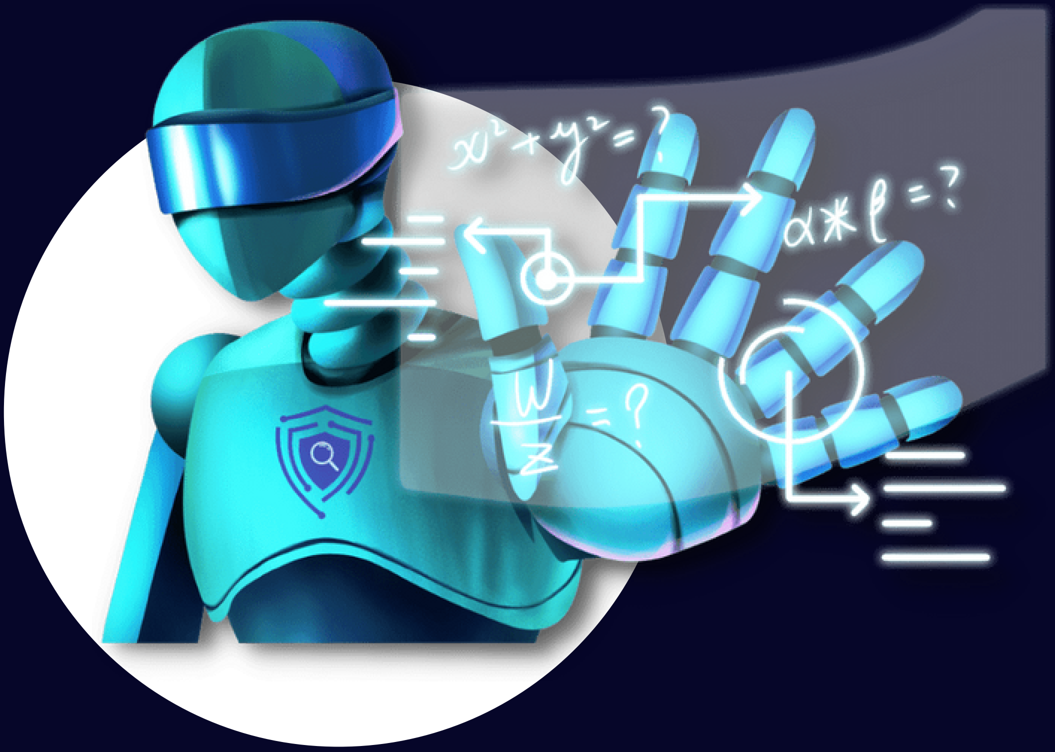
SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY

SAFUAUDIT

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY



PROJECT:	HUGO
DATE:	July 29, 2022

www.safuaudit.com

INTRODUCTION

Client	雨果.com (HUGO)
Language	Solidity
Contract address	0x88888888795cc8810169118099bcB8E4D97C6834
Owner	-
Deployer	0x684DB466C40cC6EE9B96a1462027237A4fe46F37
SHA1-Hash	4d2a423c699491fbc4cc3a7617c8617f7476942d
Decimals	0
Supply	100,000,000
Platform	Binance Smart Chain
Compiler	v0.4.21+commit.dfe3193c
Optimization	Yes with 200 runs
Website	https://www.hugo.onl/
Telegram	https://t.me/hugoshop
Twitter	https://twitter.com/Hugo_onl

OVERVIEW

Fees

- Buy Fees: 0%
- Sell Fees: 0%

Fees privileges

- Can't set fees

Ownership

- Not Owned

Minting

- No mint function

Max Tx Amount

- Can't set max Tx amount

Pause function

- Can't pause trading

Blacklist

- Can't blacklist

TABLE OF CONTENTS

APPROACH

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

Tools

- Remix IDE
- Mythril
- Open Zeppelin Code Analyzer
- Solidity Code Complier
- Hardhat

RISK CLASSIFICATION

CRITICAL

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

MEDIUM

Issues on this level could potentially bring problems and should eventually be fixed.

MINOR

Issues on this level are minor details and warning that can remain unfixed but would be better fixed at some point in the future

INFORMATIONAL

Information level is to offer suggestions for improvement of efficacity or security for features with a risk free factor.

