WINR Protocol Audit Report

May 18, 2023

WATCHPUG

Table of Contents

Summary

This report has been prepared for WINR Protocol 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	**WINR Protocol**
Codebase	**https://github.com/WINRLabs/winr-protocol**
Commit	**c75556b0dc1b500ac1139ed60909e2349bfe878f**
Language	**Solidity**

Audit Summary

Delivery Date	**May 18, 2023**
Audit Methodology	**Static Analysis, Manual Review**
Total Isssues	**21**

[WP-C1] `cancel()` can be called with a vesting that has already been cancelled

Critical

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/stakings/WINR-staking/WINRVesting.sol#L268-L314

```
268  function cancel(uint256 _index) external {
269      // Get the address of the caller
270      address sender_ = msg.sender;
271      // Declare local variables for stake and bool values
272      StakeVesting memory stake;
273
274
275      // Retrieve the stake and bool values for the given index and staker
276      (stake) = getVestingStake(sender_, _index);
277
278      // Check if the stake has already been withdrawn
279      require(!stake.withdrawn, "stake has withdrawn");
280
281      // Remove the index from the staker's active stakes list
282      _removeActiveIndex(sender_, _index);
283
284      uint256 amount_ = stake.amount;
285
286      // Mark the stake as cancelled in the mapping
287      stakes[sender_][_index].cancelled = true;
288
289      // Calculate the amount of tokens to burn and the amount of tokens to send to
     the staker
290      uint256 burnAmount_ = _computeBurnAmount(amount_);
291      uint256 sendAmount_ = amount_ - burnAmount_;
292      totalStakedVestedWINR -= amount_;
293      totalWeight -= stake.weight;
294      // claim rewards
295      uint256 reward_ = _pendingWLPOfStake(stake);
296
```

```
297         if(reward_ > 0) {
298             tokenManager.sendWLP(sender_, reward_);
299 
300             stakes[sender_][_index].profitDebt += reward_;
301             totalProfit -= reward_;
302             totalClaimed[sender_] += reward_;
303 
304             emit ClaimVesting(sender_, reward_, _index);
305         }
306         // Send the staked tokens to the staker
307         tokenManager.sendVestedWINR(sender_, sendAmount_);
308 
309         // Burn the remaining vesting tokens
310         tokenManager.burnVestedWINR(burnAmount_);
311 
312         // Emit a Cancel event to notify listeners of the cancellation
313         emit Cancel(sender_, block.timestamp, _index, burnAmount_, sendAmount_);
314     }
```

The current implementation cannot prevent a user from cancelling a vesting stake that has already been cancelled before.

An attacker can repeatedly call `cancel()` with the same `_index` until the entire contract is emptied.

Recommendation

The `amount` of the stake should always be changed whenever the stake token is sent to the staker.

In this particular case, the stake can simply be deleted.

Status

✓ Fixed

[WP-M2] `parity` can be manipulated by frontrunning `updateAddresses()`

Medium

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/strategies/MiningStrategy.sol#L122-L139

```
122  function updateAddresses(
123      IWINR _WINR,
124      IVault _vault,
125      address _pool,
126      IERC20 _pairToken
127  ) public onlyGovernance {
128      require(address(_WINR) != address(0), "WINR address zero");
129      require(address(_vault) != address(0), "Vault zero");
130      require(_pool != address(0), "Pool zero");
131      require(address(_pairToken) != address(0), "Pair Token zero");
132      WINR = _WINR;
133      vault = _vault;
134      pool = _pool;
135      pairToken = _pairToken;
136      parity = getParity();
137
138      emit AddressesUpdated(_WINR, _vault);
139  }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/strategies/MiningStrategy.sol#L223-L225

```
223  function getParity() public view returns (uint256 _value) {
224      _value = (pairToken.balanceOf(pool) * PRECISION) / WINR.balanceOf(pool);
225  }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/strategies/MiningStrategy.sol#L292-L306

```
292  function calculate(
293      address _account,
294      uint256 _amount,
295      uint256 _mintedByGames
296  ) external returns (uint256 _mintAmount) {
297      if(accountMultipliers[_account] != 0) {
298          _mintAmount = _calculate(_amount, accountMultipliers[_account]);
299      } else {
300          _mintAmount = _calculate(_amount, _getMultiplier(_mintedByGames));
301      }
302  }
303
304  function _calculate(uint256 _amount, uint256 _multiplier) internal view
     returns(uint256) {
305      return (_amount * _multiplier * PRECISION / parity) / PRECISION;
306  }
```

Using the `balanceOf` of an AMM pool (e.g., Uniswap v2) can easily be manipulated by simply sending tokens to the pool and later calling `skim()` to retrieve them at almost zero cost.

If the intention is to update the `parity` from time to time, there should be a proper function to do it. This function should not use the `balanceOf` of an AMM pool, but instead should use an oracle or TWAP price.

Status

✓ Fixed

[WP-M3] `cancel()` should not send reward

Medium

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/stakings/WINR-staking/WINRVesting.sol#L261-L314

```
261     /**
262      * @dev This function cancels vesting stakes without penalty and reward.
263      * It sends the staked amount to the staker.
264      * @param _index index to cancel vesting for
265      * @notice Throws an error if the stake has already been withdrawn
266      * @notice Emits a Cancel event upon successful execution
267      */
268     function cancel(uint256 _index) external {
269         // Get the address of the caller
@@ 270,293 @@
294         // claim rewards
295         uint256 reward_ = _pendingWLPOfStake(stake);
296 
297         if(reward_ > 0) {
298             tokenManager.sendWLP(sender_, reward_);
299 
300             stakes[sender_][_index].profitDebt += reward_;
301             totalProfit -= reward_;
302             totalClaimed[sender_] += reward_;
303 
304             emit ClaimVesting(sender_, reward_, _index);
305         }
306         // Send the staked tokens to the staker
307         tokenManager.sendVestedWINR(sender_, sendAmount_);
308 
309         // Burn the remaining vesting tokens
310         tokenManager.burnVestedWINR(burnAmount_);
311 
312         // Emit a Cancel event to notify listeners of the cancellation
313         emit Cancel(sender_, block.timestamp, _index, burnAmount_, sendAmount_);
314     }
```

While the comment states that: **This function cancels vesting stakes without penalty and reward.**

The current implementation of `cancel()` still sends the reward (L295-305).

Recommendation

Remove L294-305.

