Why enshrine Proposer-Builder Separation? A viable path to ePBS

by mike neuder and justin drake; may 25, 2023

tl;dr; Proposer-Builder Separation (PBS) decouples the task of block proposing (done by PoS validators) from block building (done most profitably by MEV searchers). PBS aims to improve access to MEV for validators by explicitly creating a permissionless market for block production. By allowing proposers to outsource block construction, validators can continue running on consumer-grade hardware without missing out on the valuable MEV exposed while they are the elected proposer. [mev-boost

](https://github.com/flashbots/mev-boost) implements out-of-protocol PBS and accounts for ≈90% of Ethereum blocks. Enshrined PBS (ePBS, sometimes referred to as in-protocol/IP PBS) evolves the consensus layer to implement PBS at the protocol level. While ePBS has been discussed since 2021 and is part of Ethereum's roadmap, recent events ([Low-Carb Crusador

](https://collective.flashbots.net/t/post-mortem-april-3rd-2023-mev-boost-relay-incident-and-related-timing-issue/1540), [Shapella mev-boost

prysm signature bug](https://collective.flashbots.net/t/impact-of-the-prysm-invalid-signature-bug-on-the-mev-boost-ecosystem-at-the-shapella-fork/1623), & relay response to unbundling) have turned the attention of the Ethereum community towards mev-boost

and the evolution of PBS.

This document aims to…

[a] outline arguments for ePBS (as opposed to continuing with mev-boost

),

[b] present and respond to the counter-arguments to ePBS,

[c] describe desirable properties of ePBS mechanisms,

[d] sketch an ePBS design based on the existing research,

[e] highlight optimistic relaying as an additional tool to build towards ePBS, and

[f]

encourage discussions around the ePBS design space.

This document does not aim to…

[a] perform an exhaustive literature review (seehere),

[b] fully spec out an ePBS implementation, or

[c] cover alternative designs for ePBS.

Many thanks to Barnabé, Dan Marzec, Terence, Chris Hager, Toni, Francesco, Rajiv, Thomas, and Jacob for comments on draft versions of this document.

Introduction

[

upload_9a682b56f269faafefdaaffcdac52a08

2496×418 126 KB

](https://ethresear.ch/uploads/default/original/2X/c/c77a0191cf5bc7bdd97aa04a22863099d3d89e33.jpeg)

Proposer-Builder Separation (PBS) allows validators to outsource their block building duties to a set of specialized builders who are well equipped to extract MEV (hence separating

the roles of proposer and builder). Proposers sell their block-production rights to builders who pay for the privilege of choosing the transaction ordering in a block. Proposers earn MEV rewards in addition to their protocol issuance, and block builders compete to assemble valuable blocks while saving a portion of the MEV for themselves as profit.

Enshrined PBS

Enshrined PBS (ePBS) advocates for implementing PBS into theconsensus layer of the Ethereum protocol. Because there was no in-protocol solution at the time of the merge, Flashbots built [mev-boost

](https://github.com/flashbots/mev-boost), which became a massively adopted out-of-protocol solution for PBS that accounts for ≈90% of Ethereum blocks produced.

[

upload_1a170bd1d4ae5f23cc52d66e4cc0ed52

1800×700 61.9 KB

](https://ethresear.ch/uploads/default/original/2X/e/ec405ebd555dec303870468b2eb3fb0123fc86b7.png)

Figure 1

– mev-boost

slot share (orange) since the merge. Sourcemevboost.pics.

mev-boost

continues to be critical infrastructure provided to grant permissionless access to the external block-building market for all validators, but it relies heavily on a small set of centralized relays to act as mutually-trusted auctioneers facilitating the block-production pipeline. We present the case for ePBS by highlighting (i) that relays are antithetical to Ethereum's core values, (ii) the risks and inefficiences of side-car software, and (iii) the costs and unsustainability of relay operations.

Reasons to enshrine

- Relays oppose Ethereum's values.

