# SKELETON ECOSYSTEM

SMART CONTRACT AUDIT

## BE THE CHAD [BTCCHAD]

**BEP 20**

0xdEe1280F26B4eBE5E626DA1E3C292896Ce4Cc87f

# Table of Contents

# 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% safaty 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 | Be The Chad |
| Ticker/Simbol | BTCchad |
| Blockchain | Binance Smart Chain BEP20 |
| Contract Address | 0xdEe1280F26B4eBE5E626DA1E3C292896Ce4Cc87f |
| Creator Address | 0xcFC46cB90FE1DF1631e536d5b8C0AC49fecfe617 |
| Current Owner Address | 0xcFC46cB90FE1DF1631e536d5b8C0AC49fecfe617 |
| Contract Explorer | https://bscscan.com/token/0x3ED1be864a7D08a3e3e72B28c567DEd1E5eE70c7#code |
| Compiler Version | v0.8.15+commit.e14f2714 |
| License | MIT |
| Optimisation | Yes with 200 Runs |
| Total Supply | 1,000,000 **BTCchad** |
| Decimals | 18 |

## Creation/Audit

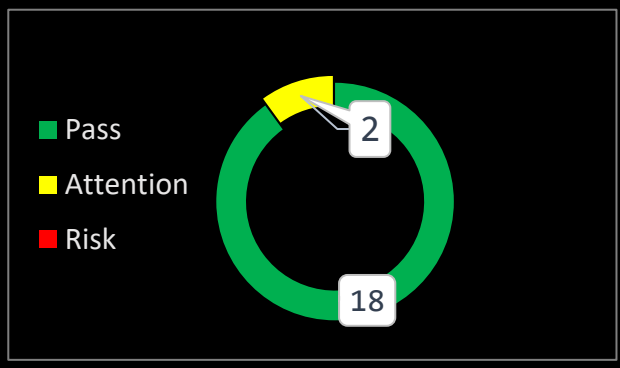
| | |
|---|---|
| Contract Deployed | 21 Aug 2023 |
| Audit Created | 29 Sept 2023 |
| Audit Update | V 1.0 |

## Verified Socials

| | |
|---|---|
| Website | http://BeTheChad.net |
| Telegram | https://t.me/BeTheChad |
| Twitter (X) | https://x.com/Bethechad_bsc |

