Cyberscope

Audit Report

Eurk

June 2023

Network	ETH
Address	0x08ed36720987c86a195f404f7073cb534a20a15d
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	Unresolved
	BT	Burns Tokens	Passed
	BC	Blacklists Addresses	Unresolved

Diagnostics

Critical Medium Minor / Informative

Severity	Code	Description	Status
	RDS	Redundant Data Structure	Unresolved
	DKO	Delete Keyword Optimization	Unresolved
	TCD	Transfers Contract's Tokens	Unresolved
	RSW	Redundant Storage Writes	Unresolved
	L04	Conformance to Solidity Naming Conventions	Unresolved
	L16	Validate Variable Setters	Unresolved

Table of Contents

Review

Contract Name	FiatTokenV3
Compiler Version	v0.6.12+commit.27d51765
Optimization	10000000 runs
Explorer	https://etherscan.io/address/0x08ed36720987c86a195f404f7073cb534a20a15d
Address	0x08ed36720987c86a195f404f7073cb534a20a15d
Network	ETH

Audit Updates

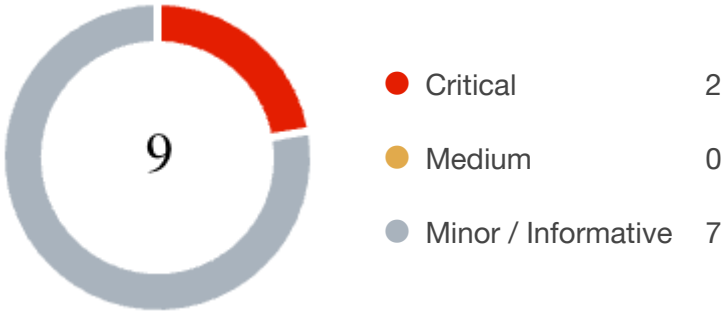
Initial Audit	06 Jun 2023

Source Files

Filename	SHA256
@openzeppelin/contracts/math/SafeMath.sol	aa64549e1edf3bd55b4a5f7404edfae8ef704f3fbd0735cdd330b380c118e067
@openzeppelin/contracts/token/ERC20/IERC20.sol	331d81d1058f809dc5b3cbae3438ed4e38cd1117778ca2ef2d207a8c46b4085c
@openzeppelin/contracts/token/ERC20/SafeERC20.sol	305ffee5fb590e99c62b777087253ddeb22eaf0d20c6e8075178210ab3eff73f
@openzeppelin/contracts/utils/Address.sol	65e6644738743b23ab1dc72cce8426f0fd6ea5dd96e87a7e16fbb0c4ccbb1633
contracts/util/ECRecover.sol	b01e9c19a9f9c6fe5019399cbd384695bd5001991f48fac6749bcfe401f0c4e7
contracts/util/EIP712.sol	c779f54e46bc7c1209619bb18999fffab74a6f6585680c02e2b592600d054752
contracts/v1.1/FiatTokenV1_1.sol	75f516c3ee9334d0ff16b36e92a36e3a381253dafbfb3dad2a18b319e6dc37e7
contracts/v1.1/Rescuable.sol	44932b3a48bc979a5e4e983abd19a07a6268fd17b1df08da2c8342f871f6a7cd
contracts/v1/AbstractFiatTokenV1.sol	180281417397c9e75dda09cd3d99a8ad35c3b42626ba69ef62704eda24e88392
contracts/v1/Blacklistable.sol	d1e8fe82c7e77ca626c27c4205dac994e6c0c97e18f86cb17559d952df8425eb
contracts/v1/FiatTokenV1.sol	8a5d5f9cd90f28f6a7fca12b5a63b5bd979fa58979b949e4b79dd2c547b1666b
contracts/v1/Ownable.sol	0e0ef6d0fc9e4e86568a281485ed73963e808f92462f72d3ce86f5e24557c0f4

