Cyberscope

Audit Report

CY9NI

June 2023

SHA256 2ca6c61650f10595bd1f4959b5717e69d59438646b97d050eb4b16c543707151

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	Unresolved

Diagnostics

● Critical ● Medium ● Minor / Informative

Severity	Code	Description	Status
●	TUU	Time Units Usage	Unresolved
●	L04	Conformance to Solidity Naming Conventions	Unresolved
●	L07	Missing Events Arithmetic	Unresolved
●	L09	Dead Code Elimination	Unresolved
●	L17	Usage of Solidity Assembly	Unresolved
●	L20	Succeeded Transfer Check	Unresolved

Table of Contents

Review

Audit Updates

Contract Name	Cy9niToken
Testing Deploy	https://testnet.bscscan.com/address/0x3fb8dc7cc17c2c73dacdbb486a27ad7fd75e5eff
Symbol	C9
Decimals	8
Total Supply	2,970,000,000

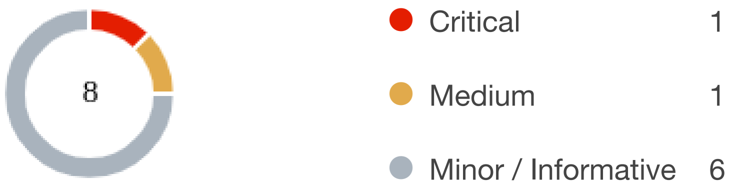
Audit Updates

Initial Audit	17 May 2023 https://github.com/cyberscope-io/audits/blob/main/cgi/v1/audit.pdf
Corrected Phase 2	21 Jun 2023

Source Files

Filename	SHA256
contracts/testingDeploy/C9.sol	2ca6c61650f10595bd1f4959b5717e69d59438646b97d050eb4b16c543707151

Findings Breakdown



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

ST - Stops Transactions

Criticality	Critical
Location	C9.sol#L1395
Status	Unresolved

Description

Initially, the contract does not allow the non-excluded addresses to transfer tokens. The restriction can be resumed once the contract owner enables them.

```
require(
    isTradingEnabled || _isExcludedFromFee[from],
    "Trading not enabled yet"
);
```

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 time-locker mechanism with a reasonable delay.
- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.
- Renouncing the ownership will eliminate the threats but it is non-reversible.

BC - Blacklists Addresses

Criticality	Medium
Location	C9.sol#L1016
Status	Unresolved

Description

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

```
function setBlackList(address addr, bool value) external
onlyOwner {
        _isBlackListed[addr] = value;
}
```

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 time-locker mechanism with a reasonable delay.
- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.
- Renouncing the ownership will eliminate the threats but it is non-reversible.

TUU - Time Units Usage

Criticality	Minor / Informative
Location	C9.sol#L633
Status	Unresolved

Description

The contract is using arbitrary numbers to form time-related values. As a result, it decreases the readability of the codebase and prevents the compiler to optimize the source code.

```
uint256 private constant MAX_COOLDOWN = 120;
```

Recommendation

It is a good practice to use the time units reserved keywords like `seconds` , `minutes` , `hours` , `days` , `weeks` and `years` to process time-related calculations.

It's important to note that these time units are simply a shorthand notation for representing time in seconds, and do not have any effect on the actual passage of time or the execution of the contract. The time units are simply a convenience for expressing time in a more human-readable form.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	C9.sol#L474,622,630,635,636,637
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.

```
function WETH() external pure returns (address);
mapping(address => bool) public _isBlackListed
uint256 private constant _tTotal = 297 * 10**6 * 10**9
string private constant _name = "cy9ni"
string private constant _symbol = "C9"
uint8 private constant _decimals = 8
```

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	C9.sol#L915
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.

```
coolDown = timeInSeconds
```

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.

L09 - Dead Code Elimination

Criticality	Minor / Informative
Location	C9.sol#L122,151,165,184,198,217,227,242,252,267,293,307,322,326
Status	Unresolved

Description

In Solidity, dead code is code that is written in the contract, but is never executed or reached during normal contract execution. Dead code can occur for a variety of reasons, such as:

- Conditional statements that are always false.
- Functions that are never called.
- Unreachable code (e.g., code that follows a return statement).

Dead code can make a contract more difficult to understand and maintain, and can also increase the size of the contract and the cost of deploying and interacting with it.

```
function sendValue(address payable recipient, uint256 amount)
internal {
        if (address(this).balance < amount) {
            revert AddressInsufficientBalance(address(this));
        }

        (bool success, ) = recipient.call{value: amount}("");
        if (!success) {
            revert FailedInnerCall();
        }
    }

function functionCall(address target, bytes memory data)
internal returns (bytes memory) {
        return functionCallWithValue(target, data, 0,
defaultRevert);
    }

...
```

Recommendation

To avoid creating dead code, it's important to carefully consider the logic and flow of the contract and to remove any code that is not needed or that is never executed. This can help improve the clarity and efficiency of the contract.

