Curated Proof Markets & Token-Curated Identities in Ocean Protocol

Trent McConaghy
@trentmc0

ocean

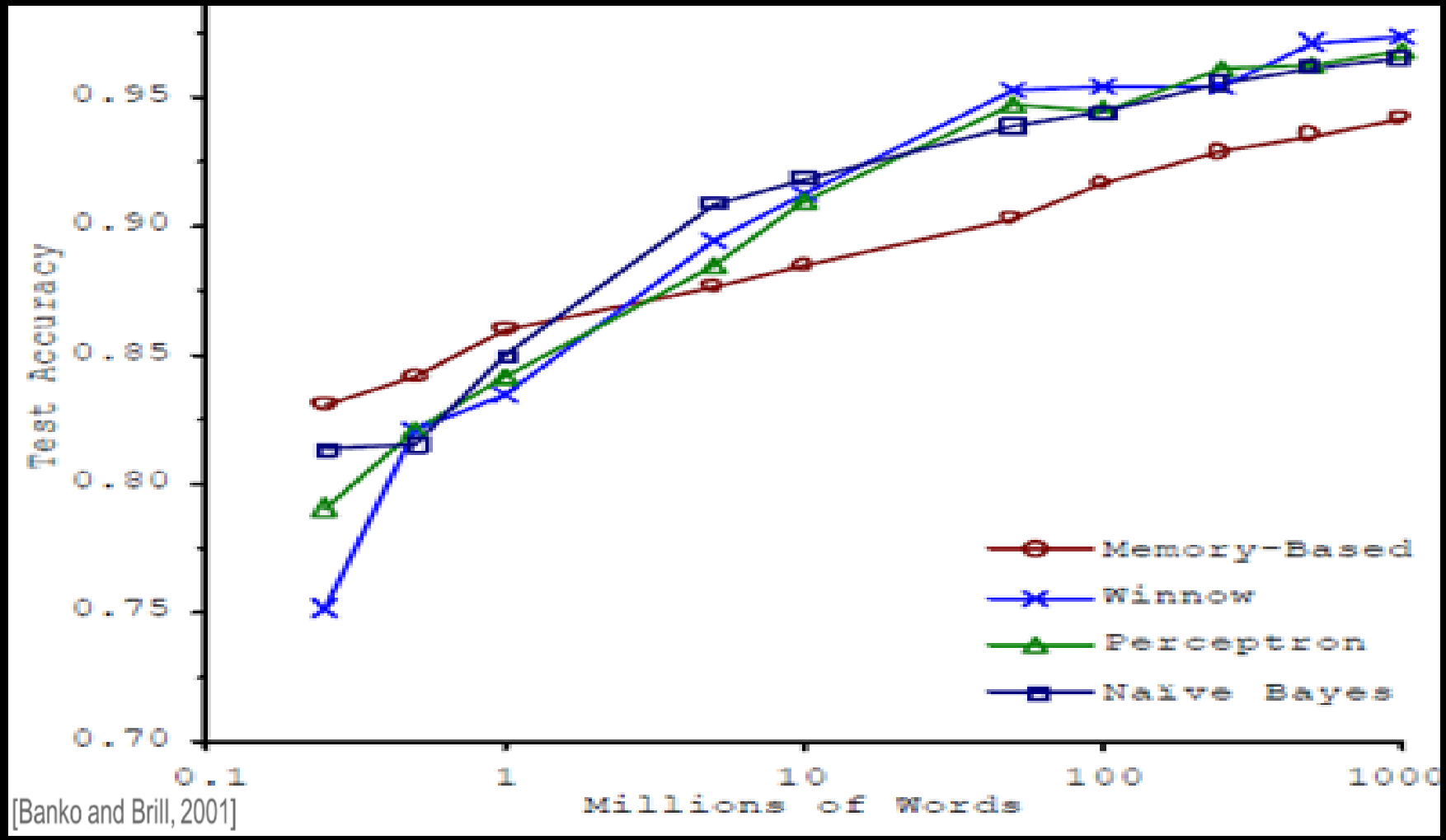
#Data
#Incentives

ocean

Audio radar

ocean

The Unreasonable Effectiveness of Data



1000% less error!

1000x *more* data

Silo mo' data

↓

Mo' accuracy

↓

Mo' $

Default incentive: hoard the data

ocean

“Show me the incentive
and I will show you the outcome.”

-Charlie Munger

You can get people to do stuff

by rewarding them with tokens.

This is a superpower.

Change the incentives!

~~Silo~~ *Pool* mo’ data

⬇

Mo’ accuracy

⬇

Mo’ $

ocean

How to design?
Here's a Process for Token Engineering

1. **Formulate the problem.** Objectives, constraints, design space.
2. **Try an existing pattern.** If needed, try different formulations or solvers.
3. **Design new pattern?**

Patterns (Building Blocks) for Token Design

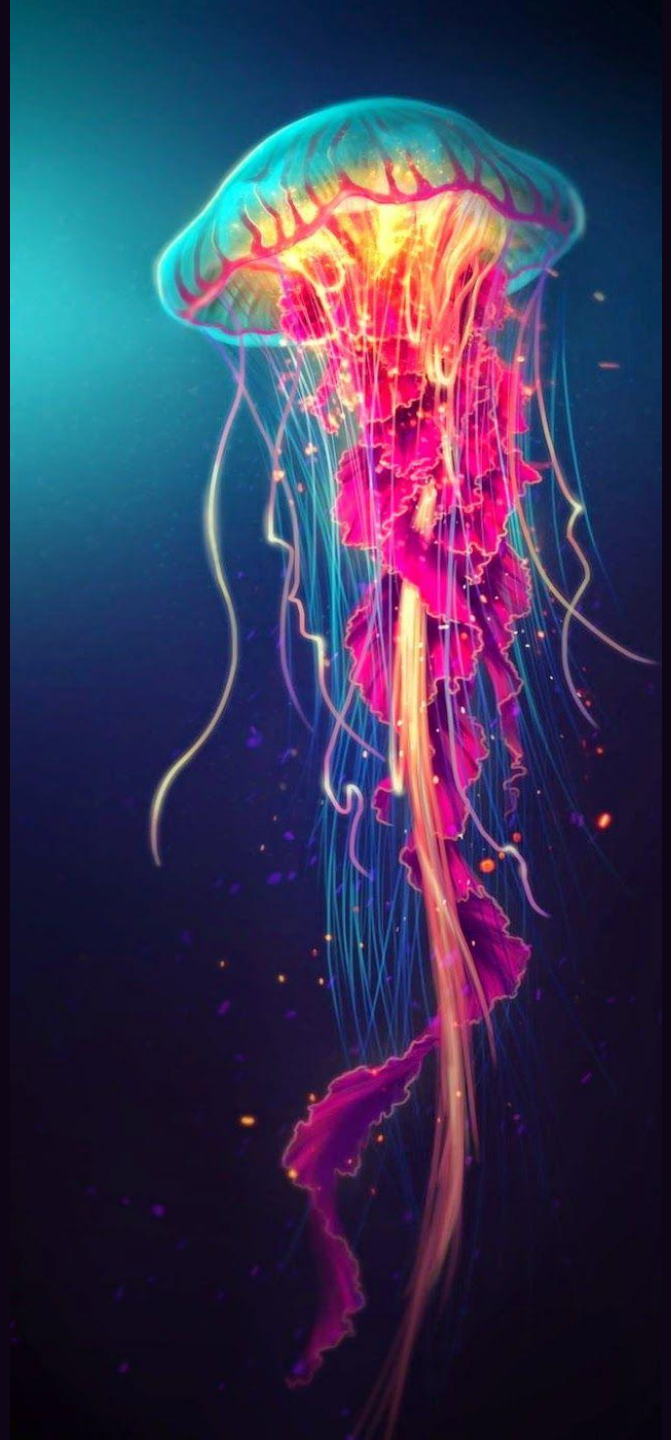
- Curation
  - Stake machines, curation markets
  - Curated proofs markets, TCRs
- Proofs
  - Bitcoin puzzle solving, Filecoin PoST, Steemit human
  - Data availability, more
- Identity
- Reputation
- Governance / software updates
- Third-party arbitration
- …

