JustBet Audit Report

Apr 26, 2023

Table of Contents

Summary

This report has been prepared for JustBet Audit Report smart contract, to discover issues and vulnerabilities in the source code of their Smart Contract as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

Overview

Project Summary

Project Name	**JustBet Audit Report**
Codebase	**https://github.com/WINRLabs/justbet-contracts**
Commit	**76d8b877d862dfdcfed10bf55ee118d3be391fbd**
Language	**Solidity**

Audit Summary

Delivery Date	**Apr 26, 2023**
Audit Methodology	**Static Analysis, Manual Review**
Total Isssues	**18**

[WP-H2] The existence of `scheduledRequestBufferSecs` allows the games to be fulfilled before the finish time

High

Issue Description

In the current implementation, there is a 2 seconds buffer time which allows anyone to trigger the randomness request BEFORE the game ends.

This creates an opportunity for attackers to place bets while the game is already finished, but before the finish time.

According to EVM Chains Average Block Time Compare, the average block time of Arbitrum is 0.363s.

That means there can be a room of 5 blocks for the fulfill transaction to arrive BEFORE the game ends.

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Wheel.sol#L40-L49

```
40    modifier whenNotClosed() {
41      _createGame();
42
43      uint256 spinStartTime = games[currentGameId].startTime + config.duration;
44      uint256 spinFinishTime = spinStartTime + config.cooldown;
45
46      require(block.timestamp < spinStartTime, "WHE: Game closed");
47
48      _;
49    }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Wheel.sol#L282-L307

```
282   function _createGame() internal {
283     uint256 finishTime = games[currentGameId].startTime + config.duration +
    config.cooldown;
284
285     /// @notice if the last game has finished
286     if (block.timestamp > finishTime) {
287       currentGameId++;
288
289       /// @notice schedules random request for game for after wagering duration
290       uint256 startTime_ = block.timestamp;
291       uint256 requestId_ = _requestScheduledRandom(1, startTime_ +
    config.duration);
292       address[] memory currencies_;
293
294       requestGamePair[requestId_] = currentGameId;
295
296       games[currentGameId] = Game(
297         currentGameId,
298         requestId_,
299         startTime_,
300         currencies_,
301         Color.IDLE,
302         Status.STARTED
303       );
304
305       emit Created(currentGameId, startTime_);
306     }
307   }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/randomizer/Router.sol#L137-L153

```
137   /// @notice requests scheduled random number(s)
138   /// @param _count random number count
139   /// @param _targetTime target time of rng response
140   function scheduledRequest(uint8 _count, uint256 _targetTime) external
    onlyConsumer returns (uint256 requestId_) {
141     IRandomizerProvider provider_ = getDefaultProvider();
142
143     requestId_ = _addRequest(provider_, _count, _targetTime, 0);
144
```

```
145       emit ScheduledRequestCreated(
146         requestId_,
147         _msgSender(),
148         address(provider_),
149         _count,
150         _targetTime,
151         _targetTime - scheduledRequestBufferSecs
152       );
153     }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/randomizer/Router.sol#L155-L164

```
155     /// @notice triggers scheduled requests
156     /// @param _requestId of scheduled request
157     function trigger(
158       uint256 _requestId
159     ) external onlyNotFilled(_requestId) onlyScheduled(_requestId)
    onTargetTime(_requestId) {
160       Request memory request_ = requests[_requestId];
161       IRandomizerProvider(request_.provider).request(_requestId, request_.count, 0);
162
163       emit Triggered(_requestId);
164     }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/randomizer/Router.sol#L58-L61

```
58     modifier onTargetTime(uint256 _requestId) {
59       require(block.timestamp >= requests[_requestId].targetTime -
    scheduledRequestBufferSecs, "RND: Not time");
60       _;
61     }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/randomizer/Router.sol#L77

