SecureWise

AUDIT REPORT

SecureWise

SMART CONTRACT AUDIT

https://github.com/securewise
https://t.me/securewise
https://securewise.info/

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: House of the Dragon (**GoT**)

Methodology: Automated Analysis, Manual Code Review

Language: Solidity

Contract Address: 0x3ea88a727215D936F55EF9778ac3a1862B635D7e

ContractLink: https://bscscan.com/address/0x3ea88a727215D936F55EF9778ac3a1862B635D7e

Network: BSC

Decimals: 9

Supply: 1,000,000,000

Website: https://www.got.army/

Twitter: https://twitter.com/got_dot_army

Telegram: https://t.me/got_dot_army

Report Date: August 10, 2022

Quick Result

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

Owner Privileges

 Auto liquidity is going to an externally owned account

 The owner can set a blacklist

 The owner can change max wallet token amount to "0"

 The owner can't set max transaction amount "0" but can set very low amount

 The owner can exclude accounts from rewards

 The owner can exclude accounts from fees

 The owner can set buy fees 10% and sell fees 25%

 The owner can change swap settings

House of the Dragon (GoT) 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

Context	Implementation			
L	_msgSender	Internal 🔒		
L	_msgData	Internal 🔒		
IERC20	Interface			
L	totalSupply	External ❗		NO❗
L	balanceOf	External ❗		NO❗
L	transfer	External ❗	🔴	NO❗
L	allowance	External ❗		NO❗
L	approve	External ❗	🔴	NO❗
L	transferFrom	External ❗	🔴	NO❗
IERC20Metadata	Interface	IERC20		
L	name	External ❗		NO❗
L	symbol	External ❗		NO❗
L	decimals	External ❗		NO❗
ERC20	Implementation	Context, IERC20, IERC20Metadata		
L		Public ❗	🔴	NO❗
L	name	Public ❗		NO❗
L	symbol	Public ❗		NO❗
L	decimals	Public ❗		NO❗
L	totalSupply	Public ❗		NO❗
L	balanceOf	Public ❗		NO❗
L	transfer	Public ❗	🔴	NO❗
L	allowance	Public ❗		NO❗
L	approve	Public ❗	🔴	NO❗
L	transferFrom	Public ❗	🔴	NO❗
L	increaseAllowance	Public ❗	🔴	NO❗
L	decreaseAllowance	Public ❗	🔴	NO❗
L	_transfer	Internal 🔒	🔴	
L	_mint	Internal 🔒	🔴	

Automated Analysis

└	_burn	Internal 🔒	🔴	
└	_approve	Internal 🔒	🔴	
└	_beforeTokenTransfer	Internal 🔒	🔴	
└	_afterTokenTransfer	Internal 🔒	🔴	
└	_changeName	Internal 🔒	🔴	
└	_changeSymbol	Internal 🔒	🔴	
Ownable	Implementation	Context		
└		Public ❗	🔴	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🔴	onlyOwner
└	transferOwnership	Public ❗	🔴	onlyOwner
└	_setOwner	Private 🔐	🔴	
IterableMapping	Library			
└	get	Public ❗		NO❗
└	getIndexOfKey	Public ❗		NO❗
└	getKeyAtIndex	Public ❗		NO❗
└	size	Public ❗		NO❗
└	set	Public ❗	🔴	NO❗
└	remove	Public ❗	🔴	NO❗
DividendPayingTokenOptionalInterface	Interface			
└	withdrawableDividendOf	External ❗		NO❗
└	withdrawnDividendOf	External ❗		NO❗
└	accumulativeDividendOf	External ❗		NO❗
DividendPayingTokenInterface	Interface			
└	dividendOf	External ❗		NO❗
└	distributeDividends	External ❗	💸	NO❗
└	withdrawDividend	External ❗	🔴	NO❗
DividendPayingToken	Implementation	ERC20, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
└		Public ❗	🔴	ERC20
└		External ❗	💸	NO❗
└	distributeDividends	Public ❗	💸	NO❗

