CODE SECURITY ASSESSMENT

ARIA PROTOCOL

Overview

Project Summary

- Name: Aria Protocol - CommemorativeNFT
- Platform: EVM-compatible chains
- Language: Solidity
- Repository: https://github.com/AriaProtocol/binance-activation-nft
- Audit Range: See Appendix - 1

Project Dashboard

Application Summary

Name	Aria Protocol - CommemorativeNFT
Version	v2
Type	Solidity
Dates	Dec 25 2024
Logs	Dec 19 2024; Dec 25 2024

Vulnerability Summary

Total High-Severity issues	0
Total Medium-Severity issues	0
Total Low-Severity issues	1
Total informational issues	1
Total	2

Contact

E-mail: support@salusec.io

Risk Level Description

High Risk	The issue puts a large number of users' sensitive information at risk, or is reasonably likely to lead to catastrophic impact for clients' reputations or serious financial implications for clients and users.
Medium Risk	The issue puts a subset of users' sensitive information at risk, would be detrimental to the client's reputation if exploited, or is reasonably likely to lead to a moderate financial impact.
Low Risk	The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
Informational	The issue does not pose an immediate risk, but is relevant to security best practices or defense in depth.

Content

Introduction

1.1 About SALUS

At Salus Security, we are in the business of trust.

We are dedicated to tackling the toughest security challenges facing the industry today. By building foundational trust in technology and infrastructure through security, we help clients to lead their respective industries and unlock their full Web3 potential.

Our team of security experts employ industry-leading proof-of-concept (PoC) methodology for demonstrating smart contract vulnerabilities, coupled with advanced red teaming capabilities and a stereoscopic vulnerability detection service, to deliver comprehensive security assessments that allow clients to stay ahead of the curve.

In addition to smart contract audits and red teaming, our Rapid Detection Service for smart contracts aims to make security accessible to all. This high calibre, yet cost-efficient, security tool has been designed to support a wide range of business needs including investment due diligence, security and code quality assessments, and code optimisation.

We are reachable on Telegram (https://t.me/salusec), Twitter (https://twitter.com/salus_sec), or Email (support@salusec.io).

1.2 Audit Breakdown

The objective was to evaluate the repository for security-related issues, code quality, and adherence to specifications and best practices. Possible issues we looked for included (but are not limited to):
  - Risky external calls
  - Integer overflow/underflow
  - Transaction-ordering dependence
  - Timestamp dependence
  - Access control
  - Call stack limits and mishandled exceptions
  - Number rounding errors
  - Centralization of power
  - Logical oversights and denial of service
  - Business logic specification
  - Code clones, functionality duplication

1.3 Disclaimer

Note that this security audit is not designed to replace functional tests required before any software release and does not give any warranties on finding all possible security issues with the given smart contract(s) or blockchain software, i.e., the evaluation result does not guarantee the nonexistence of any further findings of security issues.

Findings

2.1 Summary of Findings

ID	Title	Severity	Category	Status
1	Missing zero address checks	Low	Data Validation	Resolved
2	Missing two-step transfer ownership pattern	Informational	Business logic	Resolved

2.2 Notable Findings

Significant flaws that impact system confidentiality, integrity, or availability are listed below.

1. Missing zero address checks	
Severity: Low	Category: Data Validation
Target: - CommemorativeNFT.sol	

Description

It is considered a security best practice to verify addresses against the zero address during initialization or configuration. However, this check is missing for the signer address variables (e.g., signer). If the signer address is set to the zero address, it can lead to signature misuse because an invalid signature recovered by the recover function would also return address(0).

CommemorativeNFT.sol:L94-L97

```
function setSigner(address signer_) external onlyOwner {
    _getCommemorativeNFTStorage().signer = signer_;
    emit CommemorativeNFTSignerUpdated(signer_);
}
```

Recommendation

Consider adding zero address checks for address variables.

Status

The team has resolved this issue in commit bb14fec.

SALUS

2.3 Informational Findings

2. Missing two-step transfer ownership pattern	
Severity: Informational	Category: Business logic
Target: - CommemorativeNFT.sol	

Description

The `CommemorativeNFT` contract inherits from the Ownable contract. This contract does not implement a two-step process for transferring ownership. Thus, ownership of the contract can easily be lost when making a mistake in transferring ownership.

Recommendation

Consider using the OwnableUpgradeable contract from OpenZeppelin instead.

Status

The team has resolved this issue in commit bb14fec.

SALUS

Appendix

Appendix 1 - Files in Scope

This audit covered the following files in commit 1e6882a:

File	SHA-1 hash
CommemorativeNFT.sol	a3ddc7b74a8b7d2dece28dea04b37637b74b3e6d

SALUS