xDollar
Smart Contract Audit

coinspect

coinspect

xDollar

Smart Contract Audit

Prepared for xDollar • June 2021

v210624

1. Executive Summary

In **June 2021**, xDollar.fi engaged Coinspect to perform a source code review of the changes made on top of a fork of the Liquity Protocol. The objective of the project was to evaluate the impact of the changes on the security of the smart contracts. The project has already been deployed in the Polygon network, but the scope of the assessment was limited to source code review and it did not include post-deployment verification.

During the assessment, it was found that the **major differences** with Liquity are:

- Two constants have been modified, one for the amount of LUSD locked on trove opening to be used for **gas compensation** in case of liquidation, the other for the **minimum amount of net LUSD debt** a trove must have.
- The **allocations** of XDO (previously LQTY) tokens minted at deploy time have been modified.
- Liquity's restrictions that ensure that LQTY tokens cannot enter circulating supply and cannot be staked to earn system revenue until one year after deployment has been **reduced to half year**.
- The **fallback oracle price feed has been removed** from the `PriceFeed` contract, and **only Chainlink** was left.

The assessment identified **no security issues** introduced by the changes to the original Solidity code. Analyzing the impact of MATIC as collateral on Liquity's economic model and threats specific to the Polygon network were out of scope for this audit.

2. Introduction

xDollar is a cross-chain interest-free borrowing platform based on Liquity protocol. xDollar is a collateralized debt platform that allows users to lock up MATIC and borrow xUSD tokens. The xDollar protocol issues a stablecoin (xUSD), intended to maintain a value of 1 xUSD = 1 USD. The xDollar protocol allows users to redeem xUSD for MATIC: for N xUSD users get N USD worth of MATIC in return.

A given position associated with a Polygon address is called a "trove". When a user makes a deposit of MATIC, a trove is created with the collateral MATIC provided by the user. In normal circumstances a user can add MATIC to their collateral deposit, borrow xUSD, redeem xUSD for MATIC at face value, or withdraw MATIC from their collateral deposit, as long as a minimum collateralization ratio (MCR) of 110% is maintained. Troves with a collateralization ratio below 110% can be subject to liquidation.

The xDollar protocol also involves "stability providers'' that deposit xUSD in a pool that is used for covering the debt of liquidated troves, and as an incentive they receive excess collateral from liquidated troves (this excess should be close to 10%, since troves can be liquidated as soon as their collateralization ratio drops below 110%).

The protocol also issues a XDO token that is used to distribute among its holders a share of the revenue generated from redemption fees and xUSD issuance fees. XDO tokens are issued as incentive to third-parties that provide front-ends for the protocol (for example web front-ends or apps) and stability providers, and also (under a vesting schedule) to team members and partners.

3. Assessment

The audit started on **June 21, 2021** and was performed on the contracts in the git repository at https://github.com/xDollar-Finance/xDollar-contracts as of commit 7635f51 of **Jun 10, 2021**. During the assessment the xDollar contracts were compared with the contracts at the original Liquity repository at https://github.com/liquity/dev as of commit 9c79c9e of **May 22, 2021**.

The following is a summary of the changes in each modified contract.

`LiquityBase`

Two constants were modified in the contract `LiquityBase`:

- `LUSD_GAS_COMPENSATION` (the amount of LUSD locked on trove opening to be used for gas compensation in case of liquidation) was reduced from 200 LUSD to 50 LUSD.
- Also, `MIN_NET_DEBT` (the minimum amount of net LUSD debt a trove must have) was reduced from 1800 LUSD to 450 LUSD.

`CommunityIssuance`

The contract `CommunityIssuance` was modified reducing `LQTYSupplyCap` from 32 millions to 10 millions. This is the amount of LQTY tokens that are minted to the `CommunityIssuance` contract when `LQTYToken` is deployed.

`LookupContract`

The contract `LockupContract` was modified to decrease the minimum unlock time from “one year after system deployment” to “half a year after system deployment”. This restriction works together with a similar restriction in the `LQTYToken` contract, but `LQTYToken` was not changed accordingly. The following text was extracted from the documentation in the original `LockupContract.sol`:

> - `At construction, the contract checks that unlockTime is at least one year later than the Liquity system's deployment time;`
> - `Within the first year from deployment, the deployer of the LQTYToken (Liquity AG's address) may transfer LQTY only to valid LockupContracts, and`

```
        no other addresses (this is enforced in LQTYToken.sol's transfer()
        function).

    The above two restrictions ensure that until one year after system deployment, LQTY
    tokens originating from Liquity AG cannot enter circulating supply and cannot be
    staked to earn system revenue.
```

Although `LQTYToken` was not modified to reduce the transfer restriction to half a year to be consistent with the changes in `LockupContract`, the result is the same: in short, the change in `LockupContract` means that **XDO tokens originating from xDollar.fi could enter the circulating supply and be staked after half a year of deployment**, not one year as in the case of Liquity.

`LQTYToken`

The `LQTYToken` contract was modified to change the token's name and symbol from **"`LQTY`"** to **"`xDollar`"** and **"`XDO`"**. Token allocations during deployment were also modified. The allocation for bounties and hackathons was removed (2 millions), the allocation for community issuance (depositors and frontends) was decreased from 32 millions to 10 millions, the allocation for LP rewards was reduced from 1.33 millions to 0.05 millions, and the following allocations were added:

- 1 million for `initialSetupAddress` (the deployer)
- 15 million to `ecosystemVestingAddress`
- 7.5 million to `teamVestingAddress`
- 4 million to `partnerVestingAddress`
- 8.95 million to `treasuryAddress`

The addresses are set at deployment time, and `ecosystemVestingAddress`, `teamVestingAddress` and `partnerVestingAddress` are intended to be `TokenVesting` contracts.

The original Liquity `LQTYToken` contract allocated the remaining of 100 millions to the multisig address, which added up to 64.67 millions. Instead, xDollar's version was modified to allocate 17.5 millions for the multisig address, and the remaining 36 millions are left to be allocated in other side chains.

`LUSDToken`

The `LUSDToken` contract's name and symbol were changed from **"`LUSD Stablecoin`"** and **"`LUSD`"** to **"`xDollar Stablecoin`"** and **"`xUSD`"**.

`PriceFeed`

Liquity uses the Chainlink oracle as primary price feed, and the Tellor oracle as fallback in case Chainlink fails. However, the xDollar version for Polygon network eliminates the use of the Tellor oracle and **only relies on Chainlink**. The function `fetchPrice` was greatly simplified, and now if the Chainlink oracle fails it returns "the last good price seen" (the same behaviour of Liquity when both Chainlink and Tellor fail).

`TokenVesting`

The `TokenVesting` contract is a new contract introduced by xDollar.fi. It is a token holder contract that can release its token balance gradually like a typical vesting scheme, with a cliff and vesting period. The contract has an owner, and if the contract was created as 'revocable' the owner has the possibility to revoke the vesting by calling function `revoke(IERC20 token)` to transfer back to the owner the amount of tokens not yet vested.

Three instances of this contract are used to hold XDO token allocations (`ecosystemVestingAddress`, `teamVestingAddress`, `partnerVestingAddress` in the `LQTYToken` contract).

Some storage variables are set in the constructor and are never changed. **Coinspect recommends declaring the following variables *immutable*: `_beneficiary`, `_cliff`, `_start`, `_duration`, `_revocable`.**

4. Disclaimer

The information presented in this document is provided "as is" and without warranty. The present security audit does not cover any off-chain systems or frontends that communicate with the contracts, nor the general operational security of the organization that developed the code.