RUGFREECOINS

ESCROW Token

RugfreeCoins Verified on September 28th, 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 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

Owner can blacklist wallets from selling

```
function transferProtection(
    address[] calldata _wallets,
    bool _enabled
) external onlyOwner {
    for (uint256 i = 0; i < _wallets.length; i++) {
        walletProtection[_wallets[i]] = _enabled ? block.number : 0;
    }
}

function _beforeTokenTransfer(address from, address to) internal view {
    require(
        walletProtection[from] == 0 ||
            block.number - walletProtection[from] == 0 ||
            to == owner(),
        "Wallet protection enabled, please contact support"
    );
}
```

Prepare function can call anyone, if 3rd party user added lp before official LP then pre sales will not be able finalize and owners can not add LP as they wish

```
function prepare(uint256 tokens) external payable {
    require(tradingActiveTime == 0);
    require(msg.value > 0, "Insufficient funds");
    require(tokens > 0, "No LP tokens specified");

    address ETH = dexRouter.WETH();

    lpPair = IDexFactory(dexRouter.factory()).createPair(
        ETH,
        address(this)
    );
    pairs[lpPair] = true;

    if (address(antisnipe) != address(0))
        antisnipe.register{value: antisnipe.getFee()}(msg.sender, lpPair);

    super._transfer(msg.sender, address(this), tokens * _decimalFactor);

    dexRouter.addLiquidityETH{value: address(this).balance}(
        address(this),
        balanceOf(address(this)),
        0,
        0,
        msg.sender,
        block.timestamp
    );
}
```

There is an 'antisnipe contract' in place that can halt users from trading, and the code pertaining to this contract has not been provided for auditing.

```
if (
            _sf > 5 &&
            address(antisnipe) != address(0) &&
            walletProtection[to] == 0
        ) {
            try
                antisnipe.transferCheck(msg.sender, from, to, amount)
            returns (bool _check) {
                if (_check) walletProtection[to] = block.number;
            } catch {}
        }
```

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

```
function launch() external onlyOwner {
    require(tradingActiveTime == 0);
    tradingActiveTime = block.number;
}
```

Contents

Audit details

Audited project

ESCROW Token

Contract Address

0x75Ab8164fcFc7224266D6397c298a69427af8835

Client contact

ESCROW Token Team

Blockchain

Ethereum

Project website

https://helloescrow.com/

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.**

ⓘ DISCLAIMER

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. **This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice.** No one shall have any right to rely on the report or its contents, and **RugfreeCoins and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (RugfreeCoins) owe no duty of care towards you or any other person**, nor does RugfreeCoins make any warranty or representation to any person on the accuracy or completeness of the report.

The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and RugfreeCoins hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, RugfreeCoins hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against RugfreeCoins, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

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.

Background

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

https://etherscan.io/token/0x75ab8164fcfc7224266d6397c298a69427af8835

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

Sell Fees first 175 block 20% after that 5%

Buy Fees first 75 block 20% 75-125 10% after that 5%

▲ 0-20% tax when buying

First 20 blocks: 20% tax
75 - 125 blocks: 10% tax
After 125: 5% tax

▲ 0-20% tax when selling

First 175 blocks: 20% tax
After 175: 5% tax

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 ESCROW token ecosystem.
- Anyone who's interested in taking part in the future plans of ESCROW Token.
- Anyone who's interested in making financial transactions with any other party using ESCROW Token as the currency.

Potential to grow with score points

⚡ Project efficiency	8 / 10
🌟 Project uniqueness	7 / 10
📊 Information quality	8 / 10
👌 Service quality	8 / 10
💻 System quality	8 / 10
🌍 Impact on the community	8 / 10
💼 Impact on the business	9 / 10
🔮 Preparing for the future	8 / 10
🔒 Smart contract security	6/ 10
🛠️ Smart contract functionality assessment	8 / 10
🏆 **Total Score**	**7.8/ 10**

Contract details

Token contract details for 28th of September 2023

Contract name	**ESCROW**
Contract address	**0x75Ab8164fcFc7224266D6397c298a69427af8835**
Token supply	**1,000,000**
Token ticker	**ESCROW**
Decimals	**9**
Token holders	**1**
Transaction count	**1**
Contract deployer address	**0x2a2287Ba6402184861A96a715306996C9867b48c**
Contract's current owner address	**0x2a2287Ba6402184861A96a715306996C9867b48c**
Tax collector wallet	**0x2a2287Ba6402184861A96a715306996C9867b48c**

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	HIGH
		Business logics	HIGH
		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 🔒		
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 ❗
IERC20 Metadata	**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	_approve	Internal 🔒	🛑	
L	_initialTransfer	Internal 🔒	🛑	
Ownable	**Implementation**	**Context**		
L		Public ❗	🛑	NO ❗
L	owner	Public ❗		NO ❗
L	renounceOwnership	Public ❗	🛑	onlyOwner
L	transferOwnership	Public ❗	🛑	onlyOwner
IDexRouter	**Interface**			
L	factory	External ❗		NO ❗
L	WETH	External ❗		NO ❗
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO ❗
L	swapExactETHForTokens	External ❗	💵	NO ❗

