MythX

REPORT 60C644355536420019 11DA2C

Created	Sun Jun 13 2021 17:45:25 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	60c63fab8bfa125f70f292e8

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
30d92a24-2173-405f-b25c-cb1bdc45e9b6	DolphinPresale.sol	14

Analysis 30d92a24-2173-405f-b25c-cb1bdc45e9b6 MythX

Started	Sun Jun 13 2021 17:45:31 GMT+0000 (Coordinated Universal Time)
Finished	Sun Jun 13 2021 18:30:43 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Remythx
Main Source File	DolphinPresale.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	12	2

ISSUES

MEDIUM Function could be marked as external.

SWC-000 The function definition of "buyWithBNB" 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

DolphinPresale.sol

Locations

```
150
151 // Function to buy DLPH using BNB token
152 function buyWithBNB(uint256 buyAmount) public payable checkSaleRequirements(buyAmount) {
153 uint256 amount = calculateBNBAmount(buyAmount);
154 require(msg.value >= amount, 'Insufficient BNB balance');
155 require(buyAmount >= minPerTransaction, 'Lower than the minimal transaction amount');
156
157 uint256 sumSoFar = DLPHPerAddresses[msg.sender].add(buyAmount);
158 require(sumSoFar <= maxPerUser, 'Greater than the maximum purchase limit');
159
160 DLPHPerAddresses[msg.sender] = sumSoFar;
161 totalSold = totalSold.add(buyAmount);
162
163 DLPH.transfer(msg.sender, buyAmount);
164 emit tokensBought(msg.sender, amount, buyAmount, 'BNB', now);
165 }
166
167 //function to change the owner
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "changeOwner" 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

DolphinPresale.sol

Locations

```
167 //function to change the owner
168 //only owner can call this function
169 function changeOwner(address payable _owner) public {
170 require(msg.sender == owner);
171 owner = _owner;
172 }
173
174 // function to set the presale start date
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "setStartDate" 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

DolphinPresale.sol

Locations

```
174 // function to set the presale start date
175 // only owner can call this function
176 function setStartDate(uint256 _startDate) public {
177 require(msg.sender == owner && saleEnded == false);
178 startDate = _startDate;
179 }
180
181 // function to set the presale end date
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "setEndDate" 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

DolphinPresale.sol

Locations

```
181 // function to set the presale end date
182 // only owner can call this function
183 function setEndDate(uint256 _endDate) public {
184 require(msg.sender == owner && saleEnded == false);
185 endDate = _endDate;
186 }
187
188 // function to set the total tokens to sell
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "setTotalTokensToSell" 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

DolphinPresale.sol

Locations

```
188 // function to set the total tokens to sell
189 // only owner can call this function
190 function setTotalTokensToSell(uint256 _totalTokensToSell) public {
191 require(msg.sender == owner);
192 totalTokensToSell = _totalTokensToSell;
193 }
194
195 // function to set the minimal transaction amount
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "setMinPerTransaction" 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

DolphinPresale.sol

Locations

```
195 // function to set the minimal transaction amount
196 // only owner can call this function
197 function setMinPerTransaction(uint256 _minPerTransaction) public {
198 require(msg.sender == owner);
199 minPerTransaction = _minPerTransaction;
200 }
201
202 // function to set the maximum amount which a user can buy
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "setMaxPerUser" 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

DolphinPresale.sol

Locations

```
202 // function to set the maximum amount which a user can buy
203 // only owner can call this function
204 function setMaxPerUser(uint256 _maxPerUser) public {
205 require(msg.sender == owner);
206 maxPerUser = _maxPerUser;
207 }
208
209 // function to set the total tokens to sell
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "setTokenPricePerBNB" 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

DolphinPresale.sol

Locations

```
209 // function to set the total tokens to sell
210 // only owner can call this function
211 function setTokenPricePerBNB(uint256 _DLPHPerBnb) public {
212 require(msg.sender == owner);
213 require(_DLPHPerBnb > 0, "Invalid DLPH price per BNB");
214 DLPHPerBnb = _DLPHPerBnb;
215 }
216
217 //function to end the sale
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "endSale" 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

DolphinPresale.sol

Locations

```
217 //function to end the sale
218 //only owner can call this function
219 function endSale() public {
220 require(msg.sender == owner && saleEnded == false);
221 saleEnded = true;
222 }
223
224 //function to withdraw collected tokens by sale.
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "withdrawCollectedTokens" 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

DolphinPresale.sol

Locations

```
225 //only owner can call this function
226
227 function withdrawCollectedTokens() public {
228 require(msg.sender == owner);
229 require(address(this).balance > 0, "Insufficient balance");
230 owner.transfer(address(this).balance);
231 }
232
233 //function to withdraw unsold tokens
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "withdrawUnsoldTokens" 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

DolphinPresale.sol

Locations

```
233 //function to withdraw unsold tokens
234 //only owner can call this function
235 function withdrawUnsoldTokens() public {
236 require(msg.sender == owner);
237 uint256 remainedTokens = unsoldTokens();
238 require(remainedTokens > 0, "No remained tokens");
239 DLPH.transfer(owner, remainedTokens);
240 }
241
242 //function to return the amount of unsold tokens
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "calculateDLPHAmount" 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

DolphinPresale.sol

Locations

```
247
248 //function to calculate the quantity of DLPH token based on the DLPH price of bnbAmount
249 function calculateDLPHAmount(uint256 bnbAmount) public view returns (uint256) {
250 uint256 DLPHAmount = DLPHPerBnb.mul(bnbAmount).div(10**18);
251 return DLPHAmount;
252 }
253
254 //function to calculate the quantity of bnb needed using its DLPH price to buy `buyAmount` of DLPH tokens.
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.6.0"". 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

DolphinPresale.sol

Locations

```
1 pragma solidity ^0.6.0;
2
3 library SafeMath {
```

LOW

SWC-134

Call with hardcoded gas amount.

The highlighted function call forwards a fixed amount of gas. This is discouraged as the gas cost of EVM instructions may change in the future, which could break this contract's assumptions. If this was done to prevent reentrancy attacks, consider alternative methods such as the checks-effects-interactions pattern or reentrancy locks instead.

Source file

DolphinPresale.sol

Locations

```
228 require(msg.sender == owner);
229 require(address(this).balance > 0, "Insufficient balance");
230 owner.transfer(address(this).balance);
231 }
```