This builds upon the work of @mikeneuder on increasing the validator balance cap by making “validators” further legible to the protocol.

I’d like to discuss how enabling validators to self-certify their affiliation to a staking pool can improve staking decentralization, even if any validator is free to specify fake data

(more on this later). This does not involve any KYC, fully supports solo stakers and their privacy, and is valuable even without any economic changes.

The core of the argument is this:

By enshrining a means for validators to provide their affiliation information in a manner legible to the Ethereum protocol, any large staking entity that has their validators falsify this information can be legitimately called a Sybil attack on the Ethereum protocol

.

By having this information be legible to the protocol, and therefore enabling the protocol to potentially

enforce rules around validators with shared affiliation, this moves the discussion of what should happen to large staking providers to a consensus-rules discussion

, rather than fuzzy social-layer debates.

For example, any social-layer pressure to have staking providers self-limit becomes a protocol design decision, rather than a social mob harassing the largest staking provider of the day. This ensures those rules remain credibly neutral (every staking provider has to abide by them equally), and the rules are selected as any other protocol change is: by node operators choosing which set of protocol rules to enforce.

Won’t validators just all pretend to be unique solo stakers?

This is the obvious problem with any self-certification scheme. As with anything in consensus protocols, there are two possible answers: Cryptoeconomic measures, and social layer measures. This proposal includes both types of measures and are described in further details below.

NOTE: “Validator” does not mean “32ETH”

I believe the “32ETH per validator” design of Ethereum has framed the discussions on staking dynamics in unproductive directions.

For the purpose of this discussion, we will assume that the proposal toincrease the validator balance cap has already been implemented and is part of the consensus rules.

Currently, validators look like this:

[

3324×2804 363 KB

](https://ethresear.ch/uploads/default/original/2X/e/ed4fcc8a0084e140b6cc88a78efbe49e075cc7f8.png)

Each 32ETH validator is just a portion of a larger entity, forming a “logical validator”. “Logical validator” doesn’t necessarily refer to a single node software instance. It refers to a unique combination of validator attributes which, by their uniqueness, make this validator valuable to the Ethereum consensus. In other words, running an additional 32ETH validator on the same machine is not valuable to the Ethereum consensus compared to simply adding those 32ETH to the balance of the existing validator running on that same machine.

After the balance cap is increased, validators look more like this:

[

1280×1080 107 KB

](https://ethresear.ch/uploads/default/original/2X/3/309b4a26002aeb628944dec8bdfb9126e70a61e5.png)

In other words, the multiplicity between that the beacon chain refers to as “validators” and what a layperson would understand as a “validator” gets closer to one-to-one. Beyond the network performance benefits, this is an additional benefit of the proposal to increase the validator balance cap.

For the purpose of this proposal, “validator” will refer to “logical validator”. It represents a single node on a single machine operated by a single operator under a single entity, but is decoupled from how many ETH it has staked.

What would validator metadata look like?

New validators past a certain block height would be required to provide identification/affiliation information as part of signing up to be a validator. Every validator is free to insert unique IDs in any of the fields

, and solo stakers are encouraged to do so

to ensure their identity is seen as unique by the protocol.

This metadata could be structured as follows:

- GovernanceEntity

: The hash of a public key. This part of the ID represents the governance entity that this validator is a part of.

- Department

: Second-level numerical ID representing the branch (within GovernanceEntity

) that this validator is a part of. This would allow support for sub-DAOs and hierarchical organizations.

- HardwareOperator

: Top-level numerical ID representing the entity operating the hardware where the validator is running. This can represent a cloud provider.

- SoftwareOperator

: Top-level ID representing the entity operating the validator software stack (sysadmin). Note that the same software operator may operate validators for multiple governance entities.

- NetworkOperator

: Top-level ID representing the entity operating the network that the validator hardware relies upon. For solo stakers, this would be their ISP.

- Nonce

: A number that uniquely identifies multiple validators under the same HardwareOperator

/SoftwareOperator

/NetworkOperator

. Without this, two validators on similar setups would have identical IDs.

- Timezone

: Timezone ID. (Some staking providers already ask for this information from their node operators.)

- Jurisdiction

: Country code or some ID representing the set of real-world laws that the validator is abiding by.

The intent of these fields is to allow various dimensions of decentralization to become legible to the protocol. All fields are optional, and can be set to a special value (e.g. 0xFFFFFFFFFFFFFFFF

) to opt out of revealing this information (this special value is treated as unique from all other validators for comparison purposes). Solo stakers operating a single validator on a single machine may set all fields to this value.

How will existing validators set their metadata?

All validators that existed prior to this proposal being implemented have all of their metadata implicitly set to 0x0000000000000000

.

