0.7

Balancer Finance V2 Process Quality Review

Score: 100%

Overview

This is a Balancer V2 Process Quality Review completed on June 14th 2021. It was performed using the Process Review process (version 0.7.2) and is documented here. The review was performed by Nic of DeFiSafety. Check out our Telegram.

The final score of the review is 100%, a perfect score. 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

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.

Chain

This section indicates the blockchain used by this protocol.

> ✓ **Chain: Ethereum**

Guidance:
Ethereum
Binance

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 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.balancer.fi/developers/smart-contracts/deployment-addresses 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 labelling not clear or easy to find
0% Executing addresses could not be found

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

> Answer: 100%

Activity is 200 transactions a day on contract *Vault.sol*, as indicated in the Appendix.

Percentage Score 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/balancer-labs/.

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

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

> Answer: 100%

With 606 commits and 18 branches, this is a healthy software repository.

This 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 10 commits

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

> Answer: Yes

CEO is public: Fernando Martinelli.

https://messari.io/asset/balancer/profile

For a yes in this question the real names of some team members must be public on the website or other documentation. If the team is anonymous and then this question is a No.

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://medium.com/balancer-protocol/balancer-v2-generalizing-amms-16343c4563ff.

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

> ✓ Answer: Yes

At https://docs.balancer.fi/developers/smart-contracts/apis.

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

> ✓ Answer: 100%

At https://docs.balancer.fi/developers/smart-contracts/apis.

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: 85%

Code examples are in the Appendix. As per the SLOC, there is 75% commenting to code (CtC). An extra 10% was added to this score as their commenting on their core contracts is exeptionally well done.

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 (%)

> Answer: 100%

Robust software documentation to code traceability at https://docs.balancer.fi/developers/smart-contracts/apis.

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

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 170% 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: 96%

The Coveralls report indicates a 96% code coverage. https://coveralls.io/github/balancer-labs/balancer-core

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 improve by adding tests achieving 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

Instructions to run the tests can be found at https://github.com/balancer-labs/balancer-core

How to improve this score

Add the scripts to the repository and ensure they work. Ask an outsider to create the environment and run the tests. Improve the scripts and docs based on their feedback.

14) Report of the results (%)

> Answer: 100%

Their test report is at :https://github.com/balancer-labs/balancer-v2-monorepo/blob/master/audits/test-report.md

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: 100%

https://firebasestorage.googleapis.com/v0/b/gitbook-28427.appspot.com/o/assets%2F-MWZrc_wdLRZXvxl5Xwv%2F-MZux6kNvNHD45asnimV%2F-M_82zbLfdvhQN1imoPD%2F2021-04-19.pdf?alt=media&token=4c4fb27d-8b66-4a36-b6ab-4ef45b1464af

16) Stress Testing environment (%)

> Answer: 100%

Kovan test-net smart contract addresses can be found at https://docs.balancer.fi/developers/smart-contracts/deployment-addresses.

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%

Certora did a Balancer V2 audit on April 19th 2021.

OpenZeppelin did a Balancer V2 audit on March 15th 2021.

Trail of Bits did a Balancer V2 audit on April 5th 2021.

Balancer V2 was launched on April 20th 2021.

Most issues found were fixed and solutions implemented.

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, question

18) Is the bounty value acceptably high (%)

> Answer: 100%

Bug Bounty rewards is as high as 2M USD and is active.

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

Active program means a third party actively driving hackers to the site. Inactive program would be static mention 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?
2`) 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: 100%

https://docs.balancer.fi/core-concepts/governance

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: 100%

Everything is detailed perfectly at https://docs.balancer.fi/core-concepts/governance/multisig#context. The majority of contracts are immutable as indicated here: https://docs.balancer.fi/core-concepts/security/emergency-pause with some contracts being upgradable(which is also described in depth)

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: 100%

Contracts are indicated as mostly immutable as stated in https://docs.balancer.fi/core-concepts/security/emergency-pause

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%

Pause control explained at https://docs.balancer.fi/core-concepts/security/emergency-pause.

