

Smart Contract Security Audit V1

Voltoken Smart Contract

5/4/2022



<https://saferico.com/>

business@saferico.com

https://t.me/SFI_ANN

—

Table of Contents

Table of Contents

Background

Project Information

Token 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:** Ethereum
- **Contract Address:** 0xaDd31ffC5f20c2634d3cfC4Dd301e4bc1889B83f
- **Code Source:**
<https://github.com/Saferico/Smart-Contracts-for-Projects/blob/main/Voltoken.sol>

Token Information

- Name: VLT
- Total Supply: 1000,000,000,000,000
- Holders:
- Total transactions:

Contracts address deployed to test net (ETH)

Voltoken smart contract on Eth testnet by the auditor to test every function (ETH Test Net)

<https://rinkeby.etherscan.io/address/0xadd31ffc5f20c2634d3cfc4dd301e4bc1889b83f>

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, 1 high, 0 medium, 3 low, 0 very low-level issues and 1 note in all solidity files of the contract

The files:

Voltoken.sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
VLT.sol	9e109fa889d972aa7194a2730016763c0fcf2bd004785314e4838297df6eb658	0xaDd31ffC5f20c2634d3cfC4Dd301e4bc1889B83f

- Contract: VLT
- Inherit: ERC20, Ownable, SafeToken, LockToken
- 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
ETHRewardFee	✓	Read / public	Passed
extraFeeOnSell	✓	Read / public	Passed
automatedMarketMaker Pairs	✓	Read / public	Passed
burnFee	✓	Read / public	Passed
developmentFee	✓	Read / public	Passed

deadWallet	✓	Read / public	Passed
dividendTokenBalanceOf	✓	Read / public	Passed
dividendTracker	✓	Read / public	Passed
gasForProcessing	✓	Read / public	Passed
getAccountDividendsInfo	✓	Read / public	Passed
getAccountDividendsInfoAtIndex	✓	Read / public	Passed
getClaimWait	✓	Read / public	Passed
getLastProcessedIndex	✓	Read / public	Passed
getNumberOfDividendTokenHolders	✓	Read / public	Passed
getTotalDividendsDistributed	✓	Read / public	Passed
isExcludedFromFees	✓	Read / public	Passed
liquidityFee	✓	Read / public	Passed
isExcludedFromMaxTx	✓	Read / public	Passed
swapTokensAtAmount	✓	Read / public	Passed
isOpen	✓	Read / public	Passed
uniswapV2Pair	✓	Read / public	Passed
uniswapV2Router	✓	Read / public	Passed
projectDevWallet	✓	Read / public	Passed
swapAndLiquifyEnabled	✓	Read / public	Passed
withdrawableDividendOf	✓	Read / public	Passed
maxBuyTransactionAmount	✓	Read / public	Passed
totalFees	✓	Read / public	Passed
approve	✓	Write / public	Passed
TransferFrom	✓	Write / public	Passed
includeToWhiteList	✓	Write / public	Passed
transfer	✓	Write / public	Passed
excludeFromDividends	✓	Write / public	Passed

claim	✓	Write / public	Passed
updateDividendTracker	✓	Write / public	Passed
excludeFromFees	✓	Write / public	Passed
excludeMultipleAccountsFromFees	✓	Write / public	Passed
setExcludeFromMaxTx	✓	Write / public	Passed
renounceOwnership	✓	Write / public	Passed
transferOwnership	✓	Write / public	Passed
setAutomatedMarketMakerPair	✓	Write / public	Passed
setExcludeFromAll	✓	Write / public	Passed
setMaxBuyTx	✓	Write / public	Passed
setFee	✓	Write / public	Passed
setProjectDevWallet	✓	Write / public	Passed
setSwapTokensAtAmount	✓	Write / public	Passed
setExtraFeeOnSell	✓	Write / public	Passed
processDividendTracker	✓	Write / public	Passed
updateClaimWait	✓	Write / public	Passed
updateGasForProcessing	✓	Write / public	Passed
updateUniswapV2Router	✓	Write / public	Passed
increaseAllowance	✓	Write / public	Passed
decreaseAllowance	✓	Write / public	Passed
openTrade	✓	Write / public	Passed
setSafeManager	✓	Write / public	Passed
withdraw	✓	Write / public	Passed
withdrawBNB	N/A	Write / public	N/A

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
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:

#Contract code size exceeds 24576 bytes

Description

Contract implementation is too large in size to be deployed on main net. Ethereum with its spurious dragon release limited the size of the contracts deployable on main net to 24576 bytes.

The size of the contract Voltoken.sol goes way above this value.

