Pylon Protocol Process Quality Review

Score: 74%

Overview

This is a Pylon Protocol Process Quality Review completed on August 5th 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 final score of the review is 74%, a **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

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

Chain

This section indicates the blockchain used by this protocol.

✓ **Chain:** Terra

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.pylon.money/contract-specification/core/pool, 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 *contract.rs*, 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/pylon-protocol/core.

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: 70%

With 69 commits and 4 branches, this is an acceptable 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

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:** Yes

Location: https://medium.com/@limowooj.

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://docs.pylon.money/.

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

✓ **Answer:** Yes

All basic software functions of the core Pylon Protocol contracts can be found in the "Contract Specification" section of their documentation.

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

> ✓ **Answer:** 100%

Pylon Protocol's software function documentation covers all functions of all their deployed contracts which include the Core (Pool, Exchange Rate), and the Launchpad (Lockup, Swap).

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:** 0%

Code examples are in the Appendix. As per the SLOC, there is 1% 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:** 100%

All software functions detailed in the Pylon Protocol documentation has clear and explicit traceability as to its implementation in code. A great example of this is here.

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 248% 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:** 75%

No evidence of any Pylon Protocol third-party code coverage was found. However, they clearly have a very robust set of tests.

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 improved by adding tests that achieve 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:** Yes

Scrips/Instructions location: https://github.com/pylon-protocol/core/blob/master/README.md.

14) Report of the results (%)

⚠ **Answer:** 0%

No Pylon Protocol test report was found in their documentation or in their GitHub repositories.

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 Pylon Protocol Formal Verification test was found in any of their documentation or in subsequent web searches.

16) Stress Testing environment (%)

> ✓ **Answer:** 100%

There is evidence of Pylon Protocol's testnet usage at https://github.com/pylon-protocol/assets/blob/master/contracts.json.

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:** 60%

SVCSoft published a Pylon Protocol audit report on June 26th 2021. The audit quality appears high, however we have concern about several high severity findings that are not fixed or explained by the team. For this reason 30% is deducted or 60%.

This audit was performed before the launch of the MINE token staking.

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, (where question 1 is 0%)

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%

Pylon Protocol's Bug Bounty program is active and rewards participating users with up to 150k 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%

Pylon Protocol's access controls are clearly labelled as "Pylon Governance" in 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:** 60%

a) Contracts are labelled as upgradeable through the process of voting.

b) Governance contract is OnlyOwner of itself.

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 of the Pylon Protocol access control information is detailed in user-friendly language.

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 a Pylon Protocol Pause Control or similar function was found in their documentation or in their GitHub repositories.

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

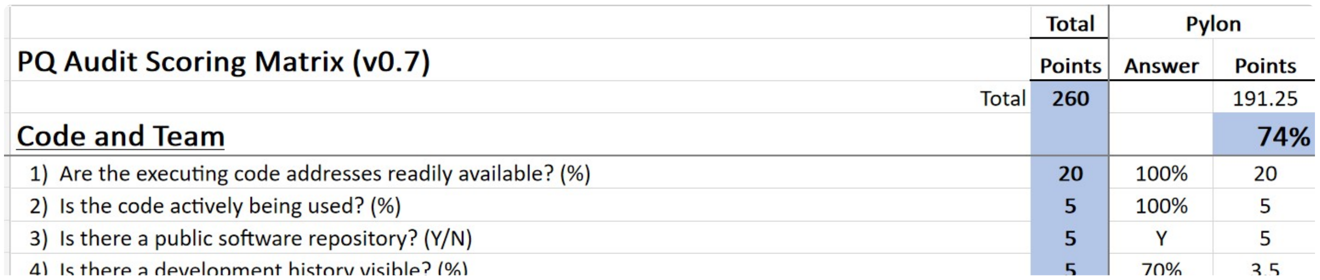
I started with Ethereum just before the DAO and that was a wonderful education. It showed the importance of code quality. The second Parity hack also showed the importance of good process. Here my aviation background offers some value. Aerospace knows how to make reliable code using quality processes.

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.

