SMART CONTRACT AUDIT REPORT

For

TY

Prepared By: SFI Team

Prepared for: Tikhe

Prepared on: 24/10/2021

Table of Content

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of 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 the team on the basis of what it says or doesn't say, or how team produced it, and it is important for you to conduct your own independent investigations before making any decisions. team go into more detail on this in the below disclaimer below – please make sure to read it in full.

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 Saferico and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (SaferICO) owe no duty of care towards you or any other person, nor does Saferico 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 Saferico 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, Saferico hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Saferico, 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.

- # Overview of the audit

The project has 1 file. It contains approx 1614 lines of Solidity code. Most of the functions and state variables are well commented on using the Nat spec documentation, but that does not create any vulnerability.

- # Attacks made to the contract

In order to check for the security of the contract, we tested several attacks in order to make sure that the contract is secure and follows best practices automatically.

1. Unit tests passing.

2. Compilator warnings;

3. Race Conditions. Reentrancy. Cross-function Race Conditions. Pitfalls in Race Condition solutions;

4. Possible delays in data delivery;

5. Transaction-Ordering Dependence (front running);

6. Timestamp Dependence;

7. Integer Overflow and Underflow;

8. DoS with (unexpected) Revert;

9. DoS with Block Gas Limit;

10. Call Depth Attack. Not relevant in modern ethereum network

11. Methods execution permissions;

12. Oracles calls;

13. Economy model. It’s important to forecast scenarios when a user is provided with additional economic motivation or faced with limitations. If application logic is based on incorrect economy model, the application will not function correctly and participants will incur financial losses. This type of issue is most often found in bonus rewards systems.
14. The impact of the exchange rate on the logic;
15. Private user data leaks.

- # Good things in smart contract

Compiler version is static: -
=> In this file, you have put “pragma solidity 0.8.4;” which is a good way to define the compiler version.

```
pragma solidity 0.8.4;
```

- **SafeMath library: -**
  - TY is using SafeMath library it is a good thing. This protects TY from underflow and overflow attacks.

```
library SafeMath {
    function add(uint256 a, uint256 b) internal pure returns (uint256)
{
        uint256 c = a + b;
        require(c >= a, "SafeMath: addition overflow");

        return c;
    }

    function sub(uint256 a, uint256 b) internal pure returns (uint256)
{
        return sub(a, b, "SafeMath: subtraction overflow");
    }
```

- **Good required condition in functions: -**
  - Here you are checking transferOwnership function

```
function transferOwnership(address newOwner) public
  virtual onlyOwner {
        require(
            newOwner != address(0),
            "Ownable: new owner is the zero address"
        );
        emit OwnershipTransferred(_owner, newOwner);
        _owner = newOwner;
      }
```

  - Here you are adding a lock and unlock function with the possibility of knowing the time of the lock

```
function getUnlockTime() public view returns (uint256) {
        return _lockTime;
    }

    function getTime() public view returns (uint256) {
        return block.timestamp;
```

```
    }

    function lock(uint256 time) public virtual onlyOwner {
        _previousOwner = _owner;
        _owner = address(0);
        _lockTime = block.timestamp + time;
        emit OwnershipTransferred(_owner, address(0));
    }

    function unlock() public virtual {
        require(
            _previousOwner == msg.sender,
            "You don't have permission to unlock"
        );
        require(block.timestamp > _lockTime, "Contract is
locked until 7 days");
        emit OwnershipTransferred(_owner, _previousOwner);
        _owner = _previousOwner;
    }
```

- Here you are Using interface IER20

```
interface IERC20 {
  function totalSupply() external view returns
(uint256);
  function balanceOf(address account) external
view returns (uint256);
  function transfer(address recipient, uint256
amount)
      external
      returns (bool);
  function allowance(address owner, address
spender)
      external
      view
      returns (uint256);
  function approve(address spender, uint256
amount) external returns (bool);
  function transferFrom(
      address sender,
      address recipient,
      uint256 amount
  ) external returns (bool);
 event Transfer(address indexed from, address
  indexed to, uint256 value);
  event Approval(
      address indexed owner,
      address indexed spender,
      uint256 value
  );
 }
```