You can read more here:

<https://github.com/ethereum/EIPs/issues/170>

Remediation

Define and use libraries for pure and view functions e.g. We can create a library which contains all the mathematical operations.

Status: **Closed**. Fixed in version 2.

Medium:

No Medium severity vulnerabilities were found.

Low:

#Pragam version not fixed

Description

It is a good practice to lock the solidity version for a live deployment (use 0.8.2 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**

#Owner privileges (In the period when the owner isn't renounced)

Description

The owner can change Buy and Sell Fees (Liquidity, ETH Reward, development, and burn) or make it = zero.

The owner can add any address to White List.

The owner can include / exclude any address from Fees or Reward.

```
function setFee(uint256 _ethRewardFee, uint256 _liquidityFee, uint256
_developmentFee, uint256 _burnFee) public onlyOwner {
    ETHRewardFee = _ethRewardFee;
    liquidityFee = _liquidityFee;
    developmentFee = _developmentFee;
    burnFee = _burnFee;
    totalFees =
    ETHRewardFee.add(liquidityFee).add(developmentFee).add(burnFee).add(burnFee); //
total fee transfer and buy}
    function setExtraFeeOnSell(uint256 _extraFeeOnSell) public onlyOwner {
        extraFeeOnSell = _extraFeeOnSell; // extra fee on sell}
    function setMaxBuytx(uint256 _maxBuyTxAmount) public onlyOwner {
        maxBuyTransactionAmount = _maxBuyTxAmount;}
function excludeFromFees(address account, bool excluded) public onlyOwner {
    require(!_isExcludedFromFees[account] != excluded, "VLT: Account is already
the value of 'excluded'");
    _isExcludedFromFees[account] = excluded;
    emit ExcludeFromFees(account, excluded);}
    function setExcludeFromMaxTx(address _address, bool value) public onlyOwner {
        _isExcludedFromMaxTx[_address] = value;}
    function setExcludeFromAll(address _address) public onlyOwner {
        _isExcludedFromMaxTx[_address] = true;
        _isExcludedFromFees[_address] = true;
        dividendTracker.excludeFromDividends(_address);}
    function excludeMultipleAccountsFromFees(address[] calldata accounts, bool
excluded) public onlyOwner {
        for(uint256 i = 0; i < accounts.length; i++) {
            _isExcludedFromFees[accounts[i]] = excluded;}
        emit ExcludeMultipleAccountsFromFees(accounts, excluded);
    }
}
function updateClaimWait(uint256 claimWait) external onlyOwner {
    dividendTracker.updateClaimWait(claimWait);
}
```

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**.

Very Low:

No Very Low severity vulnerabilities were found.

Notes:

Constant calculations in the contract

Description

recalculated initialization will save 2847 units of gas in deployment

```
uint256 public maxBuyTransactionAmount = 10000000000000000 * (10**9);
uint256 public swapTokensAtAmount = 10000000000000000 * (10**9);
_mint(owner(), 10000000000000000 * (10**9));
```

Recommendation

Replace the initialization as

```
uint256 public maxBuyTransactionAmount = 10000000000000000000000000;
uint256 public swapTokensAtAmount = 10000000000000000000000000;
_mint(owner(), 10000000000000000000000000);
```

Status: **Closed**. Fixed in version2

Automatic Testing

1- Check for security

9e109fa889d972aa7194a2730016763c0fcf2bd004785314e4838297df6eb658

File: Voltke... | Language: solidity | Size: 74947 bytes | Date: 2022-04-05T09:16:28.947Z

Critical High Medium Low Note
0 0 0 0 0



2- SOLIDITY STATIC ANALYSIS

SOLIDITY STATIC ANALYSIS

☒ Select all ☒ Autorun

Security

☒ Select Security

- ☒ Transaction origin:
'tx.origin' used
- ☒ Check-effects-interaction:
Potential reentrancy bugs
- ☒ Inline assembly:
Inline assembly used
- ☒ Block timestamp:
Can be influenced by miners
- ☒ Low level calls:
Should only be used by experienced devs
- ☒ Block hash:
Can be influenced by miners
- ☒ Selfdestruct:
Contracts using destructed contract can be broken

Gas & Economy

☒ Select Gas & Economy

- ☒ Gas costs:
Too high gas requirement of functions
- ☒ This on local calls:
Invocation of local functions via 'this'
- ☒ Delete dynamic array:
Use require/assert to ensure complete deletion
- ☒ For loop over dynamic array:
Iterations depend on dynamic array's size
- ☒ Ether transfer in loop:
Transferring Ether in a for/while/do-while loop

