Q: do we want to handle fee payments in a token-agnostic fashion or not?

Possible Token-agnostic fee model

Sequencers apply heuristic that 1st function call in a transaction is the “fee payment” transaction. They check it to see if/how much the tx will pay them.

Gas metering in a token-agnostic model

In the kernel circuit, the tx sender defines gasPrice

and gasLimit

values. The sequencer must validate these are consistent with the fee payment in the 1st transaction

Q: How do we handle refunds in this model if not all gas is used?

Depositing value into Aztec

On L1, a user calls the deposit

function of a token portal contract. The portal will write a message into the L2 message box that allows an L2 deposit

function to be executed.

The L2 deposit

function will take a portion of the deposit value (Q: how much and how is this computed?) and send it to the sequencer; the remainder being added to the depositor’s balance.

The user makes an L2 tx that calls the deposit

function on the token contract. As this is also the 1st (and only) function call in the tx, the sequencer will identify they are being paid.

i.e. depositing funds can work by following the existing above heuristics and using the message passing spec

Paying the sequencer

(idea from Mike)

We create an L2 contract that anybody can send funds to. The contract has a withdraw

function that, when called, will send its funds to a designated address. The withdraw

function asserts that msg.sender == coinbase

.

This is useful because it enables the sequencer identity to remain private.

Users send their funds to the above contract. At the end of the block, the sequencer calls withdraw

, which is a private function that does not leak their identity.