Cyberscope

Audit Report

Ring AI

March 2024

Network	ETH
Address	0xc092a137df3cf2b9e5971ba1874d26487c12626d
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	Unresolved
	MT	Mints Tokens	Passed
	BT	Burns Tokens	Passed
	BC	Blacklists Addresses	Unresolved

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	PBV	Percentage Boundaries Validation	Unresolved
	PLPI	Potential Liquidity Provision Inadequacy	Unresolved
	PMRM	Potential Mocked Router Manipulation	Unresolved
	PTRP	Potential Transfer Revert Propagation	Unresolved
	PVC	Price Volatility Concern	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved
	L07	Missing Events Arithmetic	Unresolved
	L14	Uninitialized Variables in Local Scope	Unresolved
	L16	Validate Variable Setters	Unresolved
	L20	Succeeded Transfer Check	Unresolved

Table of Contents

Review

Contract Name	RingAIToken
Compiler Version	v0.8.4+commit.c7e474f2
Optimization	200 runs
Explorer	https://etherscan.io/address/0xc092a137df3cf2b9e5971ba1874d26487c12626d
Address	0xc092a137df3cf2b9e5971ba1874d26487c12626d
Network	ETH
Symbol	RING
Decimals	18
Total Supply	100,000,000
Badge Eligibility	Must Fix Criticals

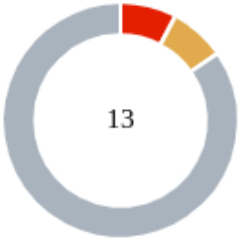
Audit Updates

Initial Audit	14 Mar 2024

Source Files

Filename	SHA256
src/active/RingAIToken.sol	4ef2d9652f45e203ee7fe21171cb72234a609b21f830df91f25cf48b01cadac3

Findings Breakdown



Critical	1
Medium	1
Minor / Informative	11

Severity	Unresolved	Acknowledged	Resolved	Other
Critical	1	0	0	0
Medium	1	0	0	0
Minor / Informative	11	0	0	0

ST - Stops Transactions

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L97,99
Status	Unresolved

Description

The contract owner has the authority to stop the sales for all users excluding the whitelist addresses. The owner may take advantage of it by setting the `tradeStartTime` or `tradeMaxAmount` to zero. As a result, the contract may operate as a honeypot.

```
require(tradeStartTime > 0 && tradeStartTime <=
block.timestamp, "Invalid time");
// Cannot transfer exceed trade max amount
require(_pAmount <= tradeMaxAmount, "Invalid amount");
```

Recommendation

The contract could embody a check for not allowing setting the `tradeMaxAmount` less than a reasonable amount and the `tradeStartTime` after its initialization. A suggested implementation could check that the minimum amount should be more than a fixed percentage of the total supply. 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.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

ELFM - Exceeds Fees Limit

Criticality	Critical
Location	src/active/RingAIToken.sol#L171
Status	Unresolved

Description

The contract owner has the authority to increase over the allowed limit of 25%. The owner may take advantage of it by calling the `setTaxFeePercent` function with a high percentage value.

```
function fConfigTax(uint _pBuyTax, uint _pSellTax, uint
_pTransferTax) external onlyOwner {
    buyTax = _pBuyTax;
    sellTax = _pSellTax;
    transferTax = _pTransferTax;
}
```

Recommendation

The contract could embody a check for the maximum acceptable value. 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.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

BC - Blacklists Addresses

Criticality	Medium
Location	src/active/RingAIToken.sol#L79
Status	Unresolved

Description

The contract owner has the authority to stop addresses from transactions. The owner may take advantage of it by calling the `fSetBlacklist` function.

```
 function fSetBlacklist(address _pAccount, bool _pStatus)
external onlyOwner {
    require(isInBlacklist[_pAccount] != _pStatus, "0x1");
    isInBlacklist[_pAccount] = _pStatus;
  }
```

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.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

