



TechRate
AUDIT COMPANY

Smart Contract Security Audit

Audit Details



Audited project

GloryDoge



Deployer address

0xcCDE780E78AcE46442524FC22A1FDdf8E1e13482



Client contacts:

GloryDoge team



Blockchain

Binance Smart Chain



Project website:

<https://glorydogecoin.com>

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 GloryDoge to perform an audit of smart contracts:

<https://bscscan.com/address/0xcc5667333f5e997ac9f0c26d41b7dda65b2b675a#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

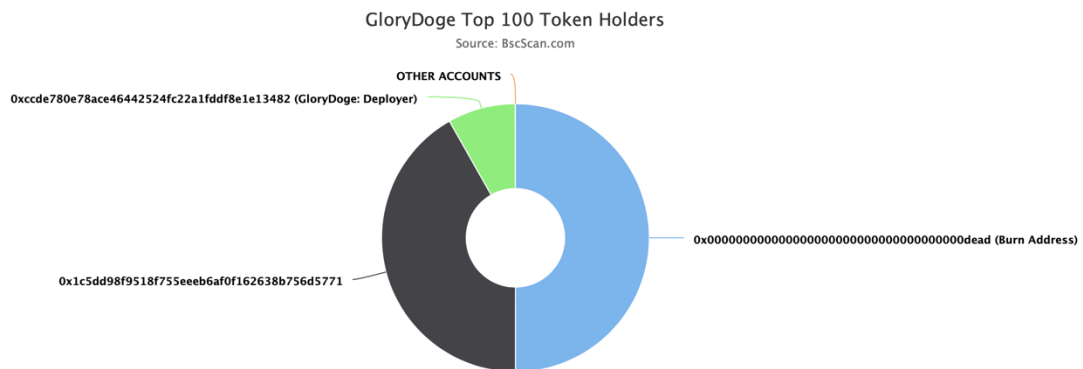
Token contract details for 19.10.2021

Contract name	GloryDoge
Contract address	0xCC5667333F5e997aC9F0C26d41b7ddA65b2b675a
Total supply	1,000,000,000,000,000
Token ticker	GLORYD
Decimals	18
Token holders	3
Transactions count	2
Top 100 holders dominance	100.00%
Marketing tax	2
Rewards tax	4
Team tax	4
Total taxes	10
Contract deployer address	0xcCDE780E78AcE46442524FC22A1FDdf8E1e13482
Contract's current owner address	0xcCDE780E78AcE46442524FC22A1FDdf8E1e13482

GloryDoge Token Distribution

💡 The top 100 holders collectively own 100.00% (1,000,000,000,000.00 Tokens) of GloryDoge

💡 Token Total Supply: 1,000,000,000,000.00 Token | Total Token Holders: 3

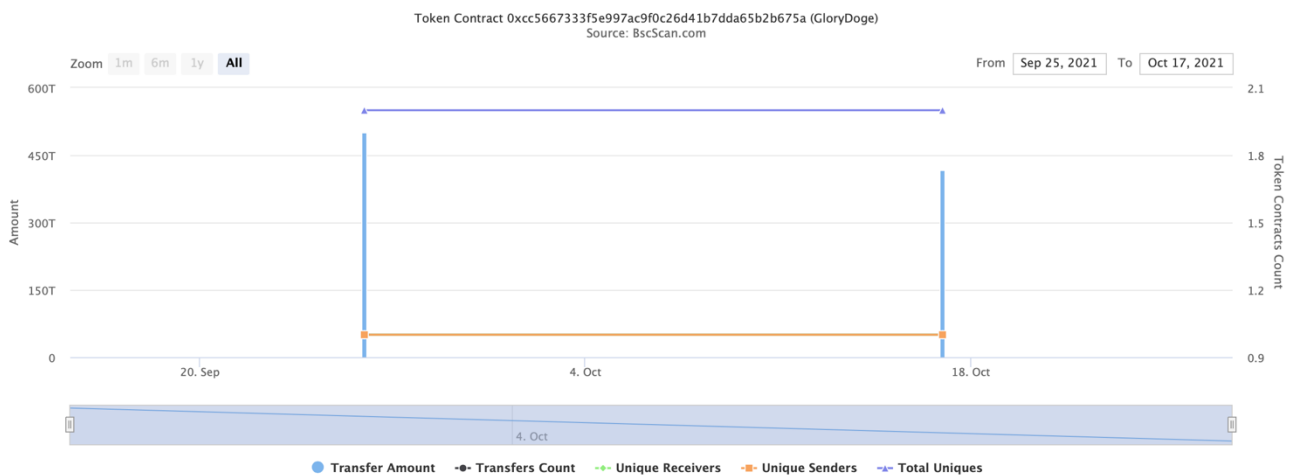


(A total of 1,000,000,000,000,000.00 tokens held by the top 100 accounts from the total supply of 1,000,000,000,000,000.00 token)


