Cyberscope

A TAC Security Company

Audit Report

Rumble Token

October 2025

Repository	https://github.com/KR-HQ/kart-rumble-token
Commit	6c983654459495e6ced3fb553789d47f3ebbd7c3
Audited by	© cyberscope

Analysis

● Critical ● Medium ● Minor / Informative ● Pass

Severity	Code	Description	Status
●	ST	Stops Transactions	Passed
●	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
⚪	MC	Missing Check	Acknowledged
⚪	CCR	Contract Centralization Risk	Acknowledged

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	6c983654459495e6ced3fb553789d47f3ebbd7c3

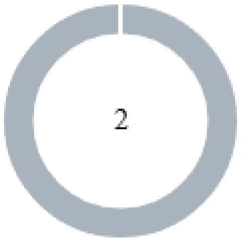
Audit Updates

Initial Audit	20 Oct 2025
Corrected Phase 2	24 Oct 2025

Source Files

Filename	SHA256
KartRumbleToken.sol	f98ec416e49c2ef0b4db3e73c72b4374f292bd2b34f885eca13a7bf3d51f4c8f

Findings Breakdown



● Critical	0
● Medium	0
● Minor / Informative	2

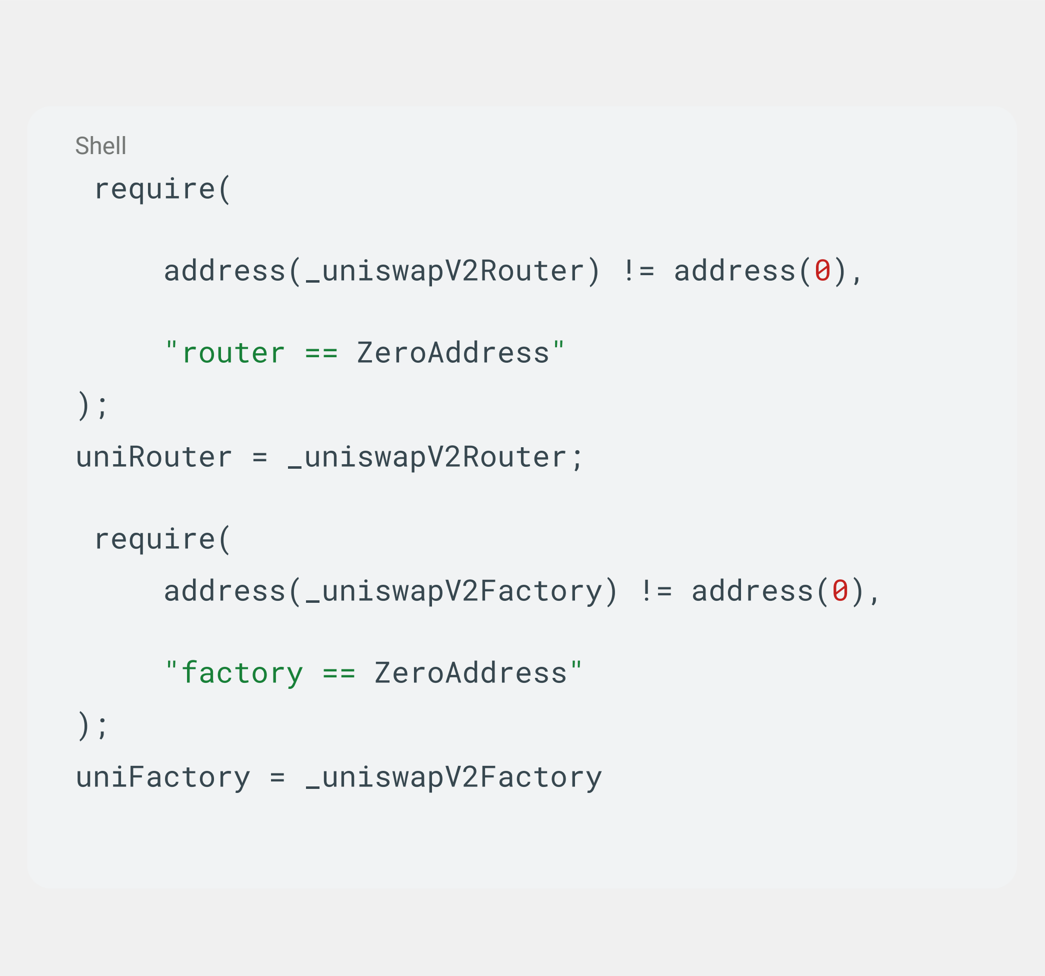
Severity	Unresolved	Acknowledged	Resolved	Other
● Critical	0	0	0	0
● Medium	0	0	0	0
● Minor / Informative	0	2	0	0