```
77     uint8 public scheduledRequestBufferSecs = 2;
```

PoC

Given:

- config.duration: 1 hour
- config.cooldown: 5 min

When:

1. **14:00:00.0** - A new round of game (id: 11) started. At `Wheel.sol` L291 `_requestScheduledRandom(1, 15:00:00 = startTime_ + config.duration)`
   - `RandomizerRouter.scheduledRequest(1, 15:00:00)` scheduled an request (id: 1011) with a `targetTime` of 15:00:00;
2. **14:59:58.0** - The attacker called `RandomizerRouter.trigger(_requestId: 1011)`
   - L59 `require(block.timestamp >= requests[_requestId].targetTime - scheduledRequestBufferSecs, "RND: Not time")` <==> `require(14:59:58 >= 15:00:00 - 2, "RND: Not time")` passed;
   - L161 `IRandomizerProvider(request_.provider).request(_requestId: 1011, _count: 1, minConfirmations: 0)`
3. **14:59:58.6** - The contract received the callback of the randomness provider;
   - Wheel L275 `games[11].color = color_ = Color.RED`
4. **14:59:58.6** - The attacker sent a `Wheel.bet(_wager: 100_000e18, _color: Color.RED)` transaction and minted at **14:59:59.6**;
   - L43 `uint256 spinStartTime = games[currentGameId].startTime + config.duration = 14:00:00 + 1 hour = 15:00:00`
   - L46 `require(block.timestamp < spinStartTime, "WHE: Game closed")` <==> `require(14:59:59 < 15:00:00, "WHE: Game closed")` passed;
   - L362 attacker successfully placed a Bet: `participants[11][attacker] = Bet(color: Color.RED, amount: 100_000e18, _tokens);`

Recommendation

The `randomizerFulfill()` transaction should NEVER been called before the game ends.

For example, this can be achieved by removing `RandomizerRouter.scheduledRequestBufferSecs` and/or adding the following requirements in `Wheel.randomizerFulfill()` :

```
require(block.timestamp >= game.startTime + duration, "...");
require(randomGeneratedTimestamp >= game.startTime + duration, "...");
```

As a side note on a more general issue:
It is critical to ensure that all games always guarantee that no bets can be placed once the `IRandomizerProvider.request()` has been sent.

Status

✓ Fixed

[WP-M3] DOS by sending a large number of small bets to deplete the protocol's balance

Medium

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/CoinFlip.sol#L114-L130

```
114     function bet(
115       uint256 _wager,
116       uint8 _count,
117       uint256 _stopGain,
118       uint256 _stopLoss,
119       bytes memory _gameData,
120       address[2] memory _tokens
121     ) external isChoiceInsideLimits(_gameData) {
122       _create(
123         _wager,
124         _count,
125         _stopGain,
126         _stopLoss,
127         _gameData,
128         _tokens
129       );
130     }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Common.sol#L208-L238

```
208     function _create(
209       uint256 _wager,
210       uint8 _count,
211       uint256 _stopGain,
212       uint256 _stopLoss,
213       bytes memory _gameData,
214       address[2] memory _tokens
215     ) internal isGameCountAcceptable(_count) isWagerAcceptable(_wager) whenNotPaused
    nonReentrant {
```

```
216     address player_ = _msgSender();
217     uint256 requestId_ = _requestRandom(_count);
@@ 218,237 @@
238   }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/core/Core.sol#L18-L21

```
18    modifier isWagerAcceptable(uint256 _wager) {
19      require(_wager <= vaultManager.getMaxWager(), "GAME: Wager too high");
20      _;
21    }
```

The current implementation allows users to place bets with no minimum bet amount requirement.

However, each settlement of the bet requires the protocol to pay for a certain amount of fee (gas fee and/or potentially additional cost for the randomness provider's service).

As a result, an attacker can initiate a huge amount of `bet()` transactions with only 1 wei each as the `wager`, to consume all the balance on the protocol's settlement wallet, and effectively prevent future settlements for other users.

Recommendation

Consider setting a minimum `_wager` requirement in order to cover the cost of `_requestRandom(_count)`:

```
18    modifier isWagerAcceptable(uint256 _wager) {
19      require(_wager >= vaultManager.getMinWager(), "GAME: Wager too low");
20      require(_wager <= vaultManager.getMaxWager(), "GAME: Wager too high");
21      _;
22    }
```

Status

✓ Fixed

[WP-L4] `VaultManager.sol#payback()` should not require `onlyWhitelistedToken`

Low

Issue Description

If the token was once a `whitelistedToken` and some users have already bet with it, and the admin chooses to `unsetWhitelistedToken()`, these users will not be able to retrieve their funds because `payback()` requires the token to be whitelisted.

The expected behavior is that when a token is no longer whitelisted, it can still pay out but cannot be accepted for bets.

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/core/VaultManagerSettings.sol#L127-L131

