

Gnosis Protocol DEX Quality Review

Score: 91%

This is a Process Quality Review of the [Gnosis DEX](#) completed on 19 January, 2021. It was performed using the Process Review process (version 0.6.1) and is documented [here](#). The review was performed by ShinkaRex of DeFiSafety. Check out our [Telegram](#).

The final score of the review is 91%, 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, token, future, option or other financial instrument or to offer or provide any investment advice or service to any person in any jurisdiction. Nothing contained in this report constitutes investment advice or offers any opinion with respect to the suitability of any security, and the views expressed in this report should not be taken as advice to buy, sell or hold any security. The information in this report should not be relied upon for the purpose of investing. In preparing the information contained in this report, we have not taken into account the investment needs, objectives and financial circumstances of any particular investor. This information has no regard to the specific investment objectives, financial situation and particular needs of any specific recipient of this information and investments discussed may not be suitable for all investors.

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


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

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://github.com/gnosis/dex-contracts/blob/master/networks.json> as indicated in the [Appendix](#). It would be better if this information was in a text file that is easier to read.

Is the code actively being used? (%)

 Answer: 100%

Activity is 29 transactions a day on contract BatchExchange.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

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

 Answer: Yes

GitHub: <https://github.com/gnosis>

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 985 commits and 20 branches in the dex-protocols repository, this is a healthy repo.

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

Information about the Gnosis founders can be found on their [career page](#).

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;

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: <https://docs.gnosis.io/protocol/docs/devguide01/>

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

 Answer: Yes

There is function documentation in their [developer guide](#).

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

 Answer: 80%

All contracts and functions are documented.


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.

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

 Answer: 46%

Gnosis has some detailed comments, but most of the contract functions do not include any.

Code examples are in the [Appendix](#). As per the [SLOC](#), there is 46% 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: 60%

The documentation lists the functions and describes them, but there is limited amounts of clear association between the code and 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

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;

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%

Gnosis's test to code ratio (TtC) is 670, demonstrating a robust testing suite.

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: 100%

Their overall report reports a 100% coverage.

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

Scripts to run testing can be found on their GitHub readme.

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.

Packaged with the deployed code (Y/N)

 Answer: Yes

How to improve this score

Improving this score requires redeployment of the code, with the tests. This score gives credit to those who test their code before deployment and release them together. If a developer adds tests after deployment they can gain full points for all test elements except this one.

Report of the results (%)

 Answer: 100%

Location: <https://travis-ci.com/github/gnosis/dex-contracts>

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.

Formal Verification test done (%)

 Answer: 0%

There has been no formal verification testing done on the DEX contracts.

Stress Testing environment (%)

 Answer: 100%

[Contracts on rinkby have been clearly stress-tested.](#)

Audits

 Answer: 90%

[G0 Group](#) has conducted an audit on Jan 27th, 2020.

Gnosis protocol was officially launched Apr 15th, 2020.

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 DeFi Safety.

Email : rex@defisafety.com Twitter : [@defisafety](https://twitter.com/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	Gnosis	
	Points	Answer	Points
Total	240		217.6
Code and Team			91%
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	80%	12
4. Are there sufficiently detailed comments for all functions within the deployed contract code (%)	10	46%	4.6
5. Is it possible to trace from software documentation to the implementation in code (%)	5	60%	3
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	100%	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	100%	10
6. Formal Verification test done (%)	5	0%	0
7. Stress Testing environment (%)	5	100%	5
Audits			
Audit done	70	90%	63
Section Scoring			
Code and Team	70	100%	
Documentation	45	77%	
Testing	55	91%	
Audits	70	90%	

