Before we get started:

Dear e-mail subscriber, due to the length of this post, it is possibly truncated by your e-mail client! If things end abruptly, take a look at the post online by clicking on the title. Thank you for reading :)

Proposer-Builder Separation (PBS) is a hotly debated topic, a broad design philosophy emphasising the relationship between protocol and non-protocol actors for the maintenance and operation of a blockchain. While the discussion has now been active for over a year, the Merge first and Devcon second brought much more resources and attention to PBS. In this post, I provide my own take on the state of PBS research, necessarily a partial view but one that I hope helps make sense of where we are and what remains to be done.

What is PBS?

Beyond a single mechanism specification or implementation, PBS is first and foremost a design philosophy, recognising that protocol actors may invoke services from third-parties in the course of their consensus duties. By “protocol actor”, we mean in Ethereum Proof-of-Stake any bonded and active validator, expected to perform duties ranging from proposing a block, producing attestations or participating in the current sync committee.

Pre-Merge: bundle searchers and mev-geth

To some extent, PBS started previous to the Merge and Ethereum turning to Proof-of-Stake. We consider

“default behaviour”the block producer behaviour which simply takes a set of transactions from their transaction pool and packs them as efficiently as possible in a block.

1

The first deviation from this default behaviour was observed as some miners opened direct, private lines of communication with transaction originators, who would agree off-chain on the conditions for inclusion of the transaction.

While such contracts were established privately, mev-geth, developed by Flashbots, soon after opened up a marketplace between block producers and transaction originators. The marketplace relied on a trusted relationship between miners (organised in PoW as mining pools) and entities known as searchers. Searchers typically obtain a user transaction from the public transaction pool or some private service they operate. The user transaction is

bundled

with the searcher’s own transactions to extract value from the user transaction, e.g., follow a user trade with arbitrage or sandwiching, or follow an oracle update with a liquidation procedure.

Bundles are communicated in the clear to marketplace participants, along with a promise of payment from the searcher to the block producer should the bundle be included. Yet the marketplace is restricted to known block producers, a whitelist of major mining pools, to prevent

bundle theft

. A block producer who replicates the searcher’s strategy, replacing the searcher’s payment with full payment to themselves, would be detected and subsequently kicked out of the marketplace, losing potentially substantial payments in the future. PBS here is expressed as block producers receiving bundles from searchers for inclusion at the top of their block.

Post-merge: block builders and mev-boost

After the Merge, and with the introduction of solo validators who are not joining staking pools, maintaining a whitelist for the marketplace proved infeasible. Solo validators infrequently propose blocks, in expectation once every two months with the current number of validators bonded in the protocol. This changes the nature of the game, from a game of repeated interactions with threats of punishment for deviations to a game with players who are often more “single shot”.

A solution to this constraint is to procure a commitment to a specific block content from a validator, without the validator gaining knowledge of what the content is. The marketplace is now known as

mev-boost

, and forwards to block

proposers

headers of blocks created by block

builders

, along with a bid made by the builder, a promise to pay the proposer some amount for choosing the block they made. At the

centre of the marketplace sit

relays

, who are responsible for checking the validity of the blocks made by builders, i.e., if the block is EVM-valid and indeed pays out the promised amount to the proposer.

To receive offers from builders participating in the mev-boost marketplace, validators run the mev-boost program alongside their consensus and execution clients. Validators select which relays to connect to, each relay enforcing its own conditions on the blocks they forward to connected validators (more on this later).

Towards “in-protocol” PBS

Faced with the prospect of relay failures, such as

validationor

liveness failures, as well as the opportunity to remove a new single point of failure from the system (inherently a centralising factor), “in-protocol” PBS

designs

were

introducedby Vitalik. In these designs, a validator once again blindly commits to use a block provided to them by builders. But instead of a relay brokering the agreement, the

protocol itself

provides assurances of two kinds:

- To the builder,

that a commitment was entered into by the proposer

, and that this commitment may only be reverted by a consensus failure, e.g., a re-org, or a safety fault with single-slot finality;

- To the proposer,

that

the builder’s promise to pay will be honoured

no matter what the builder does, e.g., eventually fail to release the block contents or release an invalid block.

To the builder,

that a commitment was entered into by the proposer

, and that this commitment may only be reverted by a consensus failure, e.g., a re-org, or a safety fault with single-slot finality;

To the proposer,

that

the builder’s promise to pay will be honoured

no matter what the builder does, e.g., eventually fail to release the block contents or release an invalid block.

PBS and the principal-agent problem

In a

recent podcastwith AltLayer’s Yaoqi Jia, I argued that instead of PBS being something we

need

, PBS appeared as a

constraint

to which the protocol had to react. The willingness of block producers to source parts of their blocks outside of the protocol, pre- or post-Merge, meant that protocol goals could become subverted.

