




SMART CONTRACT SECURITY AUDIT

Final report

Plan: Simple

Caw

April 2022

 rugdog.net

 the@rugdog.net



CONTENTS

1. Introduction	3
2. Contracts checked	3
3. Audit Process	4
4. Attacks checked	4
5. Classification of issues	6
6. Issues	6
High severity issues	6
Medium severity issues	6
Low severity issues	6
7. Conclusion	7
8. Disclaimer	8
9. Static analysis	9

INTRODUCTION

CAW is a newly released token built on the Ethereum blockchain that focuses on providing services on DeFi based payment activities. The CAW project functions based on building payment stacks on the Ethereum blockchain.

The token developers have created a basket that contains stable coins pegged with fiats using blockchain technology and algorithm to develop CAW crypto tokens. Stable coins help to withstand the high market volatility.

In addition to that, the project helps to enhance the development of open financial infrastructure and programmable payment methods.

Name	Caw
Audit date	2022-04-29 - 2022-04-29
Language	Solidity
Network	Ethereum



CONTRACTS CHECKED

Name	Address
StandardERC20	0x95EAdD76707412C27b60C3Cf4BA45f56dC426488



AUDIT PROCESS

The code was audited by the team according to the following order:

Automated analysis

-  Scanning the project's smart contracts with several publicly available automated Solidity analysis tools
-  Manual confirmation of all the issues found by the tools

Manual audit

-  Thorough manual analysis of smart contracts for security vulnerabilities
-  Smart contracts' logic check

ATTACKS CHECKED

Title	Check result
Unencrypted Private Data On-Chain	✓ passed
Code With No Effects	✓ passed
Message call with hardcoded gas amount	✓ passed
Typographical Error	✓ passed
DoS With Block Gas Limit	✓ passed
Presence of unused variables	✓ passed
Incorrect Inheritance Order	✓ passed

Requirement Violation	✓ passed
Weak Sources of Randomness from Chain Attributes	✓ passed
Shadowing State Variables	✓ passed
Incorrect Constructor Name	✓ passed
Block values as a proxy for time	✓ passed
Authorization through tx.origin	✓ passed
DoS with Failed Call	✓ passed
Delegatecall to Untrusted Callee	✓ passed
Use of Deprecated Solidity Functions	✓ passed
Assert Violation	✓ passed
State Variable Default Visibility	✓ passed
Reentrancy	✓ passed
Unprotected SELFDESTRUCT Instruction	✓ passed
Unprotected Ether Withdrawal	✓ passed
Unchecked Call Return Value	✓ passed
Floating Pragma	✓ passed
Outdated Compiler Version	✓ passed

Integer Overflow and Underflow

✓ passed

Function Default Visibility

✓ passed

CLASSIFICATION OF ISSUES

High severity

Issues leading to assets theft, locking or any other loss of assets or leading to contract malfunctioning.

Medium severity

Issues that can trigger a contract failure of malfunctioning.

Low severity

Issues that do now affect contract functionality. For example, unoptimised gas usage, outdated or unused code, code style violations, etc.

ISSUES

High severity issues

No issues were found

Medium severity issues

No issues were found

Low severity issues

No issues were found

CONCLUSION

Caw StandardERC20 contract was audited. No issues were found.

◆ **DISCLAIMER**

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to the Company in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without RugDog 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 considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts RugDog to perform a security assessment. This report 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 in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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.

STATIC ANALYSIS

INFO:Detectors:

Reentrancy in StandardERC20.constructor(string,string,uint8,uint256,address) (contracts/Token.sol#569-579):

External calls:

- ServicePayer(feeReceiver_,StandardERC20) (contracts/Token.sol#575)
- IPayable(receiver).pay{value: msg.value}(serviceName) (contracts/Token.sol#552)

State variables written after the call(s):

- _mint(_msgSender(),initialBalance_) (contracts/Token.sol#578)
- _balances[account] += amount (contracts/Token.sol#409)
- _mint(_msgSender(),initialBalance_) (contracts/Token.sol#578)
- _totalSupply += amount (contracts/Token.sol#408)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2>

INFO:Detectors:

Reentrancy in StandardERC20.constructor(string,string,uint8,uint256,address) (contracts/Token.sol#569-579):

External calls:

- ServicePayer(feeReceiver_,StandardERC20) (contracts/Token.sol#575)
- IPayable(receiver).pay{value: msg.value}(serviceName) (contracts/Token.sol#552)

Event emitted after the call(s):

- Transfer(address(0),account,amount) (contracts/Token.sol#410)
- _mint(_msgSender(),initialBalance_) (contracts/Token.sol#578)

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3>

INFO:Detectors:

Context._msgData() (contracts/Token.sol#146-148) is never used and should be removed

ERC20._burn(address,uint256) (contracts/Token.sol#426-441) is never used and should be removed

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code>

INFO:Detectors:

Pragma version^0.8.0 (contracts/Token.sol#14) necessitates a version too recent to be trusted.
Consider deploying with 0.6.12/0.7.6

Pragma version^0.8.0 (contracts/Token.sol#99) necessitates a version too recent to be trusted.
Consider deploying with 0.6.12/0.7.6

Pragma version^0.8.0 (contracts/Token.sol#129) necessitates a version too recent to be trusted.
Consider deploying with 0.6.12/0.7.6

Pragma version^0.8.0 (contracts/Token.sol#156) necessitates a version too recent to be trusted.
Consider deploying with 0.6.12/0.7.6

Pragma version^0.8.0 (contracts/Token.sol#513) necessitates a version too recent to be trusted.
Consider deploying with 0.6.12/0.7.6

Pragma version^0.8.0 (contracts/Token.sol#540) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6

Pragma version^0.8.0 (contracts/Token.sol#560) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6

solc-0.8.4 is not recommended for deployment

Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity>

INFO:Detectors:

name() should be declared external:

- ERC20.name() (contracts/Token.sol#213-215)

symbol() should be declared external:

- ERC20.symbol() (contracts/Token.sol#221-223)

totalSupply() should be declared external:

- ERC20.totalSupply() (contracts/Token.sol#245-247)

balanceOf(address) should be declared external:

- ERC20.balanceOf(address) (contracts/Token.sol#252-254)

transfer(address,uint256) should be declared external:

- ERC20.transfer(address,uint256) (contracts/Token.sol#264-267)

allowance(address,address) should be declared external:

- ERC20.allowance(address,address) (contracts/Token.sol#272-274)

approve(address,uint256) should be declared external:

- ERC20.approve(address,uint256) (contracts/Token.sol#283-286)

transferFrom(address,address,uint256) should be declared external:

- ERC20.transferFrom(address,address,uint256) (contracts/Token.sol#301-315)

increaseAllowance(address,uint256) should be declared external:

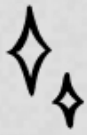
- ERC20.increaseAllowance(address,uint256) (contracts/Token.sol#329-332)

decreaseAllowance(address,uint256) should be declared external:

- ERC20.decreaseAllowance(address,uint256) (contracts/Token.sol#348-356)


Reference: <https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external>

INFO:Slither:. analyzed (8 contracts with 75 detectors), 22 result(s) found



WOOF!

 rugdog.net

 the@rugdog.net

