RugFreeCoins Audit

EverEarn Token
Smart Contract Security Audit

November 9th, 2022

Contents

Audit details

Audited project
EverEarn Token

Contract Address

0xA87Ed75C257f1ec38393bEA0A83d55Ac2279D79c

Client contact

EverEarn Team

Blockchain

Ethereum Smart chain

Project website

https://everearn.net/

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.

Overview

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

✅ The owner can't set a max transaction limit below 0.1%

✅ The owner can't pause trading.

✅ The owner can't set a max wallet limit

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

❌ The owner can set fees over 25%.

❌ The owner can blacklist wallets.

Background

Rugfreecoins was commissioned by the EverEarn Team to perform an audit of the smart contract.

https://etherscan.io/address/0xA87Ed75C257f1ec38393bEA0A83d55Ac2279D79c

The focus of this audit is to verify 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

15% when buying & selling

- 11% trade distribute among holders as rewards in USDC.
- 1% trade goes to the liquidity pool
- 1% trade goes to the buyback wallet in ETH
- 2% trade goes to the marketing wallet in ETH

15% 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 ready to staking and receive rewards.
- Anyone who's interested in taking part of the EverEarn ecosystem.
- Anyone who's interested in taking part in the future plans of EverEarn Token.
- Anyone who's interested in making financial transactions with any other party using EverEarn Token as the currency.

Potential to grow with score points

1.	Project efficiency	10/10
2.	Project uniqueness	10/10
3	Information quality	10/10
4	Service quality	10/10
5	System quality	10/10
6	Impact on the community	10/10
7	Impact on the business	10/10
8	Preparing for the future	10/10
9	Smart contract security	10/10
10	Smart contract functionality assessment	10/10
Total Points		**10/10**

Contract details

Token contract details for 9th of November 2022

Contract name	Earn
Contract address	0xA87Ed75C257f1ec38393bEA0A83d55Ac2279D79c
Token supply	100,000,000,000
Token ticker	$Earn
Decimals	9
Token holders	1
Transaction count	1
Reward	0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48
Buyback wallet	0x3193669b46F25a63ed6D32024C1409b12a26563E
Dividend tracker	0x1c009262B3bF098b7EeE9BD7fe43107b9aD80B1E
LP receiver	0x91b1DC8443848314FA9B1f3F16497521A16b59ca
Marketing wallet	0xBA3944d8f9057e4fA32c80039385d21b1A689e59
Contract deployer address	0x4958122d53756655616262168867B4AaB7496BA8
Contract's current owner address	0x4958122d53756655616262168867B4AaB7496BA8

Contract code function details

No	Category	Item	Result
1	Coding conventions	BRC20 Token standards	pass
		compile errors	pass
		Compiler version security	pass
		visibility specifiers	pass
		Gas consumption	pass
		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	Access control of owners	Centralized Risk
		Business logics	pass
		Business implementations	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
EARN	**Implementation**	**IERC20, Ownable**		
L		Public ❗	🛑	NO❗
L		External ❗	💵	NO❗
L	totalSupply	External ❗		NO❗
L	name	Public ❗		NO❗
L	symbol	Public ❗		NO❗
L	decimals	Public ❗		NO❗
L	balanceOf	Public ❗		NO❗
L	getHolderDetails	Public ❗		NO❗
L	getLastProcessedIndex	Public ❗		NO❗
L	getNumberOfTokenHolders	Public ❗		NO❗
L	totalDistributedRewards	Public ❗		NO❗
L	allowance	External ❗		NO❗
L	approve	Public ❗	🛑	NO❗
L	_approve	Internal 🔒	🛑	

L	approveMax	External ❗	🛑	NO❗
L	transfer	External ❗	🛑	NO❗
L	transferFrom	External ❗	🛑	NO❗
L	_transferFrom	Internal 🔒	🛑	
L	takeFee	Internal 🔒	🛑	
L	_basicTransfer	Internal 🔒	🛑	
L	shouldTakeFee	Internal 🔒		
L	shouldDoContractSwap	Internal 🔒		
L	___claimRewards	Public ❗	🛑	NO❗
L	claimProcess	Public ❗	🛑	NO❗
L	isRewardExcluded	Public ❗		NO❗
L	isFeeExcluded	Public ❗		NO❗
L	doContractSwap	Internal 🔒	🛑	swapping
L	swapTokensForTokens	Private 🔐	🛑	
L	swapAndLiquify	Private 🔐	🛑	
L	swapTokensForEth	Private 🔐	🛑	
L	addLiquidity	Private 🔐	🛑	
L	setIsDividendExempt	External ❗	🛑	onlyOwner
L	setIsFeeExempt	External ❗	🛑	onlyOwner
L	setDoContractSwap	External ❗	🛑	onlyOwner
L	blackListWallets	External ❗	🛑	onlyOwner

