CFG NINJA AUDITS

Security Assessment

ZOR Token

October 11, 2022

ZORAFFLE

ARE YOU IN-IT TO WIN-IT?

Table of Contents

Assessment Summary

This report has been prepared for ZOR Token on the Binance Smart Chain network. CFGNINJA 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:

- 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.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
- Thorough line-by-line manual review of the entire codebase by industry experts.

Technical Findings Summary

Classification of Risk

Severity	Description
Critical	Risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
Major	Risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
Medium	Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform
Minor	Risks can be any of the above but on a smaller scale. They generally do not compromise the overall integrity of the Project, but they may be less efficient than other solutions.
Informational	Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.

Findings

Severity	Found	Pending	Resolved
Critical	0	0	0
Major	0	0	0
Medium	0	0	0
Minor	1	1	0
Informational	0	0	0
Total	1	0	0

Project Overview

Token Summary

Parameter	Result
Address	0x54a7D204866EE5889305bf4554be13c3ef00d91f
Name	ZOR
Token Tracker	ZOR (ZOR)
Decimals	18
Supply	1,000,000,000,000
Platform	Binance Smart Chain
compiler	v0.8.7+commit.e28d00a7
Contract Name	ZoRaffle
Optimization	Yes with 200 runs
LicenseType	MIT
Language	Solidity
Codebase	https://bscscan.com/address/0x54a7d204866ee5889305bf4554be13c3ef00d91f#code
Payment Tx	0xd40dd501a834810aec627695caa317d78971c1cbf5d1f275cd4a75135d86d5d9

Project Overview

Risk Analysis Summary

Parameter	Result
Buy Tax	0%
Sale Tax	0%
Is honeypot?	Clean
Can edit tax?	Yes
Is anti whale?	No
Is blacklisted?	No
Is whitelisted?	No
Holders	Clean
Security Score	95/100
Auditor Score	95/100
Confidence Level	High

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.

