# Dexmex

# ERC20 Token Audit


## The BlockChain Auditor

Prepared by: Jorge Martinez

Version: 1
Date: Feb. 12, 2021

# Summary of Findings

This document expresses all security concerns of the Dexmex Smart ERC20 Contract as expressed by Jorge Martinez. I took care to attempt to find as many ways to improve the security, code efficiency, best practices, and overall function of the smart contracts.

## Contract Status: Safe to Use

Contract is deployed at the following address:
0x0020d80229877b495D2bf3269a4c13f6f1e1B9D3

0 Critical Issue(s) found were found.
0 Medium Issue(s) were found.
0 Low Issue(s) were found.
0 Informational Issue(s) were found.

## Solidity Code Coverage

| Jorge's Test Suite | Dexmex | Industry Standard |
|---|---|---|
| 100% | 0% | 95% |

For this audit, I wasn't provided with a testing suite but as part of my audit methodology I developed a test suite to verify the functionality of the ERC20 contract, check its security, and to help reveal any underlying issues.

This audit should be seen as one step in the development process with the intent of raising awareness on the meticulous work involved in secure development and making no material statements or guarantees to the operational state of the smart contract(s) once they are deployed. This document is not an endorsement of the reliability or effectiveness of the smart contracts. This is an assessment of the smart contract logic, implementation, and best practices. I cannot take responsibility for any potential consequences of the deployment or use of the smart contract(s) related to the audit.

# Test Suite Results

## Jorge's Test Suite

```
  Dexmex ERC20 Security Test Suite
    Deployment
      ✓ dexmex is locked upon deployment (380ms)
      ✓ Should have a total supply of 50,000,000
    Lockability functionality
      ✓ should be lockable by the owner (68ms)
      ✓ should be unlockable by the owner
    Whitelist functionality
      ✓ the contract owner can add users to whitelist
      ✓ the contract owner can remove users to whitelist
    Trading Behavior
      ✓ when the contract is locked only the owner or addresses on the whitelist can trade (86ms)
      ✓ when the contract is unlocked everyone can trade (47ms)
      ✓ when the contract is locked only the owner or addresses on the whitelist can use transferFrom() (86ms)
      ✓ when the contract is unlocked any approved addresses can use transferFrom() (78ms)
      ✓ if an address is whitelisted, an approved address could tranfer tokens from the whitelisted address (47ms)


  11 passing (873ms)

-------------|----------|----------|----------|----------|----------------|
File         |  % Stmts | % Branch |  % Funcs |  % Lines |Uncovered Lines |
-------------|----------|----------|----------|----------|----------------|
 contracts/  |      100 |      100 |      100 |      100 |                |
  Dexmex.sol |      100 |      100 |      100 |      100 |                |
-------------|----------|----------|----------|----------|----------------|
All files    |      100 |      100 |      100 |      100 |                |
-------------|----------|----------|----------|----------|----------------|
```

# Table of Contents

# Audit Methodology & Techniques

The BlockChain Auditor has the following auditing process:

1. Our audits include
   a. Review of the specifications, source code, and instructions provided to the TheBlockchainAuditor to clearly identify the desired functionality of the smart contract(s).
   b. Manual line by line review of contract code to spot potential vulnerabilities.
   c. Identification of deviations between desired functionality expressed to the TheBlockchainAuditor and what the smart contract(s) are doing.
2. Automated static and symbolic analysis, as well as verifying testing coverage using the provided test suite.
   a. Automated static and symbolic analysis help determine what inputs cause each part of the smart contract to execute. Analysis of how much of the code base is tested and comparison to industry standard.
3. Examination of smart contracts and development process as a whole, ensuring best practices are followed, allowing improved efficiency and security based on established industry and academic practices.
4. Specific, itemized, and actionable recommendations to assist in securing the smart contract(s) in question.

# Contract Checklist

## Dexmex.sol

| **Contract Vulnerability** | |
|---|---|
| Integer Overflow | Pass |
| Race Condition | Pass |
| Denial of Service | Pass |
| Logical Vulnerability | Pass |
| Hardcoded Address | Pass |
| Function Input Parameter Check | Pass |
| Function Access Control Check | Pass |
| Random Number Generation | N/A |
| Random Number Use | N/A |
| **Contract Specification** | |
| Solidity Compiler Version | Pass |
| Event Use | Pass |
| Fallback Function Use | Pass |
| Constructor Use | Pass |
| Function Visibility Declaration | Pass |
| Variable Storage Declaration | Pass |
| Deprecated Keyword Use | Pass |
| ERC20/223 Standard | Pass |
| ERC721 Standard | N/A |
| **Business Risk** | |
| Able to Arbitratrily Create Token | N/A |
| Able to Arbitrarily Destroy Token | N/A |
| Can Suspend Transactions | Pass |
| Short Address Attack | Pass |
| **Gas Optimization** | |
| assert()/require()/revert() misused | Pass |
| Loop Optimization | Pass |
| Storage Optimization | Pass |

# Issue Classification

## Critical

These issues in the smart contract can have catastrophic implications that could ruin your reputation, disrupt the contract's functionality, or impact the client and your users' sensitive information.

## Informational

This issue relates to style and security best practices but does not pose an immediate risk.

## Medium

An issue classified as medium has relatively small risk and isn't exploitable to circumvent desired functionality and could not have financial consequences but could put user's sensitive information at risk.

## Acknowledged

The issue remains in the code but is a result of an intentional business or design decision.

## Low

An issue classified as informational does not pose an immediate threat to disruption of functionality and could not be exploited on a recurring basis, however, it should be considered for security best practices or code integrity.

## Unresolved

Although the client has been informed of the risk, it was decided to accept it because it was not relevant in the functionality of the smart contract.

## Undetermined

The impact of the issue is uncertain and more investigation is required to understand the repercussions of the issue.

## Mitigated

Actions were taken to minimize the impact or likelihood of the risk.

# Executive Summary

## Overall Thoughts

Since OpenZeppelin libraries were used extensively I was able to develop a test suite that focuses on what the team added to the standard ERC20 token contract which in this case is a whitelist, lockability, and overloaded transfer() and transferFrom() functions that interact with the whitelist and lockability. I was able to verify that the added mechanisms worked as intended. This token is EIP20 compliant so users will be able to transfer the token as expected. As mentioned, this token is lockable. It also only mints during deployment. If an exploit is found the owner of the token is able to lock transactions while they figure out what is going on. All in all as long as the user is aware of lockability then this contract should be fine.

# Appendix A

## File Fingerprints

| | |
|---|---|
| Dexmex.sol | 8276f2e58fb02e764b0097f1dc3fdbf6 |

The Blockchain Auditor is honored to have the opportunity to help verify the functionality and security for Dexmex ERC20 contract and help provide security for users of the token.

I look forward to seeing how Dexmex will do next.

The BlockChain Auditor

- Jorge Martinez