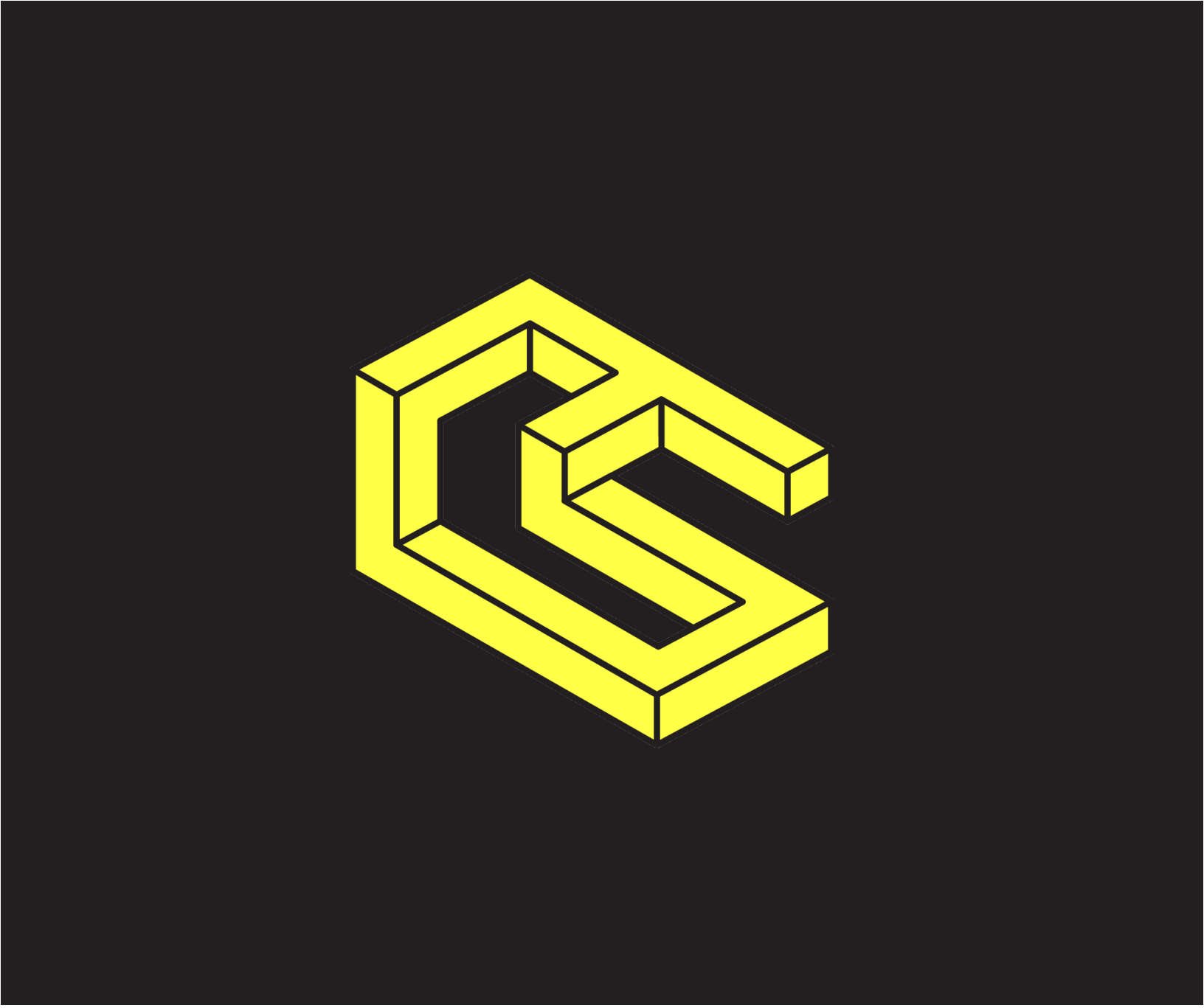

Gnosis Chain SBC Withdrawals Smart Contracts Review

By: ChainSafe Systems

May 2023

Gnosis Chain SBC Withdrawals Smart Contracts Review

Auditors: Tanya Bushenyova, Anderson Lee, Oleksii Matiiasevych

WARRANTY

Introduction

Gnosis Chain requested ChainSafe Systems to perform a review of the contracts used for SBC (Stake Beacon Chain) deposit and withdrawal. The contracts in scope can be identified as the following git commit hash:

`63d522e40dfaacde5f00891ca45c86ad474e6184 (pull request #25)`

Gnosis Chain has the following contracts in scope:

```
SBCWrapper.sol (unwrap() function)
SBCDepositContract.sol (contract modifications in the pull request
#25)
```

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

`13e155500b626612844e3d0fccc11b02b11ea785`

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 (0.8.9), that might affect the contracts' logic.

There were no critical, major or minor issues found. 11 informational/optimizational issues were identified in the contracts. The most important optimizational issue, which is using the NonReentrant modifier, was addressed along with a couple of less important ones. As there were no logic issues identified, the team decided to acknowledge most of the optimizational findings in favor of code readability.

Critical Bugs and Vulnerabilities

No critical bugs or vulnerabilities were identified in the contracts.

Line by Line Review. Fixed Issues

1. SBCDepositContract, line 290. Optimization, the `FailedWithdrawalRecord` storage variable could check if the `amount` is zero instead of storing a `processed` value.

2. SBCDepositContract, line 311. Optimization, the `processFailedWithdrawal()` function could omit using the `NonReentrant` modifier if the `failedWithdrawalRecord` update is moved to be done before the `_processWithdrawal()` call.

3. SBCDepositContract, line 348. Optimization, the `processFailedWithdrawalsFromPointer()` function could omit using the `NonReentrant` modifier if the `failedWithdrawalRecord` update is moved to be done before the `_processWithdrawal()` call, and undoing the changes in case of fail.

Line by Line Review. Acknowledged Findings.

1. SBCDepositContract, line 266. Note, the `gasLimit` parameter of the `_processWithdrawal()` function is not described.

2. SBCDepositContract, line 317. Optimization, the `processFailedWithdrawal()` function reads `FailedWithdrawalRecord` values from storage multiple times, consider using a local variable instead.

3. SBCDepositContract, line 320. Note, the `processFailedWithdrawal()` function uses a `_msgSender()` helper while the deposits processing part of the contract uses `msg.sender`.

4. SBCDepositContract, line 355. Optimization, the `processFailedWithdrawalsFromPointer()` function reads `failedWithdrawalsPointer` value from storage multiple times, consider using a local variable instead.

5. SBCDepositContract, line 358. Optimization, the `processFailedWithdrawalsFromPointer()` function reads `FailedWithdrawalRecord` values from storage multiple times, consider using a local variable instead.

6. SBCDepositContract, line 406. Optimization, the `executeSystemWithdrawals()` function reads `_addresses[i]` from calldata multiple times, it would be cheaper to store it in a local variable.

7. SBCDepositContract, line 411. Optimization, the `executeSystemWithdrawals()` function reads `numberOfFailedWithdrawals` value from storage multiple times, consider using a local variable instead.

8. SBCDepositContract, line 425. Note, the `_token` parameter of the `unwrapTokens()` function is not described.

Tanya Bushenyova

Anderson Lee

Oleksii Matiiasevych