Automated Analysis

└	withdrawDividend	Public		NO
└	_withdrawDividendOfUser	Internal		
└	dividendOf	Public		NO
└	withdrawableDividendOf	Public		NO
└	withdrawnDividendOf	Public		NO
└	accumulativeDividendOf	Public		NO
└	_transfer	Internal		
└	_mint	Internal		
└	_burn	Internal		
└	_setBalance	Internal		
IUniswapV2Router01	Interface			
└	factory	External		NO
└	WETH	External		NO
└	addLiquidity	External		NO
└	addLiquidityETH	External		NO
└	removeLiquidity	External		NO
└	removeLiquidityETH	External		NO
└	removeLiquidityWithPermit	External		NO
└	removeLiquidityETHWithPermit	External		NO
└	swapExactTokensForTokens	External		NO
└	swapTokensForExactTokens	External		NO
└	swapExactETHForTokens	External		NO
└	swapTokensForExactETH	External		NO
└	swapExactTokensForETH	External		NO
└	swapETHForExactTokens	External		NO
└	quote	External		NO
└	getAmountOut	External		NO
└	getAmountIn	External		NO
└	getAmountsOut	External		NO
└	getAmountsIn	External		NO
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External		NO
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External		NO

Automated Analysis

L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External		NO
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External		NO
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External		NO
IUniswapV2Factory	Interface			
L	feeTo	External		NO
L	feeToSetter	External		NO
L	getPair	External		NO
L	allPairs	External		NO
L	allPairsLength	External		NO
L	createPair	External		NO
L	setFeeTo	External		NO
L	setFeeToSetter	External		NO
SignedSafeMath	Library			
L	mul	Internal		
L	div	Internal		
L	sub	Internal		
L	add	Internal		
SafeMath	Library			
L	tryAdd	Internal		
L	trySub	Internal		
L	tryMul	Internal		
L	tryDiv	Internal		
L	tryMod	Internal		
L	add	Internal		
L	sub	Internal		
L	mul	Internal		
L	div	Internal		
L	mod	Internal		
L	sub	Internal		
L	div	Internal		
L	mod	Internal		

Automated Analysis

SafeCast	Library			
└	toUint224	Internal		
└	toUint128	Internal		
└	toUint96	Internal		
└	toUint64	Internal		
└	toUint32	Internal		
└	toUint16	Internal		
└	toUint8	Internal		
└	toUint256	Internal		
└	toInt128	Internal		
└	toInt64	Internal		
└	toInt32	Internal		
└	toInt16	Internal		
└	toInt8	Internal		
└	toInt256	Internal		
GoTDividendTracker	Implementation	DividendPayingToken, Ownable		
└		Public		DividendPayingToken
└	_transfer	Internal		
└	withdrawDividend	Public		NO
└	excludeFromDividends	External		onlyOwner
└	updateClaimWait	External		onlyOwner
└	getLastProcessedIndex	External		NO
└	getNumberOfTokenHolders	External		NO
└	getAccount	Public		NO
└	getAccountAtIndex	Public		NO
└	canAutoClaim	Private		
└	setBalance	External		onlyOwner
└	process	Public		NO
└	processAccount	Public		onlyOwner
GoT	Implementation	ERC20, Ownable		
└	setBuyFees	Public		onlyOwner
└	setSellFees	Public		onlyOwner
└	setMaxSellTx	Public		onlyOwner

