Smart Contract
Security Audit
V1

Raju Presale Smart Contract

27/9/2022

https://saferico.com/

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

–

Table of Contents

Table of Contents

Background

Project Information

- Smart Contract Information
- Executive Summary

File and Function Level Report

- **File in Scope:**

Issues Checking Status

- Severity Definitions
- Audit Findings

Automatic testing

- Testing proves
- Inheritance graph
- Call graph

Unified Modeling Language (UML)

Functions signature

Automatic general report

Conclusion

Disclaimer

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

- **Platform**: Polygon
- **Contract Address**: 0xb2aAF43Ebd6CE58fB3C35556DAE3B3B4416c74E7
- **Code Source:**

https://github.com/Saferico/Smart-Contracts-for-Projects/blob/main/%20Raju%20contracts/RajuPresale.sol

Contracts address deployed to test net (Polygon)

Raju Presale smart contract on Polygon test-net by the auditor to test every function (Polygon Test Net)

https://mumbai.polygonscan.com/address/0xb2aaf43ebd6ce58fb3c35556dae3b3b4416c74e7

Smart Contract Information

- Name: Raju Presale
- Phase 1: Whitelist, vesting term, buy

* Whitelist:

Only whitelisted members can buy tokens on the IDO site.

* Referral system

Any token holder can invite others.

Add invited members to the whitelist.

The inviter will receive a referral commission rate according to the buy amount of the invited member.

* Vesting term.

30% instantly, then xx% everyday till unlock 70% from Phase3.

* Buy

No selling here. Users only can buy tokens in fixed price (price1) with buying tax.

Details:

Price: 0.01$

Commission rate: 2%

Rebasing Rate: 2.81% per day

Phase1 period: 25 days

Phase 2: Referral system, buy/sell

* Referral system

Same as Phase1.

* Buy/Sell

Users can buy/sell tokens in fixed price (price 2) with buy/sell tax.

Details:

Vesting term: no

Price: 0.02$

Commission rate: 2%

Rebasing Rate: variable until starting

Phase2 period: variable until starting

Phase 3: Launching on dex

We will add liquidity on dex with funds from Phase 1,2. (by price3)

There will be buy/sell/p2p tax here.

No whitelist.

No referral system.

Start rebasing.

Details:

Price: $22

Commission rate: no

Rebasing Rate: around xx% APY

Executive Summary

According to our assessment, the customer`s solidity smart contract is **Well 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 1 note in all solidity files of the contract

The files:

RajuPresale.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
RajuPresale.sol	**a607f8b537a1ac02040cdd51fa25322159ebc1e06d236f7b9a9fbfaa0c261e5e**	0xb2aAF43Ebd6CE58fB3C35556DAE3B3B4416c74E7

- Contract: RajuPresale
- Inherit: Ownable
- Observation: All passed including security check
- Test Report: passed
- Score: passed
- Conclusion: passed

Function	Test Result	Type / Return Type	Score
dailyRate	✓	Read / public	**Passed**
getClaimableAmount	✓	Read / public	**Passed**
getDepositedAmount	✓	Read / public	**Passed**
getPercentReleased	✓	Read / public	**Passed**
getRebaseRate	✓	Read / public	**Passed**
isList	✓	Read / public	**Passed**
isPresaleOpen	✓	Read / public	**Passed**
maxTokenSell	✓	Read / public	**Passed**
maxUsdcBuy	✓	Read / public	**Passed**
minTokenSell	✓	Read / public	**Passed**
minUsdcBuy	✓	Read / public	**Passed**
owner	✓	Read / public	**Passed**

periods	✓	Read / public	**Passed**
preBuys	✓	Read / public	**Passed**
pToken	✓	Read / public	**Passed**
RATE_DECIMALS	✓	Read / public	**Passed**
rate1	✓	Read / public	**Passed**
rate2	✓	Read / public	**Passed**
REBASE_DECIMALS	✓	Read / public	**Passed**
rebaseRates	✓	Read / public	**Passed**
referralAccount	✓	Read / public	**Passed**
referralRate	✓	Read / public	**Passed**
startTimes	✓	Read / public	**Passed**
totalpTokenAmountToDistribute	✓	Read / public	**Passed**
USDC_DECIMALS	✓	Read / public	**Passed**
usdc	✓	Read / public	**Passed**
usdcRaised	✓	Read / public	**Passed**
wallet	✓	Read / public	**Passed**
whiteListed	✓	Read / public	**Passed**
buyerAccounts	✓	Read / public	**Passed**
bought	✓	Read / public	**Passed**
buy	✓	Write / public	**Passed**
sell	✓	Write / public	**Passed**
closePrsale	✓	Write / public	**Passed**
claim	✓	Write / public	**Passed**
renounceOwnership	✓	Write / public	**Passed**
retreiveExcesspToken	✓	Write /public	**Passed**
retreiveUSDC	✓	Write / public	**Passed**
setDailyRate	✓	Write / public	**Passed**
setList	✓	Write / public	**Passed**

