MythX

Analysis 87057fa4-dba3-4d12-b207-4f4fe36b9928 MythX

Started	Fri Apr 02 2021 13:35:26 GMT+0000 (Coordinated Universal Time)
Finished	Fri Apr 02 2021 14:20:26 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Remythx
Main Source File	Timelock.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	6	3

ISSUES

MEDIUM Function could be marked as external.

SWC-000

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.

Source file

Timelock.sol

Locations

```
239 receive() external payable { }
240
241 function setDelay(uint delay_) public {
242 require(msg.sender == address(this), "Timelock::setDelay: Call must come from Timelock.");
243 require(delay_ >= MINIMUM_DELAY, "Timelock::setDelay: Delay must exceed minimum delay.");
244 require(delay_ <= MAXIMUM_DELAY, "Timelock::setDelay: Delay must not exceed maximum delay.");
245 delay = delay_;
246
247 emit NewDelay(delay);
248 }
249
250 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

```
248 }
249
250 function acceptAdmin() public {
251 require(msg.sender == pendingAdmin, "Timelock::acceptAdmin: Call must come from pendingAdmin.");
252 admin = msg.sender;
253 pendingAdmin = address(0);
254
255 emit NewAdmin(admin);
256 }
257
258 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

```
256 }
257
258 function setPendingAdmin(address pendingAdmin_) public {
259 // allows one time setting of admin for deployment purposes
260 if (admin_initialized) {
261 require(msg.sender == address(this), "Timelock::setPendingAdmin: Call must come from Timelock.");
262 } else {
263 require(msg.sender == admin, "Timelock::setPendingAdmin: First call must come from admin.");
264 admin_initialized = true;
265 }
266 pendingAdmin = pendingAdmin_;
267
268 emit NewPendingAdmin(pendingAdmin);
269 }
270
271 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

```
269 }
270
271 function queueTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public returns (bytes32) {
272 require(msg.sender == admin, "Timelock::queueTransaction: Call must come from admin.");
273 require(eta >= getBlockTimestamp().add(delay), "Timelock::queueTransaction: Estimated execution block must satisfy delay.");
274
275 bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
276 queuedTransactions[txHash] = true;
277
278 emit QueueTransaction(txHash, target, value, signature, data, eta);
279 return txHash;
280 }
281
282 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

```
280 }
281
282 function cancelTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public {
283 require(msg.sender == admin, "Timelock::cancelTransaction: Call must come from admin.");
284
285 bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
286 queuedTransactions[txHash] = false;
287
288 emit CancelTransaction(txHash, target, value, signature, data, eta);
289 }
290
291 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

```
289 }
290
291 function executeTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public payable returns (bytes memory) {
292 require(msg.sender == admin, "Timelock::executeTransaction: Call must come from admin.");
293
294 bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
295 require(queuedTransactions[txHash], "Timelock::executeTransaction: Transaction hasn't been queued.");
296 require(getBlockTimestamp() >= eta, "Timelock::executeTransaction: Transaction hasn't surpassed time lock.");
297 require(getBlockTimestamp() <= eta.add(GRACE_PERIOD), "Timelock::executeTransaction: Transaction is stale.");
298
299 queuedTransactions[txHash] = false;
300
301 bytes memory callData;
302
303 if (bytes(signature).length == 0) {
304 callData = data;
305 } else {
306 callData = abi.encodePacked(bytes4(keccak256(bytes(signature))), data);
307 }
308
309 // solium-disable-next-line security/no-call-value
310 (bool success, bytes memory returnData) = target.call.value(value)(callData);
311 require(success, "Timelock::executeTransaction: Transaction execution reverted.");
312
313 emit ExecuteTransaction(txHash, target, value, signature, data, eta);
314
315 return returnData;
316 }
317
318 function getBlockTimestamp() internal view returns (uint) {
```

LOW

A floating pragma is set.

SWC-103

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

Timelock.sol

Locations

```
1 pragma solidity ^0.4.0;
2
3 /**
```

LOW

SWC-128

Loop over unbounded data structure.

Gas consumption in function "sqrt" in contract "SafeMath" depends on the size of data structures or values that may grow unboundedly. If the data structure grows too large, the gas required to execute the code will exceed 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

```
177 z = y;
178 uint256 x = y / 2 + 1;
179 while (x < z) {
180 z = x;
181 x = (y / x + x) / 2;
```

LOW

SWC-128

Potentially unbounded data structure passed to builtin.

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

```
304 callData = data;
305 } else {
306 callData = abi.encodePacked(bytes4(keccak256(bytes(signature))), data);
307 }
308
```