contracts/v1/Pausable.sol	087f0eb221cb28b36e1522d9a54ad8567db302f1f137cdcf32e2f387de1c2fe1
contracts/v2/AbstractFiatTokenV2.sol	021276c706647ba1c7e48fc0d77354c1e7236e9e5016fa53ee9fadddc2ddf5d6
contracts/v2/EIP2612.sol	39701b30144858717f89aabb67614a916d8003cd71c32caf84fd6e2ed10b004e
contracts/v2/EIP3009.sol	8ef923619fd43c62bb686d9ebadf35e51405441a06649ec0dfda25c283098bba
contracts/v2/EIP712Domain.sol	5c1126773bb9f4224ed070d691888bb5286f37fcde89de59756174da0f3df0bd
contracts/v2/FiatTokenV2_1.sol	b8e15c76105f03b9d0025f6844aa2bc24e81e8acd54af0970ad5c0cd31f72d5e
contracts/v2/FiatTokenV2.sol	85f8d9cdaef812164d07400b70fe252f9bffa0c5d0046fe9c2a19b776df37dd9
contracts/v2/FiatTokenV3.sol	1a2719d853be0661b30dbaef525b3672929593a5266d76e8580c40974a53c37e

Findings Breakdown



Severity	Unresolved	Acknowledged	Resolved	Other
Critical	2	0	0	0
Medium	0	0	0	0
Minor / Informative	7	0	0	0

ST - Stops Transactions

Criticality	Minor / Informative
Location	contracts/v1/FiatTokenV1.sol#L244,272 contracts/v2/FiatTokenV2.sol#L99,137,190
Status	Unresolved

Description

The contract pauser has the authority to pause the transactions for all users. The owner may take advantage of it by calling the `pause` function.

```
function transferFrom(
  address from,
  address to,
  uint256 value
)
  external
  override
  whenNotPaused
  notBlacklisted(msg.sender)
  notBlacklisted(from)
  notBlacklisted(to)
  returns (bool)
{
  require(
      value <= allowed[from][msg.sender],
      "ERC20: transfer amount exceeds allowance"
  );
  _transfer(from, to, value);
  allowed[from][msg.sender] = allowed[from][msg.sender].sub(value);
  return true;
}

function transfer(address to, uint256 value)
  external
  override
  whenNotPaused
  notBlacklisted(msg.sender)
  notBlacklisted(to)
  returns (bool)
{
  _transfer(msg.sender, to, value);
  return true;
}
```

Recommendation

A suggested implementation could check that the maximum 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. 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.

MT - Mints Tokens

Criticality	Critical
Location	contracts/v1/FiatTokenV1.sol#L113
Status	Unresolved

Description

The contract's master minter has the authority to give permission to minters to mint an allowed mint amount. The master minter may take advantage of it by giving minters a high mint value. As a result, the contract tokens will be highly inflated.

```
function mint(address _to, uint256 _amount)
  external
  whenNotPaused
  onlyMinters
  notBlacklisted(msg.sender)
  notBlacklisted(_to)
  returns (bool)
{
  require(_to != address(0), "FiatToken: mint to the zero address");
  require(_amount > 0, "FiatToken: mint amount not greater than 0");

  uint256 mintingAllowedAmount = minterAllowed[msg.sender];
  require(
      _amount <= mintingAllowedAmount,
      "FiatToken: mint amount exceeds minterAllowance"
  );

  totalSupply_ = totalSupply_.add(_amount);
  balances[_to] = balances[_to].add(_amount);
  minterAllowed[msg.sender] = mintingAllowedAmount.sub(_amount);
  emit Mint(msg.sender, _to, _amount);
  emit Transfer(address(0), _to, _amount);
  return true;
}
```

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	Critical
Location	contracts/v1/Blacklistable.sol#L76
Status	Unresolved

Description

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

```
function blacklist(address _account) external onlyBlacklister {
  blacklisted[_account] = true;
  emit Blacklisted(_account);
}
```

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.

RDS - Redundant Data Structure

Criticality	Minor / Informative
Location	contracts/v1/FiatTokenV1.sol#L313
Status	Unresolved

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

The contract is using two data structures. One to determine if a user is a minter and the other to point to the allowed mint amount. Since the minter has an amount greater than zero then he is minter. Hence, the `minters` data structure is redundant.

```
function configureMinter(address minter, uint256 minterAllowedAmount)
    external
    whenNotPaused
    onlyMasterMinter
    returns (bool)
{
    minters[minter] = true;
    minterAllowed[minter] = minterAllowedAmount;
    emit MinterConfigured(minter, minterAllowedAmount);
    return true;
}
```

Recommendation

The team is advised to take these segments into consideration and rewrite them so the runtime will be more performant. That way it will improve the efficiency and performance of the source code and reduce the cost of executing it. It is recommended to remove redundant data structures.

