TechRate

AUDIT COMPANY

Smart Contract Security Audit

TechRate

July, 2021

Audit Details

Audited project

ShibaKenzo

Deployer address

0xcC8f0B47Cf0E2d5428eDaE93d968eA927De626a2

Client contacts:

ShibaKenzo team

Blockchain

Binance Smart Chain

Project website:

https://shibakenzo.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 ShibaKenzo to perform an audit of smart contracts:
https://bscscan.com/address/0xAF1fC9766a15A50dBA7Cc6feF52d3e3189a117D4#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 09.07.2021

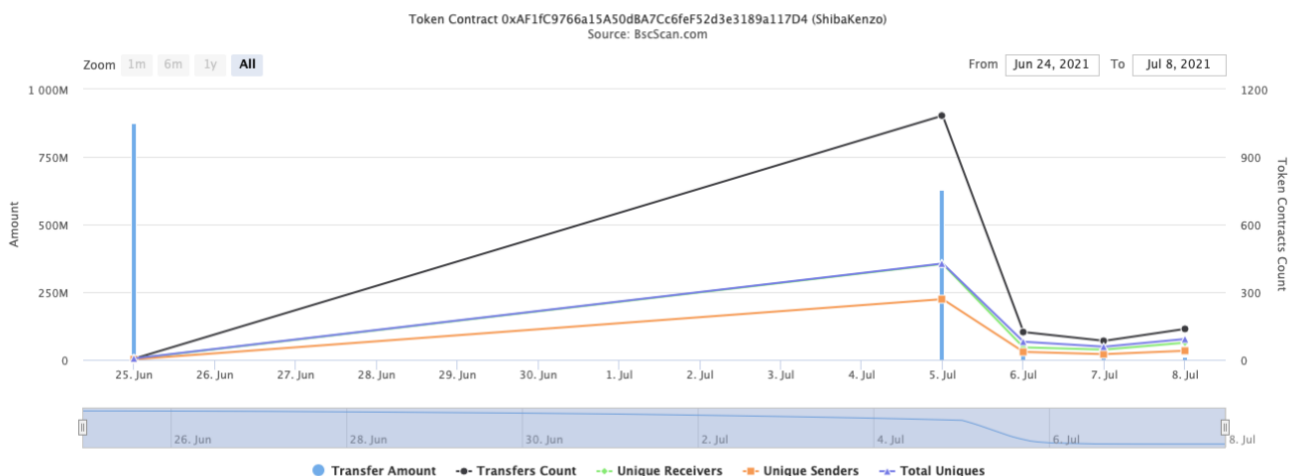|	
Contract name	**ShibaKenzo**
Contract address	**0xAF1fC9766a15A50dBA7Cc6feF52d3e3189a117D4**
Total supply	**500,000,000**
Token ticker	**KENZO**
Decimals	**18**
Token holders	**403**
Transactions count	**1,445**
Top 100 holders dominance	**98.37%**
Liquidity fee	**5**
Tax fee	**5**
Total fees	**23006793202065931332957988**
Uniswap V2 pair	**0xe5ebfd10cd29cf07c02ce19314f7e87446e91d57**
Contract deployer address	**0xcC8f0B47Cf0E2d5428eDaE93d968eA927De626a2**
Contract's current owner address	**0x081993f34b2ffcc3e09956d3aec3f33cea75a8a3**

ShibaKenzo Token Distribution

Token Total Supply: 500,000,000.00 Token | Total Token Holders: 403

ShibaKenzo Top 100 Token Holders
Source: BscScan.com

OTHER ACCOUNTS

0x02db190f9c0d02f9dd7a8209fe829f679c5d4c2f
0x0b0d3f85d8870f96a3724bf1852d8fe863790832
0xe5ebfd10cd29cf07c02ce19314f7e87446e91d57
(PancakeSwap V2: KENZO 43)
0x8dd5d35a96019f5d7309340a243035df6f1bb603

0x54889bd37eafa8daa62aea7f0e8d3f2017fd7d8b

0x000000000000000000000000000000000000dead (Burn Address)

(A total of 491,835,333.84 tokens held by the top 100 accounts from the total supply of 500,000,000.00 token)

ShibaKenzo Contract Interaction Details

Time Series: Token Contract Overview   Fri 25, Jun 2021 - Thu 8, Jul 2021

Token Contract 0xAF1fC9766a15A50dBA7Cc6feF52d3e3189a117D4 (ShibaKenzo)
Source: BscScan.com

Zoom 1m 6m 1y All   From Jun 24, 2021 To Jul 8, 2021

● Transfer Amount -●- Transfers Count -●- Unique Receivers -●- Unique Senders -▲- Total Uniques