Executing Code Appendix

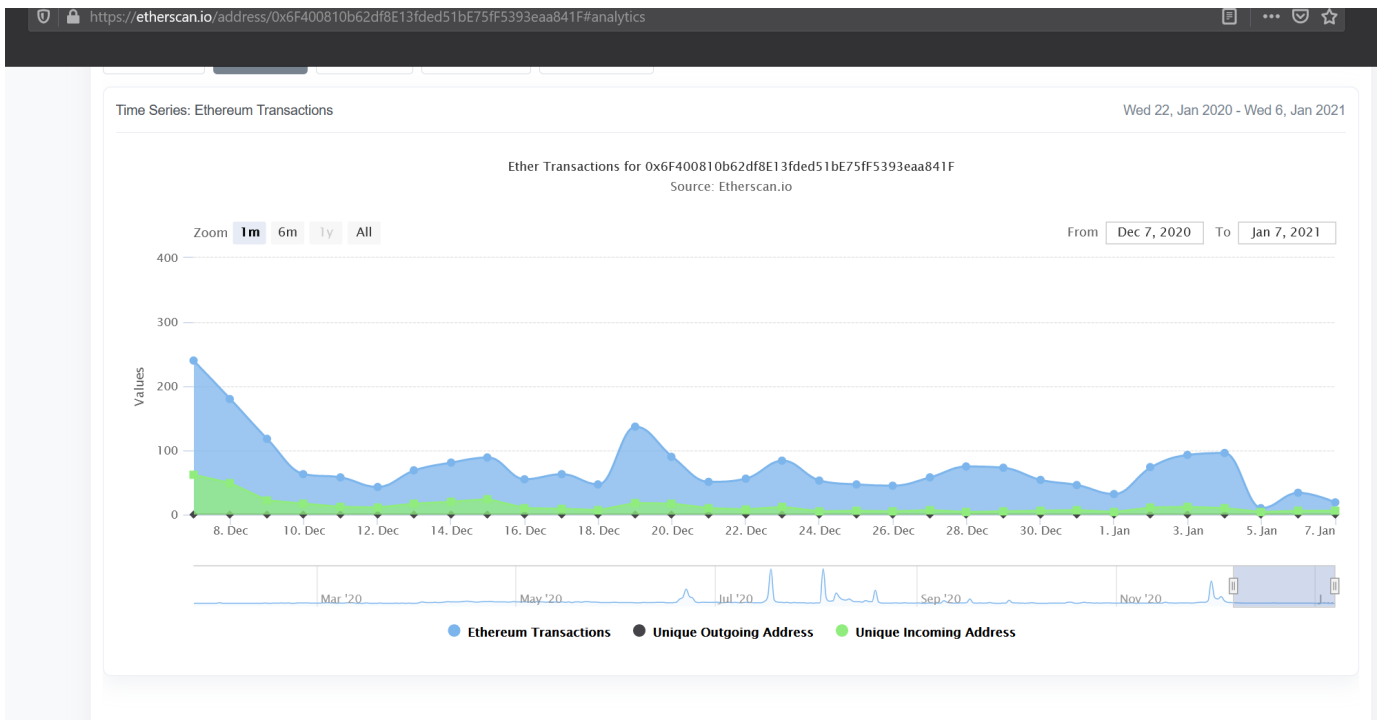
https://github.com/gnosis/dex-contracts/blob/master/networks.json

```

2   "BatchExchange": {
3     "1": {
4       "links": {
5         "IdToAddressBiMap": "0xED4d05496C71e71c2A8726af1242C22108d1761",
6         "IterableAppendOnlySet": "0xCDB32b6Bb2808D5B5115dAab207479c98d2636"
7       },
8       "address": "0x6F400810b62df8E13fded51bE75fF5393eaa841F",
9       "transactionHash": "0xe74e8e965afc67f96a38caee794f008be33a7fe7ea96b9baad7b2963250ac36a"
10    },
11    "4": {
12      "links": {
13        "IdToAddressBiMap": "0x5c4C6bf91240A5fdBf89a18ED8d43227046e2feA",
14        "IterableAppendOnlySet": "0x0D47D0548FDAD66B06E81a826EED8c687aCdd8CB"
15      },
16      "address": "0xC576eA7bd102F7E476368a5E98FA455d1Ea34dE2",
17      "transactionHash": "0xc5bf4f48747093a79b623c2a2392ccbdb73a3d788e7d1f0f53f640ea18496d90"
18    },
19    "100": {
20      "links": {
21        "IdToAddressBiMap": "0x048E53A455a058462eA58442E1D94Fbc955495cB",
22        "IterableAppendOnlySet": "0x57E6B987c2ccd421859A244d22a0D5a62D88f91"
23      },
24      "address": "0x25B06305CC4ec6AfcF3E7c0b673da1EF8ae26313",
25      "transactionHash": "0xd5e7da686ace1ae7f1917ad54b8610ee936bdc900e56b495739795fba3d5e7c1"
26    }
27  },
28  "BatchExchangeViewer": {
29    "1": {
30      "links": {},
31      "address": "0x7D071fb584b51D9c38572A04E1848afd835cD457",
32      "transactionHash": "0xd1be3b9bfc2bab273e3893ebc56c2925c2e41ab70d96694705bd90bedcfd1c3"
33    },
34    "4": {
35      "links": {},
36      "address": "0x0A7a7E35098b5F969E5d45680827AC478e6e8Ce3",
37      "transactionHash": "0x0e2ea4e0550d0b960aacc10576ac7e07c5285696f08a0b4b8f608f5b3f098a4f"
38    },
39    "100": {
40      "links": {},
41      "address": "0x51138aad07EbA8a6a1D1254cf740Df97B96bFB64",
42      "transactionHash": "0x4afb81a41e4d7633046302eece8b7a38fc6dc304d3e35db55c150c31fd3eb60"
43    }
44  },
45  "Migrations": {
46    "1": {
47      "links": {},
48      "address": "0xA3B3FBc0225f2f7cb1cF767e2bc566FA0Be4Ce9E",
49      "transactionHash": "0x0a5afd86da6b7e12d0534213c2c1a81275e8a0aeb280beb8a0f4ce1ce6ca2faa"
50    },
51  },