L	setDistributionCriteria	External !	⬤	onlyOwner
L	setDistributorSettings	External !	⬤	onlyOwner
L	changeMarketingWallet	External !	⬤	onlyOwner
L	changeBuyBackWallet	External !	⬤	onlyOwner
L	changeLPWallet	External !	⬤	onlyOwner
L	changeBuyFees	External !	⬤	onlyOwner
L	changeSellFees	External !	⬤	onlyOwner
L	changeSwapFees	External !	⬤	onlyOwner
L	setSellCollDown	External !	⬤	onlyOwner
L	changeSellLimit	External !	⬤	onlyOwner
L	changeBuyLimit	External !	⬤	onlyOwner
L	excludeFromMaxSell	External !	⬤	onlyOwner
L	excludeFromMaxBuy	External !	⬤	onlyOwner
L	enableTrading	External !	⬤	onlyOwner
L	setAuthorizedWallets	External !	⬤	onlyOwner
L	rescueEth	External !	⬤	onlyOwner
L	purgeBeforeSwitch	Public !	⬤	onlyOwner
L	depositRewards	External !	⬤	onlyOwner
IDividend Distributor	**Interface**			
L	setDistributionCriteria	External !	⬤	NO!

L	setShare	External ❗	🛑	NO❗
L	deposit	External ❗	🛑	NO❗
L	process	External ❗	🛑	NO❗
L	purge	External ❗	🛑	NO❗
Dividend Distributor	**Implementation**	**IDividend Distributor**		
L		Public ❗	🛑	NO❗
L		External ❗	💵	NO❗
L	setDistributionCriteria	External ❗	🛑	onlyToken
L	purge	External ❗	🛑	onlyToken
L	setShare	External ❗	🛑	onlyToken
L	deposit	External ❗	🛑	onlyToken
L	process	External ❗	🛑	onlyToken
L	shouldDistribute	Internal 🔒		
L	distributeDividend	Internal 🔒	🛑	
L	claimDividend	External ❗	🛑	NO❗
L	getUnpaidEarnings	Public ❗		NO❗
L	getHolderDetails	Public ❗		NO❗
L	getCumulativeDividends	Internal 🔒		
L	getLastProcessedIndex	External ❗		NO❗
L	getNumberOfTokenHolders	External ❗		NO❗

14

L	getShareHoldersList	External ❗		NO❗
L	totalDistributedRewards	External ❗		NO❗
L	addShareholder	Internal 🔒	🛑	
L	removeShareholder	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❗
IUniswapV2 Factory	**Interface**			

IUniswapV2 Router02	Interface	IUniswapV 2Router01		
L	feeTo	External ❗		NO❗
L	feeToSetter	External ❗		NO❗
L	getPair	External ❗		NO❗
L	allPairs	External ❗		NO❗
L	allPairsLength	External ❗		NO❗
L	createPair	External ❗	🛑	NO❗
L	setFeeTo	External ❗	🛑	NO❗
L	setFeeToSetter	External ❗	🛑	NO❗
IUniswapV2 Router02	**Interface**	**IUniswapV 2Router01**		
L	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
L	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
L	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO❗
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
Context	**Implementation**			
L	_msgSender	Internal 🔒		
L	_msgData	Internal 🔒		
IUniswapV2 Router01	**Interface**			

L	factory	External ❗		NO❗
L	WETH	External ❗		NO❗
L	addLiquidity	External ❗	🛑	NO❗
L	addLiquidityETH	External ❗	💵	NO❗
L	removeLiquidity	External ❗	🛑	NO❗
L	removeLiquidityETH	External ❗	🛑	NO❗
L	removeLiquidityWithPermit	External ❗	🛑	NO❗
L	removeLiquidityETHWithPermit	External ❗	🛑	NO❗
L	swapExactTokensForTokens	External ❗	🛑	NO❗
L	swapTokensForExactTokens	External ❗	🛑	NO❗
L	swapExactETHForTokens	External ❗	💵	NO❗
L	swapTokensForExactETH	External ❗	🛑	NO❗
L	swapExactTokensForETH	External ❗	🛑	NO❗
L	swapETHForExactTokens	External ❗	💵	NO❗
L	quote	External ❗		NO❗
L	getAmountOut	External ❗		NO❗
L	getAmountIn	External ❗		NO❗
L	getAmountsOut	External ❗		NO❗
L	getAmountsIn	External ❗		NO❗

