TRADING BOT **DEVELOPMENT**

ligne courteligne courteligne courteligne courteligne courte

# 

# **Table Content**

1. **Introduction**

1. **How bot works?**
2. **Profit Analysis**

## Introduction\_flashArbitrageContract

Building a trading bot can be an exciting and rewarding endeavor, enabling everyone to execute trades with precision and efficiency.

Especially triangle bots are widely used in tokenomics and their validity and profitability gradually become recognized by more and more traders.

But in the testnet stage, as there aren’t much more pairs existing in multiple Dexs for triangle bots to make profits and show its efficiency and also transaction frequency is lower than the mainnet, we’d prefer to taking flash bots to show eye catching profits and prove its efficiency.

In essence, flash bots (flash loan bots) work in the way borrowing tokens from the pool in one Dex and swap that tokens in the same pool ( by the same, I mean the pool consists of the same token pairs) in another Dex.

The reason we take these bots is that it is suitable for the testnet case where transaction frequency is lower and there are fewer pools available in.

After testnet stage, we should extend the way that the flash loan bots work and investigate how to make triangle bots as profitable as the flash ones.

## How bot works ?

We have deployed flash bot on BNB chain testnet.

First Let’s explain about the technical analysis of our flash bot.

Our flash transaction starts when there occurs swap transaction between tokens in a pool of WBNB and USDC in Pancakeswap and Sushiswap Dexs.

Flash bot periodically monitors the status of the pool by absorbing reserve values of each token in the pool for each Dex of pancakeswap and sushiswap.

If the proportion of one token to another is lower or higher than the standard ones, then the bot executes the transaction.

Monitoring the status of the pool and deciding whether to execute transaction or not is in charge of checkTrading function of flash contract.

At that time flash bots determine the amount of the tokens that they are going to borrow from pool in one Dex by using following algorithm.

* r = optimal amount in
* x\_a = reserve out of AMM A
* y\_a = reserve in of AMM A
* x\_b = reserve in of AMM B
* y\_b = reserve out of AMM B
* f = fee (0.03%)