Status

✓ **Fixed**

[WP-M4] `refundGame()` may refund a wrong amount of ReferralReward

Medium

Issue Description

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Wheel.sol#L345-L364

```
345   function bet(
346     uint256 _wager,
347     Color _color,
348     address[2] memory _tokens
349   ) external nonReentrant isWagerAcceptable(_tokens[0], _wager) whenNotPaused {
      @@ 350,357 @@
358     (uint256 _referralReward, uint256 _vWINRAmount) = _escrow(player_, _wager, _color, _tokens);
359
360     /// @notice sets players bet to the list
361     participants[currentGameId_][player_] = Bet(_color, _wager, _tokens, _referralReward, _vWINRAmount);
      @@ 362,363 @@
364   }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Wheel.sol#L319-L339

```
319   function _escrow(
320     address _player,
321     uint256 _wager,
322     Color _color,
323     address[2] memory _tokens
324   ) internal returns (uint256 referralReward_, uint256 vWINRAmount_) {
```

```
@@ 325,336 @@
337     /// @notice sets referral reward if player has referee
338     referralReward_ = vaultManager_.setReferralReward(_tokens[0], _player, _wager,
houseEdge);
339   }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/core/VaultManager.sol#L57-L67

```
57   function setReferralReward(
58     address _token,
59     address _player,
60     uint256 _amount,
61     uint64 _houseEdge
62   ) public onlyGame onlyWhitelistedToken(_token) returns (uint256 referralReward_)
{
63     if (_amount > 0) {
64       return referralStorage.setReward(_player, _token, ((_amount * _houseEdge) /
BASIS_POINTS));
65     }
66     return 0;
67   }
```

The value of the parameter `_amount` in `referralStorage.setReward()` is: _wager $\cdot \frac{houseEdge}{BASISPOINTS}$

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L402-L431

```
402     function setReward(address _player, address _token, uint256 _amount) external
onlyProtocol returns(uint256 _reward){
403         address referrer_ = returnPlayerRefferalAddress(_player);
404 
405         if (referrer_ != address(0)) {
406             // the player has a referrer
407             // calculate the rebate for the referrer tier
```

```
408             uint256 amountRebate_ = calculateRebate(
409                 _amount,
410                 tiers[referrerTiers[referrer_]].WLPRate
411             );
@@ 412,417 @@
418             // add the rebate to the rewards mapping of the referrer
419             unchecked {
420                 rewards[referrer_][_token] += amountRebate_;
421             }
@@ 422,427 @@
428             return amountRebate_;
429         }
430         emit NoRewardToSet(_player);
431     }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L294-L299

```
294     function calculateRebate(
295         uint256 _amountToDistribute,
296         uint256 _basisPointsPercentage
297     ) public pure returns (uint256 amount_) {
298         amount_ = ((_amountToDistribute * _basisPointsPercentage) /
BASIS_POINTS_DIVISOR);
299     }
```

When the player `bet()`, the corresponding referrer will be rewarded with a ReferralReward with the amount of $amountRebate_{bet}$:

$$amountRebate_{bet} = amountToDistribute \cdot \frac{basisPointsPercentage}{BASISPOINTSDIVISOR}$$

$$= \text{amount} \cdot \frac{tiers[referrerTiers[referrer]].WLPRate}{BASISPOINTSDIVISOR}$$

$$= \text{wager} \cdot \frac{houseEdge}{BASISPOINTS} \cdot \frac{tiers[referrerTiers[referrer]].WLPRate}{BASISPOINTSDIVISOR}$$

$amountRebate_{bet}$ will be stored as `participants[gameId][player].referralReward`:

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Wheel.sol#L366-L387

```
366   function refundGame(uint256 _gameId) external nonReentrant {
367     address sender_ = _msgSender();
368     Bet storage bet_ = participants[_gameId][sender_];
      @@ 369,385 @@
386     vaultManager.removeReferralReward(bet_.tokens[0], sender_,
      bet_.referralReward);
387   }
```

The value of the parameter `_amount` in `vaultManager.removeReferralReward()` is: `participants[gameId][player].referralReward` :

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/core/VaultManager.sol#L69-L75

```
69   function removeReferralReward(
70     address _token,
71     address _player,
72     uint256 _amount
73   ) public onlyGame onlyWhitelistedToken(_token) {
74     referralStorage.removeReward(_player, _token, _amount);
75   }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L433-L460

```
433 function removeReward(
434     address _player,
435     address _token,
436     uint256 _amount
437 ) external onlyProtocol {
438     address referrer_ = returnPlayerRefferalAddress(_player);
439
```

```
440     if (referrer_ != address(0)) {
441         // the player has a referrer
442         // calculate the rebate for the referrer tier
443         uint256 amountRebate_ = calculateRebate(
444             _amount,
445             tiers[referrerTiers[referrer_]].WLPRate
446         );
447         // nothing to rebate, return early
448         if (amountRebate_ == 0) {
449             return;
450         }
451 
452         if (rewards[referrer_][_token] >= amountRebate_) {
453             rewards[referrer_][_token] -= amountRebate_;
454             // remove the rebate to the referral reserves of the vault
455             IVault(vault).removeAsideReferral(_token, amountRebate_);
456         }
    @@ 457,458 @@
459     }
460 }
```

When the player `refundGame()`, the corresponding referrer will be deducted by the amount of $amountRebate_{refund}$:

$$amountRebate_{refund} = amountToDistribute \cdot \frac{basisPointsPercentage}{BASISPOINTSDIVISOR}$$

= amount $\cdot \frac{tiers[referrerTiers[referrer]].WLPRate}{BASISPOINTSDIVISOR}$
= participants[gameId][player].referralReward $\cdot \frac{tiers[referrerTiers[referrer]].WLPRate}{BASISPOINTSDIVISOR}$
= amountRebate$_{bet}$ $\cdot \frac{tiers[referrerTiers[referrer]].WLPRate}{BASISPOINTSDIVISOR}$
< amountRebate$_{bet}$

However, the expected behavior is that when refunding a game, the ReferralReward should always be deducted by the corresponding amount that was bet (i.e. $amountRebate_{bet} = participants[gameId][player].referralReward$).

Recommendation

Change to:

```
433  function removeReward(
434      address _player,
435      address _token,
436      uint256 _amountRebate
437  ) external onlyProtocol {
438      address referrer_ = returnPlayerRefferalAddress(_player);
439
440      if (referrer_ != address(0)) {
441          // nothing to rebate, return early
442          if (_amountRebate == 0) {
443              return;
444          }
445
446          if (rewards[referrer_][_token] >= _amountRebate) {
447              rewards[referrer_][_token] -= _amountRebate;
448              // remove the rebate to the referral reserves of the vault
449              IVault(vault).removeAsideReferral(_token, _amountRebate);
450          }
451
452          emit RewardRemoved(referrer_, _player, _token, _amountRebate);
453      }
454  }
```

Status

✓ Fixed

[WP-M5] When the `ReferralStorage.playerReferralCodes[player]` changes, the `refundGame()` function in the game will mistakenly refund the ReferralReward to the wrong referrer.

