

Bancor Process Quality Review

Score 96%

This is a Process Quality Review of [Bancor](#) completed on 3 November 2020. It was performed using the Process Review process (version 0.6) and is documented [here](#). The review was performed by ShinkaRex of [Caliburn Consulting](#). Check out our [Telegram](#).

The final score of the review is 96%, an excellent score. The breakdown of the scoring is in [Scoring Appendix](#).

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**

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, 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. 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.

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? (Y/N)
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)

Are the executing code addresses readily available? (Y/N)

 Answer: Yes

They are available at website <https://docs.bancor.network/ethereum-contracts/addresses> as indicated in the [Appendix](#). This review only covers the contract ContractRegistry.sol.

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.

Is the code actively being used? (%)

 Answer: 100%

Activity is well in excess of 10 transactions a day, as indicated in the [Appendix](#). This uses 0x2F9EC37d6CcFFf1caB21733BdaDEdE11c823cCB0 (Bancor Network) token.

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

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

 Answer: Yes

Location: <https://github.com/bancorprotocol/>

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.

Is there a development history visible? (%)

 Answer: 100%

With 5 branches and 3640 commits, this is a healthy 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

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

Answer: Yes

The names of the team members can be found in their [Whitepaper](#).

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 seems a No.

Documentation

This section looks at the software documentation. The document explaining these questions is [here](#).

Required questions are;

1. Is there a whitepaper? (Y/N)
2. Are the basic software functions documented? (Y/N)
3. Does the software function documentation fully (100%) cover the deployed contracts? (%)
4. Are there sufficiently detailed comments for all functions within the deployed contract code (%)
5. Is it possible to trace from software documentation to the implementation in codee (%)

Is there a whitepaper? (Y/N)



Answer: Yes

Location: Their whitepaper is both their [Documentation](#), as well as the [Whitepaper](#).

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



Answer: Yes

Location: <https://docs.bancor.network/ethereum-contracts/ethereum-api-reference>

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



Answer: 100%

The requirements are extremely well explained in the Bancor API. All of the specific contracts have detailed explanations with reference to the code within them. This is extremely well organized, and well written documentation.

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

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



Answer: 80%

There are detailed comments documenting the functions on all level of code.

Code examples are in the [Appendix](#). As per the [SLOC](#), there is 73% 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](#).

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



Answer: 100%

The API specifically defines all parts of the code. there is clear traceability throughout all of the documentation.

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;

1. Full test suite (Covers all the deployed code) (%)
2. Code coverage (Covers all the deployed lines of code, or explains misses) (%)
3. Scripts and instructions to run the tests (Y/N)
4. Packaged with the deployed code (Y/N)
5. Report of the results (%)
6. Formal Verification test done (%)
7. Stress Testing environment (%)

Is there a Full test suite? (%)



Answer: 100%

As per the [Appendix](#), The test to code ratio is 319%.

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.

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



Answer: 50%

As there is no indication of code coverage, a 50% score is given.

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.

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



Answer: Yes

Instructions on how to run the testing is found in Bancor's [documentation](#).

Packaged with the deployed code (Y/N)



Answer: Yes

The testing is packaged with the deployed code.

Report of the results (%)



Answer: 90%

A test report is found at : <https://github.com/bancorprotocol/solidity-test-reports>. The report lists all test complete on a standard run but does not indicate coverage (this ran previously but did not run when generated for this report. It may be added in the future.

How to improve this score

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

Formal Verification test done (%)



Answer: 0%

There is no evidence of formal verification testing having been done in their external documentation.

Stress Testing environment (%)



Answer: 100%

There is evidence of stress testing done on the Ropsten network.

Audits



Answer: 100%

There are 5 different audits published. Links can be found in the [Security Page](#).

Guidance:

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

Appendices

Author Details

The author of this review is Rex of [Caliburn Consulting](#).

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.

Career wise I am a business development manager for an avionics supplier.

