PYTH NETWORK: A FIRST-PARTY FINANCIAL ORACLE

PYTH DATA ASSOCIATION

VERSION 1.0

January 4th, 2022

Financial market data is often only accessible to a limited set of institutions and users. Traditional markets typically maintain strict control over live and historical price feeds. Cryptocurrency markets currently have fewer barriers, although there is no guarantee this arrangement will continue. Consequently, only a selected group of users has access to the most timely, accurate, and valuable information.

The Pyth network aims to bring this valuable financial market data to DeFi applications and the general public. The network does so by incentivizing market participants — trading firms, market makers, and exchanges — to share the price data collected as part of their existing operations. The network aggregates this first-party price data and publishes it on-chain for use by either on- or off-chain applications.

The Pyth network has already made substantial progress toward this goal. An initial version of the protocol is running on the Solana network, where it provides sub-second price updates for US equities, foreign currency pairs, commodities, and cryptocurrencies. The network's data providers include many prominent trading firms and exchanges in the traditional finance and cryptocurrency spaces. The network's price feeds already power some of the leading protocols on Solana and will soon be available on other blockchains.

This whitepaper aims to expand on the design of the protocol that powers the Pyth network. The goal of the design is to make the Pyth network self-sustaining and decentralized. The protocol consists of a system of mechanisms and incentives that coordinate the participants in the network.[1] This whitepaper describes the roles of the participants in the network and the mechanisms that coordinate them.

The Pyth Data Association (the "Association"), in collaboration with the community of Pyth network participants, has published this whitepaper to describe a vision for the future of the Pyth network. The Association and network participants will guide initial development of the protocol based on the ideas in this whitepaper, feedback from the broader crypto community, and governance input from PYTH token-holders.

1 OVERVIEW

The Pyth protocol is designed to incentivize participants to continuously publish price updates for various products (such as BTC/USD). Each product has a price feed

[1] In this whitepaper, the term "network" refers to a specific instantiation of the Pyth protocol and its participants. When an action is performed by the "Pyth network," it is in fact performed by the participants interacting with the protocol instance.

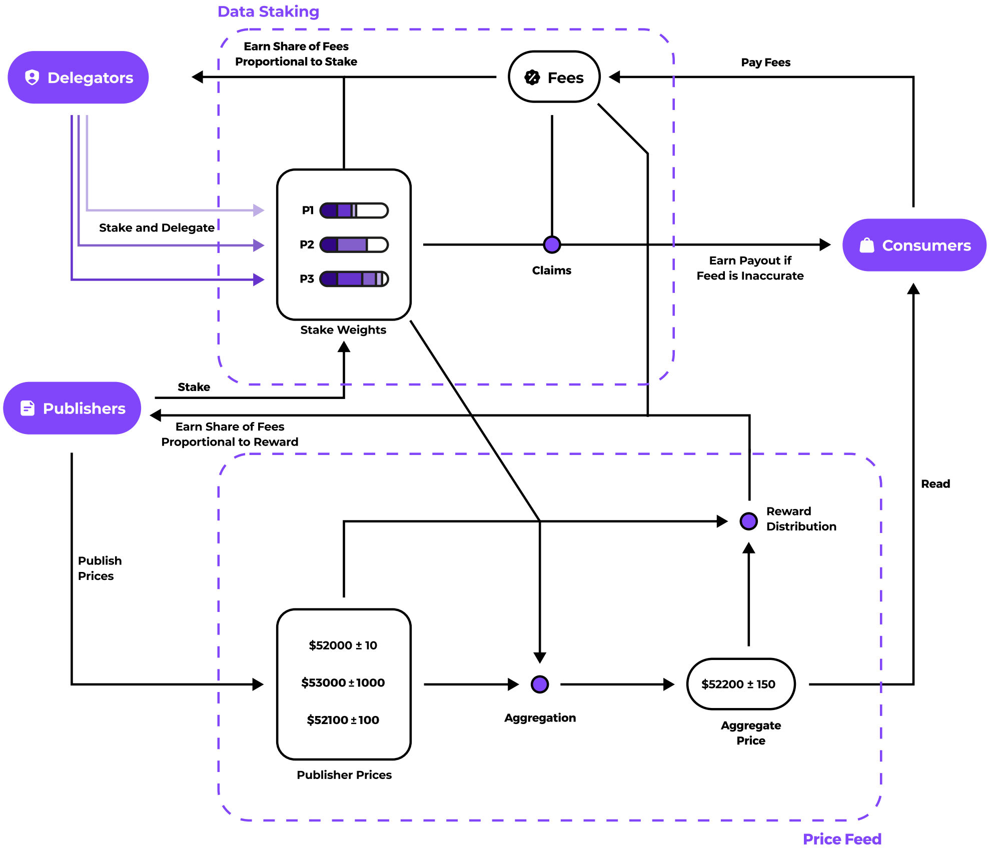

Figure 1: Overview of the Pyth protocol depicting the participants (purple ovals) and their interactions with various mechanisms (purple circles). See text for more details on each mechanism.

that continuously updates with its current price and a confidence interval representing the estimated uncertainty of the price. For example, the current BTC/USD feed may say the price is $65000 ± $50. The feed for each product is published on-chain, where it is read by *consumers*, who may be either blockchain-enabled programs or off-chain applications. An on-chain program produces the price feed for each product by aggregating the price feeds of individual *publishers*. The protocol is designed to attract publishers who are first-party data providers with the ability to source high-quality, timely pricing information.

