Aave CLSynchronicity Price Adapter Formal Verification and Manual Review

Scope

The scope of this assessment includes five contracts: `CLSynchronicityPriceAdapterBaseToPeg.sol`, `CLSynchronicityPriceAdapterPegToBase.sol`, `CLwstETHSynchronicityPriceAdapter.sol`, `ProposalPayloadStablecoinsPriceAdapter.sol` and `ArcProposalPayloadStablecoinsPriceAdapter.sol`, which were manually reviewed, as the implementation only calls functions with concrete values. One security engineer and one security researcher reviewed the code in detail.

The review was completed on the 8th of December, reviewing commit cf40a4f.

Contracts Overview

Different price feeds of asset-to-a-common-base-asset may be updated at different frequencies. Such delays in updates between them introduce unnecessary volatility in positions.

The ChainLink synchronicity Price Adapters introduce another layer of price update synchronicity between assets with high price correlation via adapter contracts.

For example, the Aave v2 pool on Ethereum uses ETH-based oracles to calculate the collateral value, debt value and health factor of a user. The adapter will replace the current ETH-based oracles for stablecoins by using USD pairs instead and normalising the USD price using the ETH oracle on the Aave v2 Ethereum and Aave Arc markets. The stabilisation is done according to one of the following formulae:

$$Price(Asset/Base) = \frac{DataFeed(Asset/Peg)}{DataFeed(Base/Peg)}$$

$$Price(Asset/Base) = DataFeed(Asset/Peg) \cdot DataFeed(Peg/Base)$$

Where `Base` is typically `ETH`, `Peg` is typical `USD`, and `Asset` is the asset of interest.

Five contracts were reviewed:

- `CLSynchronicityPriceAdapterBaseToPeg.sol` is a general price adapter for pegged assets. It uses the `Asset/Peg`, and `Base/Peg` data feeds to calculate the stabilised `Asset/Base` price according to the formula above.
- `CLSynchronicityPriceAdapterPegToBase` is a general price adapter for pegged assets that uses the `Asset/Peg` and `Peg/Base` data feeds to calculate the stabilised `Asset/Base` price according to the formula above.
- `CLwstETHSynchronicityPriceAdapter` is a dedicated price adapter for `wstETH` that uses `CLSynchronicityPriceAdapterPegToBase` and the `stETH` contract's getter to calculate the price of `wstETH/USD`.
- `ProposalPayloadStablecoinsPriceAdapter.sol` and `ArcProposalPayloadStablecoinsPriceAdapter.sol`
  - Create and deploy `CLSynchronicityPriceAdapterBaseToPeg` for each stablecoin on Aave and Arc Aave - deploying adapters on Aave, deploying adapters on Arc Aave.
  - Configuring the new price adapters on the Aave and Arc Aave oracles - see in code for aave, see in code for arc aave.

Manual Review Goals

During the code review, the following checks were performed:

ProposalPayloadStablecoinsPriceAdapter.sol and ArcProposalPayloadStablecoinsPriceAdapter.sol

1. All addresses of external contracts match the existing contracts on the Ethereum network.

   1.1. All instances of `CLSynchronicityPriceAdapterBaseToPeg` are being created with the correct aggregators.

   1.2. The oracle is set with the correct asset sources - assets and aggregators' addresses.

Correct Setting of Parameters and Values

2. All the aggregators supply prices with the exact decimals as the `Base` aggregator as required in the constructor of `CLSynchronicityPriceAdapterBaseToPeg`.
3. All `CLSynchronicityPriceAdapterBaseToPeg` were created with the correct parameters, and the `assetSource` was set with correctly correlated arrays.
4. All `CLSynchronicityPriceAdapterBaseToPeg` price adapters were created with decimals compatible with the decimals used in calculations on the Aave v2 protocol.

CLSynchronicityPricAdapterBaseToPeg.sol, CLSynchronicityPriceAdapterPegToBase.sol and CLwstETHSynchronicityPriceAdapter.sol

5. `latestAnswer()` returns a correct representation of 1 `Asset` in `Base`, i.e. the formula returns `Asset/Base`.
6. The current order of arithmetic operations in `latestAnswer()` prevents significant rounding errors in the oracle's answer.

