

Increase the MAX_EFFECTIVE_BALANCE



mike neuder – ethereum foundation
kiln rendez-vous – july 19, 2023

Outline

- Preliminaries
 - ▶ `MAX_EFFECTIVE_BALANCE`
 - ▶ `validator.effective_balance`
 - ▶ Partial withdrawals sweep
 - ▶ EIP-7002
 - ▶ Historical context
- Proposal
- Rationale
 - ▶ Validator set size
 - ▶ APYs & CL/EL rewards
 - ▶ Network stability
 - ▶ Single-slot finality, ePBS, mev-burn
 - ▶ Make solo-staking better
- Key takeaways



Preliminaries

MAX_EFFECTIVE_BALANCE (abbr. MaxEB)

Gwei values

Name	Value
MIN_DEPOSIT_AMOUNT	$\text{Gwei}(2^{**0} * 10^{**9})$ (= 1,000,000,000)
MAX_EFFECTIVE_BALANCE	$\text{Gwei}(2^{**5} * 10^{**9})$ (= 32,000,000,000)
EFFECTIVE_BALANCE_INCREMENT	$\text{Gwei}(2^{**0} * 10^{**9})$ (= 1,000,000,000)

- Sets an upper bound on the “effective balance” of a single validator
- Currently, the minimum balance to become a validator *is the same as* the MaxEB (32 ETH)

Preliminaries

`validator.effective_balance`

```
class Validator(Container):  
    pubkey: BLS Pubkey  
    withdrawal_credentials: Bytes32 # Commitment to pubkey for withdrawals  
    effective_balance: Gwei # Balance at stake  
    slashed: boolean
```

- Represents the consensus layer view of the validator stake
- Incremented by whole values of ETH
- Can go *below* 32 ETH (until the ejection balance of 16 ETH), but never *above* 32 ETH

Preliminaries

Partial withdrawals sweep

```
elif is_partially_withdrawable_validator validator, balance):
    withdrawals.append(Withdrawal(
        index=withdrawal_index,
        validator_index=validator_index,
        address=ExecutionAddress(validator.withdrawal_credentials[12:]),
        amount=balance - MAX_EFFECTIVE_BALANCE,
    ))
```

- Removes any excess balance from the validator automatically
- Sends the rewards to a “withdrawal credential”, which is an Ethereum address (EOA or contract)
- It takes about 6 days for the sweep to finish one round (0.015 ETH for most of these withdrawals)



Add EIP: Execution layer triggerable exits 104

ethereum:master ← djrtwo:el-exits

opened 🌐 May 9, 2023 L djrtwo +369 -0

- Allows validators to trigger exits from their withdrawal credential rather than their validator signing key (a BLS key-pair used for consensus layer signings)
- *safety* – if you lose your validating keys, you can still withdraw
- *convenience* – programmability of the exits

Preliminaries

Historical context

Misc

Name	Value
MAX_COMMITTEES_PER_SLOT	uint64(2**6) (= 64)
TARGET_COMMITTEE_SIZE	uint64(2**7) (= 128)
MAX_VALIDATORS_PER_COMMITTEE	uint64(2**11) (= 2,048)

- Attesting committee for a slot was broken down into sub-committees (called just `committee` in the spec) for the sharding design
- Each committee needed to be majority honest for the security of that shard (meaning having large validator balances would be difficult)
- With Danksharding, we now just use committees to aggregate attestations, so we only need one honest aggregator

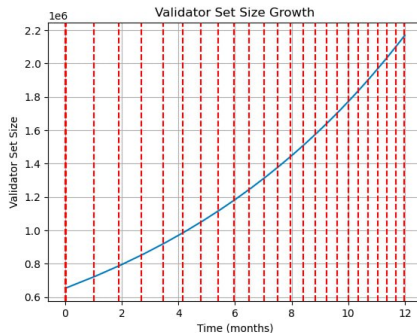
- Increase the MaxEB from 32 ETH to 2048 ETH
- Allow for arbitrary execution layer partial withdrawals

