Resisting MEV Attacks via Privacy-Preserving Techniques

David Mberingabo, Ethan Wang, Jieliang Yin, Zili Zhou

Carnegie Mellon University

Final Project Report

17631 Information, Security, Privacy & Policy

Abstract

Blockchain aims at building a fully decentralized trading environment and providing more equitable and inclusive decentralized finance services (DeFi) for everyone. However the actuality is that on-chain service providers like miners still have privileges for arbitrage called Maximal Extractable Value (MEV) because they can self-determine the content and sequence of transactions within the block. To protect users from malicious MEV attacks like front-running and sandwich attacks, we design a scheme to apply privacy-preserving techniques like commit-reveal and optimize on-chain process and structure like proposal and builder separation (PBS) to prevent users from MEV attacks and build a more regulated, usable and reliable trading order on blockchain.

Keywords: Blockchain, DeFi, MEV, Privacy-preserving, Commit-reveal, PBS

Table of Contents

A Refresher on Ethereum and DeFi

Classic blockchains, like Bitcoin, allow users to store records of their transactions amongst themselves on a public ledger, agreed upon by miners. Miners are peer-to-peer nodes that agree on a bundle of transactions to be added on the blockchain, as a new block of transactions. This formal process of agreeing is called a consensus protocol. Ethereum is another popular blockchain with a consensus protocol called Proof of Stake (PoS) and, instead of miners, it has validators who stake network tokens called Ether. Bad behavior from validators is met with the slashing of the staked ETH as punishment and good behavior is rewarded with some amount of ETH.

The total ETH amount awarded to validators depends on transaction fees paid by the users and the amount of newly minted ETH. The users can attach as much transaction fees as they want to their transaction and the higher the transaction fees the more likely the transaction will get committed into a new block by a validator, who would collect the fees once the transaction has been committed. Validators are also rewarded with newly minted ETH from each block they commit [22]. A blockchains' level of decentralization and security usually comes down to the incentives at play amongst the participants and if they are aligned with maintaining agreement on the same order of transactions over time.

Ethereum allows for Decentralized Applications (or Dapps) to run as a set of smart contracts on the Ethereum Virtual Machine. Smart contracts, like normal Ethereum transactions, cost fees depending on their computational complexity and users can add a tip for the validators to increase their chances of being committed. Many of the Dapps are finance related and they are

collectively known as Decentralized Finance (or DeFi) applications. Decentralized Exchanges (DEX) are DeFi applications that allow users to trade token pairs, NFTs and other financial assets. This Dapp layer creates novel complexity and risk to the ongoing decentralization of Ethereum. This paper aims at explaining this new risk, the solutions proposed by the community and our own solution to mitigate this risk.

The MEV Problem

When a user runs a smart contract, it is sent to validators, who execute the smart contract. The smart contract is committed on the blockchain as proof that it has been executed. Let's assume that a smart contract is an execution of a token pair transaction on a DEX that will have a large impact on the various markets across the DEXs. The validators would be able to see this transaction before they execute it. Validators can even add their own transactions, before or after executing the user's transaction, in order to extract profit from the large impact on the market. When considering validators' incentives, this extra source of revenue has to be included alongside the transaction fees, also called gas fees on Ethereum, user tips and newly minted ETH. Monetary incentives for miner's behaviors in blockchain systems were categorized as miners extractable value (MEV) by Daian et al. [12].

Gas fees are paid by users alongside tips, while newly minted ether is earned per-block. These block rewards are designed into the system incentive for miners. Other types of MEV include order optimization fees, described by Daian et al. [12], as "implicit fees from modifying transaction order". Order optimization fees are external and implicit to the system. Miners can extract OO fees by reordering blocks or injecting their own transactions in blocks. As

Ethereum's DeFi ecosystem grows bigger, one can expect large external OO fees to be introduced. Some OO fees are good for the utility of the blockchain, and some are bad for the integrity of the blockchain. Arbitrage trades are an example of OO fees extraction activity that benefit DEX markets by resolving price inefficiencies and increasing market liquidity. These OO fee opportunities also attract more users, in the form of market traders, and miners, in the form of MEV extractors.

Some OO fees extraction activity negatively affects users in terms of gas prices, trade prices and consensus layer reorganization. Gas prices are expected to be auctioned up by MEV extractors against user's transactions. In the current version of Ethereum, users trading on some DEXs are expected to receive the worst trade price as arbitrageurs have little incentive to share profits from their arbitrage extraction activity with users. Not only this but miners can simply run sandwich attacks on user's transactions. A sandwich attack is conducted by placing an order before the user's trade, then placing another trade after the user's trade has been executed, with the goal of profiting from the user's trade impact and giving the user the worst possible price they are willing to accept. An arbitrage opportunity that is exploited by an arbitrageur but not shared with the user can be considered an example of a sandwich attack on a user transaction.

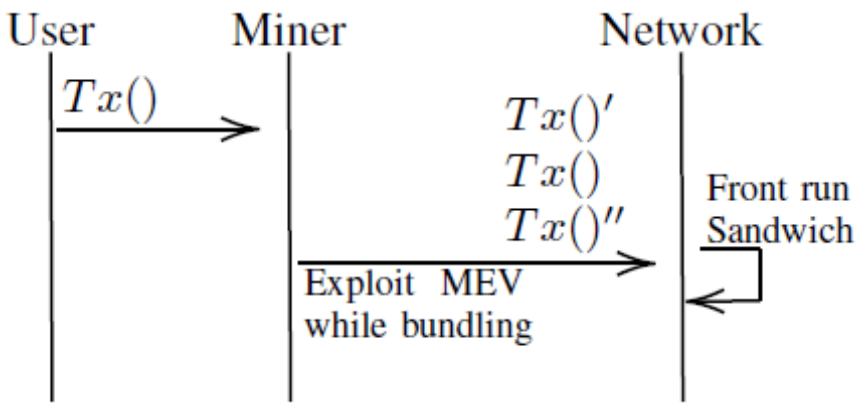


Fig 1. MEV exploitation as a Sandwich Attack (Babu Pillai, 2022 [23])

OO fees, in fact, can exceed the block reward and encourage reorganization and forking attacks [12]. Forking attacks are described as having two forms, forking/undercutting attacks [24] and Time Bandit Attacks. Time Bandit Attacks occur when a miner rebuilds the blockchain from a previous block in order to extract that block's MEV. This can lead to a miner using MEV profits to convince other miners to essentially fork the blockchain into a new blockchain with potentially lower decentralization and security. Essentially, if the external MEV extracted as OO fees is high enough and in conflict with the incentives to remain on the main blockchain, then some factions of the network will fork off, thereby reducing the security of the main blockchain over time.

MEV, more generally, is the profit that miners can extract from rearranging user transactions or injecting transactions, in a given timeframe. In order for miners to extract this value, they need the ability to (a) view the transaction details, (b) order transactions in a block, (c) inject their own transactions and (d) commit a block. These abilities are unique to validators,

and therefore, they have insider knowledge and control of which transactions will get committed and in which order. These abilities allow for user attacks such as sandwich attacks and network attacks such as forking attacks. These same abilities also allow for the normal functionalities of Ethereum and its DeFi ecosystem, such as the miner's consensus protocol and user-friendly arbitrage. It is safe to say that MEV is here to stay because the network benefits from some forms of MEV, however, there are different schools of thought on how to manage the threats MEV, especially OO fees, causes to Ethereum.

"Finally, we argue that miner extractable value, particularly in the form of order optimization fees, implicit fees from modifying transaction order, threatens blockchain consensus stability. Such fees are large enough to subsidize serious attacks on the network. They constitute an economic vulnerability that should be a current cause for concern in Ethereum." - FlashBoys 2.0 [12].

Existing Scenarios Scanning & Analysis

MEV Extraction as a Solution

There are two dimensions to the MEV problem, the first is that the miners can centralize to extract more MEV and the less powerful miners will be compelled to join the centralization efforts as MEV gets monopolized, and the second is that users do not get fair transactions and can be abused by miners. The first dimension requires a solution that shares the MEV profits equitably amongst miners. This means the network must be aware of how much MEV is being extracted by miners and the network must ensure the miners are equitably sharing the profits, i.e. not just one miner or one group of miners should be able to extract MEV. The second dimension is solved by minimizing miners' ability to conduct MEV attacks on users.

