Techrate1 Techrate

TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

November, 2021

Audit Details

Audited project

Doge Unchained

Deployer address

0x826f7cab2dfe1cc70beb879b946e4340dc5d3d28

Client contacts:

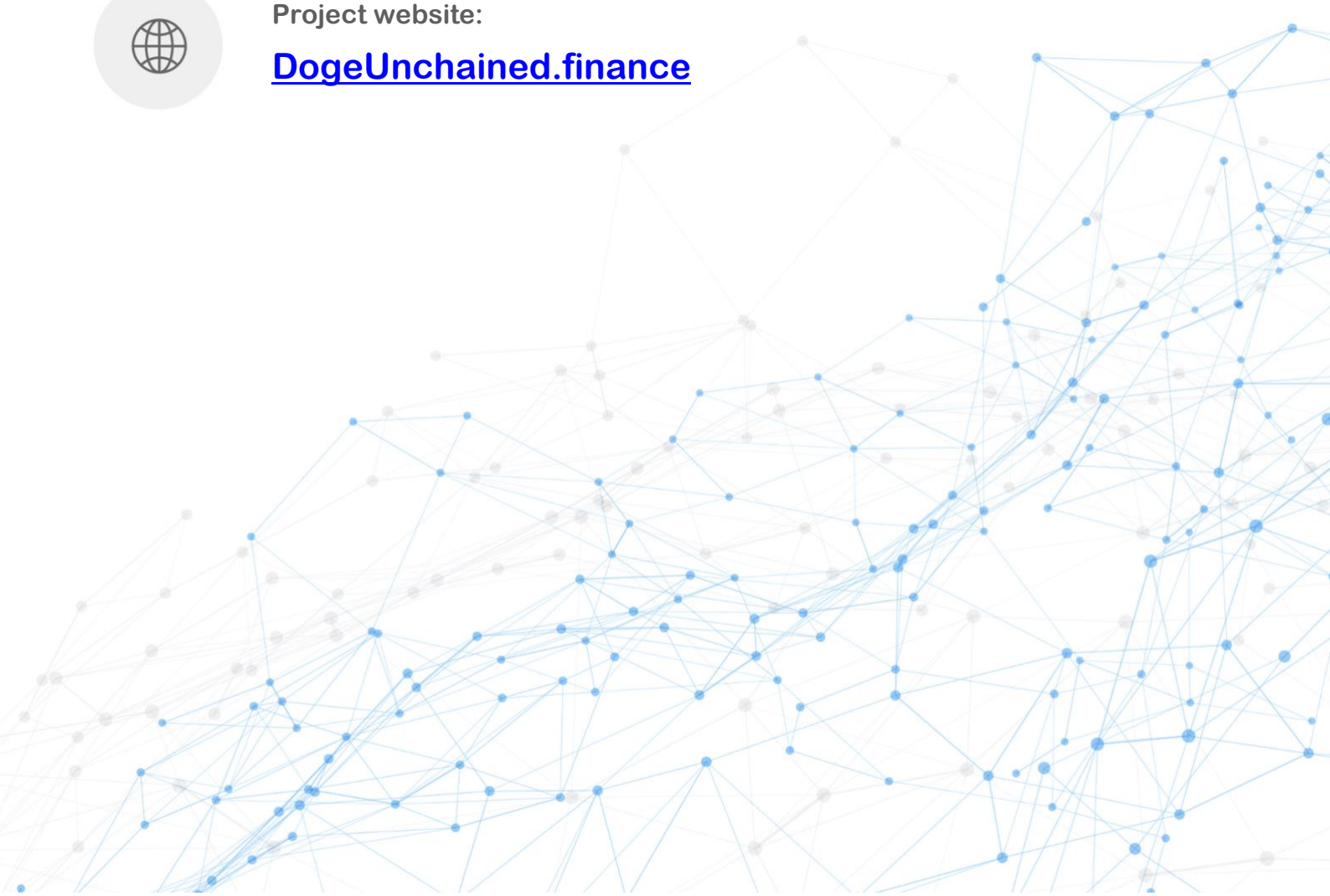
Doge Unchained team

Blockchain

Binance Smart Chain

Project website:

