0.7

Bancor Process Quality Review

Score: 90%

Overview

This is a Bancor Process Quality Review completed on July 21st 2021. It was performed using the Process Review process (version 0.7.3) and is documented here. The review was performed by Nic of DeFiSafety. Check out our Telegram. The previous review (0.6) is here.

The final score of the review is 90%, an excellent 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

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 Smart Chain
Polygon
Avalanche

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.bancor.network/developer-quick-start/working-with-bancor-network, 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 over 10 transactions a day on contract *ContractRegistry.sol*, 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/bancorprotocol.

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%

With 4836 commits and 11 branches, this is a healthy software repository.

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://icobench.com/ico/bancor/team.

For a **"Yes"** in this question, the real names of some team members must be public on the website or other documentation (LinkedIn, etc). If the team is anonymous, 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://storage.googleapis.com/website-bancor/2018/04/01ba8253-bancor_protocol_whitepaper_en.pdf.

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

✓ **Answer:** Yes

The basic software functions of Bancor Protocol are well documented in "Developer Quick Start".

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

Answer: 100%

Every single software function of Bancor Protocol is fully documented. Developer, architecture, API, SDK, and other.

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: 71%

Code examples are in the Appendix. As per the SLOC, there is 71% commenting to code (CtC).

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

Note: The CtC was calculated using code that was authored by Bancor, and did not include Interface files, or any third party resources in our calculations (ex: SafeMath).

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%

There is clear and explicit traceability between Bancor Protocol's documented software functions and their subsequent implementation in their source code. Good examples of this traceability are the Developer Quick Start, and the API Reference.

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 261% 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:** 90%

Bancor has received v2 code coverage from ConsenSys Diligence in their audit report. However, it is not the full coverage and does not explain misses or uncovered lines.

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

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

✓ **Answer:** Yes

Scrips/Instructions location: https://github.com/bancorprotocol/contracts-solidity/blob/master/README.md.

14) Report of the results (%)

✓ **Answer:** 100%

Detailed test report was found here, as well as a more extensive report in the ConsenSys Diligence audit report.

Guidance:

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

15) Formal Verification test done (%)

⚠ **Answer:** 0%

No evidence of a Bancor Formal Verification has been found in their documentation or in web searches.

16) Stress Testing environment (%)

✓ **Answer:** 100%

There is evidence of Bancor Protocol's Ropsten test-net usage at https://docs.bancor.network/developer-quick-start/working-with-bancor-network#contract-names-and-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%

Bancor Protocol has had multiple audits before deployment, both V1 and V2, as well as 2.1. A full list of reports can be found here.

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

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:** 70%

Bancor's Bug Bounty program is active and offers up to 100k for the most critical of 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:** 100%

The Bancor Protocol governance portal is clearly indicated on their website.

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 upgradeable through the DAO. 30%

b) Bancor has a multisig through which it appears DAO changes are implemented 20%

c) Capabilities for change not clearly described. 0%

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%

All governance and access control-related information is usually explained in user-friendly words.

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%

There is no evidence of Pause Control or a similar function documented in the Bancor documentation.