When mapping solutions that deal with MEV extraction and sharing, one must differentiate the theoretical schemes and techniques from the live implementations. In this paper, we chose to focus on the theoretical schemes and techniques most accepted in the community, but also reference some live implementations.

Mining Pool Services and DEX Aggregators. Miners within the Ethereum network can group together into a mining pool and promise the user certain things, such as not broadcasting transactions sent to the pool to the rest of the miners not in the pool or sharing MEV profits that do not harm the user. These pools may even extract MEV and promise to share the profits with the users or not to run any MEV extraction at all. The miners are kept fair by the pool, usually via some kind of financial or reputation staking. These mining pools can be made up of whitelisted and identifiable miners that offer services or promises to users such as private

transactions. These types of mining pools are called permissioned, however a true decentralized solution for MEV should be on a permissionless network.

Exploiting arbitrage opportunities fixes price inefficiencies in the market that arise from a trade, whether it's trading NFTs or a USDC/ETH pair trade. DEXs and DEX aggregators extract this type of MEV and compete to offer the best price to the user and make a profit. Although mining pool services and DEX aggregators alleviate some of the network inefficiencies via MEV extraction, they are also under long-term threat of increased centralization pressures. As the total MEV in the system grows, miners will be incentivized to extract maliciously MEV if it's higher than the miner's staked assets and break their promise. This ultimately leads to forking attacks as MEV grows due to the trust required when accepting promises from miners. A real solution should be trustless and permissionless.

A popular implementation that uses coincidence of wants, DEX aggregators and bond-staking mining pools is CowSwap [25]. CowSwap is able to offer traders the best prices on the market while slashing miners that do not act as expected, however the pressure to centralize still exists even in CowSwap in the long term since miners will betray users as soon as the MEV is higher than the cost of being slashed. Another popular implementation is BackRunMe by bloXroute.

PBS Model and MEV-Boost. The Proposer-Builder Separation (PBS) model is a winning model for better understanding MEV, supported by Ethereum's Vitalik Buterin and FlashBots [22]. The model assigns miners to either a builder role or a proposer role. The builders are allowed to view the mempool and build transactions into bundles. Proposers will only be able to

commit blocks they get from a market of builders. The idea is that builders will leverage their OO fees extraction ability to propose bundles along with a competitive bid. Proposers will choose which bundle to commit according to a bundle's block reward and bid, without seeing the actual transaction details in the bundles. The trust assumptions in PBS, as identified by Ethereum [22], are that (a) there is minimal or no risk that either proposers and builders will make promises and bids they do not fulfill; (b) that proposers cannot see the transaction details in the bundles until after proposers have promised to commit them into a block; and (c) consensus-layer simplicity and safety is maintained. It is also assumed that proposers have computational requirements similar or lesser to builders. PBS itself leaves a few things unanswered on purpose; to give wiggle room to future implementations. Our solution, as presented later, is a derivation of the PBS model.

A live implementation of the PBS model is MEV-Boost, by Flashbots. Flashbots is a team that runs a centralized server in Ohio, called a relay. This relay sits between a builder and a proposer (in below image validator would be proposer in PBS). Builders are considered a private mempool because all the participants in the pool promise some level of transaction privacy, however the user is in charge of choosing which builders to send their transaction through. In fact, Flashbots warns its users not to trust all builders and that they can perfectly see the users transactions and still attack them, however, the builders' track record can be maintained and they can be punished for conducting attacks. The builder bundles transactions into a block and sends them to a relay. Flashbots promises to forward only the bundle's header, which contains the block reward and bid information, to the proposers. Relays can also see transactions, but promise not to run MEV attacks on users, but under threat of legal and reputational repercussions or some

kind of penalty, they promise to not extract malicious MEV from users and so users flock to them, which allows them to potentially outbid the rogue builder or mining pool that tries to extract malicious MEV. Builders are incentivized to essentially auction their block data to the bigger fish, Flashbots' relays, and Flashbots relays ensure the builders do not extract malicious MEV and pay builders competitively better than trying to extract MEV alone. To improve the trust assumption of builders, Flashbots came up with Flashbots RPC, which is a way for users to send their transactions directly to Flashbots' own builders who promise no frontrunning and no failed transaction fees [26].

'*Centralizing MEV extraction is good because it quarantines a revenue stream that could otherwise drive centralization in other sectors*.' - Vitalik Buterin

The proposer promises to commit a bundle by sending a signed bundle-header back to the relay, thereby accepting the bid. The relay upon receiving a promise to commit then sends the rest of the bundle's data, including the transaction details, to the proposer to be committed into a block. The proposers are incentivised to get their bundles from the relay as it offers the best MEV extraction from a highly specialized builders market. Since proposers are assumed to be less technically capable, they should be getting their best MEV extractions from relays, like Flashbots. Flashbots Auctions escrows block data and fees, then slashes untrustworthy proposers, incentivizes builders by promising users fair MEV extraction and charges a business fee. The MEV-Boost relay has incomplete long-term trust assumptions because it is centralized. Again, as MEV increases, Flashbots and other mining pools will be under internal pressure to cheat and extract MEV in a detrimental manner to the network. Yet, MEV-Boost is the solution most

widely adopted by miners with over 90% of Ethereum miners using Flashbots' centralized services and over $730 million worth of safe MEV extracted since 2020 [27]. Flashbots and Ethereum have both expressed that this is not a long-term solution and have requested the community to keep working on stronger privacy and incentive assumptions that will not create pressure for centralization in the long term. Flashbots aims to one day decentralize its relay and provide true trustless MEV auction markets [28].

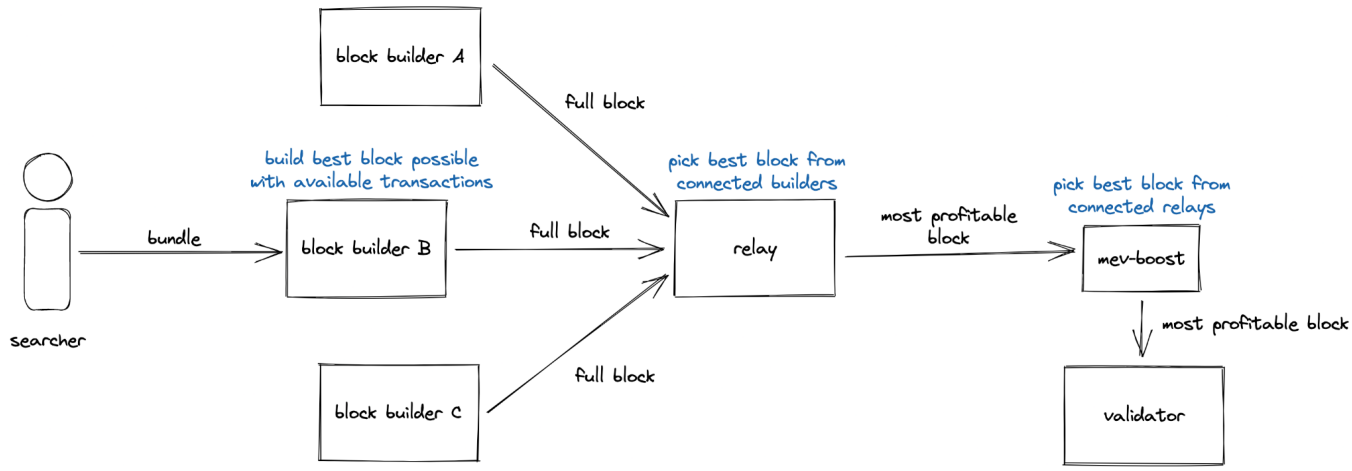


Fig 2. FlashBots Auctions and MEV-Boost [26]

There are other relays that use MEV-Boost design, not just Falshbots. The 7 major ones are Flashbots, BloXroute Max Profit, BloXroute Ethical, BloXroute Regulated, BlockNative, Manifold and Eden [29]. This further decentralizes MEV-Boost and can make a difference when centralized relays must comply with political decisions, such as economic sanctions that force relays to not include certain addresses, that are imposed by one country. Censorship is another problem that arises from the centralization of MEV extraction solutions.