- **Critical vulnerabilities found in the contract**

There not Critical severity vulnerabilities found

- **High vulnerabilities found in the contract**

There not High severity vulnerabilities found

- **Medium vulnerabilities found in the contract**

There not Medium severity vulnerabilities found

- **Low severity vulnerabilities found**

#Check-effects-interaction:

```
function _functionCallWithValue(
       address target,
       bytes memory data,
       uint256 weiValue,
       string memory errorMessage
   ) private returns (bytes memory) {
       require(isContract(target), "Address: call to non-contract");

       (bool success, bytes memory returndata) = target.call{value: weiVal
           data
       );
       if (success) {
           return returndata;
       } else {
           if (returndata.length > 0) {
               assembly {
                   let returndata_size := mload(returndata)
                   revert(add(32, returndata), returndata_size)
               }
           } else {
               revert(errorMessage);
           }}}}
```

In detail

Potential violation of Checks-Effects-Interaction pattern in Address.functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability.
For more reading:
https://docs.soliditylang.org/en/v0.8.0/security-considerations.html#re-entrancy

#Inline assembly

```
assembly {
            codehash := extcodehash(account)
        }                 }
```

In detail

The Contract uses inline assembly, this is only advised in rare cases. Additionally static analysis modules do not parse inline Assembly, this can lead to wrong analysis results.
For more reading:
https://docs.soliditylang.org/en/v0.8.0/assembly.html

#Transaction origin:

```
require(tx.origin == msg.sender, "sorry humans only");
```

In detail
Use of tx.origin: "tx.origin" is useful only in very exceptional cases.
If you use it for authentication, you usually want to replace it by "msg.sender", because otherwise any contract you call can act on your behalf.
For more reading:
https://docs.soliditylang.org/en/v0.8.4/security-considerations.html#tx-origin

- **Notes**

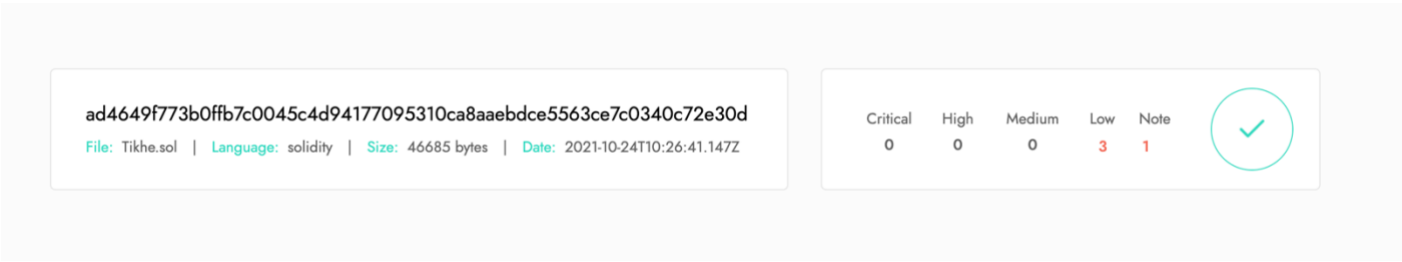
#Call

```
(       (bool success, ) = recipient.call{value: amount}("");
```