MC - Missing Check

Criticality	Minor / Informative
Location	KartRumbleToken.sol#L96
Status	Acknowledged

Description

The constructor accepts the router and factory addresses without performing validation checks. Specifically, in Uniswap-based setups, these components are typically linked within the same deployment. The contract does not ensure that the provided router corresponds to the provided factory and vice versa.



Shell

```
 require(

     address(_uniswapV2Router) != address(0),

     "router == ZeroAddress"
);
uniRouter = _uniswapV2Router;

 require(
     address(_uniswapV2Factory) != address(0),

     "factory == ZeroAddress"
);
uniFactory = _uniswapV2Factory
```

Recommendation

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

Team Update

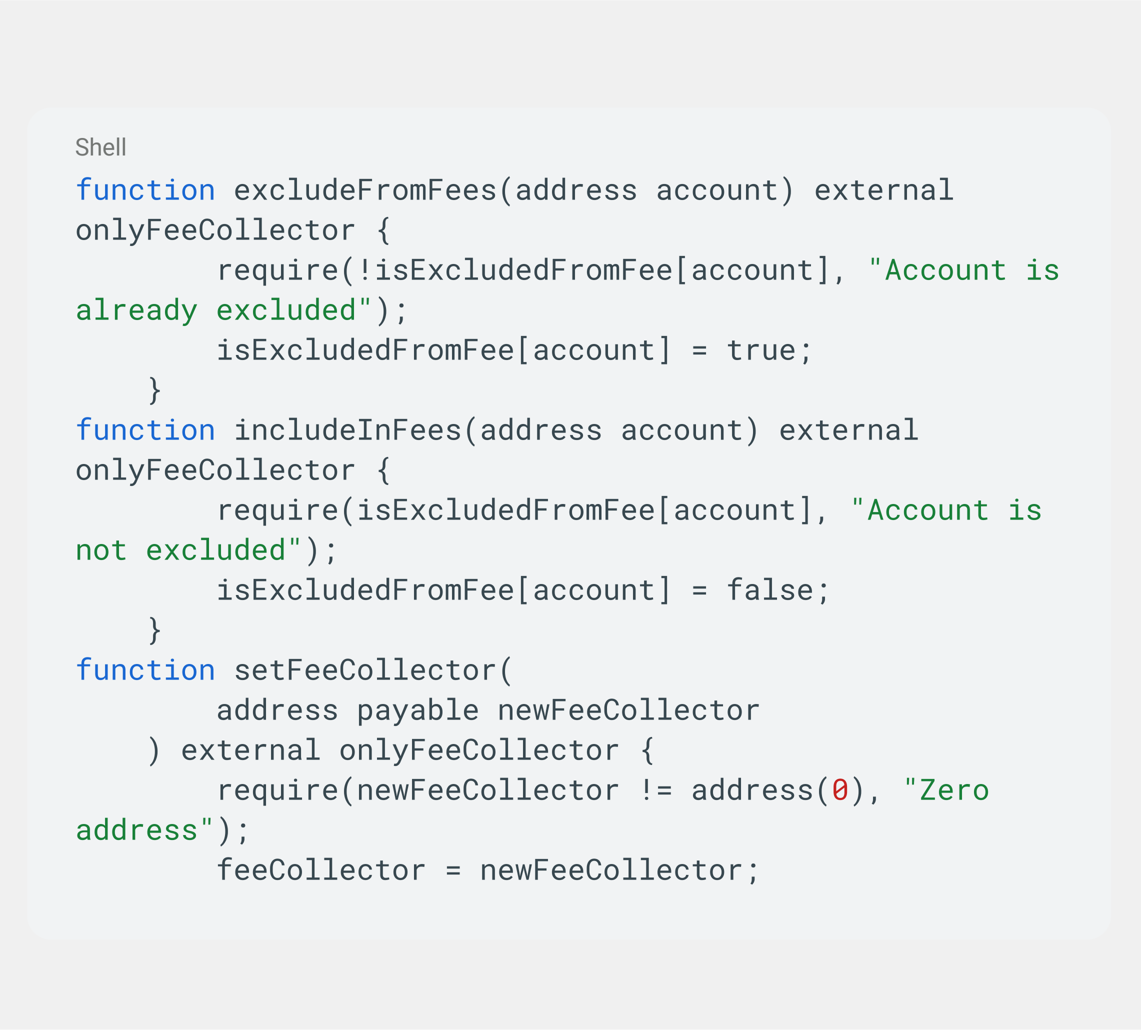
The team acknowledged the issue, we believe the check before deployment is sufficient

