TL DR; We've made a demo snark for Plasma Cash and Cashflow history compaction, that reduces a proof size from ~700 bytes

per block to ~200 + 8*N

bytes for N

blocks

Details

The SNARK itself is located here. In it's core it proves that a subtree in a block's SMT is empty for each root hash in the public inputs. Peddersen hash is used as a tree hash, with leaf values being 256 bit strings.

Inputs:

- Start of the continuous non-inclusion range
- Length of the continuous range. Currently there is no internal check that start of the interval is divisible by the length of the range, but it's trivial to extend
- Set of N root hashes for the block's SMTs for which non-inclusion is proved. Those are not compacted using the hash to reduce the number of inputs because a verification of such SNARK will be done off-chain only

Workflow:

- Alice accumulated N non-inclusion proofs from operator and prepares a SNARK proof as mentioned above
- Upon the transfer Alice sends a proof along with a list of block number for which this proof is valid to Bob
- Bob queries the chain to obtain root hashes for these blocks and use them as public inputs in verification
- Bob accepts a non-inclusion proof for this range of block numbers if SNARK proof is valid

Some numbers

- 4_270_718

constraints for 128

block of non-inclusion for 24

tree depth

- provable in ~30 seconds

on my laptop with six core i9

- Proof size is 192 bytes

(Groth16 for BN256) + N*8

bytes to encode block numbers to send to Bob (uint64 for block numbers) + 8 bytes

for a start of the slice being proved + 8 bytes

for a length of the slice

Trade-offs

- Will require to use Peddersen hash for a tree, that is slower (~ 30 microseconds

per round 2n -> n)

- Will require to use another zkSNARK to prove inclusion upon exits, that is more expensive, but can be batched. Batch size of 5 gives verification cost of 300k gas per proof

Improvements

- May transpose a zkSNARK to prove not the fact of non-inclusion of some range for a set of blocks, but a non-inclusion of the set of slices in one or more blocks

- Can be trivially extended to sum-trees