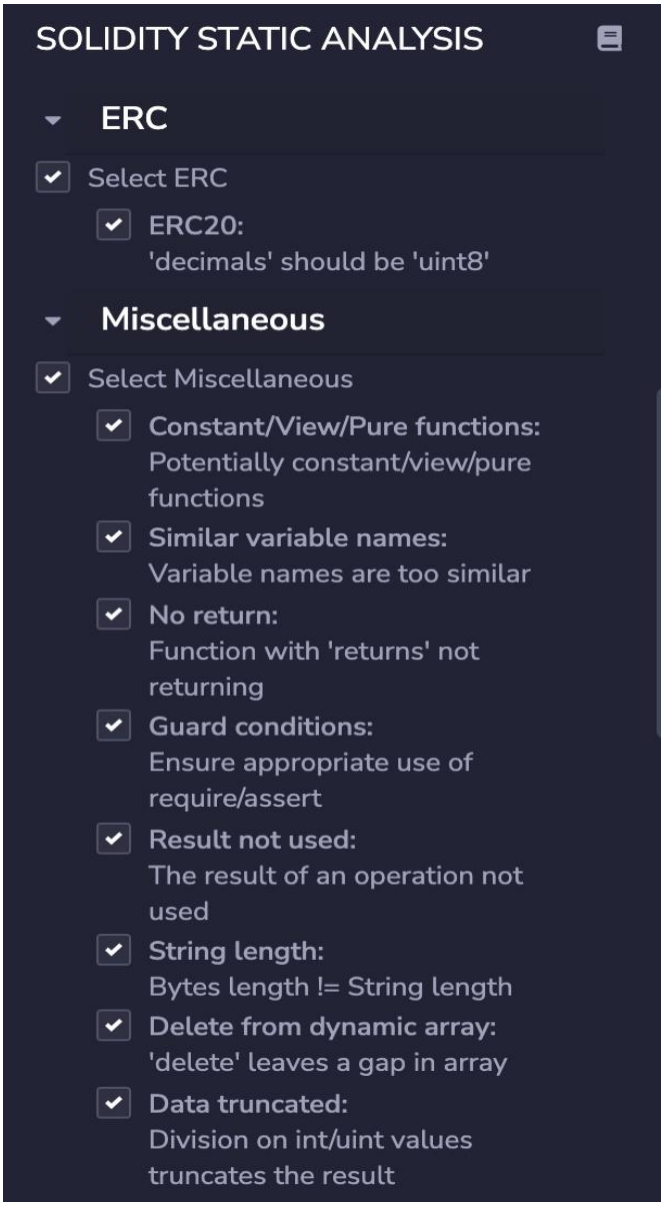
In detail
Use of "call": should be avoided whenever possible. It can lead to unexpected behavior if return value is not handled properly. Please use Direct Calls via specifying the called contract's interface.
For more reading:
https://docs.soliditylang.org/en/v0.8.0/control-structures.html#external-function-calls

Testing proves:

