

# Smart Contract Security Audit V1

## BLIZZ Token

<https://www.blizztoken.com/>

9/12/2021



<https://saferico.com/>

[business@saferico.com](mailto:business@saferico.com)

[https://t.me/SFI\\_ANN](https://t.me/SFI_ANN)

—

# Table of Contents

## **Table of Contents**

### **Background**

### **Project Information**

- Token Information
- BLIZZ Token Distribution
- Contract Interaction Details
- 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

- **Website:** <https://www.blizztoken.com/>
- **Telegram group:** <https://t.me/blizztoken>
- **Twitter:** <https://mobile.twitter.com/BlizzToken>
- **Instagram:** <https://www.instagram.com/blizztoken>
- **Facebook:** <https://www.facebook.com/profile.php?id=100075105965033>
- **Reddit:** [https://www.reddit.com/u/Blizztoken?utm\\_medium=android\\_app&utm\\_source=share](https://www.reddit.com/u/Blizztoken?utm_medium=android_app&utm_source=share)
- **Discord:** <https://discord.gg/>
- **Platform:** Polygon Network
- **Contract Address:** 0x6548eCaE23A269F90d080F4Df9431e5E84BaF8B3

## Token Information

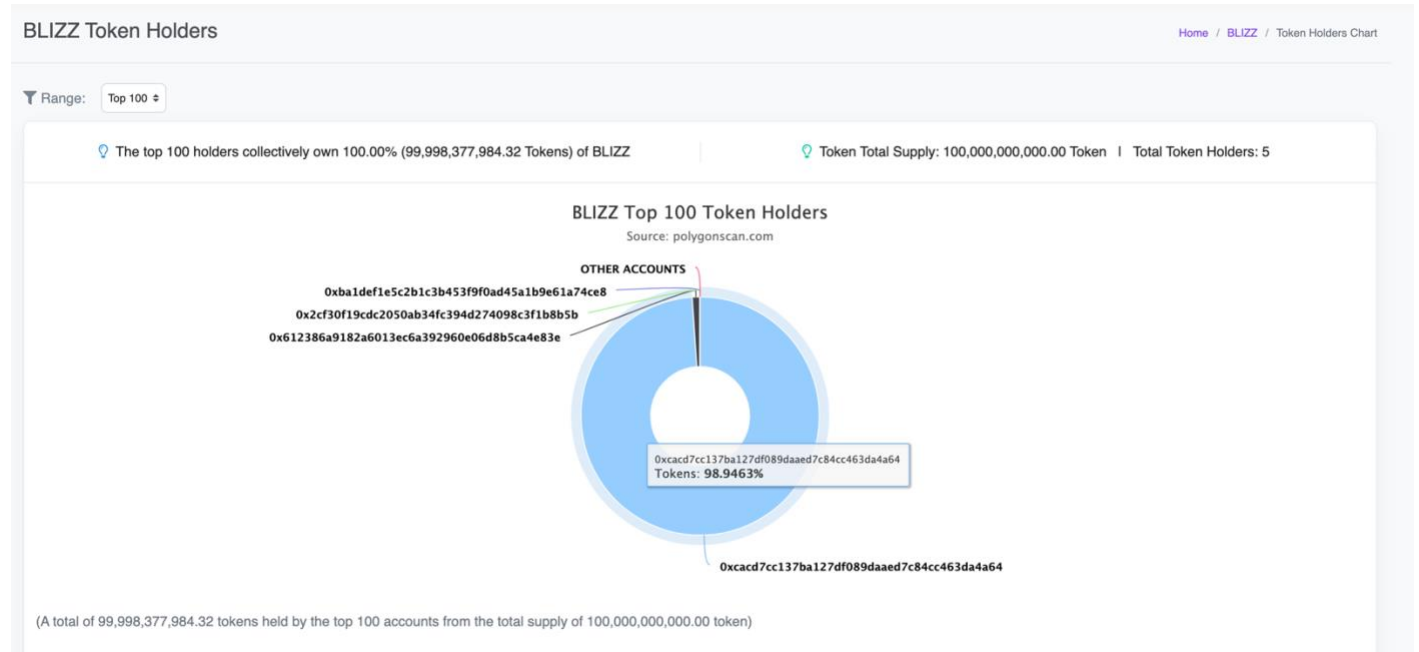
- **Name:** BLZ
- **Total Supply:** 100,000,000,000
- **Holders:** 5 address
- **Total transactions:** 10

