SafeCum Token

RugfreeCoins Verified on October 11th, 2023

Overview

✅ No mint function found, the owner cannot mint tokens after initial deployment.

✅ The owner can't set a max transaction limit

✅ The owner can't pause trading once it's enabled

❌ The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell.

✅ The owner can't change fees.

✅ The owner can't blacklist wallets.

✅ The owner can't set a max wallet limit

✅ The owner can't claim the contract's balance of its own token.

❗ HIGH SEVERITY ISSUES

The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell. : SAFU Developer is handling the contract

```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

Contents

Audit details

Audited project
SafeCum Token

Contract Address
0x9C729Dbdd6b65f53110b870Acf17b70979408F61

Client contact
SafeCum Token Team

Blockchain
Ethereum

Project website
https://safecum.vip/

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at 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 us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – **please make sure to read it in full.**

Background

RugfreeCoins was commissioned by the **SafeCum Token Team** to perform an audit of the smart contract.

https://etherscan.io/address/0x9c729dbdd6b65f53110b870acf17b70979408f61#code

This audit focuses on verifying that the smart contract is secure, resilient, and working according to the specifications.

The information in this report should be used to understand the risk exposure of the smart contract, project feasibility, and long-term sustainability, and as a guide to improving the smart contract's security posture by remediating the identified issues.

Tokenomics

▲ 0% tax when buying & selling

Target market and the concept

- Anyone who's interested in the Crypto space with long-term investment plans.
- Anyone who's ready to earn a passive income by holding tokens.
- Anyone who's interested in trading tokens.
- Anyone who's interested in taking part in the SafuCum token ecosystem.
- Anyone who's interested in taking part in the future plans of SafuCum Token.
- Anyone who's interested in making financial transactions with any other party using SafuCum Token as the currency.

Potential to grow with score points

⚡ Project efficiency	8 / 10
🌟 Project uniqueness	8 / 10
📊 Information quality	8 / 10
💧 Service quality	8 / 10
💻 System quality	8 / 10
🌍 Impact on the community	9 / 10
💼 Impact on the business	9 / 10
🔮 Preparing for the future	8 / 10
🔒 Smart contract security	10 / 10
🛠️ Smart contract functionality assessment	10 / 10
🏆 **Total Score**	**8.6** / **10**

Contract details

Token contract details for 11th of October 2023

Contract name	**SafeCum**
Contract address	**0x9C729Dbdd6b65f53110b870Acf17b70979408F61**
Token supply	**10,000,000,**
Token ticker	**SCM**
Decimals	**18**
Token holders	**1**
Transaction count	**1**
Contract deployer address	**0x6b27eB26B6B0851CF49CB1d1d15D80d89b27B985**
Contract's current owner address	**0x6b27eB26B6B0851CF49CB1d1d15D80d89b27B985**

Contract code function details

№	Category	Item	Result
1	Coding conventions	ERC20 Token standards	PASS
		Compile errors	PASS
		Compiler version security	PASS
		Visibility specifiers	PASS
		Gas consumption	LOW
		SafeMath features	PASS
		Fallback usage	PASS
		tx.origin usage	PASS
		Deprecated items	PASS
		Redundant code	PASS
		Overriding variables	PASS
2	Function call audit	Authorization of function call	PASS
		Low level function (call/delegate call) security	PASS
		Returned value security	PASS
		Self destruct function security	PASS
3	Business security & centralisation	Access control of owners	PASS
		Business logics	PASS
		Business implementation	PASS
4	Integer overflow/underflow		PASS
5	Reentrancy		PASS
6	Exceptional reachable state		PASS
7	Transaction ordering dependence		PASS
8	Block properties dependence		PASS
