**Ampnoob’s Protocol Idea**

**The Problem:** The basic problem that this protocol tries to solve is to incentivize liquidity in a sustainable manner that helps to guarantee a high long term TVL, thereby further facilitating a thriving defi ecosystem on the chosen chain/dex.

Until now, the primary means of incentivizing liquidity in defi has been through unsustainable inflation, whereby a governance token is continually emitted and paid to liquidity providers, who then sell the token, which causes the token to lose value. As the token devalues, the value of the rewards paid to liquidity providers diminishes and liquidity leaves. Various innovations have been invented to augment this basic model (most notably curve’s vote-escrow model), but with a couple of special exceptions, they have mostly proved to be unsustainable.

**The Idea 1 – Base Protocol and Tokenomics:** This protocol aims to provide liquidity providers with rewards that do not grind inevitably towards zero over the long run. Here (roughly) is the idea for how to make that possible. Note that the idea was inspired by the economic model of Nirvana Finance (on Solana), but it aims to solve what I think are fundamental problems with that model, whilst also augmenting it to solve the problem of incentivizing liquidity. There’s a lot to cover but I’ll start by describing the basic protocol and its tokenomics, before moving onto how we can use it to sustainably incentivise liquidity on a chosen chain/dex.

1. Create a token, call it SALMON (SAL) for now. The protocol contains two pools of USDC. The first is the `floor supporting liquidity (FSL) pool’. The second is the `price supporting liquidity (PSL) pool’.

PSL

FSL

1. The protocol has the ability both to mint and to burn SAL tokens. One way that this can happen is that a user can redeem a SAL token for a proportional share of the USDC in the FSL pool at any time. For instance, if there are 10 SAL tokens and 100 USDC in FSL, then a user can burn one SAL token in exchange for 10 USDC at any time, in which case we say that 10 USDC is the `floor price’ for SAL. The protocol is designed so that the floor price can never decrease, it can only increase.
2. As well as burning SAL for the floor price, users can also `sell’ SAL for the market price (determined by the protocol), which will always be at least as great as the floor price. In that case, they will burn their tokens in exchange for a proportional share of the FSL as well as some amount of the PSL (I outline how market price gets determined below). Finally, users can `buy’ SAL from the protocol by paying the protocol the market price, in exchange for which the protocol mints them one SAL token.
3. If we want to make sure that the floor price never decreases, then we have to ensure that users can never buy SAL for less than the floor price. If we set the market price according to the following function (for example), then we can be sure that market price will never drop below the floor price,

Market price = (FSL+PLS/Supply)\*(FSL+PSL/FSL)

I’m still playing around with possible market price functions (there are a lot of considerations), but this one works fine for illustration at least. For example, imagine if there is 10 USDC in FSL, 5 USDC in PSL and a supply of 10 SAL. Then the market price will be ((10+5)/10)\*((10+5)/10) = 2.25. If somebody sells a token, then they will receive 2.25 USDC. 1 USDC comes from FSL (users always get a proportional share of FSL when they sell) and 1.25 USDC comes from PSL. So the new FSL is 9, new PSL is 3.75, and the supply shrinks to 9. The new market price will be ((9+3.75)/9)\*((9+3.75)/9) = 2.006. If rather than selling, somebody buys a new token instead, then another token is minted (pushing the supply to 11), 1 USDC goes to FSL (which becomes 11) and 1.25 goes to PSL (which becomes 6.25). The new market price is ((11+6.25)/11)\* ((11+6.25)/11) = 2.459. Crucially, since users always deposit 1/FSL’s worth of USDC to mint a new token (and sellers never take more than 1/FSL out of the FSL), there is always enough USDC in the FSL to guarantee the floor price for every token, and the floor price can never decrease.

1. We’ve seen how market price can be determined, how users can mint and burn SAL, and how it’s possible to ensure that the floor price never decreases. But ideally, we want the floor price to actually increase over time. If you buy SAL for $1 and the floor price increases above $1, then you will always be in profit, whatever happens to the market price. I propose three mechanisms to increase the floor price.