SOLIDITY STATIC ANALYSIS

ERC

☒ Select ERC

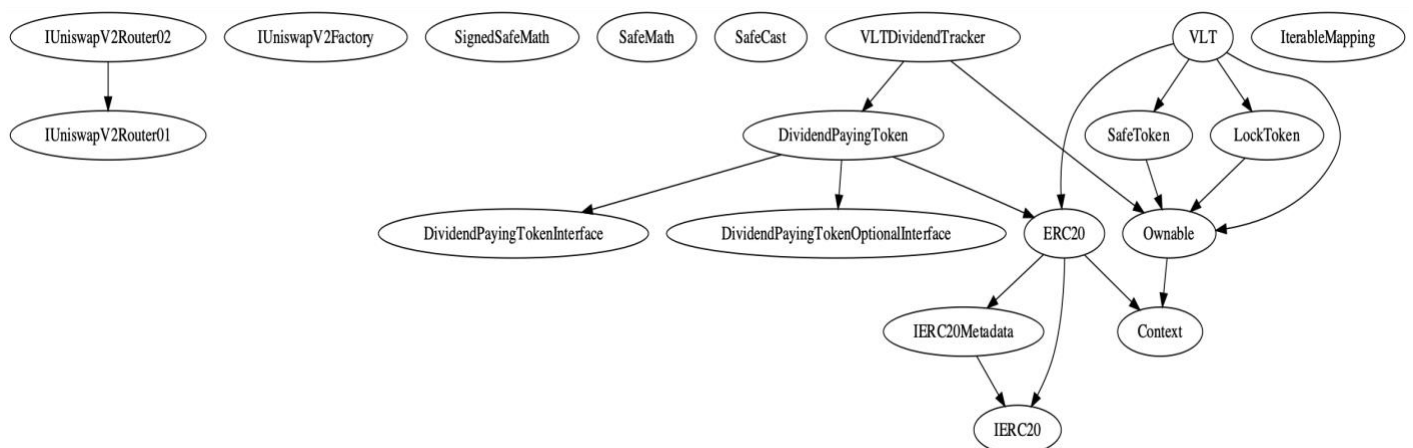
- ☒ ERC20:
'decimals' should be 'uint8'

Miscellaneous

☒ Select Miscellaneous

- ☒ Constant/View/Pure functions:
Potentially constant/view/pure functions
- ☒ Similar variable names:
Variable names are too similar
- ☒ No return:
Function with 'returns' not returning
- ☒ Guard conditions:
Ensure appropriate use of require/assert
- ☒ Result not used:
The result of an operation not used
- ☒ String length:
Bytes length != String length
- ☒ Delete from dynamic array:
'delete' leaves a gap in array
- ☒ Data truncated:
Division on int/uint values truncates the result

3- Inheritance graph



4- SOLIDITY UNIT TESTING

SOLIDITY UNIT TESTING

Test your smart contract in Solidity.

Select directory to load and generate test files.

Test directory:

☒ Select all

☒ tests/Voltoken_test.sol

Progress: 1 finished (of 1)

PASS testSuite (tests/Voltoken_test.sol)

