

SKELETON ECOSYSTEM

SMART CONTRACT AUDIT



BNBets
\$BETS
BEP20

0xfD55e48AC9B28A837627115A1F114A4b48775882



Table of Contents

Table of Contents	1
Disclaimer	2
Overview	3
Creation/Audit Date	3
Verified Socials	3
Contract Functions Analysis	4
Contract Safety and Weakness	8
Detected Vulnerability Description	12
Contract Flow Graph	13
Contract Interaction Graph	14
Inheritance Graph	15
Contract Descriptions	16
Audit Scope	21

Global Disclaimer

This document serves as a disclaimer for the crypto smart contract audit conducted by Skeleton Ecosystem. The purpose of the audit was to review the codebase of the smart contracts for potential vulnerabilities and issues. It is important to note the following:

Limited Scope: The audit is based on the code and information available up to the audit completion date. It does not cover external factors, system interactions, or changes made after the audit. The audit itself can not guarantee 100% safety and can not detect common scam methods like farming and developer sell-out.

No Guarantee of Security: While we have taken reasonable steps to identify vulnerabilities, it is impossible to guarantee the complete absence of security risks or issues. The audit report provides an assessment of the contract's security as of the audit date.

Continued Development: Smart contracts and blockchain technology are evolving fields. Updates, forks, or changes to the contract post-audit may introduce new risks that were not present during the audit.

Third-party Code: If the smart contract relies on third-party libraries or code, those components were not thoroughly audited unless explicitly stated. Security of these dependencies is the responsibility of their respective developers.

Non-Exhaustive Testing: The audit involved automated analysis, manual review, and testing under controlled conditions. It is possible that certain vulnerabilities or issues may not have been identified.

Risk Evaluation: The audit report includes a risk assessment for identified vulnerabilities. It is recommended that the development team carefully reviews and addresses these risks to mitigate potential exploits.

Not Financial Advice: This audit report is not intended as financial or investment advice. Decisions regarding the use, deployment, or investment in the smart contract should be made based on a comprehensive assessment of the associated risks.

By accessing and using this audit report, you acknowledge and agree to the limitations outlined above. Skeleton Ecosystem and its auditors shall not be held liable for any direct or indirect damages resulting from the use of the audit report or the smart contract itself.

Please consult with legal, technical, and financial professionals before making any decisions related to the smart contract.

Overview

Contract Name	BNBETS
Ticker/Symbol	\$BETS
Blockchain	Binance Smart Chain BEP20
Contract Address	0xfD55e48AC9B28A837627115A1F114A4b48775882
Creator Address	0x28E37C3Dc61D0F6d13BAb930ee69e462aeBB505a
Current Owner Address	0x28E37C3Dc61D0F6d13BAb930ee69e462aeBB505a
Contract Explorer	https://bscscan.com/address/0xfd55e48ac9b28a837627115a1f114a4b48775882#code
Compiler Version	v0.7.6+commit.7338295f
License	MIT
Optimisation	Yes with 200 Runs
Total Supply	100,000,000 \$BETS
Decimals	9




Creation/Audit

Contract Deployed	14.01.2023
Audit Created	18.01.2024
Audit Update	V 1.0

Verified Socials

Website	https://bnbets.io/
Telegram	https://t.me/Bnbetsofficial
Twitter (X)	https://x.com/bnbets2024