The following core tenants of Ethereum are eroded by the mass dependence on relays. * Decentralization

: Relays are centralized.

Six relays, operated by five different entities, account for 99% of mev-boost

blocks. This small consortium of relay operators should not play such an outsized role in the ecosystem.

- Censorship resistance

: Relays can censor blocks.

Since relays are centralized, they are exposed to regulation. This played out post-merge as some relays were pressured to censor transactions interacting with addresses on theOFAC sanctions list.

- Trustlessness

: Relays are trusted by validators and builders.

Validators trust relays to provide them a valid block header and to publish the full beacon block; builders trust relays not to steal MEV. A violation of either of these trust assumptions would be detectable, but as demonstrated by the "Low-Carb Crusader", dishonesty can be profitable, even if only through a one-time attack.

- Decentralization

: Relays are centralized.

Six relays, operated by five different entities, account for 99% of mev-boost

blocks. This small consortium of relay operators should not play such an outsized role in the ecosystem.

- Censorship resistance

: Relays can censor blocks.

Since relays are centralized, they are exposed to regulation. This played out post-merge as some relays were pressured to censor transactions interacting with addresses on theOFAC sanctions list.

- Trustlessness

: Relays are trusted by validators and builders.

Validators trust relays to provide them a valid block header and to publish the full beacon block; builders trust relays not to steal MEV. A violation of either of these trust assumptions would be detectable, but as demonstrated by the "Low-Carb Crusader", dishonesty can be profitable, even if only through a one-time attack.

- Out-of-protocol software is brittle.
- The “Low-Carb Crusader” unbundling exploited a relay vulnerability for 20+ million USD.

This attack and the general class of equivocation attacks it embodies demonstrate that relays are valuable targets outside of the protocol.

- The relay response to the unbundling attack caused consensus instability.

Due to relay-induced latency into the block-production pipeline, there was a 5x

increase in reorged blocks immediately after the attack. See [“Time, slots, and the ordering of events in Ethereum Proof-of-Stake”

](https://www.paradigm.xyz/2023/04/mev-boost-ethereum-consensus) for more details.

- During the Shapella upgrade, there was abug in the Prysm code that interacts with mev-boost

.

This resulted in a brief 10x

spike in missed slots immediately following the hard-fork. This bug was not caught because the code path for externally-built blocks is not covered by the consensus spec tests.

- There are significant core-dev coordination costs involved in maintaining compatiblity between beacon clients & relays.

Each hard-fork represents a significant amount of work from the relay and core developers to ensure mev-boost

continues functioning. This involves designing thebuilder spec, maintaining/improving therelay spec, and the software changes on the beacon clients, mev-boost

, and the mev-boost

relays. Because mev-boost

is out-of-protocol, this coordination is strictly additive to the standard ACD pipeline and usually happens later in the development cycle as a result.

- mev-boost

does not inherit the benefits of client diversity and the full consensus specification process.

Though there are multiple relay repositories, the vast majority of blocks flow through relays using the flashbotsimplementation. While this is simpler to maintain, it lacks the structural benefits of client diversity enjoyed by the beacon nodes; the full specification/spec-test infrastructure is also not leveraged by the differing relay repositories.

- The “Low-Carb Crusader” unbundling exploited a relay vulnerability for 20+ million USD.

This attack and the general class of equivocation attacks it embodies demonstrate that relays are valuable targets outside of the protocol.

- The relay response to the unbundling attack caused consensus instability.

Due to relay-induced latency into the block-production pipeline, there was a 5x

increase in reorged blocks immediately after the attack. See [“Time, slots, and the ordering of events in Ethereum Proof-of-Stake”

](https://www.paradigm.xyz/2023/04/mev-boost-ethereum-consensus) for more details.

- During the Shapella upgrade, there was abug in the Prysm code that interacts with mev-boost

.

