Polyhamster: A Deflationary Reflection Token with Automated Liquidity Acquisition on Polygon Network

hello@Polyhamster.com
http://www.polyhamster.com

Abstract

Polyhamster Protocol aims to solve the problems of prior cryptocurrencies including mining rewards, farming rewards, and liquidity provisioning. Mining equipment can be both costly and harmful to the environment, but mining remains of interest due to the opportunities afforded by it. As an easy alternative to mining rewards, we propose allowing users to participate in a smart contract token reflection to produce tokens inside their own wallet. Another challenge remains to facilitate and maintain liquidity on decentralized exchanges. By nature, decentralized exchanges require liquidity for user participation, thus the responsibility is on the developers to provide it. Historically, developers created incentives aimed at users to provide liquidity which can be outweighed by risk due to the subjectivity of impermanent loss. As a solution, we propose utilizing a smart contract function to automatically capture liquidity to be used on the decentralized exchanges and held in custody independent from user possession. Additionally, a smart contract that provides the capability to burn tokens can promote scarcity by reducing the total supply. Together, the combination of these tokenomics may afford far superior benefits for the community within the decentralized venue. Allowing these functions to be amplified and dependent on volume provides an ideal incentive to expedite adoption and foster new use cases.

1. Introduction

Decentralized finance is made possible by using decentralized exchanges in collaboration with liquidity pool smart
contracts. For any token on the smart chain to have an availability to be swapped on a decentralized exchange, it must have an available liquidity pool of tokens for swapping. The challenge remains on how to properly incentivize users to keep such liquidity pools maintained. Recognizing this, developers have attempted to satisfy these conditions by using various tokenomic structures with incentives for the user to supply liquidity into the pools. An automatic liquidity acquisition can be featured as an alternative solution compared against the traditional "farming reward" structure. An automatic liquidity acquisition function where users are offered rewards (via reflection) in lieu of traditional farming rewards. These reflections would act to distribute tokens proportional to volume, and could thus provide a more reasonable incentive for holding. Although reflection and automatic liquidity acquisition may contribute to stability, an inherent burn which can achieve token scarcity with a depreciating token supply. The combination of these tokenomics seeks to eliminate the flaws of various predecessors, while providing useful incentives for use case and adoption. Effectively, any application that is added with these smart contract functions could have the effect of amplifying Polyhamster's tokenomics.

2. Automated Liquidity Acquisition

We understand that liquidity is crucial in any trading environment. By definition, decentralized liquidity is simply the
accessibility of tokens operated and controlled by a smart contract--hosted by a decentralized exchange. Historically, market makers have been used to provide a service for buyers and sellers on traditional order book exchanges for a better user experience. The main function of these market maker services was to fill buy and sell orders promptly and reduce overall market volatility caused by large orders. However, traditional order books have long been outdated by newer technology, and have been replaced by liquidity pools in a decentralized venue. Just as market makers are compensated for providing a service in the order book environment, proper incentives for adding liquidity are a key factor in any decentralized environment. Problems arise when the liquidity pool provider loses the incentive to add tokens into the pool, which occurs after the token pair is

subjected to impermanent loss resulting from arbitrage.
As a solution, Liquidity can be taken as a function of the smart contract using market activity from all swaps and transfers. A portion of these swaps and transfers will be captured by the smart contract and utilized with the function:
"*_swapAndLiquify*". For this to happen, the portion of the 3% fee from swap and transfers can be kept in a standalone pool within the contract itself and automatically converted to the liquidity pool after the token count reaches a threshold, set at 10 million tokens. Liquidity is then managed by the contract as it is sold and paired accordingly thereby alleviating the users from having to subject themselves to any impermanent loss scenarios. Large liquidity pools act to decrease the volatility of the swap impacts against the overall available supply. Therefore, as the token matures, the auto-liquidity can be attributed toward an ever growing market stability capable of absorbing large market activity.

3. Token Reflection

Traditional mining is both costly and inconvenient for the user. Frictionless, static reflection rewards accrue by simply holding your tokens, and features an innovative hold-farming reward structure that stands out from conventional pool-farming rewards. The idea behind this function is to eliminate token dependencies that have created problems in the past, including, but not limited to:

a. Pooling funds in unverified 3rd party smart contracts.
b. External website interfaces.
c. Transaction fees needed to claim rewards.

Earlier models of decentralized finance tokens such as pool farming are costly and rely on user action to manually
compound rewards. As a solution, we propose the utilization of a compounding reward structure that requires no additional fees in a smart contract function, also known as token reflections. To achieve this, reflection must happen without cost or impact to the user. Considering the static rate of reflection set at 5%, the volume of market activity will directly impact the quantity of token reflection based upon the percentage of tokens held by the user relative to the overall supply. With the "*_excludeFromReward*" function enabled for individual addresses, accounts such as exchanges, hot wallets, dapps, etc. can be excluded from token reflection, thus granting more rewards to individual holders.

4. Depreciating Supply & Burn Address

In a decentralized smart chain environment, contract functions can be utilized to achieve token scarcity. To do this,
we propose also distributing rewards to the burn address, which is publicly verifiable for all participants to see. We can then track the depreciating supply in real-time for added transparency. In our effort to establish a baseline token burn rate, we find that these values are dependent on three important factors: reflection rate, token quantity, and market volume. The rate of reflection rewards is proportional to the total supply in each holder's wallet address. It is important to note that there are two particular variables which will affect our calculations: the increasing scarcity of tokens and the quantity of tokens absorbed into the burn address. It can be reasonably understood that these features will have synergistic effects that can stabilize the burn rate into the future.