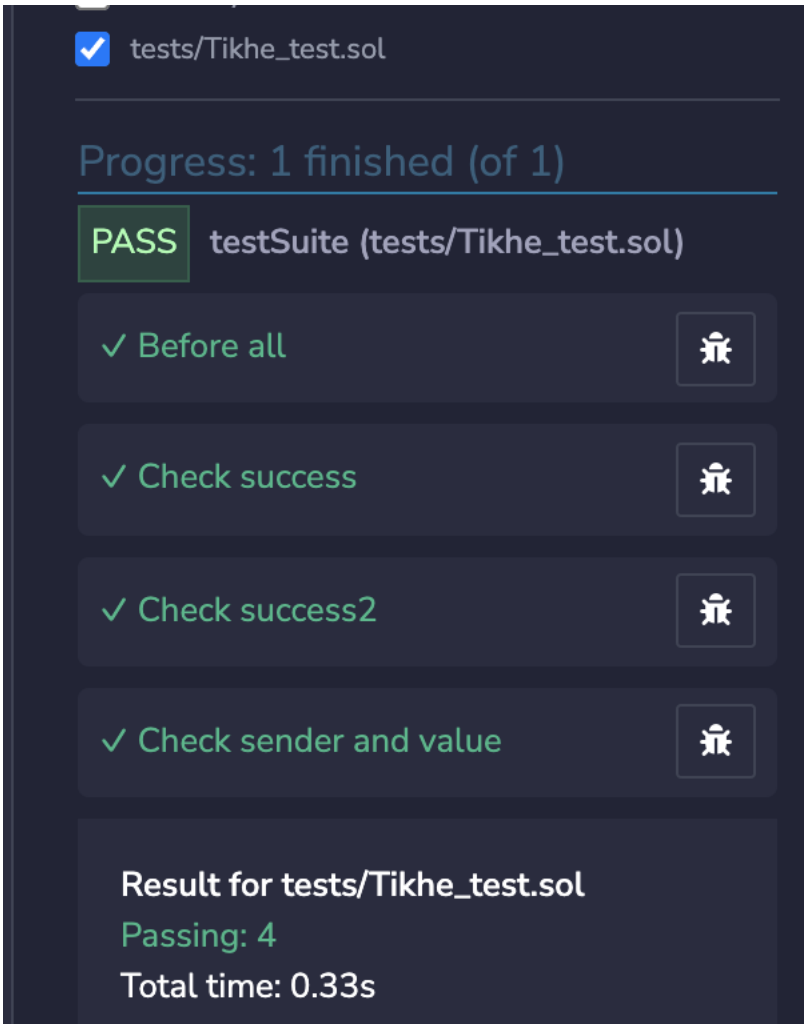
1- Check for security



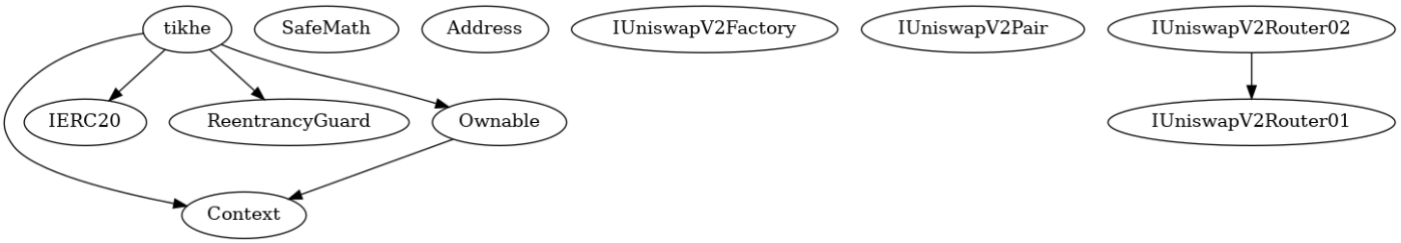
2- SOLIDITY STATIC ANALYSIS



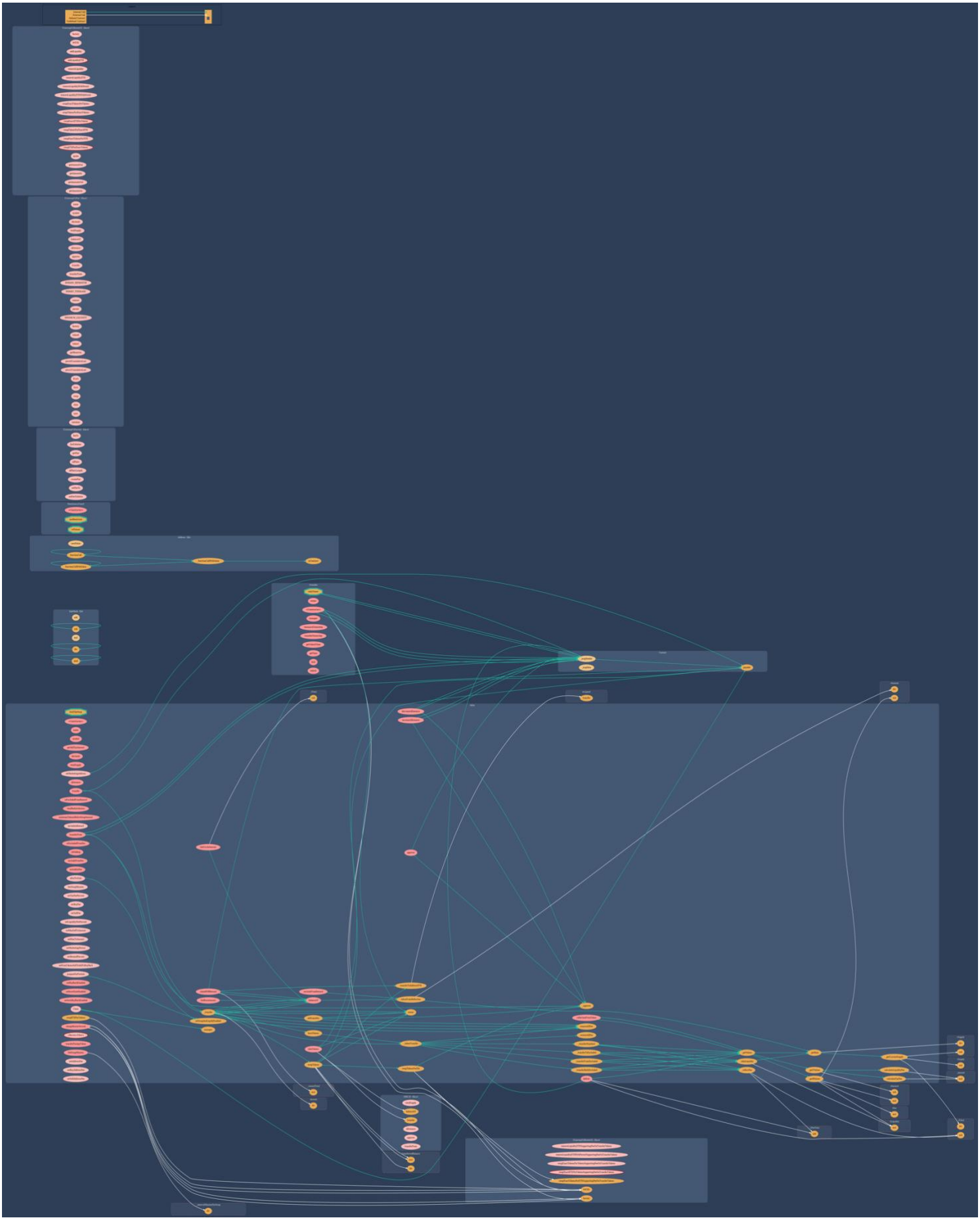
3- SOLIDITY UNIT TESTING



4- Inheritance graph



5- Call graph



Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/Tikhe.sol	6db19cf68ee4d5c4ead010b2d399506595d3d0a8

Contracts Description Table

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
SafeMath	Library			
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	

- | └ | functionCall | Internal 🔒 | 🛑 | |
- | └ | functionCall | Internal 🔒 | 🛑 | |
- | └ | functionCallWithValue | Internal 🔒 | 🛑 | |
- | └ | functionCallWithValue | Internal 🔒 | 🛑 | |
- | └ | _functionCallWithValue | Private 🔐 | 🛑 | |
- ||||||
- | **ReentrancyGuard** | Implementation | |||
- | └ | <Constructor> | Public ❗ | 🛑 | NO❗ |
- ||||||
- | **Ownable** | Implementation | Context |||
- | └ | <Constructor> | Public ❗ | 🛑 | NO❗ |
- | └ | owner | Public ❗ | | NO❗ |
- | └ | manager | Public ❗ | | NO❗ |
- | └ | renounceOwnership | Public ❗ | 🛑 | onlyOwner |
