SOLIDIFIED

Audit Report for Anchor - July 09, 2021

Summary

Audit Report prepared by Solidified covering the Ethereum Anchor smart contracts.

Process and Delivery

Three (3) independent Solidified experts performed an unbiased and isolated audit of the code. The debrief on 16 June 2021.

Fixes were verified on 9 July 2021

Audited Files

The source code has been supplied in the form of a GitHub repository:

https://github.com/Anchor-Protocol/eth-anchor-contracts/tree/develop

Commit number: `7953b4f79edc3902f9a7fcba1b3a05aff0b0c335`

The scope of the audit was limited to the following files:

```
contracts
├── assets
│   ├── WormholeAsset.sol
│   ├── WormholeAssetFactory.sol
│   └── WrappedAsset.sol
├── core
│   ├── Controller.sol
│   ├── Router.sol
│   ├── RouterV2.sol
│   └── upgrade
│       └── RouterUpgraderV1.sol
├── extensions
│   ├── ConversionPool.sol
│   ├── ConversionPoolV2.sol
│   ├── ExchangeRateFeeder.sol
│   └── upgrade
│       └── ConversionPoolUpgraderV1.sol
├── interfaces
│   ├── IAnchorAccount.sol
│   ├── IShuttleAsset.sol
│   └── IWormhole.sol
├── libraries
│   └── UniswapV2Library.sol
├── operations
│   ├── Operation.sol
│   ├── OperationACL.sol
│   ├── OperationFactory.sol
│   └── OperationStore.sol
├── swapper
│   ├── CurveSwapper.sol
│   ├── ISwapper.sol
│   └── UniswapSwapper.sol
├── test
│   ├── MockAsset.sol
│   ├── QueueTester.sol
│   ├── TestAsset.sol
│   └── TestProxy.sol
├── upgradeability
│   ├── AdminUpgradeabilityProxy.sol
│   ├── Proxy.sol
│   ├── SimpleProxy.sol
│   └── UpgradeabilityProxy.sol
└── utils
    ├── ERC20Controlled.sol
    ├── Operator.sol
    ├── Ownable.sol
    └── Queue.sol
```

Intended Behavior

The smart contracts implement Ethereum wrappaers that allow users to interact with the Terra-based Anchor protocol.

Code Complexity and Test Coverage

Smart contract audits are an important step to improve the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of a smart contract system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**.

Note, that high complexity or lower test coverage does equate to a higher risk. Certain bugs are more easily detected in unit testing than a security audit and vice versa. It is, therefore, more likely that undetected issues remain if the test coverage is low or non-existent.

Criteria	Status	Comment
Code complexity	Medium	-
Code readability and clarity	High	-
Level of Documentation	High	-
Test Coverage	Medium	-