Issues

Severity: Low

Issue:	**Unnecessary revert of `CLSynchronicityPriceAdapter` creation (commit 86f432e)**
Description:	In a case where the specified `resultDecimals` (`r`) is smaller than the difference between the `assetToPegDecimals` (`a`) and `baseToPegDecimals` (`b`), the `DECIMALS_MULTIPLIER` (`d`) will be evaluated to 0 and thus cause a revert upon constructor. $d = \frac{10^r}{10^{a-b}}$ This revert case is unnecessary since one can always decrease precision by ignoring the n least significant digits of a more precise value.
AAVE Response:	The issue was fixed in commit a6fdbd2

Severity: Low

Issue:	**Adapter may retrieve a positive price in case of two negative Chainlink oracle results (commit a6fdbd2)**
Description:	Chainlink's oracle retrieves an exchange rate in type `int`. Since it is possible to get a negative exchange rate, it is recommended to check

Issue:	Adapter may retrieve a positive price in case of two negative Chainlink oracle results (commit a6fdbd2)
	that the price returned directly from the oracle is positive. In Aave, the check is being done in `AaveOracle`; however, `CLSynchronicityPricAdapterBaseToPeg.sol` and `CLSynchronicityPriceAdapterPegToBase.sol`, return a normalized price by multiplying/dividing oracle results by one another. This allows a case where both chainlink oracles, `asset to peg` and `base to peg` or `peg to base`, return a negative result; however, the adapter returns an unwanted positive result which will pass the check in `AaveOracle` and use the wrong adapter result instead of turning to the fallback oracle.
AAVE Response:	The issue was fixed in commit cf40a4f

Conclusions

ProposalPayloadStablecoinsPriceAdapter.sol and ArcProposalPayloadStablecoinsPriceAdapter.sol

1. All asset addresses specified in the contract match relevant contracts on the Ethereum blockchain. In addition, all specified aggregators' addresses were checked to match 3 official sources: 1.1. Etherscan - the address points to a price aggregator of the specified pair and is uploaded by a familiar chainlink address. 1.2. Chainlink's data feed section on the official website. 1.3. Chainlink's price feed contract addresses list of the Ethereum mainnet on the official Chainlink documentation.

Correct Setting of Parameters and Values

2. `ETH/USD` and all `asset/USD` aggregators return a result with precision of 8 decimals as required by the `CLSynchronicityPriceAdapterBaseToPeg` constructor.
3. All `CLSynchronicityPriceAdapterBaseToPeg` are fed with the correct set of parameters. The `assetSource` is set with correctly aligned arrays of assets and their respective `CLSynchronicityPriceAdapterBaseToPeg`.
4. All `CLSynchronicityPriceAdapterBaseToPeg` price adapters are created with `result decimals = 18`, the decimals used in calculations on the Aave v2 protocol.

CLSynchronicityPricAdapterBaseToPeg.sol, CLSynchronicityPriceAdapterPegToBase.sol and CLwstETHSynchronicityPriceAdapter.sol

5. The units of measure in which `latestAnswer()` returns the data is indeed `Asset/Base`.

   5.1. In `CLSynchronicityPricAdapterBaseToPeg`, the price feeds return answers in `Asset/Peg` and `Base/Peg` units, respectively, and then divide them by one another to achieve the desired measures.

   5.2. In `CLSynchronicityPriceAdapterPegToBase`, the price feeds return answers in `Asset/Peg` and `Peg/Base` units, respectively, and then multiply them by one another to achieve the desired measures.

   5.3. `CLwstETHSynchronicityPriceAdapter` uses `CLSynchronicityPriceAdapterPegToBase` to retrieve the price of `stETH/USD` and the `stETH` contract to retrieve the price ratio of `wstETH/stETH` to achieve the desired measures.

6. In all calculations, divisions are done at the end, which reduces the rounding errors to minimum.

Disclaimer