> “⚠️ Not all builders can be trusted ⚠️ [...] The current version of Flashbots Auction contains technical limitations which prevent the network from operating in a fully trustless manner. Specifically, the properties of complete privacy and permissionlessness are required for Flashbots to be fully decentralized.” [28] - Flashbots

Price Guarantee Reward. In the PBS model, proposers can be assumed friendly, because they do not get to see the private transaction data and some MEV is being withheld in escrow by the builder or relay. The builder or relay cannot validate their blocks and collect their MEV extraction profits without sharing those profits with the proposer. One of the reasons a centralized relay is necessary is because of a lack of direct incentive for the builders and proposers to protect users from malicious MEV extraction and the network from forking attacks. An incentives-based scheme has been proposed [23], it relies on a user broadcasting the worst and best price they will accept for sending their transaction. This worst and best price is commonly known as slippage.

Simply by letting users publicly express and verify each other's slippage preferences, a users transactions market is created and builders must compete to get the rights to a transaction’s MEV, thereby making MEV profits eventually flow down to users as well. Once the user’s wanted slippage is attained and promised by a builder, the user can offer a percentage of the slippage they are given as a tip to make their transaction have a higher chance of being included first in the block. Once the user and builders have agreed on a slippage and reward, the transaction is sent. This reward scheme theoretically gets rid of the relay, since builders are now incentivised to provide the best price for the user, which includes protecting them from malicious

MEV extractions by hiding the transaction from the proposer, while conducting safe MEV extractions such as arbitrage. This was proposed this year, and it is yet to be implemented.

This reward scheme provides similar properties as PBS which are (1) block withholding, i.e. block builders have no incentives to withhold a block because they can not validate it themselves, (2) trust minimization, i.e. proposers and builders have different income streams and no conflict in splitting the total MEV and (3) fair ordering preference, in the sense that users get the best price that they have specified and they can see competing prices. This reward scheme for MEV sharing does reduce trust assumptions amongst participants while allowing for protection from malicious MEV and profits to flow down to offer the best price for the user, however miners can still have the ability to hide their MEV extraction activity from the users market as there is no cryptographic certainty that the miners will share their MEV extraction activity with the public. More research is needed to confirm that there are no downsides to this reward scheme and that it can successfully be implemented by just users, builders and proposers (below called Miner(s)), without the need for a relay or MEV minimization techniques.

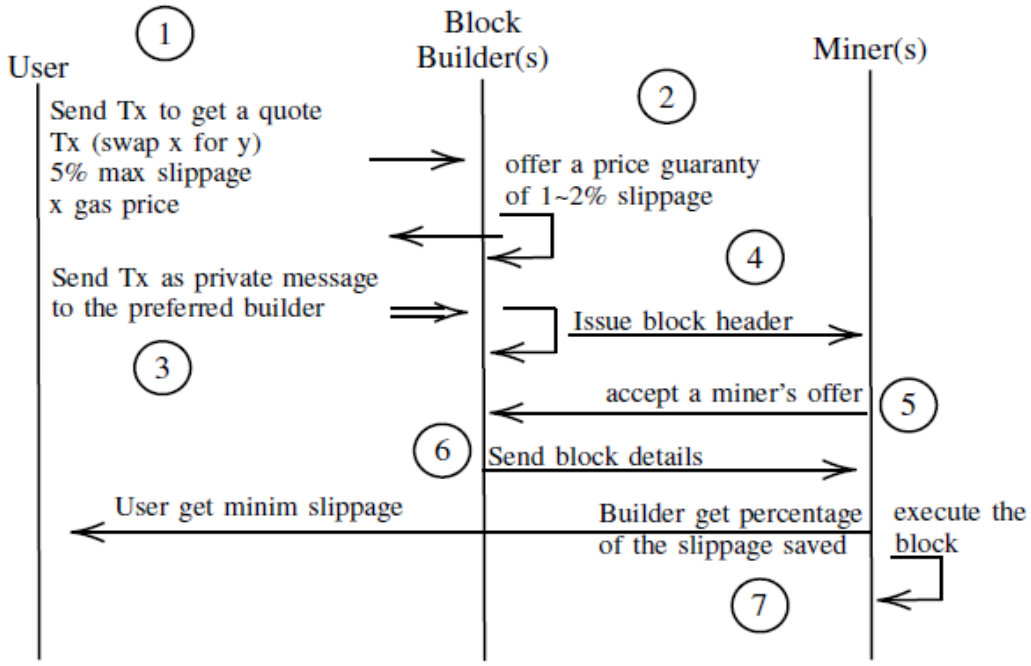


Fig 3. A price-reward scheme on top of the PBS model, without a relay, by Pillai, Babu (2022) [23]. "*Transaction flow on PBS model with price-reward scheme. The steps are: 1) user shopping around for better price, 2) a builder make an offer, 3) transaction submitted, 4) block builder issue block header, 5) a miner accept that block, 6) reveler block details 7) block get included and reward paid to miner and block builder.*"

MEV Minimization

The solutions discussed so far have been about sharing good MEV profits and either incentivising builders and proposers to play fair, or using a central relay to do this. All the solutions that rely on simply extracting the MEV and sharing it, in the long term, fall to centralization pressures as total MEV in the system rises. They are first attempt solutions that buy us time but forking attacks are increasingly a threat to Ethereum as the network grows. Overall, some users may want stronger transaction privacy guarantees, because they might not

want miners snooping into their various market moves and they would rather put their trust in a stronger privacy guarantee. To truly minimize MEV is to mathematically prove the privacy of transaction details, until after they have been committed, therefore preserving order fairness and minimizing malicious MEV attacks.This leads to the need for MEV minimization.

Order Fairness. In blockchains, order-fairness is the idea that transactions should be ordered in such a manner that a participant in the system can no longer take advantage over others [20]. Blockchains like Ethereum do not specify any transaction ordering. Some MEV minimization solutions that specify rules for miners on how to achieve order-fairness have been proposed, such as receive-order fairness [21], hybrid consensus [30] and fair committee election [31]. Receive-order fairness requires trusted client-side timestamps, which could in the future be provided by Trusted Execution Environments, but not currently. Fair committee election proposes a committee that runs an agreed-upon sequencing protocol to order transactions and a master service monitors the committee's progress and initiates reconfiguration when needed and removes participants from the committee before achieving consensus.

There are many live implementations of MEV minimization solutions that focus on order fairness, some of them are Conveyor by Automata and Fair Sequencing Service by Chainlink [32]. The issue with these services is that the transactions are still visible to the miners, so if MEV were to be high enough, they would have incentive to collude or break their promise, thereby opening the gates for a network attack.

Mempool Privacy and Commit-Reveal Schemes. The mempool is a collection of nodes that first receive transactions from users. It is within the mempool that builders search for transactions to create bundles. Currently, Ethereum transactions and smart contracts are visible in the mempool for anybody to view. To make the mempool private is to send encrypted transactions. Considering that blockchains maintain some form of a public ledger representing every transaction that occurs, the transactions must be decrypted eventually, to ensure wallets have the amount they claim to have. Decrypting these transactions in a fair manner, after they have been effectively committed, can reduce the unique opportunity for a miner to order these transactions for malicious MEV purposes, thereby ensuring order fairness. This idea is called a commit-reveal scheme.

Trusted Hardware. Trusted hardware is designed to provide a separate encrypted computing and memory unit called an enclave. This enclave can be used to generate keys, encrypt and decrypt data, run functions in an encrypted environment and return the results. Trusted Execution Environment (TEE) is hardware that isolates data and computation from the operating system and any other system, such that the host cannot observe and manipulate the data or the function.

Secret Network is an implemented Layer 1 solution that allows for encrypted smart contracts called Secret Contracts or SNIP20 [33]. It offers a transaction privacy guarantee by requiring validators to prove that they are storing decryption keys and executing encrypted contracts inside a TEE. This means the data is stored and processed in a tamper-proof and trusted, physical environment. Specifically, Secret Network leverages Intel's state of the art

Software Guard Extensions (SGX) as a TEE. Ethereum users can conduct Secret Contracts through an Inter-Blockchain Communication (IBC) protocol, so Secret Contracts are not limited to the Secret Network.

