Thanks to @EOmatan, @AlonChayet , @IttayEyal , @obpeo and the all Lightblocks Labs team and a special thanks for @sreeramkannan, @JustinDrake and @vbuterin for the inspiration

Hello world;

Ethereum and Bitcoin faced challenges that were overcome with a very cautious approach, resulting in an extraordinary case study for global human coordination. Adding real-world data through oracles as another layer on top of it is a very promising next step. However, so far, we have taken this step forward without the same caution that is fundamental to the journey toward a decentralized ecosystem. Perhaps we were too close to the problem

, focusing only on the outcome - making data accessible - and not being mindful enough about the process allowing it.

The problem with the problem

The term “oracle problem” is misleading as it conflates the problem with the solution. The actual problem is the inherent inability of blockchains to dynamically interact with mutable, external data sources, while the “oracle” refers to the proposed solution. That is, an oracle refers to the infrastructure that provides blockchains with real-world data in a trustless and secure manner.

Naive solutions come with many problems of their own, including latency, accuracy, cost, and more. These are not the oracle problem, but problems of oracles, and we as an industry haven’t been very mindful about this distinction

. As a result, we are now at the point where we have some very interesting solutions for the problems of oracles, but we don’t have a sufficient solution to the actual oracle problem.

After shifting the perspective to the right problem, we need to recognize that the latest challenges we tackled as a community were derived directly from our vision to eliminate a central point of failure and power. Solving the oracle problem with this dedication in mind is a journey that must follow the same standards.

[

4 (1)

1214×1600 116 KB

](https://ethresear.ch/uploads/default/original/2X/e/edc61b3ee5c274d504b987765fd3c01878d111e8.jpeg)

Standards Introspection

The Ethereum community holds itself to high standards for what we expect from a network.

Let’s unpack this with a few examples of protocol standards that captured our mindshare recently as they touch some of the most agreeable underlying values of the community:

- It bothers us when there is a concentration of stake in a few operators.
- We talk about "client diversity”, a topic not even on the radar for most projects.
- We see our home-stakers as incredibly valuable and cherish them as such.

These are just a few examples of our standards.

What we really want from a protocol is:

1. It should work as long as the participants follow

the rules.

1. It should make a good case for why the participants will follow

the rules.

1. It should do that with minimal assumptions about the participant’s behavior

so that we can trust it to last even with different circumstances.

We are very critical and uncompromising on that because we know—and we don’t let the day-to-day challenges cause us to deviate from our values—that our ultimate goal is to build a global, trust-minimized settlement layer.

To better understand the journey and challenges in finding a solution to the oracle problem, we should examine the significant work previously done in this field and how it aligns with our standards.

Previous Work

There has been significant work in the oracle space, characterized by good intentions, progress, and innovation. Some solutions focus predominantly on user experience, treating the oracle as merely a feature that makes data accessible to smart contracts, thereby losing sight of the core properties of decentralized applications.

Other solutions emphasize decentralization through their architecture, by distributing the process of fetching, verifying, and delivering data among multiple participants. This is a step in the right direction, but they have yet to convincingly overcome several challenges.

1. Their network is operated by a selected, well-connected, and semi-internal permissioned set of nodes. That's understandable - it's hard to bootstrap a trustless system, but a good protocol needs to explain why it will pass the bootstrapping phase, not only why it will be good when it passes that.

2. They retain centralized ownership of their smart contracts due to a lack of effective governance.

3. The computation and incentives among the participants are not fully transparent or accessible to the public. So, their crypto-economic security hinges on the trust placed in the corporation that owns the oracle.

[

Twitter post - 3 (1) (1)

1200×675 53 KB

](https://ethresear.ch/uploads/default/original/2X/e/e53f91225a3c30719fcdf1747fb6b7effdc722d8.jpeg)

Essentially, we're at a point in DeFi where oracles are many protocols' weakest link because we trust them based on reputation and history in a striking contrast to blockchains, which we trust due to their inherent design. Fortunately, although there were many oracle malfunction incidents with some causable losses for users - we haven't suffered from an FTX kind of incident regarding oracles.

For those of you who are comfortable with the way things are because of the lack of such an incident, we would love to introduce you to another battle-tested, efficient product with a fantastic business model that hasn't failed just yet.

Obviously this is not how we do things around here, we aren't chickens. - What?!

A completely Un

related tale

Once upon a time, on a farm far away, there was a chicken named Alice who lived what could only be described as the dream life. The farmer, a paragon of punctuality and care, ensured their food was always delivered on the dot, fresh, plentiful, and best of all, completely free of charge. What chicken could ask for more? Alice, having hatched into this utopia, soaked up every second of her idyllic existence. For 1000 blissful days, she enjoyed the unparalleled generosity of her human benefactor, but on day 1001, everything changed. The farmer, the very architect of her paradise, approached with a machete in hand and abruptly ended Alice's life.

