MEV-SGX: A sealed bid MEV auction design

Introduction to Flashbots

Flashbots is a research and development organization working on mitigating the negative externalities of current MEV extraction techniques and avoiding the existential risks MEV could cause to state-rich blockchains like Ethereum. Our primary focus is to enable a permissionless, transparent, and fair ecosystem for MEV extraction.

We plan to achieve this by:

- Bringing transparency to MEV activity
- Democratizing access to MEV revenue
- Distributing MEV revenue in a way the benefits the community

Flashbots has released two products so far which make progress on resolving the MEV crisis:

- Flashbots Alpha: a proof of concept communication channel between miners and users for transparent and efficient MEV extraction. Our Alpha consists of MEV-Geth, a fork of the Geth client that provides a sealed-bid blockspace auction that allows users to communicate granular transaction order preferences, and MEV-Relay, a transaction bundle relay.
- MEV-Explore: a public dashboard and live transactions explorer of MEV activity.

Today there are two primary groups of users for Flashbots' products: miners and searchers. Miners are the block producers of Ethereum who secure the network by performing work in return for the block reward, uncle rewards, and fees from transactions. Miners group together cooperatively in "mining pools" to collectively generate blocks, thereby increasing their probability of successfully mining a block.

Searchers send transactions through Flashbots for inclusion in a block. Searchers are users that "search" for MEV to extract. For example if the price of ETH/DAI is high on Uniswap and low on Sushiswap, then the searcher could buy on Sushiswap and sell it on Uniswap, thus making a profit from the price difference. The searcher would capture this profit by creating a transaction to perform the aforementioned arbitrage and send it to miners, who would include it in return for a transaction fee from the searcher.

Flashbots plays an important role in Flashbots Alpha today as the operator of the MEV-Relay that sits between miners and searchers. The relay is necessary today to prevent DOS or spam attacks on miners for reasons which will be expanded on below.

MEV-Geth and Flashbots' design goals

In early 2021 Flashbots released Flashbots Alpha. These were the design goals we launched with:

- Permissionless A permissionless design implies there are no trusted intermediaries which can censor transactions.
- Efficiency An efficient design implies MEV extraction is performed without causing unnecessary network or chain congestion.
- Pre-trade privacy Pre-trade privacy implies transactions only become publicly known after they have been included in a block. Note, this type of privacy ensures that transactions not included in a block remain private, but does not guarantee privacy from privileged actors such as transaction aggregators, gateway, or miners.
- Failed trade privacy Failed trade privacy implies losing bids are never included in a block, thus never exposed to the public. Failed trade privacy is tightly coupled to extraction efficiency.
- Complete privacy Complete privacy implies there are no privileged actors such as transaction aggregators / gateways / miners who can observe incoming transactions.
- Finality Finality implies it is infeasible for MEV extraction to be reversed once included in a block. This would protect against time-bandit chain re-org attacks.

While Flashbots Alpha has been and continues to be successful it offers incomplete trust guarantees. It is not permissionless because miners who adopt it have to be whitelisted by MEV-Relay in order to be forwarded bundles. It is not completely private because bundles can be seen by miners prior to inclusion on-chain. Lastly, Flashbots Alpha offers no finality protection against chain reorgs. While finality is important, we are focusing first on permissionless and complete privacy as the next design goals to achieve.

The properties of permissionlessness and complete privacy are difficult to achieve for three reasons. First, Flashbots introduced 0 gas price transactions which pay the miner through a smart contract transfer. These create a potential DOS vector as miners need to simulate transactions in order to determine their profitability, if any at all, and a malicious searcher could spam miners with worthless transactions at no cost to force them to expend resources simulating these transactions. In contrast with regular Ethereum transactions there is an inherent cost to sending transactions because of the fees paid due to the gas price of the transaction. Further, nodes on the network help to filter out bad transactions sent through the public mempool.