L17 - Usage of Solidity Assembly

Criticality	Minor / Informative
Location	C9.sol#L331
Status	Unresolved

Description

Using assembly can be useful for optimizing code, but it can also be error-prone. It's important to carefully test and debug assembly code to ensure that it is correct and does not contain any errors.

Some common types of errors that can occur when using assembly in Solidity include Syntax, Type, Out-of-bounds, Stack, and Revert.

```
assembly {
                let returndata_size := mload(returndata)
                revert(add(32, returndata), returndata_size)
            }
```

Recommendation

It is recommended to use assembly sparingly and only when necessary, as it can be difficult to read and understand compared to Solidity code.

L20 - Succeeded Transfer Check

Criticality	Minor / Informative
Location	C9.sol#L1149
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.

```
erc20token.transfer(owner(), balance)
```

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**
IERC20	Interface			
	totalSupply	External		-
	balanceOf	External		-
	transfer	External	✓	-
	allowance	External		-
	approve	External	✓	-
	transferFrom	External	✓	-
Address	Library			
	sendValue	Internal	✓	
	functionCall	Internal	✓	
	functionCall	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionCallWithValue	Internal	✓	
	functionStaticCall	Internal		
	functionStaticCall	Internal		
	functionDelegateCall	Internal	✓	
	functionDelegateCall	Internal	✓	

	verifyCallResultFromTarget	Internal		
	verifyCallResult	Internal		
	verifyCallResult	Internal		
	defaultRevert	Internal		
	_revert	Private		
Context	Implementation			
	_msgSender	Internal		
	_msgData	Internal		
Ownable	Implementation	Context		
		Public	✓	-
	owner	Public		-
	_checkOwner	Internal		
	renounceOwnership	Public	✓	onlyOwner
	transferOwnership	Public	✓	onlyOwner
	_transferOwnership	Internal	✓	
IUniswapV2Factory	Interface			
	feeTo	External		-
	feeToSetter	External		-
	getPair	External		-
	allPairs	External		-

	allPairsLength	External		-
	createPair	External	✓	-
	setFeeTo	External	✓	-
	setFeeToSetter	External	✓	-
IUniswapV2Router01	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		-

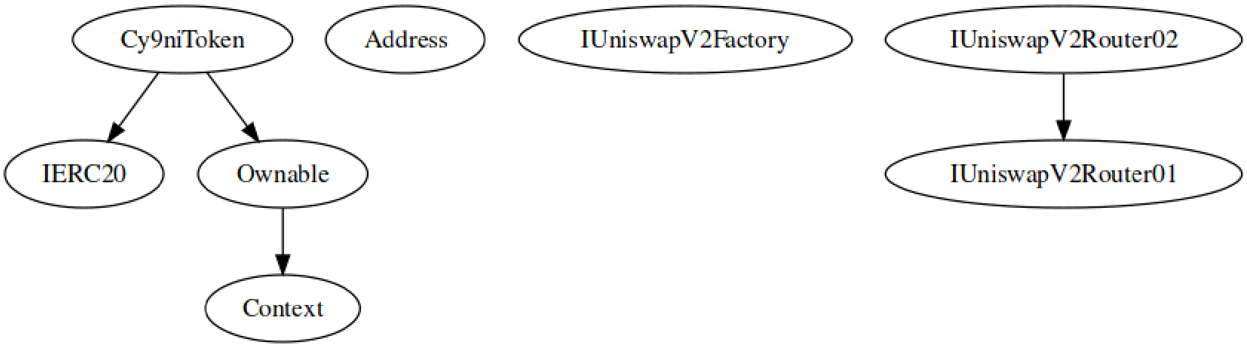
	getAmountsOut	External		-
	getAmountsIn	External		-
IUniswapV2Router02	Interface	IUniswapV2Router01		
	removeLiquidityETHSupportingFeeOnTransferTokens	External	✓	-
	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External	✓	-
	swapExactTokensForTokensSupportingFeeOnTransferTokens	External	✓	-
	swapExactETHForTokensSupportingFeeOnTransferTokens	External	Payable	-
	swapExactTokensForETHSupportingFeeOnTransferTokens	External	✓	-
Cy9niToken	Implementation	IERC20, Ownable		
		Public	✓	-
	name	Public		-
	symbol	Public		-
	decimals	Public		-
	totalSupply	Public		-
	balanceOf	Public		-
	transfer	Public	✓	-
	allowance	Public		-
	approve	Public	✓	-
	transferFrom	Public	✓	-
	_spendAllowance	Internal	✓	

