Liquity ChickenBond

Smart Contract Audit

ChickenBond - 2nd Audit

Smart Contract Audit

V220803

Prepared for Liquity • July 2022

1. Executive Summary

2. Assessment and Scope

5. Disclaimer

1. Executive Summary

In **July 2022, Liquity** engaged Coinspect to perform a second source code review of **ChickenBond**, their *autonomous self-bootstrapping treasury system*. The objective of the project was to evaluate the security of the changes performed to the project's smart contracts since Coinspect's first audit.

The modifications reviewed include fixes to the issues reported in the initial audit and new features added to the platform.

Coinspect verified that all the issues reported in the previous audit were correctly addressed and that new features did not introduce any weaknesses in the system.

As a result of the changes reviewed, the overall security of the system was improved.

No issues were identified during this second assessment.

High Risk	Medium Risk	Low Risk
0	**0**	**0**
Fixed 0	Fixed 0	Fixed 0

2. Assessment and Scope

The audit started on **Jul 25 2022** and was conducted on the **main** branch of the git repository at https://github.com/liquity/ChickenBond as of commit **`339b687a63cabd8e16c7e5f2ffd8a761689b4e42`** of **Aug 3 2022**.

The audited files have the following sha256sum hash:

```
14cbc7565f25412d9eb93c464e00dd602cd1788fee08581c118d24aefb6129f7  ChickenBondManager.sol
ca34517b329ef6d5e69cf285259b5c09e5f3d5673d0af99da2d196a478c46469  BondNFT.sol
6ade1a46b823ce6694814690bed72bb34d17ea375baea3b807799b539d58d7b6  utils/ChickenMath.sol
21d2c25d5296d74543316bdc6d58112c4e63ed29c09c8271430259bf75ef9c62  utils/BaseMath.sol
4ff150419fba46dc42d11eaab75f1149b7ab8d429dd87a25b79cf998feae163d  Proxy/ChickenBondOperationsScript.sol
ba95c52d6c24e00242e96eb0b9da242bd302039337175679222084dc7fb842c0  Interfaces/IBondNFT.sol
4274e58b63bef501a2d87fb619966bc0abcb9d49754e206107fb4a1d6470d1c7  Interfaces/IChickenBondManager.sol
b85a5aa5b27a94e13476f2a92c28a0dc55267f9be816c3735949b8b431953608  Interfaces/IYearnVault.sol
a2979fb37ed51708293e07ec00d274abe17f5fd94b7eec62e9beadc4f9ec2331  Interfaces/IBLUSDToken.sol
6c0218058b30032d1aec1e32491f71ecdb5d431eeed6f5a636b12e72ff77c9bd  Interfaces/StrategyAPI.sol
136d11cc6a428c6d9fb520b61dfa8fc6c0272d1b68079ae0911635a9e2848b95  Interfaces/IBondNFTArtwork.sol
8410326508cc0296bafd4482ef56a54ddde701d0e5903f205cb799023122e0bc  Interfaces/ILUSDToken.sol
c8813e186b5d0a7e9f09b66af976ed6992f7887481cd6e40f3b3d1978bdd1d50  Interfaces/ICurveLiquidityGaugeV4.sol
05796a085522be0a8214d845b4e9953cc25a1064c2d3eddc9d960f3a2947b9d4  Interfaces/ICurvePool.sol
1caf5ec7cf34d637a8eaabc6cfe483772a6856459191a561bd2e68676acb8988  Interfaces/IBAMM.sol
d70b12499cce02cb162203879531274e1d1509c32cc2c7aed7f83569891f46dc  Interfaces/IYearnRegistry.sol
fbe31b8d146705f41e5943fa38c49863da5ecfae66f1949adebcef167b195958  BLUSDToken.sol
```

The following documentation was consulted during the assessment:

1. https://github.com/liquity/ChickenBond/blob/main/papers/ChickenBonds%20Whitepaper.pdf
2. https://github.com/liquity/ChickenBond/blob/main/papers/LUSDChickenBonds_Shifting_Profitability.pdf
3. LUSDChickenBonds Shifting Profitability (fee=0.04%, adminFee=50%)

This section briefly outlines the changes reviewed as well as the fixes for previously reported issues. Readers interested in the protocol's basic functioning are encouraged to refer to the previous report, the Whitepaper, and the extensive documentation.

The new features introduced since the previous audit include but are not limited to:

1. Chicken Bonds are now NFTs (`ERC721Enumerable` contracts). The bonds are not burned, but their status is changed after chicken-ins and outs.
2. Yield is generated by depositing funds in the B.Protocol (https://www.bprotocol.org/) B.AMM SP vault instead of Yearn's vault. B.Protocol's code was not in scope for this audit.
3. A shift window period was introduced to limit when the shift operations can be performed in order to prevent attacks.

4. Protocol parameters are provided to the contract constructor instead of being constants.
5. Several improvements to the funds shifting mechanism. The parameters used to guarantee shifts do not force the protocol into losing funds were obtained by simulating the operations with different parameters.
6. New bootstrap locking period for redeems, shifts, and chicken ins.
7. A minimum bond amount was introduced.
8. Internal `ChickenBondManager` balances tracking instead of absolute.
9. The `shiftLUSDFromSPToCurve` function was further restricted so funds in Curve are always less than funds in the permanent bucket in order to guarantee the desired protocol's yield amplification property.
10. Events are emitted for each Bond operation.
11. Several minor refactors.
12. New tests were introduced and the existing ones were improved and/or adapted to the new features.

Regarding the issues reported in Coinspect's first audit of the project:

1. **LCB-1** is addressed by adding a `_minLUSD` parameter to the `chickenOut` function. Similarly, the `_minLUSDFromBAMMSPVault` parameter was added to the `redeem` function in order to protect the users.
2. **LCB-2** is addressed by commenting out the unnecessary import.
3. **LCB-3** is addressed by removing the `Ownable` dependency.
4. **LCB-4** is addressed by adding the missing `require` call.
5. **LCB-5** is addressed by guaranteeing shifts are always profitable. New deposit and deposit exchange rate threshold parameters were added, which are calculated based on simulations detailed in the documentation. Curve's `get_virtual_price` function is used now instead of spot prices.
6. **LCB-6** and LCB-7 are addressed by new improved tests.
7. **LCB-8** is addressed by reverting when the funds are not available.
8. **LCB-9** is addressed by preventing the funds in the pending bucket from being moved.

5. 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.