## Contract Function Analysis

Pass  Attention Item  Risky Item

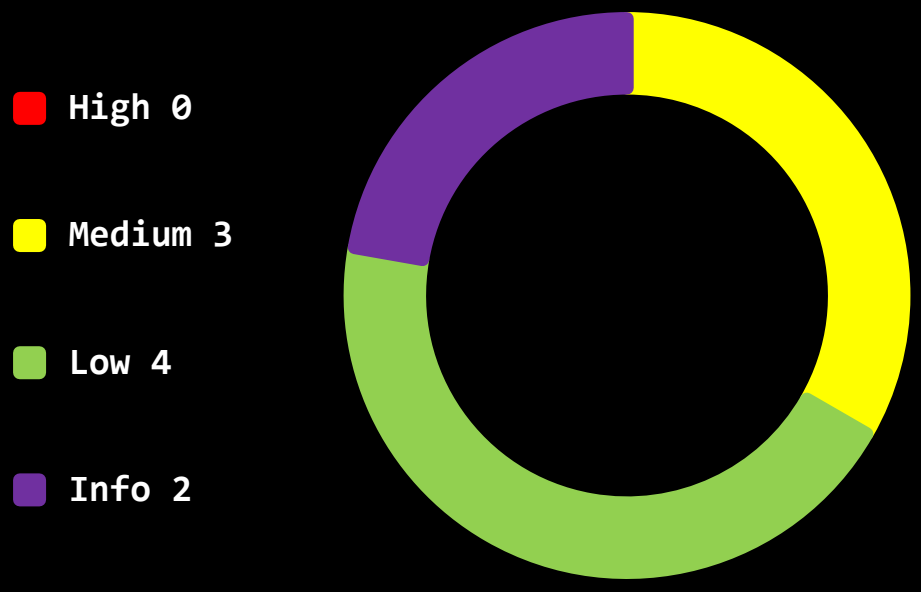



| | | |
|---|---|---|
| Contract Verified | ✔ | The contract source code is uploaded to blockchain explorer and is open source, so everybody can read it. |
| Contract Ownership | ⚠ | 0xcFC46cB90FE1DF1631e536d5b8C0AC49fecfe617 |
| Buy Tax | 6 % | Shows the taxes for purchase transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable. |
| Sell Tax | 14 % | 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 |
| Liqudity Status | ✔ | Liquidity lock status on 29.09.2023:<br>Lp Locker 1: 93.59% Mudra Locker for 64 days.<br>Lp Locker 2: 4.87% Mudra Locker for 64 days. |
| Trading Disable Functions | ✔ | No Trading suspendable function found.<br>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 | ⚠ 15% max | Fee Setting function found.<br>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). Max fee setting option: 15% |
| Proxy Contract | ✔ | Not a proxy contract! |
| Mint Function | ✔ | No Mint Function detected<br>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 | ✅ | No Balance Modifier function found.<br>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. |
| Blacklist Function | ✅ | No blacklist function found<br>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. |
| Whitelist Function | ⚠️ | Whitelist function found<br>If there is a function for this Developer can set zero fee or no max wallet size for adresses (for example team wallets can trade without fee. Can cause farming) |
| Hidden Owner Analysis | ✅ | No authorised hidden owner found.<br>For contract with a hidden owner, developer can still manipulate the contract even if the ownership has been abandoned. Fake renounce. |
| Retrieve Ownership Function | ✅ | No functions found which can retrieve ownership of the contract.<br>If this function exists, it is possible for the project owner to regain ownership even after relinquishing it. Also known as fake renounce. |
| Self Destruct Function | ✅ | No Self Destruct function found.<br>If this function exists and is triggered, the contract will be destroyed, all functions will be unavailable, and all related assets will be erased. |
| Specific Tax Changing Function | ✅ | No Specific Tax Changing Functions found.<br>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! |
| Trading Cooldown Function | ✅ | No Trading Cooldown Function found. 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. |
| Max Transaction and Holding Modify Function | ✅ | No Max Transaction and Holding Modify function found.<br>If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot |
| Transaction Limiting Function | ✅ | No Transaction Limiter Function Found.<br>The number of overall token transactions may be limited (honeypot risk) |

Contract Security

Total Findings: 9




**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: (3)

- Approve of front running attack
- TX Origin used
- Unchecked transfer

### Low severity issues: (4)

- Numeric Notation Best Practices
- Use of Floating Pragma
- Low level Calls
- Missing Events

### Informational severity issues: (2)

- Hard Coded Address
- Public Functions Should be Declared External

## 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 | Low | Passed | Passed |
| SWC-114 | Transaction Order Dependence | Passed | Passed | Passed |
