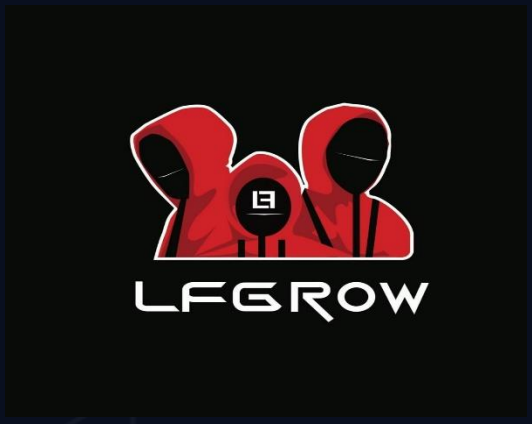
expelee

A Secure Place For Web3

SMART CONTRACT AUDIT OF

LFGROW



Contract Address

0x4C194a773d416a870Faa26c9D7ED766c60Bcac6e

Audit Summary

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**

Ownership: **RENOUNCED**

KYC Verification: NOT DONE

Audit Date: 08/08/2022

Audit Team: **EXPELEE**

Be aware that smart contracts deployed on the blockchain aren't resistant to internal exploit, external vulnerability, or hack. For a detailed understanding of risk severity, source code vulnerability, functional hack, and audit disclaimer, kindly refer to the audit.

DISCLAMER

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 guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always Do your own research and protect 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. Always 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.

Contract Review

Contract Name	**BABY TOKEN**
Compiler Version	**v0.8.4+commit.c7e474f2**
Optimization	**Yes with 200 runs**
License	**NONE license**
Explorer	**https://bscscan.com/address/0x4c194a773d416a870faa26c9d7ed766c60bcac6e#code**
Symbol	**Lfgrow**
Decimals	**18**
Total Supply	**100,000,000,000**
Domain	**https://lfgrow.info/**

Project Review

Token Name: Lfgrow

Web Site: https://lfgrow.info/

Twitter: https://twitter.com/LFGrow_BSC

Telegram: https://t.me/LFGrow_BSC

Contract Address:
0x4C194a773d416a870Faa26c9D7ED766c60Bcac6e

Platform: Binance Smart Chain

Token Type: BEP 20

Language: SOLIDITY

Audit Methodology

The scope of this report is to audit the smart contract source code. We have scanned the contract and reviewed the project for common vulnerabilities, exploits, hacks, and back-doors. Below is the list of commonly known smart contract vulnerabilities, exploits, and hacks:

Category

Smart Contract Vulnerabilities	- Unhandled Exceptions - Transaction Order Dependency - Integer Overflow - Unrestricted Action - Incorrect Inheritance Order - Typographical Errors - Requirement Violation
Source Code Review	- Gas Limit and Loops - Deployment Consistency - Repository Consistency - Data Consistency - Token Supply Manipulation
Functional Assessment	- Operations Trail & Event Generation - Assets Manipulation - Liquidity Access

Vulnerability Checklist

№	Description.	Result
1	Compiler warnings.	Passed
2	Race conditions and Re-entrancy. Cross-function raceconditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Passed
10	Methods execution permissions.	Passed
11	Economy model.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed
18	Design Logic.	Passed
19	Cross-function race conditions.	Passed
20	Safe Zeppelin module.	Passed
21	Fallback function security.	Passed

Manual Audit

Low-Risk

3 low-risk code issues found

Medium-Risk

0 medium-risk code issues found

High-Risk

0 high-risk code issues found

Audit Summary

Number of lines: 3142 (+ 0 in dependencies, + 0 in tests)
Number of assembly lines: 0
Number of contracts: 26 (+ 0 in dependencies, + 0 tests)
Number of optimization issues: 43
Number of informational issues: 64
Number of low issues: 3
Number of medium issues: 0
Number of high issues: 0
ERCs: ERC2612, ERC20

Name	# functions	ERCS	ERC20 info	Complex code	Features
SafeMath	13			No	
Clones	4			No	Assembly
IUniswapV2Factory	8			No	
IUniswapV2Router02	24			No	Receive ETH
IUniswapV2Pair	27	ERC20,ERC2612	∞ Minting Approve Race Cond.	No	
SafeMathInt	7			No	
SafeMathUint	1			No	
IterableMapping	6			No	
BABYTOKENDividendTracker	71	ERC20	No Minting Approve Race Cond.	Yes	Tokens interaction Upgradeable
BABYTOKEN	72	ERC20	No Minting Approve Race Cond.	Yes	Receive ETH Send ETH Tokens interaction

Low-Risk

1) Unused return

The return value of an external call is not stored in a local or state variable.

```
try dividendTracker.process(gas) returns (
                uint256 iterations,
                uint256 claims,
                uint256 lastProcessedIndex
            ) {
                emit ProcessedDividendTracker(
                    iterations,
                    claims,
                    lastProcessedIndex,
                    true,
                    gas,
                    tx.origin
                );
```

Recommendation

Ensure that all the return values of the function calls are used.

2) Reentrancy vulnerabilities.

Detection of the reentrancy bug.

```
uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(

            tokenAmount,
            0, // accept any amount of ETH
            path,
            address(this),
            block.timestamp
        );
    }
```

Recommendation

Apply the check-effects-interactions pattern.

3) Functions that send Ether to arbitrary destinations

Unprotected call to a function sending Ether to an arbitrary address.

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {

        // approve token transfer to cover all possible scenarios
        _approve(address(this), address(uniswapV2Router), tokenAmount);
        // add the liquidity
        uniswapV2Router.addLiquidityETH{value: ethAmount}(
            address(this),
            tokenAmount,
            0, // slippage is unavoidable
            0, // slippage is unavoidable
            address(0),
            block.timestamp
        );
    }
```

Recommendation

Ensure that an arbitrary user cannot withdraw unauthorized funds.

Important Points To Consider

Can Take Back Ownership	Not detected
Owner Change Balance	Not detected
Blacklist	Not detected
Modify Fees	Detected
Proxy	Not detected
Whitelisted	Not detected
Anti Whale	Not detected
Trading Cooldown	Not detected
Transfer Pausable	Not detected
Cannot Sell All	Not detected
Hidden Owner	Not detected
Creator Address	0xE8a285Aa8f07bD25cFD6f2dE1F94851cB652C0D2
Creator Balance	12150000000 Lfgrow
Creator Percent	12.15%
Owner Address	0xE8a285Aa8f07bD25cFD6f2dE1F94851cB652C0D2
Owner Balance	12150000000 Lfgrow
Owner Percent	12.15%
Mint	Not detected

About Expelee

Expelee is a community driven organisation dedicated to fostering an anti-rug movement. We're here to keep investment safe from fraudsters. We've encountered several rug pulls and know how it feels to be duped, which is why we don't want anybody else to go through the same experience. We are here to raise awareness through our services so that the future of cryptocurrency can be rug-free.

The auditing process focuses to the following considerations with collaboration of an expert team:

- Functionality test of the Smart Contract to determine if proper logic has been followed throughout the whole process.
- Manually detailed examination of the code line by line by experts.
- Live test by multiple clients using Test net.
- Analysing failure preparations to check how the Smart
- Contract performs in case of any bugs and vulnerabilities.
- Checking whether all the libraries used in the code are on the latest version.
- Analysing the security of the on-chain data.

Social Media

/Expelee

/ExpeleeOfficial

/expelee-co