✓ Before all

✓ Check success

✓ Check success2

✓ Check failure

✓ Check sender and value

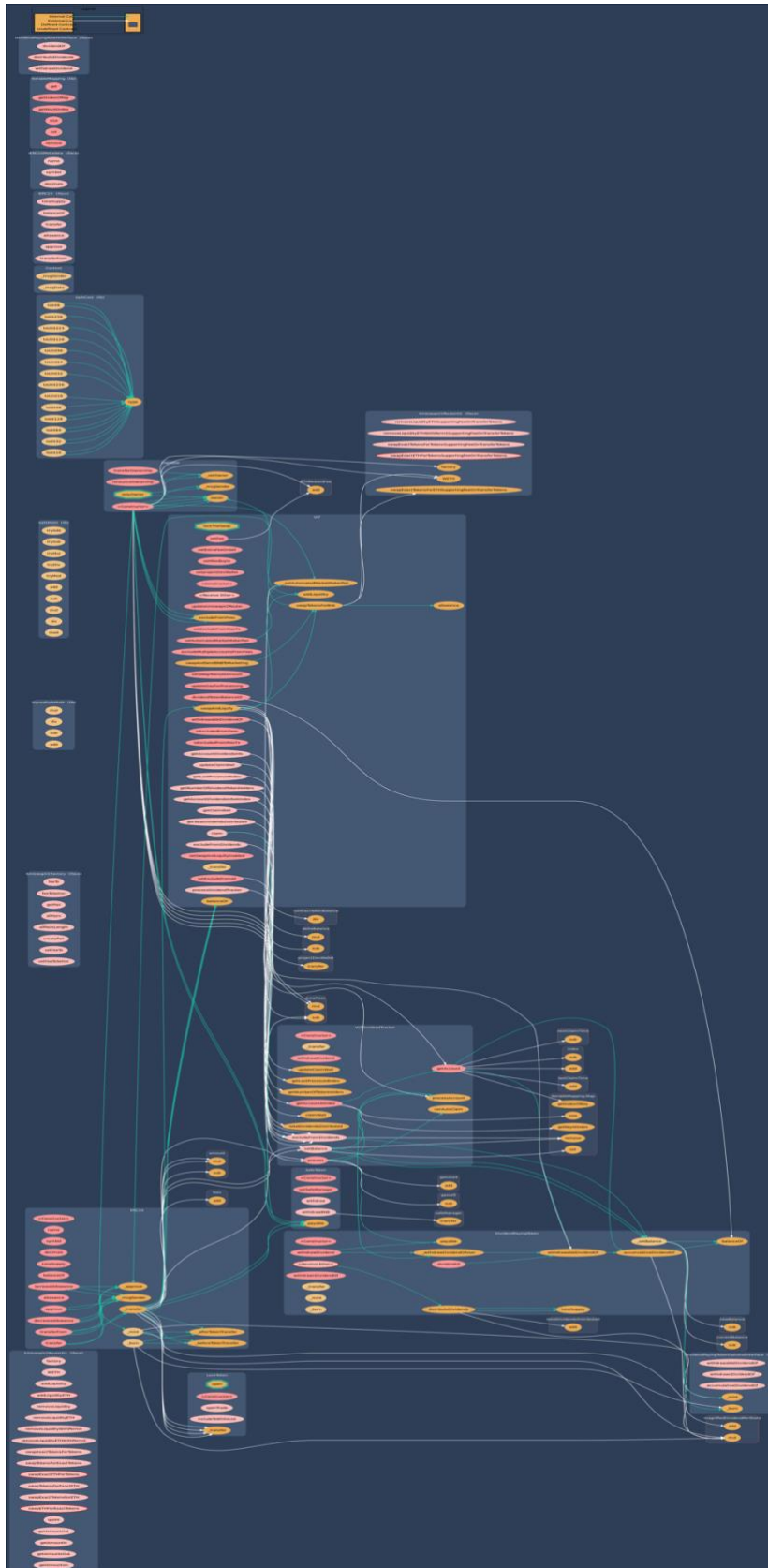
Result for tests/Voltoken_test.sol

Passed: 5

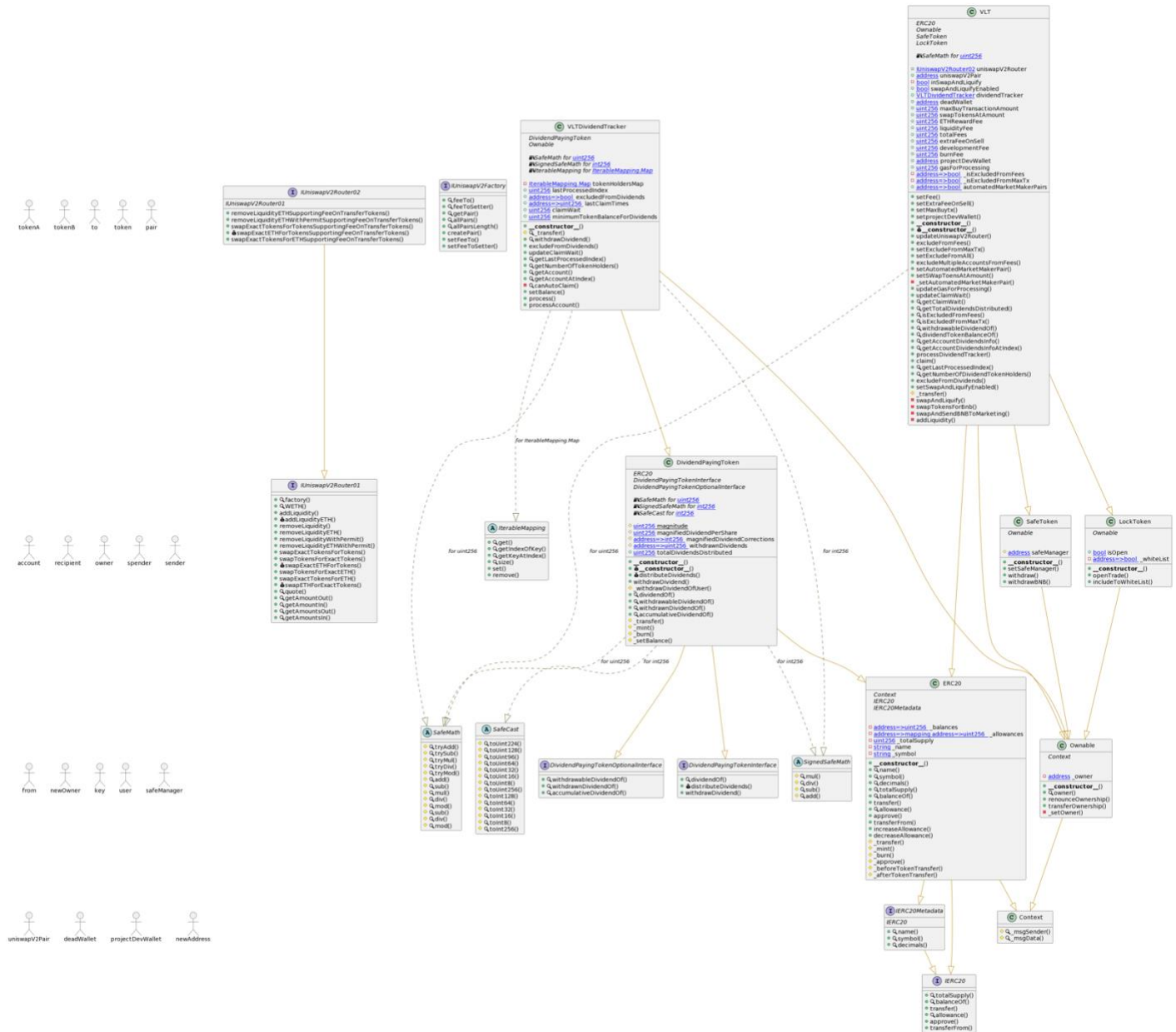
Failed: 0

Time Taken: 0.44s

5- Call graph



Unified Modeling Language (UML)



Functions signature

Sighash	Function Signature
39509351	=> increaseAllowance (address,uint256)
43509138	=> div (int256,int256)
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,boolean,uint8,bytes32,bytes32)
ded9382a	=>
	removeLiquidityETHWithPermit (address,uint256,uint256,uint256,address,uint256,boolean,uint8,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,address,uint256)
5b0d5984	=>
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens (address,uint256,uint256,uint256,address,uint256,boolean,uint8,bytes32,bytes32)
5c11d795	=>
	swapExactTokensForTokensSupportingFeeOnTransferTokens (uint256,uint256,address[],address,uint256)
b6f9de95	=>
	swapExactETHForTokensSupportingFeeOnTransferTokens (uint256,address[],address,uint256)
791ac947	=>
	swapExactTokensForETHSupportingFeeOnTransferTokens (uint256,uint256,address[],address,uint256)
017e7e58	=> feeTo ()
094b7415	=> feeToSetter ()
e6a43905	=> getPair (address,address)
1e3dd18b	=> allPairs (uint256)
574f2ba3	=> allPairsLength ()
c9c65396	=> createPair (address,address)
f46901ed	=> setFeeTo (address)
a2e74af6	=> setFeeToSetter (address)
bbe93d91	=> mul (int256,int256)
