byterocket

Smart Contract Audit Report

Float
Fee Mechanism

22nd of August 2022

Contents

Disclaimer

1. Preface

The team of **Float** contracted byterocket to conduct a smart contract audit of an upgrade to their smart contracts related a new fee mechanism that is being introduced. Float is a *"peer-to-peer, yield-enhanced, floating synthetic asset exposure mechanism"*. They describe themselves as "*the easiest and safest way for users to buy synthetic assets. Users do not need to worry about over-collateralization, or suddenly getting liquidated*".

The team of byterocket reviewed and audited the above smart contracts in the course of this audit. We started on the 19th of August and finished on the 22nd of August 2022.

The audit included the following services:

- *Manual Multi-Pass Code Review*
- *Protocol/Logic Analysis*
- *Automated Code Review*
- *Formal Report*

byterocket gained access to the code via a private GitHub repository. We based the audit on the main branch's state on August 22nd, 2022 (*commit hash 923f9b56d0d4812d8a8b2beac83ab420ff1567f2*).

2. Manual Code Review

We conducted a manual multi-pass code review of the smart contracts mentioned in section (1). Three different people went through the smart contract independently and compared their results in multiple concluding discussions.

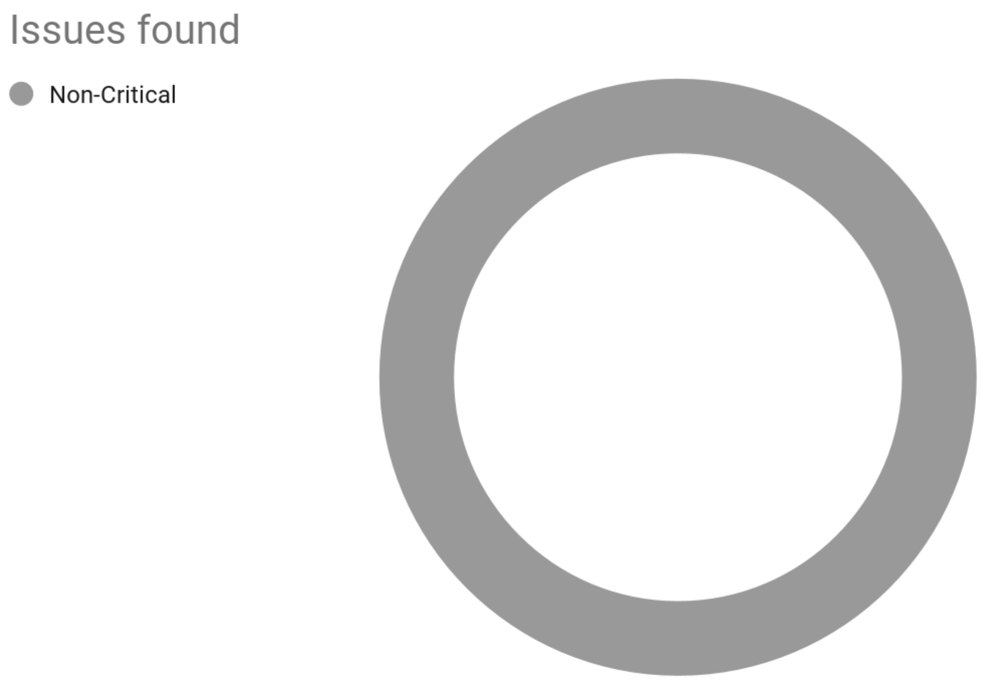
The manual review and analysis were additionally supported by multiple automated reviewing tools, like Slither, GasGauge, Manticore, and different fuzzing tools.

2.1 Severity Categories

We are categorizing our findings into four different levels of severity:

Non-Critical	Does not impose immediate risk but is relevant to security best practices. Includes issues with - Code style and clarity - Versioning - Off-chain monitoring
Low Severity	Imposes relatively small risks or could impose risks in the long-term but without assets being at risk in the current implementation. Includes issues with - State handling - Functions being incorrect as to specification - Faulty documentation or in-code comments
Medium Severity	Imposes risks on the function or availability of the protocol or imposes financial risk by leaking value from the protocol if external requirements are met.
High Severity	Imposes catastrophic risk for users and/or the protocol. Includes issues that could result in - Assets being stolen/lost/compromised - Contracts being rendered useless - Contracts being gained control of

2.2 Summary



On the code level, we **found no bugs or flaws.** Our automated systems and review tools did **not find any additional ones.** Additionally, we found **1 gas improvement**.

The contracts are written according to the latest standard used within the Ethereum community and the Solidity community's best practices. The naming of variables is very logical and understandable, which results in the contract being easy to understand. The code is sufficiently documented.

2.3 Gas Optimizations

[Gas Optimization] GO.1 - Replace division by bitshift

Description:
In order to save some gas, any divisions or multiplications by 2^x can be simplified to x bitshifts to either the left (multiplication) or right (division). As bitshifts are cheaper than divisions or multiplications, this is a good way to save some gas.

The following calculation can be optimized:

```
uint256 mintFeesForEachPool =
feesToDistributeToMarketOnNextUpdate[marketIndex] / 2;
```

To the following:

```
uint256 mintFeesForEachPool =
feesToDistributeToMarketOnNextUpdate[marketIndex] >> 1;
```

3. Protocol/Logic Review

Part of our audits are also analyses of the protocol and its logic. The byterocket team went through the implementation and documentation of the implemented protocol.

The repository itself contained tests and documentation. We found the provided unit tests that are coming with the repository execute without any issues and cover the most important parts of the protocol.

According to our analysis, the protocol and logic are working as intended, given that any findings with a severity level are fixed.

We were **not able to discover any additional problems** in the protocol implemented in the smart contract.

4. Summary

During our code review (*which was done manually and automated*), we **found no bugs or flaws.** Our automated systems and review tools did **not find any additional ones.** Additionally, we found **1 gas improvement**.

The protocol review and analysis did neither uncover any game-theoretical nature problems nor any other functions prone to abuse besides the ones that have been uncovered in our findings.

In general, there are some improvements that can be made, but we are **happy** with the overall quality of the code and its documentation. The developers have been very responsive and were able to answer any questions that we had.