

# 0.5

## mStable Process Quality Review

Score: 97%

This is an mStable Process Quality Review completed on 21 October 2020. It was performed using the Process Review process (version 0.5) and is documented [here](#). The review was performed by ShinkaRex of [Caliburn Consulting](#). Check out our [Telegram](#).

The final score of the review is 97%, a stupendous 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.

1. **Here is my smart contract on the blockchain**
2. **You can see it matches a software repository used to develop the code**
3. **Here is the documentation that explains what my smart contract does**
4. **Here are the tests I ran to verify my smart contract**
5. **Here are the audit(s) performed to review 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.

---

# Executing Code Verification

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 address(s) readily available? (Y/N)
2. Is the code actively being used? (%)
3. Are the Contract(s) Verified/Verifiable? (Y/N)
4. Does the code match a tagged version in the code hosting platform? (%)
5. Is the software repository healthy? (%)

## Are the executing code address(s) readily available? (Y/N)

✔ Answer: Yes

They are available at Address <https://docs.mstable.org/developers/deployed-addresses> as indicated in the [Appendix](#). This review only covers the contract IncentivisedVotingLockup.sol.

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

✔ Answer: 100%

Activity is 38 transactions a day, as indicated in the [Appendix](#).

## Are the Contract(s) Verified/Verifiable? (Y/N)

✔ Answer: Yes

0xaE8bC96DA4F9A9613c323478BE181FDdb2Aa0E1BF is the Etherscan verified contract address.

## Does the code match a tagged version on a code hosting platform? (%)

✔ Answer: 100%

Guidance:

- 100% All code matches and Repository was clearly labelled
- 60 % All code matches but no labelled repository. Repository was found manually
- 30% Almost all code does match perfectly and repository was found manually
- 0% Most matching Code could not be found

GitHub address : <https://github.com/mstable/mstable-contracts>

Deployed contracts in the following file;

 mstable\_deployed.rar 21KB  
Binary

Matching Repository: <https://github.com/mstable/mStable-contracts/tree/master/contracts>

### Is development software repository healthy? (%)

✔ Answer: 100%

With 4 branches and 172 commits, this is a healthy Repository.

---

## 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 application requirements documented? (Y/N)
3. Do the requirements 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 software requirements to the implementation in code (%)

### Is there a whitepaper? (Y/N)

✔ Answer: Yes

The whitepaper is featured in the documentation.

Location: <https://docs.mstable.org/>

### Are the basic application requirements documented? (Y/N)

✔ Answer: Yes

Location: <https://docs.mstable.org/developers/integrating-mstable>

The basic functions are documented on this webpage. By consulting the individual pages within these documents you can find the in-depth documentation that goes over the most of the functions within their software.

### Do the requirements fully (100%) cover the deployed contracts? (%)

✔ Answer: 100%

I found specific document requirements about the deployed contracts in the documentation. However, there are still some functions that remain undefined in their documentation. The documentation is extensive, and extremely high quality, but not 100% complete.

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

✔ Answer: 100%

Very detailed comments throughout. NatSpec in detail at the start of every function and comments within the function where required.

Code examples are in the [Appendix](#). As per the [SLOC](#), there is 126% commenting to code.

### Is it possible to trace requirements to the implementation in code (%)

✔ Answer: 100%

Clear Traceability 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

---

## 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%

Tests are clearly indicated on the GitHub along with test setups.

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

✔ Answer: 98.6%

Full coverage is documented in the audit done by [Consensys](#), page 28.

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

Location: <https://docs.mstable.org/developers/introduction/get-set-up-on-ropsten>

### Packaged with the deployed code (Y/N)

✓ Answer: Yes

### Report of the results (%)

✓ Answer: 100%

There is a report of the results in their [audit from Consensys Diligence](#).

### Formal Verification test done (%)

⚠ Answer: 0%

There is no evidence of formal verification testing done.

### Stress Testing environment (%)

✓ Answer: 100%

They have clear instructions for [ropsten setup](#).

---

## Audits

✓ Answer: 100%

Two audits have been performed by [Consensys Diligence and Bramah Systems](#). The Audit by Consensys diligence was performed before deployment. (July 2020) The audit from Bramah Systems was performed before deployment. (April 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

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](mailto: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