PBV - Percentage Boundaries Validation

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L171
Status	Unresolved

Description

The contract utilizes variables for percentage-based calculations that are required for its operations. These variables are involved in multiplication and division operations to determine proportions related to the contract's logic. If such variables are set to values beyond their logical or intended maximum limits, it could result in incorrect calculations. This misconfiguration has the potential to cause unintended behavior or financial discrepancies, affecting the contract's integrity and the accuracy of its calculations.

```
 function fConfigTax(uint _pBuyTax, uint _pSellTax, uint
_pTransferTax) external onlyOwner {
    buyTax = _pBuyTax;
    sellTax = _pSellTax;
    transferTax = _pTransferTax;
  }
```

Recommendation

To mitigate risks associated with boundary violations, it is important to implement validation checks for variables used in percentage-based calculations. Ensure that these variables do not exceed their maximum logical values. This can be accomplished by incorporating `require` statements or similar validation mechanisms whenever such variables are assigned or modified. These safeguards will enforce correct operational boundaries, preserving the contract's intended functionality and preventing computational errors.

PLPI - Potential Liquidity Provision Inadequacy

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L261
Status	Unresolved

Description

The contract operates under the assumption that liquidity is consistently provided to the pair between the contract's token and the native currency. However, there is a possibility that liquidity is provided to a different pair. This inadequacy in liquidity provision in the main pair could expose the contract to risks. Specifically, during eligible transactions, where the contract attempts to swap tokens with the main pair, a failure may occur if liquidity has been added to a pair other than the primary one. Consequently, transactions triggering the swap functionality will result in a revert.

```
address weth_ = IUniswapRouter02(dexRouter).WETH();
address[] memory path_ = new address[](2);
path_[0] = address(this);
path_[1] = weth_;
uint initialBalance_ = address(taxHolder).balance;
IUniswapRouter02(dexRouter).swapExactTokensForETHSupportingFeeOnTransferTokens(
  taxProcess,
  0,
  path_,
  taxHolder,
  block.timestamp
);
```

Recommendation

The team is advised to implement a runtime mechanism to check if the pair has adequate liquidity provisions. This feature allows the contract to omit token swaps if the pair does not have adequate liquidity provisions, significantly minimizing the risk of potential failures.

Furthermore, the team could ensure the contract has the capability to switch its active pair in case liquidity is added to another pair.

Additionally, the contract could be designed to tolerate potential reverts from the swap functionality, especially when it is a part of the main transfer flow. This can be achieved by executing the contract's token swaps in a non-reversible manner, thereby ensuring a more resilient and predictable operation.

PMRM - Potential Mocked Router Manipulation

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L129
Status	Unresolved

Description

The contract includes a method that allows the owner to modify the router address and create a new pair. While this feature provides flexibility, it introduces a security threat. The owner could set the router address to any contract that implements the router's interface, potentially containing malicious code. In the event of a transaction triggering the swap functionality with such a malicious contract as the router, the transaction may be manipulated.

```
 function fSetDexInfo(address _pDexRouter, address _pToken2)
external onlyOwner {
    dexRouter = _pDexRouter;
    IUniswapRouter02 router_ = IUniswapRouter02(dexRouter);
    IUniswapFactory factory_ =
IUniswapFactory(router_.factory());
    address lpAddress_ = factory_.getPair(address(this),
_pToken2);
    if (lpAddress_ == address(0)) {
      lpAddress_ = factory_.createPair(address(this),
_pToken2);
    }
    dexLP = lpAddress_;
  }
```

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.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

PTRP - Potential Transfer Revert Propagation

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L306
Status	Unresolved

Description

The contract sends funds to a `taxHolder` 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.

```
IUniswapRouter02(dexRouter)
    .swapExactTokensForETHSupportingFeeOnTransferTokens(
        taxProcess,
        0,
        path_,
        taxHolder,
        block.timestamp
    );
```

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.