Here is a list of compiled theoretical attacks on TEEs in general, and therefore the Secret Network:

1. Deanonymizing with ciphertext byte count
2. Two contracts with the same contract key deanonymize state
3. Transaction replay attack
4. Search-to-decision millionaire problem
5. Partial storage rollback during contract runtime
6. Transaction output data leakage
7. TEE vulnerability enabling a Byzantine node to acquire the consensus seed from the enclave

As you can see the list includes all the SGX related vulnerabilities, so here is a list of SGX-Specific Attacks:

1. Prime+Probe attack
2. Spectre-like attack
3. Enclave attack
4. MicroScope replay attack
5. Plundervolt
6. Load Value Injection
7. SGAxe
8. ÆPIC leak

Memory access-patterns can reveal information about encryption keys or about a function being executed in a trusted enclave. Automata implements Oblivious RAM (ORAM) algorithms to provide a high degree of privacy to Dapps. Automata has released multiple credible research papers on the subject [34], they propose PRO-ORAM, a better ORAM algorithm that allows for constant time for read-only data access patterns, instead of the usual logarithmic time.

These hardware solutions often require trust in a centralized entity that manufactures these trusted hardwares, and some users might not like that. As MEV grows, so too will the collective incentive to break SGX and other trusted hardwares. The problem with Secret Network and its trust in TEEs is that Intel's SGX hardware is a blackbox hardware and its security could one day break either from inside Intel's designing and manufacturing process or from outside attackers. There is a large surface of attack for trusted hardware, such as side-channel attacks, fault attacks and software attacks, especially considering the adversaries in this case would have possession of the hardware. The difficulty of acquiring the skills and the lack of incentive to break the hardware is what is currently stopping validators from breaking the trust guarantees of these technologies, instead of cryptographic certainty and strong economic incentives. What would be required for a hardware based solution is trustworthy computing, but we are far from this achievement.

Open sourcing and decentralizing trusted hardware, or their privacy guarantees, is an area for future research. Algorithms that attempt to simulate these hardware solutions are useful and do not rely on a centralized manufacturing entity, however, they add more computational cost

than their hardware counterparts. Our solution wants to avoid the requirement of TEEs and ORAM, instead we want to rely on classically cryptographic techniques to hide transaction data from validators. This is not just because there are solutions already implementing these hardware requirements, but also because in the long term, as MEV increases there is an increased incentive to break the hardware, which would be in the validator's possession.

Malicious actor(s)	MEV-Boost	Builder Release	Bonded fraud proof	ZKP	SGX
Relayer	Conceals block from proposer; Can steal MEV	Conceals block from proposer	Conceals block from proposer	Conceals block from proposer	None
Proposer	Can steal block if (MEV>cost of slashing)	None	Can steal block if (MEV>cost of slashing)	Can steal block if (MEV>cost of slashing)	Can steal block if (MEV>cost of slashing)
Builder	None	Can cause loss of block reward	Can accept slashing to cause empty slot by submitting invalid encrypted block	None	None
Relay + Proposer	None	None	Can steal MEV	None	None
Builder + Proposer	None	None	None	None	None
Builder + Relay	None	None	None	None	None

Fig 4. Comparing current MEV solutions [35].

Timelock Cryptography. Timelock cryptography allows decryption of a transaction after a certain period of time has passed. This relies on Verifiable Delay Functions, which is a cryptographic method that requires a function to have T sequential steps that can be timed and its output must be verifiable by running the invert of the function. A timelock decryption approach allows for order fairness, as encrypted transactions are pre-committed, and decrypted at the same time by all miners. Timelock cryptography is facing hardware challenges as well, since time

depends on a computer clock which can be physically hacked. The delay placed on decrypting transactions would rely on probabilistic measurements of the time a function should take, and this can result in major delays at a large enough scale as every transaction would have to go through timelock decryption, or risk being attacked.

Threshold Decryption. Threshold cryptography allows clients to share anonymous encrypted messages in a manner that allows decryption using a private key that is constructed from a threshold number of key shares held by separate, non-colluding and non-compromised nodes [36]. Private key shares must be privately generated, shared and combined, and only at a threshold t shares will the key be complete and the message decrypted by those t shareholders. t is usually 2/3rds of the node participants.

Many threshold decryption based protocols and blockchains have been proposed as solutions for MEV. Pesca uses fully homomorphic encryption, distributed key generation (Shamir secret shares), dynamic proactive secret sharing (DPSS) and zk-SNARKs for setting up the encryption and threshold decryption of the user's transactions and smart contracts [37]. Differentially private Constant Function Market Makers (CFMM) and a First Price Seal Bid Auction (FPSBA) application were built on top of Pesca, thereby proving they could make some aspects of DeFi private and useful. They identify threshold fully homomorphic encryption and zk-SNARKs frameworks and compilers as areas of future improvement. Both Pesca and Ferveo [37, 38] identify distributed key generation as a bottleneck in the efficiency of a threshold decryption based protocol. Ferveo also identifies that any commit-reveal scheme based on threshold decryption lacks reportability and penalties for validators that fail to follow the

protocol by colluding and releasing their shares early. Osmosis' Penumbra and Zcash are live implementations of threshold cryptography to achieve private transactions via commit-reveal [39].

Our solution chose threshold cryptography as a core component because it is the technique with the most implementations that allows for privacy-preserving transactions and commit-reveal schemes. It has few additional assumptions such as extra encryption and decryption steps that are negligible compared to other techniques. Threshold cryptography also relies on cryptographic proofs, instead of trust. Its one downfall is the trust assumption that 2/3rds of validators are honest and non-compromised, which is similar to normal blockchain security assumptions. Threshold decryption alone does not address miner's incentives to share MEV, and so we use PBS to address this, alongside the Price Reward scheme to align miner's incentives and flow MEV back to users and get rid of a centralized relay.

Solution method	Security cost	Delay	Additional Assumptions
TEE Encryption	Cost of breaking TEE	Block Finality	TEE manufacturer honesty
Timelock Encryption	Timelock hardware	Many blocks	All txs delayed
Threshold Encryption	2/3rds of validators	Block Finality	BFT Proof of Stake

Fig 5. Comparing privacy-preserving techniques. [38]

Approach & Methodology

Background Research and Literature Review

Background research on miner extratable value was done, including the history of miner extractable value - that in proof-of-work, miners control transaction inclusion, exclusion, and ordering and by doing so they found ways to profit from pending transactions [7] - different kinds of MEV opportunities like decentralized exchange arbitrage, liquidations, sandwich trading, and the development and the current state of MEV that till December 2022, Flashbots estimates that over 685-million-dollar MEV has been extracted since the beginning of 2020 [5].

We also performed literature review on current solutions that try to resist MEV attacks, including MEV extraction solutions like Proposer Builder Separation Model and MEV-Boost, Mining Pool Services and DEX Aggregators, Price Guarantee Reward, and MEV minimization solutions like Order Fairness, Timelock Cryptography, Threshold Decryption, and Trusted Hardware, with all of their mechanisms studied and features analyzed, as in the Existing Scenarios Scanning & Analysis section.

Solution Design

After comparing and analyzing the current solutions, we proposed our own solution by improving upon Proposer Builder Separation and combining with the improved Commit-Reveal Scheme to achieve MEV attacks resistance with high privacy-preserving degree, usability, and efficiency. The detailed solution design is introduced in the Solution Design section.

Demo Development and Testing

Initially, our plan was to develop runnable demo codes for the solution we proposed and run through previously mined blocks to calculate figures like successful MEV transactions, successful and failed MEV transactions gas fees, extracted MEV split by types and roles, etc. to

test and prove the performance of our solution. After careful review of current solutions' source codes, for example source codes for Builder API [10], we found it unachievable to finish up a runnable code demo within the semester. The detailed research we did on demo development and our plan for future implementation are included in the Implementation section.

