In Aave V3, reserve income does not automatically accrue to the treasury, unlike in the case of Aave V2 with the Collector contract. Instead, the function [mintToTreasury(address[] calldata assets)

](https://docs.aave.com/developers/core-contracts/pool#minttotreasury) on the [Pool

](https://docs.aave.com/developers/core-contracts/pool) contract needs to be called with a list of underlying assets passed in as a parameter in order for accrued reserve income to be minted to the treasury, as per the respective Reserve Factor of those assets.

Llama has set up infrastructure to call that function on a regular basis to ensure that the accrued reserve income is being minted to the treasury for all available assets on all active Aave V3 deployments (i.e Polygon, Optimism, Arbitrum and Avalanche).

Implementation

Pseudocode

Every 7 days and for each network in [Polygon, Optimism, Arbitrum, and Avalanche]:

1. Get the current list of underlying assets in the pool by querying [getReservesList()

](https://docs.aave.com/developers/core-contracts/pool#getreserveslist) on the [Pool

](https://docs.aave.com/developers/core-contracts/pool) contract.

1. Call [mintToTreasury(address[] calldata assets)

](https://docs.aave.com/developers/core-contracts/pool#minttotreasury) on the [Pool

](https://docs.aave.com/developers/core-contracts/pool) contract with the list of assets obtained in 1. and execute the transaction on-chain.

Specification

1. The infrastructure automatically runs and executes the transaction on-chain on a pre-set schedule.
2. The code is written in Python which works well with the rest of our backend and data infrastructure we're building for Aave.
3. It runs on managed cloud infrastructure instead of a local instance so that it is a long living process, and can be managed and monitored effectively.
4. It is able to dynamically query and pass in the current set of active assets into the [mintToTreasury(address[] calldata assets)

](https://docs.aave.com/developers/core-contracts/pool#minttotreasury) function on the [Pool

](https://docs.aave.com/developers/core-contracts/pool) contract.

Tech Stack

We explored a couple of options like Gelato, OpenZeppelin Autotasks, Tenderly Web3 Actions and Forge Scripting.

However, the tech stack we ended up using to satisfy the specification above was Python Web3.py (for interacting with the blockchain), and Modal (for serverless cron jobs).

Executed Treasury Claims

Here is a sample of treasury claims that have been executed across the different networks using this infrastructure:

1. Polygon
2. Optimism
3. Arbitrum
4. Avalanche

In November, we helped claim V3 revenue from Polygon, Optimism, Arbitrum, and Avalanche networks to the Aave treasury and will continue to do so on a weekly basis.

Please let us know if you have any questions.