Security Assessment Report

MINU Token Contract

January 31, 2024

Summary

The Sec3 team (formerly Soteria) was engaged to conduct a thorough security analysis of the MINU token contract smart contracts.

The artifact of the audit was the source code of the following programs, excluding tests, in https://github.com/MantleInu/minu-token-contract.

The initial audit focused on the following versions and revealed 3 issues or questions.

program	type	commit
MINU token contract	Solidity	`32f976ffda1799a577c4f93b2f156e0e6a40fe49`

This report provides a detailed description of the findings and their respective resolutions.

Table of Contents

Result Overview

Issue	Impact	Status
MINU TOKEN CONTRACT		
[I-1] Missing zero address checks	Info	Resolved
[I-2] Certain parameters should be greater than zero	Info	Resolved
[I-3] Validate setRule parameters	Info	Resolved

Findings in Detail

MINU TOKEN CONTRACT

[I-1] Missing zero address checks

```
/* src/MantleInuToken.sol */
590 | contract MantleInuToken is Ownable, ERC20 {
591 |     bool public limited;
592 |     uint256 public maxHoldingAmount;
593 |     uint256 public minHoldingAmount;
594 |     address public uniswapV2Pair;
595 |     mapping(address => bool) public blacklists;
597 |     constructor(uint256 _totalSupply) ERC20("Mantle Inu Token", "MINU") {
598 |         _mint(msg.sender, _totalSupply);
599 |     }
601 |     function blacklist(address _address, bool _isBlacklisting) external onlyOwner {
602 |         blacklists[_address] = _isBlacklisting;
603 |     }
612 |     function _beforeTokenTransfer(
613 |         address from,
614 |         address to,
615 |         uint256 amount
616 |     ) override internal virtual {
617 |         require(!blacklists[to] && !blacklists[from], "Blacklisted");
627 |     }
629 |     function burn(uint256 value) external {
630 |         _burn(msg.sender, value);
631 |     }
632 | }
```

In line 601, the variable "`_address`" should be constrained to ensure it is not the zero address. Otherwise, the "`burn`" function in line 629 will become unavailable due to the check implemented in line 617.

Resolution

The team acknowledged this finding. The issue has been fixed by commit [2d7a267](2d7a267).

MINU TOKEN CONTRACT

[I-2] Certain parameters should be greater than zero

```
/* src/MantleInuToken.sol */
590 | contract MantleInuToken is Ownable, ERC20 {
591 |     bool public limited;
592 |     uint256 public maxHoldingAmount;
593 |     uint256 public minHoldingAmount;
594 |     address public uniswapV2Pair;
595 |     mapping(address => bool) public blacklists;
596 |
597 |     constructor(uint256 _totalSupply) ERC20("Mantle Inu Token", "MINU") {
598 |         _mint(msg.sender, _totalSupply);
599 |     }
628 |
629 |     function burn(uint256 value) external {
630 |         _burn(msg.sender, value);
631 |     }
632 | }
```

In line 597, the variable "`_totalSupply`" should be constrained to ensure it is greater than zero; otherwise, the token will not be minted.

In line 629, the variable "`value`" should be restricted to values greater than zero.

Resolution

The team acknowledged this finding and clarified that, since token minting has been completed, further checks are no longer necessary.

MINU TOKEN CONTRACT

[I-3] Validate setRule parameters

```
/* src/MantleInuToken.sol */
590 | contract MantleInuToken is Ownable, ERC20 {
605 |     function setRule(bool _limited, address _uniswapV2Pair, uint256 _maxHoldingAmount,
                          uint256 _minHoldingAmount) external onlyOwner {
606 |         limited = _limited;
607 |         uniswapV2Pair = _uniswapV2Pair;
608 |         maxHoldingAmount = _maxHoldingAmount;
609 |         minHoldingAmount = _minHoldingAmount;
610 |     }
611 |
612 |     function _beforeTokenTransfer(
613 |         address from,
614 |         address to,
615 |         uint256 amount
616 |     ) override internal virtual {
617 |         require(!blacklists[to] && !blacklists[from], "Blacklisted");
618 |
619 |         if (uniswapV2Pair == address(0)) {
620 |             require(from == owner() || to == owner(), "trading is not started");
621 |             return;
622 |         }
623 |
624 |         if (limited && from == uniswapV2Pair) {
625 |             require(super.balanceOf(to) + amount <= maxHoldingAmount
                            && super.balanceOf(to) + amount >= minHoldingAmount, "Forbid");
626 |         }
627 |     }
628 |
632 | }
```

In the "`setRule`" function, the variable "`_maxHoldingAmount`" should be set to a value greater than or equal to "`_minHoldingAmount`". Otherwise, the check in line 625 will never pass.

Resolution

The team acknowledged this finding.

Appendix: Methodology and Scope of Work

The Sec3 (formerly Soteria) audit team, which consists of Computer Science professors and industrial researchers with extensive experience in smart contract security, program analysis, testing and formal verification, performed a comprehensive manual code review, software static analysis and penetration testing.

Assisted by the Sec3 Scanner developed in-house, the audit team particularly focused on the following work items:

- Check common security issues.
- Check program logic implementation against available design specifications.
- Check poor coding practices and unsafe behavior.
- The soundness of the economics design and algorithm is out of scope of this work

DISCLAIMER

The instance report ("Report") was prepared pursuant to an agreement between Coderrect Inc. d/b/a Sec3 (the "Company") and Mlabs Corp (the "Client"). This Report solely includes the results of a technical assessment of a specific build and/or version of the Client's code specified in the Report ("Assessed Code") by the Company. The sole purpose of the Report is to provide the Client with the results of the technical assessment of the Assessed Code. The Report does not apply to any other version and/or build of the Assessed Code. Regardless of the contents of the Report, the Report does not (and should not be interpreted to) provide any warranty, representation or covenant that the Assessed Code: (i) is error and/or bug free, (ii) has no security vulnerabilities, and/or (iii) does not infringe any third-party rights. Moreover, the Report is not, and should not be considered, an endorsement by the Company of the Assessed Code and/or of the Client. Finally, the Report should not be considered investment advice or a recommendation to invest in the Assessed Code and/or the Client.

This Report is considered null and void if the Report (or any portion thereof) is altered in any manner.

ABOUT

Founded by leading academics in the field of software security and senior industrial veterans, Sec3 (formerly Soteria) is a leading blockchain security company. We are also building sophisticated security tools that incorporate static analysis, penetration testing, and formal verification.

At Sec3, we identify and eliminate security vulnerabilities through the most rigorous process and aided by the most advanced analysis tools.

For more information, check out our website and follow us on twitter.

Sec3™