Smart Contract Security Audit V1

HAHAMETAVERSE

https://hahametaverse.com/

28/11/2021

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://hahametaverse.com/
- **Telegram group:** https://t.me/HAHAMETAVERSE
- **WhitePaper**: https://hahametaverse.com/whitepaper
- **Platform**: Binance Smart Chain
- **Contract Address**: 0x88009456cefc36Ad0f729316840C5Def886FbbA8

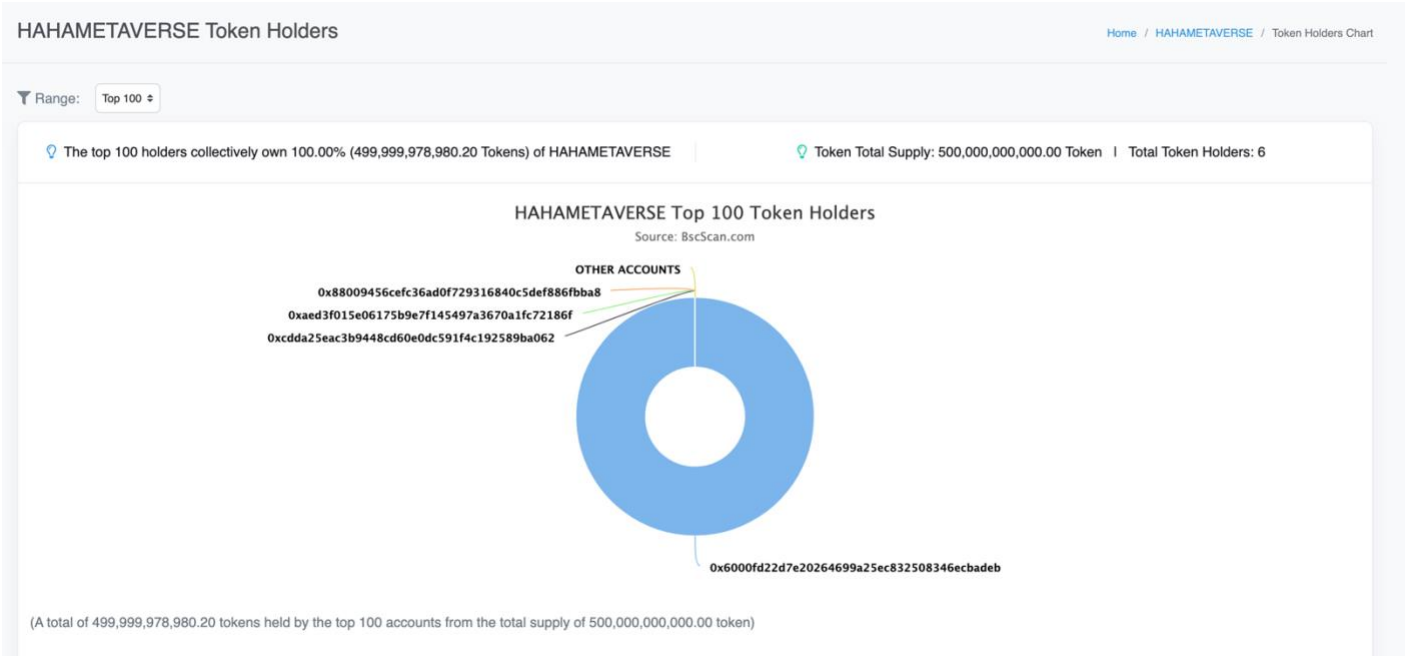
Token Information

- Name: HAHA
- Total Supply: 500,000,000,000
- Holders: 6 address
- Total transactions: 17