Pause control tests documented here https://github.com/balancer-labs/balancer-v2-monorepo/tree/6c9e24e22d0c46cca6dd15861d3d33da61a60b98/pkg/solidity-utils/contracts/helpers.

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

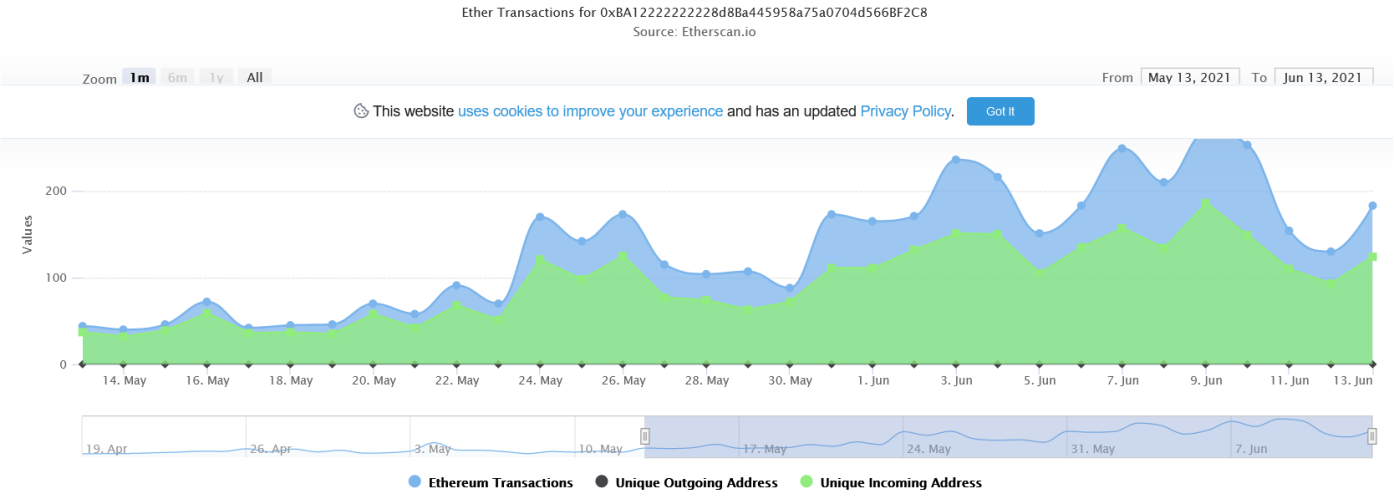
	Total	Balancer V2	
PQ Audit Scoring Matrix (v0.7)	**Points**	**Answer**	**Points**
Total	260		259.05
Code and Team			**100%**
1) Are the executing code addresses readily available? (%)	20	100%	20
2) Is the code actively being used? (%)	5	100%	5
3) Is there a public software repository? (Y/N)	5	Y	5
4) 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	85%	4.25
10) Is it possible to trace from software documentation to the implementation in code (%)	10	100%	10
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	96%	4.8
13) Scripts and instructions to run the tests? (Y/N)	5	Y	5
14) Report of the results (%)	10	100%	10
15) Formal Verification test done (%)	5	100%	5
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	100%	10
Access Controls			
19) Can a user clearly and quickly find the status of the admin controls	5	100%	5
20) Is the information clear and complete	10	100%	10
21) Is the information in non-technical terms	10	100%	10
22) Is there Pause Control documentation including records of tests	10	100%	10

Section Scoring			
Code and Team	50	100%	
Documentation	45	98%	
Testing	50	100%	
Security	80	100%	
Access Controls	35	100%	

Executing Code Appendix

Contract	Address
Vault	0xBA12222222228d8Ba445958a75a0704d566BF2C8
WeightedPoolFactory	0x8E9aa87E45e92bad84D5F8DD1bff34Fb92637dE9
WeightedPool2TokensFactory	0xA5bf2ddF098bb0Ef6d120C98217dD6B141c74EE0
Authorizer	0xA331D84eC860Bf466b4CdCcFb4aC09a1B43F3aE6 (temporary)