	increaseAllowance	Public	✓	-
	decreaseAllowance	Public	✓	-
	enableTrading	External	✓	onlyOwner
	setCooldown	External	✓	onlyOwner
	isExcludedFromReward	Public		-
	totalFees	Public		-
	deliver	Public	✓	-
	reflectionFromToken	Public		-
	tokenFromReflection	Public		-
	excludeFromReward	Public	✓	onlyOwner
	includeInReward	External	✓	onlyOwner
	excludeFromFee	External	✓	onlyOwner
	setBlackList	External	✓	onlyOwner
	includeInFee	External	✓	onlyOwner
	setBuyFee	External	✓	onlyOwner
	setSellFee	External	✓	onlyOwner
	setTransferFee	External	✓	onlyOwner
	updateRouter	External	✓	onlyOwner
	setMaxWallet	External	✓	onlyOwner
	setMaxBuyAmount	External	✓	onlyOwner
	setMaxSellAmount	External	✓	onlyOwner
	setTreasuryWallet	External	✓	notZeroAddress onlyOwner
	setTeamWallet	External	✓	notZeroAddress onlyOwner

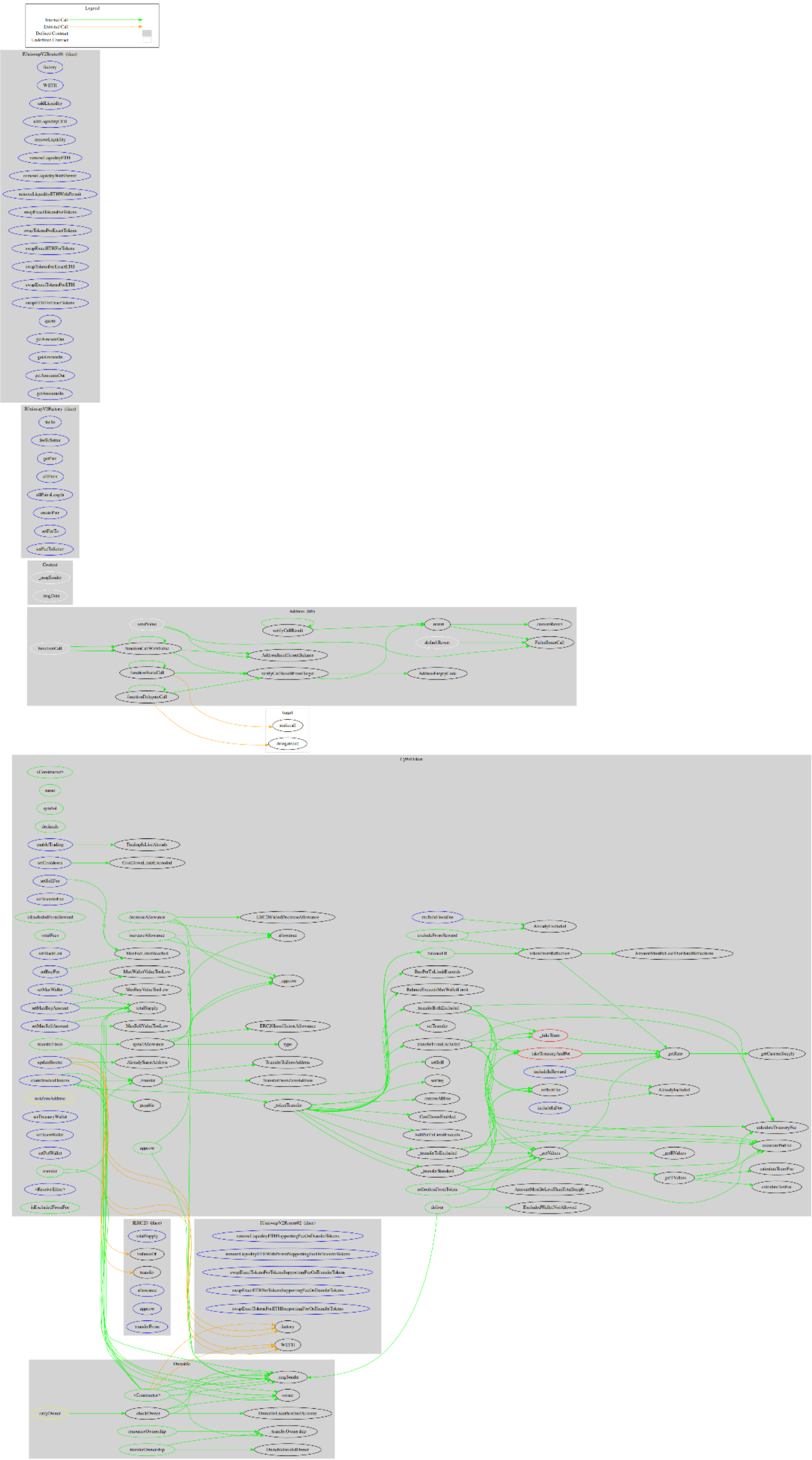
	setPotWallet	External	✓	notZeroAddress onlyOwner
	claimStuckedTokens	External	✓	onlyOwner
		External	Payable	-
	_reflectFee	Private	✓	
	_getValues	Private		
	_getTValues	Private		
	_getRValues	Private		
	_getRate	Private		
	_getCurrentSupply	Private		
	_takeTeam	Private	✓	
	_takeTreasuryAndPot	Private	✓	
	calculateTaxFee	Private		
	calculateTeamFee	Private		
	calculateTreasuryFee	Private		
	calculatePotFee	Private		
	removeAllFee	Private	✓	
	setBuy	Private	✓	
	setSell	Private	✓	
	setTransfer	Private	✓	
	isExcludedFromFee	Public		-
	_approve	Private	✓	
	_transfer	Private	✓	
	_tokenTransfer	Private	✓	

