Edit

: Note, this discussion precedes the current specs and those should be referred to for concrete details - this post is an artifact of initial brainstorming . The post also makes use of the term “kettle” which should be thought of as a TEE running the MEVM and “brewer” which is the operator of the kettle infrastructure.

While kettles provide integrity guarantees, brewers (kettle operators) still have degrees of freedom over inputs to the kettle. For instance, a brewer may bid in an OFA and attempt to censor competing bids. We have thought to combat this by providing a DA module which allows contracts to encode conditions like “this function will only execute if all relevant inputs in the DA layer have been processed.” Enforcing this would require some way of ensuring that kettles are executing against an up-to-date version of the MEVM & DA module.

Questions:

- How do we ensure that up-to-date DA/MEVM contracts were used to form an export block? Verifying a TEE certificate?
- How do we support a contract on SUAVE allowing for a certain transaction to be included in an ETH block at SUAVE block height n

, but then the transaction is no longer allowed at n+1

(e.g. because of an oracle update)?

- We may want heterogenous sources of DA - i.e. multiple DA committees doing DA for different use cases. For example, a UniX committee and a committee for CoWswap-on-Solana. This could be useful for scaling, but also satisfying the varying preferences and trust assumptions of the apps who may want cheaper DA, lower latency or trust a certain set of nodes. How do we support multiple sources for DA? We could possibly do this via encoding DA sources in contracts.
- If we can do the above, can we allow contracts to live anywhere, not just on SUAVE consensus? This would have SUAVE resemble Anoma a lot more closely.
- One solution would be to have the proposer pass in the ETH parent block and latest SUAVE block hash in the getHeader call. However, it isn’t clear if its in the proposer’s incentive to do (not doing so might enable them to decrypt and exploit some transactions, especially when colluding with a kettle). The nice part of this approach is that this could mean that the target chain could own more of the stack - builder contracts and DA while kettles just handle execution.