SecureWise

AUDIT REPORT

SecureWise

BABY WIKICAT

SecureWise

Table of Contents

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.

Overview

Token Name: Baby WikiCat **(BWKC)**

Methodology: Automated Analysis, Manual Code Review

Language: Solidity

Contract Address: 0x99e4ffeE5367592A82B5d4Be6F30144F400388fC

ContractLink: https://bscscan.com/address/0x99e4ffeE5367592A82B5d4Be6F30144F400388fC

Network: Binance Smart Chain (BSC)

Decimals: 9

Supply: 1.000.000.000.000.000

Website: https://www.babywikicat.co

Twitter: https://twitter.com/Babywkc

Telegram: https://t.me/Babywkc

Report Date: February 21, 2023

renounceOwnership

https://bscscan.com/tx/0x9bf4e2f7e2c4b10d6602f877e9a989bb298af96a1e94e7de5e4aac9ed828d2cc

Quick Result

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

⚠	Auto liquidity is going to an externally owned account

visit **page 13** for all details

renounceOwnership

https://bscscan.com/tx/0x9bf4e2f7e2c4b10d6602f877e9a989bb298af96a1e94e7de5e4aac9ed828d2cc

Contract has renounced ownership. Before renounce some function were called..
https://bscscan.com/txs?a=0x99e4ffee5367592a82b5d4be6f30144f400388fc&p=21

0xdbc2774001a12328f9...	Set Is Timelock ...	24809739	37 days 9 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.00021888
0x212816be94cf7c162f9...	Set Treasury Fee...	24809563	37 days 9 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.0001637
0x2178c305995c199637...	Set Marketing Wa...	24809425	37 days 9 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.00031
0x522c2049238ebacbad...	Set Is Tx Limit ...	24809198	37 days 9 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.00021865
0x6dffa8148d76308381f...	Set Is Dividend ...	24809179	37 days 9 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.00028343
0xcb863407f9301a5d05f...	Set Is Fee Exemp...	24809156	37 days 9 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.0002191
0x82d4ab266177c6ca93...	Set Buy Tax	24808739	37 days 10 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.0001368
0x6c291685568685e62f...	Trading Status	24810563	37 days 8 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.00014115
0x0990b5a7eeb9f41545...	Set Buy Tax	24810545	37 days 8 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.0001368
0x5c971c250aaef1ba42f...	Set Is Fee Exemp...	24809762	37 days 9 hrs ago	0xeb25e9a829a656b3bf...	IN	0x99e4ffee5367592a82b...	0 BNB	0.0002191

then ownership was renounced

Input Data:

```
Function: transferOwnership(address adr)

MethodID: 0xf2fde38b
[0]:  000000000000000000000000000000000000000000000000000000000000dead
```

Baby WikiCat (BWKC) has succesfully **PASSED** the smart contract audit with **MEDIUM** and **LOW** severity issue

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 fxed as soon as possible.

Low: Effects are minimal in isolation and do not pose a signifcant danger to the project or its users. Issues under this classifcation are recommended to be fixed nonetheless.

Automated Analysis

Symbol	Meaning
🔴	Function can modify state
💵	Function is payable

Contract	Type	Bases		
L	Function Name	Visibility	Mutability	Modifiers
Project	Implementation			
SafeMath	Library			
L	add	Internal 🔒		
L	sub	Internal 🔒		
L	sub	Internal 🔒		
L	mul	Internal 🔒		
L	div	Internal 🔒		
L	div	Internal 🔒		
IBEP20	Interface			
L	totalSupply	External ❗		NO ❗
L	decimals	External ❗		NO ❗
L	symbol	External ❗		NO ❗
L	name	External ❗		NO ❗
L	getOwner	External ❗		NO ❗
L	balanceOf	External ❗		NO ❗
L	transfer	External ❗	🔴	NO ❗
L	allowance	External ❗		NO ❗
L	approve	External ❗	🔴	NO ❗
L	transferFrom	External ❗	🔴	NO ❗
Auth	Implementation			
L		Public ❗	🔴	NO ❗
L	authorize	Public ❗	🔴	onlyOwner
L	unauthorize	Public ❗	🔴	onlyOwner
L	isOwner	Public ❗		NO ❗
L	isAuthorized	Public ❗		NO ❗