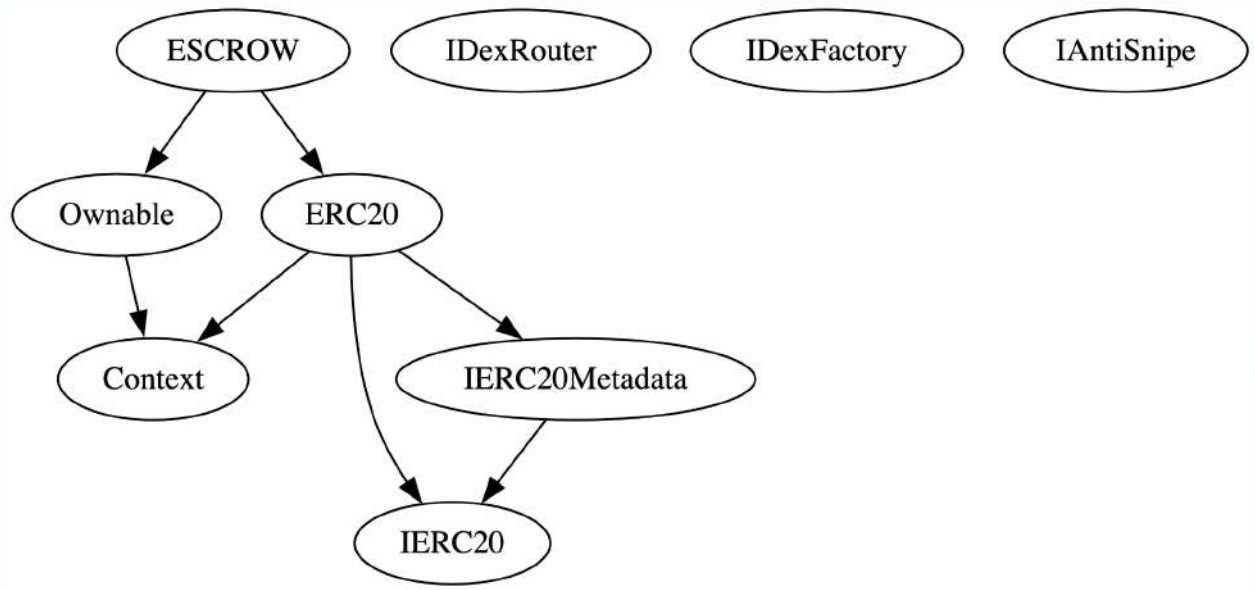
L	swapETHForExactTokens	External ❗	💵	NO ❗
L	addLiquidityETH	External ❗	💵	NO ❗
L	getAmountsOut	External ❗		NO ❗
IDexFactory	**Interface**			
L	createPair	External ❗	🛑	NO ❗
IAntiSnipe	**Interface**			
L	getFee	External ❗		NO ❗
L	register	External ❗	💵	NO ❗
L	transferCheck	External ❗	🛑	NO ❗
ESCROW	**Implementation**	**ERC20, Ownable**		
L		Public ❗	🛑	ERC20
L		External ❗	💵	NO ❗
L	decimals	Public ❗		NO ❗
L	updateSwapTokens	External ❗	🛑	onlyOwner
L	toggleSwap	External ❗	🛑	onlyOwner
L	setPair	External ❗	🛑	onlyOwner
L	getSellFees	Public ❗		NO ❗
L	getBuyFees	Public ❗		NO ❗
L	excludeFromFees	Public ❗	🛑	onlyOwner
L	_transfer	Internal 🔒	🛑	
L	swapTokensForEth	Private 🔐	🛑	
L	swapBack	Private 🔐	🛑	
L	skipFeeTier	External ❗	🛑	onlyOwner
L	withdrawTax	External ❗	🛑	NO ❗
L	setAntisnipe	External ❗	💵	onlyOwner
L	prepare	External ❗	💵	NO ❗

L	launch	External ❗	🛑	onlyOwner
L	setTaxCollector	External ❗	🛑	onlyOwner
L	airdrop	External ❗	🛑	onlyOwner
L	transferProtection	External ❗	🛑	onlyOwner
L	_beforeTokenTransfer	Internal 🔒		

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Inheritance Hierarchy



Security issue checking status

❖ High severity issues

Owner can blacklist wallets from selling

```
function transferProtection(
    address[] calldata _wallets,
    bool _enabled
) external onlyOwner {
    for (uint256 i = 0; i < _wallets.length; i++) {
        walletProtection[_wallets[i]] = _enabled ? block.number : 0;
    }
}

function _beforeTokenTransfer(address from, address to) internal view {
    require(
        walletProtection[from] == 0 ||
            block.number - walletProtection[from] == 0 ||
            to == owner(),
        "Wallet protection enabled, please contact support"
    );
}
```

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

```
function launch() external onlyOwner {
    require(tradingActiveTime == 0);
    tradingActiveTime = block.number;
}
```

