This is an edited copy/paste from omisego/research.

It seems like it'll be necessary for the user that submits a mass exit to attach a summation of the value of all referenced UTXOs. For example, if exiting UTXOs worth (10 ETH, 20 ETH, 15 ETH), then the user will also attach the value "45 ETH" in some way. When the mass exit processes, this sum will be "reserved" for mass exit to be processed on a per-UTXO basis later. This is necessary so that invalid exits that process after 2 weeks can't steal money while the mass exit is still being processed.

Unfortunately, it isn't easy to prove that this summation is valid. The above user might attach "50 ETH", which would obviously be invalid. We don't want users to be able to steal money in this manner.

One possible solution to this problem is a sum Merkle tree of sorts. Each leaf node in the tree would contain the tuple (utxo_value

, total_sum

), where total_sum

represents the sum of all leaf nodes to the left of this node, inclusive of the node itself. For example, if the UTXO values are 10 ETH, 20 ETH, 15 ETH, then the leaf nodes would be (10, 10), (20, 30), (15, 45).

These leaves would be Merklized and the final sum + tree root would be published along with the mass exit. The tree could be challenged in a TrueBit-esque game where two users iterate down the tree until they find the first leaf node at which they disagree. They reveal this node as well as the leaf node to the left of this node. The root chain makes a calculation to determine which party is correct (left_total_sum

- right_utxo_value

= right_total_sum

).

Note that this requires log(n) transactions to the root-chain in the worst case, which is not ideal. This may be way too many back-and-forth responses for the average user, although we might not consider mass-exit submitters to be average users (?). We could trivially prove fraud in 1 O(n)-sized transaction by revealing the entire UTXO set, but this is almost definitely too big. It may be possible to construct more concise proofs, but I haven't figured out anything better (yet).

I'd love to know if research on this topic has been done before.