```
127  function unsetWhitelistedToken(address _token) external onlyGovernance {
128      delete whitelistedTokens[_token];
129
130      emit TokenRemoved(_token);
131  }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/core/VaultManagerSettings.sol#L24-L27

```
24  modifier onlyWhitelistedToken(address _token) {
25      require(whitelistedTokens[_token], "VM: unknown token");
26      _;
27  }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/core/VaultManager.sol#L85-L87

```
85    function transferOut(address _token, address _recipient, uint256 _amount) public
      onlyGame onlyWhitelistedToken(_token) {
86        require(IERC20(_token).transfer(_recipient, _amount), "VM: Transfer out
      error");
87    }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/core/VaultManager.sol#L46-L51

```
46  function payback(address _token, address _recipient, uint256 _amount) public
    onlyGame onlyWhitelistedToken(_token) {
47      _decreaseEscrow(_token, _amount);
48      transferOut(_token, _recipient, _amount);
49
50      emit Payback(_recipient, _token, _amount);
51  }
```

Recommendation

Consider removing the `onlyWhitelistedToken(_token)` modifier for `payback()` :

```
46  function payback(address _token, address _recipient, uint256 _amount) public
    onlyGame {
47      _decreaseEscrow(_token, _amount);
48      transferOut(_token, _recipient, _amount);
49
50      emit Payback(_recipient, _token, _amount);
51  }
```

Status

✓ Fixed

[WP-L5] `_tokens[1]` was not required to be whitelisted in `bet()`, but required when paying out

Low

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Common.sol#L208-L238

```
208   function _create(
209       uint256 _wager,
210       uint8 _count,
211       uint256 _stopGain,
212       uint256 _stopLoss,
213       bytes memory _gameData,
214       address[2] memory _tokens
215     ) internal isGameCountAcceptable(_count) isWagerAcceptable(_wager) whenNotPaused
    nonReentrant {
216       address player_ = _msgSender();
217       uint256 requestId_ = _requestRandom(_count);
218
219       games[gameId] = Game(
220         requestId_,
221         player_,
222         _wager,
223         _count,
224         _stopGain,
225         _stopLoss,
226         _gameData,
227         _tokens,
228         false
229       );
230
231       requestGamePair[requestId_] = gameId;
232       gameId++;
233
234       /// @notice escrows total wager to Vault Manager
235       vaultManager.escrow(_tokens[0], player_, _count * _wager);
236
237       emit Created(player_, requestId_, _wager);
```

```
238     }
```

In `bet()` , only `_tokens[0]` is required to be whitelisted in `vaultManager.escrow()` .

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/core/VaultManager.sol#L64-L66

```
64  function payout(address[2] memory _tokens, address _recipient, uint256
    _escrowAmount, uint256 _totalAmount) public onlyGame
    onlyWhitelistedTokens(_tokens) {
65      vault.payout(_tokens, address(this), _escrowAmount, _recipient, _totalAmount);
66    }
```

However, in `payout()` , both `_tokens[0]` and `_tokens[1]` are required to be whitelisted.

Recommendation

See the **Recommendation** of [WP-L4].

Status

✓ Fixed

[WP-L7] Plinko: Lack of sanity check for the length of `_multiplier`

Low

Issue Description

The length of `_multiplier` MUST be greater than or equal to `index_ + 1` .

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Plinko.sol#L54-L61

```
54  /// @notice updates row multipliers
55    /// @param _index row count
56    /// @param _multipliers of row. generally 1 more or same size
57    function updateMultipliers(uint8 _index, uint256[] memory _multipliers) external
    onlyGovernance {
58      multipliers[_index] = _multipliers;
59    }
```

The elements of the `_multipliers` array are used by the `calcReward()` function in the `play()` function.

In each iteration of the inner for loop, consider the following extreme cases:

- `index_` is increased by `rows_` times. Since the initial value of `index_` is `rows_` , then at this point in line 143, `index_ = 2 * rows_` , `index_/2 = rows_` .
- `index_` is decreased by `rows_` times. Since the initial value of `index_` is `rows_` , then at this point in line 143, `index_ = 0` , `index_/2 = 0` .

