Techrate1 Techrate

TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

November, 2021

Audit Details

Audited project

Holiday Token

Deployer address

0x4610e98049f028393b8e5f40ba9a580b4891e38b

Client contacts:

Holiday Token team

Blockchain

Binance Smart Chain

Project website:

https://holidaytoken.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 Holiday Token to perform an audit of smart contracts:

https://bscscan.com/address/0x7B8656C95944f1d6e2E6dEaDeDD0392A4138D8dd#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 22.11.2021

Contract name	Holiday Token
Contract address	0x7B8656C95944f1d6e2E6dEaDeDD0392A4138D8dd
Total supply	1,000,000,000,000,000
Token ticker	HOL
Decimals	9
Token holders	2,722
Transactions count	11,305
Top 100 holders dominance	82.58%
Multiplied fee	1400
Autoliquidity fee receiver	0x4610e98049f028393b8e5f40ba9a580b4891e38b
Marketing fee receiver	0xeb32f32ccd81b7c4d2a1df789fadc741e88b7946
Pair	0x233355020c30359f4eeb64112551d9f55ca059f3
Contract deployer address	0x4610e98049f028393b8e5f40ba9a580b4891e38b
Contract's current owner address	0x4610e98049f028393b8e5f40ba9a580b4891e38b

Holiday Token Distribution

The top 100 holders collectively own 82.58% (825,831,055,553,802.00 Tokens) of Holiday Token

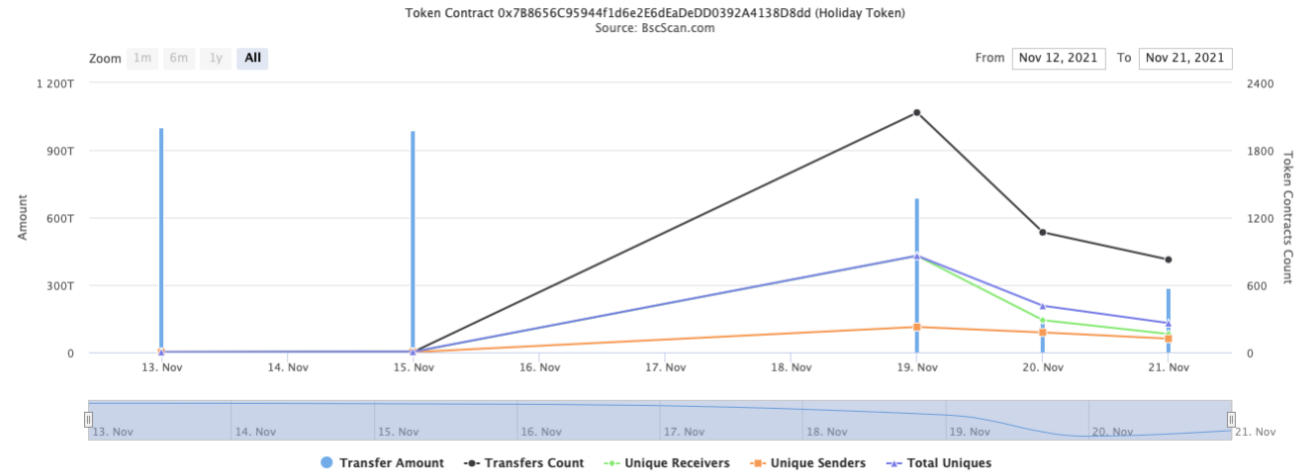
Token Total Supply: 1,000,000,000,000,000.00 Token | Total Token Holders: 2,722

(A total of 825,831,055,553,802.00 tokens held by the top 100 accounts from the total supply of 1,000,000,000,000,000.00 token)

Holiday Token Contract Interaction details

Time Series: Token Contract Overview

Sat 13, Nov 2021 - Sun 21, Nov 2021



Holiday Token Top 10 Token Holders

Rank	Address	Quantity (Token)	Percentage
1	Burn Address	397,000,000,000,000	39.7000%
2	PancakeSwap V2: HOL 11	100,851,154,086,471.139004607	10.0851%
3	0x7bd9cc54f131031d369d44d6931a30fac4074edf	83,094,000,000,000	8.3094%
4	UniCrypt: Token Vesting	65,140,000,000,000	6.5140%
5	0x1026fbe9c9666b1c192acbf726d70505df774bd2	12,137,214,120,000.000150733	1.2137%
6	0xf2b8e653ea93dcf26cce25d02e73906ed000f587	6,618,351,225,478.717760677	0.6618%
7	0x7547c9e3a87620489c80dc3047394c645639c271	5,695,976,970,004.119945596	0.5696%
8	0xefaab8e042ca8b45bc2a1c1f92e69110959631e2	5,588,259,885,035.523360164	0.5588%
9	0x4b04213c2774f77e60702880654206b116d00508	5,200,000,000,000	0.5200%
10	0xf8550f0a458ee374803ac3fe84ae7b717527397f	5,069,109,181,468.977880495	0.5069%

