

ADC Token

Security Audit

March 23rd, 2023

V 1.0.0

Prepared by
Mayank Meena
(fiverr.com/mayankmeena)
(<https://mayank-meena.vercel.app/>)

Introduction	3
Additional Info about Audited Project	4
Source Code	5
Methodology	6
Issues Descriptions and Recommendations	7
Report	8
Functions (Read & Write)	10
Disclaimer	11

Introduction

This document includes the results of the security audit for [smart contract](#) (0x62d0dEe9871B21EE92D8838C5d1c302746F176aD on Core chain provided by [mike](#) on fiverr) code as found in the section titled 'Source Code'. The security audit was performed by the [Mayank's](#) security team from Mar 23, 2023 to Mar 23, 2023. The purpose of this audit is to review the source code of certain Solidity contracts, and provide feedback on the design, architecture, and quality of the source code with an emphasis on validating the correctness and security of the software in its entirety.

Disclaimer:

While [Mayank's](#) team review is comprehensive and has surfaced some changes that should be made to the source code, this audit should not solely be relied upon for security, as no single audit is guaranteed to catch all possible bugs.

Overall Assessment

We identified some of low to high-severity issues.

Specification

Our understanding of the specification was based on the following sources:

- Discussions on Fiverr with [Mike](#)
- The [Source code](#) found on the blockchain explorer.

Additional Info about Audited Project

Contract Name - ADC Token

Contract Type - ERC20

Contract Add. - 0x62d0dEe9871B21EE92D8838C5d1c302746F176aD

Blockchain explorer link - [link](#)

Compiler Version - v0.8.4

License - MIT

Decimal - 18

Total Supply - 5e+26(50000000000000000000000000000000)

Contract Owner - [0xa76D5942452225505Dd833f3B0F87986C87B1A62](#)

Source Code

the following source code was reviewed during the audit:

Link:- [Source Code](#)

Note: This document contains an audit solely of the Solidity contracts listed above. Specifically, the audit pertains only to the contracts themselves and does not pertain to any other programs or scripts, including deployment scripts and libraries.

Libraries used in Contract

<pre>"@openzeppelin/contracts/token/ERC20/IERC20.sol"; "@openzeppelin/contracts/access/Ownable.sol"; "@openzeppelin/contracts/utils/math/SafeMath.sol";</pre>

Methodology

The audit was conducted in several steps.

First, we reviewed in detail all available documentation and specifications for the project, as described in the ‘Specification’ section above.

Second, we performed a thorough manual review of the code, checking that the code matched the specification and the spirit of the contract (i.e. the intended behavior). During this manual review portion of the audit, we primarily searched for security vulnerabilities, unwanted behavior vulnerabilities, and problems with systems of incentives.

Third, we performed the automated portion of the review consisting of measuring test coverage (while also assessing the quality of the test suite) and evaluating the results of various symbolic execution tools against the code.

Lastly, we performed a final line-by-line inspection of the code – including comments –in an effort to find any minor issues with code quality, documentation, or best practices.

Issues Descriptions and Recommendations

Severity Level Reference:

Level	Description
High	The issue poses an existential risk to the project, and the issue identified could lead to massive financial or reputational repercussions. We highly recommend fixing the reported issue. If you have already deployed, you should upgrade or redeploy your contracts.
Medium	The potential risk is large, but there is some ambiguity surrounding whether or not the issue would practically manifest. We recommend considering a fix for the reported issue.
Low	The risk is small, unlikely, or not relevant to the project in a meaningful way. Whether or not the project wants to develop a fix is up to the goals and needs of the project.
Code Quality	The issue identified does not pose any obvious risk, but fixing it would improve overall code quality, conform to recommended best practices, and perhaps lead to fewer development issues in the future.
Gas Optimizations	The presented optimization suggestion would save an amount of gas significant enough, in our opinion, to be worth the development cost of implementing it.

Based on these Levels we have not found any particular issue in smart contract after we have found after checking all parameters and given information by [Mike](#).

Report on Smart Contract for [ADC Token](#):

Code Audit Report

The contract code is a standard implementation of the ERC-20 token. The contract is implemented in Solidity version 0.8.4.

