Formal Verification of Aave's rescue mission Phase 1

Summary

This document describes the specification and verification of Aave's rescue mission phase 1 using the Certora Prover. The work was undertaken from January 6th to January 12th. The latest commit reviewed and run through the Certora Prover was 4c59da8d.

The scope of our verification was the following contracts:

- AaveTokenV2.sol
- LendToAaveMigrator.sol
- StakedTokenV2Rev4.sol

The Certora Prover proved the implementation is correct with respect to the formal rules written by the Certora team. During the verification process, the Certora Prover alerted on several issues listed in the table below. All issues were promptly corrected, and the fixes were verified to satisfy the specifications up to the limitations of the Certora Prover. The next section formally defines high-level specifications of the contracts. All the rules are publically available in the Aave-rescue-mission github repositoryXXX.

List of Main Issues Discovered

Recommendations

There's a risk of draining more funds than intended from contracts that hold some underlying asset. Trying to rescue the underlying token from there may cause under-collateralisation that can cause serious issues; therefore, we suggest adding the following checks:

Potential Issue:	**Rescue of AAVE tokens from the `StakedTokenV2Rev4` contract**
Rules Broken:	Property #3: StkAaveIsBackedByAave

Potential Issue:	**Rescue of AAVE tokens from the `StakedTokenV2Rev4` contract**
Recommendation:	Add a requirement that demands the holding token (stkAAVE) is always backed with a sufficient amount of underlying asset (AAVE), i.e: $stkAAVE.totalSupply() \leq AAVE.balanceOf(stkAAVE)$
Mitigation/Fix:	Check added in commit 4c59da8d

Potential Issue:	**Rescue of AAVE and LEND tokens from the `LendToAaveMigrator` contract**
Rules Broken:	Property #1: LendIsBackedByAave
Recommendation:	Add a requirement that demands all the LEND that wasn't sent for swap in the migrator be fully collateralised with AAVE. This means that: $\frac{LEND.totalSupply() - LEND.balanceOf(LEND)}{LEND\_AAVE\_RATIO} \leq AAVE.balanceOf(migrator)$
Mitigation/Fix:	Check added in commit 4c59da8d

Potential Issue:	**Potential miscount of burnt LEND (introduced in commit 759275c)**
Rules Broken:	Property #1: LendIsBackedByAave
Recommendation:	The current form of requirement in the `initialize` function demand: $\frac{LEND.totalSupply() - LEND.balanceOf(LEND) - LEND.balanceOf(AAVE)}{LEND\_AAVE\_RATIO} \leq AAVE.balanceOf(migrator)$ This means that the AAVE locked in the migrator should be accountable for at least the entire circulating LEND, i.e. all the LEND that wasn't burnt by sending it to the LEND contract. Still, it should also **not** be accountable for the entire LEND locked in the AAVE contract. This addition was introduced since the two upgrades will take effect a-synchronously - first, the LEND migrator rescue will take place, then the AAVE rescue. However, the payload of the rescue from the AAVE contract specify a concrete amount of LEND to rescue rather than the entire LEND balance of this contract (`LEND.balanceOf(AAVE)`). This can lead to a miscount in the circulating LEND (the left side of the inequality), which will result in demanding we have at least a smaller amount of AAVE locked in the contract than there should be.
Mitigation/Fix:	Fixed in commit c905eaf

Disclaimer

The Certora Prover takes as input a contract and a specification and formally proves that the contract satisfies the specification in all scenarios. Importantly, 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.

We hope that this information is useful but 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.

Overview of the verification

Description of the system

The project introduces upgrades to existing contracts to which community members have mistakenly sent tokens in the past. The changes will be made to the initialised method, aiming to rescue these locked assets and pass them on to their rightful owners.

Assumptions and Simplifications

We made the following assumptions during the verification process:

- We unroll loops. Violations that require executing a loop more than once will not be detected.
- We assumed the `onTransfer` hook doesn't change the state of any contract.

Notations

✔️ indicates the rule is formally verified on the latest reviewed commit. We write ✔️* when the rule was verified on a simplified version of the code (or under some assumptions).

❌ indicates the rule was violated under one of the tested versions of the code.

✍️ indicates the rule is not yet formally specified.

🔁 indicates the rule is postponed (<due to other issues, low priority>) .

We use Hoare triples of the form {p} C {q}, which means that if the execution of program C starts in any state satisfying p, it will end in a state satisfying q. In Solidity, p is similar to require, and q is similar to assert.

The syntax {p} (C1 ~ C2) {q} is a generalisation of Hoare rules, called relational properties. {p} is a requirement on the states before C1 and C2, and {q} describes the states after their executions. Notice that C1 and C2 result in different states. As a special case, C1~op C2, where op is a getter, indicating that C1 and C2 result in states with the same value for op.

Formal Properties

`LendToAaveMigrator.sol`

1. LendIsBackedByAave ^[reqs] ❌

All the LEND that wasn't sent for swap in the migrator must be fully collateralised with AAVE.

```
(LEND.totalSupply() - LEND.balanceOf(LEND)) / LEND_AAVE_RATIO() ≤ AAVE.balance
```

2. LendIsBackedByAaveIncInitialize ^[reqs] ✔

A direct extension of the previous property - `LendIsBackedByAave`, showing that after calling `initialize` in AAVE token the invariant is preserved.^[reqs2]

```
{
    (LEND.totalSupply() - LEND.balanceOf(LEND)) / LEND_AAVE_RATIO() ≤ AAVE.bal
}
    ( < call to initialize > ∧ < call to AAVE.initialize > )
    ∨
    < call to any other function >
{
    (LEND.totalSupply() - LEND.balanceOf(LEND)) / LEND_AAVE_RATIO() ≤ AAVE.bal
}
```

`StakedTokenV2Rev4.sol`

3. StkAaveIsBackedByAave ✔

The holding token (stkAAVE) is always backed with a sufficient amount of the staked asset (AAVE).

```
totalSupply() ≤ STAKED_TOKEN.balanceOf(address(this))
```

^[reqs]: We assume the initiator of the action (`msg.sender`) is neither the LEND token contract, nor the AAVE V2 token contract.