Bskt Decentralized Autonomous Fund

by *Nemil Dalal <nemild@cryptofin.io>*, *Daniel Que <daniel@cryptofin.io>*

Abstract

We propose a *decentralized autonomous fund* (DAF) that allows an ERC20 token to represent a diversified, regularly rebalanced portfolio of other ERC20 tokens.

A DAF is segregated into two separate components: a *portfolio data component* that dictates what tokens compose a portfolio and a *fund component* that independently rebalances the portfolio using periodic auctions to match the latest *portfolio data*. The *fund contract* allows easy entry and exit, ensuring that investors can exit in advance if they disagree with the proposed allocation in the *portfolio data contract*. Our approach also handles a number of common failure states in the Ethereum ERC20 ecosystem such as airdrops and frozen tokens.

Introduction

An investment fund allows parties to pool capital and invest in projects collectively, reducing transaction costs and allowing them to diversify their assets.

These centralized investment funds have been difficult to replicate in a decentralized form. At one extreme, fully decentralized funds have been too limited, with critical features like rebalancing missing. At the other, these funds have centralized elements that increase the risk of fund loss. Funds such as the infamous DAO contract have also been highly complex, with commensurately large likelihood of bugs and challenging user experiences.

We propose a simple, but powerful *decentralized autonomous fund* (DAF) that allows cryptocurrency investors to track a pre-specified *portfolio data manager*. Portfolio data managers receive fees for periodically updating ERC20 token portfolios. The fund allows investors to easily enter and exit, with suitable notification before rebalancing — periodic, on-chain auctions — begins.

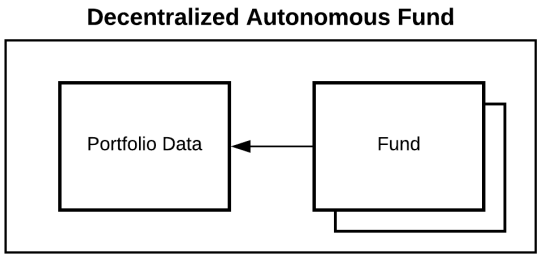

Figure 1: A Decentralized Autonomous Fund (DAF) is composed of one or more *funds* that follow a *portfolio data* contract. The *portfolio data component* provides regularly updated token addresses and weights only. *Funds* track the *portfolio data*, allowing investors to hold the underlying tokens with regular rebalancing.

Unlike traditional funds, the *portfolio data manager* does not exercise custody over the fund, with their role limited to a paid data provider. Fund investors can vote against a proposed rebalancing by redeeming their tokens and exiting.

Our structure enables both regularly rebalanced passive structures (such as a top 10 market cap weighted index), as well as active management. As such, the same smart contract system can enable annually rebalanced, low cost index funds, and sophisticated, continuously rebalanced hedge funds.

Example

First, a *portfolio data manager* publicizes that they are launching a DAF, such as the top prediction market projects.

They deploy a *portfolio data contract* that indicates the ERC20 tokens and weights that comprise the initial composition. At a set interval dictated by the contract, the *portfolio data manager* has the ability — but not obligation — to update the composition and weighting. They can choose to include any ERC20 tokens, including short tokens and tokenized ERC721 portfolios.

Anyone can then deploy an ERC20-compliant *fund contract* that pays the *portfolio data contract* for access to its information.

This *fund contract* allows *market makers* and *investors* to create and redeem *fund* tokens by surrendering the underlying ERC20 tokens that currently comprise the *portfolio data contract*. Most investors will likely acquire or sell their token on exchanges without interacting with the contract directly.

The *fund contract* runs auctions when a *portfolio data contract* has been updated, to rebalance the portfolio to the new allocation. The *fund contract* allows investor to redeem their tokens after a *portfolio data manager* has updated the tokens/weights, but before the rebalancing has occurred. This allows investors to withdraw their money if they disagree with a *portfolio data manager*'s decisions.

Overview

A Bskt DAF is composed of two main components:

1. **Portfolio Data**: In this contract, a pre-selected *portfolio data manager* indicates the ERC20 tokens and weights that compose the portfolio for a given period (e.g., Q1 2018). External parties pay this portfolio data manager to read these regularly updated allocations on chain. While this structure is centrally managed, it can also be trivially replaced by a community managed token curated registry (TCR).
b. **Fund**: The *fund* makes investment decisions based on the portfolio data. To enter, market makers can create/redeem ERC20 fund tokens by surrendering/requesting the underlying tokens. During rebalancing periods, creation/redemption is suspended while the contract runs a auction to convert its previous portfolio to the new portfolio, based on the changes in the *Portfolio Data*

Key Roles

Four main parties interact with the DAF:

- **Portfolio Data Manager**: This is the person or group who determines the composition of a portfolio during preselected change times. They manage the *Portfolio Data* and are paid by one or more *Funds* to read the contract periodically. Over time, they build a reputation by composing a portfolio and regularly updating selected tokens/weights in the contract.
- **Market maker**: This sophisticated investor provides two key services: fulfilling the auctions during rebalancing periods and creating/redeeming tokens when third-party exchange prices go above/below the net asset value.
- **Investor**: Most investors get exposure to a fund by finding the token on an ERC20-compliant exchange. The most sophisticated investors may want to buy the underlying tokens and create *Fund* tokens themselves.
- **Admin**: A few functions are reserved for weakly trusted third-parties, who will help maintain the fund. These generally are meant to only handle unexpected failure states, with the expectation that they will be fully removed over time.

