Techrate1 Techrate

TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

June, 2021

Audit Details

Audited project

Shibby

Deployer address

0xd8976283F0A0fA7e43a0910711B907a159D00CF1

Client contacts:

Shibby team

Blockchain

Binance Smart Chain

Project website:

https://shibby.finance/

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

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 TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (TechRate) owe no duty of care towards you or any other person, nor does TechRate 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 TechRate 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, TechRate hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against TechRate, 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.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Background

TechRate was commissioned by Shibby to perform an audit of smart contracts:

https://bscscan.com/address/0xB1035523a844371C2877f8a3b2F2f8d337403b6F#code

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contracts Details

Token contract details for 14.06.2021

Contract name	Shibby
Contract address	0xB1035523a844371C2877f8a3b2F2f8d337403b6F
Total supply	1,000,000,000,000,000
Token ticker	SHIBBY
Decimals	9
Token holders	11,671
Transactions count	52,138
Top 100 holders dominance	84.83%
Liquidity fee	2
Tax fee	4
Total fees	90126003841203513993742
Pancake V2 pair	0x3b458002be8bc1a10a846c12a24924b0df7c234a
Contract deployer address	0xd8976283F0A0fA7e43a0910711B907a159D00CF1
Contract's current owner address	0x470b69f05d2c70736337f839944cd6b2692450c5

Shibby Token Distribution

The top 100 holders collectively own 84.83% (848,251,387,330,926.00 Tokens) of Shibby

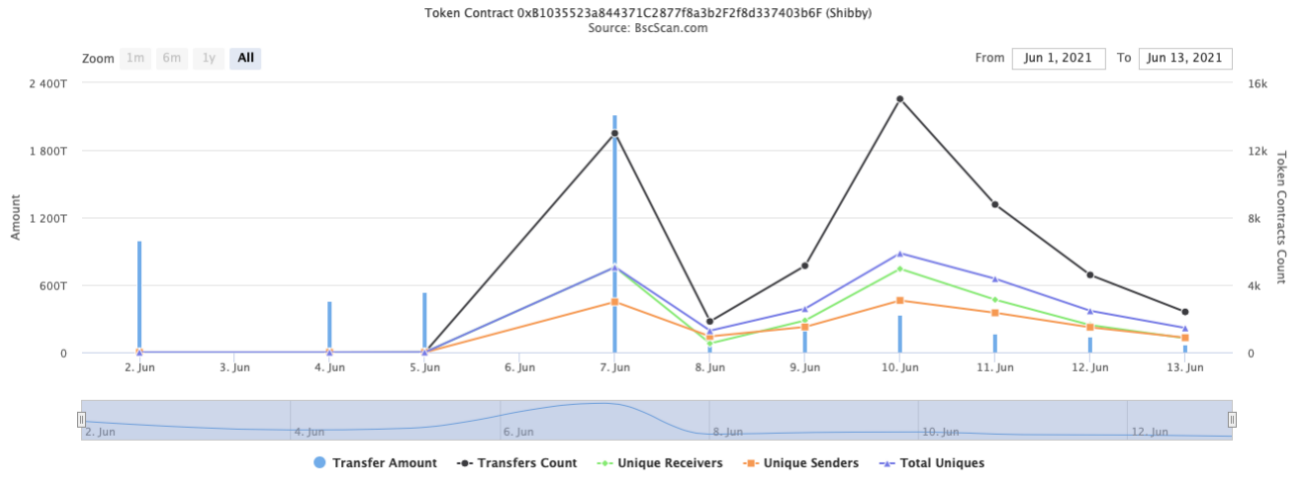
Token Total Supply: 1,000,000,000,000,000.00 Token | Total Token Holders: 11,671

(A total of 848,251,387,330,926.00 tokens held by the top 100 accounts from the total supply of 1,000,000,000,000,000.00 token)

