0.7

Loopring V3 Process Quality Review

Score: 86%

Overview

This is a Loopring Process Quality Review completed on 07/10/2021. It was performed using the Process Review process (version 0.7.3) and is documented here. The review was performed by Nick of DeFiSafety. Check out our Telegram.

The final score of the review is **86%**, a **PASS.** The breakdown of the scoring is in Scoring Appendix. For our purposes, a pass is **70%.**

Summary of the Process

Very simply, the review looks for the following declarations from the developer's site. With these declarations, it is reasonable to trust the smart contracts.

- **Here are my smart contracts on the blockchain**
- **Here is the documentation that explains what my smart contracts do**
- **Here are the tests I ran to verify my smart contract**
- **Here are the audit(s) performed on my code by third party experts**
- **Here are the admin controls and strategies**

Disclaimer

This report is for informational purposes only and does not constitute investment advice of any kind, nor does it constitute an offer to provide investment advisory or other services. Nothing in this report shall be considered a solicitation or offer to buy or sell any security, token, future, option or other financial instrument or to offer or provide any investment advice or service to any person in any jurisdiction. Nothing contained in this report constitutes investment advice or offers any opinion with respect to the suitability of any security, and the views expressed in this report should not be taken as advice to buy, sell or hold any security. The information in this report should not be relied upon for the purpose of investing. In preparing the information contained in this report, we have not taken into account the investment needs, objectives and financial circumstances of any particular investor. This information has no regard to the specific investment objectives, financial situation and particular needs of any specific recipient of this information and investments discussed may not be suitable for all investors.

Any views expressed in this report by us were prepared based upon the information available to us at the time such views were written. The views expressed within this report are limited to DeFiSafety and the author and do not reflect those of any additional or third party and are strictly based upon DeFiSafety, its authors, interpretations and evaluation of relevant data. Changed or additional information could cause such views to change. All information is subject to possible correction. Information may quickly become unreliable for various reasons, including changes in market conditions or economic circumstances.

This completed report is copyright (c) DeFiSafety 2021. Permission is given to copy in whole, retaining this copyright label.

Chain

This section indicates the blockchains used by this protocol. This report covers all of the blockchains upon which the protocol is deployed.

✓ **Chain:** Ethereum (+ zkRollup)

Guidance:

Ethereum
Binance Smart Chain
Polygon
Avalanche
Terra
Celo
Arbitrum
Solana

Code and Team

This section looks at the code deployed on the Mainnet that gets reviewed and its corresponding software repository. The document explaining these questions is here. This review will answer the following questions:

1) Are the executing code addresses readily available? (%)
2) Is the code actively being used? (%)
3) Is there a public software repository? (Y/N)
4) Is there a development history visible? (%)
5) Is the team public (not anonymous)? (Y/N)

1) Are the executing code addresses readily available? (%)

✓ **Answer:** 100%

They are available at website https://docs.loopring.io/en/basics/contracts.html , as indicated in the /Appendix.

Guidance:-

-100%	Clearly labelled and on website, docs or repo, quick to find
70%	Clearly labelled and on website, docs or repo but takes a bit of looking
40%	Addresses in mainnet.json, in discord or sub graph, etc
20%	Address found but labeling not clear or easy to find
0%	Executing addresses could not be found

2) Is the code actively being used? (%)

✓ **Answer:** 100%

Activity is 20 transactions a day on contract *ExchangeV3*, as indicated in the Appendix.

Guidance:

100%	More than 10 transactions a day
70%	More than 10 transactions a week
40%	More than 10 transactions a month
10%	Less than 10 transactions a month
0%	No activity

3) Is there a public software repository? (Y/N)

✓ **Answer:** Yes

GitHub: https://github.com/Loopring

Is there a public software repository with the code at a minimum, but also normally test and scripts. Even if the repository was created just to hold the files and has just 1 transaction, it gets a **"Yes"**. For teams with private repositories, this answer is **"No"**.

4) Is there a development history visible? (%)

✓ **Answer:** 100%

At 15 branches and 3,491 commits, it's entirely possible Loopring themselves are looking into a layer 2 solution for navigating their bountiful development history.

This metric checks if the software repository demonstrates a strong steady history. This is normally demonstrated by commits, branches and releases in a software repository. A healthy history demonstrates a history of more than a month (at a minimum).

