Cyberscope

Audit Report

IllumiShare SRG

November 2022

Type	BEP20
Network	BSC
Address	0x5AE6862B92Fe443D2C4addD9C6e65Fc0C7ccdDc0
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	SrgToken
Compiler Version	v0.8.17+commit.8df45f5f
Optimization	200 runs
Explorer	https://bscscan.com/token/0x5AE6862B92Fe443D2C4addD9C6e65Fc0C7ccdDc0
Symbol	SRG
Decimals	18
Total Supply	7,951,696,555
Ownership	MultiSig

Audit Updates

Initial Audit	24th June 2022 https://github.com/cyberscope-io/audits/blob/main/illumi/v1/audit.pdf
Corrected Phase 1	11th October 2022 https://github.com/cyberscope-io/audits/blob/main/illumi/v2/audit.pdf
Corrected Phase 2	2nd November 2022 https://github.com/cyberscope-io/audits/blob/main/illumi/v3/audit.pdf
Corrected Phase 3	22nd November 2022

Source Files

Filename	SHA256
@openzeppelin/contracts-0.8/access/Ownable.sol	dc6ecf2fb375c223c78b1eecb52d9ddf2397a622af29aa39597bf9fa5e800ad4
@openzeppelin/contracts-0.8/token/ERC20/ERC20.sol	ec5fc414e1e0bfdbf6ac510d75dcdcef25dd5da440fef0be67cfd7b9fe038611
@openzeppelin/contracts-0.8/token/ERC20/IERC20.sol	5f4e89bc7ee8aeb26b724218151ebe2b5787f2c73b084d3e2b54ef5716223b18
@openzeppelin/contracts-0.8/token/ERC20/utils/SafeERC20.sol	7b5c5895d3b1080dbe29753c6ffea0aa9618ca847f9e0bc0c2f29d68a5cceaae
@openzeppelin/contracts-0.8/utils/Address.sol	3cd9dd62a5fdc865d8b069f36ee9977c726932b1f6ad9e9bb3acb819dfa6fa59
@openzeppelin/contracts-0.8/utils/Context.sol	543c46d0f81fd4e5d9d6a92beef3d2be18badb483b0b4718c819fe3dbbc37587
@openzeppelin/contracts-0.8/utils/math/SafeMath.sol	3bf9042f6d35f2cf0389fb8bef53b3ff29d60740a60e92d423798a62ec57cdc9
contracts/SrgToken.sol	fd64f6bcdf2526b48021f6bb483dbe5be824cee5e09c46353045c238b263d061

Contract Analysis

Critical Medium Minor Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	MultiSig
●	OCTD	Transfers Contract's Tokens	MultiSig
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	ULTW	Transfers Liquidity to Team Wallet	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

ST - Stops Transactions

Criticality	medium
Location	SrgToken-1.sol#L156
Status	MultiSig

Description

The contract owner has the authority to stop the transactions. The owner may take advantage by setting the coldStakingAddress to null. As a result, the contract will revert. Additionally, the balance of coldStakingAddress might manipulate the transaction flow and stop transactions. This could happen by setting the balance of the sender to the maximum possible amount.

```
function _beforeTokenTransfer(
        address from,
        address,
        uint256 amount
    ) internal virtual override {
        if (from != address(0)) {
            require(
                balanceOf(from) >
                    IColdStaking(coldStakingAddress).balanceOf(from) + amount,
                "Not enough unlocked"
            );
        }
    }
```

Recommendation

The contract could embody a check for not allowing setting the coldStakingAddress to an invalid address. Additionally, the contract could embody the external call on a try-catch statement and sanitize the returned value.

Instead of using Untrusted Source to manipulate the transaction flow. It is recommended to add a max transaction amount to a fixed percentage of the total supply. For example, an acceptable max transaction amount is greater than 0.1% of the total supply.