setMaxBuyUsdc	✓	Write / public	**Passed**
setMinBuy	✓	Write / public	**Passed**
setMaxSellToken	✓	Write / public	**Passed**
setMinSellToken	✓	Write / public	**Passed**
setPhaseSetting	✓	Write / public	**Passed**
setPresaleTokenAddress	✓	Write / public	**Passed**
setRate1	✓	Write / public	**Passed**
setRate2	✓	Write / public	**Passed**
setReferralRate	✓	Write / public	**Passed**
setWallet	✓	Write / public	**Passed**
setWhitelist	✓	Write / public	**Passed**
startPresale	✓	Write / public	**Passed**
transferOwnership	✓	Write / public	**Passed**
update	✓	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	Design Logic.	**Passed**
6	Timestamp dependence.	**Passed with notes**
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**

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:

#Missing zero address validation

Description

When the owner wants to add addresses to whitelist, he has to check for the zero address to make he didn't add the zero address.

```
function setWhitelist(address[] memory addresses, bool value) public onlyOwner{
        require(isList, "Whitelist disable");
        for (uint i = 0; i < addresses.length; i++) {
                        whiteListed[addresses[i]] = value;
        } }
```

Remediation

Use the require statement to check for zero addresses.

Status: Closed. Fixed in version 2.

#Pragam version not fixed

Description

It is a good practice to lock the solidity version for a live deployment (use 0.8.16 instead of ^0.8.0). 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: Closed. Fixed in version2

#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

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

#Missing SPDX-License-Identifier:

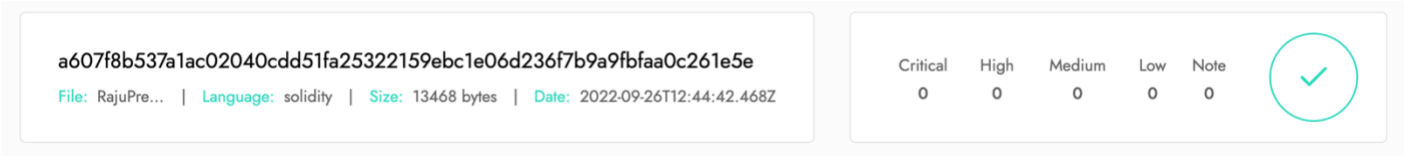
Warning: SPDX license identifier not provided in source file. Before publishing, consider adding a comment containing "SPDX-License-Identifier: <SPDX-License>" to each source file. Use "SPDX-License-Identifier: UNLICENSED" for non-open-source code. Please see https://spdx.org for more information .