Guidance:

100%	Any one of 100+ commits, 10+branches
70%	Any one of 70+ commits, 7+branches
50%	Any one of 50+ commits, 5+branches
30%	Any one of 30+ commits, 3+branches
0%	Less than 2 branches or less than 30 commits

5) Is the team public (not anonymous)? (Y/N)

✓ **Answer:** Yes

Location: https://medium.com/loopring-protocol/loopring-is-growing-78daeba7f09a

---

Documentation

This section looks at the software documentation. The document explaining these questions is here.

Required questions are;

6) Is there a whitepaper? (Y/N)
7) Are the basic software functions documented? (Y/N)
8) Does the software function documentation fully (100%) cover the deployed contracts? (%)
9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)
10) Is it possible to trace from software documentation to the implementation in code (%)

6) Is there a whitepaper? (Y/N)

✓ **Answer:** Yes

Location: https://github.com/Loopring/protocols/blob/master/packages/loopring_v3/DESIGN.md

7) Are the basic software functions documented? (Y/N)

✓ **Answer:** Yes

The basic software functions are covered in the documentation. Loopring's living, breathing design document is a robust way to organize such documentation and should be commended for the considerable effort put in.

8) Does the software function documentation fully (100%) cover the deployed contracts? (%)

> ✓ **Answer:** 100%

Contracts as varied as minting NFTs on Loopring's network and on-chain signatures to simple deposits are covered by this document. There is also in-depth software documentation at https://github.com/Loopring/protocols/blob/master/packages/loopring_v3/circuit/statements.md, as well as API documentation at https://docs.loopring.io/en/REST_APIS.html.

Guidance:

100%	All contracts and functions documented
80%	Only the major functions documented
79-1%	Estimate of the level of software documentation
0%	No software documentation

9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)

> ! **Answer:** 25%

Code examples are in the Appendix. As per the SLOC, there is 25% commenting to code (CtC).

The Comments to Code (CtC) ratio is the primary metric for this score.

Guidance:

100%	CtC > 100	Useful comments consistently on all code
90-70%	CtC > 70	Useful comment on most code
60-20%	CtC > 20	Some useful commenting
0%	CtC < 20	No useful commenting

How to improve this score

This score can improve by adding comments to the deployed code such that it comprehensively covers the code. For guidance, refer to the SecurEth Software Requirements.

10) Is it possible to trace from software documentation to the implementation in code (%)

> i **Answer:** 60%

There is clear association between the code and the documentation, though it is lacking explicit traceability.

Guidance:

100%	Clear explicit traceability between code and documentation at a requirement level for all code
60%	Clear association between code and documents via non explicit traceability
40%	Documentation lists all the functions and describes their functions
0%	No connection between documentation and code

How to improve this score:

This score can improve by adding traceability from documentation to code such that it is clear where each outlined function is coded in the source code. For reference, check the SecurEth guidelines on traceability.

Testing

This section looks at the software testing available. It is explained in this document. This section answers the following questions;

11) Full test suite (Covers all the deployed code) (%)
12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)
13) Scripts and instructions to run the tests (Y/N)
14) Report of the results (%)
15) Formal Verification test done (%)
16) Stress Testing environment (%)

11) Is there a Full test suite? (%)

✓ **Answer:** 100%

Code examples are in the Appendix. As per the SLOC, there is 606% testing to code (TtC).

This score is guided by the Test to Code ratio (TtC). Generally a good test to code ratio is over 100%. However the reviewers best judgement is the final deciding factor.

Guidance:

100%	TtC > 120% Both unit and system test visible
80%	TtC > 80% Both unit and system test visible
40%	TtC < 80% Some tests visible
0%	No tests obvious

12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)

ⓘ **Answer:** 50%

The most recent published code coverage test was from 2017, making this not applicable to V3. Nevertheless, at an astonishing 18,000 lines of test code, it is clear that significant testing has been conducted.

Guidance:

100%	Documented full coverage
99-51%	Value of test coverage from documented results
50%	No indication of code coverage but clearly there is a reasonably complete set of tests
30%	Some tests evident but not complete
0%	No test for coverage seen

How to improve this score:

This score can improved by adding tests that achieve full code coverage. A clear report and scripts in the software repository will guarantee a high score.