Evaluation

To systematically evaluate our solution and compare it with current solutions that mitigate MEV problem, we referred to the seven measures taken by Heimbach and Wattenhofer [6] to construct our evaluation matrix – decentralization, security, scope, jostling, goodput, delay, and cost.

Decentralization measures "the approach's impact on the blockchain's decentralization", security measures "how susceptible the approach is to attacks", scope measures "how wide-reaching the approach is", jostling measures "the competition between traders for block inclusion", goodput measures "whether the approach impacts the number of genuine transactions processed by the application or blockchain per time unit", delay measures "the time between a trade submission and its execution", cost measures "the additional cost the approach places directly on traders for executing their transaction" (Heimbach and Wattenhofer [6]). Together we examined our solution and compared it with current solutions by these seven standards, to show its better performance on MEV mitigation and great reaction to all sorts of possible security vulnerabilities. Here we chose Proposer-Builder Separation and Builder API for evaluation and comparison, because they are the two solutions proposed to reduce the negative impacts of MEV after The Merge [7].

Solution Design

Rationale

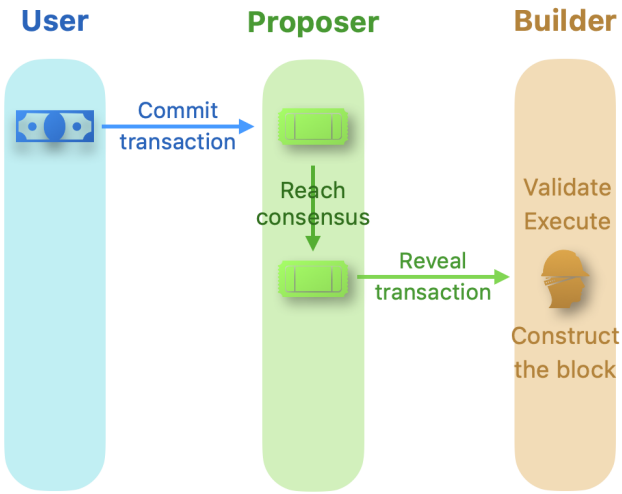
To address the MEV issues above, in our research, we will look into different kinds of privacy-preserving techniques like commit-reveal and optimize them to enhance their usability, efficiency, and reliability. On the other hand, by improving PBS and combining it with optimized privacy-preserving techniques, we make PBS MEV resistible, and improved PBS can better implement optimized privacy-preserving techniques.

For a better commit-reveal scheme, we design a new commit-reveal scheme with improved Proposer Builder Separation (PBS) that optimizes the whole on-chain process. Proposers reach consensus together on the encrypted transactions users commit first, then decrypt and reveal them for the builder to verify, execute and append on chain. With corresponding technical evaluation and economic analysis, we will prove that it will resist MEV better and benefit users, proposers, and builders.

In our commit-reveal design, users send encrypted transactions to proposers. Based on the improved PBS, proposers will first accept and reach consensus on them. They can't do any MEV attacks because the content of transactions is hidden. After proposers confirm which transactions will be put on chain, those transactions can be revealed and sent to the builder for validation and execution. Knowing the details of transactions after being revealed, builders can process them at lower cost and better efficiency, but they can't extract MEV because their ordering and content have already been determined by proposers during the commit phase. In this way, users could be protected from MEV attacks when proposers commit their transactions first for consensus, and maintain transparency and usability when builders reveal their transactions to process and put them on chain.

Design

The process of putting transactions on chain using our scheme is:



1. Users commit their transactions via cryptography like threshold signature, and send to proposers;
2. Proposers reach consensus on what transactions should be included in the next block and in which order via PoW or PoS;
3. Builders can reveal transactions, validate and execute them according to proposers' consensus to construct the new block.
4. Proposers verify the new block and append it on the chain.

By improving and combining PBS and commit-reveal, our solution aims at enabling usable, efficient, and reliable on-chain service that protects users' interests from MEV and regulates proposers and builders without privileges. To the best of our knowledge, our work is the first to:

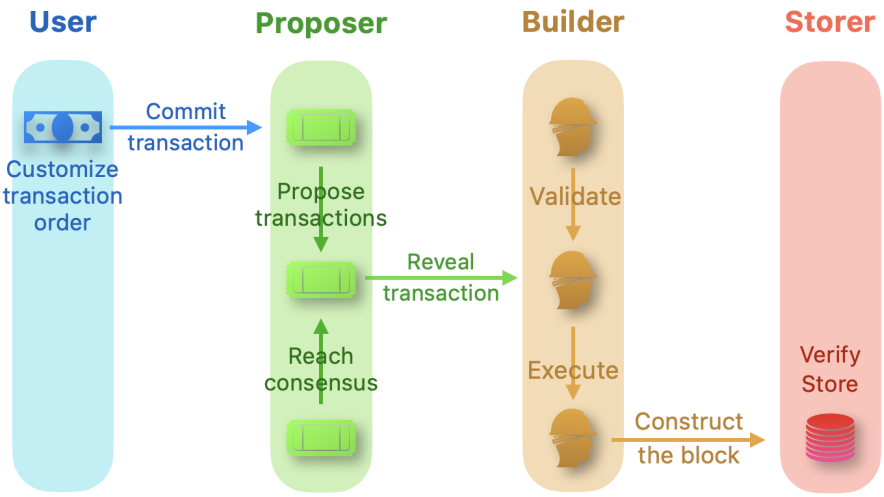
1. improve PBS including applying commit-reveal technique into PBS to make PBS MEV resistible.
2. improve commit-reveal technique with improved PBS for better MEV resistance in usability, efficiency, transparency, and reliability.
3. optimize the on-chain process to balance the relationship among users, proposers, and builders and avoid anyone having privileges for maximizing their self-interests but harmful to others within the blockchain.
4. design the incentive mechanism to make our solution benefit every participant on blockchain including users, proposers, and builders.

Improvement

Furthermore, each builder and proposer should have equal opportunities to co-determine the content and ordering of transactions within the block. Each builder can submit any transactions with MEV they extract to form a block for proposers to choose, and each proposer can vote for different blocks builders propose based on their voting rights like the number of stakes in PoS. In this way, the content and sequence of transactions within each block on the blockchain will be proposed and decided by all builders and proposers together rather than any single builder and proposer.

As for order fairness [21], it's quite subjective that different users and applications have different definitions and understandings about it. Some may think transactions should be ordered by time, while others may favor ordering by transaction amount or user level. Therefore users and applications should be given the right and freedom to order their transactions in their own ways according to their different kinds of actual demands, and builders and proposers should support their ways of ordering for their order fairness.

In this way the on-chain process could be:



1. Users submit their transactions to proposers. They can optionally commit their transactions to not be revealed until confirmed by proposers for MEV resistance. They can also customize other on-chain requirements like the ordering of their transactions.

2. Proposers co-determine the transactions within the block and reach consensus on them by voting for the transactions they accept, and transactions getting enough votes will be included in the block constructed by builders and put on chain by storers.

3. Builders receive and follow the proposed transactions agreed upon and confirmed by proposers. They validate and receive those transactions and construct the block for them accordingly. Different builders can process different transactions at least one and combine them to construct the whole block in the end. They can reveal and view committed transactions to better process them but are unable to change any of them within the block.

4. Storers receive the blocks constructed by builders and check their validity. If valid they will put them on the blockchain and update the states of the blockchain. Different storers can keep different pieces of blockchain according to their storage capacity and everyone can retrieve

and check them. They can also optionally provide extra storage for the original block content including transactions and data if users have such demands and pay for them.

In this way we can better achieve:

1. Decentralized on-chain process: No single builder or proposer has privileges in self-controlling and self-censoring transactions within the block including their content and orderings, which should be co-determined by every builder and proposer equally and collectively.
2. Self-determined transaction sequence: Users and applications can propose their own requirements and preferences like the way of ordering their related transactions, and builders and proposers need to satisfy their customized needs.
3. MEV attacks resistance: Builders and proposers should be regulated and users' transactions should be protected to avoid builders and proposers knowing the content of transactions and extracting MEV on them before putting them on the chain.