CCR - Contract Centralization Risk

Criticality	Minor / Informative
Location	KartRumbleToken.sol#L136,141,265
Status	Acknowledged

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.

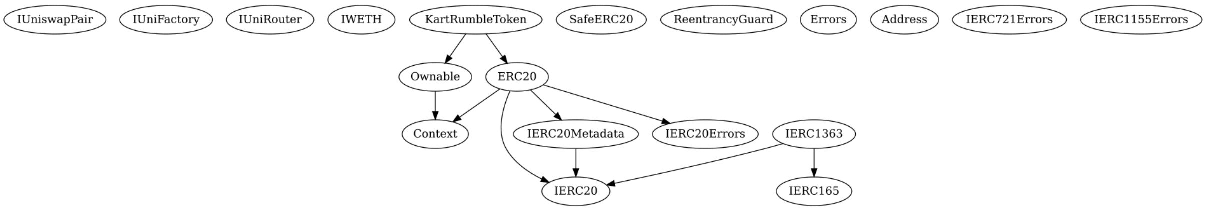
Team Update

The team is aware of this centralized behavior, it is necessary to achieve the normal intended operations of the token, and could not be used in a malicious way to disturb users' transactions.

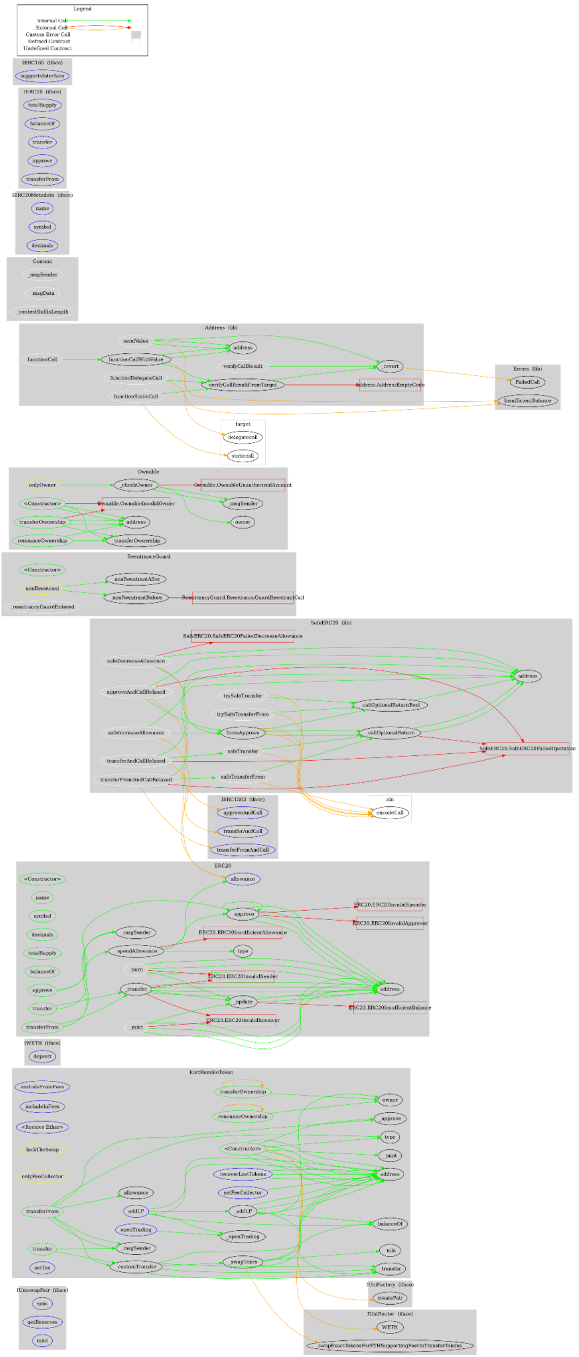
Functions Analysis

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


Legend
Internal Call
External Call
Custom Error Call
Defined Contract
Undefined Contract
IERC165 (iface)
supportsInterface
IERC20 (iface)
totalSupply
balanceOf
transfer
approve
transferFrom
IERC20Metadata (iface)
name
symbol
decimals
Context
_msgSender
msgData
_contextSuffixLength
Address (lib)
sendValue
address
functionCall
functionCallWithValue
verifyCallResult
_revert
functionDelegateCall
verifyCallResultFromTarget
Address.AddressEmptyCode
functionStaticCall
Errors (lib)
FailedCall
InsufficientBalance
target
delegatecall
staticcall
Ownable
onlyOwner
_checkOwner
Ownable.OwnableUnauthorizedAccount
<Constructor>
Ownable.OwnableInvalidOwner
_msgSender
transferOwnership
address
owner
renounceOwnership
_transferOwnership
ReentrancyGuard
<Constructor>
_nonReentrantAfter
nonReentrant
_nonReentrantBefore
ReentrancyGuard.ReentrancyGuardReentrantCall
_reentrancyGuardEntered
SafeERC20 (lib)
SafeERC20.SafeERC20FailedDecreaseAllowance
safeDecreaseAllowance
address
approveAndCallRelaxed
trySafeTransfer