CONTRACT INSPECTION 🔎

Imported contracts or frameworks used:

```
| **EIP20Interface** | Implementation |  |||
| └ | balanceOf | Public ❗️ |   |NO❗️ |
| └ | transfer | Public ❗️ | 🛑  |NO❗️ |
| └ | transferFrom | Public ❗️ | 🛑  |NO❗️ |
| └ | approve | Public ❗️ | 🛑  |NO❗️ |
| └ | allowance | Public ❗️ |   |NO❗️ |
```

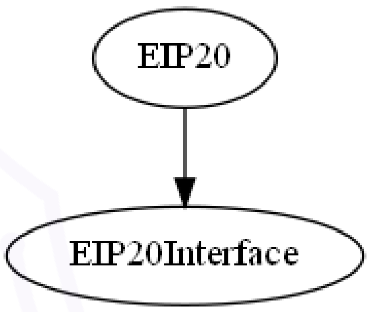
Tested Contract File:

```
|  File Name  |  SHA-1 Hash  |
|-------------|--------------|
| Hugo.sol | 4d2a423c699491fbc4cc3a7617c8617f7476942d |
```

```
| **EIP20** | Implementation | EIP20Interface |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
| └ | transfer | Public ❗️ | 🛑  |NO❗️ |
| └ | transferFrom | Public ❗️ | 🛑  |NO❗️ |
| └ | balanceOf | Public ❗️ |   |NO❗️ |
| └ | approve | Public ❗️ | 🛑  |NO❗️ |
| └ | allowance | Public ❗️ |   |NO❗️ |
```

```
| Symbol |          Meaning              |
|--------|-------------------------------|
|   🛑    | Function can modify state |
|   💵   | Function is payable |
|   🔐    | Private function |
|   🔒    | Internal function |
|   NO❗️ | Function has no modifier |
```

INHERITANCE TREE



Inheritance is a feature of the object-oriented programming language. It is a way of extending the functionality of a program, used to separate the code, reduces the dependency, and increases the re-usability of the existing code. Solidity supports inheritance between smart contracts, where multiple contracts can be inherited into a single contract.

MANUAL FUNCTIONS ANALYSIS

The contract is verified to check if functions do and work as they should and malicious code is not inserted.

	Tested	**Result**
Transfer	Yes	Passed
Total Supply	Yes	N/A
Buy Back	Yes	N/A
Burn	Yes	N/A
Mint	Yes	N/A
Rebase	Yes	N/A
Pause	Yes	N/A
Blacklist	Yes	N/A
Lock	Yes	N/A
Max Transaction	Yes	N/A
Transfer Ownership	Yes	N/A
Renounce Ownership	Yes	N/A

VULNERABILITIES TEST

ID	Description	
V-01	Function Default Visibility	Passed
V-02	Integer Overflow and Underflow	Passed
V-03	Outdated Compiler Version	Passed
V-04	Floating Pragma	Minor
V-05	Unchecked Call Return Value	Passed
V-06	Unprotected Ether Withdrawal	Passed
V-07	Unprotected SELF-DESTRUCT Instruction	Passed
V-08	Re-entrancy	Passed
V-09	State Variable Default Visibility	Passed
V-10	Uninitialized Storage Pointer	Passed
V-11	Assert Violation	Passed
V-12	Use of Deprecated Solidity Functions	Passed
V-13	Delegate Call to Untrusted Callee	Passed
V-14	DoS with Failed Call	Passed
V-15	Transaction Order Dependence	Passed
V-16	Authorization through tx.origin	Passed
V-17	Block values as a proxy for time	Passed

V-18	Signature Malleability	**Passed**
V-19	Incorrect Constructor Name	**Passed**
V-20	Shadowing State Variables	**Passed**
V-21	Weak Sources of Randomness from Chain Attributes	**Passed**
V-22	Missing Protection against Signature Replay Attacks	**Passed**
V-23	Lack of Proper Signature Verification	**Passed**
V-24	Requirement Violation	**Passed**
V-25	Write to Arbitrary Storage Location	**Passed**
V-26	Incorrect Inheritance Order	**Passed**
V-27	Insufficient Gas Griefing	**Passed**
V-28	Arbitrary Jump with Function Type Variable	**Passed**
V-29	DoS With Block Gas Limit	**Passed**
V-30	Typographical Error	**Passed**
V-31	Right-To-Left-Override control character (U+202E)	**Passed**
V-32	Presence of unused variables	**Passed**
V-33	Unexpected Ether balance	**Passed**
V-34	Hash Collisions With Multiple Variable Length Arguments	**Passed**
V-35	Message call with the hardcoded gas amount	**Passed**
V-36	Code With No Effects (Irrelevant/Dead Code)	**Passed**
V-37	Unencrypted Private Data On-Chain	**Passed**

FINDINGS

ID	Category	Issue	Severity
V-01	Vulnerabilities	Floating Pragma	Minor

V-01: Floating Pragma

Line #9

```
pragma solidity ^0.4.21;
```

Description

Contract uses old version of Solidity.

- Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

- Solc frequently releases new compiler versions. Using an old version prevents access to new Solidity security checks.

Recommendation

- Lock the pragma version and also consider known bugs (https://github.com/ethereum/solidity/releases) for the compiler version that is chosen.
- Deploy with any of the following Solidity versions:
  - 0.5.16 - 0.5.17
  - 0.6.11 - 0.6.12
  - 0.7.5 - 0.7.6
  - 0.8.4 - 0.8.7 Use a simple pragma version that allows any of these versions. Consider using the latest version of Solidity for testing.

WEBSITE 🌐

Website	https://www.hugo.onl/
Domain Registry	http://www.enom.com
Domain Expiry Date	2022-11-08
Response Code	200
SSL Checker and HTTPS Test	Passed
Deprecated HTML tags	Passed
Robots.txt	Informative
Sitemap Test	Informative
SEO Friendly URL	Passed
Responsive Test	Passed
JS Error Test	Passed
Console Errors Test	Passed
Site Loading Speed Test	1.21 seconds - Passed
HTTP2 Test	Passed
Safe Browsing Test	Passed

DISCLAIMER

SafuAudit.com is not a financial institution and the information provided on this website does not constitute investment advice, financial advice, trading advice, or any other sort of advice. You should not treat any of the website's content as such. Investing in crypto assets carries a high level of risk and does not hold guarantees for not sustaining financial loss due to their volatility.

