



Preliminary Audit Report

CagedBeasts Security Assessment

Real Cybersecurity
Protecting digital assets



SECURI LAB
(THAILAND) contact@securi-lab.com



PRELIMINARY AUDIT REPORT

| | |
|---|----------|
| Table of Contents | 1 |
| ▪ Report Information | 2 |
| ▪ Disclaimer | 3 |
| ▪ Executive Summary | 4 |
| NVD CVSS Scoring | |
| Audit Result | |
| ▪ Project Introduction | 5 |
| Scope Information | |
| Audit Information | |
| Audit Version History | |
| ▪ Initial Audit Scope | 6-7 |
| ▪ Security Assessment Procedure | 8 |
| ▪ Risk Rating | 9 |
| ▪ Vulnerability Severity Summary | 10 |
| ▪ Vulnerability Findings | 11-16 |
| SWC & SEC & Non-severity level | |
| ▪ SWC Findings | 17-19 |
| ▪ Visibility, Mutability, Modifier function testing | 20-23 |
| Component, Exposed Function | |
| StateVariables, Capabilities, Contract Descripton Table | |
| ▪ Inheritate Function Relation Graph | 24 |
| ▪ UML Diagram | 25 |
| ▪ About Securi | 26 |



PRELIMINARY AUDIT REPORT

Report Information

| | |
|-------------------------------|--|
| About Report | CagedBeasts Security Assessment |
| Version | v1.1 |
| Client | CagedBeasts |
| Language | Solidity |
| Confidentiality | Public |
| Contract File | Cagedbeasts.sol SHA1 - 56d305e8e5acee1383ea2139e5aafcb986f075e5 Smart Contract Are Not Deployed |
| Audit Method | Whitebox |
| Security Assessment Author | Auditor  Mark K. [Security Researcher Redteam] Kevin N. [Security Researcher Web3 Dev] Yusheng T. [Security Researcher Incident Response] Approve Document Ronny C. CTO & Head of Security Researcher Chinnakit J. CEO & Founder |

*Audit Method

Whitebox: SECURI LAB Team receives all source code from the client to provide the assessment.
Blackbox: SECURI LAB Team receives only bytecode from the client to provide the assessment.

Digital Sign (Only Full Audit Report)

PRELIMINARY AUDIT REPORT

Disclaimer

Regarding this security assessment, there are no guarantees about the security of the program instruction received from the client is hereinafter referred to as **"Source code"**.

And **SECURI Lab** hereinafter referred to as **"Service Provider"**, the **Service Provider** will not be held liable for any legal liability arising from errors in the security assessment. The responsibility will be the responsibility of the **Client**, hereinafter referred to as **"Service User"** and the **Service User** agrees not to be held liable to the **service provider** in any case. By contract **Service Provider** to conduct security assessments with integrity with professional ethics, and transparency to deliver security assessments to users The **Service Provider** has the right to postpone the delivery of the security assessment. If the security assessment is delayed whether caused by any reason and is not responsible for any delayed security assessments.

If the **service provider** finds a vulnerability The **service provider** will notify the **service user** via the Preliminary Report, which will be kept confidential for security. The **service provider** disclaims responsibility in the event of any attacks occurring whether before conducting a security assessment. Or happened later All responsibility shall be sole with the **service user**.

Security Assessment Not Financial/Investment Advice Any loss arising from any investment in any project is the responsibility of the investor.

SECURI LAB disclaims any liability incurred. Whether it's Rugpull, Abandonment, Soft Rugpull

The SECURI LAB team has conducted a comprehensive security assessment of the vulnerabilities. This assessment is tested with an expert assessment. Using the following test requirements

1. Smart Contract Testing with Expert Analysis By testing the most common and uncommon vulnerabilities.
2. Automated program testing It includes a sample vulnerability test and a sample of the potential vulnerabilities being used for the most frequent attacks.
3. Manual Testing with AST/WAS/ASE/SMT and reviewed code line by line
4. Visibility, Mutability, Modifier function testing, such as whether a function can be seen in general, or whether a function can be changed and if so, who can change it.
5. Function association test It will be displayed through the association graph.
6. This safety assessment is cross-checked prior to the delivery of the assessment results.