| SWC-115 | Authorization through tx.origin | Medium | Medium | Medium |
| 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 | **Passed** | **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.

### ⚠ Approve of front running attack (2 Items)

| Item: 1 | Location: | Line 266-274 | Severity: | Medium |
|---|---|---|---|---|

| | |
|---|---|
| **Function** | The approve() method overrides current allowance regardless of whether the spender already used it or not, so there is no way to increase or decrease allowance by a certain value atomically unless the token owner is a smart contract, not an account.<br>This can be abused by a token receiver when they try to withdraw certain tokens from the sender's account. Meanwhile, if the sender decides to change the amount and sends another approve transaction, the receiver can notice this transaction before it's mined and can extract tokens from both the transactions, therefore, ending up with tokens from both the transactions. This is a front-running attack affecting the ERC20 Approve function.<br>The function approve can be front-run by abusing the _approve function. |
| **Remedation** | 1. Introduce mechanisms that limit the maximum acceptable gas price for transactions. This can help prevent front-runners from drastically increasing the gas fees to prioritize their transactions.<br><br>2. Use transaction taxes to prevent against front-run attack |

```
266     function approve(address spender, uint256 amount)
267         public
268         virtual
269         override
270         returns (bool)
271     {
272         _approve(_msgSender(), spender, amount);
273         return true;
274     }
```

| Item: 2 | Location: | Line 276-291 | Severity: | Medium |
|---|---|---|---|---|

| | |
|---|---|
| **Function** | The transferFrom() method overrides current allowance regardless of whether the spender already used it or not, so there is no way to increase or decrease allowance by a certain value atomically unless the token owner is a smart contract, not an account.<br>This can be abused by a token receiver when they try to withdraw certain tokens from the sender's account. Meanwhile, if the sender decides to change the amount and sends another approve transaction, the receiver can notice this transaction before it's mined and can extract tokens from both the transactions, therefore, ending up with tokens from both the transactions. This is a front-running attack affecting the ERC20 Approve function.<br>The function transferFrom can be front-run by abusing the _approve function. |
| **Remedation** | 1. Introduce mechanisms that limit the maximum acceptable gas price for transactions. This can help prevent front-runners from drastically increasing the gas fees to prioritize their transactions.<br><br>2. Use transaction taxes to prevent against front-run attack |

```
276     function transferFrom(
277         address sender,
278         address recipient,
279         uint256 amount
280     ) public virtual override returns (bool) {
281         _transfer(sender, recipient, amount);
282         _approve(
283             sender,
284             _msgSender(),
285             _allowances[sender][_msgSender()].sub(
286                 amount,
287                 "ERC20: transfer amount exceeds allowance"
288             )
289         );
290         return true;
291     }
```