A Modest Proposal



Rationale

Validator set size

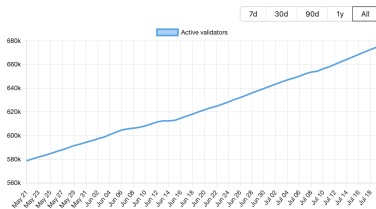


- If the activation queue remains full, the validator set grows exponentially

Rationale

Validator set size

Active Validators

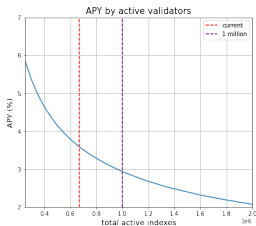


Supply Staked

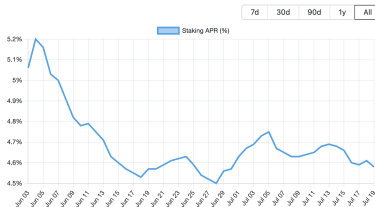


Rationale

APYs & CL and EL rewards

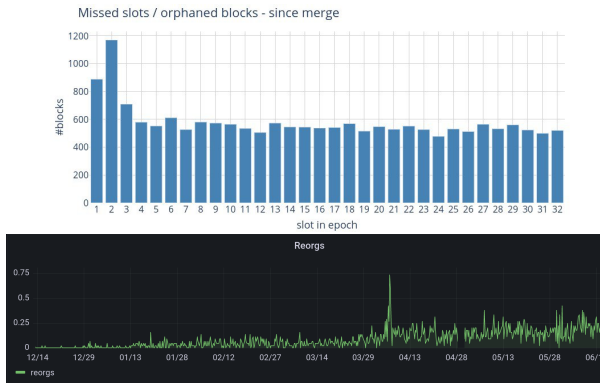


Staking APR



Rationale

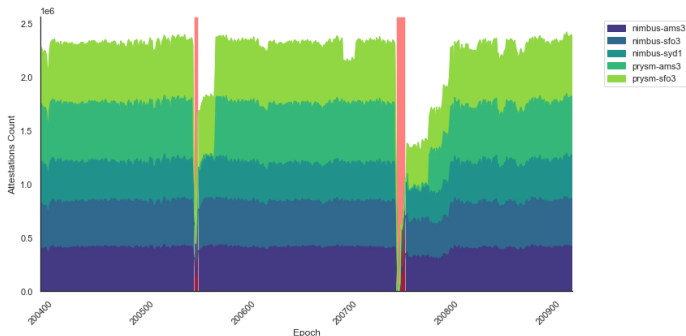
Network stability



- Reorgs happen more frequently
- Epoch processing is especially unstable

Rationale

Network stability



- Non-finality event on May 11-12 partially attributable to processing so many activations
- Large validator set means: large beacon state, long signature aggregations, and more p2p networking overhead

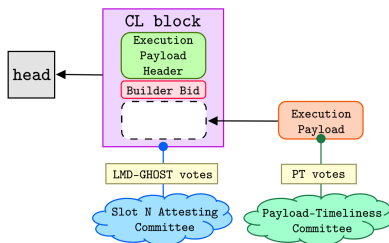
Rationale

Single-slot finality

- ☞ **Bad news: hybrid consensus mechanisms actually have many unavoidable problems**
- Current consensus mechanism has issues around UX, safety, and protocol complexity
- How do we do SSF?
 - ▶ Rotating validator set
 - ▶ Validator set capping (how do you choose)
 - ▶ Change validator economics directly (negative issuance?!)
- All of these changes modify core elements of the protocol

Rationale

ePBS & mev-burn



- ePBS designs change consensus rules, add some complexity
- ePBS directly enables mev-burn, which is potentially a better way to address the validator economics

<https://ethresear.ch/t/payload-timeliness-committee-ptc-an-epbs-design/16054>

<https://ethresear.ch/t/mev-burn-a-simple-design/15590>

Rationale

Make solo-staking better

POOL	NETWORK PENETRATION	CONSENSUS CLIENT DISTRIBUTION	BLOCK SPACE DISTRIBUTION	BACKWARD APR %
Lido 30 entities ▾	32.40 %			4.79 %
Coinbase	9.85 %			4.74 %
Binance	5.90 %			4.60 %
Kraken	3.56 %			4.36 %
Rocketpool 1921 deposit addresses ▾ ⓘ	3.06 %			4.61 %

- Pools provide auto-compounding and flexible staking amounts
- This quickly overcomes the fixed-rate fee taken by the pool
- Solo-stakers should also have these benefits, without needing to join a pool

<https://www.rated.network/?network=mainnet&view=pool&timeWindow=1d&page=1>

<https://blog.rated.network/blog/solo-stakers>

Key takeaways

- Increasing the MaxEB is an easy tech-debt fix that slows the growth of the validator set
- A much better solution than directly changing the validator economics
- Added benefit of making solo-staking better
- Fits perfectly into the roadmap of SSF, ePBS, and mev-burn



Thanks!