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

+ [Int] IBEP20
  - [Ext] totalSupply
  - [Ext] decimals
  - [Ext] symbol
  - [Ext] name
  - [Ext] getOwner
  - [Ext] balanceOf
  - [Ext] transfer #
  - [Ext] allowance
  - [Ext] approve #
  - [Ext] transferFrom #

+  Auth
  - [Pub] <Constructor> #
  - [Pub] authorize #
    - modifiers: onlyOwner
  - [Pub] unauthorize #
    - modifiers: onlyOwner
  - [Pub] isOwner
  - [Pub] isAuthorized
  - [Pub] transferOwnership #
    - modifiers: onlyOwner

+ [Int] IDEXFactory
  - [Ext] createPair #

+ [Int] IDEXRouter
  - [Ext] factory
  - [Ext] WETH
  - [Ext] addLiquidity #
  - [Ext] addLiquidityETH ($)
  - [Ext] swapExactTokensForTokensSupportingFeeOnTransferTokens #
  - [Ext] swapExactETHForTokensSupportingFeeOnTransferTokens ($)
  - [Ext] swapExactTokensForETHSupportingFeeOnTransferTokens #

+ [Int] IDividendDistributor
```

- [Ext] setDistributionCriteria #
- [Ext] setShare #
- [Ext] deposit ($)
- [Ext] process #

+ DividendDistributor (IDividendDistributor)
- [Pub] <Constructor> #
- [Ext] setDistributionCriteria #
  - modifiers: onlyToken
- [Ext] setShare #
  - modifiers: onlyToken
- [Ext] deposit ($)
  - modifiers: onlyToken
- [Ext] process #
  - modifiers: onlyToken
- [Int] shouldDistribute
- [Int] distributeDividend #
- [Ext] claimDividend #
- [Pub] getUnpaidEarnings
- [Int] getCumulativeDividends
- [Int] addShareholder #
- [Int] removeShareholder #

+ HolidayToken (IBEP20, Auth)
- [Pub] <Constructor> #
  - modifiers: Auth
- [Ext] <Fallback> ($)
- [Ext] totalSupply
- [Ext] decimals
- [Ext] symbol
- [Ext] name
- [Ext] getOwner
- [Pub] balanceOf
- [Ext] allowance
- [Pub] approve #
- [Ext] approveMax #
- [Ext] transfer #
- [Ext] transferFrom #
- [Int] _transferFrom #
- [Int] _basicTransfer #
- [Int] checkTxLimit
- [Int] shouldTakeFee
- [Pub] getTotalFee
- [Pub] getMultipliedFee
- [Int] takeFee #
- [Int] shouldSwapBack
- [Int] swapBack #
  - modifiers: swapping
- [Int] shouldAutoBuyback
- [Ext] triggerZeusBuyback #
  - modifiers: authorized
- [Ext] clearBuybackMultiplier #
  - modifiers: authorized
- [Int] triggerAutoBuyback #
- [Int] buyTokens #

  - modifiers: swapping
- [Ext] setAutoBuybackSettings #
  - modifiers: authorized
- [Ext] setBuybackMultiplierSettings #
  - modifiers: authorized
- [Int] launched
- [Pub] launch #
  - modifiers: authorized
- [Ext] setTxLimit #
  - modifiers: authorized
- [Ext] setIsDividendExempt #
  - modifiers: authorized
- [Ext] setIsFeeExempt #
  - modifiers: authorized
- [Ext] setIsTxLimitExempt #
  - modifiers: authorized
- [Ext] setFees #
  - modifiers: authorized
- [Ext] setFeeReceivers #
  - modifiers: authorized
- [Ext] setSwapBackSettings #
  - modifiers: authorized
- [Ext] setTargetLiquidity #
  - modifiers: authorized
- [Ext] setDistributionCriteria #
  - modifiers: authorized
- [Ext] setDistributorSettings #
  - modifiers: authorized
- [Pub] getCirculatingSupply
- [Pub] getLiquidityBacking
- [Pub] isOverLiquified

($) = 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

No low severity issues found.

Notes

- No transfer event emitted when basic transfer called.