9	Pseudo random number generator (PRNG)		PASS
10	DoS (Denial of Service)		PASS
11	Token vesting implementation		PASS
12	Fake deposit		PASS
13	Event security		PASS

Contract description table

The below table represents the summary of the contracts and methods in the token contract. We scanned the whole contract and listed down all the Interfaces, functions, and implementations with their visibility and mutability.

Contract	Type	Bases		
L	Function Name	Visibility	Mutability	Modifiers
Context	**Implementation**			
L	_msgSender	Internal 🔒		
L	_msgData	Internal 🔒		
Ownable	**Implementation**	**Context**		
L		Public ❗	🛑	NO ❗
L	owner	Public ❗		NO ❗
L	_checkOwner	Internal 🔒		
L	renounceOwnership	Public ❗	🛑	onlyOwner
L	transferOwnership	Public ❗	🛑	onlyOwner
L	_transferOwnership	Internal 🔒	🛑	
IERC20	**Interface**			
L	totalSupply	External ❗		NO ❗
L	balanceOf	External ❗		NO ❗
L	transfer	External ❗	🛑	NO ❗
L	allowance	External ❗		NO ❗
L	approve	External ❗	🛑	NO ❗
L	transferFrom	External ❗	🛑	NO ❗

IERC20Meta data	Interface	IERC20		
L	name	External ❗		NO ❗
L	symbol	External ❗		NO ❗
L	decimals	External ❗		NO ❗
ERC20	**Implementation**	**Context, IERC20, IERC20 Metadata**		
L		Public ❗	🛑	NO ❗
L	name	Public ❗		NO ❗
L	symbol	Public ❗		NO ❗
L	decimals	Public ❗		NO ❗
L	totalSupply	Public ❗		NO ❗
L	balanceOf	Public ❗		NO ❗
L	transfer	Public ❗	🛑	NO ❗
L	allowance	Public ❗		NO ❗
L	approve	Public ❗	🛑	NO ❗
L	transferFrom	Public ❗	🛑	NO ❗
L	increaseAllowance	Public ❗	🛑	NO ❗
L	decreaseAllowance	Public ❗	🛑	NO ❗
L	_transfer	Internal 🔒	🛑	
L	_mint	Internal 🔒	🛑	
L	_burn	Internal 🔒	🛑	
L	_approve	Internal 🔒	🛑	
L	_spendAllowance	Internal 🔒	🛑	
L	_beforeTokenTransfer	Internal 🔒	🛑	
L	_afterTokenTransfer	Internal 🔒	🛑	
DexFactory	**Interface**			

L	createPair	External ❗	🛑	NO ❗
DexRouter	**Interface**			
L	factory	External ❗		NO ❗
L	WETH	External ❗		NO ❗
L	addLiquidityETH	External ❗	💵	NO ❗
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
SafeCum	**Implementation**	**ERC20, Ownable**		
L		Public ❗	🛑	ERC20
L	enableTrading	External ❗	🛑	onlyOwner
L	setWhitelistStatus	External ❗	🛑	onlyOwner
L	checkWhitelist	External ❗		NO ❗
L	_transfer	Internal 🔒	🛑	
L		External ❗	💵	NO ❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Inheritance Hierarchy
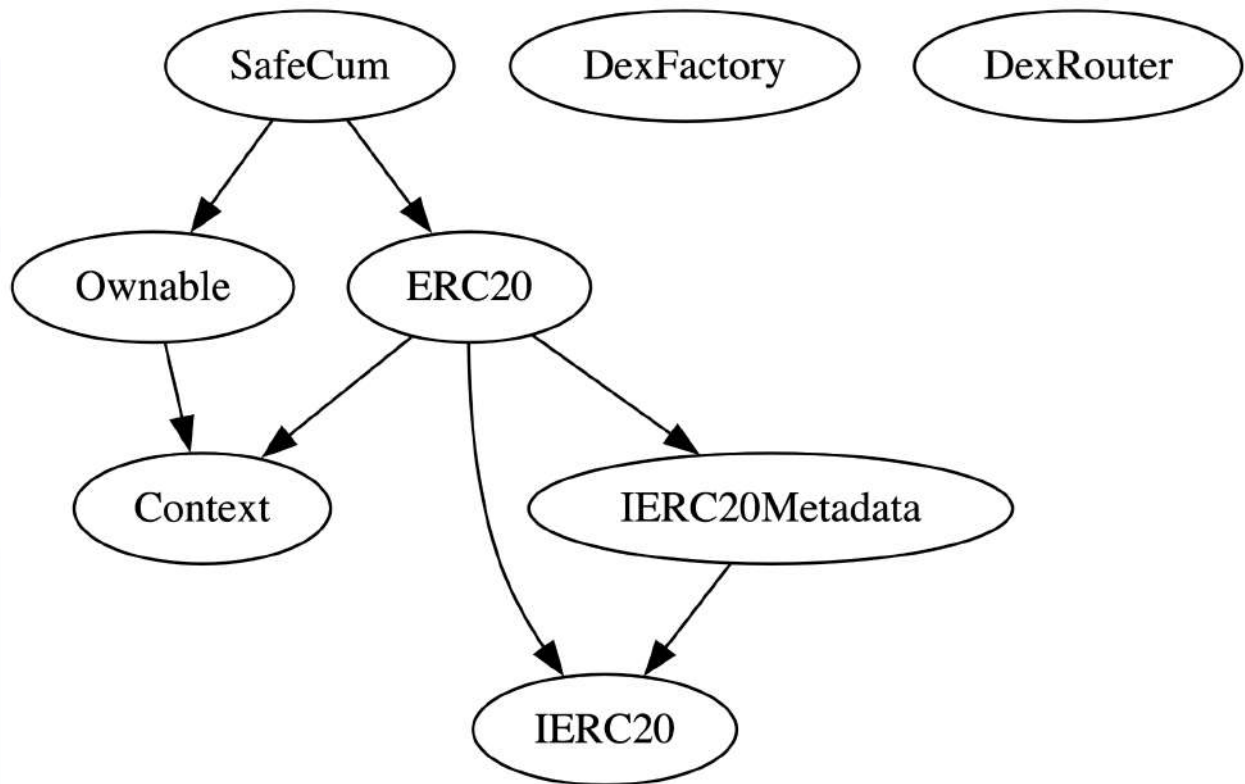
Security issue checking status

❖ High severity issues

The owner must enable trade for the holders, if trading remains disabled, no one would be able to buy and sell. : <mark>SAFU Developer is handling the contract</mark>

```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

❖ Medium severity issues

No medium severity issues found

❖ Low severity issues

startTradingBlock variable setting value but it is not used anywhere in the contract

```
uint256 public startTradingBlock;
```

Owner privileges

❖ The owner can enable trading, and once enabled can not disable it again

```
function enableTrading() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

❖ The owner can add or remove whitelisted wallets (who can do transactions before enabling trading)

```
function setWhitelistStatus(
    address _wallet,
    bool _status
) external onlyOwner {
    whitelisted[_wallet] = _status;
    emit Whitelist(_wallet, _status);
}
```

Audit conclusion

RugFreeCoins team has performed in-depth testing, line-by-line manual code review, and automated audit of the smart contract. The smart contract was analyzed mainly for common smart contract vulnerabilities, exploits, manipulations, and hacks. According to the smart contract audit.

Smart contract functional Status:	PASS ▾
Smart contract security Status:	SAFE (Handled by a SAFU developer) ▾
Number of risk issues:	1
Solidity code functional issue level:	PASS ▾
Number of owner privileges:	2
Centralization risk correlated to the active owner:	HIGH ▾
Smart contract active ownership:	ACTIVE ▾