KISHIELD

Security Audit

Godfather Token

April 21, 2022

Table of Contents

Audit Summary

This report has been prepared for Godfather Token on the Binance Chain network. KISHIELD provides both client-centered and user-centered examination of the smart contracts and their current status when applicable. This report represents the security assessment made to find issues and vulnerabilities on the source code along with the current liquidity and token holder statistics of the protocol.

A comprehensive examination has been performed, utilizing Cross Referencing, Static Analysis, In-House Security Tools, and line-by-line Manual Review.

The auditing process pays special attention to the following considerations:

- Ensuring contract logic meets the specifications and intentions of the client without exposing the user's funds to risk.
- Testing the smart contracts against both common and uncommon attack vectors.
- Inspecting liquidity and holders statistics to inform the current status to both users and client when applicable.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Verifying contract functions that allow trusted and/or untrusted actors to mint, lock, pause, and transfer assets.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Project Overview

Token Summary

Parameter	Result
Address	0xC100099bc314aED676A03634587c2C39E0af961F
Name	Godfather
Token Tracker	Godfather (GF)
Decimals	5
Supply	1,000,000
Platform	Binance Chain
compiler	v0.7.6+commit.7338295f
Optimization	Yes with 200 runs
LicenseType	Unlicense
Language	Solidity
Codebase	https://bscscan.com/address/0xc100099bc314aed676a03634587c2c39e0af961f#code
Url	https://godfather.app/

Main Contract Assessed

Name	Contract	Live
Godfather	0xC100099bc314aED676A03634587c2C39E0af961F	Yes

Smart Contract Vulnerability Checks

Vulnerability	Automatic 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 Griefing	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

Vulnerability	Automatic Scan	Manual Scan	Result
Authorization through tx.origin	Complete	Complete	✓ Low / No Risk
Delegatecall to Untrusted Callee	Complete	Complete	✓ Low / No Risk
Use of Deprecated Solidity Functions	Complete	Complete	✓ Low / No Risk
Assert Violation	Complete	Complete	✓ Low / No Risk
Reentrancy	Complete	Complete	✓ Low / No Risk
Unprotected SELFDESTRUCT Instruction	Complete	Complete	✓ Low / No Risk
Unprotected Ether Withdrawal	Complete	Complete	✓ Low / No Risk
Unchecked Call Return Value	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

Contract Ownership

The contract ownership of Godfather 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 0x7FB82B34F283e4ddD7D5F2Ec287D9A81Ed5dFa64 which can be viewed from:
HERE

The owner wallet has the power to call the functions displayed on the priviliged functions chart below, if the owner wallet is compromised this privileges could be exploited.

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

Important Notes To The Users:

- The owner cannot mint tokens after intial deployment.
- The transfer function is implemented correctly.
- The owner cannot stop Trading.
- The owner cannot change the max tx amount.
- The owner cannot change the fees amount.
- autoRebase and autoAddLiquidity are by default true at deployment time.
- Liquidity is added 2 days after the last liquidity addition.
- Once the owner renounces ownership of the contract, none of the following are applicable.
- Owner can withdraw all tokens from the contract to the treasuryReceiver address.
- Owner can enable/disable autoRebase and AutoAddLiquidity
- Owner can add and remove contracts from the bot blacklist.
- Owner can set wallets for fee exempt in setWhitelist function.
- Owner can transfer tokens and BNB from the contract.
- No high-risk Exploits/Vulnerabilities Were Found in token Source Code.

Audit Passed

Findings Summary

Classification of Issues

Severity	Description
🔴 High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, or impairment of the contract and its functions. Issues under this classification are recommended to be fixed with utmost urgency
🟠 Medium	Bugs or issues with that may be subject to exploit, though their impact is somewhat limited.Issues under this classification are recommended to be fixed as soon as possible.
🟡 Low	Effects are minimal in isolation and do not pose a significant danger to the project or its users. Issues under this classification are recommended to be fixed nonetheless.
🟣 Info	Consistency, syntax or style best practices. Generally pose a negligible level of risk, if any.

Findings

Severity	Found
🔴 High	0
🟠 Medium	0
🟡 Low	1
🟣 Info	4
Total	5

Findings

Variables could be declared as constant

ID	Severity	Contract	Function
01	● Informational	Godfather	variables feeDenominator, firePitFee, DEAD, ZERO, liquidityFee, insuranceFundFee, rebaseCycle, sellFee, treasuryFee

Description

Gas Optimization. Variables that are never changed could be declared as constant.