Automated Analysis

└	transferOwnership	Public !	🔴	onlyOwner
IDEXFactory	Interface			
└	createPair	External !	🔴	NO !
IDEXRouter	Interface			
└	factory	External !		NO !
└	WETH	External !		NO !
└	addLiquidity	External !	🔴	NO !
└	addLiquidityETH	External !	💵	NO !
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !	🔴	NO !
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External !	💵	NO !
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External !	🔴	NO !
IDividendDistributor	Interface			
└	setDistributionCriteria	External !	🔴	NO !
└	setShare	External !	🔴	NO !
└	deposit	External !	💵	NO !
└	process	External !	🔴	NO !
DividendDistributor	Implementation	IDividendDistributor		
└		Public !	🔴	NO !
└	setDistributionCriteria	External !	🔴	onlyToken
└	setShare	External !	🔴	onlyToken
└	deposit	External !	💵	onlyToken
└	process	External !	🔴	onlyToken
└	shouldDistribute	Internal 🔒		
└	distributeDividend	Internal 🔒	🔴	
└	claimDividend	External !	🔴	NO !
└	getUnpaidEarnings	Public !		NO !

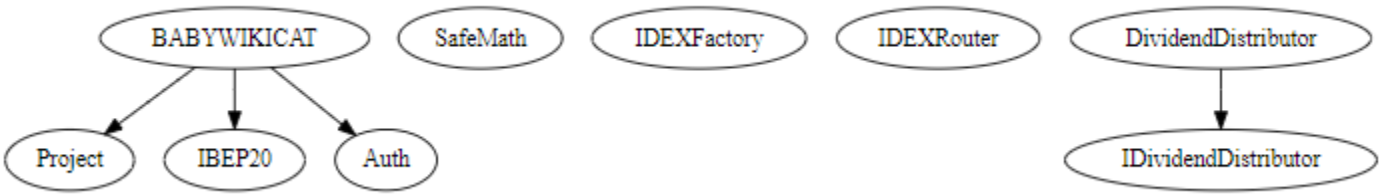
Automated Analysis

└	getCumulativeDividends	Internal 🔒		
└	addShareholder	Internal 🔒	🔴	
└	removeShareholder	Internal 🔒	🔴	
BABYWIKICAT	Implementation	Project, IBEP20, Auth		
└		Public ❗	🔴	Auth
└		External ❗	💵	NO ❗
└	totalSupply	External ❗		NO ❗
└	decimals	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	name	External ❗		NO ❗
└	getOwner	External ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	allowance	External ❗		NO ❗
└	approve	Public ❗	🔴	NO ❗
└	approveMax	External ❗	🔴	NO ❗
└	transfer	External ❗	🔴	NO ❗
└	transferFrom	External ❗	🔴	NO ❗
└	setMaxWalletPercent_base1000	External ❗	🔴	onlyOwner
└	setMaxTxPercent_base1000	External ❗	🔴	onlyOwner
└	setBuyTax	External ❗	🔴	onlyOwner
└	setTxLimit	External ❗	🔴	authorized
└	setBurnTo	External ❗	🔴	onlyOwner
└	setBuyBurnFee	External ❗	🔴	onlyOwner
└	setSwapBurnFee	External ❗	🔴	onlyOwner
└	_transferFrom	Internal 🔒	🔴	
└	_basicTransfer	Internal 🔒	🔴	
└	checkTxLimit	Internal 🔒		
└	shouldTakeFee	Internal 🔒		
└	takeFee	Internal 🔒	🔴	

