NOTE Design

Since Note cannot be created, only borrowed, and the Supply rate is the same as the Borrow rate, a new mechanism was necessary to keep track of how much of the interest paid by to borrow Note goes to the Accountant (and ultimately the Community Treasury) versus how much is paid to external suppliers of Note.

This is achieved by supplying and redeeming liquidity during every action taken by an external user of cNote, using an additional call implemented in the mintFresh/redeemFresh, borrowFresh/repayBorrowFresh internal methods in CToken.sol:

1. When usersborrow Note or redeem cNote
2. , the function in CNote will call another function in the Accountant to supply cNote in the exact amount required to offset the request.
3. When Users supply or repay Note to the market, the function in CNote will call another function in Accountant to redeem exactly the same amount of Note from the market.
4.

As a result, there is never any Note present in the CNote market other than during function calls, but there is an infinite amount of Note that can be borrowed or redeemed.

This also prevents supply inflation, as idle Note in the lending market pool would otherwise earn interest despite not being lent out.

It is important to note that the internal price in the lending market used to calculate liquidity for Note is always set to 1 USD regardless of the value of Note in the DEX.

Architecture diagram

Our implementation of Note’s architecture is described below:

Previous Gravity Bridge Next Canto Lending Protocol Last updated3 months ago On this page