MythX

REPORT 60668A47DD06F7001141408D

Created	Fri Apr 02 2021 03:06:47 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	mysteryfarm063@gmail.com

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
73b9852b-4170-4174-a2d3-1d3950e17d0e	MYSTERYToken.sol	24

Analysis 73b9852b-4170-4174-a2d3-1d3950e17d0e MythX

Started	Fri Apr 02 2021 03:06:52 GMT+0000 (Coordinated Universal Time)
Finished	Fri Apr 02 2021 03:53:04 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Remythx
Main Source File	MYSTERYToken.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	16	8

ISSUES

MEDIUM Function could be marked as external.

SWC-000 The function definition of "mint" 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
IBEP20.sol
Locations

```
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  /**
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "burn" 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

IBEP20.sol

Locations

```
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);
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "transfer" 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

IBEP20.sol

Locations

```
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 /**
47 * @dev Returns the remaining number of tokens that `spender` will be
48 * allowed to spend on behalf of `owner` through {transferFrom}. This is
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "decimals" 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

BEP20.sol

Locations

```
80  * @dev Returns the token decimals.
81  */
82  function decimals() public override view returns (uint8) {
83  return _decimals;
84  }
85
86  /**
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "symbol" 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

BEP20.sol

Locations

```
87  * @dev Returns the token symbol.
88  */
89  function symbol() public override view returns (string memory) {
90  return _symbol;
91  }
92
93  /**
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "allowance" 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

BEP20.sol

Locations

```
121 * @dev See {BEP20-allowance}.
122 */
123 function allowance(address owner, address spender) public override view returns (uint256) {
124 return _allowances[owner][spender];
125 }
126
127 /**
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "approve" 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

BEP20.sol

Locations

```
132 * - `spender` cannot be the zero address.
133 */
134 function approve(address spender, uint256 amount) public override returns (bool) {
135 _approve(_msgSender(), spender, amount);
136 return true;
137 }
138
139 /**
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "transferFrom" 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

BEP20.sol

Locations

```
149 * `amount`.
150 */
151 function transferFrom(
152 address sender,
153 address recipient,
154 uint256 amount
155 ) public virtual override returns (bool) {
156 _transfer(sender, recipient, amount);
157 _approve(
158 sender,
159 _msgSender(),
160 _allowances[sender][_msgSender()].sub(amount, 'BEP20: transfer amount exceeds allowance')
161 );
162 return true;
163 }
164
165 /**
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "increaseAllowance" 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

BEP20.sol

Locations

```
175 * - `spender` cannot be the zero address.
176 */
177 function increaseAllowance(address spender, uint256 addedValue) public returns (bool) {
178 _approve(_msgSender(), spender, _allowances[_msgSender()][spender].add(addedValue));
179 return true;
180 }
181
182 /**
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "decreaseAllowance" 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

BEP20.sol

Locations

```
194 * `subtractedValue`.
195 */
196 function decreaseAllowance(address spender, uint256 subtractedValue) public returns (bool) {
197 _approve(
198 _msgSender(),
199 spender,
200 _allowances[_msgSender()][spender].sub(subtractedValue, 'BEP20: decreased allowance below zero')
201 );
202 return true;
203 }
204
205 /**
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "addMinter" 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

BEP20.sol

Locations

```
381 }
382
383 function addMinter(address account) public onlyMinter {
384 _addMinter(account);
385 }
386
387 function removeMinter(address account) public onlyMinter {
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "removeMinter" 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

BEP20.sol

Locations

```
385 }
386
387 function removeMinter(address account) public onlyMinter {
388 _removeMinter(account);
389 }
390
391 function renounceMinter() public {
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "renounceMinter" 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

BEP20.sol

Locations

```
389 }
390
391 function renounceMinter() public {
392 _removeMinter(msg.sender);
393 }
394
395 function _addMinter(address account) internal {
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "owner" 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

Ownable.sol

Locations

```
34 * @dev Returns the address of the current owner.
35 */
36 function owner() public view returns (address) {
37 return _owner;
38 }
39
40 /**
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "renounceOwnership" 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

Ownable.sol

Locations

```
53  * thereby removing any functionality that is only available to the owner.
54  */
55  function renounceOwnership() public onlyOwner {
56  emit OwnershipTransferred(_owner, address(0));
57  _owner = address(0);
58  }
59
60  /**
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "transferOwnership" 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

Ownable.sol

Locations

```
62  * Can only be called by the current owner.
63  */
64  function transferOwnership(address newOwner) public onlyOwner {
65  _transferOwnership(newOwner);
66  }
67
68  /**
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.4.0"". It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.

Source file

IBEP20.sol

Locations

```
1  // SPDX-License-Identifier: GPL-3.0-or-later
2
3  pragma solidity ^0.4.0;
```