The following is a largely application-layer counter-proposal to Enshrined Eth2 price feeds, though there are some protocol-layer features (eg. larger graffiti, in-VM accessibility of the validator set) that could help make it easier to implement.

A lot of existing and proposed oracles (Augur, Kleros, UMA) work in roughly the same way:

- [Initial reporting]

Suppose a result on some event E needs to be reported. First, anyone (or at least one of many actors) can make a report R by putting down a deposit. The deposit is only returned if the oracle ends up returning R.

- If the report is unchallenged, the oracle returns R.
- [Escalation]

One of many actors can challenge the report, and there is some escalating challenge process where the deposits required to challenge challenges keep increasing. If there are enough challenges with high enough deposits, the system starts a global coin vote.

- [Global coin vote]

In a global coin vote, everyone who holds some coin can vote on the report. The oracle reports the winning result, and everyone who agreed with the winning result sees their coin balance greatly increase and everyone who disagreed sees their balance greatly decrease.

- [Market backstop]

In the event of a 51% attack on the system, anyone can fork the system and create a “parallel universe” where the minority report won and the minority coinholders see their balances greatly increase and the majority coinholders see their balances greatly decrease. If the minority answer was actually correct, the theory is that the market will value the forked system more highly because its coinholders have a track record of honesty even under high risk, and the original system will be less important than the fork.

There are two ways to measure security of such oracles:

- Budget

: how much money do you need to attack the system?

- Cost

: how much does attacking the system cost you?

These oracles have a budget requirement and

cost of attack roughly equal to the market cap of the oracle token (eg. REP, MKR, PNK). The unincentivized coin vote oracle proposed in the enshrined feed proposal has a higher budget requirement, because ETH has a higher value than tokens, but zero cost because there is no penalty to voting incorrectly.

The goal of this proposal is to combine these two systems, creating a system with a budget requirement consisting of a substantial portion of all staked ETH plus

an oracle token, and a cost of attack based on the oracle token.

The proposal

An oracle where to participate in the global coinvote, you need to have both

32 ETH worth of oracle tokens and

an active validator slot. That is, one unit of participation requires both signing a message with a key that controls oracle tokens and signing a message with a key that is connected to a validator slot.

How to verify?

There are a few goals here:

- Prevent validators from easily selling off their oracle participation rights as a separate token (as that would break the security of the scheme)
- Avoid adding centralization/pooling pressure to validating itself
- Avoid imposing excessive risk on validators

One option is to require signing a message with the signing key of a validator (this could even be done by checking the graffiti of the last beacon or shard block that they created). This goes pretty far in preventing selling participation rights, because giving someone else your signing key lets them (profitably) slash you. However, it does increase the complexity of validation itself.

A second option is to have some kind of signature scheme where instead of the key being

the signing key, the key is a pair of signatures of conflicting block headers (so the key could be used to slash, but not to commit any other mischief to the chain). This would not increase people's willingness to delegate participation rights, but in the event that people do so anyway, it would decrease risks, as they would only be giving out a self-slashing key, not a key that could be used to attack the blockchain.