QuillAudits

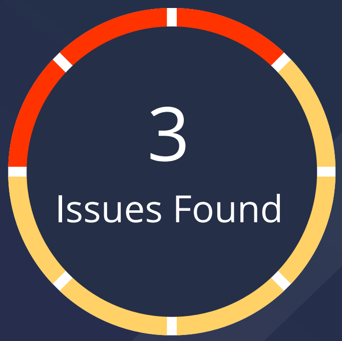
Audit Report
July, 2023

For

Table of Content

Executive Summary

Project Name	Kiros Token
Project URL	*https://kiroscoin.com/*
Overview	Kiros is a global exchange token with a new burn mechanism different from traditional dead wallet methods. The burned amount doesn't go to the dead wallet or zero address but it'll be sent back to their main wallet from their the tokens will be introduced again in the market as a token.
Audit Scope	Sol File was provided for audit
Contracts in Scope	Kiros.sol
Commit Hash	NA
Language	Solidity
Blockchain	Avalanche
Method	Manual Analysis, Functional Testing, Automated Testing
Review 1	3 July 2023 - 6 July 2023
Updated Code Received	13 July 2023
Review 2	17 July 2023
Fixed In	*https://github.com/BlackWalletLTD/token-contracts-evm/blob/main/contracts/Kiros.sol*



🟥 High 🟨 Medium
🟩 Low 🟪 Informational

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	0	0	0
Resolved Issues	**1**	**2**	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
- ✓ Compiler version not fixed
- ✓ Address hardcoded
- ✓ Divide before multiply
- ✓ Integer overflow/underflow
- ✓ 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
Remix IDE, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis.

Manual Testing

A. Contract - Kiros Token

High Severity Issues

A.1 Token is not erc-20 compatible

Description

Kiros token does not follow erc-20 token standard. ERC20 has defined an interface for being compatible but the kiros token is missing some of the implementation. Even though the token may work without standard definition it can cause issues if it is to work with other protocols.

Remediation

Please make sure to make it ERC20 compatible. You can use Open-Zeppelin libraries to do so.

Status

Resolved

Medium Severity Issues

A.2 Fee amount is not sent to black wallet

Description

In kiros whitepaper, there is mention of: `*Anytime there is a transaction, there is a small amount of KIROS burned. This acts as an anti-inflationary tool for the system. The burn, however, is not a traditional burn. The burn goes into a "black wallet" which is not dead*` but while transferring the small amount which is smartly burned is not getting sent back to the black wallet as mentioned, making a loss for the protocol.

Remediation

Please make sure to correct the logic for the fee amount transfer.

Status

Resolved

A.3 Centralization Issue

Description

In kiros contract functions mint() and changeBurnRate() are called by black_wallet which is handled by the team and is the main wallet. If, for some reason, the wallet key gets compromised, then the malicious actor can mint any number of tokens also by changing the burn rate can get a much higher amount of tokens sent back to black_wallet and can do harm to the protocol. Even though the likelihood of this happening is low, it is better to take measures.

Remediation

Please make sure to use multisig if possible.

Note: As discussed with the team, they will be changing the code to mint the total Supply of the tokens at once eliminating the need of the mint function.

Kiros Team: To address the multisig suggestion, the code uses OpenZeppelin's Ownable functions so that it can be transferred to a multisig wallet after creation.

Status

Resolved

Low Severity Issues

No issues found

Informational Issues

No issues found

Functional Tests

Some of the tests performed are mentioned below

- ✓ test_CheckValuesAfterBurnRateChange() (gas: 116862)
- ✓ test_OnlyBlackWalletShouldBeAbleToMint() (gas: 28631)
- ✓ test_transferFunction() (gas: 46900)
- ✓ test_transferFunctionForUsers() (gas: 80637)

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.

Summary

In this report, we have considered the security of the Kiros. We performed our audit according to the procedure described above.Some issues of High, Medium, Low and informational severity were found, Some suggestions and best practices are also provided in order to improve the code quality and security posture.

In conclusion, the Kiros Team resolved all of the issues as recommended.

Disclaimer

QuillAudits Smart contract security audit provides services to help identify and mitigate potential security risks in Kiros smart contracts. However, it is important to understand that no security audit can guarantee complete protection against all possible security threats. QuillAudits audit reports are based on the information provided to us at the time of the audit, and we cannot guarantee the accuracy or completeness of this information. Additionally, the security landscape is constantly evolving, and new security threats may emerge after the audit has been completed.

Therefore, it is recommended that multiple audits and bug bounty programs be conducted to ensure the ongoing security of Kiros smart contracts. One audit is not enough to guarantee complete protection against all possible security threats. It is important to implement proper risk management strategies and stay vigilant in monitoring your smart contracts for potential security risks.

QuillAudits cannot be held liable for any security breaches or losses that may occur as a result of using our audit services. It is the responsibility of the Kiros to implement the recommendations provided in our audit reports and to take appropriate steps to mitigate potential security risks.

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.

850+
Audits Completed

$30B
Secured

800K
Lines of Code Audited

Follow Our Journey

Audit Report July, 2023

For

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com

audits@quillhash.com