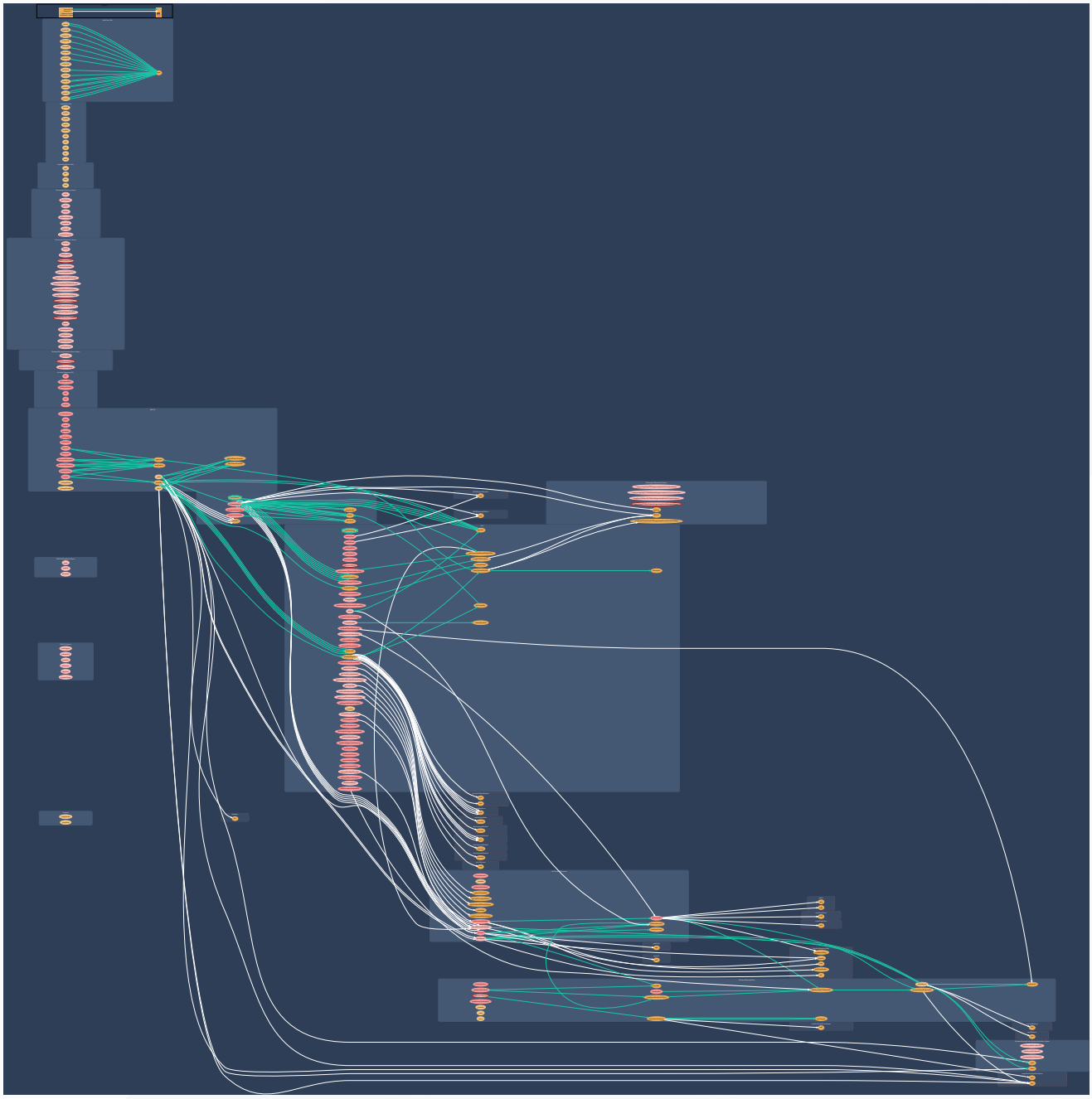
Automated Analysis

L	setMaxBuyTx	Public		onlyOwner
L		Public		ERC20
L		External		NO
L	changeName	External		onlyOwner
L	changeSymbol	External		onlyOwner
L	updateUniswapV2Router	Public		onlyOwner
L	excludeFromFees	Public		onlyOwner
L	setExcludeFromMaxTx	Public		onlyOwner
L	setExcludeFromAll	Public		onlyOwner
L	excludeMultipleAccountsFromFees	Public		onlyOwner
L	setAutomatedMarketMakerPair	Public		onlyOwner
L	_setAutomatedMarketMakerPair	Private		
L	updateGasForProcessing	Public		onlyOwner
L	updateClaimWait	External		onlyOwner
L	getClaimWait	External		NO
L	getTotalDividendsDistributed	External		NO
L	isExcludedFromFees	Public		NO
L	isExcludedFromMaxTx	Public		NO
L	withdrawableDividendOf	Public		NO
L	dividendTokenBalanceOf	Public		NO
L	getAccountDividendsInfo	External		NO
L	getAccountDividendsInfoAtIndex	External		NO
L	processDividendTracker	External		NO
L	claim	External		NO
