Smart Contract Security Audit V1

Baby Dogelon Token

https://www.babydogelon.xyz/

29/1/2022

https://saferico.com/

business@saferico.com
https://t.me/SFI_ANN

–

Table of Contents

Background

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.

Project Information

- **Website**: https://www.babydogelon.xyz/
- **Telegram group:** https://telegram.org/realbabydogelon
- **Twitter**: https://twitter.com/realbabydogelon
- **Platform**: Binance Smart Chain
- **Contract Address**: 0xc19db217d96C10173070D435861a9b4138242980
- **Code Source:**
https://testnet.bscscan.com/address/0xc19db217d96c10173070d435861a9b4138242980#code

Token Information

- Name: BELON
- Total Supply: 1,000,000,000,000
- Holders: address
- Total transactions:

Contracts address deployed to test net (BSC)

Baby Dogelon smart contract on testnet.bsc (BSC Test Net)

https://testnet.bscscan.com/address/0xc19db217d96c10173070d435861a9b4138242980

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Secured**.

Well Secured	
Secured	✓
Poor Secured	
Insecure	

Automated checks are with remix IDE. All issues were performed by the team, which included the analysis of code functionality, manual audit found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the audit overview section. The general overview is presented in the Project Information section and all issues found are located in the audit overview section.

Team found 0 critical, 0 high, 0 medium, 3 low, 0 very low-level issues and 2 notes in all solidity files of the contract

The files:

BabyDogelon.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
BabyDogelon.sol	**ef271d5dbfed68a79ddc0e31273da3742c988c9f1fbba9af8faf7b1ec5c8b433**	0xc19db217d96C10173070D435861a9b4138242980

- Contract: BabyDogelon
- Inherit: Context, IERC20, Ownable
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
name	✓	Read / public	**Passed**
symbol	✓	Read / public	**Passed**
decimals	✓	Read / public	**Passed**
totalSupply	✓	Read / public	**Passed**
allowance	✓	Read / public	**Passed**
balanceOf	✓	Read / public	**Passed**
Owner	✓	Read / public	**Passed**
uniswapV2Pair	✓	Read / public	**Passed**
swapAndLiquifyEnabled	✓	Read / public	**Passed**
uniswapV2Router	✓	Read / public	**Passed**
reflectionFromToken	✓	Read / public	**Passed**
isExcludedFromReward	✓	Read / public	**Passed**

tokenFromReflection	✓	Read / public	**Passed**
maxWalletAmount	✓	Read / public	**Passed**
tradingStartBlock	✓	Read / public	**Passed**
isExcludedFromFees	✓	Read / public	**Passed**
maxBuyAmount	✓	Read / public	**Passed**
maxSellAmount	✓	Read / public	**Passed**
totalFees	✓	Read / public	**Passed**
_communityGrowthWallet	✓	Read / public	**Passed**
_isBlackListed	✓	Read / public	**Passed**
BLOCKCOUNT	✓	Read / public	**Passed**
buyFee	✓	Read / public	**Passed**
sellFee	✓	Read / public	**Passed**
approve	✓	Write / public	**Passed**
transferFrom	✓	Write / public	**Passed**
transfer	✓	Write / public	**Passed**
deliver	✓	Write / public	**Passed**
excludeFromFees	✓	Write / public	**Passed**
excludeFromReward	✓	Write / public	**Passed**
includeInFee	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**
transferOwnership	✓	Write / public	**Passed**
includeInReward	✓	Write / public	**Passed**
enableTrading	✓	Write / public	**Passed**
decreaseAllowance	✓	Write / public	**Passed**
setSellFee	✓	Write / public	**Passed**
setMaxBuyAmount	✓	Write / public	**Passed**
setBlackList	✓	Write / public	**Passed**

setSwapAndLiquifyEnabled	✓	Write / public	**Passed**
increaseAllowance	✓	Write / public	**Passed**
updateRouter	✓	Write / public	**Passed**
setBuyFee	✓	Write / public	**Passed**
setNumTokensSellToAddToLiquidity	✓	Write / public	**Passed**
setMaxSellAmount	✓	Write / public	**Passed**
setMaxWalletAmount	✓	Write / public	**Passed**
setCommunityGrowthWallet	✓	Write / public	**Passed**
claimStuckTokens	✓	Write / public	**Passed**

Issues Checking Status

No.	Issue Description	Checking Status
1	Compiler warnings.	**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 with notes**
10	Methods execution permissions.	**Passed**
11	Economy model. If application logic is based on an incorrect economic model, the application would not function correctly and participants would incur financial losses. This type of issue is most often found in bonus rewards systems, Staking and Farming contracts, Vault and Vesting contracts, etc.	**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**

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to tokens loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial functions
Medium	Medium-level vulnerabilities are important to fix; however, they can’t lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can’t have significant impact on execution
Note	Lowest-level vulnerabilities, code style violations and info statements can’t affect smart contract execution and can be ignored.

