6756bc71a47c0217e08b6b884dc2ba805ed6923045d44bbc13b7908bdd6eb5(

File: SmartChefInitia | Language:solidity | Size:11886 bytes | Date:2022-06-15T16:13:33.569Z

Critical	High	Medium	Low	Note
0	0	0	2	11

Issues

Severity	Issue	Analyzer	Code Lines
Low	SWC-100	Achilles	73 - 75
Low	SWC-103	Achilles	2
Note	SWC-116	Achilles	231, 258, 270, 272, 291, 292, 305, 312, 316, 319, 341

Code

1. SWC-100 / lines: 73 - 75 Low Achilles

```
A security vulnerability has been detected.

72      */
73      constructor() {
74          SMART_CHEF_FACTORY = msg.sender;
75      }
76
```

In detail

Functions that do not have a function visibility type specified are public by default. This can lead to a vulnerability if a developer forgot to set the visibility and a malicious user is able to make unauthorized or unintended state changes.

2. SWC-103 / lines: 2 Low Achilles

```
A security vulnerability has been detected.

1   // SPDX-License-Identifier: MIT
2   pragma solidity ^0.8.0;
3   pragma abicoder v2;
```

In detail

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

3. SWC-116 / lines: 231 Note Achilles

```
A security vulnerability has been detected.

230     function stopReward() external onlyOwner {
231         bonusEndBlock = block.number;
232     }
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some

degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

4. SWC-116 / lines: 258 Note Achilles

A security vulnerability has been detected.

```
257     function updateRewardPerBlock(uint256 _rewardPerBlock) external onlyOwner {
258         require(block.number < startBlock, "Pool has started");
259         rewardPerBlock = _rewardPerBlock;
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

5. SWC-116 / lines: 270 Note Achilles

A security vulnerability has been detected.

```
269     function updateStartAndEndBlocks(uint256 _startBlock, uint256 _bonusEndBlock) external onlyOwner {
270         require(block.number < startBlock, "Pool has started");
271         require(_startBlock < _bonusEndBlock, "New startBlock must be lower than new endBlock");
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

6. SWC-116 / lines: 272 Note Achilles

A security vulnerability has been detected.

```
271         require(_startBlock < _bonusEndBlock, "New startBlock must be lower than new endBlock");
272         require(block.number < _startBlock, "New startBlock must be higher than current block");
273
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

7. SWC-116 / lines: 291 Note Achilles

A security vulnerability has been detected.

```
290         uint256 stakedTokenSupply = stakedToken.balanceOf(address(this));
291         if (block.number > lastRewardBlock && stakedTokenSupply != 0) {
292             uint256 multiplier = _getMultiplier(lastRewardBlock, block.number);
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

8. SWC-116 / lines: 292 Note Achilles

A security vulnerability has been detected.

```
291         if (block.number > lastRewardBlock && stakedTokenSupply != 0) {
292             uint256 multiplier = _getMultiplier(lastRewardBlock, block.number);
293             uint256 tokenReward = multiplier * rewardPerBlock;
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

9. SWC-116 / lines: 305 Note Achilles

A security vulnerability has been detected.

```
304     function _updatePool() internal {
305         if (block.number <= lastRewardBlock) {
306             return;
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

10. SWC-116 / lines: 312 Note Achilles

A security vulnerability has been detected.

```
311         if (stakedTokenSupply == 0) {
312             lastRewardBlock = block.number;
313             return;
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

11. SWC-116 / lines: 316 Note Achilles

A security vulnerability has been detected.

```
315
316         uint256 multiplier = _getMultiplier(lastRewardBlock, block.number);
317         uint256 tokenReward = multiplier * rewardPerBlock;
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

12. SWC-116 / lines: 319 Note Achilles

A security vulnerability has been detected.

```
318         accTokenPerShare = accTokenPerShare + (tokenReward * PRECISION_FACTOR) / stakedTokenSupply;
319         lastRewardBlock = block.number;
320     }
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.

13. SWC-116 / lines: 341 Note Achilles

A security vulnerability has been detected.

```
340     function hasUserLimit() public view returns (bool) {
341         if (!userLimit || (block.number >= (startBlock + numberBlocksForUserLimit))) {
342             return false;
```

In detail

Contracts often need access to the current timestamp to trigger time-dependent events. As Ethereum is decentralized, nodes can synchronize time only to some degree. Moreover, malicious miners can alter the timestamp of their blocks, especially if they can gain advantages by doing so. However, miners can't set timestamp smaller than the previous one (otherwise the block will be rejected), nor can they set the timestamp too far ahead in the future. Taking all of the above into consideration, developers can't rely on the preciseness of the provided timestamp.