Medium

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L89-L126

```
@@ 89,94 @@
95      function setPlayerReferralCode(
96          address _account,
97          bytes32 _code
98      ) external override onlySupport {
99          _setPlayerReferralCode(_account, _code);
100     }
@@ 101,106 @@
107     function setPlayerReferralCodeByUser(bytes32 _code) external {
108         _setPlayerReferralCode(msg.sender, _code);
109     }
@@ 110,116 @@
117     function _setPlayerReferralCode(address _account, bytes32 _code) private {
@@ 118,121 @@
122         // Set the player's referral code.
123         playerReferralCodes[_account] = _code;
@@ 124,125 @@
126     }
```

Player and Support can modify `playerReferralCodes[_account]` (the code corresponding to the player).

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Wheel.sol#L345-L364

```
345   function bet(
346     uint256 _wager,
347     Color _color,
348     address[2] memory _tokens
349   ) external nonReentrant isWagerAcceptable(_tokens[0], _wager) whenNotPaused {
      @@ 350,357 @@
358     (uint256 _referralReward, uint256 _vWINRAmount) = _escrow(player_, _wager, _color, _tokens);
359
360     /// @notice sets players bet to the list
361     participants[currentGameId_][player_] = Bet(_color, _wager, _tokens, _referralReward, _vWINRAmount);
      @@ 362,363 @@
364   }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Wheel.sol#L319-L339

```
319   function _escrow(
320     address _player,
321     uint256 _wager,
322     Color _color,
323     address[2] memory _tokens
324   ) internal returns (uint256 referralReward_, uint256 vWINRAmount_) {
      @@ 325,336 @@
337     /// @notice sets referral reward if player has referee
338     referralReward_ = vaultManager_.setReferralReward(_tokens[0], _player, _wager, houseEdge);
339   }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/core/VaultManager.sol#L57-L67

```
57    function setReferralReward(
58      address _token,
59      address _player,
60      uint256 _amount,
61      uint64 _houseEdge
62    ) public onlyGame onlyWhitelistedToken(_token) returns (uint256 referralReward_)
{
63      if (_amount > 0) {
64        return referralStorage.setReward(_player, _token, ((_amount * _houseEdge) /
BASIS_POINTS));
65      }
66      return 0;
67    }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L402-L431

```
402  function setReward(address _player, address _token, uint256 _amount) external
     onlyProtocol returns(uint256 _reward){
403      address referrer_ = returnPlayerRefferalAddress(_player);
404
405      if (referrer_ != address(0)) {
     @@ 406,417 @@
418          // add the rebate to the rewards mapping of the referrer
419          unchecked {
420              rewards[referrer_][_token] += amountRebate_;
421          }
     @@ 422,428 @@
429      }
430      emit NoRewardToSet(_player);
431  }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L389-L394

```
389 function returnPlayerRefferalAddress(
390     address _player
391 ) public view returns (address referrer_) {
392     (, referrer_) = getPlayerReferralInfo(_player);
393     return referrer_;
394 }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L334-L352

```
334 function getPlayerReferralInfo(
335     address _account
336 ) public view override returns (bytes32 code_, address referrer_) {
337     // Retrieve the player's referral code from the playerReferralCodes mapping.
338     code_ = playerReferralCodes[_account];
339 
340     // If the player has a referral code, retrieve the referrer address from the
    codeOwners mapping.
341     if (code_ != bytes32(0)) {
342         referrer_ = codeOwners[code_];
343     }
    @@ 344,350 @@
351     return (code_, referrer_);
352 }
```

When the player `bet()` , a ReferralReward will be added to the $referrer_1$ corresponding to $code_{time1}$ `ReferralStorage.playerReferralCodes[player]` :

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Wheel.sol#L366-L387

```
366   function refundGame(uint256 _gameId) external nonReentrant {
367     address sender_ = _msgSender();
368     Bet storage bet_ = participants[_gameId][sender_];
```

```
@@ 369,385 @@
386     vaultManager.removeReferralReward(bet_.tokens[0], sender_,
bet_.referralReward);
387   }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/core/VaultManager.sol#L69-L75

```
69    function removeReferralReward(
70      address _token,
71      address _player,
72      uint256 _amount
73    ) public onlyGame onlyWhitelistedToken(_token) {
74      referralStorage.removeReward(_player, _token, _amount);
75    }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L433-L460

```
433     function removeReward(
434         address _player,
435         address _token,
436         uint256 _amount
437     ) external onlyProtocol {
438         address referrer_ = returnPlayerRefferalAddress(_player);
439
440         if (referrer_ != address(0)) {
@@ 441,451 @@
452             if (rewards[referrer_][_token] >= amountRebate_) {
453                 rewards[referrer_][_token] -= amountRebate_;
454                 // remove the rebate to the referral reserves of the vault
455                 IVault(vault).removeAsideReferral(_token, amountRebate_);
456             }
```

```
@@ 457,458 @@
459          }
460      }
```

When the player calls `refundGame()`, the refund amount is deducted from the corresponding $referrer_2$ account for the ReferralReward associated with $code_{time2}$ of `ReferralStorage.playerReferralCodes[player]`.

If `ReferralStorage.playerReferralCodes[player]` has changed between $time\_1$ and $time\_2$, then $referrer\_2$ will be different from $referrer\_1$.

However, the expected behavior is that ReferralReward should always be deducted from the $referrer_1$ account corresponding to the bet when issuing a refund during the game.

Recommendation

See the Recommendation of [WP-L10].

Status

✓ Fixed

[WP-M7] If the limit is exceeded after this mint, it should mint as much as possible instead of not mining at all

Medium

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/core/TokenManager.sol#L272-L292

```
272  function mintVestedWINR(
273      address _input,
274      uint256 _amount,
275      address _recipient
276  ) external nonReentrant onlyProtocol returns(uint256 _mintAmount){
277      //mint with mining strategy
278      uint256 _feeAmount = feeStrategy.calculate(_input, _amount);
279      _mintAmount = miningStrategy.calculate(_recipient, _feeAmount, mintedByGames);
280      // get referral rate
281      uint256 _vWINRRate = referralStorage.getPlayerVestedWINRRate(_recipient);
282      // add vested WINR rate to mint amount
283      if (_vWINRRate > 0) {
284          _mintAmount += (_mintAmount * _vWINRRate) / BASIS_POINTS;
285      }
286      // mint Vested WINR
287      if (mintedByGames + _mintAmount <= MAX_MINT) {
288          vWINR.mint(_recipient, _mintAmount);
289          accumFee += _mintAmount / mintDivider;
290          mintedByGames += _mintAmount;
291      }
292  }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/core/VaultManager.sol#L164-L170

```
164    function mintVestedWINR(
165      address _input,
166      uint256 _amount,
```

```
167       address _recipient
168     ) public onlyGame returns (uint256 mintedAmount_) {
169       mintedAmount_ = tokenManager.mintVestedWINR(_input, _amount, _recipient);
170     }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Moon.sol#L246-L261