Audit Findings

Critical:

No critical severity vulnerabilities were found.

High:

No High severity vulnerabilities were found

Medium:

No Medium severity vulnerabilities were found.

Low:

#Use of block.timestamp for comparisons
Description
The value of block.timestamp can be manipulated by the miner.
And conditions with strict equality is difficult to achieve - block.timestamp
Remediation
Avoid use of block.timestamp

Status: Acknowledged

#Owner privileges (In the period when the owner isn't renounced)
Description
Owner can change Buy and Sell Fees.
Owner can enable the trading.
Owner can add any address to Blacklist.
Owner can include / exclude any address from Fees or Reward.

```
function enableTrading() external onlyOwner {
        isTradingEnabled = true;
        tradingStartBlock = block.number;}
    function isExcludedFromReward(address account) public view returns (bool) {
        return _isExcluded[account];}
    function totalFees() public view returns (uint256) {
        return _tFeeTotal;}
function excludeFromReward(address account) public onlyOwner {
        require(!_isExcluded[account], "Account is already excluded");
        if (_rOwned[account] > 0) {
            _tOwned[account] = tokenFromReflection(_rOwned[account]); }
```

```
        _isExcluded[account] = true;
        _excluded.push(account);}
    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;}}}
function excludeFromFee(address account) external onlyOwner {
        _isExcludedFromFee[account] = true;}
    function setBlackList(address addr, bool value) external onlyOwner {
        _isBlackListed[addr] = value;}
    function includeInFee(address account) external onlyOwner {
        _isExcludedFromFee[account] = false;}
    function setBuyFee(uint16 market, uint16 tax) external onlyOwner {
        buyFee.communityGrowthFee = market;
        buyFee.taxFee = tax;}
    function setSellFee(
        uint16 liq,
        uint16 market,
        uint16 tax
    ) external onlyOwner {
        sellFee.liquidityFee = liq;
        sellFee.communityGrowthFee = market;
        sellFee.taxFee = tax;}
```

Remediation

Make these functions internal in next version or the team should announce the investors before change the fees and give them time if they want to use the old fees.

P.S: This issue is common to the majority of rewards smart contracts.

Status: Acknowledged.

#Pragam version not fixed

Description

It is a good practice to lock the solidity version for a live deployment (use 0.8.10 instead of ^0.8.10). contracts should be deployed with the same compiler version and flags that they have been tested the most with. Locking the pragma helps ensure that contracts do not accidentally get deployed using, for example, the latest compiler which may have higher risks of undiscovered bugs. Contracts may also be deployed by others and the pragma indicates the compiler version intended by the original authors.

Remediation

Remove the ^ sign to lock the pragma version

Status: Acknowledged.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Naming Conventions

Description

The contract follows a consistent naming convention where we are private variables with leading"_" and public variables without it. But we have missed to comply to the condition for certain variable names "__isBlacklisted" which is public

Remediation

Remove "_" from external variable names and add it to private variable names
Status: Acknowledged

 # Constant calculations in the contract

Description

recalculated initialization will save 2847 units of gas in deployment

```
uint256 private _tTotal = 1 * 10**12 * 10**9;
```

Recommendation

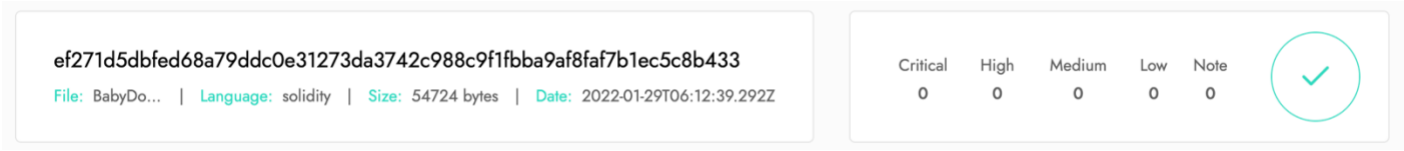
Replace the initialization as

```
uint256 private _tTotal = 1000000000000000000000;
```

