Security Audit Report

PureFi Farming Contracts

24/01/2022

PREPARED FOR:
PureFi
https://purefi.io/

ARCADIA CONTACT INFO
Email: audits@arcadiamgroup.com
Telegram: https://t.me/thearcadiagroup

Table of Contents

Executive Summary **2**

Findings **3**

Missing check 3

Action Recommended: 3

Only call the Token Buyer contract with a msg.value>0, or require commissionAmount and buyTokenAddress to be both non zero when updated. 3

ETH may get stuck in contract 4

Action Recommended: 4

Send back msg.value to the user if the buyTokenAddress is not set. 4

Gas optimization 4

Action Recommended: 5

Remove the if check and only require(msg.value >= pool.commissionAmount) 5

Adding an existent LP token 5

Action Recommended: 6

Track added LP tokens and check whether the parameter _lpToken has already been already added to the pool or not. 6

Conclusion **7**

Disclaimer **7**

Executive Summary

A Representative Party of **PUREFI** ("**CLIENT**") engaged The Arcadia Group ("Arcadia"), a software development, research, and security company, to conduct a review of the following **PUREFIPROTOCOL** smart contracts on the **TOKEN** github repository at Commit #**4dc26b682007f1220e91e498ca780617a8f4cc16**.

The scope of this audit included the following files:

1. **PureFiFarming2**.sol

Arcadia completed this security review using various methods primarily consisting of dynamic and static analysis. This process included a line-by-line analysis of the in-scope contracts, optimization analysis, analysis of key functionalities and limiters, and reference against intended functionality.

There were **4** issues found, **0** of which were deemed to be 'critical', and **0** of which were rated as 'high'.

Severity Rating	Number of Original Occurrences	Number of Remaining Occurrences
CRITICAL	0	0
HIGH	0	0
MEDIUM	0	0
LOW	4	4
INFORMATIONAL	0	0

Findings

1. Missing check

Issue: **PUREFIFARMING2**-1
Severity: **LOW**
Likelihood: **LOW**
Impact: **HIGH**

Target: **PureFiFarming2.sol**
Category: **LOW**
Finding Type: **DYNAMIC**

If a pool's commission amount is set to 0 but already has a buyTokenAddress (!=0), the buy Token function will revert if the user doesn't send a msg.value > 0.

While the user shouldn't be required to send funds while depositing and no commission is set, in this case, he won't be able to deposit his LP tokens to the pool with a msg.value=0.

```
    // Deposit LP tokens to PureFiFarming for Token allocation.
    function depositTo(uint16 _pid, uint256 _amount, address _beneficiary) public payable override whenNotPaused {
        PoolInfo storage pool = poolInfo[_pid];
        UserInfo storage user = userInfo[_pid][_beneficiary];
        require(_amount + user.amount <= maxStakingAmountForPool[_pid], "Deposited amount exceeded limits for this pool");
        if(pool.commissionAmount > 0) {
            require(msg.value >= pool.commissionAmount, "Insufficient commission sent");
        }

        uint256 tokensOutAmount = 0;

        if(pool.buyTokenAddress!= address(0)){
            require(tokenBuyer != address(0),"Token buyer not set");
            tokensOutAmount = ITokenBuyer(tokenBuyer).buyToken{value:msg.value}(pool.buyTokenAddress, _beneficiary);
        }

        emit DepositCommission(_beneficiary, _pid, msg.value, pool.buyTokenAddress, tokensOutAmount);
```

```
    function updatePoolStakingCommission(uint16 _pid, address _buyTokenAddress, uint256 _commission) public onlyManager {
        poolInfo[_pid].buyTokenAddress = _buyTokenAddress;
        poolInfo[_pid].commissionAmount = _commission;
    }
```

Action Recommended:

Only call the Token Buyer contract with a msg.value>0, or require commissionAmount and buyTokenAddress to be both non zero when updated.

2. ETH may get stuck in contract

Issue: **PUREFIFARMING2**-2
Severity: **LOW**
Likelihood: **LOW**
Impact: **HIGH**