Contracts address deployed to test net (Polygon)

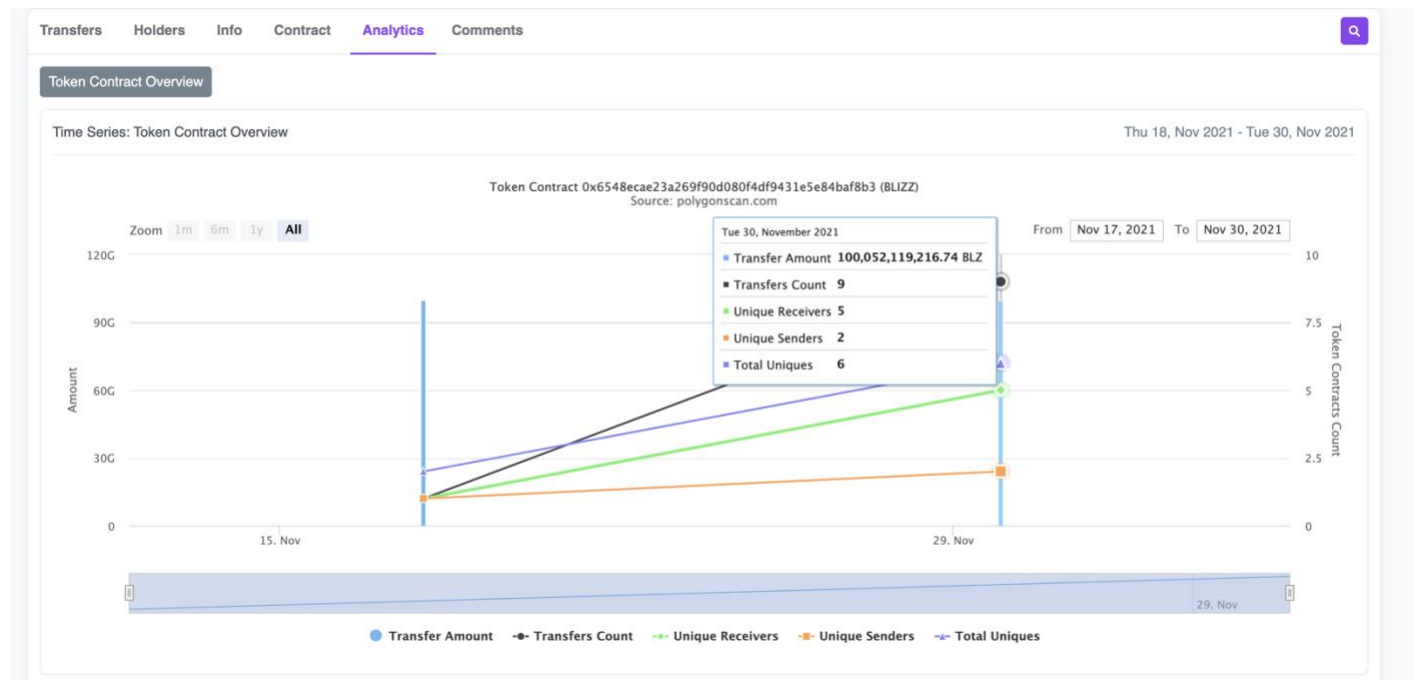
BLIZZ token contract on munbai.polygon (Polygon Test Net)

<https://mumbai.polygonscan.com/address/0x6ba4a3826dea64a4953a3f4bd4b4ce7f72f33711>

# BLZ Token Distribution



# Contract Interaction Details



## Executive Summary

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

Well Secured	
<b>Secured</b>	✓
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, 0 low, 1 very low-level issues and 1 note in all solidity files of the contract

The files:

BLIZZ.sol

# File and Function Level Report

## File in Scope:

Contract Name	SHA 256 hash	Contract Address
BLIZZ.sol	7e76938b0c52f9417c29290d3403e2350aae60359e1dcae91044a325ec6cff80	0x6548eCaE23A269F90d080F4Df9431e5E84BaF8B3