Historically, it wouldn't be the first time. An important protocol goal is

reward fairness, or the ability for protocol actors to all receive in expectation the same amount of rewards, relative to their size. While this held true in Proof-of-Work, with rewards obtained proportionally to the hashrate brought to the network, the reward variance was high enough to incentivise most if not all miners to join mining pools, which share profits with their members. There is however no guarantee that the block made by the pool will respect the preferences of the pool's members, unless the block's construction is distributed across members of the pool. Remains the right to exit the pool if the pool acts against its members wishes, another reason why

withdrawalsare important.

Anytime protocol actors outsource some of their control over block construction, a new instance of the

principal-agent problemappears. In such a framing, the principal requires the agent to perform some action, yet the incentives of the principal and the agent may not be fully aligned. An employer wants their employees to perform their job as well as possible, but an employee wants to minimise their effort, so appropriate compensation structures help re-align incentives. A block proposer wants transactions to be included, but the builder may wish to censor these transactions, so mechanisms enforcing proposer preferences help re-align incentives.

Market structure and allocation mechanism

In my opinion, it makes sense to think of PBS in a modular fashion, to better appreciate its design space. This approach lets us consider more clearly the structure of the principal-agent problem, and how far the protocol intervenes to resolve the problem.

- PBS as a market structure/legal system.

The protocol defines the conditions under which proposers may engage with third-parties during block construction. For instance, the protocol provides enforceability of agreements, e.g.. enforces the promises to pay made by proposers or third-parties. The protocol also provides a facility for third-parties to recognise that a commitment was made by proposers. This facility comes from the protocol's consensus mechanism, which

makes safe

commitments, i.e., prevents the commitment from being reverted.

- PBS as an allocation mechanism/business logic.

The protocol defines the space of contracts proposers and third-parties may enter into. For instance, the protocol determines that only the whole rights of making the block can be sold, and organises the auction allocating these rights.

PBS as a market structure/legal system.

The protocol defines the conditions under which proposers may engage with third-parties during block construction. For instance, the protocol provides enforceability of agreements, e.g.. enforces the promises to pay made by proposers or third-parties. The protocol also provides a facility for third-parties to recognise that a commitment was made by proposers. This facility comes from the protocol's consensus mechanism, which

makes safe

commitments, i.e., prevents the commitment from being reverted.

PBS as an allocation mechanism/business logic.

The protocol defines the space of contracts proposers and third-parties may enter into. For instance, the protocol determines that only the whole rights of making the block can be sold, and organises the auction allocating these rights.

Under mev-boost, the protocol does not intervene into the principal-agent problem. It does not recognise the role of builders as an entity interfacing with proposers. At best, consensus layer clients provide the ability for stakers to blindly sign a block header, and perhaps additional services such as local profit-switching, yet neither of these are part of the protocol (to convince yourself, the word "builder"

only appears in preliminary "sharding" parts of the consensus-specs repo).

This non-intervention implies that there does not exist trust-minimised, in-protocol ways for proposers to be made whole when the other side of the market does not honour their agreements. Whether this is the responsibility of the protocol or not is entirely debatable. My views on this are plural:

- On the one hand, proposers seeking maximum profit may engage in risky behaviour, such as delaying block

publication or summoning external entities to help them create the block. The protocol may not care to backstop such activities, and in fact, may wish to prevent

moral hazardas much as possible.

- On the other hand, the asymmetry between block construction and block verification (discussed in Vitalik's

Endgame post)

does

open up design possibilities that cannot be realised without an appeal to external entities. If we intend to fully lean in to the PBS design philosophy, for instance requiring powerful builders to produce Danksharding blocks, then the requirement of interfacing with such third-parties will be imposed

by the protocol

upon proposers. It would be a rough bargain if some proposers weren't able to fulfil the duties required of them by the protocol (in particular, proposers in resource-constrained environments), while on top of this these proposers weren't compensated when they inevitably invoke third-parties to fulfil the duties on their behalf. At least making an execution block, even one that does not maximise extracted value, is at hand for all proposers. Making a Danksharding megablock is not, however.

If proposers are made whole in case of builder failure, what about risks to the protocol? If builders are less trusted to deliver on their commitments, we could end up with a protocol realising a much smaller social welfare compared to one where proposers are fully engaged to realise their duties. I am not as worried about this part. First, builders are profit-making entities, even in a competitive market. They care for their blocks to make it into the protocol, especially when the payment is committed and will happen regardless of their performance. Second, while it is a loss to the protocol when a block is missed due to builder failure, if only because of the greater real-world latency, it is always possible to "make up" for lost time, e.g., with an

accumulative, time-based EIP-1559.

On the one hand, proposers seeking maximum profit may engage in risky behaviour, such as delaying block publication or summoning external entities to help them create the block. The protocol may not care to backstop such activities, and in fact, may wish to prevent

moral hazardas much as possible.

