CRACKEN

Smart Contract Security Audit

Project:Hallelujah Coin

Jul 23, 2022

hallelujah

Contract Address

0x0a724B9144D5d3C190584071d44001Af7198488F

Table of Contents

Disclaimer

The contents of this report reflect only the CRACKEN TECH audit team's understanding of the current progress and status of the security of the code audited, to verify the integrity of the code provided for the scope of this audit. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Given the size of the project, the findings detailed here are not to be considered exhaustive, and further testing and audit are recommended after the issues covered are fixed. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence regardless of the findings presented in this report.

The review does not address the compiler layer, any other areas beyond the programming language, or other programming aspects that could present security risks. If the audited source files are smart contract files, risks or issues introduced by using data feeds from off-chain sources are not extended by this review either.

Audit Review

The source code of the Hallelujah Coin was audited in order to acquire a clear impression of how the project was implemented. The Cracken Tech audit team conducted in-depth research, analysis, and scrutiny, resulting in a series of observations. A detailed list of each issue found, and vulnerabilities in the source code will be included in the audit report. The problems and potential solutions are given in this report, we will identify common sources for such problems and comments for improvement.

The auditing process will follow a routine as special considerations by Cracken:

- Review of the specifications, sources, and instructions provided to Cracken to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Cracken describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code is exercised when we run the test cases.
- Symbolic execution is analyzing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

Project Review

Token Summary

Parameter	Result
Token Name	Hallelujah Coin
Token Symbol	Alleluia
Token Decimal	18
Total Supply	1,000,000,000,000,000
Platform	BSC
Buy Tax Fee	6%
Sell Tax Fee	6%
Contract Creation Date	Jul 15, 2022
Liquidity Status	Not Available
Liquidity Lockup Time	Not Available
Compiler Version	v0.8.9+commit.e5eed63a
Optimization	Yes with 200 runs
Contract Address	0x0a724B9144D5d3C190584071d44001Af7198488F
Deployer Address	0x65455f81Bb4Dd22fd3c90f6320142e92BccceAEa
Owner Address	0xc120b9847b32e7bc7af9854dd59bdf63ce2939d1

Source Code

CRACKEN was commissioned by Hallelujah Coin to perform an audit based on the following smart contract:

https://bscscan.com/address/0x0a724B9144D5d3C190584071d44001Af7198488F

Smart Contract Vulnerability Checks

Vulnerability	Auto-Scan	Manual-Scan	Result
Unencrypted Private Data On-Chain	Complete	Complete	Low / No Risk
Code With No Effects	Complete	Complete	Low / No Risk
Message call with hardcoded gas amount	Complete	Complete	Low / No Risk
Hash Collisions with Multiple Variable Length Arguments	Complete	Complete	Low / No Risk
Unexpected Ether balance	Complete	Complete	Low / No Risk
Presence of unused variables	Complete	Complete	Low / No Risk
Right-To-Left-Override control character (U+202E)	Complete	Complete	Low / No Risk
Typographical Error	Complete	Complete	Low / No Risk
DoS With Block Gas Limit	Complete	Complete	Low / No Risk
Arbitrary Jump with Function Type Variable	Complete	Complete	Low / No Risk
Insufficient Gas Grieving	Complete	Complete	Low / No Risk
Incorrect Inheritance Order	Complete	Complete	Low / No Risk
Write to Arbitrary Storage Location	Complete	Complete	Low / No Risk
Requirement Violation	Complete	Complete	Low / No Risk
Missing Protection against Signature Replay Attacks	Complete	Complete	Low / No Risk
Weak Sources of Randomness from Chain Attributes	Complete	Complete	Low / No Risk
Authorization through tx. origin	Complete	Complete	Low / No Risk
Delegate call to Untrusted Callee	Complete	Complete	Low / No Risk

Vulnerability	Auto-Scan	Manual-Scan	Result
Use of Deprecated Solidity Functions	Complete	Complete	Low / No Risk
Assert Violation	Complete	Complete	Low / No Risk
Reentrancy	Complete	Complete	Low / No Risk
Unprotected SELF-DESTRUCT Instruction	Complete	Complete	Low / No Risk
Unprotected Ether Withdrawal	Complete	Complete	Low / No Risk
Outdated Compiler Version	Complete	Complete	Low / No Risk
Integer Overflow and Underflow	Complete	Complete	Low / No Risk
Function Default Visibility	Complete	Complete	Low / No Risk