In addition to publishing these price feeds, the Pyth protocol will allow consumers to optionally pay data fees. In exchange for these fees, consumers may receive a payout from *delegators* if the oracle price is inaccurate. Data fees enable projects that use the protocol's prices to hedge against oracle inaccuracies on behalf of their users. A share of data fees go to publishers, which enables them to monetize their data.

Thus, the protocol will have three different sets of participants:

- **Publishers** publish price feeds and earn a share of data fees in exchange. Publishers are typically market participants with access to accurate and timely price information. The protocol rewards publishers in proportion to the quantity of new pricing information that they share.

- **Consumers** read price feeds, incorporate data into smart contracts or dApps, and optionally pay data fees. Consumers can either be on-chain protocols or off-chain applications.

- **Delegators** stake tokens and earn data fees in exchange for potentially losing their stake if the oracle is inaccurate.

Note that a single actor may have multiple roles in the protocol; for example, publishers may simultaneously be delegators.

These participants will interact via four mechanisms. All of these mechanisms will be implemented on-chain:

- **Price aggregation** combines the price feeds of individual publishers into a single price feed for the product. This mechanism is designed to produce robust price feeds, that is, feeds whose prices cannot be significantly influenced by small groups of publishers.
- **Data staking** allows delegators to stake tokens to earn data fees. The delegators in aggregate also determine the level of influence that each publisher has on the aggregate price. In addition, this mechanism determines whether delegators' stakes are slashed. Finally, the mechanism collects data fees from consumers and distributes a share to delegators. The remainder goes into a reward pool that is distributed to publishers.
- **Reward distribution** determines the share of the reward pool earned by each publisher. This mechanism preferentially rewards publishers with higher-quality price feeds and reduces the likelihood that uninformed publishers will earn rewards.
- **Governance** determines high-level parameters of the other three mechanisms.

A critical challenge is designing these mechanisms to be robust to various forms of adversarial behavior. Three specific attacks to consider are:

1. Participants could onboard as publishers and attempt to manipulate the oracle price. The price aggregation mechanism is designed to guard against this attack by limiting the influence of publishers on the aggregate price.
2. Uninformed participants could onboard as publishers to earn rewards without contributing useful pricing information. The reward distribution mechanism is designed to guard against this attack by reducing the possibility that uninformed participants can earn rewards.
3. Participants could pay data fees and seek to manipulate the claims process to trigger an invalid payout. The mechanism's claims process is designed to make this attack difficult.

The mechanisms introduced above will depend on two core functions of the protocol. First, parts of the Pyth protocol will run in epochs. An epoch is a number of Solana slots corresponding to one week of real-time. Second, the protocol will require users to stake PYTH tokens to participate in some activities. Staking locks the user's tokens immediately and makes them available for downstream activities at the beginning of the next epoch. At any point, stakers can request to unstake their tokens. Upon unstaking, the tokens will remain locked in the contract for the remainder of then-current and subsequent epoch. This staking design guarantees that the quantity of PYTH tokens staked toward any given activity remains constant within an epoch. Additional mechanisms such as stake pools may enable stakers to delegate their staked tokens to another user. However, these mechanisms are not core to the protocol (and could be built as entirely separate programs), so they are not described in this whitepaper.

2 PRICE AGGREGATION

The price aggregation mechanism combines each individual publisher's price and confidence feed into a single aggregate price and confidence feed. For example, one

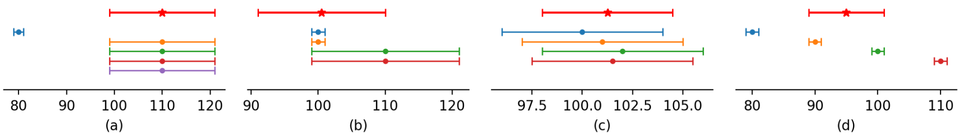

Figure 2: Scenarios for the aggregation procedure. The lower thin bars represent each publisher's prices and confidence intervals, and the bold red bar represents the resulting aggregate price and confidence.

publisher may say that the price of BTC/USD is $\$52000 \pm 10$ and another that it is $\$53000 \pm 20$, and price aggregation may combine these two prices into an aggregate price of $\$52500 \pm 500$. This mechanism is part of the on-chain program and triggers when publishers submit price updates: the first price update on a given slot automatically aggregates the prices from the previous slot.

The price aggregation algorithm is designed to have three properties:

1. It is resistant to manipulation — both accidental and intentional — by publishers. For example, if most publishers submit a price of $100 and one publisher submits a price of $80, the aggregate price remains near $100. This property increases the likelihood that the aggregate price remains accurate even if a small percentage of publishers submit a price far from the market. Figure 2(a) depicts this scenario.