Firstly, there is a `target ratio’ T such that when the ratio PSL/FSL hits T, we move some of the PSL over to FSL, thereby increasing the floor price. More specifically, we could move e.g. 10% of PSL as soon as the PSL/FSL ratio hits T. When this happens, market price will drop slightly (using the market price function above) but the floor price will increase permanently – we essentially lock in the gains of the token forever. Also, the PSL/FSL ratio will decrease back below T. It will probably also make sense to then raise the target ratio slightly (maybe by 10%). Now, if the PSL/FSL ratio is stuck below T for a period of time, we will gradually decrease T so that it becomes easier to reach it, thereby ensuring that the floor never stops rising. For instance, we could decrease T by n% each day, where n is the number of days since the last time we hit T. This is one of the mechanisms that I think can be improved upon significantly from Nirvana finance. Their price curve made it possible for the target ratio to stop decreasing – right now, the price has been stuck below the target ratio for weeks and there is no sign of it reaching it again in the short term, which means that there is no floor increase and the protocol stagnates. We should ensure that the target ratio never stops decreasing when we’re below it. Constant floor increases are the rocket fuel for the protocol.

Secondly, I like the idea of adding in a sell tax of around 5% that gets sent to the FSL with every sale, thereby raising the floor. This will ensure that in drastic sell offs, as the market price dips, the floor price is constantly increasing. As the market price approaches the floor price, the risk/reward ratio of buying becomes infinitely positive (at floor price, there is zero downside and uncapped upside). It also helps to ensure that people who make money from the protocol are never able to simply extract 100% of the value, they have to leave some behind, which I like. The sales tax will also synergize with the strategy for incentivizing liquidity.

There is also a third potential strategy for encouraging a rising floor price, but I’ll leave that until later.

1. This is where things start to get really cool. The protocol will also have a staking mechanism. But the rewards are not more SAL tokens that are minted out of thin air (that would dilute the floor price). Rather, they are in a second token, call it `Honey’ (HON) for now. What HON does is give you the right to buy SAL from the protocol *at floor price*. So no matter what SAL’s market price is, if you have one HON token, you can use it to mint one SAL token at the floor price. Thus, the intrinsic value of HON is the difference between SAL’s floor price and its market price. The bigger the gap, the greater the value of HON. Again, the key thing is that we can give these staking rewards without diluting the floor price of SAL, which can never decrease. If we followed the Nirvana model, we’d pay SAL stakers 0.005 HON tokens a day for every SAL token they have staked (0.5% of their SAL bag a day, about 180% apr).
2. When you stake your SAL, you can also borrow up to the full floor price of your staked SAL. So if you have 100 SAL staked and the backing per SAL token is 1 USDC, then you can borrow 100 USDC from the FSL. Your staked SAL is then locked and can’t be withdrawn or sold until the loan is fully repaid, but you will still receive your staking rewards. This means that when the market price is low, you can buy SAL and immediately recoup nearly all of your investment as a loan against your staked SAL, and then receive the daily apr. For instance, if the floor price is 1 USDC and the market price is 1.1 USDC, you can pay 1.1 USDC to buy a SAL token, stake it, and borrow 1 USDC against it to do what you like with. The borrowed USDC are still counted as part of the FSL for the purpose of calculating market price etc because they need to be repaid before the tokens that they are backing can be sold. Borrowing doesn’t dilute the floor price in any way.

Crucially, there is zero risk of liquidation with these loans. Users can borrow the floor price of their tokens, no more and no less. Since floor price can never decrease, there is never any need to liquidate users (compared to standard lending protocols where borrowing limits are determined by market price) Also, as the floor price increases, the amount that users can borrow is constantly increasing. This is important, because it gives users access to more funds to buy dips, use HON to mint SAL etc. Also, you can expect a lot of users to loop leverage to increase their bag size by a lot.

