certora

Security Assessment Report

aave

sGHO Vault

September 2025

Prepared for:
Aave DAO

Code developed by:

kpk

Table of content

Project Summary

Project Scope

Project Name	Repository (link)	Latest Commit	Platform
sGHO Vault	https://github.com/aave-dao/gho-origin	PR#5 (initial) 0993a90 (fix)	Solidity

Project Overview

This document describes the manual code review findings of **sGHO ERC4626 Vault**. The following contract list is included in our scope:

`src/contracts/sgho/sGHO.sol`

The work was undertaken from **September 3, 2025,** to **September 8, 2025**. During this time, Certora's security researchers performed a manual audit of all the Solidity contracts and discovered several bugs in the codebase, which are summarized in the subsequent section.

Protocol Overview

The sGHO contract is a typical ERC4626 implementation with a few modifications. Users deposit assets and get shares based on the current yield index. Over time, the yield index will grow based on the target rate set by the yield manager which allows users to receive more assets than deposited. The vault works on a first-come-first-served basis as the contract might not have enough GHO balance to support all withdrawals with their interest included.

Findings Summary

The table below summarizes the findings of the review, including type and severity details.

Severity	Discovered	Confirmed	Fixed
Critical	-	-	-
High	-	-	-
Medium	-	-	-
Low	1	1	-
Informational	1	1	1
Total	2	2	1

Severity Matrix

Impact	High	Medium	High	Critical
	Medium	Low	Medium	High
	Low	Low	Low	Medium
		Low	Medium	High
		Likelihood		

Detailed Findings

ID	Title	Severity	Status
L-01	Users can DoS vault actions by triggering maxAction() requires	Low	Acknowledged
I-01	Lack of pausability mechanism limits admin emergency action	Informational	Fixed

Low Severity Issues

L-01. Users can DoS vault actions by triggering maxAction() requires

Severity: **Low**	Impact: **Medium**	Likelihood: **Low**
Files: sGHO.sol	Status: Acknowledged	

Description

The sGHO.sol implements ERC4626, which has deposit()/mint() and withdraw()/redeem() actions. These actions are bound by maxDeposit()/maxMint(), maxWithdraw()/maxRedeem() checks, respectively.

These checks perform the following validation:

- For deposit/mint, it checks that the deposit will not put the vault above $.supplyCap
- For withdraw/redeem, it checks that the vault has enough underlying token balance to fulfill the withdrawal

The only requirements for someone to deposit or withdraw are, respectively, that they must have enough GHO to deposit, or sGHO to withdraw. So any user can DoS vault actions by triggering the maxAction() check. For example:

- Innocent user wants to deposit 1,000 GHO into the vault
- Attacker front-runs their deposit, making a deposit such that $.supplyCap - currentAssets < 1000
- The innocent user's transaction will revert with a ERC4626ExceededMaxDeposit() error
- Attacker back-runs this transaction, withdrawing the GHO back

Recommendation

There is no way of completely eliminating this vector, but the following mitigations limit the impact or feasibility of this attack:

- Include deposit or withdrawal fees: this will make it expensive for an attacker to perform this attack
- Prevent deposits and withdrawals in the same block: this will introduce an additional constraint for the attacker, as they must hold the funds for more than a block in order to perform it

Informational Issues

I-01. Lack of pausability mechanism limits admin emergency action

Description

The sGHO.sol contract relies on active admin action, such as setting target rate, supply cap, adding the funds equivalent to the accrued yield, and if necessary, rescuing tokens mistakenly sent to the contract.

However, the contract does not implement a pausability mechanism. As a result, the admin cannot temporarily halt user interactions in emergency situations, which may limit their ability to respond to unexpected risks.

Recommendation

Consider implementing pausability on external user actions - deposit(), mint(), withdraw(), and redeem().

Fix Review

Fixed in 0993a90.

Disclaimer

Even though we hope this information is helpful, we provide no warranty of any kind, explicit or implied. The contents of this report should not be construed as a complete guarantee that the contract is secure in all dimensions. In no event shall Certora or any of its employees be liable for any claim, damages, or other liability, whether in an action of contract, tort, or otherwise, arising from, out of, or in connection with the results reported here.

About Certora

Certora is a Web3 security company that provides industry-leading formal verification tools and smart contract audits. Certora's flagship security product, Certora Prover, is a unique SaaS product that automatically locates even the most rare & hard-to-find bugs on your smart contracts or mathematically proves their absence. The Certora Prover plugs into your standard deployment pipeline. It is helpful for smart contract developers and security researchers during auditing and bug bounties.

Certora also provides services such as auditing, formal verification projects, and incident response.