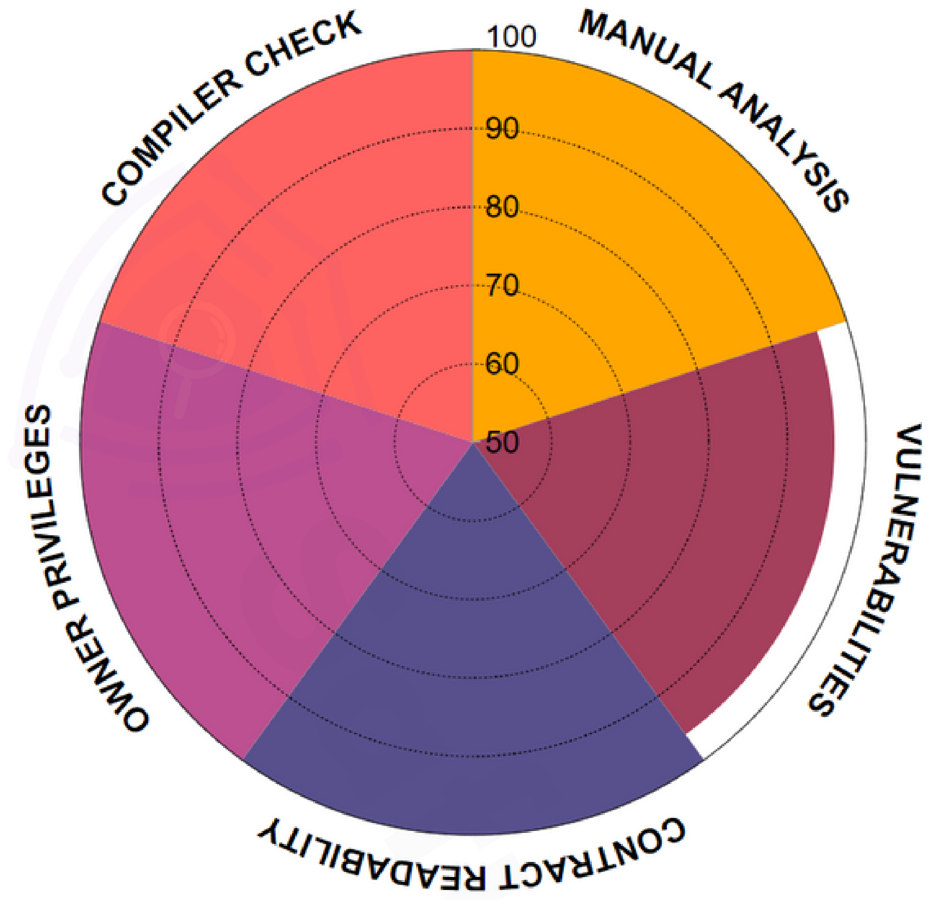
Accuracy of Information

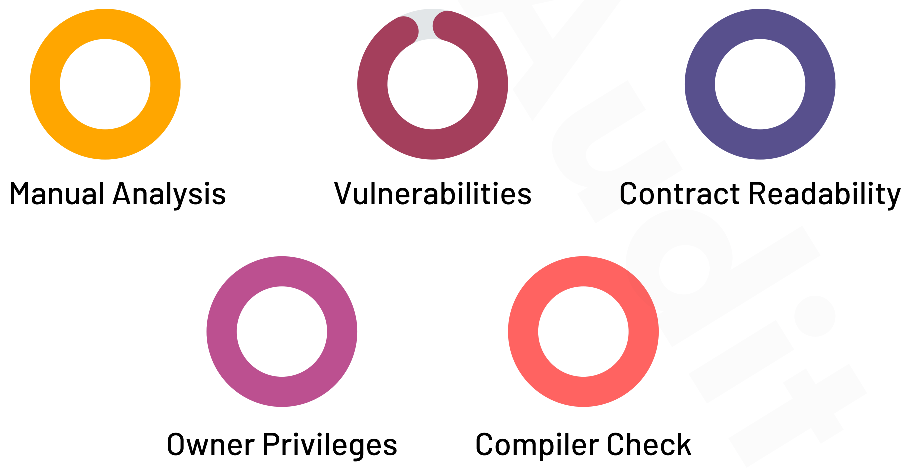
SafuAudit will strive to ensure the accuracy of the information listed on this website although it will not hold any responsibility for any missing or wrong information. SafuAudit provides all information as is. You understand that you are using any and all information available here at your own risk. Any use or reliance on our content and services is solely at your own risk and discretion.

The purpose of the audit is to analyze the on-chain smart contract source code and to provide a basic overview of the project.

While we have used all the information available to us for this straightforward investigation, you should not rely on this report only — we recommend proceeding with several independent audits Be aware that smart contracts deployed on a blockchain aren't secured enough against external vulnerability or a hack. Be aware that active smart contract owner privileges constitute an elevated impact on the smart contract safety and security. Therefore, SafuAudit does not guarantee the explicit security of the audited smart contract. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