Coverage Report:

```
-------------------------------|----------|----------|----------|----------|----------------|
File                           |  % Stmts | % Branch |  % Funcs |  % Lines |Uncovered Lines |
-------------------------------|----------|----------|----------|----------|----------------|
 assets/                       |        0 |        0 |        0 |        0 |                |
  WormholeAsset.sol            |        0 |        0 |        0 |        0 |... 51,52,55,56 |
  WormholeAssetFactory.sol     |        0 |      100 |        0 |        0 |... 25,29,30,31 |
  WrappedAsset.sol             |      100 |      100 |      100 |      100 |                |
 core/                         |    57.27 |       50 |    31.37 |    57.27 |                |
  Controller.sol               |       60 |       25 |       50 |       60 |... 105,107,108 |
  Router.sol                   |    68.75 |    66.67 |    40.91 |    68.75 |... 303,345,356 |
  RouterV2.sol                 |    42.86 |    41.67 |    14.29 |    42.86 |... 344,360,372 |
 core/upgrade/                 |      100 |       50 |      100 |      100 |                |
  RouterUpgraderV1.sol         |      100 |       50 |      100 |      100 |                |
 extensions/                   |     42.5 |    21.43 |    37.14 |    42.62 |                |
  ConversionPool.sol           |     62.5 |       20 |    53.33 |    63.27 |... 163,164,218 |
  ConversionPoolV2.sol         |        0 |        0 |        0 |        0 |... 186,187,196 |
  ExchangeRateFeeder.sol       |    80.77 |    66.67 |      100 |    80.77 | 88,90,91,92,93 |
 extensions/upgrade/           |      100 |       50 |      100 |      100 |                |
  ConversionPoolUpgraderV1.sol |      100 |       50 |      100 |      100 |                |
 interfaces/                   |      100 |      100 |      100 |      100 |                |
  IAnchorAccount.sol           |      100 |      100 |      100 |      100 |                |
  IShuttleAsset.sol            |      100 |      100 |      100 |      100 |                |
  IWormhole.sol                |      100 |      100 |      100 |      100 |                |
 libraries/                    |    58.82 |       25 |     62.5 |    58.82 |                |
  UniswapV2Library.sol         |    58.82 |       25 |     62.5 |    58.82 |... 130,131,133 |
 operations/                   |    86.05 |    66.67 |    84.44 |    86.83 |                |
  Operation.sol                |     87.3 |    71.88 |       75 |     87.1 |... 283,326,331 |
  OperationACL.sol             |    90.91 |       50 |     87.5 |    93.33 |             50 |
  OperationFactory.sol         |    94.44 |      100 |    83.33 |    94.44 |             77 |
  OperationStore.sol           |     82.5 |    65.79 |    93.33 |    83.33 |... 260,277,292 |
 swapper/                      |       50 |    16.67 |    42.86 |       50 |                |
  CurveSwapper.sol             |        0 |        0 |        0 |        0 |... 102,110,111 |
  ISwapper.sol                 |      100 |      100 |      100 |      100 |                |
  UniswapSwapper.sol           |    94.44 |       50 |       75 |    94.44 |             30 |
 test/                         |    76.92 |      100 |       75 |    76.92 |                |
  MockAsset.sol                |        0 |      100 |        0 |        0 |       14,18,22 |
  QueueTester.sol              |      100 |      100 |      100 |      100 |                |
  TestAsset.sol                |      100 |      100 |      100 |      100 |                |
  TestProxy.sol                |      100 |      100 |      100 |      100 |                |
 upgradeability/               |    81.48 |       40 |    89.47 |    84.85 |                |
  AdminUpgradeabilityProxy.sol |    70.59 |     37.5 |       80 |       75 |... 131,134,136 |
  Proxy.sol                    |      100 |      100 |      100 |      100 |                |
  SimpleProxy.sol              |      100 |      100 |      100 |      100 |                |
  UpgradeabilityProxy.sol      |      100 |       50 |      100 |      100 |                |
 utils/                        |    66.67 |    33.33 |    62.86 |    65.38 |                |
  ERC20Controlled.sol          |       80 |      100 |       75 |       80 |             31 |
  Operator.sol                 |    91.67 |    33.33 |       90 |    86.67 |          32,34 |
  Ownable.sol                  |    36.36 |        0 |    33.33 |    33.33 |... 64,72,76,77 |
  Queue.sol                    |       65 |      100 |    53.33 |       65 |... 63,67,71,85 |
-------------------------------|----------|----------|----------|----------|----------------|
All files                      |    66.67 |    46.04 |    60.25 |    66.87 |                |
-------------------------------|----------|----------|----------|----------|----------------|
```

Issues Found

Solidified found that the EthAnchor contracts contain no critical issues, no major issue, 5 minor issues, in addition to 3 informational notes.

We recommend all issues are amended, while the notes are up to the team's discretion, as they refer to best practices.

Issue #	Description	Severity	Status
1	Inheritance Mismatch due to duplicate libraries	Minor	Resolved
2	CurveSwapper.sol: Token swaps does not validate the pool existence	Minor	Resolved
3	Index 0 of an enumerable set is not in any order	Minor	Acknowledged
4	OperationStore.sol: allocate() does not check if the operation already exists in other queues	Minor	Acknowledged
5	Missing input validations	Minor	Resolved
6	WrappedAsset.sol: Interface not used	Note	Resolved
7	Pragma allows for a wide range of compiler versions	Note	Resolved
8	Queue.sol: Consider deleting the items from Queue Store	Note	Acknowledged

Critical Issues

No critical issues have been found.