DKO - Delete Keyword Optimization

Criticality	Minor / Informative
Location	contracts/v1/FiatTokenV1.sol#L330
Status	Unresolved

Description

The contract resets variables to the default state by setting the initial values. Setting values to state variables increases the gas cost.

```
function removeMinter(address minter)
    external
    onlyMasterMinter
    returns (bool)
{
    minters[minter] = false;
    minterAllowed[minter] = 0;
    emit MinterRemoved(minter);
    return true;
}
```

Recommendation

The team is advised to use the `delete` keyword instead of setting variables. This can be more efficient than setting the variable to a new value, using delete can reduce the gas cost associated with storing data on the blockchain.

TCD - Transfers Contract's Tokens

Criticality	Minor / Informative
Location	contracts/v1.1/Rescuable.sol#L60
Status	Unresolved

Description

The contract rescuer has the authority to claim all the balance of the contract. The owner may take advantage of it by calling the `rescueERC20` function.

```
function rescueERC20(
    IERC20 tokenContract,
    address to,
    uint256 amount
) external onlyRescuer {
    tokenContract.safeTransfer(to, amount);
}
```

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.

RSW - Redundant Storage Writes

Criticality	Minor / Informative
Location	contracts/v2/FiatTokenV3.sol#L51contracts/v1.1/Rescuable.sol#L72contracts/v1/Pausable.sol#L86contracts/v1/Blacklistable.sol#L76,85,90
Status	Unresolved

Description

There are code segments that could be optimized. A segment may be optimized so that it becomes a smaller size, consumes less memory, executes more rapidly, or performs fewer operations.

The contract modifies the state of variables without checking if their current value is the same as the one given as an argument. As a result, the contract performs redundant storage writes.

```
function updateEvidenceArchive(address _newEvidenceArchive)
    external
    onlyOwner
{
    evidenceArchive = _newEvidenceArchive;
    emit EvidenceArchiveChanged(evidenceArchive);
}

function updateRescuer(address newRescuer) external onlyOwner {
    require(
        newRescuer != address(0),
        "Rescuable: new rescuer is the zero address"
    );
    _rescuer = newRescuer;
    emit RescuerChanged(newRescuer);
}

function updatePauser(address _newPauser) external onlyOwner {
    require(
        _newPauser != address(0),
        "Pausable: new pauser is the zero address"
    );
    pauser = _newPauser;
    emit PauserChanged(pauser);
}

function blacklist(address _account) external onlyBlacklister {
    blacklisted[_account] = true;
    emit Blacklisted(_account);
}

function unBlacklist(address _account) external onlyBlacklister {
    blacklisted[_account] = false;
    emit UnBlacklisted(_account);
}

function updateBlacklister(address _newBlacklister) external onlyOwner
{
    require(
        _newBlacklister != address(0),
        "Blacklistable: new blacklister is the zero address"
    );
    blacklister = _newBlacklister;
    emit BlacklisterChanged(blacklister);
}
```

Recommendation

The team is advised to take these segments into consideration and rewrite them so the runtime will be more performant. That way it will improve the efficiency and performance of the source code and reduce the cost of executing it.

L04 - Conformance to Solidity Naming Conventions

Criticality	Minor / Informative
Location	contracts/v2/FiatTokenV3.sol#L43,51contracts/v2/FiatTokenV2.sol#L39contracts/v2/FiatTokenV2_1.sol#L35,40contracts/v1/Pausable.sol#L86contracts/v1/FiatTokenV1.sol#L113,347,363contracts/v1/Blacklistable.sol#L68,76,85,90contracts/v1.1/FiatTokenV1_1.sol#L34
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 _evidenceArchive
address _newEvidenceArchive
uint8 internal _initializedVersion

...
```

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.

L16 - Validate Variable Setters

Criticality	Minor / Informative
Location	contracts/v2/FiatTokenV3.sol#L46,55
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.

