

REPORT 60767F25595385001293A369

Created	Wed Apr 14 2021 05:35:33 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	contact@balldefi.com

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
39268d03-8ec9-4c60-934b-925916ed5249	Timelock.sol	8

Started	Wed Apr 14 2021 05:35:36 GMT+0000 (Coordinated Universal Time)
Finished	Wed Apr 14 2021 05:50:49 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Client Tool	Remythx
Main Source File	Timelock.sol

DETECTED VULNERABILITIES



ISSUES

MEDIUM

Function could be marked as external.
The function definition of "setDelay" 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.

SWC-000

Source file
Timelock.sol
Locations

```
206 | receive() external payable { }
207 |
208 | function setDelay(uint delay_) public {
209 |     require(msg.sender == address(this), "Timelock::setDelay: Call must come from Timelock.");
210 |     require(delay_ >= MINIMUM_DELAY, "Timelock::setDelay: Delay must exceed minimum delay.");
211 |     require(delay_ <= MAXIMUM_DELAY, "Timelock::setDelay: Delay must not exceed maximum delay.");
212 |     delay = delay_;
213 |
214 |     emit NewDelay(delay);
215 | }
216 |
217 | function acceptAdmin() public {
```

MEDIUM Function could be marked as external.

SWC-000

The function definition of "acceptAdmin" 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

Timelock.sol

Locations

```
215 | }
216 |
217 | function acceptAdmin() public {
218 |     require(msg.sender == pendingAdmin, "Timelock::acceptAdmin: Call must come from pendingAdmin.");
219 |     admin = msg.sender;
220 |     pendingAdmin = address(0);
221 |
222 |     emit NewAdmin(admin);
223 | }
224 |
225 | function setPendingAdmin(address pendingAdmin_) public {
```

MEDIUM Function could be marked as external.

SWC-000

The function definition of "setPendingAdmin" 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

Timelock.sol

Locations

```
223 | }
224 |
225 | function setPendingAdmin(address pendingAdmin_) public {
226 |     // allows one time setting of admin for deployment purposes
227 |     if (admin_initialized) {
228 |         require(msg.sender == address(this), "Timelock::setPendingAdmin: Call must come from Timelock.");
229 |     } else {
230 |         require(msg.sender == admin, "Timelock::setPendingAdmin: First call must come from admin.");
231 |         admin_initialized = true;
232 |     }
233 |     pendingAdmin = pendingAdmin_;
234 |
235 |     emit NewPendingAdmin(pendingAdmin);
236 | }
237 |
238 | function queueTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public returns (bytes32) {
```

MEDIUM Function could be marked as external.

SWC-000

The function definition of "queueTransaction" 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

Timelock.sol

Locations

```
236 }  
237  
238 function queueTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public returns (bytes32) {  
239     require(msg.sender == admin, "Timelock::queueTransaction: Call must come from admin.");  
240     require(eta >= getBlockTimestamp().add(delay), "Timelock::queueTransaction: Estimated execution block must satisfy delay.");  
241  
242     bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));  
243     queuedTransactions[txHash] = true;  
244  
245     emit QueueTransaction(txHash, target, value, signature, data, eta);  
246     return txHash;  
247 }  
248  
249 function cancelTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public {
```

MEDIUM Function could be marked as external.

SWC-000

The function definition of "cancelTransaction" 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

Timelock.sol

Locations

```
247 }  
248  
249 function cancelTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public {  
250     require(msg.sender == admin, "Timelock::cancelTransaction: Call must come from admin.");  
251  
252     bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));  
253     queuedTransactions[txHash] = false;  
254  
255     emit CancelTransaction(txHash, target, value, signature, data, eta);  
256 }  
257  
258 function executeTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public payable returns (bytes memory) {
```

MEDIUM Function could be marked as external.

SWC-000

The function definition of "executeTransaction" 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

Timelock.sol

Locations

```
256 }
257
258 function executeTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public payable returns (bytes memory) {
259     require(msg.sender == admin, "Timelock::executeTransaction: Call must come from admin.");
260
261     bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
262     require(queuedTransactions[txHash], "Timelock::executeTransaction: Transaction hasn't been queued.");
263     require(getBlockTimestamp() >= eta, "Timelock::executeTransaction: Transaction hasn't surpassed time lock.");
264     require(getBlockTimestamp() <= eta.add(GRACE_PERIOD), "Timelock::executeTransaction: Transaction is stale.");
265
266     queuedTransactions[txHash] = false;
267
268     bytes memory callData;
269
270     if (bytes(signature).length == 0) {
271         callData = data;
272     } else {
273         callData = abi.encodePacked(bytes4(keccak256(bytes(signature))), data);
274     }
275
276     // solium-disable-next-line security/no-call-value
277     (bool success, bytes memory returnData) = target.call.value(value)(callData);
278     require(success, "Timelock::executeTransaction: Transaction execution reverted.");
279
280     emit ExecuteTransaction(txHash, target, value, signature, data, eta);
281
282     return returnData;
283 }
284
285 function getBlockTimestamp() internal view returns (uint) {
```

LOW A floating pragma is set.

SWC-103

The current pragma Solidity directive is "">=0.6.0<0.8.0"". 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

Timelock.sol

Locations

```
1 // SPDX-License-Identifier: MIT
2
3 pragma solidity >=0.6.0 <0.8.0
4
5 /**
```

LOW

Potentially unbounded data structure passed to builtin.

SWC-128

Gas consumption in function "executeTransaction" in contract "Timelock" 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

Timelock.sol

Locations

```
271 | callData = data;  
272 | } else {  
273 | callData = abi.encodePacked(bytes4(keccak256(bytes(signature))), data);  
274 | }  
275 |
```