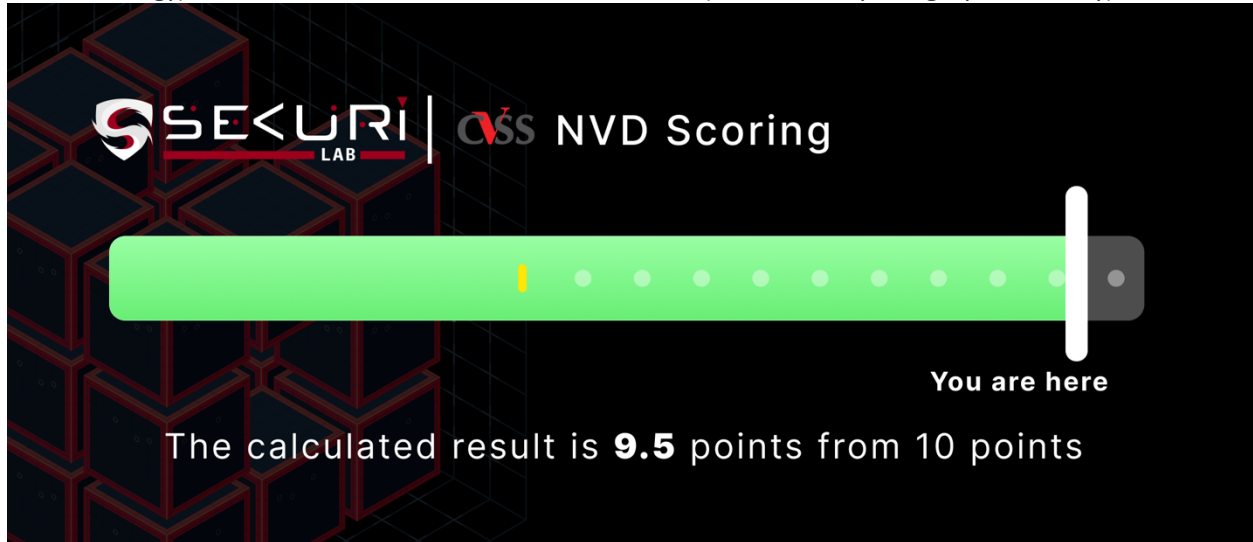
PRELIMINARY AUDIT REPORT

Executive Summary

For this security assessment, SECURI LAB received a request from CagedBeasts on Thursday, JUNE 01, 2023.

NVD CVSS Scoring

The score was calculated using the NVD (National Vulnerability Database) of NIST (National Institute of Standards and Technology) under the CVSS 3.1 standard, based on the CIA (Confidentiality, Integrity, Availability).



Audit Result

SECURI LAB evaluated the smart contract security of the project and found: [Total : 0]

| Critical | High | Medium | Low | Very Low | Informational |
|----------|------|--------|-----|----------|---------------|
| 0 | 0 | 0 | 0 | 0 | 0 |

SECURI LAB has assessed the security of this smart contract.

The results of the security assessment revealed

No Vulnerabilities.

Full Audit Report by SECURI LAB on June 06, 2023

PRELIMINARY AUDIT REPORT

Project Introduction

Scope Information:

| | |
|--------------|---|
| Project Name | CagedBeasts |
| Website | https://cagedbeasts.com/ |
| Chain | - |
| Language | Solidity |

Audit Information:

| | |
|--------------------|-------------------------|
| Request Date | Thursday, June 01, 2023 |
| Audit Date | Saturday, June 03, 2023 |
| Re-assessment Date | - |

Audit Version History:

| Version | Date | Description |
|---------|-----------------------|--------------------|
| 1.0 | Sunday, June 4, 2023 | Preliminary Report |
| 1.1 | Tuesday, June 6, 2023 | Full Audit Report |

PRELIMINARY AUDIT REPORT

Initial Audit Scope:

| | |
|---------------------|--|
| Smart Contract File | Cagedbeasts.sol |
| | SHA1- 56d305e8e5acee1383ea2139e5aafcb986f075e5 |
| Compiler Version | v0.8.18 |

Source Units Analyzed: 1

Source Units in Scope: 1 (100%)

| Type | File | Logic Contracts | Interfaces | Lines | nLines | nSLOC | Comment Lines | Complex. Score | Capabilities |
|---|---------------------------|-----------------|------------|-------|--------|-------|---------------|----------------|--------------|
|  | contracts/Cagedbeasts.sol | 1 | | 197 | 181 | 120 | 26 | 83 | Σ |
|  | Totals | 1 | | 197 | 181 | 120 | 26 | 83 | Σ |

