0.7

Biswap Process Quality Review

Score: 11%

Overview

This is a Biswap Process Quality Review completed on June 30th 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 11%, a terrible 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: Binance Smart Chain**

Guidance:
Ethereum
Binance Smart Chain
Polygon

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: 0%

No publicly available executing smart contract addresses were found on their website/in any of their documentation.

Note: Token contracts are not executing contract addresses. A good example of a executing address is the MasterChef contract.

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

How to improve this score

Make the Ethereum addresses of the smart contract utilized by your application available on either your website or your GitHub (in the README for instance). Ensure the addresses is up to date. This is a very important question wrt to the final score.

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

⚠ Answer: 0%

No executing code addresses are publicly available in Biswap's documentation, and therefore we cannot evaluate the code's usage.

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/biswap-org

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: 0%

With 13 commits and 1 branch, this is an unhealthy 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

How to improve this score

Continue to test and perform other verification activities after deployment, including routine maintenance updating to new releases of testing and deployment tools. A public development history indicates clearly to the public the level of continued investment and activity by the developers on the application. This gives a level of security and faith in the application.

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

⚠ Answer: No

No public team info was found in their documentation or through web searches.

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://docs.biswap.org/

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

⚠ Answer: No

There is no evident software documentation available.

How to improve this score

Write the document based on the deployed code. For guidance, refer to the SecurEth System Description Document.

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

⚠ Answer: 0%

There is no evident software documentation available.

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

How to improve this score

This score can improve by adding content to the requirements document such that it comprehensively covers the requirements. For guidance, refer to the SecurEth System Description Document . Using tools that aid traceability detection will help.

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

ⓘ Answer: 70%

Code examples are in the Appendix. As per the SLOC, there is 70% 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 (%)

> ⚠ Answer: 0%

There is no evident software documentation available.

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 requirements to code such that it is clear where each requirement is coded. 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: 40%

Code examples are in the Appendix. As per the SLOC, there is 7% 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

How to improve this score

This score can improve by adding tests to fully cover the code. Document what is covered by traceability or test results in the software repository.

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

> ⓘ Answer: 30%

No evidence of code coverage in their GitHub repository or in their Certik audit.

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: No

No scripts or instructions to run tests were found in their GitHub repository.

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: 0%

No test report or coverage report in their GitHub repository.

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%

No evidence of a Biswap Formal Verification test was found in their documentation/GitHub.

16) Stress Testing environment (%)

⚠ Answer: 0%

No evidence of test-net smart contract address usage was found in any Biswap documentation.

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: 0%

Certik published a Biswap security assessment on June 9th 2021. Since the contract addresses were not found, the score is 0% because the executing code could be different than the audited code and we can't verify.

Note: Biswap was released at the end of April 2021.

Note 2: All fix recommendations were acknowledged but not 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: 70%

Bug Bounty program found at https://docs.biswap.org/bug-bounty.

Reward as high as 100k worth of their native token.

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: 0%

No admin access control information was found in any of their documentation.

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: 0%

No admin access control information was found in any of their documentation, this includes a lack of ownership type, clear labelling, and description of capabilities for change in contracts.

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: 0%

No admin control information was found.

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: 0%

No evidence of Pause Control was found in their documentation or in their GitHub repository.

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

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.

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

PQ Audit Scoring Matrix (v0.7)	Total Points	BiSwap Answer	BiSwap Points
Total	260		29.5
Code and Team			**11%**
1) Are the executing code addresses readily available? (%)	20	0%	0
2) Is the code actively being used? (%)	5	0%	0
3) Is there a public software repository? (Y/N)	5	Y	5
4) Is there a development history visible? (%)	5	0%	0
5) Is the team public (not anonymous)? (Y/N)	15	N	0
Code Documentation			
6) Is there a whitepaper? (Y/N)	5	Y	5
7) Are the basic software functions documented? (Y/N)	10	N	0
8) Does the software function documentation fully (100%) cover the deployed contracts? (%)	15	0%	0
9) Are there sufficiently detailed comments for all functions within the deployed contract code (%)	5	0%	0
10) Is it possible to trace from software documentation to the implementation in code (%)	10	40%	4
Testing			
11) Full test suite (Covers all the deployed code) (%)	20	40%	8
12) Code coverage (Covers all the deployed lines of code, or explains misses) (%)	5	30%	1.5
13) Scripts and instructions to run the tests? (Y/N)	5	N	0
14) Report of the results (%)	10	0%	0
15) Formal Verification test done (%)	5	0%	0
16) Stress Testing environment (%)	5	0%	0
Security			
17) Did 3rd Party audits take place? (%)	70	0%	0
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	0%	0
20) Is the information clear and complete	10	0%	0
21) Is the information in non-technical terms	10	0%	0
22) Is there Pause Control documentation including records of tests	10	0%	0
Section Scoring			
Code and Team	50	10%	
Documentation	45	20%	
Testing	50	19%	
Security	80	8%	
Access Controls	35	0%	

