RUGDOG

SMART CONTRACT SECURITY AUDIT

Final report   Plan: Simple

Mara

June 2022

🌐 rugdog.net

✉ the@rugdog.net

✦ CONTENTS

✦ INTRODUCTION

A fungible token of ERC20 standard with antibot functionality.

Name	Mara
Audit date	2022-06-24 – 2022-06-24
Language	Solidity
Network	Binance Smart Chain

✦ CONTRACTS CHECKED

Name	Address
AntiBotStandardToken	0x868149c7eacd7eb0ab71a43f7b9ff25ec1dc8023

✦ 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Outdated Compiler Version	✓ passed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 For example, unoptimised gas usage, outdated or unused code, code styleviolations, etc.

✦ ISSUES

High severity issues

No issues were found

Medium severity issues

No issues were found

Low severity issues

1. Antibot may block transfers (AntiBotStandardToken)

The contract calls an external contract for antibot protection. The antibot contract is deployed via proxy and it's coe can be changed. The antibot may potentially block transfers.

```
function _transfer(
    address sender,
    address recipient,
    uint256 amount
) internal virtual {
    ...

    if (enableAntiBot) {
        pinkAntiBot.onPreTransferCheck(sender, recipient, amount);
    }
    ...
}
```

✦ CONCLUSION

Mara AntiBotStandardToken contract was audited. 1 low severity issue was 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:
AntiBotStandardToken.allowance(address,address).owner (Mara .sol#590) shadows:
   - Ownable.owner() (Mara .sol#150-152) (function)
AntiBotStandardToken._approve(address,address,uint256).owner (Mara .sol#795)
shadows:
   - Ownable.owner() (Mara .sol#150-152) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-
variable-shadowing
INFO:Detectors:
AntiBotStandardToken.constructor(string,string,uint8,uint256,address,address,uint256)
.serviceFeeReceiver_ (Mara .sol#491) lacks a zero-check on :
    - address(serviceFeeReceiver_).transfer(serviceFee_) (Mara .sol#510)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-
zero-address-validation
INFO:Detectors:
Reentrancy in AntiBotStandardToken._transfer(address,address,uint256)
(Mara .sol#716-736):
  External calls:
  - pinkAntiBot.onPreTransferCheck(sender,recipient,amount) (Mara .sol#725)
  State variables written after the call(s):
  - _balances[sender] = _balances[sender].sub(amount,ERC20: transfer amount exceeds
balance) (Mara .sol#730-733)
  - _balances[recipient] = _balances[recipient].add(amount) (Mara .sol#734)
Reentrancy in AntiBotStandardToken.constructor(string,string,uint8,uint256,address,ad
dress,uint256) (Mara .sol#485-511):
  External calls:
  - pinkAntiBot.setTokenOwner(owner()) (Mara .sol#500)
  State variables written after the call(s):
  - enableAntiBot = true (Mara .sol#501)
Reentrancy in AntiBotStandardToken.transferFrom(address,address,uint256)
(Mara .sol#630-645):
  External calls:
  - _transfer(sender,recipient,amount) (Mara .sol#635)
```

```
      - pinkAntiBot.onPreTransferCheck(sender,recipient,amount) (Mara .sol#725)
    State variables written after the call(s):
    - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20:
transfer amount exceeds allowance)) (Mara .sol#636-643)
      - _allowances[owner][spender] = amount (Mara .sol#802)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-2
INFO:Detectors:
Reentrancy in AntiBotStandardToken._transfer(address,address,uint256)
(Mara .sol#716-736):
    External calls:
    - pinkAntiBot.onPreTransferCheck(sender,recipient,amount) (Mara .sol#725)
    Event emitted after the call(s):
    - Transfer(sender,recipient,amount) (Mara .sol#735)
Reentrancy in AntiBotStandardToken.constructor(string,string,uint8,uint256,address,ad
dress,uint256) (Mara .sol#485-511):
    External calls:
    - pinkAntiBot.setTokenOwner(owner()) (Mara .sol#500)
    Event emitted after the call(s):
    - TokenCreated(owner(),address(this),TokenType.antiBotStandard,VERSION)
(Mara .sol#503-508)
Reentrancy in AntiBotStandardToken.transferFrom(address,address,uint256)
(Mara .sol#630-645):
    External calls:
    - _transfer(sender,recipient,amount) (Mara .sol#635)
      - pinkAntiBot.onPreTransferCheck(sender,recipient,amount) (Mara .sol#725)
    Event emitted after the call(s):
    - Approval(owner,spender,amount) (Mara .sol#803)
      - _approve(sender,_msgSender(),_allowances[sender]
[_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance))
(Mara .sol#636-643)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-
vulnerabilities-3
INFO:Detectors:
AntiBotStandardToken._burn(address,uint256) (Mara .sol#768-779) is never used and
should be removed
AntiBotStandardToken._setupDecimals(uint8) (Mara .sol#813-815) is never used and
```