On the other hand, the asymmetry between block construction and block verification (discussed in Vitalik's

Endgame post)

does

open up design possibilities that cannot be realised without an appeal to external entities. If we intend to fully lean in to the PBS design philosophy, for instance requiring powerful builders to produce Danksharding blocks, then the requirement of interfacing with such third-parties will be imposed

by the protocol

upon proposers. It would be a rough bargain if some proposers weren't able to fulfil the duties required of them by the protocol (in particular, proposers in resource-constrained environments), while on top of this these proposers weren't compensated when they inevitably invoke third-parties to fulfil the duties on their behalf. At least making an execution block, even one that does not maximise extracted value, is at hand for all proposers. Making a Danksharding megablock is not, however.

If proposers are made whole in case of builder failure, what about risks to the protocol? If builders are less trusted to deliver on their commitments, we could end up with a protocol realising a much smaller social welfare compared to one where proposers are fully engaged to realise their duties. I am not as worried about this part. First, builders are profit-making entities, even in a competitive market. They care for their blocks to make it into the protocol, especially when the payment is committed and will happen regardless of their performance. Second, while it is a loss to the protocol when a block is missed due to builder failure, if only because of the greater real-world latency, it is always possible to "make up" for lost time, e.g., with an

accumulative, time-based EIP-1559.

In any case, answering these questions would answer what Ethereum, as a protocol, is or should be. What does it require of its actors? What is its worldview with respect to the economic organisation around protocol duties? What are the boundaries of its concerns?

In the next few sections, I will discuss some of the mechanisms we might consider in our search for Ethereum's final form.

The space of allocation mechanisms

Keeping the market structure constant (e.g., in two-slot PBS world, the proposer commits to a builder bid and the builder reveals their part after seeing a safe commitment), the space of allocation mechanisms (who gets what) offers rich possibilities.

Whole vs partial block building

The initial goal of PBS, “proposer naivety”, drove the design of a whole block auction, where the proposer lets go entirely of their right to make the block. This whole block auction is the model used today under mev-boost. Yet as we’ve seen, relinquishing entirely the proposer’s input in block construction can lead to mis-aligned preferences over the contents of the block.

With the goal of allowing for proposer expression during the construction of the execution payload, we could determine that the proposer makes one part of the block, while the builder makes another part. It is fairly hard to conceive a block construction procedure where proposer- and builder-submitted transactions are intermingled, so we may restrict the design to two options: proposer prefixes and proposer suffixes.

Proposer prefixes

seem easy enough to work out. The proposer’s commitment could come with a block prefix, indicating to the builder

a)

how much gas they can still spend as well as

b)

the state root their block must build upon. The downside is that a builder only gets to know these two pieces of information as the proposer makes the commitment, unlike whole block auctions where this data is known as soon as the previous builder reveals their block. This can however be mitigated with the proposer communicating off-band the prefix they intend to make, as deviating from this prefix when making their commitment only hurts the profitability of the builder, who must readjust the block they are building.

Proposer suffixes

look more complex. If we intend the proposer to build upon the prefix provided to them by the builder, we need an extra protocol round to make safe the proposer suffix, after the builder’s prefix is revealed.

Vitalik suggested a different approach, where the proposer commits first to a list of transactions, which are appended to the end of the block in the order of the list, skipping any transaction already included by the builder and stopping when gas runs out. In his scheme, the proposer commits only to a root of the list, or some KZG commitment, so this may still require an extra step of communication if the proposer is expected to reveal the list eventually. Yet by committing to a list, the proposer has no space to extract MEV based on the builder prefix, so naivety is somewhat preserved.

Block vs slot auctions

This is an orthogonal axis to whole vs partial block building. Here we determine whether the builder must commit to block contents

before

making their bid, or

after

.

Block auctions

mean the builder has an exact view of the contents of their block when placing their bid. They know precisely how much they are extracting, and make the bid based on the extraction amount. This is the current model used by mev-boost, where the builder forwards their whole block to the relay, which forwards the header to the proposer. The header contains a block root, which does not allow for a different block to be revealed once the proposer’s signature is received.

Slot auctions

mean the builder is only bidding for the

right

to make the block, but not for any specific block. The proposer commitment only determines

which

builder is expected to release the block, but does not commit the builder to a particular block. This scheme may be more flexible, allowing for

“just-in-time” block buildingbased on the certainty that the right is owned by the builder, facilitating cross-domain MEV extraction. The downside of course is uncertainty over the true value of the block, once the state of the world is revealed, i.e., once the state of other domains is known. A builder may overpay, based on a wrong belief that they can extract more once they build the block. Could a

winner’s cursescenario take hold? It does not seem possible to continuously have actors in the system operate at a loss, so likely not, but ongoing research on the question is in progress.

