

REPORT 60437204F31D800018A7F8DF

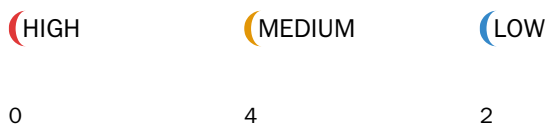
Created Sat Mar 06 2021 12:13:56 GMT+0000 (Coordinated Universal Time)
Number of analyses 1
User admin@fullsail.finance

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
e0f2503a-3a2d-4eef-b561-b6e50d225533	localhost/contracts/SailReferral.sol	6

Started	Sat Mar 06 2021 12:14:03 GMT+0000 (Coordinated Universal Time)
Finished	Sat Mar 06 2021 12:29:13 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Client Tool	Remythx
Main Source File	Localhost/Contracts/SailReferral.Sol

DETECTED VULNERABILITIES



ISSUES

MEDIUM Function could be marked as external.

SWC-000 The function definition of "recordReferral" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

localhost/contracts/libs/IBEP20.sol

Locations

```
32  * @dev Returns the amount of tokens owned by 'account'.
33  */
34  function balanceOf(address account) external view returns (uint256)
35
36  /**
37  * @dev Moves 'amount' tokens from the caller's account to 'recipient'.
38  */
39  * Returns a boolean value indicating whether the operation succeeded.
40  */
41  * Emits a {Transfer} event.
42  */
43  function transfer(address recipient, uint256 amount) external returns (bool)
44
45  /**
46  * @dev Returns the remaining number of tokens that 'spender' will be
47  * allowed to spend on behalf of 'owner' through {transferFrom}. This is
48  * zero by default.
```

MEDIUM Function could be marked as external.

SWC-000

The function definition of "getReferrer" is marked "public". However, it is never directly called by another function in the same contract or in any of its descendants. Consider to mark it as "external" instead.

Source file

localhost/contracts/libs/IBEP20.sol

Locations

```
45  /**
46   * @dev Returns the remaining number of tokens that `spender` will be
47   * allowed to spend on behalf of `owner` through {transferFrom}. This is
48   * zero by default.
49   *
50   * This value changes when {approve} or {transferFrom} are called.
51   */
52  function allowance(address _owner, address spender) external view returns (uint256);
```

LOW

Requirement violation.

A requirement was violated in a nested call and the call was reverted as a result. Make sure valid inputs are provided to the nested call (for instance, via passed arguments).

SWC-123

Source file

localhost/contracts/libs/IBEP20.sol

Locations

```
7  * @dev Returns the amount of tokens in existence.
8  */
9  function totalSupply() external view returns (uint256);
10
11  /**
12   * @dev Returns the token decimals.
13   */
14  function decimals() external view returns (uint8);
15
16  /**
17   * @dev Returns the token symbol.
18   */
19  function symbol() external view returns (string memory);
20
21  /**
22   * @dev Returns the token name.
23   */
24  function name() external view returns (string memory);
25
26  /**
27   * @dev Returns the bep token owner.
28   */
29  function getOwner() external view returns (address);
30
31  /**
32   * @dev Returns the amount of tokens owned by 'account'.
33   */
34  function balanceOf(address account) external view returns (uint256);
35
36  /**
37   * @dev Moves 'amount' tokens from the caller's account to 'recipient'.
38   *
39   * Returns a boolean value indicating whether the operation succeeded.
40   *
41   * Emits a {Transfer} event.
42   */
43  function transfer(address recipient, uint256 amount) external returns (bool);
44
45  /**
46   * @dev Returns the remaining number of tokens that 'spender' will be
47   * allowed to spend on behalf of 'owner' through {transferFrom}. This is
48   * zero by default.
49   *
50   * This value changes when {approve} or {transferFrom} are called.
51   */
52  function allowance(address _owner, address spender) external view returns (uint256);
53
54  /**
55   * @dev Sets 'amount' as the allowance of 'spender' over the caller's tokens.
56   *
57   * Returns a boolean value indicating whether the operation succeeded.
58   *
59   * IMPORTANT: Beware that changing an allowance with this method brings the risk
60   * that someone may use both the old and the new allowance by unfortunate
61
```

* transaction ordering. One possible solution to mitigate this race

* condition is to first reduce the spender's allowance to 0 and set the