2. The aggregate price appropriately weights data sources with different levels of accuracy. The Pyth protocol allows publishers to submit a confidence interval because they have varying levels of accuracy in observing the price of a product. For example, some publishers are expected to be exchanges. Exchanges have different levels of liquidity, and less liquid exchanges tend to have wider bid/offer spreads than more liquid ones. This property can result in situations where one exchange reports $\$101 \pm 1$, and another reports $\$110 \pm 10$. In these cases, the aggregate price is closer to $101 than $110. Figure 2(b) depicts this scenario.

3. The aggregate confidence interval reflects the variation between publishers' prices. In reality, there is no single price for any given product. Products trade at slightly different prices at various venues at any given time. Furthermore, the product's spread is a fundamental limit on the precision of the product's price. The aggregate confidence interval reflects both the variation across venues and these limitations. Figures 2(c) and (d) depict two different cases where there are price variations across exchanges.

The price aggregation algorithm uses a variant of the weighted median. An input to the algorithm is a stake-weight for each publisher. The data staking mechanism (described below) generates this weight to maximize the robustness of the price feed. This weight is designed to account for hard-to-quantify factors that may influence robustness, such as publisher reputation. The first step of the algorithm computes the aggregate price by giving each publisher three votes – one vote at their price and one vote at each of their price plus and minus their confidence interval – then taking the stake-weighted median of the votes. The second step computes the distance from the aggregate price to the stake-weighted 25th and 75th percentiles of the votes, then selects the larger as the aggregate confidence interval.

This simple algorithm is a generalization of the ordinary median. Most people understand the median as the middle value in the data set, that is, the 50th percentile. However, the median is also the value R that minimizes the objective function $\sum_i |R - p_i|$ where p_i is the price of the ith publisher. This function penalizes R based on its distance from the publisher's price p_i. The proposed algorithm computes the aggregate price R that minimizes $\frac{1}{3}\Sigma_i s_i|R - p_i| + \frac{2}{3}\Sigma_i s_i \max(|R - p_i| - c_i, 0)$, where s_i is the publisher's stake-weight and c_i is the publisher's confidence interval. This objective does two different things. First, it weights publishers according to their stake, such that low-stake publishers have minimal influence on the price. Second, it combines the ordinary median objective with a second term that only assigns a penalty to R if it lies outside the publisher's confidence interval.

3 DATA STAKING

The data staking mechanism collects data fees and distributes payouts if the oracle publishes an incorrect price for a product.[2] The mechanism will also define a claims process that determines when consumers receive a data staking payout; this process compares the oracle price to external reference data to measure its accuracy.

There are several challenges in designing this mechanism. First, data fees should be fairly priced — i.e., the price should reflect the failure risk. Second, the mechanism needs to align the incentives of publishers and delegators, as failures on the part of publishers could lead to losses for delegators. Finally, the claims process needs to be robust to attempts to trigger payouts on false claims.

The data staking mechanism will permit or require users to perform the following actions:

1. Consumers opt to pay data fees for a product. They pay a fee to the Pyth protocol in any governance-approved token, which may include PYTH , USDC, or other tokens. In exchange, the consumer may earn a payout if the chosen product has a problem over the next 4 epochs ($\sim$ 1 month).

2. Publishers will be required to stake a minimum quantity of PYTH tokens per product they price. These PYTH tokens will form a portion of the assets available for payouts but will only be slashed if the publisher publishes an inaccurate price. In other words, the protocol will only punish publishers for failures if the on-chain mechanism determines that they are at fault.

3. Delegators can use their staked PYTH tokens to back products. For each staked PYTH token, delegators select (1) the product to back and (2) a publisher in that product. The publisher selection helps improve the product's security (as described in the text below). Delegators earn a share of the data fees for the products they back; governance will determine the share (initially 80%).

4. Anyone can raise a claim that the oracle published an incorrect price. Once raised, a vote of PYTH token holders ratifies claims. In the event of a successful claim against the product, the protocol will slash the stakes of the product's delegators. The protocol will distribute the slashed amount to consumers in proportion to the dollar-denominated value of the data fees in the epoch during which the event occurred. The process for adjudicating claims is designed to deem a claim successful if Pyth's aggregate price and confidence interval were substantially incorrect for any period of time. The process will further determine which publishers were incorrect during this time and slash their stakes accordingly.

This functionality will be implemented as part of the on-chain program that codifies the rules of the Pyth protocol. For example, a consumer will invoke a function on this program to hedge product failures. This function will accept payment and

[2] Note that this whitepaper does not consider outages, i.e., situations where the oracle fails to publish a price. It may expand in the future to include outages caused by publisher failures and not underlying blockchain failures.

store the consumer's wallet address for future claims payouts. The claims process involves two separate on-chain actions: raising and ratifying a claim. The former of these actions will require specific input data computed off-chain; the Pyth network developers will provide an open-source software package to construct this data. The incentives of the claims process are designed to ensure that these off-chain activities are performed correctly.

This mechanism is intended to produce a market price for data fees. In step (3), delegators will compare products on the anticipated data fees relative to the perceived risk of a successful claim. A favorable tradeoff will attract more delegators to the product, thereby lowering the price. As this process reaches an equilibrium, the relationship between the fees for a product and its total stake should reflect its claim risk.