Scoring Appendix

PQ Audit Scoring Matrix (v0.6)	Total	Bancor	
	Points	Answer	Points
Total	240		229.5
Code and Team			96%
1. Are the executing code addresses readily available? (Y/N)	30	Y	30
2. Is the code actively being used? (%)	10	100%	10
3. Is there a public software repository? (Y/N)	5	Y	5
4. Is there a development history visible? (%)	5	100%	5
Is the team public (not anonymous)? (Y/N)	20	Y	20
Code Documentation			
1. Is there a whitepaper? (Y/N)	5	Y	5
2. Are the basic software functions documented? (Y/N)	10	Y	10
3. Does the software function documentation fully (100%) cover the deployed contracts? (%)	15	100%	15
4. Are there sufficiently detailed comments for all functions within the deployed contract code (%)	10	80%	8
5. Is it possible to trace from software documentation to the implementation in code (%)	5	100%	5
Testing			
1. Full test suite (Covers all the deployed code) (%)	20	100%	20
2. Code coverage (Covers all the deployed lines of code, or explains misses) (%)	5	50%	2.5
3. Scripts and instructions to run the tests? (Y/N)	5	Y	5
4. Packaged with the deployed code (Y/N)	5	Y	5
5. Report of the results (%)	10	90%	9
6. Formal Verification test done (%)	5	0%	0
7. Stress Testing environment (%)	5	100%	5
Audits			
Audit done	70	100%	70
Section Scoring			
Executing Code Verification	70	100%	
Documentation	45	96%	
Testing	55	85%	
Audits	70	100%	

Executing Code Appendix

https://docs.bancor.network/ethereum-contracts/addresses

...

Addresses

This section contains the list of the deployed contract addresses on both the Ethereum mainnet and Ropsten testnet.

Below are the core Bancor Network contract addresses that are required to initiate any interaction with the network:

Mainnet

Contract Name	Contract Address
ContractRegistry	0x52Ae12ABe5D8BD778BD5397F99cA900624CfADD4
BNT Token	0x1F573D6Fb3F13d689FF844B4cE37794d79a7FF1C
ETH Token	0xEeeeeEeeeEeEeeEeEeEeEeEeEeEeEeEeEeEeEeE
EtherToken (Deprecated)	0xc0829421C1d2608D3cB3E0F06cFE2D52db2cE315

As of version 28 (released June 16, 2020), EtherToken has been deprecated and replaced with ETH.

Follow the "Working with Bancor Network" guide to have a full view of the relevant contracts that make the network.

