Security Audit

Name: BNB Farm

Date: 04/20/2021

Contract: https://bscscan.com/address/0x3383dC1B85863b662b0f9420C849a6C527a74E4c

https://lemonsec.com

Lemon Sec

Issues Checking Status

Compiler Errors: **PASSED**

Race conditions: **PASSED**

Possible delays in data delivery: **PASSED**

Oracle calls: **PASSED**

Front running: **PASSED**

Timestamp dependence: **PASSED**

Integer Overflow and Underflow: **PASSED**

DoS with Revert: **PASSED**

Economy model of the contract: **PASSED**

Methods execution permissions: **PASSED**

DoS with block gas limit: **1 LOW ISSUES**

The impact of the exchange rate on the logic: **PASSED**

Private user data leaks: **PASSED**

Malicious Event log: **PASSED**

Issues Checking Status

Scoping and Declarations: **PASSED**

Uninitialized storage pointers: **PASSED**

Arithmetic accuracy: **PASSED**

Design Logic: **PASSED**

Cross-function race conditions: **PASSED**

Fallback function security: **PASSED**

Security Issues

CriticalSeverity Issues: 0

High Severity Issues: 0

Medium Severity Issues: 0

Low Severity Issues: 2

The function **forceWithdraw()** uses the loop to find and remove deposits from the user deposits list. Function will be aborted with OUT_OF_GAS exception if there will be a long excluded deposits list.

```
function forceWithdraw(uint256 index) public beforeStarted() {
    User storage user = users[msg.sender];

    require(index < user.deposits.length, "Invalid index");
    require(user.deposits[index].force == true, "Force is false");

    uint256 depositAmount = user.deposits[index].amount;
    uint256 penaltyAmount =
        depositAmount.mul(PENALTY_STEP).div(PERCENTS_DIVIDER);
    msg.sender.transfer(depositAmount.sub(penaltyAmount));

    emit FeePayed(msg.sender, penaltyAmount);

    penaltyDeposits[msg.sender].push(user.deposits[index]);

    user.deposits[index] = user.deposits[user.deposits.length - 1];
    user.deposits.pop();
    emit ForceWithdrawn(
        msg.sender,
        depositAmount,
        penaltyAmount,
        penaltyDeposits[msg.sender].length
    );
}
```

Security Issues

Recommendation:

Use EnumerableSet instead of array or do not use long arrays.

No vulnerabilities were found.
No Backdoors or Scam Scripts found.
The Smart Contract is Safe to use in the Binance Smart Chain.

Conclusion

APPROVED

No vulnerabilities were found.
No Backdoors or Scam Scripts found.
The Smart Contract is Safe to use in the Binance Smart Chain.

If you are interested in Auditing your project contact us on:

https://t.me/lemonsec

https://lemonsec.com

https://twitter.com/lemon_sec