



Building the Futuristic **Blockchain** Ecosystem

SECURITY AUDIT REPORT

FROGO

TABLE OF CONTENTS

02	Table of Contents	
03	Overview	
04	Contract Details	
05	Owner Privileges	
06	Audit Methodology	
07	Vulnerabilities Checklist	
08	Risk Classification	
09	Inheritance Trees & Risk Overview	
10	Testnet Version	
12	Function Details	
15	Manual Review	
16	Findings	
22	About Expelee	
23	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	DONE
Audit Date	17 May 2023

CONTRACT DETAILS

Token Name: Crazy Frogo

Symbol: FROGO

Network: Binance smart chain

Language: Solidity

Contract Address:

0x5e7f09B2D6b0571Cdcc3Cf9fC0bf4F8e2f77ca48

Total Supply: 420,101,101,101,101

Contract SHA-256 Checksum: -

965be1bc06988cec5155ebe718030c0c8f933457

Owner's Wallet:

0x4CA5daE77F3833F4e52C7f5c1a9d790Eef58302B

Deployer's Wallet:

0x4CA5daE77F3833F4e52C7f5c1a9d790Eef58302B

OWNER PRIVILEGES

- Contract owner can set marketing wallet address
- Contract owner can set buy, sell, and transfer taxes with a maximum of 12% for buy/sell and 5% for transfer
- Contract owner can set the swap threshold amount (minimum 0 and maximum 0.5% of total supply)
- Contract owner can toggle swapping on and off
- Contract owner can whitelist or unwhitelist wallets
- Whitelisted wallets are exempt from taxes
- Taxes are taken during transfers based on the type of transfer (buy, sell, or transfer)
- Swapping is triggered when the contract balance reaches the swap threshold and the recipient is the pair address
- Swapped tokens are converted to BNB and sent to the marketing wallet
- Contract owner can withdraw stuck BNB and bep20 tokens from the contract
- The contract accepts incoming BNB payments

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.

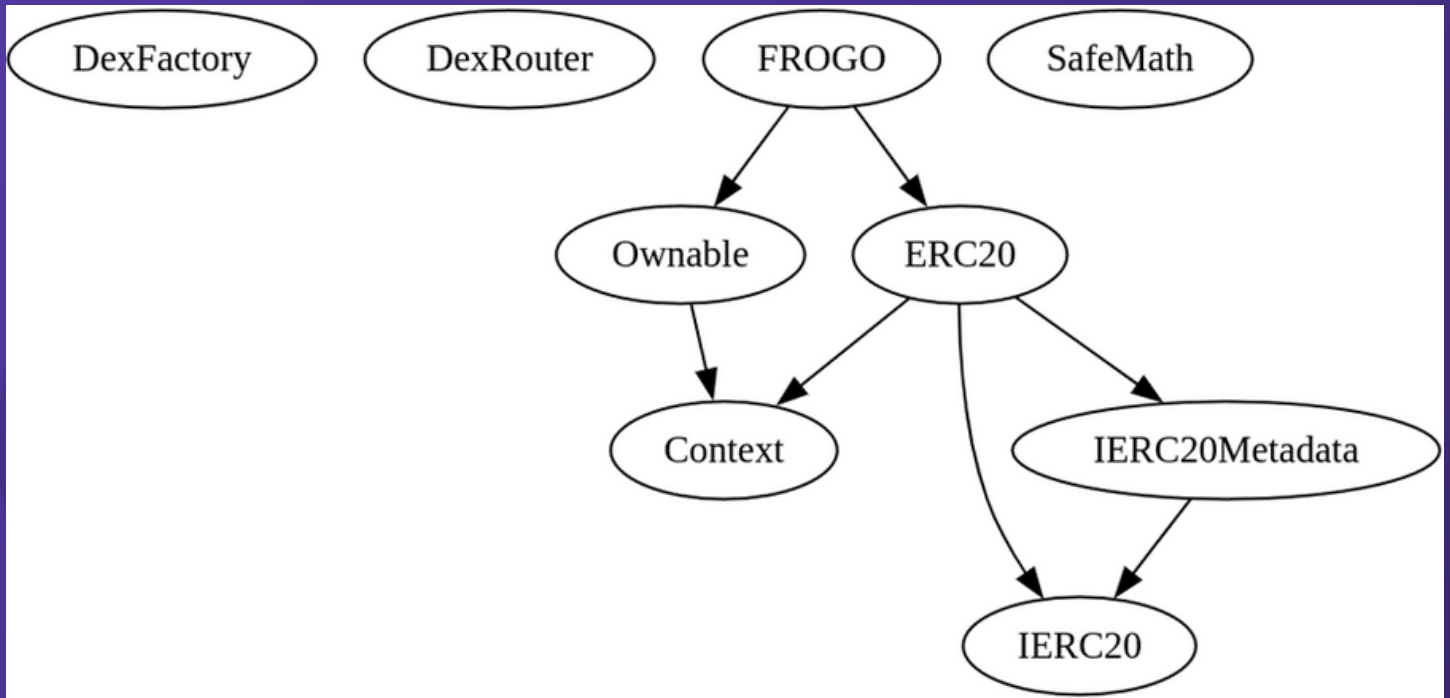
Low Risk

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



TESTNET VERSION

Adding Liquidity ✓

Tx:

<https://testnet.bscscan.com/tx/0xf65606d2fca74fdc4d2593d5e8ea6fdf7e63e189d6c2ad91ea46a2334faa9177>

=====

Buying from a fee excluded wallet ✓

Tx (0% tax):

<https://testnet.bscscan.com/tx/0xe6f68a7728b79188712e287153783102b6715fef09bd9202c1bc3e52c7ab5788>