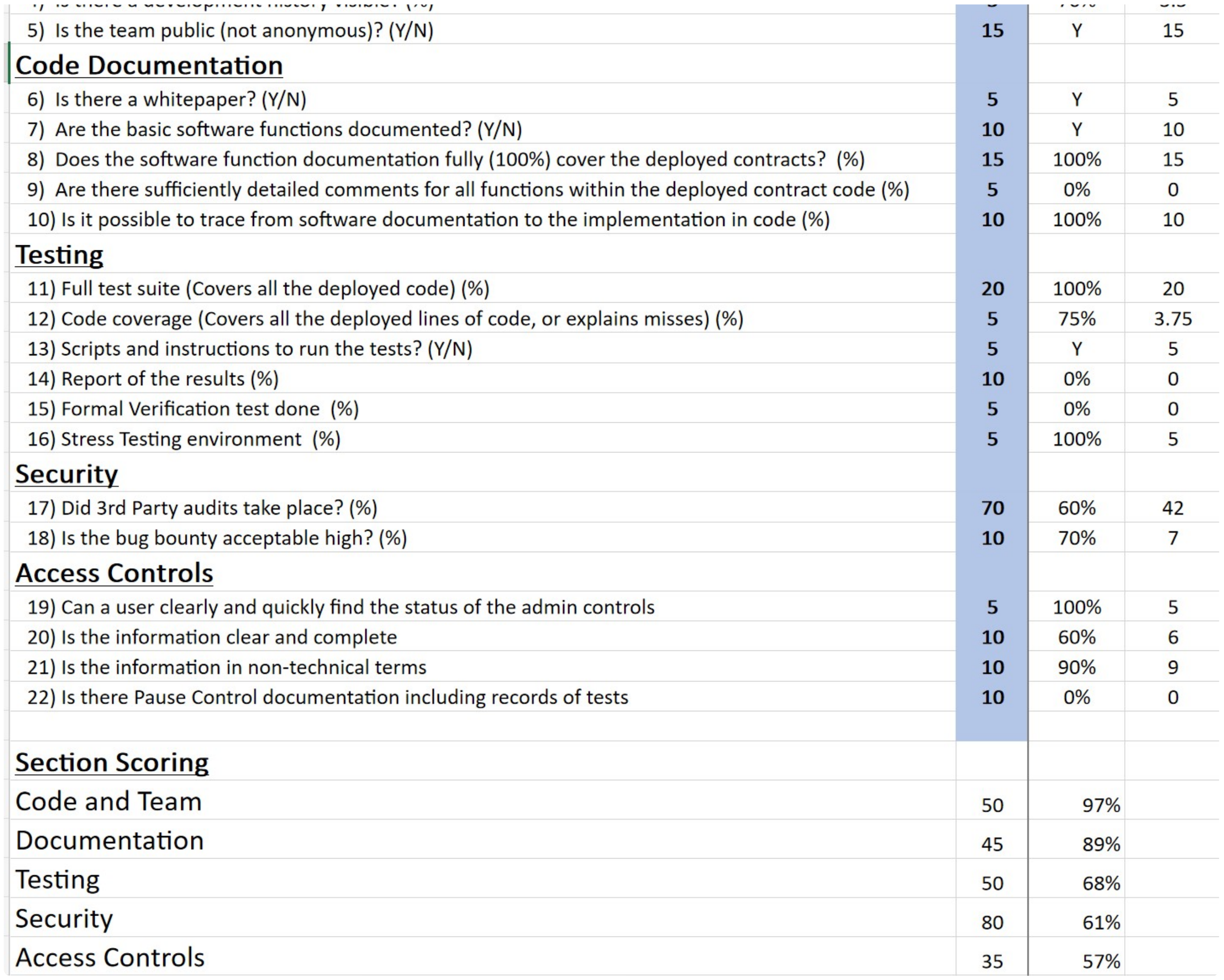
Process Quality Reviews are an extension of the SecurEth guidelines that will further increase the quality processes in Solidity and Vyper development.

DeFiSafety is my full time gig and we are working on funding vehicles for a permanent staff.