```
246     function _escrow(
247       address _player,
248       uint256 _wager,
249       address[2] memory _tokens
250     ) internal returns (uint256 referralReward_, uint256 vWINRAmount_) {
251       IVaultManager vaultManager_ = vaultManager;
252 
253       /// @notice escrows total wager to Vault Manager
254       vaultManager_.escrow(_tokens[0], _player, _wager);
255       /// @notice mints the vWINR rewards
256       vWINRAmount_ = vaultManager_.mintVestedWINR(_tokens[0], _wager, _player);
257       /// @notice sets referral reward if player has referee
258       referralReward_ = vaultManager_.setReferralReward(_tokens[0], _player, _wager,
    houseEdge);
259 
260       totalAmounts[currentGameId][_tokens[0]] += _wager;
261     }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Moon.sol#L267-L289

```
267     function bet(
268       uint256 _wager,
269       uint256 _multiplier,
270       address[2] calldata _tokens
271     )
272       external
273       nonReentrant
274       isWagerAcceptable(_tokens[0], _wager)
```

```
275       isChoiceInLimits(_multiplier)
276       whenNotPaused
277       whenNotClosed
278     {
279       address player_ = _msgSender();
280       require(participantIndex[currentGameId][player_] == 0, "MOO: Bet cannot
    change");
281
282       (uint256 _referralReward, uint256 _vWINRAmount) = _escrow(player_, _wager,
    _tokens);
283
284       /// @notice sets players bet to the list
285       participants[currentGameId].push(Bet(_multiplier, _wager, _tokens,
    _referralReward, _vWINRAmount));
286       participantIndex[currentGameId][player_] = participants[currentGameId].length;
287
288       emit Participated(currentGameId, player_, _wager, _multiplier, _tokens);
289     }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/multiplayer/Moon.sol#L291-L310

```
291   function refundGame(uint256 _gameId) external nonReentrant {
292     address sender_ = _msgSender();
293     (uint256 index_, Bet memory bet_) = getParticipant(_gameId, sender_);
294     Game storage game_ = games[_gameId];
295     if (game_.status != Status.REFUNDED) {
296       require(
297         game_.startTime + refundCooldown < block.timestamp,
298         "MOO: Game is not refundable yet"
299       );
300       require(game_.status == Status.STARTED, "MOO: Game can not refund");
301       game_.status = Status.REFUNDED;
302     }
303     require(bet_.amount != 0, "MOO: Cant refund zero");
304
305     participants[_gameId][index_].amount = 0;
306     refunds[_gameId][sender_] = true;
307
308     vaultManager.refund(bet_.tokens[0], bet_.amount, bet_.mintedVWINR,  sender_);
```

```
309       vaultManager.removeReferralReward(bet_.tokens[0], sender_,
      bet_.referralReward);
310     }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/core/VaultManager.sol#L77-L92

```
77    function refund(
78      address _token,
79      uint256 _amount,
80      uint256 _vWINRAmount,
81      address _player
82    ) public onlyGame {
83      _decreaseEscrow(_token, _amount);
84      tokenManager.decreaseVolume(_token, _amount);
85      transferOut(_token, _player, _amount);
86      if (_vWINRAmount != 0) {
87        tokenManager.takeVestedWINR(_player, _vWINRAmount);
88        tokenManager.burnVestedWINR(_vWINRAmount);
89      }
90
91      emit Refunded(_msgSender(), _player, _token, _amount);
92    }
```

PoC

Given:

- vWINR_MAX_MINT = 100
- mintedByGames = 95

1. Alice `bet()` 10 tokens on Moon.
    - As 12 vWINR tokens need to be minted, and the total already minted is 95, which is more than the maximum allowed mint of 100, no vWINR was minted.
    - But the `_mintAmount` returned is 12.
2. Alice decides to refund her bet.
    - The `_vWINRAmount` is equal to `bet_.mintedVWINR`, which equals 12, indicating that

`_vWINRAmount` is not equal to 0.

- If Alice does not possess any vWINR tokens, `tokenManager.takeVestedWINR(_player, _vWINRAmount)` will revert. This will result in an unsuccessful refund.

Recommendation

Change to:

```
272  function mintVestedWINR(
273      address _input,
274      uint256 _amount,
275      address _recipient
276  ) external nonReentrant onlyProtocol returns(uint256 _mintAmount){
277      //mint with mining strategy
278      uint256 _feeAmount = feeStrategy.calculate(_input, _amount);
279      _mintAmount = miningStrategy.calculate(_recipient, _feeAmount, mintedByGames);
280      // get referral rate
281      uint256 _vWINRRate = referralStorage.getPlayerVestedWINRRate(_recipient);
282      // add vested WINR rate to mint amount
283      if (_vWINRRate > 0) {
284          _mintAmount += (_mintAmount * _vWINRRate) / BASIS_POINTS;
285      }
286      // mint Vested WINR
287      if (mintedByGames + _mintAmount > MAX_MINT) {
288          _mintAmount = MAX_MINT - mintedByGames
289      }
290      vWINR.mint(_recipient, _mintAmount);
291      accumFee += _mintAmount / mintDivider;
292      mintedByGames += _mintAmount;
293  }
```

Status

✓ Fixed

[WP-M8] `decreaseVolume()` may revert

Medium

Issue Description

As `dailyVolumes` only tracks the volume for the current day, a refund can occur for a bet placed on the previous day. Therefore, if:

1. No bets have been placed yet on the current day;
2. A bet placed on the previous day is being refunded because `dailyVolumes[_dayIndex]` is currently 0;

Then executing `dailyVolumes[_dayIndex] -= _dollarValue` will result in a revert due to underflow.