Inclusion lists (f.k.a crLists)

A simple way to reassert proposer input into the block construction is simply to declare a block invalid unless it contains transactions which the proposer flags previous to the block being made by the builder.

The “straw man” inclusion list design invalidates the whole block if it does not contain all transactions flagged by the proposer. With this mechanism, a proposer could inadvertently reduce the profit obtained through the builder, when their inclusion list prevents the builder from packing more profitable transactions due to lack of space. This reduces the incentive for the proposer to add anything to their inclusion list, as the rational move (with max profit as a motive) is to keep the list empty.

To go around this, we could instead let a different proposer make the list, for instance the previous proposer makes the list for the current proposer. Let’s call such lists

forward inclusion lists

. With the straw man inclusion list, this opens the door to griefing, where the previous proposer packs the list and hurts the current proposer’s profit. But note that packing the list beyond what currently sits in the mempool is costly! The previous proposer cannot add phony, gas-consuming transactions to the list for free. EIP-1559 requires the transaction originator to pay at least the basefee to the protocol. Still, what sits in the mempool may be enough to force the current proposer to include a sub-optimal set of transactions in their block.

But we can use EIP-1559 to design a more flexible inclusion list mechanism, where, given an inclusion list

IL

, either

1. All transactions of

IL

are included in the block, or

1. The block must be full enough that no extra transaction from

IL

may be included.

All transactions of

IL

are included in the block, or

The block must be full enough that no extra transaction from

IL

may be included.

This way, adding a transaction to the inclusion cannot hurt the profit of the proposer: if the transaction participates to the max profit allocation given the gas limit constraint, it is included; if it does not, then the block must be full, since it is always better to include a fee-paying transaction than not.

The question is whether builders can continually censor a transaction by filling their blocks to capacity. Once again, EIP-

1559 prevents it. Blocks filling up increase basefee exponentially, until it becomes no longer possible to find enough transactions willing to pay basefee. In addition, the censoring builders stuffing their blocks incur exponential costs.

Censoring-best builders and relays

If making an inclusion list does not hurt your profit, is there still a reason for using forward inclusion lists rather than letting a proposer make the list for themselves?

There is one reason. Censoring relays or builders who must respect an inclusion list could decide to forego relaying or building a block entirely. This is fine when censoring builders are not competitive, which, in a market without private order flows/information asymmetries, should be true, since censoring builders leave fee-paying transactions on the table.

But in the presence of a censoring-best builder, a builder which

despite their censorship

, builds the highest-paying blocks, this is not fine. In this case, a proposer may not be willing to switch off the censoring-best builder for their block, or to hurt the builder's profits (which are forwarded to the proposer) by forcing the builder to fill up their block with garbage so that they satisfy the second validity condition of inclusion lists.

To see how forward inclusion lists mitigate this issue and spread the cost of censorship across all proposers, not only honest ones who take the hit, we introduce a simple model of player types in the censorship game. We assume that an inclusion list of censored transactions is given for the current proposer/builder pair, at first without specifying who is responsible for making the list. Proposers may be:

- Censoring:

They do not want some transactions to enter their blocks

- Greedy:

They will take the max profit block, whether the block censors or not

- Non-censoring:

They will require either of the two validity conditions of inclusion lists to be satisfied, given some list for their slot.

Censoring:

They do not want some transactions to enter their blocks

Greedy:

They will take the max profit block, whether the block censors or not

Non-censoring:

They will require either of the two validity conditions of inclusion lists to be satisfied, given some list for their slot.

While builders may be:

- Censoring:

They do not build blocks with some specific transactions contained in the inclusion list

- Non-censoring:

They include any and all transactions

Censoring:

They do not build blocks with some specific transactions contained in the inclusion list

Non-censoring:

They include any and all transactions

Note that builders are all greedy, but censoring builders add more constraints to their optimisation problem. We assume here that there exist a censoring-best builder realising a greater profit than non-censoring builders.

From this simple model, we can determine some properties:

- It may be rational to build or propose blocks which "waste space" by filling up the block, to satisfy the second validity

condition of inclusion lists. Indeed, a censoring-best builder could find it worthwhile to pay the basefee for the extra gas necessary to fill up the block, and still win the block auction. Let's call such blocks

stuffed blocks

. While this would likely not be a common occurrence, it has the bad side-effect of raising network costs for everyone.

- When the inclusion list applies to the proposer who makes the list, only non-censoring proposers make the list. Greedy proposers do not wish to hurt their profit, and so do not make the list. Censoring proposers do not make the list for themselves either.
- When the inclusion list applies to some other proposer than the one making the list (forward inclusion list), greedy proposers no longer mind making the list. Thus both non-censoring and greedy proposers now bind other greedy or censoring proposers, increasing the "zone of influence" of the list. Proposers making the list (either non-censoring or greedy) still cannot force the hand of censoring proposers, who could prefer to not make a block at all, or make a stuffed block instead if it nets them a positive profit. Still, the economics look much better: Non-censoring proposers incur an implicit cost when a censoring-best builder exists, by shirking the option which realises the maximum profit for themselves. Now, this cost is also borne by the rest of the network.
- When there is no censoring-best builder, only censoring proposers are hurt.