PVC - Price Volatility Concern

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L292
Status	Unresolved

Description

The contract accumulates tokens from the taxes to swap them for ETH. The variable `taxThreshold` 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.

```
function _processAllTax() private {
    uint taxProcess = totalTaxCollected();
    if (taxProcess >= taxThreshold) {
        // Reset tax collected
        _resetAllTax();
        // Swap to ETH
        _approve(address(this), dexRouter, taxProcess);
        address weth_ = IUniswapRouter02(dexRouter).WETH();
        address[] memory path_ = new address[](2);
        path_[0] = address(this);
        path_[1] = weth_;
        uint initialBalance_ = address(taxHolder).balance;
        IUniswapRouter02(dexRouter)

.swapExactTokensForETHSupportingFeeOnTransferTokens(
                taxProcess,
                0,
                path_,
                taxHolder,
                block.timestamp
            );
```

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.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L129,143,151,161,171,180,187,195,202,210
Status	Unresolved

Description

The Solidity style guide is a set of guidelines for writing clean and consistent Solidity code. Adhering to a style guide can help improve the readability and maintainability of the Solidity code, making it easier for others to understand and work with.

The followings are a few key points from the Solidity style guide:

1. Use camelCase for function and variable names, with the first letter in lowercase (e.g., myVariable, updateCounter).
2. Use PascalCase for contract, struct, and enum names, with the first letter in uppercase (e.g., MyContract, UserStruct, ErrorEnum).
3. Use uppercase for constant variables and enums (e.g., MAX_VALUE, ERROR_CODE).
4. Use indentation to improve readability and structure.
5. Use spaces between operators and after commas.
6. Use comments to explain the purpose and behavior of the code.
7. Keep lines short (around 120 characters) to improve readability.

```
address _pDexRouter
address _pToken2
address _pAccount
bool _pStatus
uint _pMaxAmount
uint _pStartTime
uint _pSellTax
uint _pTransferTax
uint _pBuyTax
uint _pTaxThreshold
uint _pTaxEndTime
address _pTaxHolder
uint256 _pAmount
address _pTo

...
```

Recommendation

By following the Solidity naming convention guidelines, the codebase increased the readability, maintainability, and makes it easier to work with.
Find more information on the Solidity documentation
https://docs.soliditylang.org/en/v0.8.17/style-guide.html#naming-convention.

L07 - Missing Events Arithmetic

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L162,172,181,188
Status	Unresolved

Description

Events are a way to record and log information about changes or actions that occur within a contract. They are often used to notify external parties or clients about events that have occurred within the contract, such as the transfer of tokens or the completion of a task.

It's important to carefully design and implement the events in a contract, and to ensure that all required events are included. It's also a good idea to test the contract to ensure that all events are being properly triggered and logged.

```
tradeStartTime = _pStartTime
buyTax = _pBuyTax
taxThreshold = _pTaxThreshold
taxEndTime = _pTaxEndTime
```

Recommendation

By including all required events in the contract and thoroughly testing the contract's functionality, the contract ensures that it performs as intended and does not have any missing events that could cause issues with its arithmetic.

L14 - Uninitialized Variables in Local Scope

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L107
Status	Unresolved

Description

Using an uninitialized local variable can lead to unpredictable behavior and potentially cause errors in the contract. It's important to always initialize local variables with appropriate values before using them.

```
uint taxAmount_
```

Recommendation

By initializing local variables before using them, the contract ensures that the functions behave as expected and avoid potential issues.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L60,130,196
Status	Unresolved

Description

The contract performs operations on variables that have been configured on user-supplied input. These variables are missing of proper check for the case where a value is zero. This can lead to problems when the contract is executed, as certain actions may not be properly handled when the value is zero.

```
taxHolder = sender_
dexRouter = _pDexRouter
taxHolder = _pTaxHolder
```

Recommendation

By adding the proper check, the contract will not allow the variables to be configured with zero value. This will ensure that the contract can handle all possible input values and avoid unexpected behavior or errors. Hence, it can help to prevent the contract from being exploited or operating unexpectedly.

L20 - Succeeded Transfer Check

Criticality	Minor / Informative
Location	src/active/RingAIToken.sol#L217
Status	Unresolved

Description

According to the ERC20 specification, the transfer methods should be checked if the result is successful. Otherwise, the contract may wrongly assume that the transfer has been established.

```
token_.transfer(_pTo, _pAmount)
```

Recommendation

The contract should check if the result of the transfer methods is successful. The team is advised to check the SafeERC20 library from the Openzeppelin library.

