MythX

REPORT 600F426ADFD70E00119EE9DC

Created	Mon Jan 25 2021 22:12:58 GMT+0000 (Coordinated Universal Time)
Number of analyses	1
User	contact@rubicon.finance

REPORT SUMMARY

Analyses ID	Main source file	Detected vulnerabilities
a653bc21-2138-49da-a2ca-de54b7bd0481	C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol	18

Analysis a653bc21-2138-49da-a2ca-de54b7bd0481 MythX

Started	Mon Jan 25 2021 22:13:05 GMT+0000 (Coordinated Universal Time)
Finished	Mon Jan 25 2021 22:59:21 GMT+0000 (Coordinated Universal Time)
Mode	Deep
Client Tool	Mythx-Cli-0.6.22
Main Source File	C:\Users\Benjamin Hughes\Workspace\Rubicon\Rubicon_protocol\Contracts\SenateAlpha.Sol

DETECTED VULNERABILITIES

HIGH	MEDIUM	LOW
0	12	6

ISSUES

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "propose" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
153 }
154
155 function propose(
156 address[] memory targets,
157 uint256[] memory values,
158 string[] memory signatures,
159 bytes[] memory calldatas,
160 string memory description
161 ) public returns (uint256) {
162 require(
163 RBCN.getPriorVotes(msg.sender, sub256(block.number, 1)) >
164 proposalThreshold(),
165 "SenateAlpha::propose: proposer votes below proposal threshold"
166 );
167 require(
168 targets.length == values.length &&
169 targets.length == signatures.length &&
170 targets.length == calldatas.length,
171 "SenateAlpha::propose: proposal function information arity mismatch"
172 );
173 require(
174 targets.length != 0,
175 "SenateAlpha::propose: must provide actions"
176 );
177 require(
178 targets.length <= proposalMaxOperations(),
179 "SenateAlpha::propose: too many actions"
180 );
181
182 uint256 latestProposalId = latestProposalIds[msg.sender];
183 if (latestProposalId != 0) {
184 ProposalState proposersLatestProposalState =
185 state(latestProposalId);
186 require(
187 proposersLatestProposalState != ProposalState.Active,
188 "SenateAlpha::propose: one live proposal per proposer, found an already active proposal"
189 );
190 require(
191 proposersLatestProposalState != ProposalState.Pending,
192 "SenateAlpha::propose: one live proposal per proposer, found an already pending proposal"
193 );
194 }
195
196 uint256 startBlock = add256(block.number, votingDelay());
197 uint256 endBlock = add256(startBlock, votingPeriod());
198
199 proposalCount++;
200 Proposal memory newProposal =
201 Proposal({
202 id: proposalCount,
203 proposer: msg.sender,
204 eta: 0,
205 targets: targets,
206 values: values,
207 signatures: signatures,
```

```
        calldatas: calldatas,
        startBlock: startBlock,
        endBlock: endBlock,
        forVotes: 0,
        againstVotes: 0,
        canceled: false,
        executed: false
    });

    proposals[newProposal.id] = newProposal;
    latestProposalIds[newProposal.proposer] = newProposal.id;

    emit ProposalCreated(
    newProposal.id,
    msg.sender,
    targets,
    values,
    signatures,
    calldatas,
    startBlock,
    endBlock,
    description
    );
    return newProposal.id;
    }

