Dai-xDai Bridge Contracts Update Diff Review

By: ChainSafe Systems

August 2023

Dai-xDai Bridge Contracts Update Diff Review

Auditors: Anderson Lee, Oleksii Matiiasevych

WARRANTY

Introduction

Gnosis Chain requested ChainSafe Systems to perform a review of the contracts update diff used for Dai-xDai bridging from Ethereum Mainnet to Gnosis which includes the depositing of Dai on Ethereum into a DSR and periodic bridging of the interest to Gnosis. The contracts in scope can be identified as the following git diff:

```
908a48107919d4ab127f9af07d44d47eac91547e original
9eb8f1d00741271b44b3c83f042fb9f6882705f1 update
```

After the initial review, Gnosis Chain team applied a number of updates which can be identified by the following git commit hash:

```
b778a4a3823c2ae8111270af280a2e865762eb71
```

Additional verification was performed after that.

Disclaimer

The review makes no statements or warranties about the utility of the code, safety of the code, suitability of the business model, regulatory regime for the business model, or any other statements about the fitness of the contracts for any specific purpose, or their bug free status.

Executive Summary

There are no known compiler bugs for the specified compiler version (0.4.24), that might affect the contracts' logic.

There were no critical or major issues found. 2 minor and 2 optimizational issues were identified in the contracts that were fixed and are not present in the final version.

Critical Bugs and Vulnerabilities

No critical issues were identified.

Line by Line Review. Fixed Issues

1. ERC20Bridge, line 35. Minor, the `_relayInterest()` function requires the balance of the bridge to be greater than the relayed interest. Those are unrelated units and should not be compared, as the balance could be 0 and interest still be positive.

2. InterestConnector, line 148. Minor, in the `payInterest()` function second if clause should be (`balance + interest > minCash`). Greater or equal is not right because it could result in a zero withdrawal. And the comparison with the `minInterestPaid(_token)` is always held at this point. `minCash > balance -> minCash - balance > 0 -> interest + 1+ >= minInterestPaid(_token)` because of the require statement on line 142.

3. SavingsDaiConnector, line 11. Optimization, the `SavingsDaiConnector` contracts' constant `SUCCESS` is not used.

4. XDaiForeignBridge, line 44. Optimization, the `refillBridge()` function reads `minCashThreshold` variable value from storage twice.

Anderson Lee

Oleksii Matiiasevych