Delegators have a secondary role in this process, which is to determine the stake-weights of publishers in the aggregation procedure. This process again happens in step (3) when they select which publisher in the product to back. A publisher's stake-weight is the sum of the publisher's stake and their backers' stake, normalized such that the total stake-weight per product is 1. Note, however, that the delegators are insuring the *product* against failures, not the specific publisher — if the other publishers fail, the protocol will still slash the delegator in the resulting claim. Therefore, the protocol incentivizes delegators to distribute their stake amongst publishers to minimize the product's overall failure risk. They will need to balance multiple competing factors when making this decision. On the one hand, reputable publishers with demonstrated historical performance may warrant a more substantial stake. On the other hand, distributing stake amongst publishers ensures that a small number of publisher failures do not cause the product to fail.

Governance may allow delegators to back several products with each staked PYTH token. This change would make data staking more capital-efficient, as it is unlikely that multiple products will have simultaneous successful claims.

3.1 Claims

The claims process will determine whether a payout occurs. The purpose of this process is to verify that the aggregate price and confidence interval for a product were incorrect in comparison to some ground-truth off-chain data. Designing the claims process is difficult because a successful claim may adversely impact PYTH token-holders. Consequently, PYTH token-holders are not impartial judges in this process. On the other hand, an adversary could bribe impartial — that is, completely financially-unmotivated — judges to trigger an incorrect payout.

The proposed claims process reconciles the tension between these two competing concerns. The process will use HUMAN protocol to collect the necessary off-chain information from impartial judges, then feed that information into a predetermined algorithm that determines the outcome of the claim. PYTH token-holders will then vote to ratify the outcome. On ratification, a payout occurs. This mechanism relies on social pressure to incentivize PYTH token-holders to behave in the long-term interest of the protocol. Intuitively, if PYTH token-holders fail to ratify a payout, then the result of the algorithm is strong evidence to consumers that data fees are useless. Consequently, data fees will stop, and the protocol as a whole is likely to fail. However, if an adversary attempts to bribe the judges, PYTH token-holders can vote against ratification and broadcast the evidence of manipulation.

Anyone will be able to file a claim against the protocol to (possibly) trigger a payout. A claim is an assertion that the aggregate price and confidence interval were incorrect for a specific period of time. To prevent spam, the claimant will be required to bond some PYTH tokens; the protocol will return the bond if the claim is ratified. The claimant will also be required to prepay for the HUMAN task; the protocol will reimburse this payment if the claim is ratified.

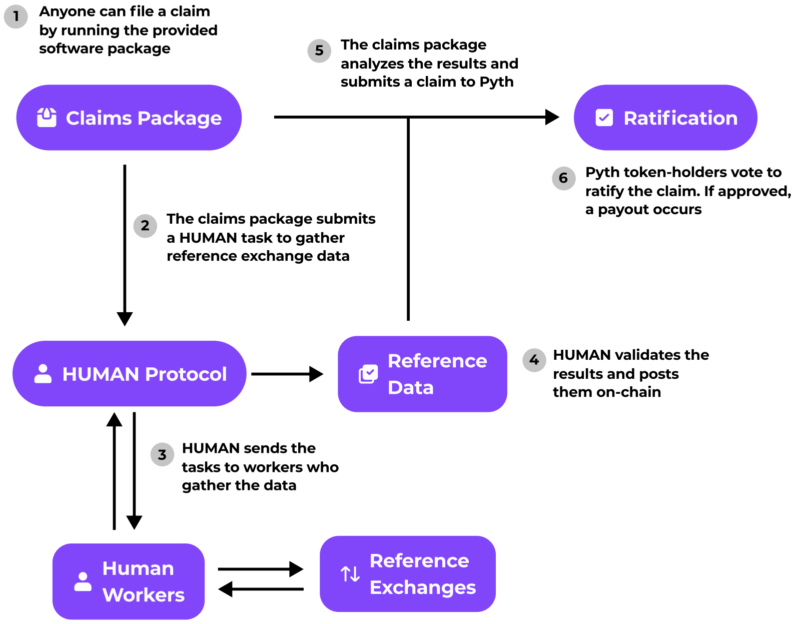
1 Anyone can file a claim by running the provided software package
Claims Package
5 The claims package analyzes the results and submits a claim to Pyth
Ratification
6 Pyth token-holders vote to ratify the claim. If approved, a payout occurs
2 The claims package submits a HUMAN task to gather reference exchange data
HUMAN Protocol
Reference Data
4 HUMAN validates the results and posts them on-chain
3 HUMAN sends the tasks to workers who gather the data
Human Workers
Reference Exchanges

Figure 3: Flowchart of the steps in the claims process.

A claim will include the following fields:

1. The product for which the incorrect data appeared
2. The time interval when the incorrect data appeared. This interval should be relatively short, e.g., 1 second.
3. The results of a HUMAN protocol task that determines the truth of the claim and which publishers were at fault. Pyth network developers will provide a github repository that allows anyone to quickly run the necessary HUMAN task and combine their results in a verifiable way.