Similar to the withdrawal address type update message, a new beacon chain message type is created that allows validators to set their metadata. This message can only be sent by validators that currently have no metadata set, so it can be sent at most once per validator.

After a certain block height months past the introduction of validator metadata, a small reward penalty is imposed on all

validators that still have their metadata set to 0x0000000000000000

.

How are IDs mapped back to human-readable concepts?

Simple way

: Assign IDs like EIP/ERC numbers are assigned today. Much like EIP numbers, people will naturally remember the significant ones without having to look them up.

Harder way

: Enshrine a “validator ID registration contract” that anyone may make a transaction to (providing a human-friendly string) and which returns a unique number. The contract doesn’t require fees (other than gas for the registration transaction). The string doesn’t have to mean anything or reveal any personal information; it just has to be unique. Then, when a validator registers with metadata, it uses this number, and provides a Merkle proof that the number exists within the contract.

Again, note that all metadata fields are optional. If the user doesn’t wish to register their own ID, they can set the field to 0xFFFFFFFFFFFFFFFF

. This special value is considered as different from all other validators’ value, including other validators which also have 0xFFFFFFFFFFFFFFFF

. (For the programmers out there, this is similar semantics as how NaN

operates in the IEEE Standard for Floating-Point Arithmetic.)

What prevents validators of competing staking pools to pretend to be part of a pool that they’re not a part of?

Governance entities are represented by the hash of a public key. (This can be a key split up in multiple parts.)

Cryptographic approach

: To certify that they belong to a governance entity, a validator must provide a signature from the public key of that governance entity in order to be accepted into the beacon chain in the first place.

Slashing approach

: When a validator self-certifies as belonging to a governance entity that they are not a part of, that governance entity may sign and broadcast a message to force-exit the validator (with or without a slashing penalty).

What about distributed validators (DVT)?

Some fields (HardwareOperator

, SoftwareOperator

, NetworkOperator

, Timezone

, Jurisdiction

) could be structured either as bloom filters, either as a hash of the concatenation of multiple values. This way, they can (lossily) encode multiple values, despite only retaining a fixed amount of bytes per validator. From a protocol perspective, this makes every subset of these fields appear unique.

What prevents validators of large staking pools to self-certify as solo stakers? (Sybil attack)

This section describes the social layer dynamics of this question; the cryptoeconomic possibilities are described later.

In short, we need to establish the following social norms:

- Any large staking provider that obfuscates the flow of funds from delegators to the deposit contract will be slashed

by the social layer. * In other words, the interface for delegated staking should be: deposit ETH into contract, observe a corresponding amount of ETH flowing to the deposit contract as a direct consequence (no Tornado Cash in the middle).

- Importantly, this does not prevent delegators from having financial privacy.Privacy for the weak, transparency for the powerful.
- This does mean centralized exchanges should not participate in staking, unless they do so transparently, on a one-to-one basis with customer requests. Coinbase gets the closest to that ideal, but there is still no transparency as to whether each cbETH issued is truly the outcome of customers actively wanting to stake their ETH.
- In other words, the interface for delegated staking should be: deposit ETH into contract, observe a corresponding amount of ETH flowing to the deposit contract as a direct consequence (no Tornado Cash in the middle).
- Importantly, this does not prevent delegators from having financial privacy.Privacy for the weak, transparency for the powerful.
- This does mean centralized exchanges should not participate in staking, unless they do so transparently, on a one-to-one basis with customer requests. Coinbase gets the closest to that ideal, but there is still no transparency as to whether each cbETH issued is truly the outcome of customers actively wanting to stake their ETH.
- Any large staking provider that does not provide honest metadata for its validators will be slashed

by the social layer. (This is easy to verify thanks to the first social norm.)

Creating these two social norms also has the downstream effect of making it risky for individuals to participate in any such staking enterprise, as their capital is now at risk of being slashed. This hopefully prevents such schemes from getting off the ground in the first place.

Why hasn't social pressure worked in the past?

I believe that a large part of this is simply that validator identification metadata is not currently a protocol-level concept

. The protocol is

effectively under a Sybil attack right now, in the sense that large staking providers are "pretending" to be many individual 32ETH validators. But they do this for one of two reasons:

A. Because they have no other choice: This is the only type of validator that the current rules allow, and they have more than 32ETH.

B. Because they actively want to Sybil attack the protocol.

I'd argue that all major staking providers today

are doing this because of reason (A). Once weincrease the validator balance cap, only staking providers that are doing this for reason (B) remains.

What about small staking providers that have their validators falsely claim to be as solo stakers?

Those will inevitably crop up, since they are the economically rational thing to do (if you ignore social slashing risk). The goal of the social layer is to educate individuals to understand why they should not participate in such schemes.

