Cyberscope

Audit Report

FirstDay

September 2022

Type	BEP20
Network	BSC
Address	0xb8E99e216CC8Cf342009ff7aFbd960135A1D072F
Audited by	© cyberscope

Table of Contents

Contract Review

Contract Name	FirstDate
Compiler Version	v0.8.4+commit.c7e474f2
Optimization	200 runs
Licence	None
Explorer	https://bscscan.com/token/0xb8E99e216CC8Cf342009ff7aFbd960135A1D072F
Symbol	FD
Decimals	9
Total Supply	100,000,000
Domain	firstdate.space

Source Files

Filename	**SHA256**
contract.sol	45f533bc888cd046bf66008b5d28c12f0c7cf51c565dbd8bae06c1e424644d3a

Audit Updates

Initial Audit	23rd September 2022
Corrected	

Contract Analysis

● Critical ● Medium ● Minor / Informative ● Pass

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

OCTD - Transfers Contract's Tokens

Criticality	minor / informative
Location	contract.sol#L315
Status	Unresolved

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 `rescueForeignTokens` function.

```
function rescueForeignTokens(address _tokenAddr, address _to, uint _amount) public
onlyDev() {
        emit tokensRescued(_tokenAddr, _to, _amount);
        Token(_tokenAddr).transfer(_to, _amount);
}
```

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.

ELFM - Exceeds Fees Limit

Criticality	medium
Location	contract.sol#L403
Status	Unresolved

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `setFee` function with a high percentage value.

```
function setFee(uint256 redisFeeOnBuy, uint256 redisFeeOnSell, uint256 taxFeeOnBuy,
uint256 taxFeeOnSell) public onlyDev {
            require(redisFeeOnBuy < 31, "Redis cannot be more than 30.");
            require(redisFeeOnSell < 31, "Redis cannot be more than 30.");
            require(taxFeeOnBuy < 21, "Tax cannot be more than 20.");
            require(taxFeeOnSell < 21, "Tax cannot be more than 20.");
      _redisFeeOnBuy = redisFeeOnBuy;
      _redisFeeOnSell = redisFeeOnSell;
      _taxFeeOnBuy = taxFeeOnBuy;
      _taxFeeOnSell = taxFeeOnSell;
}
```

Recommendation

The contract could embody a check for the maximum acceptable value.

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.

ULTW - Transfers Liquidity to Team Wallet

Criticality	minor / informative
Location	contract.sol#L391,397
Status	Unresolved

Description

The contract owner has the authority to transfer funds without limit to the team wallet. These funds have been accumulated from fees collected from the contract. The owner may take advantage of it by calling the `manualswap` and `manualsend` methods.

```
function manualswap() external {
        require(_msgSender() == _developmentAddress || _msgSender() == _marketingAddress
|| _msgSender() == owner());
        uint256 contractBalance = balanceOf(address(this));
        swapTokensForEth(contractBalance);
    }

    function manualsend() external {
        require(_msgSender() == _developmentAddress || _msgSender() == _marketingAddress
|| _msgSender() == owner());
        uint256 contractETHBalance = address(this).balance;
        sendETHToFee(contractETHBalance);
    }
```

Recommendation

The contract could embody a check for the maximum amount of funds that can be swapped. Since a huge amount may volatile the token's price.

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.

Contract Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	STC	Succeeded Transfer Check	Unresolved
	BLC	Business Logic Concern	Unresolved
	CO	Code Optimization	Unresolved
	L01	Public Function could be Declared External	Unresolved
	L02	State Variables could be Declared Constant	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved
	L05	Unused State Variable	Unresolved

STC - Succeeded Transfer Check

Criticality	minor / informative
Location	contract.sol#L316
Status	Unresolved

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
function rescueForeignTokens(address _tokenAddr, address _to, uint _amount) public
onlyDev() {
        emit tokensRescued(_tokenAddr, _to, _amount);
        Token(_tokenAddr).transfer(_to, _amount);
    }
```

Recommendation

The contract should check if the result of the transfer methods is successful.

BLC - Business Logic Concern

Criticality	medium
Location	contract.sol#L357,364
Status	Unresolved

Description

The business logic seems peculiar. The implementation may not follow the expected behavior.

Misleading use of arguments on the function _getTValues.