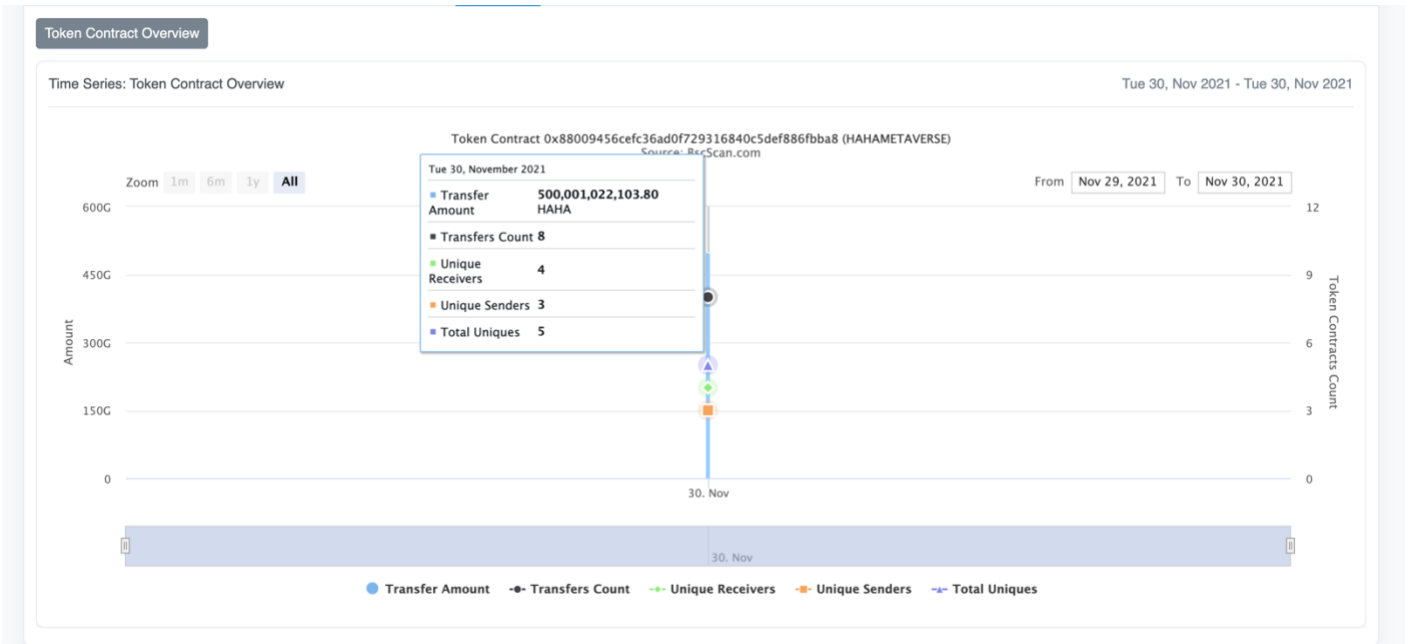
Contracts address deployed to test net (BSC)

HAHAMETAVERSE token (HAHA)contract on testnet.bsc (BSC Test Net)
https://testnet.bscscan.com/address/0x7481e0e68f6a4f9a7cddf4025942f6e09fe506a2#code

HAHA Token Distribution



Contract Interaction Details



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

The files:

HAHAMETAVERSE .sol

File and Function Level Report

File in Scope:

Contract Name	SHA 256 hash	Contract Address
HAHAMETAVERSE .sol	**520cb21ee62a1aa1f76e5c210d1657564746e28227a9067b9ccee140c17e1f91**	0x88009456cefc36Ad0f729316840C5Def886FbbA8