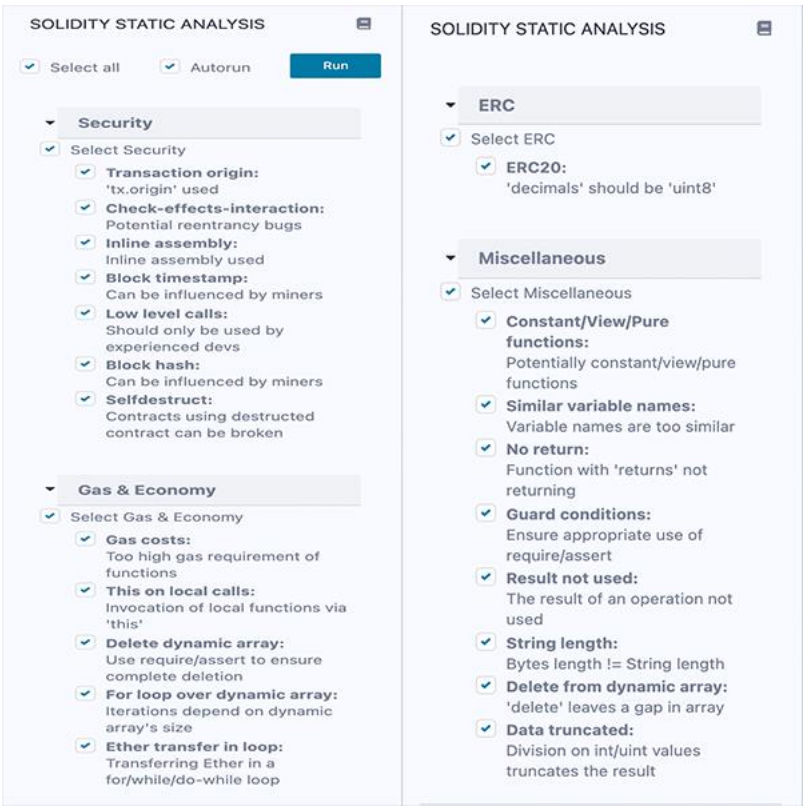
Status: Acknowledged

Automatic Testing

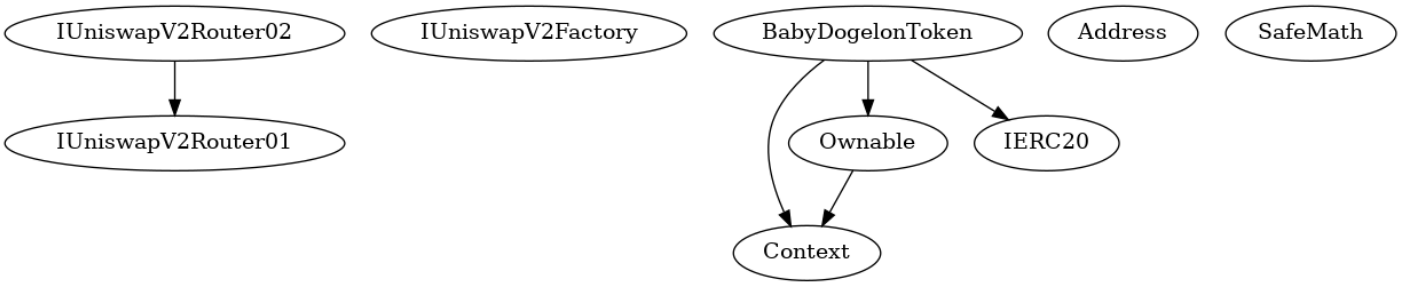
1- Check for security



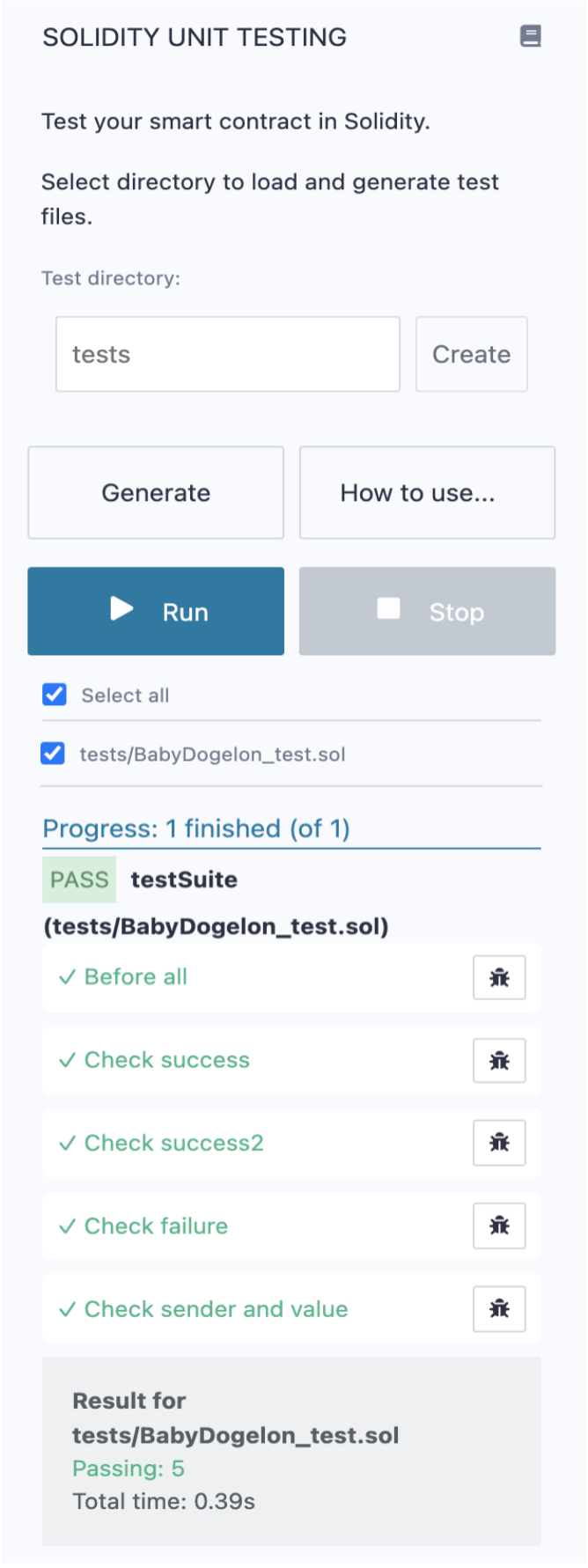
2- SOLIDITY STATIC ANALYSIS



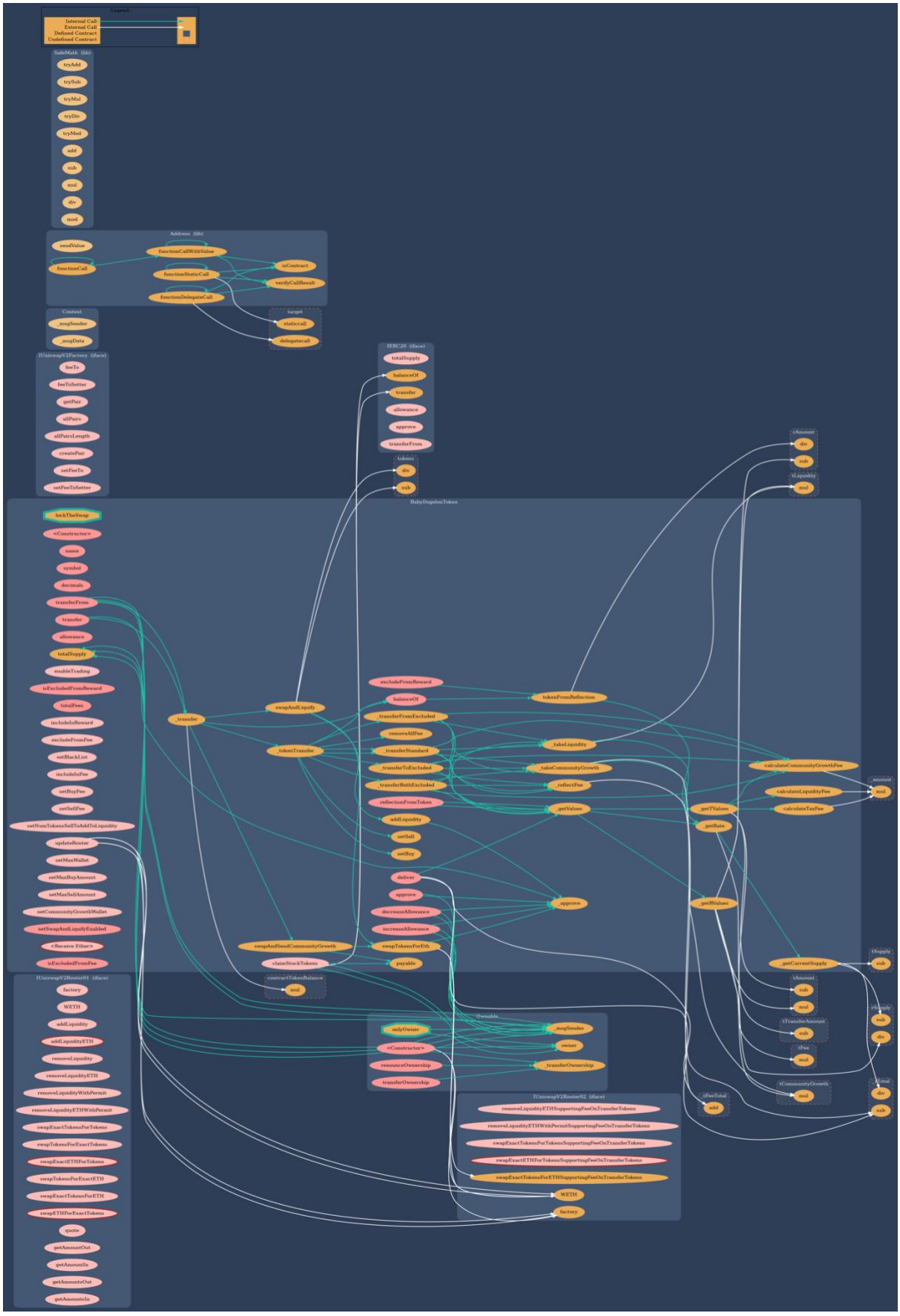
3- Inheritance graph



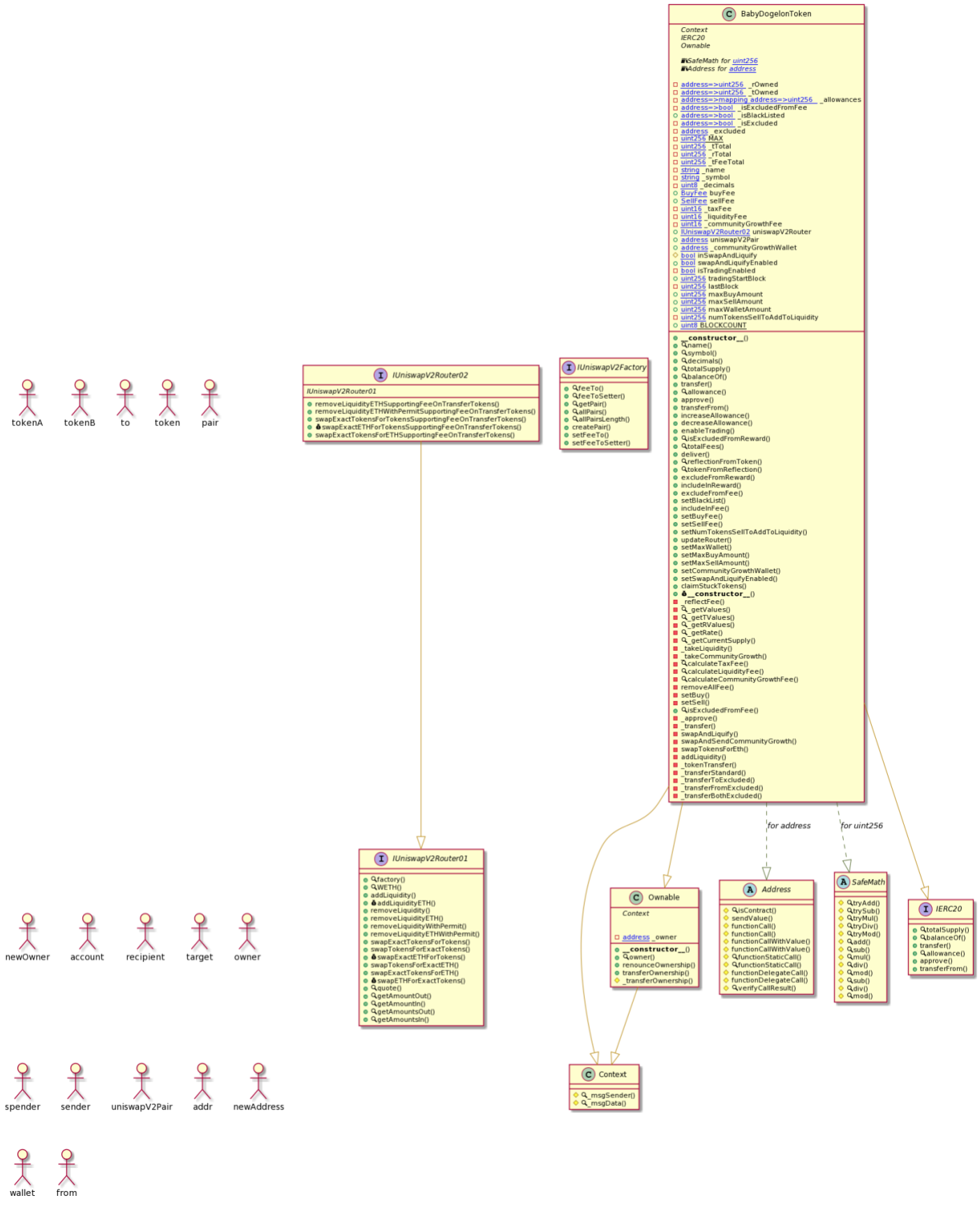
4- SOLIDITY UNIT TESTING



5- Call graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
SafeMath (lib)
tryAdd
trySub
tryMul
tryDiv
tryMod
add
sub
mul
div
mod
Address (lib)
sendValue
functionCall
functionCallWithValue
functionStaticCall
functionDelegateCall
isContract
verifyCallResult
Context
_msgSender
_msgData
target
staticcall
delegatecall
IUniswapV2Factory (iface)
feeTo
feeToSetter
getPair
allPairs
allPairsLength
createPair
setFeeTo
setFeeToSetter
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
tokens
div
sub
BabyDogelonToken
lockTheSwap
<Constructor>
name
symbol
decimals
transferFrom
transfer
allowance
totalSupply
enableTrading
isExcludedFromReward
totalFees
includeInReward
excludeFromFee
setBlackList
includeInFee
setBuyFee
setSellFee
setNumTokensSellToAddToLiquidity
updateRouter
setMaxWallet
setMaxBuyAmount
setMaxSellAmount
setCommunityGrowthWallet
setSwapAndLiquifyEnabled
<Receive Ether>
isExcludedFromFee
transfer
swapAndLiquify
_tokenTransfer
