Cyberscope

A TAC Security Company

Audit Report

Kart Rumble Token

October 2025

Repository	https://github.com/KR-HQ/kart-rumble-token
Commit	d3481d9ca2709a43c93dd11ae868486f38d1e100
Audited by	© cyberscope

Analysis

Critical Medium Minor / Informative Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Unresolved
●	OTUT	Transfers User's Tokens	Passed
●	ELFM	Exceeds Fees Limit	Passed
●	MT	Mints Tokens	Passed
●	BT	Burns Tokens	Passed
●	BC	Blacklists Addresses	Passed

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	PTRP	Potential Transfer Revert Propagation	Unresolved
	MC	Missing Check	Unresolved
	MCM	Misleading Comment Messages	Unresolved
	CCR	Contract Centralization Risk	Unresolved
	MEE	Missing Events Emission	Unresolved
	PVC	Price Volatility Concern	Unresolved

Table of Contents

Risk Classification

The criticality of findings in Cyberscope's smart contract audits is determined by evaluating multiple variables. The two primary variables are:

1. **Likelihood of Exploitation**: This considers how easily an attack can be executed, including the economic feasibility for an attacker.
2. **Impact of Exploitation**: This assesses the potential consequences of an attack, particularly in terms of the loss of funds or disruption to the contract's functionality.

Based on these variables, findings are categorized into the following severity levels:

1. **Critical**: Indicates a vulnerability that is both highly likely to be exploited and can result in significant fund loss or severe disruption. Immediate action is required to address these issues.
2. **Medium**: Refers to vulnerabilities that are either less likely to be exploited or would have a moderate impact if exploited. These issues should be addressed in due course to ensure overall contract security.
3. **Minor**: Involves vulnerabilities that are unlikely to be exploited and would have a minor impact. These findings should still be considered for resolution to maintain best practices in security.
4. **Informative**: Points out potential improvements or informational notes that do not pose an immediate risk. Addressing these can enhance the overall quality and robustness of the contract.

Severity	Likelihood / Impact of Exploitation
Critical	Highly Likely / High Impact
Medium	Less Likely / High Impact or Highly Likely/ Lower Impact
Minor / Informative	Unlikely / Low to no Impact

Review

Repository	https://github.com/KR-HQ/kart-rumble-token
Commit	d3481d9ca2709a43c93dd11ae868486f38d1e100

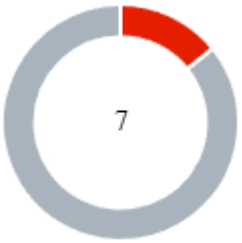
Audit Updates

Initial Audit	20 Oct 2025

Source Files

Filename	SHA256
KartRumbleToken.sol	6e99141bad877cb6eb5eeb8e3dfe905bc43c5b8a7f011937d6d096b393c2904c

Findings Breakdown



Critical	1
Medium	0
Minor / Informative	6

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	1	0	0	0
Medium	0	0	0	0
Minor / Informative	6	0	0	0

ST - Stops Transactions

Criticality	Critical
Location	KartRumbleToken.sol#L251
Status	Unresolved

Description

The transactions are initially disabled for all users excluding the authorized addresses. The owner can enable the transactions for all users. Once the transactions are enable the owner will not be able to disable them again.

```
Shell
function openTrading() external onlyOwner {
require(tradingStatus == TRADING_DISABLED, "Trading
already enabled");
   _openTrading();
}
```

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. Some suggestions are:

- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

PTRP - Potential Transfer Revert Propagation

Criticality	Minor / Informative
Location	KartRumbleToken.sol#L209
Status	Unresolved

Description

The contract sends funds to a `feeCollector` as part of the transfer flow. This address can either be a wallet address or a contract. If the address belongs to a contract then it may revert from incoming payment. As a result, the error will propagate to the token's contract and revert the transfer.

```
Shell
function _swapTaxes() internal lockTheSwap {
    uint256 tokenAmount = _min(balanceOf(address(this)),
MAX_TO_SWAP);
    if (tokenAmount < MIN_TO_SWAP) {
        return;
    }
    address ;
    path[0] = address(this);
    path[1] = weth;

    try
uniRouter.swapExactTokensForETHSupportingFeeOnTransferToke
ns(
        tokenAmount,
        0,
        path,
        feeCollector,
        block.timestamp
    ) {} catch {}
```

Recommendation

The contract should tolerate the potential revert from the underlying contracts when the interaction is part of the main transfer flow. This could be achieved by not allowing set contract addresses or by sending the funds in a non-revertable way.