Shibby Contract Interaction Details

Time Series: Token Contract Overview

Wed 2, Jun 2021 - Sun 13, Jun 2021



Shibby Top 10 Token Holders

Rank	Address	Quantity (Token)	Percentage
1	Burn Address	501,167,602,690,636.975696248	50.1168%
2	PancakeSwap V2: SHIBBY	54,002,343,667,063.739991282	5.4002%
3	0x68448bf87ed9ab46f004c15c1e007db34679da47	30,000,000,000,000	3.0000%
4	0xfc5bb64c56a879983d7351b4d2ab3920a829f62d	22,708,907,153,430.244743253	2.2709%
5	0x4021b258c787f439a304619b4d8304cf9fdc0f14	20,000,000,000,000	2.0000%
6	0x5c19d2175cf60832e0d22e1f142c2a5a8e5abe0e	20,000,000,000,000	2.0000%
7	0xb28d27a9968fb80beaf24a1f7aaf12dff4d9e98a	18,763,770,537,611.434647653	1.8764%
8	0xc263edd593ae7478deb885c2600104eda948c9e8	15,000,481,148,965.54478546	1.5000%
9	0x8cf530100c9106a630a3696d5a8f99668ec7bc68	10,074,786,884,255.820382992	1.0075%
10	0xfaf9cb4eb87a48fb552ad7f4bfc8a488f0ac82e9	9,183,370,614,350.600207916	0.9183%

Contract functions details

+ [Lib] SafeMath
  - [Int] tryAdd
  - [Int] trySub
  - [Int] tryMul
  - [Int] tryDiv
  - [Int] tryMod
  - [Int] add
  - [Int] sub
  - [Int] mul
  - [Int] div
  - [Int] mod
  - [Int] sub
  - [Int] div
  - [Int] mod

+ Context
  - [Int] _msgSender
  - [Int] _msgData

+ [Lib] Address
  - [Int] isContract
  - [Int] sendValue #
  - [Int] functionCall #
  - [Int] functionCall #
  - [Int] functionCallWithValue #
  - [Int] functionCallWithValue #
  - [Int] functionStaticCall
  - [Int] functionStaticCall
  - [Int] functionDelegateCall #
  - [Int] functionDelegateCall #
  - [Prv] _verifyCallResult

+ [Int] IERC20
  - [Ext] totalSupply
  - [Ext] balanceOf
  - [Ext] transfer #
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transferFrom #

+ Ownable (Context)
  - [Pub] <Constructor> #
  - [Pub] owner
  - [Pub] renounceOwnership #
    - modifiers: onlyOwner
  - [Pub] transferOwnership #
    - modifiers: onlyOwner

+ [Int] IPancakeRouter02 (IPancakeRouter01)
  - [Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
  - [Ext] removeLiquidityETHWithPermitSupportingFeeOnTransferTokens #

- [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
- [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens ($)
- [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #

+ [Int] IPancakeRouter01
- [Ext] factory
- [Ext] WETH
- [Ext] addLiquidity #
- [Ext] addLiquidityETH ($)
- [Ext] removeLiquidity #
- [Ext] removeLiquidityETH #
- [Ext] removeLiquidityWithPermit #
- [Ext] removeLiquidityETHWithPermit #
- [Ext] swapExactTokensForTokens #
- [Ext] swapTokensForExactTokens #
- [Ext] swapExactETHForTokens ($)
- [Ext] swapTokensForExactETH #
- [Ext] swapExactTokensForETH #
- [Ext] swapETHForExactTokens ($)
- [Ext] quote
- [Ext] getAmountOut
- [Ext] getAmountIn
- [Ext] getAmountsOut
- [Ext] getAmountsIn

+ [Int] IPancakePair
- [Ext] name
- [Ext] symbol
- [Ext] decimals
- [Ext] totalSupply
- [Ext] balanceOf
- [Ext] allowance
- [Ext] approve #
- [Ext] transfer #
- [Ext] transferFrom #
- [Ext] DOMAIN_SEPARATOR
- [Ext] PERMIT_TYPEHASH
- [Ext] nonces
- [Ext] permit #
- [Ext] MINIMUM_LIQUIDITY
- [Ext] factory
- [Ext] token0
- [Ext] token1
- [Ext] getReserves
- [Ext] price0CumulativeLast
- [Ext] price1CumulativeLast
- [Ext] kLast
- [Ext] mint #
- [Ext] burn #
- [Ext] swap #
- [Ext] skim #
- [Ext] sync #
- [Ext] initialize #

+ [Int] IPancakeFactory

- [Ext] feeTo
- [Ext] feeToSetter
- [Ext] getPair
- [Ext] allPairs
- [Ext] allPairsLength
- [Ext] createPair #
- [Ext] setFeeTo #
- [Ext] setFeeToSetter #