swapAndSendCommunityGrowth
claimStuckTokens
contractTokenBalance
mul
excludeFromReward
balanceOf
_transferFromExcluded
removeAllFee
_transferStandard
_transferToExcluded
_transferBothExcluded
reflectionFromToken
addLiquidity
setSell
setBuy
deliver
approve
decreaseAllowance
increaseAllowance
swapTokensForEth
payable
_approve
tokenFromReflection
_takeLiquidity
_takeCommunityGrowth
_reflectFee
_getValues
tAmount
div
sub
tLiquidity
mul
calculateCommunityGrowthFee
calculateLiquidityFee
calculateTaxFee
_amount
mul
_getTValues
_getRate
_getRValues
_getCurrentSupply
tSupply
sub
tAmount
sub
mul
tTransferAmount
sub
tFee
mul
rSupply
sub
div
tCommunityGrowth
mul
rTotal
div
sub
rFeeTotal
add
IUniswapV2Router01 (iface)
factory
WETH
addLiquidity
addLiquidityETH
removeLiquidity
removeLiquidityETH
removeLiquidityWithPermit
removeLiquidityETHWithPermit
swapExactTokensForTokens
swapTokensForExactTokens
swapExactETHForTokens
swapTokensForExactETH
swapExactTokensForETH
swapETHForExactTokens
quote
getAmountOut
getAmountIn
getAmountsOut
getAmountsIn
Ownable
_msgSender
owner
onlyOwner
<Constructor>
renounceOwnership
transferOwnership
_transferOwnership
IUniswapV2Router02 (iface)
removeLiquidityETHSupportingFeeOnTransferTokens
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens
swapExactTokensForTokensSupportingFeeOnTransferTokens
swapExactETHForTokensSupportingFeeOnTransferTokens
swapExactTokensForETHSupportingFeeOnTransferTokens
WETH
factory

Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
11902160  =>  _getTValues(uint256)
16279055  =>  isContract(address)
39509351  =>  increaseAllowance(address,uint256)
75128141  =>  calculateTaxFee(uint256)
c45a0155  =>  factory()
ad5c4648  =>  WETH()
e8e33700  =>
addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256)
f305d719  =>  addLiquidityETH(address,uint256,uint256,uint256,address,uint256)
baa2abde  =>
removeLiquidity(address,address,uint256,uint256,uint256,address,uint256)
02751cec  =>  removeLiquidityETH(address,uint256,uint256,uint256,address,uint256)
2195995c  =>
removeLiquidityWithPermit(address,address,uint256,uint256,uint256,address,uint256,b
ool,uint8,bytes32,bytes32)
ded9382a  =>
removeLiquidityETHWithPermit(address,uint256,uint256,uint256,address,uint256,bool,u
int8,bytes32,bytes32)
38ed1739  =>  swapExactTokensForTokens(uint256,uint256,address[],address,uint256)
8803dbee  =>  swapTokensForExactTokens(uint256,uint256,address[],address,uint256)
7ff36ab5  =>  swapExactETHForTokens(uint256,address[],address,uint256)
4a25d94a  =>  swapTokensForExactETH(uint256,uint256,address[],address,uint256)