```
function _basicTransfer(address sender, address recipient, uint256 amount) internal returns (bool) {
    _balances[sender] = _balances[sender].sub(amount, "Insufficient Balance");
    _balances[recipient] = _balances[recipient].add(amount);
      emit Transfer(sender, recipient, amount);
    return true;
}
```

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

- Owner can authorize / unauthorize addresses.

```
/**
 * Authorize address. Owner only
 */
function authorize(address adr) public onlyOwner {
    authorizations[adr] = true;
}

/**
 * Remove address' authorization. Owner only
 */
function unauthorize(address adr) public onlyOwner {
    authorizations[adr] = false;
}
```

- **Authorized addresses can call triggerZeusBuyback that's initiate buyback.**

```
function triggerZeusBuyback(uint256 amount, bool triggerBuybackMultiplier) external authorized {
    buyTokens(amount, DEAD);
    if(triggerBuybackMultiplier){
        buybackMultiplierTriggeredAt = block.timestamp;
        emit BuybackMultiplierActive(buybackMultiplierLength);
    }
}
```

- **Authorized addresses can clear buyback multiplier.**

```
function clearBuybackMultiplier() external authorized {
    buybackMultiplierTriggeredAt = 0;
}
```

- **Authorized addresses can change auto buyback settings.**

```
function setAutoBuybackSettings(bool _enabled, uint256 _cap, uint256 _amount, uint256 _period) external authorized {
    autoBuybackEnabled = _enabled;
    autoBuybackCap = _cap;
    autoBuybackAccumulator = 0;
    autoBuybackAmount = _amount;
    autoBuybackBlockPeriod = _period;
    autoBuybackBlockLast = block.number;
}
```

- **Authorized addresses can change buyback multiplier settings.**

```
function setBuybackMultiplierSettings(uint256 numerator, uint256 denominator, uint256 length) external authorized {
    require(numerator / denominator <= 2 && numerator > denominator);
    buybackMultiplierNumerator = numerator;
    buybackMultiplierDenominator = denominator;
    buybackMultiplierLength = length;
}
```

- **Authorized addresses can change the maximum transaction amount.**

```
function setTxLimit(uint256 amount) external authorized {
    require(amount >= _totalSupply / 1000);
    _maxTxAmount = amount;
}
```

- **Authorized addresses can include in and exclude from dividends.**

```
function setIsDividendExempt(address holder, bool exempt) external authorized {
    require(holder != address(this) && holder != pair);
    isDividendExempt[holder] = exempt;
    if(exempt){
        distributor.setShare(holder, 0);
    }else{
        distributor.setShare(holder, _balances[holder]);
    }
}
```

- **Authorized addresses can include in and exclude from fee.**

```
function setIsFeeExempt(address holder, bool exempt) external authorized {
    isFeeExempt[holder] = exempt;
}
```

- Authorized addresses can include in and exclude from transaction amount limit.

```
function setIsTxLimitExempt(address holder, bool exempt) external authorized {
    isTxLimitExempt[holder] = exempt;
}
```

- Authorized addresses can change fees.

```
function setFees(uint256 _liquidityFee, uint256 _buybackFee, uint256 _reflectionFee, uint256 _marketingFee, uint256 _feeDenominator) external authorized {
    liquidityFee = _liquidityFee;
    buybackFee = _buybackFee;
    reflectionFee = _reflectionFee;
    marketingFee = _marketingFee;
    totalFee = _liquidityFee.add(_buybackFee).add(_reflectionFee).add(_marketingFee);
    feeDenominator = _feeDenominator;
    require(totalFee < feeDenominator/4);
}
```

- Authorized addresses can change fee receivers.

```
function setFeeReceivers(address _autoLiquidityReceiver, address _marketingFeeReceiver) external authorized {
    autoLiquidityReceiver = _autoLiquidityReceiver;
    marketingFeeReceiver = _marketingFeeReceiver;
}
```

- Authorized addresses can change swap threshold and disable/enable swap.

```
function setSwapBackSettings(bool _enabled, uint256 _amount) external authorized {
    swapEnabled = _enabled;
    swapThreshold = _amount;
}
```

- Authorized addresses can change target liquidity values.

```
function setTargetLiquidity(uint256 _target, uint256 _denominator) external authorized {
    targetLiquidity = _target;
    targetLiquidityDenominator = _denominator;
}
```

- Authorized addresses can change distribution criteria.

```
function setDistributionCriteria(uint256 _minPeriod, uint256 _minDistribution) external authorized {
    distributor.setDistributionCriteria(_minPeriod, _minDistribution);
}
```

- Authorized addresses can change distribution GAS.

```
function setDistributorSettings(uint256 gas) external authorized {
    require(gas < 750000);
    distributorGas = gas;
}
```

Conclusion

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

Liquidity locking details are 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.