- Contract: HAHAMETAVERSE
- 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**
_airaddress	✔	Read / private	**Passed**
sale	✔	Read / public	**Passed**

deadWallet	✔	Read / public	**Passed**
swapAndLiquifyEnabled	✔	Read / public	**Passed**
rewards	✔	Read / public	**Passed**
reflectionFromToken	✔	Read / public	**Passed**
rate	✔	Read / public	**Passed**
isExcludedFromReward	✔	Read / public	**Passed**
airdrop	✔	Read / public	**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**
tokenFromReflection	✔	Read / public	**Passed**
approve	✔	Write / public	**Passed**
transferFrom	✔	Write / public	**Passed**
blacklist	✔	Write / public	**Passed**
transfer	✔	Write / public	**Passed**
airdropTokens	✔	Write / public	**Passed**
buyTokens	✔	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**
removeFromBlacklist	✔	Write / public	**Passed**
setLiquiditFeePercent	✔	Write / public	**Passed**
setMaxAmount	✔	Write / public	**Passed**
setDrop	✔	Write / public	**Passed**
Sale_start	✔	Write / public	**Passed**
Sale_stop	✔	Write / public	**Passed**
setCharityFeePercent	✔	Write / public	**Passed**
setMaxTxAmount	✔	Write / public	**Passed**
setPrice	✔	Write / public	**Passed**
setSwapAndLiquifyEnabled	✔	Write / public	**Passed**
increaseAllowance	✔	Write / public	**Passed**
decreaseAllowance	✔	Write / public	**Passed**
setTaxFeePercent	✔	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:

IERC20.transfer(address,uint256), transferFrom(address,uint256)and approve(address,uint256) : Potentially should be constant/view/pure but is not. Note: Modifiers are currently not considered by this static analysis.

```
function transfer(address recipient, uint256 amount) external returns (bool);
function transferFrom(address sender, address recipient, uint256 amount) external returns (bool);
function approve(address spender, uint256 amount) external returns (bool);
```

Notes:

#Note1

#ERC20:

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

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

Automatic Testing

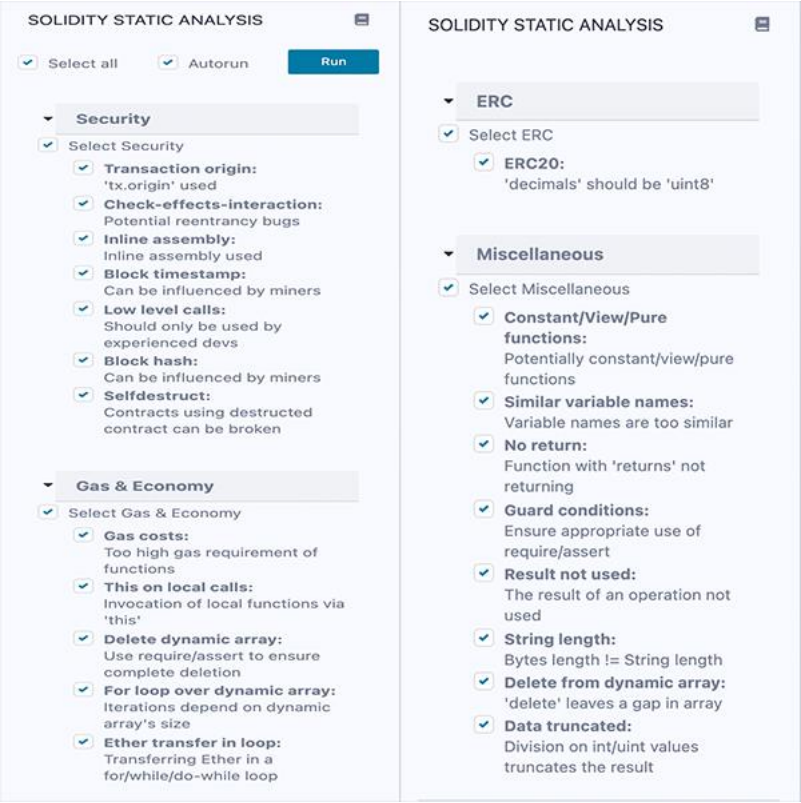
1- Check for security

520cb21ee62a1aa1f76e5c210d1657564746e28227a9067b9ccee140c17e1f91

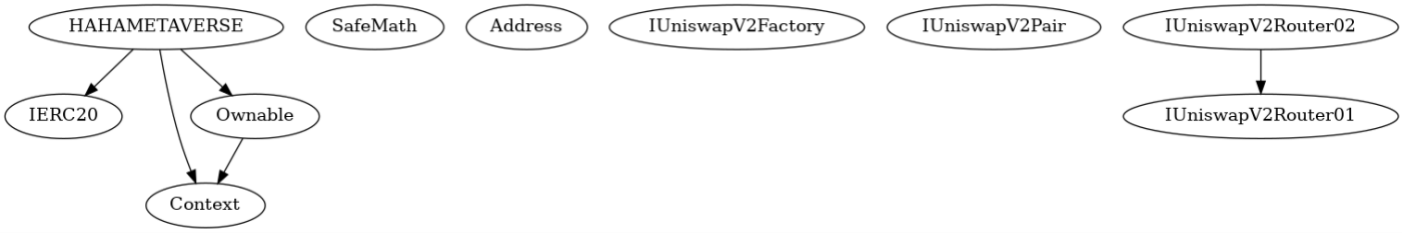
File: HAHA... | Language: solidity | Size: 49694 bytes | Date: 2021-11-28T04:55:33.425Z