    function queue(uint256 proposalId) public {
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "queue" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
232 }
233
234 function queue(uint256 proposalId) public {
235 require(
236 state(proposalId) == ProposalState.Succeeded,
237 "SenateAlpha::queue: proposal can only be queued if it is succeeded"
238 );
239 Proposal storage proposal = proposals[proposalId];
240 uint256 eta = add256(block.timestamp, timelock.delay());
241 for (uint256 i = 0; i < proposal.targets.length; i++) {
242 _queueOrRevert(
243 proposal.targets[i],
244 proposal.values[i],
245 proposal.signatures[i],
246 proposal.calldatas[i],
247 eta
248 );
249 }
250 proposal.eta = eta;
251 emit ProposalQueued(proposalId, eta);
252 }
253
254 function _queueOrRevert(
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "execute" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
268 }
269
270 function execute(uint256 proposalId) public payable {
271 require(
272 state(proposalId) == ProposalState.Queued,
273 "SenateAlpha::execute: proposal can only be executed if it is queued"
274 );
275 Proposal storage proposal = proposals[proposalId];
276 proposal.executed = true;
277 for (uint256 i = 0; i < proposal.targets.length; i++) {
278 timelock.executeTransaction.value(proposal.values[i])(
279 proposal.targets[i],
280 proposal.values[i],
281 proposal.signatures[i],
282 proposal.calldatas[i],
283 proposal.eta
284 );
285 }
286 emit ProposalExecuted(proposalId);
287 }
288
289 function cancel(uint256 proposalId) public {
```

MEDIUM Function could be marked as external.

SWC-000 The function definition of "cancel" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
287 }
288
289 function cancel(uint256 proposalId) public {
290 ProposalState state = state(proposalId);
291 require(
292 state != ProposalState.Executed,
293 "SenateAlpha::cancel: cannot cancel executed proposal"
294 );
295
296 Proposal storage proposal = proposals[proposalId];
297 require(
298 msg.sender == guardian ||
299 RBCN.getPriorVotes(proposal.proposer, sub256(block.number, 1)) <
300 proposalThreshold(),
301 "SenateAlpha::cancel: proposer above threshold"
302 );
303
304 proposal.canceled = true;
305 for (uint256 i = 0; i < proposal.targets.length; i++) {
306 timelock.cancelTransaction(
307 proposal.targets[i],
308 proposal.values[i],
309 proposal.signatures[i],
310 proposal.calldatas[i],
311 proposal.eta
312 );
313 }
314
315 emit ProposalCanceled(proposalId);
316 }
317
318 function getActions(uint256 proposalId)
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "getActions" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
316 }
317
318 function getActions(uint256 proposalId)
319 public
320 view
321 returns (
322 address[] memory targets,
323 uint256[] memory values,
324 string[] memory signatures,
325 bytes[] memory calldatas
326 )
327 {
328 Proposal storage p = proposals[proposalId];
329 return (p.targets, p.values, p.signatures, p.calldatas);
330 }
331
332 function getReceipt(uint256 proposalId, address voter)
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "getReceipt" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
330 }
331
332 function getReceipt(uint256 proposalId, address voter)
333 public
334 view
335 returns (Receipt memory)
336 {
337 return proposals[proposalId].receipts[voter];
338 }
339
340 function state(uint256 proposalId) public view returns (ProposalState) {
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "castVote" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
368 }
369
370 function castVote(uint256 proposalId, bool support) public {
371 return _castVote(msg.sender, proposalId, support);
372 }
373
374 function castVoteBySig(
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "castVoteBySig" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
372 }
373
374 function castVoteBySig(
375 uint256 proposalId,
376 bool support,
377 uint8 v,
378 bytes32 r,
379 bytes32 s
380 ) public {
381 bytes32 domainSeparator =
382 keccak256(
383 abi.encode(
384 DOMAIN_TYPEHASH,
385 keccak256(bytes(name)),
386 getChainId(),
387 address(this)
388 )
389 );
390 bytes32 structHash =
391 keccak256(abi.encode(BALLOT_TYPEHASH, proposalId, support));
392 bytes32 digest =
393 keccak256(
394 abi.encodePacked("\x19\x01", domainSeparator, structHash)
395 );
396 address signatory = ecrecover(digest, v, r, s);
397 require(
398 signatory != address(0),
399 "SenateAlpha::castVoteBySig: invalid signature"
400 );
401 return _castVote(signatory, proposalId, support);
402 }
403
404 function _castVote(
```

MEDIUM

SWC-000

Function could be marked as external.