- Contract: BLIZZ
- 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
geUnlockTime	✓	Read / public	Passed
swapAndLiquifyEnabled	✓	Read / public	Passed
developmentWallet	✓	Read / public	Passed
reflectionFromToken	✓	Read / public	Passed
isExcludedFromReward	✓	Read / public	Passed

tokenFromReflection	✓	Read / public	Passed
_developmentFee	✓	Read / private	Passed
_burnFee	✓	Read / private	Passed
_maxTxAmount	✓	Read / private	Passed
_taxFee	✓	Read / private	Passed
isExcludedFromFees	✓	Read / public	Passed
_liquidityFee	✓	Read / private	Passed
_marketingFee	✓	Read / private	Passed
totalFees	✓	Read / public	Passed
uniswapV2Pair	✓	Read / public	Passed
uniswapV2Router	✓	Read / public	Passed
unLock	✓	Write / 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
setAllFee	✓	Write / public	Passed
decreaseAllowance	✓	Write / public	Passed
enableAllFees	✓	Write / public	Passed
lock	✓	Write / public	Passed
disableAllFees	✓	Write / public	Passed
setMaxTxAmount	✓	Write / public	Passed

setSwapAndLiquifyEnabled	✓	Write / public	Passed
increaseAllowance	✓	Write / public	Passed
setdevelopmentWallet	✓	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
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:**

No Low severity vulnerabilities were found.

### **Very Low:**

#### **Issue #1. Constant/View/Pure functions:**

SafeMath.sub(uint256,uint256) : Is constant but potentially should not be. Note: Modifiers are currently not considered by this static analysis.

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

### **Notes:**

#### **#Note1**

#### **#ERC20:**

In detail

ERC20 contract's "decimals" function should have "uint8" as return type.

```
function decimals() external pure returns  
(uint8);
```

# Automatic Testing

## 1- Check for security

7e76938b0c52f9417c29290d3403e2350aae60359e1dcae91044a325ec6cff80

File: BLIZZ.sol | Language: solidity | Size: 44639 bytes | Date: 2021-12-09T05:58:24.026Z

Critical	High	Medium	Low	Note
0	0	0	0	1

## 2- SOLIDITY STATIC ANALYSIS

SOLIDITY STATIC ANALYSIS

☒ Select all ☒ Autorun Run

**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

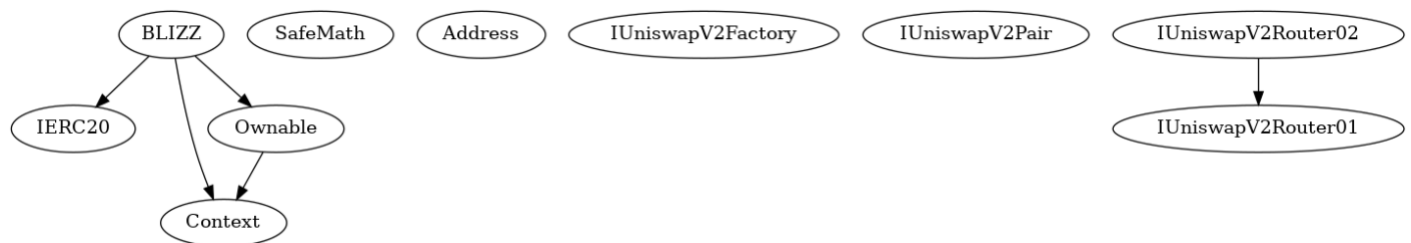
- ☒ **ERC20:**  
'decimals' should be 'uint8'

**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:

CreateGenerateHow to use... Run Stop

☒ Select all

☒ tests/BLIZZ\_test.sol

Progress: 1 finished (of 1)

