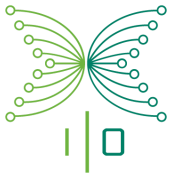
Backend - Transaction Fees structure



Introduction

This document gives a basic idea of the fee management for the IOHK Backend System. The reader should know the structure of that system in order to follow the explanation.

BackEnd Transactions Types

There are essentially three Transactions Types in the app:

1. **Holding**: Moves funds from one Holding Wallet address to one or more Invoice Wallets Addresses (could be either to the same Wallet or different ones)
2. **Invoice Split**: Moves funds from one invoice address to the Exodus address and the corresponding Commission address
3. **Commission Payout**: Moves funds from one Commission address to Distributors addresses (up to four).

Each of these movements requires bitcoin transaction fees to be paid, and as each order Balance must close to zero, it's important to handle them carefully.

Fees Calculation

Currently Bitgo Service is used estimate the "*Fees per kilobyte*", using the method described in Bitgo documentation[1].
And then, the Tx Kbs are calculated using the following formula:

```
txKb = inputsAmount * 148 + outputsAmount * 34 + 10 + inputsAmount
```

Based on this[2] Stackexchange post.

[1] https://www.bitgo.com/api/#estimate-transaction-fees
[2] http://bitcoin.stackexchange.com/questions/1195/how-to-calculate-transaction-size-before-sending

Fees Application

Each Transaction Type takes the fees amount from different places, let's review each one:

1) Holding Transaction Fee

As Holding movements are supposed to be spaced over time, meaning that by the time a transaction is performed the previous one is completed and confirmed, the risks[3] of using a fixed “fee address” to takes fees from, and using that same address as “change address” (to be able to reused it) should be mitigated; by the “atomic” and isolated nature of these transactions.

There is a fixed holding address (in Wallet chain = 1, internal) with some funds, and this address unspents are appended to the Transactions Inputs. Where in position one, you have the Holding Address with the amount you want to move. This same address is set as the change address, so when the transactions is completed, the change amount will be put back in the fee address for next transaction.

Example:

Fees: 0.001 BTC

Input:

Holding Address X	50 BTC
Holding Fee Address	1 BTC

Output:

Invoice Address 0	Sum 50 BTC in Total
...	
Invoice Address N	
Change Address	0.999 BTC

[3] http://bitcoin.stackexchange.com/questions/40670/spending-unconfirmed-output-from-a-change-address

2) **Invoice Split Fee**

This movement splits an Invoice Address **totals** into an exodus address (80%) and a particular commission address (20%) associated with it. The fees are taken from the commission portion.

Example:

Fees: 0.001 BTC

Input:

Invoice Address X	10 BTC

Only one Invoice Address, all funds are split

Output:

Exodus Address	8 BTC
Commission Address X	1.999 BTC

Note that there is no Change Address, cause the Transaction closes to zero

```
Invoice Address Amount = Exodus Address Amount + Commission Address Amount + Tx Fee
```

3) **Commission Payout Fee**

This Transaction spreads a Commission Address amount **total** into some (one to four) distributor's address/es. The fees paid for these transactions, as well as the Split Tx's one, are distributed evenly for each distributor.

Example:

Imagine this ideal scenario; Invoice receives 10 BTC and the Commission must be paid equally amongst two distributors. In an ideal scenario with no fees, the exodus should receive 8 BTC, Commision 2 BTC and then Distributor 1 BTC each. In reality, each Tx costs a fee, so that the operations would be:

Invoice Receives

Invoice X	10 BTC

Invoice Splits

Fee = 0.001

Exodus	8 BTC
Commission X	1.999 BTC

Commission Payout

Fee = 0.003

Distrib 1	0.998 BTC
Distrib 2	0.998 BTC

Where 0.998 cames from

```
Dist Total - (Split Fee + Payout Fee ) / #Distributors =
1 - ( 0.001 + 0.003) / 2 = 0.998
```

In this case `(Split Fee + Payout Fee ) / #Distributors` is a decimal number, it would be floored, and the last distributor would compensate the extra reminder. For example, if we had 4001 satoshis of fees to be distributed between two address, the first one would take 2000 satoshis and the last one the remaining 2001.

Notice that these Transactions are balanced, all inputs are spend completely, meaning no change address is needed.

$$\text{Commission X Amount} = \sum_i (\text{Distrib}_i \text{ Address Amount}) + \text{Tx Fee}$$