



Building the Futuristic **Blockchain Ecosystem**

SECURITY AUDIT REPORT



Green Whale Challenge

TABLE OF CONTENTS

02 Table of Contents

03 Overview

04 Project Description

05 Social Media Profiles

06 Contract Details

07 Owner Privileges

08 Audit Methodology

09 Vulnerabilities Checklist

10 Risk Classification

11 Inheritance Tree

12 Function Details

14 Manual Review

15 Findings

17 About Expelee

18 Disclaimer

OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

Audit Result	Passed
KYC Verification	-
Audit Date	24 April 2023

PROJECT DESCRIPTION

To know more about Green Whale Challenge visit their Social Media handles.



SOCIAL MEDIA PROFILES

GREEN WHALE CHALLENGE



<https://t.me/greenwhalechallenge>



<https://twitter.com/gwctoken>

It's always good to check the social profiles of the project, before making your investment.

Team Expelee

CONTRACT DETAILS

Token Name: Green Whale Challenge

Symbol: GWC

Network: Binance Smart Chain

Language: Solidity

Contract Address: --

Total Supply: 10,000,000

Contract SHA-256 Checksum:

706a26798c495521786ebd6546ec01ed34758f8d

Owner's Wallet: --

Deployer's Wallet: --

Testnet:

<https://testnet.bscscan.com/token/0x49eab73b24aa104927c29223836a07ab0323b4c1>

OWNER PRIVILEGES

- Contract owner is not able to set buy/sell taxes over 12% each
- Contract owner is not able to set transfer tax (0%)
- Contract owner is not able to blacklist an arbitrary wallet
- Contract owner is not able to disable trades/transfers
- Contract owner is not able to mint new tokens
- **Contract owner must enable trades for public**

AUDIT METHODOLOGY

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

Tools

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

VULNERABILITY CHECKS

Design Logic	Passed
Compiler warnings	Passed
Private user data leaks	Passed
Timestamps dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front Running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zepplin module	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

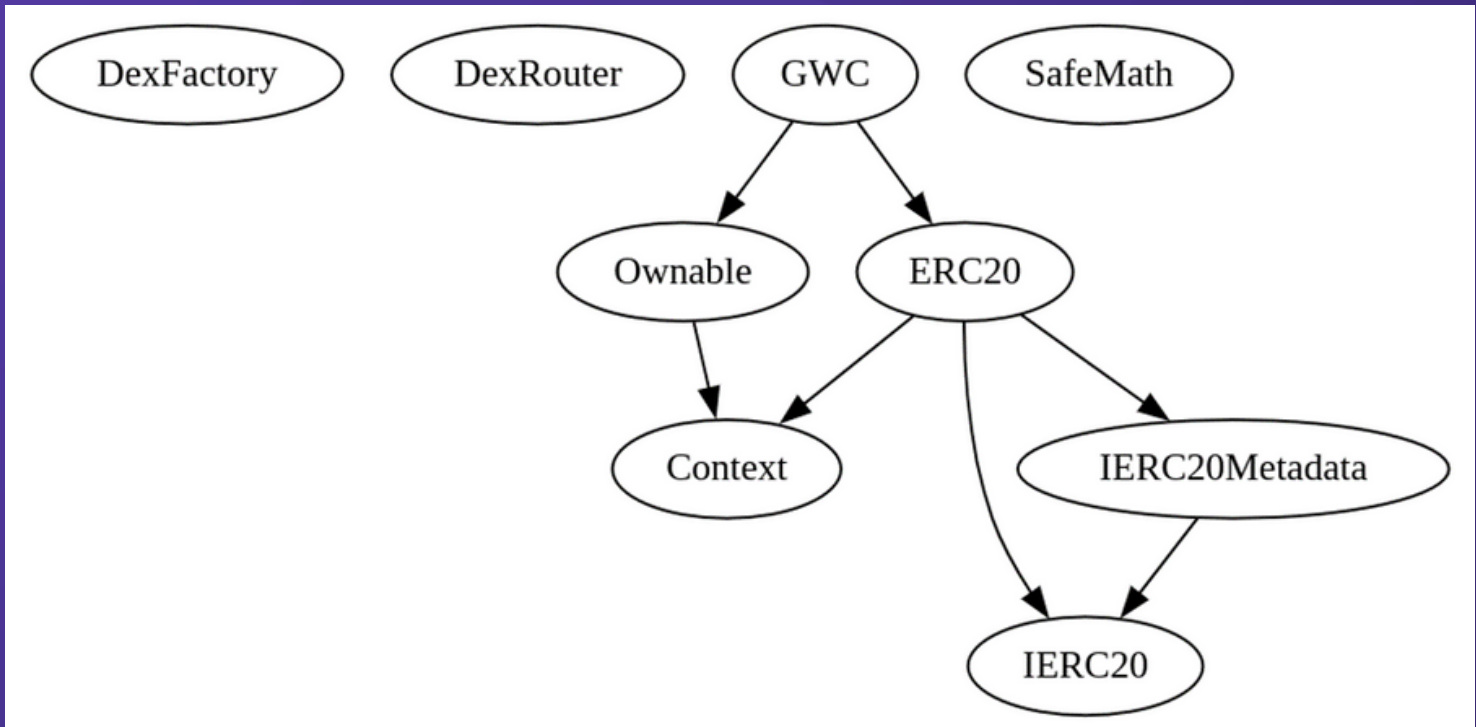
Low Risk

Issues on this level are minor details and warnings that can remain unfixed.



Informational

Issues on this level are minor details and warnings that can remain unfixed.