Main Contract Assessed Contract Name

Name	Contract	Live
ZOR	0x54a7D204866EE5889305bf4554be13c3ef00d91f	Yes

TestNet Contract Assessed Contract Name

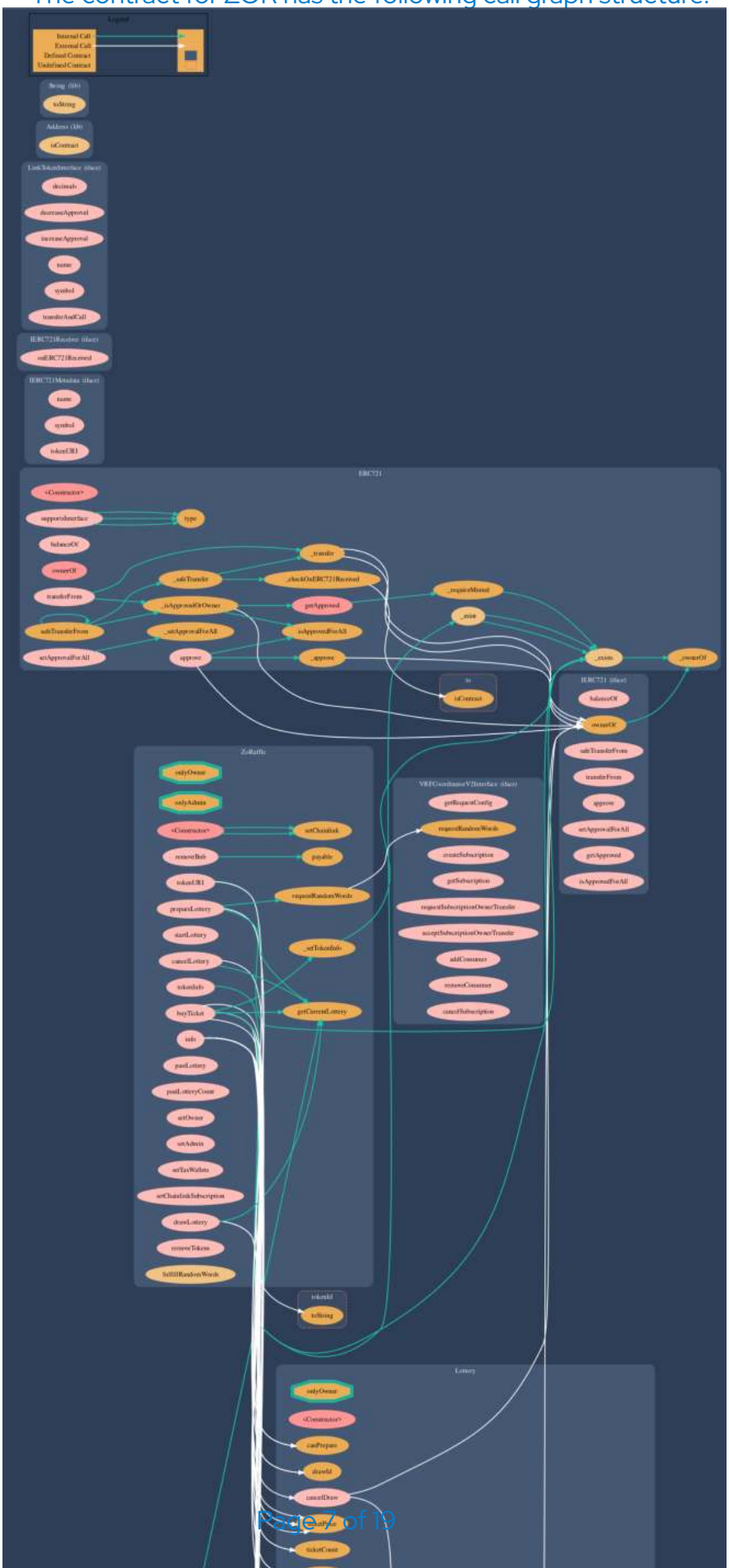
Name	Contract	Live
ZOR	0x9e9A1E1e9f2E54c43aB52cB4499Ab5A9dE151551	Yes

Solidity Code Provided

SolID	File Sha-1	FileName
zooraffle	14a0688d0a90951c4918d974a6b26ce089c997ca	zooraffle.sol

Call Graph

The contract for ZOR has the following call graph structure.



KYC Information

The Project Owners of ZOR have provided KYC Documentation.

KYC Certificated can be found on the Following: KYC Data

KYC Information Notes:

Auditor Notes: Customer is KYC

Project Owner Notes:

Smart Contract Vulnerability Checks

ID	Severity	Name	File	location
SWC-100	Pass	Function Default Visibility	zooraffle.sol	L: 0 C: 0
SWC-101	Pass	Integer Overflow and Underflow.	zooraffle.sol	L: 0 C: 0
SWC-102	Pass	Outdated Compiler Version file.	zooraffle.sol	L: 0 C: 0
SWC-103	Pass	A floating pragma is set.	zooraffle.sol	L: 0 C: 0
SWC-104	Pass	Unchecked Call Return Value.	zooraffle.sol	L: 0 C: 0
SWC-105	Pass	Unprotected Ether Withdrawal.	zooraffle.sol	L: 0 C: 0
SWC-106	Pass	Unprotected SELFDESTRUCT Instruction	zooraffle.sol	L: 0 C: 0
SWC-107	Pass	Read of persistent state following external call.	zooraffle.sol	L: 0 C: 0
SWC-108	Pass	State variable visibility is not set..	zooraffle.sol	L: 304 C: 8, L: 304 C: 42, L: 304 C: 56
SWC-109	Pass	Uninitialized Storage Pointer.	zooraffle.sol	L: 0 C: 0
SWC-110	Pass	Assert Violation.	zooraffle.sol	L: 0 C: 0
SWC-111	Pass	Use of Deprecated Solidity Functions.	zooraffle.sol	L: 0 C: 0

