Related work

1. Payload Timeliness Committee (PTC): An EPBS Design, by Mike and Francesco – July 2023
2. Payload Boosts in EPBS, by Potuz – Feb 2024
3. reth: High-Performance Ethereum Implementation, by Georgios – April 2024
4. EPBS Annotated Spec Guide, by Potuz – April 2023
5. Builder Reveal Timing Game in EPBS, by Terence – April 2024
6. Annotated Validator Guide for EPBS, by Terence – April 2024

Background

Pipelining consensus and execution validations into different stages is a lesser known concept in Ethereum today, but it may become crucial in the future, especially in any Proposer-Builder Separation (PBS) roadmap, whether it's block auction, slot auction, or ticket auction. As this separation seems inevitable in the long term, what does it mean to separate and pipeline consensus and execution validations into different stages? This post will explore the details, going into the benefits and downsides.

Since Ethereum moved to proof of stake, a node runs two pieces of software: a consensus client and an execution client. The consensus client verifies the consensus part of the block, while the execution client verifies the execution part. Although the validations are separated in two different software, the validation result is coupled. If either the consensus validation or the execution validation fails, the whole block fails. Timeline-wise, both validations must be completed before the AttestationDeadline

(4 seconds into the beacon slot) for the block to be considered as head for the attestation vote. This separation of concern for the two layers allows a node to verify consensus and execution in parallel. But despite parallel validation, both the consensus and execution parts of the block MUST

be validated before the AttestationDeadline

. Currently, the average block arrives at the 1 to 2-second mark into the slot, giving the running node less than 2 seconds to complete the verifications. This is called the validation

hot path

.

[

Screenshot 2024-05-23 at 6.47.57 AM

2517×1120 153 KB

](https://ethresear.ch/uploads/default/original/3X/5/9/59b830c5f23687d39a664ce8187c4d5672bbee06.png)

How long it took the beacon block + execution payload to arrive on my local NUC node

[

Screenshot 2024-05-23 at 6.50.00 AM

2519×1125 148 KB

](https://ethresear.ch/uploads/default/original/3X/c/7/c7581900ce6df801b5261dc709b5344e35480859.png)

How long it took the execution layer client to finish processing the execution payload on my local NUC node

Consensus and execution pipelining in PBS

In this section, we'll focus on block-auction-based EPBS. The anatomy of a 12 second beacon chain slot changes to the following:

- Second 0: proposer broadcasts a consensus block that commits to the builder's block
- Second 3: AttestationDeadline

for the consensus block to be processed; beacon attester casts vote. If the consensus block is not processed on time, the beacon attester votes for an older slot block as head

- Second 6: builder reveals the winning execution block
- Second 9: PtcDeadline

for the execution block to be seen;PTC attester casts vote that the builder revealed the execution block on time without verifying the block, only the verify block hash. If the builder fails to reveal, the PTC attester votes the current slot contains an empty execution block

Now let’s examine how different consensus actors

perceive their time under the new anatomy:

As a proposer

for the current slot, assuming block propagation takes 1 second, I have 2 seconds left for everyone to verify the consensus block. Note that only the consensus block needs to be propagated here, not the execution block, so the network load is lighter. The processing time is also reduced as it is bound by consensus validation, not the worst case of either consensus or execution validation.

As a beacon attester

for the current slot, I cast my vote on the consensus block that is head at the 3-second mark Instead of 4-second mark before. Nothing else changes here.

As a winning builder

for the current slot, I reveal my execution block at the 6-second mark. I may choose to withhold my execution block to gain a withhold boost if the beacon block is very weak, but I will reveal it as long as there is 40% of the current slot beacon committee voted for my execution header as canonical.

As a PTC attester

for the current slot, I cast my vote on the execution block revealed on time and verify that the execution block hash is valid. I don’t need to perform complete execution verification tasks like heavy state root validation before casting my vote. More on how PTC attester votes here.

As a builder

trying to win the next slot, as soon as the current slot builder reveals its winning execution block, I can begin building. This happens in less than 6 seconds. Before I build my block, I need to verify that the execution block I’m building on is valid.

As a proposer

for the next slot, I package the current slot PTC attestations into my block and pick the best bid that reflects my local chain view, which includes the beacon head root and execution reveal status. I should also verify the execution block to ensure I do not choose a header that builds on top of an invalid execution block as a parent.

Execution validation hot path

Now you ask… When is the deadline to verify the execution block for slot

n

? As it turns out, you don’t need to complete verifying the execution until slot n+1

's beacon attestation deadline. (Cheaper validations like block hash can still be done) If the n+1

proposer reveals the consensus block that builds on top of the execution payload and the attester casts its vote on the beacon block, the attesters will apply the builder’s boost to the execution block if the previous PTC has voted reveal on time and the execution block is valid. This represents a significant paradigm shift from today’s mental model.

To summarize:

- Proposer proposes consensus block B_c

that commits header h

- Beacon attesters verify B_c

and cast votes

- Builder reveals execution block B_e
- PTC attesters cast votes for B_e

being on time and having a valid block hash

- Proposer proposes B_{c+1}

that builds on top of { B_c

, B_e

}

- Beacon attesters verify B_{c+1}

is valid, and if there are sufficient votes to signal B_e

revealed on time, then B_e

is valid. If that is true, then RevealBoost

is applied to B_e

, and ProposeBoost

is applied to B_{c+1}

Here is a table to compare the timings with those of today.

	Today	ePBS
Time to verify consensus block validity for slot n	3s	2s
Time to verify execution block reveal on time for slot n	3s	3s
Time to verify execution block validity for slot n	3s	9s
Time to build next block for slot n+1	8s	5s

Benefits of pipelining consensus and execution

- Less network load at peak: Consensus and execution blocks are sent at different times. Blobs can even be sent earlier than reveal time.
- Less processing load at peak: Consensus and execution blocks are processed and verified at different times.
- More importantly, the execution layer has more time for state root computation withoutdiverging from Ethereum L1 state root behavior. This provides an easy execution state root performance increase without needing complex constructions like parallelized state root, less frequent state root, or trailing state root. You get this state root performance increase for free under ePBS without major execution layer overhaul and without second-order effects on light clients, L2 bridges, and other protocols that rely on frequent account and storage proofs.

[

Screenshot 2024-05-23 at 6.41.27 AM

1526×413 53.5 KB

](https://ethresear.ch/uploads/default/original/3X/9/d/9d58d9498b0d901cd675aaef0191266650ae476d.png)

Top: block production today. Bottom: block production under block-auction based ePBS.

One downside that is worth mentioning is that builders competing for the next slot now have less time to prepare for the block. Assuming honest builder behavior and no builder timing games, this time decreases from today's 8 seconds to 6 seconds.

I hope this post was able to illustrate the key point that by separating consensus and execution into different stages, the network reaps additional benefits like those mentioned above. This is one positive side effect of PBS, although it may not have been what PBS was initially intended to solve. Given that there's more time to process the execution block, it's up to the execution layer to innovate and determine what benefits it could bring, such as increasing the gas limit once state growth has a clear solution or other improvements.