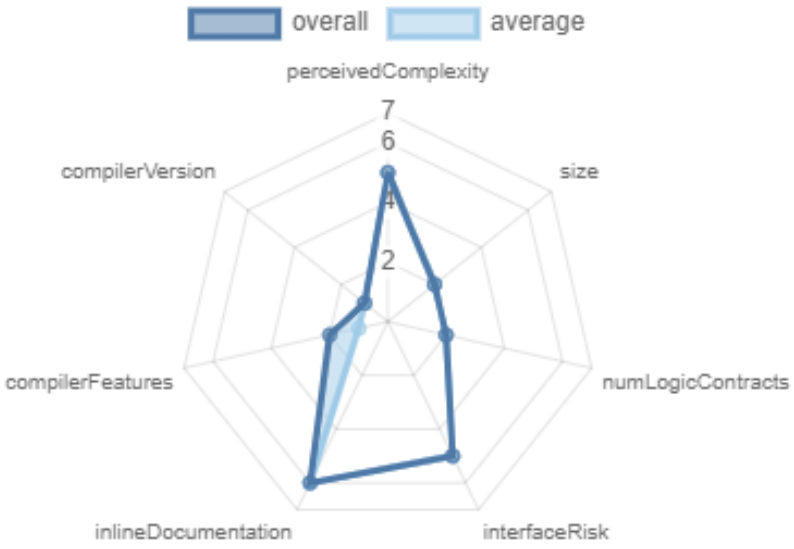
Automated Analysis

L	shouldSwapBack	Internal 🔒		
L	clearStuckBalance	External !	●	authorized
L	clearStuckBalance_sender	External !	●	authorized
L	tradingStatus	Public !	●	onlyOwner
L	cooldownEnabled	Public !	●	onlyOwner
L	swapBack	Internal 🔒	●	swapping
L	setIsDividendExempt	External !	●	authorized
L	manage*blacklist*and*dividend*exempt	Public !	●	onlyOwner
L	manage*burn*exempt	Public !	●	onlyOwner
L	enable_blacklist	Public !	●	onlyOwner
L	manage_blacklist	Public !	●	onlyOwner
L	setIsFeeExempt	External !	●	authorized
L	setIsTxLimitExempt	External !	●	authorized
L	setIsTimelockExempt	External !	●	authorized
L	setSwapFees	External !	●	authorized
L	setTreasuryFeeReceiver	External !	●	authorized
L	setMarketingWallet	External !	●	authorized
L	setFeeReceivers	External !	●	authorized
L	setSwapBackSettings	External !	●	authorized
L	setTargetLiquidity	External !	●	authorized
L	setDistributionCriteria	External !	●	authorized
L	setDistributorSettings	External !	●	authorized
L	getCirculatingSupply	Public !		NO !
L	getLiquidityBacking	Public !		NO !
L	isOverLiquified	Public !		NO !
L	multiTransfer	External !	●	onlyOwner
L	multiTransfer_fixed	External !	●	onlyOwner

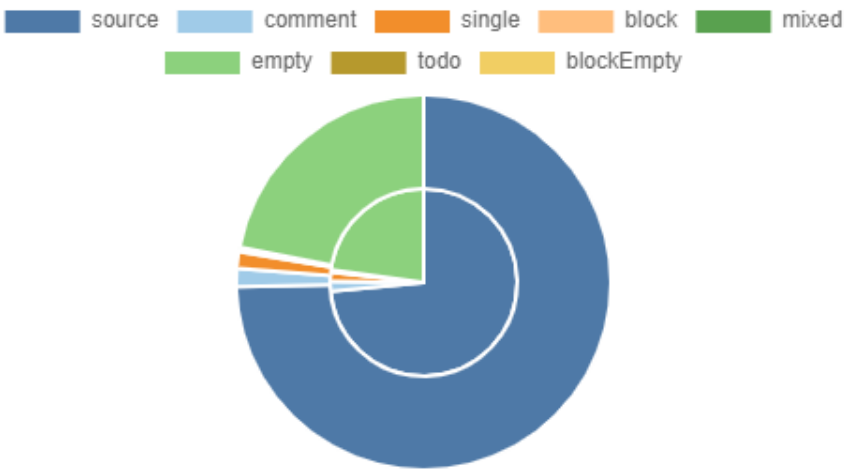
Inheritance Graph



Risk



Source Lines



Contract Summary

Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
5	4	815	745	560	13	579	
5	4	815	745	560	13	579	