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

Inheritance Hierarchy
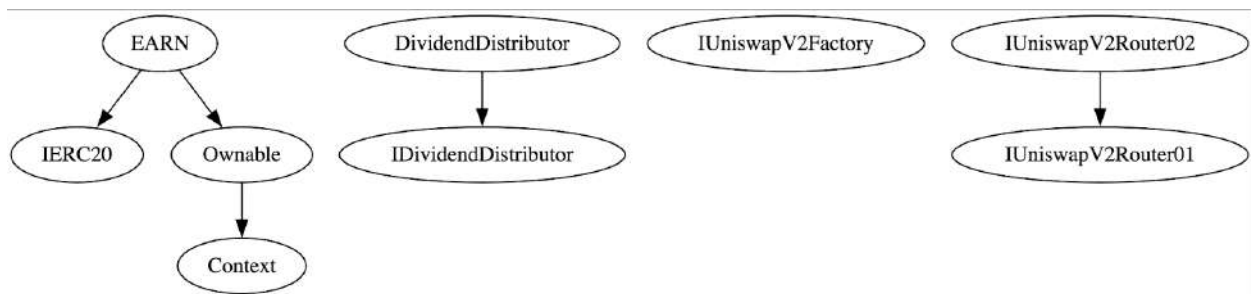
Security issue checking status

❖ **High severity issues**
   No High severity issues found

❖ **Medium severity issues**
   No medium severity issues found

❖ **Low severity issues**
   No low severity issues found

❖ **Centralization Risk**

The owner can blacklist any wallet from the contract

```
ftrace | funcSig
function blackListWallets(address _wallet↑, bool _status↑)
    external
    onlyOwner
{
    isBlacklisted[_wallet↑] = _status↑;
}
```

Auto LP tokens are not going to an unreachable address

```
ftrace | funcSig
function addLiquidity(uint256 tokenAmount↑, uint256 bnbAmount↑) private {
    _approve(address(this), address(router), tokenAmount↑);

    // add the liquidity
    router.addLiquidityETH{value: bnbAmount↑}(
        address(this),
        tokenAmount↑,
        0, // slippage is unavoidable
        0, // slippage is unavoidable
        lpReceiver,
        block.timestamp
    );
}
```

Owner privileges

❖ The owner can include/exclude wallets from the rewards.

```
ftrace | funcSig
function setIsDividendExempt(address holder↑, bool exempt↑)
    external
    onlyOwner
{
    require(
        holder↑ != address(this) && holder↑ != pair,
        "can not add pair and token address as share holder"
    );
    isDividendExempt[holder↑] = exempt↑;
    if (exempt↑) {
        dividendTracker.setShare(holder↑, 0);
    } else {
        dividendTracker.setShare(holder↑, _balances[holder↑]);
    }

    emit SetIsDividendExempt(holder↑, exempt↑);
}
```

❖ The owner can include/exclude wallets from the fees

```
ftrace | funcSig
function setIsFeeExempt(address holder↑, bool exempt↑) external onlyOwner {
    isFeeExempt[holder↑] = exempt↑;

    emit SetIsFeeExempt(holder↑, exempt↑);
}
```

❖ The owner can enable/disable swapping

```
ftrace | funcSig
function setDoContractSwap(bool _enabled↑) external onlyOwner {
    contractSwapEnabled = _enabled↑;

    emit SetDoContractSwap(_enabled↑);
}
```

20

❖ The owner can blacklist/whitelist wallets from the contract

```
ftrace | funcSig
function blackListWallets(address _wallet↑, bool _status↑)
    external
    onlyOwner
{
    isBlacklisted[_wallet↑] = _status↑;
}
```

❖ The owner can change minimum distribution period and minimum distribution amount

```
ftrace | funcSig
function setDistributionCriteria(
    uint256 _minPeriod↑,
    uint256 _minDistribution↑
) external onlyOwner {
    dividendTracker.setDistributionCriteria(_minPeriod↑, _minDistribution↑);

    emit ChangeDistributionCriteria(_minPeriod↑, _minDistribution↑);
}
```

❖ The owner can change max gas limit for auto distribution

```
ftrace | funcSig
function setDistributorSettings(uint256 gas↑) external onlyOwner {
    require(gas↑ < 750000);
    distributorGas = gas↑;
}
```