Q Audit Scoring Matrix (v0.4 and 0.5)	Total	mStable	
	Points	Answer	Points
Total	240		233.443
<b><u>Executing Code Verification</u></b>			<b>97%</b>
. Is the executing code address(s) readily available? (Y/N)	30	Y	30
. Is the code actively being used? (%)	5	100%	5
. Are the Contract(s) Verified/Verifiable? (Y/N)	5	Y	5
. Does the code match a tagged version on a code hosting platform? (%)	20	100%	20
. Is development software repository healthy? (%)	10	100%	10
<b><u>Code Documentation</u></b>			
. Is there a whitepaper? (Y/N)	5	Y	5
. Are the basic application requirements documented? (Y/N)	10	Y	10
. Do the requirements fully (100%) cover the deployed contracts? (%)	15	90%	13.5
. Are there sufficiently detailed comments for all functions within the deployed contract code (%)	10	100%	10
. Is it possible to trace requirements to the implementation in code (%)	5	100%	5
<b><u>Testing</u></b>			
. Full test suite (Covers all the deployed code) (%)	20	100%	20
. Code coverage (Covers all the deployed lines of code, or explains misses) (%)	5	99%	4.943
. Scripts and instructions to run the tests? (Y/N)	5	Y	5
. Packaged with the deployed code (Y/N)	5	Y	5
. Report of the results (%)	10	100%	10
. Formal Verification test done (%)	5	0%	0
. Stress Testing environment (%)	5	100%	5
<b><u>Audits</u></b>			
Audit done	70	100%	70

<b>action Scoring</b>		
Executing Code Verification	70	100%
Documentation	45	97%
Testing	55	91%
Audits	70	100%
State		21-Oct-20
Private Repo		Y
Version		0.5

## Executing Code Appendix

docs.mstable.org/developers/deployed-addresses

**mStable** [Twitter](#) [Discord](#) [Contact](#)

- Getting Started
- MSTABLE
  - mASSETS >
  - Meta >
  - Interfacing with mStable >
- META REWARDS AND GRANTS
  - Public MTA Rewards >
  - Grants Program
- DEVELOPERS
  - Introduction >
  - mStable-protocol >
  - mStable-js
  - Deployed addresses**
- PROTOCOL
  - mStable Improvement Proposals

Powered by GitBook

### Deployed addresses

List of deployed contract addresses

Mainnet **Ropsten** Kovan

Contract	Address
mUSD	<a href="#">0xe2f2a5c287993345a840db3b0845fbc70f5935a5</a>
mUSD SAVE	<a href="#">0xcf3f73290803fc04425bee135a4caeb2bab2c2a1</a>
mStable Helper	<a href="#">0xe15aad5d6b7433e5988415274529311f6bf6e8a3</a>
mUSD Basket Manager	<a href="#">0x66126b4a2a1c07536ef8e5e8bd4efda1fdEA96D</a>
MTA	<a href="#">0xa3bed4e1c75d00fa6f4e5e6922db7261b5e9acd2</a>
MTA Staking	<a href="#">0xae8bc96da4f9a9613c323478be181fdb2aa0e1bf</a>
Nexus	<a href="#">0xAfCE80b19A8cE13DEc0739a1aaB7A028d6845Eb3</a>
Proxy Admin	<a href="#">0x5C8eb57b44C1c6391fC7a8A0cf44d26896f92386</a>

## Code Used Appendix

https://etherscan.io/address/0x0EC1f1573f3a2dB0Ad396c843E6a079e2a53e557#analytics

Getting Started  Process Quality Review...  
 :ether value: \$0.00 Contract Creator: [0x142151d3b15b8961f...](#) at txn [0xe56f5bca8964365d8...](#)

oken:

\$/6,333.84 47

Ad  
Up To \$170 Welcome Bonus [Grab Now](#)   

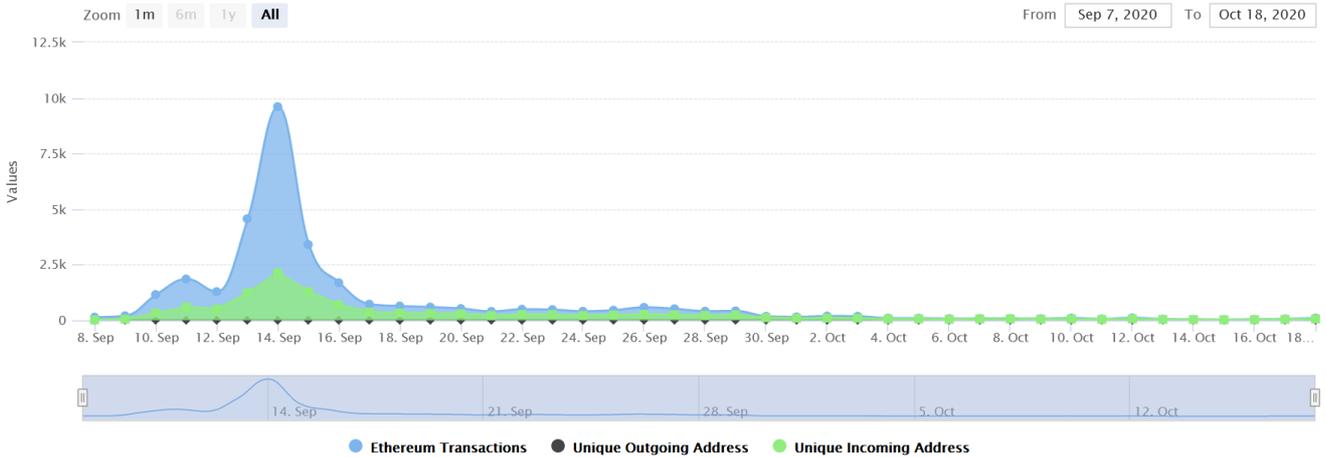
Transactions Internal Txns Erc20 Token Txns Contract 👍 Events **Analytics** Comments