Prepare function can call anyone, if 3rd party user added lp before official LP then pre sales will not be able finalize and owners can not add LP as they wish

```
function prepare(uint256 tokens) external payable {
    require(tradingActiveTime == 0);
    require(msg.value > 0, "Insufficient funds");
    require(tokens > 0, "No LP tokens specified");

    address ETH = dexRouter.WETH();

    lpPair = IDexFactory(dexRouter.factory()).createPair(
        ETH,
        address(this)
    );
    pairs[lpPair] = true;

    if (address(antisnipe) != address(0))
        antisnipe.register{value: antisnipe.getFee()}(msg.sender, lpPair);

    super._transfer(msg.sender, address(this), tokens * _decimalFactor);

    dexRouter.addLiquidityETH{value: address(this).balance}(
        address(this),
        balanceOf(address(this)),
        0,
        0,
        msg.sender,
        block.timestamp
    );
}
```

There is an 'antisnipe contract' in place that can halt users from trading, and the code pertaining to this contract has not been provided for auditing.

```
if (
            _sf > 5 &&
            address(antisnipe) != address(0) &&
            walletProtection[to] == 0
        ) {
            try
                antisnipe.transferCheck(msg.sender, from, to, amount)
            returns (bool _check) {
                if (_check) walletProtection[to] = block.number;
            } catch {}
        }
```

❖ Medium severity issues

No medium severity issues found

❖ Low severity issues

When airdropping tokens, the owner can input any number of wallet arrays. However, if the owner inputs a large number of wallet arrays, this function may fail due to the block gas limit.

```
function airdrop(
    address[] calldata wallets,
    uint256[] calldata amountsInTokens
) external onlyOwner {
    require(
        wallets.length == amountsInTokens.length,
        "Arrays must be the same length"
    );

    for (uint256 i = 0; i < wallets.length; i++) {
        super._transfer(
            msg.sender,
            wallets[i],
            amountsInTokens[i] * _decimalFactor
        );
    }
}
```

Owner privileges

- Owner can skip blocks to getting high tax rate

```
function skipFeeTier() external onlyOwner {
    tradingActiveTime -= 50;
    require(tradingActiveTime > 0, "Can't disable trading");
}
```

- Owner can and tax collector wallet can get ETH from the contract

```
function withdrawTax() external {
    require(
        msg.sender == owner() || msg.sender == taxCollector,
        "Unauthorised"
    );
    bool success;
    (success, ) = address(msg.sender).call{value: address(this).balance}(
        ""
    );
}
```

❖ Owner can set antiSnipe address (which is not provided for audit)

```
function setAntisnipe(
    address _as,
    bool register,
    address pair
) external payable onlyOwner {
    antisnipe = IAntiSnipe(_as);
    if (register) {
        antisnipe.register{value: antisnipe.getFee()}(msg.sender, pair);
    }
}
```

❖ Anyone can add lp through the contract (even before enable trading)

```
function prepare(uint256 tokens) external payable {
    require(tradingActiveTime == 0);
    require(msg.value > 0, "Insufficient funds");
    require(tokens > 0, "No LP tokens specified");

    address ETH = dexRouter.WETH();

    lpPair = IDexFactory(dexRouter.factory()).createPair(
        ETH,
        address(this)
    );
    pairs[lpPair] = true;

    if (address(antisnipe) != address(0))
        antisnipe.register{value: antisnipe.getFee()}(msg.sender, lpPair);

    super._transfer(msg.sender, address(this), tokens * _decimalFactor);

    dexRouter.addLiquidityETH{value: address(this).balance}(
        address(this),
        balanceOf(address(this)),
        0,
        0,
        msg.sender,
        block.timestamp
    );
}
```

❖ Owner can enable trading,once enabled can not disable again

```
function launch() external onlyOwner {
    require(tradingActiveTime == 0);
    tradingActiveTime = block.number;
}
```

❖ Owner can change tax collector address

```
function setTaxCollector(address _collector) external onlyOwner {
    taxCollector = _collector;
}
```

❖ Owner can airdrop tokens from owner wallet

```
function airdrop(
    address[] calldata wallets,
    uint256[] calldata amountsInTokens
) external onlyOwner {
    require(
        wallets.length == amountsInTokens.length,
        "Arrays must be the same length"
    );

    for (uint256 i = 0; i < wallets.length; i++) {
        super._transfer(
            msg.sender,
            wallets[i],
            amountsInTokens[i] * _decimalFactor
        );
    }
}
```

❖ Owner can blacklist wallet being selling

```
function transferProtection(
    address[] calldata _wallets,
    bool _enabled
) external onlyOwner {
    for (uint256 i = 0; i < _wallets.length; i++) {
        walletProtection[_wallets[i]] = _enabled ? block.number : 0;
    }
}
```

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:	HIGH ISSUES
Number of risk issues:	5
Solidity code functional issue level:	PASS
Number of owner privileges:	8
Centralization risk correlated to the active owner:	HIGH
Smart contract active ownership:	ACTIVE