Illiquidly Labs - P2P-Trading Contract - Audit Report

Prepared for Illiquidly Labs, 11 August 2022

Table of Contents

Introduction

SCV was engaged by Illiquidly Labs to assist in identifying security threats and vulnerabilities that have the potential to affect their security posture. Additionally, SCV will assist the team in understanding the risks and identifying potential mitigations.

Scope

SCV performed the security assessment on the following codebase:

- https://github.com/illiquidly/illiquidlabs-contracts
- Code Freeze: *1f52c02c446c95b1d6cdc70a6762bb8fa30809b2*

Remediations were applied into the same commit hash due a codebase migration to a new location.

Methodologies

SCV performs a combination of automated and manual security testing based on the scope of testing. The testing performed is based on the extensive experience and knowledge of the auditor to provide the greatest coverage and value to Illiquidly Labs. Testing includes, but is not limited to, the following:

- Understanding the application and its code base purpose;
- Deploying SCV in-house tooling to automate dependency analysis and static code review;
- Analyse each line of the code base and inspect application security perimeter;
- Review underlying infrastructure technologies and supply chain security posture;

Code Criteria and Test Coverage

SCV used a scale from **0** to **10** that represents how **SUFFICIENT(6-10)** or **NOT SUFFICIENT(0-5)** each code criteria was during the assessment:

Criteria	Status	Scale Range	Notes
Provided Documentation	**Sufficient**	7-8	N/A
Code Coverage Test	**Sufficient**	7-8	N/A
Code Readability	**Sufficient**	6-8	N/A
Code Complexity	**Sufficient**	6-7	N/A

Vulnerabilities Summary

	Title and Summary	Risk	Status
1	Enforce validation checks on add_asset to prevent misconfigures message from caller.	**Medium**	**Remediated**
2	get_last_trade_id_created cannot guarantee the last trade_id will be returned	**Medium**	**Remediated**
3	The _are_assets_in_trade does not appropriately handle duplicate assets	**Low**	**Remediated**
4	Ensure trade has assets added before publishing	**Informational**	**Acknowledged**
5	Codebase with spelling mistakes	**Informational**	**Remediated**
6	Overflow checks not set for release profile	**Informational**	**Remediated**

Detailed Vulnerabilities

1. Enforce validation checks on add_asset to prevent misconfigures message from caller.

Likelihood	Impact	Risk
Possible	Moderate	**Medium**

Description

The *add_asset* function in *illiquidly-contracts/contracts/p2p-trading/src/contract.rs:290* provides general functionality for adding an asset to a trade or a counter trade. While it is valid and effective functionality, we recommend implementing message validation before adding assets.

The *trade_id*, *counter_id*, *to_last_trade*, and *to_last_counter* are options specified by the caller and are evaluated in sequential order such that a caller cannot reach an unexpected state. A caller may pass an incorrectly configured *AddAsset* message such that the parameters are conflicting with each other and the best outcome is to return an error rather than a potentially unexpected outcome such as adding assets to the wrong trade.

For example, if the caller specifies *to_last_counter* as true and also specifies a *counter_id* the second condition in the if-else chain would be evaluated. There are other parameter combinations that could conflict as well, and the outcome would be determined by the order of the if-else chain.

Recommendations

We recommend explicitly checking that the conditions in each step of the if-else chain are correctly configured before adding assets and returning an error if the AddAsset message is incorrectly configured.

2. get_last_trade_id_created cannot guarantee the last trade_id will be returned

Likelihood	Impact	Risk
Possible	Moderate	**Medium**

Description

The *get_last_trade_id_created* function in *illiquidly-contracts/contracts/p2p-trading/src/trade.rs:28* is specified to only be used in the same transaction as the trade creation. But it is called in *prepare_trade_modification* which is called in both *add_asset_to_trade* and *add_nfts_wanted* which cannot guarantee that the calls are being made in the same transaction as the trade creation. Furthermore, if either the *AddNFTsWanted* or *AddAsset* messages are passed without a *trade_id* after over 100 trades have occurred, the *get_last_trade_id_created* will return an error because *query_all_trades_raw* takes a *BASE_LIMIT* of 100. So if the trader's last trade is not within the last 100 trades it will not be found.

The *GetAllTrades* query calls the *query_all_trades* function so it is important that the limits are in place to prevent unbounded iteration.

Recommendations

We recommend enforcing that *trade_id* is specified in both the *AddAsset* and *AddNFTsWanted* messages or defining separate filter functionality to find the caller's last trade id rather than using the general function that is exposed via the query entry-point.