```
evidenceArchive = _evidenceArchive
evidenceArchive = _newEvidenceArchive
```

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.

Functions Analysis

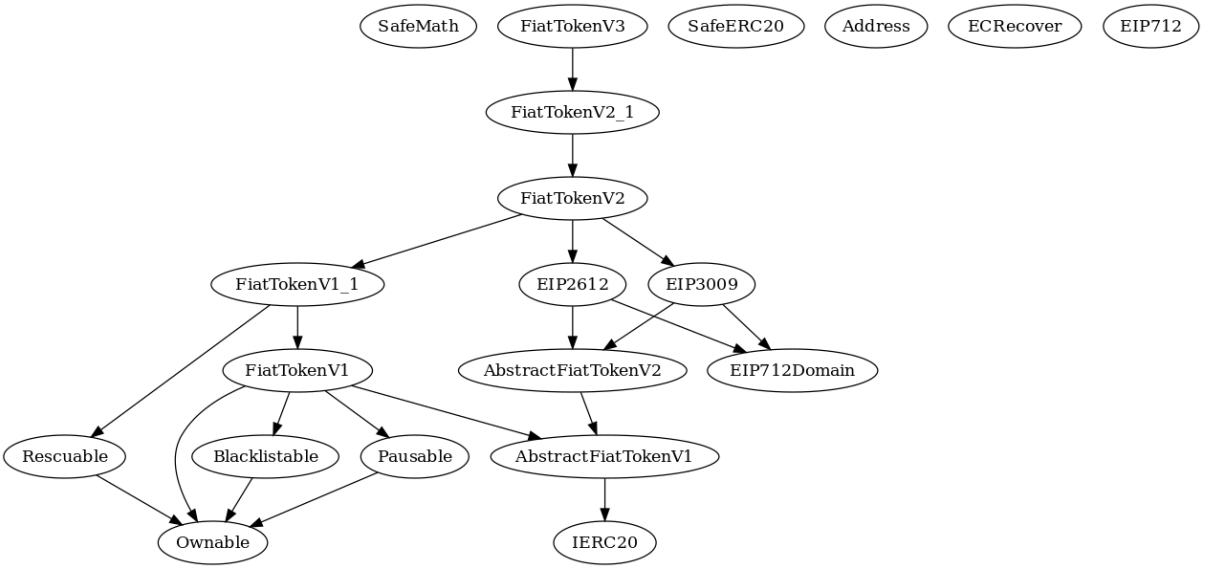
Contract	Type	Bases		
	Function Name	**Visibility**	**Mutability**	**Modifiers**
FiatTokenV1_1	Implementation	FiatTokenV1, Rescuable		
Rescuable	Implementation	Ownable		
	rescuer	External		-
	rescueERC20	External	✓	onlyRescuer
	updateRescuer	External	✓	onlyOwner
AbstractFiatTokenV1	Implementation	IERC20		
	_approve	Internal	✓	
	_transfer	Internal	✓	
Blacklistable	Implementation	Ownable		
	isBlacklisted	External		-
	blacklist	External	✓	onlyBlacklister
	unBlacklist	External	✓	onlyBlacklister
	updateBlacklister	External	✓	onlyOwner

FiatTokenV1	Implementation	AbstractFiatTokenV1, Ownable, Pausable, Blacklistable		
	initialize	Public	✓	-
	mint	External	✓	whenNotPaused onlyMinters notBlacklisted notBlacklisted
	minterAllowance	External		-
	isMinter	External		-
	allowance	External		-
	totalSupply	External		-
	balanceOf	External		-
	approve	External	✓	whenNotPaused notBlacklisted notBlacklisted
	_approve	Internal	✓	
	transferFrom	External	✓	whenNotPaused notBlacklisted notBlacklisted notBlacklisted
	transfer	External	✓	whenNotPaused notBlacklisted notBlacklisted
	_transfer	Internal	✓	
	configureMinter	External	✓	whenNotPaused onlyMasterMinter
	removeMinter	External	✓	onlyMasterMinter
	burn	External	✓	whenNotPaused onlyMinters notBlacklisted

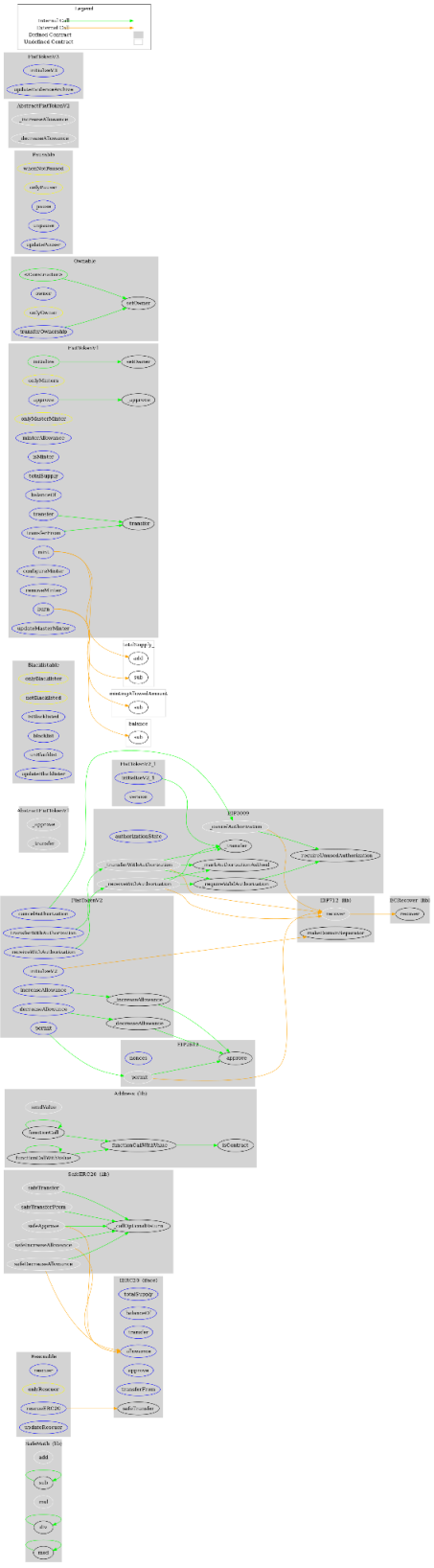
	updateMasterMinter	External	✓	onlyOwner
Ownable	Implementation			
		Public	✓	-
	owner	External		-
	setOwner	Internal	✓	