**PASS** testSuite (tests/BLIZZ\_test.sol)

✓ Before all



✓ Check success



✓ Check success2



✓ Check sender and value



**Result for tests/BLIZZ\_test.sol**

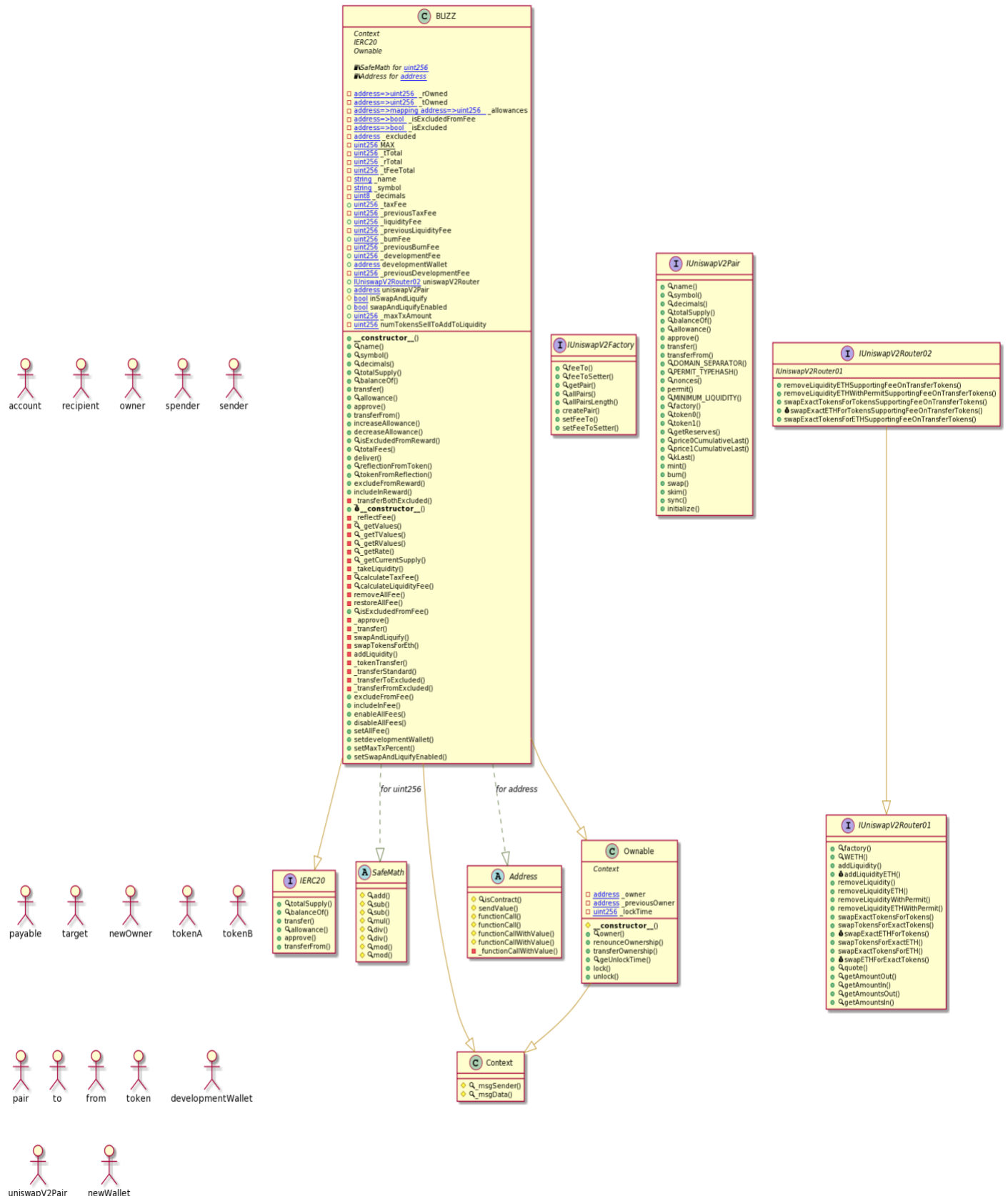
Passing: 4

Total time: 0.32s

5- Call graph



# Unified Modeling Language (UML)



## Functions signature

```
11902160 => _getTValues(uint256)
16279055 => isContract(address)
39509351 => increaseAllowance(address,uint256)
75128141 => calculateTaxFee(uint256)
18160ddd => totalSupply()
70a08231 => balanceOf(address)
a9059cbb => transfer(address,uint256)
dd62ed3e => allowance(address,address)
095ea7b3 => approve(address,uint256)
23b872dd => transferFrom(address,address,uint256)
771602f7 => add(uint256,uint256)
b67d77c5 => sub(uint256,uint256)
e31bdc0a => sub(uint256,uint256,string)
c8a4ac9c => mul(uint256,uint256)
a391c15b => div(uint256,uint256)
b745d336 => div(uint256,uint256,string)
f43f523a => mod(uint256,uint256)
71af23e8 => mod(uint256,uint256,string)
119df25f => _msgSender()
8b49d47e => _msgData()
24a084df => sendValue(address,uint256)
a0b5ffb0 => functionCall(address,bytes)
241b5886 => functionCall(address,bytes,string)
2a011594 => functionCallWithValue(address,bytes,uint256)
d525ab8a => functionCallWithValue(address,bytes,uint256,string)
36455e42 => _functionCallWithValue(address,bytes,uint256,string)
8da5cb5b => owner()
715018a6 => renounceOwnership()
f2fde38b => transferOwnership(address)
b6c52324 => getUnlockTime()
dd467064 => lock(uint256)
a69df4b5 => unlock()
017e7e58 => feeTo()
094b7415 => feeToSetter()
e6a43905 => getPair(address,address)
1e3dd18b => allPairs(uint256)
574f2ba3 => allPairsLength()
c9c65396 => createPair(address,address)
f46901ed => setFeeTo(address)
a2e74af6 => setFeeToSetter(address)
06fdde03 => name()
95d89b41 => symbol()
313ce567 => decimals()
3644e515 => DOMAIN_SEPARATOR()
30adf81f => PERMIT_TYPEHASH()
7ecebe00 => nonces(address)
d505accf => permit(address,address,uint256,uint256,uint8,bytes32,bytes32)
ba9a7a56 => MINIMUM_LIQUIDITY()
c45a0155 => factory()
0dfe1681 => token0()
d21220a7 => token1()
0902f1ac => getReserves()
5909c0d5 => price0CumulativeLast()
5a3d5493 => price1CumulativeLast()
```



```

7464fc3d => kLast()
6a627842 => mint(address)
89afcb44 => burn(address)
022c0d9f => swap(uint256,uint256,address,bytes)
