This is my understanding of the existing MEV problem, given various talks from the flashbots team. I want to describe my understanding of this problem so far, because if I or others want to think about this problem, we need a clear description of what the issue is and what has/hasn't worked so far. In this post, "users" here refers only to people sending transactions, not miners or searchers (searchers build blocks and have access to MEV).

While some may say we don't want MEV at all, a better solution is that we want them to be able to extract MEV so that MEV is not extracted 100% extractively in the next block anyways, but be able to share those profits. The advantage of a searcher getting MEV instead of the next block builder, is that it's possible for the protocol to get them to pay users for access to their transaction data for MEV. For instance, users could progressively reveal their private information that searchers bid on. Although users will get MEVd either way, this way they get compensated for their valuable asset, which is the commitment to a private transaction that only they know that they will execute.

Currently, the proposed solution (which doesn't make sense to me) is that all block builders run SGX so they can privately build their blocks. This will theoretically (but not in practice) keep transactions private. You can even keep txs private over SGX-holders by having each SGX generate deterministic private randomness, generate sk-pk pairs from that, and create a join oublic key with other searchers via El-Gamal algorithm over all other SGXs who are approved to be searchers. Then, users can encrypt their transactoins with this shared public key, ensuring that all of these SGX's and only these SGX enclaves can read these transactions, not even the searchers themselves. The reason we want > 1 SGX is two fold: 1) flashbots should move away from being a centralized point of failure for Ethereum and 2) if everyone knew the specific searcher algo or were able to reverse engineer it, then this would be akin to running a deterministic strategy in poker, where it is exploitable and 3) there are many types of MEV, and different searchers may be the best at different types. We want blockbuilders to run mixed strategies emergently, the best way of which to do so, is via allowing multiple searchers. While this solution seems OK if SGX algorithms are somehow audited or ensured they don't leak data, any proposal with SGX makes no sense because the algorithm on the SGX itself they can decrypt whatever they want via arbitrary SGX code and output it via logs (you don't even need side channel attacks, but also sgx repeatedly gets hacked for every 2 years so it's not particularly safe here either). In a nutshell, SGX's gaurantee of verifiable execution is not sufficient to solve this problem, you need semi-private or semi-auditable execution.

Since the user is the only one who knows the transaction, ideally we have some system that allows them to commit to it, but prove things about it. One way to do this is fast SNARK proofs on the client wallet. This isn't really feasible in metamask right now, since SNARKs take too long and need specialized hardware to parallelize. For the sake of argument, let's say we figure out private remote snark proving efficiently and we can SNARK progressive info about the transaction super fast. One mental model of the benefit of this, is a back and forth with searchers, where the user produces a SNARK where they can prove they encrypted a valid transaction where they promise to reveal and send it after this back and forth game. One way to make that promise is to publicly VDF-encrypt the tx as public output, ensuring that it will be revealed after predefined N seconds. You can also prove its encrypted with a key (tx_key), and publicly output a VDF on the encryption of the tx_key with the searcher's shared key (similarly to how the SGX key was generated), so only the searcher enclaves can decrypt in N seconds. Then, searchers pay for access to that data, and bid for the next "reveal". Users selectively release increasingly more properties of that tx i.e. the exchange, then the amount, then the coin, or whatever app specific game. This back and forth game seems hard with latency though, and apps need to invest eng time to determine how to play this game, and though it seems useful and easy for specific protocols or exchanges, it seems hard (if not impossible) to generalize. One early way to do this might just be like, access to a preset order of the fields of the tx (including calldata, which will probably be the most valuable field), and punt the EV calculation to MEV searchers.

You can even systematize this early solution via tiers. One way to get these layers of information revealed without back and forth with searchers, is to have searcher tiers. Searchers opt into higher tiers to get more information dense txn data. This amount can be determined by a completely blind vickrey auction per block/tx for instance. i.e. searchers can bid to get different amounts of standardized information: i.e. maybe nothing, or the primary contract that they are interacting with, or the amount of eth being transferred, or all contracts touched in the txn, or plaintext txns. They only get masked txdata with a ZK proof that it corresponds to a valid VDF-encrypted tx. They each try to do MEV off of the info they have, and bid on the right to send to validators for inclusion, say via a revenue sharing model with the validators. The VDF can be calibrated per-tx to be solved after searchers have to submit but before validator has to validate, allowing it to be revealed before touching the existing L1 consensus layer. This however, requires a user to generate several snarks fast and in parallel. It might need outsourcing to an untrusted machine or something, which should be possible given recent advances. However, this requires finding tiers that are generic enough to work on defi, gaming, etc, which may require too much opinionation.

Although there are 3 types of MEV, MEV is app specific, and we shouldn't be prescriptive about that – since there are optimal routing algorithms that make backrunning irrelevant, and I didnt really understand the utility in seperating "useful MEV" out, since this requires very detailed information on the tx/mev that i dont think we have access to, and is not our responsibility when optimal routing algorithms exist that make "positive MEV" irrelevant.

I didn't und understand how SGX or TSS could ever work. SGX has no way to commit to an algorithm, and TX senders need to be part of an N of N TSS to avoid giving specific TSS key holders unilateral power to extract MEV. It seems the only way forwards here is for custom wallet integrations and ultra-efficient SNARKs, which is on the horizon. For instance, we are working on fast secp256k1 verification in a SNARK which will allow fast verification of txs, and some related work like proof of ethereum state change can also prove useful here.

Two questions that might help illuminate paths forwards are: how do miners currently choose which searcher bundle is canonical? And in the TSS solution, what exactly is the thing that's being TSS'd out of the user txs, searcher algos, and the searcher blocks?