The Future of Royalties, the Creator Pool.

Author: z.ftm

Date: 17/11/2022

Contact: https://ohmki.t.me/

Table of Contents

Description of the Model

A basic Creator Pool

A Creator Pool (referred to as a CP or pool from this point forward) is a royalties model which works more as a pay check, rather than a constant stream of income. The basic idea behind this model is that royalties do not go directly to the creator, instead, royalties are sent to a pool. This pool gets emptied out in intervals, and every NFT collection's creator is paid out in proportion to what they contributed.

As an example of how this would work is:

1) Both Alice and Bob create NFT collections. One and Two, respectively.
2) One gets $100 in volume at 3% royalty, and therefore contributes $3 to the pool.
3) Two gets $100 in volume at 7% royalty, and therefore contributes $7 to the pool.
4) The pool now holds $10 inside it.
5) One contributed 30%, and Two 70%, so Alice and Bob will get paid out $3 and $7, respectively.

But why is this better than just paying out the $3 or $7 at the time of the sale? Right now, it isn't, but it is going to define a helpful equation for later on:

Let *c* be the amount contributed to the pool

Let *t* be the total amount in the pool

Let *P* be the amount the creator will be paid out

$$\frac{c}{t} \cdot t = P$$

This equation seems overcomplicated, but it is just a base that we will expand later on. For now, let's just label this equation as the bVCP – basic Vested Creator Pool equation.

Improvements

Right now, there is no reason to use this model, though there are many improvements to be made. These improvements benefit both the marketplace and the creators.

Investment fund

An investment fund is a no-brainer. Why have all that money lying around if it isn't working for anyone? This is where an investment fund comes in. The royalty funds would be invested and produce yield for the artists or creator. The gains of the investment will be the total amount in the pool minus the total amount deposited. So, we need to update the equation, let's call this one the 'iVCP (invested Vested Creator Pool) formula'.

Let *d* be the total amount deposited by all creators.

$$\frac{c}{d} \cdot d + \frac{c(t-d)}{d} = P$$

Let's just walk through what each part of the formula represents:

1) $\frac{c}{d} \cdot d$ can be simplified to just c, which means that a creator should always get what they contribute to the pool. $\frac{c}{d}$ just represents the creator's weight in the pool in a decimal/percent form.
2) $\frac{c(t-d)}{d}$ is the amount of yield the creator's royalties have accrued. We already know that $\frac{c}{d}$ is a creator's weight in the pool, so we need only multiply the total yield for the pool by that to find the creator's allocation of the yield.

Let's re-arrange that formula just for aesthetics.

$$\frac{ct}{d} = P$$

Now, in Solidity code.

```
function calculatePayout(

        uint64 contributed,

        uint64 total,

        uint64 totalDeposited

) private pure returns(uint64) {

        return (contributed * total)/totalDeposited;

}
```

Great! Now we have a way for artists to earn on their royalties. But why can't they just do this themselves? Well, this method also provides value to the NFT marketplace. If the marketplace has enough capital to invest, protocols are going to want that sweet crypto in their pools. If – in this case – Magic Eden was to use a governance proposal to decide where to put the assets, it would incentivise protocols to buy up the governance token (or whatever method is used) to direct the capital to their platform. Alternatively, the protocol in question could bribe the governance voters to get the voters to pick their protocol (not unlike what happened in the Curve wars, BEETS wars, etc.). This provides value to the creators, and the users of the NFT marketplace.

This would bring artists, developers, users and protocols to the NFT markeplace.

Caveats

The Spotify problem

The model described above is similar to that which Spotify uses. Spotify is a nightmare for smaller artists, who don't get paid the same rate per stream as big-name artists, just because their share in the 'Spotify Revenue Pool' is smaller. While this will not happen by our description, it is important that one party does not control a large portion of the pool.

The solution

This is where we introduce a 'cap' per collection. Let's say each collection can only control up to 5% of the pool at once. This means that any royalty revenue after 5% control will be considered as yield, and will be distributed evenly to all the artists in the pool. This means more changes to our equation.