=====

Selling from a fee excluded wallet ✓

Tx (0% tax):

<https://testnet.bscscan.com/tx/0x6f26210f02b2f09d81881a794af9a2b95ad1553763ead621516c2a394101384c>

=====

Transferring using a fee excluded wallet ✓

Tx (0% tax):

<https://testnet.bscscan.com/tx/0xf6498445b26f8862e9ea0e1c76197703f127882b3a1f73355fdf7522055f38f2>

=====

TESTNET VERSION

Buying from a regular wallet ✓

Tx (0-12% tax):

<https://testnet.bscscan.com/tx/0xf5881253aeb325c732379794319ddd99a0ccf5aa9658aa0dc30fba41bbf03216>

=====

Selling from a regular wallet ✓

Tx (0-12% tax):

<https://testnet.bscscan.com/tx/0x0a05d23135cd68628561443c9456f81a597c8fc4a8844cf1191187f215f73014>

=====

Transferring a regular wallet ✓

Tx (0-12%):

<https://testnet.bscscan.com/tx/0x990e2132b3f1da7745d6cb7417e9da64750c422eb4f62731de4cb5e688ab4aae>

=====

FUNCTION DETAILS

Contract	Type	Bases			
----- :----- :----- :----- :-----					
L	**Function Name**	**Visibility**	**Mutability**	**Modifiers**	
DexFactory Interface					
L	createPair	External	!	●	[NO !]
DexRouter Interface					
L	factory	External	!		[NO !]
L	WETH	External	!		[NO !]
L	addLiquidityETH	External	!	8	[NO !]
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External	!	●	[NO !]
FROGO Implementation ERC20, Ownable					
L	<Constructor>	Public	!	●	ERC20
L	setmarketingWallet	External	!	●	onlyOwner
L	setBuyTaxes	External	!	●	onlyOwner
L	setSellTaxes	External	!	●	onlyOwner
L	setTransferFees	External	!	●	onlyOwner
L	setSwapTokensAtAmount	External	!	●	onlyOwner
L	toggleSwapping	External	!	●	onlyOwner
L	setWhitelistStatus	External	!	●	onlyOwner
L	checkWhitelist	External	!		[NO !]
L	_takeTax	Internal	🔒	●	
L	_transfer	Internal	🔒	●	
L	internalSwap	Internal	🔒	●	
L	swapToETH	Internal	🔒	●	

FUNCTION DETAILS

```

| L | withdrawStuckETH | External ! | ● | onlyOwner |
| L | withdrawStuckTokens | External ! | ● | onlyOwner |
| L | <Receive Ether> | 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 🔒 | ● | |
|||||
| **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 ! |
|||||
| **IERC20Metadata** | Interface | IERC20 |||
| L | name | External ! | | NO ! |
| L | symbol | External ! | | NO ! |
| L | decimals | External ! | | NO ! |
|||||
| **Context** | Implementation | |||
| L | _msgSender | Internal 🔒 | | |
| L | _msgData | Internal 🔒 | | |
|||||
| **Ownable** | Implementation | Context |||
| L | <Constructor> | Public ! | ● | NO ! |
| L | owner | Public ! | | NO ! |
| L | _checkOwner | Internal 🔒 | | |

```

FUNCTION DETAILS

```

| L | renounceOwnership | Public ! | ● | onlyOwner |
| L | transferOwnership | Public ! | ● | onlyOwner |
| L | _transferOwnership | 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 🔒 | | |
| L | mod | Internal 🔒 | | |
| L | sub | Internal 🔒 | | |
| L | div | Internal 🔒 | | |
| L | mod | Internal 🔒 | | |

```

Legend

Symbol	Meaning
●	Function can modify state
🔒	Function is payable

MANUAL REVIEW

Severity Criteria

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

Vulnerabilities are divided into three primary risk categories:

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	2
Suggestion & discussion	● Informational	3
Gas Optimizations	● Gas Opt.	0

LOW RISK FINDING

Category: Centralization

Subject: Centralized control over stuck tokens and BNB withdrawal

Severity: Low

Overview:

The contract allows the owner to withdraw stuck tokens and BNB from the contract. This centralizes control over the recovery of stuck assets.

Code:

- withdrawStuckBNB()
- withdrawStuckTokens(address bep20_token)

Suggestion:

Consider implementing a time-locked multisig wallet to handle the recovery of stuck assets. This would reduce the centralization risk and provide additional security.

LOW RISK FINDING

Category: Centralization

Subject: Centralized control over fees and whitelisting

Severity: Low

Overview:

The contract allows the owner to set buy, sell, and transfer fees, as well as whitelist addresses. This centralizes control over the token's fee structure and exemptions.

Code:

- setBuyTaxes(uint256 _marketingTax) => 0-12%
- setSellTaxes(uint256 _marketingTax) => 0-12%
- setTransferFees(uint256 _marketingTax) => 0-5%
- setWhitelistStatus(address _wallet, bool _status)

SUGGESTIONS

Category: Informational

Subject: Centralized control over swapping and swap threshold

Overview:

The contract allows the owner to toggle swapping and set the swap threshold. This centralizes control over the token's swapping mechanism which can affect trade slippage and token price.

Code:

- setSwapTokensAtAmount(uint256 _newAmount)
- toggleSwapping()

SUGGESTIONS

Category: Informational

Subject: Centralized control over marketing wallet

Overview:

The contract allows the owner to set the marketing wallet address. This centralizes control over the token's marketing funds.

Code:

– `setmarketingWallet(address _newmarketing)`

SUGGESTIONS

Although the fees are adjustable in a safe range but it would be a good practice to consider implementing a decentralized governance mechanism to allow token holders to vote on fee structures and whitelisting. This would reduce the centralization risk and give more control to the community.

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**