MythX

REPORT 60646027DD06F70011413027

Created Wed Mar 31 2021 11:42:31 GMT+0000 (Coordinated Universal Time)

Number of analyses 1

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
8ec4054b-7cf1-416f-9cce-71b648fcb105	/contracts/timelock.sol	7

Analysis 8ec4054b-7cf1-416f-9cce-71b648fcb105 MythX

Started	Wed Mar 31 2021 11:42:40 GMT+0000 (Coordinated Universal Time)
Finished	Wed Mar 31 2021 11:57:49 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Client Tool	Mythx-Vscode-Extension
Main Source File	/Contracts/Timelock.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	6	1

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

/contracts/timelock.sol

Locations

```
52
53  function setDelay(uint delay_) public {
54  require(msg.sender == address(this), "Timelock::setDelay: Call must come from Timelock.");
55  require(delay_ >= MINIMUM_DELAY, "Timelock::setDelay: Delay must exceed minimum delay.");
56  require(delay_ <= MAXIMUM_DELAY, "Timelock::setDelay: Delay must not exceed maximum delay.");
57  delay = delay_;
58
59  emit NewDelay(delay);
60  }
61
62  function acceptAdmin() public {
63  require(msg.sender == pendingAdmin, "Timelock::acceptAdmin: Call must come from pendingAdmin.");
64  admin = msg.sender;
65  pendingAdmin = address(0);
```

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

/contracts/timelock.sol

Locations

```
61
62  function acceptAdmin() public {
63  require(msg.sender == pendingAdmin, "Timelock::acceptAdmin: Call must come from pendingAdmin.");
64  admin = msg.sender;
65  pendingAdmin = address(0);
66
67  emit NewAdmin(admin);
68  }
69
70  function setPendingAdmin(address pendingAdmin_) public {
71  // allows one time setting of admin for deployment purposes
72  if (admin_initialized) {
73  require(msg.sender == address(this), "Timelock::setPendingAdmin: Call must come from Timelock.");
```

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

/contracts/timelock.sol

Locations

```
69
70  function setPendingAdmin(address pendingAdmin_) public {
71  // allows one time setting of admin for deployment purposes
72  if (admin_initialized) {
73  require(msg.sender == address(this), "Timelock::setPendingAdmin: Call must come from Timelock.");
74  } else {
75  require(msg.sender == admin, "Timelock::setPendingAdmin: First call must come from admin.");
76  admin_initialized = true;
77  }
78  pendingAdmin = pendingAdmin_;
79
80  emit NewPendingAdmin(pendingAdmin);
81  }
82
83  function queueTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public returns (bytes32) {
84  require(msg.sender == admin, "Timelock::queueTransaction: Call must come from admin.");
85  require(eta >= getBlockTimestamp().add(delay), "Timelock::queueTransaction: Estimated execution block must satisfy delay.");
```

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

/contracts/timelock.sol

Locations

```
81  }
82
83  function queueTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public returns (bytes32) {
84  require(msg.sender == admin, "Timelock::queueTransaction: Call must come from admin.");
85  require(eta >= getBlockTimestamp().add(delay), "Timelock::queueTransaction: Estimated execution block must satisfy delay.");
86
87  bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
88  queuedTransactions[txHash] = true;
89
90  emit QueueTransaction(txHash, target, value, signature, data, eta);
91  return txHash;
92  }
93
94  function cancelTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public {
95  require(msg.sender == admin, "Timelock::cancelTransaction: Call must come from admin.");
```

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

/contracts/timelock.sol

Locations

```
92  }
93
94  function cancelTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public {
95  require(msg.sender == admin, "Timelock::cancelTransaction: Call must come from admin.");
96
97  bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
98  queuedTransactions[txHash] = false;
99
100 emit CancelTransaction(txHash, target, value, signature, data, eta);
101 }
102
103 function executeTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public payable returns (bytes memory) {
104 require(msg.sender == admin, "Timelock::executeTransaction: Call must come from admin.");
```

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

/contracts/timelock.sol

Locations

```
101 }
102
103 function executeTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public payable returns (bytes memory) {
104 require(msg.sender == admin, "Timelock::executeTransaction: Call must come from admin.");
105
106 bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
107 require(queuedTransactions[txHash], "Timelock::executeTransaction: Transaction hasn't been queued.");
108 require(getBlockTimestamp() >= eta, "Timelock::executeTransaction: Transaction hasn't surpassed time lock.");
109 require(getBlockTimestamp() <= eta.add(GRACE_PERIOD), "Timelock::executeTransaction: Transaction is stale.");
110
111 queuedTransactions[txHash] = false;
112
113 bytes memory callData;
114
115 if (bytes(signature).length == 0) {
116 callData = data;
117 } else {
118 callData = abi.encodePacked(bytes4(keccak256(bytes(signature))), data);
119 }
120
121 // solium-disable-next-line security/no-call-value
122 (bool success, bytes memory returnData) = target.call.value(value)(callData);
123 require(success, "Timelock::executeTransaction: Transaction execution reverted.");
124
125 emit ExecuteTransaction(txHash, target, value, signature, data, eta);
126
127 return returnData;
128 }
129
130 function getBlockTimestamp() internal view returns (uint) {
131 // solium-disable-next-line security/no-block-members
132 return block.timestamp;
```

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

/contracts/timelock.sol

Locations

```
119 }
120
121 // solium-disable-next-line security/no-call-value
122 (bool success, bytes memory returnData) = target.call.value(value)(callData);
123 require(success, "Timelock::executeTransaction: Transaction execution reverted.");
```