This resulted in a brief 10x

spike in missed slots immediately following the hard-fork. This bug was not caught because the code path for externally-built blocks is not covered by the consensus spec tests.

- There are significant core-dev coordination costs involved in maintaining compatiblity between beacon clients & relays.

Each hard-fork represents a significant amount of work from the relay and core developers to ensure mev-boost

continues functioning. This involves designing thebuilder spec, maintaining/improving therelay spec, and the software changes

on the beacon clients, mev-boost

, and the mev-boost

relays. Because mev-boost

is out-of-protocol, this coordination is strictly additive to the standard ACD pipeline and usually happens later in the development cycle as a result.

- mev-boost

does not inherit the benefits of client diversity and the full consensus specification process.

Though there are multiple relay repositories, the vast majority of blocks flow through relays using the flashbots implementation. While this is simpler to maintain, it lacks the structural benefits of client diversity enjoyed by the beacon nodes; the full specification/spec-test infrastructure is also not leveraged by the differing relay repositories.

- Relays are expensive public goods.
- Relay operational costs range from ≈20k-100k USD per-year depending on the desired performance.

This doesn't include engineering and DevOps costs associated with running a highly-available production service.

- Relays are public goods that don't have a clear funding model.

While there are discussions around guilds, grants, and other funding vehicles, there is no obvious way to support relay development and operation (similar to the issues faced in supporting core-dev teams).

- Relay operational costs range from ≈20k-100k USD per-year depending on the desired performance.

This doesn't include engineering and DevOps costs associated with running a highly-available production service.

- Relays are public goods that don't have a clear funding model.

While there are discussions around guilds, grants, and other funding vehicles, there is no obvious way to support relay development and operation (similar to the issues faced in supporting core-dev teams).

~~ ePBS resolves these issues by eliminating the relay. ~~

Note: MEV burn is currently being explored for its economic & security benefits. If we decide to pursue MEV burn, these benefits serve as yet another carrot on a stick for enshrinement.

Reasons not

to enshrine

We believe it is important to highlight the counter-point of enshrinement by addressing the main arguments and presenting our responses.

- "If it ain't broke don't fix it."

mev-boost

has worked incredibly well given the scale of its adoption. As the implementation continues to harden, we can gain confidence in its security properties and build out the specification. If we can find a credibly neutral way to fund a set of relays, then we could continue depending on them into the future. * Response –

mev-boost

has worked well, but there are no guarantees that this stability will continue. Another major censorship event, further attacks, and continued centralization pressures of relays, builders, and searchers pose significant risks to Ethereum. There is value in having clarity about ePBS to handle a situation where there is a pressing need for faster enshrinement. Additionally, ePBS will take time to design and implement – we should start formalizing it now, even if we continue with the relays for the next O(1-2 \; years)

as ePBS progresses.

- Response –

mev-boost

has worked well, but there are no guarantees that this stability will continue. Another major censorship event, further attacks, and continued centralization pressures of relays, builders, and searchers pose significant risks to Ethereum. There is value in having clarity about ePBS to handle a situation where there is a pressing need for faster enshrinement. Additionally, ePBS will take time to design and implement – we should start formalizing it now, even if we continue with the relays for the next O(1-2 \; years)

as ePBS progresses.

- Could MEV be addressed with different tools?

There is a growing discourse around protecting users from MEV on the application/transaction level.SUAVE, CoW swap, and MEVBlocker are three of many solutions that are continuing to gain usage. If a significant portion of MEV can be protected against, perhaps enshrining PBS is an unnecessary step on an already ambitious roadmap. * Response –

We hope that this line of work can help protect users from “toxic” MEV, but we don’t expect on-chain MEV to ever be fully eliminated. Further, some MEV is extracted off-chain, requiring sophistication beyond just computational power. For example, in order to execute a CEX-DEX arbitrage, a validator would need liquidity and connectivity with a CEX in addition to the algorithmic resources to find and execute such an opportunity. We don’t envision a future in which there is little to no MEV in Ethereum or a solo-staking validator could meaningfully extract it on their own.