Major Issues

No major issues have been found.

Minor Issues

1. Inheritance Mismatch due to duplicate libraries

Several OpenZeppelin dependencies are included from the module dependencies whilst a separate version from the project’s codebase is also imported. This applies to the following contracts:

`Ownable.sol`
`Proxy.sol`

Recommendation
Clean up file imports.

Update
Resolved

2. CurveSwapper.sol: Token swaps does not validate the pool existence

The function `swapToken()` does not validate if the Curve pool exists and this will skip the token swap and transfer whatever target token the user has specified. Since the input token is not validated the user can use any ERC20 tokens, even custom tokens, to swap for actual tokens already present in the contract.

Recommendation

Consider validating the existence of the pool before transferring any tokens present in the contract.

Update
Resolved

3. Index 0 of an enumerable set is not in any order

The functions throughout the contract fetch an item at index `0` from an enumerable set. The enumerable set library from OpenZeppelin does not guarantee the order of the items and in most cases will return the element which is inserted last. This can even make the items at the bottom of the stack to be fetched much later than expected.

Some functions that follow this practice are `OperationFactory.fetchTerraAddress()` and `OperationStore.init()`

Recommendation
Consider using a proper stack or queue that aligns with the use case if the mentioned issue is not intended.

Update
Acknowledged: “*We uses EnumerableSet to specific parts that does not need to care about order.*”

4. OperationStore.sol: `allocate()` does not check if the operation already exists in other queues

The method `allocation()` does not check if the input operation address is already present in any other contract. This also allows changing the status of such addresses without much validation.

Recommendation
Consider validating if the operation is already present as part of `optRunning` or `optStopped`.

Update
Acknowledged: “*As long as Controller.sol pushes only deployed Operations in same tx, it's not gonna happen.*”

5. Missing input validations

The contracts are missing input validation in many functions, specifically the ones where address is passed as a parameter. The effect can range from simple gas wastage to unintentional ownership renounce.

In particular:

- No guard in `Operator.sol` to prevent setting owner/operator to `address(0)`
- No guard in `OperationACL.sol` to prevent setting owner/router/controller to `address(0)`
- No guards in any of the admin setters in both `Router.sol` and `RouterV2.sol` to prevent `address(0)`
- No guard in `UniswapSwapper.sol` to prevent setting `swapFactory` to `address(0)`
- `ConversionPool.sol` and `ConversionPoolV2.so`l: `address(0)` checks for `setSwapper`, `setOperationRouter`, `setExchangeFeeder`, `setDepositAllowance`, `setRedemptionAllowance`, and `migrate`

Recommendation
Consider adding input validation in the cases identified above.

Update
Resolved

Informational Notes

6. WrappedAsset.sol: Interface not used

The `WrappedAsset.sol` file defines an interface. However, `WormholeAsset.sol` and `WormholeAssetFactory.sol` do not use this interface and declare another interface for this purpose inline.
Moreover, it is common convention to start interface file names with a capital I, in order to identify them more easily.

Recommendation
Clean up the interface descriptions and consider changing the file name to `IWrappedAsset.sol`.

Update
Resolved

7. Pragma allows for a wide range of compiler versions

The `pragma` statement allows for a very large range of compiler versions, including some versions with known bugs. In addition, the language syntax has changed since the earlier versions that are allowed.

Recommendation
Consider limiting the compiler to at least a single major version number.

Update
Resolved

8. `Queue.sol`: Consider deleting the items from Queue Store

The queue implementation does not delete the items from the `Queue.store` map once the item is dequeued.

Recommendation
It is recommended to delete the items from the queue store once it is dequeued.

Update
Acknowledged: “*Acknowledged. We need these items for debug & backfill.*”

Disclaimer

Solidified audit is not a security warranty, investment advice, or an endorsement of TerraformLabs/ Anchor or its products. This audit does not provide a security or correctness guarantee of the audited smart contract. Securing smart contracts is a multistep process, therefore running a bug bounty program as a complement to this audit is strongly recommended.

The individual audit reports are anonymized and combined during a debrief process, in order to provide an unbiased delivery and protect the auditors of Solidified platform from legal and financial liability.

Solidified Technologies Inc.