GloryDoge Contract Interaction Details

Time Series: Token Contract Overview

Sun 26, Sept 2021 - Sun 17, Oct 2021



GloryDoge Top 10 Token Holders

Rank	Address	Quantity	Percentage
1	Burn Address	500,000,000,000,000	<u>50.0000%</u>
2	 0x1c5dd98f9518f755eeeb6af0f162638b756d5771	417,970,000,000,000	<u>41.7970%</u>
3	GloryDoge: Deployer	82,030,000,000,000	<u>8.2030%</u>



Contract functions details

+ Context

- [Int] _msgSender
- [Int] _msgData

+ Ownable (Context)

- [Pub] <Constructor> #
- [Pub] owner
- [Pub] renounceOwnership #
 - modifiers: onlyOwner
- [Pub] transferOwnership #
 - modifiers: onlyOwner
- [Prv] _setOwner #

+ [Int] IERC20

- [Ext] totalSupply
- [Ext] balanceOf
- [Ext] transfer #
- [Ext] allowance
- [Ext] approve #
- [Ext] transferFrom #

+ [Int] IERC20Metadata (IERC20)

- [Ext] name
- [Ext] symbol
- [Ext] decimals

+ [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 #
- + [Int] IUniswapV2Factory
 - [Ext] feeTo
 - [Ext] feeToSetter
 - [Ext] getPair
 - [Ext] allPairs
 - [Ext] allPairsLength
 - [Ext] createPair #
 - [Ext] setFeeTo #
 - [Ext] setFeeToSetter #
- + GloryDoge (Context, IERC20, IERC20Metadata, Ownable)
 - [Pub] <Constructor> #
 - [Pub] name
 - [Pub] symbol
 - [Pub] decimals
 - [Pub] totalSupply
 - [Pub] rewardsTax
 - [Pub] marketingTax
 - [Pub] teamTax
 - [Pub] totalTaxes
 - [Pub] taxSwapThreshold
 - [Pub] excludedFromRewards
 - [Pub] excludedFromTaxes
 - [Pub] distributionAddress
 - [Pub] pair
 - [Pub] balanceOf
 - [Pub] transfer #
 - [Pub] allowance
 - [Pub] approve #
 - [Pub] transferFrom #
 - [Pub] increaseAllowance #
 - [Pub] decreaseAllowance #
 - [Pub] setMarketingTaxAddress #
 - modifiers: onlyOwner
 - [Pub] setTeamTaxAddress #
 - modifiers: onlyOwner
 - [Pub] setTaxes #
 - modifiers: onlyOwner
 - [Ext] distribute #
 - [Pub] addDistributor #
 - modifiers: onlyOwner
 - [Pub] removeDistributor #
 - modifiers: onlyOwner
 - [Pub] includeInRewards #
 - modifiers: onlyOwner
 - [Pub] excludeFromRewards #
 - modifiers: onlyOwner
 - [Pub] includeInTaxes #
 - modifiers: onlyOwner
 - [Pub] excludeFromTaxes #
 - modifiers: onlyOwner
 - [Pub] enableTransferLimit #

- modifiers: onlyOwner
- [Pub] disableTransferLimit #
 - modifiers: onlyOwner
- [Pub] addPair #
 - modifiers: onlyOwner
- [Pub] setTaxSwapThreshold #
 - modifiers: onlyOwner
- [Int] _transfer #
- [Int] _approve #
- [Int] _swapTransfer #
- [Int] _swapTaxes #
 - modifiers: swapLock
- [Int] _takeTaxes #
- [Int] _getRewardsRate
- [Int] _getRewardPointsFromBalance
- [Int] _getBalanceFromRewardPoints

(\$) = 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.	Passed
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 issue
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. Remove distributor not work

Issue:

- The function `removeDistributor()` set value true for account.

```
function removeDistributor(address account) public onlyOwner {
    require(_distributionAddress[account], "Address is not a distributor");
    _distributionAddress[account] = true;

    emit RemoveDistributor(account);
}
```

Recommendation:

Set value false when removing distributor.

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

- Owner can change marketing tax address.

```
function setMarketingTaxAddress(address marketingTaxAddress)
    public
    onlyOwner
{
    address _oldMarketingTaxAddress = _marketingTaxAddress;

    includeInRewards(_oldMarketingTaxAddress);
    includeInTaxes(_oldMarketingTaxAddress);

    excludeFromRewards(marketingTaxAddress);
    excludeFromTaxes(marketingTaxAddress);

    _marketingTaxAddress = marketingTaxAddress;

    emit MarketingTaxAddressChange(
        _oldMarketingTaxAddress,
        _marketingTaxAddress
    );
}
```

- Owner can change team tax address.

```
function setTeamTaxAddress(address teamTaxAddress) public onlyOwner {
    address _oldTeamTaxAddress = _teamTaxAddress;

    includeInRewards(_oldTeamTaxAddress);
    includeInTaxes(_oldTeamTaxAddress);

    excludeFromRewards(teamTaxAddress);
    excludeFromTaxes(teamTaxAddress);

    _teamTaxAddress = teamTaxAddress;

    emit MarketingTaxAddressChange(_oldTeamTaxAddress, _teamTaxAddress);
}
```