- Response –

We hope that this line of work can help protect users from “toxic” MEV, but we don’t expect on-chain MEV to ever be fully eliminated. Further, some MEV is extracted off-chain, requiring sophistication beyond just computational power. For example, in order to execute a CEX-DEX arbitrage, a validator would need liquidity and connectivity with a CEX in addition to the algorithmic resources to find and execute such an opportunity. We don’t envision a future in which there is little to no MEV in Ethereum or a solo-staking validator could meaningfully extract it on their own.

- There are other roadmap items that should take precedence.

The roadmap has many goals beyond ePBS. If we choose to go ahead with ePBS, it begs the question of where this can fit on the roadmap and what upgrades will be pushed down the line as a result. * Response –

We believe that ePBS depends on Single-Slot Finality (SSF) for security and complexity reasons. Additionally, a validator set consolidation is a prerequisite for any SSF progress (see [Increase the MAX_EFFECTIVE_BALANCE

](https://notes.ethereum.org/@mikeneuder/increase-maxeb)). The resource allocation problem is difficult, but we believe that ePBS should be part of these discussions, especially in the context of being bundled (pun-intended) with a larger consensus upgrade.

- Response –

We believe that ePBS depends on Single-Slot Finality (SSF) for security and complexity reasons. Additionally, a validator set consolidation is a prerequisite for any SSF progress (see [Increase the MAX_EFFECTIVE_BALANCE

](https://notes.ethereum.org/@mikeneuder/increase-maxeb)). The resource allocation problem is difficult, but we believe that ePBS should be part of these discussions, especially in the context of being bundled (pun-intended) with a larger consensus upgrade.

- What is the right thing to enshrine?

From a protocol design perspective, there are many mechanisms that could be implemented. Barnabé explores these concepts in [“Unbundling PBS”

](https://ethresear.ch/t/unbundling-pbs-towards-protocol-enforced-proposer-commitments-pepc/13879), [“Notes on Proposer-Builder Separation”

](https://barnabe.substack.com/p/pbs), and [“Seeing like a protocol”

](https://barnabe.substack.com/p/seeing-like-a-protocol). One takeaway from this work is that mev-boost

implements a block-auction, which is not the only option for ePBS. Julian explores this further in [“Block vs Slot Auction PBS”

]
(https://mirror.xyz/0x03c29504CEcCa30B93FF5774183a1358D41fbeB1/CPYI91s98cp9zKFkanKs_qotYzw09kWvouaAa9GXBrQ).
* Response –

ePBS is only useful insofar as it is adopted by builders and validators; the worst-case scenario is that ePBS is sidestepped by out-of-protocol solutions we didn’t foresee. While we acknowledge that any protocol upgrade has unknown-unknowns, we believe that by opening the discussion, working to achieve rough community consensus, and taking the next step in formalizing the design space of ePBS will improve confidence around what we are working towards. We also present the optimistic relay roadmap below, which takes a more iterative approach at evolving mev-boost

.

- Response –

ePBS is only useful insofar as it is adopted by builders and validators; the worst-case scenario is that ePBS is sidestepped by out-of-protocol solutions we didn’t foresee. While we acknowledge that any protocol upgrade has unknown-unknowns, we believe that by opening the discussion, working to achieve rough community consensus, and taking the next step in formalizing the design space of ePBS will improve confidence around what we are working towards. We also present the optimistic relay

roadmap below, which takes a more iterative approach at evolving mev-boost

.

ePBS design space

For extensive ePBS literature links see [“Bookmarks relevent for Proposer-Builder Separation researchers”

](https://collective.flashbots.net/t/bookmarks-relevant-for-proposer-builder-separation-pbs-researchers/1319). We define the following properties as desirable:

1. honest builder publication safety

– If an honest builder wins the auction, the builder (i) must

have an opportunity to create a block, and (ii) must

be confident that any payload contents they release become canonical (i.e., protection from unbundling &equivocation attacks from the proposer).