The team should carefully manage the private keys of the owner’s account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Team update

The ownership of the contract has been moved to a multi-signature wallet. It requires multiple wallets to sign and approve the owner functions. This is an extra security mechanism.

OCTD - Transfers Contract's Tokens

Criticality	minor / informative
Location	SrgToken-1.sol#L91
Status	MultiSig

Description

The contract owner has the authority to claim all the balance of the contract. The owner may take advantage of it by calling the `withdrawTokens` function.

```
function withdrawTokens(IERC20 token) external onlyOwner {
        token.safeTransfer(owner(), token.balanceOf(address(this)));
}
```

Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. That risk can be prevented by temporarily locking the contract or renouncing ownership.

Team update

The ownership of the contract has been moved to a multi-signature wallet. It requires multiple wallets to sign and approve the owner functions. This is an extra security mechanism.

Contract Diagnostics

Critical Medium Minor

Severity	Code	Description	Status
●	US	Untrusted Source	MultiSig

US - Untrusted Source

Criticality	critical
Location	SrgToken-1.sol#L24
Status	MultiSig

Description

The contract uses an external contract in order to determine the transaction's flow. The external contract is untrusted. As a result, it may produce security issues and harm the transactions.

```
address public coldStakingAddress;
```

Recommendation

The contract should use a trusted external source. A trusted source could be either a commonly recognized or an audited contract. The pointing addresses should not be able to change after the initialization.

Team update

The ownership of the contract has been moved to a multi-signature wallet. It requires multiple wallets to sign and approve the owner functions. This is an extra security mechanism.

Contract Functions

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
ERC20	Implementation	Context, IERC20		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
IERC20	Interface			
	totalSupply	External		-

	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
SafeERC20	Library			
	safeTransfer	Internal	✓	
	safeTransferFrom	Internal	✓	
	safeApprove	Internal	✓	
	safeIncreaseAllowance	Internal	✓	
	safeDecreaseAllowance	Internal	✓	
	_callOptionalReturn	Private	✓	
Address	Library			
	isContract	Internal		
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	
	_verifyCallResult	Private		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
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		
IColdStaking	Interface			
	balanceOf	External		-
SrgToken	Implementation	ERC20, Ownable		
	<Constructor>	Public	✓	ERC20
	burn	External	✓	onlyOwner
	setTransferFee	External	✓	onlyOwner
	setColdStakingAddress	External	✓	onlyOwner
	withdrawTokens	External	✓	onlyOwner
	getTransferFee	External		-
	transfer	Public	✓	-
	transferFrom	Public	✓	-
	_transferWithFees	Private	✓	
	_beforeTokenTransfer	Internal	✓	

Contract Flow

Summary

There are some functions that can be abused by the owner like stopping transactions and transferring tokens to the team's wallet. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats. There is also a limit of max 18% fees.

When the audit was created, the ownership of the contract has been moved to a multi-signature wallet. It requires multiple wallets to sign and approve the owner functions. This is an extra security mechanism.

The ownership transfer

https://bscscan.com/tx/0x8d39cf6703b52c26e2c4ccd6ab554dce2119bbc74bc5f6af20c65effe28d9a75

The multisign wallet

https://bscscan.com/address/0x3b41e7f2f2975affdcae95217c2dc938be36b415

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment.

Cyberscope team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed.

The Cyberscope team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Cyberscope receive a payment to manipulate those results or change the awarding badge that we will be adding in our website.

Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token.

The Cyberscope team disclaims any liability for the resulting losses.

About Cyberscope

Coinscope audit and K.Y.C. service has been rebranded to Cyberscope.

Coinscope is the leading early coin listing, voting and auditing authority firm. The audit process is analyzing and monitoring many aspects of the project. That way, it gives the community a good sense of security using an informative report and a generic score.

Cyberscope and Coinscope are aiming to make crypto discoverable and efficient globally. They provide all the essential tools to assist users draw their own conclusions.

The Cyberscope team

https://www.cyberscope.io