This one remains the same, for collections that control under 5%.

$$\frac{c}{d} \cdot d + \frac{c(t-d)}{d} = P$$

However, this one just considers their stake in the pool as 5%.

$$0.05 \cdot d + \frac{0.05d(t-d)}{d} = P$$

Let's simplify that again.

$$0.05t = P$$

Just to look pretty, let's put it in a neat formula using conditionals.

$$0.05t \cdot \left[\frac{c}{d} \geq 0.05\right] + \frac{ct}{d} \cdot \left[\frac{c}{d} < 0.05\right] = P$$

And to express it without fractions (for fun).

$$0.05t \cdot [d^{-c} \geq 0.05] + d^{-ct}[d^{-c} < 0.05] = P$$

Let's put this in Solidity in a neat one-liner.

```
        function calculatePayout(

                uint64 contributed,

                uint64 total,

                uint64 totalDeposited

        ) private pure returns(uint64) {

                return (contributed/totalDeposited >= 0.05) : (0.05 * total) ?
((contributed * total)/totalDeposited);

        }
```

So, if someone contributes more than 5%, they just get 5% of the pool at the end of the interval. This means that if the creator wants to earn more after they obtain over 5% of the pool, they either have to wait until their contribution is under 5%, or grow the whole pool, which will allow the smaller creators to have a piece of the larger creators' revenue. Let's call this one the 'iICVCP (invested Individually Capped Vested Creator Pool) formula'.

This seems counterproductive, however. Why would any project big enough to control 5% of the pool want to continue contributing after that? Sure, it helps smaller creators, but let's say this project was in it solely for the money, they don't care about furthering the ecosystem at all. For this, the marketplace in question should establish two separate pools, one where the iVCP formula is implemented, and another where the iICVCP formula is implemented. This allows collections a choice to whether they want to contribute to the wider ecosystem, or just the marketplace (with what was described in the 'Investment fund' section.

Timing

The bCVP model works on intervals for payouts. In between intervals, creator royalties are accrued. But how can we assure the royalties are paid out on time? On-chain, timed events are tricky to pull off, and while there are solutions like Gelato, which allow for transactions to be queued and sent at specific times, most of these are exclusive to Ethereum, so we must come up with our own solution, one which is both gas-efficient, and automatic.

The solution

Currently, royalties are paid out when the buying transaction occurs, so let's do the same here. We still have to respect the interval system. So what we can do, is store the last time the pool was emptied and make sure it has been at least the interval time since then – and only then empty the pool. We can do this procedure every time a sale goes through (where royalties get sent to the pool). Let's put this in some code – a basic solidity contract.

```
contract interval {

        uint256 last_payout;

        uint256 constant interval_time = 30 days;


        function verify_payout() internal pure returns(bool) {

                return (block.timestamp >= (last_payout + interval_time));

        }


        function payout() external {

                if (verify_payout()) {

                        // implement payout logic here

                        last_payout = block.timestamp;

                }

                // do nothing if the interval time has not passed

        }

}
```

All this does is reset the `last_payout` variable to the current timestamp, but only if 30 days have passed since the last set. All we need to do now, is call the `payout()` function whenever an NFT is sold, after royalties are sent to the pool.

Gas costs

The bCVP model calculates the payout for each creator, which means that we have to send the funds individually to each creator. This costs a lot of gas. How can we change this?

The solution

There are a few possible ways to multi-send funds:

1. Iterate over all the creators in one transaction
2. Have the creators claim their royalties
3. Make a permissionless claiming system, and have the team execute the payouts for each creator
4. Make a permissionless claiming system, and crowd-fund the claims for each creator

Let’s go through the pros and cons of each, and explain how they work:

1. Iterate over all the creators in one transaction

This is easy, just store a list of all the creators, and send them their payout when you get to them in the list. This would be great – all the creators get paid instantly, with no involvement. The problem with this method is the gas cost for the user that executes the payout would be incredibly high. This seems unviable, so let’s move on.

2. Have the creators claim their funds

This strategy is a bit more difficult to execute but removes all the problems with gas. In this method, at the end of each interval, the claimable amount for each creator is stored. After that, the creator claims the funds themselves. This would be implemented best by storing the amount owed to each creator per interval, and have the creator claim the funds for a specified interval. The only downside is that if creators forget to claim their royalties, they will just remain in the pool forever. To facilitate claiming of forgotten royalties, a multicall function of sorts could be used.

3. Make a permissionless claiming system, and have the team execute the payouts for each creator

This system is much like the previous one, but the payout step is not performed by the creator, instead the team of the NFT marketplace claims the royalties in the stead of the creator.

4. Make a permissionless claiming system, and crowd-fund the claims for each creator

This system is like the previous one, but the claims for each creator are crowd-funded, and incentivised. This would be implemented in a similar way to how PancakeSwap crowd-funds autocompounds for its vaults. The contract would have to automatically select the next creator to payout, so front running claims would be ineffective.

Conclusion

The current NFT landscape relies on an outdated royalty payment method, which does not function as the second income stream most creators treat it as. Instead, we have defined a way for NFT creators to earn their royalties in a more traditional payslip method. We have also defined multiple improvements to this model, including a way for creators to earn on their royalties, while providing value for the marketplace, users of the marketplace. All the finished code for this can be found at (link).