Manual Code Review

Classification of Issues

Severity	Description
🔴 High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
🟠 Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
🟡 Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
🔵 Informational	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
🔴 High-Risk	0
🟠 Medium-Risk	0
🟡 Low-Risk	0
🔵 Informational	2
Total	2

● **Informational: Implementation of certain corrective actions or accepting the risk.**

Set max buy / sell tax fee

Description:

The owner can change the buy & sell fees up to 20%

```
function updateBuyFees(uint256 _operationsFee, uint256 _liquidityFee, uint256 _devFee)
external onlyOwner {
    buyOperationsFee = _operationsFee;
    buyLiquidityFee = _liquidityFee;
    buyDevFee = _devFee;

    buyTotalFees = buyLiquidityFee + buyOperationsFee + buyDevFee;
    require(buyTotalFees <= 200, "Must keep fees at 20% or less");
 }

 function updateSellFees(uint256 _operationsFee, uint256 _liquidityFee, uint256 _devFee)
external onlyOwner {
    sellOperationsFee = _operationsFee;
    sellLiquidityFee = _liquidityFee;
    sellDevFee = _devFee;

    sellTotalFees = sellLiquidityFee + sellOperationsFee + sellDevFee;
    require(sellTotalFees <= 200, "Must keep fees at 20% or less");
 }
```

Recommendation: fees are set to a reasonable amount

🔵 **Informational: Implementation of certain corrective actions or accepting the risk.**

Set max transaction amount

Description:

The max transaction amount must higher than 0.5%

```
function updateSwapTokensAtAmount(uint256 newAmount) external onlyOwner returns (bool){
 // require(newAmount >= totalSupply() * 1 / 100000, "Swap amount cannot be lower than 0.001% total supply.");
   require(newAmount <= totalSupply() * 5 / 1000, "Swap amount cannot be higher than 0.5% total supply.");
   swapTokensAtAmount = newAmount;
   return true;
 }

 function updateMaxAmount(uint256 newNum) external onlyOwner {
   require(newNum >= (totalSupply() * 5 / 1000) / 1e18, "Cannot set maxTransactionAmount lower than 0.5%");
   maxTransactionAmount = newNum * (10 ** 18);
 }
```

Privileged Functions

onlyOwner

Function Name	Parameters	Visibility
decreaseAllowance	address spender, uint256 subtractedValue	Public
enableTrading		Public
excludedFromFees	address account, bool excluded	Public
excludeFromMaxTransaction	address updAds, bool isEx	Public
increaseAllowance	address spender, uint256 addedValue	Public
transferFrom	address sender,address recipient,uint256 amount	Public
transferOwnership	address payable wallet	External
updateBuyFees	uint256 _operationsFee, uint256 _liquidityFee, uint256 _devFee	External
updateDevWallet	address newDevWallet	External
updateLiquidityWallet	address newLiquidityWallet	External
updateMarketingWallet	address newMarketingWallet	External
updateMaxAmount	uint256 newNum	External
updateSellFees	uint256 _operationsFee, uint256 _liquidityFee, uint256 _devFee	External

Contract Ownership

The contract ownership of Hallelujah Coin is not currently renounced. The ownership of the contract grants special powers to the protocol creators, making them the sole addresses that can call sensible ownable functions that may alter the state of the protocol.

The current owner is the address 0xc120b9847b32e7bc7af9854dd59bdf63ce2939d1 which can be viewed: HERE

The owner wallet has the power to call the functions displayed on the privileged functions list above, if the owner wallet is compromised these privileges could be exploited.

We recommend the team renounce ownership at the right timing if possible, or gradually migrate to a time lock with governing functionalities in respect of transparency and safety considerations.

