Cyberscope

Audit Report

JOJO

Aug 2023

Network	BSC
Address	0xb4303e22cb305008efee6a26218f6f376fa4cf9a
Audited by	© cyberscope

Analysis

🔴 Critical 🟠 Medium ⚪ Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

🔴 Critical 🟠 Medium ⚪ Minor / Informative

Severity	Code	Description	Status
⚪	PDIF	Potential DAO Interaction Failure	Unresolved
⚪	RVA	Redundant Variable Assignment	Unresolved
⚪	IDI	Immutable Declaration Improvement	Unresolved

Table of Contents

Review

Contract Name	BEP20Standard
Compiler Version	v0.5.16+commit.9c3226ce
Optimization	200 runs
Explorer	https://bscscan.com/address/0xb4303e22cb305008efee6a26218f6f376fa4cf9a
Address	0xb4303e22cb305008efee6a26218f6f376fa4cf9a
Network	BSC
Symbol	JOJO
Decimals	9
Total Supply	21,000,000,000

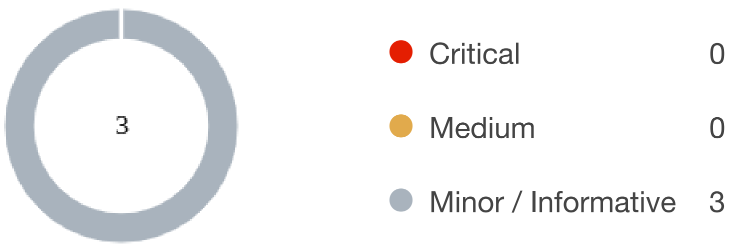
Audit Updates

Initial Audit	02 Aug 2023

Source Files

Filename	SHA256
BEP20Standard.sol	a56f74bbe7d2f2e0cc0edd7e770afdb107d382edea9fc85b0a5371053ce9462b

Findings Breakdown



Severity		Unresolved	Acknowledged	Resolved	Other
●	Critical	0	0	0	0
●	Medium	0	0	0	0
●	Minor / Informative	3	0	0	0

PDIF - Potential DAO Interaction Failure

Criticality	Minor / Informative
Location	BEP20Standard.sol#L183
Status	Unresolved

Description

The contract implements an `approve` function which sets an allowance for a given spender. However, the function contains a `require` statement that requires the `amount` to be approved is either zero or the user has not previously set an allowance for the specified `spender`. This design can introduce complications when interacting with other Decentralized Autonomous Organizations (DAOs), such as presale contracts. Specifically, if a user has already executed an approve action and later attempts to modify the approved amount, the function will revert due to the aforementioned requirement. This implementation can hinder the ability to adjust allowances dynamically, potentially leading to failed interactions with other DAOs.

```
    function approve(address spender, uint256 amount) public returns
(bool) {
        require(amount == 0 || _allowances[_msgSender()][spender] == 0,
"BEP20: approve non-zero allowance");
        _approve(_msgSender(), spender, amount);
        return true;
}
```

Recommendation

It is recommended to modify the `approve` function to allow users to adjust their allowances without the restrictive condition. This can be achieved by removing or altering the `require` statement. By doing so, the contract will offer greater flexibility and compatibility, ensuring smoother interactions with other DAOs and systems that might need dynamic allowance adjustments.

RVA - Redundant Variable Assignment

Criticality	Minor / Informative
Location	BEP20Standard.sol#L136,143
Status	Unresolved

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

The contract initializes the private variable `_maxSupply` within its constructor function. However this variable is neither utilized in any of the contract's functions nor accessed externally. Consequently, `_maxSupply` serves no functional purpose within the contract and is redundant.

```
   uint256 private _maxSupply;

    constructor() public {
        ...
        _maxSupply = 21000000000 * (10 ** uint256(_decimals));
        ...
    }
```

Recommendation

The team is advised to take these segments into consideration and rewrite them so the runtime will be more performant. That way it will improve the efficiency and performance of the source code and reduce the cost of executing it.

It is recommended to remove the `_maxSupply` variable from the contract to simplify the codebase and reduce potential areas of confusion.

IDI - Immutable Declaration Improvement

Criticality	Minor / Informative
Location	BEP20Standard.sol#L139,140,141,142,143
Status	Unresolved

Description

The contract declares state variables that their value is initialized once in the constructor and are not modified afterwards. The `immutable` is a special declaration for this kind of state variables that saves gas when it is defined.

```
_name
_symbol
_decimals
_totalSupply
_maxSupply
```

Recommendation

By declaring a variable as immutable, the Solidity compiler is able to make certain optimizations. This can reduce the amount of storage and computation required by the contract, and make it more gas-efficient.