- | └ | transferOwnership | Public ❗ | 🛑 | onlyOwner |
- | └ | getUnlockTime | Public ❗ | | NO❗ |
- | └ | getTime | Public ❗ | | NO❗ |
- | └ | lock | Public ❗ | 🛑 | onlyOwner |
- | └ | unlock | Public ❗ | 🛑 | NO❗ |
- ||||||
- | **IUniswapV2Factory** | Interface | |||
- | └ | feeTo | External ❗ | | NO❗ |
- | └ | feeToSetter | External ❗ | | NO❗ |
- | └ | getPair | External ❗ | | NO❗ |
- | └ | allPairs | External ❗ | | NO❗ |
- | └ | allPairsLength | External ❗ | | NO❗ |
- | └ | createPair | External ❗ | 🛑 | NO❗ |
- | └ | setFeeTo | External ❗ | 🛑 | NO❗ |
- | └ | setFeeToSetter | External ❗ | 🛑 | NO❗ |
- ||||||
- | **IUniswapV2Pair** | Interface | |||
- | └ | name | External ❗ | | NO❗ |
- | └ | symbol | External ❗ | | NO❗ |
- | └ | decimals | External ❗ | | NO❗ |
- | └ | totalSupply | External ❗ | | NO❗ |
- | └ | balanceOf | External ❗ | | NO❗ |
- | └ | allowance | External ❗ | | NO❗ |
- | └ | approve | External ❗ | 🛑 | NO❗ |
- | └ | transfer | External ❗ | 🛑 | NO❗ |
- | └ | transferFrom | External ❗ | 🛑 | NO❗ |

