Gambleswap

Gambleswap Team

March 2022

Abstract

Gambleswap is a gambling powered Constant Product Automated Market Making protocol. Liquidity providers will be given $GMB as a form of passive income and may enter into a gambling game to bet with their $GMBs. Liquidity providers also can lend their LP tokens to those who do not have enough LP tokens for participating in the gambling game, thus, unlocking leveraging the value locked in liquidity pools even more. The chance of winning in the game is proportional to the $GMBs each user has put in the game. Also, it's designed to have the ability to control the total supply and price of $GMB.

1 Decentralized Exchange

As a fork of Uniswap, Gambleswap is, in its core, a decentralized liquidity pool based exchange. Liquidity pools contain a huge amounts of two different assets like $aToken$ and $bToken$. Users can use these pools to swap their $aTokens$ with $bTokens$ or vice versa. The ratio between the amount of token a user deposits and the amount of the token they take out is calculated through an Automated Market Maker (AMM) which will be discussed more later (For more details, see Appendix 4). In Gambleswap, in addition to LP tokens, they also receive $GMB. These tokens can be used to participate in the gambling and later moderating the Gambleswap ecosystem.

2 Tokenomics

Gambleswap has introduced GMB as the main token for particpation to empower its underlying automated market maker. LP token holders will receive $GMB as rewards. Later, they can use them to participate in the gambling.

2.1 Mint GMB

A fixed number of $GMB is minted in each block. Each liquidity provider will receive a fraction of these tokens which is proportional to the amount of their LP tokens. In order to get these $GMBs, the user needs to claim them. When a user u claims that they should receive GMB tokens, the contract calculates the effective number of LP tokens that the user has and then find out their available number of GMB tokens to be minted. Effective LP tokens includes the amount of LP tokens that user holds at the time of calling claim, plus what he already has locked in the gambling game of the last recent 10 rounds, plus what he has lent:

$$\begin{gathered}\text{number of GMB tokens } u \text{ gets in a claim} = \\ \text{GMB to mint per LP token} \\ \times \text{ effective number of LP tokens for the user} u \\ - \text{ number of all the GMB tokens that } u \text{ minted so far}\end{gathered}$$

To make it easier for the users to receive their \$GMB, the pool contract automatically claims these tokens for them whenever they update their LP token value, i.e. when they deposit liquidity to or withdraw liquidity from the pool.

2.2 Burn GMB

To be able to control the supply of \$GMB, the gambling game designed to burns some of the minted tokens on a regularly basis. We will discuss it in more details in the following sections.

3 Gambling Game

The gambling game is the powerful core of the enhancements added to Gambleswap's AMM logic. The rules of the game can be basically listed as follows:

- Participants send their GMB tokens into the gambling contract. These tokens are accumulated to form our jackpot. They also send their bet values, which is a number they have guessed within a particular interval.
- Participants need to either own some LP tokens of the certain set of liquidity pools (called authorised pools) or borrow them from the LP lending pool to be able to play the game. Their LP tokens are locked during the game, and afterward, they can unlock them regardless of whether they win or lose. This rule intends to motivate users to provide liquidity in our pools.
- A random number is generated using some properties of the block like block number, block difficulty, and block timestamp.
- If the number a user has bet on falls inside a certain distance from the randomly generated number, that user is one of the winners of the game. This distance gets larger depending on the amount of GMB tokens they have deposited into the jackpot.

- 25% of the jackpot of each round, and all the accumulated GMBs after every fourth game will be burnt. This is done in order to control the GMB total supply.
- If a game doesn't have any winners, the jackpot's contents remain in the jackpot and will be used in the next round. This motivates users to participate in the game if they see there is a non-empty jackpot before they start the game. Therefore, the number of participants can grow substantially.
- The maximum chance of winning is 50%. It means if a player owns all the $GMBs in the jackpot, they have a maximum 50% chance of winning.