Legend: []

- **Lines:** total lines of the source unit
- **nLines:** normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
- **nSLOC:** normalized source lines of code (only source-code lines; no comments, no blank lines)
- **Comment Lines:** lines containing single or block comments
- **Complexity Score:** a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

Dependencies / External Imports

| Dependency / Import Path | Count |
|---|-------|
| @openzeppelin/contracts/access/Ownable.sol | 1 |
| @openzeppelin/contracts/token/ERC20/extensions/IERC20Metadata.sol | 1 |
| @openzeppelin/contracts/token/ERC20/utils/SafeERC20.sol | 1 |

PRELIMINARY AUDIT REPORT

Security Assessment Procedure

Securi has the following procedures and regulations for conducting security assessments:

1.Request Audit Client submits a form request through the Securi channel. After receiving the request, Securi will discuss a security assessment. And drafting a contract and agreeing to sign a contract together with the Client

2.Auditing Securi performs security assessments of smart contracts obtained through automated analysis and expert manual audits.

3.Preliminary Report At this stage, Securi will deliver an initial security assessment. To report on vulnerabilities and errors found under Audit Scope will not publish preliminary reports for safety.

4.Reassessment After Securi has delivered the Preliminary Report to the Client, Securi will track the status of the vulnerability or error, which will be published to the Final Report at a later date with the following statuses:

a.Acknowledge The client has been informed about errors or vulnerabilities from the security assessment.

b.Resolved The client has resolved the error or vulnerability. Resolved is probably just a commit, and Securi is unable to verify that the resolved has been implemented or not.

c.Decline Client has rejected the results of the security assessment on the issue.

5.Final Report Securi providing full security assessment report and public



PRELIMINARY AUDIT REPORT

Risk Rating

Risk rating using this commonly defined: $Risk\ rating = impact * confidence$

Impact The severity and potential impact of an attacker attack

Confidence Ensuring that attackers expose and use this vulnerability

Both have a total of 3 levels: **High, Medium, Low**. By *Informational* will not be classified as a level

| Confidence Impact [Likelihood] | Low | Medium | High |
|--------------------------------------|----------|--------|----------|
| Low | Very Low | Low | Medium |
| Medium | Low | Medium | High |
| High | Medium | High | Critical |



PRELIMINARY AUDIT REPORT

Vulnerability Severity Summary

Severity is a risk assessment It is calculated from the Impact and Confidence values using the following calculation methods,

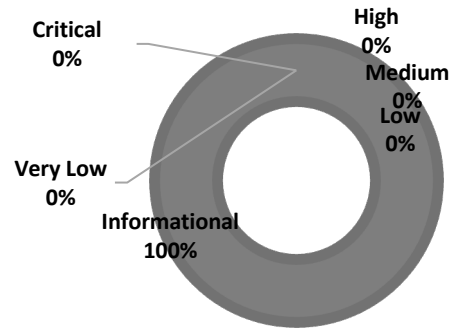
$$\text{Risk rating} = \text{impact} * \text{confidence}$$

It is categorized into

5 categories based on the lowest severity:

Very Low, Low, Medium, High, Critical.

For **Informational** & will **Non-class/Optimization/Best-practices** will not be counted as severity



| Vulnerability Severity Level | Total |
|--|-------|
| Critical | 0 |
| High | 0 |
| Medium | 0 |
| Low | 0 |
| Very Low | 0 |
| Informational | 0 |
| Non-class/Optimization/Best-practices | 1 |

Category information:

| | | | | | |
|--|---|--|---|--|--|
| Centralization Centralization Risk is The risk incurred by a sole proprietor, such as the Owner being able to change something without permission | Economics Risk Economics Risk is Risks that may affect the economic mechanism system, such as the ability to increase Mint token | Logical Issue Logical Issue is that can cause errors to core processing, such as any prior operations that cause background processes to crash. | Authorization Authorization is Possible pitfalls from weak coding allows unrelated people to take any action to modify the values. | Mathematical Mathematical Any erroneous arithmetic operations affect the operation of the system or lead to erroneous values. | Naming Conventions Naming Conventions naming variables that may affect code understanding or naming inconsistencies |
| Security Risk Security Risk of loss or damage if it's no mitigate | Coding Style Coding Style is Tips coding for efficiency performance | Best Practices Best Practices is suggestions for improvement | Optimization Optimization is performance improvement | Gas Optimization Gas Optimization is increase performance to avoid expensive gas | Dead Code Dead Code having unused code This may result in wasted resources and gas fees. |

