MythX

REPORT 607F1A424E31260011C1DE1C

Created	Tue Apr 20 2021 18:15:30 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	607f174267db361c31bacfdc

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
623e9e36-fde3-45fb-a44c-abd77bc1c784	/contracts/timelock.sol	7

Analysis 623e9e36-fde3-45fb-a44c-abd77bc1c784 MythX

Started	Tue Apr 20 2021 18:15:33 GMT+0000 (Coordinated Universal Time)
Finished	Tue Apr 20 2021 18:17:38 GMT+0000 (Coordinated Universal Time)
Mode	Quick
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

```
78
79  function setDelay(uint delay_) public {
80  require(msg.sender == address(this), "Timelock::setDelay: Call must come from Timelock.");
81  require(delay_ >= MINIMUM_DELAY, "Timelock::setDelay: Delay must exceed minimum delay.");
82  require(delay_ <= MAXIMUM_DELAY, "Timelock::setDelay: Delay must not exceed maximum delay.");
83  delay = delay_;
84
85  emit NewDelay(delay);
86  }
87
88  function acceptAdmin() public {
89  require(msg.sender == pendingAdmin, "Timelock::acceptAdmin: Call must come from pendingAdmin.");
90  admin = msg.sender;
91  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

```
87
88  function acceptAdmin() public {
89  require(msg.sender == pendingAdmin, "Timelock::acceptAdmin: Call must come from pendingAdmin.");
90  admin = msg.sender;
91  pendingAdmin = address(0);
92
93  emit NewAdmin(admin);
94  }
95
96  function setPendingAdmin(address pendingAdmin_) public {
97  // allows one time setting of admin for deployment purposes
98  if (admin_initialized) {
99  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

```
95
96  function setPendingAdmin(address pendingAdmin_) public {
97  // allows one time setting of admin for deployment purposes
98  if (admin_initialized) {
99  require(msg.sender == address(this), "Timelock::setPendingAdmin: Call must come from Timelock.");
100 } else {
101 require(msg.sender == admin, "Timelock::setPendingAdmin: First call must come from admin.");
102 admin_initialized = true;
103 }
104 pendingAdmin = pendingAdmin_;
105
106 emit NewPendingAdmin(pendingAdmin);
107 }
108
109 function queueTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public returns (bytes32) {
110 require(msg.sender == admin, "Timelock::queueTransaction: Call must come from admin.");
111 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

```
107 }
108
109 function queueTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public returns (bytes32) {
110 require(msg.sender == admin, "Timelock::queueTransaction: Call must come from admin.");
111 require(eta >= getBlockTimestamp().add(delay), "Timelock::queueTransaction: Estimated execution block must satisfy delay.");
112
113 bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
114 queuedTransactions[txHash] = true;
115
116 emit QueueTransaction(txHash, target, value, signature, data, eta);
117 return txHash;
118 }
119
120 function cancelTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public {
121 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

```
118 }
119
120 function cancelTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public {
121 require(msg.sender == admin, "Timelock::cancelTransaction: Call must come from admin.");
122
123 bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
124 queuedTransactions[txHash] = false;
125
126 emit CancelTransaction(txHash, target, value, signature, data, eta);
127 }
128
129 function executeTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public payable returns (bytes memory) {
130 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

```
127 }
128
129 function executeTransaction(address target, uint value, string memory signature, bytes memory data, uint eta) public payable returns (bytes memory) {
130 require(msg.sender == admin, "Timelock::executeTransaction: Call must come from admin.");
131
132 bytes32 txHash = keccak256(abi.encode(target, value, signature, data, eta));
133 require(queuedTransactions[txHash], "Timelock::executeTransaction: Transaction hasn't been queued.");
134 require(getBlockTimestamp() >= eta, "Timelock::executeTransaction: Transaction hasn't surpassed time lock.");
135 require(getBlockTimestamp() <= eta.add(GRACE_PERIOD), "Timelock::executeTransaction: Transaction is stale.");
136
137 queuedTransactions[txHash] = false;
138
139 bytes memory callData;
140
141 if (bytes(signature).length == 0) {
142 callData = data;
143 } else {
144 callData = abi.encodePacked(bytes4(keccak256(bytes(signature))), data);
145 }
146
147 // solium-disable-next-line security/no-call-value
148 (bool success, bytes memory returnData) = target.call.value(value)(callData);
149 require(success, "Timelock::executeTransaction: Transaction execution reverted.");
150
151 emit ExecuteTransaction(txHash, target, value, signature, data, eta);
152
153 return returnData;
154 }
155
156 function getBlockTimestamp() internal view returns (uint) {
157 // solium-disable-next-line security/no-block-members
158 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

```
145 }
146
147 // solium-disable-next-line security/no-call-value
148 (bool success, bytes memory returnData) = target.call.value(value)(callData);
149 require(success, "Timelock::executeTransaction: Transaction execution reverted.");
```