Code Used Appendix



Example Code Appendix

```
1 // SPDX-License-Identifier: GPL-3.0-or-later
2 // This program is free software: you can redistribute it and/or modify
3 // it under the terms of the GNU General Public License as published by
4 // the Free Software Foundation, either version 3 of the License, or
5 // (at your option) any later version.
6
7 // This program is distributed in the hope that it will be useful,
8 // but WITHOUT ANY WARRANTY; without even the implied warranty of
9 // MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
10
```

```
// GNU General Public License for more details.

// You should have received a copy of the GNU General Public License
// along with this program.  If not, see <http://www.gnu.org/licenses/>.

pragma solidity ^0.7.0;
pragma experimental ABIEncoderV2;

import "@balancer-labs/v2-solidity-utils/contracts/misc/IWETH.sol";

import "./interfaces/IAuthorizer.sol";

import "./VaultAuthorization.sol";
import "./FlashLoans.sol";
import "./Swaps.sol";

/**
 * @dev The `Vault` is Balancer V2's core contract. A single instance of it exists for the
 * entity used to interact with Pools by Liquidity Providers who join and exit them, Trade
 * Managers who withdraw and deposit tokens.
 *
 * The `Vault`'s source code is split among a number of sub-contracts, with the goal of imp
 * understanding the system easier. Most sub-contracts have been marked as `abstract` to ex
 * the full `Vault` is meant to be deployed.
 *
 * Roughly speaking, these are the contents of each sub-contract:
 *
 *  - `AssetManagers`: Pool token Asset Manager registry, and Asset Manager interactions.
 *  - `Fees`: set and compute protocol fees.
 *  - `FlashLoans`: flash loan transfers and fees.
 *  - `PoolBalances`: Pool joins and exits.
 *  - `PoolRegistry`: Pool registration, ID management, and basic queries.
 *  - `PoolTokens`: Pool token registration and registration, and balance queries.
 *  - `Swaps`: Pool swaps.
 *  - `UserBalance`: manage user balances (Internal Balance operations and external balance
 *  - `VaultAuthorization`: access control, relayers and signature validation.
 *
 * Additionally, the different Pool specializations are handled by the `GeneralPoolsBalance
 * `MinimalSwapInfoPoolsBalance` and `TwoTokenPoolsBalance` sub-contracts, which in turn ma
 * `BalanceAllocation` library.
 *
 * The most important goal of the `Vault` is to make token swaps use as little gas as poss
 * multitude of design decisions, from minor things like the format used to store Pool IDs
 * the different Pool specialization settings.
 *
 * Finally, the large number of tasks carried out by the Vault means its bytecode is very l
 * the contract size limit imposed by EIP 170 (https://eips.ethereum.org/EIPS/eip-170). Mar
 * was required to improve code generation and bring the bytecode size below this limit. Th
 * utilization of `internal` functions (particularly inside modifiers), usage of named retu
 * storage access methods, dynamic revert reason generation, and usage of inline assembly,
 */
contract Vault is VaultAuthorization, FlashLoans, Swaps {
    constructor(
```

```
63
          IAuthorizer authorizer,
64        IWETH weth,
65        uint256 pauseWindowDuration,
66        uint256 bufferPeriodDuration
67    ) VaultAuthorization(authorizer) AssetHelpers(weth) TemporarilyPausable(pauseWindowDura
68        // solhint-disable-previous-line no-empty-blocks
69    }
70
71    function setPaused(bool paused) external override nonReentrant authenticate {
72        _setPaused(paused);
73    }
74
75    // solhint-disable-next-line func-name-mixedcase
76    function WETH() external view override returns (IWETH) {
77        return _WETH();
78    }
79 }
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Solidity	100	16041	2290	5898	7853	744

Comments to Code 5898/7853 = 75%

Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
JavaScript	77	17403	3396	641	13366	609

Tests to Code 13366/7853 = 170%