Scoring Appendix

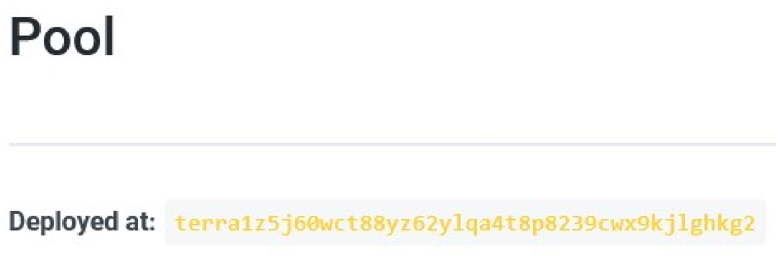


	Total	Pylon	
PQ Audit Scoring Matrix (v0.7)	Points	Answer	Points
Total	260		191.25
Code and Team			74%
1) Are the executing code addresses readily available? (%)	20	100%	20
2) Is the code actively being used? (%)	5	100%	5
3) Is there a public software repository? (Y/N)	5	Y	5
4) Is there a development history visible? (%)	5	70%	3.5



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**	0%	0
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**	75%	3.75
13) Scripts and instructions to run the tests? (Y/N)	**5**	Y	5
14) Report of the results (%)	**10**	0%	0
15) Formal Verification test done (%)	**5**	0%	0
16) Stress Testing environment (%)	**5**	100%	5
Security			
17) Did 3rd Party audits take place? (%)	**70**	60%	42
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**	60%	6
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	97%	
Documentation	45	89%	
Testing	50	68%	
Security	80	61%	
Access Controls	35	57%	