## ⚠ Use of Tx. Origin (6 Items)

| | | | | |
|---|---|---|---|---|
| Item: 1 | Location: | Line 1401 | Severity: | Medium |
| Item: 2 | Location: | Line 1495 | Severity: | Medium |
| Item: 3 | Location: | Line 1497 | Severity: | Medium |
| Item: 4 | Location: | Line 1511 | Severity: | Medium |
| Item: 5 | Location: | Line 1512 | Severity: | Medium |
| Item: 6 | Location: | Line 1592 | Severity: | Medium |

| | |
|---|---|
| **Function** | In Solidity, tx.origin is a global variable that returns the address of the account that sent the transaction. Using the variable for authorization could make a contract vulnerable. For example, if an authorized account calls a malicious contract which triggers it to call the vulnerable contract that passes an authorization check since tx.origin returns the original sender of the transaction which in this case is the authorized account. |
| Remedation | tx.origin should not be used for authorization in smart contracts. It does have some legitimate use cases, for example, To prevent external contracts from calling the current contract, you can implement a require of the form require(tx.origin == msg.sender). This prevents intermediate contracts from calling the current contract, thus limiting the contract to regular codeless addresses. |

```
1400            gas,
1401            tx.origin
```

```
1494        if (limitsInEffect) {
1495        require(block.timestamp >= _holderLastTransferTimestamp[tx.origin] + cooldowntimer,
1496                "cooldown period active");
1497        _holderLastTransferTimestamp[tx.origin] = block.timestamp;
```

```
1511        require(_holderLastTransferBlock[tx.origin] != block.number,"Too many TX in block");
1512        _holderLastTransferBlock[tx.origin] = block.number;
```

```
1591            gas,
1592            tx.origin
```

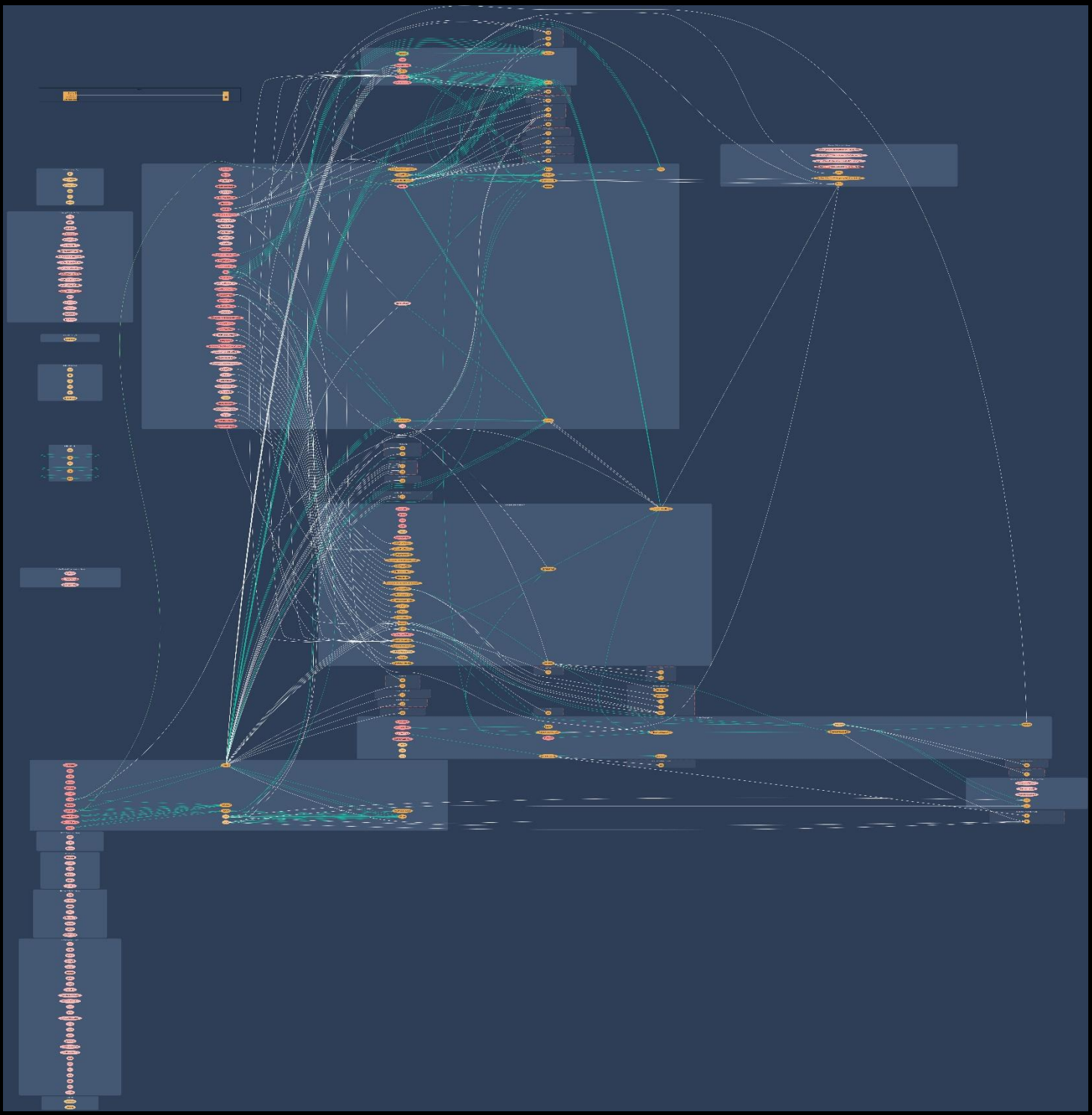
## ⚠ Unchecked Transfers (1 Item)