bc25cf77 => skim(address)
fff6cae9 => sync()
485cc955 => initialize(address,address)
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,bool,uint8,bytes32,bytes32)
ded9382a =>
removeLiquidityETHWithPermit(address,uint256,uint256,uint256,address,uint256,bool,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,bool,uint8,bytes32,bytes32)
5c11d795 =>
swapExactTokensForTokensSupportingFeeOnTransferTokens(uint256,uint256,address[],address,uint256)
b6f9de95 =>
swapExactETHForTokensSupportingFeeOnTransferTokens(uint256,address[],address,uint256)
791ac947 =>
swapExactTokensForETHSupportingFeeOnTransferTokens(uint256,uint256,address[],address,uint256)
a457c2d7 => decreaseAllowance(address,uint256)
88f82020 => isExcludedFromReward(address)
13114a9d => totalFees()
3bd5d173 => deliver(uint256)
4549b039 => reflectionFromToken(uint256,bool)
2d838119 => tokenFromReflection(uint256)
52390c02 => excludeFromReward(address)
3685d419 => includeInReward(address)
6ff6cdf4 => _transferBothExcluded(address,address,uint256)
184d894e => _reflectFee(uint256,uint256)
d4780e36 => _getValues(uint256)
1d5671e4 => _getRValues(uint256,uint256,uint256,uint256)
94e10784 => _getRate()

```

```
97a9d560 => _getCurrentSupply()
c432df5e => _takeLiquidity(uint256)
cc126a23 => calculateLiquidityFee(uint256)
301370af => removeAllFee()
e7e3e3a7 => restoreAllFee()
5342acb4 => isExcludedFromFee(address)
104e81ff => _approve(address,address,uint256)
30e0789e => _transfer(address,address,uint256)
173865ad => swapAndLiquify(uint256)
b28805f4 => swapTokensForEth(uint256)
9cd441da => addLiquidity(uint256,uint256)
f147aa74 => _tokenTransfer(address,address,uint256)
2852df65 => _transferStandard(address,address,uint256)
16f1cc83 => _transferToExcluded(address,address,uint256)
c7d9be66 => _transferFromExcluded(address,address,uint256)
437823ec => excludeFromFee(address)
ea2f0b37 => includeInFee(address)
ffc78635 => enableAllFees()
741af87f => disableAllFees()
830d7e45 => setAllFee(uint256,uint256,uint256,uint256)
4830aa17 => setdevelopmentWallet(address)
d543dbeb => setMaxTxPercent(uint256)
c49b9a80 => setSwapAndLiquifyEnabled(bool)
```

# Automatic general report

## Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/BLIZZ.sol	88d61016496708bc115f973f01d702ca2c441748