Therefore, for the `calcReward` function, the range of the second parameter is `[0, rows_]` .

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Plinko.sol#L113-L151

```
113  function play(
```

```
@@ 114,127 @@

    for (uint32 i = 0; i < _game.count; i ++) {
      index_ = rows_;

      /// @notice calculates the final index by moving the ball movements sequentially on the index
      for (uint32 s = 0; s < rows_; s ++) {
        randomIndex_ = (i * rows_) + s;

        if (_resultNumbers[randomIndex_] == 0) {
          index_--;
        } else {
          index_++;
        }
      }

      payouts_[i] = calcReward(rows_, index_ / 2, _game.wager);
@@ 144,149 @@
    }
  }
```

Due to the fact that the second parameter, `_index`, of the `calcReward()` function is the array index of corresponding item in the `_rows`, and since the range of `_index` is `[0, _rows]`, it is necessary to ensure that the array does not go out of bounds when using `getMultiplier()`.

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Plinko.sol#L87-L89

```
  function calcReward(uint8 _rows, uint32 _index, uint256 _wager) public view returns (uint256 reward_) {
    reward_ = (_wager * getMultiplier(_rows, _index)) / PRECISION;
  }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Plinko.sol#L76-L81

```
76    /// @notice returns multiplier according to row's index
77    /// @param _rows row count
78    /// @param _index multiplier index
79    function getMultiplier(uint8 _rows, uint32 _index) public view returns (uint256
    multiplier_) {
80        multiplier_ = multipliers[_rows][_index];
81    }
```

Recommendation

Consider adding a check when updating the array:

```
54  /// @notice updates row multipliers
55  /// @param _index row count
56  /// @param _multipliers of row. generally 1 more or same size
57  function updateMultipliers(uint8 _index, uint256[] memory _multipliers) external
    onlyGovernance {
58      require(_multipliers.length > _index, "insufficient _multipliers length");
59      multipliers[_index] = _multipliers;
60  }
```

Status

✓ Fixed

[WP-L8] Lack of sanity check for the multiplier

Low

Issue Description

There is a lack of a sanity check for the multiplier, which can result in reverts when the user wins the game.

Affected games:

- CoinFlip
- Dice
- Range

When a win occurs, the reward is calculated based on the Multiplier. The Multiplier should always be greater than or equal to 1e18. Otherwise, the payout will revert due to underflow `reward_ - wager` .

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Dice.sol#L52-L57

```
52    /// @notice updates win multiplier
53    /// @param _sideCount multiplier changes according to side count
54    /// @param _multiplier multiplier
55    function updateWinMultiplier(uint16 _sideCount, uint256 _multiplier) external
    onlyGovernance {
56      multipliers[_sideCount] = _multiplier;
57    }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Dice.sol#L103-L129

```
103   function play(
104     Game memory _game,
105     uint256[] memory _resultNumbers
106   ) public view override returns (
107     uint256 payout_,
108     uint32 playedGameCount_,
```

```
109       uint256[] memory payouts_
110     ) {
111       payouts_ = new uint[](_game.count);
112       playedGameCount_ = _game.count;
113
114       uint8[] memory choices_ = decodeGameData(_game.gameData);
115       uint256 reward_ = calcReward(choices_.length, _game.wager);
116
117       for (uint8 i = 0; i < _game.count; i++) {
118         if (isWon(choices_, _resultNumbers[i])) {
119           int _payout = int256(reward_) - int256(_game.wager);
120           payouts_[i] = uint256(abs(_payout));
121           payout_ += reward_;
122         }
123
124         if (shouldStop(payout_, (i + 1) * _game.wager, _game.stopGain,
    _game.stopLoss)) {
125           playedGameCount_ = i + 1;
126           break;
127         }
128       }
129     }
```

Recommendation

```
52     /// @notice updates win multiplier
53     /// @param _sideCount multiplier changes according to side count
54     /// @param _multiplier multiplier
55     function updateWinMultiplier(uint16 _sideCount, uint256 _multiplier) external
    onlyGovernance {
56       require(_multiplier >= 1e18, "_multiplier should be greater than or equal to
    1e18")
57       multipliers[_sideCount] = _multiplier;
58     }
```