ShibaKenzo Top 10 Token Holders

Rank	Address	Quantity (Token)	Percentage
1	Burn Address	349,903,728.838745462554143732	69.9807%
2	0x54889bd37eafa8daa62aea7f0e8d3f2017fd7d8b	50,000,000	10.0000%
3	0x8dd5d35a96019f5d7309340a243035df6f1bb603	50,000,000	10.0000%
4	📄 PancakeSwap V2: KENZO 43	13,506,966.422848245819867116	2.7014%
5	0x0b0d3f85d8870f96a3724bf1852d8fe863790832	11,080,657.207760394824814954	2.2161%
6	0x02db190f9c0d02f9dd7a8209fe829f679c5d4c2f	1,423,278.771694374221254674	0.2847%
7	0x3c3e9858e034e655d3824fd1500e2f4da38aa1ff	1,109,041.732556158439088868	0.2218%
8	0xf2da7f3cd4c57c9da50c7f3b1e3e84439fc74878	1,095,585.343028970625765071	0.2191%
9	0xe70be69b76cd7e1f2c96587a077f62fd38da57fa	1,078,214.625948315820525037	0.2156%
10	0x2f6339cd62b82211f160d47cb38629f57e01548e	1,070,962.972194532423096303	0.2142%

ShibaKenzo LP Token Holders

Rank	Address	Quantity	Percentage
1	Burn Address	27,328.591807093350723909	87.7231%
2	📄 0x8655e5c4d701186d16765d1cdcef6d5287e4679a	3,741.657386773941384583	12.0105%
3	0x07d80ae6f36a5e08dca74ce884a24d39db9934ed	82.993528934636158063	0.2664%
4	📄 0x0000000000000000000000000000000000000000	0.000000000000001	0.0000%

Contract functions details

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

+ [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

+ [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]** getUnlockTime
 - **[Pub]** getTime
 - **[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 **#**

+ ShibaKenzo (Context, IERC20, Ownable)
- **[Pub]** <Constructor> **#**
- **[Pub]** name
- **[Pub]** symbol
- **[Pub]** decimals
- **[Pub]** totalSupply
- **[Pub]** balanceOf
- **[Pub]** transfer **#**
- **[Pub]** allowance
- **[Pub]** approve **#**
- **[Pub]** blacklistAddress **#**
  - modifiers: onlyOwner
- **[Pub]** removeBlacklist **#**
  - modifiers: onlyOwner
- **[Pub]** transferFrom **#**
- **[Pub]** increaseAllowance **#**
- **[Pub]** decreaseAllowance **#**
- **[Pub]** isExcludedFromReward
- **[Pub]** totalFees
- **[Pub]** minimumTokensBeforeSwapAmount
- **[Pub]** buyBackUpperLimitAmount
- **[Pub]** deliver **#**
- **[Pub]** reflectionFromToken
- **[Pub]** tokenFromReflection
- **[Pub]** excludeFromReward **#**
  - modifiers: onlyOwner
- **[Ext]** includeInReward **#**
  - modifiers: onlyOwner
- **[Prv]** _approve **#**
- **[Prv]** _transfer **#**
- **[Prv]** swapTokens **#**
  - modifiers: lockTheSwap
- **[Prv]** buyBackTokens **#**
  - modifiers: lockTheSwap
- **[Prv]** swapTokensForEth **#**
- **[Prv]** swapETHForTokens **#**
- **[Prv]** addLiquidity **#**
- **[Prv]** _tokenTransfer **#**
- **[Prv]** _transferStandard **#**
- **[Prv]** _transferToExcluded **#**
- **[Prv]** _transferFromExcluded **#**
- **[Prv]** _transferBothExcluded **#**
- **[Prv]** _reflectFee **#**
- **[Prv]** _getValues
- **[Prv]** _getTValues
- **[Prv]** _getRValues
- **[Prv]** _getRate
- **[Prv]** _getCurrentSupply
- **[Prv]** _takeLiquidity **#**
- **[Prv]** calculateTaxFee
- **[Prv]** calculateLiquidityFee

- **[Prv]** removeAllFee **#**
- **[Prv]** restoreAllFee **#**
- **[Pub]** isExcludedFromFee
- **[Pub]** excludeFromFee **#**
  - modifiers: onlyOwner
- **[Pub]** includeInFee **#**
  - modifiers: onlyOwner
- **[Ext]** setTaxFeePercent **#**
  - modifiers: onlyOwner
- **[Ext]** setLiquidityFeePercent **#**
  - modifiers: onlyOwner
- **[Ext]** setMaxTxAmount **#**
  - modifiers: onlyOwner
- **[Ext]** setMarketingDivisor **#**
  - modifiers: onlyOwner
- **[Ext]** setNumTokensSellToAddToLiquidity **#**
  - modifiers: onlyOwner
- **[Ext]** setBuybackUpperLimit **#**
  - modifiers: onlyOwner
- **[Ext]** setMarketingAddress **#**
  - modifiers: onlyOwner
- **[Pub]** setSwapAndLiquifyEnabled **#**
  - modifiers: onlyOwner
- **[Pub]** setBuyBackEnabled **#**
  - modifiers: onlyOwner
- **[Ext]** prepareForPreSale **#**
  - modifiers: onlyOwner
- **[Ext]** afterPreSale **#**
  - modifiers: onlyOwner
- **[Prv]** transferToAddressETH **#**
- **[Ext]** **<Fallback>** **($)**

($) = 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

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.

```solidity
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.

