We want to be able to read L1-to-L2 messages from both private and

public contracts. The way we currently read messages in private contracts might not be acceptable in public contracts.

Reminder: private execution and proof-generation is performed client-side (in the PXE), but public execution and is performed by a Sequencer and proven by a Prover.

Today, if a private

contract reads an L1-to-L2 message, it calls an oracle and provides the message key. The PXE checks its database for a message with matching key. If an entry for the key is found, the full message and its sibling path are returned and become circuit inputs.

In public, if an AVM opcode (READL1TOL2MESSAGE(key)

) existed to do this same thing, it would somehow need to require that the sequencer/prover prove non-existence of key

, otherwise the sequencer could return empty/no-message without proof. Non-membership proofs (given only a leaf value

) are not feasible in an ordinary (non-sparse/indexed) merkle tree.

I believe are options are the following:

1. Modify the l1tol2message tree to be an “indexed” merkle tree

so that such membership checks are feasible.

1. Require that the READL1TOL2MESSAGE

opcode accepts leafIndex

as argument.

Then it would be the responsibility of private execution to determine the leaf index of a message and forward it to an enqueued public call via calldata.