1. honest builder payment safety

– If an honest builder payment is processed, the builder must

be able to publish a block that becomes canonical.

1. honest proposer safety

– If an honest proposer commits to a block on-time, they must

receive a payment at least as large as specified by the bid they selected.

1. permissionless

– Any builder can participate in the auction and any validator can outsource block production.

1. censorship resistance

– There must

be a mechanism by which honest proposers can force through transactions they suspect are being censored without significantly sacrificing on their own rewards (“If we rely on altruism, don’t make altruism expensive”–Vitalik).

1. roadmap compatibility

– The design must

be compatible with future roadmap upgrades (SSF, mev-burn, distributed block-building, SSLE, DAS, etc).

One design instantiation – Two-Block HeadLock (TBHL)

While there are many proposed ePBS implementations, we present a small modification of the originaltwo-slot design from Vitalik. We call it Two-Block HeadLock (TBHL) because it uses a single slot to produce two blocks. The first is a proposer block that contains a commitment to a specific execution payload and the second is a builder block that contains the actual transaction contents (here we just call the overall pair of blocks a “single” slot because only one execution payload is produced). Note that with a second round of attestations, the slot time will likely need to increase. TBHL also incorporates some of the features of headlock to protect builders from proposer equivocations. TBHL shares many components with the current mechanism (HLMD-GHOST) and satisfies the six properties specified above.

Note: This is a sketch of the design; it is intentionally brief to improve readability. If we gain confidence that TBHL is a good overall approach, we can begin the specification and full security analysis. The aim is to present a simple, concrete example of a mechanism that satisfies the ePBS design properties, without overloading the reader with implementation details.

[

upload_8955ce1c08fe500e9aa8cbe27d31032f

1205×1314 119 KB

](https://ethresear.ch/uploads/default/original/2X/9/95f9159fbfc5fd513899f8581527a7253a6d1acc.png)

Figure 2

– The slot anatomy of TBHL. A proposer block is proposed and attested to in the purple phase, while a builder block is proposed and attested to in the yellow phase. The proposers, attesters, and builders each make different observations at various

timestamps in the slot.

TBHL has the notion of proposer

and builder

blocks. Each slot can contain at most: one proposer block + one builder block, each of which receives attestations. The slot duration is divided into 4 periods.

t=t_0

: The proposer chooses winning bid and publishes a proposer block.

The proposer starts by observing the bidpool, which is a p2p topic where builders send their bids. The proposer selects one of these bids and includes it in a block they publish before t_1

.

t=t_1

: The attesting committee for the proposer block observes for a timely proposal.

This is the equivalent of the “attestation deadline” at t=4

in the current mechanism. If at least one block is seen, the attesting committee votes for the first one that they saw. If no block is observed, the attesting committee votes for an empty slot (this requires [block, slot

](https://github.com/ethereum/consensus-specs/pull/2197) voting).

t=t_{1.5}

: The attesting committee for the builder block checks for equivocations.

If the attesting committee sees (i) more than one proposer block or (ii) no proposer blocks, they give no proposer boost to any subsequent builder block. If the attesting committee sees a unique proposer block, they give proposer boost to the builder associated with that bid (see [“Headlock in ePBS

”](https://ethresear.ch/t/equivocation-attacks-in-mev-boost-and-epbs/15338#headlock-in-epbs-8) for more details).

t=t_2

: The builder checks if they are the unique winner.

If a builder sees an equivocation, they produce a block that includes the equivocation as proof that their unconditional payment should be reverted. Otherwise, the builder can safely publish their builder block with a payload (the transaction contents). If the builder does not

see the proposer block as the head of the chain, they publish an empty block extending their head (see [“Headlock in ePBS

”](https://ethresear.ch/t/equivocation-attacks-in-mev-boost-and-epbs/15338#headlock-in-epbs-8) for more details).

t=t_3

: The attesting committee for the builder block observes for a timely proposal.