	transferOwnership	External	✓	onlyOwner
Pausable	Implementation	Ownable		
	pause	External	✓	onlyPauser
	unpause	External	✓	onlyPauser
	updatePauser	External	✓	onlyOwner
AbstractFiatTokenV2	Implementation	AbstractFiatTokenV1		
	_increaseAllowance	Internal	✓	
	_decreaseAllowance	Internal	✓	
FiatTokenV2_1	Implementation	FiatTokenV2		
	initializeV2_1	External	✓	-
	version	External		-
FiatTokenV2	Implementation	FiatTokenV1_1, EIP3009, EIP2612		
	initializeV2	External	✓	-

	increaseAllowance	External	✓	whenNotPaused notBlacklisted notBlacklisted
	decreaseAllowance	External	✓	whenNotPaused notBlacklisted notBlacklisted
	transferWithAuthorization	External	✓	whenNotPaused notBlacklisted notBlacklisted
	receiveWithAuthorization	External	✓	whenNotPaused notBlacklisted notBlacklisted
	cancelAuthorization	External	✓	whenNotPaused
	permit	External	✓	whenNotPaused notBlacklisted notBlacklisted
	_increaseAllowance	Internal	✓	
	_decreaseAllowance	Internal	✓	
FiatTokenV3	Implementation	FiatTokenV2_1		
	initializeV3	External	✓	-
	updateEvidenceArchive	External	✓	onlyOwner

Inheritance Graph



Flow Graph



Summary

Eurk 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, mint tokens and blacklist addresses. if the contract owner abuses the mint functionality, then the contract will be highly inflated. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats.

During the audit assessment, the proxy address

https://etherscan.io/address/0xFA99E789aa0164a7Eac475f3eda666BE1d0669F2#code

Points to the audited contract

https://etherscan.io/address/0x08ed36720987c86a195f404f7073cb534a20a15d#code

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