Well, that was an unexpected ending to the story - wasn't it?

This is akin to Hume's 'problem of induction': We are told to trust oracle X to execute well tomorrow since it executed well yesterday. This is not the kind of reasoning we are looking for.

[

trust design meme

958×500 69.1 KB

](https://ethresear.ch/uploads/default/original/2X/3/369783344325b0eb0d4385cd1841d89c559708cc.jpeg)

eoracle foundations

'A Problems First' Approach

In light of our standards reflection and the evident misalignment in previous work, we advocate for a first principle approach to tackling the oracle problem. This strategy zeroes in on solving the fundamental issue of blockchain's interaction with external off-chain data in the most rigorous and demanding manner possible. By adhering strictly to the high standards we've set, this approach prioritizes addressing the core challenge of trustless data integration. Other challenges around data and implementation are crucial, respected, and important to solve, but they can't be of any significance if countering the essence of oracles in a decentralized ecosystem. These problems should be solved on top of and not instead of the core solution.

How do Ethereum values translate to oracles?

Armed with a new perspective, and laser-focused approach on the problem, we should now make a list of guidelines our protocol should follow in order for it to align with Ethereum standards and be able to rise to its crucial and complex role.

1. Decentralized Ownership
2. Oracles as evolving protocols need a way to adapt in the face of industry changes. A fully immutable protocol is not practical, and a decentralized architecture is only worth doing with decentralized ownership.
3. Minimal Trust Assumptions
4. The security model should avoid additional assumptions on the entities involved other than - 'they like making money and dislike losing it

'.

1. Permissionless Participation
2. All roles within the protocol should be entirely open for anyone to participate.
3. Maximum Visibility of the Process
4. To trust how data is being aggregated, we must provide an immutable and "accessible for all" layer for the process to occur on.
5. Transparent Incentive Mechanism
6. The positive and negative incentives should be known to all and tracked publicly, as they are the driving forces of a well-designed system.
7. Proper Bootstrapping Model
8. The design should present a compelling story about how the protocol will work in a fully decentralized way and how it will get there.

Justin Drake's Suggestion for Enshrined Eth2 Price Feeds

We came across @JustinDrake suggestion about an enshrined Ethereum stakers' oracle, which sparked our thinking. The suggestion is about, in essence, adding another field to [BeaconBlockBody]

for price feeds. This original approach has some performance limitations. Most importantly as@vbuterin pointed out, it increases the workload and potentially raises the entry barrier for home stakers to take part, once again emphasizing Ethereum's uncompromising approach to security and decentralization.

On the other side, this suggestion spotlights a new, bright and refreshing approach - use the same group of stakers, that we already trust (under minimal assumptions) and have them be the trust quorum for delivering data on chain. This way we can minimize additional trust assumptions for this crucial task.

There was a lot to do in order for it to become something practical but the good news is that we realized that the biggest downside of the suggestion might be solvable - we don't actually need all stakers, just a permissionless diverse sub-set of stakers that are up for another "job".

Enter EigenLayer - The Trust Marketplace

The change from 'work' to 'capital' as a commitment guarantee introduces the PoS ecosystem with the benefits of the efficiency of money, one of its features being that you don't have to consume it in order to create value with it.

We will stop the hand waving here because we still struggle to explain to our moms the innovation of restaking in traditional finance terms.

In the eyes of a protocol, which has some kind of decentralized security model, trust is in fact the set of individual entities coordinating

and their incentives to follow the process.

For this trust-seeking protocol, EigenLayer suggests a Trust Marketplace. Here, we have to ask - Is some trust better than others?

The short answer is - Yes.

We should think of two ways to evaluate trust:

Economic Trust

- amount of capital at stake = Quantity

Diversity Of Trust

- number of different nodes and their decorrelation = Quality

In order to achieve security and liveness in a decentralized way, a protocol will need a sufficient amount of economic trust and a diverse, well-balanced set of nodes to carry it.

Although it’s theoretically possible to achieve a decentralized trust quorum for a protocol, in reality, only Bitcoin and Ethereum have achieved a sufficient level of decentralization. This is despite numerous protocols attempting to do so. This struggle could be referred to as the challenge of proper bootstrapping. Some great oracles fall into this category of having the right idea, and have a good explanation of how they will operate once they get past bootstrapping but don’t have a compelling plan about how they’re going to get to there.

We believe that EigenLayer and restaking is exactly the missing piece for a superior oracle design - that could actually get there.