└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🔘	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗
└	price1CumulativeLast	External ❗		NO❗
└	kLast	External ❗		NO❗
└	burn	External ❗	🔘	NO❗
└	swap	External ❗	🔘	NO❗
└	skim	External ❗	🔘	NO❗
└	sync	External ❗	🔘	NO❗
└	initialize	External ❗	🔘	NO❗
IUniswapV2Router01	Interface			
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidity	External ❗	🔘	NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	removeLiquidity	External ❗	🔘	NO❗
└	removeLiquidityETH	External ❗	🔘	NO❗
└	removeLiquidityWithPermit	External ❗	🔘	NO❗
└	removeLiquidityETHWithPermit	External ❗	🔘	NO❗
└	swapExactTokensForTokens	External ❗	🔘	NO❗
└	swapTokensForExactTokens	External ❗	🔘	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapTokensForExactETH	External ❗	🔘	NO❗
└	swapExactTokensForETH	External ❗	🔘	NO❗
└	swapETHForExactTokens	External ❗	💵	NO❗
└	quote	External ❗		NO❗
└	getAmountOut	External ❗		NO❗
└	getAmountIn	External ❗		NO❗
└	getAmountsOut	External ❗		NO❗
└	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	Interface	IUniswapV2Router01		

- | └ | removeLiquidityETHSupportingFeeOnTransferTokens | External ❗ | 🔘 |NO❗ |
- | └ | removeLiquidityETHWithPermitSupportingFeeOnTransferTokens | External ❗ | 🔘 |NO❗ |
- | └ | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ❗ | 🔘 |NO❗ |
- | └ | swapExactETHForTokensSupportingFeeOnTransferTokens | External ❗ | 💵 |NO❗ |
- | └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🔘 |NO❗ |
- ||||||
- | **tikhe** | Implementation | Context, IERC20, Ownable, ReentrancyGuard |||
- | └ | <Constructor> | Public ❗ | 🔘 |NO❗ |
- | └ | name | Public ❗ | |NO❗ |
- | └ | symbol | Public ❗ | |NO❗ |
- | └ | getSellTaxAmount | Public ❗ | |NO❗ |
- | └ | decimals | Public ❗ | |NO❗ |
- | └ | totalSupply | Public ❗ | |NO❗ |
- | └ | balanceOf | Public ❗ | |NO❗ |
- | └ | transfer | Public ❗ | 🔘 |NO❗ |
- | └ | allowance | Public ❗ | |NO❗ |
- | └ | approve | Public ❗ | 🔘 |NO❗ |
- | └ | transferFrom | Public ❗ | 🔘 |NO❗ |
- | └ | increaseAllowance | Public ❗ | 🔘 |NO❗ |
- | └ | decreaseAllowance | Public ❗ | 🔘 |NO❗ |
- | └ | isExcludedFromReward | Public ❗ | |NO❗ |
- | └ | totalRedistribution | Public ❗ | |NO❗ |
- | └ | minimumTokensBeforeSwapAmount | Public ❗ | |NO❗ |
- | └ | deliver | Public ❗ | 🔘 |NO❗ |
- | └ | reflectionFromToken | Public ❗ | |NO❗ |
- | └ | claimBNBReward | Public ❗ | 🔘 | isHuman nonReentrant |
- | └ | tokenFromReflection | Public ❗ | |NO❗ |
- | └ | excludeFromReward | Public ❗ | 🔘 | onlyOwner |
- | └ | includeInReward | External ❗ | 🔘 | onlyOwner |
- | └ | _approve | Private 🔐 | 🔘 | |
- | └ | _transfer | Private 🔐 | 🔘 | |
- | └ | swapTokens | Private 🔐 | 🔘 | lockTheSwap |
- | └ | burnTokens | Private 🔐 | 🔘 | lockTheSwap |
- | └ | swapTokensForEth | Private 🔐 | 🔘 | |
- | └ | swapETHForTokens | Private 🔐 | 🔘 | |
- | └ | addLiquidity | Private 🔐 | 🔘 | |
- | └ | _tokenTransfer | Private 🔐 | 🔘 | |
- | └ | _transferStandard | Private 🔐 | 🔘 | |
- | └ | takeTokens | Public ❗ | 🔘 | onlyOwner |
- | └ | _transferToExcluded | Private 🔐 | 🔘 | |