It may be rational to build or propose blocks which "waste space" by filling up the block, to satisfy the second validity condition of inclusion lists. Indeed, a censoring-best builder could find it worthwhile to pay the basefee for the extra gas necessary to fill up the block, and still win the block auction. Let's call such blocks

stuffed blocks

. While this would likely not be a common occurrence, it has the bad side-effect of raising network costs for everyone.

When the inclusion list applies to the proposer who makes the list, only non-censoring proposers make the list. Greedy proposers do not wish to hurt their profit, and so do not make the list. Censoring proposers do not make the list for themselves either.

When the inclusion list applies to some other proposer than the one making the list (forward inclusion list), greedy proposers no longer mind making the list. Thus both non-censoring and greedy proposers now bind other greedy or censoring proposers, increasing the "zone of influence" of the list. Proposers making the list (either non-censoring or greedy) still cannot force the hand of censoring proposers, who could prefer to not make a block at all, or make a stuffed block instead if it nets them a positive profit. Still, the economics look much better: Non-censoring proposers incur an implicit cost when a censoring-best builder exists, by shirking the option which realises the maximum profit for themselves. Now, this cost is also borne by the rest of the network.

When there is no censoring-best builder, only censoring proposers are hurt.

This rough model could be developed further, to provide accurate bounds and dynamics for how revenue is obtained by various players in the game. A work-in-progress is found

here, and generally relates to our

ROP-2(RIG open question).

Open questions for inclusion lists

As argued previously, a forward inclusion list model where a previous proposer selects the list for a future proposer seems better suited to a world where a censoring-best builder reigns supreme. There remains implementation questions. What happens when transactions from a forward inclusion list are included by the current builder? How should we select transactions to feature in the list?

In the case of mev-boost, all these questions apply, on top of which we must reason about the incentives due to the optionality of the mechanism. mev-boost is a network of builders, organised around the mev-boost builder client. Nothing prevents the introduction of a new builder client, which may prove to be a dominant strategy for (greedy) builders and proposers to switch to, in particular if profit-hurting constraints are applied to the mev-boost network as a whole.

The

mev-boost plan todayis indeed for the current proposer to make the inclusion list for their own builders, as forward inclusion lists would induce greedy proposers to switch to a different builder client, e.g., "MEV-boost classic", without such lists. We must then recognise that we cannot realise the same outcomes with out-of-protocol solutions, as forward inclusion lists only work if they are not opt-in, and the protocol alone can enforce globally such constraints.

2

What does PBS value?

After this long exploration of the various ways PBS might be done in- or out-of-protocol, it's natural to wonder which version is best suited for our purposes. In my opinion, the answer lies in understanding what is gained and what is lost with each version. We've hinted at this in a few places before, but what we are really interested in, as mechanism designers for the protocol, is maximising the total value realised, or social welfare.

But what is value? A proper answer to this question requires a lot more work, so we hint here only at some ways the protocol might realise value for its stakeholders:

- The value derived from the execution of user transactions:

It is worth $10 to me that I am able to send a payment over Ethereum, and though I may pay a fraction of that, this value is realised once the payment goes through.

- The value derived from holding ETH:

Perhaps this is a protocol concern too! ETH is used to secure the protocol, so maximising its value benefits the protocol directly, in addition to benefiting ETH holders. There are disagreements over how this is achieved, relating to the nature of ETH as an asset.

- Other ways?

The value derived from the execution of user transactions:

It is worth $10 to me that I am able to send a payment over Ethereum, and though I may pay a fraction of that, this value is realised once the payment goes through.

The value derived from holding ETH:

Perhaps this is a protocol concern too! ETH is used to secure the protocol, so maximising its value benefits the protocol directly, in addition to benefiting ETH holders. There are disagreements over how this is achieved, relating to the nature of ETH as an asset.

Other ways?

In this section, we'll often use what we know about the EIP-1559 mechanism to understand what PBS values and how it realises this value. For instance,

[EIP-1559 elicits the value a marginal user is willing to pay to be includedright now]

. This marginal user is the "last one" that can make it through while the chain preserves both a burst limit (the 30 million gas limit) and long-term targets (15 million gas per block on average). So what does PBS elicit?

Does PBS

really

value the MEV in the block? Off-chain agreements between proposers and builders

Here we think of PBS with the whole block auction model. Builders compete over the value they are able to return to the proposer, by making a block satisfying the validity conditions imposed by the protocol. This value really is MEV, the value a proposer would be able to obtain from the special position they hold as temporary leader over the chain's contents. Or if not the whole MEV, a fraction of it, e.g., the realisable MEV.