Evaluation Metrics

Evaluation metrics for Proposer-Builder Separation, Builder API, and our proposed solution are constructed using seven standards - decentralization, security, scope, jostling, goodput, delay, and cost in the format of spider web graphs, as mentioned in Methodology. In that way, we hope to quantify the performance of the current solutions and our solution for a clear comparison.

Proposer-Builder Separation

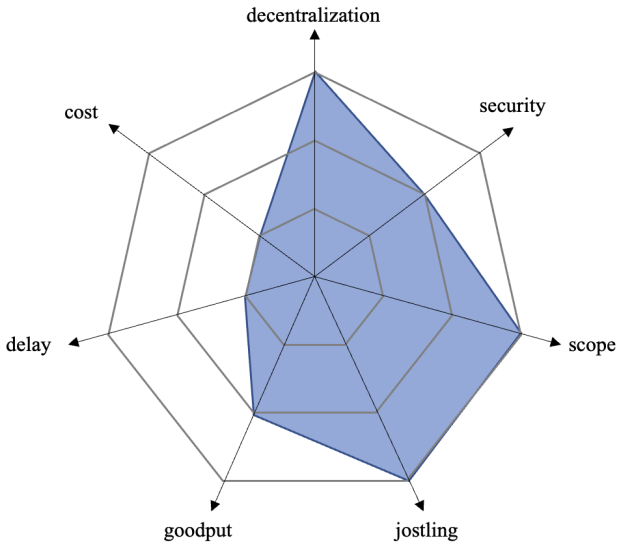


Figure 1: Evaluation Metrics of PBS. The spyder web's inner level represents the lowest score and the outer layer the highest score [6].

We find that PBS does not impact decentralization negatively, but rather, it promotes decentralization as it minimizes the compute overhead required to become a validator and therefore lowers the barrier to entry as a validator and incentives a more diverse group of participants [8]. PBS makes it more difficult for proposers to censor transactions or manipulate transaction order, and therefore improves security [9]. Attacks like front-running and sandwich attacks would be minimized since it's now hard to preview users' transactions, censor, or reorder

them. While PBS does democratize MEV among block builders, it does not prevent blockers from performing malicious MEV on users' transactions.

In current PBS designs, builders only publish a cryptographic commitment to a block header along with their bids [7]. As transaction contents are encrypted, ideally, we expect minimal jostling. A drawback of the solution is that the state of a decentralized exchange, relying on commit-revel schemes for ordering, is unknown to users when they submit transactions, and we would expect an increase in transaction failures due to price movements and therefore a reduced goodput [6]. Delay in transactions is unavoidable, given that the main feature of PBS is the separation of block producer and block proposer roles, and the "auction" leads to delay. Introductions of new roles also bring new costs, like the proposing fee paid to proposers, and the increased transaction fee caused by additional space needed on-chain, so PBS cannot be highly rated with respect

to cost.

Builder API

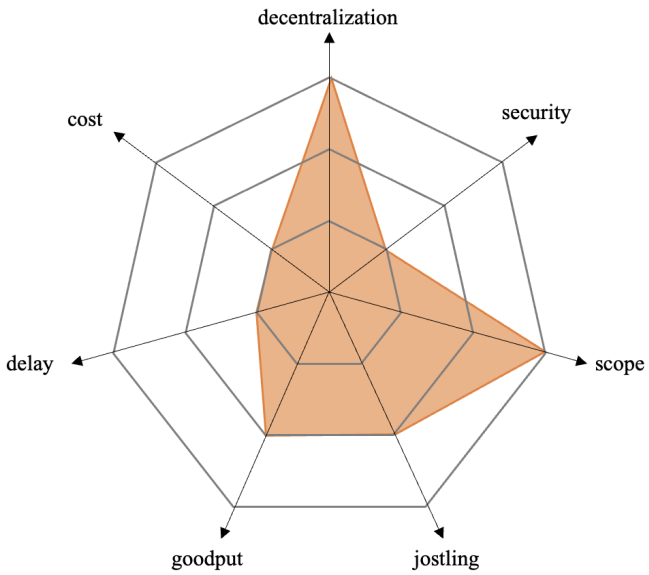


Figure 2: Evaluation Metrics of Builder API

Builder API is essentially a temporary solution aimed at providing a working implementation of PBS without changing the consensus protocol [7]. Its evaluation will be very similar to that for PBS. Builder API also contributes to decentralization with its potential to democratize access to MEV opportunities - by employing commit-reveal schemes, it also reduces entry barriers for validators seeking to benefit from MEV [7].

The main difference between PBS and builder API would be their security. While PBS requires modifications to the Beacon chain and could not be implemented at The Merge, Builder API provides the proposer with a blind execution layer header to incorporate into a block and a value amount which will be transferred to the proposer once they create a block with the header [10]. With more information provided to proposers, we have higher trust assumptions that proposers will be bound to their choice once they sign a block with the header, and higher trust assumptions bring lower security levels.

Although more information is provided to proposers, bids from builders will only contain a cryptographic commitment to the payload's contents and a fee to be paid. Ideally the Beacon block proposal created should be blinded, so there should not be much increase in jostling. However, while the Builder API acts as a middleware between validators and executors, the extra steps it brings to the transaction still make the state of a decentralized exchange unknown to the traders at first, and the transaction failures and low goodput are still expected. Also, serving as a working implementation of PBS, Builder API does not solve the extra delay and cost brought by PBS.

Our Solution

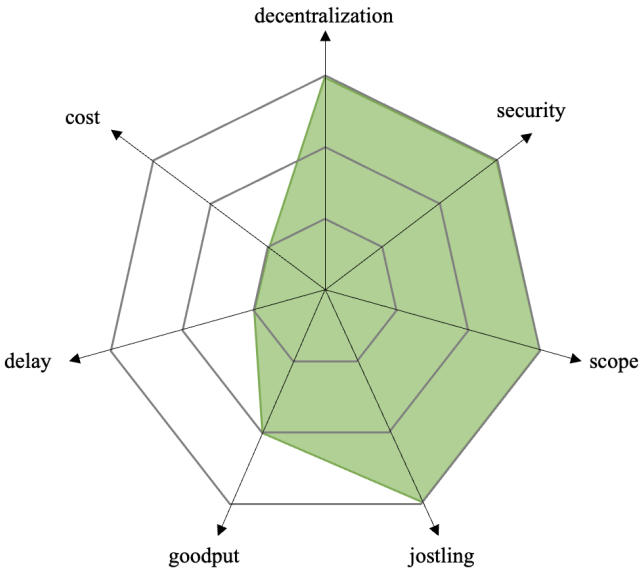


Figure 3: Evaluation Metrics of Our Solution

Our solution would be an improvement of PBS and it also employs commit-reveal design and thus will not lose its privilege in decentralization. Still dividing the roles into users, proposers, and builders, our solution continues to allow different actors to specialize on their

own strengths and tasks, which could lead to a more capable network with fewer external dependencies [8].

Our solution aims to achieve security and preserve privacy, in a way that proposers will first accept and reach consensus on the incoming encrypted transaction, during which MEV attacks should be greatly avoided because the contents of transactions are hidden to them. Contents of the transactions shall only be revealed and sent to builders for validation and execution after the proposers confirm and reach consensus on the transaction to be put on chain, but builders won't be able to extract MEV from them since the transaction order and contents have already been determined by proposers during the commit phase.

As for jostling, just like PBS, our solution keeps the transaction contents hidden so we still expect minimal jostling to happen. Similar to PBS, however, since our solution also relies on the Commit-Reveal schemes, price movements brought by it cannot be avoided and the increased transaction failures and reduced goodput are not solved. Also, while privacy-preserving techniques bring security, it also causes delay in transactions and increased cost. By revealing the details of transactions to builders after consensus has been reached, we expect builders to process them at lower cost and better efficiency, and therefore, there will not be a significant increase in delay or cost.