Second, many proposed solutions to provide complete privacy or permissionlessness introduce latency and reduce overall

system performance, which could lead to a higher rate of mining uncle blocks. Lastly even worse than mining an uncle block is mining a block that is altogether invalid. A completely private system will need to provide miners additional guarantees that the blocks they are mining without seeing are valid and profitable to mitigate the risk of mining invalid blocks.

Approaches to completely private and permissionless MEV auctions

There are a few different proposals for completely private and permissionless MEV auctions. In this section we will briefly review each as well as their associated tradeoffs. Some further discussion can be found in our #MEV-Research Discord channel as well as the “Privacy Solutions for MEV Minimization” roast.

Using block headers for privacy One proposal for achieving a permissionless and completely private MEV auction is for searchers to craft full blocks and send the headers of those blocks to miners while withholding the content of the transaction trie. Using these headers miners would be able to perform proof-of-work, but they would not see the full block or the transactions that went into it. After a miner finds a proof-of-work solution the searcher would reveal the full block data to the miner, at which point the transactions would become visible to the miner and the block would be sent to the network for inclusion. Although this achieves our desired privacy guarantees, miners are unable to verify the validity of a block before performing work and therefore this proposal does not solve for permissionlessness. Thus without some kind of permissioned system or whitelist a malicious searcher could send invalid blocks to the miner and DOS the network.

Bonded block headers A variation of the block header mechanism would be to require searchers to post bonds before they can send block headers to miners. If searchers act maliciously (e.g. spam invalid blocks, withhold block content), their bond could be either slashed or claimed by the miner, thereby introducing a cost to bad behavior. A drawback of this mechanism is it increases the barriers to entry for new users by requiring them to lock up a significant amount of capital (commensurate to the opportunity cost of DOS). We are still exploring this solution in parallel. MiningDAO has implemented a variant of this system and other community proposals can be found here and here.

Timelock encryption Timelock encryption, often referred to as “commit reveal scheme,” is a cryptographic primitive that ensures an encrypted message can only be decrypted after a certain period of time has passed, using for example Verifiable Delay Functions. In our context searchers could send their encrypted transactions to miners where they would be added to the block immediately. After a period of time (or number of blocks), miners and other users of the network could decrypt the transaction to reveal its content. Since transactions would be encrypted for a period of time then current permissionless methods for propagating Ethereum transactions could be used.

Using timelock encryption has several drawbacks. First, many MEV extraction opportunities are timebound (e.g. they will be captured in the next block) and a solution that delays the execution of searchers’ transactions would make it impossible to extract these opportunities. Making all transactions timelocked could level the playing field, but that would severely degrade user experiences. Furthermore there is an inverse correlation between user experience and security: the longer the delay between transaction inclusion and execution, the stronger the security offered by timelock encryption.

Threshold encryption Another cryptographic solution would be to use threshold encryption such that a committee of block producers is required to decrypt encrypted transactions sent by searchers. Each miner would have a share of a decryption key and some threshold (e.g. n of m) would be needed to decrypt transactions. While this solution provides some additional privacy and validity guarantees, it relies on an honest majority assumption that the key holders won’t collude to break the encryption, therefore it is difficult to make joining the set of key holders a permissionless process. Threshold encryption by committee also introduces a bandwidth intensive step to block production which may be found to be untenable. Threshold encryption could be a promising path forward if these concerns are resolved.

Secure enclaves Secure enclaves, like Intel’s SGX or AMD’s SEV, could also be leveraged. Searchers would use an enclave to validate that their bundles are valid and profitable, thus mitigating DOS and enabling permissionless interactions between searchers and miners. Miners would use an enclave to store encrypted blocks from searchers, and receive truncated header hashes they can use to perform proof-of-work. The miner’s enclave would unencrypt and seal the searcher’s block when provided a proof-of-work solution.

A drawback of this solution is that the searcher’s privacy guarantees are only as strong as the security of the miner’s enclave and it would be difficult for the searcher to determine if the miner has broken the enclave. Furthermore, using enclaves may introduce latency into the system that make it infeasible to use.

