freshcoins

SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

SHIBACCA

$BACCA

16/02/2022

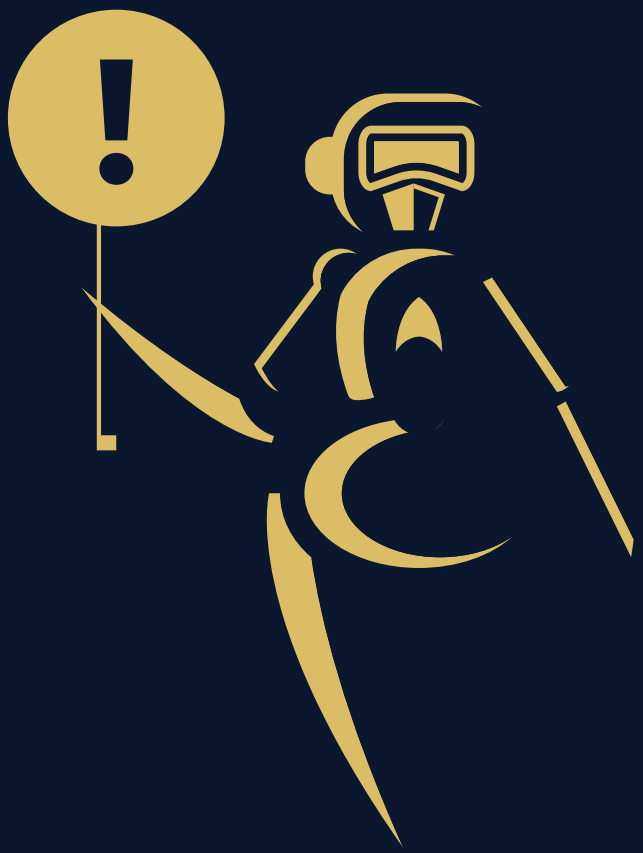
TABLE OF CONTENTS

DISCLAIMER

The information provided on this analysis document is only for general information and should not be used as a reason to invest.

FreshCoins Team will take no payment for manipulating the results of this audit.

The score and the result will stay on this project page information on our website https://freshcoins.io
FreshCoins Team does not guarantees that a project will not sell off team supply, or any other scam strategy (RUG or Honeypot etc)



INTRODUCTION

FreshCoins (Consultant) was contracted by SHIBACCA (Customer) to conduct a Smart Contract Code Review and Security Analysis.

0xfD2d4925a9f9a21c500085F4A55f7F7237A2fcb3

Network: Binance Smart Chain (BSC)

This report presents the findings of the security assessment of Customer's smart contract and its code review conducted on 16/02/2022

WEBSITE DIAGNOSTIC

https://shibacca.net/

0-49 | 50-89 | 90-100

84	97	92	93	NA
Performance	Accessibility	Best Practices	SEO	Progressive Web App

Metrics

First Contentful Paint
3.1 s

Time to interactive
21.1 s

Speed Index
2.6 s

Total Blocking Time
8,941 ms

Large Contentful Paint
15.2 s

Cumulative Layout Shift
0

WEBSITE IMPROVEMENTS

Reduce unused CSS

Reduce unused JavaScript

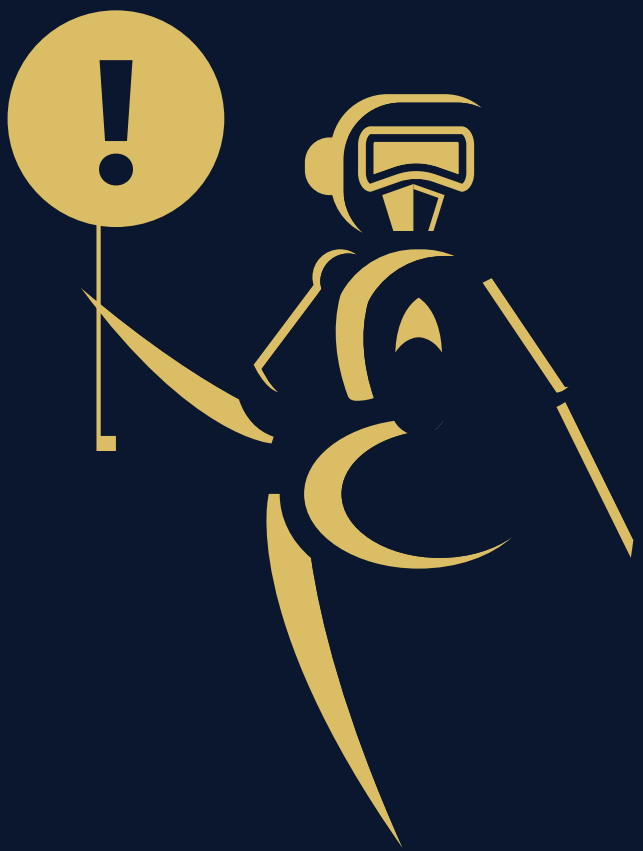
Ensure text remains visible during webfont load