Comparison and Discussion

Comparing figure 1 and 3, we would like to prove that our solution is secure and efficient, and essentially an improvement of PBS without significantly worsening its drawbacks like delay in transaction and extra cost. By improving and combining PBS and commit-reveal, our solution aims at enabling usable, efficient, and reliable on-chain service that protects users'

interests from MEV and regulates proposers and builders without privileges. To the best of our knowledge, our work is the first to:

- improve PBS including applying commit-reveal technique into PBS to make PBS MEV resistible.
- improve commit-reveal technique with improved PBS for better MEV resistance in usability, efficiency, transparency, and reliability.
- optimize the on-chain process to balance the relationship among users, proposers, and builders and avoid anyone having privileges for maximizing their self-interests but harmful to others within the blockchain.
- design the incentive mechanism to make our solution benefit every participant on blockchain including users, proposers, and builders.

Limitation/Challenge

Plan	People	Achievements	Limitations & Challenges
Research & Analysis	David Mberingabo	Scanned and compared existing schemes.	No specific analysis on their effectiveness of MEV mitigation or resistance
Solution Design	Jieliang Yin	Ideally can combat MEV attacks in a better way	Need to change the overall on-chain process and structure
Development & Implementation	Ethan Wang	Made a plan on how to develop and implement in the future	No work on real development and implementation yet
Testing & Evaluation	Zili Zhou	Learned and drew from existing evaluation metrics	Not applying them for analyzing our proposed solution yet

In the research and analysis section done by David, although we scanned and investigated current MEV solutions in both academia and industry, we mainly focused on what techniques they used but didn't relate them to MEV specifically like how much MEV they can deal with, what kinds of MEV issues they can address.

In the solution design part done by Jieliang, although we put forward our MEV solution which can ideally better regulate miners and help mitigate MEV attacks for users via privacy-preserving techniques, it changes the whole on-chain process and structure which may need a huge amount of work.

As for the development and implementation done by Ethan, although initially we expected to develop a demo to implement and test our solution in practice, finally we just designed a plan on how to develop a demo for the solution we designed in the future but had no actual work on any realistic development and implementation in the end.

To evaluate the effectiveness of our solution done by Zili, although we found and learned several metrics and indicators used in other similar evaluation work from existing literature, we didn't really apply them in practice to test our solution and evaluate how much MEV we can address and how many kinds of MEV attacks we can prevent.

Future Work

Implementation

Overview

In order to fully demonstrate and evaluate our solution in the real world, we would need to create a space with a large enough user base to allow competition. This way, we could actually measure the amount of MEV extracted, as well as the fairness of MEV extracted and compare it to other protocols like MEV-Boost.

Unfortunately, getting enough users to simulate a competitive environment or building a simulation that approaches a real world environment is out of the time frame and scope of our project. Writing a program that is able to submit transactions to a blockchain would not be useful unless there were enough nodes to simulate competition. However, in this section we will discuss and compare different ways of how we could implement this solution in the future.

Security Concerns

The integrity and security of blockchain relies on the reliability of cryptography used and the engineering rigor required to ensure the cryptographic functions are applied correctly. We recognize that getting cryptography and security right is a difficult task, and in our implementation we would try to keep our protocol as simple as possible and only use vetted, publicly available cryptography libraries.

Our Implementation Goal

Efficiency and ease of implementation will determine the infrastructure, libraries, and coding languages we use. There is a trade-off between developing in simpler but slower languages like python, or more complex but built for distributed applications languages like golang. If we were to implement our solution as a proof of concept for testing, we would focus on creating a simple, easy to validate solution in Python.

In the next section we will discuss two possibilities of implementation, either adding on to an existing blockchain like Ethereum, or developing a blockchain from scratch. While Python may not be the best solution for a fully deployed system, there are many publicly available libraries for cryptography like hashlib[2] and APIs for communicating with Ethereum like Web3.py[3] which will make implementing and validating our solution easier.

Implementation as an Extension of an Existing Blockchain

MEV-Boost as an Example

MEV-Boost operates as a sidecar to the Ethereum validator stack. This means that MEV-Boost uses an existing blockchain. It does not alter the blockchain's validation process. Rather, it outsources block building and implements proposer-builder separation on top of the existing block chain. In this case, users are not required to use MEV-boost, but some are incentivized to do so as MEV-boost can provide more equitable MEV extraction. MEV-Boost is a great example of how one can implement a different proposer/validator scheme on top of an existing blockchain.

Implementing our Solution on Ethereum

If we were to take this route, we would follow a similar implementation strategy as MEV-Boost. As we would not have to implement the blockchain ourselves, we just need to implement the middleware. We would use Web3.py to get transactions from the Mempool and submit transactions to the Ethereum network. The design of our own module and how it would interact with the Ethereum network is shown below.

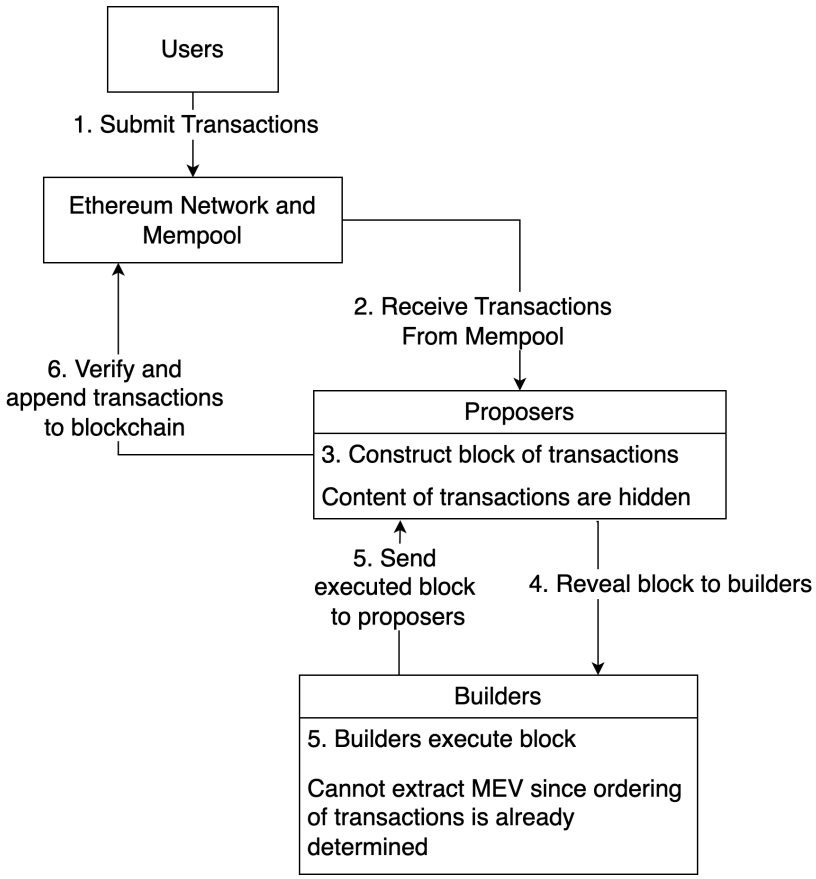


Figure 4: Solution Interface with Ethereum

Building a Blockchain From Scratch

It is relatively straightforward to implement a simple blockchain from scratch. There are many resources available[1] which demonstrate how this can be done. A simple solution is to use the Python hashlib library to create a proof of work system to verify blocks on the blockchain. The blockchain itself can be represented by simple objects in a list which are cryptographically linked. To enable communication between nodes, Flask[4] will be used to create a web application which uses HTTP requests to allow nodes to communicate. Once the simple blockchain is set up, we can implement our middleware solution on top of the blockchain.

Comparison of Implementations

There are both pros and cons to building a blockchain from scratch and implementing our solution on top of an existing blockchain. A benefit to building our solution on top of an existing blockchain is that the existing blockchain already has a solid user base. It will be relatively easy for users to use our middleware to provide a different way of interacting with the blockchain. Additionally, an already publicly established blockchain like Ethereum is constantly maintained and issues like security are taken care of. This will allow us to focus on developing and implementing our specific solution. However, since we cannot control the simulation environment, it can be more difficult to validate our solution.

A benefit to developing our own blockchain is that we have full control over our testing environment. We can perform smaller scale tests and it will be easier to validate our solution. However, it will be hard to simulate a competitive market of builders and proposers.