adefc37b	=> sub (int256,int256)
a5f3c23b	=> add (int256,int256)
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)
5bb79860 => toUint224(uint256)
809fdd33 => toUint128(uint256)
1cf887fc => toUint96(uint256)
2665fad0 => toUint64(uint256)
c8193255 => toUint32(uint256)
9374068f => toUint16(uint256)
0cc4681e => toUint8(uint256)
fdcf791b => toUint256(int256)
dd2a0316 => toInt128(int256)
d6bd32aa => toInt64(int256)
9c6f59be => toInt32(int256)
cf65b4d3 => toInt16(int256)
f136dc02 => toInt8(int256)
dfbe873b => toInt256(uint256)
119df25f => _msgSender()
8b49d47e => _msgData()
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)
30e0789e => _transfer(address,address,uint256)
4e6ec247 => _mint(address,uint256)
6161eb18 => _burn(address,uint256)
104e81ff => _approve(address,address,uint256)
cad3be83 => _beforeTokenTransfer(address,address,uint256)
8f811a1c => _afterTokenTransfer(address,address,uint256)
8da5cb5b => owner()
715018a6 => renounceOwnership()
f2fde38b => transferOwnership(address)
fc201122 => _setOwner(address)
268d8e2e => get(Map,address)
b45dad3d => getIndexOfKey(Map,address)
7596720f => getKeyAtIndex(Map,uint256)
b1b533f3 => size(Map)
6b06f325 => set(Map,address,uint256)
0eac8729 => remove(Map,address)
a8b9d240 => withdrawableDividendOf(address)
aafd847a => withdrawnDividendOf(address)
27ce0147 => accumulativeDividendOf(address)
91b89fba => dividendOf(address)
03c83302 => distributeDividends()
```