+ ShibbyToken (Context, IERC20, Ownable)
- [Pub] <Constructor> #
- [Pub] name
- [Pub] symbol
- [Pub] decimals
- [Pub] totalSupply
- [Pub] balanceOf
- [Pub] transfer #
- [Pub] allowance
- [Pub] approve #
- [Pub] transferFrom #
- [Pub] increaseAllowance #
- [Pub] decreaseAllowance #
- [Pub] isExcludedFromReward
- [Pub] totalFees
- [Pub] reflectionFromToken
- [Pub] tokenFromReflection
- [Pub] excludeFromReward #
  - modifiers: onlyOwner
- [Ext] includeInReward #
  - modifiers: onlyOwner
- [Pub] excludeFromFee #
  - modifiers: onlyOwner
- [Pub] includeInFee #
  - modifiers: onlyOwner
- [Ext] setCharityFeePercent #
  - modifiers: onlyOwner
- [Ext] setTaxFeePercent #
  - modifiers: onlyOwner
- [Ext] setLiquidityFeePercent #
  - modifiers: onlyOwner
- [Ext] setMaxTxPercent #
  - modifiers: onlyOwner
- [Pub] setSwapAndLiquifyEnabled #
  - modifiers: onlyOwner
- [Ext] <Fallback> ($)
- [Prv] _reflectFee #
- [Prv] _getValues
- [Prv] _getTValues
- [Prv] _getRValues
- [Prv] _getRate
- [Prv] _getCurrentSupply
- [Prv] _takeCharity #
- [Prv] _takeLiquidity #
- [Prv] calculateCharityFee
- [Prv] calculateTaxFee

- [Prv] calculateLiquidityFee
- [Prv] removeAllFee #
- [Prv] restoreAllFee #
- [Pub] isExcludedFromFee
- [Prv] _approve #
- [Prv] _transfer #
- [Ext] adminRescueTokens #
  - modifiers: onlyOwner
- [Ext] swapBalanceToLiquidity #
  - modifiers: onlyOwner
- [Prv] swapAndLiquify #
  - modifiers: lockTheSwap
- [Ext] swapWBNBForTokensAndBurn #
  - modifiers: onlyOwner
- [Ext] swapBNBForTokensAndBurn #
  - modifiers: onlyOwner
- [Prv] swapTokensForEth #
- [Prv] addLiquidity #
- [Prv] _tokenTransfer #
- [Prv] _transferStandard #
- [Prv] _transferToExcluded #
- [Prv] _transferFromExcluded #
- [Prv] _transferBothExcluded #

($) = payable function
= non-constant function

Issues Checking Status

	Issue description	Checking status
1.	Compiler errors.	Passed
2.	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3.	Possible delays in data delivery.	Passed
4.	Oracle calls.	Passed
5.	Front running.	Passed
6.	Timestamp dependence.	Passed
7.	Integer Overflow and Underflow.	Passed
8.	DoS with Revert.	Passed
9.	DoS with block gas limit.	Low issues
10.	Methods execution permissions.	Passed
11.	Economy model of the contract.	Passed
12.	The impact of the exchange rate on the logic.	Passed
13.	Private user data leaks.	Passed
14.	Malicious Event log.	Passed
15.	Scoping and Declarations.	Passed
16.	Uninitialized storage pointers.	Passed
17.	Arithmetic accuracy.	Passed
18.	Design Logic.	Passed
19.	Cross-function race conditions.	Passed
20.	Safe Open Zeppelin contracts implementation and usage.	Passed
21.	Fallback function security.	Passed

Security Issues

High Severity Issues

No high severity issues found.

Medium Severity Issues

No medium severity issues found.

Low Severity Issues

1. Out of gas

Issue:

- The function includeInReward() uses the loop to find and remove addresses from the _excluded list. Function will be aborted with OUT_OF_GAS exception if there will be a long excluded addresses list.

```
function includeInReward(address account⬆) external onlyOwner() {
    require(_isExcluded[account⬆], "Account is already excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account⬆) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account⬆] = 0;
            _isExcluded[account⬆] = false;
            _excluded.pop();
            break;
        }
    }
}
```

- The function _getCurrentSupply also uses the loop for evaluating total supply. It also could be aborted with OUT_OF_GAS exception if there will be a long excluded addresses list.

```
function _getCurrentSupply() private view returns (uint256, uint256) {
    uint256 rSupply = _rTotal;
    uint256 tSupply = _tTotal;
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (
            _rOwned[_excluded[i]] > rSupply ||
            _tOwned[_excluded[i]] > tSupply
        ) return (_rTotal, _tTotal);
        rSupply = rSupply.sub(_rOwned[_excluded[i]]);
        tSupply = tSupply.sub(_tOwned[_excluded[i]]);
    }
    if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
    return (rSupply, tSupply);
}
```

Recommendation:

Check that the excluded array length is not too big.

