Contracts for Curve DAO

Curve DAO consists of multuple smart contracts connected by Aragon. Apart from that, standard Aragon's 1 token = 1 vote method is replaced with the voting weight also proportional to locktime, as will be described below.

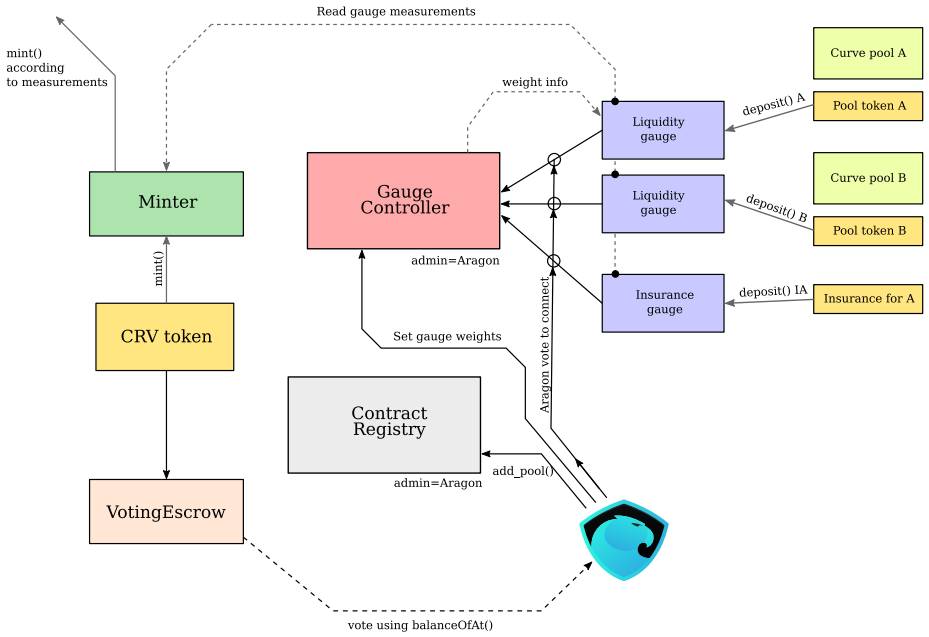

Figure 1: Curve DAO contracts managed by Aragon

Curve DAO has a token CRV which is used for both governance and value accrual.

Time-weighted voting. Vote-locked tokens in VotingEscrow

Instead of voting with token amount a, in Curve DAO tokens are lockable in a VotingEscrow for a selectable locktime t_l, where $t_l < t_{\max}$, and $t_{\max} = 4$ years. After locking, the time *left to unlock* is $t \leq t_l$. The voting weight is equal to:

$$w = a\frac{t}{t_{\max}}.$$

In other words, the vote is both amount- and time-weighted, where the time counted is how long the tokens cannot be moved in fututre.

The account which locks the tokens cannot be a smart contract (because can be tradable and/or tokenized), unless it is one of whitelisted smart contracts (for example, widely used multi-signature wallets).

VotingEscrow tries to resemble Aragon's Minime token. Most importantly, `balanceOf() / balanceOfAt()` and `totalSupply() / totalSupplyAt()` return the time-weighted voting weight w and the sum of all of those weights $W = \sum w_i$ respectively. Aragon can interface VotingEscrow as if it was a typical governance token.

Locks can be created or extended in time or token amount with `deposit()`, and `withdraw()` can remove tokens from the escrow when the lock is expired.

Inflation schedule. ERC20CRV

Token ERC20CRV follows a piecewise linear inflation schedule with inflation dropping by sqrt(2) every year. Only Minter contract can directly mine ERC20CRV but only within the limits defined by inflation.

Each time the inflation changes, a new mining epoch starts.

System of Gauges. LiquidityGauge and GaugeController

In Curve, inflation is going towards users who use it. The usage is measured with Gauges. Currently there is just LiquidityGauge which measures, how much liquidity does the user provide.

For LiquidityGauge to measure user liquidity over time, the user deposits his LP tokens into the gauge using `deposit()` and can withdraw using `withdraw()`.

Coin rates which the gauge is getting depends on current inflation rate, and gauge type weights (which get voted on in Aragon). Each user gets inflation proportional to his LP tokens locked. We do iterated calculations of inverse locked supply integral, so it is not required for user to periodically check in.

GaugeController keeps a list of Gauges and their types, with weights of each gauge and type.

Gauges are per pool (each pool has an individual gauge).

Weight voting for gauges

Instead of simply voting for weight change in Aragon, users can allocate their vote-locked tokens towards one or other Gauge (pool). That pool will be getting a fraction of CRV tokens minted proportional to how much vote-locked tokens are allocated to it. Eeach user with tokens in VotingEscrow can change his/her preference at any time.