13) Scripts and instructions to run the tests (Y/N)

✓ **Answer:** Yes

Scripts/Instructions location:
https://github.com/Loopring/protocols/tree/master/packages/loopring_v3#run-unit-tests

14) Report of the results (%)

⚠ **Answer:** 0%

No test dissertation was located.

Guidance:

100%	Detailed test report as described below
70%	GitHub code coverage report visible
0%	No test report evident

How to improve this score

Add a report with the results. The test scripts should generate the report or elements of it.

15) Formal Verification test done (%)

⚠ **Answer:** 0%

Loopring V2 underwent formal verification, but V3 has not yet.

16) Stress Testing environment (%)

> ✓ **Answer:** 100%

Loopring V3 was deployed on Ropsten testnet.

Security

This section looks at the 3rd party software audits done. It is explained in this document. This section answers the following questions;

17) Did 3rd Party audits take place? (%)
18) Is the bounty value acceptably high?

17) Did 3rd Party audits take place? (%)

> ✓ **Answer:** 100%

Two audits are public. The first audit was completed before the code was shipped, the second was completed shortly after, but started beforehand. In these audits all recommendations were acted upon and changed were made.

Guidance:

100%	Multiple Audits performed before deployment and results public and implemented or not required
90%	Single audit performed before deployment and results public and implemented or not required
70%	Audit(s) performed after deployment and no changes required. Audit report is public
50%	Audit(s) performed after deployment and changes needed but not implemented
20%	No audit performed
0%	Audit Performed after deployment, existence is public, report is not public and no improvements deployed OR smart contract address' not found, (where question 1 is 0%)

Deduct 25% if code is in a private repo and no note from auditors that audit is applicable to deployed code

18) Is the bounty value acceptably high (%)

Answer: 60%

Loopring had a 3 month bug bounty period that has now expired, meaning there is no bug bounty currently offered. The value of the aforementioned Bug Bounty was around 300k for the most critical of bu finds.

Guidance:

100%	Bounty is 10% TVL or at least $1M AND active program (see below)
90%	Bounty is 5% TVL or at least 500k AND active program
80%	Bounty is 5% TVL or at least 500k
70%	Bounty is 100k or over AND active program
60%	Bounty is 100k or over
50%	Bounty is 50k or over AND active program
40%	Bounty is 50k or over
20%	Bug bounty program bounty is less than 50k
0%	No bug bounty program offered

An active program means that a third party (such as Immunefi) is actively driving hackers to the site. An inactive program would be static mentions on the docs.

Access Controls

This section covers the documentation of special access controls for a DeFi protocol. The admin access controls are the contracts that allow updating contracts or coefficients in the protocol. Since these contracts can allow the protocol admins to "change the rules", complete disclosure of capabilities is vital for user's transparency. It is explained in this document. The questions this section asks are as follow;

19) Can a user clearly and quickly find the status of the admin controls?
20) Is the information clear and complete?
21) Is the information in non-technical terms that pertain to the investments?
22) Is there Pause Control documentation including records of tests?

19) Can a user clearly and quickly find the status of the access controls (%)

Answer: 70%

Admin control information was found at the bottom of the Loopring website. You can view it at https://loopring.io/#/document/risks_en.md.

Guidance:

100%	Clearly labelled and on website, docs or repo, quick to find
70%	Clearly labelled and on website, docs or repo but takes a bit of looking
40%	Access control docs in multiple places and not well labelled

20%	Access control docs in multiple places and not labelled
0%	Admin Control information could not be found

20) Is the information clear and complete (%)

Answer: 50%

a) All contracts are clearly labelled as upgradeable (or not) -- 10% -- Only the upgradeability of the smart wallet contract is described.

b) The type of ownership is clearly indicated (OnlyOwner / MultiSig / Defined Roles) -- 30% -- Multiple guardian, MultSig, and Controller roles are described.

c) The capabilities for change in the contracts are described -- 10% -- Only the smart wallet contract has its capabilities for change described.

Guidance:

All the contracts are immutable -- 100% OR

a) All contracts are clearly labelled as upgradeable (or not) -- 30% AND
b) The type of ownership is clearly indicated (OnlyOwner / MultiSig / Defined Roles) -- 30% AND
c) The capabilities for change in the contracts are described -- 30%

How to improve this score:

Create a document that covers the items described above. An example is enclosed.