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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
432 }
433
434 function __acceptAdmin() public {
435 require(
436 msg.sender == guardian,
437 "SenateAlpha::__acceptAdmin: sender must be gov guardian"
438 );
439 timelock.acceptAdmin();
440 }
441
442 function __abdicate() public {
```

MEDIUM

SWC-000

Function could be marked as external.

The function definition of "__abdicate" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
440 }
441
442 function __abdicate() public {
443 require(
444 msg.sender == guardian,
445 "SenateAlpha::__abdicate: sender must be gov guardian"
446 );
447 guardian = address(0);
448 }
449
450 function __queueSetTimelockPendingAdmin(
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "__queueSetTimelockPendingAdmin" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
448 }
449
450 function __queueSetTimelockPendingAdmin(
451 address newPendingAdmin,
452 uint256 eta
453 ) public {
454 require(
455 msg.sender == guardian,
456 "SenateAlpha::__queueSetTimelockPendingAdmin: sender must be gov guardian"
457 );
458 timelock.queueTransaction(
459 address(timelock),
460 0,
461 "setPendingAdmin(address)",
462 abi.encode(newPendingAdmin),
463 eta
464 );
465 }
466
467 function __executeSetTimelockPendingAdmin(
```

MEDIUM

Function could be marked as external.

SWC-000

The function definition of "__executeSetTimelockPendingAdmin" 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
465 }
466
467 function __executeSetTimelockPendingAdmin(
468 address newPendingAdmin,
469 uint256 eta
470 ) public {
471 require(
472 msg.sender == guardian,
473 "SenateAlpha::__executeSetTimelockPendingAdmin: sender must be gov guardian"
474 );
475 timelock.executeTransaction(
476 address(timelock),
477 0,
478 "setPendingAdmin(address)",
479 abi.encode(newPendingAdmin),
480 eta
481 );
482 }
483
484 function add256(uint256 a, uint256 b) internal pure returns (uint256) {
```

LOW

SWC-103

A floating pragma is set.

The current pragma Solidity directive is ""^0.5.16"". 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
1 pragma solidity ^0.5.16;
2 pragma experimental ABIEncoderV2;
```

LOW

SWC-120

Potential use of "block.number" as source of randonmness.

The environment variable "block.number" looks like it might be used as a source of randomness. 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
161 ) public returns (uint256) {
162 require(
163 RBCN.getPriorVotes(msg.sender, sub256(block.number, 1)) >
164 proposalThreshold(),
165 "SenateAlpha::propose: proposer votes below proposal threshold"
```

LOW

SWC-120

Potential use of "block.number" as source of randonmness.

The environment variable "block.number" looks like it might be used as a source of randomness. 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
194 }
195
196 uint256 startBlock = add256(block.number, votingDelay());
197 uint256 endBlock = add256(startBlock, votingPeriod());
```

LOW

SWC-120

Potential use of "block.number" as source of randonmness.

The environment variable "block.number" looks like it might be used as a source of randomness. 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
297 require(
298 msg.sender == guardian ||
299 RBCN.getPriorVotes(proposal.proposer, sub256(block.number, 1)) <
300 proposalThreshold(),
301 "SenateAlpha::cancel: proposer above threshold"
```

LOW

SWC-120

Potential use of "block.number" as source of randonmness.

The environment variable "block.number" looks like it might be used as a source of randomness. 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
346 if (proposal.canceled) {
347 return ProposalState.Canceled;
348 } else if (block.number <= proposal.startBlock) {
349 return ProposalState.Pending;
350 } else if (block.number <= proposal.endBlock) {
```

LOW

SWC-120

Potential use of "block.number" as source of randonmness.

The environment variable "block.number" looks like it might be used as a source of randomness. 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

C:\Users\Benjamin Hughes\workspace\rubicon\rubicon_protocol\contracts\SenateAlpha.sol

Locations

```
348 } else if (block.number <= proposal.startBlock) {
349 return ProposalState.Pending;
350 } else if (block.number <= proposal.endBlock) {
351 return ProposalState.Active;
352 } else if (
```