Components

Contracts	Libraries	Interfaces	Abstract
2	1	4	2

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

Public	Payable
77	5

External	Internal	Private	Pure	View
60	78	0	11	22

StateVariables

Total	Public
63	33

Capabilities

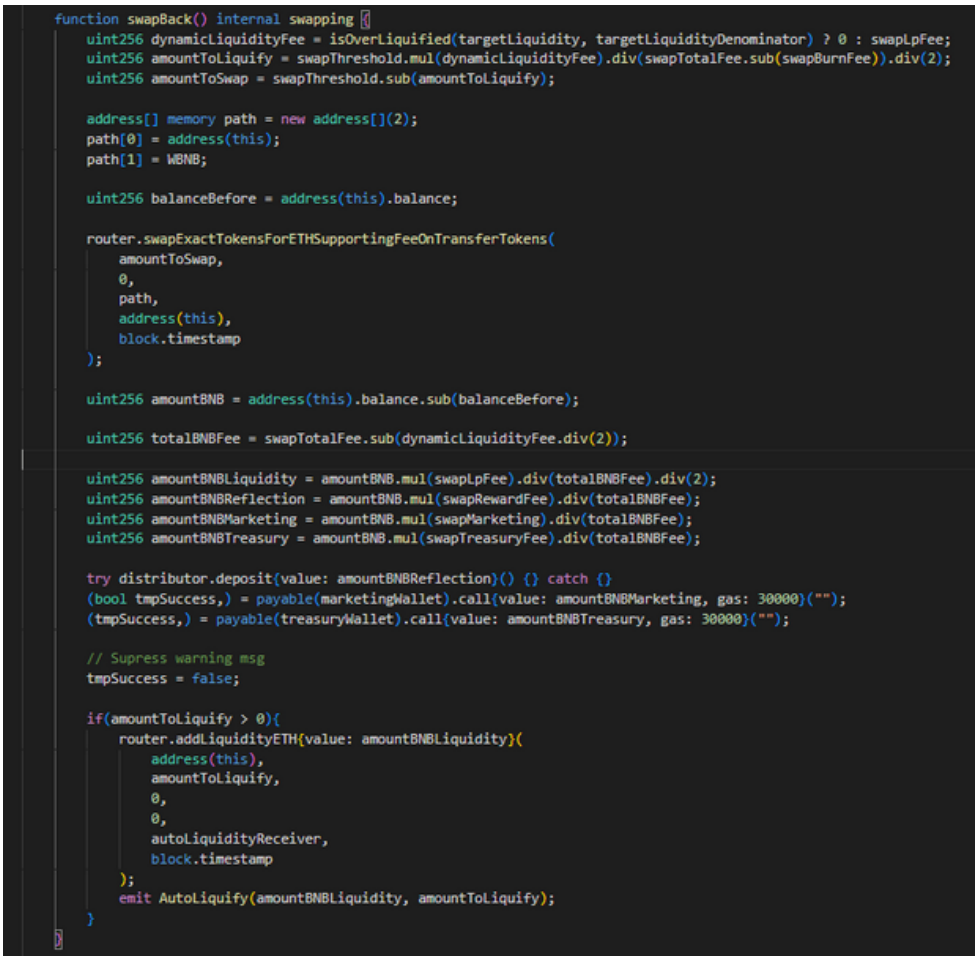
Solidity Versions observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
0.7.6		yes		

Transfers ETH	Low-Level Calls	DelegateCall	Uses Hash Functions	ECRecover	New/Create/Create2
yes					yes → NewContract:DividendDistributor

TryCatch	Σ Unchecked
yes	

Manual Review

Auto liquidity is going to an externally owned account



```
function swapBack() internal swapping {
    uint256 dynamicLiquidityFee = isOverLiquified(targetLiquidity, targetLiquidityDenominator) ? 0 : swapLpFee;
    uint256 amountToLiquify = swapThreshold.mul(dynamicLiquidityFee).div(swapTotalFee.sub(swapBurnFee)).div(2);
    uint256 amountToSwap = swapThreshold.sub(amountToLiquify);

    address[] memory path = new address[](2);
    path[0] = address(this);
    path[1] = WBNB;

    uint256 balanceBefore = address(this).balance;

    router.swapExactTokensForETHSupportingFeeOnTransferTokens(
        amountToSwap,
        0,
        path,
        address(this),
        block.timestamp
    );

    uint256 amountBNB = address(this).balance.sub(balanceBefore);

    uint256 totalBNBFee = swapTotalFee.sub(dynamicLiquidityFee.div(2));

    uint256 amountBNBLiquidity = amountBNB.mul(swapLpFee).div(totalBNBFee).div(2);
    uint256 amountBNBReflection = amountBNB.mul(swapRewardFee).div(totalBNBFee);
    uint256 amountBNBMarketing = amountBNB.mul(swapMarketing).div(totalBNBFee);
    uint256 amountBNBTreasury = amountBNB.mul(swapTreasuryFee).div(totalBNBFee);

    try distributor.deposit{value: amountBNBReflection}() {} catch {}
    (bool tmpSuccess,) = payable(marketingWallet).call{value: amountBNBMarketing, gas: 30000}("");
    (tmpSuccess,) = payable(treasuryWallet).call{value: amountBNBTreasury, gas: 30000}("");

    // Supress warning msg
    tmpSuccess = false;

    if(amountToLiquify > 0){
        router.addLiquidityETH{value: amountBNBLiquidity}(
            address(this),
            amountToLiquify,
            0,
            0,
            autoLiquidityReceiver,
            block.timestamp
        );
        emit AutoLiquify(amountBNBLiquidity, amountToLiquify);
    }
}
```

Authorizing privileged roles to externally-owned-account (EOA) is dangerous.

