Published in New Alchemy

New Alchemy Follow
Apr 16, 2018 · 6 min read · Listen

The Abyss DAICO Smart Contract Audit (New)

New Alchemy

During April 2018, The Abyss Team engaged New Alchemy to perform an additional audit of changes made to the contracts that support their digital distribution platform. The engagement was technical in nature and focused specifically on locations where functionality has been added, removed, or altered. New Alchemy's assessment focused primarily on code review of the contracts, with an emphasis on discovery of security flaws, differences between the contracts and public documentation, and any other issues that may impact contract trustworthiness.

The audit was performed over four days. This document describes the issues discovered in the audit. After receiving the report, The Abyss made several changes to the contracts in order to address issues. The report has been updated to reflect these changes.

`b31a0f2172d10c9343c5a3b644a584e74b409683`.

The Abyss deployed changes at commit hash `21d6232a802cd966b86629cadeac0b7f3816f719`, which New Alchemy audited as well.

New Alchemy's audit was additionally guided by the following documents:

- The Abyss Whitepaper, version 1.9 (March 2018)
- The Abyss Frequently Asked Questions
- A Guide to The Abyss DAICO Smart Contract (7 April 2018)
- The Abyss Contract Audit

> *The smart contract audit did not discover any issues of critical severity.*

New Alchemy identified several moderate and minor issues that revolve around transparency, minor timing attacks, and coding best practices. While the overall security impact of these issues is negligible, The Abyss should correct them if possible.

The Abyss uses code from publicly-available libraries including OpenZeppelin and a DateTime conversion contract. Since these libraries have received significant analysis, this helped to reduce the overall effort required to audit the project.

General Discussion

The Abyss contracts have been altered since New Alchemy's initial assessment, with the key change being the removal of Oracle voting. This feature revolved around how to control the DAICO model of 'tapping' funds. While the tap can still be controlled, it has been changed to use token holders as opposed to 'Oracles' from the gaming industry.

A breakdown of the contract changes are as follows:

Removed contracts

- ISimpleCrowdsale.sol
- ReservationFund.sol
- fund/ICrowdsaleReservationFund.sol

Modified/Renamed contracts

- RefundVoting.sol renamed to RefundPoll.sol
- TapVoting.sol renamed to TapPoll.sol
- fund/IVotingManagedFund.sol renamed to fund/IPollManagedFund.sol
- voting/BaseVoting.sol renamed to voting/BasePoll.sol
- VotingManagedFund.sol renamed to PollManagedFund.sol

Many of the changes are simple refactoring adjustments, such as altering variable/function/contract names to more accurately reflect their usage.

The code initially used Solidity version 0.4.18. Upon receiving the report, The Abyss updated the contracts to use the most recent version, 0.4.21.

Moderate Issues

Fixed: ReservationFund owner can change crowdsale address at any time

`ReservationFund.sol` includes a function that allows the contract owner to specify the address of the token crowdsale:

```
function setCrowdsaleAddress(address crowdsaleAddress) public
onlyOwner {
        crowdsale = ISimpleCrowdsale(crowdsaleAddress);
    }
```

There are no restrictions on this function being used more than once, which could

crowdsale contributions. Since this function should only be called once, add a check to ensure that it is not called a second time.

Re-test results: The Abyss corrected this issue by adding the suggested check.

Minor Issues

Not Fixed: Crowdsale logic depends on Ethereum block timestamp

The logic for determining the stage of the token sale and whether the sale has started or ended is made by using `now`, an alias for `block.timestamp`. For instance, in `Crowdsale.sol`:

```
function isValidContribution() internal view returns(bool) {
        if(now < SALE_START_TIME || now > SALE_END_TIME) {
            return false;

        }
        uint256 currentUserContribution = safeAdd(msg.value,
            userTotalContributed[msg.sender]);
        if(whiteList[msg.sender] && msg.value >= ETHER_MIN_CONTRIB)
{
            if(now <= MAX_CONTRIB_CHECK_END_TIME &&
currentUserContribution >
                ETHER_MAX_CONTRIB ) {
                    return false;
            }
            return true;

        }
        if(privilegedList[msg.sender] && msg.value >=
ETHER_MIN_CONTRIB_PRIVATE) {
            if(now <= MAX_CONTRIB_CHECK_END_TIME &&
currentUserContribution >
                ETHER_MAX_CONTRIB_PRIVATE ) {
                    return false;
            }
            return true;
        }
```

```
                    return false;
                }
                return true;
            }

            return false;
        }
```

In addition to this, the `processReservationContribution`, `finalizeCrowdsale`, and `getBonus` implement similar behavior. The 'Poll' functionality uses the same methodology to determine the duration of the poll.

Per the Ethereum technical specification, this value can be manipulated by miners up to 900 seconds per block. While unlikely, it is possible that a miner could maliciously alter the behavior of the token sale by altering the block timestamp. See https://github.com/ethereum/wiki/wiki/Safety#timestamp-dependence for more information.

To fix this issue, consider defining the duration for the token sale in terms of block height, instead of timestamp values. While using block height is just an approximation, it prevents a malicious miner from manipulating timestamps.

Re-test results: The Abyss decided not to fix this issue, as the risk is negligible.

Not Fixed: Hard and soft cap are not defined

The Abyss uses a model that defines a soft and hard cap for the crowdsale in terms of USD. Since the crowdsale operates in Ether this value will fluctuate, forcing The Abyss to set the prices directly before the crowdsale. While this is a standard practice, it requires purchasers to verify the information after the crowdsale has started. New Alchemy verified that these values can only be set once. Consider specifying exactly when this information will be set so that token purchasers know what to expect.

Re-test results: The Abyss did not fix this issue, opting to set the values manually before the crowdsale.

Not Fixed: Contracts do not follow a strict format

```
function addToLists(
        address _wallet,
        bool isInWhiteList,
        bool isInPrivilegedList,
        bool isInLimitedList,
        bool hasAdditionalBonus
    ) public onlyOwner {
        ...
    }
```

and

```
function processPayment(address contributor, uint256 etherAmount,
uint256
    tokenAmount, uint256 tokenBonusAmount, bool
additionalBonusApplied)
    internal {
        ...
    }
```

While this issue does not have a direct security impact, consistent formatting makes code review easier.

Re-test results: The Abyss did not implement changes to formatting for contract functions/constructors.

Fixed: Code repository does not adhere to MIT license

The Abyss uses code from multiple projects, including OpenZeppelin and DateTime. These projects are licensed under the MIT license, which requires that terms be supplied with any redistribution of the code. Currently, The Abyss has not included this information, which is a violation of the license.

Re-test results: This issue has been fixed.

PollManagedFund.sol

Line 73:

Commented out parameter name in function should be removed if not used.

Re-test results: The Abyss did not fix this issue.

Crowdsale.sol

Line 143–146, 224–226, 324:

Consider using a modifier to implement this functionality.

Re-test results: This issue has been fixed.

Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug-free status. The audit documentation is for discussion purposes only.

New Alchemy *is a strategy and technology advisory group specializing in tokenization. One of the only companies to offer a full spectrum of guidance from tactical technical execution to high-level theoretical modeling, New Alchemy provides blockchain technology, token game theory, smart contracts, security audits, and ICO advisory to the most innovative startups worldwide.* ***Get in touch with us at Hello@NewAlchemy.io***