```
should be removed
Context._msgData() (Mara .sol#110-112) is never used and should be removed
SafeMath.div(uint256,uint256) (Mara .sol#324-326) is never used and should be
removed
SafeMath.div(uint256,uint256,string) (Mara .sol#380-389) is never used and should be
removed
SafeMath.mod(uint256,uint256) (Mara .sol#340-342) is never used and should be
removed
SafeMath.mod(uint256,uint256,string) (Mara .sol#406-415) is never used and should be
removed
SafeMath.mul(uint256,uint256) (Mara .sol#310-312) is never used and should be
removed
SafeMath.sub(uint256,uint256) (Mara .sol#296-298) is never used and should be
removed
SafeMath.tryAdd(uint256,uint256) (Mara .sol#211-217) is never used and should be
removed
SafeMath.tryDiv(uint256,uint256) (Mara .sol#253-258) is never used and should be
removed
SafeMath.tryMod(uint256,uint256) (Mara .sol#265-270) is never used and should be
removed
SafeMath.tryMul(uint256,uint256) (Mara .sol#236-246) is never used and should be
removed
SafeMath.trySub(uint256,uint256) (Mara .sol#224-229) is never used and should be
removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version=0.8.4 (Mara .sol#461) 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:
Parameter AntiBotStandardToken.setEnableAntiBot(bool)._enable (Mara .sol#513) is not
in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-
to-solidity-naming-conventions
INFO:Detectors:
```

Variable AntiBotStandardToken._totalSupply (Mara .sol#480) is too similar to AntiBotS
tandardToken.constructor(string,string,uint8,uint256,address,address,uint256).totalSu
pply_ (Mara .sol#489)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-
names-are-too-similar
INFO:Detectors:
renounceOwnership() should be declared external:
  - Ownable.renounceOwnership() (Mara .sol#169-171)
transferOwnership(address) should be declared external:
  - Ownable.transferOwnership(address) (Mara .sol#177-180)
name() should be declared external:
  - AntiBotStandardToken.name() (Mara .sol#520-522)
symbol() should be declared external:
  - AntiBotStandardToken.symbol() (Mara .sol#528-530)
decimals() should be declared external:
  - AntiBotStandardToken.decimals() (Mara .sol#545-547)
totalSupply() should be declared external:
  - AntiBotStandardToken.totalSupply() (Mara .sol#552-554)
balanceOf(address) should be declared external:
  - AntiBotStandardToken.balanceOf(address) (Mara .sol#559-567)
transfer(address,uint256) should be declared external:
  - AntiBotStandardToken.transfer(address,uint256) (Mara .sol#577-585)
allowance(address,address) should be declared external:
  - AntiBotStandardToken.allowance(address,address) (Mara .sol#590-598)
approve(address,uint256) should be declared external:
  - AntiBotStandardToken.approve(address,uint256) (Mara .sol#607-615)
transferFrom(address,address,uint256) should be declared external:
  - AntiBotStandardToken.transferFrom(address,address,uint256) (Mara .sol#630-645)
increaseAllowance(address,uint256) should be declared external:
  - AntiBotStandardToken.increaseAllowance(address,uint256) (Mara .sol#659-670)
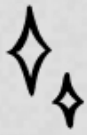decreaseAllowance(address,uint256) should be declared external:
  - AntiBotStandardToken.decreaseAllowance(address,uint256) (Mara .sol#686-700)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-
function-that-could-be-declared-external
INFO:Slither:Mara .sol analyzed (7 contracts with 75 detectors), 40 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github
integration

Mara

RUGDOG

WOOF!

🌐 rugdog.net

✉ the@rugdog.net