```
function _getValues(uint256 tAmount) private view returns (uint256, uint256, uint256,
uint256, uint256, uint256) {
        (uint256 tTransferAmount, uint256 tFee, uint256 tTeam) = _getTValues(tAmount,
_redisFee, _taxFee);
        uint256 currentRate =  _getRate();
        (uint256 rAmount, uint256 rTransferAmount, uint256 rFee) = _getRValues(tAmount,
tFee, tTeam, currentRate);
        return (rAmount, rTransferAmount, rFee, tTransferAmount, tFee, tTeam);
    }

function _getTValues(uint256 tAmount, uint256 taxFee, uint256 TeamFee) private pure
returns (uint256, uint256, uint256) {
        uint256 tFee = tAmount.mul(taxFee).div(100);
        uint256 tTeam = tAmount.mul(TeamFee).div(100);
        uint256 tTransferAmount = tAmount.sub(tFee).sub(tTeam);
        return (tTransferAmount, tFee, tTeam);
    }
```

Recommendation

The team is advised to carefully check if the implementation follows the expected business logic.

CO - Code Optimization

Criticality	minor / informative
Location	contract.sol#L311
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 function `_tokenTransfer` i redundant.

```
function _tokenTransfer(address sender, address recipient, uint256 amount) private {
        _transferStandard(sender, recipient, amount);
}
```

Recommendation

Rewrite some code segments so the runtime will be more performant.

L01 - Public Function could be Declared External

Criticality	minor / informative
Location	contract.sol#L312,204,318,122,230,221,325,410,200,216,225,208,128,196,414,399
Status	Unresolved

Description

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

```
rescueForeignTokens
decimals
setNewDevAddress
renounceOwnership
transferFrom
allowance
setNewMarketingAddress
toggleSwap
symbol
...
```

Recommendation

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

L02 - State Variables could be Declared Constant

Criticality	minor / informative
Location	contract.sol#L105
Status	Unresolved

Description

Constant state variables should be declared constant to save gas.

```
_previousOwner
```

Recommendation

Add the constant attribute to state variables that never change.

L04 - Conformance to Solidity Naming Conventions

Criticality	minor / informative
Location	contract.sol#L158,312,317,410,144,157,46,159,324,311
Status	Unresolved

Description

Solidity defines a naming convention that should be followed. Rule exceptions:

- Allow constant variable name/symbol/decimals to be lowercase.
- Allow _ at the beginning of the mixed_case match for private variables and unused parameters.

```
_symbol
_tokenAddr
devAddressUpdated
_amount
_swapEnabled
_tTotal
_name
WETH
_decimals
...
```

Recommendation

Follow the Solidity naming convention.
https://docs.soliditylang.org/en/v0.4.25/style-guide.html#naming-conventions.

L05 - Unused State Variable

Criticality	minor / informative
Location	contract.sol#L139,105
Status	Unresolved

Description

There are segments that contain unused state variables.