PRELIMINARY AUDIT REPORT

Vulnerability Findings

| ID | Vulnerability Detail | Severity | Category | Status |
|--------|----------------------|----------|------------------|--------|
| GAS-01 | Use Custom Errors | - | Gas Optimization | - |



PRELIMINARY AUDIT REPORT

GAS-01: Use Custom Errors

| Vulnerability Detail | Severity | Location | Category | Status |
|----------------------|----------|------------------|------------------|--------|
| Use Custom Errors | - | Check on finding | Gas Optimization | - |

Finding:

```

113:         require(currentAllowance >= subtractedValue, "ERC20:allowance<0");
126:         require(from != address(0), "ERC20:From 0");
127:         require(to != address(0), "ERC20:To 0");
130:         require(fromBalance >= amount, "ERC20:amount>balance");
141:         require(account != address(0), "ERC20:address(0)");
152:         require(account != address(0), "ERC20:address(0)");
156:         require(accountBalance >= amount, "ERC20:amount>balance");
172:         require(owner != address(0), "ERC20:FromAddress(0)");
173:         require(spender != address(0), "ERC20:ToAddress(0)");
186:         require(currentAllowance >= amount, "ERC20: insufficient allowance");

```

Recommendation:

[Source](<https://blog.soliditylang.org/2021/04/21/custom-errors/>)

Instead of using error strings, to reduce deployment and runtime cost, you should use Custom Errors. This would save both deployment and runtime cost.

Alleviation:

-

PRELIMINARY AUDIT REPORT

SWC Findings

| ID | Title | Scanning | Result |
|---------|--------------------------------------|----------|---------|
| SWC-100 | Function Default Visibility | Complete | No risk |
| SWC-101 | Integer Overflow and Underflow | Complete | No risk |
| SWC-102 | Outdated Compiler Version | Complete | No risk |
| SWC-103 | Floating Pragma | Complete | No risk |
| SWC-104 | Unchecked Call Return Value | Complete | No risk |
| SWC-105 | Unprotected Ether Withdrawal | Complete | No risk |
| SWC-106 | Unprotected SELFDESTRUCT Instruction | Complete | No risk |
| SWC-107 | Reentrancy | Complete | No risk |
| SWC-108 | State Variable Default Visibility | Complete | No risk |
| SWC-109 | Uninitialized Storage Pointer | Complete | No risk |
| SWC-110 | Assert Violation | Complete | No risk |
| SWC-111 | Use of Deprecated Solidity Functions | Complete | No risk |
| SWC-112 | Delegatecall to Untrusted Callee | Complete | No risk |
| SWC-113 | DoS with Failed Call | Complete | No risk |
| SWC-114 | Transaction Order Dependence | Complete | No risk |
| SWC-115 | Authorization through tx.origin | Complete | No risk |

PRELIMINARY AUDIT REPORT

| | | | |
|---------|---|----------|---------|
| SWC-116 | Block values as a proxy for time | Complete | No risk |
| SWC-117 | Signature Malleability | Complete | No risk |
| SWC-118 | Incorrect Constructor Name | Complete | No risk |
| SWC-119 | Shadowing State Variables | Complete | No risk |
| SWC-120 | Weak Sources of Randomness from Chain Attributes | Complete | No risk |
| SWC-121 | Missing Protection against Signature Replay Attacks | Complete | No risk |
| SWC-122 | Lack of Proper Signature Verification | Complete | No risk |
| SWC-123 | Requirement Violation | Complete | No risk |
| SWC-124 | Write to Arbitrary Storage Location | Complete | No risk |
| SWC-125 | Incorrect Inheritance Order | Complete | No risk |
| SWC-126 | Insufficient Gas Griefing | Complete | No risk |
| SWC-127 | Arbitrary Jump with Function Type Variable | Complete | No risk |
| SWC-128 | DoS With Block Gas Limit | Complete | No risk |
| SWC-129 | Typographical Error | Complete | No risk |
| SWC-130 | Right-To-Left-Override control character (U+202E) | Complete | No risk |
| SWC-131 | Presence of unused variables | Complete | No risk |
| SWC-132 | Unexpected Ether balance | Complete | No risk |