Recommendation

We recommend declaring those variables as constant.

Public function that could be declared external

ID	Severity	Contract	Function
02	● Informational	Godfather	Functions getLiquidityBacking,renounceOwnership, transferOwnership, setPairAddress

Description

Gas Optimization. Public function that could be declared external

Recommendation

Public functions that are never called by the contract should be declared external to save gas.

Uncommon decimals

ID	Severity	Contract	Function
03	● Informational	Godfather	DECIMALS = 5

Description

Most tokens use 18 or 10 decimal places, by having such a low value precision may be lost.

Recommendation

We advice making use of 18 or 10 decimal places.

Division before Multiplication

ID	Severity	Contract	Function
04	● Low	Godfather	function rebase(), takeFee(), getLiquidityBacking()

Description

Precision Loss. 'uint256 times = deltaTime.div(rebaseCycle); => uint256 epoch = times.mul(5);', 'gonAmount.div(feeDenominator).mul(_totalFee)', 'uint256 liquidityBalance = _gonBalances[pair].div(_gonsPerFragment);' Division before multiplication can result in truncation and less accurate results

Recommendation

Multiplication should be performed before division to not lose precision.

Uninitialized local variables

ID	Severity	Contract	Function
05	● Informational	Godfather	function rebase()

Description

Variable rebaseRate.

Recommendation

Initialize all the variables. If a variable is meant to be initialized to zero, explicitly set it to zero to improve code readability.

Priviliged Functions (onlyOwner)

Function Name	Parameters	Visibility
renounceOwnership	none	public
transferOwnership	address newOwner	public
withdrawAllToTreasury	none	external
withdrawAllToTreasury	none	external
setAutoRebase	bool _flag	external
setAutoAddLiquidity	bool _flag	external
setFeeReceivers	address _autoLiquidityReceiver, address _treasuryReceiver, address _insuranceFundReceiver, address _firePitAddress	external
setWhitelist	address userAddress	external
setBotBlacklist	address _botAddress, bool _flag	external
setPairAddress	address _pairAddress	public
setLP	address _address	external
setAutoLiquidityReceiver	address newAddress	external
setTreasuryReceiver	address newAddress	external
setInsuranceFundReceiver	address newAddress	external
setFirePitAddress	address newAddress	external
retrieveERC20Token	address tokenAddress, uint256 amount, address receiveAddress	external

Function Name	Parameters	Visibility
retrieveMainBalance	address receiveAddress	external

Statistics

Liquidity Info

Parameter	Result
Pair Address	0x0e78eA18B151d660a2FF5caBE9b5491F82A7F108
GF Reserves	0.00 GF
BNB Reserves	0.00 BNB
Liquidity Value	$0 USD

Token (GF) Holders Info

Parameter	Result
GF Percentage Burnt	0.00%
GF Amount Burnt	0 GF
Top 10 Percentage Own	100.00%
Top 10 Amount Owned	1,000,000 GF
Top 10 Aprox Value	$NaN USD

LP (GF/BNB) Holders Info

Parameter	Result
GF/BNB % Burnt	0.00%
GF/BNB Amount Burnt	0 GF
Top 10 Percentage Owned	0.00%
Top 10 Amount Owned	0 GF
Locked Tokens Percentage	0.00%
Locked Tokens Amount	0 GF

* All the data diplayed above was taken on-chain at block 17140134
* The tokens on industry-standard burn wallets are not included on the top 10 wallets calculations

Liquidity Ownership

The token does not have liquidity at the moment of the audit, block 17140134

KISHIELD

Disclaimer

KISHIELD has conducted an independent audit to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the codes that were provided for the scope of this audit. This audit report does not constitute agreement, acceptance or advocation for the Project that was audited, and users relying on this audit report should not consider this as having any merit for financial advice in any shape, form or nature. The contracts audited do not account for any economic developments that may be pursued by the Project in question, and that the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are completely free of exploits, bugs, vulnerabilities or deprecation of technologies.

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. Information is provided 'as is', and KISHIELD is under no covenant to the completeness, accuracy or solidity of the contracts audited. In no event will KISHIELD or its partners, employees, agents or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions and/or actions with regards to the information provided in this audit report.

The assessment services provided by KISHIELD is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.