Techrate1 Techrate

TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

October, 2021

Audit Details

Audited project

VIP TOKEN

Deployer address

0xada19a693368Ffb2B1e1e6e22B1DaE36CE943229

Client contacts:

VIP TOKEN team

Blockchain

Binance Smart Chain

Project website:

https://viptoken.io/

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 VIP TOKEN to perform an audit of smart contracts:

https://bscscan.com/address/0x6759565574De509b7725ABb4680020704B3F404e#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

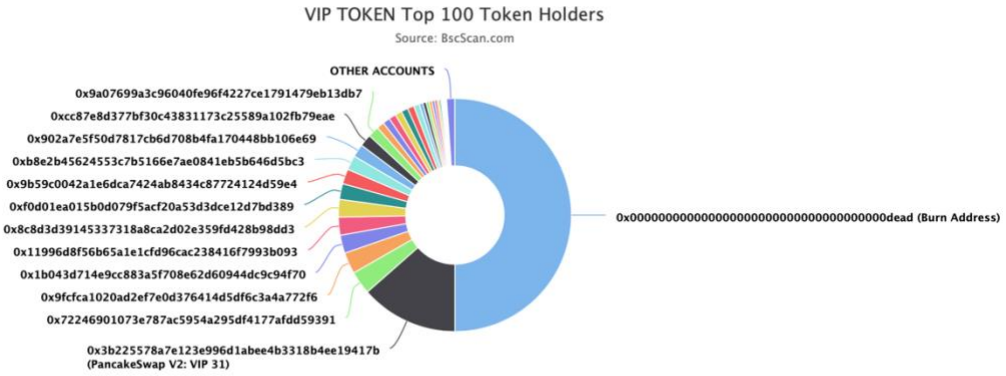
Token contract details for 30.10.2021

Contract name	VIP TOKEN
Contract address	0x6759565574De509b7725ABb4680020704B3F404e
Total supply	1,000,000,000,000,000
Token ticker	VIP
Decimals	9
Token holders	44
Transactions count	667
Top 100 holders dominance	98.86%
Liquidity fee	2
Tax fee	2
Burn fee	0
Marketing fee	4
Total fees	7,508,427,385,928.883357386
Uniswap V2 pair	0x3b225578a7E123e996d1AbEe4B3318B4ee19417b
Contract deployer address	0xada19a693368Ffb2B1e1e6e22B1DaE36CE943229
Contract's current owner address	0xada19a693368Ffb2B1e1e6e22B1DaE36CE943229

VIP TOKEN Token Distribution

The top 100 holders collectively own 98.86% (988,604,550,801,806.00 Tokens) of VIP TOKEN

Token Total Supply: 1,000,000,000,000,000.00 Token | Total Token Holders: 44

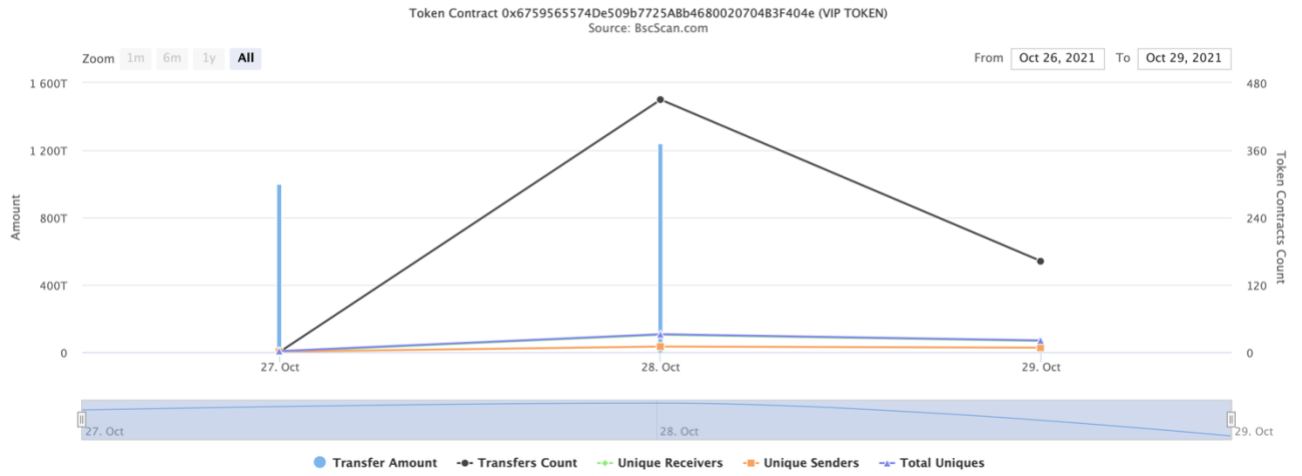


(A total of 988,604,550,801,806.00 tokens held by the top 100 accounts from the total supply of 1,000,000,000,000,000.00 token)