_callOptionalReturnBool
trySafeTransferFrom
safeIncreaseAllowance
forceApprove
_callOptionalReturn
safeTransfer
SafeERC20.SafeERC20FailedOperation
transferAndCallRelaxed
transferFromAndCallRelaxed
safeTransferFrom
IERC1363 (iface)
approveAndCall
transferAndCall
transferFromAndCall
abi
encodeCall
ERC20
<Constructor>
allowance
name
symbol
approve
ERC20.ERC20InvalidSpender
ERC20.ERC20InvalidApprover
decimals
_msgSender
ERC20.ERC20InsufficientAllowance
totalSupply
_spendAllowance
type
balanceOf
_burn
ERC20.ERC20InvalidSender
approve
_transfer
address
transfer
_update
ERC20.ERC20InsufficientBalance
transferFrom
_mint
ERC20.ERC20InvalidReceiver
IWETH (iface)
deposit
KartRumbleToken
excludeFromFees
transferOwnership
owner
includeInFees
renounceOwnership
_approve
<Receive Ether>
type
lockTheSwap
<Constructor>
_mint
onlyFeeCollector
recoverLostTokens
address
setFeeCollector
allowance
transferFrom
addLP
_addLP
balanceOf
openTrading
_openTrading
transfer
_msgSender
min
setTax
customTransfer
_swapTaxes
transfer
IUniFactory (iface)
createPair
IUniswapPair (iface)
sync
getReserves
mint
IUniRouter (iface)
WETH
swapExactTokensForETHSupportingFeeOnTransferTokens

Summary

Kart Rumble contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. Kart Rumble is an interesting project that has a friendly and growing community. The Smart Contract analysis reported no compiler error or critical issues. The contract Owner can access some admin functions that can not be used in a malicious way to disturb the users' transactions.

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