Ether Balance **Transactions** TxnFees New Ether Transfers Token Transfers

Time Series: Ethereum Transactions

Mon 7, Sept 2020 - Sat 17, Oct 2020

Ether Transactions for 0x0EC1f1573f3a2dB0Ad396c843E6a079e2a53e557  
Source: Etherscan.io



 This website uses cookies to improve your experience and has an updated [Privacy Policy](#). [Got It](#)

## Example Code Appendix

```
1  /**
2   * @dev Sets the address of the proxy admin.
3   * @param newAdmin Address of the new proxy admin.
4   */
5  function _setAdmin(address newAdmin) internal {
6      bytes32 slot = ADMIN_SLOT;
7
8      assembly {
9          sstore(slot, newAdmin)
10     }
11 }
12
13 /**
14 * @dev Only fall back when the sender is not the admin.
15 */
16 function _willFallback() internal {
17     require(msg.sender != _admin(), "Cannot call fallback function from the proxy admin");
18     super._willFallback();
19 }
20 }
21
22 /**
23 * @title InitializableUpgradeabilityProxy
```

```

24 * @dev Extends BaseUpgradeabilityProxy with an initializer for initializing
25 * implementation and init data.
26 */
27 contract InitializableUpgradeabilityProxy is BaseUpgradeabilityProxy {
28     /**
29     * @dev Contract initializer.
30     * @param _logic Address of the initial implementation.
31     * @param _data Data to send as msg.data to the implementation to initialize the proxied
32     * It should include the signature and the parameters of the function to be called, as de
33     * https://solidity.readthedocs.io/en/v0.4.24/abi-spec.html#function-selector-and-argumen
34     * This parameter is optional, if no data is given the initialization call to proxied co
35     */
36     function initialize(address _logic, bytes memory _data) public payable {
37         require(_implementation() == address(0));
38         assert(IMPLEMENTATION_SLOT == bytes32(uint256(keccak256('eip1967.proxy.implementation'
39         _setImplementation(_logic);
40         if(_data.length > 0) {
41             (bool success,) = _logic.delegatecall(_data);
42             require(success);
43         }
44     }
45 }
46
47 /**
48 * @title InitializableAdminUpgradeabilityProxy
49 * @dev Extends from BaseAdminUpgradeabilityProxy with an initializer for
50 * initializing the implementation, admin, and init data.
51 */
52 contract InitializableAdminUpgradeabilityProxy is BaseAdminUpgradeabilityProxy, Initializal
53     /**
54     * Contract initializer.
55     * @param _logic address of the initial implementation.
56     * @param _admin Address of the proxy administrator.
57     * @param _data Data to send as msg.data to the implementation to initialize the proxied
58     * It should include the signature and the parameters of the function to be called, as de
59     * https://solidity.readthedocs.io/en/v0.4.24/abi-spec.html#function-selector-and-argumen
60     * This parameter is optional, if no data is given the initialization call to proxied co
61     */
62     function initialize(address _logic, address _admin, bytes memory _data) public payable {
63         require(_implementation() == address(0));
64         InitializableUpgradeabilityProxy.initialize(_logic, _data);
65         assert(ADMIN_SLOT == bytes32(uint256(keccak256('eip1967.proxy.admin')) - 1));
66         _setAdmin(_admin);
67     }
68 }
69
70 /**
71 * @notice MassetProxy delegates calls to a Masset implementation
72 * @dev Extending on OpenZeppelin's InitializableAdminUpgradabilityProxy
73 * means that the proxy is upgradable through a ProxyAdmin. MassetProxy upgrades
74 * are implemented by a DelayedProxyAdmin, which enforces a 1 week opt-out period.
75 * All upgrades are governed through the current mStable governance.
76 */

```

```

77 contract MasetProxy is InitializableAdminUpgradeabilityProxy {
78 }
79
80 /**
81  * @notice BasketManagerProxy delegates calls to a BasketManager implementation
82  * @dev      Extending on OpenZeppelin's InitializableAdminUpgradabilityProxy
83  * means that the proxy is upgradable through a ProxyAdmin. BasketManagerProxy upgrades
84  * are implemented by a DelayedProxyAdmin, which enforces a 1 week opt-out period.
85  * All upgrades are governed through the current mStable governance.
86  */
87 contract BasketManagerProxy is InitializableAdminUpgradeabilityProxy {

```

## SLOC Appendix

### Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Solidity	4	2640	311	1299	1030	97

Comments to Code 1299/ 1030 = 126%

### Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
JavaScript	8	2397	251	199	1947	53

Tests to Code 1947/ 1030 = 10%