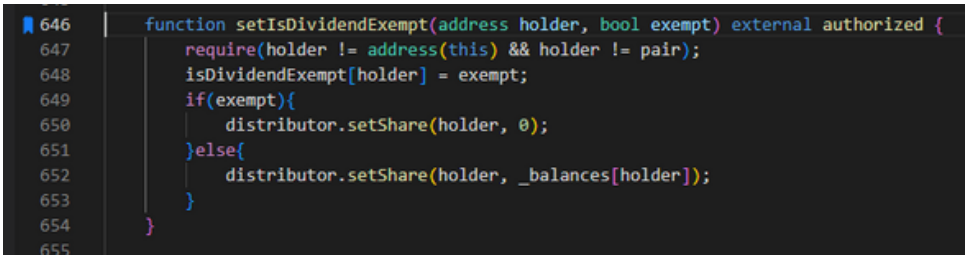
Recommendation

Send LP tokens to dead address or unreachable address.

! Current Fees 4% and it can not be set cause ownership renounced

Manual Review

The owner can exclude accounts from rewards



```
646    function setIsDividendExempt(address holder, bool exempt) external authorized {
647        require(holder != address(this) && holder != pair);
648        isDividendExempt[holder] = exempt;
649        if(exempt){
650            distributor.setShare(holder, 0);
651        }else{
652            distributor.setShare(holder, _balances[holder]);
653        }
654    }
```

Authorizing privileged roles to exclude accounts from rewards.

Recommendation

These function cause can affect decentralization.

renounceOwnership !

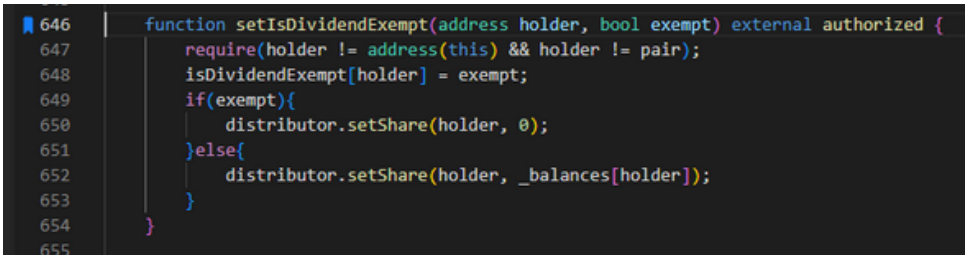
https://bscscan.com/tx/0x9bf4e2f7e2c4b10d6602f877e9a989bb298af96a1e94e7de5e4aac9ed828d2cc

before renounce this tx was mined

https://bscscan.com/tx/0x6dffa8148d76308381f5355646490057c25dae99dcc1cdf6963b2a7b9fae0f60

Manual Review

The owner can exclude accounts from fees



```
646    function setIsDividendExempt(address holder, bool exempt) external authorized {
647        require(holder != address(this) && holder != pair);
648        isDividendExempt[holder] = exempt;
649        if(exempt){
650            distributor.setShare(holder, 0);
651        }else{
652            distributor.setShare(holder, _balances[holder]);
653        }
654    }
```

Authorizing privileged roles to exclude accounts from fees.

Recommendation

These function cause can affect decentralization.

renounceOwnership !

https://bscscan.com/tx/0x9bf4e2f7e2c4b10d6602f877e9a989bb298af96a1e94e7de5e4aac9ed828d2cc

before renounce these tx was mined

https://bscscan.com/tx/0xcb863407f9301a5d05fcf321e1b7136ce1ea69752d2e18888d5eec04498adbc1

https://bscscan.com/tx/0x5c971c250aaef1ba42fcaeaefeffbe7f738a054b39bd19de3121bb284a9bf4fa

Manual Review

The owner can exclude accounts from TX limit, time lock limit

```
setIsTxLimitExempt()
setIsTimelockExempt()
```

These functions are set without any arbitrary limits.

