DataBrokerDAO Smart Contracts Audit by ZK Labs

MATTHEW DI FERRANTE

2018-03-18

Introduction

On 2018-03-18, Matthew Di Ferrante performed an audit of the DataBrokerDAO smart contracts. My findings are detailed below.

I, Matthew Di Ferrante have no stake or vested interest in DataBrokerDAO. This audit was performed under a contracted rate with no other compensation.

Authenticity

This document should have an attached cryptographic signature to ensure it has not been tampered with. The signature can be verified using the public key from http://keybase.io/mattdf

Audit Goals and Focus

Smart Contract Best Practices

This audit will evaluate whether the codebase follows the current established best practices for smart contract development.

Code Correctness

This audit will evaluate whether the code does what it is intended to do.

Code Quality

This audit will evaluate whether the code has been written in a way that ensures readability and maintainability.

Security

This audit will look for any exploitable security vulnerabilities, or other potential threats to either the operators of ChainLink or its users.

Testing and testability

This audit will examine how easily tested the code is, and review how thoroughly tested the code is.

About DataBrokerDAO

DataBroker DAO is a blockchain-backed marketplace to sell & buy sensor data. As a decentralized marketplace for IoT sensor data using Blockchain technology, Databroker DAO enables sensor owners to turn generated data into revenue streams.

Terminology

This audit uses the following terminology.

Likelihood

How likely a bug is to be encountered or exploited in the wild, as specified by the OWASP risk rating methodology.

Impact

The impact a bug would have if exploited, as specified by the OWASP risk rating methodology.

Severity

How serious the issue is, derived from Likelihood and Impact as specified by the OWASP risk rating methodology.

Overview

Source Code

The DataBrokerDAO smart contract source code was made available in the https://github.com/DataBrokerDAO/dtx-crowdsale-contracts Github repository.

The following files were audited:

```
1 4ae9568962813a5ef1616804ed8778497cbe7e29704fa71ccb7145a9c3ebb487  DTXToken
    .sol
2 6fc02820963f1b2228d98e80c00b800ec1b1538e72d3f88de0712ffb1ee77826
    TokenSale.sol
```

The code makes use of OpenZeppelin and MiniMe library code, which was *not* audited as part of this audit.

General Notes

The code is generally well structured, self contained and easy to read. It makes use of the MiniMe and OpenZeppelin smart contracts, which reduces the count of lines that need to be independently audited and the risk of bugs.

Contracts

`DTXToken` is simply an instantiation of a MiniMe token with the standard parameters and the symbol "DTX". No issues in this contract.

`TokenSale` is the primary logic for the crowdsale. The entry point is through `doPayment`, and a sale with 3 phases is implemented: a presale, day one, and main sale phase. For successful buys, tokens are issued directly to the purchase address, and ether is sent to the vault address.

Beyond `doPayment`, the sale has a `handleExternalBuyers` function, which registers private and bitcoin buyers with locked/vested tokens if applicable, and an `handleEarlySaleBuyers` function, which registers contributions from the initial presale. Both of these functions may be called only by the controller.

Once either the sale has hit the cap or end time has been reached, `finalizeSale` may be called, which registers a number of vested tokens to the vault address, and issues any left over amount which was not bought to the vault.

Any vested tokens can be claimed by their owners through `claimTokens` once the vesting period has elapsed.

The crowdsale allows the controller to change the Token Controller through the `changeTokenController` function.

Testing

Test coverage is fairly complete for all TokenSale functionality.

Findings

We found 1 note issue.

Note Issues

The token controller can change to an arbitrary address

- Likelihood: low
- Impact: low

The token controller can change during or after the crowdsale to any arbitrary address, so one must trust the crowdsale operators to not ever break constraints or assumptions introduced in the crowdsale.

Low Issues

None found.

Medium Issues

None found.

High Issues

None found.

Critical Issues

None found.