This is a round of attestations that vote for the builder block. This makes t_3

a second attestation deadline.

We can assert that this mechanism satisfies the ePBS design properties.

1. honest builder publication safety

– The only situation where builder safety could be in question is if the proposer equivocates. For brevity, the details of the equivocation protection are left out of this document. Please see [“Headlock in ePBS”

](https://ethresear.ch/t/equivocation-attacks-in-mev-boost-and-epbs/15338#headlock-in-epbs-8).

1. honest builder payment safety

– If an honest builder is selected and their payment is processed, they will either (i) see no equivocations and have the opportunity to create a block with confidence that they are the unique recipient of proposer boost or (ii) see an equivocation and use it as proof to revert the payment. Again, please see [“Headlock in ePBS”

](https://ethresear.ch/t/equivocation-attacks-in-mev-boost-and-epbs/15338#headlock-in-epbs-8) for further details.

1. honest proposer safety

– If an honest proposer commits to a block on-time, their block will receive attestations and the unconditional payment will go through without reversion because the builder will not have any proof of an equivocation. Even if the builder block is not produced, the bid payment occurs so long as no equivocation proof is presented.

1. permissionless

– The p2p layer is permissionless and any builder can submit bids to the bidpool. Any validator can listen to the bidpool if they want to outsource block building, or they can choose to build locally instead.

1. censorship resistance

– The proposal is compatible with censorship resistance schemes. For example, the proposer block could contain a [forward inclusion list

](https://notes.ethereum.org/@fradamt/forward-inclusion-lists). See [“PBS censorship-resistance alternatives”

](https://notes.ethereum.org/@fradamt/H1TsYRfJc) for more context.

1. roadmap compatiblity

– SSF fits naturally with this proposal by adding a third round of attestations after the builder block attestation round. The third round includes the full validator set and justifies the block immediately with a supermajority consensus. See [“A simple Single Slot Finality protocol”

](https://ethresear.ch/t/a-simple-single-slot-finality-protocol/14920) for more details. This mechanism is also highly compatible with mev-burn, as the base fee floor deadline, D

, could precede t_0

.

Optimistic relaying – an iterative approach to PBS

The design framework and TBHL presented above provide a “top-down” approach to ePBS. This has historically been the way R&D is done in Ethereum. Once the design is fleshed out, a spec is written, and the client teams implement it.

The existence of mev-boost

and in particular mev-boost

relays gives us an interesting additional angle to approach the problem – “bottom-up”. We can imagine there are many PBS implementations that lie on a spectrum between the original mev-boost

implementation and full ePBS. By modifying the relay, we can move “up” towards an ePBS implementation without needing to modify the spec and make changes to the consensus node software. This allows us to forerun and derisk some of the features of a full ePBS system (e.g., are builders OK with us removing cancellations?) while also remaining agile. This objective has already been presented in the optimistic roadmap.

The main theme of the optimistic roadmap is to remove relay responsibilities

. This has the added benefit of improving the operational efficiency of running a relay. As mentioned in “Reasons to enshrine,” relay operation is expensive and is currently being done only as a public good. By lowering the barrier to entry for relay operaters, we enable a more sustainable future for mev-boost

as we flesh out the details of ePBS.

Block submission in mev-boost

Before describing optimistic relaying, we briefly present the builder bid submission pipeline in the [mev-boost-relay

](https://github.com/flashbots/mev-boost-relay). Processing builder bids is the main function of the relay, and incurs the highest latency and compute costs. When a builder submits a bid to the relay the following occurs.