❖ The owner can change marketing/buyback and LP wallet

```
ftrace | funcSig
function changeMarketingWallet(address _wallet↑) external onlyOwner {
    marketingWallet = _wallet↑;
}

ftrace | funcSig
function changeBuyBackWallet(address _wallet↑) external onlyOwner {
    buyBackWallet = _wallet↑;
}

ftrace | funcSig
function changeLPWallet(address _wallet↑) external onlyOwner {
    lpReceiver = _wallet↑;
}
```

❖ The owner can change buy and sell fees maximum up to 15% (cannot change reward fee)

```
ftrace | funcSig
function changeBuyFees(
    uint256 _liquidityFee↑,
    uint256 _buyBackFee↑,
    uint256 _marketingFee↑
) external onlyOwner {
    buyLiquidityFee = _liquidityFee↑;
    buyBuyBackFee = _buyBackFee↑;
    buyMarketingFee = _marketingFee↑;

    buyTotalFee = rewardFee + _liquidityFee↑ + _buyBackFee↑ + _marketingFee↑;

    require(buyTotalFee <= 15, "Total fees can not greater than 15%");
}

ftrace | funcSig
function changeSellFees(
    uint256 _liquidityFee↑,
    uint256 _buyBackFee↑,
    uint256 _marketingFee↑
) external onlyOwner {
    sellLiquidityFee = _liquidityFee↑;
    sellBuyBackFee = _buyBackFee↑;
    sellMarketingFee = _marketingFee↑;

    sellTotalFee = rewardFee + _liquidityFee↑ + _buyBackFee↑ + _marketingFee↑;

    require(sellTotalFee <= 15, "Total fees can not greater than 15%");
}
```

❖ The owner can enable/disable contract sell cool down and can change cool down time

```
ftrace | funcSig
function setSellCollDown(bool _status↑, uint256 _coolDownTime↑)
    external
    onlyOwner
{
    isSellCoolDownEnabled = _status↑;
    sellCoolDownTime = _coolDownTime↑;
}
```

❖ The owner can change max buy and sell limit minimum up to 0.1%

```
ftrace | funcSig
function changeSellLimit(uint256 _limit↑) external onlyOwner {
    if (_limit↑ > 0)
        require(
            _limit↑ >= 100 * 10**6 * 10**_decimals,
            "Limit can not less than 250 million"
        );

    maxSellLimit = _limit↑;
}

ftrace | funcSig
function changeBuyLimit(uint256 _limit↑) external onlyOwner {
    if (_limit↑ > 0)
        require(
            _limit↑ >= 100 * 10**6 * 10**_decimals,
            "Limit can not less than 250 million"
        );
    maxBuyLimit = _limit↑;
}
```

❖ The owner can include/exclude wallets from max buy and sell limit

```
ftrace | funcSig
function excludeFromMaxSell(address _wallet↑, bool _status↑)
    external
    onlyOwner
{
    isMaxSellLimitExcluded[_wallet↑] = _status↑;
}

ftrace | funcSig
function excludeFromMaxBuy(address _wallet↑, bool _status↑)
    external
    onlyOwner
{
    isMaxBuyLimitExcluded[_wallet↑] = _status↑;
}
```

❖ The owner can enable trading once enabled cannot disable again

```
ftrace | funcSig
function enableTrading() external onlyOwner {
    isTradeEnabled = true;
}
```

❖ The owner can add/remove authorized wallets (authorized wallets can do transactions when trading is disable)

```
ftrace | funcSig
function setAuthorizedWallets(address _wallet↑, bool _status↑)
    external
    onlyOwner
{
    isAuthorized[_wallet↑] = _status↑;
}
```

❖ The owner can take ETH from the contract

```
ftrace | funcSig
function rescueEth() external onlyOwner {
    uint256 balance = address(this).balance;
    require(balance > 0, "No enough ETH to transfer");

    payable(msg.sender).transfer(balance);
}
```

❖ The owner can manually deposit reward tokens for the rewards

```
ftrace | funcSig
function depositRewards(uint256 _rewardAmount⬆) external onlyOwner {
    IERC20(REWARD).transferFrom(
        msg.sender,
        address(dividendTracker),
        _rewardAmount⬆
    );

    try dividendTracker.deposit(_rewardAmount⬆) {} catch {}
}
```

❖ The owner can take reward tokens from the reward tracker

```
ftrace | funcSig
function purgeBeforeSwitch() public onlyOwner {
    dividendTracker.purge(msg.sender);
}
```

25

Audit conclusion

RugFreeCoins team has performed in-depth testings, 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**

Number of risk issues: **2**

Solidity code functional issue level: **PASS**

Number of owner privileges: **16**

Centralization risk correlated to the active owner: **HIGH**

Smart contract active ownership: **ACTIVE**