The HUMAN task will ask a random sample of workers to report several pieces of off-chain information:

1. The maximum and minimum price for the product during the time interval in question on a fixed set of reference exchanges. These reference exchanges will be selected in advance per-product by Pyth protocol governance.
2. The maximum and minimum aggregate price and maximum confidence interval during the time in question.
3. The maximum and minimum price per publisher and maximum confidence interval during the time in question.

The claim will be successful if (1) the price feed published an aggregate price during the claim interval, and (2) the published price, incorporating any uncertainty provided by the confidence interval, disagrees with the reference prices. The agreement algorithm operates on two price ranges. The Pyth network price range extends from the minimum aggregate price minus 3 confidence intervals to the maximum aggregate price plus 3 confidence intervals. The reference price range extends from the minimum to the maximum reference price. The algorithm will deem the claim successful if these two ranges do not overlap; this check indicates that the reference price was highly improbable according to the Pyth network. If the claim is successful, the algorithm will additionally identify a set of at-fault publishers. The algorithm will judge publishers using the same range overlap criterion, except with their quoted price and confidence instead of the aggregate. This computation results in (1) a yes/no decision on whether the claim is successful and (2), if yes, a list of publishers who are at fault.

The HUMAN task will be configured to increase the difficulty for an adversary to influence the process. Specifically, the task will be instantaneously distributed to judges across the globe and will require the judges to have a certain level of agreement before the answer is accepted. HUMAN protocol will sign and post the results of this task on-chain, where PYTH token-holders can verify them in the ratification step.

This portion of the claims process will be implemented in a software package that anyone can run. The software package will (1) instantiate the necessary HUMAN tasks with the appropriate parameters, (2) gather the results from users of the HUMAN protocol, and (3) run the algorithm described above to determine the success of the claim. Users of the package can directly submit its output for ratification.

Finally, PYTH token-holders will vote to ratify the output of the algorithm. The token-holders should check the on-chain results of the HUMAN task, then run their own instance of the claims software against the results. They should also not be aware of any circumstances indicating that anyone is trying to manipulate the claims process, for example, by broadcasting a bribe to HUMAN judges. (Token-holders are likely to be aware of a bribe offer if one exists, due to the configuration of the HUMAN task.) If the results are genuine, the token-holders should vote to ratify the claim. If ratified, the protocol will slash the stakes of the product's delegators and any at-fault publishers. The protocol will distribute the slashed tokens to the consumers who paid data fees.

From the perspective of the on-chain implementation of the Pyth protocol, the claims process only has two steps. The first step is submitting a claim, and the second step is ratifying it. No off-chain computation or discussion is strictly required to run either step, but the incentives encourage these off-chain activities. For example, a user could elect to file a claim without running the HUMAN task. However, PYTH token-holders are unlikely to ratify this claim. Similarly, PYTH token-holders could vote to reject every claim. However, they are strongly incentivized to accept claims when the results of the HUMAN task indicate that the claim should be successful.

4 REWARD DISTRIBUTION

The reward distribution mechanism will determine the share of the reward pool earned by each publisher. As a reminder, the data staking mechanism will distribute a portion of the data fees per product into a reward pool for that product. This portion initially will be 20% but may be adjusted by governance. The reward pool may additionally include bonus rewards to bootstrap the protocol (discussed in further detail below).

The reward distribution mechanism is designed to achieve the following goals:

1. **Preferentially reward higher-quality publishers**. Publishers are not uniform, and some have access to more accurate or timely pricing information than others. The reward system should preferentially reward these publishers so that the best publishers are incentivized to contribute to the protocol.

2. **Prevent bad actors from earning rewards**. Publishers may attempt to exploit the system for personal gain. The reward mechanism must penalize bad actors to disincentivize them from participating in the protocol.

3. **Encourage honest reporting of private information**. Publishers typically have private pricing information, such as their recent trades on various exchanges. The incentives should encourage publishers to honestly report prices using

this private information, as the combination of publishers' honest reports will result in the most accurate aggregate price.

Existing oracle mechanisms fail to accomplish these three goals. Other oracle mechanisms reward publishers for agreeing, i.e., reporting the same price. However, rewarding agreement creates a perverse incentive for publishers to misreport their private price information. Imagine a publisher thinks the price is currently $110 but notices that the aggregate price on the previous slot was $100. The publisher knows that the price is unlikely to move by $10 in a single slot, so they can infer that the aggregate price on this slot is also likely to be around $100. The publisher may choose to maximize agreement by reporting a price close to $100 instead of their actual estimate of $110. Formalizing this argument shows that rewarding agreement incentivizes publishers to report *their best estimate of what everyone else will report* and not their private price.

The second problem is that agreement cannot separate bad actors from good actors. Prices are often stable for periods of time, so the aggregate price from the previous Solana slot is a reasonable estimate of the current price. Therefore, if the protocol rewards agreement, bad actors can trivially earn rewards by replaying aggregate prices with a delay. The protocol also should not simply penalize disagreement because honest publishers will occasionally diverge from the actual price.