18cbafe5  =>  swapExactTokensForETH(uint256,uint256,address[],address,uint256)
fb3bdb41  =>  swapETHForExactTokens(uint256,address[],address,uint256)
ad615dec  =>  quote(uint256,uint256,uint256)
054d50d4  =>  getAmountOut(uint256,uint256,uint256)
85f8c259  =>  getAmountIn(uint256,uint256,uint256)
d06ca61f  =>  getAmountsOut(uint256,address[])
1f00ca74  =>  getAmountsIn(uint256,address[])
af2979eb  =>
removeLiquidityETHSupportingFeeOnTransferTokens(address,uint256,uint256,uint256,add
ress,uint256)
5b0d5984  =>
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens(address,uint256,uint256,u
int256,address,uint256,bool,uint8,bytes32,bytes32)
5c11d795  =>
swapExactTokensForTokensSupportingFeeOnTransferTokens(uint256,uint256,address[],add
ress,uint256)
b6f9de95  =>
swapExactETHForTokensSupportingFeeOnTransferTokens(uint256,address[],address,uint25
6)
791ac947  =>
swapExactTokensForETHSupportingFeeOnTransferTokens(uint256,uint256,address[],addres
s,uint256)
017e7e58  =>  feeTo()
094b7415  =>  feeToSetter()
e6a43905  =>  getPair(address,address)
1e3dd18b  =>  allPairs(uint256)
574f2ba3  =>  allPairsLength()
c9c65396  =>  createPair(address,address)
f46901ed  =>  setFeeTo(address)
a2e74af6  =>  setFeeToSetter(address)
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
```