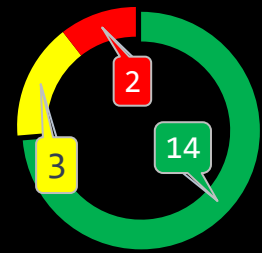
Contract Function Analysis








 Pass
  Attention Item
  Risky Item

■ Pass

■ Attention

■ Risk



Contract Verified		The contract source code is uploaded to blockchain explorer and is open source, so everybody can read it.
Contract Ownership		0x28E37C3Dc61D0F6d13BAb930ee69e462aeBB505a Deployer
Buy Tax	9 %	Shows the taxes for purchase transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable
Sell Tax	9 %	Shows the taxes for sell transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable.
Honeypot Analyse		Holder is able to buy and sell. If honeypot: The contract blocks sell transfer from holder wallet. Multiple events may cause honeypot. Trading disabled, extremely high tax
Liquidity Status		Liquidity Locker status 17.01.2024: 100.00% Pinklock for 365 days.
Trading Disable Functions		Trading suspendable function found If a suspendable code is included, the token maybe neither be bought or sold (honeypot risk). If contract is renounced this function can't be used
Set Fees function		Fee Setting function found. The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens may not be able to be traded (honeypot risk).
Proxy Contract		Not a proxy contract!
Mint Function		No Mint Function detected Mint function is transparent or non-existent. Hidden mint functions may increase the amount of tokens in circulation and effect the price of the token. Owner can mint new tokens and sell.

Balance Modifier Function		<p>No Balance Modifier function found.</p> <p>If there is a function for this, the contract owner can have the authority to modify the balance of tokens at other addresses. For example revoke the bought tokens from the holders wallet. Common form of scam: You buy the token, but it's disappearing from your wallet.</p>
Blacklist Function		<p>No Blacklist Setting function found.</p> <p>If there is a blacklist, some addresses may not be able to trade normally. Example: you buy the token and right after your Wallet getting blacklisted. Like so you will be unable to sell. Honeypot Risk.</p>
Whitelist Function		<p>Whitelist Setting function found.</p> <p>If there is a function for this Developer can set zero fee or no max wallet size for addresses (for example team wallets can trade without fee. Can cause farming)</p>
Owner Analysis		<p>Contract leaves deployer authorised even if ownership is transferred.</p> <p>For contract with a hidden owner, developer can still manipulate the contract even if the ownership has been abandoned.</p>
Retrieve Ownership Function		<p>No functions found which can retrieve ownership of the contract.</p> <p>If this function exists, it is possible for the project owner to regain ownership even after relinquishing it. Also known as fake renounce.</p>
Self Destruct Function		<p>No Self Destruct function found.</p> <p>If this function exists and is triggered, the contract will be destroyed, all functions will be unavailable, and all related assets will be erased.</p>
Specific Tax Changing Function		<p>No Specific Tax Changing Functions found.</p> <p>If it exists, the contract owner may set a very outrageous tax rate for assigned address to block it from trading. Can assign all wallets at once!</p>
Trading Cooldown Function		<p>No Trading Cooldown Function found.</p> <p>If there is a trading cooldown function, the user will not be able to sell the token within a certain time or block after buying. Like a temporary honeypot.</p>
Max Transaction and Holding Modify Function		<p>Max Transaction and Holding Modify function found.</p> <p>If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot</p>
Transaction Limiting Function		<p>No Transaction Limiter Function Found.</p> <p>The number of overall token transactions may be limited (honeypot risk)</p>

Details of Risk - Attention Items

The following functions are listed as functions may be triggered by the owner of the contract.

⚠ Whitelist (Set excluded)

[Remediation: No remediation, based on deployer authorisations]

If there is a function for this Developer can set zero fee or no max wallet size for addresses (for example team wallets can trade without fee)

```
496     ftrace | funcSig
497     function changeIsFeeExempt(address holder!, bool exempt!) external authorized {
498         | isFeeExempt[holder!] = exempt!;
499     }

500     ftrace | funcSig
501     function changeIsTxLimitExempt(address holder!, bool exempt!) external authorized {
502         | isTxLimitExempt[holder!] = exempt!;
503     }
```

⚠ Max Transaction and Holding Modify function

[Remediation: No remediation, based on deployer authorisations]

If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot

```
484     ftrace | funcSig
485     function changeTxLimit(uint256 newLimit!) external authorized {
486         | maxTxAmount = newLimit!;
487     }

488     ftrace | funcSig
489     function changeWalletLimit(uint256 newLimit!) external authorized {
490         | walletMax = newLimit!;
491     }
```

⚠ Set Trading Fees

[Remedation: No remediation, based on deployer authorisations]

The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens may not be able to be traded (honeypot risk).

```

515   ftrace | funcSig
516   function changeFees(uint256 newLiqFee!, uint256 newRewardFee!, uint256 newMarketingFee!, uint256 newExtraSellFee!) external authorized {
517       liquidityFee = newLiqFee!;
518       rewardsFee = newRewardFee!;
519       marketingFee = newMarketingFee!;
520       extraFeeOnSell = newExtraSellFee!;
521
522       totalFee = liquidityFee.add(marketingFee).add(rewardsFee);
523       totalFeeIfSelling = totalFee.add(extraFeeOnSell);
524   }
  