Status

✓ Fixed

[WP-L9] Lack of sanity check for the `units` , `angle_` and `unitHeight`

Low

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Wheel.sol#L217-L224

```
217     /// @notice calculates reward according to winning multiplier
218     /// @param _random raw random number
219     function getUnit(uint256 _random) public view returns (uint32 angle_, uint32
    index_) {
220         Configuration memory config_ = config;
221 
222         angle_ = uint32(_random % config_.range);
223         index_ = (angle_ - (angle_ % config_.unitHeight)) / config_.unitHeight;
224     }
```

There is a chance that if the `angle_` and `unitHeight` are not set properly, then the result `index` of `getUnit()` may be out-of-bound for the storage variable `units` .

As a result, the game will not be able to be fulfilled.

Status

✓ Fixed

[WP-L10] Plinko: Lack of existence check for the multiplier of the corresponding rows

Low

Issue Description

In Plinko, there is a lack of existence check for the multiplier of the corresponding rows. This means that there is no validation in place to ensure that a multiplier exists for each row.

This issue can lead to errors in the game, such as incorrect payouts or revert when a player lands on a row with no multiplier.

To prevent these errors and ensure fair gameplay, it is important to implement a validation process that checks for the existence of a multiplier for each row in Plinko.

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Plinko.sol#L54-L66

```
54    /// @notice updates row multipliers
55    /// @param _index row count
56    /// @param _multipliers of row. generally 1 more or same size
57    function updateMultipliers(uint8 _index, uint256[] memory _multipliers) external
    onlyGovernance {
58      multipliers[_index] = _multipliers;
59    }
60
61    /// @notice updates row selection limits
62    /// @param _min minimum selectable row
63    /// @param _max maxiumum selectable row
64    function updateRowLimits(uint32 _min, uint32 _max) external onlyGovernance {
65      rowLimits = RowLimits(_min, _max);
66    }
```

Recommendation

```
64  function updateRowLimits(uint32 _min, uint32 _max) external onlyGovernance {
65      for (int i = _min; i <= _max; i++) {
```

```
66          require(multipliers[i].length > 0, "multipliers in row " + i + " not
    exist");
67      }
68      rowLimits = RowLimits(_min, _max);
69    }
```

Status

✓ Fixed

[WP-L11] Moon: Lack of sanity check in `updateConfig()` to prevent div by 0

Low

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Moon.sol#L196-L204

```
196   function getResult(uint256 _random) public view returns (uint256 multiplier_) {
197     uint256 H = modNumber(_random, (config.maxMultiplier - config.minMultiplier +
    1));
198     uint256 E = config.maxMultiplier / 100;
199     multiplier_ = (E * config.maxMultiplier - H) / (E * 100 - H);
200
201     if (modNumber(_random, 66) == 0) {
202       multiplier_ = 1;
203     }
204   }
```

The initial config will not result in a denominator of zero, but if `updateConfig()` is called to make modifications of the configs to certain values, it may result in a denominator of zero.

For example:

- max = 9900
- min = 90
- H's range is [0, 9901)
- If `_random` = 9900, then H = 9900
- E = 9990 / 100 = 99
- E * 100 - H = 0

Hence, it is recommended to add a check during the setup or calculation phase to prevent this from happening.

https://github.com/WINRLabs/justbet-contracts/blob/

1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Moon.sol#L115

```
115     Configuration public config = Configuration(100, 10000, 20, 30);
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Moon.sol#L128-L130

```
128     function updateConfig(Configuration memory _config) external onlyGovernance {
129       config = _config;
130     }
```

Status

ⓘ Acknowledged

[WP-L12] Changing the game settings will impact the ongoing games

Low

Issue Description

For certain games, there are settings or parameters that can be changed by the `governance`. For example, `Wheel.sol#updateUnits()` can change the `units` of the Wheel game.

We believe that once the game contract is published, calling `updateUnits()` will certainly have an impact on the game that was started earlier and has not yet been fulfilled.

For example, if the governance decides to change the `units` from `5 Black and 5 Red` to `8 Black and 2 Red`, then all current ongoing game participants will suddenly have a much lower chance of getting Red as the result.

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Wheel.sol#L200-L202