```
8da5cb5b  =>  owner()
715018a6  =>  renounceOwnership()
f2fde38b  =>  transferOwnership(address)
d29d44ee  =>  _transferOwnership(address)
24a084df  =>  sendValue(address,uint256)
a0b5ffb0  =>  functionCall(address,bytes)
241b5886  =>  functionCall(address,bytes,string)
2a011594  =>  functionCallWithValue(address,bytes,uint256)
d525ab8a  =>  functionCallWithValue(address,bytes,uint256,string)
c21d36f3  =>  functionStaticCall(address,bytes)
dbc40fb9  =>  functionStaticCall(address,bytes,string)
ee33b7e2  =>  functionDelegateCall(address,bytes)
57387df0  =>  functionDelegateCall(address,bytes,string)
946b5793  =>  verifyCallResult(bool,bytes,string)
884557bf  =>  tryAdd(uint256,uint256)
a29962b1  =>  trySub(uint256,uint256)
6281efa4  =>  tryMul(uint256,uint256)
736ecb18  =>  tryDiv(uint256,uint256)
38dc0867  =>  tryMod(uint256,uint256)
771602f7  =>  add(uint256,uint256)
b67d77c5  =>  sub(uint256,uint256)
c8a4ac9c  =>  mul(uint256,uint256)
a391c15b  =>  div(uint256,uint256)
f43f523a  =>  mod(uint256,uint256)
e31bdc0a  =>  sub(uint256,uint256,string)
b745d336  =>  div(uint256,uint256,string)
71af23e8  =>  mod(uint256,uint256,string)
18160ddd  =>  totalSupply()
70a08231  =>  balanceOf(address)
a9059cbb  =>  transfer(address,uint256)
dd62ed3e  =>  allowance(address,address)
095ea7b3  =>  approve(address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
06fdde03  =>  name()
95d89b41  =>  symbol()
313ce567  =>  decimals()
a457c2d7  =>  decreaseAllowance(address,uint256)
8a8c523c  =>  enableTrading()
88f82020  =>  isExcludedFromReward(address)
13114a9d  =>  totalFees()
3bd5d173  =>  deliver(uint256)
4549b039  =>  reflectionFromToken(uint256,bool)
2d838119  =>  tokenFromReflection(uint256)
52390c02  =>  excludeFromReward(address)
3685d419  =>  includeInReward(address)
437823ec  =>  excludeFromFee(address)
68092bd9  =>  setBlackList(address,bool)
ea2f0b37  =>  includeInFee(address)
7afad249  =>  setBuyFee(uint16,uint16)
2e222fa2  =>  setSellFee(uint16,uint16,uint16)
f0f165af  =>  setNumTokensSellToAddToLiquidity(uint256)
c851cc32  =>  updateRouter(address)
5d0044ca  =>  setMaxWallet(uint256)
f34eb0b8  =>  setMaxBuyAmount(uint256)
e99c9d09  =>  setMaxSellAmount(uint256)
8920731b  =>  setCommunityGrowthWallet(address)
c49b9a80  =>  setSwapAndLiquifyEnabled(bool)
f9d0831a  =>  claimStuckTokens(address)
184d894e  =>  _reflectFee(uint256,uint256)
```