3.1 Determining winners

We introduce a variable as *coveragePerGMB*. This value demonstrates the length of the interval that each GMB token covers. Participants can increase their coverage by having more GMBs in the game. User's coverage shows the maximum distance the user's bet value can have with the correct random number for the user to win. The greater this value, the more chance the user has for winning. This value is calculated as follows.

$$\begin{aligned}\text{user's coverage} = min(&\frac{1}{4} \times \text{maximum random number},\\ &coveragePerGMB \times \text{user's GMB tokens})\end{aligned}$$

the $\frac{1}{4} \times$ maximum random number is there to bound the maximum chance of winning to 50 percent.

Users have more chance to win by putting more GMB in the jackpot; each user wins if their bet value falls inside this interval:

$$(\text{correct random number} - \text{user's coverage},$$

$$\text{correct random number} + \text{user's coverage})$$

The *coveragePerGMB* value in each round is calculated based on the value of the jackpot in the previous game. For the first game, we assume the coveragePerGMB value is 10, and for games after that, we have:

$$coveragePerGMB = \frac{\text{maximum random number}}{4 \times \text{total jackpot of the previous game}}$$

In the other words the coveragePerGMB is a dynamic value depending on the previous game. In each game, this value is the same for all users. If the number of participants in one game is small, the chance of winning in the next game goes higher; Consequently, more users are encouraged to participate in the next round. Figure 1 illustrates the process of determining the winners.

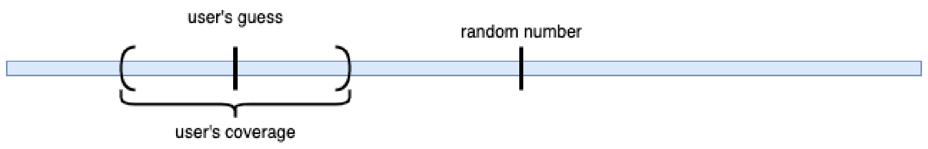

Figure 1: The blue interval shows the entire interval the random number can fall into. If the random number falls into the user's coverage, user will be one of the winners.

3.2 LP Lending

Gambleswap introduces users a LP lending protocol to participate in gambling game without the need to provide liquidity. There are two user's types in this lending protocol:

- Lenders: Users who have LPs can lend their tokens from different pools; In return, after each round of the game, they will be given interest per borrow in form of GMB tokens.
- Borrowers: Users who do not have any LP tokens and can participate in each round of the game by borrowing lenders LP tokens. In return, they have to pay the interest to the lenders.

Each round of the game has its own capacity on the lending contract, and after each round, the capacity will be released for the upcoming round of the game. Every authorised liquidity pool has a separate lending pool and interest per borrow. Whenever a user attempts to borrow LP token, the lending contract finds the cheapest available pool which has enough capacity, and lends the needed amount of LP token to the user. Lenders share the accumulated interest of their corresponding pool proportional to their share of the lending pool.

3.3 Overflow/Underflow in the interval

The generated interval for each user may exceed the $MaximumRandomNumber$ or may become less than $zero$. There are three scenarios for an interval:

- Overflow: It happens when

  $CorrectRandomNumber + UserCoverage > MaximumRandomNumber$

  In this scenario, the remaining amount of the user's coverage will be considered from 0 to the OverflowValue.

$$\begin{aligned}\text{OverflowValue} = {} & \text{UserCoverage} \\ & + \text{CorrectRandomNumber} \\ & - \text{MaxRandomNumber}\end{aligned}$$