L	getLastProcessedIndex	External		NO
L	getNumberOfDividendTokenHolders	External		NO
L	excludeFromDividends	External		onlyOwner
L	setSwapAndLiquifyEnabled	Public		onlyOwner
L	_transfer	Internal		
L	swapAndLiquify	Private		lockTheSwap
L	swapTokensForBnb	Private		
L	addLiquidity	Private		
L	swapETHForTokens	Private		
L	buyBackTokens	Private		lockTheSwap

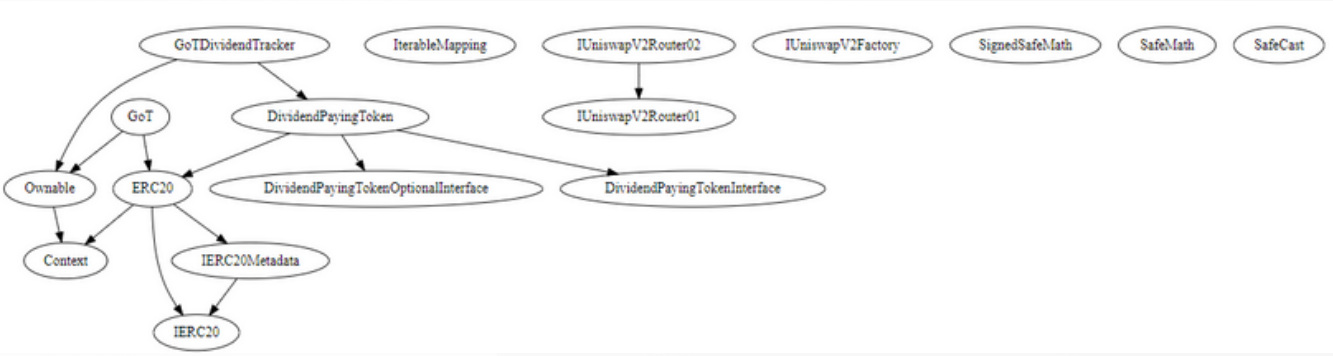
Automated Analysis

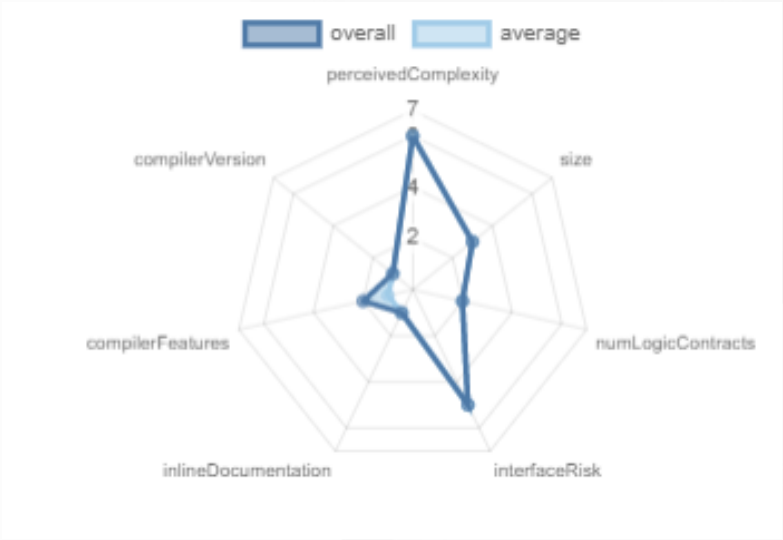
L	setBuyBackEnabled	Public ❗	🔴	onlyOwner
L	SetBuyBackUpperLimitAmount	Public ❗	🔴	NO❗
L	setMaxWalletTokend	External ❗	🔴	onlyOwner
L	buyBackUpperLimitAmount	Public ❗		NO❗
L	buyBackDivisor	Public ❗		NO❗
L	setBuyBackDivisor	Public ❗	🔴	onlyOwner
L	setmarketingWallet	Public ❗	🔴	onlyOwner
L	setdevelopmentWallet	Public ❗	🔴	onlyOwner
L	setDevShareInPercentage	Public ❗	🔴	onlyOwner
L	setSwapTokensAtAmount	Public ❗	🔴	onlyOwner
L	blacklistAddress	External ❗	🔴	onlyOwner