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

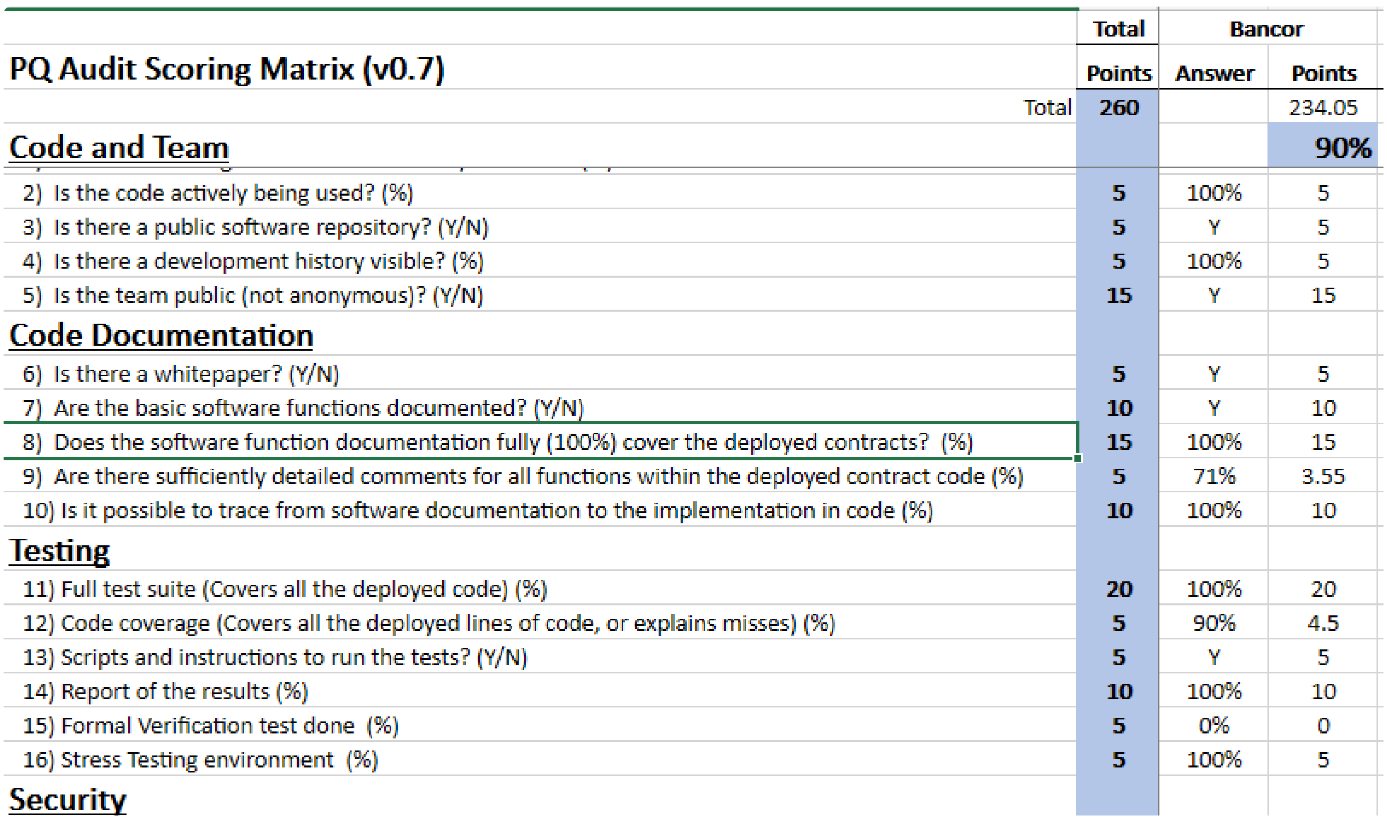
I started with Ethereum just before the DAO and that was a wonderful education. It showed the importance of code quality. The second Parity hack also showed the importance of good process. Here my aviation background offers some value. Aerospace knows how to make reliable code using quality processes.

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



	Total	Bancor	
PQ Audit Scoring Matrix (v0.7)	**Points**	**Answer**	**Points**
Total	**260**		234.05
Code and Team			**90%**
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**	71%	3.55
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**	90%	4.5
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**	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**	70%	7
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**	50%	5
21) Is the information in non-technical terms	**10**	90%	9
22) Is there Pause Control documentation including records of tests	**10**	0%	0
Section Scoring			
Code and Team	50	100%	
Documentation	45	97%	
Testing	50	89%	
Security	80	96%	
Access Controls	35	54%	

Executing Code Appendix

Contract Names and Addresses

ContractRegistry address

Network	Contract Address
`Mainnet`	`0x52Ae12ABe5D8BD778BD5397F99cA900624CfADD4`
`Ropsten`	`0xA6DB4B0963C37Bc959CbC0a874B5bDDf2250f26F`

ContractRegistry also indicates its own address, which means you should run the same process to make sure you are working with the latest version of the ContractRegistry

List of Contracts Names and Bytes32 Representation

Contract Name	bytes32 Representation
`ContractRegistry`	`0x436f6e74726163745265676973747279`
`BancorNetwork`	`0x42616e636f724e6574776f726b`
`BancorFormula`	`0x42616e636f72466f726d756c61`
`BancorConverterRegistry`	`0x42616e636f72436f6e7665727465725265676973747279`
`BancorNetworkPathFinder`	`0x42616e636f724e6574776f726b5061746846696e646572`

Code Used Appendix

Parent Txn Hash	Block	Age	From		To
0xd423fc81be73ff59ea4...	12871189	1 min ago	0x5dafb315d9c358d628f...	→	0x52ae12abe5d8bd778...
0xd423fc81be73ff59ea4...	12871189	1 min ago	0xc0205e203f423bcd8b...	→	0x52ae12abe5d8bd778...
0xd423fc81be73ff59ea4...	12871189	1 min ago	0xf7d28faa1fe9ea53279...	→	0x52ae12abe5d8bd778...
0x783e767becb1d7e916...	12871189	1 min ago	0xcba47689202d31575b...	→	0x52ae12abe5d8bd778...
0x783e767becb1d7e916...	12871189	1 min ago	0x2f9ec37d6ccfff1cab21...	→	0x52ae12abe5d8bd778...
0x18c70a3bf3052cea03...	12871181	3 mins ago	0x6a74941c1cf4151b3f1...	→	0x52ae12abe5d8bd778...
0x18c70a3bf3052cea03...	12871181	3 mins ago	0x2f9ec37d6ccfff1cab21...	→	0x52ae12abe5d8bd778...
0xed6477c954ac22f8e0...	12871181	3 mins ago	0x6a74941c1cf4151b3f1...	→	0x52ae12abe5d8bd778...
0xed6477c954ac22f8e0...	12871181	3 mins ago	0x2f9ec37d6ccfff1cab21...	→	0x52ae12abe5d8bd778...
0x09aac429eec090b841...	12871179	3 mins ago	0xe31643ebd9fd29b668...	→	0x52ae12abe5d8bd778...
0x09aac429eec090b841...	12871179	3 mins ago	0x2f9ec37d6ccfff1cab21...	→	0x52ae12abe5d8bd778...
0xcd6f74cdfd222a6fdfe1...	12871179	3 mins ago	0xab7ae646063087317c...	→	0x52ae12abe5d8bd778...
0xcd6f74cdfd222a6fdfe1...	12871179	3 mins ago	0x2f9ec37d6ccfff1cab21...	→	0x52ae12abe5d8bd778...