3. The _are_assets_in_trade does not appropriately handle duplicate assets

Likelihood	Impact	Risk
Possible	Low	**Low**

Description

The *are_assets_in_trade* function in *illiquidly-contracts/contracts/p2p-trading/src/trade.rs:271* does not check to ensure that the assets list provided does not contain duplicate values for a single position. For example, a caller may create a message with multiple entries in the asset list for the same position index. This condition will pass the *are_assets_in_trade* function silently and then throw a panic *'attempt to subtract with overflow'* if the sum of both amounts is greater than the amount of the specified asset in the trade. This condition would successfully cause a subtraction with overflow if the workspace *Cargo.toml* did not contain *overflow-checks* or if that flag were to get removed in the future during refactoring or for any other reason. We classify this as minor because the *overflow-checks* flag will prevent the overflow from occurring.

Recommendations

We recommend either first deduplicating the asset list or to performed checked subtraction in *are_assets_in_trade*.

4. Ensure trade has assets added before publishing

Likelihood	Impact	Risk
Unlikely	Informational	**Informational**

Description

Both *confirm_counter_trade* and *confirm_trade* do not confirm that the trade or counter trade being published have assets associated with them before being published. This may allow for the creation of empty spam trades or counter trades.

Recommendations

We recommend considering implementing a check to confirm that trades or counter trades have associated assets before allowing them to be published.

5. Codebase with spelling mistakes

Likelihood	Impact	Risk
Unlikely	Informational	**Informational**

Description

Throughout the codebase, there are minor spelling errors that should be corrected before going to production. The following are misspellings within the codebase:

- additionnal_info - *illiquidly-contracts/contracts/p2p-trading/src/trade.rs:67*
- mecanism - *illiquidly-contracts/contracts/p2p-trading/src/counter_trade.rs:173*
- ve - *illiquidly-contracts/contracts/p2p-trading/src/trade.rs:691*
- coounter - *illiquidly-contracts/contracts/p2p-trading/src/trade.rs:675*
- withdrawnable - *illiquidly-contracts/contracts/p2p-trading/src/counter_trade.rs:241*
- funtion - *illiquidly-contracts/contracts/p2p-trading/src/trade.rs:766*
- withlisted - *illiquidly-contracts/contracts/p2p-trading/src/trade.rs:75*

Recommendations

We recommend updating the spelling errors mentioned above.

6. Overflow checks not set for release profile

Likelihood	Impact	Risk
Unlikely	Informational	**Informational**

Description

Even though this check is implicitly applied to all packages from the workspace's *Cargo.toml*, we recommend also explicitly enabling overflow checks in every individual package. That helps prevent unintended consequences when the codebase is refactored in the future.

Recommendations

We recommend explicitly enabling overflow checks in *illiquidly-contracts/contracts/p2p-trading/Cargo.toml*.

Document control

Document changes

Version	Date	Name	Changes
0.1	2022-08-08	Vinicius Marino	Initial report
0.2	2022-08-09	Vinicius Marino	Team communication and Pre-Release
1.0	2022-08-11	Vinicius Marino	Document Release

Document contributors

Name	Role	Email address
Vinicius Marino	Security Specialist	vini@scv.services

Appendices

Appendix A: Report Disclaimer

The content of this audit report is provided “As is”, without representations and warranties of any kind.

The author and their employer disclaim any liability for damage arising out of, or in connection with, this audit report.

Appendix B: Risk assessment methodology

A qualitative risk assessment is performed on each vulnerability to determine the impact and likelihood of each.

Risk rate will be calculated on a scale. As per criteria Likelihood vs Impact table below:

Impact \ Likelihood	Rare	Unlikely	Possible	Likely
Critical	Medium	High	Critical	Critical
Severe	Low	Medium	High	High
Moderate	Low	Medium	Medium	High
Low	Low	Low	Low	Medium
Informational	Informational	Informational	Informational	Informational

LIKELIHOOD:

- **Likely**: likely a security incident will occur;
- **Possible**: It is possible a security incident can occur;
- **Unlikely**: Low probability a security incident will occur;
- **Rare**: In rare situations, a security incident can occur;

IMPACT:

- **Critical**: May cause a significant and critical impact;
- **Severe**: May cause a severe impact;
- **Moderate**: May cause a moderated impact;
- **Low**: May cause low or none impact;
- **Informational**: May cause very low impact or none.


SCV