LIDO

# Lido Stonks Security Analysis

by Pessimistic

This report is public

December 7, 2023

## Abstract

In this report, we consider the security of smart contracts of Lido Stonks project. Our task is to find and describe security issues in the smart contracts of the platform.

## Disclaimer

The audit does not give any warranties on the security of the code. A single audit cannot be considered enough. We always recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts. Besides, a security audit is not investment advice.

## Summary

In this report, we considered the security of Lido Stonks smart contracts. We described the audit process in the section below.

The initial audit showed two issues of medium severity: Depeg of stablecoins, Price timestamp. Also, several low-severity issues were found.

After the audit, the developers provided us with a new version of the code, in which they fixed one medium-severity issue, addressed the other one, and resolved all low-severity issues.

The overall code quality is good.

## General recommendations

We do not have any additional recommendations.

# Project overview

## Project description

For the audit, we were provided with Lido Stonks project on a public repository, commit 8dc3711a4a11653b3d4e66b4b4e7dc0747e0c86e.

The scope of the audit included the entire repository.

The documentation for the project consists of a private link to a markdown file.

36 tests out of 37 pass successfully (1 test pending). The code coverage is 83.33%.

The total LOC of audited sources is 476.

## Codebase update

After the initial audit, the developers provided us with a new version of the code, commit ad6a9e83c095f5052e404bc13585ad2c752f242f. This update includes a fix for M02 as well as fixes to all low-severity issues. Additionally, the developers commented on the medium-severity issue that has not been fixed.

77 tests out of 77 pass successfully. The code coverage is 93.2%.

# Audit process

We started the audit on November 28, 2023 and finished on December 4, 2023.

We inspected the materials provided for the audit. Then, we contacted the developers for an introduction to the project. After a discussion, we manually analyzed all the contracts within the scope of the audit and checked their logic. Among other, we verified the following properties of the contracts:

- The correctness of Chainlink integration;
- Correctness of CoW Swap integration and their security considerations;
- Edge cases in price calculation;
- Correctness of type conversions;
- Potential consequences of improper use of privileged roles (see N02 note). We verified that tokens cannot be stuck or sent to an arbitrary address;
- Implementation of struct hashing in **GPv2Order.sol**;
- The possibility of breaking the system by preemptively sending tokens to either **Stonks** or **Order** contract;
- The logic of the price check security: the usage of `marginInBasisPoints` and `priceToleranceInBasisPoints` variables;
- Potential gas optimizations in the contract.

We scanned the project with the following tools:

- Static analyzer Slither;
- Our plugin Slitherin with an extended set of rules;
- Semgrep rules for smart contracts.

We ran tests and calculated the code coverage.

We combined in a private report all the verified issues we found during the manual audit or discovered by automated tools.

After the initial audit, we discussed the results with the developers. On December 6, 2023, the developers provided us with an updated version of the code. In this update, they fixed almost all issues, commented on the M01 issue, and implemented additional tests.

We conducted a review of the updated codebase and scanned the project using the static analyzer Slither, along with our plugin Slitherin with an extended set of rules.

After the review, we updated the report.

# Manual analysis

The contracts were completely manually analyzed, their logic was checked. Besides, the results of the automated analysis were manually verified. All the confirmed issues are described below.

## Critical issues

Critical issues seriously endanger project security. They can lead to loss of funds or other catastrophic consequences. The contracts should not be deployed before these issues are fixed.

**The audit showed no critical issues.**

## Medium severity issues

Medium severity issues can influence project operation in the current implementation. Bugs, loss of potential income, and other non-critical failures fall into this category, as well as potential problems related to incorrect system management. We highly recommend addressing them.

### M01. Depeg of stablecoins (addressed)

Chainlink prices are denoted in the currency (for example, USD). When the order is created, the **Stonks** contract assumes that the prices of stablecoins are always equal to 1 dollar. However, in reality, there are rare cases when the price of the stablecoin deviates significantly. This may result in a loss of funds on bad trades.

> *Comment from the developers: We accept the associated risk related to the potential depegging of any stablecoin. We aim to mitigate this risk operationally, relying on the token management committee.*

### M02. Price timestamp (fixed)

The project heavily relies on prices from Chainlink. However, there are no checks on the timestamp of the received price from the feed. According to Chainlink documentation:

```
Application should track the latestTimestamp variable or use the
updatedAt value from the latestRoundData() function to make sure
that the latest answer is recent enough for your application to use
it.
```

*The issue has been fixed and is not present in the latest version of the code.*

## Low severity issues

Low severity issues do not directly affect project operation. However, they might lead to various problems in future versions of the code. We recommend fixing them or explaining why the team has chosen a particular option.

### L01. Inconsistency between variable name and limitation (fixed)

In **Stonks.sol**, there is a variable named `MIN_POSSIBLE_ORDER_DURATION_IN_SECONDS`. Judging by its name, its value should represent a possible order duration. However, according to line 70, this does not seem to be the case.

*The issue has been fixed and is not present in the latest version of the code.*

### L02. Inconsistent timestamp limitations (fixed)

According to line 129 of **Order.sol**, the order remains valid if `validTo < block.timestamp`. However, tokens can be withdrawn from the contract when `validTo > block.timestamp`. We recommend using complementary equations: either `<=` and `>` or `<` and `>=`.

*The issue has been fixed and is not present in the latest version of the code.*

### L03. Storage packing (fixed)

In **IStonks.sol**, in `OrderParameters` struct, there is an option to pack storage variables. According to line 88 in **Order.sol**, `orderDurationInSeconds` should not exceed `uint32`. Additionally, the values for `marginInBasisPoints` and `priceToleranceInBasisPoints` cannot be greater than `1000`.

*The issue has been fixed and is not present in the latest version of the code.*

## Notes

### N01. Possible gas optimization

The **AmountConverter** contract utilizes the **FeedRegistry** from the Chainlink project, which calls **AggregatorV3Interface**.There is an option to call **AggregatorV3Interface** directly, potentially reducing gas costs. In such a case, project owners should actively monitor changes in the **FeedRegistry**.

### N02. Project roles

The project heavily relies on the `Agent` and `Manager` roles. They have the following powers:

- Placing orders;
- Retrieving ether, ERC20, ERC721, and ERC1155 tokens from the contracts. This implies that privileged roles are able to withdraw tokens intended for trade from the **Stonks** contract.

Note that the mentioned roles are not centralized.

### N03. CoW Swap Liquidity

At the time of writing the report, CoW Swap project does not have a partial fill option. Hence, we recommend tracking the available liquidity and carefully choosing the size of the order.

This analysis was performed by Pessimistic:

Evgeny Marchenko, Senior Security Engineer
Pavel Kondratenkov, Senior Security Engineer
Yhtyyar Sahatov, Security Engineer
Irina Vikhareva, Project Manager
Konstantin Zherebtsov, Business Development Lead
Alexander Seleznev, Founder

December 7, 2023