## Contracts Description Table

Contract	Type	Bases	
:	:	:	:
:	:	:	:
:	:	:	:
L	**Function Name**	**Visibility**	**Mutability**
**Modifiers**			
**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 !
**SafeMath**	Library		
L	add	Internal	
L	sub	Internal	
L	sub	Internal	
L	mul	Internal	
L	div	Internal	
L	div	Internal	
L	mod	Internal	
L	mod	Internal	
**Context**	Implementation		
L	_msgSender	Internal	
L	_msgData	Internal	
**Address**	Library		
L	isContract	Internal	
L	sendValue	Internal	
L	functionCall	Internal	
L	functionCall	Internal	
L	functionCallWithValue	Internal	
L	functionCallWithValue	Internal	
L	_functionCallWithValue	Private	
**Ownable**	Implementation	Context	
L	<Constructor>	Internal	
L	owner	Public !	NO !
L	renounceOwnership	Public !	onlyOwner
L	transferOwnership	Public !	onlyOwner
L	geUnlockTime	Public !	NO !
L	lock	Public !	onlyOwner
L	unlock	Public !	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 |
| | | |
| **IUniswapV2Pair** | Interface | | |
| L | name | External | | NO |
| L | symbol | External | | NO |
| L | decimals | External | | NO |
| L | totalSupply | External | | NO |
| L | balanceOf | External | | NO |
| L | allowance | External | | NO |
| L | approve | External | | NO |
| L | transfer | External | | NO |
| L | transferFrom | External | | NO |
| L | DOMAIN_SEPARATOR | External | | NO |
| L | PERMIT_TYPEHASH | External | | NO |
| L | nonces | External | | NO |
| L | permit | External | | NO |
| L | MINIMUM_LIQUIDITY | External | | NO |
| L | factory | External | | NO |
| L | token0 | External | | NO |
| L | token1 | External | | NO |
| L | getReserves | External | | NO |
| L | price0CumulativeLast | External | | NO |
| L | price1CumulativeLast | External | | NO |
| L | kLast | External | | NO |
| L | mint | External | | NO |
| L | burn | External | | NO |
| L | swap | External | | NO |
| L | skim | External | | NO |
| L | sync | External | | NO |
| L | initialize | External | | NO |
| | | |
| **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! |
| NO! |
| L | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ! | ⬛ | NO! |
|
| L | swapExactETHForTokensSupportingFeeOnTransferTokens | External ! | ⬛ | NO! |
| L | swapExactTokensForETHSupportingFeeOnTransferTokens | External ! | ⬛ | NO! |
| | | |
| **BLIZZ** | Implementation | Context, IERC20, Ownable | | |
| 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 | isExcludedFromReward | Public ! | | NO! |
| L | totalFees | Public ! | | NO! |
| L | deliver | Public ! | ⬛ | NO! |
| L | reflectionFromToken | Public ! | | NO! |
| L | tokenFromReflection | Public ! | | NO! |
| L | excludeFromReward | Public ! | ⬛ | onlyOwner |
| L | includeInReward | External ! | ⬛ | onlyOwner |
| L | _transferBothExcluded | Private ! | ⬛ |
| L | <Receive Ether> | External ! | ⬛ | NO! |
| L | _reflectFee | Private ! | ⬛ |
| L | _getValues | Private ! | |
| L | _getTValues | Private ! | |
| L | _getRValues | Private ! | |
| L | _getRate | Private ! | |
| L | _getCurrentSupply | Private ! | |
| L | _takeLiquidity | Private ! | ⬛ |
| L | calculateTaxFee | Private ! | |
| L | calculateLiquidityFee | Private ! | |
| L | removeAllFee | Private ! | ⬛ |
| L | restoreAllFee | Private ! | ⬛ |
| L | isExcludedFromFee | Public ! | | NO! |
| L | _approve | Private ! | ⬛ |
| L | _transfer | Private ! | ⬛ |
| L | swapAndLiquify | Private ! | ⬛ | lockTheSwap |
| L | swapTokensForEth | Private ! | ⬛ |
| L | addLiquidity | Private ! | ⬛ |
| L | _tokenTransfer | Private ! | ⬛ |
| L | _transferStandard | Private ! | ⬛ |
| L | _transferToExcluded | Private ! | ⬛ |
| L | _transferFromExcluded | Private ! | ⬛ |
| L | excludeFromFee | Public ! | ⬛ | onlyOwner |
| L | includeInFee | Public ! | ⬛ | onlyOwner |
| L | enableAllFees | External ! | ⬛ | onlyOwner |

```

L	disableAllFees	External	!		onlyOwner
L	setAllFee	Public	!		onlyOwner
L	setdevelopmentWallet	External	!		onlyOwner
L	setMaxTxPercent	External	!		onlyOwner
L	setSwapAndLiquifyEnabled	Public	!		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 “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.