```

⚠ Trading Suspensible Function

[Remedation: No remediation, based on deployer authorisations]

If there is a trading cooldown function, the user will not be able to sell the token within a certain time or block after buying. Like a temporary honeypot.

```

619   ftrace | funcSig
620   function tradingStatus(bool newStatus!) public onlyOwner {
621       tradingOpen = newStatus!;
622   }
623
  
```

⚠ Hidden or multi owner.

[Remedation: No remediation, based on deployer authorisations]

Contract leaves deployer authorised even if ownership is transferred. For contract with a hidden owner, developer can still manipulate the contract even if the ownership has been abandoned.

```

336   ftrace | funcSig
337   function isAuthorized(address adr!) public view returns (bool) {
338       return authorizations[adr!];
339   }
340
341   /**
342    * Transfer ownership to new address. Caller must be owner. Leaves old owner authorized
343    */
344   ftrace | funcSig
345   function transferOwnership(address payable adr!) public onlyOwner {
346       owner = adr!;
347       authorizations[adr!] = true;
348       emit OwnershipTransferred(adr!);
349   }
350
351   event OwnershipTransferred(address owner);
  
```


Contract Security

Total Findings: 7

■ High 0

■ Medium 1

■ Low 4

■ Info 2



■ **High Severity Issues:** High possibility to cause problems, need to be resolved.

■ **Medium Severity Issue:** Will likely cause problems, recommended to resolve.

■ **Low Severity Issues:** Won't cause problems, but for improvement purposes could be adjusted.

■ **Informational Severity Issues:** Not harmful in any way, information for the developer team.

Contract Security

List of Found Issues

High severity Issues: (0)

Medium severity issues: (1)

- Unchecked Array Lenght

Low severity issues: (4)

- Missing Events
- Floating Pragma
- Long Number Literals
- Outdated Compiler Version

Informational severity issues: (2)

- Public Functions Should be Declared External
- State Variables Shouds be Declared Constant

Contract Weakness Classisication

THE SMART CONTRACT WEAKNESS CLASSIFICATION REGISTRY (SWC REGISTRY) IS AN IMPLEMENTATION OF THE WEAKNESS CLASSIFICATION SCHEME PROPOSED IN EIP-1470. IT IS LOOSELY ALIGNED TO THE TERMINOLOGIES AND STRUCTURE USED IN THE COMMON WEAKNESS ENUMERATION (CWE) WHILE OVERLAYING A WIDE RANGE OF WEAKNESS VARIANTS THAT ARE SPECIFIC TO SMART CONTRACTS.

ID	Description	AI	Manual	Result
SWC-100	Function Default Visibility	Passed	Passed	Passed
SWC-101	Integer Overflow and Underflow	Passed	Passed	Passed
SWC-102	Outdated Compiler Version	Low	Passed	Passed
SWC-103	Floating Pragma	Low	Passed	Passed
SWC-104	Unchecked Call Return Value	Passed	Passed	Passed
SWC-105	Unprotected Ether Withdrawal	Passed	Passed	Passed
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed	Passed	Passed
SWC-107	Reentrancy	Passed	Passed	Passed
SWC-108	State Variable Default Visibility	Passed	Passed	Passed
SWC-109	Uninitialized Storage Pointer	Passed	Passed	Passed
SWC-110	Assert Violation	Passed	Passed	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed	Passed	Passed
SWC-112	Delegatecall to Untrusted Callee	Passed	Passed	Passed
SWC-113	DoS with Failed Call	Passed	Passed	Passed
SWC-114	Transaction Order Dependence	Passed	Passed	Passed
SWC-115	Authorization through tx.origin	Passed	Passed	Passed
SWC-116	Block values as a proxy for time	Passed	Passed	Passed
SWC-117	Signature Malleability	Passed	Passed	Passed
SWC-118	Incorrect Constructor Name	Passed	Passed	Passed
SWC-119	Shadowing State Variables	Passed	Passed	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed	Passed	Passed