Should any such entity get large enough to matter, they will eventually be identifiable by fingerprinting/forensics techniques. This can include factors like IP addresses and subnets of the nodes, commonality in node software and version upgrade patterns, commonality in attestation misses, and investigative reporting from users (eg. by creating trial deposits and withdrawals and watching for deposits/withdrawals of corresponding amounts on the beacon chain). Once identified, they are susceptible to be socially slashed.

What if a large provider forks itself into a differently-named GovernanceEntity

but uses the same set of governors?

It depends on the rebranded version's approach to specifying validator metadata.

If the rebranded version specifies validator metadata truthfully, this is where the structure of the metadata comes in handy. Separating out governance-related identification from operator-related identification means that it's possible to detect overlap across multiple governance entities, and have the protocol treat those governance entities as identical.

Alternatively, if the rebranded version specifies falsely-new IDs for operator metadata, then that is an equivalent attack on the protocol as a small staking provider would. It is up to the social layer to detect this. Detecting this via forensics should be fairly easy, since the nodes would be managed under the same infrastructure.

What if a staking provider does

manage to get large while evading detection?

In order for this to happen, this hypothetical large provider must have:

- Carefully distributed its validators geographically
- Carefully distributed its validators in different IP subnets, including likely residential IP regions
- Carefully avoid registering any real-world legal entity that could be investigated and/or prosecuted
- Carefully chose a diverse set of operators each with near-perfect opsec

… And if all of that happens, well:Mission. Fucking. Accomplished. Ethereum is now very resiliently decentralized.

Node operators may never agree to economic policy changes; is this proposal still worth doing despite that?

Yes, this proposal is valuable even without any economic changes or implications derived from validator metadata

. Having validator metadata enriches the Ethereum protocol by giving it the following capabilities:

- The ability to measure decentralization

by another way than current forensics-based approaches. The forensics-based approach will always work, but now it can be compared against self-certified data. The difference between those two methods can be used to estimate the accuracy of either method, and can potentially be used to catch large staking providers that do not correctly identify their validators.

- Better data on decentralization within honest staking providers

. By having structured fields like Jurisdiction

and Timezone

and HardwareOperator

/SoftwareOperator

/NetworkOperator

, we can estimate Ethereum's level of decentralization along these respective axes individually. Forensics-based approaches are less precise here; for example, it is difficult to tell how many different software operators are operating all the nodes running in AWS. While solo stakers are free to use 0xFFFFFFFFFFFFFFFF

for these fields, this still allows precise measurement of these axes of decentralization within

a staking provider.

- A clearer view on the proportion of stake which is aligned with possible future consensus changes

. In world where validator metadata can be provided but for which there are no cryptoeconomic consequences for doing so, the possibility

of rules being imposed later means that we can observe the behavior of staking providers to gauge how aligned they feel about the current or future direction of consensus rule changes. If they decide to not provide metadata, then a conflict which was previously implicit is now made explicit.

Cryptoeconomic design space of validators with metadata

This section aims to explore cryptoeconomic measures that attempt to simultaneously encourage validators to be honest about their self-certified metadata, while penalizing staking entities that gain disproportionate power over the total stake.

I am not actively proposing that we do any of these changes. As noted earlier, I believe this proposal is valuable to implement even without any cryptoeconomic changes

. The purpose of this section is to explore the design space of validator data on cryptoeconomic incentives, which could provide further motivation to implement these changes. Additionally, even if no cryptoeconomic change is implemented, the mere possibility of them being implementable at all is valuable

(see section above on this).

Tweak slashing penalties for correlated failures

Ethereum currently has a correlation slashing penalty which aims to punish centralization by slashing more capital from validators the larger the proportion of validators are operating incorrectly (or dishonestly). The intuition is that this will force validators to consider this possibility as a long-tail risk, and self-limit such that the likelihood of accidental correlated failure of their validators results in lower slashing penalties.

This penalty has so far proven ineffective at convincing staking providers to self-limit before hitting certain critical stake thresholds. It may instead be possible to adapt this penalty to take operator data into account

, and turn it into an incentivization mechanism for motivating operators to properly self-certify their metadata. For example:

- Reduce

the per-validator slashing penalty for simultaneous failure the more validators with the same hardware/software/network operator fail at the same time.

- Increase

the slashing penalty for validators that do not

share the same hardware/software/network operator yet also fail at the same time.

This simultaneously provides a carrot for validators to self-certify their operator correctly (they lower their own long-tail slashing risk by doing so), and a stick for validators to not falsify their hardware/software/network operators (if they falsify their data and their seemingly-unique validators end up failing simultaneously, they will get slashed more than they would if they had truthfully specified their validator metadata).

