---|---|---|
| Open Issues | 0 | 0 | 0 | 0 |
| Acknowledged Issues | 0 | 0 | 0 | 4 |
| Partially Resolved Issues | 0 | 0 | 0 | 0 |
| Resolved Issues | 0 | 0 | 0 | 0 |

## Types of Severities

### High

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

### Medium

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

### Low

Low-level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

### Informational

These are severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

## Types of Issues

### Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

### Resolved

These are the issues identified in the initial audit and have been successfully fixed.

### Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

### Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

# Checked Vulnerabilities

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- Exception Disorder
- Gasless Send
- Use of tx.origin
- Malicious libraries
- Compiler version not fixed
- Address hardcoded
- Divide before multiply
- Integer overflow/underflow
- ERC20 transfer() does not return boolean
- ERC20 approve() race
- Dangerous strict equalities
- Tautology or contradiction
- Return values of low-level calls
- Missing Zero Address Validation
- Private modifier
- Revert/require functions
- Using block.timestamp
- Multiple Sends
- Using SHA3
- Using suicide
- Using throw
- Using inline assembly

# Techniques and Methods

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

### Structural Analysis

In this step, we have analysed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

### Static Analysis

Static analysis of smart contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

### Code Review / Manual Analysis

Manual analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analysed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

### Gas Consumption

In this step, we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

### Tools and Platforms used for Audit

Mythril, Slither, SmartCheck,RemixIDE, Surya, Solhint.

# Manual Testing

## High Severity Issues

No issues were found

## Medium Severity Issues

No issues were found

## Low Severity Issues

No issues were found

## Informational Issues

### A.1 Multiple pragma directives have been used.

**Remediation**

Contracts should be deployed using the same compiler version/flags with which they have been tested. Locking the pragma (for e.g. by not using ^ in pragma solidity 0.8.4) ensures that contracts do not accidentally get deployed using an older compiler version with unfixed bugs. Ref: *Security Pitfall 2*

**Status**

**Acknowledged**

### A.2 No check or guard For renounceOwnership .

**Remediation**

Contracts should have a double check for renounceOwnership function , because this will make the contract to have address ox000 as the admin, when Admin mistakenly calls the function.

**Status**

**Acknowledged**

## A.3 Centralize functionality of Blacklist users

### Description

This causes a centralized issue: blacklisting can also be used for unintentional purposes. Causing the loss of Decentralization which is the core features of the Blockchain.

### Remediation

Recommendation is removing access control  of Blacklist and favor decentralization. That means adding tactics to do it automatically by analyzing certain traits from an address. But as we also know, this will be a very challenging task to fulfill.

### Status

**Acknowledged**

## A.4 No error message

```
18      modifier notBlacklisted(address _account) {
19          require(!isBlacklisted(_account));
20          _;
21      }
```

### Remediation

Contracts should have an error message when isBlaclisted fails.

### Status

**Acknowledged**

# Functional Testing

Complete functional testing report has been attached *here*

Some of the tests performed are mentioned below:

- should be able to mint 2750000000 token at deployment
- should be able to approve a contract to spend token
- should be able transfer
- should be able to transferFrom
- should be able to transferOwnership
- should be able to Blacklist an address only called by owner
- should be able to removeBlacklist an address only called by owner
- Should revert if not owner try to Blacklist an address
- Should revert if not owner try to removeBlacklist an address
- Should revert if an address is Blacklist and is trying to transfer token

# Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

# Closing Summary

In this report, we have considered the security of the Metal Token Smart Contract by drunk Robots. We performed our audit according to the procedure described above.

Some issues of informational severity were found, Some suggestions and best practices are also provided in order to improve the code quality and security posture.
At the end,Drunk Robots team acknowledged all the issues

# Disclaimer

QuillAudits smart contract audit is not a security warranty, investment advice, or an endorsement of the Drunk Robots Platform. This audit does not provide a security or correctness guarantee of the audited smart contracts.

The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the Drunk Robot Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

# About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies. We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem.

**500+**
Audits Completed

**$15B**
Secured

**500K**
Lines of Code Audited

## Follow Our Journey

# Audit Report
# April, 2022

For

DRUNK ROBOTS

QuillAudits


Canada, India, Singapore, United Kingdom

audits.quillhash.com

audits@quillhash.com