

# AUDIT REPORT

SecureWise

VIBEZONEONLINE (VZO)



# Quick Result

Quick Result	Status
Owner can mint new token?	Not Detected
Owner can update tax over 25% ?	Pass
Owner can pause trade ?	Not Detected
Owner can enable trading ?	Not Detected
Owner can add Blacklist ?	Not Detected
Owner can set Max Tx ?	Not Detected
Owner can set Max Wallet Amount ?	Not Detected
KYC ?	<u>Done</u>

Page 10 for more details

VibeZoneOnline (VZO) as **PASSED** the smart contract audit.

# Findings

Risk Classification	Description
High	Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, of the contract and its functions. Must be fixed as soon as possible.
Medium	Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Must 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.
Informational	A vulnerability that have informational character but is not effecting any of the code

Severity	Found	Pending	Resolved
High	0	0	0
Medium	0	0	0
Low	2	0	0
Informational	3	0	0
Total	5	0	0

# Contents

01	Quick Result
02	Findings
04	Overview
05	Auditing Approach and Methodologies
06	Findings Summary
07	Function Privileges
08	Inheritance Graph
10	Manual Review
15	Disclaimer

# Overview

**Token Name:** VibeZoneOnline (**VZO**)

**Language:** Solidity

**Contract Address:** 0xF394C9F6b719E4756fEa32F7436c43c71cd8297A

**Network:** Binance Smart Chain

**Supply:** 100000000000

**KYC:** Done

**Website:** <https://www.vibezone.online>

**Twitter:** [https://twitter.com/vibezone\\_app](https://twitter.com/vibezone_app)

**Telegram:** [https://t.me/VibeZone\\_App](https://t.me/VibeZone_App)

**Report Date:** June 21, 2023

**Testnet:**

<https://testnet.bscscan.com/address/0xD7dCF5449D1E3b37E42a725A5De861DbB9877F2D>

# Auditing Approach and Methodologies

SecureWise has performed starting with analyzing the code, issues, code quality, and libraries. Reviewed line-by-line by our team. Finding any potential issue like race conditions, transaction-ordering dependence, timestamp dependence, and denial of service attacks.

## Methodology

- Understanding the size, scope and functionality of your project's source code
- Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Testing and automated analysis of the Smart Contract to determine proper logic has been followed throughout the whole process
- Deploying the code on testnet using multiple live test
- Analyzing a program to determine the specific input that causes different parts of a program to execute its functions.
- Checking whether all the libraries used in the code are on the latest version.

## Goals

Smart Contract System is secure, resilient and working according to the specifications and without any vulnerabilities.

## Risk Classification

**High:** Exploits, vulnerabilities or errors that will certainly or probabilistically lead towards loss of funds, control, of the contract and its functions. Must be fixed as soon as possible.

**Medium:** Bugs or issues with that may be subject to exploit, though their impact is somewhat limited. Must 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.

**Informational:** A vulnerability that have informational character but is not affecting any of the code


# Findings Summary

SecureWise has applied the automated and manual analysis of Smart Contract and were reviewed for common contract vulnerabilities and centralized exploits

## Centralization Findings

	Owner can set fees with a maximum limit of 5%.
	Owner is capable of withdrawing claim stuck tokens from the contract.
	Owner has the ability to exclude accounts from being charged fees.
	Swap token at amount can be modified by the owner.

## Logical Findings

	Old versions of Solidity
---	--------------------------

**Page 10** for more details

VibeZoneOnline (VZO) as **PASSED** the smart contract audit.