The Portfolio Data Contract

The *portfolio data contract* allows a *portfolio data manager* to receive payments in exchange for periodically picking tokens/weights.

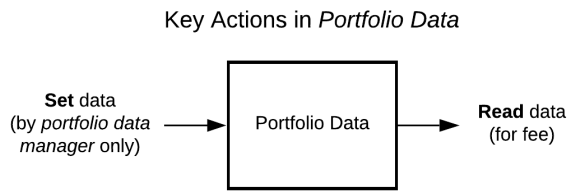

Figure 2: A *portfolio data* contract allows a *portfolio data manager* to set token addresses and weights periodically. External parties can read the data for a fee.

Actions

The *portfolio data contract* allows the *portfolio data manager* to do the following:

- **Update data**: At preset intervals (the *data update period*), the *portfolio data manager* can update the ERC20 token addresses and the amount of each ERC20 token that compose a portfolio. At other times, the *portfolio data manager* cannot change these values.
- **Set fee**: Determine the fee that should be charged for third-parties that want to read data (set only during the *data update period*)
- **Add/remove from whitelist**: Add or remove addresses from a whitelist of parties able to read the data for a minimum fee
- **Close**: Activate a boolean flag that indicates that the fund is closed

Initial Deployment

When initially deploying the contract, a *portfolio data manager* sets:

- The initial token addresses and token quantities
- The fee (either a fixed or asset under management fee to read from the contract)
- The periodic timeframe when the token addresses/quantities can be updated
- Other metadata (a symbol, a string URL where more information can be found)

A third-party can execute a single function that allows them to read the data for a pre-specified fee. This returns an array of the token addresses and selected balances.

Fees

The *portfolio data contract* allows two types of fees: fixed fees and asset under management fees. *Fixed fees* are set by charging an amount to read the data, and can be adjusted by varying the set fee. This fee is paid in ether.

Asset under management (AUM) fees can be charged by allowing whitelisted addresses to read the data, and verifying off chain that they are paying commensurate with their fund size. The *fund contract (below)* is automatically configured to provide these fees to the owner of the *portfolio data contract*.

This fee is paid in the underlying tokens in the fund. The*fund contract* will set aside tokens on behalf of the owner of the *portfolio data contract*.

Alternate Governance models

The standard *portfolio data contract* is configured much like a traditional investment fund: a single person or a small group of people make investment decisions. In other cases, there may be value to an open process, where any number of external parties vote and debate what investments should be made.

The standard *portfolio data contract* can be easily replaced by a *token curated registry (TCR)* with any external party able to propose investment choices and participate in the challenge process.

Fund Contract

The *fund contract* allows sophisticated investors to create and redeem tokens in a fund that tracks a particular *portfolio data contract*.

Actions

Investors can call the following actions on a *fund contract*:

- **Create**: Investors can get fund tokens in exchange for surrendering to the *fund contract* the ERC20 tokens/weights indicated in the current *portfolio data contract*
- **Redeem**: Investors can surrender fund tokens and get underlying ERC20 tokens
- **getComposition**: Investors can request the current composition

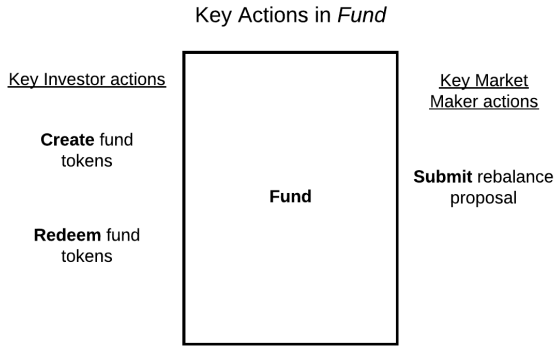

Figure 3: A *fund* lets investors create and redeem fund tokens. Creation lets you trade the underlying tokens for fund tokens, while redeem lets you surrender fund tokens for the underlying tokens. Market makers periodically submit rebalancing proposals, which take the existing tokens and convert their weights to the new weights in the *portfolio data contract*.

Initially, an admin can suspend creation/rebalancing and only allow redemption. Over time, this functionality will be removed.