This is the intent of the PBS bid: returning to the proposer the value they could extract, but which due to their lack of sophistication or knowledge they are unable to. This value may be quantified. But is that intent always realised? Is the winning bid of the PBS auction equal or close to that value?

Given the assumptions we've made so far, it is not the case. A proposer may receive bids at the protocol auction, and decide to forego any of them in favour of a private agreement with a builder. From the protocol's perspective, it's equivalent to the proposer accepting a 0 ETH bid and receiving the value somewhere else, for instance via a transaction in the builder's block.

These situations could happen when the proposer and the builder have a special, trusted relationship. Without the builder submitting a binding bid to the protocol, the proposer may forego their assurance that payment will be guaranteed regardless of the builder's actions. This could be helpful when the MEV present in the block is very large, and the builder does not have the upfront capital to submit their binding bid to the protocol, but is trusted enough by the proposer. The vertical integration between sophisticated proposers and builders favour such conditions, as Jolene Dunne noted in her Devcon talk

Evaluating the PBS Experiment: Early insights from MEV-Boost and the Builder Market

.

Understanding these types of off-chain agreements was

critical in the study of EIP-1559. When is it to the block producer and user's advantage to collude, in order to defeat the mechanism's constraints? In the case of EIP-1559, the effects are particularly severe. Collusion between users and block producers could make the basefee value completely irrelevant, defeating its use as an objective oracle of the current congestion price. In the case of PBS, it could be argued that an objective oracle for current MEV does not bring about UX improvements as important as objective pricing, but this argument relies perhaps on a lack of imagination of what could be done with such an oracle.

Recovering the true PBS value via consensus bids and protocol capture

One interpretation of the EIP-1559 oracle is that it "captures" the value expended by users to compensate for the congestion they create. Without this mandatory capture, inclusion lists would not work, a

really

nice side-effect coming from the impossibility of block producers to stuff their blocks for free. When the protocol "sees" more, introspection allows for a more resilient system, indeed often as a side-effect.

So how can we make the PBS bid as truthfully revealing as EIP-1559's basefee? In much the same way: the protocol must capture it. For PBS, solutions include coming to consensus over the highest bid received by the proposer. The proposer is then bound to propose a block corresponding to a bid at least equal to the

consensus bid

. This consensus relies on a honest majority assumption from the attesters in the proposer's committee, the same committee making safe the proposer's commitment to some builder block. The rationale is that if at least half of all attesters in the committee have seen such a bid, the proposer must have, too.

Given a consensus bid, which the proposer must match, off-chain agreements with builders no longer hold. If the proposer's trusted builder makes an off-chain promise to return 10 ETH to the proposer, but a separate builder sends in a 5 ETH bid through the protocol's auction, the proposer is bound by the protocol to accept the 5 ETH bid instead. In this case, it would be rational for the trusted builder to make a bid for 5.000001 ETH via the protocol auction, and return 4.999999 ETH to the proposer somehow.

Clearly, the PBS bid was not able to elicit the full 10 ETH value the proposer is receiving from the trusted builder, but this problem is not specific to consensus bids. Generally, in a model of rational builders, a proposer should expect to receive the second-highest bid value, as the best builder only needs to beat the second-best to win the auction. Competitiveness of the builder market then ensures the delta between the best bid and the second-best remains small, with both bids close to the true value of the block.

Given a consensus bid, and so an objective value of the block's value (assuming again the competitiveness of the builder market), the protocol

knows

how much value the block returns. What can we do with this value?

- Can we return it entirely to the block's proposer? If we did so, as happens with EIP-1559, the oracle may no longer objectively report the true block's value. A trusted builder could bid an absurdly high value to win the auction, with the proposer and builder profit-sharing privately once the auction is over.
- Can we smooth it? This case was also studied for EIP-1559, known as

l

-smoothing in Tim Roughgarden's

landmark paper. With PBS, this is the suggestion of

committee-driven smoothingby Francesco, known as MEV-smoothing, where the value is not smoothed over time across proposers, but across all attesters participating in the slot.

- Can we burn it? This is how EIP-1559 functions today. Domothy recently wrote up a mechanism where the value is captured then burned, known as

MEV-burn. The consensus bid is obtained by making some number of proposers compete over who can burn the most, so that attesters need only attest for the proposer with the highest burn. This is a recent proposal, and it is currently discussed

to gain a better understanding of its properties and guarantees.

Can we return it entirely to the block's proposer? If we did so, as happens with EIP-1559, the oracle may no longer objectively report the true block's value. A trusted builder could bid an absurdly high value to win the auction, with the proposer and builder profit-sharing privately once the auction is over.