## Function Privileges

```

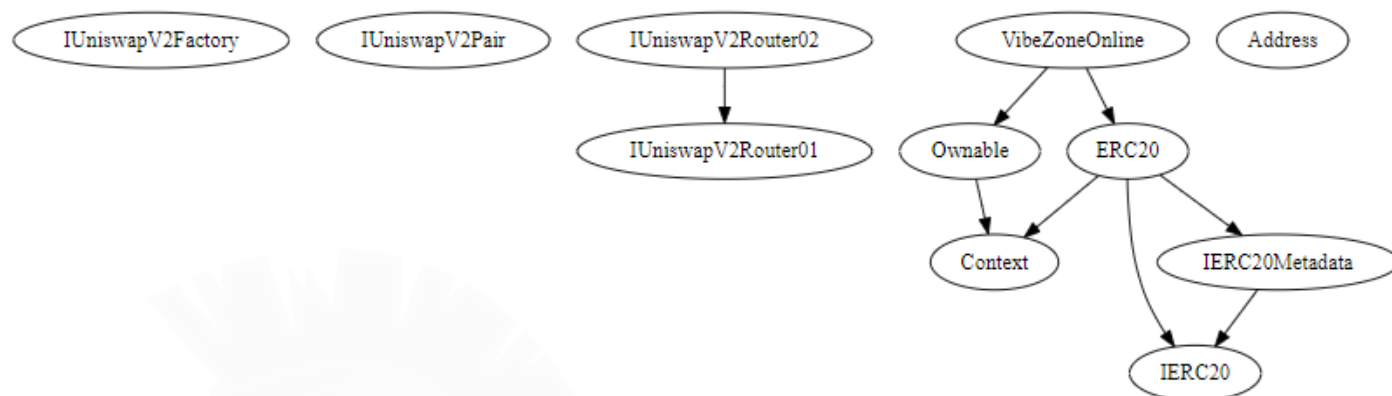
| **VibeZoneOnline** | Implementation | ERC20, Ownable |||
| L | <Constructor> | Public ! | ● | ERC20 |
| L | <Receive Ether> | External ! | ● | NO ! |
| L | claimStuckTokens | External ! | ● | onlyOwner |
| L | excludeFromFees | External ! | ● | onlyOwner |
| L | isExcludedFromFees | Public ! | | NO ! |
| L | updateBuyTax | External ! | ● | onlyOwner |
| L | updateSellTax | External ! | ● | onlyOwner |
| L | updateWalletToWalletTransferTax | External ! | ● | onlyOwner |
| L | changeTaxReceiver | External ! | ● | onlyOwner |
| L | _transfer | Internal 🔒 | ● | |
| L | setSwapTokensAtAmount | External ! | ● | onlyOwner |
| L | swapAndSendTax | Private 🔒 | ● | |

```

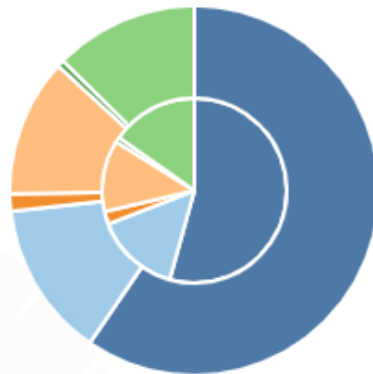




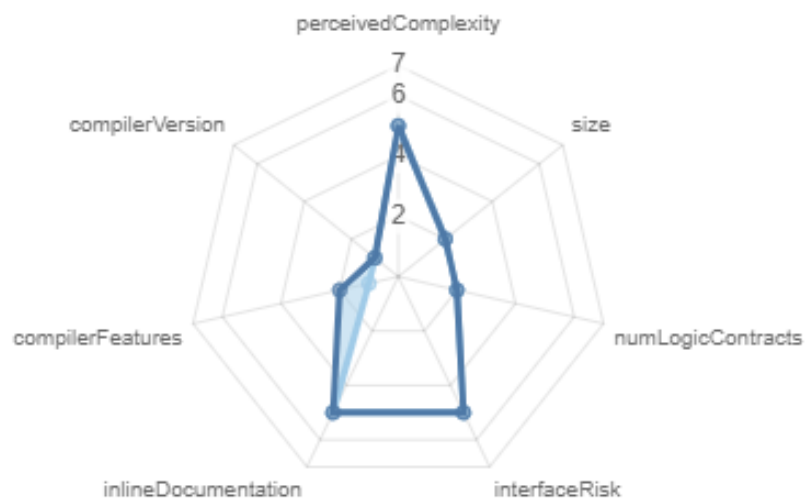
# Inheritance Graph



## Source Lines



## Risk



# Manual Review

## Low

**Owner can set fees with a maximum limit of 5%.**

```
function updateBuyTax(uint256 _taxOnBuy) external onlyOwner {
    require(_taxOnBuy <= 5, "Buy Tax cannot be more than 5%");
    taxOnBuy = _taxOnBuy;
    emit UpdateBuyTax(taxOnBuy);
}

function updateSellTax(uint256 _taxOnSell) external onlyOwner {
    require(_taxOnSell <= 5, "Sell Tax cannot be more than 5%");
    taxOnSell = _taxOnSell;
    emit UpdateSellTax(taxOnSell);
}

function updateWalletToWalletTransferTax(uint256 _walletToWalletTransferTax) external onlyOwner {
    require(_walletToWalletTransferTax <= 5, "Wallet to Wallet Transfer Tax cannot be more than 5%");
    walletToWalletTransferTax = _walletToWalletTransferTax;
    emit UpdateWalletToWalletTransferTax(walletToWalletTransferTax);
}
```

## Description

**updateBuyTax** **updateSellTax** **updateWalletToWalletTransferTax** functions allows the contract owner to update the tax rate applied to buy/sell/transfer. The function ensures that the tax rate provided is not more than 5%.