| Item: 1 | Location: | Line 1746-1786 | Severity: | Medium |
|---|---|---|---|---|

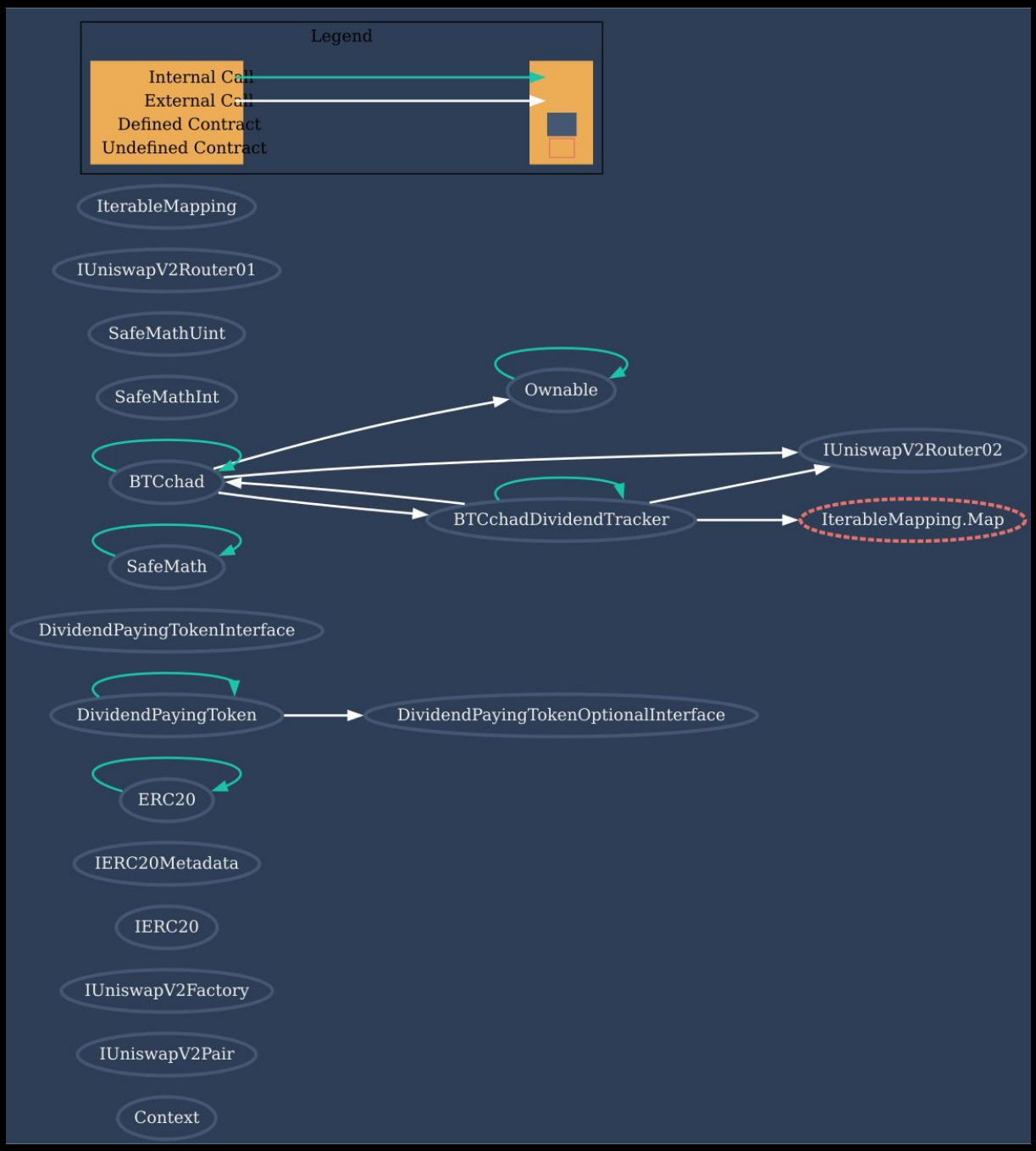
| | |
|---|---|
| **Function** | Some tokens do not revert the transaction when the transfer or transferFrom fails and returns False. Hence we must check the return value after calling the transfer or transferFrom function. |
| **Remedation** | Use OpenZeppelin SafeERC20's safetransfer and safetransferFrom functions. |

```
1746    function multiSend(
1747        address[] memory _contributors,
1748        uint256[] memory _balances
1749    ) public onlyOwner {
1750        require(
1751            _contributors.length == _balances.length,
1752            "Contributors and balances must be same size"
1753        );
1754        // Max 200 sends in bulk, uint8 in loop limited to 255
1755        require(
1756            _contributors.length <= 200,
1757            "Contributor list length must be <= 200"
1758        );
1759        uint256 sumOfBalances = 0;
1760        for (uint8 i = 0; i < _balances.length; i++) {
1761            sumOfBalances = sumOfBalances.add(_balances[i]);
1762        }
1763        require(
1764            balanceOf(msg.sender) >= sumOfBalances,
1765            "Account balance must be >= sum of balances. "
1766        );
1767        require(
1768            allowance(msg.sender, address(this)) >= sumOfBalances,
1769            "Contract allowance must be >= sum of balances. "
1770        );
1771        address contributor;
1772        uint256 origBalance;
1773        for (uint8 j; j < _contributors.length; j++) {
1774            contributor = _contributors[j];
1775            require(
1776                contributor != address(0) &&
1777                    contributor != 0x000000000000000000000000000000000000dEaD,
1778                "Cannot airdrop to a dead address"
1779            );
1780            origBalance = balanceOf(contributor);
1781            this.transferFrom(msg.sender, contributor, _balances[j]);
1782            require(
1783                balanceOf(contributor) == origBalance + _balances[j],
1784                "Contributor must recieve full balance of airdrop"
1785            );
1786            emit Airdrop(contributor, _balances[j]);
```

