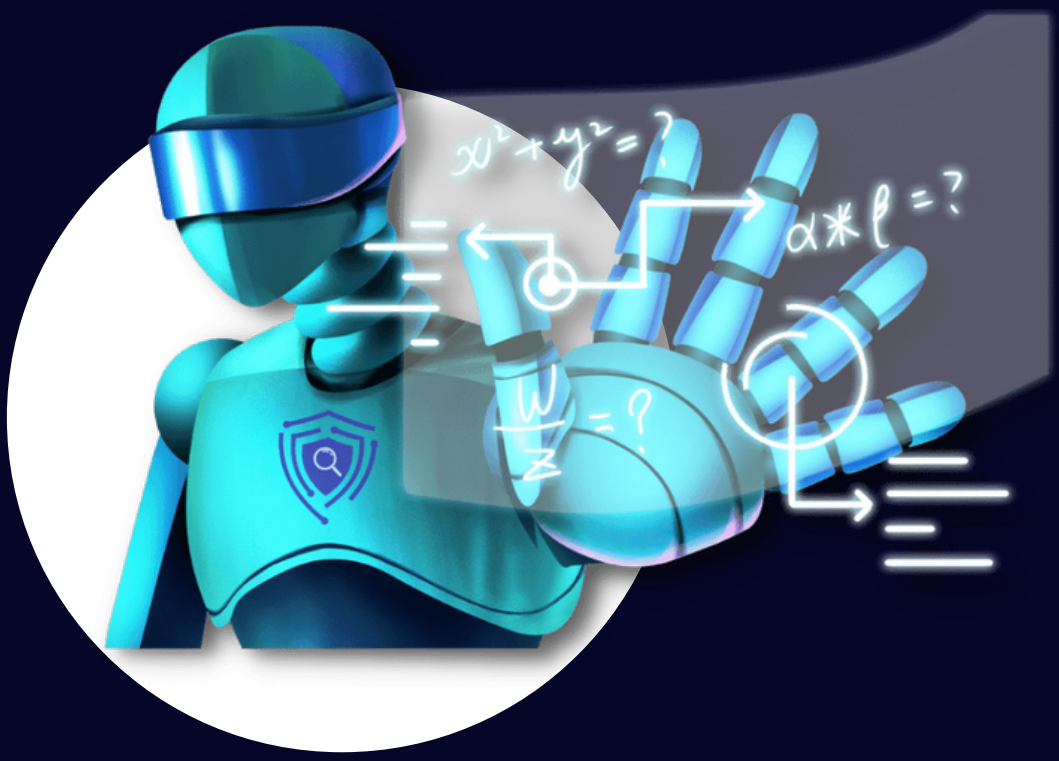
SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY

SAFUAUDIT

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY



PROJECT: BULLSTEP

DATE: July 11, 2022

BULLSTEP
EVERY STEP COUNTS

www.safuaudit.com

INTRODUCTION

Client	BullStep ($BSTEP)
Language	Solidity
Contract address	0xd999B5144860e14c11460De64D00c10c66a57cA7
Owner	0x377A214e3b40e1CC0Cba0494B40B9A03822c4DA0
Deployer	0x377A214e3b40e1CC0Cba0494B40B9A03822c4DA0
SHA1-Hash	5b31b7bc244ead4dd90437d8c13fa65f0222b662
Decimals	9
Supply	10,000,000,000
Platform	Binance Smart Chain
Compiler	v0.8.11+commit.d7f03943
Optimization	Yes with 200 runs
Website	https://bullstep.net/
Telegram	https://t.me/bullstepofficial
Twitter	https://twitter.com/BullStep_App

TABLE OF CONTENTS

APPROACH

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

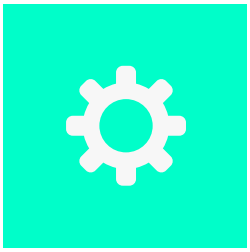
Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability



Tools

- Remix IDE
- Mythril
- Open Zeppelin Code Analyzer
- Solidity Code Complier
- Hardhat

RISK CLASSIFICATION

CRITICAL

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

MEDIUM

Issues on this level could potentially bring problems and should eventually be fixed.

MINOR

Issues on this level are minor details and warning that can remain unfixed but would be better fixed at some point in the future

INFORMATIONAL

Information level is to offer suggestions for improvement of efficacity or security for features with a risk free factor.