Remediation
Add License Identifier
// SPDX-License-Identifier: MIT

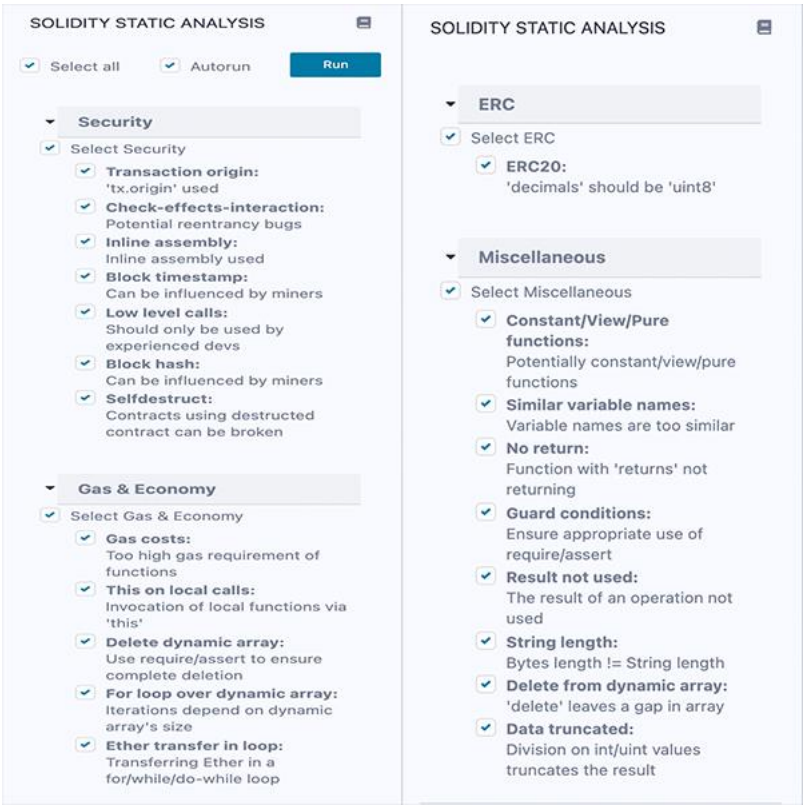
Status: Closed. Fixed In version 2.