Critical	High	Medium	Low	Note
0	0	0	0	1

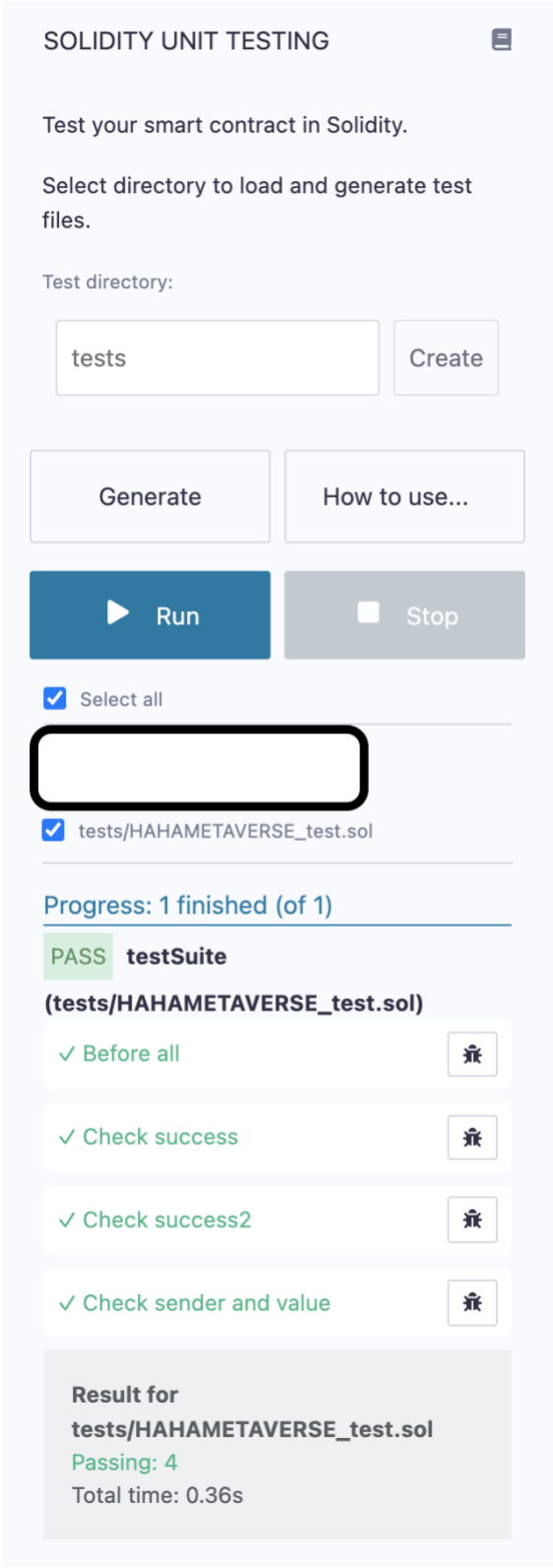
2- SOLIDITY STATIC ANALYSIS



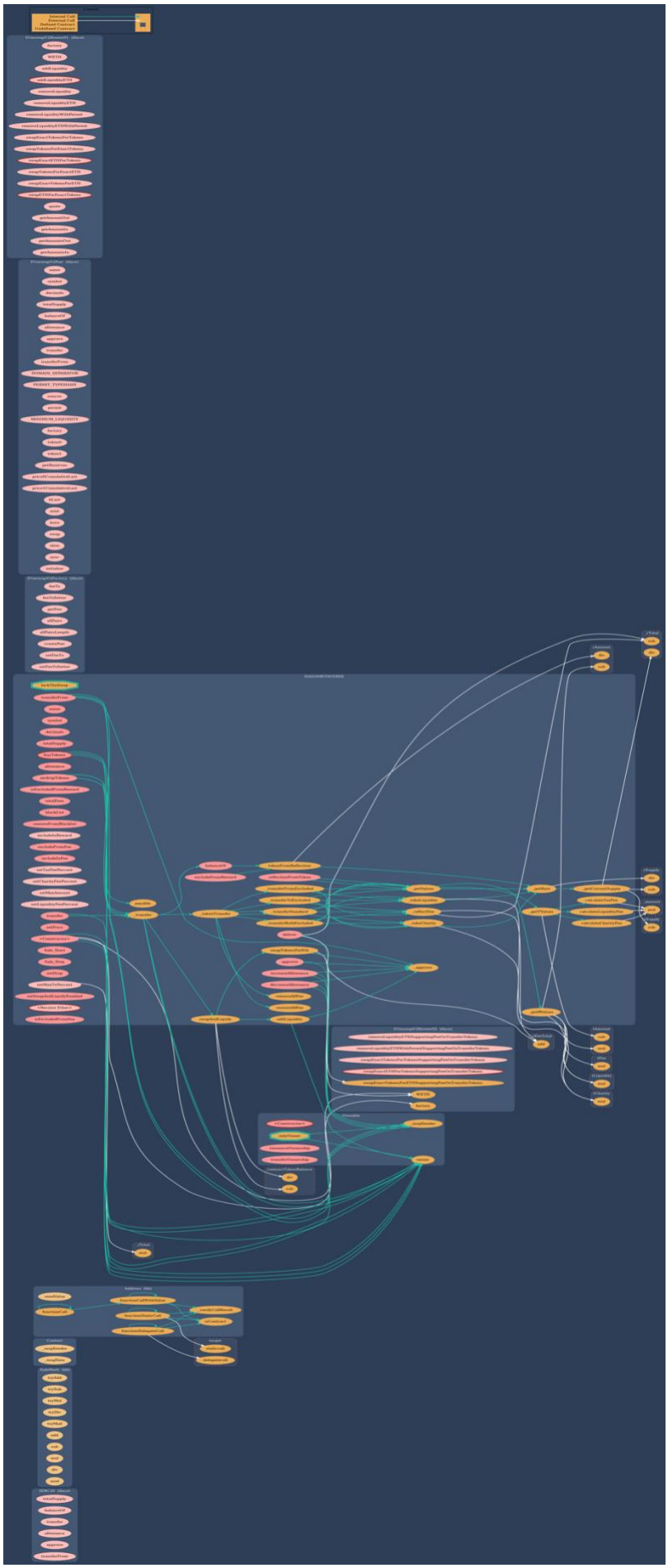
3- Inheritance graph



4- SOLIDITY UNIT TESTING



5- Call graph



Automatic general report

Files Description Table

File Name	SHA-1 Hash
/Users/macbook/Desktop/smart contracts/HAHAMETAVERSE.sol	e62461bf435d12eca83dd238dfe17f5d9a6c6323

