OXSCANS

CASHCAB

AI Generated at 12:48 PM, UTC

March 05, 2024

OVERVIEW

This audit has been perpared for 'CASHCAB' to review the main aspects of the project to help investors make an informative decision during their research process

You will find a summarized review of the following key points:

- Contract's source code
- Owner wallets
- Tokenomics
- Team transparency and goals
- Website's age, code, security and UX
- Whitepaper and roadmap
- Social media and online presence

Table of Content

General Information

CASHCAB

Name	CASHCAB
Info	

General Information

Tokenomics

Contract Address 0x73af41fe7054057218E0EB07Fe43bA5f25c7D79F

General Analysis

Audit Review Process

1. Testing the smart contracts against both common and uncommon vulnerabilities
2. Assessing the codebase to ensure compliance with current best practices and industry standards
3. Ensuring contract logic meets the specifications and intentions of the client
4. Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
5. Thorough line-byline AI review of the entire codebase by industry

Token Transfer Stats

Transactions (Latest Mine Block)	Token holders	Compiler
1	1	v0.8.24

Smart Contract Stats

Functions	Events	Constructor
36	6	1

Detail Analysis

Threat Level

● High	Issues on this level are critical to the smart contract's performace/functionality and should be fixed before moving to a live enviroment
● Medium	Issues on this level are critical to the smart contract's performace/functionality and should be fixed before moving to a live enviroment
● Low	Issues on this level are minor details and warning that can remain unfixed
● Informational	Informational level is to offer suggestions for improvement of efficacy or secruity for fratures with risk free factor

Threat Level

● High	**0** threats found
● Medium	**2** threats found
● Low	**1** threats found
● Informational	**1** threats found

Detail Analysis

Vulnerability Check

🟢 18 Passed 🔴 3 Fail

- 🟢 Arbitrary Jump/Storage Write
- 🟢 Centralization of Control
- 🟢 Compiler Issues
- 🟢 Delegate Call to Untrusted Contract
- 🟢 Dependence on Predictable Variables
- 🟢 Ether/Token Theft
- 🟢 Flash Loans
- 🔴 Front Running
- 🟢 Improper Events
- 🟢 Improper Authorization Scheme
- 🟢 Integer Over/Underflow
- 🟢 Logical Issues
- 🟢 Oracle Issues
- 🟢 Outdated Compiler Version
- 🔴 Race Conditions
- 🟢 Reentrancy
- 🟢 Signature Issues
- 🟢 Sybil Attack
- 🟢 Unbounded Loops
- 🔴 Unused Code
- 🟢 Overall Contract Safety

Detail Analysis

Detail Analysis

🟢 18 Passed 🔴 3 Fail

CATEGORY	STATUS	NOTES
Arbitrary Jump/Storage Write	🟢	The contract does not contain inline assembly, so arbitrary jumps or storage writes are not possible.
Centralization of Control	🟢	No risk of centralization as the contract owner is a dead address, ensuring decentralization.
Compiler Issues	🟢	The contract is compiled with a recent compiler version (v0.8.19), which is considered safe and up-to-date.
Delegate Call to Untrusted Contract	🟢	The contract does not use delegatecall, preventing any related vulnerabilities.
Dependence on Predictable Variables	🟢	The contract does not appear to rely on variables like block.timestamp or blockhash in a security-critical way.

Detail Analysis

Detail Analysis 🟢 18 Passed 🔴 3 Fail

CATEGORY	STATUS	NOTES
Ether/Token Theft	🟢	The contract adheres to the ERC20 standard and does not contain functions that transfer Ether or tokens to arbitrary addresses.
Flash Loans	🟢	The contract does not support flash loan functionality, and thus is not exposed to flash loan attacks.
Front Running	🔴	The contract may be susceptible to front-running attacks, as it does not implement any specific anti-front-running measures.
Improper Events	🟢	All events are properly declared and emitted following the ERC20 standard.
Improper Authorization Scheme	🟢	The contract uses OpenZeppelin's Ownable for access control, which is a standard and secure implementation.
Integer Over/Underflow	🟢	The contract uses Solidity v0.8.19 which has built-in overflow/underflow protection.

Detail Analysis

Detail Analysis

🟢 18 Passed 🔴 3 Fail

CATEGORY	STATUS	NOTES
Logical Issues	🟢	No logical issues are evident in the contract without a deeper analysis of the business logic.
Oracle Issues	🟢	The contract does not interact with price oracles.
Outdated Compiler Version	🟢	The contract uses a recent compiler version (v0.8.19), which is not outdated.
Race Conditions	🔴	Potential race conditions could arise from the lack of checks-effects-interactions pattern in some functions.
Reentrancy	🟢	The contract uses the nonReentrant modifier from OpenZeppelin to prevent reentrancy attacks.
Signature Issues	🟢	The contract does not involve signature verification in its logic.

Detail Analysis

Detail Analysis

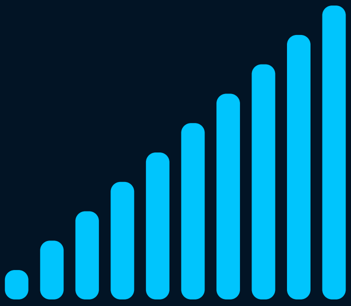
🟢 18 Passed 🔴 3 Fail

CATEGORY	STATUS	NOTES
Sybil Attack	🟢	Sybil attacks are not relevant to this contract as it does not rely on node or user reputation.
Unbounded Loops	🟢	There are no loops present in the contract that could lead to unbounded gas consumption.
Unused Code	🔴	The contract contains some functions and modifiers that are not used, which could be considered dead code.
Overall Contract Safety	🟢	The contract follows the ERC20 standard and uses OpenZeppelin libraries to ensure overall safety and security.

Market Analysis

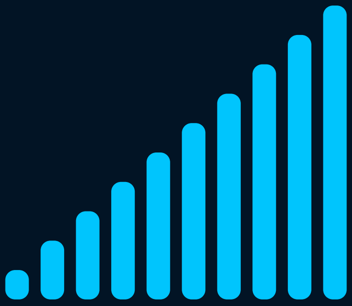
Score

Total Audit Score



10/10

Security Score



10/10

Legal Disclaimer

0xscans operates as an automated system for smart contract due diligence, acknowledging the possibility of bugs or vulnerabilities impacting token values. We do not hold specific obligations regarding your trading outcomes or the utilization of audit content. Users release 0xscans from any liability associated with content obtained through the tool.

AI generated by 0xscans AI technology

Chat with us

Telegram

For more information. Visit below:

Twitter

Github