```
200     function updateUnits(Color[] memory _units) external onlyGovernance {
201       units = _units;
202     }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Wheel.sol#L194-L196

```
194     function updateColorMultiplier(Color _color, uint256 _multiplier) public
      onlyGovernance {
195       multipliers[_color] = _multiplier;
196     }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/

contracts/games/multiplayer/Wheel.sol#L352-L367

```
352     function bet(
353       uint256 _wager,
354       Color _color,
355       address[2] memory _tokens
356     ) external nonReentrant isWagerAcceptable(_wager) whenNotPaused whenNotClosed {
357       address player_ = _msgSender();
358       require(participants[currentGameId][player_].amount == 0, "Bet cannot
    change");
359       Game memory game_ = games[currentGameId];
360
361       /// @notice sets players bet to the list
362       participants[game_.id][player_] = Bet(_color, _wager, _tokens);
363
364       _escrow(player_, _wager, _color, _tokens);
365
366       emit Participated(game_.id, player_, _wager, _color);
367     }
```

Recommendation

To ensure that the player receives the same terms at the time of settling the game as when they placed the bet, consider binding a fixed `units` and `multipliers` state to `bet()`, such as adding a snapshot of `units` and `multipliers` to `games[currentGameId]`.

See also the **Recommendation** of [WP-I13].

Status

✓ **Fixed**

[WP-I13] `currentGameId` should be specificed as a param in the bet function

Informational

Issue Description

> This issue is related to [WP-L12] as both issues addressed the problem of insufficient control for the change of terms and conditions of the transaction.

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Wheel.sol#L282-L307

```
282   function _createGame() internal {
283     uint256 finishTime = games[currentGameId].startTime + config.duration +
    config.cooldown;
284
285     /// @notice if the last game has finished
286     if (block.timestamp > finishTime) {
287       currentGameId++;
288
289       /// @notice schedules random request for game for after wagering duration
290       uint256 startTime_ = block.timestamp;
291       uint256 requestId_ = _requestScheduledRandom(1, startTime_ +
    config.duration);
292       address[] memory currencies_;
293
294       requestGamePair[requestId_] = currentGameId;
295
296       games[currentGameId] = Game(
297         currentGameId,
298         requestId_,
299         startTime_,
300         currencies_,
301         Color.IDLE,
302         Status.STARTED
303       );
304
```

```
305         emit Created(currentGameId, startTime_);
306       }
307     }
```

It may not be the player's intention to bet in another round (i.e., `currentGameId`) than the one they anticipated joining when the `bet()` transaction was sent.

Status

ⓘ Acknowledged

[WP-G15] Removing unnecessary code can save gas

Gas

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Dice.sol#L83-L95

```
83    /// @notice makes the decision about choices
84    /// @param _choices selected side's by player
85    /// @param _result modded random number
86    function isWon(uint8[] memory _choices, uint256 _result) public pure returns
    (bool result_) {
87      for (uint8 i = 0; i < _choices.length; i++) {
88        if (_choices[i] == _result) {
89          result_ = true;
90          break;
91        } else {
92          result_ = false;
93        }
94      }
95    }
```

There is no need to set `result_ = false;` in the else branche as that's the default value.

Recommendation

```
83    /// @notice makes the decision about choices
84    /// @param _choices selected side's by player
85    /// @param _result modded random number
86    function isWon(uint8[] memory _choices, uint256 _result) public pure returns
    (bool result_) {
87      for (uint8 i = 0; i < _choices.length; i++) {
88        if (_choices[i] == _result) {
89          result_ = true;
90          break;
91        } // else result_ is false(default)
92      }
```

```
93     }
```

Status

✓ Fixed

[WP-G16] RPS: Using `else if` can save some gas

Gas

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/RPS.sol#L79-L98

```
79    /// @notice makes the decision about choice
80    /// @param _choice players choice 0 or 1
81    /// @param _result modded random number
82    function isWon(uint16 _choice, uint256 _result) public pure returns (Result
    result_) {
83      if (_choice == 0) {
84        if (_result == 0) result_ = Result.DRAW;
85        if (_result == 1) result_ = Result.LOSE;
86        if (_result == 2) result_ = Result.WIN;
87      }
88      if (_choice == 1) {
89        if (_result == 0) result_ = Result.WIN;
90        if (_result == 1) result_ = Result.DRAW;
91        if (_result == 2) result_ = Result.LOSE;
92      }
93      if (_choice == 2) {
94        if (_result == 0) result_ = Result.LOSE;
95        if (_result == 1) result_ = Result.WIN;
96        if (_result == 2) result_ = Result.DRAW;
97      }
98    }
```

Recommendation