	_transferStandard	Private	✓	
	_transferToExcluded	Private	✓	
	_transferFromExcluded	Private	✓	
	_transferBothExcluded	Private	✓	

Inheritance Graph



Flow Graph


Legend
Internal Call
External Call
Defined Contract
Undefined Contract
IUniswapV2Router01 (iface)
factory
WETH
addLiquidity
addLiquidityETH
removeLiquidity
removeLiquidityETH
removeLiquidityWithPermit
removeLiquidityETHWithPermit
swapExactTokensForTokens
swapTokensForExactTokens
swapExactETHForTokens
swapTokensForExactETH
swapExactTokensForETH
swapETHForExactTokens
quote
getAmountOut
getAmountIn
getAmountsOut
getAmountsIn
IUniswapV2Factory (iface)
feeTo
feeToSetter
getPair
allPairs
allPairsLength
createPair
setFeeTo
setFeeToSetter
Context
_msgSender
_msgData
Address (lib)
sendValue
verifyCallResult
revert
customRevert
functionCall
functionCallWithValue
defaultRevert
FailedInnerCall
AddressInsufficientBalance
functionStaticCall
verifyCallResultFromTarget
AddressEmptyCode
functionDelegateCall
target
staticcall
delegatecall
CY9nToken
<Constructor>
name
symbol
decimals
enableTrading
TradingIsLiveAlready
setCooldown
CoolDownLimitExceeded
setSellFee
setTransferFee
isExcludedFromReward
decreaseAllowance
ERC20FailedDecreaseAllowance
excludeFromFee
AlreadyExcluded
totalFees
increaseAllowance
allowance
excludeFromReward
setBlackList
MaxFeeLimitReached
balanceOf
tokenFromReflection
AmountMustBeLessThanTotalReflections
setBuyFee
MaxWalletValueTooLow
BuyPerTxLimitExceeds
setMaxWallet
MaxBuyValueTooLow
_approve
BalanceExceedsMaxWalletLimit
setMaxBuyAmount
totalSupply
_transferBothExcluded
setMaxSellAmount
MaxSellValueTooLow
ERC20InsufficientAllowance
setTransfer
_takeTeam
transferFrom
spendAllowance
type
_transferFromExcluded
_takeTreasuryAndPot
_getRate
_getCurrentSupply
updateRouter
AlreadySameAddress
TransferToZeroAddress
setSell
includeInReward
claimStuckedTokens
_transfer
TransferFromZeroAddress
setBuy
reflectFee
AlreadyIncluded
notZeroAddress
payable
_tokenTransfer
removeAllFee
includeInFee
setTreasuryWallet
CoolDownEnabled
setTeamWallet
SellPerTxLimitExceeds
calculateTreasuryFee
setPotWallet
approve
_transferToExcluded
_getValues
_getRValues
calculatePotFee
transfer
_transferStandard
getTValues
calculateTeamFee
<Receive Ether>
reflectionFromToken
AmountMustBeLessThanTotalSupply
calculateTaxFee
isExcludedFromFee
deliver
ExcludedWalletNotAllowed
IERC20 (iface)
totalSupply
balanceOf
transfer
allowance
approve
transferFrom
IUniswapV2Router02 (iface)
removeLiquidityETHSupportingFeeOnTransferTokens
removeLiquidityETHWithPermitSupportingFeeOnTransferTokens
swapExactTokensForTokensSupportingFeeOnTransferTokens
swapExactETHForTokensSupportingFeeOnTransferTokens
swapExactTokensForETHSupportingFeeOnTransferTokens
factory
WETH
Ownable
_msgSender
<Constructor>
owner
onlyOwner
checkOwner
OwnableUnauthorizedAccount
renounceOwnership
_transferOwnership
transferOwnership
OwnableInvalidOwner

Summary

CY9NI 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 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. There is also a limit of max 20% fees.

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