These problems occur because existing oracle mechanisms are designed for the case where every publisher has access to the same information. In these cases, agreement suffices to validate that the publishers have reported it correctly, as there is no reason to expect honest publishers to disagree. In Pyth's case, publishers have private information, and are not expected to report the exact same price. There are real-world circumstances in which the prices on exchanges disagree and the oracle should reflect those circumstances.

This whitepaper proposes a new oracle mechanism for the private information setting based on a critical insight: publishers should be rewarded for sharing new information, i.e., changes to the current price. The mechanism measures new information by calculating how well a price series predicts future changes in the aggregate price. This mechanism is also difficult for bad actors to exploit under the assumption that they cannot easily predict future prices from historical prices.

4.1 Score Computation

The reward distribution mechanism will divide the reward pool for a product amongst publishers in proportion to three quantities:

1. The publisher's stake-weight s, as determined by the data staking mechanism. This number is between 0 and 1.

2. A quality score q measuring how well the publisher's price series predicts future price changes. This score lies between -1 and 1, and a positive score implies some predictive power.

3. The calibration c of the publisher's confidence intervals. This quantity is a number between 0 and 1.

The mechanism will distribute rewards at the end of each epoch. The on-chain program will estimate q and c for each publisher during the epoch. At the end of the epoch, each publisher earns a share of the reward pool in proportion to $s \times q \times c$. If this quantity is negative, the mechanism will slash the publisher's stake by the corresponding amount.

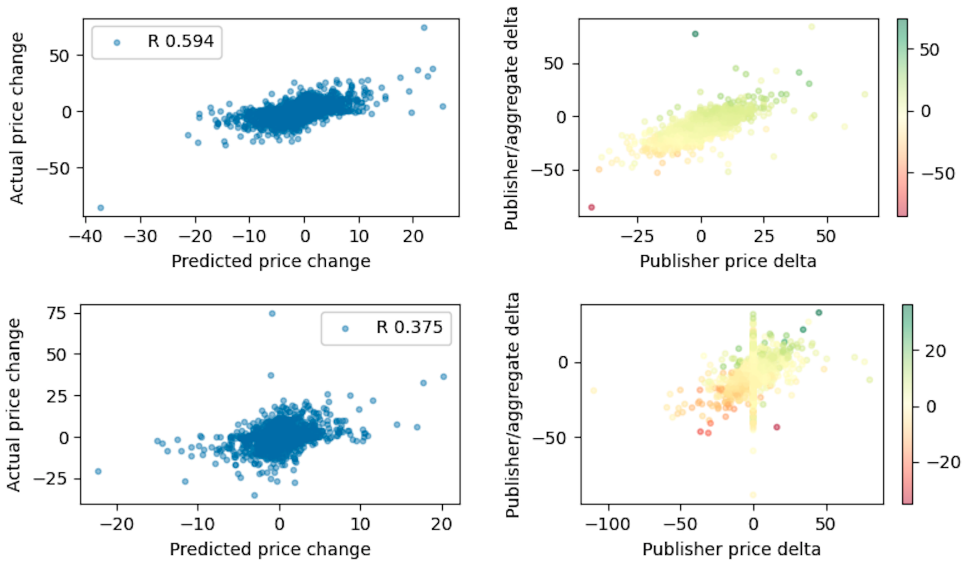

Figure 4: Visualization of the quality score for two publishers in a historical data sample. The left graph is a scatterplot of the predicted and actual price changes. The tighter the linear relationship between these quantities, the higher the quality score. The axes of the right graph correspond to the two non-bias features of the regression model and the color denotes the actual price change. A smooth color gradient indicates a high quality score. The top publisher obtains a higher score than the bottom one in this sample.

4.1.1 Quality Score

The quality score measures how well a publisher's price series predicts future changes in the aggregate price. The mechanism will compute this score by training an online regression model that predicts the future price from several features of the publisher's price series. Let p_t, σ_t be the publisher's price and confidence on slot t, and let $\bar{p}_t$ be the aggregate price on that slot. That is, the aggregation algorithm computes $\bar{p}_t$ from all of the publishers' prices p_t. On slot t, the regression model performs the following set of updates:

$$\begin{aligned} f_t &\leftarrow [p_t - \bar{p}_{t-1}, p_t - p_{t-1}, 1] \\ \hat{p}_t &\leftarrow w_{t-1}^T f_t + \bar{p}_{t-1} \\ w_t &\leftarrow \text{clip}(\frac{\alpha}{\sigma_t}(\hat{p}_t - \bar{p}_t) f_t, -0.1, 0.1) \end{aligned}$$

In the above equations, $\hat{p}_t$ represents a prediction for the aggregate price on the current timestep. A linear regression model computes this prediction from two change-in-price features that compare the publisher's current price p_t to their own price and the aggregate from the previous slot. w_t represents the regression model's weights. After each prediction, the final equation updates these weights using a standard clipped gradient update.