```
6a474002 => withdrawDividend()
373de4aa => _withdrawDividendOfUser(address)
ab86e0a6 => _setBalance(address,uint256)
31e79db0 => excludeFromDividends(address)
e98030c7 => updateClaimWait(uint256)
e7841ec0 => getLastProcessedIndex()
09bbedde => getNumberOfTokenHolders()
fbcbc0f1 => getAccount(address)
5183d6fd => getAccountAtIndex(uint256)
77fdb837 => canAutoClaim(uint256)
e30443bc => setBalance(address,uint256)
ffb2c479 => process(uint256)
bc4c4b37 => processAccount(address,bool)
91b32b32 => setSafeManager(address)
f3fef3a3 => withdraw(address,uint256)
127f4b2e => withdrawBNB(uint256)
fb201b1d => openTrade()
0b6bb6f5 => includeToWhiteList(address[])
a2a957bb => setFee(uint256,uint256,uint256,uint256)
03a280fc => setExtraFeeOnSell(uint256)
a68bacf3 => setMaxBuytx(uint256)
2bd61186 => setprojectDevWallet(address)
65b8dbc0 => updateUniswapV2Router(address)
c0246668 => excludeFromFees(address,bool)
5b89029c => setExcludeFromMaxTx(address,bool)
49928a50 => setExcludeFromAll(address)
c492f046 => excludeMultipleAccountsFromFees(address[],bool)
9a7a23d6 => setAutomatedMarketMakerPair(address,bool)
5290c250 => setSWapToensAtAmount(uint256)
a7f7b36f => _setAutomatedMarketMakerPair(address,bool)
871c128d => updateGasForProcessing(uint256)
a26579ad => getClaimWait()
30bb4cff => getTotalDividendsDistributed()
4fbee193 => isExcludedFromFees(address)
658c27a9 => isExcludedFromMaxTx(address)
6843cd84 => dividendTokenBalanceOf(address)
ad56c13c => getAccountDividendsInfo(address)
f27fd254 => getAccountDividendsInfoAtIndex(uint256)
700bb191 => processDividendTracker(uint256)
4e71d92d => claim()
64b0f653 => getNumberOfDividendTokenHolders()
c49b9a80 => setSwapAndLiquifyEnabled(bool)
173865ad => swapAndLiquify(uint256)
0de423da => swapTokensForBnb(uint256,address)
a30aa203 => swapAndSendBNBToMarketing(uint256)
9cd441da => addLiquidity(uint256,uint256)
```

Automatic general report

















































Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/Voltoken.sol	111367d8ac043642211ea2f54e034a9b7b306808

Contracts Description Table

Contract	Type	Bases	
:-----: :-----: :-----: :-----:			
L	**Function Name**	**Visibility**	**Mutability**
Modifiers			
IUniswapV2Router01 Interface			
L	factory	External !	NO !
L	WETH	External !	NO !
L	addLiquidity	External !	NO !
L	addLiquidityETH	External !	NO !
L	removeLiquidity	External !	NO !
L	removeLiquidityETH	External !	NO !
L	removeLiquidityWithPermit	External !	NO !
L	removeLiquidityETHWithPermit	External !	NO !
L	swapExactTokensForTokens	External !	NO !
L	swapTokensForExactTokens	External !	NO !
L	swapExactETHForTokens	External !	NO !
L	swapTokensForExactETH	External !	NO !
L	swapExactTokensForETH	External !	NO !
L	swapETHForExactTokens	External !	NO !
L	quote	External !	NO !
L	getAmountOut	External !	NO !
L	getAmountIn	External !	NO !
L	getAmountsOut	External !	NO !
L	getAmountsIn	External !	NO !
IUniswapV2Router02 Interface IUniswapV2Router01			
L	removeLiquidityETHSupportingFeeOnTransferTokens	External !	NO !
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External !	NO !
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External !	NO !
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External !	NO !
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External !	NO !
IUniswapV2Factory Interface			
L	feeTo	External !	NO !
L	feeToSetter	External !	NO !
L	getPair	External !	NO !
L	allPairs	External !	NO !
L	allPairsLength	External !	NO !
L	createPair	External !	NO !
L	setFeeTo	External !	NO !
L	setFeeToSetter	External !	NO !