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/strategies/MiningStrategy.sol#L213-L219

```
213  function decreaseVolume(address _input, uint256 _amount) external onlyProtocol {
214      uint256 _dayIndex = getVolumeDayIndex(); // Get the current day index to
     update the  volume
215      uint256 _dollarValue = computeDollarValue(_input, _amount); // Calculate the
     dollar value of the token amount using the computeDollarValue function
216      dailyVolumes[_dayIndex] -= _dollarValue; // Decrease the  volume of the
     current day index by the calculated dollar value
217
218      emit VolumeDecreased(_dollarValue, dailyVolumes[_dayIndex], _dayIndex); //
     Emit a VolumeDecreased event with the updated volume
219  }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/core/TokenManager.sol#L322-L324

```
322  function decreaseVolume(address _input, uint256 _amount) external nonReentrant
     onlyProtocol {
323      miningStrategy.decreaseVolume(_input, _amount);
324  }
```

https://github.com/JustbSerbia/justbet-contracts/blob/76d8b877d862dfdcfed10bf55ee118d3be391fbd/contracts/games/core/VaultManager.sol#L77-L92

```
77    function refund(
78      address _token,
79      uint256 _amount,
80      uint256 _vWINRAmount,
81      address _player
82    ) public onlyGame {
83      _decreaseEscrow(_token, _amount);
84      tokenManager.decreaseVolume(_token, _amount);
85      transferOut(_token, _player, _amount);
86      if (_vWINRAmount != 0) {
87        tokenManager.takeVestedWINR(_player, _vWINRAmount);
88        tokenManager.burnVestedWINR(_vWINRAmount);
89      }
90
91      emit Refunded(_msgSender(), _player, _token, _amount);
92    }
```

Recommendation

Change to:

```
213  function decreaseVolume(address _input, uint256 _amount) external onlyProtocol {
214      uint256 _dayIndex = getVolumeDayIndex(); // Get the current day index to
     update the  volume
215      uint256 _dollarValue = computeDollarValue(_input, _amount); // Calculate the
     dollar value of the token amount using the computeDollarValue function
216      if (dailyVolumes[_dayIndex] > _dollarValue) {
217          dailyVolumes[_dayIndex] -= _dollarValue;
218      } else {
219          dailyVolumes[_dayIndex] = 0;
220      }
221          // Decrease the  volume of the current day index by the calculated dollar
     value
222
```

```
223         emit VolumeDecreased(_dollarValue, dailyVolumes[_dayIndex], _dayIndex); //
    Emit a VolumeDecreased event with the updated volume
224     }
```

Status

✓ Fixed

[WP-L9] Consider adding a slippage control parameter `minWlp` for `claim()`

Low

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/referrals/ReferralStorage.sol#L470-L502

```
470  function claim() public whenNotPaused nonReentrant {
471      address referrer_ = _msgSender();
472
473      require(!_referrerOnBlacklist(referrer_), "Referrer is blacklisted");
474
475      uint256 lastWithdrawTime_ = lastWithdrawTime[referrer_];
476      require(
477          block.timestamp >= lastWithdrawTime_ + withdrawInterval,
478          "Rewards can only be withdrawn once per withdrawInterval"
479      );
480
481      // check: update last withdrawal time
482      lastWithdrawTime[referrer_] = block.timestamp;
483
484      // effects: calculate total WLP amount and update withdrawn rewards
485      uint256 totalWlpAmount_;
486      address[] memory wlTokens_ = allWhitelistedTokens;
487      for (uint256 i = 0; i < wlTokens_.length; ++i) {
488          address token_ = wlTokens_[i];
489          uint256 amount_ = rewards[referrer_][token_] -
     withdrawn[referrer_][token_];
490          withdrawn[referrer_][token_] = rewards[referrer_][token_];
491
492          // interactions: convert token rewards to WLP
493          if (amount_ > 0) {
494              totalWlpAmount_ += _convertReferralTokensToWLP(token_, amount_);
495          }
496      }
497      // transfer WLP tokens to referrer
498      if (totalWlpAmount_ > 0) {
```

```
499             wlp.transfer(referrer_, totalWlpAmount_);
500         }
501         emit Claim(referrer_, totalWlpAmount_);
502     }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/referrals/ReferralStorage.sol#L565-L582

```
565     function _convertReferralTokensToWLP(
566         address _token,
567         uint256 _amount
568     ) internal returns (uint256 wlpAmount_) {
569         uint256 currentWLPBalance_ = wlp.balanceOf(address(this));
570 
571         // approve WLPManager to spend the tokens
572         IERC20(_token).approve(address(wlpManager), _amount);
573 
574         // WLPManager returns amount of WLP minted
575         wlpAmount_ = wlpManager.addLiquidity(_token, _amount, 0, 0);
576 
577         // note: if we want to check if the mint was successful and the WLP actually
        sits in this contract, we should do it like this:
578         require(
579             wlp.balanceOf(address(this)) == currentWLPBalance_ + wlpAmount_,
580             "ReferralStorage: WLP mint failed"
581         );
582     }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/referrals/ReferralStorage.sol#L510-L557

```
510     function getPendingWLPRewards(
511         address _referrer
512     ) public view returns (uint256 totalWlpAmount_) {
513         address[] memory wlTokens_ = allWhitelistedTokens;
514 
```

```
515       // Loop through each whitelisted token
516       for (uint256 i = 0; i < wlTokens_.length; ++i) {
      @@ 517,554 @@
555       }
556       return totalWlpAmount_;
557   }
```

In the `_convertReferralTokensToWLP()` function, both `_minUsdw` and `_minWlp` were set to 0 when calling `addLiquidity` . To prevent significant changes in the conditions, it is recommended to add a condition control parameter `minWlp` to the `claim()` function, which can be obtained from the frontend using `getPendingWLPRewards()` .

Status

✓ Fixed

[WP-L10] Constrains can be bypassed by `setCodeOwner()` to an new address

Low

Issue Description

The `withdrawInterval` constraint can easily be bypassed by setting the owner address for the same code to another address (`setCodeOwner()`) before each `claim()` :

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/referrals/ReferralStorage.sol#L151-L166

```
151  function setCodeOwner(bytes32 _code, address _newAccount) external {
152      // Ensure that the code is not empty.
153      require(_code != bytes32(0), "ReferralStorage: invalid _code");
154
155      // Get the current account owner of the code.
156      address account = codeOwners[_code];
157
158      // Ensure that the caller is the current account owner.
159      require(msg.sender == account, "ReferralStorage: forbidden");
160
161      // Set the new account owner for the code.
162      codeOwners[_code] = _newAccount;
163
164      // Emit an event to log the code owner change.
165      emit SetCodeOwner(msg.sender, _newAccount, _code);
166  }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L470-L502

```
470  function claim() public whenNotPaused nonReentrant {
471      address referrer_ = _msgSender();
472
```

```
473         require(!_referrerOnBlacklist(referrer_), "Referrer is blacklisted");
474
475         uint256 lastWithdrawTime_ = lastWithdrawTime[referrer_];
476         require(
477             block.timestamp >= lastWithdrawTime_ + withdrawInterval,
478             "Rewards can only be withdrawn once per withdrawInterval"
479         );
480
481         // check: update last withdrawal time
482         lastWithdrawTime[referrer_] = block.timestamp;
483
484         // effects: calculate total WLP amount and update withdrawn rewards
485         uint256 totalWlpAmount_;
486         address[] memory wlTokens_ = allWhitelistedTokens;
487         for (uint256 i = 0; i < wlTokens_.length; ++i) {
488             address token_ = wlTokens_[i];
489             uint256 amount_ = rewards[referrer_][token_] -
    withdrawn[referrer_][token_];
490             withdrawn[referrer_][token_] = rewards[referrer_][token_];
491
492             // interactions: convert token rewards to WLP
493             if (amount_ > 0) {
494                 totalWlpAmount_ += _convertReferralTokensToWLP(token_, amount_);
495             }
496         }
497         // transfer WLP tokens to referrer
498         if (totalWlpAmount_ > 0) {
499             wlp.transfer(referrer_, totalWlpAmount_);
500         }
501         emit Claim(referrer_, totalWlpAmount_);
502     }
```

