When transferring someone else’s notes, it seems likely that, in order to prevent contention, the owner may want to break that note into smaller ones. This way, different users interacting with the owner’s allowed balance won’t step on each others’ toes, as each of them can operate on a different utxo

note.

However, breaking a note into many smaller ones is an expensive operation: destroying a 1000-value note to create 1000 notes with value 1 will require 1000 commitments added to the tree. But we don’t need to do that.

We can add a new break_notes

function that consumes a note and “creates” many smaller notes. These notes get stored in the owner’s PXE, but they are not broadcasted and not added as commitments

to the data tree. Instead, the function computes a merkle tree out of all the small notes, and stores the root as a commitment in the data tree.

This requires making a change to get_notes

though. When retrieving a note, the oracle will need to provide either the merkle membership proof of the note in the data tree (for “standard” notes) or will need to additionally

provide the merkle membership proof of the note to its corresponding leaf in the data tree (for “small” notes). This means the PXE needs to be aware of this optimization, which is not nice, but may be useful for other use cases.