Summary of approaches and their tradeoffs

Permissionless Complete privacy Latency Drawbacks

Secure Enclaves Yes Yes Low - Medium Non-falsifiable guarantees for the searcher, potentially latency

Timelock Yes Correlated to delay Low High transaction execution delay

Threshold Encryption Yes Yes High Assumes an honest committee and is bandwidth expensive

Block headers No Yes Low Malicious searchers can spam miners or send invalid blocks

Bonded block headers High startup cost Yes Low Requires significant capital (commensurate with the opportunity cost of a DOS) to become a searcher

While all of these solutions deserve to be researched further, this post presents a proposal for how permissionlessness and complete privacy guarantees can be achieved through the use of secure enclaves. Other designs are not being ruled out, but we are seeking to implement a system that achieves our design goals as soon as possible.

MEV-SGX

MEV-SGX uses secure enclaves, specifically Intel's SGX, to provide complete privacy and permissionlessness. In MEV-SGX both searchers and miners have their own SGX, and here we use "miners" interchangeably for "mining pool." Searchers craft a full block with their transaction(s) included in it and validate that block in their SGX. The searcher's SGX then encrypts the block and sends it to miners along with the unencrypted truncated header hash of that block. Miners would store the encrypted block in their SGX and use the truncated header hash for proof-of-work. Only after finding a proof-of-work solution and providing that to their SGX can miners decrypt the encrypted block, seal it, and propagate it to the network. Sealing is the process of adding the mix hash and nonce to a block header.

Prior to interacting, searchers and miners perform a handshake and use cryptographic attestations to ensure that each party is running the right software in an SGX. Due to the tamperproof nature of SGX these attestations provide cryptographic trust to searchers and miners, as they have guarantees that their counterparty is running specific code in an environment that cannot be tampered with or broken into. The searcher's use of the enclave allows them to permissionlessly interact with miners by ensuring the blocks that searchers send them are valid. The miners' use of the enclave provides complete privacy to searchers.

MEV-SGX Architecture and Process

The architecture for MEV-SGX looks like this:

Building off of the architecture, these are the steps involved in the process of MEV-SGX:

1. Handshake: The searcher and miner perform a handshake to discover each other, exchange public keys, and verify that each party is running MEV-SGX in an SGX through cryptographic attestations.
2. MEV Opportunity Detection: The searcher detects some MEV then creates transactions that extract it and pay the miner upon successful extraction.
3. Block Creation: Searchers use their node to generate a block including their MEV-extracting transactions. They also generate the block witness and input both the block and the witness into their SGX.
4. Block Validation: The searcher's SGX processes the block to ensure it is valid and profitable for the miner by using the block witness generated in (3).
5. Block Transfer: The searcher's SGX encrypts the block with the miner's public key and sends it to the miner. The searcher's SGX also sends the miner the block's truncated header hash and how profitable that block was.
6. Block Selection: The miner selects the most profitable block they've received.
7. Proof-of-Work: The miner uses the most profitable block's truncated header hash to request proof-of-work from workers.
8. Block Sealing: After finding the proof-of-work solution the miner passes that into their SGX. With this solution the miner's SGX decrypts, seals, and exports the block outside of their SGX.
9. Block propagation: The miner's node propagates the decrypted and sealed block to the network.

A detailed list of steps, definitions, and the inputs and outputs of each step can be found in the appendix.

MEV-SGX block production process diagram MEV-SGX changes the process by which blocks are produced, validated, and propagated to the network. The block production process is as follows:

A higher resolution version of this diagram can be viewed here.

Limitations of MEV-SGX

Below are several limitations of MEV-SGX in bold, along with mitigating actions in non-bolded text:

- Increased latency from using SGX: Minimize what is done in the SGX, explore new generations of SGX, perform rigorous benchmarking, and move towards low-level implementation for optimal performance.
- Searchers could break their SGX: Falsifiable by the miner by checking for invalid blocks. If it happens the systems falls back to Flashbots Alpha.
- Miners could break their SGX: This is non-falsifiable by the searcher. Currently exploring solutions like threshold committees that mitigate this limitation.
- Searchers must propose full blocks: If searchers need to propose full blocks then it is much more difficult to do bundle merging between multiple searchers. We are researching methods of partial block proposals for searchers and privacy preserving bundle merging in the miner's SGX as well.
- Reliance on SGX: Explore feasibility of offering multiple types of secure enclaves as well as continuing research into software and cryptoeconomic based methods of achieving complete privacy.
- Miners can see searchers transactions after sealing the block, but before propagating it: Explore alternative structures that don't rely on the miner for data availability