Liquidity Overview

Liquidity Information

Parameter	Result
Pair Address	0xdbd5053ffa2b9824d46465026e1e06e7d9637e18
Alleluia Reserves	0.00 Alleluia
BNB Reserves	0.00 BNB
Liquidity Value	$0 USD
Liquidity Ownership	The token does not have liquidity at the moment of the audit

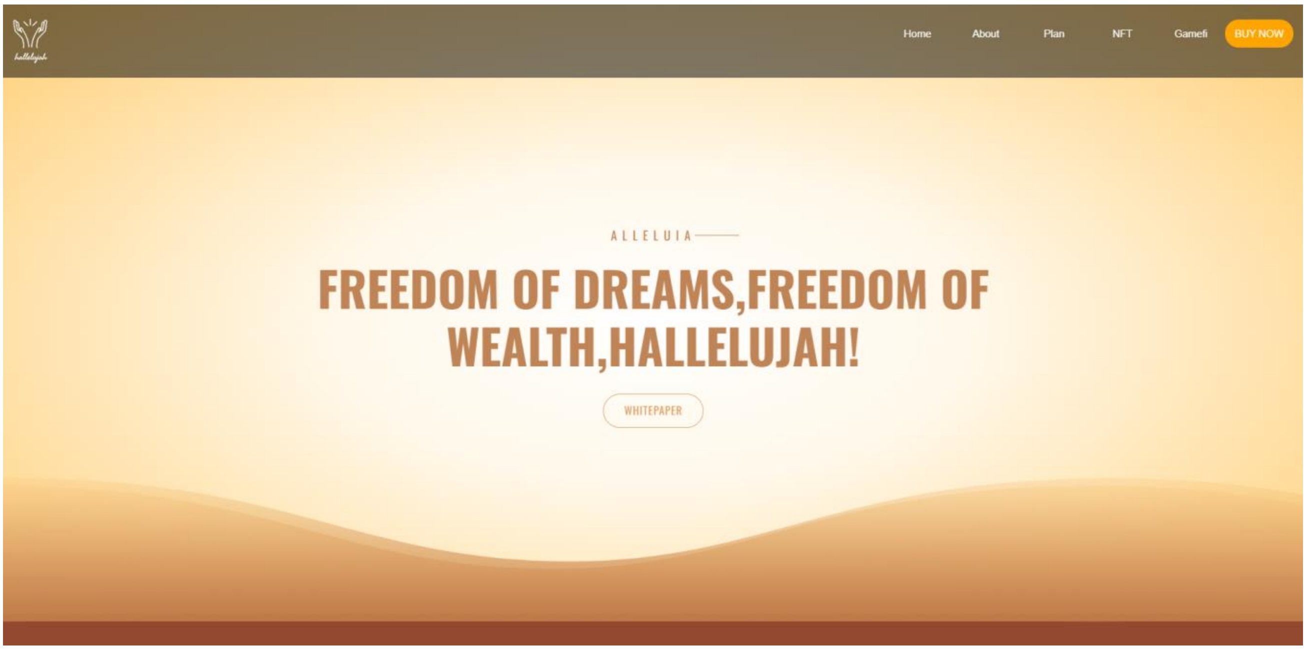
Tokenomics

Rank	Address	Quantity (Token)	Percentage
1	0x000000000000000000000000000000000000dead	400,000,000,000,000	40.0000%
2	0x884411285113b451f9146824ed9bf1adf748f1be	300,600,000,000,000	30.0600%
3	Pinksale: PinkLock V2	250,000,000,000,000	25.0000%
4	0xc120b9847b32e7bc7af9854dd59bdf63ce2939d1	49,400,000,000,000	4.9400%

Social Media Check

Social Media Type	Link	Result
Website	https://www.alleluia.top/	Checked
Twitter	https://twitter.com/Hallelujah_coin/	Checked
Telegram	https://t.me/hallelujah_global/	Checked

Website Review



- Mobile Friendly
- Contains no code errors
- SSL Secured
- No spelling errors

Audit Conclusion

- The owner cannot pause trading
- The owner cannot mint new tokens
- The owner cannot blacklist users
- The owner cannot set the max transaction amount lower than 0.5%
- The owner can change the buy/sell fee up to 20%.

AUDIT PASSED