- | └ | _transferFromExcluded | Private 🔐 | 🛑 | |
- | └ | _transferBothExcluded | Private 🔐 | 🛑 | |
- | └ | _reflectFee | Private 🔐 | 🛑 | |
- | └ | _getValues | Private 🔐 | | |
- | └ | _getTValues | Private 🔐 | | |
- | └ | _getRValues | Private 🔐 | | |
- | └ | _getRate | Private 🔐 | | |
- | └ | _getCurrentSupply | Private 🔐 | | |
- | └ | _takeLiquidity | Private 🔐 | 🛑 | |
- | └ | calculateTaxFee | Private 🔐 | | |
- | └ | calculateLiquidityFee | Private 🔐 | | |
- | └ | removeAllFee | Private 🔐 | 🛑 | |
- | └ | restoreAllFee | Private 🔐 | 🛑 | |
- | └ | isExcludedFromFee | Public ❗ | |NO❗ |
- | └ | isHolding | Public ❗ | |NO❗ |
- | └ | excludeFromFee | Public ❗ | 🛑 | onlyOwner |
- | └ | includeInFee | Public ❗ | 🛑 | onlyOwner |
- | └ | GetBurnAmount | Public ❗ | |NO❗ |
- | └ | GetCircleAmount | Public ❗ | |NO❗ |
- | └ | GetSwapMinutes | Public ❗ | |NO❗ |
- | └ | SetSwapMinutes | External ❗ | 🛑 | onlyOwner |
- | └ | setTaxFeePercent | External ❗ | 🛑 | onlyOwner |
- | └ | setBuyFee | External ❗ | 🛑 | onlyOwner |
- | └ | setSellFee | External ❗ | 🛑 | onlyOwner |
- | └ | setLiquidityFeePercent | External ❗ | 🛑 | onlyOwner |
- | └ | setMaxSellTxAmount | External ❗ | 🛑 | onlyOwner |
- | └ | setMaxTxAmount | External ❗ | 🛑 | onlyOwner |
- | └ | setMarketingDivisor | External ❗ | 🛑 | onlyOwner |
- | └ | setRewardPercent | External ❗ | 🛑 | onlyOwner |
- | └ | setNumTokensSellToAddToBuyBack | External ❗ | 🛑 | onlyOwner |
- | └ | setMarketingAddress | External ❗ | 🛑 | onlyOwner |
- | └ | setSwapAndLiquifyEnabled | Public ❗ | 🛑 | onlyOwner |
- | └ | setBuyBackEnabled | Public ❗ | 🛑 | onlyOwner |
- | └ | setburnStateEnabled | Public ❗ | 🛑 | onlyOwner |
- | └ | setAutoBuyBackEnabled | Public ❗ | 🛑 | onlyOwner |
- | └ | prepareForPreSale | External ❗ | 🛑 | onlyOwner |
- | └ | afterPreSale | External ❗ | 🛑 | onlyOwner |
- | └ | transferToAddressETH | Private 🔐 | 🛑 | |
- | └ | changeRouterVersion | Public ❗ | 🛑 | onlyOwner |
- | └ | <Receive Ether> | External ❗ | 💷 |NO❗ |

```
|  └ | transferForeignToken | Public ❗ | 🛑  | onlyOwner |
|  └ | Claim | External ❗ | 🛑  | onlyOwner |
|  └ | setAddressFee | External ❗ | 🛑  | onlyOwner |
|  └ | setBuyAddressFee | External ❗ | 🛑  | onlyOwner |
|  └ | setSellAddressFee | External ❗ | 🛑  | onlyOwner |


 Legend

| Symbol | Meaning |
|:--------:|-----------|
|  🛑  | Function can modify state |
|  💵  | Function is payable |
```

- # Summary of the Audit

According to automatically test, the customer`s solidity smart contract is **Very Secured**.

The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

The test found 0 critical, 0 high, 0 medium, 3 low issues, and 1 notes.