Decrease attestation rewards past critical stake proportion thresholds

This is the obvious one in order to avoid over-concentration of any particular staking provider. Similar to an informal proposal from Vitalik Buterin to have staking providers progressively charge higher fees the larger they get (past certain critical thresholds), the protocol can impose it on them. There is a purely logical reason for it to do this:

N

attestations to the validity of the chain head from validators under a single entity are less valuable to the protocol

than 1 attestation to the validity of the chain head from validators under N

entities.

If an attestation is less valuable to the protocol, it should be less rewarded.

Ethereum values decentralization and plurality, and this encodes this value into the consensus rules.

How rewards should decrease as a single entity grows is for people better at incentive design to correctly model out, but my sense is that a curve much like progressive tax schemes may be the appropriate template: no taxation at lower thresholds, and then the tax rate progressively increases as the taxpayer passes certain thresholds.

The other relevant question is how to define “same entity”. This proposal contains many aspects of validator identity (governance entity, operator, jurisdiction, etc.). It may be possible to come up with a “likeness score” as a function of these fields in order to quantify how redundant each validator is to every other validator

, although performing this computation and storing the resulting matrix may be too computationally intensive for nodes to perform; unless perhaps the validator set is capped, or significantly reduced in size thanks to increasing the validator balance cap.

Incentivize creation of uniquely-identified validators

Creating a validator with unique metadata attributes (GovernanceEntity

, HardwareOperator

, SoftwareOperator

, Jurisdiction

, etc.) could get boosted rewards in the first few slots of their existence, as a function of how unique their metadata is relative to the current validator set.

The rewards would have to be small enough to discourage large staking providers from “rebranding” their existing validator

set to a new name (which would come with a new GovernanceEntity

ID and therefore appear new to the protocol). This means the reward must be smaller than the opportunity cost of attestation rewards from leaving a validator active vs the delay of exiting and re-entering the beacon chain.

Criticism of this proposal

Some criticism of this proposal that I can foresee, along with some rebuttals:

"This proposal assumes that large staking entities will act honestly!"

Indeed. This assumption is based on the observation that this appears to be the case today. This can be explained by either:

- Coincidence
- Conscious decisions from delegators
- Lack of reasons to act dishonestly

Adding validator metadata by itself does not add a reason to act dishonestly. It is adding (some) cryptoeconomic changes that depend on this metadata that may add a reason to do so. Therefore, this objection is only an argument against the cryptoeconomic changes that validator metadata could

enable; it's not a valid argument against the addition of metadata.

Large staking entities have also shown a willingness to care for the health of the Ethereum ecosystem in the past; for example, to contribute to better client diversity.

"This proposal relies too much on the social layer for enforcement!"

I agree. By itself, this proposal is insufficient to fully secure Ethereum's future and values. However, hopefully this proposal is at least convincing enough that having even partially-correct

validator metadata is a net improvement over the status quo, and this does not require social layer enforcement (it helps, but it's not necessary).

"This proposal paints an even larger target on consensus client teams to become captured"

Agreed. My counterpoint to this would be that this process is already happening. Some consensus client teams are already part of large staking providers themselves, and get financial upside and equity from them.

This proposal makes this problem worse, but also forces this capture process to become more explicit, and its staking footprint more transparently visible onchain.

"Why not maintain this information by a third party or a non-enshrined contract outside of the consensus protocol?"

If the goal is to only add data, and to not perform any economic changes based on that data, then this could theoretically be possible. However, I'd argue two things:

- The social pressure for staking providers to standardize adding their information is much weaker compared to the possibility of adding penalties for not eventually specifying it; it's likely that only a minority of the total stake would consider voluntarily registering in an out-of-protocol registry.
- The mere possibility

of later being able to change the protocol economics based on this information is valuable in the present (to see which staking providers are aligned with likely future consensus changes).

Conclusion

I believe this proposal is a natural extension of the one toincrease or remove the validator balance cap What that proposal does, in essence, is to make logical

validators individually legible to the protocol. This proposal extends this concept further by adding optional metadata to these validators, without requiring KYC or impacting the privacy of solo stakers.

Having this data on its own is valuable, even if not all validators provide it. It provides valuable insights into the network's health, decentralization, and values alignment. It provides a schelling point for the social layer to rally around: Large staking

operators must identify their validators honestly, as doing anything else is now an explicit Sybil attack on the protocol

.

It also enables potential cryptoeconomic changes that can more precisely reward attestations as a function of how valuable they actually are to the Ethereum consensus. In doing so, this change can be used to reflect the Ethereum network's values of plurality, resilience, and decentralization.

Thanks for reading!