ocean

Patterns (Building Blocks) for Token Design

- Curation
  - Stake machines, curation markets
  - **This talk: Curated proofs markets, TCRs**
- Proofs
  - Bitcoin puzzle solving, Filecoin PoST, Steemit human
  - **This talk: Data availability, more**
- Identity
- Reputation
- Governance / software updates
- Third-party arbitration
- …

ocean



Curated Proofs Markets: Formulation

ocean

Objective: maximize supply of relevant data

- We convert this objective to block rewards.
- Then, how about this reward:

"Reward making relevant data available when asked"

- But: How to know what's relevant? Algorithms capture this poorly.
- Solution: leave it to the crowd. Let them put their money where their mouth is! (Staking)

- Revised reward:

"Reward curating data (staking on it) + making it available"

ocean

ocean

Objective: maximize supply of relevant data

ocean

- **Reward curating data (staking on it) + making it available**
- **New pattern: curated proofs market**

$$E(R_{ij}) \alpha \ \log10(S_{ij}) * \log10(D_j) * T * R_i$$

Expected reward for user *i* on dataset *j*

S_{ij} = predicted popularity = user's curation market stake in dataset *j*

D_j = proofed popularity = # times made dataset available

 # tokens during interval

ocean

From AI data to AI *services*

ocean

Motivations:

- **Privacy, so compute on-premise or decentralized**
- **Data is heavy, so compute on-premise**
- **Link in emerging decentralized AI compute**

Objective function: Maximize supply of relevant *services*

=reward curating *services* + proving that it was delivered

$$E(R_{ij}) \alpha \log10(S_{ij}) * \log10(D_j) * T * R_i$$

predicted popularity of *service*

proofed popularity of *service*

ocean

ocean

$$E(R_{ij}) \alpha \log10(S_{ij}) * \log10(D_j) * T * R_i$$

S_{ij} = expected popularity = stake in dataset j

Key block #1:

Curation market

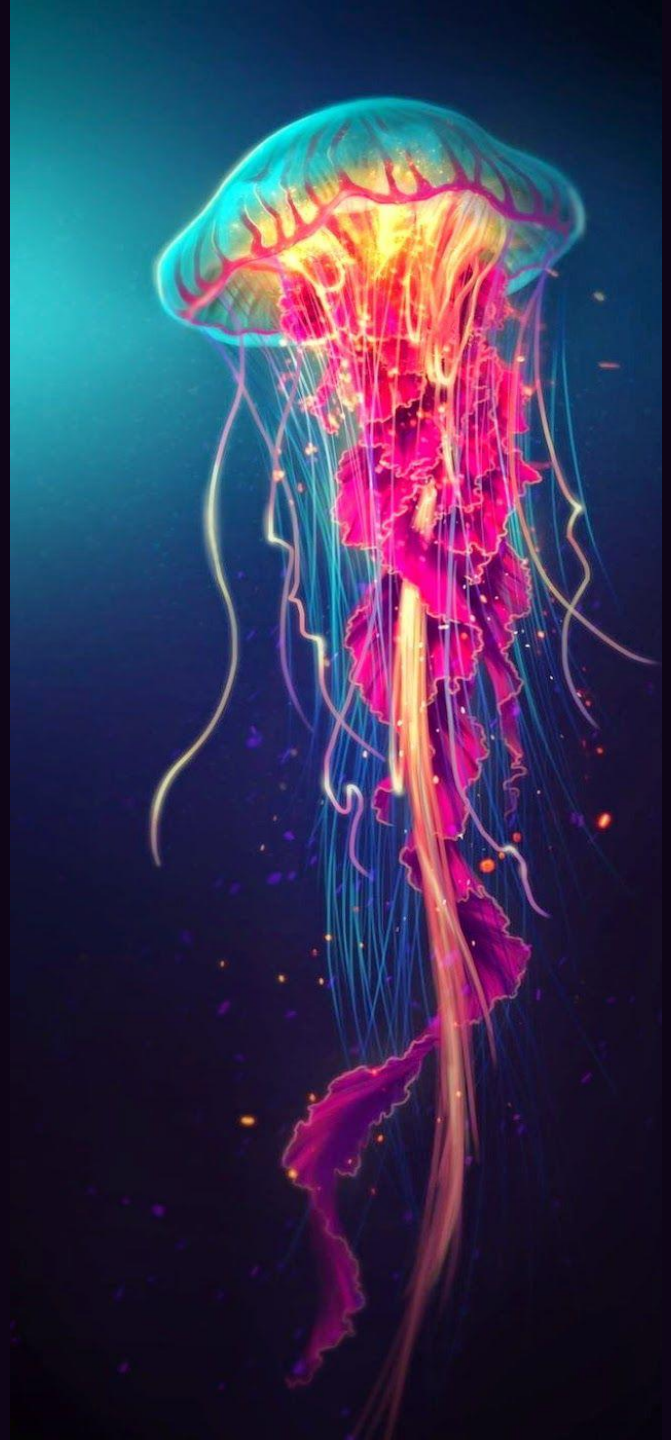
D_j = actual popularity = # downloads

Key block #2:

Proof of availability

(All other blocks are ancillary to these blocks)

ocean



Proofs

ocean

ocean

$$E(R_{ij}) \alpha \log10(S_{ij}) * \log10(D_j) * T * R_i$$

S_{ij} = predicted popularity = stake in dataset j

Key block #1:

Curation market

D_j = actual popularity = # downloads

Key block #2:

Proof of availability

(All other blocks are ancillary to these blocks)

ocean

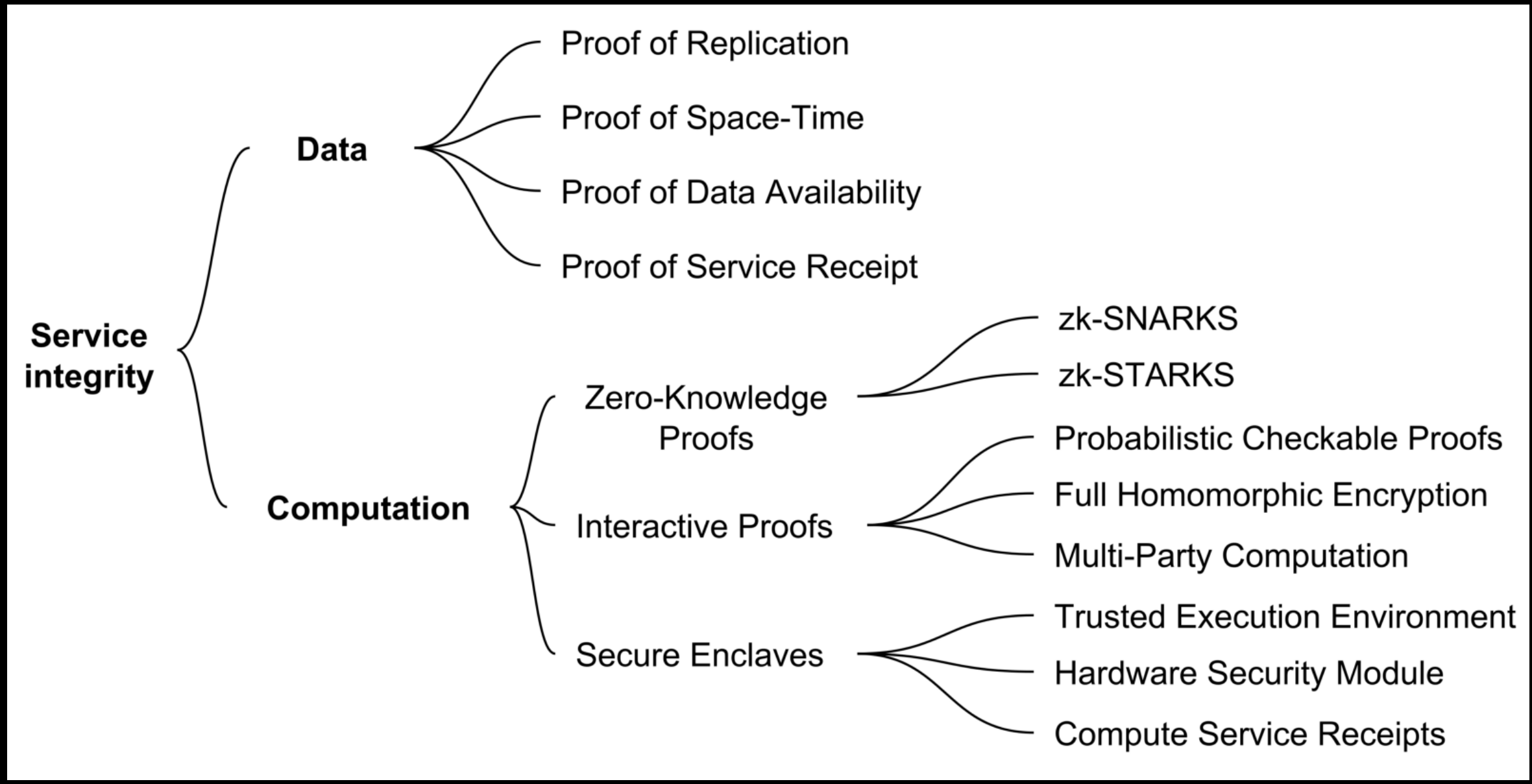
Proof of Data Availability

Approach A: FileCoin Proof of Space-Time (PoST), though that's overkill

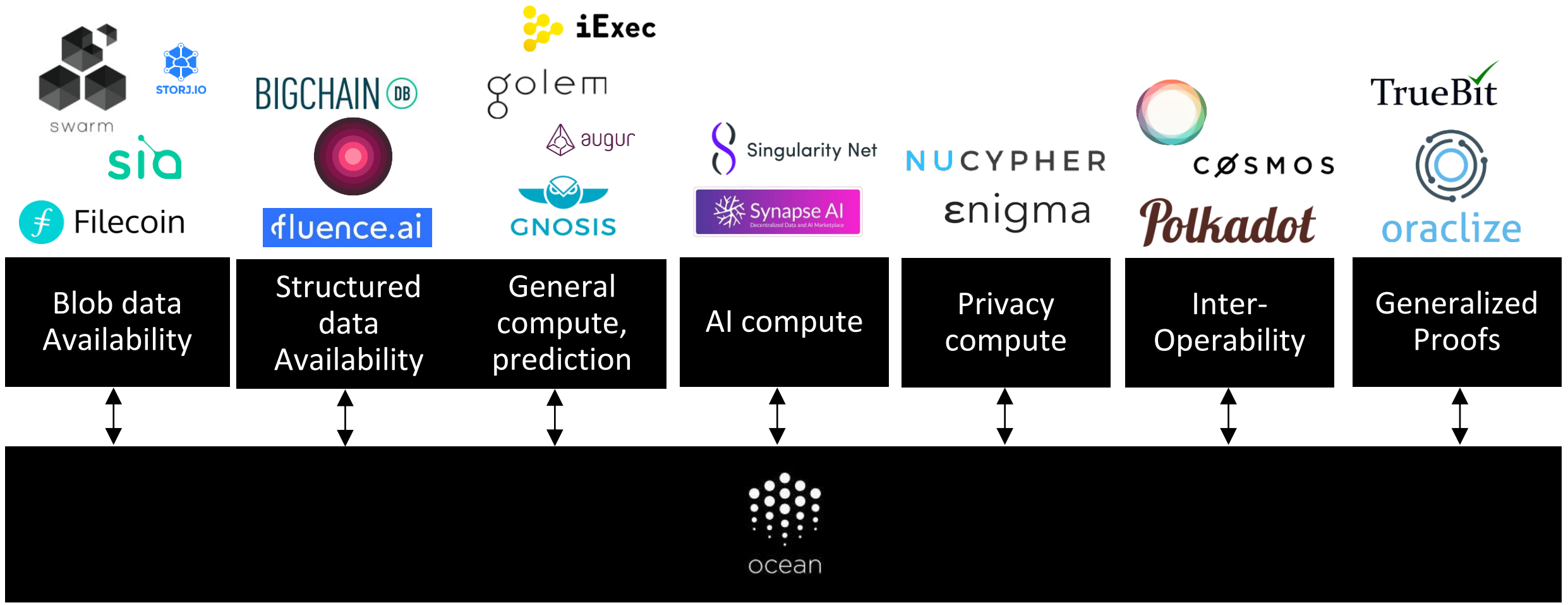
Approach B: TrueBit-style challenge-response:

1. I serve data to you
2. If you don't think you got it, you challenge (with stake)
    1. Randomly choose two actors to vet
    2. Actors vote on whether served
    3. If they agree that I served it, you lose stake
    4. If they agree that I didn't serve it , I lose stake
    5. Else

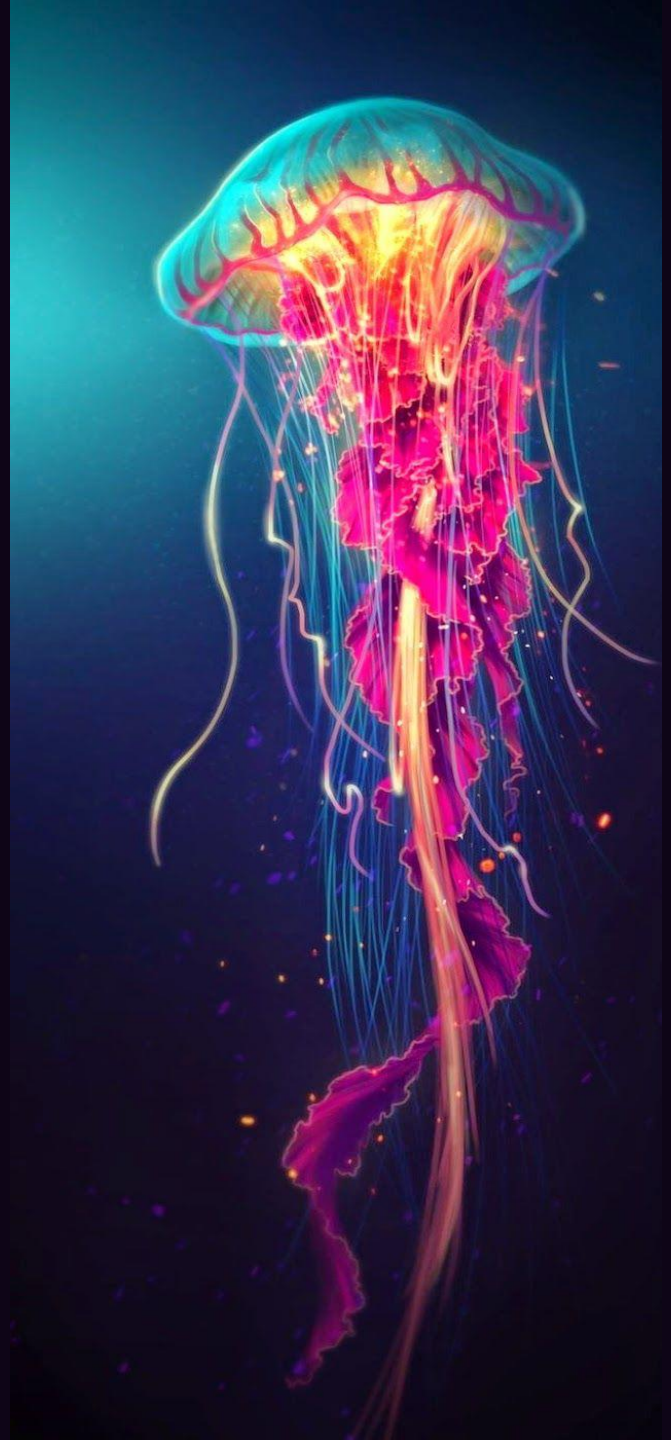
More Proofs


Service integrity
Data
Proof of Replication
Proof of Space-Time
Proof of Data Availability
Proof of Service Receipt
Computation
Zero-Knowledge Proofs
zk-SNARKS
zk-STARKS
Interactive Proofs
Probabilistic Checkable Proofs
Full Homomorphic Encryption
Multi-Party Computation
Secure Enclaves
Trusted Execution Environment
Hardware Security Module
Compute Service Receipts

With more proofs, more services!



*Note: logos shown are examples and do not imply partnerships or integrations



Token Curated Registry (TCR)

ocean

First, some attack vectors

elsa and anna attack - Go ×

Trent

Secure | https://www.google.de/search?q=elsa+and+anna+attack&tbm=isch&tbas=0&source=lnt&sa=X&ved=0...

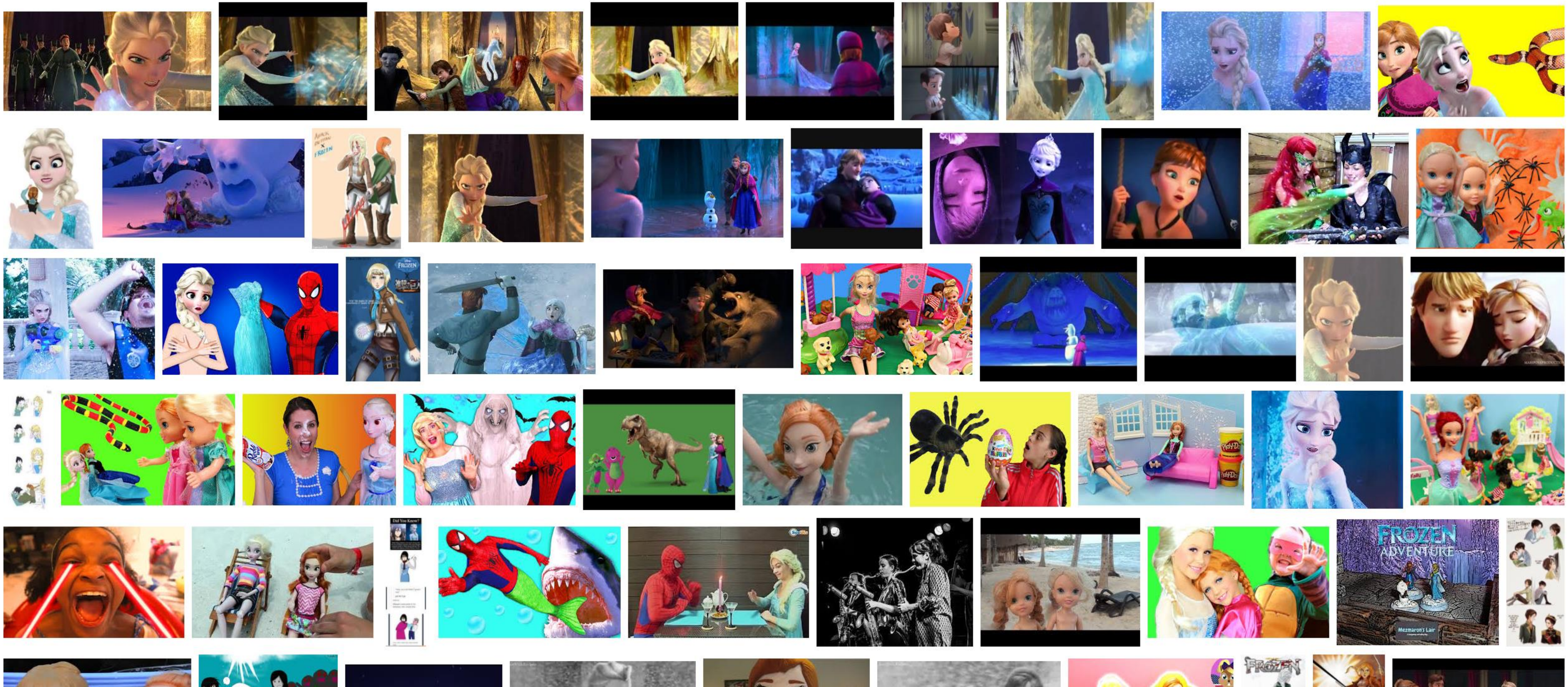
Google

elsa and anna attack

All | Images | Videos | Shopping | News | More | Settings | Tools

View saved | SafeSearch ▾

titan elsa | barbie | crossover | mikasa | hans | rapunzel | spiderman | aot | disney | disney frozen | gif | anime