PRELIMINARY AUDIT REPORT

| | | | |
|---------|---|----------|---------|
| SWC-133 | Hash Collisions With Multiple Variable Length Arguments | Complete | No risk |
| SWC-134 | Message call with hardcoded gas amount | Complete | No risk |
| SWC-135 | Code With No Effects | Complete | No risk |
| SWC-136 | Unencrypted Private Data On-Chain | Complete | No risk |



PRELIMINARY AUDIT REPORT



Visibility, Mutability, Modifier function testing

Components


| | | | |
|---|---|--|--|
|  Contracts |  Libraries |  Interfaces |  Abstract |
| 1 | 0 | 0 | 0 |

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.












| | | | | |
|--|---|---------|------|------|
|  Public |  Payable | | | |
| 12 | 0 | | | |
| External | Internal | Private | Pure | View |
| 1 | 17 | 0 | 0 | 6 |

StateVariables

| | |
|-------|--|
| Total |  Public |
| 6 | 0 |



Capabilities

| | | | | | |
|--|--|--|--|--|---|
| <div>Solidity Versions observed</div> | <div> Experimental Features</div> | <div> Can Receive Funds</div> | <div> Uses Assembly</div> | <div> Has Destroyable Contracts</div> | |
| <div>0.8.18</div> | | | | | |
| <div> Transfers ETH</div> | <div> Low-Level Calls</div> | <div> DelegateCall</div> | <div> Uses Hash Functions</div> | <div> ECR recover</div> | <div> New/Create/Create2</div> |
| | | | | | |
| <div> TryCatch</div> | <div>Σ Unchecked</div> | | | | |
| | <div>yes</div> | | | | |

PRELIMINARY AUDIT REPORT



Contracts Description Table

| Contract | Type | Bases | | |
|--------------------|-------------------|-------------------------|------------|-----------|
| L | Function Name | Visibility | Mutability | Modifiers |
| | | | | |
| Cagedbeasts | Implementation | IERC20Metadata, Ownable | | |
| L | | Public ! | ● | NO ! |
| L | name | Public ! | | NO ! |
| L | symbol | Public ! | | NO ! |
| L | decimals | Public ! | | NO ! |
| L | totalSupply | Public ! | | NO ! |
| L | balanceOf | Public ! | | NO ! |
| L | transfer | Public ! | ● | NO ! |
| L | allowance | Public ! | | NO ! |
| L | approve | Public ! | ● | NO ! |
| L | transferFrom | Public ! | ● | NO ! |
| L | increaseAllowance | Public ! | ● | NO ! |
| L | decreaseAllowance | Public ! | ● | NO ! |
| L | _transfer | Internal 🔒 | ● | |
| L | _mint | Internal 🔒 | ● | |
| L | _burn | Internal 🔒 | ● | |
| L | _approve | Internal 🔒 | ● | |
| L | _spendAllowance | Internal 🔒 | ● | |
| L | withdrawToken | External ! | ● | onlyOwner |



