TechRate
Blockchain solutions and consulting

Smart Contract Security Audit

Audit details:

Audited project:	GarudaSwap
Deployer address	0x6d21333444fccef0aceab0bb9d7f25d5c2fa7afd
Blockchain:	Binance Smart Chain
Project website:	https://garudaswap.finance

May, 2021
TechRate

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 GarudaSwap to perform an audit of smart contracts:

- *https://bscscan.com/address/0x743d5428eE14d95AE8f737D9D9832246344Ced59#code*
- *https://bscscan.com/address/0xf6afB97aC5eAfAd60d3ad19c2f85E0Bd6b7eAcCf#code*
- *https://bscscan.com/address/0x854086dC841e1bfae50Cb615bF41f55BF432a90b#code*
- *https://bscscan.com/address/0xa3DC5D19BAAE61dDb8245D3eB3bb655480F327d1#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 06.05.2021.

Contract name:	GarudaSwap
Compiler version:	v0.6.12+commit.27d51765
Contract address:	0x854086dC841e1bfae50Cb615bF41f55BF432a90b
Total supply:	200000000000000000000000
Token ticker:	GARUDA
Decimals:	18
Token holders:	1182
Transactions count:	18631
Maximum transfer tax rate:	1000
Burn rate:	12
Garuda swap pair:	0x1ab609b5d930c64be12a05d040a9822b0da84a8f
Garuda swap router:	0x10ed43c718714eb63d5aa57b78b54704e256024e
Max transfer amount:	0
Max transfer amount rate:	0
Min amount to liquify:	500_000_000_000_000_000_000
Operator	0x6d21333444fccef0aceab0bb9d7f25d5c2fa7afd
Transfer tax rate:	800
Contract deployer address:	0x6d21333444fccef0aceab0bb9d7f25d5c2fa7afd
Contract’s current owner address:	0xf6afb97ac5eafad60d3ad19c2f85e0bd6b7eaccf

Masterchef contract details for 06.05.2021.

Contract name:	MasterChef
Compiler version:	v0.6.12+commit.27d51765
Contract address:	0xf6afB97aC5eAfAd60d3ad19c2f85E0Bd6b7eAcCf
Dev address:	0x78211a692cd7f3be44343c5b0a7fcb0e7691d170
Fee address:	0xe7391f0b3e1c7811bcd30b7d4d48f2f21594aa63
Garuda contract address:	0x854086dc841e1bfae50cb615bf41f55bf432a90b
Garuda per block:	100_000_000_000_000_000_000
Contract owner address:	0x743d5428ee14d95ae8f737d9d9832246344ced59
Pool length:	15
Start block:	7241540
Total alloc point:	10400
Garuda referral:	0xa3dc5d19baae61ddb8245d3eb3bb655480f327d1
Bonus multiplier:	1
Referral commission rate:	300
Total locked up rewards:	0

MasterChef functions outline

+ [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 #

+  ReentrancyGuard
  - [Int] <Constructor> #

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

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

+ [Int] IGarudaReferral
  - [Ext] recordReferral #
  - [Ext] recordReferralCommission #
  - [Ext] getReferrer

+ [Lib] Address
```

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

+ [Lib] SafeBEP20
  - [Int] safeTransfer #
  - [Int] safeTransferFrom #
  - [Int] safeApprove #
  - [Int] safeIncreaseAllowance #
  - [Int] safeDecreaseAllowance #
  - [Prv] _callOptionalReturn #

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

+ [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

```
+  BEP20 (Context, IBEP20, Ownable)
  - [Pub] <Constructor> #
  - [Ext] getOwner
  - [Pub] name
  - [Pub] decimals
  - [Pub] symbol
  - [Pub] totalSupply
  - [Pub] balanceOf
  - [Pub] transfer #
  - [Pub] allowance
  - [Pub] approve #
  - [Pub] transferFrom #
  - [Pub] increaseAllowance #
  - [Pub] decreaseAllowance #
  - [Pub] mint #
    - modifiers: onlyOwner
  - [Int] _transfer #
  - [Int] _mint #
  - [Int] _burn #
  - [Int] _approve #
  - [Int] _burnFrom #