21) Is the information in non-technical terms that pertain to the investments (%)

Answer: 90%

The admin control documentation is written in clear language that relates to the users' investment safety. It can be found at https://loopring.io/#/document/risks_en.md.

Guidance:

100%	All the contracts are immutable
90%	Description relates to investments safety and updates in clear, complete non-software l language
30%	Description all in software specific language
0%	No admin control information could not be found

How to improve this score:

Create a document that covers the items described above in plain language that investors can understand.

An example is enclosed.

22) Is there Pause Control documentation including records of tests (%)

> ✓ **Answer:** 100%

A "withdrawal mode" is detailed, and it operates as a pause control function for the AMM. The last test was performed around 2 months ago at https://github.com/Loopring/protocols/blob/master/packages/loopring_v3/test/testExchangeWithdrawalMode.ts.

Guidance:

100%	All the contracts are immutable or no pause control needed and this is explained OR
100%	Pause control(s) are clearly documented and there is records of at least one test within 3 months
80%	Pause control(s) explained clearly but no evidence of regular tests
40%	Pause controls mentioned with no detail on capability or tests
0%	Pause control not documented or explained

Appendices

Author Details

The author of this review is Rex of DeFi Safety.

Email : rex@defisafety.com Twitter : @defisafety

I started with Ethereum just before the DAO and that was a wonderful education. It showed the importance of code quality. The second Parity hack also showed the importance of good process. Here my aviation background offers some value. Aerospace knows how to make reliable code using quality processes.

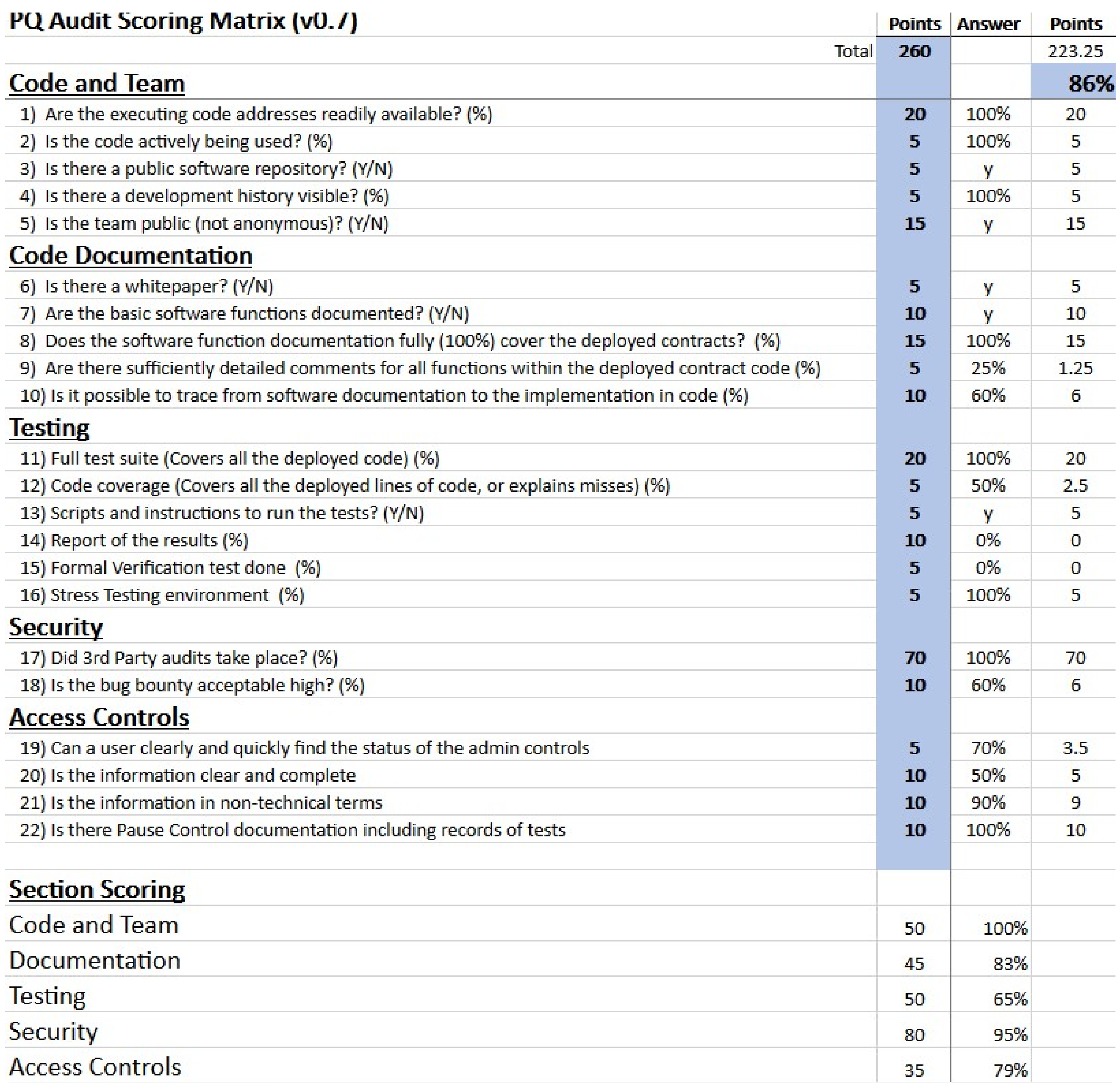
I was coaxed to go to EthDenver 2018 and there I started SecuEth.org with Bryant and Roman. We created guidelines on good processes for blockchain code development. We got EthFoundation funding to assist in their development.