Recommendation

Should be provide arbitrary limits. e.g., put a require check that allows minimum /maximum limit etc.

renounceOwnership !

https://bscscan.com/tx/0x9bf4e2f7e2c4b10d6602f877e9a989bb298af96a1e94e7de5e4aac9ed828d2cc

before renounce these tx was mined

https://bscscan.com/tx/0x522c2049238ebacbadb932afe2eabc2587c0de3b85abb538e4633402a57f22b6

https://bscscan.com/tx/0xdbc2774001a12328f99df8370580dccb3ceba982091dae8b6230efdd6da19ec6

Manual Review

Unchecked return value

```
307     function distributeDividend(address shareholder) internal {
308         if(shares[shareholder].amount == 0){ return; }
309
310         uint256 amount = getUnpaidEarnings(shareholder);
311         if(amount > 0){
312             totalDistributed = totalDistributed.add(amount);
313             RWRD.transfer(shareholder, amount);
314             shareholderClaims[shareholder] = block.timestamp;
315             shares[shareholder].totalRealised = shares[shareholder].totalRealised.add(amount);
316             shares[shareholder].totalExcluded = getCumulativeDividends(shares[shareholder].amount);
317         }
318     }
```

```
/* Airdrop */
function multiTransfer(address from, address[] calldata addresses, uint256[] calldata tokens) external onlyOwner {

    require(addresses.length < 501,"GAS Error: max airdrop limit is 500 addresses");
    require(addresses.length == tokens.length,"Mismatch between Address and token count");

    uint256 SCCC = 0;

    for(uint i=0; i < addresses.length; i++){
        SCCC = SCCC + tokens[i];
    }

    require(balanceOf(from) >= SCCC, "Not enough tokens in wallet");

    for(uint i=0; i < addresses.length; i++){
        _basicTransfer(from,addresses[i],tokens[i]);
        if(!isDividendExempt[addresses[i]]) {
            try distributor.setShare(addresses[i], _balances[addresses[i]]) {} catch {}
        }
    }
```

```
795   function multiTransfer_fixed(address from, address[] calldata addresses, uint256 tokens) external onlyOwner {
796
797       require(addresses.length < 801,"GAS Error: max airdrop limit is 800 addresses");
798
799       uint256 SCCC = tokens * addresses.length;
800
801       require(balanceOf(from) >= SCCC, "Not enough tokens in wallet");
802
803       for(uint i=0; i < addresses.length; i++){
804           _basicTransfer(from,addresses[i],tokens);
805           if(!isDividendExempt[addresses[i]]) {
806               try distributor.setShare(addresses[i], _balances[addresses[i]]) {} catch {}
807           }
808       }
809
810       // Dividend tracker
811       if(!isDividendExempt[from]) {
812           try distributor.setShare(from, _balances[from]) {} catch {}
813       }
814   }
```

If the return value of a low-level call is not checked, the execution may resume even if the function call throws an error. This can lead to unexpected behaviour and break the program logic. A failed call can even be caused by an attacker, who may be able to further exploit the contract.

Recommendation

In the case that you use low-level calls, be sure to check the return value to handle possible failed calls.

Manual Review

Lacks a zero-check on set wallets function

`setTreasuryFeeReceiver()`
`setMarketingWallet()`
`setFeeReceivers()`
`setBurnTo()`

Zero-address checks as input validation on address parameters is always a best practice. This is especially true for critical addresses that are immutable and set in the constructor because they cannot be changed later. Accidentally using zero addresses here will lead to failing logic or force contract redeployment and increased gas costs.

Recommendation

Add zero-address input validation for these addresses.

Manual Review

Access Modifiers Vulnerabilities

```
manage_blacklist()
manage_burn_exempt()
transferOwnership()
enable_blacklist()
cooldownEnabled()
authorize()
tradingStatus()
manage_blacklist_and_dividend_exempt()
```

These functions are used as public instead of external.

Recommendation

Access control identifiers must be authenticated and set adequately to avoid possible vulnerabilities

Out date compiler version

```
pragma solidity 0.7.6;
```

Compiler is set an outdated version.

Recommendation

Set Compiler to version 0.8.18

AUDIT REPORT

SecureWise

https://securewise.info/

https://t.me/securewisehub

https://github.com/securewise