Some attack vectors

- **Elsa & Anna Attack:** Someone uploads popular content that they don’t have rights to
- **Data Escapes:** People take data out of the system
- **Curation Clones:** Others create a new market for an existing dataset
- **Sybil Downloads:** Someone (or a ring of buddies) downloads a dataset they own >>1 times, to get >>rewards

How to approach?

Idea 1: network directly has legal arbitration

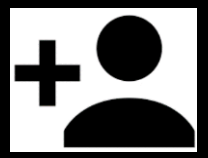
- Problem: it binds the network to jurisdictions (eek).
- *Need* to be permissionless and borderless!!

Idea 2: "sunny day tech, rainy day law"

- Core network itself has self-regulation by community. Incentivize good acting. Zero sum → positive sum. 99% case.
- Higher levels can have arbitration tied to jurisdictions. 1% case

TCR: a whitelist of good actors, curated by the actors themselves

New user B proposes for B to join



10 tokens stake by B

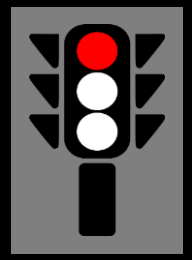
class TokenCuratedRegistry

def propose(data)
def challenge(proposal)
def vote(challenge)

User	status
A (Alice)	“Challenger”
B (Bob)	“New” + “Proposed”
Mallory	“OK”
Trent	“OK”

10 tokens stake by A

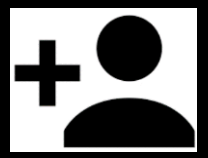
Existing user A challenges



ocean

TCR: a whitelist of good actors, curated by the actors themselves

New user B proposes for B to join



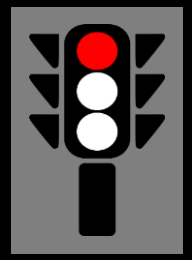
10 tokens stake by B

class TokenCuratedRegistry
def propose(data)
def challenge(proposal)
def vote(challenge)

User	status
A (Alice)	"Challenger"
B (Bob)	"New" + "Proposed"
Mallory	"OK"
Trent	"OK"

10 tokens stake by A

Existing user A challenges



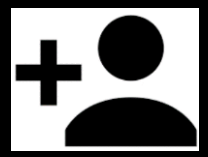
Concern:
High friction onboarding
7-28 day waiting period

ocean

TCR: a whitelist of good actors

"Trust is risk" for low-friction onboarding: vouch for others

Existing user
A *vouches*
for new user
B to join



10 tokens
stake by A,
B can run
away with it
anytime

class TokenCuratedRegistry

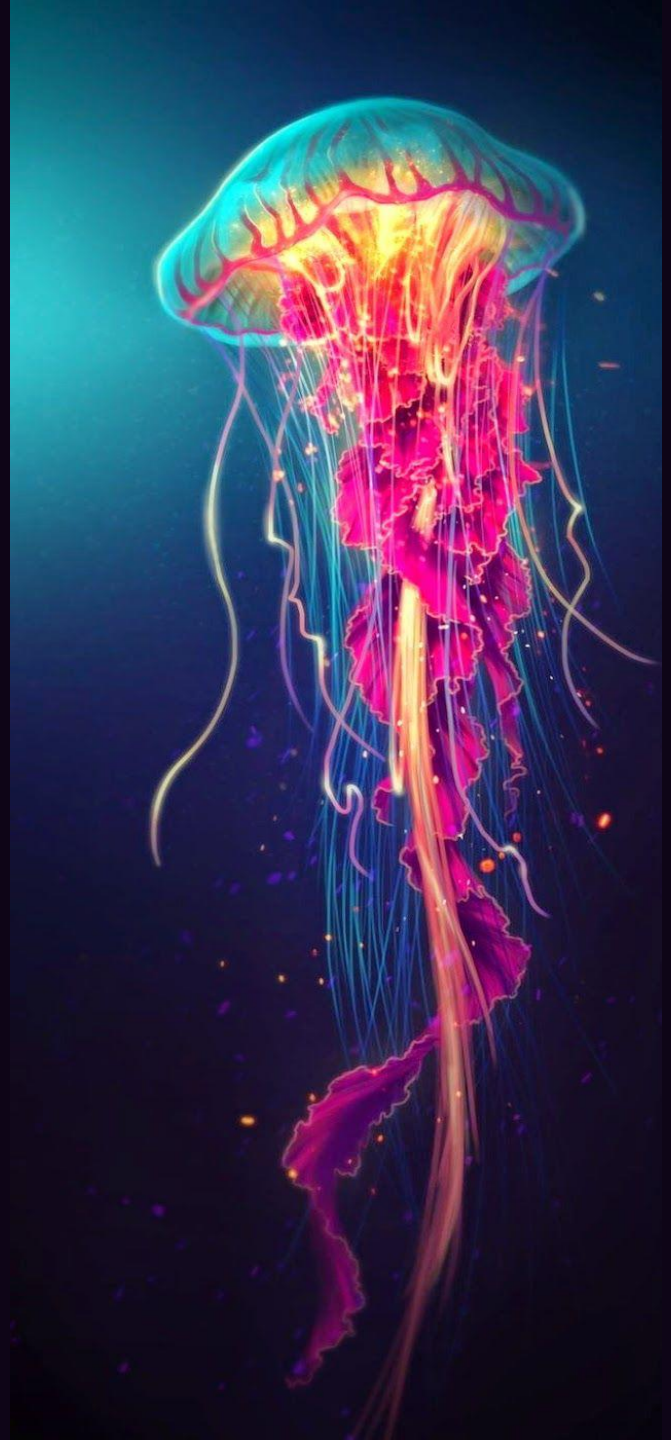
def **vouch**(data)
def challenge(proposal)
def vote(challenge)

User	status
A (Alice)	"Vouches"
B (Bob)	"New"
Mallory	"OK"
Trent	"OK"