Can we smooth it? This case was also studied for EIP-1559, known as

l

-smoothing in Tim Roughgarden's

landmark paper. With PBS, this is the suggestion of

committee-driven smoothingby Francesco, known as MEV-smoothing, where the value is not smoothed over time across proposers, but across all attesters participating in the slot.

Can we burn it? This is how EIP-1559 functions today. Domothy recently wrote up a mechanism where the value is captured then burned, known as

MEV-burn. The consensus bid is obtained by making some number of proposers compete over who can burn the most, so that attesters need only attest for the proposer with the highest burn. This is a recent proposal, and it is currently discussed to gain a better understanding of its properties and guarantees.

Note that all of these mechanisms implicitly rely on PBS featuring the whole block auction. It is trivial otherwise to use the proposer's partial block as a way of "hiding" the value in the proposer's part, which the protocol cannot mechanically capture from.

Can PBS maximise protocol value?

In the previous sections, we assumed the ideal PBS bid represented the whole MEV of the block, and asked whether the bid received in-protocol objectively revealed this value. In this section, we review our initial assumption, and ask whether, even if objectively revealed, the PBS value matches the true MEV of a block.

Latency.

The first way this might fail is due to latency. Rounds of communication between builders and proposers, even with fast connections, loses precious milliseconds which could be used to integrate new information, such as newly pending transactions, into block construction. Value may simply be lost by a user who would have been included by the proposer, but didn't make it in time for inclusion in the builder's block. Such losses are likely small, as all parties have the incentive to minimise latency as much as possible. Yet the incentive to waste as little time could induce a timing game, the subject of

ROP-0, where proposers or builders delay the fulfilment of their duties for as long as they possibly can. This incentive could also induce builder colocation, to minimise round-trips.

Private order flow and value elicitation.

The second way it might fail comes from the format of the mechanism. Whatever way the builder block is constructed, a single bid must arrive for the PBS whole block auction. This single bid must be emitted from some entity able to aggregate the value of the separate bundles and transactions it receives from searchers and users. What if the entity is not trusted with the separate order flows it receives? In the example given by Quintus,

in this mev-boost repo issue, a monolithic block builder is able to make a bid for 10 ETH. Meanwhile, the dynamics of an auction between searchers mean that a block merging the searchers' bundles realises a total value greater than 10 ETH, yet the bid expressed is lower than 10 ETH.

This situation clearly points to a failure of market competition at surfacing the true, current value of the block. Efforts towards making order flow non-exclusive would likely recover such value, as would

efforts towards distributed builderswho could aggregate various order flows towards maximising the value returned.

3

Blockspace futures.

Another type of economic value which may not be realised is the value unlocked by trust-minimised blockspace futures. The discussion relates to arguments made in the

block vs slot auctions section. In the slot auction model, the builder essentially obtains a "one-slot" blockspace future from the proposer, the guaranteed right to build any block for the next slot. No proposer is able to offer such futures for more than one slot ahead, at least trustlessly:

- With “vanilla” in-protocol PBS, the proposer could always renege on their commitment to let the builder make their block at the last minute (though they could stake some money on whether they honour the commitment or not…)
- With PBS featuring “consensus bids”, the proposer might not have the choice over which builder to select, so it is definitely not possible for them to make such commitments in advance.

With “vanilla” in-protocol PBS, the proposer could always renege on their commitment to let the builder make their block at the last minute (though they could stake some money on whether they honour the commitment or not…)

With PBS featuring “consensus bids”, the proposer might not have the choice over which builder to select, so it is definitely not possible for them to make such commitments in advance.

The question is whether there is much value in obtaining five-slot or ten-slot blockspace futures, vs the one-slot future. Are there economic activities which cannot be conducted without a guarantee over what happens 10 slots in the future? I’m not sure about this one. See also

a postfrom our RIG researcher Julian Ma on blockspace derivatives!

Combinatorial preferences.

And finally, we may find ourselves in a situation where a builder cares for either owning the right to make the next two blocks, or none at all. We’re talking here about multi-block MEV, where value is realised by performing operations in two separate blocks. In the single-slot, whole-block PBS auction, a builder may win auction for slot 1, but not know whether they can also win the auction for slot 2. Again there is a distinction between the vanilla PBS and the consensus bids PBS, in that a proposer controlling two slots in a row could make an off-chain promise to a builder for the control of both blocks under vanilla PBS, but clearly cannot honour that promise in PBS with consensus bids.

So in case the auction is truly combinatorial, and builders have preferences over bundles of blocks rather than single-blocks, some value may be lost from not allowing builders to express these preferences via the PBS auction mechanism. Multi-block MEV appears greatly centralising, and combinatorial bids may then be a Pandora’s box we’re not willing to open, yet social welfare (the total value realised) is lower if we are not able to realise this value. Could combinatorial bids for

partial

blocks be feasible? i.e., mev-geth with bundles spanning two blocks? Would this reduce centralisation risks? IMO some fascinating open questions here!