OVERVIEW

Fees

- Buy Fees: 10%
- Sell Fees: 10%

Fees privileges

- Can't set fees over 25%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount

- Can't set max Tx amount lower than 1%

Pause function

- Can't pause trading

Blacklist

- Can't blacklist

Other privileges

- Can include/exclude from fees

CONTRACT INSPECTION 🔎

Imported contracts or frameworks used:

Context	Implementation				
IERC20	Interface				
IFactoryV2	Interface				
IV2Pair	Interface				
IRouter01	Interface				
IRouter02	Interface	IRouter01			
AntiSnipe	Interface				
BullStep	Implementation	Context, IERC20			

Tested Contract File:

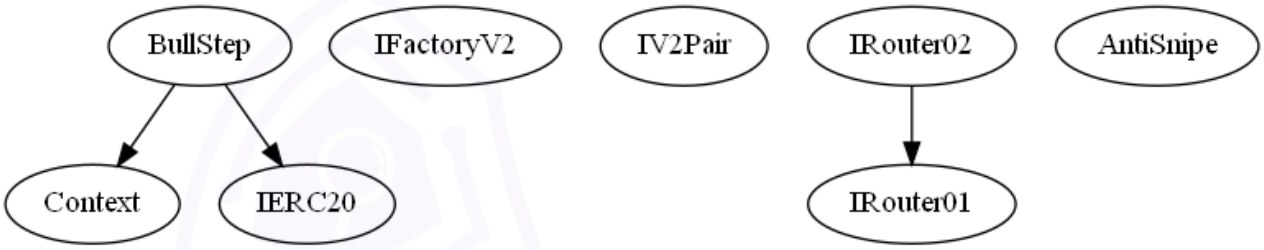
File Name	SHA-1 Hash
BullStep.sol	5b31b7bc244ead4dd90437d8c13fa65f0222b662

BullStep	Implementation	Context, IERC20			
└	<Constructor>	Public ❗	🟩	NO❗	
└	<Receive Ether>	External ❗	🟩	NO❗	
└	withdrawBUSD	Public ❗	🛑		onlyOwner
└	owner	Public ❗		NO❗	
└	transferOwner	External ❗	🛑		onlyOwner
└	renounceOwnership	Public ❗	🛑		onlyOwner
└	totalSupply	External ❗		NO❗	
└	decimals	External ❗		NO❗	
└	symbol	External ❗		NO❗	
└	name	External ❗		NO❗	
└	getOwner	External ❗		NO❗	
└	allowance	External ❗		NO❗	
└	balanceOf	Public ❗		NO❗	
└	transfer	Public ❗	🛑	NO❗	
└	approve	Public ❗	🛑	NO❗	
└	_approve	Private 🔐	🛑		
└	approveContractContingency	Public ❗	🛑		onlyOwner
└	transferFrom	External ❗	🛑	NO❗	
└	increaseAllowance	Public ❗	🛑	NO❗	
└	decreaseAllowance	Public ❗	🛑	NO❗	
└	setNewRouter	Public ❗	🛑		onlyOwner
└	setLpPair	External ❗	🛑		onlyOwner
└	changeRouterContingency	External ❗	🛑		onlyOwner
└	getCirculatingSupply	Public ❗		NO❗	
└	isExcludedFromFees	Public ❗		NO❗	
└	setExcludedFromFees	Public ❗	🛑		onlyOwner

└	setInitializer	External ❗	🛑	onlyOwner
└	removeBlacklisted	External ❗	🛑	onlyOwner
└	isBlacklisted	Public ❗		NO❗
└	getSniperAmt	Public ❗		NO❗
└	removeSniper	External ❗	🛑	onlyOwner
└	setProtectionSettings	External ❗	🛑	onlyOwner
└	setGasPriceLimit	External ❗	🛑	onlyOwner
└	setTaxes	External ❗	🛑	onlyOwner
└	setRatios	External ❗	🛑	onlyOwner
└	setMaxTxPercent	External ❗	🛑	onlyOwner
└	getMaxTX	Public ❗		NO❗
└	setSwapSettings	External ❗	🛑	onlyOwner
└	setWallets	External ❗	🛑	onlyOwner
└	setContractSwapEnabled	Public ❗	🛑	onlyOwner
└	excludePresaleAddresses	External ❗	🛑	onlyOwner
└	_hasLimits	Private 🔐		
└	_transfer	Internal 🔒	🛑	
└	cTokens	Public ❗	🛑	onlyOwner
└	a_checkCBalance	Public ❗	🛑	onlyOwner
└	contractSwap	Private 🔐	🛑	lockTheSwap
└	_checkLiquidityAdd	Private 🔐	🛑	
└	enableTrading	Public ❗	🛑	onlyOwner
└	sweepContingency	External ❗	🛑	onlyOwner
└	multiSendTokens	External ❗	🛑	NO❗
└	multiSendPercents	External ❗	🛑	NO❗
└	takeTaxes	Internal 🔒	🛑	
└	_finalizeTransfer	Private 🔐	🛑	