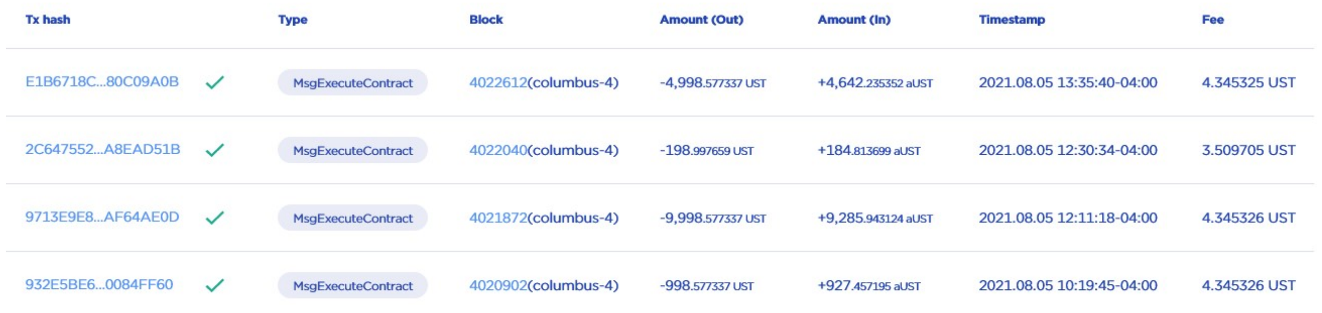
Executing Code Appendix



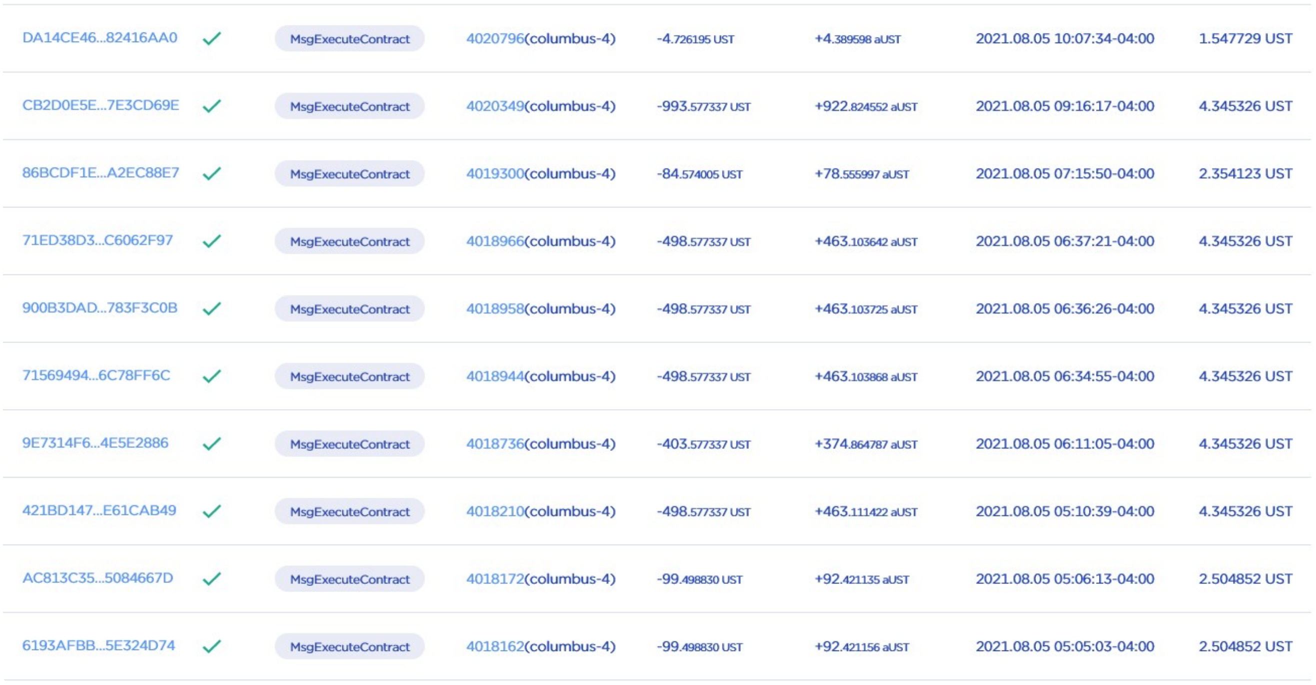
Pool

Deployed at: `terra1z5j60wct88yz62ylqa4t8p8239cwx9kjlghkg2`

Code Used Appendix



Tx hash		Type	Block	Amount (Out)	Amount (In)	Timestamp	Fee
E1B6718C...80C09A0B	✓	MsgExecuteContract	4022612(columbus-4)	-4,998.577337 UST	+4,642.235352 aUST	2021.08.05 13:35:40-04:00	4.345325 UST
2C647552...A8EAD51B	✓	MsgExecuteContract	4022040(columbus-4)	-198.997659 UST	+184.813699 aUST	2021.08.05 12:30:34-04:00	3.509705 UST
9713E9E8...AF64AE0D	✓	MsgExecuteContract	4021872(columbus-4)	-9,998.577337 UST	+9,285.943124 aUST	2021.08.05 12:11:18-04:00	4.345326 UST
932E5BE6...0084FF60	✓	MsgExecuteContract	4020902(columbus-4)	-998.577337 UST	+927.457195 aUST	2021.08.05 10:19:45-04:00	4.345326 UST



DA14CE46...82416AA0	✓	MsgExecuteContract	4020796(columbus-4)	-4.726195 UST	+4.389598 aUST	2021.08.05 10:07:34-04:00	1.547729 UST
CB2D0E5E...7E3CD69E	✓	MsgExecuteContract	4020349(columbus-4)	-993.577337 UST	+922.824552 aUST	2021.08.05 09:16:17-04:00	4.345326 UST
86BCDF1E...A2EC88E7	✓	MsgExecuteContract	4019300(columbus-4)	-84.574005 UST	+78.555997 aUST	2021.08.05 07:15:50-04:00	2.354123 UST
71ED38D3...C6062F97	✓	MsgExecuteContract	4018966(columbus-4)	-498.577337 UST	+463.103642 aUST	2021.08.05 06:37:21-04:00	4.345326 UST
900B3DAD...783F3C0B	✓	MsgExecuteContract	4018958(columbus-4)	-498.577337 UST	+463.103725 aUST	2021.08.05 06:36:26-04:00	4.345326 UST
71569494...6C78FF6C	✓	MsgExecuteContract	4018944(columbus-4)	-498.577337 UST	+463.103868 aUST	2021.08.05 06:34:55-04:00	4.345326 UST
9E7314F6...4E5E2886	✓	MsgExecuteContract	4018736(columbus-4)	-403.577337 UST	+374.864787 aUST	2021.08.05 06:11:05-04:00	4.345326 UST
421BD147...E61CAB49	✓	MsgExecuteContract	4018210(columbus-4)	-498.577337 UST	+463.111422 aUST	2021.08.05 05:10:39-04:00	4.345326 UST
AC813C35...5084667D	✓	MsgExecuteContract	4018172(columbus-4)	-99.498830 UST	+92.421135 aUST	2021.08.05 05:06:13-04:00	2.504852 UST
6193AFBB...5E324D74	✓	MsgExecuteContract	4018162(columbus-4)	-99.498830 UST	+92.421156 aUST	2021.08.05 05:05:03-04:00	2.504852 UST