VIP TOKEN Contract Interaction Details

Time Series: Token Contract Overview

Wed 27, Oct 2021 - Fri 29, Oct 2021



VIP TOKEN Top 10 Token Holders

Rank	Address	Quantity	Percentage
1	Burn Address	500,000,000,000,000	50.0000%
2	PancakeSwap V2: VIP 31	135,316,828,944,672.357682365	13.5317%
3	0x72246901073e787ac5954a295df4177afdd59391	32,158,821,305,062.002056563	3.2159%
4	0x9fcfca1020ad2ef7e0d376414d5df6c3a4a772f6	29,568,046,560,165.417975116	2.9568%
5	0x1b043d714e9cc883a5f708e62d60944dc9c94f70	25,233,574,444,046.919410013	2.5234%
6	0x11996d8f56b65a1e1cfd96cac238416f7993b093	24,979,132,731,627.49372158	2.4979%
7	0x8c8d3d39145337318a8ca2d02e359fd428b98dd3	24,490,770,979,602.11943163	2.4491%
8	0xf0d01ea015b0d079f5acf20a53d3dce12d7bd389	21,747,781,884,164.837576226	2.1748%
9	0x9b59c0042a1e6dca7424ab8434c87724124d59e4	20,644,690,415,278.324559699	2.0645%
10	0xb8e2b45624553c7b5166e7ae0841eb5b646d5bc3	20,157,517,223,123.186879394	2.0158%

VIP TOKEN LP Token Holders

Rank	Address	Quantity	Percentage
1	0xada19a693368ffb2b1e1e6e22b1dae36ce943229	3,037.300206295070541862	99.9906%
2	0xb1b9b4bbe8a92d535f5df2368e7fd2ecfb3a1950	0.286632901260412837	0.0094%
3	0x0000000000000000000000000000000000000000	0.000000000000001	0.0000%

Contract functions details

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

+ [Lib] SafeMath
  - [Int] add
  - [Int] sub
  - [Int] sub
  - [Int] mul
  - [Int] div
  - [Int] div
  - [Int] mod
  - [Int] mod

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

+ [Lib] Address
  - [Int] isContract
  - [Int] sendValue #
  - [Int] functionCall #
  - [Int] functionCall #
  - [Int] functionCallWithValue #
  - [Int] functionCallWithValue #
  - [Prv] _functionCallWithValue #

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

+ [Int] IUniswapV2Factory
  - [Ext] feeTo
  - [Ext] feeToSetter
  - [Ext] getPair
  - [Ext] allPairs
  - [Ext] allPairsLength
  - [Ext] createPair #
  - [Ext] setFeeTo #

- [Ext] setFeeToSetter #

+ [Int] IUniswapV2Pair
  - [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] burn #
  - [Ext] swap #
  - [Ext] skim #
  - [Ext] sync #
  - [Ext] initialize #

+ [Int] IUniswapV2Router01
  - [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] IUniswapV2Router02 (IUniswapV2Router01)
  - [Ext] removeLiquidityETHSupportingFeeOnTransferTokens #
  - [Ext] removeLiquidityETHWithPermitSupportingFeeOnTransferTokens #
  - [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #

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

+ VIPTOKEN (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] deliver #
- [Pub] reflectionFromToken
- [Pub] tokenFromReflection
- [Pub] excludeFromReward #
  - modifiers: onlyOwner
- [Ext] includeInReward #
  - modifiers: onlyOwner
- [Prv] _transferBothExcluded #
- [Ext] <Fallback> ($)
- [Prv] _reflectFee #
- [Prv] _getValues
- [Prv] _getTValues
- [Prv] _getRValues
- [Prv] _getRate
- [Prv] _getCurrentSupply
- [Prv] _takeLiquidity #
- [Prv] calculateTaxFee
- [Prv] calculateLiquidityFee
- [Prv] removeAllFee #
- [Prv] restoreAllFee #
- [Pub] isExcludedFromFee
- [Prv] _approve #
- [Prv] _transfer #
- [Prv] swapAndLiquify #
  - modifiers: lockTheSwap
- [Prv] swapTokensForEth #
- [Prv] addLiquidity #
- [Prv] _tokenTransfer #
- [Prv] _transferStandard #
- [Prv] _transferToExcluded #
- [Prv] _transferFromExcluded #
- [Pub] excludeFromFee #
  - modifiers: onlyOwner
- [Pub] includeInFee #
  - modifiers: onlyOwner
- [Ext] disableAllFees #
  - modifiers: onlyOwner

- [Ext] enableAllFees #
  - modifiers: onlyOwner
- [Ext] setMarketingWallet #
  - modifiers: onlyOwner
- [Ext] setTaxFeePercent #
  - modifiers: onlyOwner
- [Ext] setLiquidityFeePercent #
  - modifiers: onlyOwner
- [Ext] setMarketingFeePercent #
  - modifiers: onlyOwner
- [Ext] setBurnFeePercent #
  - modifiers: onlyOwner
- [Ext] setMinSell #
  - modifiers: onlyOwner
- [Pub] setRouterAddress #
  - modifiers: onlyOwner
- [Pub] setSwapAndLiquifyEnabled #
  - modifiers: onlyOwner

($) = 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.	Low issues
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.