Owner privileges (In the period when the owner is not renounced)

- Owner can change the tax, charity and liquidity fee.

```
ftrace | funcSig
function setCharityFeePercent(uint256 fee⬆) external onlyOwner() {
    require(fee⬆ <= _maxCharityFee, 'Tax exceeds maximum');
    _charityFee = fee⬆;
}

ftrace | funcSig
function setTaxFeePercent(uint256 taxFee⬆) external onlyOwner() {
    require(taxFee⬆ <= _maxTaxFee, 'Tax exceeds maximum');
    _taxFee = taxFee⬆;
}

ftrace | funcSig
function setLiquidityFeePercent(uint256 liquidityFee⬆) external onlyOwner() {
    require(liquidityFee⬆ <= _maxLiquidityFee, 'Tax exceeds maximum');
    _liquidityFee = liquidityFee⬆;
}
```

- Owner can change the maximum transaction amount.

```
function setMaxTxPercent(uint256 maxTxPercent⬆) external onlyOwner() {
    _maxTxAmount = _tTotal.mul(maxTxPercent⬆).div(
        10**2
    );
}
```

- Owner can exclude from the fee.

```
function excludeFromFee(address account⬆) public onlyOwner {
    _isExcludedFromFee[account⬆] = true;
}
```

- Owner can withdraw tokens from the contract.

```
function adminRescueTokens(address token⬆, uint256 amount⬆) external onlyOwner() {
    require(token⬆ != address(this) && token⬆ != address(pancakeRouter.WETH()), 'Cannot withdraw primary tokens');
    IERC20(token⬆).transfer(msg.sender, amount⬆);
}
```

- **Owner can manually swap balance to liquidity.**

```
function swapBalanceToLiquidity(uint256 amount⬆) external onlyOwner() {
    require(amount⬆ <= balanceOf(address(this)), 'Not enough tokens for swap');

    // split the contract balance into halves
    uint256 half = amount⬆.div(2);
    uint256 otherHalf = amount⬆.sub(half);

    // capture the contract's current BNB balance.
    // this is so that we can capture exactly the amount of BNB that the
    // swap creates, and not make the liquidity event include any BNB that
    // has been manually sent to the contract
    // Leftover BNB can always be swapped to token in another external method
    uint256 initialBalance = address(this).balance;

    // swap tokens for ETH
    swapTokensForEth(half); // <- this breaks the ETH -> HATE swap when swap+liquify is triggered

    // how much ETH did we just swap into?
    uint256 newBalance = address(this).balance.sub(initialBalance);

    // add liquidity to pancake
    addLiquidity(otherHalf, newBalance);

    emit SwapAndLiquify(half, newBalance, otherHalf);
}
```

- **Owner can swap wBNB and burn.**

```
function swapWBNBForTokensAndBurn(uint256 amountBNB⬆, uint256 minTokenAmountOut⬆) external onlyOwner() {
    address[] memory path = new address[](2);
    path[0] = pancakeRouter.WETH();
    path[1] = address(this);

    // make the swap
    pancakeRouter.swapExactTokensForTokensSupportingFeeOnTransferTokens(
        amountBNB⬆,
        minTokenAmountOut⬆,
        path,
        BURN_ADDRESS,
        block.timestamp
    );
}
```

- **Owner can swap BNB and burn.**

```
function swapBNBForTokensAndBurn(uint256 amountBNB⬆, uint256 minTokenAmountOut⬆) external onlyOwner() {
    address[] memory path = new address[](2);
    path[0] = pancakeRouter.WETH();
    path[1] = address(this);

    // make the swap
    pancakeRouter.swapExactETHForTokensSupportingFeeOnTransferTokens{value: amountBNB⬆}(
        minTokenAmountOut⬆,
        path,
        BURN_ADDRESS,
        block.timestamp
    );
}
```

Conclusion

Smart contracts contain low severity issues! Liquidity pair contract's security is not checked due to out of scope.

Liquidity locking details provided by the team:

- **45.80 % permanently burned**
  - **proof: http://shibby.me/SU9**
- **28.00 % sold in Presale**
- **17.64 % Locked liquidity until 2042**
- **8.00 % team funds are locked for 3-24 months**
  - **proof: https://github.com/shibbyfinance/shibby-tokenlock**
- **0.56 % DXSale fee**

TechRate note:

Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.