Automatic Testing

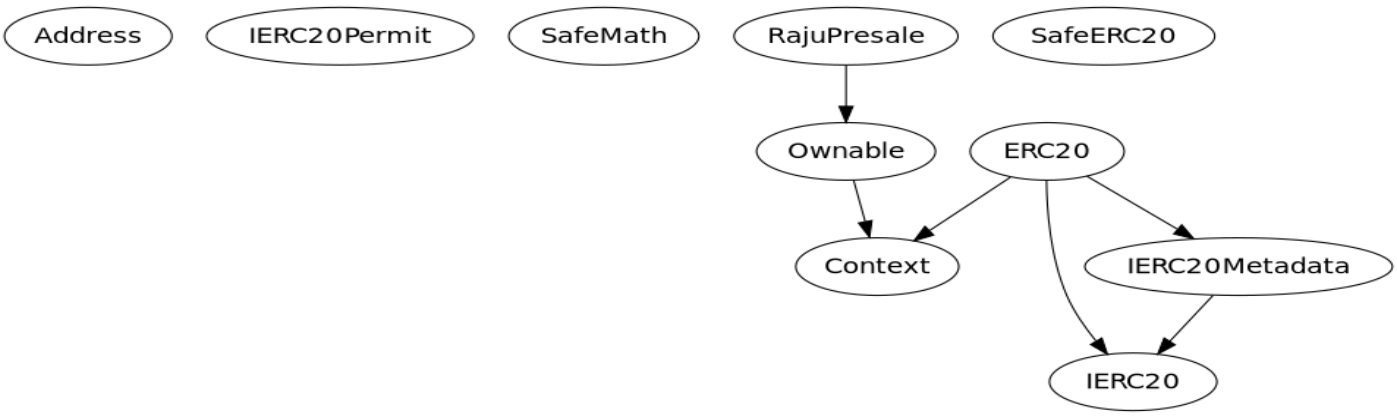
1- Check for security



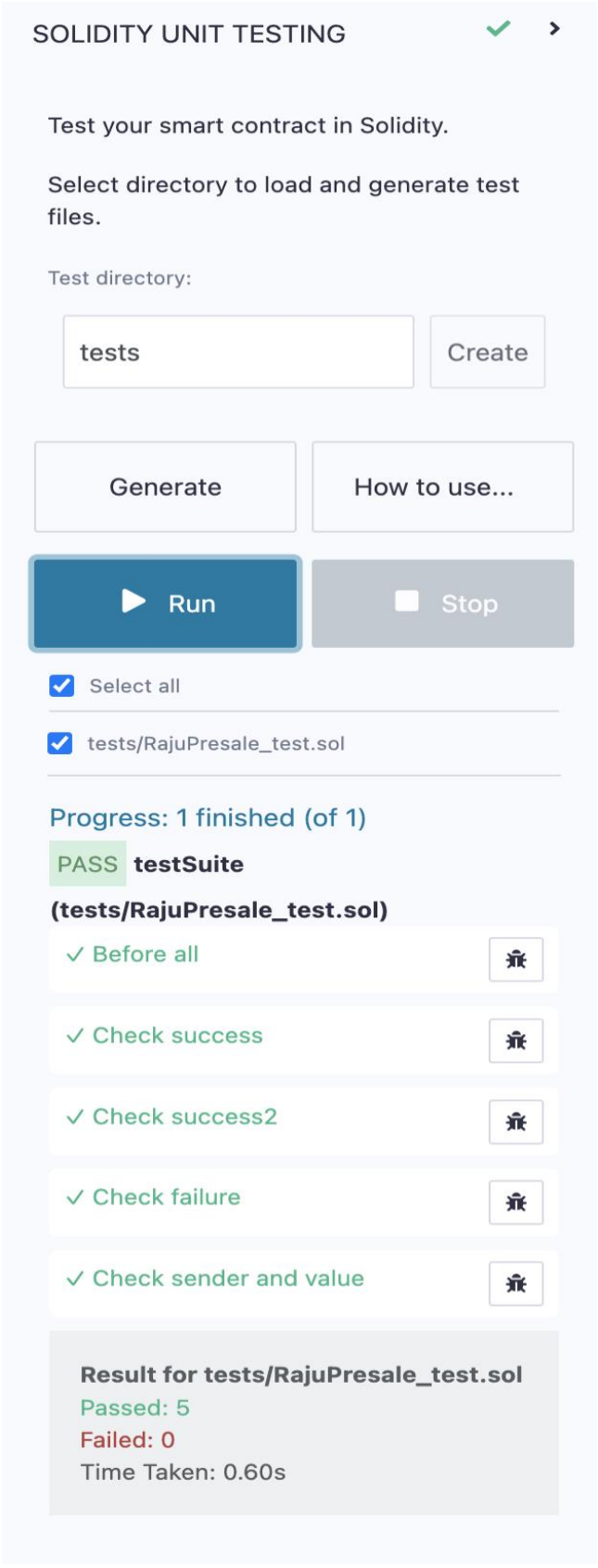
2- SOLIDITY STATIC ANALYSIS



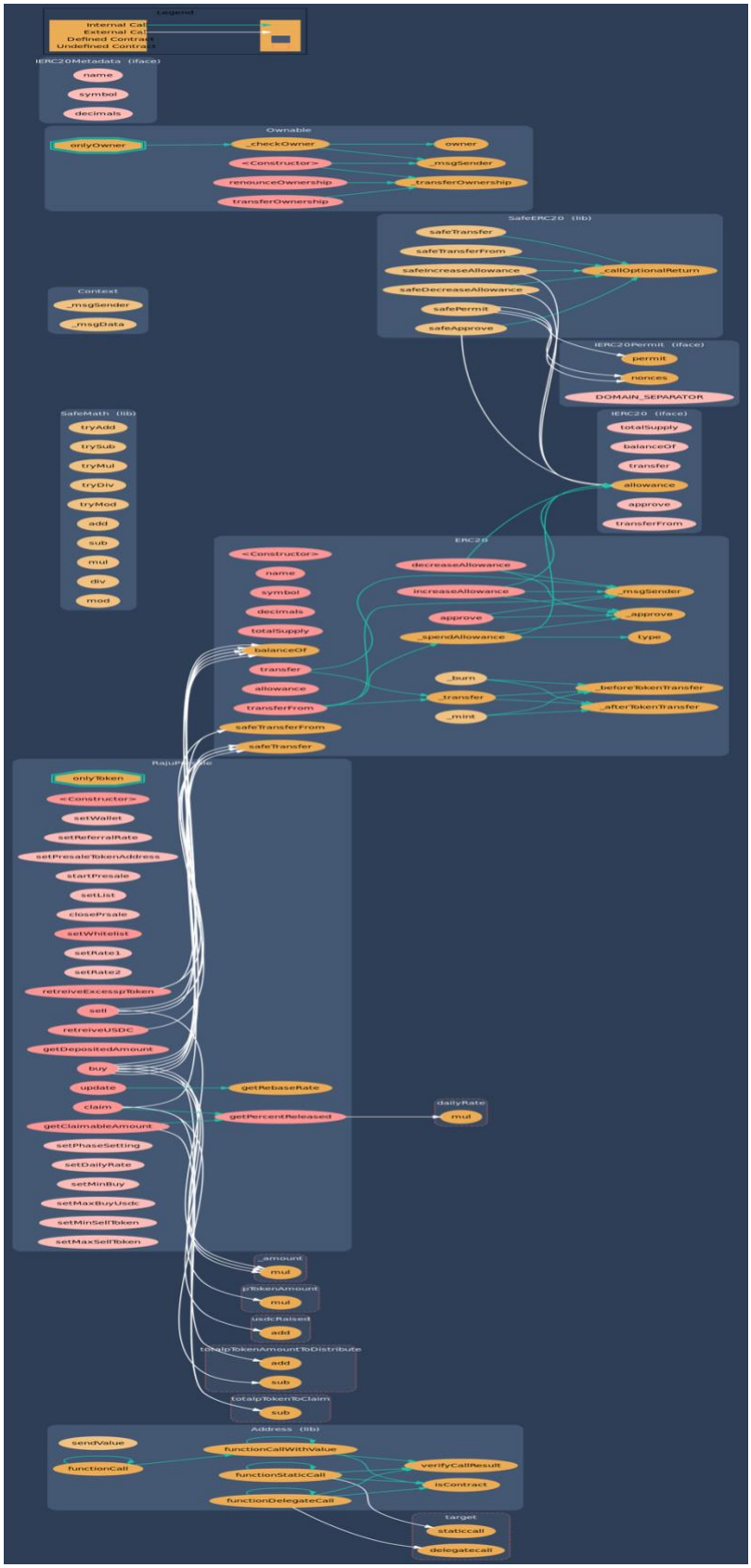
3- Inheritance graph



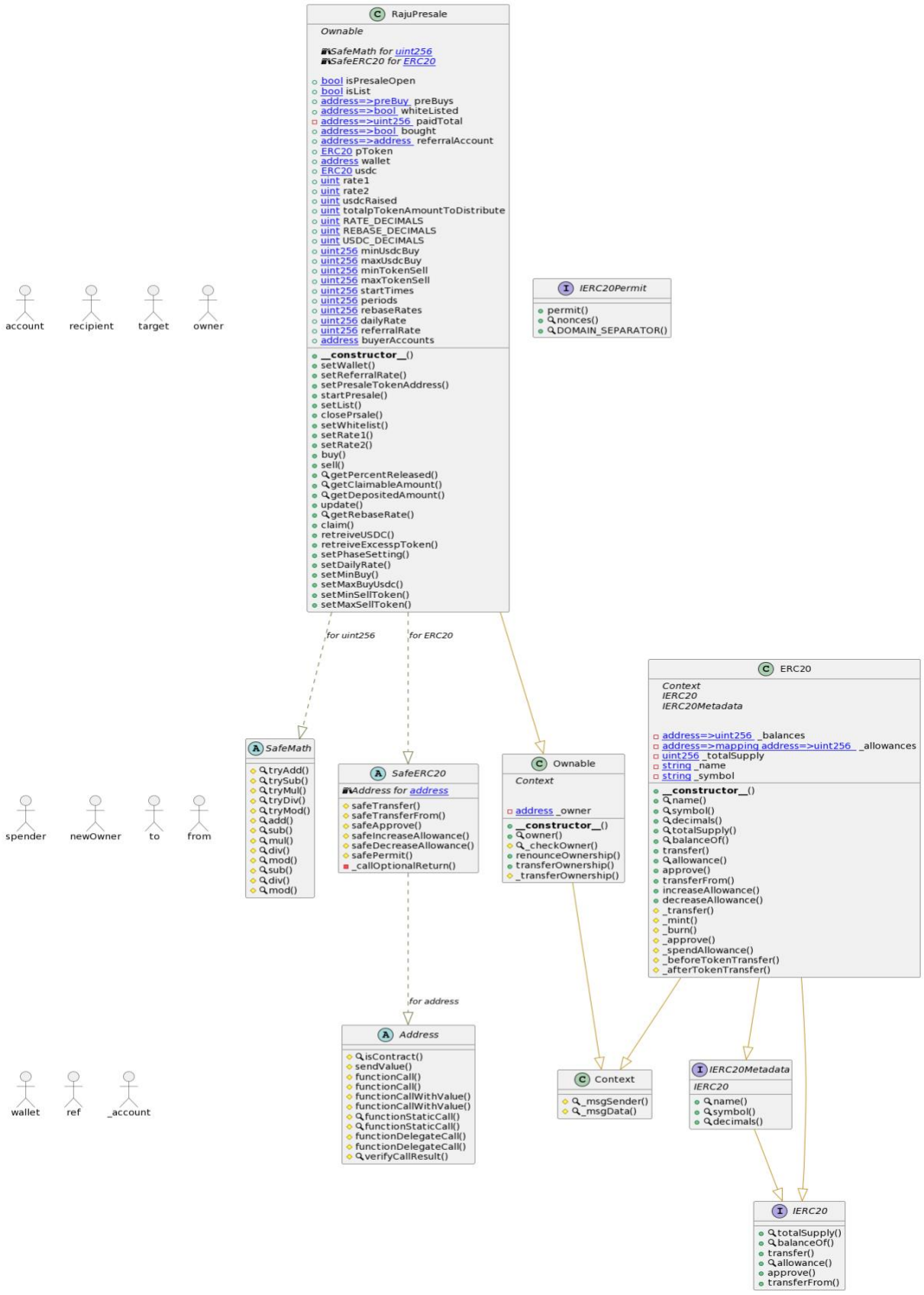
4- SOLIDITY UNIT TESTING



5- Call graph


Legend
IERC20Metadata (iface)
name
symbol
decimals
Ownable
onlyOwner
_checkOwner
owner
<Constructor>
_msgSender
renounceOwnership
_transferOwnership
transferOwnership
SafeERC20 (lib)
safeTransfer
safeTransferFrom
safeIncreaseAllowance
_callOptionalReturn
safeDecreaseAllowance