Example Code Appendix

```
/**
 * @dev This contract maintains contract addresses by name.
 *
 * The owner can update contract addresses so that a contract name always points to the la
 * of the given contract.
 *
 * Other contracts can query the registry to get updated addresses instead of depending on
 * addresses.
 *
 * Note that contract names are limited to 32 bytes UTF8 encoded ASCII strings to optimize
 */
contract ContractRegistry is IContractRegistry, Owned, Utils {
    struct RegistryItem {
        address contractAddress; // contract address
        uint256 nameIndex; // index of the item in the list of contract names
    }

    mapping(bytes32 => RegistryItem) private items; // name -> RegistryItem mapping
    string[] public contractNames; // list of all registered contract names

    /**
     * @dev triggered when an address pointed to by a contract name is modified
     *
     * @param _contractName    contract name
     * @param _contractAddress new contract address
```

```solidity
     */
    event AddressUpdate(bytes32 indexed _contractName, address _contractAddress);

    /**
     * @dev returns the number of items in the registry
     *
     * @return number of items
     */
    function itemCount() public view returns (uint256) {
        return contractNames.length;
    }

    /**
     * @dev returns the address associated with the given contract name
     *
     * @param _contractName    contract name
     *
     * @return contract address
     */
    function addressOf(bytes32 _contractName) public view override returns (address) {
        return items[_contractName].contractAddress;
    }

    /**
     * @dev registers a new address for the contract name in the registry
     *
     * @param _contractName     contract name
     * @param _contractAddress  contract address
     */
    function registerAddress(bytes32 _contractName, address _contractAddress)
        public
        ownerOnly
        validAddress(_contractAddress)
    {
        // validate input
        require(_contractName.length > 0, "ERR_INVALID_NAME");

        // check if any change is needed
        address currentAddress = items[_contractName].contractAddress;
        if (_contractAddress == currentAddress) {
            return;
        }

        if (currentAddress == address(0)) {
            // update the item's index in the list
            items[_contractName].nameIndex = contractNames.length;

            // add the contract name to the name list
            contractNames.push(bytes32ToString(_contractName));
        }

        // update the address in the registry
        items[_contractName].contractAddress = _contractAddress;

        // dispatch the address update event
        emit AddressUpdate(_contractName, _contractAddress);
    }

    /**
     * @dev removes an existing contract address from the registry
     *
     * @param _contractName contract name
     */
    function unregisterAddress(bytes32 _contractName) public ownerOnly {
        // validate input
        require(_contractName.length > 0, "ERR_INVALID_NAME");
        require(items[_contractName].contractAddress != address(0), "ERR_INVALID_NAME");

        // remove the address from the registry
        items[_contractName].contractAddress = address(0);

        // if there are multiple items in the registry, move the last element to the delet
        // and modify last element's registryItem.nameIndex in the items collection to poi
        if (contractNames.length > 1) {
            string memory lastContractNameString = contractNames[contractNames.length - 1]
            uint256 unregisterIndex = items[_contractName].nameIndex;

            contractNames[unregisterIndex] = lastContractNameString;
            bytes32 lastContractName = stringToBytes32(lastContractNameString);
            RegistryItem storage registryItem = items[lastContractName];
            registryItem.nameIndex = unregisterIndex;
        }

        // remove the last element from the name list
        contractNames.pop();
        // zero the deleted element's index
        items[_contractName].nameIndex = 0;

        // dispatch the address update event
        emit AddressUpdate(_contractName, address(0));
    }

    /**
     * @dev utility, converts bytes32 to a string
     * note that the bytes32 argument is assumed to be UTF8 encoded ASCII string
     *
     * @return string representation of the given bytes32 argument
     */
    function bytes32ToString(bytes32 _bytes) private pure returns (string memory) {
        bytes memory byteArray = new bytes(32);
        for (uint256 i = 0; i < 32; i++) {
            byteArray[i] = _bytes[i];
        }

        return string(byteArray);
    }

    /**
     * @dev utility, converts string to bytes32
     * note that the bytes32 argument is assumed to be UTF8 encoded ASCII string
     *
     * @return string representation of the given bytes32 argument
     */
    function stringToBytes32(string memory _string) private pure returns (bytes32) {
        bytes32 result;
        assembly {
            result := mload(add(_string, 32))
        }
        return result;
    }
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Solidity	37	11025	1295	4050	5680	546

Comments to Code 4050/5680 = 71%

Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
JavaScript	62	17736	2689	231	14816	708

Tests to Code 14816/5680 = 261%