PEPC: Market structure guarantees for flexible proposer commitments

PEPC (pronounced “pepsi”), stands for

Protocol-Enforced Proposer Commitments

, a recent post I wrote on ethresear.ch

.

PEPC is a direct application of the modular framework for PBS splitting it between the market structure and the allocation mechanism. PEPC attempts to provide the market structure without enforcing any kind of allocation mechanism, leaving it up to the proposer to make their own out-of-protocol commitments should they wish to do so. The protocol will only be there to secure these commitments, i.e., ensure the proposer cannot walk back their commitments after observing the builder deliver, letting the proposer to steal from the builder.

The main thrust behind PEPC is to generalise the market structure to more commitments than simply the execution payload. In the near future, some builders will be summoned to produce Danksharding blocks. Later on, we may find it useful to have state providers support block construction and witness computation for stateless proposers. We may also require validity proofs based on a canonical zkEVM to be attached to an execution payload. All such pieces of data will have to come from third-party entities running specialised hardware, potentially inaccessible to the average proposer.

To ensure long-term sustainability of protocol development, figuring out a flexible form of commitment provisioning could help with avoiding to build from scratch a new facility each time a new item is added to the block construction checklist. In its most radical incarnation, PEPC could allow the proposer to control precisely how their block is being made. Parallel execution-builders could be summoned, limited by gas use and a list of state accesses, to ensure compatibility of transaction flows. Block building rights could be auctioned off ahead of time, up to the proposer’s choice (as long as it is known when they are expected to propose, up to 2 epochs in advance).

The way PEPC works is by allowing a proposer to include additional validity conditions to their block, e.g., “the execution payload must be signed by builder

x

” or “the execution payload starts with prefix

abc

and is then filled by builder

y

” or “the block witness is signed by state provider

z

”. I am hoping we can make use of EVM powers to provide a language to express such commitments, with many questions currently un-answered:

- In the benchmark case of reproducing the PBS whole block auction via PEPC, how should builder promises of payment be represented? Would an EVM “context-aware” transaction allow for conditional payment, based on the existence of a proposer commitment? Can the proposer always be made whole? What does the process look like end-to-end? (see also

ROP-4!)

- How far can we go with the commitments? Can we succinctly represent complex commitments such as “transaction 1 is given by builder

x

, transaction 2 is given by builder

y

, etc”? Is there a limit to commitment complexity, such as “commitment gas”? Can the fork choice handle discriminating between all partial deliveries of some subset of commitments?

- How does PEPC mesh with protocol-captured value? Can the value received by the proposer be “smoothed”? Should it be anyways?

In the benchmark case of reproducing the PBS whole block auction via PEPC, how should builder promises of payment be represented? Would an EVM “context-aware” transaction allow for conditional payment, based on the existence of a proposer commitment? Can the proposer always be made whole? What does the process look like end-to-end? (see also

ROP-4!)

How far can we go with the commitments? Can we succinctly represent complex commitments such as “transaction 1 is given by builder

x

, transaction 2 is given by builder

y

, etc”? Is there a limit to commitment complexity, such as “commitment gas”? Can the fork choice handle discriminating between all partial deliveries of some subset of commitments?

How does PEPC mesh with protocol-captured value? Can the value received by the proposer be “smoothed”? Should it be anyways?

So what now?

Research continues into the exciting world of PBS. As our lens widens, we also find more focus on the various mechanisms at our disposal, hopefully providing us with the theory we need to evaluate what should be the final form of PBS in Ethereum. In most of what was discussed here, the protocol-side of PBS was emphasised, but distributed building, user-side privacy, smart transactions and other constructions are sure to yield fascinating ways to make the most out of our favourite coordination engine. In the meantime, a strong focus is placed on censorship resistance, to augment the current out-of-protocol market with ways for the proposer to have more control over the block built for them.

Thank you for reading this long post and make sure to check out our

Open Problems, reach out if you’re interested to join this collective research effort!

Many thanks to Francesco d’Amato and Julian Ma for comments on the post, and of course to the many people working on

and thinking about this topic too!

Note that this default behaviour is

not

a protocol specification, but a rational move by block producers in the absence of other order flows.

But a protocol without forward inclusion lists could presumably attract away validators from the Ethereum protocol, in a meta-game! Protocols are sovereign over their domain, yet remain in competition with one another, h/t to Nikete for this line of thinking.

Such builders can likely do more, such as returning value to the users themselves. Note that this discussion is somewhat orthogonal to the topic at hand, PBS intervening only to capture the value that

wasn’t

returned or otherwise captured. Mitigation remains a cardinal direction for all parties concerned.