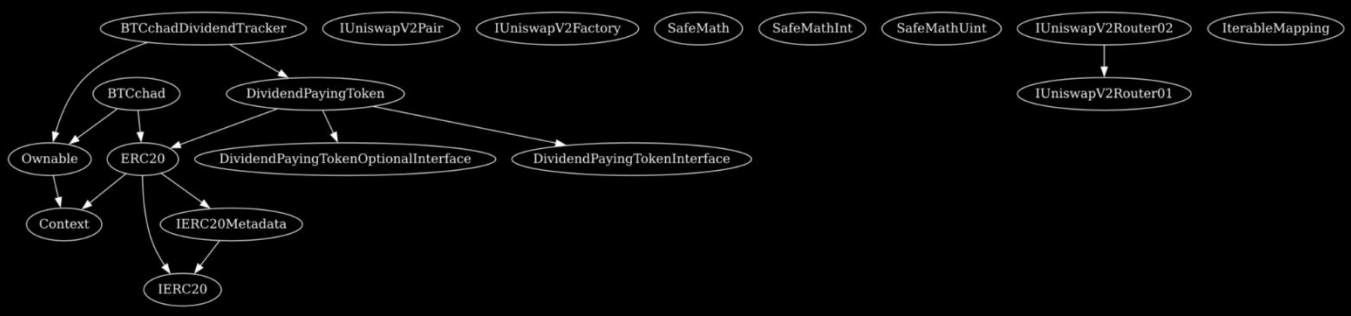
## Contract Flow Graph



## Contract Interaction Graph



## Inheritance Graph



## Contract Functions

| Contract | Type | Bases | | |
|---|---|---|---|---|
| | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
| | | | | |
| **Context** | Implementation | | | |
| | _msgSender | Internal 🔒 | | |
| | _msgData | Internal 🔒 | | |
| | | | | |
| **IUniswapV2Pair** | Interface | | | |
| | name | External ❗ | | NO❗ |
| | symbol | External ❗ | | NO❗ |
| | decimals | External ❗ | | NO❗ |
| | totalSupply | External ❗ | | NO❗ |
| | balanceOf | External ❗ | | NO❗ |
| | allowance | External ❗ | | NO❗ |
| | approve | External ❗ | 🛑 | NO❗ |
| | transfer | External ❗ | 🛑 | NO❗ |
| | transferFrom | External ❗ | 🛑 | NO❗ |
| | DOMAIN_SEPARATOR | External ❗ | | NO❗ |
| | PERMIT_TYPEHASH | External ❗ | | NO❗ |
| | nonces | External ❗ | | NO❗ |
| | permit | External ❗ | 🛑 | NO❗ |
| | MINIMUM_LIQUIDITY | External ❗ | | NO❗ |
| | factory | External ❗ | | NO❗ |
| | token0 | External ❗ | | NO❗ |
| | token1 | External ❗ | | NO❗ |
| | getReserves | External ❗ | | NO❗ |
| | price0CumulativeLast | External ❗ | | NO❗ |
| | price1CumulativeLast | External ❗ | | NO❗ |
| | kLast | External ❗ | | NO❗ |
| | mint | External ❗ | 🛑 | NO❗ |
| | burn | External ❗ | 🛑 | NO❗ |
| | swap | External ❗ | 🛑 | NO❗ |

| | | | | |
|---|---|---|---|---|
| | skim | External ! | 🛑 | NO! |
| | sync | External ! | 🛑 | NO! |
| | initialize | External ! | 🛑 | NO! |
| | | | | |
| **IUniswapV2Factory** | Interface | | | |
| | feeTo | External ! | | NO! |
| | feeToSetter | External ! | | NO! |
| | getPair | External ! | | NO! |
| | allPairs | External ! | | NO! |
| | allPairsLength | External ! | | NO! |
| | createPair | External ! | 🛑 | NO! |
| | setFeeTo | External ! | 🛑 | NO! |
| | setFeeToSetter | External ! | 🛑 | NO! |
| | | | | |
| **IERC20** | Interface | | | |
| | totalSupply | External ! | | NO! |
| | balanceOf | External ! | | NO! |
| | transfer | External ! | 🛑 | NO! |
| | allowance | External ! | | NO! |
| | approve | External ! | 🛑 | NO! |
| | transferFrom | External ! | 🛑 | NO! |
| | | | | |
| **IERC20Metadata** | Interface | IERC20 | | |
| | name | External ! | | NO! |
| | symbol | External ! | | NO! |
| | decimals | External ! | | NO! |
| | | | | |
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata | | |
| | | Public ! | 🛑 | NO! |
| | name | Public ! | | NO! |
| | symbol | Public ! | | NO! |
| | decimals | Public ! | | NO! |
| | totalSupply | Public ! | | NO! |
| | balanceOf | Public ! | | NO! |
| | transfer | Public ! | 🛑 | NO! |
| | allowance | Public ! | | NO! |
| | approve | Public ! | 🛑 | NO! |

| | | | | |
|---|---|---|---|---|
| | transferFrom | Public ❗ | 🛑 | NO❗ |
| | increaseAllowance | Public ❗ | 🛑 | NO❗ |
| | decreaseAllowance | Public ❗ | 🛑 | NO❗ |
| | _transfer | Internal 🔒 | 🛑 | |
| | _mint | Internal 🔒 | 🛑 | |
| | _burn | Internal 🔒 | 🛑 | |
| | _approve | Internal 🔒 | 🛑 | |
| | _beforeTokenTransfer | Internal 🔒 | 🛑 | |
| | | | | |
| **DividendPayingTokenOptionalInterface** | Interface | | | |
| | withdrawableDividendOf | External ❗ | | NO❗ |
| | withdrawnDividendOf | External ❗ | | NO❗ |
| | accumulativeDividendOf | External ❗ | | NO❗ |
| | | | | |
| **DividendPayingTokenInterface** | Interface | | | |
| | dividendOf | External ❗ | | NO❗ |
| | distributeDividends | External ❗ | 💵 | NO❗ |
| | withdrawDividend | External ❗ | 🛑 | NO❗ |
| | | | | |
| **SafeMath** | Library | | | |
| | add | Internal 🔒 | | |
| | sub | Internal 🔒 | | |
| | sub | Internal 🔒 | | |
| | mul | Internal 🔒 | | |
| | div | Internal 🔒 | | |
| | div | Internal 🔒 | | |
| | mod | Internal 🔒 | | |
| | mod | Internal 🔒 | | |
| | | | | |
| **Ownable** | Implementation | Context | | |

