DAV Crowdsale Smart Contracts Audit by ZK Labs

MATTHEW DI FERRANTE

2018-05-06

Introduction

On 2018-05-06, Matthew Di Ferrante performed an audit of the DAV Foundation smart contracts. My findings are detailed below.

I, Matthew Di Ferrante have no stake or vested interest in DAV Foundation. 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 the crowdsale 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 DAV Foundation

DAV is a Blockchain-based transportation protocol, enabling a Decentralized, Peer to Peer, Global Transportation Network.

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 DAV Foundation Crowdsale smart contract source code was made available in the DAVFoundation/contracts Github repository.

The following files were audited:

```
1 d16a03fc4039bfe152bee5b6d42d35fe95738730d512267bdc5c9a05e987ff34
    DAVCrowdsale.sol
2 4d0d26100933c681b0225644e2a6563192bd3d5e401f36a79098a180bd379dff  DAVToken
    .sol
3 b39647cb6dd1630f57209114cdefea82cc41aacd1e731b23396571c700d33fdc
    OwnedPausableToken.sol
```

```
4 72065afed9e746d854273803d4e9760a7b6d0a7ba273f86257c20377da4dae46
    PausableCrowdsale.sol
```

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

General Notes

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

Contracts

DAVCrowdsale.sol

The `DAVCrowdsale` contract implements the main crowdsale logic. It inherits from OpenZeppelin's `FinalizableCrowdsale` and implements a `Pausable` pattern.

The crowdsale has two different whitelists:

- A, which allows beneficiaries to participate at any time once the crowdsale is operational
- B, which allows beneficiaries to participate after an opening time specific to B (which must be later than A's opening time)

The whitelists can be modified prior to and during the crowdsale's operation.

The crowdsale restrictions also include:

- a minimal contribution amount
- a cumulative maximum contribution amount per address
- a gas price limit per tx
- a total raise limit in wei and vinci

The crowdsale owner is able to pause the sale at any time, and is also able to end the sale early through the `closeEarly` function.

The functions `_preValidatePurchase`, `_updatePurchasingState`, and `finalization` are overloaded from OpenZeppelin's Crowdsale to apply the restrictions above, record contributions and finalize the crowdsale, respectively.

The only other function is `recordSale`, which is used to allocate presale purchases to a `lockedTokensWallet` address.

Upon finalization, up to 1.5x of the sold tokens are transferred to the foundation wallet - and if there are any unsold tokens left at the end of crowdsale beyond the foundation allocation, they are burned.

DAVToken.sol

The `DAVToken` contract implements a pausable, burnable, ownable token - with almost all functionality inherited from OpenZeppelin contracts. The main functionl additions are:

- An initial supply set in the constructor, assigned to the deployer of the contract
- a `setPauseCutoffTime` function, which once called ensures the token can no longer be paused beyond the specified timestamp

OwnedPausableToken.sol

The `OwnedPausableToken` contract is inherited by `DAVToken`, and simply adds the `whenNotPaused` modifier to all ERC20 functions, along with adding a `whenNotPausedOrOwner` modifier to the `transfer` function.

PausableCrowdsale.sol

The `PausableCrowdsale` contract inherits from `Pausable` and `Crowdsale` by OpenZeppelin, and adds the `whenNotPaused` modifier to the invocation of `_preValidatePurchase`, making purchases impossible if the crowdsale contract is paused.

Testing

Test coverage is adequate, covering end-to-end functionality and tests for each individual contract.

Findings

We found 1 note issue.

Note Issues

The crowdsale owner can end the sale early

- Likelihood: low

- Impact: low

Having the ability to end the crowdsale early may have impact on investors buying decisions and should be clearly stated in the participation terms, so as to not surprise possible investors.