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable
🔐	Private function
🔒	Internal function
NO❗	Function has no modifier

INHERITANCE TREE



Inheritance is a feature of the object-oriented programming language. It is a way of extending the functionality of a program, used to separate the code, reduces the dependency, and increases the re-usability of the existing code. Solidity supports inheritance between smart contracts, where multiple contracts can be inherited into a single contract.

MANUAL FUNCTIONS ANALYSIS

The contract is verified to check if functions do and work as they should and malicious code is not inserted.

	Tested	Result
Transfer	Yes	Passed
Total Supply	Yes	Passed
Buy Back	Yes	N/A
Burn	Yes	N/A
Mint	Yes	N/A
Rebase	Yes	N/A
Pause	Yes	N/A
Blacklist	Yes	N/A
Lock	Yes	N/A
Max Transaction	Yes	Passed
Transfer Ownership	Yes	Passed
Renounce Ownership	Yes	Passed

VULNERABILITIES TEST

ID	Description	
V-01	Function Default Visibility	Passed
V-02	Integer Overflow and Underflow	Passed
V-03	Outdated Compiler Version	Passed
V-04	Floating Pragma	Minor
V-05	Unchecked Call Return Value	Passed
V-06	Unprotected Ether Withdrawal	Passed
V-07	Unprotected SELF-DESTRUCT Instruction	Passed
V-08	Re-entrancy	Passed
V-09	State Variable Default Visibility	Minor
V-10	Uninitialized Storage Pointer	Passed
V-11	Assert Violation	Passed
V-12	Use of Deprecated Solidity Functions	Passed
V-13	Delegate Call to Untrusted Callee	Passed
V-14	DoS with Failed Call	Passed
V-15	Transaction Order Dependence	Passed
V-16	Authorization through tx.origin	Passed
V-17	Block values as a proxy for time	Passed

V-18	Signature Malleability	**Passed**
V-19	Incorrect Constructor Name	**Passed**
V-20	Shadowing State Variables	**Passed**
V-21	Weak Sources of Randomness from Chain Attributes	**Passed**
V-22	Missing Protection against Signature Replay Attacks	**Passed**
V-23	Lack of Proper Signature Verification	**Passed**
V-24	Requirement Violation	**Passed**
V-25	Write to Arbitrary Storage Location	**Passed**
V-26	Incorrect Inheritance Order	**Passed**
V-27	Insufficient Gas Griefing	**Passed**
V-28	Arbitrary Jump with Function Type Variable	**Passed**
V-29	DoS With Block Gas Limit	**Passed**
V-30	Typographical Error	**Passed**
V-31	Right-To-Left-Override control character (U+202E)	**Passed**
V-32	Presence of unused variables	**Informational**
V-33	Unexpected Ether balance	**Passed**
V-34	Hash Collisions With Multiple Variable Length Arguments	**Passed**
V-35	Message call with the hardcoded gas amount	**Passed**
V-36	Code With No Effects (Irrelevant/Dead Code)	**Informational**
V-37	Unencrypted Private Data On-Chain	**Passed**

FINDINGS

ID	Category	Issue	Severity
CE-OF	Centralization	Owner Accessible Functions	Minor
V-01	Vulnerabilities	Unchecked transfer	Minor
V-02	Vulnerabilities	Unlocked Compiler	Minor
V-03	Vulnerabilities	State Variable Default Visibility	Minor
CS-01	Coding Standards	Unused State Variable	Informational
CS-02	Coding Standards	Dead Code	Informational
GO-03	Gas Optimization	State Variables that could be declared constant	Informational
GO-04	Gas Optimization	Public Function could be Declared External	Informational