Functions Analysis

Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
RingAIToken	Implementation	ERC20, Ownable		
		External	Payable	-
		Public	✓	ERC20
	totalTaxCollected	Public		-
	_transfer	Internal	✓	
	fSetDexInfo	External	✓	onlyOwner
	fSetBlacklist	External	✓	onlyOwner
	fSetWhitelist	External	✓	onlyOwner
	fConfigTrade	External	✓	onlyOwner
	fConfigTax	External	✓	onlyOwner
	fConfigTaxThreshold	External	✓	onlyOwner
	fConfigTaxEndTime	External	✓	onlyOwner
	fConfigTaxHolder	External	✓	onlyOwner
	fEmergencyEth	External	✓	onlyOwner
	fEmergencyToken	External	✓	onlyOwner
	fBurnAllTax	External	✓	onlyGranted
	fClaimAllTax	External	✓	onlyGranted
	_resetAllTax	Private	✓	
	_processAllTax	Private	✓	

IWETH	Interface			
	deposit	External	Payable	-
	transfer	External	✓	-
	withdraw	External	✓	-
	approve	External	✓	-
	balanceOf	External		-
IUniswapRouter02	Interface	IUniswapRouter01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
IUniswapRouter01	Interface			
	factory	External		-
	WETH	External		-
	addLiquidity	External	✓	-
	addLiquidityETH	External	Payable	-
	removeLiquidity	External	✓	-
	removeLiquidityETH	External	✓	-

	removeLiquidityWithPermit	External	✓	-
	removeLiquidityETHWithPermit	External	✓	-
	swapExactTokensForTokens	External	✓	-
	swapTokensForExactTokens	External	✓	-
	swapExactETHForTokens	External	Payable	-
	swapTokensForExactETH	External	✓	-
	swapExactTokensForETH	External	✓	-
	swapETHForExactTokens	External	Payable	-
	quote	External		-
	getAmountOut	External		-
	getAmountIn	External		-
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapFactory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-
	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-

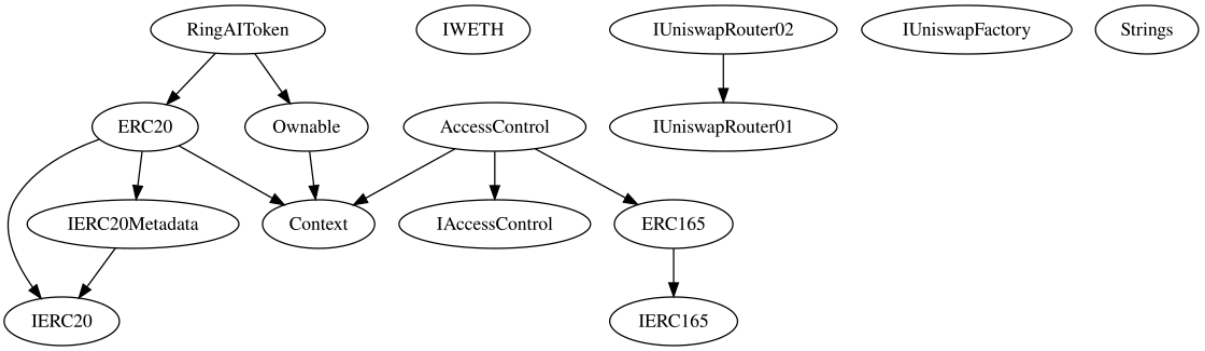
Strings	Library			
	toString	Internal		
	toHexString	Internal		
	toHexString	Internal		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
IERC165	Interface			
	supportsInterface	External		-
ERC165	Implementation	IERC165		
	supportsInterface	Public		-
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-

ERC20	Implementation	Context, IERC20, IERC20Metadata		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	_transfer	Internal	✓	
	_mint	Internal	✓	
	_burn	Internal	✓	
	_approve	Internal	✓	
	_beforeTokenTransfer	Internal	✓	
	_afterTokenTransfer	Internal	✓	
IERC20Metadata	Interface	IERC20		