k = (1-f)\*x\_b + (1-f)\*\*2\*x\_a

a = k\*\*2

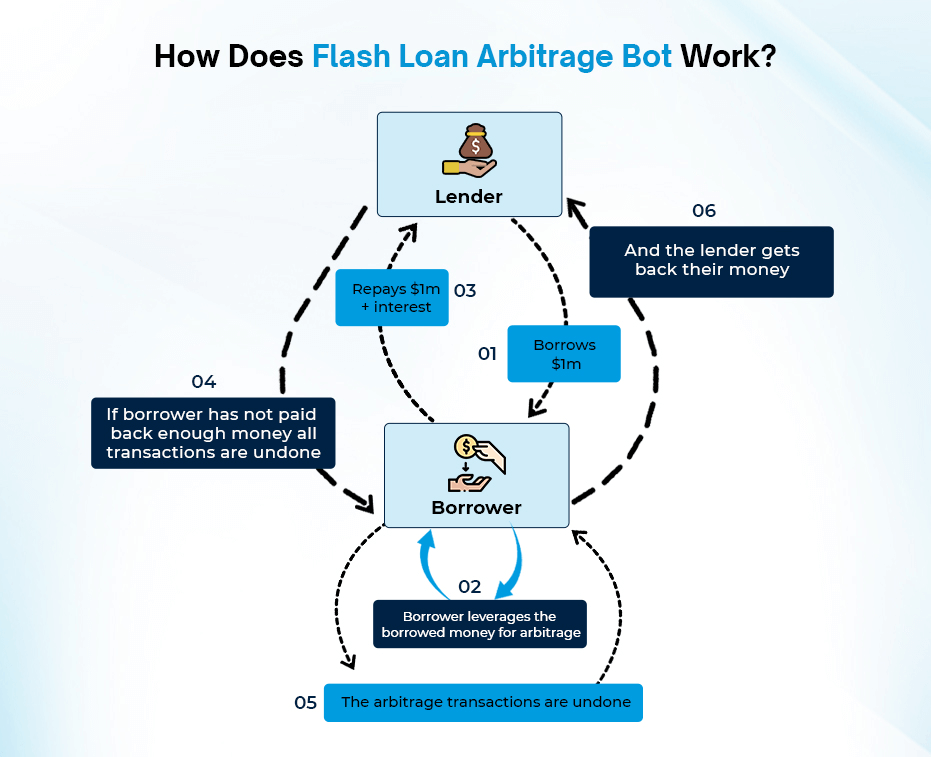
b = 2\*k\*y\_a\*x\_b

c = (y\_a\*x\_b)\*\*2 - (1-f)\*\*2\*x\_a\*y\_b\*y\_a\*x\_b

r = (-b + sqrt(b\*\*2 - 4\*a\*c)) / (2\*a) - the opitmal amout.

After calculating the optimal amount the bots start to execute transaction.

Transaction occurs in the way as described in the following diagram.



Here Borrower is the flash bot and he first borrow optimal amount, with which transaction can be profitable, from lender which in this case is the pool from certain Dex.

Borrowerr , i.e, the bot then swap borrowed amount token in the pool in the opposite Dex and get another token corresponding to the amount in the pool of the other Dex.

If the transaction meets unpredicted error, all the minor transactions will be rolled up so the flash bot never affect the pool status in Dexs.

## Profit Analysis

And Now Let’s dig into the transaction and calculate the profit and loss carefully.

First Let’s navigate to the following url which is the deployed flash bot contract address in bnb testnet.

<https://testnet.bscscan.com/address/0x06962B5c89dE26b0E7C3614D29BC25F966D5f0A9>

This first transaction’s hash is 0x6030195005d06e5a8af4f11900bd730a76d0a10c4c2c1370f7ff5622c7e95b4e and you can check transaction in the explorer by navigating following url.

Here are flash transactions and I am going to explain about the Profit and Loss of those transactions.

<https://testnet.bscscan.com/tx/0x6030195005d06e5a8af4f11900bd730a76d0a10c4c2c1370f7ff5622c7e95b4e>

<https://testnet.bscscan.com/tx/0xd46190303b5fceeba4f614a55394b19ac60fcfd0d43a0b661884e5f21c7ddbba>

<https://testnet.bscscan.com/tx/0x766aaac4c71eb984cc95706d2b0ee3b071082e3efaf0b55a79c8340d656ffb41>

The following is the main stages of that transaction.



As you can see, our flash transaction bundles 5 minor transactions and each minor one has its own structure as a transaction itself.

Let me explain about the individual address one by one.

- 0xBBbEA32C9f6400d36ABA0804F907bd860BA8a96f

This is WBNB contract address on bnb chain testnet

- 0x06962B5c89dE26b0E7C3614D29BC25F966D5f0A9

This is the flash bot contract address deployed on the bnb testnet.

-  [0xF7741aec412a17b5794EE6F16fF9224D1016b9EF](https://testnet.bscscan.com/address/0xf7741aec412a17b5794ee6f16ff9224d1016b9ef)

This is the bnb testnet USDC token contract address.

So our flash bot first borrow 0.007096584733196246 WBNB from WBNB/USDC pool in Pancakeswap.

And next flash contract send borrowed amount of WBNB to the WBNB/USDC contract in Sushiswap and get the 0.044393878800473023 USDC.

Flash bot contract send amount of the equivalent price of borrowed WBNB to WBNB/USDC pool in Pancakeswap contract, that is 0.04277492004541904 but it has remaining 0.001618958755053983 WBNB.

Profit - 0.001618958755053983 WBNB.

Transaction Fee - 001125355 WBNB.

And transaction fee of this flash transaction cost 0.001125355 WBNB so we get  
 0.001618958755053983 - 0.001125355 = **0.0005 WBNB** as a reward of our flash transaction.