Security Assessment

Galaxy Finance

Jun 3rd, 2021

Table of Contents

Summary

This report has been prepared for Galaxy Finance smart contracts, to discover issues and vulnerabilities in the source code of their Smart Contract 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 given they are currently missing in the repository;
- Provide more comments per each function for readability, especially contracts are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Galaxy Finance
Description	Galaxy Finance is the innovative blockchain financial product with "stranger trust network" as its underlying architecture.
Platform	Ethereum
Language	Solidity
Codebase	https://github.com/GFC-Eco/gfc-doucuments/blob/main/code/LpPool.full.sol
Commits	<1a121e86132b4adb8d478ed8cff08885afb0d54b> <c90f7b346e1076e07aab0ea67886402b8b49270a>

Audit Summary

Delivery Date	Jun 03, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	

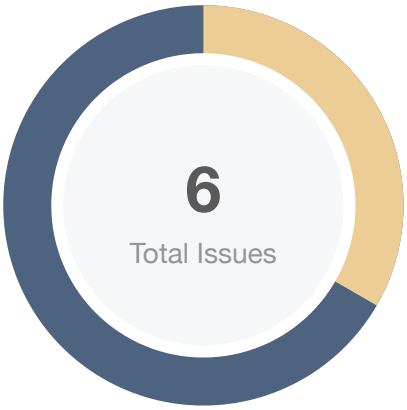
Vulnerability Summary

Total Issues	6
Critical	0
Major	0
Medium	0
Minor	2
Informational	4
Discussion	0

Audit Scope

ID	file	SHA256 Checksum
LPG	LpPool.full.sol	01f348127650b45fef753bf387fc4cf13be31b3392dd6f4b16dc734edc9ffa19

Findings



Critical	0 (0.00%)
Major	0 (0.00%)
Medium	0 (0.00%)
Minor	2 (33.33%)
Informational	4 (66.67%)
Discussion	0 (0.00%)

ID	Title	Category	Severity	Status
LPG-01	Lack of Input Validation	Volatile Code	Informational	Resolved
LPG-02	Risk For Weak Randomness	Logical Issue	Minor	Acknowledged
LPG-03	Redundant code	Logical Issue	Informational	Resolved
LPG-04	Missing Emit Events	Optimization	Informational	Resolved
LPG-05	SafeMath Not Used	Mathematical Operations	Informational	Resolved
LPG-06	Potential Calculation Overflow	Volatile Code	Minor	Resolved

LPG-01 | Lack of Input Validation

Category	Severity	Location	Status
Volatile Code	Informational	LpPool.full.sol: 630~632	Resolved

Description

The assigned value to `_lp` in the constructor of `LpPool.full.sol` should be verified as a non-zero value to prevent error.

The assigned value to `_gfv` in the constructor of `LpPool.full.sol` should be verified as a non-zero value to prevent error.

The assigned value to `_nft` in the constructor of `LpPool.full.sol` should be verified as a non-zero value to prevent error.

Recommendation

Check that the passed-in values are non-zero values.

Example:

```
require(lp != address(0), "lp is a zero value");
require(gfv != address(0), "gfv is a zero value");
require(nft != address(0), "nft is a zero value");
```

Alleviation

The development team heeded our advice and resolved this issue in commit c90f7b346e1076e07aab0ea67886402b8b49270a.

LPG-02 | Risk For Weak Randomness

Category	Severity	Location	Status
Logical Issue	Minor	LpPool.full.sol: 799	Acknowledged

Description

It is now well understood that the difficulty or timestamp is not a source of randomness. They can be manipulated if the attacker is also a miner.

Recommendation

Consider mixing a seed value based on the chainlink random service(https://docs.chain.link/docs/get-a-random-number/).

Alleviation

No alleviation.

LPG-03 | Redundant code

Category	Severity	Location	Status
Logical Issue	Informational	LpPool.full.sol: 674	Resolved

Description

The code `mul(20).div(100)` can be replaced by `div(5)`

Recommendation

Consider simplifying the code as below:

```
earnGFV = amount.mul(profitTime).div(5).div(duration);
```

Alleviation

The development team resolved this issue in commit c90f7b346e1076e07aab0ea67886402b8b49270a.

LPG-04 | Missing Emit Events

Category	Severity	Location	Status
Optimization	Informational	LpPool.full.sol: 704, 693	Resolved

Description

Several key actions are defined without event declarations.

Recommendation

Consider emitting events for key actions.

Alleviation

The development team heeded our advice and resolved this issue in commit c90f7b346e1076e07aab0ea67886402b8b49270a.

LPG-05 | SafeMath Not Used

Category	Severity	Location	Status
Mathematical Operations	Informational	LpPool.full.sol: 800	Resolved

Description

SafeMath is not used making it possible for overflow/underflow, which will lead to an inaccurate message.

Recommendation

Consider using SafeMath library

```
randomNum.mod(length)
```

Alleviation

The development team heeded our advice and resolved this issue in commit c90f7b346e1076e07aab0ea67886402b8b49270a.

LPG-06 | Potential Calculation Overflow

Category	Severity	Location	Status
Volatile Code	Minor	LpPool.full.sol: 672, 681, 684	Resolved

Description

The calculation of `profitTime` below may cause overflow:

```
672  profitTime = startTime.add(duration).sub(_gfvProfit[account].updateTime);
```

Because the `updateTime` could be assigned to be a value greater than the end time.

```
684  _gfvProfit[account].updateTime = now;
```

Recommendation

Consider ensuring the `updateTime` less than or equals to the activity end time.

Alleviation

The development team heeded our advice and resolved this issue in commit c90f7b346e1076e07aab0ea67886402b8b49270a.

Appendix

Finding Categories

Mathematical Operations

Mathematical Operation findings relate to mishandling of math formulas, such as overflows, incorrect operations etc.

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 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 without CertiK's prior written consent.

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.

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.