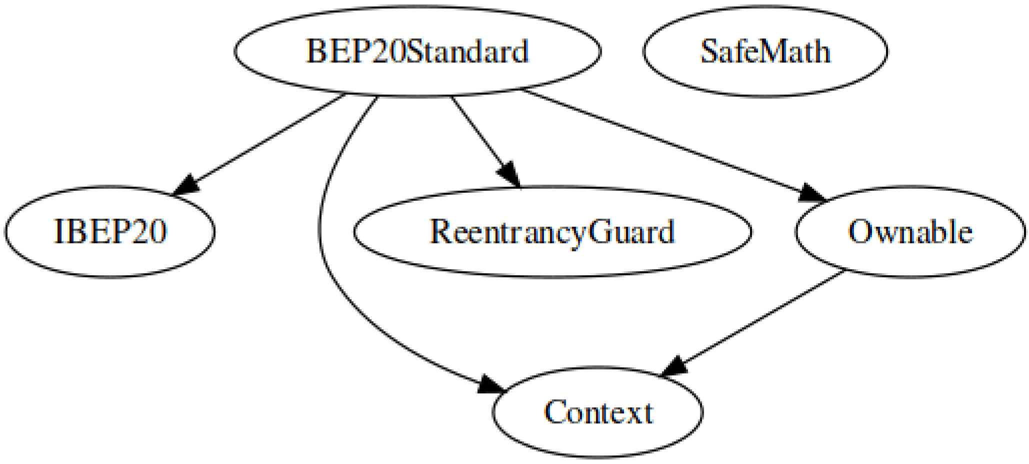
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
IBEP20	Interface			
	totalSupply	External		-
	decimals	External		-
	symbol	External		-
	name	External		-
	getOwner	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Context	Implementation			
		Internal	✓	
	_msgSender	Internal		
	_msgData	Internal		
SafeMath	Library			

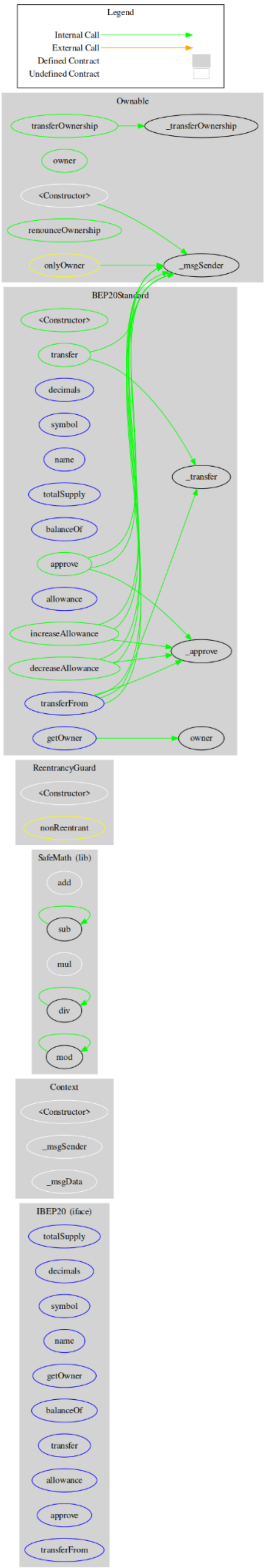
	add	Internal		
	sub	Internal		
	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
	mod	Internal		
	mod	Internal		
ReentrancyGuard	Implementation			
		Internal	✓	
Ownable	Implementation	Context		
		Internal	✓	
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
BEP20Standard	Implementation	Context, IBEP20, Ownable, ReentrancyGuard		
		Public	✓	-
	getOwner	External		-

	decimals	External		-
	symbol	External		-
	name	External		-
	totalSupply	External		-
	balanceOf	External		-
	transfer	Public	✓	nonReentrant
	allowance	External		-
	approve	Public	✓	-
	transferFrom	External	✓	nonReentrant
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_approve	Internal	✓	

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
Ownable
transferOwnership
_transferOwnership
owner
<Constructor>
renounceOwnership
onlyOwner
_msgSender
BEP20Standard
<Constructor>
transfer
decimals
symbol
name
totalSupply
balanceOf
approve
allowance
increaseAllowance
decreaseAllowance
transferFrom
getOwner
_transfer
_approve
owner
ReentrancyGuard
<Constructor>
nonReentrant
SafeMath (lib)
add
sub
mul
div
mod
Context
<Constructor>
_msgSender
_msgData
IBEP20 (iface)
totalSupply
decimals
symbol
name
getOwner
balanceOf
transfer
allowance
approve
transferFrom

Summary

JOJO contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. JOJO is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope’s prior written consent. This report is not nor should be considered an “endorsement” or “disapproval” of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope’s position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

The Cyberscope team

https://www.cyberscope.io