INHERITANCE TREES



FUNCTION DETAILS

Symbol	Meaning
	Function can modify state
	Function is payable

```

| Contract | Type | Bases | | |
|-----|-----|-----|-----|-----|
| L | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
| **GWC** | Implementation | ERC20, Ownable |||
| L | <Constructor> | Public ! | ● | ERC20 |
| L | startTrading | External ! | ● | onlyOwner |
| L | setDevWallet | External ! | ● | onlyOwner |
| L | setMarketing | External ! | ● | onlyOwner |
| L | setBuyFees | External ! | ● | onlyOwner |
| L | setSellFees | External ! | ● | onlyOwner |
| L | setSwapTokensAtAmount | External ! | ● | onlyOwner |
| L | toggleSwapping | External ! | ● | onlyOwner |
| L | setWhitelist | External ! | ● | onlyOwner |
| L | checkWhitelist | External ! | | NO ! |
| L | _takeTax | Internal 🔒 | ● | |
| L | _transfer | Internal 🔒 | ● | |
| L | manageTaxes | Internal 🔒 | ● | |
| L | swapAndLiquify | Internal 🔒 | ● | |
| L | swapToETH | Internal 🔒 | ● | |
| L | addLiquidity | Private 🔒 | ● | |
| L | withdrawStuckETH | External ! | ● | onlyOwner |
| L | withdrawStuckTokens | External ! | ● | onlyOwner |
| L | <Receive Ether> | External ! | 🟢 | NO ! |
|||||
| **Ownable** | Implementation | Context |||
| L | <Constructor> | Public ! | ● | NO ! |
| L | owner | Public ! | | NO ! |
| L | _checkOwner | Internal 🔒 | | |
| L | renounceOwnership | Public ! | ● | onlyOwner |
| L | transferOwnership | Public ! | ● | onlyOwner |
| L | _transferOwnership | Internal 🔒 | ● | |
|||||
| **Context** | Implementation | |||
| L | _msgSender | Internal 🔒 | | |
| L | _msgData | Internal 🔒 | | |
|||||
| **SafeMath** | Library | |||
| L | tryAdd | Internal 🔒 | | |
| L | trySub | Internal 🔒 | | |
| L | tryMul | Internal 🔒 | | |
| L | tryDiv | Internal 🔒 | | |
| L | tryMod | Internal 🔒 | | |
| L | add | Internal 🔒 | | |
| L | sub | Internal 🔒 | | |
| L | mul | Internal 🔒 | | |
| L | div | Internal 🔒 | | |

```

FUNCTION DETAILS

```

| L | mod | Internal | 🔒 | | |
| L | sub | Internal | 🔒 | | |
| L | div | Internal | 🔒 | | |
| L | mod | Internal | 🔒 | | |
|||||
| **IERC20** | Interface | |||
| L | totalSupply | External | ! | | NO ! |
| L | balanceOf | External | ! | | NO ! |
| L | transfer | External | ! | ● | NO ! |
| L | allowance | External | ! | | NO ! |
| L | approve | External | ! | ● | NO ! |
| L | transferFrom | External | ! | ● | NO ! |
|||||
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata |||
| L | <Constructor> | Public | ! | ● | NO ! |
| L | name | Public | ! | | NO ! |
| L | symbol | Public | ! | | NO ! |
| L | decimals | Public | ! | | NO ! |
| L | totalSupply | Public | ! | | NO ! |
| L | balanceOf | Public | ! | | NO ! |
| L | transfer | Public | ! | ● | NO ! |
| L | allowance | Public | ! | | NO ! |
| L | approve | Public | ! | ● | NO ! |
| L | transferFrom | Public | ! | ● | NO ! |
| L | increaseAllowance | Public | ! | ● | NO ! |
| L | decreaseAllowance | Public | ! | ● | NO ! |
| L | _transfer | Internal | 🔒 | ● | |
| L | _mint | Internal | 🔒 | ● | |
| L | _burn | Internal | 🔒 | ● | |
| L | _approve | Internal | 🔒 | ● | |
| L | _spendAllowance | Internal | 🔒 | ● | |
| L | _beforeTokenTransfer | Internal | 🔒 | ● | |
| L | _afterTokenTransfer | Internal | 🔒 | ● | |
|||||
| **IERC20Metadata** | Interface | IERC20 |||
| L | name | External | ! | | NO ! |
| L | symbol | External | ! | | NO ! |
| L | decimals | External | ! | | NO ! |

```

MANUAL REVIEW

Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categroies:

High

Medium

Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

FINDINGS

Findings	Severity	Found
High Risk	● High	0
Medium Risk	● Medium	0
Low Risk	● Low	0
Suggestion & discussion	● Informational	1
Gas Optimizations	● Gas Opt.	0

INFORMATIONAL FINDING

Enabling trades is not guaranteed

Severity : Informational

Overview

The owner of the contract must enable trades for public, otherwise no one would be able to buy/sell/transfer their tokens except whitelisted wallets.

```
function startTrading() external onlyOwner {  
    tradingStatus = true;  
}
```

Suggestion :

To mitigate this issue there are several options:

- Enable tradings before presale

Issue Status: **Open**

ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

 www.expelee.com



expeleeofficial



expelee



Expelee



expelee



expelee_official



expelee-co

expelee

Building the Futuristic **Blockchain Ecosystem**

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools.

Under no circumstances did Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway do your own research and protect yourselves from scams.

This document should not be presented as a reason to buy or not buy any particular token. The Expelee team disclaims any liability for the resulting losses.

The logo for Expelee, featuring the word "expelee" in a stylized font. The "ex" is in white, and "pelee" is in orange. The letters are bold and modern.

Building the Futuristic **Blockchain Ecosystem**