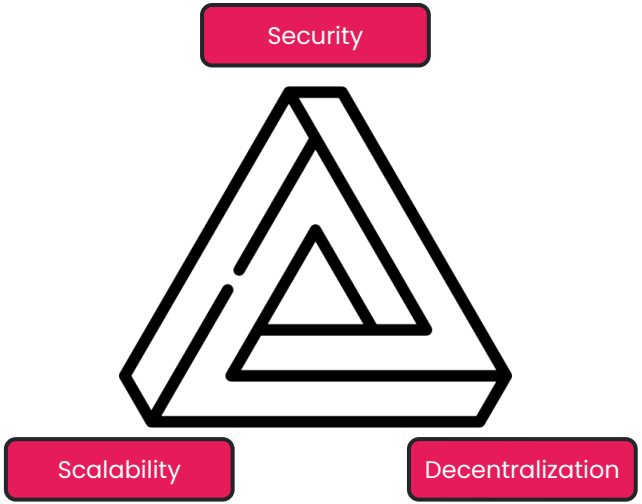
flare

Security in Decentralized Oracles: Challenges and Solutions

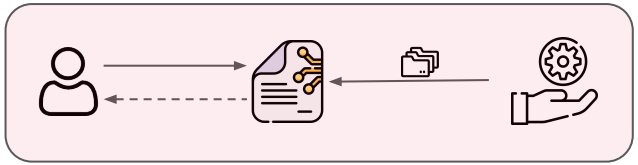
The Problem

- How do you bring external data to a blockchain environment?
- Key desirable properties:
  - Accuracy
  - Availability
  - Resistance to Manipulation

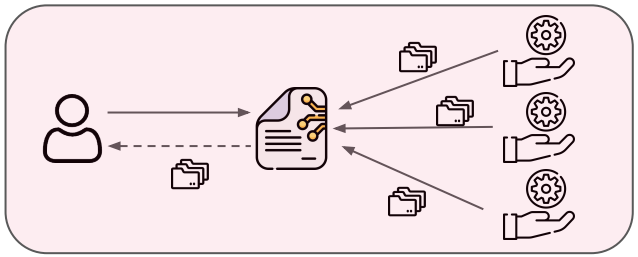


Partial Solutions

- **The “oldest” solution: a single data provider**
  - Low latency
  - Very simple logic
  - Cheap (low fees)
  - Major con: single point of failure



- **Improved solution: aggregate data from multiple providers**
  - Better **security**
  - Reliable data
  - Requires incentivization mechanism
  - Higher latency
  - More complex on-chain operations



Oracle Security

- **Academic literature usually focuses on:**
  - Safety: protocol reaches correct outcome.
  - Liveness: protocol eventually makes progress.

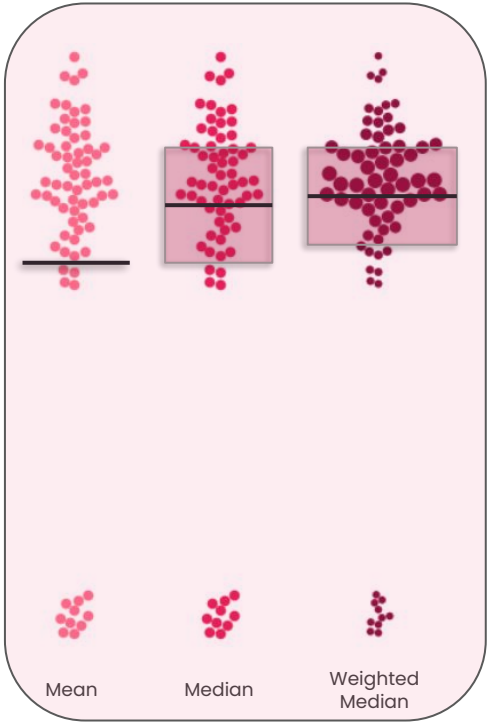
We will focus on **time-series** Oracles (*e.g.* price feeds) secured by *N* data providers, with a data aggregation process using a *weight system*.

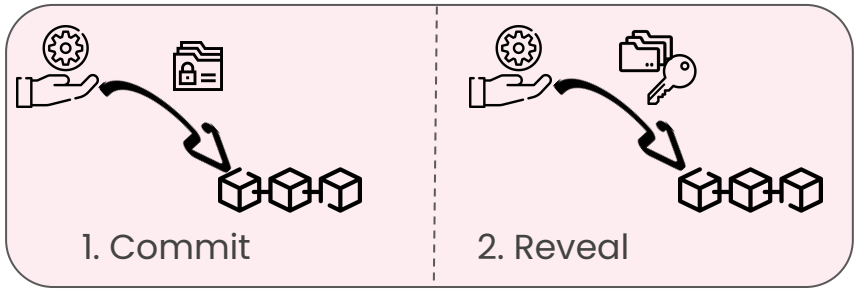
Definition 1. A decentralized oracle is safe with $\omega \in (0, 1]$ if a weight ω is required to publish arbitrary values.

Definition 2. A decentralized oracle is live with $\omega \in (0, 1]$ if a weight ω is required to stall the protocol indefinitely.

Robust Oracle Design

- Aggregation needs to be robust to outliers.
  - *e.g.* mean is not secure – arbitrarily small weight can completely change outcome.
- **Challenge:**
  - Scalability – no complex operations on-chain.
- **Solution:**
  - Offload computations!