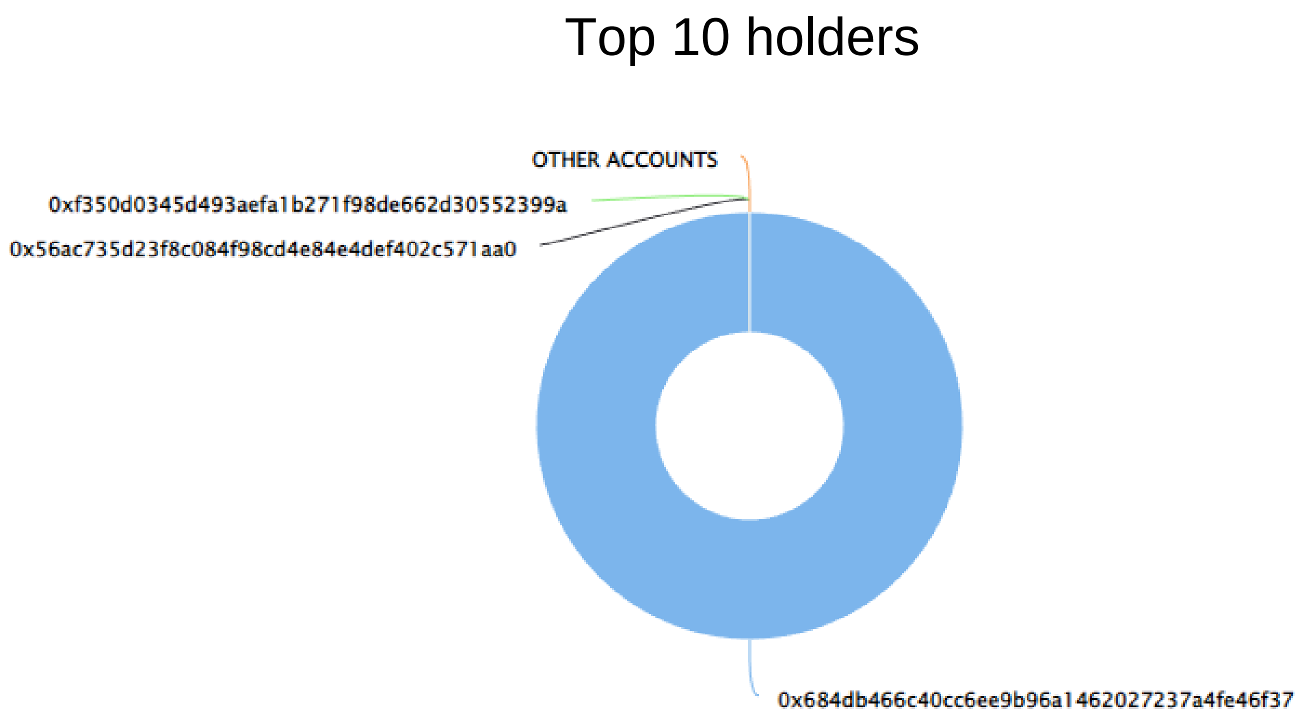
RATING





Final Score: **99.2**

SUMMARY



Rank	Address	Quantity (Token)	Percentage
1	0x684db466c40cc6ee9b96a1462027237a4fe46f37	99,999,998	100.0000%
2	0x56ac735d23f8c084f98cd4e84e4def402c571aa0	1	0.0000%
3	0xf350d0345d493aefa1b271f98de662d30552399a	1	0.0000%

CONCLUSION

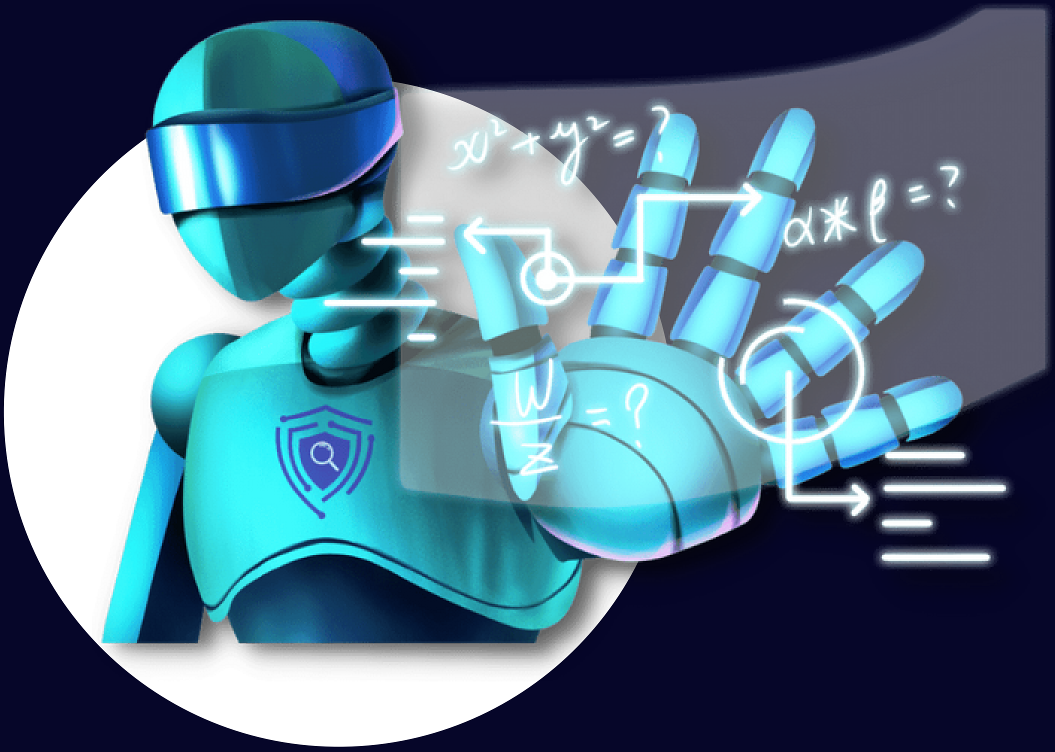
Project 雨果.com (HUGO) has no critical or medium severity issues or risk characteristics.

SafuAudit has tested the security based on manual and automated tests. Please note that we don't offer any warranties for the business model.

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY

SAFUAUDIT

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY



"Only in growth, reform, and change, paradoxically enough, is true security to be found."

www.safuaudit.com