Given the pros and cons of both implementation strategies, we determined that building our solution on top of an existing blockchain would be the best route since the solution will be more robust and secure, and there is already a user space to simulate a competitive environment.

Collecting data

Flashboots MEV-Boost keeps track of data metrics relating to their service [5]. It monitors metrics like daily extracted MEV, number of successful MEV transitions, and extracted MEV split by type. In a similar way, we will collect these same metrics which will be discussed in the metrics and evaluation section. Then, we can use these metrics to compare and evaluate our solution in a real environment to MEV-Boost and other protocols.

References

[1] A. Ridwan, "How to create a simple blockchain using python," *Section*, 13-Oct-2021. [Online]. Available: https://www.section.io/engineering-education/how-to-create-a-blockchain-in-python/. [Accessed: 01-Nov-2022].

[2] "hashlib — Secure hashes and message digests — Python 3.8.4rc1 documentation," docs.python.org. https://docs.python.org/3/library/hashlib.html

[3] "Introduction — Web3.py 5.31.1 documentation," web3py.readthedocs.io. https://web3py.readthedocs.io/en/v5/

[4] "Welcome to flask," *Welcome to Flask - Flask Documentation (2.0.x)*. [Online]. Available: https://flask.palletsprojects.com/en/2.0.x/. [Accessed: 30-Nov-2022].

[5] "Flashbots Data Metrics," *MeV explore*. [Online]. Available: https://explore.flashbots.net/data-metrics. [Accessed: 30-Nov-2022].

[6] Heimbach, Lioba, and Roger Wattenhofer. "SOK: Preventing Transaction Reordering Manipulations in Decentralized Finance." *ArXiv.org*, 21 Sept. 2022, arxiv.org/abs/2203.11520.

[7] EridianAlpha, et al. "Maximal Extractable Value (MeV)." *Ethereum.org*, 7 Nov. 2022, ethereum.org/en/developers/docs/mev/.

[8] Blocknative. "What Is Proposer/Builder Separation (PBS) on Ethereum?" *What Is Proposer/Builder Separation (PBS) on Ethereum?*, 24 Oct. 2022, www.blocknative.com/blog/proposer-builder-separation-ethereum.

[9] Alchemy. “What Is Proposer / Builder Separation (PBS)? - Alchemy.com.” *What Is Proposer / Builder Separation (PBS)?*, 4 Aug. 2022, www.alchemy.com/overviews/proposer-builder-separation.

[10] Github Contributors. “Ethereum/Builder-Specs: Specification for the External Block Builders.” *GitHub*, Nov. 2022, github.com/ethereum/builder-specs.

[11] Lewis, Michael. Flash boys: a Wall Street revolt. WW Norton & Company, 2014. https://books.google.com/books?hl=en&lr=&id=UcIkAwAAQBAJ&oi=fnd&pg=PA1&dq=flash+boys&ots=wfQbsbXAyo&sig=YAEvhG3hx7jvbXyccxgpQwO5BMQ#v=onepage&q=flash%20boys&f=false

[12] Daian, Philip, et al. "Flash boys 2.0: Frontrunning, transaction reordering, and consensus instability in decentralized exchanges." arXiv preprint arXiv:1904.05234 (2019). https://arxiv.org/abs/1904.05234

[13] EridianAlpha, MAXIMAL EXTRACTABLE VALUE (MEV), 2022, https://ethereum.org/en/developers/docs/mev/

[14] PAT DANIEL, Miner Extractable Value: The Evolution of MEV in 2021, 2022, https://coinculture.com/au/tech/miner-extractable-value-mev-2021/

[15] MEV Wiki, 2021, https://www.mev.wiki/attack-examples

[16] vbuterin, State of research: increasing censorship resistance of transactions under proposer/builder separation (PBS), 2022 https://notes.ethereum.org/@vbuterin/pbs_censorship_resistance

[17] MEV Wiki, 2021, https://www.mev.wiki/attack-examples

[18] Upgrading Ethereum to radical new heights, 2022, https://ethereum.org/en/upgrades/

[19] Heimbach, Lioba, and Roger Wattenhofer. "SoK: Preventing Transaction Reordering Manipulations in Decentralized Finance." arXiv preprint arXiv:2203.11520 (2022). https://arxiv.org/pdf/2203.11520.pdf

[20] Baum, Carsten, et al. "Sok: Mitigation of front-running in decentralized finance." Cryptology ePrint Archive (2021). https://eprint.iacr.org/2021/1628

[21] Kelkar, Mahimna, et al. "Order-fairness for byzantine consensus." Annual International Cryptology Conference. Springer, Cham, 2020. https://eprint.iacr.org/2020/269

[22] "Minting Ether." Ethereum Organization. Last edited November 1, 2022. https://ethereum.org/en/developers/docs/intro-to-ether/#minting-ether

[23] Pillai, Babu. “Blockchain MEV minimisation solution with price guarantee reward.” October 24, 2022. TechRxiv. Preprint. https://doi.org/10.36227/techrxiv.21345306.v1

[24] Carlsten, Miles and Kalodner, Harry and Weinberg, S. Matthew and Narayanan, Arvind. "On the Instability of Bitcoin Without the Block Reward." 24 October 2016. https://doi.org/10.1145/2976749.2978408

[25] Gnosis. "Introducing Gnosis Protocol V2 and Balancer-Gnosis-Protocol." GnosisDAO, Medium. Apr 28, 2021. https://blog.gnosis.pm/introducing-gnosis-protocol-v2-and-balancer-gnosis-protocol-f693b2938ae4

[26] Flashbots. "Overview". https://docs.flashbots.net/flashbots-protect/overview

[27] BitMEX Research. "Flashbots." May 6, 2022. https://blog.bitmex.com/flashbots/

[28] Flashbots. "Trust Assumptions." https://docs.flashbots.net/flashbots-auction/overview#trust-assumptions

[29] Agnihotri, Anish. "Top relayers." MEV-Boost. https://www.mevboost.org/

[30] Pass, Rafael and Shi, Elaine. "Hybrid Consensus: Efficient Consensus in the Permissionless Model." 31st International Symposium on Distributed Computing (DISC 2017). Date of publication: 12.10.2017. http://dx.doi.org/10.4230/LIPIcs.DISC.2017.39

[31] Lev-Ari, Kfir and Spiegelman, Alexander and Keidar, Idit and Malkhi, Dahlia. “FairLedger: A Fair Blockchain Protocol for Financial Institutions.” 31st International Symposium on Distributed Computing (DISC 2017). Date of publication: 12.10.2017. https://doi.org/10.4230/LIPIcs.OPODIS.2019.4

[32] Juels, Ari and Breidenbach, Lorenz, and Tramèr, Florian. “Fair Sequencing Services: Enabling a Provably Fair DeFi Ecosystem.” ChanLink Labs. September 11, 2020. https://blog.chain.link/chainlink-fair-sequencing-services-enabling-a-provably-fair-defi-ecosystem/

[33] “Secret Network: A Privacy-Preserving Secret Contract & Decentralized Application Platform.” Secret Network. https://scrt.network/graypaper

[34] C. W. Fletcher et al., "A Low-Latency, Low-Area Hardware Oblivious RAM Controller," 2015 IEEE 23rd Annual International Symposium on Field-Programmable Custom Computing Machines, 2015, pp. 215-222. https://doi.org/10.1109/FCCM.2015.58

[35] “Approaches to complete privacy for MEV-Boost.” IC3 Ethereum Summer Camp. Ethresearch. August 14, 2022. https://ethresear.ch/t/approaches-to-complete-privacy-for-mev-boost/13376

[36] Boneh, D. et al. (2018). “Threshold Cryptosystems from Threshold Fully Homomorphic Encryption.” In: Shacham, H., Boldyreva, A. (eds) Advances in Cryptology – CRYPTO 2018.

CRYPTO 2018. Lecture Notes in Computer Science(), vol 10991. Springer, Cham. https://doi.org/10.1007/978-3-319-96884-1_19

[37] Dai, Wei. “PESCA: A Privacy-Enhancing Smart-Contract Architecture.” Bain Capital Crypto, Cryptology ePrint Archive. August 29, 2022. https://eprint.iacr.org/2022/1119