CE-OF: Owner Accessible Functions

Description

The owner has the permission through onlyOwner modifier to the following:

1. withdrawBUSD()
2. transferOwner()
3. renounceOwnership()
4. setNewRouter()
5. setLpPair()
6. changeRouterContingency()
7. setExcludedFromFees()
8. setInitializer()
9. removeBlacklisted()
10. removeSniper()
11. setProtectionSettings()
12. setProtectionSettings()
13. setGasPriceLimit()
14. setTaxes()
15. setRatios()
16. setMaxTxPercent()
17. setSwapSettings()
18. setWallets()
19. setContractSwapEnabled()
20. excludePresaleAddresses()
21. cTokens()
22. enableTrading()
23. sweepContingency()

The role OnlyOwner has authority over the above functions that can manipulate the project functionality without restrictions. Any compromise to the owner account may allow a hacker to take advantage of this authority.

Recommendation

- We advise the client to carefully manage the privilege accounts' private key to avoid any potential risks of being hacked.
- Renounce Ownership at some point in time.

V-01: Unchecked transfer

Line #370

```
function withdrawBUSD(address tAddress, uint amount, uint tDecimals) public onlyOwner
{
    token= IERC20(tAddress);
    token.transfer(msg.sender,amount*10**tDecimals);
}
```

Description

Function BullStep.withdrawBUSD() (Line #370) does not check the value returned by token.transfer().
Some tokens don't revert in case of failure. In case one of these tokens needs to be withdrawn from contract and transfer fails (exceeding amount, gas, etc.), it will not revert, but return false.

Recommendation

- We recommend using SafeERC20 library for transfers or add multiple checks on token.transfer().

V-02: Unlocked Compiler

Line #6

```
pragma solidity >=0.6.0 <0.9.0;
```

Description

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Recommendation

- Lock the pragma version and also consider known bugs (https://github.com/ethereum/solidity/releases) for the compiler version that is chosen.

V-03: State Variable Default Visibility

Line #222

```
mapping (address => bool) lpPairs;
```

Line #225

```
IERC20 token;
```

Line #301

```
bool inSwap;
```

Line #313

```
AntiSnipe antiSnipe;
```

Description

Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.

Recommendation

Variables can be specified as being public, internal or private. Explicitly define visibility for all state variables.

CS-01: Unused State Variable

Line #229

```
address[] private _excluded;
```

Description

Unused state variable.

Recommendation

- Remove unused state variables for code clarity and easier review.

CS-02: Dead Code

Line #696 - 707

```
function _checkLiquidityAdd(address from, address to) private {
    require(!_hasLiqBeenAdded, "Liquidity already added and marked.");
    if (!_hasLimits(from, to) && to == lpPair) {
        _liquidityHolders[from] = true;
        _hasLiqBeenAdded = true;
        if(address(antiSnipe) == address(0)){
            antiSnipe = AntiSnipe(address(this));
        }
        contractSwapEnabled = true;
        emit ContractSwapEnabledUpdated(true);
    }
}
```

Description

_checkLiquidityAdd(from ,to) is not used in the contract, and makes the review more difficult.

Recommendation

- Remove unused functions

GO-01: State Variables could be declared constant

Line #232, 235, 239

```
bool private allowedPresaleExclusion = true;
uint256 private startingSupply = 10_000_000_000;
uint8 private _decimals = 9;
```

Description

The above variables should be declared constant. This is especially for the **_tTotal, swapThreshold** and **swapAmount** pre-construction variables that are set afterwards.

Recommendation

- Add the constant attributes to state variables that never change.

GO-02: Public Function could be Declared External

Line #370, 399, 416, 421, 434, 447, 452, 457, 493, 511, 515, 556, 572, 641, 646, 709

Description

The above functions are public functions that are never called by the contract should be declared external to save gas.

Recommendation

- Use the external attribute for functions never called from the contract.