```

Code Used Appendix



Example Code Appendix

```

1 pragma solidity ^0.5.10;
2
3 import "@openzeppelin/contracts/math/SafeMath.sol";
4 import "@openzeppelin/contracts/token/ERC20/ERC20Detailed.sol";
5 import "solidity-bytes-utils/contracts/BytesLib.sol";
6 import "./BatchExchange.sol";
7
8 contract BatchExchangeViewer {
9     using BytesLib for bytes;
10    using SafeMath for uint256;
11
12    uint8 public constant AUCTION_ELEMENT_WIDTH = 112;
13    // Contains the orderId on top of the normal auction element data
14    uint8 public constant INDEXED_AUCTION_ELEMENT_WIDTH = AUCTION_ELEMENT_WI
15    uint8 public constant ADDRESS_WIDTH = 20;
16    uint16 public constant LARGE_PAGE_SIZE = 5000;
17    // Can be used by external contracts to indicate no filter as it doesn't
18    // to create an empty memory array in solidity.
19    uint16[] public ALL_TOKEN_FILTER;
20
21    BatchExchange batchExchange;
22
23    constructor(BatchExchange exchange) public {
24        batchExchange = exchange;
25    }
26
27    /** @dev Queries the orderbook for the auction that is still accepting (
28     * @param tokenFilter all returned order will have buy *and* sell token
29     * @return encoded bytes representing orders, maxed at 5000 elements
30     */