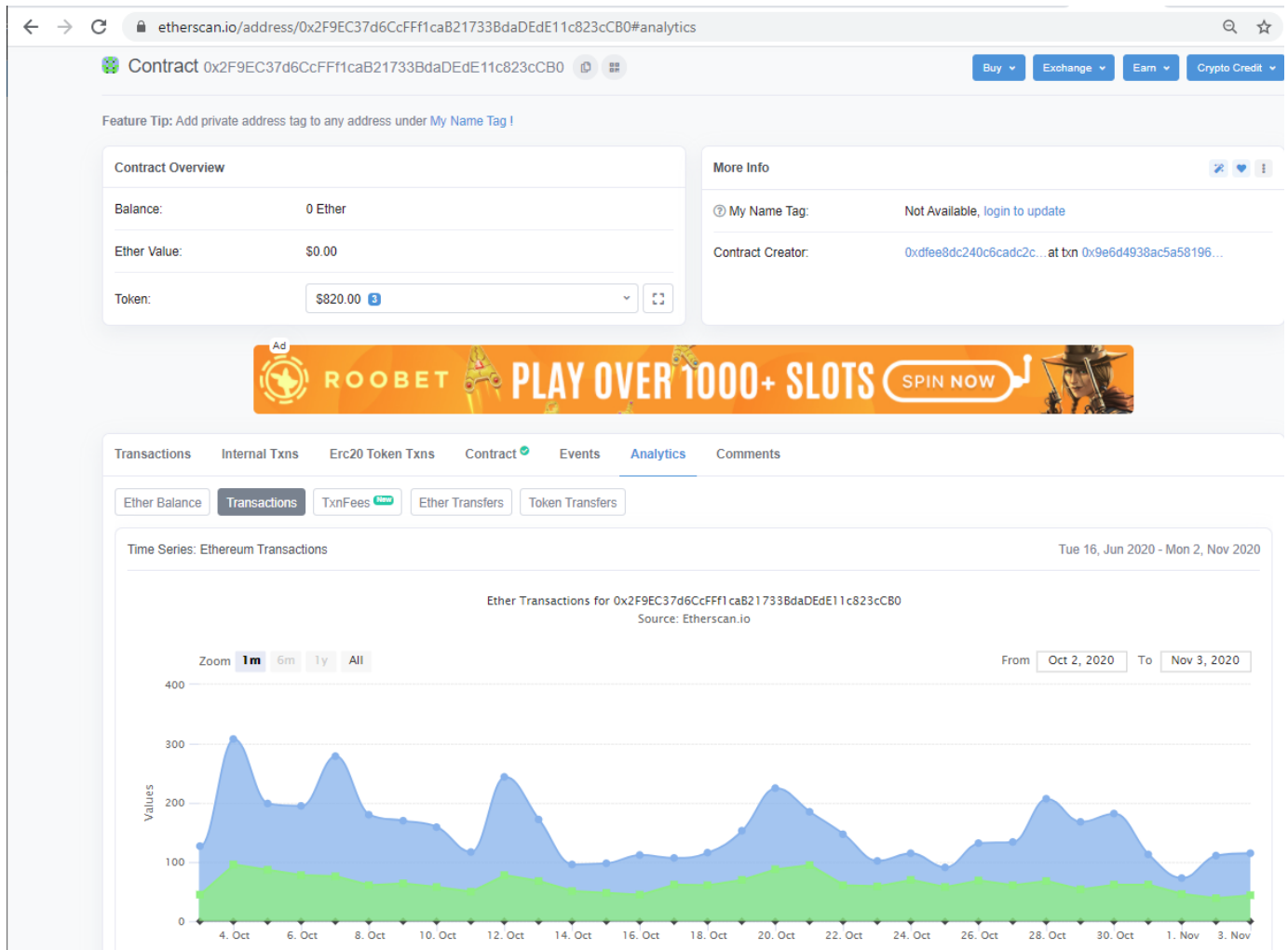
Ropsten

Contract Name	Contract Address
ContractRegistry	0xA6DB4B0963C37Bc959CbC0a874B5bDDF2250f26F
BNT Token	0xF35cCfbCE1228014F66609EDaFCDB836BF388F5

Code Used Appendix

https://docs.defisafety.com/finished-reviews/bancor-process-quality-review

12/17



Example Code Appendix

```

1 // SPDX-License-Identifier: SEE LICENSE IN LICENSE
2 pragma solidity 0.6.12;
3 import "./interfaces/IBancorXUpgrader.sol";
4 import "./interfaces/IBancorX.sol";
5 import "../utility/ContractRegistryClient.sol";
6 import "../utility/SafeMath.sol";
7 import "../utility/TokenHandler.sol";
8 import "../utility/TokenHolder.sol";
9
10 /**
11  * @dev The BancorX contract allows cross chain token transfers.
12  *
13  * There are two processes that take place in the contract -
14  * - Initiate a cross chain transfer to a target blockchain (locks tokens)
15  * - Report a cross chain transfer initiated on a source blockchain (releases tokens)
16  *
17  * Reporting cross chain transfers works similar to standard multisig contracts.
18  * callers are required to report a transfer before tokens are released to the target.
19  */
20 contract BancorX is IBancorX, TokenHandler, TokenHolder, ContractRegistryClient {
21     using SafeMath for uint256;
22

```

```

23 // represents a transaction on another blockchain where tokens were dest
24 struct Transaction {
25     uint256 amount;
26     bytes32 fromBlockchain;
27     address to;
28     uint8 numOfReports;
29     bool completed;
30 }
31
32 uint16 public constant version = 4;
33
34 uint256 public maxLockLimit;           // the maximum amount of tokens
35 uint256 public maxReleaseLimit;        // the maximum amount of tokens
36 uint256 public minLimit;               // the minimum amount of tokens
37 uint256 public prevLockLimit;          // the lock limit *after* the l
38 uint256 public prevReleaseLimit;       // the release limit *after* the
39 uint256 public limitIncPerBlock;        // how much the limit increases
40 uint256 public prevLockBlockNumber;    // the block number of the last
41 uint256 public prevReleaseBlockNumber; // the block number of the last
42 uint8 public minRequiredReports;       // minimum number of required re
43
44 IERC20Token public override token;     // erc20 token
45
46 bool public xTransfersEnabled = true;  // true if x transfers are enab
47 bool public reportingEnabled = true;    // true if reporting is enabled
48
49 // txId -> Transaction
50 mapping (uint256 => Transaction) public transactions;
51
52 // xTransferId -> txId
53 mapping (uint256 => uint256) public transactionIds;
54
55 // txId -> reporter -> true if reporter already reported txId
56 mapping (uint256 => mapping (address => bool)) public reportedTxes;
57
58 // address -> true if address is reporter
59 mapping (address => bool) public reporters;
60
61 /**
62  * @dev triggered when tokens are locked in smart contract
63  *
64  * @param _from    wallet address that the tokens are locked from
65  * @param _amount  amount locked
66 */
67 event TokensLock(
68     address indexed _from,
69     uint256 _amount
70 );
71
72 /**
73  * @dev triggered when tokens are released by the smart contract
74  *
75  * @param _to      wallet address that the tokens are released to
76  * @param _amount  amount released
77 */