GOOD PRACTICES

- The owner cannot mint new tokens after deployment
- The owner cannot set taxes above 25%
- The owner cannot stop or pause the contract
- The owner cannot set a transaction limit under 1%

WEBSITE

Website	https://bullstep.net/
Domain Registry	http://www.namecheap.com
Domain Expiry Date	2023-06-10
Response Code	200
SSL Checker and HTTPS Test	**Passed**
Deprecated HTML tags	**Passed**
Robots.txt	**Passed**
Sitemap Test	**Passed**
SEO Friendly URL	**Informative**
Responsive Test	**Passed**
JS Error Test	**Passed**
Console Errors Test	**Passed**
Site Loading Speed Test	**2.29 seconds - Passed**
HTTP2 Test	**Passed**
Safe Browsing Test	**Passed**

DISCLAIMER

SafuAudit.com is not a financial institution and the information provided on this website does not constitute investment advice, financial advice, trading advice, or any other sort of advice. You should not treat any of the website's content as such. Investing in crypto assets carries a high level of risk and does not hold guarantees for not sustaining financial loss due to their volatility.

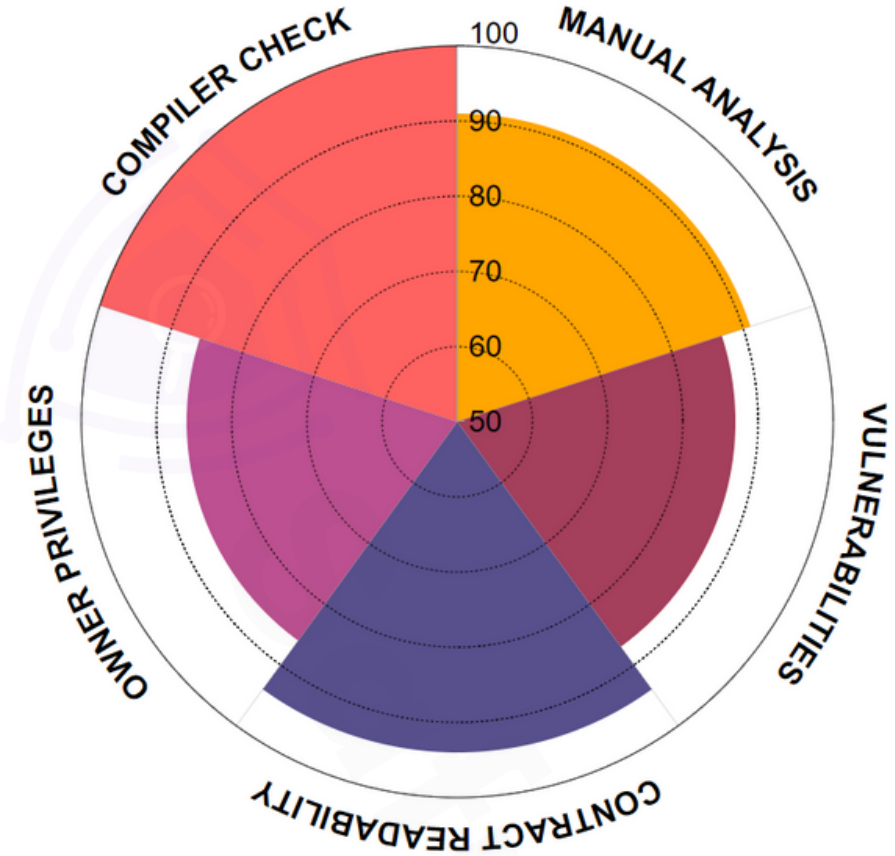
Accuracy of Information

SafuAudit will strive to ensure the accuracy of the information listed on this website although it will not hold any responsibility for any missing or wrong information. SafuAudit provides all information as is. You understand that you are using any and all information available here at your own risk. Any use or reliance on our content and services is solely at your own risk and discretion.

The purpose of the audit is to analyze the on-chain smart contract source code and to provide a basic overview of the project.

While we have used all the information available to us for this straightforward investigation, you should not rely on this report only — we recommend proceeding with several independent audits Be aware that smart contracts deployed on a blockchain aren't secured enough against external vulnerability or a hack. Be aware that active smart contract owner privileges constitute an elevated impact on the smart contract safety and security. Therefore, SafuAudit does not guarantee the explicit security of the audited smart contract. 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.