+  GarudaToken (BEP20)
  - [Pub] <Constructor> #
    - modifiers: BEP20
  - [Pub] mint #
    - modifiers: onlyOwner
  - [Int] _transfer #
    - modifiers: antiWhale
  - [Prv] swapAndLiquify #
    - modifiers: lockTheSwap,transferTaxFree
  - [Prv] swapTokensForEth #
  - [Prv] addLiquidity #
  - [Pub] maxTransferAmount
  - [Pub] isExcludedFromAntiWhale
  - [Ext] <Fallback> ($)
  - [Pub] updateTransferTaxRate #
    - modifiers: onlyOperator
  - [Pub] updateBurnRate #
    - modifiers: onlyOperator
  - [Pub] updateMaxTransferAmountRate #
    - modifiers: onlyOperator
  - [Pub] updateMinAmountToLiquify #
    - modifiers: onlyOperator
  - [Pub] setExcludedFromAntiWhale #
    - modifiers: onlyOperator
  - [Pub] updateSwapAndLiquifyEnabled #
```

    - modifiers: onlyOperator
  - [Pub] updateGarudaSwapRouter #
    - modifiers: onlyOperator
  - [Pub] operator
  - [Pub] transferOperator #
    - modifiers: onlyOperator
  - [Ext] delegates
  - [Ext] delegate #
  - [Ext] delegateBySig #
  - [Ext] getCurrentVotes
  - [Ext] getPriorVotes
  - [Int] _delegate #
  - [Int] _moveDelegates #
  - [Int] _writeCheckpoint #
  - [Int] safe32
  - [Int] getChainId

+ MasterChef (Ownable, ReentrancyGuard)
  - [Pub] <Constructor> #
  - [Ext] poolLength
  - [Pub] add #
    - modifiers: onlyOwner
  - [Pub] set #
    - modifiers: onlyOwner
  - [Pub] getMultiplier
  - [Ext] pendingGaruda
  - [Pub] canHarvest
  - [Pub] massUpdatePools #
  - [Pub] updatePool #
  - [Pub] deposit # - modifiers: nonReentrant
  - [Pub] withdraw # - modifiers: nonReentrant
  - [Pub] emergencyWithdraw #
    - modifiers: nonReentrant
  - [Int] payOrLockupPendingGaruda #
  - [Int] safeGarudaTransfer #
  - [Pub] setDevAddress #
  - [Pub] setFeeAddress #
  - [Pub] updateEmissionRate # - modifiers: onlyOwner
  - [Pub] setGarudaReferral #
    - modifiers: onlyOwner
  - [Pub] setReferralCommissionRate #
    - modifiers: onlyOwner
  - [Int] payReferralCommission #

($) = 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

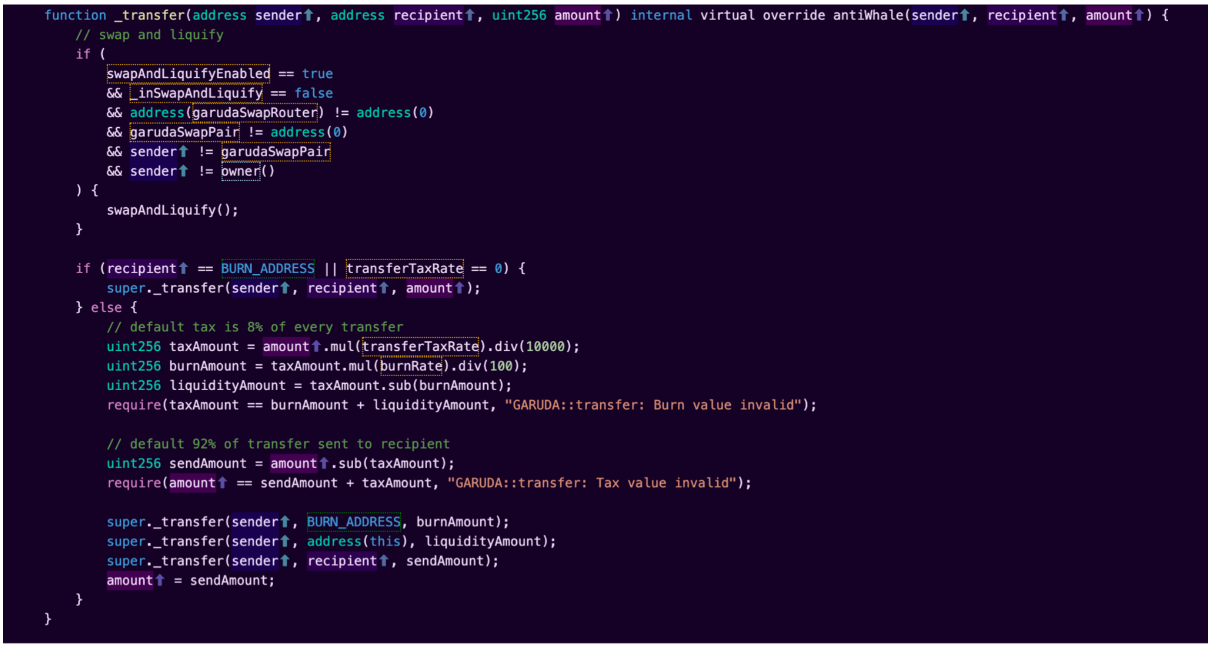
No high severity issues found.

Medium Severity Issues

1. Wrong burning

Issue:

There is sending tokens to the dead address in overridden _transfer functions, instead of burning them in the token contract.



