Security Assessment & Formal Verification Report

Aave Delivery Infrastructure Shuffle System

June-2024

Prepared for

AAVE

Table of content

Project Summary

Project Scope

Project Name	Repository (link)	Latest Commit Hash	Platform
aDI-Shuffle	aDI	a875de6	EVM/Solidity 0.8

Project Overview

This document describes the specification and verification of the **aDI-Shuffle** using the Certora Prover and manual code review findings. The work was undertaken from **23 May 2024 to 5 June 2024**.

The following contract list is included in our scope:

- Utils
- CrossChainForwarder

The Certora Prover demonstrated that the implementation of the Solidity contracts above is correct with respect to the formal rules written by the Certora team. In addition, the team performed a manual audit of all the Solidity contracts**.** During the verification process and the manual audit, no bug was discovered. (Anyhow we have one informational issue that we list below.)

For more information about the modifications please refer to the following PR.

Protocol Overview

The contracts under review are part of the Aave Delivery Infrastructure and introduce new shuffling logic to the bridge adapters used to forward a message and includes:

- Adding optimalBandwidth which specifies for each chain what is the optimal number of bridges through which an envelope could be sent. If the optimalBandwidth is less than the number of forwarders and greater than 0, then the number of bridge adapters will be optimalBandwidth, and these adapters will be chosen pseudo-randomly. With this new logic aDI can have any number of allowed forwarded for a specific destination, without increasing the cost of forwarding a message.
- Adding the shuffling logic which is used to choose the forwarders.

Coverage

1. We wrote several new rules in order to check the shuffling mechanism. See more information later.
2. We ran the already existing rules of the aDI.
3. With respect to manual auditing we have checked the following:
   - We have checked that setting the optimal bandwidth to 0 or larger/equal to the length of the forwarders means that all adapters will be used.
   - We have checked that setting the optimal bandwidth to 0< bandwidth< LengthOfForwardes will use bandwidth forwarders which are selected pseudo-randomly.
   - We have checked that the distribution which pseudo-randomize the forwarders is uniformal.
   - We have checked that the new shuffling logic is being used when forwarding a message.

Findings Summary

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

Severity	Discovered	Confirmed	Fixed
Critical			
High			
Medium			
Low			
Informational	1		
Total			

Severity Matrix

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

Likelihood

Detailed Findings

ID	Title	Severity	Status
I-01	Non optimal technique for choosing adapters	Informational	

Informational Severity Issues

I-01. Non optimal technique for choosing adapters

Description: In order to choose the adapters that are used to send a transaction the contract uses a non optimal technique that we now describe. Assume that there are n potential adapters, and we need to use only k of them (0<k<n). Starting with the array (of indexes of adapters) arr=[0,1,..,n-1], the algorithm iterates n time over arr, and in iteration i it does the following:
Randomly choose a position t in the array, and then swap the values arr[i] and arr[t].
Finally the contract uses the k adapters whose indexes are arr[0],,..,arr[k-1].

Recommendation: We suggested the following more efficient algorithm, that only iterates k times over the array of indexes. Here is its pseudo code:

```
arr = [0,1,...,n-1];
for (i=0; i<k; ++i) {
  int pos = rand(n-i);  // rand(t) returns a random number in 0,..,t-1
  swap (arr[i+pos] , arr[i]);
}
return arr[0..k-1]; // namely, use the k adapters whose indexes are arr[0],,..,arr[k-1].
```

BGD Labs response: Suggestion was implemented on this PR.

Formal Verification

Verification Notations

Formally Verified	The rule is verified for every state of the contract(s), under the assumptions of the scope/requirements in the rule.
Formally Verified After Fix	The rule was violated due to an issue in the code and was successfully verified after fixing the issue
Violated	A counter-example exists that violates one of the assertions of the rule.

Formal Verification Properties

In the table below we specify all the formally verified rules that we wrote for the verification of the aDI-shuffle, and give a detailed description for them. A link to the Certora's prover report can be found here.

P-01. shuffle__amount_of_bridges

Status: Verified	Property Assumptions:

Rule Name	Status	Description	Rule Assumptions
shuffle__amount_of_bridges	Verified	*Check that the amount of bridges is in accordance with the value of the optimal-bandwidth. Namely, if the amount is 0 or bigger than the total number of bridges we use all the bridges, and otherwise the number of bridges is the optimal-bandwidth.*	

P-02. shuffle__uniqueness_of_bridges

Status: Verified	Property Assumptions:

Rule Name	Status	Description	Rule Assumptions
_shuffle__uniqueness_of_bridges	Verified	*Check that the shuffling process doesn't produce the same adapter more than once. Namely we check that all the adapters that are passed to the function _bridgeTransaction are different from each other.*	*We only check for the case that the number of bridge-adapters is 4, and the optimal bandwidth is 3.*

Disclaimer

The Certora Prover takes a contract and a specification as input and formally proves that the contract satisfies the specification in all scenarios. Notably, the guarantees of the Certora Prover are scoped to the provided specification and the Certora Prover does not check any cases not covered by the specification.

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.