Stepping into the Arena

In creating the architecture for a fully decentralized oracle network, our design is guided towards system that is inherently aware of its operational dynamics. This necessitates a comprehensive tracking mechanism for participant actions, their incentives (both positive and negative), and the resulting outcomes, all within a framework that guarantees transparency and immutability. In the near future, it should be run and managed by users and not a central authority.

Ideally the best oracle infrastructure is Ethereum itself, where we have a set of nodes that take some observations about off-chain data, declare (submit/post) it on-chain for a smart contract that applies some aggregation and verification logic on the votes and concludes the aggregated and hopefully (if built correctly) accurate and secure outcome for smart contracts to use. This simple architecture has everything we want as guidelines for the protocol. Unfortunately, this solution is infeasible, the cost and latency resulting in the recording of all this process on-chain would be way above what a useful oracle can allow. So we needed to come up with a database that we could afford to offload the computation

to in order to minimize the target chain activity.

Does that sounds like a scaling problem? We think so.

Scaling problem? → Scaling solution

We concluded that establishing another PoS blockchain as an independent data and settlement layer will allow us to build an oracle that is mindful of the process, fully transparent, and eventually completely decentralized and owned by the users. This PoS blockchain, that we named “eoracle chain”, together with the task of fetching data, will be operated by Eigen operators - Ethereum validators that are willing to restake their ETH on the protocol and secure the network and the data processes. eoracle chain will allow us to maintain decentralized incentives for the oracle operators and to have all the calculations on-chain instead of on the client’s software. To this end, we store all their relevant data submissions and the aggregation of their data, and either reward or punish them according to their behavior.

After outlining our desired design, we’ll now address some of the challenges and opportunities associated with building an PoS based oracle with EigenLayer.

Challenges and Improvements

Who said Reconfiguration?!

Changing the set of validators over time is classically known as reconfiguration

. Known solutions basically involve agreeing on updates to the committee as part of the consensus process for new blocks. That is, the decision of a block i

includes the details of the committee that will generate block i+1

.

However, the validators of our chain are not determined on our chain but rather on Ethereum through the restaking and unstaking operations there. Therefore, each block includes a reference (hash pointer) to the latest Ethereum block. This implicitly determines the committee for the next block: the set of restakers at that Ethereum block.

This is where the problem diverges from classical instances: The committee defined in that Ethereum block is temporary, and becomes deprecated once its members unstake. If this occurs, our blockchain might remain without an active committee.

We overcome this by introducing a novel design we call Aegis,

the algorithm behind eoracle chain, that protects (like the mythical shield) a derivative chain (eoracle chain) using a primary one (Ethereum).

Aegis relies on routine checkpoints between Ethereum and Aegis. If Aegis is unable to update due to asynchrony, we trigger a reset on both chains to maintain progress without violating other consensus properties.

[

Untitled

2304×526 98.9 KB

](https://ethresear.ch/uploads/default/original/2X/9/9535b02546a77ecb1d55a5e52bbf197664f9a794.jpeg)

Can you add Long Range Resistance in the mix?

A central challenge of Proof of Stake blockchains is that of Long-Range Attacks

. In a PoS blockchain, operators get to extend the chain based on their current staked tokens. This allows stakers from a long time ago to quickly produce a fictitious long branch of the blockchain. For a newly joining node, the validity of this branch is indistinguishable from the correct one generated over time by the network. Ethereum overcomes this by having a vast network of operators that monitor the blockchain and will discard such a fictitious branch. A newly joining node can always find a critical mass of reliable nodes to help with this.

However, in a nascent PoS blockchain that does not yet have sufficiently many operators, long-range attacks are a concern. Aegis overcomes this by deriving its security from Ethereum itself. The checkpointing mechanism on Ethereum prevents an attacker from rolling back the Aegis (eoracle chain): they would need to roll back Ethereum itself to perform a long-range attack on Aegis.

Unbundling Token Utilities

Decentralized oracle networks (DON) are protocols that act as a two-sided marketplace, hence a well-designed token design is a load barring for them. A design that doesn't include a utility token will be much less efficient in aligning the interests of the different kinds of operators and users of the protocol. In addition, an oracle must have decentralized governance.

However, although a token is fundamental for an oracle, the use of a native token for the security of the protocol comes with major risks. A native token suffers from two major disadvantages in relation to ETH when used for security. The first con is its fluctuating value that could cause major shifts in the oracle's Cost of Corruption.

The second weakness is its smaller market cap, resulting in a better ability for an attacker to extract additional gain by short selling, resulting in higher possible Profit from Corruption.

These are very important factors from a cryptoeconomic security point of view.