```
function _transfer(address sender, address recipient, uint256 amount) internal virtual override antiWhale(sender, recipient, amount) {
    // swap and liquify
    if (
        swapAndLiquifyEnabled == true
        && _inSwapAndLiquify == false
        && address(garudaSwapRouter) != address(0)
        && garudaSwapPair != address(0)
        && sender != garudaSwapPair
        && sender != owner()
    ) {
        swapAndLiquify();
    }

    if (recipient == BURN_ADDRESS || transferTaxRate == 0) {
        super._transfer(sender, recipient, amount);
    } else {
        // default tax is 8% of every transfer
        uint256 taxAmount = amount.mul(transferTaxRate).div(10000);
        uint256 burnAmount = taxAmount.mul(burnRate).div(100);
        uint256 liquidityAmount = taxAmount.sub(burnAmount);
        require(taxAmount == burnAmount + liquidityAmount, "GARUDA::transfer: Burn value invalid");

        // default 92% of transfer sent to recipient
        uint256 sendAmount = amount.sub(taxAmount);
        require(amount == sendAmount + taxAmount, "GARUDA::transfer: Tax value invalid");

        super._transfer(sender, BURN_ADDRESS, burnAmount);
        super._transfer(sender, address(this), liquidityAmount);
        super._transfer(sender, recipient, sendAmount);
        amount = sendAmount;
    }
}
```

Recommendation:

There should be a burn instead of sending to the dead address.

Low Severity Issues

1. Block gas limit

Issue:

The updateEmissionRate function can fail due to block gas limit if the pool size is too big.

2. add function issue

Issue:

If some LP token is added to the contract twice using function add, then the total amount of reward garudaReward in function updatePool will be incorrect.

Recommendation:

Add the mapping from address to bool and check that same address will not be added twice.

Owner privileges

- ❑ Owner can drain tokens that are sent to the referral contract which is useful for withdrawing tokens sent by mistake to the contract.
- ❑ Owner can change the operator of the referral contract.
- ❑ Operator can change the transfer tax rate to maximum 10%.
- ❑ Operator can change the burn rate to maximum 10%.
- ❑ Operator can change the max transfer amount rate.
- ❑ Operator can change the min amount to liquify.
- ❑ Operator can change the router and pair contract addresses, which could be not audited contract

Conclusion

Smart contracts contain medium severity issues and owner privileges!

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.