[

upload_29875d098550a358c78c57236b487925

1489×902 72.7 KB

](https://ethresear.ch/uploads/default/original/2X/1/187fd5a4aff59926fe62abac607dec7ab01de73a.png)

Figure 3

– Once the builder block is received by the relay, it is validated against an execution layer (EL) client. Once it is validated, the block is eligible to win the auction and may be signed by the proposer. Once the relay receives the signed header, it publishes the block to the p2p through a consensus layer (CL) client.

Since hundreds of bids are submitted each slot, the relay must (i) handle the ingress bytes of all the builder submissions, (ii) simulate the blocks on the EL clients, and (iii) serve as a data availablity layer for the execution payloads. Additionally, the validator relies on the relay to publish the block in a timely manner once they sign the header.

Optimistic relaying v1

The first version of optimistic relaying simply removes the block validation step from the block submission pipeline.

[

upload_7118043a7e81e563e4e0812a5833d129

1193×785 62 KB

](https://ethresear.ch/uploads/default/original/2X/6/68033336ae917a1637f68bf3cc185ade427a7843.png)

Figure 4

– Once the builder block is received by the relay, it is immediately eligible to win the auction and be signed by the proposer. Once the relay receives the signed header, it publishes the block to the p2p network through a consensus layer (CL) client.

The risk incurred by skipping the block validation is that an invalid block may be unknownly signed by the validator. This results in a missed slot because the attesting committee will reject the invalid block. The relay financially protects the validator against this situation by holding builder collateral. If a bad builder block results in a proposer missing a slot, the relay uses the builder collateral to refund the proposer. Optimistic relaying v1 is already upstreamed into the Flashbot's mev-boost-relay

repository and running on ultra sound relay. See ["An optimistic weekend"

](https://github.com/ultrasoundmoney/mev-boost-relay/blob/f428b3d5c7098bd1691bfbc2d658595b4abf9c82/docs/optimistic/an-optimistic-weekend.md) and ["Optimistic relays and where to find them"

](https://frontier.tech/optimistic-relays-and-where-to-find-them) for more details.

Optimistic relaying endgame

The final iteration of optimistic relaying behaves more like TBHL. Instead of the attesting committee enforcing the rules, the relay serves as a centralized "oracle" for the timeliness of events that take place in the bidpool. The flow of a block proposal is diagrammed below.

[

upload_dc0b3bfcb4652c9e14e6f4b370911eeb

1076×818 77.9 KB

](https://ethresear.ch/uploads/default/original/2X/0/0cebc393410dfb1d486b7d6cb513eab6112f1ff7.png)

Figure 5

– Builders now directly submit bids to the p2p layer (instead of the relay). Proposers observe these bids and sign the corresponding header of the winning bid. The builder of that signed header publishes the full block. The relay observes the bidpool and checks for timeliness of (i) the proposer's signed header and (ii) the builder's block publication. Notice that these observations are exactly what the attesting committee is responsible for in TBHL. The relay still holds builder collateral to refund a proposer if they sign a header on-time, but the builder doesn't produce a valid block.

Endgame optimistic relaying contains some of the ePBS machinery; proposers and builders will be interacting directly through the bidpool and relays will be implementing the validity conditions that the attesting committee would enforce at the consensus layer. Additionally, relay operation at that point is reduced to a collateralized mempool oracle service, which should be much cheaper and easier to run than the full validating relays of today.

Conclusion

Proposer-Builder Separation is an important piece of Ethereum's roadmap and continues to gain momentum in the public discourse. This document aims to present the arguments for and against enshrinement, lay out design goals of an ePBS mechanism, present Two-Block HeadLock (a minor variant of Two-slot PBS), and describe the utility of the optimistic relay roadmap. We hope to open up the enshrinement discussion and solicit alternative ePBS proposals from the community. While these "top-down" design and specification discussions continue, we hope to move forward on the "bottom-up" approach of optimistic relaying with the goal of making relays cheaper and more sustainable in the medium-term.

For any questions, concerns, or corrections, please don’t hesitate to reach out ontwitter or through telegram.

thanks for reading!

-mike & justin