```solidity
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.

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

- **Owner can change tax and liquidity fees.**

```
ftrace | funcSig
function setTaxFeePercent(uint256 taxFee↑) external onlyOwner() {
    _taxFee = taxFee↑;
}

ftrace | funcSig
function setLiquidityFeePercent(uint256 liquidityFee↑) external onlyOwner() {
    _liquidityFee = liquidityFee↑;
}
```

- **Owner can change maximum transaction amount.**

```
ftrace | funcSig
function setMaxTxAmount(uint256 maxTxAmount↑) external onlyOwner() {
    _maxTxAmount = maxTxAmount↑;
}
```

- **Owner can exclude from the fee.**

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

- **Owner can change marketingDivisor.**

```
ftrace | funcSig
function setMarketingDivisor(uint256 divisor↑) external onlyOwner() {
    marketingDivisor = divisor↑;
}
```

- **Owner can change minimum number of tokens to add to liquidity.**

```
ftrace | funcSig
function setNumTokensSellToAddToLiquidity(uint256 _minimumTokensBeforeSwap↑) external onlyOwner() {
    minimumTokensBeforeSwap = _minimumTokensBeforeSwap↑;
}
```

- **Owner can change buyBackUpperLimit.**

```
ftrace | funcSig
function setBuybackUpperLimit(uint256 buyBackLimit⬆) external onlyOwner() {
    buyBackUpperLimit = buyBackLimit⬆ * 10**18;
}
```

- **Owner can change marketing address.**

```
ftrace | funcSig
function setMarketingAddress(address _marketingAddress⬆) external onlyOwner() {
    marketingAddress = payable(_marketingAddress⬆);
}
```

- **Owner can enable and disable buyBack.**

```
ftrace | funcSig
function setBuyBackEnabled(bool _enabled⬆) public onlyOwner {
    buyBackEnabled = _enabled⬆;
    emit BuyBackEnabledUpdated(_enabled⬆);
}
```

- **Owner can enable before and after presale modes.**

```
ftrace | funcSig
function prepareForPreSale() external onlyOwner {
    setSwapAndLiquifyEnabled(false);
    _taxFee = 0;
    _liquidityFee = 0;
    _maxTxAmount = 1000000000 * 10**6 * 10**9;
}
```

```
ftrace | funcSig
function afterPreSale() external onlyOwner {
    setSwapAndLiquifyEnabled(true);
    _taxFee = 2;
    _liquidityFee = 9;
    _maxTxAmount = 3000000 * 10**6 * 10**9;
}
```

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

```solidity
//Locks the contract for owner for the amount of time provided
function lock(uint256 time) public virtual onlyOwner {
    _previousOwner = _owner;
    _owner = address(0);
    _lockTime = now + 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(now > _lockTime , "Contract is locked until 7 days");
    emit OwnershipTransferred(_owner, _previousOwner);
    _owner = _previousOwner;
}
```

- **Owner can blacklist and unblacklist addresses.**

```solidity
ftrace | funcSig
function blacklistAddress(address account) public onlyOwner(){
require(!_isBlacklisted[account], "Account is already blacklisted");
require(account != owner(), "Owner cannot be blacklisted");
_isBlacklisted[account] = true;
_blacklisted.push(account);
}

ftrace | funcSig
function removeBlacklist(address account) public onlyOwner() {
    require(_isBlacklisted[account], "Account is not blacklisted");
    for (uint256 i = 0; i < _blacklisted.length; i++) {
        if (_blacklisted[i] == account) {
            _blacklisted[i] = _blacklisted[_blacklisted.length - 1];
            _isBlacklisted[account] = false;
            _blacklisted.pop();
            break;
        }
    }
}
```

Conclusion

Smart contracts contain low severity issues! Liquidity pair contract's security is not checked due to out of scope. One third of the liquidity goes to marketing address.

Liquidity locking details provided by the team:
https://dxsale.app/app/v2_9/dxlockview?id=0&add=0x081993F34b2 FFcc3e09956D3AeC3f33CEa75a8a3&type=lplock&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.