DogeUnchained.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 Doge Unchained to perform an audit of smart contracts:

https://bscscan.com/address/0x726f7BFa5f88dcAB97aCEB60d499e0CFf3BDc458#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

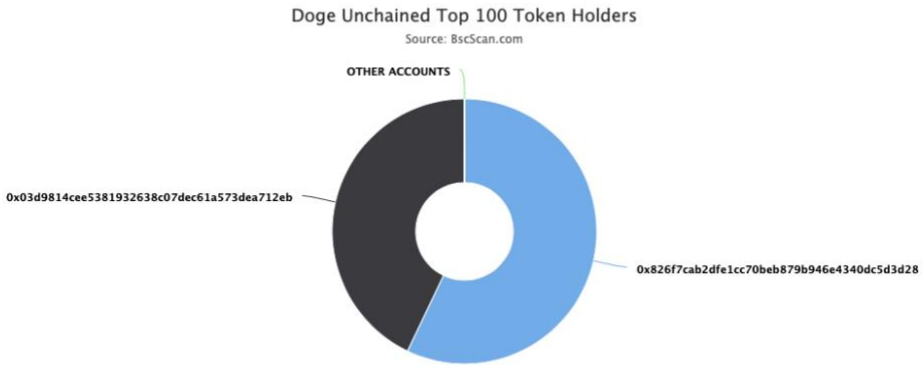
Token contract details for 02.11.2021

Contract name	Doge Unchained
Contract address	0x726f7BFa5f88dcAB97aCEB60d499e0CFf3BDc458
Total supply	100,000,000,000
Token ticker	DUC
Decimals	9
Token holders	2
Transactions count	1
Top 100 holders dominance	100.00%
Liquidity fee	2
Tax fee	2
Total fees	0
Uniswap V2 pair	0x62b79e4f51d57848bf8f006510436537c178abf5
Contract deployer address	0x826f7cab2dfe1cc70beb879b946e4340dc5d3d28
Contract's current owner address	0x826f7cab2dfe1cc70beb879b946e4340dc5d3d28

Doge Unchained Token Distribution

The top 100 holders collectively own 100.00% (100,000,000,000.00 Tokens) of Doge Unchained

Token Total Supply: 100,000,000,000.00 Token | Total Token Holders: 2



(A total of 100,000,000,000.00 tokens held by the top 100 accounts from the total supply of 100,000,000,000.00 token)

Doge Unchained Contract Interaction Details

Time Series: Token Contract Overview

Thu 28, Oct 2021 - Thu 28, Oct 2021

Doge Unchained Top 10 Token Holders

Rank	Address	Quantity (Token)	Percentage
1	0x826f7cab2dfe1cc70beb879b946e4340dc5d3d28	57,094,840,000	57.0948%
2	0x03d9814cee5381932638c07dec61a573dea712eb	42,905,160,000	42.9052%

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

+ [Int] IForward

- [Ext] retrieve #

+ DogeUnchained (Context, IERC20, Ownable)
  - [Pub] <Constructor> #
  - [Pub] updateAutoAddLiquidityRouter #
    - modifiers: onlyOwner
  - [Pub] updateAutoAddLiquidityAddie #
    - modifiers: onlyOwner
  - [Pub] createAutoAddUniswapV2pair #
    - modifiers: onlyOwner
  - [Pub] setAutomatedMarketMakerPair #
    - modifiers: onlyOwner
  - [Prv] _setAutomatedMarketMakerPair #
  - [Pub] distribute #
    - modifiers: onlyOwner
  - [Pub] name
  - [Pub] symbol
  - [Pub] decimals
  - [Pub] totalSupply
  - [Pub] balanceOf
  - [Pub] setWallet #
  - [Pub] contains
  - [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
  - [Pub] excludeFromFee #
    - modifiers: onlyOwner
  - [Pub] includeInFee #
    - modifiers: onlyOwner
  - [Ext] setSellFeePercents #
    - modifiers: onlyOwner
  - [Ext] openTrading #
    - modifiers: onlyOwner
  - [Ext] setTaxFeePercent #
    - modifiers: onlyOwner
  - [Ext] setLiquidityFeePercent #
    - modifiers: onlyOwner
  - [Ext] setCharityFeePercent #
    - modifiers: onlyOwner
  - [Ext] setCommunityFeePercent #
    - modifiers: onlyOwner
  - [Ext] setBurnFeePercent #
    - modifiers: onlyOwner