```

```

31     function getOpenOrderBook(address[] memory tokenFilter) public view returns
32         (bytes memory elements, bool, address) = getOpenOrderBookPaginated(tokenFilter);
33     require(
34         elements.length < uint256(LARGE_PAGE_SIZE) * INDEXED_AUCTION_ELEMENTS,
35         "Orderbook too large, use paginated view functions"
36     );
37     return elements;
38 }
39
40 /** @dev Queries a page of the orderbook for the auction that is still active.
41     * @param tokenFilter all returned order will have buy *and* sell token
42     * @param previousPageUser address taken from nextPageUser return value
43     * @param previousPageUserOffset offset taken from nextPageUserOffset return value
44     * @param maxPageSize count of elements to be returned per page (same as maxOrderSize)
45     * @return encoded bytes representing orders and page information for the page
46     */
47     function getOpenOrderBookPaginated(
48         address[] memory tokenFilter,
49         address previousPageUser,
50         uint16 previousPageUserOffset,
51         uint16 maxPageSize
52     )
53     public
54     view
55     returns (
56         bytes memory elements,
57         bool hasNextPage,
58         address nextPageUser,
59         uint16 nextPageUserOffset
60     )
61     {
62         uint32 batch = batchExchange.getCurrentBatchId();
63         return
64             getFilteredOrdersPaginated(
65                 [batch, batch, batch + 1],
66                 getTokenIdsFromAddresses(tokenFilter),
67                 previousPageUser,
68                 previousPageUserOffset,
69                 maxPageSize
70             );
71     }
72
73 /** @dev Queries the orderbook for the auction that is currently being finalized.
74     * @param tokenFilter all returned order will have buy *and* sell token
75     * @return encoded bytes representing orders, maxed at 5000 elements
76     */
77     function getFinalizedOrderBook(address[] memory tokenFilter) public view returns
78         (bytes memory elements, bool, address) = getFinalizedOrderBookPaginated(tokenFilter);
79     require(
80         elements.length < uint256(LARGE_PAGE_SIZE) * INDEXED_AUCTION_ELEMENTS,
81         "Orderbook too large, use paginated view functions"
82     );
83     return elements;
84 }
85

```

```

86  /** @dev Queries a page of the orderbook for the auction that is current
87  * @param tokenFilter all returned order will have buy *and* sell token
88  * @param previousPageUser address taken from nextPageUser return value
89  * @param previousPageUserOffset offset taken nextPageUserOffset return value
90  * @param maxPageSize count of elements to be returned per page (same as
91  * @return encoded bytes representing orders and page information for
92  */
93  function getFinalizedOrderBookPaginated(
94      address[] memory tokenFilter,
95      address previousPageUser,
96      uint16 previousPageUserOffset,
97      uint16 maxPageSize
98  )
99      public
100     view
101     returns (
102         bytes memory elements,
103         bool hasNextPage,
104         address nextPageUser,
105         uint16 nextPageUserOffset
106     )
107 {
108     uint32 batch = batchExchange.getCurrentBatchId();
109     return
110         getFilteredOrdersPaginated(
111             [batch - 1, batch - 1, batch],
112             getTokenIdsFromAddresses(tokenFilter),
113             previousPageUser,
114             previousPageUserOffset,
115             maxPageSize
116         );
117 }
118
119 /** @dev Queries a page in the list of all orders
120 * @param batchIds Triple with the following values [maxValidFrom, minValidUntil,
121 * Batched together as we are running out of local variables (Solidity
122 *     - maxValidFrom: all returned orders will have a validFrom <= the
123 *     - minValidUntil all returned orders will have a validUntil >= the
124 *     - sellBalanceTargetBatchIndex the batchIndex at which we are expecting
125 * (e.g. in the current live orderbook we want to include sellBalances
126 * @param tokenFilter all returned order will have buy *and* sell token
127 * @param previousPageUser address taken from nextPageUser return value
128 * @param previousPageUserOffset offset taken nextPageUserOffset return value
129 * @param maxPageSize maximum count of elements to be returned per page
130 * @return encoded bytes representing orders and page information for
131 * maxPageSize if remaining gas is low.
132 */

```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complexity
Solidity	7	1817	144	529	1144	124

Comments to Code 529/1144 = 46%

Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complexity
Typescript	21	8610	647	294	7669	111

Tests to Code 7669/1144 = 670%