SWC-121	Missing Protection against Signature Replay Attacks	Passed	Passed	Passed
SWC-122	Lack of Proper Signature Verification	Passed	Passed	Passed
SWC-123	Requirement Violation	Passed	Passed	Passed
SWC-124	Write to Arbitrary Storage Location	Passed	Passed	Passed
SWC-125	Incorrect Inheritance Order	Passed	Passed	Passed
SWC-126	Insufficient Gas Griefing	Passed	Passed	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed	Passed	Passed
SWC-128	DoS With Block Gas Limit	Passed	Passed	Passed
SWC-129	Typographical Error	low	Passed	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed	Passed	Passed
SWC-131	Presence of unused variables	Passed	Passed	Passed
SWC-132	Unexpected Ether balance	Passed	Passed	Passed
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed	Passed	Passed
SWC-134	Message call with hardcoded gas amount	Passed	Passed	Passed
SWC-135	Code With No Effects	Passed	Passed	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed	Passed	Passed

Detected High and Medium Severity Vulnerability Description.

⚠️ Unchecked Array Length (1 Items)

Item: 1	Location:	Line 224	Severity:	Medium
---------	-----------	----------	-----------	--------

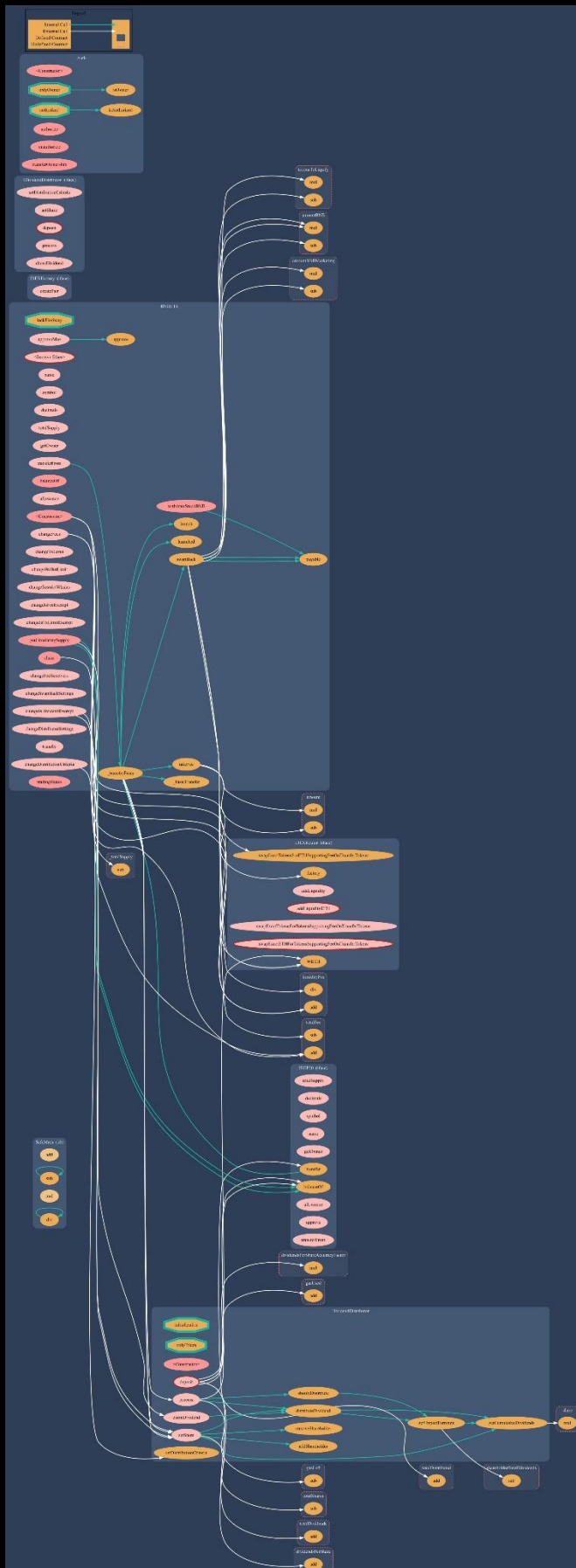
Function	<p>Ethereum is a very resource-constrained environment. Prices per computational step are orders of magnitude higher than with centralized providers. Moreover, Ethereum miners impose a limit on the total number of Gas consumed in a block. If array.length is large enough, the function exceeds the block gas limit, and transactions calling it will never be confirmed.</p> <pre>for (uint256 i = 0; i < array.length ; i++) { cosltyFunc(); }</pre> <p>This becomes a security issue if an external actor influences array.length. E.g., if an array enumerates all registered addresses, an adversary can register many addresses, causing the problem described above.</p>
Remediation	<p>Either explicitly or just due to normal operation, the number of iterations in a loop can grow beyond the block gas limit, which can cause the complete contract to be stalled at a certain point. Therefore, loops with a bigger or unknown number of steps should always be avoided.</p>

```

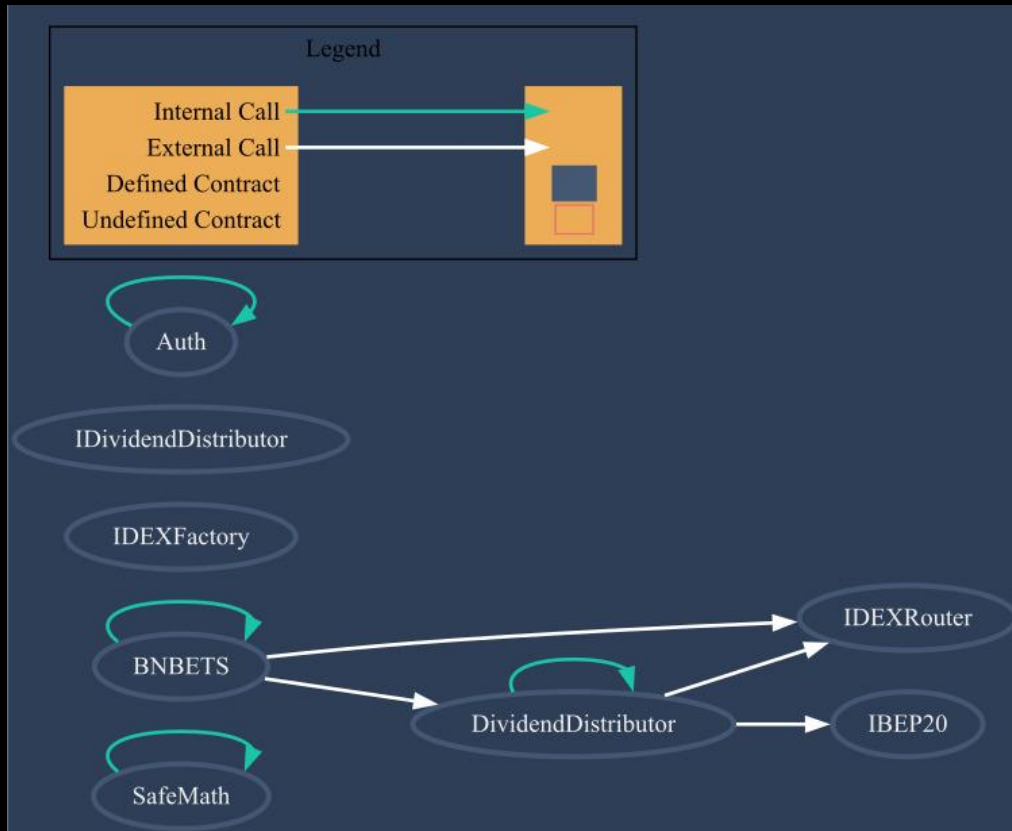
223
224     while(gasUsed < gas! && iterations < shareholderCount) {
225