| | | | | |
|---|---|---|---|---|
| | | Public ❗ | 🛑 | NO❗ |
| | owner | Public ❗ | | NO❗ |
| | renounceOwnership | Public ❗ | 🛑 | onlyOwner |
| | transferOwnership | Public ❗ | 🛑 | onlyOwner |
| | | | | |
| **SafeMathInt** | Library | | | |
| | mul | Internal 🔒 | | |
| | div | Internal 🔒 | | |
| | sub | Internal 🔒 | | |
| | add | Internal 🔒 | | |
| | abs | Internal 🔒 | | |
| | toUint256Safe | Internal 🔒 | | |
| | | | | |
| **SafeMathUint** | Library | | | |
| | toInt256Safe | 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! |
| | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ! | 🛑 | NO! |
| | swapExactETHForTokensSupportingFeeOnTransferTokens | External ! | 💵 | NO! |
| | swapExactTokensForETHSupportingFeeOnTransferTokens | External ! | 🛑 | NO! |
| | | | | |
| **DividendPayingToken** | Implementation | ERC20, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface | | |
| | | Public ! | 🛑 | ERC20 |
| | | External ! | 💵 | NO! |
| | distributeDividends | Public ! | 💵 | NO! |

| | | | | |
|---|---|---|---|---|
| | 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 🔒 | 🛑 | |
| | | | | |
| **BTCchad** | Implementation | ERC20, Ownable | | |
| | | Public ❗ | 🛑 | ERC20 |
| | decimals | Public ❗ | | NO❗ |
| | | External ❗ | 💵 | NO❗ |
| | updateStakingAmounts | Public ❗ | 🛑 | onlyOwner |
| | enableTrading | External ❗ | 🛑 | onlyOwner |
| | setPresaleWallet | External ❗ | 🛑 | onlyOwner |
| | setExcludeFees | Public ❗ | 🛑 | onlyOwner |
| | setExcludeDividends | Public ❗ | 🛑 | onlyOwner |
| | setIncludeDividends | Public ❗ | 🛑 | onlyOwner |
| | setCanTransferBefore | External ❗ | 🛑 | onlyOwner |
| | setLimitsInEffect | External ❗ | 🛑 | onlyOwner |
| | setGasPriceLimit | External ❗ | 🛑 | onlyOwner |
| | setcooldowntimer | External ❗ | 🛑 | onlyOwner |
| | setmaxWallet | External ❗ | 🛑 | onlyOwner |
| | enableStaking | Public ❗ | 🛑 | onlyOwner |
| | stake | Public ❗ | 🛑 | NO❗ |
| | setSwapTriggerAmount | Public ❗ | 🛑 | onlyOwner |

| | | | | |
|---|---|---|---|---|
| | enableSwapAndLiquify | Public ❗ | 🛑 | onlyOwner |
| | setAutomatedMarketMakerPair | Public ❗ | 🛑 | onlyOwner |
| | setAllowCustomTokens | Public ❗ | 🛑 | onlyOwner |
| | setAllowAutoReinvest | Public ❗ | 🛑 | onlyOwner |
| | _setAutomatedMarketMakerPair | Private 🔐 | 🛑 | |
| | updateGasForProcessing | Public ❗ | 🛑 | onlyOwner |
| | transferAdmin | Public ❗ | 🛑 | onlyOwner |
| | updateTransferFee | Public ❗ | 🛑 | onlyOwner |
| | updateFees | Public ❗ | 🛑 | onlyOwner |
| | getStakingInfo | External ❗ | | NO❗ |
| | getTotalDividendsDistributed | External ❗ | | NO❗ |
| | isExcludedFromFees | Public ❗ | | NO❗ |
| | withdrawableDividendOf | Public ❗ | | NO❗ |
| | dividendTokenBalanceOf | Public ❗ | | NO❗ |
| | getAccountDividendsInfo | External ❗ | | NO❗ |
| | getAccountDividendsInfoAtIndex | External ❗ | | NO❗ |
| | processDividendTracker | External ❗ | 🛑 | NO❗ |
| | claim | External ❗ | 🛑 | NO❗ |
| | getLastProcessedIndex | External ❗ | | NO❗ |
| | getNumberOfDividendTokenHolders | External ❗ | | NO❗ |
| | setAutoClaim | External ❗ | 🛑 | NO❗ |
| | setReinvest | External ❗ | 🛑 | NO❗ |
| | setDividendsPaused | External ❗ | 🛑 | onlyOwner |

