SOLIDIFIED

Audit Report for Polymath. March 4, 2018.

Summary

Audit Report prepared by Solidified for Polymath covering the Polymath USDTieredSTO module and it's auxiliary files.

Process and Delivery

Three (3) independent Solidified experts performed an unbiased and isolated audit of the below token swap. The debrief took place on March 04, 2019 and the final results are presented here.

Audited Files

- USDTieredSTO.sol
- USDTieredSTOFactory.sol
- USDTieredSTOStorage
- CappedSTO.sol
- CappedSTOFactory.sol

The files were audited on commit `a8b71e575526284c08803e156ab0c3feca198989` and solidity compiler version `0.4.24`

Intended Behavior

The purpose of these contracts is to offer a tiered USD based Security Token Offering
The audit was based on commit.

Issues Found

Critical

No critical issues were found.

Major

No major issues were found.

Minor

1. `Wallet` is defined both on ISTOStorage and USDTieredSTOStorage

In the current implementation, there's no risk of variable shadowing but it could happen as development continues.

Recommendation:
Remove the declaration on `USDTieredSTOStorage`

Amended:
The issue was fixed and is no longer present in commit `a657773768c19a941d291816334943127c8448b2`.

2. Capped STO last transaction must be exact

The transaction which reaches the token cap must be exact otherwise it fails. It's recommended that the contract accepts the transaction and give the exceeding amount as change.

Amended:
The issue was fixed and is no longer present in commit `a657773768c19a941d291816334943127c8448b2`.

Notes

3. Poly can be added as a `usdToken`

In USDTieredSTO the poly address can be added as a valid `usdToken` by administrators. If that happens, the purchases go through, but some of the accounting misbehaves.

Recommendation:
Require that the address is different than the POLY when adding the `usdTokens`

Amended:
The issue was fixed and is no longer present in commit `a657773768c19a941d291816334943127c8448b2`.

4. Granularity is not enforced

If the end of the tier is reached, the token granularity is not enforced, because the buyer takes all remaining tokens on the tier.

Amended:
The issue was fixed and is no longer present in commit `a657773768c19a941d291816334943127c8448b2`.

5. Remove unused functions

It's not clear if CappedSTO is meant to be extensible by other developers. If that's not the case, it's recommended to remove unused functions:

* `updatePurchasingState`
* `postValidatePurchase`

Amended:
The issue was fixed and is no longer present in commit `a657773768c19a941d291816334943127c8448b2`.

6. Use external functions

Marking functions as `external` saves gas when being called with big data arrays. This could be optimized in some functions

* `changeAccreditedLimit`
* `changeNonAccreditedLimit`

But also it's advised to all the functions that should not be called from within the contract, like:

* `getTokensSoldFor`

* `getTokensMintedByTier`
* `getTokensSoldByTier`

Amended:
The issue was fixed and is no longer present in commit `a657773768c19a941d291816334943127c8448b2`.

7. No upper array bound check in `_setFundRaiseType()`

In ISTO.sol, there is statement checking that the number of elements in the argument array does not exceed three. This means that duplicate types may be set.

Consider adding: `require(_fundRaiseTypes.length <= 3)`;

Amended:
The issue was fixed and is no longer present in commit `a657773768c19a941d291816334943127c8448b2`.

8. ISTO contains implementation

It's widely accepted in the community that files that are prefixed with `I` are interfaces that merely define a contracts function, but the ISTO file also contains function implementations. It's recommended that this file is renamed to suit its content.

Amended:
The issue was fixed and is no longer present in commit `a657773768c19a941d291816334943127c8448b2`.

Closing Summary

Although no critical or major issues were found, there are some minor issues that can affect the desired behavior and it's recommended that they're addressed before proceeding to deployment.

Disclaimer

Solidified audit is not a security warranty, investment advice, or an endorsement of the Polymath platform or its products. This audit does not provide a security or correctness guarantee of the audited smart contracts. 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.