```

| | | | | | |
| **SignedSafeMath** | Library | | |
| L | mul | Internal |  | | |
| L | div | Internal |  | | |
| L | sub | Internal |  | | |
| L | add | Internal |  | | |
| | | |
| **SafeMath** | Library | | |
| L | tryAdd | Internal |  | | |
| L | trySub | Internal |  | | |
| L | tryMul | Internal |  | | |
| L | tryDiv | Internal |  | | |
| L | tryMod | Internal |  | | |
| L | add | Internal |  | | |
| L | sub | Internal |  | | |
| L | mul | Internal |  | | |
| L | div | Internal |  | | |
| L | mod | Internal |  | | |
| L | sub | Internal |  | | |
| L | div | Internal |  | | |
| L | mod | Internal |  | | |
| | | |
| **SafeCast** | Library | | |
| L | toUint224 | Internal |  | | |
| L | toUint128 | Internal |  | | |
| L | toUint96 | Internal |  | | |
| L | toUint64 | Internal |  | | |
| L | toUint32 | Internal |  | | |
| L | toUint16 | Internal |  | | |
| L | toUint8 | Internal |  | | |
| L | toUint256 | Internal |  | | |
| L | toInt128 | Internal |  | | |
| L | toInt64 | Internal |  | | |
| L | toInt32 | Internal |  | | |
| L | toInt16 | Internal |  | | |
| L | toInt8 | Internal |  | | |
| L | toInt256 | Internal |  | | |
| | | |
| **Context** | Implementation | | |
| L | _msgSender | Internal |  | | |
| L | _msgData | Internal |  | | |
| | | |
| **IERC20** | Interface | | |
| L | totalSupply | External |  | NO! |
| L | balanceOf | External |  | NO! |
| L | transfer | External |   | NO! |
| L | allowance | External |  | NO! |
| L | approve | External |   | NO! |
| L | transferFrom | External |   | NO! |
| | | |
| **IERC20Metadata** | Interface | IERC20 | | |
| L | name | External |  | NO! |
| L | symbol | External |  | NO! |
| L | decimals | External |  | NO! |
| | | |
| **ERC20** | Implementation | Context, IERC20, IERC20Metadata | | |
| L | <Constructor> | Public |   | NO! |
| L | name | Public |  | NO! |

```

```

| L | symbol | Public ! | | NO! |
| L | decimals | Public ! | | NO! |
| L | totalSupply | Public ! | | NO! |
| L | balanceOf | Public ! | | NO! |
| L | transfer | Public ! | ● | NO! |
| L | allowance | Public ! | | NO! |
| L | approve | Public ! | ● | NO! |
| L | transferFrom | Public ! | ● | NO! |
| L | increaseAllowance | Public ! | ● | NO! |
| L | decreaseAllowance | Public ! | ● | NO! |
| L | _transfer | Internal ! | ● | |
| L | _mint | Internal ! | ● | |
| L | _burn | Internal ! | ● | |
| L | _approve | Internal ! | ● | |
| L | _beforeTokenTransfer | Internal ! | ● | |
| L | _afterTokenTransfer | Internal ! | ● | |
| | | | |
| **Ownable** | Implementation | Context | | |
| L | <Constructor> | Public ! | ● | NO! |
| L | owner | Public ! | | NO! |
| L | renounceOwnership | Public ! | ● | onlyOwner |
| L | transferOwnership | Public ! | ● | onlyOwner |
| L | _setOwner | Private ! | ● | |
| | | | |
| **IterableMapping** | Library | | | |
| L | get | Public ! | | NO! |
| L | getIndexOfKey | Public ! | | NO! |
| L | getKeyAtIndex | Public ! | | NO! |
| L | size | Public ! | | NO! |
| L | set | Public ! | ● | NO! |
| L | remove | Public ! | ● | NO! |
| | | | |
| **DividendPayingTokenOptionalInterface** | Interface | | | |
| L | withdrawableDividendOf | External ! | | NO! |
| L | withdrawnDividendOf | External ! | | NO! |
| L | accumulativeDividendOf | External ! | | NO! |
| | | | |
| **DividendPayingTokenInterface** | Interface | | | |
| L | dividendOf | External ! | | NO! |
| L | distributeDividends | External ! | ● | NO! |
| L | withdrawDividend | External ! | ● | NO! |
| | | | |
| **DividendPayingToken** | Implementation | ERC20, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface | | |
| L | <Constructor> | Public ! | ● | ERC20 |
| L | <Receive Ether> | External ! | ● | NO! |
| L | distributeDividends | Public ! | ● | NO! |
| L | withdrawDividend | Public ! | ● | NO! |
| L | _withdrawDividendOfUser | Internal ! | ● | |
| L | dividendOf | Public ! | | NO! |
| L | withdrawableDividendOf | Public ! | | NO! |
| L | withdrawnDividendOf | Public ! | | NO! |
| L | accumulativeDividendOf | Public ! | | NO! |
| L | _transfer | Internal ! | ● | |
| L | _mint | Internal ! | ● | |
| L | _burn | Internal ! | ● | |
| L | _setBalance | Internal ! | ● | |
| | | | |

```



