Thanks to the great work that has gone intoEIP-3074

One my motivations for wanting in-protocol meta-transactions is the goal to remove the ability to observe gas in the EVM. The simplest version of this would either remove the GAS

opcode, or turn it into a no-op.

There are bad

EVM patterns that use gas observation. Basically, any contract that hard codes a gas value for a sub-call is bad

because it can break if the gas schedule changes. This happened with one of the Aragon contracts a while back.

There are however, legitimate patterns for gas observation in the EVM as it exists today, specifically meta transactions. We have the “signer” who is the party who wants a transaction to be sent and the “invoker” who actually sends the transaction on behalf of the signer. Both of these parties have certain guarantees they need for this to work.

As the signer, you need to know that your transaction will be executed exactly as specified and that the invoker cannot intentionally cause your transaction to fail (when it otherwise should succeed) and still get paid their fee.

As the invoker, you need to know that you will be compensated for the gas costs, regardless of whether the transactions succeeds or fails.

The idea of enshrining meta-transactions into the protocol was originally inspired by previous attempts to remove gas observability. Unfortunately, it appears this is still an open problem. Here’s why.

The majority of the requirements for both invoker and signer under 3074 seems to be fulfilled by “a verified trusted invoker” meaning that the signer has verified that the invoker’s code does indeed respect these values. The one

place this seems to fall short is the gas stipend for the subcall. CALL

opcode semantics are that the gas

parameter to a CALL

is not guaranteed to actually be the value passed into the opcode since we do roughly

gas = min(specified_gas, gas_remaining - (gas_remaining // 64))

. This means that if there is not enough gas left, the call will get less gas than whatever was specified.

This inability to strictly require the CALL

operation be executed with at least X gas

is where gas observability seems to remain critically required for meta-transactions to not require trust, and subsequently. Stated differently, it looks like one blocker to removing gas observability would be the ability to strictly require a subcall be executed with >= minimum_required_gas

which currently can only be done imperfectly by manually checking how much gas is remaining with the GAS

opcode.

here’s the code from py-evm for how the gas value is determined:py-evm/call.py at 7afe050f0849ed87a4282038d394212f6029d3b9 · ethereum/py-evm · GitHub

A naive solution would be modifying the AUTHCALL semantics such that the opcode causes a REVERT if gas_remaining < required_call_gas

. This seems to allow for the signer to retain their ability to verify an invoker contract and know their transaction will be executed as intended, and the responsibility falls on the invoker to provide adequate gas, otherwise the call will revert without them receiving their payment. It’s still unclear to me whether this is sufficient. It would still require auditing the invoker contract to verify that it will indeed revert when not enough gas is provided.