```
79    /// @notice makes the decision about choice
80    /// @param _choice players choice 0 or 1
81    /// @param _result modded random number
82    function isWon(uint16 _choice, uint256 _result) public pure returns (Result
    result_) {
83      if (_choice == 0) {
```

```
84        if (_result == 0) result_ = Result.DRAW;
85        else if (_result == 1) result_ = Result.LOSE;
86        else if (_result == 2) result_ = Result.WIN;
87      }
88      else if (_choice == 1) {
89        if (_result == 0) result_ = Result.WIN;
90        else if (_result == 1) result_ = Result.DRAW;
91        else if (_result == 2) result_ = Result.LOSE;
92      }
93      else if (_choice == 2) {
94        if (_result == 0) result_ = Result.LOSE;
95        else if (_result == 1) result_ = Result.WIN;
96        else if (_result == 2) result_ = Result.DRAW;
97      }
98    }
```

Status

✓ Fixed

[WP-G17] Moon: When all the players win, `payin()` can be skipped

Gas

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Moon.sol#L210-L228

```
210     function randomizerFulfill(uint256 _requestId, uint256[] calldata _randoms)
    override internal {
211       uint256 gameId_ = requestGamePair[_requestId];
212       Game memory game_ = games[gameId_];
213
214       game_.multiplier = getResult(_randoms[0]);
215       game_.status = Status.FINISHED;
216
217       uint256[] memory wonAmounts_ = collectWonAmounts(game_);
218
219       for (uint256 i = 0; i < wonAmounts_.length; i++) {
220         address token_ = game_.currencies[i];
221         vaultManager.payin(token_, totalAmounts[gameId_][token_] - wonAmounts_[i]);
222         totalAmounts[gameId_][token_] = wonAmounts_[i];
223       }
224
225       games[gameId_] = game_;
226
227       emit Settled(gameId_, game_.multiplier);
228     }
```

Recommendation

```
210     function randomizerFulfill(uint256 _requestId, uint256[] calldata _randoms)
    override internal {
211       uint256 gameId_ = requestGamePair[_requestId];
212       Game memory game_ = games[gameId_];
```

```
213
214       game_.multiplier = getResult(_randoms[0]);
215       game_.status = Status.FINISHED;
216
217       uint256[] memory wonAmounts_ = collectWonAmounts(game_);
218
219       for (uint256 i = 0; i < wonAmounts_.length; i++) {
220         address token_ = game_.currencies[i];
221         uint256 totalLoss = totalAmounts[gameId_][token_] - wonAmounts_[i];
222         if (totalLoss > 0) {
223           vaultManager.payin(token_, totalLoss);
224         }
225         totalAmounts[gameId_][token_] = wonAmounts_[i];
226       }
227
228       games[gameId_] = game_;
229
230       emit Settled(gameId_, game_.multiplier);
231     }
```

Status

✓ Fixed

[WP-G19] `Wheel.bet()` Removing unnecessary SLOAD of game can simply the code and save gas

Gas

Issue Description

Only the `id` field of `game_` was used. Therefore, using `currentGameId` directly can save gas.

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Wheel.sol#L352-L367

```
352   function bet(
353     uint256 _wager,
354     Color _color,
355     address[2] memory _tokens
356   ) external nonReentrant isWagerAcceptable(_wager) whenNotPaused whenNotClosed {
357     address player_ = _msgSender();
358     require(participants[currentGameId][player_].amount == 0, "Bet cannot
    change");
359     Game memory game_ = games[currentGameId];
360
361     /// @notice sets players bet to the list
362     participants[game_.id][player_] = Bet(_color, _wager, _tokens);
363
364     _escrow(player_, _wager, _color, _tokens);
365
366     emit Participated(game_.id, player_, _wager, _color);
367   }
```

Recommendation

Consider changing to:

```
352   function bet(
353     uint256 _wager,
354     Color _color,
355     address[2] memory _tokens
```