A dual token approach as suggested by @vbuterin, is to utilize ETH as the lion's share of the security, ensuring high 'Budget

' for attacking the protocol and a 'Cost'

of Attack based on the oracle native token. In addition, the native token will be used for incentivizing positive behaviour, penalizing malicious actors and for decentralized ownership and governance. This way, we can enjoy stability, crypto-economic security, and flexibility of stake with ETH and alignment with the native token.

"Nothing at stake"

To avoid oracles being manipulated and encourage honest behavior among data providers, we need a mechanism to detect and deter participants from submitting false data.

Unlike well-defined protocol rules that can be programmatically verify based on information stored on-chain, real-world data seems to have a more subjective nature. For example, two operators could disagree on a piece of data without any malicious intent. They simply have different perceptions of what is “true”. Of course, resorting to a third-party ‘referee’ to settle such disagreements would contradict the goal of avoiding a central authority governing the protocol. How do we detect false reports in this subjective setting?

We believe the first step towards a solution is to separate the aggregation from the basic task of securely introducing off-chain data into the blockchain. Traditionally, oracle operators fetch, aggregate, and publish data to the DON. We, however, adopt a different approach and divide this complex process into distinct phases.

This approach brings several benefits. Firstly, specifying the task in detail (“submit event ‘E’ from source ‘XYZ’” instead of a general “submit data”) reduces the expected variance in submissions. This, in turn, enhances the robustness of fraud detection and our ability to measure “honesty” among data providers. Additionally, handling raw data, rather than aggregated data, provides more modularity and flexibility in the logic layer, as we’ll discuss next.

Modularity and the problem of ever changing demand

Predicting short and long-term data demand is challenging. This includes determining the types of data needed for DApps, its usage, security requirements, and associated costs. Furthermore, each data field has its unique complexities. For instance, pricing data has distinctive properties and nuances compared to weather or RWA data. A well-designed oracle must support this flexibility and modularity requirement, ensuring that changes or adjustments in the logic layer do not compromise the system’s core security. We insisted on a fully transparent and immutable data process using the eoracle execution layer, with all raw data submissions on-chain. The result is a modular system that can accommodate any computation or data source users may require. The complexities and subtleties of different data types are crucial. With an oracle dedicated to continuous improvement of these solutions, we’ve created a system where flexibility and modularity are inherent. We encourage our users and the community to be meticulous and critical of this process, and to actively participate.

eoracle End Game

The use and demand for off-chain data will change drastically in the next few years. New use cases will arise, different types of data will become relevant on-chain, and the vulnerabilities from this evolution will dictate new and complex security measures. The answer to the question of how some crucial piece of data is being aggregated, secured, priced, and incentivized is not a static one. A brilliant group of people could dedicate their career to analyze data demand, to establish algorithmic solutions for the right aggregation, and they could attract a specialized set of nodes to do the data provisioning. In the current state they will most definitely not be able to cover all the potential market needs. Instead, they will have to put a huge amount of effort in establishing the infrastructure to allow those algorithms.

eoracle has a different plan in mind. We want to leverage the power of permissionless innovation and market dynamics to solve the oracle problem. We envision a perfect market for decentralized data and computation, deeply rooted in the belief that the complex oracle problem cannot be solved by a fixed product or algorithm. Instead, it requires a dynamic free market and a platform designed for permissionless innovation. Leveraging principles of perfect competition, eoracle aims to create an ecosystem where supply and demand seamlessly meet. We are building a decentralized platform that serves as the coordination and settlement layer for innovation around bringing data and compute onto the blockchain.

Our system will trustlessly act as a layer where operators could submit their raw observations on data. Smart contracts will be the algorithmic layer ensuring a fully transparent and immutable process of data aggregation and verification. Developers could request any kind of data with any kind of properties like security mechanisms or guarantees, insurance, latency, diversity, sourcing, and more.

The use of EigenLayer operators as the active participants of the system is a natural choice, allowing a very flexible supply of work and stake - a crucial ingredient for a perfect competition marketplace.

Ending - but essentially just the beginning

Guided by Ethereum’s foundational values and the trust marketplace enabled by EigenLayer, we seek to address the critical challenge of integrating off-chain data and compute without compromising on trust and decentralization. In addition, we envision the value paid by dapps using our oracle solution, will get paid directly to Ethereum validators who are providing the validation, keeping this value within the Ethereum ecosystem instead of going to the pockets of third party intermediaries. With the collective effort of the Ethereum community and the pursuit of solutions that adhere to core cypherpunk values, we hope to help build the next iteration of decentralized oracle networks and enable permissionless innovation through new use cases of decentralized applications, backed by Ethereum security.