I recently had the privilege of delivering a talk at ETH Chicago, focusing on the potential direction of the consensus layer post-Deneb in 2024.

Post-deneb enters a fun time, teeming with intriguing protocol problems and justifications for various solutions to them. Notably, there isn't a singular, universally agreed-upon feature destined for Electra.

Capella

:

During the Capella phase, it was evident that the primary feature post-merge should be enabling withdrawals. This fulfills the original promise of the beacon chain from 2020, allowing a validator to stake, reap rewards, and subsequently withdraw their ETH. Meanwhile, Proto-danksharding (EIP4844) demanded more time for review and proper implementation, the community prioritized withdrawals first, postponing EIP4844 to the subsequent release - Deneb.

Deneb

:

Undoubtedly Deneb's main feature is EIP4844, which scales rollup and applications that are powered by data publishing on the parent chain. Ethereum increases its scalability by introducing blob storage to decrease data publishing expenses. This solution effectively mitigates the scalability issues for the foreseeable future. There are other features like exposing consensus beacon block root to the EVM, which in turn makes staking pools or restaking protocols more trustless forward. It also capped validator activation churn to 8.

Electra

:

As we zoom in on Electra, three primary areas emerge:

1. Strengthening consensus stability
2. Increasing scalability
3. Addressing client tech debts

Let's delve into some three areas:

Increase Max Effective Balance:

Increasing validator max effective balance reduces beacon attestation p2p messages and paves the way for ePBS and single slot finality updates. It also optimizes client code performance by diminishing validator state size. Smaller validator size in state results in faster hashing and reduced memory consumption. This opt-in feature will empower node operators to determine consolidation preferences, though some design parts remain debated. Dependencies include reusing validator indices and permitting execution layer partial withdrawal. The drafting of the

EIPis ongoing, and there is a

FAQfor answering any initial questions and an analysis of

slashing penalty.

Inclusion List

:

Strengthening the chain's censorship resistance is imperative. Ethereum's censorship resistance is relatively weak, where a censored transaction is

included eventually, albeit with a degraded user experience. (See Tornado case) With enhanced interaction between execution and consensus clients over the builder override flag, the consensus client can prioritize its local transactions. However, strong robust censorship resistance is crucial, and we must relentlessly pursue it. Over time, the inclusion list has undergone extensive research, refining its mechanisms. There has been ongoing research on the implementable

mechanismthat solves free DA problems,

extension, and

game theory analysisaround the inclusion list. As for spec writing, ongoing work on

consensus,

validator, and

networkspec with further ePBS in mind. There are many open questions, like does the inclusion list have its gas limit? If yes, does it open up to mev-boost or PEPC-like side market for the inclusion list? There's a certain tradeoff between spec and implementation complexities.

Secret Single Leader Election

:

Secret single leader election ensures that the proposer assignment remains private until the assigned slot time, eliminating risks like proposers attacking each other or stealing MEV. Despite the lack of concrete evidence of such occurrences on the mainnet, this attribute is crucial for the chain's future. The candidate algorithm for SSLE is

Whisk, and it treats proposer shuffling as private throughout phases from candidate selection to shuffling to selection. There is a

specand even a

testnetlive!

Fork Choice Improvements

:

Clients like Lighthouse and Prysm re-org late blocks. Soon all the client implementations will adopt re-org late block behavior pending this

PRmerge. This behavior can be extended, allowing attesters to vote against late blocks. Proposals like view-merge,

RLMD GHOST, and

advancementsbeyond LMD GHOST and FFG GHOST underscore the fork choice's significance. The fork choice is the chain's most critical component, determining its head and underpinning its economic security. Hence, a robust fork choice is indispensable for maintaining stable consensus.

ePBS

:

Approximately 95% of blocks are constructed using out-of-protocol mechanisms like MEV-boost and relayer. These methods have associated challenges, such as centralization and potential governance threats. The ultimate goal is to integrate the relayer role within the protocol. Achieving ePBS, even in its most basic form, is a monumental task, surpassing even the complexity of the merge. Yet, it's a goal worth pursuing. Potuz, myself, and Ethereum protocol fellows have been putting in work on the specifications and, hopefully soon implementations. The design consists of:

- Staked Builders that also perform validator duties
- PTC Committee
- Max Effective Balance and its Dependencies
- Forced Inclusion List
- (Block, Slot) Voting from Attesters

Staked Builders that also perform validator duties

PTC Committee

Max Effective Balance and its Dependencies

Forced Inclusion List

(Block, Slot) Voting from Attesters

The main complication resides in fork choice and how to deal with equivocation.

Danksharding

:

In terms of scalability, we can progressively build towards Danksharding. Data availability sampling (DAS) and missing data

sample reconstruction are arguably the biggest unknowns to realize Danksharding.

PeerDas, as outlined by Danny, offers a straightforward data availability sampling network construction. If benchmarks validate its feasibility, it might be promptly implemented.

EIP4844+

:

We could consider increasing blob sizes from the current 3/6 once demand escalates and doing so on the mainnet becomes safer. Additionally, we can explore innovations like erasure coding and constructing an overlay network based on EIP4844 blobs. These initiatives allow nodes to practice sampling and reconstruction using the mainnet as a sandbox without awaiting full danksharding.

Client Code Improvements

:

Hypothetically speaking, we could allocate approximately six months for client teams to address technical debts, which seems beneficial. As the Ethereum beacon chain evolves, we must prioritize the long-term health of client code. With about 25M ETH staked, safeguarding immense value in DeFi, NFTs, and L2s, it's crucial to focus on sustaining client code health. As we learned from time again, it's hard to focus on new features while improving existing code health. Design decisions back in 2019/2020 may become stale and require a big overhaul

Conclusion

:

My two cents? Post-Deneb should be all about improving the stability of the consensus layer. The focus should be laser-sharp on the inclusion list, increasing the validator’s Effective Balance and its dependencies, and improving fork choice. As for the subsequent "F-Star" upgrade, it could be the perfect place for ePBS and danksharding increments. If blob data demand exists, we can also increase the blob target and max allocation to address short-term scalability needs.

Exciting times ahead

!