- [Ext] setCommunityAddress #
  - modifiers: onlyOwner
- [Ext] setCharityAddress #
  - modifiers: onlyOwner
- [Ext] setLiquidityTaxAddress #
  - modifiers: onlyOwner
- [Ext] setSwapForwardAddress #
  - modifiers: onlyOwner
- [Ext] setAutoSellAddress #
  - modifiers: onlyOwner
- [Ext] setMaxTxPercent #
  - modifiers: onlyOwner
- [Ext] setSwapAndLiquifyEnabled #
  - modifiers: onlyOwner
- [Ext] setNumTokensSellToAddToLiquidity #
  - modifiers: onlyOwner
- [Ext] setAutoSellForCommunity #
  - modifiers: onlyOwner
- [Ext] setAutoSellForCharity #
  - modifiers: onlyOwner
- [Ext] setBurnToBurnAddress #
  - modifiers: onlyOwner
- [Ext] <Fallback> ($)
- [Prv] _reflectFee #
- [Prv] _getValues
- [Prv] _getTValues
- [Prv] _getRValues
- [Prv] _getRate
- [Prv] _getCurrentSupply
- [Prv] _takeFee #
- [Prv] calculateTaxFee
- [Prv] calculateBurnFee
- [Prv] calculateFeeToTake
- [Prv] removeAllFee #
- [Prv] applySellFee #
- [Prv] restoreAllFee #
- [Pub] isExcludedFromFee
- [Prv] _approve #
- [Prv] _transfer #
- [Prv] _getFeeAmounts
- [Prv] swapAndLiquify #
  - modifiers: lockTheSwap
- [Pub] triggerSwapAndLiquify #
  - modifiers: onlyOwner
- [Prv] swapTokensForTokens #
- [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.	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⬆], "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.

- The function distribute() uses the loop to transfer token amounts from array to addresses from list. It also could be aborted with OUT_OF_GAS exception if there will be a long addresses list.

```
function distribute(address[] memory _addresses⬆, uint256[] memory _balances⬆)
    public
    onlyOwner
{
    uint16 i;
    uint256 count = _addresses⬆.length;

    if (count > 100) {
        count = 100;
    }

    for (i = 0; i < count; i++) {
        //_addresses.length
        _tokenTransfer(_msgSender(), _addresses⬆[i], _balances⬆[i], false);
    }
}
```

Recommendation:

Check that the excluded array length is not too big.

2. Array inconsistency

- The function distribute() do not compare _addresses length with _balances length to make sure there is no inconsistency.

```
function distribute(address[] memory _addresses⬆, uint256[] memory _balances⬆)
    public
    onlyOwner
{
    uint16 i;
    uint256 count = _addresses⬆.length;

    if (count > 100) {
        count = 100;
    }

    for (i = 0; i < count; i++) {
        //_addresses.length
        _tokenTransfer(_msgSender(), _addresses⬆[i], _balances⬆[i], false);
    }
}
```

Recommendation:

Check that the arrays' length are equal.

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

- Owner can change Uniswap router.
- Owner can change _liquidityPairAddie and recreate uniswapV2Pair.
- Owner can include addresses in automatedMarketMakerPairs array.
- Owner can distribute to multiple addresses.
- Owner can include and exclude from rewards.
- Owner can exempt addresses from paying the tax.
- Owner can change fee percents and fee addresses.
- Owner can enable trading.
- Owner can change swap forward and auto sell addresses.
- Owner can change maximum transaction amount.
- Owner can change numTokensSellToAddToLiquidity.
- Owner can enable/disable autoSellForCommunity, burnToBurnAddress and autoSellForCharity.
- Owner can manually swap and liquify.

Conclusion

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

Liquidity locking details NOT provided by the team.

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.