RATING



Manual Analysis

Vulnerabilities

Contract Readability

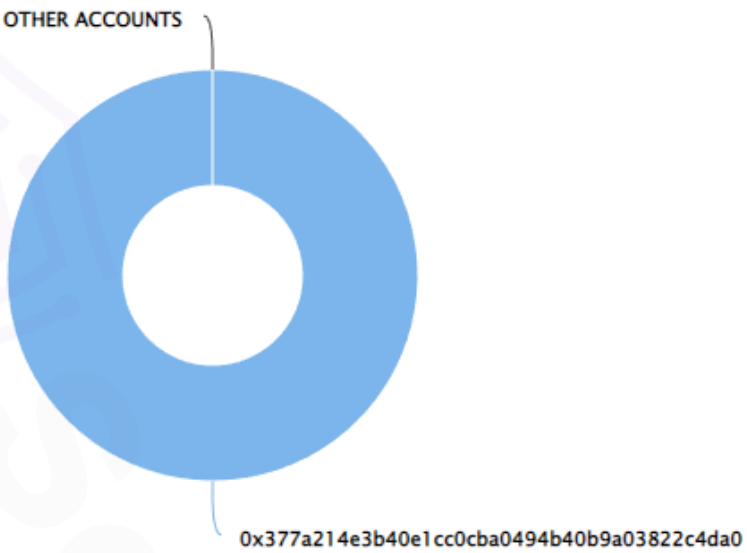
Owner Privileges

Compiler Check

Final Score: **91.6**

SUMMARY

Top 10 holders



Rank	Address	Quantity (Token)	Percentage
1	0x377a214e3b40e1cc0cba0494b40b9a03822c4da0	10,000,000,000	100.0000%

CONCLUSION

Project BullStep does not contain any severe issues or risk characteristics.

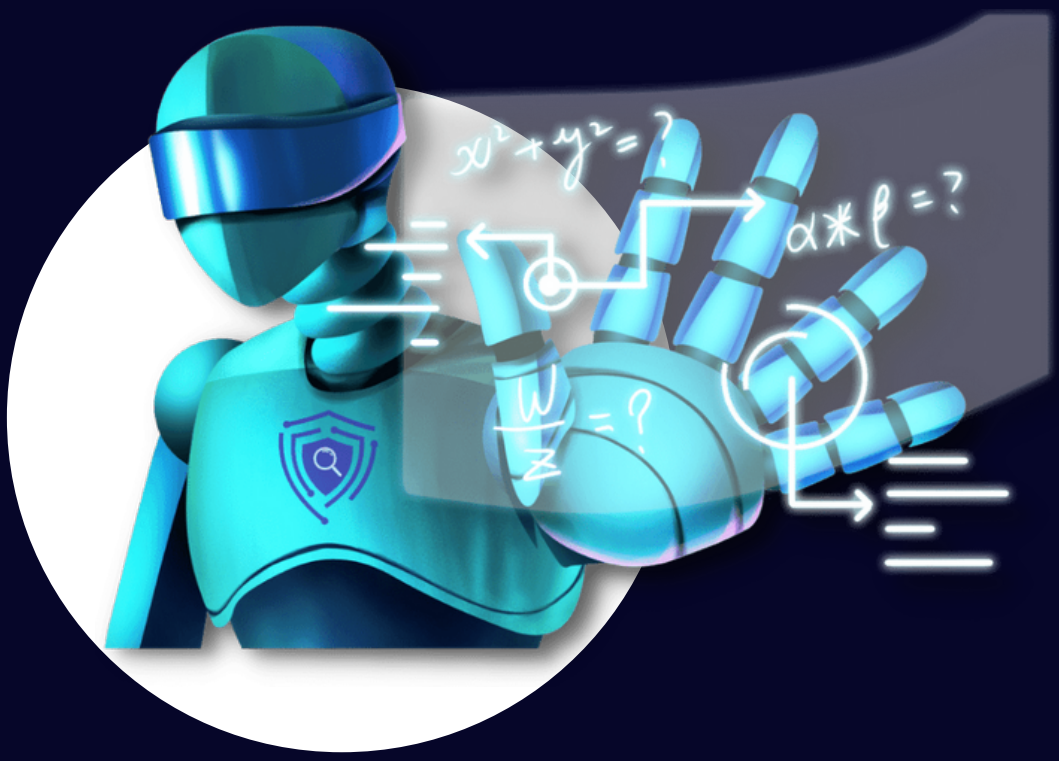
SafuAudit has tested the security based on manual and automated tests.

Please note that we don't offer any warranties for business model.

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY

SAFUAUDIT

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY



"Only in growth, reform, and change, paradoxically enough, is true security to be found."

www.safuaudit.com