QuillAudits

AUDIT REPORT

February, 2025

For

Table of Content

Executive Summary 03

Number of security issues per severity. 04

Check Vulnerabilities 05

Techniques & Methods 08

Types of Severity 10

Types of Issues 11

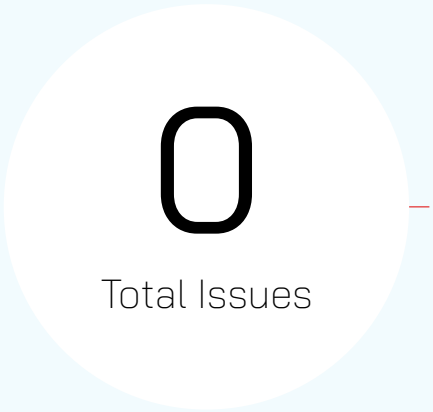
Functional Tests 13

Closing Summary & Disclaimer 14

Executive Summary

Project name	Wow Token
Overview	The WOW project introduces a custom ERC20 token named 'WOW' with the symbol 'WOW'. It leverages OpenZeppelin's ERC20 implementation to ensure security and standard compliance. Upon deployment, the contract mints a total supply of 1 billion WOW tokens, each with 18 decimal places, directly to a specified receiver address.
Source Code	https://ether-scan.io/token/0x8c5d1963e41eb351495d6a7a068052103a47b40c#
Audit Scope	The scope of this audit was to analyse the Wow Token Contract for quality, security, and correctness.
Blockchain	Ethereum
Method	Manual Review, Functional Testing, Automated Testing, etc. All the raised flags were manually reviewed and re-tested to identify any false positives.
Timeline	21st February 2025
Second Review	NA
Fixed In	NA

Number of Issues per Severity



High	0 (NaN%)
Medium	0 (NaN%)
Low	0 (NaN%)
Informational	0 (NaN%)

Severity

Issues	High	Medium	Low	Informational
Open	0	0	0	0
Resolved	0	0	0	0
Acknowledged	0	0	0	0
Partially Resolved	0	0	0	0

Checked Vulnerabilities

- ☑ Re-entrancy
- ☑ Timestamp Dependence
- ☑ Gas Limit and Loops
- ☑ DoS with Block Gas Limit
- ☑ Transaction-Ordering Dependence
- ☑ Use of tx.origin
- ☑ Exception Disorder
- ☑ Gasless Send
- ☑ Balance Equality
- ☑ Compiler Version Not Fixed
- ☑ Send Instead of Transfer
- ☑ Style Guide Violation
- ☑ Unchecked External Call
- ☑ Unchecked Math
- ☑ Access Management
- ☑ Centralization of Control
- ☑ Implicit Visibility Level
- ☑ Malicious Libraries
- ☑ Revert/Require Functions
- ☑ Missing Zero Address Validation

Techniques and Methods

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

- The overall quality of code.
- Use of best practices.
- Code documentation and comments, match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC 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 analyzed 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

A 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 analyzed, 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 behavior 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, Foundry, Solhint, Mythril, Slither, Solidity statistical analysis.

Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below

High Severity Issues

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 Severity Issues

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 Severity Issues

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

Informational Issues

These are four 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

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

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.

No Issues Found

Functional Tests

Some of the tests performed are mentioned below:

- Should get the Name of Token
- Should get the Token Decimal
- Should be able to get Token
- Should get the TotalSupply
- Should be able to get balanceOf
- Should be able to approve
- Should be able to allowance
- Should be able to transfer
- Should be able to TransferForm

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 Wow Token contract. We performed our audit according to the procedure described above.

Code Looks Good,No Issues Found.

Disclaimer

At QuillAudits, we have spent years helping projects strengthen their smart contract security. However, security is not a one-time event—threats evolve, and so do attack vectors. Our audit provides a security assessment based on the best industry practices at the time of review, identifying known vulnerabilities in the Received Wow Token contract source code, including its compilation and intended functionality.

This report does not serve as a security guarantee, investment advice, or an endorsement of any platform. It reflects our findings based on the provided code at the time of analysis and may no longer be relevant after any modifications. The presence of an audit does not imply that the contract is free of vulnerabilities or fully secure.

While we have conducted a thorough review, security is an ongoing process. We strongly recommend multiple independent audits, continuous monitoring, and a public bug bounty program to enhance resilience against emerging threats.

Stay proactive. Stay secure.

About QuillAudits

QuillAudits is a leading name in Web3 security, offering top-notch solutions to safeguard projects across DeFi, GameFi, NFT gaming, and all blockchain layers. With six years of expertise, we've secured over 1000 projects globally, averting over $30 billion in losses. Our specialists rigorously audit smart contracts and ensure DApp safety on major platforms like Ethereum, BSC, Arbitrum, Algorand, Tron, Polygon, Polkadot, Fantom, NEAR, Solana, and others, guaranteeing your project's security with cutting-edge practices.

6+ Years of Expertise	**1M+** Lines of Code Audited
$30B+ Secured in Digital Assets	**1K+** Projects Secured

Follow Our Journey

AUDIT REPORT

February, 2025

For

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com audits@quillaudits.com