Security

The code is written using the OpenZeppelin library that provides well-audited, battle-tested security implementations for various functionalities.

The contract uses SafeMath to prevent overflow and underflow vulnerabilities. Also, the transferFrom function implements a check to ensure that the spender has the allowance to spend the tokens.

Style

The code follows the standardized Solidity naming conventions for functions and variables.

The code also includes inline comments, which help understand the code.

Gas Optimization

The contract uses “using” to minimize gas consumption for the SafeMath library.

Recommendations

There are no critical issues found in the code, and the implementation is satisfactory. However, some improvements could be made for readability and optimization.

It would be better to add more inline comments to explain the code's working and functions' purpose.







The version variable is unused, and it could be removed.

It would be better to include the version of the ERC-20 interface that is being implemented, for clarity purposes.

Overall, the code is secure and follows the standard conventions. It can be deployed safely.

Note - We have only checked and audited the contract (0x62d0dEe9871B21EE92D8838C5d1c302746F176aD).

The contract is similar to 6 well-known contracts.

 pinksale / StandardToken.sol	100% ▾
 pinksale Legacy / StandardToken.sol	100% ▾
 pinksale / AntiBotStandardToken.sol	96% ▾
 pinksale Legacy / AntiBotStandardToken.sol	96% ▾
 dxsale / StandardToken.sol	88% ▾
 coinscope / StandardToken.sol	87% ▾

Audit

Can Set Fees	✔ Safe
Can Mint	✔ Safe
Can Burn	✔ Safe
Can Blacklist	✔ Safe
Can Blacklist Massively	✔ Safe
Can Whitelist	✔ Safe
Can Cooldown Transfers	✔ Safe
Can Pause Transfers	✔ Safe
Can change max tx amount	No

All Functions of Smart Contract

READ Functions

1. Version - Shows Version of smart contract
2. Balance Of - shows the token balance of the wallet
3. allowance - check allowance of a particular wallet
4. Decimals - shows the decimal quantity of token
5. Name - Name of contract
6. Owner - shows wallet address of contract owner
7. Symbol - Symbol of contract
8. Total supply - shows the total quantity of tokens

Write Functions

1. Transfer Ownership - Transfer ownership of contract to someone else.
2. Transfer / TransferFrom - for transferring token
3. Renounce Ownership - Renounce ownership to 0 address
4. Increase / decrease Allowance - change the allowance of wallet
5. Approve - for approving token for transfer/ trading.

Disclaimer

Mayank's team makes no warranties, either express, implied, statutory, or otherwise, with respect to the services or deliverables provided in this report, and Mayank's team specifically disclaims all implied warranties of merchantability, fitness for a particular purpose, noninfringement and those arising from a course of dealing, usage or trade with respect thereto, and all such warranties are hereby excluded to the fullest extent permitted by law. Mayank's team will not be liable for any lost profits, business, contracts, revenue, goodwill, production, anticipated savings, loss of data, or costs of procurement of substitute goods or services or for any claim or demand by any other party. In no event will Mayank's team be liable for consequential, incidental, special, indirect, or exemplary damages arising out of this agreement or any work statement, however, caused and (to the fullest extent permitted by law) under any theory of liability (including negligence), even if Mayank's team has been advised of the possibility of such damages. The scope of this report and review is limited to a review of only the code presented by the Emergent team and only the source code Mayank's team notes as being within the scope of Mayank's team's review within this report. This report does not include an audit of the deployment scripts used to deploy the Solidity contracts in the repository corresponding to this audit. Specifically, for the avoidance of doubt, this report does not constitute investment advice, is not intended to be relied upon as investment advice, is not an endorsement of this project or team, and it is not a guarantee as to the absolute security of the project. In this report you may through hypertext or other computer links, gain access to websites operated by persons other than Mayank's team. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such websites' owners. You agree that Mayank's team is not responsible for the content or operation of such websites, and that Mayank's team shall have no liability to your or any other person or entity for the use of third party websites. Mayank's team assumes no responsibility for the use of third party software and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any outcome generated by such software.