Target: **PureFiFarming2.sol**
Category: **LOW**
Finding Type: **DYNAMIC**

When a pool's "buyTokenAddress" is not set, the payable depositTo function won't purchase tokens for the user and thus the value sent by the user (if any) will get stuck in the contract and forever lost.

```
237     // Deposit LP tokens to PureFiFarming for Token allocation.
238     function depositTo(uint16 _pid, uint256 _amount, address _beneficiary) public payable override whenNotPaused {
239         PoolInfo storage pool = poolInfo[_pid];
240         UserInfo storage user = userInfo[_pid][_beneficiary];
241         require(_amount + user.amount <= maxStakingAmountForPool[_pid], "Deposited amount exceeded limits for this pool");
242         if(pool.commissionAmount > 0) {
243             require(msg.value >= pool.commissionAmount, "Insufficient commission sent");
244         }
245 
246         uint256 tokensOutAmount = 0;
247 
248         if(pool.buyTokenAddress!= address(0)){
249             require(tokenBuyer != address(0),"Token buyer not set");
250             tokensOutAmount = ITokenBuyer(tokenBuyer).buyToken{value:msg.value}(pool.buyTokenAddress, _beneficiary);
251         }
252 
253         emit DepositCommission(_beneficiary, _pid, msg.value, pool.buyTokenAddress, tokensOutAmount);
254 
255         updatePool(_pid);
256         if (user.amount > 0) {
257             user.pendingReward += user.amount * pool.accTokenPerShare / 1e12 - user.rewardDebt;
258         }
259         if(_amount > 0) {
260             pool.lpToken.safeTransferFrom(address(msg.sender), address(this), _amount);
261             user.amount += _amount;
262             pool.totalDeposited += _amount;
263         }
264         user.rewardDebt = user.amount * pool.accTokenPerShare / 1e12;
265         userStakedTime[_pid][_beneficiary] = uint64(block.timestamp); //save last user staked time;
266         emit Deposit(_beneficiary, _pid, _amount);
267     }
268 
```

Action Recommended:

Send back msg.value to the user if the buyTokenAddress is not set.

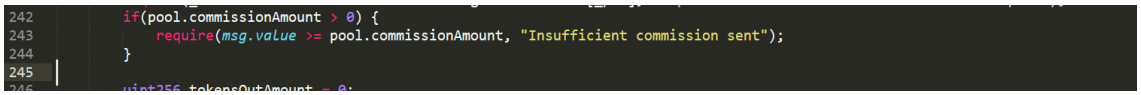
3. Gas optimization

Issue: **PUREFIFARMING2**-3
Severity: **LOW**
Likelihood: **LOW**
Impact: **LOW**

Target: **PureFiFarming2.sol**
Category: **LOW**
Finding Type: **DYNAMIC**

In the depositTo function, there's no need to check whether the pool commission amount is set or not, the 'require' statement will always be enough even if the commission is 0. The if check will only cost more gas to the function caller.

Note: Even if the pool has no commission, the user can still buy or not tokens.



Action Recommended:

Remove the if check and only require(msg.value >= pool.commissionAmount)

4. Adding an existent LP token

Issue: **PUREFIFARMING2**-4
Severity: **LOW**
Likelihood: **LOW**
Impact: **HIGH**

Target: **PureFiFarming2.sol**
Category: **LOW**
Finding Type: **DYNAMIC**

While a comment preceding this function is present in the smart contract showing the awareness of this issue, no checks have been made.

```
149     // Add a new lp to the pool. Can only be called by the owner.
150     // XXX DO NOT add the same LP token more than once. Rewards will be messed up if you do.
151     function addPool(uint64 _allocPoint, address _lpTokenAddress, uint64 _startBlock, uint64 _endBlock, uint64 _minStakingTime, uint256 _maxStak
152         require (block.number < _endBlock, "Incorrect endblock number");
153         IERC20Upgradeable _lpToken = IERC20Upgradeable(_lpTokenAddress);
154         if (_withUpdate) {
155             massUpdatePools();
156         }
157         uint64 lastRewardBlock = block.number > _startBlock ? uint64(block.number) : _startBlock;
158         totalAllocPoint += _allocPoint;
159         poolInfo.push(PoolInfo({
160             lpToken: _lpToken,
161             allocPoint: _allocPoint,
162             startBlock: _startBlock,
163             endBlock: _endBlock,
164             lastRewardBlock: lastRewardBlock,
165             accTokenPerShare: 0,
166             totalDeposited: 0,
167             buyTokenAddress: address(0),
168             commissionAmount: 0
169         }));
170         minStakingTimeForPool[uint16(poolInfo.length-1)] = _minStakingTime;
171         maxStakingAmountForPool[uint16(poolInfo.length-1)] = _maxStakingAmount;
172 
173         emit PoolAdded(poolInfo.length-1);
174     }
```

Action Recommended:

Track added LP tokens and check whether the parameter _lpToken has already been already added to the pool or not.

Conclusion

Arcadia identified issues that occurred at hash **#4dc26b682007f1220e91e498ca780617a8f4cc16**.

Disclaimer

While best efforts and precautions have been taken in the preparation of this document, The Arcadia Group and the Authors assume no responsibility for errors, omissions, or damages resulting from the use of the provided information. Additionally, Arcadia would like to emphasize that the use of Arcadia's services does not guarantee the security of a smart contract or set of smart contracts and does not guarantee against attacks. One audit on its own is not enough for a project to be considered secure; that categorization can only be earned through extensive peer review and battle testing over an extended period.