Process Quality Reviews are an extension of the SecurEth guidelines that will further increase the quality processes in Solidity and Vyper development.

DeFiSafety is my full time gig and we are working on funding vehicles for a permanent staff.

Scoring Appendix

	Total	Loopring V3



PQ Audit Scoring Matrix (v0.7)	Points	Answer	Points
Total	**260**		223.25
Code and Team			**86%**
1) Are the executing code addresses readily available? (%)	**20**	100%	20
2) Is the code actively being used? (%)	**5**	100%	5
3) Is there a public software repository? (Y/N)	**5**	y	5
4) Is there a development history visible? (%)	**5**	100%	5
5) Is the team public (not anonymous)? (Y/N)	**15**	y	15
Code Documentation			
6) Is there a whitepaper? (Y/N)	**5**	y	5
7) Are the basic software functions documented? (Y/N)	**10**	y	10
8) Does the software function documentation fully (100%) cover the deployed contracts? (%)	**15**	100%	15
9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)	**5**	25%	1.25
10) Is it possible to trace from software documentation to the implementation in code (%)	**10**	60%	6
Testing			
11) Full test suite (Covers all the deployed code) (%)	**20**	100%	20
12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)	**5**	50%	2.5
13) Scripts and instructions to run the tests? (Y/N)	**5**	y	5
14) Report of the results (%)	**10**	0%	0
15) Formal Verification test done (%)	**5**	0%	0
16) Stress Testing environment (%)	**5**	100%	5
Security			
17) Did 3rd Party audits take place? (%)	**70**	100%	70
18) Is the bug bounty acceptable high? (%)	**10**	60%	6
Access Controls			
19) Can a user clearly and quickly find the status of the admin controls	**5**	70%	3.5
20) Is the information clear and complete	**10**	50%	5
21) Is the information in non-technical terms	**10**	90%	9
22) Is there Pause Control documentation including records of tests	**10**	100%	10
Section Scoring			
Code and Team	50	100%	
Documentation	45	83%	
Testing	50	65%	
Security	80	95%	
Access Controls	35	79%	

