Audit of The BigDataProtocol Contracts

a report of findings by

Arcadia

innovative fortuna iuvat

March 6th, 2021

Table of Contents

Document Info

Client	BigDataProtocol
Title	Smart Contract Audit of BDP Contracts
Approved By	Rasikh Morani

Contact

For more information on this report, contact us below

Rasikh Morani
rasikh@arcadiamgroup.com
https://t.me/thearcadiagroup

Executive Summary

A Representative Party of BigDataProtocol ("BDP") engaged The Arcadia Group ("Arcadia"), a software development, research, and security company, to conduct a review of the following BDP smart contracts at the github repository at commit hash 9fc9ed6ee7e5d403f990c95db62b005b56086a32.

BDPMaster.sol
BDPToken.sol
bAlphaMaster.sol
bAlphaToken.sol

There were 1 issue found, 0 of which were deemed to be ‘critical’, and 0 of which were rated as ‘high’, 1 recommendation for the deployment and execution of the contracts.

The audit also reviewed the deployed contracts and verified code on etherscan to verify whether the issues can be mitigated.

Severity Rating	Number Of Original Occurrences	Number Of Remaining Occurrences
Critical	0	0
High	0	0
Medium	0	0
Low	0	0
Notice	1	0
Informational	0	0

Arcadia completed the reviews using various methods primarily consisting of dynamic and static analysis. This process included a line by line analysis of the in-scope contracts, optimization analysis, analysis of key functionalities and limiters, and reference against intended functionality.

Findings

1. Input parameter is not checked for its value range

- BDP-1
- Severity: Low
- Likelihood: Low
- Impact: Low
- Target: BDPMaster.sol, bAlphaMaster.sol
- Category: Input parameter
- Finding Type: Dynamic

Input parameter `_startBlock` of constructors of contracts BPDMaster and bAlphaMaster should be validated for its value range. The validation is to avoid unexpected input parameters taken as input in the contracts constructors.

Moreover, input parameter `_rewardPerBlock` should be also checked for its input value to avoid unexpected manual mistakes that can make reward per block too high.

```
constructor(
        bAlphaToken _bAlpha,
        uint256 _rewardPerBlock,
        uint256 _startBlock,
        uint256 _halvingAfterBlock
    ) public {
        bAlpha = _bAlpha;
        REWARD_PER_BLOCK = _rewardPerBlock;
        START_BLOCK = _startBlock;
        for (uint256 i = 0; i < REWARD_MULTIPLIER.length - 1; i++) {
            uint256 halvingAtBlock = _halvingAfterBlock.mul(i + 1).add(_startBlock);
            HALVING_AT_BLOCK.push(halvingAtBlock);
        }
        FINISH_BONUS_AT_BLOCK = _halvingAfterBlock.mul(REWARD_MULTIPLIER.length - 1).add(_startBlock);
        HALVING_AT_BLOCK.push(uint256(-1));
    }

constructor(
        BDPToken _BDP,
        uint256 _rewardPerBlock,
        uint256 _startBlock
    ) public {
        BDP = _BDP;
```

```
        REWARD_PER_BLOCK = _rewardPerBlock;
        START_BLOCK = _startBlock;
    }
```

Action Recommended: Add validity checks for `_startBlock` to ensure that `_startBlock` for starting rewards is after the current block.

In the deployed contracts, the input parameters are properly input and initialized, thus the issue is mitigated to have no security risks.

Recommendations

2. Contract ownership should be transferred to a multisignature wallet

- BDP-3
- Severity: Recommendation
- Impact: Recommendation
- Target: BDPMaster.sol, BDPToken.sol, bAlphaMaster.sol, bAlphaToken.sol
- Category: Contract ownership
- Finding Type: Dynamic

In the contracts, the ownership of the contracts should be a timelock contract or a multisignature wallet. This is a recommendation to the deployment and execution of the contracts. Whenever all changes for the initial configuration and initialization of the contracts and when pools are added to the farming, the ownership of those contracts should be transferred to a timelock contract and a multisignature wallet.

Using a multisignature wallet ensures that changes to the contracts do not depend on a single private key.

The ownership of bAlphaToken should be transferred to 0x00 address or be transferred to a multisignature wallet.d

Conclusion

Arcadia identified some issues that occurred at git hash #`9fc9ed6ee7e5d403f990c95db62b005b56086a32` and reviewed the deployed contracts and verified code on etherscan. All found issues are resolved. A recommendation is also advised to the team to ensure the best practices are applied.

Disclaimer

While best efforts and precautions have been taken in the preparation of this document, The Arcadia Group and the Authors assume no responsibility for errors, omissions, or damages resulting from the use of the provided information. Additionally, Arcadia would like to emphasize that the use of Arcadia's services does not guarantee the security of a smart contract or set of smart contracts and does not guarantee against attacks. One audit on its own is not enough for a project to be considered secure; that categorization can only be earned through extensive peer review and battle testing over an extended period.