MythX

REPORT 60C64428723693001850C7DA2

Created	Sun Jun 13 2021 17:45:12 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	60c63fab8bfa125f70f292e8

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
3c20474b-2bd1-4b6e-841d-00674bdf69e9	DolphinToken.sol	20

Analysis 3c20474b-2bd1-4b6e-841d-00674bdf69e9 MythX

Started	Sun Jun 13 2021 17:45:20 GMT+0000 (Coordinated Universal Time)
Finished	Sun Jun 13 2021 18:30:55 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Remythx
Main Source File	DolphinToken.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	13	7

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
SafeMath.sol
Locations

```
8  *
9  * Arithmetic operations in Solidity wrap on overflow. This can easily result
10 * in bugs, because programmers usually assume that an overflow raises an
11 * error, which is the standard behavior in high level programming languages.
12 * `SafeMath` restores this intuition by reverting the transaction when an
13 * operation overflows.
14 *
```

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
DolphinToken.sol
Locations

```
60
61 /**
62 * @notice Delegate votes from `msg.sender` to `delegatee`
63 * @param delegatee The address to delegate votes to
64 */
65 function delegate(address delegatee) external {
66 return _delegate(msg.sender, delegatee);
67 }
```

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

DolphinToken.sol

Locations

```
65 function delegate(address delegatee) external {
66 return _delegate(msg.sender, delegatee);
67 }
68
69 /**
70 * @notice Delegates votes from signatory to `delegatee`
71 * @param delegatee The address to delegate votes to
72 * @param nonce The contract state required to match the signature
73 * @param expiry The time at which to expire the signature
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "totalSupply" 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

DolphinToken.sol

Locations

```
70 * @notice Delegates votes from signatory to `delegatee`
71 * @param delegatee The address to delegate votes to
72 * @param nonce The contract state required to match the signature
73 * @param expiry The time at which to expire the signature
74 * @param v The recovery byte of the signature
75 * @param r Half of the ECDSA signature pair
```

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

DolphinToken.sol

Locations

```
86 external
87 {
88 bytes32 domainSeparator = keccak256(
89 abi.encode(
90 DOMAIN_TYPEHASH,
91 keccak256(bytes(name())),
92 getChainId(),
93 address(this)
94 )
95 );
```

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

DolphinToken.sol

Locations

```
95  );
96
97  bytes32 structHash = keccak256(
98  abi.encode(
99  DELEGATION_TYPEHASH,
100 delegatee,
101 nonce,
102 expiry
103 )
104 );
```

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

DolphinToken.sol

Locations

```
107 abi.encodePacked(
108 "\x19\x01",
109 domainSeparator,
110 structHash
111 )
112 );
113
114 address signatory = ecrecover(digest, v, r, s);
115 require(signatory != address(0), "DLPH::delegateBySig: invalid signature");
116 require(nonce == nonces[signatory]++, "DLPH::delegateBySig: invalid nonce");
117 require(now <= expiry, "DLPH::delegateBySig: signature expired");
```

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

DolphinToken.sol

Locations

```
122 * @notice Gets the current votes balance for `account`
123 * @param account The address to get votes balance
124 * @return The number of current votes for `account`
125 */
126 function getCurrentVotes(address account)
127 external
128 view
129 returns (uint256)
130 {
131 uint32 nCheckpoints = numCheckpoints[account];
132 return nCheckpoints > 0 ? checkpoints[account][nCheckpoints - 1].votes : 0;
133 }
134
135 /**
136 * @notice Determine the prior number of votes for an account as of a block number
137 * @dev Block number must be a finalized block or else this function will revert to prevent misinformation.
138 * @param account The address of the account to check
```

MEDIUM

Function could be marked as external.

SWC-000

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

DolphinToken.sol

Locations

```
141 */
142 function getPriorVotes(address account, uint blockNumber)
143 external
144 view
145 returns (uint256)
146 {
147 require(blockNumber < block.number, "DLPH::getPriorVotes: not yet determined");
148
149 uint32 nCheckpoints = numCheckpoints[account];
150 if (nCheckpoints == 0) {
151 return 0;
152 }
```

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

DolphinToken.sol

Locations

```
164 uint32 lower = 0;
165 uint32 upper = nCheckpoints - 1;
166 while (upper > lower) {
167 uint32 center = upper - (upper - lower) / 2; // ceil, avoiding overflow
168 Checkpoint memory cp = checkpoints[account][center];
169 if (cp.fromBlock == blockNumber) {
170 return cp.votes;
171 } else if (cp.fromBlock < blockNumber) {
172 lower = center;
173 } else {
```

MEDIUM

SWC-000

Function could be marked as external.

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

DolphinToken.sol

Locations

```
178 }
179
180 function _delegate(address delegator, address delegatee)
181 internal
182 {
183 address currentDelegate = _delegates[delegator];
184 uint256 delegatorBalance = balanceOf(delegator); // balance of underlying DLPHs (not scaled);
185 _delegates[delegator] = delegatee;
```

LOW

SWC-116

A control flow decision is made based on The block.timestamp environment variable.

The block.timestamp environment variable is used to determine a control flow decision. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source file

SafeMath.sol

Locations

```
129 * Reverts when dividing by zero.
130 *
131 * Counterpart to Solidity's `%` operator. This function uses a `revert`
132 * opcode (which leaves remaining gas untouched) while Solidity uses an
133 * invalid opcode to revert (consuming all remaining gas).
134 *
```

LOW

SWC-128

Potentially unbounded data structure passed to builtin.

Gas consumption in function "delegateBySig" in contract "DolphinToken" depends on the size of data structures that may grow unboundedly. Specifically the "1-st" argument to builtin "keccak256" may be able to grow unboundedly causing the builtin to consume more gas than the block gas limit, effectively causing a denial-of-service condition.Consider that an attacker might attempt to cause this condition on purpose.

Source file

SafeMath.sol

Locations

```
109 * division by zero. The result is rounded towards zero.
110 *
111 * Counterpart to Solidity's `/` operator. Note: this function uses a
112 * `revert` opcode (which leaves remaining gas untouched) while Solidity
113 * uses an invalid opcode to revert (consuming all remaining gas).
```