Some attack vectors

- **Data Escapes:** People take data out of the system
- **Curation Clones:** Others create a new market for an existing dataset
- **Elsa & Anna Attack:** Someone uploads popular content that they don't have rights to
- **Sybil Downloads:** Someone (or a ring of buddies) downloads a dataset they own >>1 times, to get >>rewards

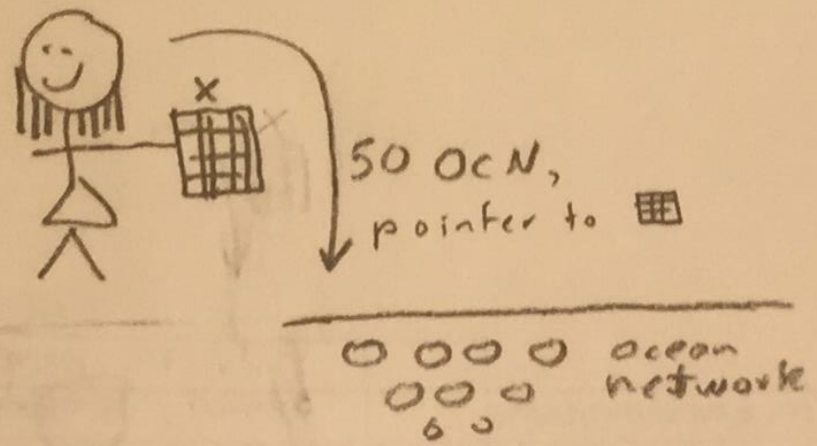
Whitelist of good actors addresses all of these!



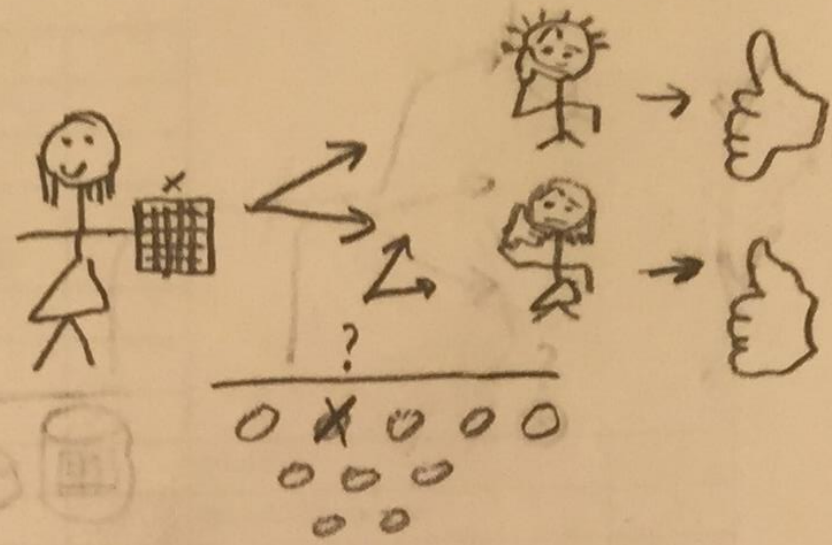
Walk Through

ocean

[Claim a dataset]



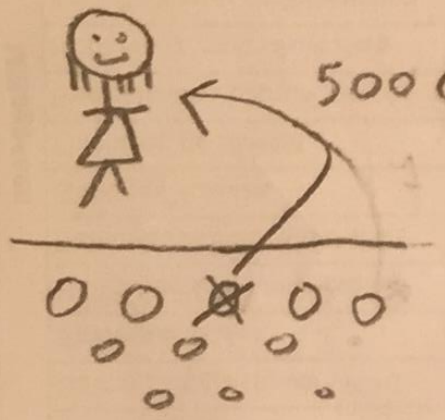
Alice claims copyright on dataset X.
and stakes 50 OCN to do so.

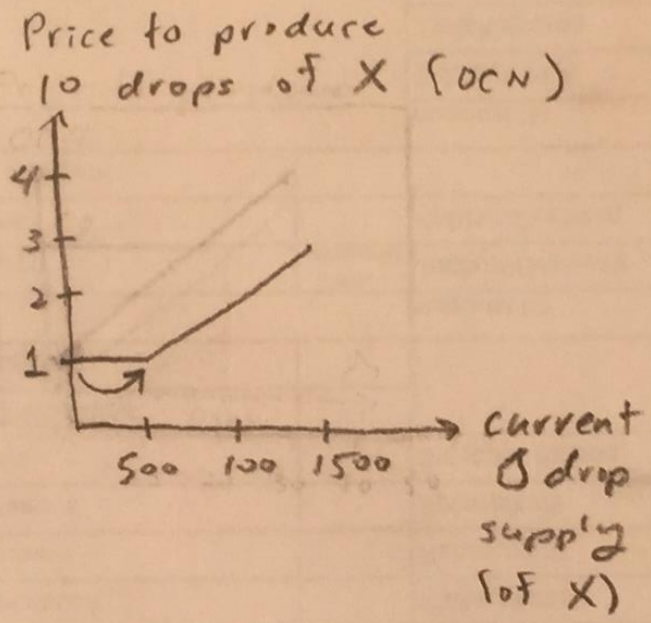


Others check. Data is ok!
5 day period concludes.

[Initiate curation, invest more]

①



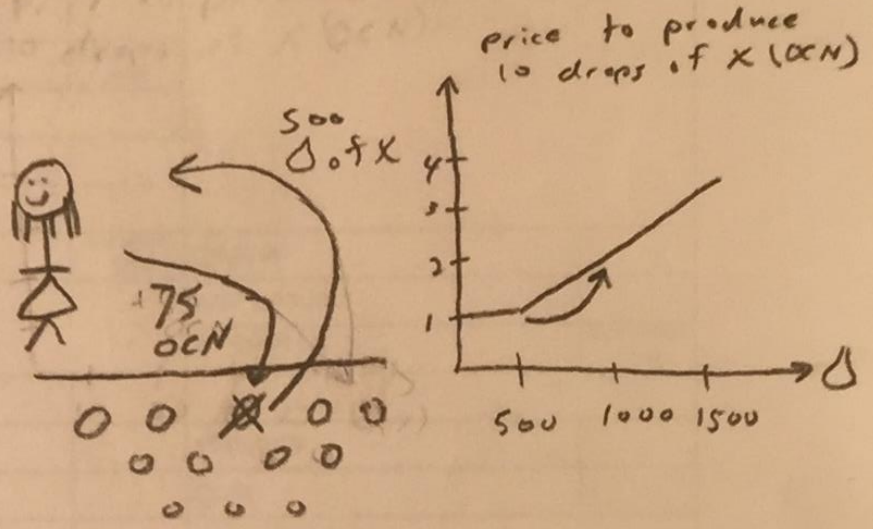


The curation market for X is initialized.

Because Alice staked 50 OCN, she gets $10 \cdot 50 = 500$ drops of X.

