Aave V3 MAI and FRAX Verificaton and Listing Stewards Audit

Scope

The scope of the assessment is the contracts `QiStablecoin.sol` (`miMATIC`), `CrossChainCanonicalFRAX.sol` (`FRAX`) which were formally verified, and `AaveV3FantomMIMATICListingSteward.sol`, `AaveV3AvaMAIListingSteward.sol`, `MiMaticPayload.sol`, `AaveV3FantomFRAXListingSteward.sol`, `AaveV3AvaFRAXListingSteward.sol` and `FraxPayload.sol` which were manually audited due to the fact that the implementation involves only calls to function with concrete values. The latter 6 contracts defines `MAI` and `FRAX` configuration on the Aave V3 platform on Avalanche, Fantom and Polygon networks.

These contracts had 2 security engineers and 1 security researcher reviewing the code in detail.

The verification of the token was finished on the 28th of June, reviewing the deployed contracts of `MAI` and `FRAX`.

The audit was finished on the 11th of August, reviewing commits 380db3b5, 2d6b9d34, d70d69cf, a29eb520, a29eb520 and d70d69cf of the respective listing stewards and payloads.

Contracts Overview

As part of our continues formal verification for Aave, we've inspected the `MAI`, `FRAX` tokens for security issues and non-trivial features. You can see the results in our Aave dashboard.

The audited contracts' purpose is to configure `MAI` as a borrowing asset and `FRAX` as a borrowing and collateral asset on the Aave V3 platform on Avalanche, Fantom and Polygon networks.

Six contracts were audited:

1. `AaveV3FantomMIMATICListingSteward.sol`, `AaveV3AvaMAIListingSteward.sol` and `MiMaticPayload.sol`, which configures `MAI` as a borrowing asset on all three chains, and as a collateral asset on Polygon for the Aave V3 platform. All three contracts contains 2 steps:

   1.1 Setting a price feed on the Aave oracle for the `MAI` token - see on avalanche, see on fantom, see on polygon.

   1.2 Listing the token onto Aave V3 protocol and configuring it as a brrowing asset - see on avalanche, see on fantom, see on polygon.

   1.3 `MAI` 's Polygon payload extends the usability of the token with a third step which configures the token as a collateral asset in the stablecoins' e-mode category - see on polygon.

2. `AaveV3AvaFRAXListingSteward.sol.sol`, `AaveV3FantomFRAXListingSteward.sol` and `FraxPayload.sol`, which configures `FRAX` as a collateral asset on Aave V3 platform. The contract contains 2 steps:

   2.1. Setting a price feed on the Aave oracle for the `FRAX` token - see on avalanche, see on fantom, see on polygon.

   2.2. Listing the token onto Aave V3 protocol and configuring it as a collateral asset in the stablecoins e-mode category - see on avalanche, see on fantom, see on polygon.

Audit Goals

Since all 6 contracts are rather similar, the same set of criteria and checks were performed on each of them. Below is the list of checks that were performed:

Addresses

1. All addresses of external contracts being used are matching the existing contracts on the relevant networks.

Correct Setting Of Parameters And Values

2. The asset is being added to the system correctly with all the correct `InitReserveInput` struct values. Additional parameters are being passed in the correct methods and with the right decimals, e.g. `SUPPLY_CAP`, `RESERVE_FACTOR` and `LIQ_PROTOCOL_FEE`.

3. `MAI` (on Polygon) and `FRAX` are being configured as collateral with proper relations between the LTV, threshold and liquidation bonus.

Privileges

Polygon

4. `ACL_ADMIN` has the neccesery role to grant itself the `POOL_ADMIN` role.

Avalanche and Fantom

5.1. The contracts has the necessary roles to execute `listAssetAddingOracle()` without reverting.

5.2. The roles are being renounced from the contract at the end.

Findings And Recommendations

Severity: high

Issue:	Wrong address set for underlying asset in `MAI` on Avalanche.
Description:	In `AaveV3AvaMAIListingSteward.sol`, the address used as the `MAI` underlying asset on Avalanche refers to a non-official `MAI` token. The address in use points to relay `MAI` - a “synthetic” `MAI` minted by the relay bridge that was in common use before `MAI` officially deployed on Avalanche.
Aave Response:	This issue was fixed in commit 2d6b9d34.

Recommendation

Issue:	Verify that the contracts has necessary privileges.
Description:	Many function calls in the listing process require the listing contract to have both `Asset Listing Admin` and `Risk Admin` roles in order to be successfully executed. It is important to remember granting those roles prior to calling `listAssetAddingOracle()`.
Recommendation:	Add a dedicated require condition at the beginning of `listAssetAddingOracle()` in order to give a clearer error message in case of such a revert case.

Informational

Issue:	Mismatch between the proposal's snapshot and values on the payload contracts.

Issue:	Mismatch between the proposal's snapshot and values on the payload contracts.
Description:	There exist several mismatches between the values voted on in the snapshot proposals and the actual values configured in the payloads for both MAI & FRAX on Polygon: 1. The snapshot proposal suggested that `MAI` will be configured as an isolated collateral asset, however the contracts configure the token only as a borrowable asset on the Fantom chain. 2. While the proposal voted on a 5% reserve factor and a 50M$ debt ceiling, the payload actually configures a 10% reserve factor and a 2M$ debt ceiling. The snapshots and payloads can be found here: MAI's snapshot, MAI's payload, FRAX's snpashot, FRAX's payload)
Aave Response:	1. Aave technical contributor (BGD) conversed with the `MAI` team and decided that it's better to list the token with a more conservative approach. As part of that it was decided not to list the token as collateral on Fantom due to relatively low liquidity. The asset could be configured as a collateral in the future if needed. 2. The parameters that were proposed on these snapshots are'nt aligned with other assets of the market. It is usually better to start with conservative values and adjust later. Additionally, 10% reserve factor is a standard across stable coins.

Informational - Non-Standard Behavior:

FRAX

1. The token is mintable.
2. The token is burnable with a `burnFrom()` method.
3. The token has a `permit()` method which allows changing the allowance via singed approval.

You can see the full result in our AAVE dashboard.

Conclusions

Addresses

1. In 5 out of 6 contracts, the addresses specified in the contracts fully matched the existing relevant contracts on the respective blockchains. However for `MIA` on Avalanche the underlying asset address was found to point on an older relay bridge

token instead of the original `MIA` token. This issue is explained in details in the table above.

Correct Setting Of Parameters And Values

2. The asset has been added to the system correctly with all the correct `InitReserveInput` struct values, and all parameters are being passed in the correct methods and with the right decimals.
3. The relations between the LTV, Threshold and Liquidation bonus for `FRAX` and `MAI` on Polygon found to be configured reasonably.

Privileges

Polygon

4. `ACL_ADMIN` was granted with the `POOL_ADNIM` 's admin role so it can grant itself the `POOL_ADMIN` role.

Avalanche and Fantom

5.1. The roles to the contracts are being given externally. Therefore, the executor should make sure to delegate the necessary privileges before trying to execute.

5.2. At the end of the process the contract renounce it's privileges in a correct manner.

Disclaimer