Executing Code Appendix

LRC

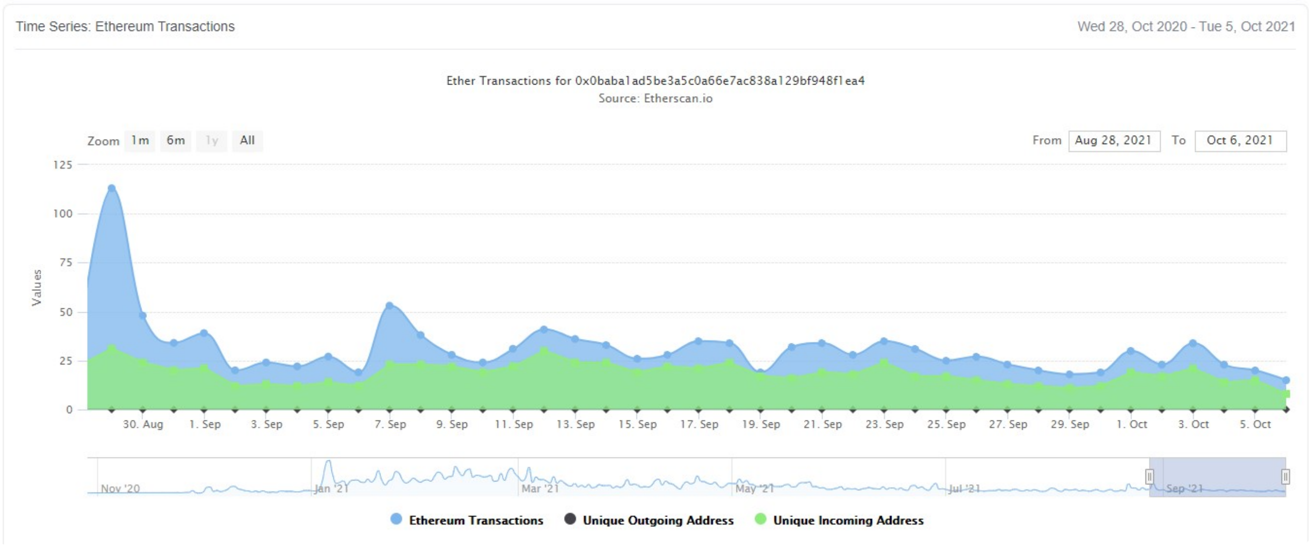
- LRC Address: 0xBBbbCA6A901c926F240b89EacB641d8Aec7AEafD (lrctoken.eth)

Loopring Exchange V2 (Added at 2020-12-21)

- ExchangeV3: 0x0BABA1Ad5bE3a5C0a66E7ac838a129Bf948f1eA4 (exchange2.loopring.eth)
- DefaultDepositContract: 0x674bdf20A0F284D710BC40872100128e2d66Bd3f (deposit2.loopring.eth)
- LoopringIOExchangeOwner: 0x5c367c1b2603ed166C62cEc0e4d47e9D5DC1c073
- ExchangeV3 Implementation: 0x2fefbeF4d1445F523941c56349C2414cd5e9675d
- LoopringV3: 0xe56D6ccab6551932C0356E4e8d5dAF0630920C71
- BlockVerifier: 0x6150343E0F43A17519c0327c41eDd9eBE88D01ef (verifier2.loopring.eth)

- AgentRegistry: 0x39B9bf169a7e225ba037C443A40460c77438ea14 (agents2.loopring.eth)
- FastWithdrawalAgent: 0xec3Cc6Cf0252565b56FC7AC396017Df5b9B78a31 (fastwithdraw2.loopring.eth)