- Owner can change rewards, marketing and team taxes.

```
function setTaxes(
    uint256 rewardsTax_,
    uint256 marketingTax_,
    uint256 teamTax_
) public onlyOwner {
    require(
        rewardsTax_ + marketingTax_ + teamTax_ <= 10,
        "Total taxes should never be more than 10%."
    );

    _rewardsTax = rewardsTax_;
    _marketingTax = marketingTax_;
    _teamTax = teamTax_;

    emit TaxesChange(_rewardsTax, _marketingTax, _teamTax);
}
```

- Owner can add / remove distributor address.

```
function addDistributor(address account) public onlyOwner {
    require(
        !_distributionAddress[account],
        "Address is already a distributor"
    );
    _distributionAddress[account] = true;

    emit AddDistributor(account);
}

/**
 * Removes an address from distributors.
 * @param account The address to be removed from distributors.
 */
function removeDistributor(address account) public onlyOwner {
    require(_distributionAddress[account], "Address is not a distributor");
    _distributionAddress[account] = false;

    emit RemoveDistributor(account);
}
```

- Owner can include in and exclude from taxes.

```
function includeInTaxes(address account) public onlyOwner {
    if (!_excludedFromTaxes[account]) return;
    _excludedFromTaxes[account] = false;

    emit IncludeInTaxes(account);
}

/**
 * Excludes an address from taxes.
 * @param account The address to be excluded from taxes.
 */
function excludeFromTaxes(address account) public onlyOwner {
    if (_excludedFromTaxes[account]) return;
    _excludedFromTaxes[account] = true;

    emit ExcludeFromTaxes(account);
}
```

- Owner and distributors can distribute an amount from the sender's wallet to all holders.

```
function distribute(uint256 amount) external {
    require(
        _distributionAddress[_msgSender()] || owner() == _msgSender(),
        "Only owner and distribution addresses can call this function."
    );

    uint256 balance = balanceOf(_msgSender());

    require(balance >= amount, "Distribution amount exceeds balance");

    uint256 rate = _getRewardsRate();

    uint256 balanceRewardPoints = _getRewardPointsFromBalance(
        balance,
        rate
    );
    uint256 amountRewardPoints = _getRewardPointsFromBalance(amount, rate);

    if (_excludedFromRewards[_msgSender()]) {
        _supplyOwned[_msgSender()] -= amount;
        _excludedSupply -= amount;
        _excludedRewardPoints -= amountRewardPoints;
    } else
        _rewardPointsOwned[_msgSender()] =
            balanceRewardPoints -
            amountRewardPoints;

    _rewardPoints -= amountRewardPoints;

    emit Distribution(_msgSender(), amount);
}
```

- Owner can enable / disable transfer limit.

```
function enableTransferLimit() public onlyOwner {
    require(
        _maxTransferLimit == _totalSupply,
        "Transfer limit already enabled"
    );
    _maxTransferLimit = _totalSupply / 500;

    emit EnableTransferLimit(_maxTransferLimit);
}

/**
 * Disables the 0.2% transfer amount limit.
 */
function disableTransferLimit() public onlyOwner {
    require(
        _maxTransferLimit != _totalSupply,
        "Transfer limit already disabled"
    );
    _maxTransferLimit = _totalSupply;

    emit DisableTransferLimit(_maxTransferLimit);
}
```

- Owner can include in and exclude from rewards.

```
function includeInRewards(address account) public onlyOwner {
    if (!_excludedFromRewards[account]) return;

    uint256 rate = _getRewardsRate();
    uint256 accountSupplyBalance = balanceOf(account);
    uint256 accountRewardPoints = _getRewardPointsFromBalance(
        accountSupplyBalance,
        rate
    );

    _rewardPointsOwned[account] = accountRewardPoints;
    _excludedSupply -= accountSupplyBalance;
    _excludedRewardPoints -= accountRewardPoints;

    _excludedFromRewards[account] = false;

    emit IncludeInRewards(account);
}

function excludeFromRewards(address account) public onlyOwner {
    if (_excludedFromRewards[account]) return;

    uint256 rate = _getRewardsRate();
    uint256 accountSupplyBalance = balanceOf(account);
    uint256 accountRewardPoints = _getRewardPointsFromBalance(
        accountSupplyBalance,
        rate
    );

    _supplyOwned[account] = accountSupplyBalance;
    _excludedSupply += accountSupplyBalance;
    _excludedRewardPoints += accountRewardPoints;

    _excludedFromRewards[account] = true;

    emit ExcludeFromRewards(account);
}
```


- Owner can add pair address.

```
function addPair(address pairAddress) public onlyOwner {  
    _pair[pairAddress] = true;  
    excludeFromRewards(pairAddress);  
  
    emit AddPair(pairAddress);  
}
```

- Owner can change tax swap threshold.

```
function setTaxSwapThreshold(uint256 threshold) public onlyOwner {  
    _taxSwapThreshold = threshold;  
  
    emit TaxSwapThresholdChange(_taxSwapThreshold);  
}
```

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://www.pinksale.finance/#/launchpad/0x1c5dd98f9518F755eEeB6af0F162638b756D5771?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.



[Techrate1](#)



[Techrate](#)



[Techrate audits](#)