```
356   ) external nonReentrant isWagerAcceptable(_wager) whenNotPaused whenNotClosed {
357     address player_ = _msgSender();
358     uint256 currentGameId_ = currentGameId;
359     require(participants[currentGameId_][player_].amount == 0, "Bet cannot
      change");
360
361     /// @notice sets players bet to the list
362     participants[currentGameId_][player_] = Bet(_color, _wager, _tokens);
363
364     _escrow(player_, _wager, _color, _tokens);
365
366     emit Participated(currentGameId_, player_, _wager, _color);
367   }
```

Status

✓ Fixed

[WP-N20] ERC20 tokens with no bool return value are not supported

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/core/VaultManager.sol#L77-L79

```
77  function transferIn(address _token, address _sender, uint256 _amount) public
    onlyGame onlyWhitelistedToken(_token) {
78      require(IERC20(_token).transferFrom(_sender, address(this), _amount), "VM:
    Transfer in error");
79  }
```

There are some ERC20 tokens do not return a bool (e.g. USDT, BNB, OMG) on ERC20 methods. see here for a comprehensive (if somewhat outdated) list.

Recommendation

Consider using `SafeERC20` if the system intends to support these tokens.

Status

✓ Fixed

[WP-N21] Dice: Consider requiring `_sideCount` to be within bounds (`[1, 5]`) in `updateWinMultiplier()`

Issue Description

Setting the multiplier for a `_sideCount` out of the `[1,5]` bound is meaningless, as the choices_.length can only be <= 5.

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Dice.sol#L29-L35

```
29    constructor(IRandomizerRouter _router) CommonSolo(_router) {
30      multipliers[1] = 588e16;
31      multipliers[2] = 294e16;
32      multipliers[3] = 196e16;
33      multipliers[4] = 147e16;
34      multipliers[5] = 1176e15;
35    }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Dice.sol#L55-L57

```
55    function updateWinMultiplier(uint16 _sideCount, uint256 _multiplier) external
    onlyGovernance {
56      multipliers[_sideCount] = _multiplier;
57    }
```

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Dice.sol#L62-L64

```
62    function calcReward(uint256 _sideCount, uint256 _wager) public view returns
    (uint256 reward_) {
63      reward_ = (_wager * multipliers[_sideCount]) / PRECISION;
64    }
```

https://github.com/WINRLabs/justbet-contracts/blob/

1e9869726faf296724b927d930823434012a1558/contracts/games/solo/Dice.sol#L103-L130

```
103   function play(
104     Game memory _game,
105     uint256[] memory _resultNumbers
106   ) public view override returns (
107     uint256 payout_,
108     uint32 playedGameCount_,
109     uint256[] memory payouts_
110   ) {
111     payouts_ = new uint[](_game.count);
112     playedGameCount_ = _game.count;
113 
114     uint8[] memory choices_ = decodeGameData(_game.gameData);
115     uint256 reward_ = calcReward(choices_.length, _game.wager);
116 
117     for (uint8 i = 0; i < _game.count; i++) {
118       if (isWon(choices_, _resultNumbers[i])) {
119         // uint _payout = reward_  > wager ?
120         int _payout = int256(reward_) - int256(_game.wager);
121         payouts_[i] = uint256(abs(_payout));
122         payout_ += reward_;
123       }
124 
125       if (shouldStop(payout_, (i + 1) * _game.wager, _game.stopGain,
    _game.stopLoss)) {
126         playedGameCount_ = i + 1;
127         break;
128       }
129     }
130   }
```

Status

✓ Fixed

[WP-N22] Unused code

Issue Description

https://github.com/WINRLabs/justbet-contracts/blob/1e9869726faf296724b927d930823434012a1558/contracts/games/multiplayer/Moon.sol#L38-L47

```
38    modifier whenNotClosed() {
39      _createGame();
40
41      uint256 spinStartTime = games[currentGameId].startTime + config.duration;
42      uint256 spinFinishTime = spinStartTime + config.cooldown;
43
44      require(block.timestamp < spinStartTime, "MOO: Game closed");
45
46      _;
47    }
```

`spinFinishTime` is never used.

Status

✓ Fixed

Appendix

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by WatchPug; however, WatchPug does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication.

Disclaimer

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Smart Contract technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security. No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.