Contracts Description Table

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
IERC20	Interface			
└	totalSupply	External ❗	NO❗	
└	balanceOf	External ❗	NO❗	
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗	NO❗	
└	approve	External ❗	🛑	NO❗
└	transferFrom	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 🔒		
Address	Library			
└	isContract	Internal 🔒		
└	sendValue	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCall	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionCallWithValue	Internal 🔒	🛑	
└	functionStaticCall	Internal 🔒		
└	functionStaticCall	Internal 🔒		
└	functionDelegateCall	Internal 🔒	🛑	
└	functionDelegateCall	Internal 🔒	🛑	
└	_verifyCallResult	Private 🔐		

```
|||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗ |  💵 |NO❗ |
| └ | owner | Public ❗ |  |NO❗ |
| └ | renounceOwnership | Public ❗ | 🛑  | onlyOwner |
| └ | transferOwnership | Public ❗ | 🛑  | onlyOwner |
|||||
| **IUniswapV2Factory** | Interface |  |||
| └ | feeTo | External ❗ |  |NO❗ |
| └ | feeToSetter | External ❗ |  |NO❗ |
| └ | getPair | External ❗ |  |NO❗ |
| └ | allPairs | External ❗ |  |NO❗ |
| └ | allPairsLength | External ❗ |  |NO❗ |
| └ | createPair | External ❗ | 🛑  |NO❗ |
| └ | setFeeTo | External ❗ | 🛑  |NO❗ |
| └ | setFeeToSetter | External ❗ | 🛑  |NO❗ |
|||||
| **IUniswapV2Pair** | Interface |  |||
| └ | name | External ❗ |  |NO❗ |
| └ | symbol | External ❗ |  |NO❗ |
| └ | decimals | External ❗ |  |NO❗ |
| └ | totalSupply | External ❗ |  |NO❗ |
| └ | balanceOf | External ❗ |  |NO❗ |
| └ | allowance | External ❗ |  |NO❗ |
| └ | approve | External ❗ | 🛑  |NO❗ |
| └ | transfer | External ❗ | 🛑  |NO❗ |
| └ | transferFrom | External ❗ | 🛑  |NO❗ |
| └ | DOMAIN_SEPARATOR | External ❗ |  |NO❗ |
| └ | PERMIT_TYPEHASH | External ❗ |  |NO❗ |
| └ | nonces | External ❗ |  |NO❗ |
| └ | permit | External ❗ | 🛑  |NO❗ |
| └ | MINIMUM_LIQUIDITY | External ❗ |  |NO❗ |
| └ | factory | External ❗ |  |NO❗ |
| └ | token0 | External ❗ |  |NO❗ |
| └ | token1 | External ❗ |  |NO❗ |
| └ | getReserves | External ❗ |  |NO❗ |
| └ | price0CumulativeLast | External ❗ |  |NO❗ |
| └ | price1CumulativeLast | External ❗ |  |NO❗ |
| └ | kLast | External ❗ |  |NO❗ |
| └ | mint | External ❗ | 🛑  |NO❗ |
| └ | burn | External ❗ | 🛑  |NO❗ |
| └ | swap | External ❗ | 🛑  |NO❗ |
| └ | skim | External ❗ | 🛑  |NO❗ |
| └ | sync | External ❗ | 🛑  |NO❗ |
| └ | initialize | External ❗ | 🛑  |NO❗ |
|||||
| **IUniswapV2Router01** | Interface |  |||
| └ | factory | External ❗ |  |NO❗ |
| └ | WETH | External ❗ |  |NO❗ |
| └ | addLiquidity | External ❗ | 🛑  |NO❗ |
| └ | addLiquidityETH | External ❗ |  💵 |NO❗ |
| └ | removeLiquidity | External ❗ | 🛑  |NO❗ |
| └ | removeLiquidityETH | External ❗ | 🛑  |NO❗ |
| └ | removeLiquidityWithPermit | External ❗ | 🛑  |NO❗ |
| └ | removeLiquidityETHWithPermit | External ❗ | 🛑  |NO❗ |
| └ | swapExactTokensForTokens | External ❗ | 🛑  |NO❗ |
| └ | swapTokensForExactTokens | External ❗ | 🛑  |NO❗ |
| └ | swapExactETHForTokens | External ❗ |  💵 |NO❗ |
```