Example Code Appendix

```rust
ub fn init<S: Storage, A: Api, Q: Querier>(
    deps: &mut Extern<S, A, Q>,
    env: Env,
    msg: InitMsg,
) -> StdResult<InitResponse> {
    let sender = env.message.sender;
    let raw_sender = deps.api.canonical_address(&sender)?;

    let mut config = config::Config {
        this: deps.api.canonical_address(&env.contract.address)?,
        owner: raw_sender,
        beneficiary: deps.api.canonical_address(&msg.beneficiary)?,
        fee_collector: deps.api.canonical_address(&msg.fee_collector)?,
        exchange_rate_feeder: deps.api.canonical_address(&msg.exchange_rate_feeder)?,
        moneymarket: deps.api.canonical_address(&msg.moneymarket)?,
        stable_denom: String::default(),
        atoken: CanonicalAddr::default(),
        dp_token: CanonicalAddr::default(),
    };

    let market_config = querier::anchor::config(deps, &config.moneymarket)?;

    config.stable_denom = market_config.stable_denom.clone();
    config.atoken = deps.api.canonical_address(&market_config.aterra_contract)?;

    config::store(&mut deps.storage, &config)?;

    Ok(InitResponse {
        messages: vec![CosmosMsg::Wasm(WasmMsg::Instantiate {

            code_id: msg.dp_code_id,
            send: vec![],
            label: None,
            msg: to_binary(&Cw20InitMsg {
                name: format!("Deposit Token - {}", msg.pool_name),
                symbol: "PylonDP".to_string(),
                decimals: 6u8,
                initial_balances: vec![],
                mint: Some(MinterResponse {
                    minter: env.contract.address.clone(),
                    cap: None,
                }),
                init_hook: Some(Cw20InitHook {
                    contract_addr: env.contract.address,
                    msg: to_binary(&HandleMsg::RegisterDPToken {})?,
                }),
            })?,
        })],
        log: vec![],
    })
}

pub fn handle<S: Storage, A: Api, Q: Querier>(
    deps: &mut Extern<S, A, Q>,
    env: Env,
    msg: HandleMsg,
) -> StdResult<HandleResponse> {
    match msg {
        HandleMsg::Receive(msg) => CoreHandler::receive(deps, env, msg),
        HandleMsg::Deposit {} => CoreHandler::deposit(deps, env),
        HandleMsg::ClaimReward {} => CoreHandler::claim_reward(deps, env),
        HandleMsg::RegisterDPToken {} => CoreHandler::register_dp_token(deps, env),
    }
}

pub fn query<S: Storage, A: Api, Q: Querier>(
    deps: &Extern<S, A, Q>,
    msg: QueryMsg,
) -> StdResult<Binary> {
    match msg {
        QueryMsg::DepositAmountOf { owner } => QueryHandler::deposit_amount(deps, owner),
        QueryMsg::TotalDepositAmount {} => QueryHandler::total_deposit_amount(deps), // dp
        QueryMsg::Config {} => QueryHandler::config(deps),                           // co
        QueryMsg::ClaimableReward {} => QueryHandler::claimable_reward(deps), // config.st
    }
}

pub fn migrate<S: Storage, A: Api, Q: Querier>(
    _deps: &mut Extern<S, A, Q>,
    _env: Env,
    _msg: MigrateMsg,
) -> MigrateResult {
    Ok(MigrateResponse::default())
```

```
}
```

SLOC Appendix

Solidity Contracts

Language	Files	Lines	Blanks	Comments	Code	Complex
Rust	31	2050	283	23	1744	56

Comments to Code 23/1744 = 1%

Javascript Tests

Language	Files	Lines	Blanks	Comments	Code	Complex
Rust	16	5054	609	117	4328	31

Tests to Code 4328/1744 = 248%