| | | | | |
|---|---|---|---|---|
| | isExcludedFromAutoClaim | External ❗ | | NO❗ |
| | isReinvest | External ❗ | | NO❗ |
| | _transfer | Internal 🔒 | 🛑 | |
| | getStakingBalance | Private 🔐 | | |
| | swapAndLiquify | Private 🔐 | 🛑 | |
| | swapTokensForEth | Private 🔐 | 🛑 | |
| | updatePayoutToken | Public ❗ | 🛑 | onlyOwner |
| | getPayoutToken | Public ❗ | | NO❗ |
| | setMinimumTokenBalanceForAutoDividends | Public ❗ | 🛑 | onlyOwner |
| | setMinimumTokenBalanceForDividends | Public ❗ | 🛑 | onlyOwner |
| | addLiquidity | Private 🔐 | 🛑 | |
| | forceSwapAndSendDividends | Public ❗ | 🛑 | onlyOwner |
| | swapAndSendDividends | Private 🔐 | 🛑 | |
| | multiSend | Public ❗ | 🛑 | onlyOwner |
| | airdropToWallets | External ❗ | 🛑 | onlyOwner |
| | | | | |
| **BTCchadDividendTracker** | Implementation | DividendPayingToken, Ownable | | |
| | | Public ❗ | 🛑 | DividendPayingToken |
| | decimals | Public ❗ | | NO❗ |
| | name | Public ❗ | | NO❗ |
| | symbol | Public ❗ | | NO❗ |
| | _transfer | Internal 🔒 | | |
| | withdrawDividend | Public ❗ | | NO❗ |
| | isExcludedFromAutoClaim | External ❗ | | onlyOwner |
| | isReinvest | External ❗ | | onlyOwner |

| | | | | |
|---|---|---|---|---|
| | setAllowCustomTokens | External ❗ | 🛑 | onlyOwner |
| | setAllowAutoReinvest | External ❗ | 🛑 | onlyOwner |
| | excludeFromDividends | External ❗ | 🛑 | onlyOwner |
| | includeFromDividends | External ❗ | 🛑 | onlyOwner |
| | setAutoClaim | External ❗ | 🛑 | onlyOwner |
| | setReinvest | External ❗ | 🛑 | onlyOwner |
| | setMinimumTokenBalanceForAutoDividends | External ❗ | 🛑 | onlyOwner |
| | setMinimumTokenBalanceForDividends | External ❗ | 🛑 | onlyOwner |
| | setDividendsPaused | External ❗ | 🛑 | onlyOwner |
| | getLastProcessedIndex | External ❗ | | NO❗ |
| | getNumberOfTokenHolders | External ❗ | | NO❗ |
| | getAccount | Public ❗ | | NO❗ |
| | getAccountAtIndex | Public ❗ | | NO❗ |
| | setBalance | External ❗ | 🛑 | onlyOwner |
| | process | Public ❗ | 🛑 | NO❗ |
| | processAccount | Public ❗ | 🛑 | onlyOwner |
| | updateUniswapV2Router | Public ❗ | 🛑 | onlyOwner |
| | updatePayoutToken | Public ❗ | 🛑 | onlyOwner |
| | getPayoutToken | Public ❗ | | NO❗ |
| | _reinvestDividendOfUser | Private 🔐 | 🛑 | |
| | _withdrawDividendOfUser | Internal 🔒 | 🛑 | |
| | | | | |
| **IterableMapping** | Library | | | |
| | get | Internal 🔒 | | |
| | getIndexOfKey | Internal 🔒 | | |

| | | | | |
|---|---|---|---|---|
| | getKeyAtIndex | Internal 🔒 | | |
| | size | Internal 🔒 | | |
| | set | Internal 🔒 | 🛑 | |
| | remove | Internal 🔒 | 🛑 | |

🛑 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 vulnaribilities in the code. Findings getting reported and improvements getting suggested.

**Automatic and Manual Review**
We are using automated tools to scan functions and weeknesses 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