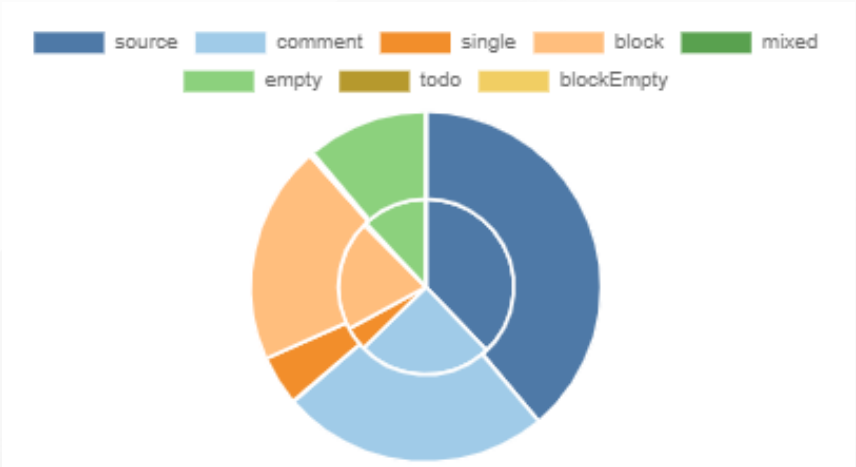
Inheritance Graph



Inheritance Graph







Contract Summary

Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
10	7	2325	1975	1007	774	852	
10	**7**	**2325**	**1975**	**1007**	**774**	**852**	

Components

Contracts	Libraries	Interfaces	Abstract
4	4	7	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
124	8

External	Internal	Private	Pure	View
68	136	8	38	51

StateVariables

Total	Public
54	20

Capabilities

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

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

TryCatch	Σ Unchecked
`yes`	`yes`

Manual Review

Auto liquidity is going to an externally owned account

```
2242    function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private
2243    {
2244        // add the liquidity
2245        uniswapV2Router.addLiquidityETH{value: ethAmount}(
2246            address(this),
2247            tokenAmount,
2248            0, // slippage is unavoidable
2249            0, // slippage is unavoidable
2250            owner(),
2251            block.timestamp
2252        );
2253    }
```

Recommendation

Authorizing privileged roles to externally-owned-account (EOA) is dangerous.
Send LP tokens to dead address or unreachable address.

The owner can set blacklist

```
2324    function blacklistAddress(address account, bool value) external onlyOwner{
2325        _isBlacklisted[account] = value;
2326    }
```

Recommendation

Authorizing privileged roles to add an account to blacklist and pause trade for any account. These cause can affect decentralization. Remove blacklist function

Manual Review

The owner can change max wallet token amount to "0"

```
2287    function setMaxWalletTokend(uint256 _maxToken) external onlyOwner {
2288        maxWalletToken = _maxToken * (10**9);
2289    }
```

Recommendation

Authorizing privileged roles to change max wallet amount without limit and these cause by semi pause trading. shouldn't be 0. Put a **require** check that allows minimum reasonable limit etc.