The quality score q is the product of two terms. The first term is the correlation between the predicted price $\hat{p}_t$ and the aggregate price $\bar{p}_t$ over the entire epoch, and the second term is the fraction of slots in which the publisher contributed a price. The correlation measures how well the predicted prices agree with the aggregate. If the predictions are perfect, the correlation is 1. If the predictions are random, the correlation is 0. If (somehow) the predictions are worse than random, the correlation is negative. Honest publishers are unlikely to encounter this third case, but it guards against adversarial behavior (see Discussion, below).

The quality score proposed here permits multiple variations. First, it could predict price changes more than 1 slot ahead in the future, which are presumably harder to predict. Second, it could incorporate additional features. For example, some exchanges often have minor but persistent price differences. This mechanism could avoid punishing such price differences if it included a feature that tracks them. The Pyth network developers have experimented with this mechanism on historical data, and those experiments suggest that the version presented here produces reasonable results. However, the general approach is flexible and can be tuned in the future.

4.1.2 *Calibration Score*

The calibration score c measures whether the publisher's confidence interval accurately represents their uncertainty. The score interprets the confidence interval as the standard error of a Laplace distribution within which the publisher expects to find the aggregate price. (The Laplace distribution is a heavy-tailed distribution that better represents the actual distribution of prices relative to the normal distribution.) The score uses a simple frequency test to measure how closely the aggregate price follows the implied distribution. Specifically, it computes a z-score by taking the difference between the aggregate and prediction, then normalizing by the publisher's confidence interval. The resulting z-score is then binned to produce a histogram; the z-score thresholds for each bin are chosen such that each bin has equal probability under the standard Laplace distribution. This procedure applied to a perfect publisher should produce a uniform histogram. The calibration score is therefore defined as one minus the earth-mover distance between the publisher's histogram and uniform.

Note that the calibration score does not benefit publishers for producing tighter confidence intervals. The quality score already solves this problem: publishers with tighter confidence intervals should also have more accurate price predictions. Instead, the calibration score c captures whether the reported confidence interval corresponds to the publisher's "true" confidence.

4.2 Discussion

This reward distribution mechanism has many beneficial properties that align with the goals above:

1. The mechanism assigns higher rewards to publishers with more predictive price series and better-calibrated confidence intervals. These publishers are also likely to attract a higher percentage of the stake-weight. Consequently, these publishers will earn a larger share of the reward pool. The mechanism thereby incentivizes the best publishers to participate in the protocol.

2. The properties of the correlation coefficient reduce the likelihood that bad actors will earn rewards in the system. This analysis requires two assumptions: (1) future prices are hard to predict from historical prices, and (2) the Pyth aggregate price tracks the actual price. The first assumption is the efficient market hypothesis, which is broadly true on the small timescales required. The second assumption should be valid because the aggregation algorithm is robust; thus, an attacker has limited ability to manipulate the Pyth aggregate price away from the actual price. Under these two assumptions, an adversary with access to only historical Pyth data cannot earn rewards as a Pyth publisher: their predictions $\hat{p}_t$ will be random, so their correlation q will also be a random variable with 0 expectation. The reward/penalty for each publisher is proportional to q, so their expected reward in each epoch is also 0. Furthermore, the reward mechanism penalizes publishers with negative quality scores by slashing their stake, such that the expected reward over multiple

epochs is also zero. The penalty is indispensable: otherwise, bad actors could play the game over multiple epochs, and their quality score would be positive through sheer chance in some epochs. The penalty is designed so that losses in other epochs balance out rewards from these epochs.

3. The adaptive nature of the predictions eliminates the primary incentives for dishonesty. Publishers no longer have to agree with the aggregate as long as the prediction computed by the regression agrees with the aggregate.

This reward mechanism does have some minor weaknesses. One weakness is that an attacker could copy a publisher's price within the same slot. This attack would require the attacker to read a publisher's price update and submit their own price update within the Solana slot time. The protocol could defend against this attack using a commit-reveal system for publisher price updates. A second weakness is that the mechanism encourages publishers to submit predictions of future prices, which may not agree with the actual price. This attack seems unlikely: if a publisher could predict future prices, it would be more profitable to trade on their predictions instead of publishing them. It also exposes the publisher to getting slashed in a data staking claim.

5 GOVERNANCE

The Pyth Data Association will initially govern the protocol while it is under development. Over time, the Association will transfer full control of the protocol to an on-chain governance mechanism. Once the transfer occurs, all protocol governance will be on-chain. However, on-chain governance systems have various failings — e.g., borrowing coins to vote — that do not have effective technical solutions. Therefore, the general design philosophy is to reduce the necessity for governance input.

The on-chain governance mechanism will approve or reject proposals using a coin-voting system. Anyone with a minimum quantity of staked PYTH tokens will be able to make a governance proposal. PYTH stakers will then be allowed to vote on these proposals. They will also be able to delegate their votes to others. Governance votes will run for 2 weeks from the proposal date, and only tokens staked in prior epochs will be allowed to vote. Users will be permitted to vote with PYTH tokens that are staked toward other applications (e.g., data staking). Additionally, some users currently own locked PYTH tokens, which will also be permitted to vote.