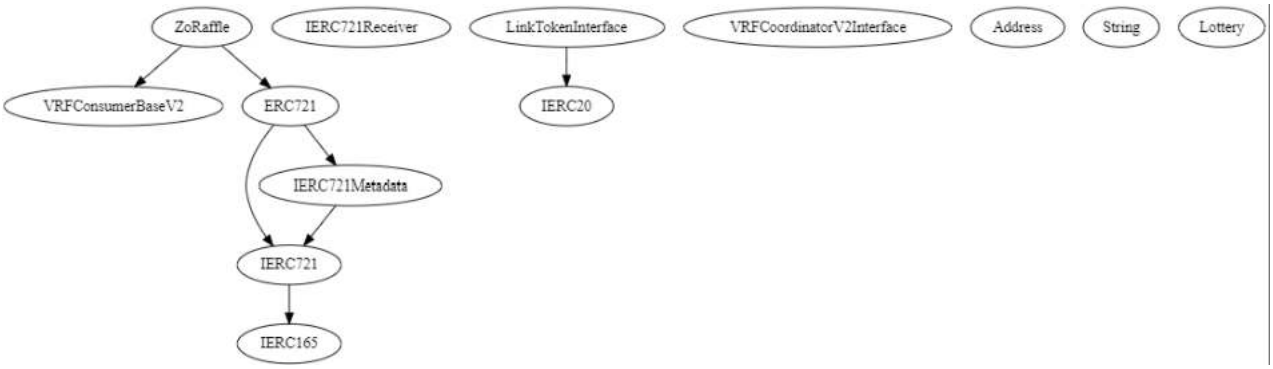
ID	Severity	Name	File	location
SWC-112	Pass	Delegate Call to Untrusted Callee.	zooraffle.sol	L: 0 C: 0
SWC-113	Pass	Multiple calls are executed in the same transaction.	zooraffle.sol	L: 0 C: 0
SWC-114	Pass	Transaction Order Dependence.	zooraffle.sol	L: 0 C: 0
SWC-115	Pass	Authorization through tx.origin.	zooraffle.sol	L: 0 C: 0
SWC-116	Pass	A control flow decision is made based on The block.timestamp environment variable.	zooraffle.sol	L: 0 C: 0
SWC-117	Pass	Signature Malleability.	zooraffle.sol	L: 0 C: 0
SWC-118	Pass	Incorrect Constructor Name.	zooraffle.sol	L: 0 C: 0
SWC-119	Pass	Shadowing State Variables.	zooraffle.sol	L: 0 C: 0
SWC-120	Pass	Potential use of block.number as source of randonmness.	zooraffle.sol	L: 0 C: 0
SWC-121	Pass	Missing Protection against Signature Replay Attacks.	zooraffle.sol	L: 0 C: 0
SWC-122	Pass	Lack of Proper Signature Verification.	zooraffle.sol	L: 0 C: 0
SWC-123	Pass	Requirement Violation.	zooraffle.sol	L: 0 C: 0
SWC-124	Pass	Write to Arbitrary Storage Location.	zooraffle.sol	L: 0 C: 0
SWC-125	Pass	Incorrect Inheritance Order.	zooraffle.sol	L: 0 C: 0
SWC-126	Pass	Insufficient Gas Griefing.	zooraffle.sol	L: 0 C: 0