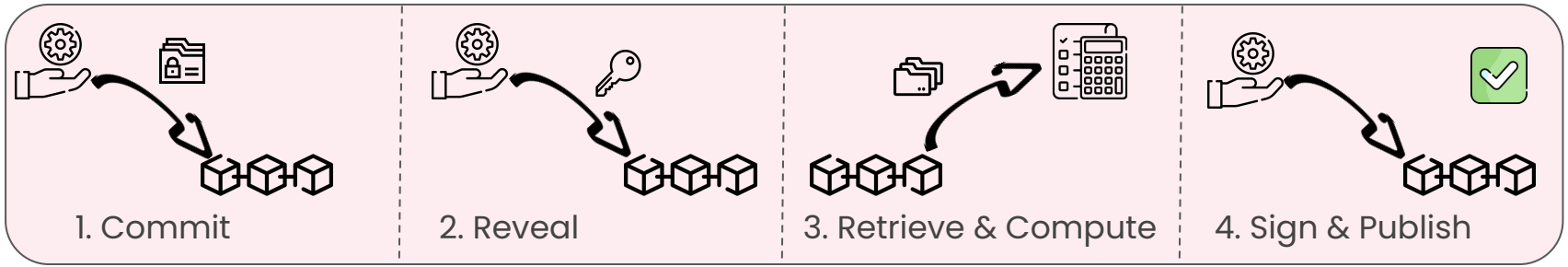


Robust Oracle Design

- Aggregation needs to be robust to outliers.
  - *e.g.* mean is not secure – arbitrarily small weight can completely change outcome.

- **Challenge:**
  - Scalability – no complex operations on-chain.

- **Solution:**
  - Offload computations!



Enshrining

- So far: robust aggregation + off-chain compute + on-chain verification

Theorem 1. The weight threshold for a signing protocol which minimizes any type of adversarial influence is 50%.

Proof. Let T_0 *be the signing threshold (and hence the safety threshold). If* $T_0 > 0.5$*, then the liveness threshold is* $(1-T_0) < 0.5$*. As such, the value of* T_0 *that maximizes* $\min(T_0, 1-T_0)$ *is 0.5.*

- Challenges:
  - Sybil resistance
  - Incentivization (honest participation)
  - Weight/reputation systems can be vulnerable to exploitation

Solution: **Enshrining!**

The Latency Problem

Q: How do we improve latency while maintaining high degrees of security, decentralization, and scalability?

A: Randomness!

The Latency Problem

Q: How do we improve latency while maintaining high degrees of security, decentralization, and scalability?
A: **Randomness!**

The FTSO protocol uses *cryptographic sortition* for selecting data-providers eligible for updating price feeds *every block*.

Block-latency Updates

- To avoid arbitrarily large price changes, price feeds are updated as:

$$P(t+1) = (1+p)^{\delta} P(t) \ ,$$

  where p is the "precision" and the delta-updates are: $\delta(t) \in \{-1, 0, 1\}$.

- The protocol checks all points considered so far:
  - Scalability
  - Decentralization
  - Low-latency

- What makes it difficult to manipulate is the added randomness!

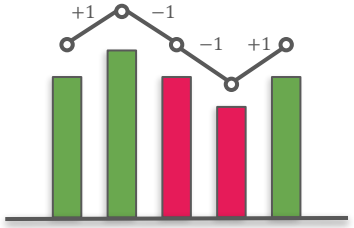
Did We Sacrifice Security for Speed?

Lemma 1. (Liveness) The weight needed to stop progress in the protocol is at most 50%.

Sketch of Proof. Roughly speaking, any "positive" update can be immediately negated by an adversary, and vice versa. Let's call this the "opposite" strategy.

$$P(t+2) = (1+p)^{-\delta} P(t+1) = (1+p)^{-\delta}(1+p)^{\delta} P(t) \ .$$

Thus, the price remains the same in expectation.



Did We Sacrifice Security for Speed?

- The price feed can be described as a 1d **random walk** where each state has an **intrinsic value** – the price. For the "opposite strategy":

$$\mathbb{E}\left[P(t+1)\right] = (1-\omega_m)(1+p)^{\delta}P(t) + \omega_m(1+p)^{-\delta}P(t) \equiv F(\delta,\omega_m)\times P(t)\ .$$

$$\boxed{\mathbb{E}\left[P(t+N)\right] = F(\delta,\omega_m)^{N}\times P(t)\ .}$$

Theorem 2. The expected deviation from the optimal protocol behaviour caused by a malicious weight ω in N blocks, in a steady uptrend, is ***at most*** *2ωpN.*

- With *p = 1/2^13,* deviations of the order of *1%* require extended time frames to develop, even in cases of large malicious weight.

Did We Sacrifice Security for Speed?

- On Flare:
  - 75% of the FLR circulating supply is used in the FTSO weight system.
  - Expected number of updates per block is set to 1.
  - A new block is produced on average every 1.8s.

N \ ω	1% (i.e. $5.6m)	10% (i.e. $56m)	30% (i.e. $167m)
5 blocks	$0.001\% \pm 0.00\%$	$0.012\% \pm 0.01\%$	$0.037\% \pm 0.02\%$
10 blocks	$0.002\% \pm 0.01\%$	$0.024\% \pm 0.02\%$	$0.073\% \pm 0.03\%$
50 blocks	$0.012\% \pm 0.02\%$	$0.122\% \pm 0.05\%$	$0.366\% \pm 0.08\%$

Block-latency Feeds

DOGE/USD

Block-latency Feeds

DOGE/USD

A 0.4% deviation in CeX prices within 3-4 blocks!

Mean dev. (%)	Median dev. (%)	Largest dev. (%)
0.0412	0.0295	0.3960

Key takeaways

- Security must be the primary focus, but it should be addressed alongside scalability and decentralization.
- Enshrining enhances not only security but also performance.
- Keep the chain light: perform heavy computations off-chain and verify on-chain.
- Randomness disrupts attackers' ability do devise effective strategies.

flare

Thank you!