```

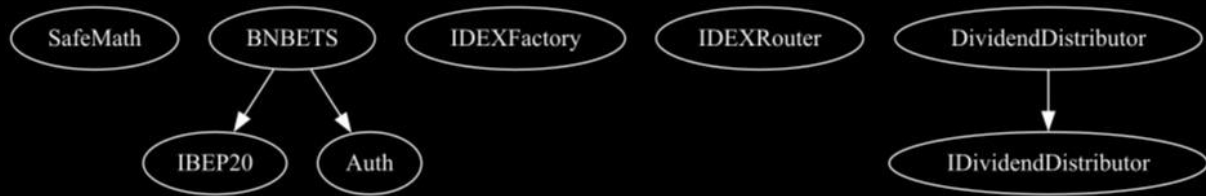
Contract Flow Graph
































Contract Interaction Graph













































































Inheritance Graph













































Contract Functions

Contract	Type	Bases		
L	Function Name	Visibility	Mutability	Modifiers
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 
IDEXFactory	Interface			

Contract	Type	Bases		
L	createPair	External 		NO 
IDEXRouter	Interface			
L	factory	External 		NO 
L	WETH	External 		NO 
L	addLiquidity	External 		NO 
L	addLiquidityETH	External 		NO 
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External 		NO 
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External 		NO 
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External 		NO 
IDividendDistributor	Interface			
L	setDistributionCriteria	External 		NO 
L	setShare	External 		NO 
L	deposit	External 		NO 
L	process	External 		NO 
L	claimDividend	External 		NO 
DividendDistributor	Implementation	IDividendDistributor		

Contract	Type	Bases		
L		Public 		NO 
L	setDistributionCriteria	External 		onlyToken
L	setShare	External 		onlyToken
L	deposit	External 		onlyToken
L	process	External 		onlyToken
L	shouldDistribute	Internal 		
L	distributeDividend	Internal 		
L	getUnpaidEarnings	Public 		NO 
L	getCumulativeDividends	Internal 		
L	addShareholder	Internal 		
L	removeShareholder	Internal 		
L	claimDividend	External 		NO 
Auth	Implementation			
L		Public 		NO 
L	authorize	Public 		onlyOwner
L	unauthorize	Public 		onlyOwner
L	isOwner	Public 		NO 
L	isAuthorized	Public 		NO 
L	transferOwnership	Public 		onlyOwner

Contract	Type	Bases		
BNBETS	Implementation	IBEP20, Auth		
L		Public 		Auth
L		External 		NO 
L	name	External 		NO 
L	symbol	External 		NO 
L	decimals	External 		NO 
L	totalSupply	External 		NO 
L	getOwner	External 		NO 
L	getCirculatingSupply	Public 		NO 
L	balanceOf	Public 		NO 
L	allowance	External 		NO 
L	approve	Public 		NO 
L	approveMax	External 		NO 
L	claim	Public 		NO 
L	launched	Internal 		
L	launch	Internal 		
L	changeTxLimit	External 		authorized
L	changeWalletLimit	External 		authorized
L	changeRestrictWhales	External 		authorized
L	changeFeeExempt	External 		authorized

Contract	Type	Bases		
L	changeTxLimit Exempt	External 		authorized
L	changeDivide ndExempt	External 		authorized
L	changeFees	External 		authorized
L	changeFeeRecei vers	External 		authorized
L	changeSwapBac kSettings	External 		authorized
L	changeDistribut ionCriteria	External 		authorized
L	changeDistribut orSettings	External 		authorized
L	transfer	External 		NO 
L	transferFrom	External 		NO 
L	_transferFrom	Internal 		
L	_basicTransfer	Internal 		
L	takeFee	Internal 		
L	tradingStatus	Public 		onlyOwner
L	swapBack	Internal 		lockTheSwap
L	withdrawStuckB NB	Public 		onlyOwner



Function
can modify
state



Function
is payable

Audit Scope

Audit Method.

Our smart contract audit is an extensive methodical examination and analysis of the smart contract's code that is used to interact with the blockchain. Goal: discover errors, issues and security vulnerabilities in the code. Findings getting reported and improvements getting suggested.

Automatic and Manual Review

We are using automated tools to scan functions and weaknesses of the contract. Transfers, integer over-undeflow checks such as all CWE events.

Tools we use:

Visual Studio Code

CWE

SWC

Solidity Scan

SVD

In manual code review our auditor looking at source code and performing line by line examination. This method helps to clarify developer's coding decisions and business logic.

Skeleton Ecosystem

<https://skeletonecosystem.com>

<https://github.com/SkeletonEcosystem/Audits>