ID	Severity	Name	File	location
SWC-127	Pass	Arbitrary Jump with Function Type Variable.	zooraffle.sol	L: 0 C: 0
SWC-128	Pass	DoS With Block Gas Limit.	zooraffle.sol	L: 0 C: 0
SWC-129	Pass	Typographical Error.	zooraffle.sol	L: 0 C: 0
SWC-130	Pass	Right-To-Left-Override control character (U +202E).	zooraffle.sol	L: 0 C: 0
SWC-131	Pass	Presence of unused variables.	zooraffle.sol	L: 0 C: 0
SWC-132	Pass	Unexpected Ether balance.	zooraffle.sol	L: 0 C: 0
SWC-133	Pass	Hash Collisions with Multiple Variable Length Arguments.	zooraffle.sol	L: 0 C: 0
SWC-134	Pass	Message call with hardcoded gas amount.	zooraffle.sol	L: 0 C: 0
SWC-135	Pass	Code With No Effects (Irrelevant/Dead Code).	zooraffle.sol	L: 0 C: 0
SWC-136	Pass	Unencrypted Private Data On-Chain.	zooraffle.sol	L: 0 C: 0

We scan the contract for additional security issues using MYTHX and industry-standard security scanning tools.

Inheritance

The contract for ZOR has the following inheritance structure.



Privileged Functions (onlyOwner)

Function Name	Parameters	Visibility
buyTicket		external
prepareLotto		external
drawLotto		external
cancelDraw		external
soldTickets		external
canBuy		external
canDraw		external
canPrepare		external
startLottery		external
prepareLottery		external
drawLottery		external
cancelLottery		external
setOwner		external
setAdmin		external

Function Name	Parameters	Visibility
setTaxwallets		external
setChainlink		external
setChainlinksubscription		external
removebnb		external
removetokens		external

Assessment Results

- Contract is a Lottery Contract.
- No high-risk Exploits/Vulnerabilities Were Found in the Source Code.
- Contract developed by Adam Leroix-Sainz.

Audit Passed

ZOR-01 | Potential Sandwich Attacks.

Category	Severity	Location	Status
Security	Minor	zooraffle.sol: 594,13	In Progress

Description

A sandwich attack might happen when an attacker observes a transaction swapping tokens or adding liquidity without setting restrictions on slippage or minimum output amount. The attacker can manipulate the exchange rate by frontrunning (before the transaction being attacked) a transaction to purchase one of the assets and make profits by back running (after the transaction being attacked) a transaction to sell the asset. The following functions are called without setting restrictions on slippage or minimum output amount, so transactions triggering these functions are vulnerable to sandwich attacks, especially when the input amount is large:

- swapExactTokensForETHSupportingFeeOnTransferTokens()
- addLiquidityETH()

Remediation

We recommend setting reasonable minimum output amounts, instead of 0, based on token prices when calling the aforementioned functions.

Referrences:

What Are Sandwich Attacks in DeFi — and How Can You Avoid Them?.

Social Media Checks

Social Media	URL	Result
Twitter	https://twitter.com/Zoraffle	Pass
Instagram	https://www.instagram.com/zoraffle/	Pass
Website	https://www.zorraffle.com	Pass
Telegram	https://t.me/nutgainofficial/113	Pass

We recommend to have 3 or more social media sources including a completed working websites.

Social Media Information Notes:

Auditor Notes: undefined

Project Owner Notes:

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that actagainst the nature of decentralization, such as explicit ownership or specialized access roles incombination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimalEVM opcodes resulting in a reduction on the total gas cost of a transaction.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on howblock.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functionsbeing invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that mayresult in a vulnerability.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to makethe codebase more legible and, as a result, easily maintainable.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code,such as a constructor assignment imposing different require statements on the input variables than a setterfunction.

Coding Best Practices

ERC 20 Conding Standards are a set of rules that each developer should follow to ensure the code meet a set of creterias and is readable by all the developers.

Disclaimer

CFGNINJA has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocation for the Project, and users relying on this 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 the Project in question may pursue, and 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 entirely 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. Information is provided 'as is, and CFGNINJA is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will CFGNINJA 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 or actions with regards to the information provided in this audit report.

The assessment services provided by CFGNINJA are subject to dependencies and are under continuing development. You agree that your access 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 with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.