```

| **VLTDividendTracker** | Implementation | DividendPayingToken, Ownable |||
| L | <Constructor> | Public ! |  | DividendPayingToken |
| L | _transfer | Internal  | | |
| L | withdrawDividend | Public ! | | NO! |
| L | excludeFromDividends | External ! |  | onlyOwner |
| L | updateClaimWait | External ! |  | onlyOwner |
| L | getLastProcessedIndex | External ! | | NO! |
| L | getNumberOfTokenHolders | External ! | | NO! |
| L | getAccount | Public ! | | NO! |
| L | getAccountAtIndex | Public ! | | NO! |
| L | canAutoClaim | Private  | | |
| L | setBalance | External ! |  | onlyOwner |
| L | process | Public ! |  | NO! |
| L | processAccount | Public ! |  | onlyOwner |
| | | |
| **SafeToken** | Implementation | Ownable |||
| L | <Constructor> | Public ! |  | NO! |
| L | setSafeManager | Public ! |  | onlyOwner |
| L | withdraw | External ! |  | NO! |
| L | withdrawBNB | External ! |  | NO! |
| | | |
| **LockToken** | Implementation | Ownable |||
| L | <Constructor> | Public ! |  | NO! |
| L | openTrade | External ! |  | onlyOwner |
| L | includeToWhiteList | External ! |  | onlyOwner |
| | | |
| **VLT** | Implementation | ERC20, Ownable, SafeToken, LockToken |||
| L | setFee | Public ! |  | onlyOwner |
| L | setExtraFeeOnSell | Public ! |  | onlyOwner |
| L | setMaxBuytx | Public ! |  | onlyOwner |
| L | setprojectDevWallet | Public ! |  | onlyOwner |
| L | <Constructor> | Public ! |  | ERC20 |
| L | <Receive Ether> | External ! |  | NO! |
| L | updateUniswapV2Router | Public ! |  | onlyOwner |
| L | excludeFromFees | Public ! |  | onlyOwner |
| L | setExcludeFromMaxTx | Public ! |  | onlyOwner |
| L | setExcludeFromAll | Public ! |  | onlyOwner |
| L | excludeMultipleAccountsFromFees | Public ! |  | onlyOwner |
| L | setAutomatedMarketMakerPair | Public ! |  | onlyOwner |
| L | setSWAPToensAtAmount | Public ! |  | onlyOwner |
| L | _setAutomatedMarketMakerPair | Private  |  |
| L | updateGasForProcessing | Public ! |  | onlyOwner |
| L | updateClaimWait | External ! |  | onlyOwner |
| L | getClaimWait | External ! | | NO! |
| L | getTotalDividendsDistributed | External ! | | NO! |
| L | isExcludedFromFees | Public ! | | NO! |
| L | isExcludedFromMaxTx | Public ! | | NO! |
| L | withdrawableDividendOf | Public ! | | NO! |
| L | dividendTokenBalanceOf | Public ! | | NO! |
| L | getAccountDividendsInfo | External ! | | NO! |
| L | getAccountDividendsInfoAtIndex | External ! | | NO! |
| L | processDividendTracker | External ! |  | NO! |
| L | claim | External ! |  | NO! |
| L | getLastProcessedIndex | External ! | | NO! |
| L | getNumberOfDividendTokenHolders | External ! | | NO! |
| L | excludeFromDividends | External ! |  | onlyOwner |
| L | setSwapAndLiquifyEnabled | Public ! |  | onlyOwner |
| L | _transfer | Internal  |  | open |

```

	L		swapAndLiquify		Private					lockTheSwap	
	L		swapTokensForBnb		Private						
	L		swapAndSendBNBToMarketing		Private						
	L		addLiquidity		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.

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.