CERTIK

Security Assessment

Lucky Lion Audit 2

Dec 2nd, 2021

Table of Contents

Summary

This report has been prepared for Lucky Lion to discover issues and vulnerabilities in the source code of the Lucky Lion Audit 2 project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Lucky Lion Audit 2
Description	Revenue Sharing Pool
Platform	BSC
Language	Solidity
Codebase	https://github.com/LuckyLionIO/LuckyLion-RevenueSharing/blob/main/contracts/RevenueSharingPool.sol
Commit	a1d134a9367191b04b2def9a84fc3734c3752dca c87022db0fd23de93cc367edfa5cfdd676265ec9 5b9970794f29d562ad1f219b646bb64afceadfb5

Audit Summary

Delivery Date	Dec 02, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

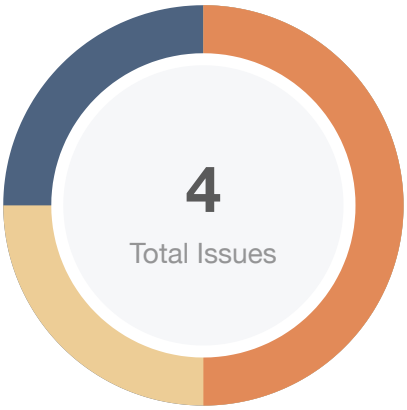
Vulnerability Summary

Vulnerability Level	Total	Pending	Declined	Acknowledged	Partially Resolved	Resolved
Critical	0	0	0	0	0	0
Major	2	0	0	1	0	1
Medium	0	0	0	0	0	0
Minor	1	0	0	0	0	1
Informational	1	0	0	1	0	0
Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
RSR	RevenueSharingPool.sol	6b854390d2caf6fbfa8bcdfedb8e88fe88d67a0144b0158047941161655ad841

Findings



Critical	0 (0.00%)
Major	2 (50.00%)
Medium	0 (0.00%)
Minor	1 (25.00%)
Informational	1 (25.00%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
RSR-01	Centralization Risk in `depositRevenue()`	**Centralization / Privilege**	**Major**	Acknowledged
RSR-02	`removeStakeholder()` not called in function `emergencyWithdraw()`	Logical Issue	Minor	Resolved
RSR-03	Reward token not trasferred to user in function `claimReward()`	Logical Issue	Major	Resolved
RSR-04	Local Variable Shadowing	Volatile Code	Informational	Acknowledged

RSR-01 | Centralization Risk in `depositRevenue()`

Category	Severity	Location	Status
Centralization / Privilege	Major	LuckyLion/contracts/RevenueSharingPool/RevenueSharingPool.sol (e97bf75): 164~177	Acknowledged

Description

In the contract `RevenueSharingPool`, the role `owner` has the authority over the following function:

- `updateMaxDate()`
- `addWhitelist()`
- `removeWhitelist()`
- `depositRevenue()`

The parameters of a revenue-sharing round are completely controlled by the whitelisted addresses through the privileged function `depositRevenue()`. This function allows the caller to create a new round of revenue sharing, which will prevent users to deposit to old sharing rounds.

```
1   function depositRevenue(
2         string memory symbol,
3         uint256 amount,
4         uint256 winLoss,
5         uint256 TPV,
6         uint256 percentOfRevshare
7     ) external isWhitelisted(msg.sender) {
8         uint256 roundId = getCurrentRoundId();
9         totalLuckyRevenue[roundId] += amount;
10        updatePoolInfo(winLoss, TPV, symbol, amount, percentOfRevshare, roundId); // update
round pool info
11        START_ROUND_DATE = block.timestamp;
12        updateRoundId();
13        uint256 currentRoundId = getCurrentRoundId();
14        updateTotalStake(currentRoundId); // update new round total stake
15        emit DistributeLuckyRevenue(msg.sender, address(this), amount);
16    }
```

Any compromise to the `owner` or `whitelisted` accounts may allow the hacker to take advantage of this and disrupt the operation of the pool. E.g., block a high APY revenue sharing round by calling `depositRevenue()`.