As well as helping to make SAL an extremely good buy as soon as it nears the floor price, the borrowing mechanic has some other benefits. Specifically, if we follow Nirvana’s model, there’s a 3% loan origination fee that goes to the team/protocol. This helps to ensure that the team can get paid without needing to be given any free tokens that they dump on the investors down the line.

1. So those are the basic tokenomics. Constantly rising minimum price floor, high apr that doesn’t dilute floor price, sell tax to boost price floor, liquidation free loans etc. Now, everything I just described assumes that the base asset for the token (the token held in the FSL and PSL) is USDC (in Nirvana, it’s a mixture of stables). But we can run the whole thing again with different base assets, such as BTC, ETH, maybe even DRAKMA! In fact, I really like the idea of having at least three of these tokens, corresponding to USDC, BTC, ETH. As we’ll see in a bit, I think doing that can be a really useful tool in the liquidity incentivising strategy .

**The Idea 2 – Liquidity Incentives:** So suppose that we have these three tokens running smoothly and everything goes as planned. How does that help bring liquidity to a chain/dex? Here’s how.

1. Create a yield farm for the chosen dex. Call our three tokens SAL1, SAL2 and SAL3 for now (with corresponding reward tokens HON1, HON2, HON3). Liquidity providers are given high yields, paid in HON1, HON2 and HON3. We’d have to think about this more, but I imagine that for each of these reward tokens, the total emissions going to the yield farms should be about half of the emissions going to the stakers for the corresponding SAL token. It may be that we need to make the emissions rate dynamic and sensitive to the gap between market and floor price in order to keep yields within an attractive range that doesn’t suppress SAL market price too hard. Crucially, because we are paying HON and not SAL to liquidity providers, these emissions do nothing to dilute or damage the floor price of the corresponding SAL tokens. And it always remains true that buying SAL tokens close to floor is an extremely good bet because you can immediately recoup your investment with a loan and then claim yield on your locked SAL.
2. Because there are four separate tokens, the price burden of supporting the emissions is shared 4 ways. Furhtermore, unlike most yield farm reward tokens, these tokens can never go to zero. They are backed by a monotonically increasing floor price that never goes down and constantly goes up. This is why this way of incentivizing liquidity is so different from and more sustainable than existing methods. Previous reward tokens lacked a mechanism to prevent them from going to zero. This has such a method.
3. There’s no doubt that using HON as a farming reward token will produce selling pressure on SAL and push down its market price. But all that matters for SAL is that the floor price keeps increasing, and it always will. In fact, the system is kind of designed to prevent the market price from ever going too far above the floor price, which makes sure that nobody ever gets rekt by buying in way above floor before a crash. Furthermore, the selling pressure from rewards will drive more taxes constantly towards the FSL, which is why I think the sell tax synergises well with the liquidity incentives. And again, paying the emissions in three or four tokens ensures that the sell pressure on any one is mitigated.
4. In the end then, we have at least three tokens, each with a corresponding reward token, that have rising price floors, high staking apr’s, and liquidation free loans, and the reward tokens are used both to pay stakers and to incentivize liquidity on a chosen dex/chain in a sustainable way. There is a lot of variation possible in the basic design, depending on exactly how we set the apr’s, the market price function, the sell tax etc, and it all needs to be thought through carefully.
5. We can also think about extra mechanics to relieve sell pressure on SAL, like e.g. locking for extended periods of time in order to get higher apr’s etc.
6. One nice thing about the protocol from an engineering perspective is that it doesn’t (to my untrained eye) require us to use any oracles or mess with actual liquidity pools containing multiple tokens etc.
7. Of course, if we wanted to, we could make our own uniswap fork and incentivize liquidity on that, and then use the trading fees to help push up the price floor. Or we could introduce a bribing system so that different dex’s and protocols pay bribes to direct emissions to their dex’s/trading pairs, and those bribes could be used to increase the floor price.

This is already way too long – hope I haven’t bored you – excited to hear your thought.

Ampnoob