Reduce JavaScript execution time 2.4 s

Avoid enormous network payloads Total size was 3,638 KiB

Links do not have a discernible name

Heading elements are not in a sequentially-descending order



AUDIT OVERVIEW

95

Security Score

93

Static Scan
Automatic scanning for common vulnerabilities

98

ERC Scan
Automatic checks for ERC's conformance

0 High

0 Medium

0 Low

0 Optimizations

0 Informational

No.	Issue description	Checking Status
1	Compiler Errors / Warnings	Passed
2	Reentrancy and Cross-function	Passed
3	Front running	Passed
4	Timestamp dependence	Passed
5	Integer Overflow and Underflow	Passed
6	Reverted DoS	Passed
7	DoS with block gas limit	Passed
8	Methods execution permissions	Passed
9	Exchange rate impact	Passed
10	Malicious Event	Passed
11	Scoping and Declarations	Passed
12	Uninitialized storage pointers	Passed
13	Design Logic	Passed
14	Safe Zeppelin module	Passed

OWNER PRIVILEGES

Contract owner can exclude wallet from dividends

```
function excludeFromDividends(address account) public onlyOwner {
        dividendTracker.excludeFromDividends(account);
}
```

Contract owner can exclude/include wallet from fees

```
function excludeFromFees(address account, bool excluded) public onlyOwner {
        require(_isExcludedFromFee[account] != excluded, "BACCA: Account is already the value of 'exclud-
ed'");
        _isExcludedFromFee[account] = excluded;
        emit ExcludeFromFeesChange(account, excluded);
}
```

Contract owner can exclude/include wallet from tx limitations

```
function excludeFromMaxTransactionLimit(address account, bool excluded) public onlyOwner {
        require(_isExcludedFromMaxTransactionLimit[account] != excluded, "BACCA: Account is already the
value of 'excluded'");
        _isExcludedFromMaxTransactionLimit[account] = excluded;
        emit ExcludeFromMaxTransferChange(account, excluded);
}
```

Contract owner can exclude/include wallet from wallet limitations

```
function excludeFromMaxWalletLimit(address account, bool excluded) public onlyOwner {
        require(_isExcludedFromMaxWalletLimit[account] != excluded, "BACCA: Account is already the value
of 'excluded'");
        _isExcludedFromMaxWalletLimit[account] = excluded;
        emit ExcludeFromMaxWalletChange(account, excluded);
}
```

Contract owner can change max tx amount

```
function setMaxTransactionAmount(uint256 newValue) public onlyOwner {
        require(newValue != maxTxAmount, "BACCA: Cannot update maxTxAmount to same value");
        emit MaxTransactionAmountChange(newValue, maxTxAmount);
        maxTxAmount = newValue;
}
```

Contract owner can change max wallet amount

```
function setMaxWalletAmount(uint256 newValue) public onlyOwner {
        require(newValue != maxWalletAmount, "BACCA: Cannot update maxWalletAmount to same
value");
        emit MaxWalletAmountChange(newValue, maxWalletAmount);
        maxWalletAmount = newValue;
}
```

Contract owner can change liquidityWallet address

Current address:

liquidityWallet: 0xffffe48c0c3612dc8f2897a565fb8d527593d854

```
function setWallets(address newLiquidityWallet) public onlyOwner {
        if(liquidityWallet != newLiquidityWallet) {
                require(newLiquidityWallet != address(0), "BACCA: The liquidityWallet cannot be 0");
                emit WalletChange('liquidityWallet', newLiquidityWallet, liquidityWallet);
                liquidityWallet = newLiquidityWallet;
        }
}
```