```
d4780e36  =>  _getValues(uint256)
65c63d72  =>  _getRValues(uint256,uint256,uint256,uint256,uint256)
94e10784  =>  _getRate()
97a9d560  =>  _getCurrentSupply()
c432df5e  =>  _takeLiquidity(uint256)
9fe808d1  =>  _takeCommunityGrowth(uint256)
cc126a23  =>  calculateLiquidityFee(uint256)
38a39203  =>  calculateCommunityGrowthFee(uint256)
301370af  =>  removeAllFee()
e2437408  =>  setBuy()
a2b1f062  =>  setSell()
5342acb4  =>  isExcludedFromFee(address)
104e81ff  =>  _approve(address,address,uint256)
30e0789e  =>  _transfer(address,address,uint256)
173865ad  =>  swapAndLiquify(uint256)
9b85ec1b  =>  swapAndSendCommunityGrowth(uint256)
b28805f4  =>  swapTokensForEth(uint256)
9cd441da  =>  addLiquidity(uint256,uint256)
b09bbc79  =>  _tokenTransfer(address,address,uint256,bool)
2852df65  =>  _transferStandard(address,address,uint256)
16f1cc83  =>  _transferToExcluded(address,address,uint256)
c7d9be66  =>  _transferFromExcluded(address,address,uint256)
6ff6cdf4  =>  _transferBothExcluded(address,address,uint256)
```

Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/BabyDogelon.sol	4c347861d034198a091d808a628afad440a9c40c

Contracts Description Table

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
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❗
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❗


Context | Implementation |
└ | _msgSender | Internal 🔒 |
└ | _msgData | Internal 🔒 |
Ownable | Implementation | Context
└ | <Constructor> | Public ❗ | 🛑 | NO❗
└ | owner | Public ❗ | NO❗
└ | renounceOwnership | Public ❗ | 🛑 | onlyOwner
└ | transferOwnership | Public ❗ | 🛑 | onlyOwner
└ | _transferOwnership | Internal 🔒 | 🛑
Address | Library |
└ | isContract | Internal 🔒 |
└ | sendValue | Internal 🔒 | 🛑
└ | functionCall | Internal 🔒 | 🛑
└ | functionCall | Internal 🔒 | 🛑
└ | functionCallWithValue | Internal 🔒 | 🛑
└ | functionCallWithValue | Internal 🔒 | 🛑
└ | functionStaticCall | Internal 🔒 |
└ | functionStaticCall | Internal 🔒 |
└ | functionDelegateCall | Internal 🔒 | 🛑
└ | functionDelegateCall | Internal 🔒 | 🛑
└ | verifyCallResult | Internal 🔒 |
SafeMath | Library |
└ | tryAdd | Internal 🔒 |
└ | trySub | Internal 🔒 |
└ | tryMul | Internal 🔒 |
└ | tryDiv | Internal 🔒 |
└ | tryMod | Internal 🔒 |
└ | add | Internal 🔒 |
└ | sub | Internal 🔒 |
└ | mul | Internal 🔒 |
└ | div | Internal 🔒 |
└ | mod | Internal 🔒 |
└ | sub | Internal 🔒 |
└ | div | Internal 🔒 |
└ | mod | Internal 🔒 |
IERC20 | Interface |
└ | totalSupply | External ❗ | NO❗
└ | balanceOf | External ❗ | NO❗
└ | transfer | External ❗ | 🛑 | NO❗
└ | allowance | External ❗ | NO❗
└ | approve | External ❗ | 🛑 | NO❗
└ | transferFrom | External ❗ | 🛑 | NO❗
BabyDogelonToken | Implementation | Context, IERC20, Ownable
└ | <Constructor> | Public ❗ | 🛑 | NO❗
└ | name | Public ❗ | NO❗
└ | symbol | 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❗
└	enableTrading	External ❗	🛑	onlyOwner
└	isExcludedFromReward	Public ❗		NO❗
└	totalFees	Public ❗		NO❗
└	deliver	Public ❗	🛑	NO❗
└	reflectionFromToken	Public ❗		NO❗
└	tokenFromReflection	Public ❗		NO❗
└	excludeFromReward	Public ❗	🛑	onlyOwner
└	includeInReward	External ❗	🛑	onlyOwner
└	excludeFromFee	External ❗	🛑	onlyOwner
└	setBlackList	External ❗	🛑	onlyOwner
└	includeInFee	External ❗	🛑	onlyOwner
└	setBuyFee	External ❗	🛑	onlyOwner
└	setSellFee	External ❗	🛑	onlyOwner
└	setNumTokensSellToAddToLiquidity	External ❗	🛑	onlyOwner
└	updateRouter	External ❗	🛑	onlyOwner
└	setMaxWallet	External ❗	🛑	onlyOwner
└	setMaxBuyAmount	External ❗	🛑	onlyOwner
└	setMaxSellAmount	External ❗	🛑	onlyOwner
└	setCommunityGrowthWallet	External ❗	🛑	onlyOwner
└	setSwapAndLiquifyEnabled	Public ❗	🛑	onlyOwner
└	claimStuckTokens	External ❗	🛑	onlyOwner
└	<Receive Ether>	External ❗	💵	NO❗
└	_reflectFee	Private 🔐	🛑	
└	_getValues	Private 🔐		
└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	_takeLiquidity	Private 🔐	🛑	
└	_takeCommunityGrowth	Private 🔐	🛑	
└	calculateTaxFee	Private 🔐		
└	calculateLiquidityFee	Private 🔐		
└	calculateCommunityGrowthFee	Private 🔐		
└	removeAllFee	Private 🔐	🛑	
└	setBuy	Private 🔐	🛑	
└	setSell	Private 🔐	🛑	
└	isExcludedFromFee	Public ❗		NO❗
└	_approve	Private 🔐	🛑	
└	_transfer	Private 🔐	🛑	
└	swapAndLiquify	Private 🔐	🛑	lockTheSwap
└	swapAndSendCommunityGrowth	Private 🔐	🛑	lockTheSwap
└	swapTokensForEth	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	_transferStandard	Private 🔐	🛑	
└	_transferToExcluded	Private 🔐	🛑	
└	_transferFromExcluded	Private 🔐	🛑	
└	_transferBothExcluded	Private 🔐	🛑	

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Conclusion

The contracts are written systematically. Team found no critical issues. So, it is good to go for production.

Since possible test cases can be unlimited and developer level documentation (code flow diagram with function level description) not provided, for such an extensive smart contract protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan Everything.

Security state of the reviewed contract is “secured”.

✓ No mint function.
✓ No volatile code.
✓ Not many high severity issues were found.
✓ Contract Ownership Renounced.

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 s) 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.