Summary of MEV-SGX proposal

MEV-SGX could enable Flashbots to achieve its design goals of creating a completely private and permissionless system. Searchers would craft blocks with their bundles included, validate and encrypt those blocks in their SGX, and send them to miners alongside block truncated header hashes. Miners receive truncated header hashes and encrypted blocks they know are valid and profitable for them to mine. They use the truncated header hashes to perform proof-of-work on blocks without seeing them, and upon finding a proof-of-work solution are able to decrypt and seal blocks.

Next Steps and Open Questions

MEV-SGX is in early stages of development. Flashbots intends to take this design and implement a proof-of-concept that can help us learn more, pressure test our assumptions, and carry out latency benchmarks. From there we will release our findings and start work on a version of MEV-SGX that searchers and miners can test, and will keep testing and iterating on MEV-SGX thereafter. In parallel we intend to continue to research alternatives for ensuring complete privacy and permissionlessness.

There are several open technical and theoretical questions with MEV-SGX that we are seeking to explore in the coming months (please reach out if you want to contribute!).

MEV-SGX

1. What is the latency overhead from MEV-SGX compared to the normal block production process?
2. How can MEV-SGX provide complete privacy and merge non-competing bundles at the same time?
3. In the current design for MEV-SGX the searcher cannot provably tell if miners have broken into their SGX. In other words the trust guarantees of the miner's SGX are non-falsifiable. How can we either provide falsifiable guarantees or mitigate the risk of a miner breaking their SGX?
4. MEV-SGX relies on the miner to propagate blocks to the network although the miner has no role in crafting or validating that block. Can we rely on the searcher for propagating sealed blocks instead?

ETH2.0

1. How does the design of MEV-SGX change in ETH2.0?
2. Does ETH2.0 make it easier to achieve our design goals with cryptographic or cryptoeconomic methods besides secure enclaves?
3. In ETH1 miners have to perform proof-of-work before they are able to decrypt and seal the block, and this gives them an incentive to immediately propagate the block instead of frontrunning the searcher and mining a new block with their transaction in it. In ETH2 validators would not have to expend resources before they can decrypt and seal the block. How should our design change to account for this?

System design

1. What are the alternatives to secure enclaves for achieving our design goals of permissionless and complete privacy? Which of these can be achieved without a protocol change?
2. Should MEV-SGX be considered an add on to existing systems, or a replacement? In other words, should MEV-SGX be run in parallel to the existing non-private MEV-Relay or should it replace it?

Appendix

Please see this link to find a list of definitions and a table detailing the process of MEV-SGX step by step with data inflows and outflows for each step.

Call for Feedback & Contributions

- Contribute to MEV-Research We invite you to review our MEV-Research GitHub repo to learn about our MEV Fellowship program. Start contributing through opening or answering a Github issue, and/or writing a Flashbots Research Proposal (FRP), and join our discussion on our MEV-Research discord community.
- Try our proof of concept We hope to have a proof of concept released shortly. We encourage traders and miners to review our code base and try MEV-SGX out. Join our Flashbots discord community or contact us at info@flashbots.net
- Subscribe to MEV Ship Calendar You can follow the latest updates and events by subscribing to our MEV Ship Calendar: join us on our semi-monthly community call "MEV Ship Treasure Map Roast", semi-weekly core dev call, weekly research workshop, and the upcoming unconference: MEV.wtf
- Work with us Flashbots has several open jobs, including a systems engineer role that encompasses MEV-SGX work. If you are a self-directed individual who puts collective success above your own and are motivated by solving hard problems with asymmetric impact, you will fit right in.