Contract owner can change the fees up to 100%

```
function setBaseFeesOnBuy(uint256 _liquidityFeeOnBuy, uint256 _burnFeeOnBuy, uint256 _holdersFeeOn-
Buy) public onlyOwner {
        _setCustomBuyTaxPeriod(_base, _liquidityFeeOnBuy, _burnFeeOnBuy, _holdersFeeOnBuy);
        emit FeeChange('baseFees-Buy', _liquidityFeeOnBuy, _burnFeeOnBuy, _holdersFeeOnBuy);
}

function setBaseFeesOnSell(uint256 _liquidityFeeOnSell, uint256 _burnFeeOnSell, uint256 _holdersFeeOn-
Sell) public onlyOwner {
        _setCustomSellTaxPeriod(_base, _liquidityFeeOnSell, _burnFeeOnSell, _holdersFeeOnSell);
        emit FeeChange('baseFees-Sell', _liquidityFeeOnSell, _burnFeeOnSell, _holdersFeeOnSell);
}
```

Contract owner can enable/disable fees on token transfers

```
function setFeeOnWalletTransfers(bool value) public onlyOwner {
        emit FeeOnWalletTransferChange(value, _feeOnWalletTranfers);
        _feeOnWalletTranfers = value;
}
```

Contract owner can exclude/include an address from transactions.

```
function blockAccount(address account) public onlyOwner {
        uint256 currentTimestamp = _getNow();
        require(!_isBlocked[account], "BACCA: Account is already blocked");
        if (_isLaunched) {
                require(currentTimestamp.sub(_launchStartTimestamp) < _blockedTimeLimit, "BACCA: Time
to block accounts has expired");
        }
        _isBlocked[account] = true;
        emit BlockedAccountChange(account, true);
}

function unblockAccount(address account) public onlyOwner {
        require(_isBlocked[account], "BACCA: Account is not blcoked");
        _isBlocked[account] = false;
        emit BlockedAccountChange(account, false);
}
```

Contract owner can enable/disable trading

```
function activateTrading() public onlyOwner {
        isTradingEnabled = true;
}

function deactivateTrading() public onlyOwner {
        isTradingEnabled = false;
        _tradingPausedTimestamp = _getNow();
}
```

Contract owner can allow specific wallet to trade even if trading is disabled

```
function allowTradingWhenDisabled(address account, bool allowed) public onlyOwner {
        _isAllowedToTradeWhenDisabled[account] = allowed;
        emit AllowedWhenTradingDisabledChange(account, allowed);
}
```

Contract owner can reset all fees at default value

```
function resetAllFees() public onlyOwner {
        _setCustomBuyTaxPeriod(_base, _default.liquidityFeeOnBuy, _default.burnFeeOnBuy,
_default.holdersFeeOnBuy);
        emit FeeChange('baseFees-Buy', _default.liquidityFeeOnBuy, _default.burnFeeOnBuy,
_default.holdersFeeOnBuy);
        _setCustomSellTaxPeriod(_base, _default.liquidityFeeOnSell, _default.burnFeeOnSell,
_default.holdersFeeOnSell);
        emit FeeChange('baseFees-Sell', _default.liquidityFeeOnSell, _default.burnFeeOnSell, _default.holder-
sFeeOnSell);
}
```

Contract owner can change the fees after the launch for the first 60 minutes.

```
function setLaunch2FeesOnBuy(uint256 _liquidityFeeOnBuy, uint256 _burnFeeOnBuy, uint256 _holders-
FeeOnBuy) public onlyOwner {
        _setCustomBuyTaxPeriod(_launch2, _liquidityFeeOnBuy, _burnFeeOnBuy, _holdersFeeOnBuy);
        emit FeeChange('launch2Fees-Buy', _liquidityFeeOnBuy, _burnFeeOnBuy, _holdersFeeOnBuy);
}

function setLaunch2FeesOnSell(uint256 _liquidityFeeOnSell, uint256 _burnFeeOnSell, uint256 _holders-
FeeOnSell) public onlyOwner {
        _setCustomSellTaxPeriod(_launch2, _liquidityFeeOnSell, _burnFeeOnSell, _holdersFeeOnSell);
        emit FeeChange('launch2Fees-Sell', _liquidityFeeOnSell, _burnFeeOnSell, _holdersFeeOnSell);
}
```

