Mira
ERC20

FINAL REPORT

February ‘2025

Disclaimer:

Security assessment projects are time-boxed and often reliant on information that may be provided by a client, its affiliates, or its partners. As a result, the findings documented in this report should not be considered a comprehensive list of security issues, flaws, or defects in the target system or codebase.

The content of this assessment is not an investment. The information provided in this report is for general informational purposes only and is not intended as investment, legal, financial, regulatory, or tax advice. The report is based on a limited review of the materials and documentation provided at the time of the audit, and the audit results may not be complete or identify all possible vulnerabilities or issues. The audit is provided on an "as-is," "where-is," and "as-available" basis, and the use of blockchain technology is subject to unknown risks and flaws.

The audit does not constitute an endorsement of any particular project or team, and we make no warranties, expressed or implied, regarding the accuracy, reliability, completeness, or availability of the report, its content, or any associated services or products. We disclaim all warranties, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

We assume no responsibility for any product or service advertised or offered by a third party through the report, any open-source or third-party software, code, libraries, materials, or information linked to, called by, referenced by, or accessible through the report, its content, and the related services and products. We will not be liable for any loss or damages incurred as a result of the use or reliance on the audit report or the smart contract.

The contract owner is responsible for making their own decisions based on the audit report and should seek additional professional advice if needed. The audit firm or individual assumes no liability for any loss or damages incurred as a result of the use or reliance on the audit report or the smart contract. The contract owner agrees to indemnify and hold harmless the audit firm or individual from any and all claims, damages, expenses, or liabilities arising from the use or reliance on the audit report or the smart contract.

By engaging in a smart contract audit, the contract owner acknowledges and agrees to the terms of this disclaimer.

1. Project Details

Important:
Please ensure that the deployed contract matches the source-code of the last commit hash.

Project	Mira – ERC20
Website	mira.network
Language	Solidity
Methods	Manual Analysis
Github repository	https://basescan.org/token/0x7aafd31a321d3627b30a8e2171264b56852187fe#code (Users should verify that the deployed source code matches the code from the audited commit)

2. Detection Overview

Severity	Found	Resolved	Partially Resolved	Acknowledged (no change made)	Failed Resolution
High					
Medium					
Low					
Informational					
Governance					
Total					

2.1 Detection Definitions

Severity	Description
High	The problem poses a significant threat to the confidentiality of a considerable number of users' sensitive data. It also has the potential to cause severe damage to the client's reputation or result in substantial financial losses for both the client and the affected users.
Medium	While medium-level vulnerabilities may not be easy to exploit, they can still have a major impact on the execution of a smart contract. For instance, they may allow public access to critical functions, which could lead to serious consequences.
Low	Poses a very low-level risk to the project or users. Nevertheless, the issue should be fixed immediately.
Informational	Effects are small and do not pose an immediate danger to the project or users.
Governance	Governance privileges which can directly result in a loss of funds or other potential undesired behavior.

3. Detection

Mira

The Mira contract is a simple ERC20 contract with permit functionality, extended with ERC20Votes which allows for the delegation of voting power.

The following contracts are inherited in an effort to support the ERC20Votes module:

- ERC20Votes
- Votes
- Checkpoints
- Math
- Time
- Panic

The following contracts are inherited in an effort to supper the permit functionality:

- Nonces
- ERC20Permit
- ShortStrings
- ECDSA
- EIP712
- MessageHashUtils
- Strings
- StorageSlot

All files originate from OpenZeppelin and are not modified.

Appendix: ERC20Votes

ERC20Votes is an OpenZeppelin extension that adds governance voting capabilities to a standard ERC20 token.
It keeps a historical record of each account’s voting power and makes that power available for on-chain governance processes.
Users can delegate their votes to themselves or another address, which allows flexible representation in governance decisions.
Voting units typically map 1:1 to token balances, but only delegated balances actively count in voting outcomes.
The module limits total token supply to type(uint208).max by default, preventing overflows and ensuring accurate checkpoint math.

Deployed Blockchain:

Ethereum

Trust Assumptions:

This contract is permissionless

Decimals:

18

Initial mint:

1000000000

Minting occasions:

Minting only happens during the deployment

Privileged Functions

- none

No issues found. All contracts are 1:1 OpenZeppelin contracts.