Executing Code Appendix

null

Code Used Appendix

null

Example Code Appendix

```
pragma solidity 0.6.12;
library SafeBEP20 {
    using SafeMath for uint256;
    using Address for address;

    function safeTransfer(
        IBEP20 token,
        address to,
        uint256 value
    ) internal {
        _callOptionalReturn(token, abi.encodeWithSelector(token.transfer.selector, to, valu
    }

    function safeTransferFrom(
        IBEP20 token,
        address from,
        address to,
        uint256 value
    ) internal {
        _callOptionalReturn(token, abi.encodeWithSelector(token.transferFrom.selector, fro
    }

    /**
     * @dev Deprecated. This function has issues similar to the ones found in
     * {IBEP20-approve}, and its usage is discouraged.
     *
     * Whenever possible, use {safeIncreaseAllowance} and
     * {safeDecreaseAllowance} instead.
     */
    function safeApprove(
        IBEP20 token,
        address spender,
        uint256 value
    ) internal {
        // safeApprove should only be called when setting an initial allowance,
        // or when resetting it to zero. To increase and decrease it, use
        // 'safeIncreaseAllowance' and 'safeDecreaseAllowance'
        // solhint-disable-next-line max-line-length
        require(
            (value == 0) || (token.allowance(address(this), spender) == 0),
            'SafeBEP20: approve from non-zero to non-zero allowance'
        );
        _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender,
    }

    function safeIncreaseAllowance(
        IBEP20 token,
        address spender,
        uint256 value
    ) internal {
```

```
51          uint256 newAllowance = token.allowance(address(this), spender).add(value);
52          _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender,
53      }
54
55      function safeDecreaseAllowance(
56          IBEP20 token,
57          address spender,
58          uint256 value
59      ) internal {
60          uint256 newAllowance = token.allowance(address(this), spender).sub(
61              value,
62              'SafeBEP20: decreased allowance below zero'
63          );
64          _callOptionalReturn(token, abi.encodeWithSelector(token.approve.selector, spender,
65      }
66
67      /**
68       * @dev Imitates a Solidity high-level call (i.e. a regular function call to a contrac
69       * on the return value: the return value is optional (but if data is returned, it must
70       * @param token The token targeted by the call.
71       * @param data The call data (encoded using abi.encode or one of its variants).
72       */
73      function _callOptionalReturn(IBEP20 token, bytes memory data) private {
74          // We need to perform a low level call here, to bypass Solidity's return data size
75          // we're implementing it ourselves. We use {Address.functionCall} to perform this c
76          // the target address contains contract code and also asserts for success in the lo
77
78          bytes memory returndata = address(token).functionCall(data, 'SafeBEP20: low-level c
79          if (returndata.length > 0) {
80              // Return data is optional
81              // solhint-disable-next-line max-line-length
82              require(abi.decode(returndata, (bool)), 'SafeBEP20: BEP20 operation did not suc
83          }
84      }
85  }
86  import "./BSWToken.sol";
87
88  interface IMigratorChef {
89      function migrate(IBEP20 token) external returns (IBEP20);
90  }
91
92  // MasterChef is the master of BSW. He can make BSW and he is a fair guy.
93  //
94  // Note that it's ownable and the owner wields tremendous power. The ownership
95  // will be transferred to a governance smart contract once BSW is sufficiently
96  // distributed and the community can show to govern itself.
97  //
98  // Have fun reading it. Hopefully it's bug-free. God bless.
99  contract MasterChef is Ownable {
100     using SafeMath for uint256;
101     using SafeBEP20 for IBEP20;
102     // Info of each user.
103     struct UserInfo {
```

```
104
        uint256 amount; // How many LP tokens the user has provided.
105         uint256 rewardDebt; // Reward debt. See explanation below.
106         //
107         // We do some fancy math here. Basically, any point in time, the amount of BSWs
108         // entitled to a user but is pending to be distributed is:
109         //
110         //   pending reward = (user.amount * pool.accBSWPerShare) - user.rewardDebt
111         //
112         // Whenever a user deposits or withdraws LP tokens to a pool. Here's what happens:
113         //   1. The pool's `accBSWPerShare` (and `lastRewardBlock`) gets updated.
114         //   2. User receives the pending reward sent to his/her address.
115         //   3. User's `amount` gets updated.
116         //   4. User's `rewardDebt` gets updated.
117     }
118     // Info of each pool.
119     struct PoolInfo {
120         IBEP20 lpToken; // Address of LP token contract.
121         uint256 allocPoint; // How many allocation points assigned to this pool. BSWs to d
122         uint256 lastRewardBlock; // Last block number that BSWs distribution occurs.
123         uint256 accBSWPerShare; // Accumulated BSWs per share, times 1e12. See below.
124     }
125     // The BSW TOKEN!
126     BSWToken public BSW;
127     //Pools, Farms, Dev, Refs percent decimals
128     uint256 public percentDec = 1000000;
129     //Pools and Farms percent from token per block
130     uint256 public stakingPercent;
131     //Developers percent from token per block
132     uint256 public devPercent;
133     //Referrals percent from token per block
134     uint256 public refPercent;
135     //Safu fund percent from token per block
136     uint256 public safuPercent;
137     // Dev address.
138     address public devaddr;
139     // Safu fund.
140     address public safuaddr;
141     // Refferals commision address.
142     address public refAddr;
143     // Last block then develeper withdraw dev and ref fee
144     uint256 public lastBlockDevWithdraw;
145     // BSW tokens created per block.
146     uint256 public BSWPerBlock;
147     // Bonus muliplier for early BSW makers.
148     uint256 public BONUS_MULTIPLIER = 1;
149     // The migrator contract. It has a lot of power. Can only be set through governance (ow
150     IMigratorChef public migrator;
151     // Info of each pool.
152     PoolInfo[] public poolInfo;
153     // Info of each user that stakes LP tokens.
154     mapping(uint256 => mapping(address => UserInfo)) public userInfo;
155     // Total allocation poitns. Must be the sum of all allocation points in all pools.
156     uint256 public totalAllocPoint = 0;
```

```
157
        // The block number when BSW mining starts.
158     uint256 public startBlock;
159     // Deposited amount BSW in MasterChef
160     uint256 public depositedBsw;
161
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Solidity	9	2865	330	1042	1493	164

Comments to Code 1042/1493 = 70%

Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
JavaScript	2	303	44	160	99	8

Tests to Code 99/1493 = 7%