The blacklist can easily be bypassed by setting the owner address for the same code to another address:

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/referrals/ReferralStorage.sol#L389-L395

```
389  function returnPlayerRefferalAddress(
390      address _player
391  ) public view returns (address referrer_) {
392      (, referrer_) = getPlayerReferralInfo(_player);
393      return referrer_;
394  }
395
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/referrals/ReferralStorage.sol#L334-L352

```
334  function getPlayerReferralInfo(
335      address _account
336  ) public view override returns (bytes32 code_, address referrer_) {
337      // Retrieve the player's referral code from the playerReferralCodes mapping.
338      code_ = playerReferralCodes[_account];
339
340      // If the player has a referral code, retrieve the referrer address from the
     codeOwners mapping.
341      if (code_ != bytes32(0)) {
342          referrer_ = codeOwners[code_];
343      }
344
345      // Check if the referrer is on the blacklist, if yes, set the referrer address
     to 0x0.
346      if (_referrerOnBlacklist(referrer_)) {
347          referrer_ = address(0);
348      }
349
350      // Return the player's referral code and referrer address.
351      return (code_, referrer_);
352  }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/referrals/ReferralStorage.sol#L277-L279

```
277 function _referrerOnBlacklist(address _referrer) internal view returns (bool
    onBlacklist_) {
278     onBlacklist_ = referrerOnBlacklist[_referrer];
279 }
```

The ”cannot set own code” constraint can easily be bypassed by using another address to create a code and set that code as the referral code for the original address. Then, the owner address for the code can be set to the original address using the `setCodeOwner()` function.:

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L117-L126

```
117 function _setPlayerReferralCode(address _account, bytes32 _code) private {
118     // Ensure that the player is not setting their own code.
119     require(codeOwners[_code] != _account, "ReferralStorage: can not set own
    code");
120     // Ensure that the code exists.
121     require(codeOwners[_code] != address(0), "ReferralStorage: code does not
    exist");
122     // Set the player's referral code.
123     playerReferralCodes[_account] = _code;
124     // Emit an event to log the referral code setting.
125     emit SetPlayerReferralCode(_account, _code);
126 }
```

Recommendation

Consider getting rid of the concept of `referralCode` and using the `referralAddress` directly.

Status

✓ Fixed

[WP-L11] `calculateDistribution()` Leftover tokens in the `FeeCollector` contract due to precision loss

Low

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/core/FeeCollector.sol#L714-L721

```
714  function _setAmountsForWager(uint256 _amount) internal {
715      reserves.staking += calculateDistribution(_amount,
     wagerDistributionConfig.staking);
716      reserves.buybackAndBurn += calculateDistribution(
717          _amount,
718          wagerDistributionConfig.buybackAndBurn
719      );
720      reserves.core += calculateDistribution(_amount, wagerDistributionConfig.core);
721  }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/core/FeeCollector.sol#L406-L411

```
406  function calculateDistribution(
407      uint256 _amountToDistribute,
408      uint64 _basisPointsPercentage
409  ) public pure returns (uint256 amount_) {
410      amount_ = ((_amountToDistribute * _basisPointsPercentage) /
     BASIS_POINTS_DIVISOR);
411  }
```

Recommendation

Change to:

```
714  function _setAmountsForWager(uint256 _amount) internal {
715      uint256 forStaking = calculateDistribution(_amount,
     wagerDistributionConfig.staking);
716      reserves.staking += forStaking;
717      uint256 forBuybackAndBurn = calculateDistribution(
718          _amount,
719          wagerDistributionConfig.buybackAndBurn
720      );
721      reserves.buybackAndBurn += forBuybackAndBurn;
722      reserves.core += _amount - forStaking - forBuybackAndBurn;
723  }
```

Status

✓ Fixed

[WP-L12] Redundant code

Low

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/strategies/FeeStrategy.sol#L200-L211

```
200  if (index > 2) {
201      PeriodReserve memory prevReserve_ = periodReserves[index - 1]; // Get the
     previous day's reserve
202      _totalProfit - prevReserve_.profit; // Subtract the previous day's profit from
     the total profit
203
204      // Determine whether the reserve change type should be set to PROFIT or LOSS
     based on the difference between
205      // the total profit and total loss for the current day and the previous day
206      if (_totalProfit - prevReserve_.profit > _totalLoss - prevReserve_.loss) {
207          changeType_ = ReserveChangeType.PROFIT;
208      } else {
209          changeType_ = ReserveChangeType.LOSS;
210      }
211  }
```

Status

✓ Fixed

[WP-I13] The value of the `Reward` event may not be as expected

Informational

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L402-L431

```
402 function setReward(address _player, address _token, uint256 _amount) external
    onlyProtocol returns(uint256 _reward){
403     address referrer_ = returnPlayerRefferalAddress(_player);
404
405     if (referrer_ != address(0)) {
406         // the player has a referrer
407         // calculate the rebate for the referrer tier
408         uint256 amountRebate_ = calculateRebate(
409             _amount,
410             tiers[referrerTiers[referrer_]].WLPRate
411         );
412         // nothing to rebate, return early but emit event
413         if (amountRebate_ == 0) {
414             emit Reward(referrer_, _player, _token, 0, 0);
415             return 0;
416         }
417
418         // add the rebate to the rewards mapping of the referrer
419         unchecked {
420             rewards[referrer_][_token] += amountRebate_;
421         }
422
423         // add the rebate to the referral reserves of the vault (to keep it aside
    from the wagerFeeReserves)
424         IVault(vault).setAsideReferral(_token, amountRebate_);
425
426         emit Reward(referrer_, _player, _token, _amount, amountRebate_);
427
428         return amountRebate_;
429     }
430     emit NoRewardToSet(_player);
```

```
431 }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/interfaces/referrals/IReferralStorage.sol#L19

```
19     event Reward(address referrer, address player, address token, uint256 amount,
    uint256 rebateAmount);
```

It may be better to replace 0 `amount` at L414 with `_amount` .

Only `rebateAmount` can be confirmed as `0` , `amount` may not be `0` .

Emitting a constant `0` may be misleading for the `amount` variable.