MC - Missing Check

Criticality	Minor / Informative
Location	KartRumbleToken.sol#L64,90
Status	Unresolved

Description

The constructor accepts the router and factory addresses without performing validation checks. It does not ensure that these addresses are non-zero or that the router's associated factory matches the supplied factory address. In Uniswap-based setups, these components are typically linked within the same deployment

```shell
uniRouter = _uniswapV2Router;

uniFactory = _uniswapV2Factory;
```

Recommendation

It is suggested to include basic validation in the constructor to help ensure correct setup and minimize configuration errors.

MCM - Misleading Comment Messages

Criticality	Minor / Informative
Location	KartRumbleToken.sol#L191
Status	Unresolved

Description

The contract is using misleading comment messages. These comment messages do not accurately reflect the actual implementation, making it difficult to understand the source code. As a result, the users will not comprehend the source code's actual implementation.

```shell
// swap taxes first on buys or regular transfers
```

Recommendation

The team is advised to carefully review the comment in order to reflect the actual implementation. To improve code readability, the team should use more specific and descriptive comment messages.

CCR - Contract Centralization Risk

Criticality	Minor / Informative
Location	KartRumbleToken.sol#L119,124,241
Status	Unresolved

Description

The contract's functionality and behavior are heavily dependent on external parameters or configurations. While external configuration can offer flexibility, it also poses several centralization risks that warrant attention. Centralization risks arising from the dependence on external configuration include Single Point of Control, Vulnerability to Attacks, Operational Delays, Trust Dependencies, and Decentralization Erosion.

Shell

```
function excludeFromFees(address account) external
onlyFeeCollector {
        require(!isExcludedFromFee[account], "Account is
already excluded");
        isExcludedFromFee[account] = true;
    }
function includeInFees(address account) external
onlyFeeCollector {
        require(isExcludedFromFee[account], "Account is
not excluded");
        isExcludedFromFee[account] = false;
    }
function setFeeCollector(
        address payable newFeeCollector
    ) external onlyFeeCollector {
        require(newFeeCollector != address(0), "Zero
address");
        feeCollector = newFeeCollector;
```

Recommendation

To address this finding and mitigate centralization risks, it is recommended to evaluate the feasibility of migrating critical configurations and functionality into the contract's codebase itself. This approach would reduce external dependencies and enhance the contract's self-sufficiency. It is essential to carefully weigh the trade-offs between external configuration flexibility and the risks associated with centralization..

MEE - Missing Events Emission

Criticality	Minor / Informative
Location	KartRumbleToken.sol#L120,126,236,241,251
Status	Unresolved

Description

The contract performs actions and state mutations from external methods that do not result in the emission of events. Emitting events for significant actions is important as it allows external parties, such as wallets or dApps, to track and monitor the activity on the contract. Without these events, it may be difficult for external parties to accurately determine the current state of the contract. The `setFeeCollector()` function updates the `feeCollector` address without emitting an event `(FeeCollectorUpdated)`, reducing traceability of administrative changes.

```shell
function includeInFees(address account)

function excludeFromFees(address account)

function setTax(uint256 newTax)

function setFeeCollector( address payable
newFeeCollector)

function openTrading()
feeCollector = newFeeCollector
```

Recommendation

It is recommended to include events in the code that are triggered each time a significant action is taking place within the contract. These events should include relevant details such as the user's address and the nature of the action taken. By doing so, the contract will be more transparent and easily auditable by external parties. It will also help prevent potential issues or disputes that may arise in the future.

PVC - Price Volatility Concern

Criticality	Minor / Informative
Location	KartRumbleToken.sol#L209
Status	Unresolved

Description

The contract accumulates tokens from the taxes to swap them for ETH. The variable `MAX_TO_SWAP` sets a threshold where the contract will trigger the swap functionality. If the variable is set to a big number, then the contract will swap a huge amount of tokens for ETH. It is important to note that the price of the token representing it, can be highly volatile. This means that the value of a price volatility swap involving Ether could fluctuate significantly at the triggered point, potentially leading to significant price volatility for the parties involved.

```
Shell
uint256 constant MAX_TO_SWAP = (TOTAL_SUPPLY * 1) /
100;
function _swapTaxes() internal lockTheSwap {
        uint256 tokenAmount =
_min(balanceOf(address(this)), MAX_TO_SWAP);
        if (tokenAmount < MIN_TO_SWAP) {
            return;
```