safePermit
safeApprove
Context
_msgSender
_msgData
IERC20Permit (iface)
permit
nonces
DOMAIN_SEPARATOR
SafeMath (lib)
tryAdd
trySub
tryMul
tryDiv
tryMod
add
sub
mul
div
mod
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
ERC20
<Constructor>
name
symbol
decimals
totalSupply
balanceOf
transfer
allowance
transferFrom
decreaseAllowance
increaseAllowance
approve
_spendAllowance
_msgSender
_approve
type
_burn
_transfer
_mint
_beforeTokenTransfer
_afterTokenTransfer
safeTransferFrom
safeTransfer
onlyToken
<Constructor>
setWallet
setReferralRate
setPresaleTokenAddress
startPresale
setList
closePrsale
setWhitelist
setRate1
setRate2
retreiveExcesspToken
sell
retreiveUSDC
getDepositedAmount
buy
update
claim
getClaimableAmount
setPhaseSetting
setDailyRate
setMinBuy
setMaxBuyUsdc
setMinSellToken
setMaxSellToken
getRebaseRate
getPercentReleased
dailyRate
mul
_amount
mul
pTokenAmount
mul
usdcRaised
add
totalpTokenAmountToDistribute
add
sub
totalpTokenToClaim
sub
Address (lib)
sendValue
functionCall
functionCallWithValue
functionStaticCall
functionDelegateCall
verifyCallResult
isContract
target
staticcall
delegatecall

Unified Modeling Language (UML)