Recommendation

Change to:

```
402 function setReward(address _player, address _token, uint256 _amount) external
    onlyProtocol returns(uint256 _reward){
403     address referrer_ = returnPlayerRefferalAddress(_player);
404 
405     if (referrer_ != address(0)) {
406         // the player has a referrer
407         // calculate the rebate for the referrer tier
408         uint256 amountRebate_ = calculateRebate(
409             _amount,
410             tiers[referrerTiers[referrer_]].WLPRate
411         );
412         // nothing to rebate, return early but emit event
413         if (amountRebate_ == 0) {
414             emit Reward(referrer_, _player, _token, _amount, 0);
415             return 0;
416         }
```

Status

✓ Fixed

[WP-N14] Inconsistent usage of `_msgSender()` / `msg.sender` to retrieve the sender's address.

Issue Description

There are instances where `_msgSender()` and `msg.sender` are both used to obtain the sender's address, leading to inconsistency. It is recommended to standardize the usage of either one of the methods for consistency and clarity in code.

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/stakings/WINR-staking/WINRVesting.sol#L444-L511

```
444     function withdrawVesting(uint256 _index) external whenNotPaused nonReentrant {
445         address sender_ = _msgSender();
@@ 446,510 @@
511     }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/stakings/WINR-staking/WINRVesting.sol#L517-L589

```
517     function withdrawVestingBatch(
518         uint256[] calldata indexes
519     ) external whenNotPaused nonReentrant {
520         address sender = msg.sender;
@@ 521,588 @@
589     }
```

Status

✓ Fixed

[WP-I15] If a `token` is removed from the whitelist (`allWhitelistedTokens`), users will not be able to claim the reward.

Informational

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/referrals/ReferralStorage.sol#L470-L502

```
470  function claim() public whenNotPaused nonReentrant {
     @@ 471,482 @@
483
484      // effects: calculate total WLP amount and update withdrawn rewards
485      uint256 totalWlpAmount_;
486      address[] memory wlTokens_ = allWhitelistedTokens;
487      for (uint256 i = 0; i < wlTokens_.length; ++i) {
488          address token_ = wlTokens_[i];
489          uint256 amount_ = rewards[referrer_][token_] -
     withdrawn[referrer_][token_];
490          withdrawn[referrer_][token_] = rewards[referrer_][token_];
491
492          // interactions: convert token rewards to WLP
493          if (amount_ > 0) {
494              totalWlpAmount_ += _convertReferralTokensToWLP(token_, amount_);
495          }
496      }
497      // transfer WLP tokens to referrer
498      if (totalWlpAmount_ > 0) {
499          wlp.transfer(referrer_, totalWlpAmount_);
500      }
501      emit Claim(referrer_, totalWlpAmount_);
502  }
```

Recommendation

Consider allowing users to specify the list of tokens they would like to claim through a parameter in the claim function.

Status

ⓘ Acknowledged

[WP-I17] `setWeightMultipliers()` can malfunction `unstake()`

Informational

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/stakings/WINR-staking/WINRVesting.sol#L321-L335

```
321  function setWeightMultipliers(
322      WeightMultipliers memory _weightMultipliers
323  ) external onlyGovernance {
324      require(_weightMultipliers.vWinr != 0, "vWINR dividend multiplier can not be
     zero");
325      require(
326          _weightMultipliers.vWinrVesting != 0,
327          "vWINR vesting multiplier can not be zero"
328      );
329      require(_weightMultipliers.winr != 0, "WINR multiplier can not be zero");
330      // Set the weight multipliers to the provided values
331      weightMultipliers = _weightMultipliers;
332
333      // Emit an event to notify listeners of the update
334      emit WeightMultipliersUpdate(_weightMultipliers);
335  }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/stakings/WINR-staking/WINRStaking.sol#L241-L280

```
241  function unstake(uint256 _amount, bool _isVested) external nonReentrant
     whenNotPaused {
242      address sender_ = _msgSender();
243      StakeDividend storage stake_ = _isVested
244          ? dividendVestedWINRStakes[sender_]
245          : dividendWINRStakes[sender_];
246      require(stake_.amount >= _amount, "Insufficient stake amount");
247      ITokenManager tokenManager_ = tokenManager;
```

```
248
249     // Compute the amount of tokens to be burned and sent to the staker.
250     uint256 burnAmount_ = _computeBurnAmount(_amount);
251     uint256 sendAmount_ = _amount - burnAmount_;
252     // Compute the weight of the unstaked tokens and update the total staked
    amount and weight.
253     uint256 unstakedWeight_;
254
255     // Claim dividends for the staker.
256     _claimDividend(sender_, _isVested);
257
258     // Burn the necessary amount of tokens and send the remaining unstaked tokens
    to the staker.
259     if (_isVested) {
260         tokenManager_.burnVestedWINR(burnAmount_);
261         tokenManager_.sendVestedWINR(sender_, sendAmount_);
262         unstakedWeight_ = _amount * weightMultipliers.vWinr;
263         totalStakedVestedWINR -= _amount;
264     } else {
265         tokenManager_.burnWINR(burnAmount_);
266         tokenManager_.sendWINR(sender_, sendAmount_);
267         unstakedWeight_ = _amount * weightMultipliers.winr;
268         totalStakedWINR -= _amount;
269     }
270
271     totalWeight -= unstakedWeight_;
272
273     // Update the stake details after unstaking tokens.
274     stake_.amount -= _amount;
275     stake_.weight -= unstakedWeight_;
276     stake_.profitDebt = _calcDebt(stake_.weight);
277
278     // Emit an Unstake event with the details of the unstaked tokens.
279     emit Unstake(sender_, block.timestamp, sendAmount_, burnAmount_, _isVested);
280 }
```

When the new `weightMultipliers` is higher than the original values, L275 may revert.

Recommendation

Consider making the `weightMultipliers` constants or immutable variables so that they cannot be changed.

Status

✓ Fixed

[WP-N18] Misleading comment

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/stakings/WINR-staking/WINRVesting.sol#L444-L511

```
444  function withdrawVesting(uint256 _index) external whenNotPaused nonReentrant {
     @@ 445,468 @@
469
470      uint256 amountToBurn = stake_.amount - redeemable_;
471
472      // Mint WINR tokens to decrease total supply
473      if (amountToBurn > 0) {
474          // this code piece is used to decrease burn amount from WINR total supply
475          tokenManager.mintWINR(address(tokenManager), amountToBurn);
476          tokenManager.burnWINR(amountToBurn);
477      }
478
479      // Burn vested WINR tokens
480      tokenManager.burnVestedWINR(stake_.amount);
481
     @@ 482,510 @@
511  }
```

`Mint WINR tokens to decrease total supply` should be `Mint WINR tokens to decrease MAX_SUPPLY` :

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/mocks/WINRMock.sol#L54-L58

```
54  function burn(uint256 amount) external {
55      _burn(msg.sender, amount);
56      MAX_SUPPLY -= amount;
57      emit Burn(msg.sender, amount);
58  }
```

Status

✓ Fixed

[WP-N19] Consider making `WINRVesting` an abstract contract to improve readability.