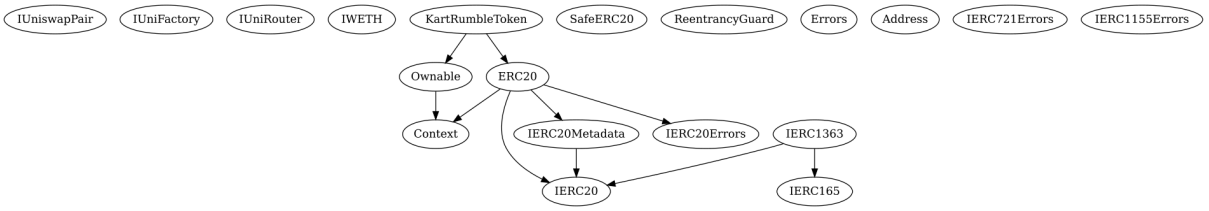
Recommendation

The contract could ensure that it will not sell more than a reasonable amount of tokens in a single transaction. A suggested implementation could check that the maximum amount should be less than a fixed percentage of the exchange reserves. Hence, the contract will guarantee that it cannot accumulate a huge amount of tokens in order to sell them.

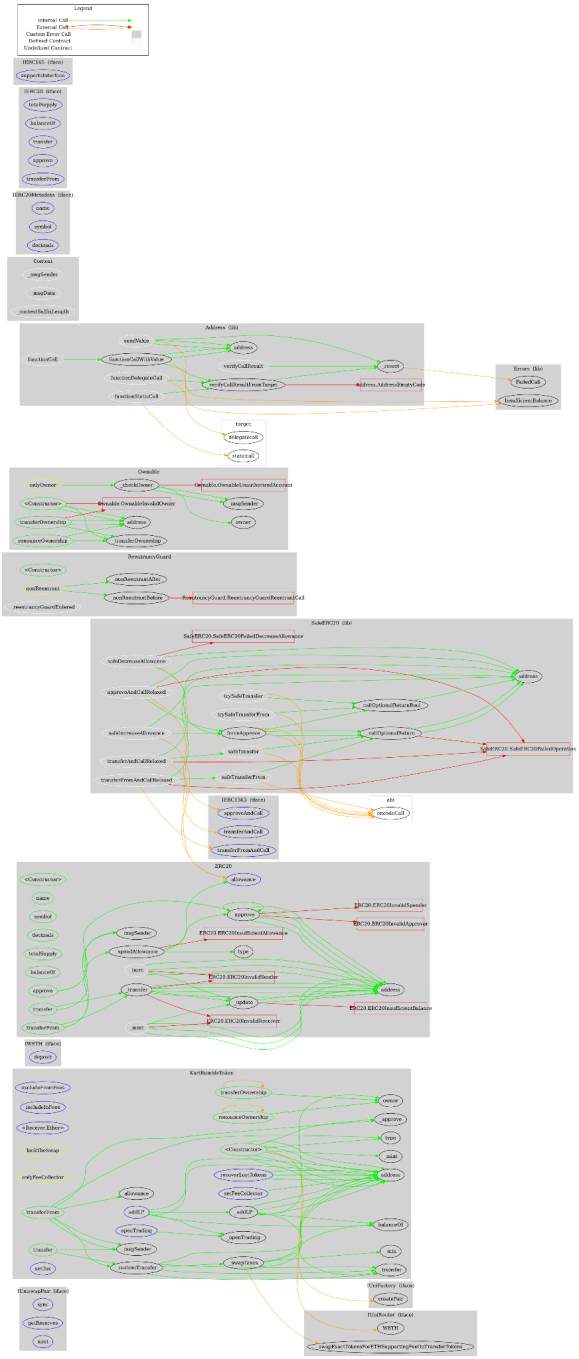
Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
KartRumbleToken	Implementation	ERC20, Ownable		
	transferOwnership	Public	✓	onlyOwner
	renounceOwnership	Public	✓	onlyOwner
	excludeFromFees	External	✓	onlyFeeCollector
	includeInFees	External	✓	onlyFeeCollector
	transfer	Public	✓	-
	transferFrom	Public	✓	-
	_customTransfer	Internal	✓	
	_swapTaxes	Internal	✓	lockTheSwap
	_min	Internal		
	setTax	External	✓	onlyFeeCollector
	setFeeCollector	External	✓	onlyFeeCollector
	openTrading	External	✓	onlyOwner
	_openTrading	Internal	✓	
	addLP	External	Payable	onlyOwner
	_addLP	Internal	✓	
	recoverLostTokens	External	✓	onlyFeeCollector

Inheritance Graph



Flow Graph



Summary

Kart Rumble contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. There are some functions that can be abused by the owner like stop transactions. A multi-wallet signing pattern will provide security against potential hacks. Renouncing ownership will eliminate all the contract threats.

Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. This report is not nor should be considered an "endorsement" or "disapproval" of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security Cyberscope's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

About Cyberscope

Cyberscope is a TAC blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.

A **TAC Security** Company

The Cyberscope team

cyberscope.io