## Recommendation

Tax rate is already capped at 5%, no specific recommendation is required for this function at this time. However, it is important to periodically review and reassess the tax rate to ensure it aligns with the market dynamics and does not discourage potential buyers.

# Manual Review

## Low

**Owner is capable of withdrawing claim stuck tokens from the contract.**

```
function claimStuckTokens(address token) external onlyOwner {
    if (token == address(0x0)) {
        payable(msg.sender).sendValue(address(this).balance);
        return;
    }
    IERC20 ERC20token = IERC20(token);
    uint256 balance = ERC20token.balanceOf(address(this));
    ERC20token.transfer(msg.sender, balance);
}
```

## Description

usage of **payable(msg.sender).sendValue()** to transfer native tokens is generally acceptable. However, this method has a gas limit constraint, and if the contract's balance exceeds this limit, the transfer may fail. Additionally, using **sendValue** can result in loss of funds if the recipient address does not have a fallback function to receive the transferred funds. **claimStuckTokens** function provides the contract owner with the ability to claim stuck tokens, either native tokens or ERC20 tokens, that are held by the contract

## Recommendation

Consider using **transfer** or **transferFrom** functions instead of **sendValue** to transfer native tokens. This approach will provide more reliability and prevent the potential loss of funds. Additionally, ensure that the recipient address has a fallback function or is a trusted contract capable of handling the received tokens. I recommend adding a **require** statement to explicitly check that the token provided is not the contract itself. This can be achieved by comparing the token address with the contract's address (**address(this)**).

# Manual Review

## Informational

Owner has the ability to exclude accounts from being charged fees.

```
function excludeFromFees(address account, bool excluded) external onlyOwner{
    require(!_isExcludedFromFees[account] != excluded, "Account is already the value of 'excluded'");
    _isExcludedFromFees[account] = excluded;

    emit ExcludeFromFees(account, excluded);
}
```

## Description

**excludeFromFees** allows the contract owner to modify the exclusion status of an account from fees by updating the `_isExcludedFromFees` mapping.

## Recommendation

No specific recommendation is necessary for the **excludeFromFees** function at this time. However, it is important to ensure that the function is being used appropriately and that the owner's ability to exclude or include accounts from fees is clearly documented and understood.

# Manual Review

## Informational

Swap token at amount can be modified by the owner.

```
function setSwapTokensAtAmount(uint256 newAmount) external onlyOwner{
    require(newAmount > totalSupply() / 1000000, "SwapTokensAtAmount must be greater than 0.0001% of total supply");
    swapTokensAtAmount = newAmount;
    emit SwapTokensAtAmountUpdated(swapTokensAtAmount);
}
```

## Description

**swapTokensAtAmount** variable. The function includes a requirement that ensures the new amount is greater than a specific threshold based on a percentage calculation of the total supply.

## Recommendation

Given that the range of values is safe and the function appears to be working as intended, no specific recommendation is necessary for the **setSwapTokensAtAmount** function at this time.

# Manual Review

## Informational

### Old Version of Solidity

`pragma solidity 0.8.18;`

### Description

*Using an old version prevents access to new Solidity security checks. We also recommend avoiding complex pragma statement. Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly.*

### Recommendation

*Use a simple pragma version that allows any of these versions. Consider using the latest version of Solidity for testing.*

# Disclaimer

SecureWise provides the smart contract audit of solidity. Audit and report are for informational purposes only and not, nor should be considered, as an endorsement to engage with, invest in, participate, provide an incentive, or disapprove, criticise, discourage, or purport to provide an opinion on any particular project or team.

This audit report doesn't provide any warranty or guarantee regarding the nature of the technology analysed. These reports, in no way, provide investment advice, nor should be used as investment advice of any sort. Investors must always do their own research and manage their risk.

**DISCLAIMER:** By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and SecureWise and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) SecureWise owe no duty of care towards you or any other person, nor does SecureWise make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and SecureWise hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, SecureWise hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against SecureWise, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.





# AUDIT REPORT

SecureWise



[securewise.org](https://securewise.org)



[t.me/securewisehub](https://t.me/securewisehub)



[twitter.com/securewiseAudit](https://twitter.com/securewiseAudit)