```
_tOwned
_previousOwner
```

Recommendation

Remove unused state variables.

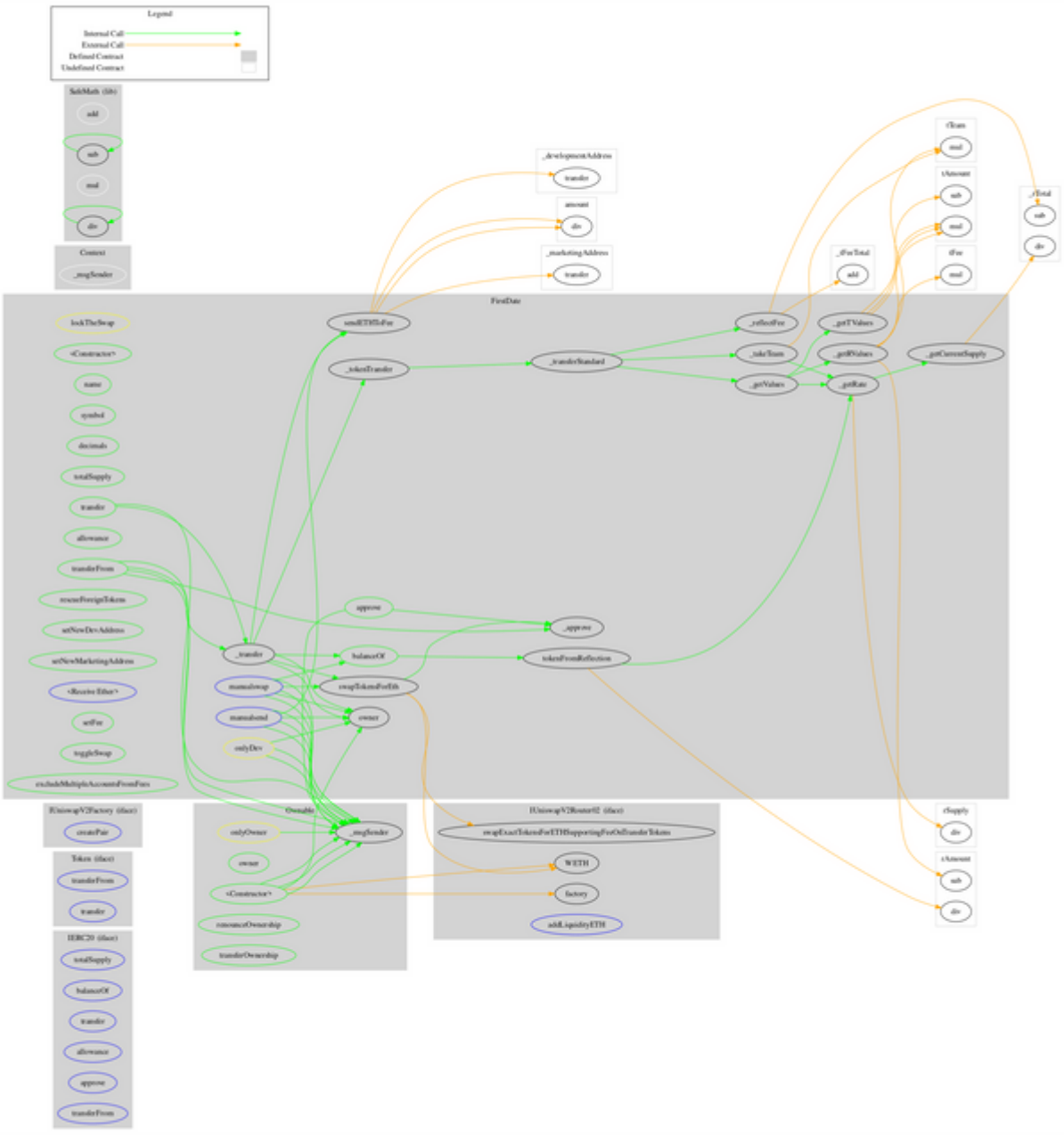
Contract Functions

Contract	Type	Bases		
	Function Name	Visibility	Mutability	Modifiers
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Token	Interface			
	transferFrom	External	✓	-
	transfer	External	✓	-
IUniswapV2Factory	Interface			
	createPair	External	✓	-
IUniswapV2Router02	Interface			
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
	factory	External		-
	WETH	External		-
	addLiquidityETH	External	Payable	-
Context	Implementation			
	_msgSender	Internal		
SafeMath	Library			
	add	Internal		
	sub	Internal		

	sub	Internal		
	mul	Internal		
	div	Internal		
	div	Internal		
Ownable	Implementation	Context		
	<Constructor>	Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
FirstDate	Implementation	Context, IERC20, Ownable		
	<Constructor>	Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	tokenFromReflection	Private		
	_approve	Private	✓	
	_transfer	Private	✓	
	swapTokensForEth	Private	✓	lockTheSwap
	sendETHToFee	Private	✓	
	_tokenTransfer	Private	✓	
	rescueForeignTokens	Public	✓	onlyDev
	setNewDevAddress	Public	✓	onlyDev
	setNewMarketingAddress	Public	✓	onlyDev
	_transferStandard	Private	✓	
	_takeTeam	Private	✓	

	_reflectFee	Private	✓	
	<Receive Ether>	External	Payable	-
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	manualswap	External	✓	-
	manualsend	External	✓	-
	setFee	Public	✓	onlyDev
	toggleSwap	Public	✓	onlyDev
	excludeMultipleAccountsFromFees	Public	✓	onlyOwner

Contract Flow



Domain Info

Domain Name	firstdate.space
Registry Domain ID	D324002746-CNIC
Creation Date	2022-09-22T10:10:47.0Z
Updated Date	2022-09-22T10:10:49.0Z
Registry Expiry Date	2023-09-22T23:59:59.0Z
Registrar WHOIS Server	whois.hostinger.com
Registrar URL	https://www.hostinger.com/
Registrar	Hostinger, UAB
Registrar IANA ID	1636

The domain was created about 20 hours before the creation of the audit. It will expire in 12 months.

There is no public billing information, the creator is protected by the privacy settings.

Summary

There are some functions that can be abused by the owner like transferring tokens to the team’s wallet, manipulating fees and transferring funds 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.

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