The owner can't set max transaction amount "0" but can set very low amount

```
1854    function setMaxSellTx(uint256 _maxSellTxAmount) public onlyOwner {
1855        require(_maxSellTxAmount > 0, "maxSellTransactionAmount must be greater than zero");
1856        maxSellTransactionAmount = _maxSellTxAmount * (10**9);
1857
1858    }
1859
1860    function setMaxBuyTx(uint256 _maxBuyTxAmount) public onlyOwner {
1861        require(_maxBuyTxAmount > 0, "maxBuyTransactionAmount must be greater than zero");
1862        maxBuyTransactionAmount = _maxBuyTxAmount * (10**9);
1863    }
```

Recommendation

Authorizing privileged roles to set max transaction very low amount. Put a **require** check that allows minimum limit etc.

The owner can exclude accounts from rewards

```
1554    function excludeFromDividends(address account) external onlyOwner {
1555        require(!excludedFromDividends[account]);
1556        excludedFromDividends[account] = true;
1557
1558        _setBalance(account, 0);
1559        tokenHoldersMap.remove(account);
1560
1561        emit ExcludeFromDividends(account);
1562    }
```

Recommendation

Authorizing privileged roles to exclude accounts from rewards. These cause can affect decentralization.

Manual Review

The owner can exclude accounts from fees

```
1947    function excludeFromFees(address account, bool excluded) public onlyOwner {
1948        require(_isExcludedFromFees[account] != excluded, "GoT: Account is already the value of 'excluded'");
1949        _isExcludedFromFees[account] = excluded;
1950
1951        emit ExcludeFromFees(account, excluded);
1952    }
```

Recommendation

Authorizing privileged roles to exclude accounts from fees. These cause can affect decentralization.

The owner can set buy fees 10% and sell fees 25%

```
1832    function setBuyFees(uint256 _bnbRewardFee, uint256 _liquidityFee, uint256 _marketingFee, uint256 _developmentFee, uint256 _buybackFee)
1833        public onlyOwner {
1834        BNBRewardsBuyFee = _bnbRewardFee;
1835        liquidityBuyFee = _liquidityFee;
1836        marketingBuyFee = _marketingFee;
1837        developmentBuyFee = _developmentFee;
1838        buybackBuyFee = _buybackFee;
1839        totalBuyFees = BNBRewardsBuyFee.add(liquidityBuyFee).add(marketingBuyFee).add(developmentBuyFee).add(buybackBuyFee); // total buy fee
1840        require(totalBuyFees <= 10, "tax too high");
1841    }
1842
1843    function setSellFees(uint256 _bnbRewardFee, uint256 _liquidityFee, uint256 _marketingFee, uint256 _developmentFee, uint256 _buybackFee)
1844        public onlyOwner {
1845        BNBRewardsSellFee = _bnbRewardFee;
1846        liquiditySellFee = _liquidityFee;
1847        marketingSellFee = _marketingFee;
1848        developmentSellFee = _developmentFee;
1849        buybackSellFee = _buybackFee;
1850        totalSellFees = BNBRewardsSellFee.add(liquiditySellFee).add(marketingSellFee).add(developmentSellFee).add(buybackSellFee); // total sell fee
1851        require(totalSellFees <= 25, "tax too high");
1852    }
```

Recommendation

-

The owner can change swap settings

```
2071    function setSwapAndLiquifyEnabled(bool _enabled) public onlyOwner {
2072        swapAndLiquifyEnabled = _enabled;
2073        emit SwapAndLiquifyEnabledUpdated(_enabled);
2074    }
```

Recommendation

Authorizing privileged roles to enable or disable the swap. These cause can affect decentralization.

AUDIT REPORT

SecureWise

SMART CONTRACT AUDIT

https://github.com/securewise
https://t.me/securewise
https://securewise.info/

SecureWise