```

```
78     event TokensRelease(  
79         address indexed _to,  
80         uint256 _amount  
81     );  
82  
83     /**  
84      * @dev triggered when xTransfer is successfully called  
85      *  
86      * @param _from          wallet address that initiated the xtransfer  
87      * @param _toBlockchain   target blockchain  
88      * @param _to             target wallet  
89      * @param _amount         transfer amount  
90      * @param _id            xtransfer id  
91     */  
92     event XTransfer(  
93         address indexed _from,  
94         bytes32 _toBlockchain,  
95         bytes32 indexed _to,  
96         uint256 _amount,  
97         uint256 _id  
98     );  
99  
100    /**  
101     * @dev triggered when report is successfully submitted  
102     *  
103     * @param _reporter       reporter wallet  
104     * @param _fromBlockchain source blockchain  
105     * @param _txId           tx id on the source blockchain  
106     * @param _to             target wallet  
107     * @param _amount         transfer amount  
108     * @param _xTransferId    xtransfer id  
109    */  
110    event TxReport(  
111        address indexed _reporter,  
112        bytes32 _fromBlockchain,  
113        uint256 _txId,  
114        address _to,  
115        uint256 _amount,  
116        uint256 _xTransferId  
117    );  
118  
119    /**  
120     * @dev triggered when final report is successfully submitted  
121     *  
122     * @param _to target wallet  
123     * @param _id xtransfer id  
124    */  
125    event XTransferComplete(  
126        address _to,  
127        uint256 _id  
128    );  
129  
130    /**  
131     * @dev initializes a new BancorX instance  
132     *
```

```

133      * @param _maxLockLimit          maximum amount of tokens that can be
134      * @param _maxReleaseLimit        maximum amount of tokens that can be
135      * @param _minLimit                minimum amount of tokens that can be
136      * @param _limitIncPerBlock        how much the limit increases per block
137      * @param _minRequiredReports      minimum number of reporters to report
138      * @param _registry                address of contract registry
139      * @param _token                  ERC20 token
140      */
141      constructor(
142          uint256 _maxLockLimit,
143          uint256 _maxReleaseLimit,
144          uint256 _minLimit,
145          uint256 _limitIncPerBlock,
146          uint8 _minRequiredReports,
147          IContractRegistry _registry,
148          IERC20Token _token
149      ) ContractRegistryClient(_registry)
150      public
151      greaterThanZero(_maxLockLimit)
152      greaterThanZero(_maxReleaseLimit)
153      greaterThanZero(_minLimit)
154      greaterThanZero(_limitIncPerBlock)
155      greaterThanZero(_minRequiredReports)
156      validAddress(address(_token))
157      notThis(address(_token))
158      {
159          // validate input
160          require(_minLimit <= _maxLockLimit && _minLimit <= _maxReleaseLimit);
161
162          // the maximum limits, minimum limit, and limit increase per block
163          maxLockLimit = _maxLockLimit;
164          maxReleaseLimit = _maxReleaseLimit;
165          minLimit = _minLimit;
166          limitIncPerBlock = _limitIncPerBlock;
167          minRequiredReports = _minRequiredReports;
168
169          // previous limit is _maxLimit, and previous block number is current
170          prevLockLimit = _maxLockLimit;
171          prevReleaseLimit = _maxReleaseLimit;
172          prevLockBlockNumber = block.number;
173          prevReleaseBlockNumber = block.number;
174
175          token = _token;
176      }
177
178      // validates that the caller is a reporter
179      modifier reporterOnly {
180          _reporterOnly();
181          _;
182      }
183
184      // error message binary size optimization
185      function _reporterOnly() internal view {
186          require(reporters[msg.sender], "ERR_ACCESS_DENIED");
187      }

```



```
188
189 // allows execution only when x transfers are enabled
190 modifier xTransfersAllowed {
191     _xTransfersAllowed();
192     _;
193 }
194
195 // error message binary size optimization
196 function _xTransfersAllowed() internal view {
197     require(xTransfersEnabled, "ERR_DISABLED");
198 }
199
200 // allows execution only when reporting is enabled
201 modifier reportingAllowed {
202     _reportingAllowed();
203     _;
204 }
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complexity
Solidity	14	6663	744	2517	3402	396

Comments to Code 2517/ 3402 = 73%

Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complexity
JavaScript	45	13798	2773	171	10854	731

Tests to Code 10854 / 3402 = 319%