Anyone (not just investors) can request a *rebalance* during the *rebalancing period*. When a rebalance is requested, the following happens:

- **Pause creation and redemption**: Creation/redemption is paused during the rebalancing period, and enabled as soon as the last auction is complete
- **Collect new portfolio data**: Collect the latest portfolio data after paying a fee to the *portfolio data contract*
- **Determine delta**: Compare the new portfolio data to the existing, and determine how each ERC20 position needs to change
- **Begin rebalancing**: Start a rebalancing process for each changed position that allows you to meet the new composition
- **Pay the party that initiated the rebalance**

Rebalancing

The fund must rebalance trustlessly, exchanging any old tokens for the required weights. The DAF is built to allow a number of different approaches, including:

- **Pooled English Auction**: In a pooled English auction, the fund offers some or all of the current balance of tokens, in exchange for other tokens that achieve the target rebalancing. The bidders compete to provide the best offer, and make money by getting a spread on the offer they make to the *fund*. As long as there is arbitrage potential, bidders compete to get closest to the market prices net of transaction costs.
- **Dutch Auction**: Dutch auctions start at a high value and progressively drop the price until a buyer appears. In the context of trading fungible tokens, it relies on arbitrageurs to make the market when the price difference can turn a profit. These auctions transform each token singly into other tokens, rather than as a pool.
- **On-chain DEX**: On-chain decentralized exchanges, like Kyber and Bancor, provide instant access to market rates and trades. Like many on-chain DEXs, these provide a ready pool of liquidity at the risk of front running. Currently, they also have limited volume and tokens.

The DAF provides a generic interface that allows any of these methods (or others) to be used. Each method will also be paired with a scoring function that determines which bid is deemed the best.

Initially, The DAF will use the *pooled English auction*. Winning bids will submit what tokens/balances they

want to withdraw from the fund, and what tokens/balances they will offer in exchange.

The Fund will select the winning bid with the following initial scoring function:

```
max(min(A1...An), min(B1...Bn), ...)
```

- `A, B,…` are bids submitted by one or many parties
- `A1...An` is the proportion of offered token balance of *n* over target token balance of *n* for a given bid

The bid whose smallest token proportion is the largest wins the auction.

Fees

A fund investor must pay three sets of ongoing fees:

- **Payment to the *portfolio data manager***: This fee is paid either as a fixed fee (e.g., a passive index fund) for reading the data contract or a variable AUM fee (e.g., an actively managed fund) during every rebalancing period. The AUM fee is paid in the underlying tokens in the fund, set aside for the *portfolio data manager* to collect
- **Rebalancing initiation fee**: Rebalancing requires a third-party to pay to begin a rebalancing during the *rebalancing period*. These third-parties are incentivized with an Ether reward that covers their gas costs and provides additional return. A small portion of every fund is set aside in Ether, regardless of the values set in the data contract. Relative to fund sizes, this number is expected to be small.
- **Rebalancing fee**: Beyond explicit fees, rebalancing generally will have higher transaction costs than exchanges due to the gas costs of rebalancing on chain

Airdrops

Airdrops are a regular occurrence in the current ecosystem. Airdrops are carried out in many ways, but the bulk of them simply deploy a token contract with ownership of tokens to users.

The Bskt DAF has two ways for dealing with this:

- The data curator adds an entry with the airdropped token's address and the exact amount that was airdropped.
  - This allows the token to be tracked, without requiring any rebalancing actions.
  - The data curator can choose to sell off the airdropped tokens by setting the entry with an amount of 0
  - Users claim the airdropped token without needing the data curator to do anything (details pending)

Common ERC20 Scenarios

Freezing

When a token is frozen, the data curator flags it as such. The Fund treats flagged tokens as though they weren't in the Bskt, without setting the token's amount to 0 which would trigger a rebalance that fails due to the paused token.

Similar to Bskt, redeem allows a skipList to be specified so that redemption isn't blocked. Flagged tokens are added to the skipList by default. In order to prevent griefing by the data curator, we allow redeemers to specify an override skipList that doesn't consider the default ones. This prevents the curator from holding funds hostage by flagging them as frozen.

Token Upgrades

Some tokens (eg. REP) upgrade by pausing the old token contract and airdropping a new one to match token balances. This is essentially a freeze followed by an airdrop, which are covered by the previous sections.

Risks

Fund Investor Protection

If at any time a fund investor doesn't agree with a rebalance about to take place they may "vote with their wallet" by exiting the fund. They can either sell tokens on the open market or exit by redeeming.

Since not every investor can be expected to stay up to date, investors can delegate redeem authority to a sentry actor. If the sentry exits the fund to protect investors, the underlying tokens will be sent to the investors.

Upgrading

An upgrade may be required to fix newly discovered flaws — or issues with underlying ERC20 tokens.

In the *fund*, the admin can suspend continued rebalancing and creation, allowing users to redeem their tokens.

The *portfolio data manager* can also similarly mark a *portfolio data contract* closed. Initially, this is just a signal to the *fund* contract that the data contract will no longer be updated or is untrustworthy, but does not actually change the fund's behavior.

Terminology

- *Dutch auction*: An auction in which an auctioneer begins with a high asking price, and lowers it until some participant accepts the price

States:

- *Data update period*: Period during which a *portfolio data manager* can optionally change the proposed composition
- *Wait period*: Period after rebalance data updated, when anyone can request redemption from the fund to "vote" against the proposed allocation
- *Rebalancing period*: period during which rebalancing can occur