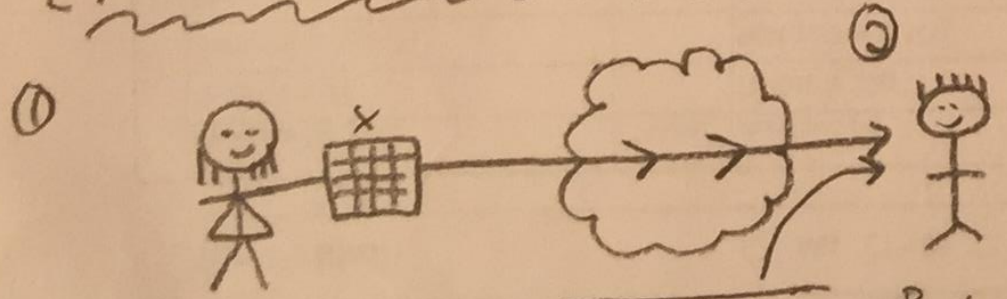
Drops have value for block rewards and when un-staking, as we will see.

②



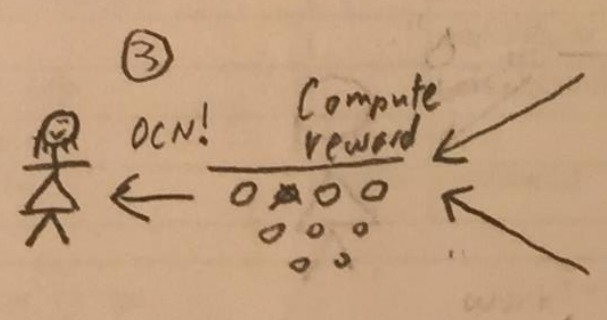
Alice has high confidence in the future popularity of X. So, she gets another 500 drops of X by staking $\frac{(1+2)}{2} \cdot 50 = 75$ OCN.

[Make data available, block rewards I]



Alice makes dataset X available for download.

Bob downloads the dataset for free, from Alice

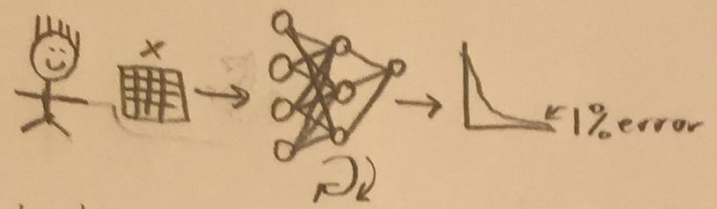


Alice gets block rewards (on average) because she staked on X, and served it when requested.

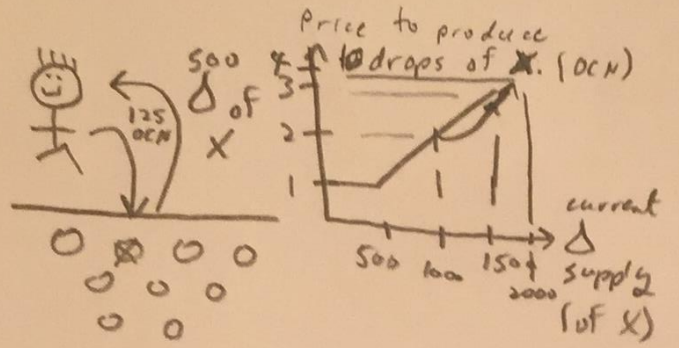
$$\text{Reward} \propto \frac{\text{Alice's } \Delta \text{ stake in X}}{\text{difficulty}^*} = \frac{1000}{\text{diff.}}$$

* details in whitepaper

[Referral / curation]

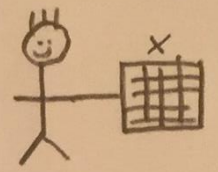
① 

Bob tries the dataset out on his AI model, and finds that it's really useful. Cool!

② 

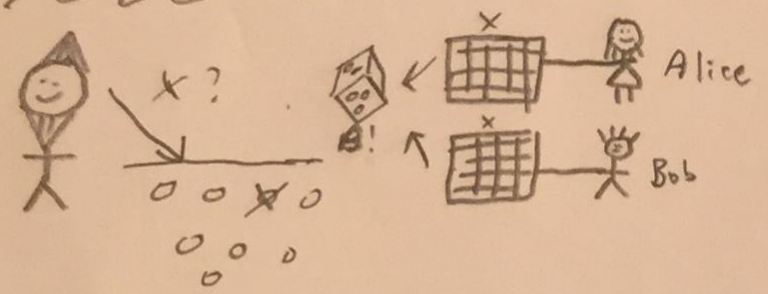
Bob realizes that others will find the dataset useful too. So, he gets 1000 drops of X by stating an average price of $\left(\frac{2+4}{2}\right) = 3$ OCN for every 10 drops of X.
So he spends 300 OCN for 1000 drops.

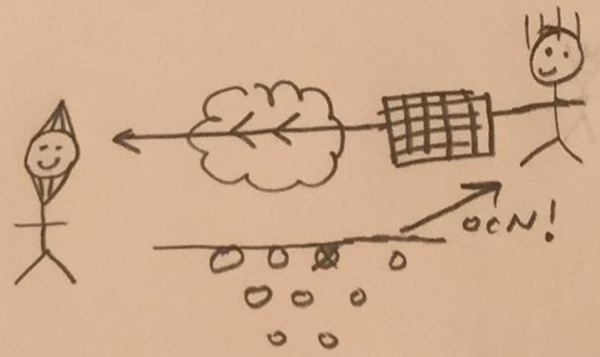
$\frac{3 \text{ OCN}}{10 \text{ drops}} = \frac{300}{1000}$

③ 

Bob makes dataset X available for download too.

[Block rewards II]





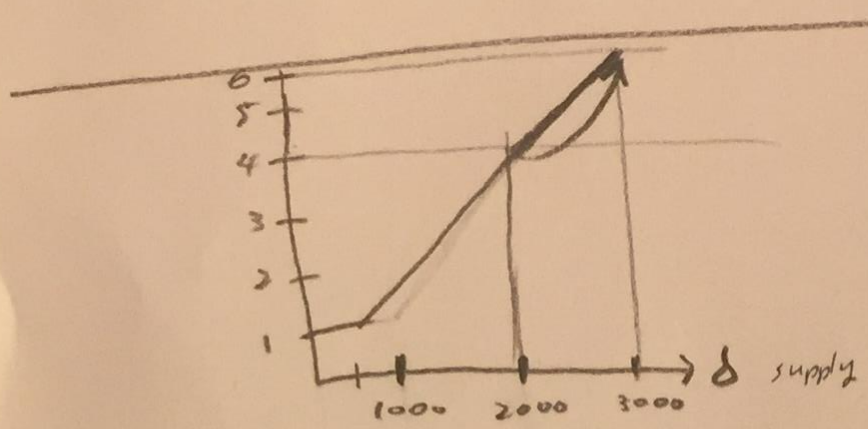
① Joe requests dataset X.