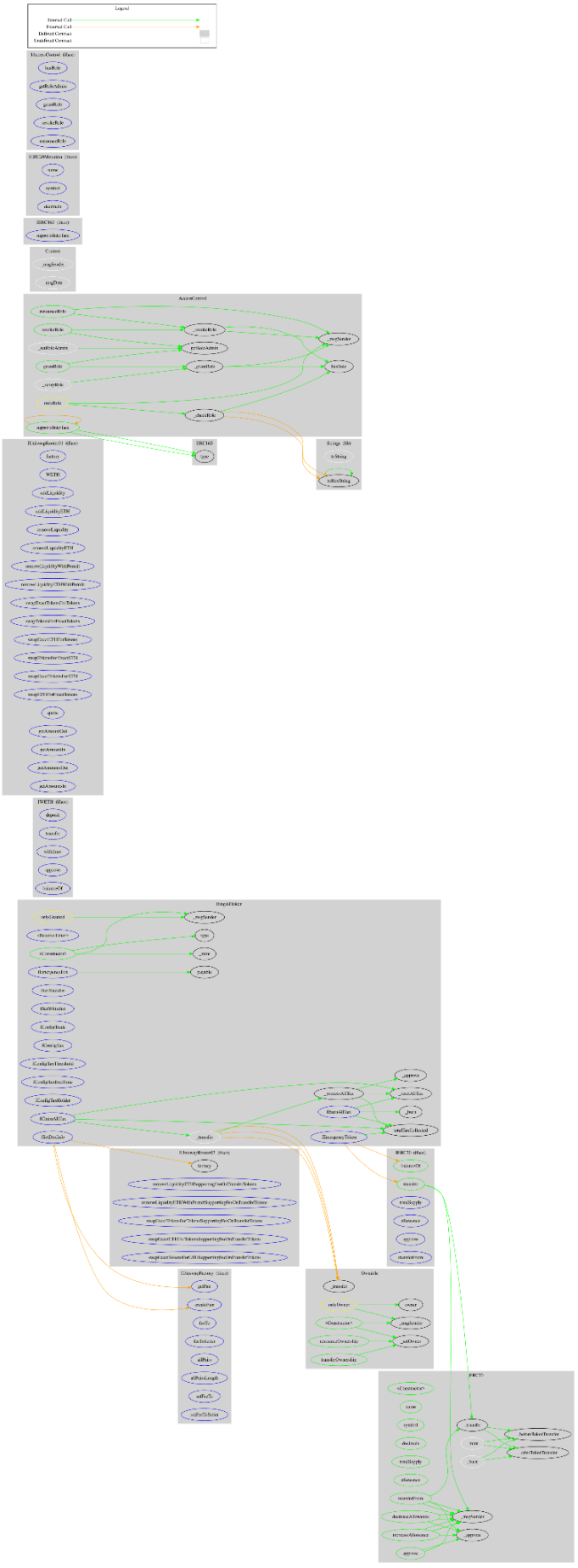
	name	External		-
	symbol	External		-
	decimals	External		-
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_setOwner	Private	✓	
IAccessControl	Interface			
	hasRole	External		-
	getRoleAdmin	External		-
	grantRole	External	✓	-
	revokeRole	External	✓	-
	renounceRole	External	✓	-
AccessControl	Implementation	Context, IAccessControl, ERC165		
	supportsInterface	Public		-
	hasRole	Public		-
	_checkRole	Internal		
	getRoleAdmin	Public		-

	grantRole	Public	✓	onlyRole
	revokeRole	Public	✓	onlyRole
	renounceRole	Public	✓	-
	_setupRole	Internal	✓	
	_setRoleAdmin	Internal	✓	
	_grantRole	Private	✓	
	_revokeRole	Private	✓	

Inheritance Graph



Flow Graph



Summary

Ring AI 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, manipulate the fees and blacklist addresses. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or 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 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.

The Cyberscope team

https://www.cyberscope.io