2. Wrong transfer

Issue:

- The function `_tokenTransfer()` uses _transferStandard to send burn and marketing amounts without any checking addresses to be excluded from reward. If them would be, this is the high issue.

```
    _transferStandard(sender, address(0), burnAmt);
    uint256 amountToSend = marketingAmount/4;
    _transferStandard(sender, marketingWallet1, amountToSend);
    _transferStandard(sender, marketingWallet2, amountToSend);
    _transferStandard(sender, marketingWallet3, amountToSend);
    _transferStandard(sender, marketingWallet4, amountToSend);
```

Recommendation:
Check addresses to be excluded from reward and use proper functions to send amounts.

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

- Owner can lock and unlock. By the way, using these functions the owner could retake privileges even after the ownership was renounced.

```
/* Locks the contract for owner for the amount of time provided */
function lock(uint256 time) public virtual onlyOwner {
    _previousOwner = _owner;
    _owner = address(0);
    _lockTime = block.timestamp + time;
    emit OwnershipTransferred(_owner, address(0));
}

/* Unlocks the contract for owner when _lockTime is exceeds */
function unlock() public virtual {
    require(_previousOwner == msg.sender, "You don't have permission to unlock");
    require(block.timestamp > _lockTime , "Contract is locked until 7 days");
    emit OwnershipTransferred(_owner, _previousOwner);
    _owner = _previousOwner;
}
```

- Owner can include in and exclude from fees.

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

function includeInFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = false;
}
```

- Owner can include in and exclude from rewards.

```
function excludeFromReward(address account) public onlyOwner() {
    require(account != 0x10ED43C718714eb63d5aA57B78B54704E256024E, 'We can not exclude Pancake router.');
    require(!_isExcluded[account], "Account is already excluded");
    if(_rOwned[account] > 0) {
        _tOwned[account] = tokenFromReflection(_rOwned[account]);
    }
    _isExcluded[account] = true;
    _excluded.push(account);
}

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;
        }
    }
}
```

- Owner can disable fees.

```
function disableAllFees() external onlyOwner() {
    _taxFee = 0;
    _previousTaxFee = _taxFee;
    _liquidityFee = 0;
    _previousLiquidityFee = _liquidityFee;
    _burnFee = 0;
    _previousBurnFee = _taxFee;
    _marketingFee = 0;
    _previousMarketingFee = _marketingFee;
    inSwapAndLiquify = false;
    emit SwapAndLiquifyEnabledUpdated(false);
}
```

- Owner can enable fees.

```
function enableAllFees() external onlyOwner() {
    _taxFee = 2;
    _previousTaxFee = _taxFee;
    _liquidityFee = 2;
    _previousLiquidityFee = _liquidityFee;
    _burnFee = 0;
    _previousBurnFee = _taxFee;
    _marketingFee = 4;
    _previousMarketingFee = _marketingFee;
    inSwapAndLiquify = true;
    emit SwapAndLiquifyEnabledUpdated(true);
}
```

- Owner can change marketing wallet addresses.

```
function setMarketingWallet(address newWallet1,address newWallet2,address newWallet3,address newWallet4) external onlyOwner() {
    marketingWallet1 = newWallet1;
    marketingWallet2 = newWallet2;
    marketingWallet3 = newWallet3;
    marketingWallet4 = newWallet4;
}
```

- Owner can change liquidity, marketing and burn fees.

```
function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {
    _liquidityFee = liquidityFee;
}

function setMarketingFeePercent(uint256 marketingFee) external onlyOwner() {
    _marketingFee = marketingFee;
}

function setBurnFeePercent(uint256 burnFee) external onlyOwner() {
    _burnFee = burnFee;
}
```

- Owner can change numTokensSellToAddToLiquidity value.

```
function setMinSell(uint256 amount) external onlyOwner() {
    numTokensSellToAddToLiquidity = amount;
}
```

- Owner can change UniswapV2Router and UniswapV2Pair.

```
function setRouterAddress(address newRouter) public onlyOwner() {
    IUniswapV2Router02 _newPancakeRouter = IUniswapV2Router02(newRouter);
    uniswapV2Pair = IUniswapV2Factory(_newPancakeRouter.factory()).createPair(address(this), _newPancakeRouter.WETH());
    uniswapV2Router = _newPancakeRouter;
}
```

- Owner can enable / disable swap and liquify.

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

Conclusion

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

Liquidity locking details provided by the team:
https://dxsale.app/app/v3/dxlplocksearch?id=0&add=0x6759565574De509b7725ABb4680020704B3F404e&type=lplock0&chain=BSC

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.