Second of the Offslack Specs.

What

There are a host of new challenges that present themselves in making the MEV supply network programmable. One in particular, and which this note targets, is that of the interface between a block building program and a solver program. This is relevant as we may eventually see multiple solvers on the SUAVE network, but even more relevant today, in the short term we may have a path towards a more fair bottom of block auction via a “network solver” that block building programs can hook into, or maybe even vice versa where a Solver could register a backrun hook on the block building program.

How?

[

Screenshot 2024-04-20 at 6.50.30 PM

2366×1316 147 KB

](https://collective.flashbots.net/uploads/default/original/2X/c/c9c08483f195a25a05280d6c00671e9eacb05d8d.png)

An example Block Builder Program can be seen in our SUAPP examples repo. The question this note poses is how can we allow for arbitrary solvers to communicate with block builder programs to register their intent or availability to back run the block. Ideally the block builder could in parallel send to the solver and make available to the relay so as to parallelize the work and not block on back running.

This note does not suggest a concrete implementation, but ideally the solution looks like a standard for the Solver and Builder to communicate through. Some other requirements:

1. Solver Programs should be able to “register hooks” somewhere
2. Block Building Programs should be able to get list of “fresh hooks”
3. Communication between the two should be private and secure
4. Solver program should adhere to the security requirements of the Block Builder Program, ideally Solver cannot leak any information at all.(
5. Potentially an object capabilities model with some security features could be relevant here?)
6. Potentially an object capabilities model with some security features could be relevant here?)

A solution to this post should be in the form of a spec for an API and potentially a SUAPP example pattern. Necessary to solving this spec is also considering how the block data is passed to the solver program, whether it’s the entire block, the state diff, or maybe even a massive list of pool balances!

Discussion

How do we handle refunds? Prior work has shown that[rebates in the permissionless setting](#) can be particularly challenging.