Issue Description

This will make it clear to readers that `WINRVesting` is not a concrete contract and cannot be deployed on its own.

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/stakings/WINR-staking/WINRVesting.sol#L11

```
11  contract WINRVesting is IWINRStaking, Pausable, ReentrancyGuard, AccessControlBase
    {
```

Recommendation

```
11  abstract contract WINRVesting is IWINRStaking, Pausable, ReentrancyGuard,
    AccessControlBase {
```

Status

✓ Fixed

[WP-G20] Redundant code wastes gas

Gas

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/strategies/FeeStrategy.sol#L264-L330

```
264  function _getMultiplier() internal returns (uint256) {
265      // Get the current period index
266      uint256 index  = getPeriodIndex();
267
268      // If the current period index is the same as the last calculated index,
     return the current multiplier
269      if (lastCalculatedIndex == index) {
270          return currentMultiplier;
271      }
272
273      // Set the reserve for the current period index
274      setReserve(index);
275
276      // If the period index is less than 1, set the current multiplier to the
     minimum multiplier value
277      if (index <= 2) {
278          currentMultiplier = config.maxMultiplier;
279          periodReserves[index].currentMultiplier = config.maxMultiplier;
280      } else {
     @@ 281,319 @@
320      }
321
322      // Update the last calculated index to the current period index
323      lastCalculatedIndex = index;
324
325      // Set the wager fee for the current period index and current multiplier
326      _setWagerFee(index, currentMultiplier);
327
328      // Return the current multiplier
329      return currentMultiplier;
330  }
```

https://github.com/WINRLabs/winr-protocol/blob/a3af15596b1e442092164b61e55f247f43083ec2/contracts/strategies/FeeStrategy.sol#L153-L157

```
153 function _setWagerFee(uint256 _index, uint256 _wagerFee) internal {
154     periodReserves[_index].currentMultiplier = _wagerFee;
155     vault.setWagerFee(_wagerFee);
156     emit FeeMultiplierChanged(currentMultiplier);
157 }
```

Recommendation

Remove L279.

Status

✓ Fixed

[WP-L21] `ReferralStorage.removeReward()` When `rewards[referrer_][_token] < amountRebate_` , `rewards[referrer_][_token]` should also get deducted

Low

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/referrals/ReferralStorage.sol#L433-L460

```
433  function removeReward(
434      address _player,
435      address _token,
436      uint256 _amount
437  ) external onlyProtocol {
438      address referrer_ = returnPlayerRefferalAddress(_player);
439
440      if (referrer_ != address(0)) {
441          // the player has a referrer
442          // calculate the rebate for the referrer tier
443          uint256 amountRebate_ = calculateRebate(
444              _amount,
445              tiers[referrerTiers[referrer_]].WLPRate
446          );
447          // nothing to rebate, return early
448          if (amountRebate_ == 0) {
449              return;
450          }
451
452          if (rewards[referrer_][_token] >= amountRebate_) {
453              rewards[referrer_][_token] -= amountRebate_;
454              // remove the rebate to the referral reserves of the vault
455              IVault(vault).removeAsideReferral(_token, amountRebate_);
456          }
457
458          emit RewardRemoved(referrer_, _player, _token, _amount);
459      }
460  }
```

Recommendation

Consider reducing `rewards[referrer_][_token]` to 0 when `rewards[referrer_][_token] < amountRebate_` , and reflecting in the event that ”not fully reduced” (distinguished from normal reduction).

Status

✓ Fixed

[WP-I22] `MiningStrategy._updateHalvings()` Lack of sanity check for `_percentages.length`

Informational

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/strategies/MiningStrategy.sol#L77-L95

```
77  function _updateHalvings(uint256[] memory _percentages, Config[] memory _configs)
    internal {
78      require(_percentages.length == _configs.length, "Lengths must be equal");
79      for (uint256 i = 0; i < _percentages.length; i++) {
80          require(_configs[i].maxMultiplier != 0, "Max zero");
81          require(_configs[i].minMultiplier != 0, "Min zero");
82          require(
83              _configs[i].minMultiplier < _configs[i].maxMultiplier,
84              "Min greater than max"
85          );
86          halvings[_percentages[i]] = _configs[i];
87      }
88      percentages = _percentages;
89
90      if (currentMultiplier == 0) {
91          currentMultiplier = _configs[0].maxMultiplier;
92      }
93
94      emit ConfigUpdated(_percentages, _configs);
95  }
```

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/strategies/MiningStrategy.sol#L316-L330

```
316 function findIndex(uint256 ratio) internal view returns (uint8 index) {
317     uint8 min = 0;
318     uint8 max = uint8(percentages.length) - 1;
319
```

```
320         while (min < max) {
321             uint8 mid = (min + max) / 2;
322             if (ratio < percentages[mid]) {
323                 max = mid;
324             } else {
325                 min = mid + 1;
326             }
327         }
328 
329         return min;
330     }
```

Recommendation

Consider adding a sanity check for `_updateHalvings()` :

- The upper limit is `_percentages.length` must be less than or equal to `type(uint8).max`
- `_percentages` must be in ascending order.

Status

ⓘ Acknowledged

[WP-G23] Using swap instead of shifting can save a lot of gas

Gas

Issue Description

https://github.com/WINRLabs/winr-protocol/blob/c75556b0dc1b500ac1139ed60909e2349bfe878f/contracts/stakings/WINR-staking/WINRVesting.sol#L641-L660

```
641  function _removeActiveIndex(address staker, uint index) internal {
642      uint[] storage indexes;
643
644      indexes = activeVestingIndexes[staker];
645
646      uint length = indexes.length;
647
648      // Find the index to remove
649      for (uint i = 0; i < length; i++) {
650          if (indexes[i] == index) {
651              // Shift all subsequent elements left by one position
652              for (uint j = i; j < length - 1; j++) {
653                  indexes[j] = indexes[j + 1];
654              }
655              // Remove the last element
656              indexes.pop();
657              return;
658          }
659      }
660  }
```

Given that the order of the indices is not being used, it is not necessary to maintain the order when `_removeActiveIndex()` .

By using a swap instead of shifting all subsequent elements left by one position, it saves a lot of gas.

Recommendation

```
641  function _removeActiveIndex(address staker, uint index) internal {
642      uint[] storage indexes = activeVestingIndexes[staker];
643
644      uint length = indexes.length;
645
646      // Find the index to remove
647      for (uint i = 0; i < length; i++) {
648          if (indexes[i] == index) {
649              // Swap with the last element
650              indexes[i] = indexes[length - 1];
651
652              // Remove the last element
653              indexes.pop();
654              return;
655          }
656      }
657  }
```

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.