└	swapTokensForExactETH	External 🔒	🛑	NO❗
└	swapExactTokensForETH	External 🔒	🛑	NO❗
└	swapETHForExactTokens	External 🔒	💵	NO❗
└	quote	External 🔒		NO❗
└	getAmountOut	External 🔒		NO❗
└	getAmountIn	External 🔒		NO❗
└	getAmountsOut	External 🔒		NO❗
└	getAmountsIn	External 🔒		NO❗
IUniswapV2Router02	Interface	IUniswapV2Router01		
└	removeLiquidityETHSupportingFeeOnTransferTokens	External 🔒	🛑	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External 🔒	🛑	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External 🔒	🛑	NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External 🔒	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External 🔒	🛑	NO❗
HAHAMETAVERSE	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❗
└	isExcludedFromReward	Public 🔒		NO❗
└	totalFees	Public 🔒		NO❗
└	blackList	Public 🔒	🛑	onlyOwner
└	removeFromBlacklist	Public 🔒	🛑	onlyOwner
└	deliver	Public 🔒	🛑	NO❗
└	reflectionFromToken	Public 🔒		NO❗
└	tokenFromReflection	Public 🔒		NO❗
└	excludeFromReward	Public 🔒	🛑	onlyOwner
└	includeInReward	External 🔒	🛑	onlyOwner
└	_transferBothExcluded	Private 🔐	🛑	
└	excludeFromFee	Public 🔒	🛑	onlyOwner
└	includeInFee	Public 🔒	🛑	onlyOwner
└	setTaxFeePercent	External 🔒	🛑	onlyOwner
└	setCharityFeePercent	External 🔒	🛑	onlyOwner
└	setMaxAmount	External 🔒	🛑	onlyOwner
└	setLiquidityFeePercent	External 🔒	🛑	onlyOwner
└	setMaxTxPercent	External 🔒	🛑	onlyOwner
└	setPrice	Public 🔒	🛑	onlyOwner
└	buyTokens	Public 🔒	💵	NO❗
└	Sale_Start	Public 🔒	🛑	onlyOwner
└	Sale_Stop	Public 🔒	🛑	onlyOwner
└	setDrop	Public 🔒	🛑	onlyOwner
└	airdropTokens	Public 🔒	🛑	NO❗
└	setSwapAndLiquifyEnabled	Public 🔒	🛑	onlyOwner
└	<Receive Ether>	External 🔒	💵	NO❗
└	_reflectFee	Private 🔐	🛑	
└	_getValues	Private 🔐		

└	_getTValues	Private 🔐		
└	_getRValues	Private 🔐		
└	_getRate	Private 🔐		
└	_getCurrentSupply	Private 🔐		
└	_takeLiquidity	Private 🔐	🛑	
└	_takeCharity	Private 🔐	🛑	
└	calculateTaxFee	Private 🔐		
└	calculateCharityFee	Private 🔐		
└	calculateLiquidityFee	Private 🔐		
└	removeAllFee	Private 🔐	🛑	
└	restoreAllFee	Private 🔐	🛑	
└	isExcludedFromFee	Public ❗		NO❗
└	_approve	Private 🔐	🛑	
└	_transfer	Private 🔐	🛑	
└	swapAndLiquify	Private 🔐	🛑	lockTheSwap
└	swapTokensForEth	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	_tokenTransfer	Private 🔐	🛑	
└	_transferStandard	Private 🔐	🛑	
└	_transferToExcluded	Private 🔐	🛑	
└	_transferFromExcluded	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 “Well-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.