Contract owner can change the fees after the first hour of launch for 24 hours.

```
function setLaunch3FeesOnBuy(uint256 _liquidityFeeOnBuy, uint256 _burnFeeOnBuy, uint256 _holders-
FeeOnBuy) public onlyOwner {
        _setCustomBuyTaxPeriod(_launch3, _liquidityFeeOnBuy, _burnFeeOnBuy, _holdersFeeOnBuy);
        emit FeeChange('launch3Fees-Buy', _liquidityFeeOnBuy, _burnFeeOnBuy, _holdersFeeOnBuy);
}

function setLaunch3FeesOnSell(uint256 _liquidityFeeOnSell, uint256 _burnFeeOnSell, uint256 _holders-
FeeOnSell) public onlyOwner {
        _setCustomSellTaxPeriod(_launch3, _liquidityFeeOnSell, _burnFeeOnSell, _holdersFeeOnSell);
        emit FeeChange('launch3Fees-Sell', _liquidityFeeOnSell, _burnFeeOnSell, _holdersFeeOnSell);
}
```

Contract owner can change all fees value to zero

```
function setAllFeesToZero() public onlyOwner {
        _setCustomBuyTaxPeriod(_base, 0, 0, 0);
        emit FeeChange('baseFees-Buy', 0, 0, 0);
        _setCustomSellTaxPeriod(_base, 0, 0, 0);
        emit FeeChange('baseFees-Sell', 0, 0, 0);
}
```

Contract owner can renounce ownership

```
function renounceOwnership() public virtual onlyOwner {
        emit OwnershipTransferred(_owner, address(0));
        _owner = address(0);
}
```

Contract owner can transfer ownership

```
function transferOwnership(address newOwner) public virtual onlyOwner {
        require(newOwner != address(0), "Ownable: new owner is the zero address");
        emit OwnershipTransferred(_owner, newOwner);
        _owner = newOwner;
}
```

CONCLUSION AND ANALYSIS

Smart Contracts within the scope were manually reviewed and analyzed with static tools.

Audit report overview contains all found security vulnerabilities and other issues in the reviewed code.

Found no issue during the first review.

TOKEN DETAILS

Details

launch1 - first 3 transactions: 100%
launch2 - after launch1 first 60 minutes: 10%
launch3 - after launch2 for 24 hours: 12%

Base Buy fees: 12%

launch1 - N/A
launch2 - after launch1 first 60 minutes: 30%
launch3 - after launch2 for 24 hours: 20%

Base Sell fees: 12%

Max TX: 2,500,000

Max Sell: N/A

Honeypot Risk

Ownership: Owned

Blacklist: Detected

Modify Max TX: Detected

Modify Max Sell: Not detected

Disable Trading: Detected

Rug Pull Risk

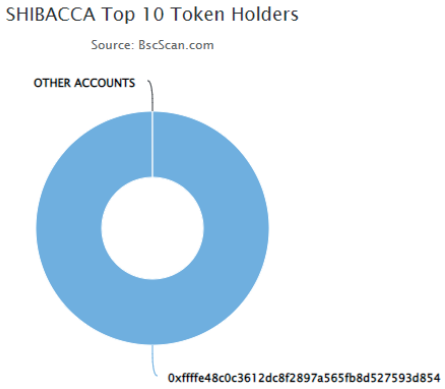
Liquidity: N/A

Holders: Clean

SHIBACCA TOKEN ANALYTICS & TOP 10 TOKEN HOLDERS

The top 10 holders collectively own 100.00% (1,000,000,000.00 Tokens) of SHIBACCA

Token Total Supply: 1,000,000,000.00 Token | Total Token Holders: 1



(A total of 1,000,000,000.00 tokens held by the top 10 accounts from the total supply of 1,000,000,000.00 token)

Rank	Address	Quantity (Token)	Percentage
1	0xffffe48c0c3612dc8f2897a565fb8d527593d854	1,000,000,000	100.0000%

TECHNICAL DISCLAIMER

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have its vulnerabilities that can lead to hacks. The audit can't guarantee the explicit security of the audited project / smart contract.