Functions signature

```
Sighash   |   Function Signature
========================
16279055  =>  isContract(address)
29028696  =>  setList(bool)
39509351  =>  increaseAllowance(address,uint256)
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)
d505accf  =>  permit(address,address,uint256,uint256,uint8,bytes32,bytes32)
7ecebe00  =>  nonces(address)
3644e515  =>  DOMAIN_SEPARATOR()
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)
119df25f  =>  _msgSender()
8b49d47e  =>  _msgData()
8da5cb5b  =>  owner()
53a72975  =>  _checkOwner()
715018a6  =>  renounceOwnership()
f2fde38b  =>  transferOwnership(address)
d29d44ee  =>  _transferOwnership(address)
18160ddd  =>  totalSupply()
70a08231  =>  balanceOf(address)
a9059cbb  =>  transfer(address,uint256)
dd62ed3e  =>  allowance(address,address)
095ea7b3  =>  approve(address,uint256)
23b872dd  =>  transferFrom(address,address,uint256)
d0c407e1  =>  safeTransfer(IERC20,address,uint256)
5beae096  =>  safeTransferFrom(IERC20,address,address,uint256)
d6dcec8d  =>  safeApprove(IERC20,address,uint256)
390cc046  =>  safeIncreaseAllowance(IERC20,address,uint256)
5164ffed  =>  safeDecreaseAllowance(IERC20,address,uint256)
aecb368d  =>
safePermit(IERC20Permit,address,address,uint256,uint256,uint8,bytes32,bytes32)
becc5a20  =>  _callOptionalReturn(IERC20,bytes)
06fdde03  =>  name()
95d89b41  =>  symbol()
313ce567  =>  decimals()
a457c2d7  =>  decreaseAllowance(address,uint256)
```

```
30e0789e  =>  _transfer(address,address,uint256)
4e6ec247  =>  _mint(address,uint256)
6161eb18  =>  _burn(address,uint256)
104e81ff  =>  _approve(address,address,uint256)
1532335e  =>  _spendAllowance(address,address,uint256)
cad3be83  =>  _beforeTokenTransfer(address,address,uint256)
8f811a1c  =>  _afterTokenTransfer(address,address,uint256)
deaa59df  =>  setWallet(address)
ca4863fd  =>  setReferralRate(uint256)
3c0c134d  =>  setPresaleTokenAddress(address)
04c98b2b  =>  startPresale()
75305fc6  =>  closePrsale()
3c271a05  =>  setWhitelist(address[],bool)
b86b14cd  =>  setRate1(uint256)
79e32a62  =>  setRate2(uint256)
7deb6025  =>  buy(uint256,address)
e4849b32  =>  sell(uint256)
9cfd81f6  =>  getPercentReleased()
e12f3a61  =>  getClaimableAmount(address)
e9081ec7  =>  getDepositedAmount(address)
a2e62045  =>  update()
9e675d17  =>  getRebaseRate(uint256)
4e71d92d  =>  claim()
e71f683a  =>  retreiveUSDC(uint256)
3c9a26d0  =>  retreiveExcesspToken()
7e02899a  =>  setPhaseSetting(uint256,uint256,uint256,uint256)
187ab7a2  =>  setDailyRate(uint256)
860779e7  =>  setMinBuy(uint256)
52d0c881  =>  setMaxBuyUsdc(uint256)
978a4081  =>  setMinSellToken(uint256)
626df490  =>  setMaxSellToken(uint256)
```

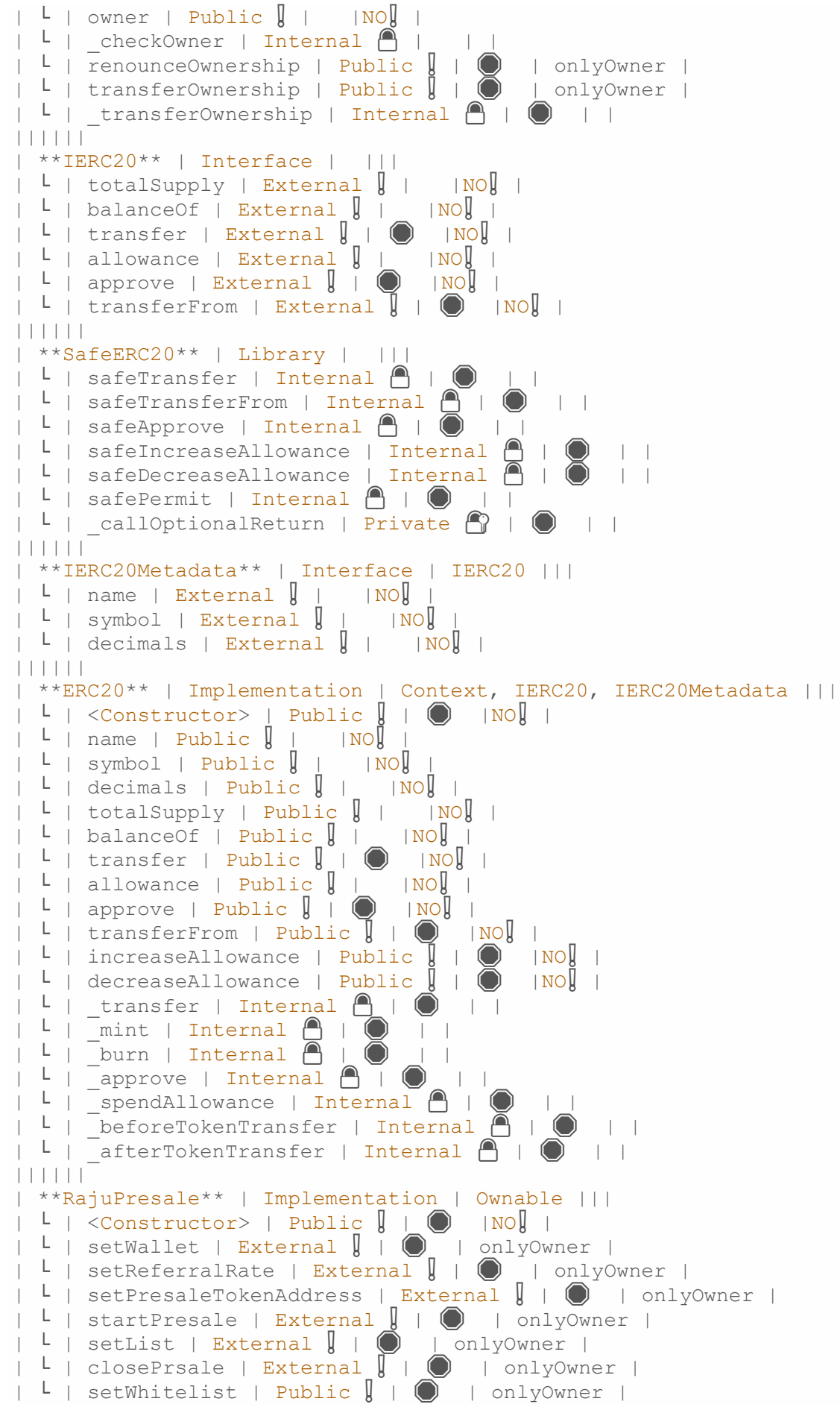
Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/RajuPresale.sol	d5f683aa8dc2b6d1c6b2c5c8edda211b922317cd