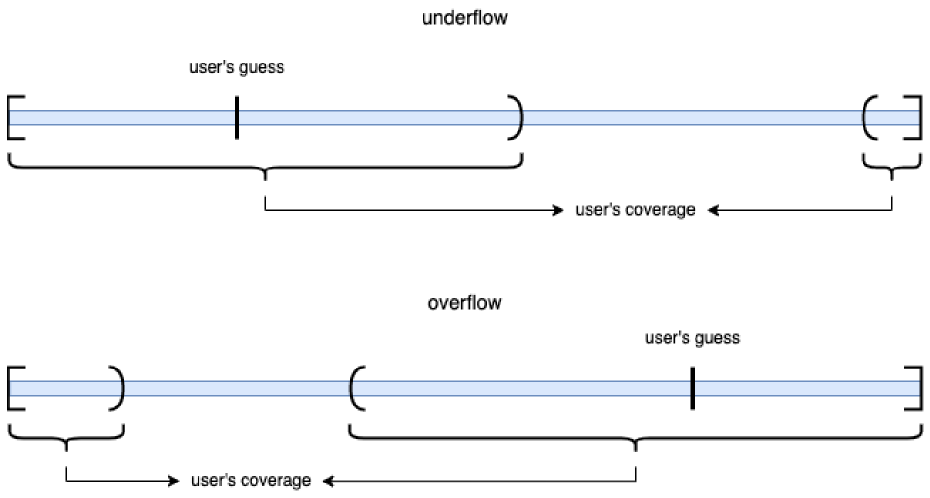

Figure 2: User's coverage in underflow and overflow cases.

Users will win when the below condition holds:

$$(\text{betValue} > (\text{correctRandomNumber} - \text{UserCoverage})$$

$$\vee$$

$$\text{betValue} < \text{overflowValue}$$

- Underflow: This happens when

  $$UnderflowValue = UserInterval - CorrectRandomNumber$$

  Users will win when the below condition holds:

  $$(\text{betValue} > (\text{maxRandomNumber} - \text{UnderflowValue}))$$

  $$\vee$$

  $$(\text{betValue} < (\text{CorrectRandomNumber} + \text{UserCoverage}))$$

  Figure 2 shows the coverage of the user in cases of underflow and overflow.

- Normal: In this scenario users will win when their $betValue$ is in the below interval:

  $$CorrectRandomNumber \pm UserInterval$$

3.4 Expected value of net profit

Let's assume user u_i deposits a_i GBM tokens in the gambling contract. The random number can be between zero and the maximum. Let's assume the maximum is M and the number of participants is n and number of winners is W. Also, $\frac{1}{4}$ of the tokens are burnt. Therefore, the probability of u_i being one of the winners is:

$$p_i = a_i \times \frac{1}{4 \times (\sum_{i=0}^{n} a_i)_{lastRound}}$$

Hence, the expected value of u_i's profit in one round of gambling is:

$$\begin{aligned} E[profit(u_i)] &= p_i \times \frac{3\sum_{i=0}^{n} a_i}{4W} \\ &= a_i \times \frac{1}{4 \times (\sum_{i=0}^{n} a_i)_{lastRound}} \times \frac{3\sum_{i=0}^{n} a_i}{4W} \\ &= \frac{a_i}{16W} \times \frac{3\sum_{i=0}^{n} a_i}{(\sum_{i=0}^{n} a_i)_{lastRound}} \end{aligned}$$

As we see, when the number of players increases in one round in respect to the last round, the probability of winning increases. If we assume the $\frac{\sum_{i=0}^{n} a_i}{(\sum_{i=0}^{n} a_i)_{lastRound}}$ factor to be close to one, we can see that

$$E[profit(u_i)] \approx \frac{3a_i}{16W}$$

So, the expected net profit of the the u_i is approximately $\frac{3a_i}{16W} - a_i$.

4 Conclusion

Gambleswap seeks to unlock the hundreds of millions of dollars in value locked in a decentralized exchange. Up until now, the community has shown great enthusiasm toward gambling and lottery games. The economic dynamics behind Gambleswap's gambling game also incentivizes people to provide even more liquidity.

A Automated Market Maker

As mentioned before there are two tokens in each liquidity pool in an AMM-based exchange. The ratio between the amount of one token a user takes out and the amount of the other token the user endows, are calculated using a constant product equation. Assume the amounts of the tokens are R_a and R_b before the swap. A user can deposit x tokens of R_a in the pool and take y tokens of R_b out. A small portion of x is taken as a fee. This fee is divided between all liquidity providers proportional to their share in the pool. The fee ratio is shown by μ. Now, because the product of the two tokens amounts should remain equal to k, we can calculate y like this:

$$\begin{aligned} R_a R_b &= (R_a + (1-\mu)x)(R_b - y) \\ \rightarrow (1-\mu)x R_b &= (1-\mu)xy + yR_1 \\ \rightarrow y &= R_b - \frac{R_a R_b}{R_a + (1-\mu)x} \end{aligned}$$