PRELIMINARY AUDIT REPORT

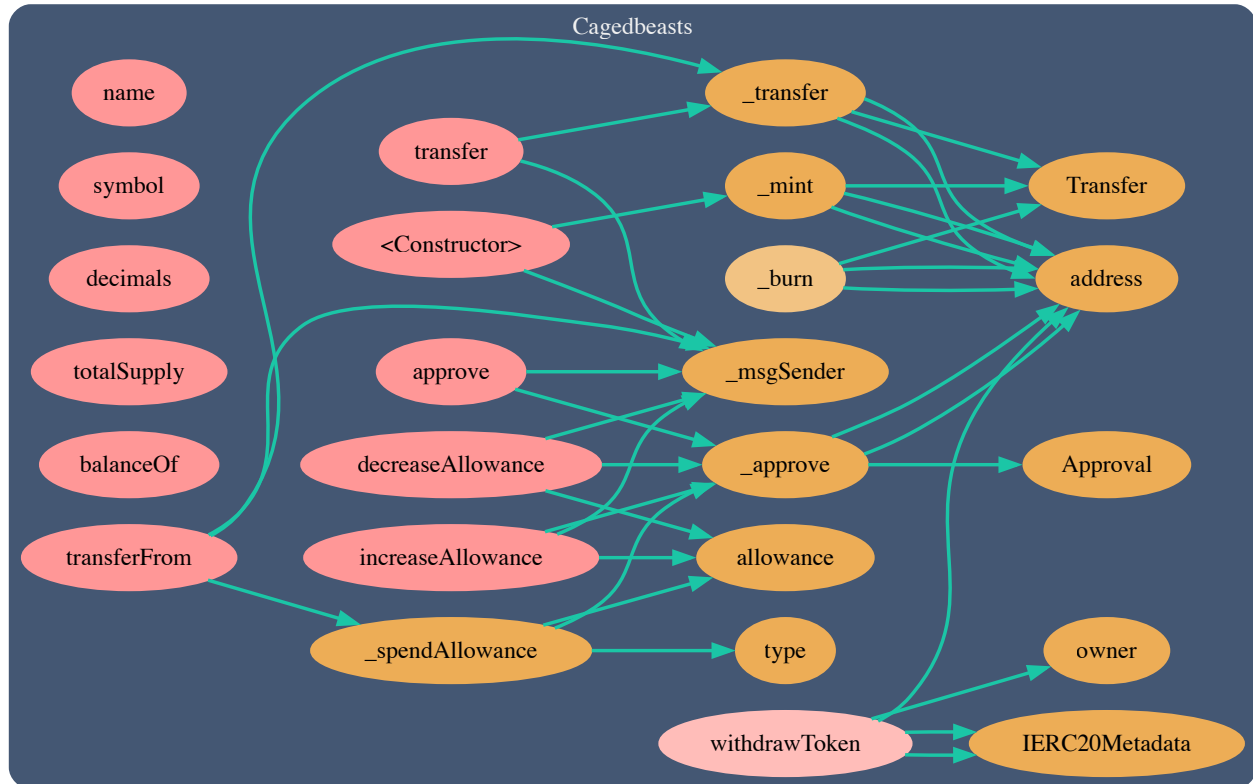
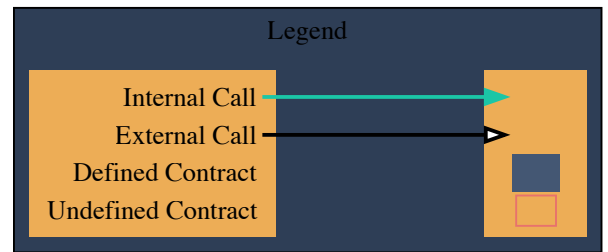
Legend

| Symbol | Meaning |
|---|---------------------------|
|  | Function can modify state |
|  | Function is payable |



PRELIMINARY AUDIT REPORT

Inheritate Function Relation Graph



PRELIMINARY AUDIT REPORT

UML Class Diagram

| Cagedbeasts contracts/Cagedbeasts.sol |
|--|
| <p>Private:</p> <p>_balances: mapping(address=>uint256)</p> <p>_allowances: mapping(address=>mapping(address=>uint256))</p> <p>_totalSupply: uint256</p> <p>_name: string</p> <p>_symbol: string</p> <p>_decimals: uint8</p> |
| <p>Internal:</p> <p>_transfer(from: address, to: address, amount: uint256)</p> <p>_mint(account: address, amount: uint256)</p> <p>_burn(account: address, amount: uint256)</p> <p>_approve(owner: address, spender: address, amount: uint256)</p> <p>_spendAllowance(owner: address, spender: address, amount: uint256)</p> <p>External:</p> <p>withdrawToken(_token: address) <<onlyOwner>></p> <p>Public:</p> <p>constructor(__name: string, __symbol: string, __totalSupply: uint256, __decimals: uint8)</p> <p>name(): string</p> <p>symbol(): string</p> <p>decimals(): uint8</p> <p>totalSupply(): uint256</p> <p>balanceOf(account: address): uint256</p> <p>transfer(to: address, amount: uint256): bool</p> <p>allowance(owner: address, spender: address): uint256</p> <p>approve(spender: address, amount: uint256): bool</p> <p>transferFrom(from: address, to: address, amount: uint256): bool</p> <p>increaseAllowance(spender: address, addedValue: uint256): bool</p> <p>decreaseAllowance(spender: address, subtractedValue: uint256): bool</p> |

PRELIMINARY AUDIT REPORT

About SECURI LAB

SECURI LAB is a group of cyber security experts providing cyber security consulting, smart contract security audits, and KYC services.



Follow Us On:

| | |
|----------|---|
| Website | https://securi-lab.com/ |
| Twitter | https://twitter.com/SECURI_LAB |
| Telegram | https://t.me/securi_lab |
| Medium | https://medium.com/@securi |