Code Used Appendix



Example Code Appendix

```
// SPDX-License-Identifier: Apache-2.0
// Copyright 2017 Loopring Technology Limited.
pragma solidity ^0.7.0;
pragma experimental ABIEncoderV2;

import "../aux/access/ITransactionReceiver.sol";
import "../core/iface/IAgentRegistry.sol";
import "../lib/ReentrancyGuard.sol";
import "../lib/TransferUtil.sol";
import "./libamm/AmmAssetManagement.sol";
import "./libamm/AmmData.sol";
import "./libamm/AmmExitRequest.sol";
import "./libamm/AmmJoinRequest.sol";
import "./libamm/AmmPoolToken.sol";
import "./libamm/AmmStatus.sol";
import "./libamm/AmmTransactionReceiver.sol";
import "./libamm/AmmWithdrawal.sol";
import "./PoolToken.sol";


/// @title LoopringAmmPool
contract LoopringAmmPool is
    PoolToken,
    IAgent,
    ITransactionReceiver,
    ReentrancyGuard
{
    using AmmAssetManagement     for AmmData.State;
    using AmmJoinRequest         for AmmData.State;
    using AmmExitRequest         for AmmData.State;
    using AmmPoolToken           for AmmData.State;
    using AmmStatus              for AmmData.State;
    using AmmTransactionReceiver for AmmData.State;
    using AmmWithdrawal          for AmmData.State;
    using TransferUtil           for address;

    event PoolJoinRequested(AmmData.PoolJoin join);
    event PoolExitRequested(AmmData.PoolExit exit, bool force);
    event ForcedExitProcessed(address owner, uint96 burnAmount, uint96[] amounts);
    event Shutdown(uint timestamp);

    IAmmController public immutable controller;
    IAssetManager  public immutable assetManager;
    bool           public immutable joinsDisabled;

    modifier onlyFromExchangeOwner()
    {
        require(msg.sender == state.exchangeOwner, "UNAUTHORIZED");
        _;
    }

    modifier onlyFromAssetManager()
    {
        require(msg.sender == address(assetManager), "UNAUTHORIZED");
        _;
    }

    modifier onlyFromController()
    {
        require(msg.sender == address(controller), "UNAUTHORIZED");
        _;
    }

    modifier onlyWhenOnline()
    {
        require(state.isOnline(), "NOT_ONLINE");
        _;
    }

    modifier onlyWhenOffline()
    {
        require(!state.isOnline(), "NOT_OFFLINE");
        _;
    }

    constructor(
        IAmmController _controller,
        IAssetManager  _assetManager,
        bool           _joinsDisabled
    )
    {
        require(_controller != IAmmController(0), "ZERO_ADDRESS");
        controller = _controller;
        assetManager = _assetManager;
        joinsDisabled = _joinsDisabled;
    }

    function isOnline()
        public
        view
        returns (bool)
    {
        return state.isOnline();
    }

    receive() payable external {}

    function setupPool(AmmData.PoolConfig calldata config)
        external
        nonReentrant
    {
        require(state.accountID == 0 || msg.sender == address(controller), "UNAUTHORIZED")
        state.setupPool(config);
    }

    function enterExitMode(bool enabled)
        external
        onlyFromController
    {
        require(state.exitMode != enabled, "INVALID_STATE");
        state.exitMode = enabled;
    }

    // Anyone is able to shut down the pool when requests aren't being processed any more.
    function shutdown(address exitOwner)
        external
        payable
        onlyWhenOnline
        nonReentrant
    {
        state.shutdownByLP(exitOwner);
    }

    function shutdownByController()
        external
        onlyWhenOnline
        nonReentrant
        onlyFromController
    {
        state.shutdownByController();
    }

    function joinPool(
        uint96[]     calldata joinAmounts,
        uint96                mintMinAmount,
        uint96                fee
        )
        external
        payable
        onlyWhenOnline
        nonReentrant
    {
        state.joinPool(joinAmounts, mintMinAmount, fee);
    }

    function exitPool(
        uint96            burnAmount,
        uint96[] calldata exitMinAmounts
        )
        external
        payable
        onlyWhenOnline
        nonReentrant
    {
        state.exitPool(burnAmount, exitMinAmounts, false);
    }

    function forceExitPool(
        uint96            burnAmount,
        uint96[] calldata exitMinAmounts
        )
        external
        payable
        onlyWhenOnline
        nonReentrant
    {
        state.exitPool(burnAmount, exitMinAmounts, true);
    }

    function onReceiveTransactions(
        bytes             calldata txsData,
        bytes             calldata callbackData
        )
        external
        override
        onlyWhenOnline
        onlyFromExchangeOwner
        // nonReentrant     // Not needed, does not do any external calls
                            // and can only be called by the exchange owner.
    {
        AmmData.Settings memory settings = AmmData.Settings({
            controller: controller,
            assetManager: assetManager,
            joinsDisabled: joinsDisabled
        });
        state.onReceiveTransactions(txsData, callbackData, settings);
    }

    function withdrawWhenOffline()
        external
        onlyWhenOffline
        nonReentrant
    {
        state.withdrawWhenOffline();
    }

    function transferOut(
        address to,
        address token,
        uint    amount
        )
        external
        nonReentrant
        onlyFromAssetManager
    {
        state.transferOut(to, token, amount);
    }

    function setBalanceL1(
        address token,
        uint96  balance
        )
        external
        nonReentrant
        onlyFromAssetManager
    {
        state.balancesL1[token] = balance;
    }

    function getBalanceL1(
        address token
        )
        public
        view
        returns (uint96)
    {
        return state.balancesL1[token];
    }
}
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex

Solidity	24	4930	487	783	3120	187

Comments to Code 783/3120 = 25%

Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
TypeScript	40	22415	2450	1107	18858	1536
JSON	1	46	0	0	46	0
Python	1	41	4	29	8	3
Total	42	22502	2454	1136	18912	1539

Tests to Code 18912/3120 = 606%