On-chain governance is expected to be responsible for taking the following actions:

- Approving the types of tokens that may be used for data fees.
- Determining which products are listed on Pyth and their reference data (e.g., number of decimal places in the price, reference exchanges).
- Determining the share of data fees allocated to publishers, delegators, and other uses.
- Approving software updates to the on-chain program.
- Determining the number of PYTH tokens that publishers must stake.
- Determining the number of products that a delegator can back per staked PYTH token.
- Enabling claims to be filed against a product. Governance should take this action once the product has enough publishers to produce a robust price feed.
- Permissioning publishers to provide price feeds.

6 INCENTIVES

This section summarizes the incentives for the various participants in the proposed protocol.

Publishers are incentivized to publish accurate and timely prices by the data staking and reward distribution mechanisms. Participants must stake PYTH tokens to become publishers. The data staking mechanism can slash their stake if (1) they ever publish a price far from the reference price and (2) that price subsequently causes a problem with the oracle's price feed. This possible penalty encourages publishers to publish prices close to the overall market. The reward distribution mechanism also preferentially rewards publishers whose prices predict future changes in the aggregate price.

Publishers are incentivized to participate in the protocol to earn a share of the rewards. Publishers earn a share of the data fees for the products they price. The data fees for a product will likely grow in proportion to consumer usage of the price feed, while the capital required (the staked PYTH tokens) and publishing cost remain fixed. Popular products with few publishers could produce an attractive level of payments.

The primary attacks that publishers could perform are (1) attempting to manipulate the aggregate price and (2) attempting to earn rewards without contributing new pricing information. Delegators will guard against the first attack by setting each publisher's stake weights to limit their influence on the aggregate price. The protocol incentivizes the delegators to set these weights such that no single publisher or small group can manipulate the price. The rewards mechanism will guard against the second attack by only rewarding publishers whose price series predict future price changes.

Consumers are incentivized to pay data fees for two reasons. First, data fees enable applications to reduce the risk of using Pyth price feeds. Users of applications are highly risk-sensitive and will make usage decisions accordingly. Second, paying data fees attracts more publishers to the product, which improves the robustness of the price feed.

It is possible that consumers consider paying data fees and then attack the system to trigger an invalid payout. The claims process is carefully designed to reduce the likelihood of this outcome.

Delegators are incentivized to participate in the protocol to earn data fees. Delegators will initially earn attractive payments, but competition between them will reduce the payments over time as the market becomes more efficient. In the long run, the payments earned by delegators will reflect the activity's inherent risks and the broader investment climate.

Delegators also have a secondary role, which is to set the stake-weights of publishers. The protocol incentivizes delegators to set these weights in a way that maximizes the robustness of the price feed because delegators risk losing their stake if the aggregate price feed is incorrect.

7 TOKEN DISTRIBUTION

There are 10,000,000,000 PYTH tokens and this total supply will not increase. Furthermore, 85% of the tokens will initially be contractually locked. These tokens will unlock monthly over 7 years with an initial 1-year cliff. This schedule is designed to produce a gradual increase the unlocked token supply over time. The remaining 15% of PYTH tokens will initially be unlocked. The supply of both locked and unlocked tokens will be allocated according to the categories shown in Table 1.

	Unlocked	Locked	Total
On-chain Rewards	8%	14%	22%
Ecosystem Participation	5%	28%	33%
Team and Advisors		25%	25%
Launch Partners	2%	8%	10%
Private Sale		10%	10%

Table 1: Allocation of locked and unlocked PYTH tokens. Locked tokens unlock monthly over 7 years with an initial 1-year cliff.

7.1 Bootstrapping rewards

The Pyth protocol will likely provide additional incentives to early participants in the protocol. Specifically, the protocol faces a cold-start problem for new products. These products have neither publishers nor consumers to pay data fees. However, publishers have no incentive to price a new product without data fees. One way for the protocol to address this problem is to incentivize early publishers for new products.

Various incentive systems could be applied to solve the cold-start problem. For example, the protocol could add bonus tokens to the reward pools for new products. The protocol could initially take these tokens from a preallocated reward pool or perhaps a portion of the data fees for mature products. The protocol could distribute these bonus tokens in various ways, for example, to reflect the relative difficulty of sourcing different types of data. The Association will reserve a substantial portion (22%) of the total PYTH token supply for distribution in such incentive systems.

New products will also go through two stages. When the product is first added, the protocol will not permit claims against the product. When a product only has a small number of publishers, the price feed is not sufficiently robust, as each publisher has substantial influence over the aggregate price. Once the product has a sufficient number of publishers, governance will vote to enable claims. However, the protocol will allow consumers to pay data fees on products whose claims are currently disabled as a signaling mechanism. For example, a protocol that wanted their governance token listed could use this mechanism to provide an added incentive for publishers.

8 CONCLUSION

This whitepaper proposes an oracle protocol to make accurate, high-resolution financial market data easily accessible on-chain. The protocol is designed to be a self-sustaining decentralized network that coordinates data publishers and consumers. A critical element of the design is a data staking mechanism for consumers that distributes a share of data fees to publishers. The protocol is also designed to attract publishers with high-quality pricing data. The mechanisms help prevent malicious attackers from manipulating the protocol to their benefit in various ways, such as manipulating the price.