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level in term of short-term and long-term:

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key;
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.

Alleviation

[Lucky Lion Team]:

Acknowledge: This project has one owner. We have security to protect the private key and the seed of owner's wallet. We have announcement to the community before depositRevenue() every week. This case doesn't have any impact on the LP of users.

RSR-02 | `removeStakeholder()` not called in function `emergencyWithdraw()`

Category	Severity	Location	Status
Logical Issue	Minor	LuckyLion/contracts/RevenueSharingPool/RevenueSharingPool.sol (e97bf75): 120	Resolved

Description

Function `removeStakeholder()` is not called in `emergencyWithdraw`, which conflicts with the counterpart in function `withdrawToken()`

```
1   function withdrawToken() external {
2         ...
3         uint256 amount = user.amount;
4         user.amount = 0;
5         removeStake(roundId);
6         removeStakeholder(msg.sender);
7         luckyBusd.safeTransfer(msg.sender, amount);
8         emit WithdrawStake(msg.sender, amount, block.timestamp);
9     }
10
11  function emergencyWithdraw() external {
12        ...
13        uint256 amount = user.amount;
14        user.amount = 0;
15        user.lastUpdateRoundId = roundId;
16        removeStake(roundId);
17        luckyBusd.safeTransfer(msg.sender, amount);
18        emit WithdrawStake(msg.sender, amount, block.timestamp);
19    }
```

Recommendation

We recommend the client calling `removStakeHolder()` in function emergencyWithdraw() to keep consistent.

Alleviation

LuckyLion team has resolved this issue as suggested in commit `c87022db0fd23de93cc367edfa5cfdd676265ec9`

RSR-03 | Reward token not trasferred to user in function `claimReward()`

Category	Severity	Location	Status
Logical Issue	Major	LuckyLion/contracts/RevenueSharingPool/RevenueSharingPool.sol (e97bf75): 140	Resolved

Description

In the current implementation of function `claimReward()` there is no reward token transfer related funtion, which means users can not actually receive their reward by calling `claimReward()`.

```
126    function claimReward() external {
127          UserInfo storage user = userInfo[msg.sender];
128          uint256 roundId = getCurrentRoundId();
129
130          if (user.amount > 0) {
131              if (!isStakeUpToDate(roundId)) {
132                 updatePendingStake();
133              }
134              updatePendingReward();
135          }
136
137          uint256 claimableLuckyReward = user.pendingReward;
138          require(claimableLuckyReward > 0, "Not enough claimable LUCKY reward!");
139          user.rewardDept += claimableLuckyReward;
140          user.pendingReward -= claimableLuckyReward;
141          emit ClaimReward(msg.sender, claimableLuckyReward, block.timestamp);
142      }
```

Recommendation

We advise the team to explain explicitly how the reward tokens will be distributed to the user.

Alleviation

[Lucky Lion Team]:

Business logic: The revenue sharing pool pay the reward to users in form of Credit to play a game. When users claim rewards, the smart contract will sending the reward amount to Backend to generate Credit. Users can change the credit to Lucky token via Withdraw function in Game System.

RSR-04 | Local Variable Shadowing

Category	Severity	Location	Status
Volatile Code	Informational	LuckyLion/contracts/RevenueSharingPool/RevenueSharingPool.sol (e97bf75): 24	Acknowledged

Description

The linked variable `totalLuckyRevenue` shadows with `PoolInfo.totalLuckyRevenue`.

```
24  mapping(uint256 => uint256) public totalLuckyRevenue
```

```
40    struct PoolInfo {
41          uint256 winLoss;
42          uint256 TPV;
43          string symbol; // must be changes (waiting for P'Book to confirm with customer)
44          uint256 TVL;
45          uint256 percentOfRevshare;
46          uint256 participants;
47          uint256 totalLuckyRevenue;
48      }
```

Recommendation

Consider renaming the linked variables to avoid shadowing.

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.