DECENTER

Audit report for BetDex

DECENTER

1. Summary

BetDex has engaged Decenter in the period starting on September 20th and ending on September 25th 2018 to assess and audit their platform's Solidity smart contracts. This document describes the issues discovered during the audit. Decenter's assessment is focused primarily on code review of the contract, with an emphasis on security, gas usage optimization and overall code quality.

2. Audit

2.1 Authenticity

The audited contracts can be found in the Gitlab repository: https://gitlab.com/bet-dex/smart-contracts; the version used has commit hash *330eda5988e2bb5eba0b4d27515d6e014da0345a*.

2.2 Scope

This audit covered only the delivered *.sol files mentioned in the previous section.

2.3 Legend

🔴High priority issues
🟠Medium priority issues
🟡Minor issues and optimisations
🟢Notes and recommendations

3. Issues Found

1. Contract owner can manipulate and change betting start and end times

File name: BetDex.sol
(lines 114-120): updateEventStartsTime() function
(lines 122-128): updateBettingOpensTime() function

The contract owner has the power to open and close betting for a certain event at any time and as many times as they want in the current revision of the code. This leaves room for potential manipulation and can be seen by the community as a backdoor for abuse of the contract.

We recommend removing the function that allows the contract owner to change the betting start time and modify the function that allows them to change the betting end time so that it only allows changing to a later point in time.

2. The totalReward property in Event structure is redundant and increases gas usage

File name: BetDex.sol
(lines 12-31): Event structure

This is a redundant property as its value can be calculated and equals the sum of the totalBet values of both scenarios.

Removing this property and calculating the value instead would reduce gas usage.

3. The totalBettors property in Event structure is redundant and increases gas usage

File name: BetDex.sol
(lines 12-31): Event structure

This property is redundant and is only used to be incremented by 1 each time a bettor who hasn't previously participated in the bet joins.

Since it's not used anywhere, removing this function would reduce gas usage.

4. The winnerPoolTotal property in Event structure is redundant and increases gas usage

File name: BetDex.sol
(lines 12-31): Event structure

This is a redundant property as its value equals the total bet value of the winning scenario.

Removing this property would reduce gas usage.

5. The houseFeePaid property in Event structure is redundant and increases gas usage

File name: BetDex.sol
(lines 12-31): Event structure

This is a redundant property, as it serves no function to the contract code. It is only used once, after event ends, where it's simply set to true.

Since we know that the house fee is paid every time an event ends, except in the cases when the contract owner calls the event a tie or cancels the event, the property is unnecessary and removing it would reduce gas usage.

6. The scenarioId property in Scenario structure is redundant and increases gas usage

File name: BetDex.sol
(lines 33-37): Scenario structure

This property is redundant and serves no function to the contract code.

Removing the property would reduce gas usage.

7. The totalBet property in betterInfo structure is redundant and increases gas usage

File name: BetDex.sol
(lines 39-44): betterInfo structure

The totalBet property is unnecessarily introduced as it can be calculated by summary of the values of two betting options.

Removing the property and simply calculating it this way instead would reduce gas usage.

9. The Div() function from SafeMath library is redundant and increases gas usage

File name: BetDex.sol
(lines 130-159): setWinnerAndEndEvent() function
(lines 198-203): calculateWinnings() function

The div() function in these cases is redundant and increases gas usage for no reason.

Replacing the function call with a simple division operation would reduce gas usage.

10. The eventHasEnded property is set to true redundantly the second time

File name: BetDex.sol
(lines 130-159): setWinnerAndEndEvent() function

Setting the eventHasEnded value to true at line 145 is redundant and unnecessary as the variable was already set to true earlier at line 138.

Removing the redundant code at line 145 would make the code cleaner.

11. The if check at line 152 is redundant

File name: BetDex.sol
(lines 130-159): setWinnerAndEndEvent() function

The if check at line 152 is redundant as the house fee is automatically paid out every time an event ends, making this check unnecessary.

Removing this if check would reduce processing resources needed.

12. The claimWinnings function should only run for those who bet on the winning scenario

File name: BetDex.sol
(lines 170-184): claimWinnings() function

Running the claimWinnings function for all bettors instead only for those who bet on the winning scenario is wasting processing resources unnecessarily.

14. winningScenarioId property should be renamed to winningScenarioName

File name: BetDex.sol
(lines 12-31): Event structure

We recommend renaming the winningScenarioId variable to winningScenarioName for a clearer naming scheme, because its value will indeed be taken from and equal to one of the scenario names.

15. bettorInfo structure should be renamed to BettorInfo

File name: BetDex.sol
(lines 39-44): betterInfo Structure

We recommend changing the bettorInfo structure name to the capitalized variant BettorInfo for naming convention reasons.

4. Audit

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of business model, or any other statements about fitness of the contracts to purpose or their bugfree status. Securing smart contracts is a multistep process, therefore running a bug bounty program as a complement to this audit is strongly recommended.

5. Closing Summary

It is the conclusion of this review that the codebase of the platform is well organized and sufficiently modular. In general the smart contract code adapts all the relevant best practices and has clean, legible code which will be further improved with the mentioned recommended updates. There are currently many redundant elements in the code and removing them will optimize its execution and greatly reduce gas usage. Additionally, the documentation can also be improved slightly. Aside for the remarks in this audit, the security of the contracts is sufficient given the assumed requirements. Main issue found by the audit is item marked with medium priority.