Contracts Description Table

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	verifyCallResult	Internal 🔒		
IERC20Permit	Interface			
└	permit	External ❗	🛑	NO❗
└	nonces	External ❗		NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
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 🔒		
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗


└	owner	Public ❗		NO❗
└	_checkOner	Internal 🔒		
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
IERC20	Interface		
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
SafeERC20	Library		
└	safeTransfer	Internal 🔒	🛑	
└	safeTransferFrom	Internal 🔒	🛑	
└	safeApprove	Internal 🔒	🛑	
└	safeIncreaseAllowance	Internal 🔒	🛑	
└	safeDecreaseAllowance	Internal 🔒	🛑	
└	safePermit	Internal 🔒	🛑	
└	_callOptionalReturn	Private 🔐	🛑	
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<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❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_spendAllowance	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
RajuPresale	Implementation	Ownable		
└	<Constructor>	Public ❗	🛑	NO❗
└	setWallet	External ❗	🛑	onlyOwner
└	setReferralRate	External ❗	🛑	onlyOwner
└	setPresaleTokenAddress	External ❗	🛑	onlyOwner
└	startPresale	External ❗	🛑	onlyOwner
└	setList	External ❗	🛑	onlyOwner
└	closePrsale	External ❗	🛑	onlyOwner
└	setWhitelist	Public ❗	🛑	onlyOwner

└	setRate1	External ❗	🛑	onlyOwner
└	setRate2	External ❗	🛑	onlyOwner
└	buy	Public ❗	🛑	NO❗
└	sell	Public ❗	🛑	NO❗
└	getPercentReleased	Public ❗		NO❗
└	getClaimableAmount	Public ❗		NO❗
└	getDepositedAmount	Public ❗		NO❗
└	update	Public ❗	🛑	onlyToken
└	getRebaseRate	Public ❗		NO❗
└	claim	Public ❗	🛑	NO❗
└	retreiveUSDC	Public ❗	🛑	NO❗
└	retreiveExcesspToken	Public ❗	🛑	NO❗
└	setPhaseSetting	External ❗	🛑	onlyOwner
└	setDailyRate	External ❗	🛑	onlyOwner
└	setMinBuy	External ❗	🛑	onlyOwner
└	setMaxBuyUsdc	External ❗	🛑	onlyOwner
└	setMinSellToken	External ❗	🛑	onlyOwner
└	setMaxSellToken	External ❗	🛑	onlyOwner

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 “Well Secured”.

✓ No mint function.
✓ No volatile code.
✓ No high severity issues were found.

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.