② From providers, Bob is randomly chosen

③ Joe downloads X from Bob

④ Bob gets block rewards (on average) because he staked on X, and served it when requested.

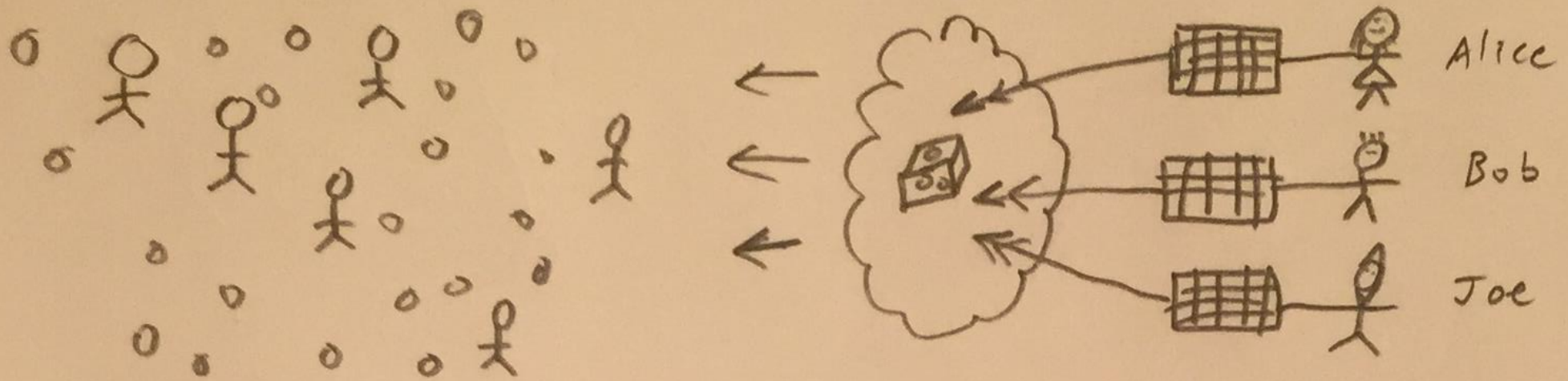
$$\text{Reward} \approx \frac{\text{Bob's } \delta\text{stake in X}}{\text{difficulty}} = \frac{1000}{\text{diff.}}$$



⑤ Joe likes the dataset too, he believes in its future popularity, so he buys 1000 drops of X, at an average price of $\left(\frac{4+6}{2}\right) = 5$ OCN / 10 drops, or 500 OCN for 1000 drops.

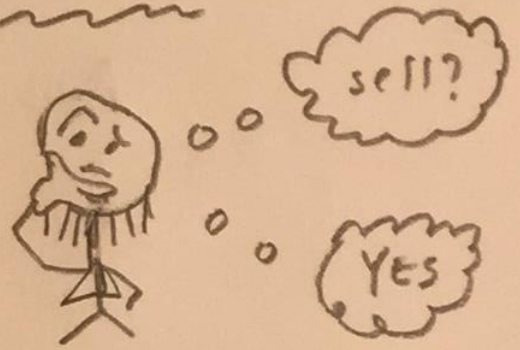
Note how Alice and Bob get the same reward on average because they both have 1000 δ. However, Alice staked 125 OCN versus 300 for Bob, because she was an earlier adopter!

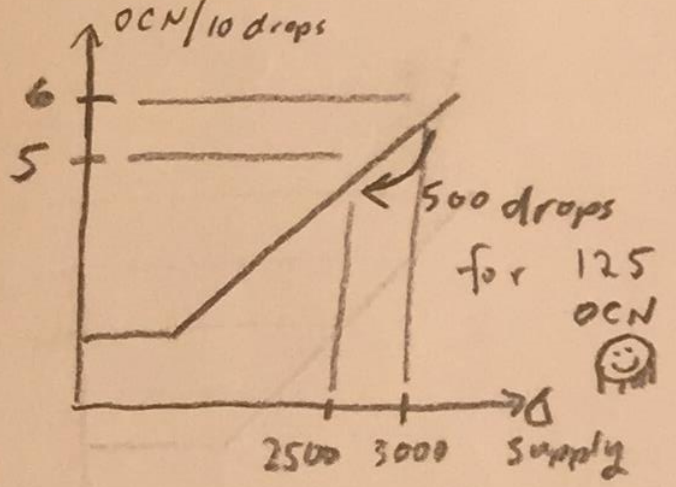
[Block rewards III: payoff on average]



Another 100 people download the dataset. It's becoming popular!
Alice, Bob, and Joe each serve up & approximately 1/3 the time,
and get equal rewards (because equal δ).

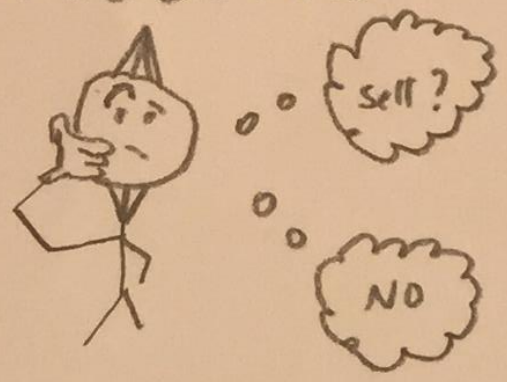
[Sell stake I]

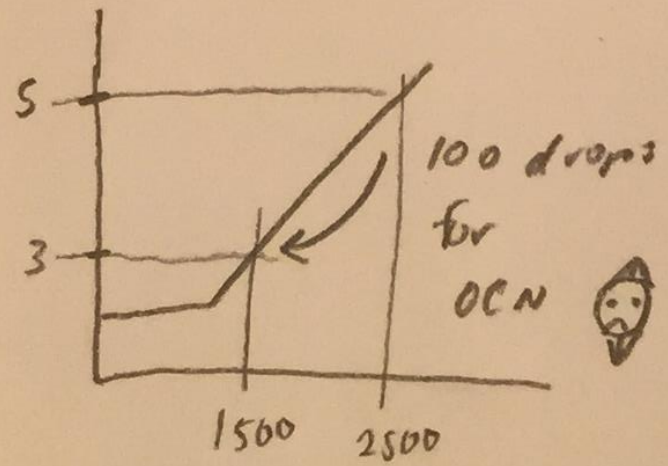




Alice considers selling 500 Δ of X.
Supply is currently 3000 Δ,
therefore average price is $\left(\frac{5+6}{2}\right) = 5.5$ OCN / 10 drops.
She sells, getting $(5.5)\cdot(50) = 275$ OCN.
Wow! She originally invested $50 + 75 = 125$ OCN,
and she's already got that back, and more!
Plus she's retained half her Δ.
Early adopter FTW!

[Sell stake II]





Joe considers selling 1000 ♁.

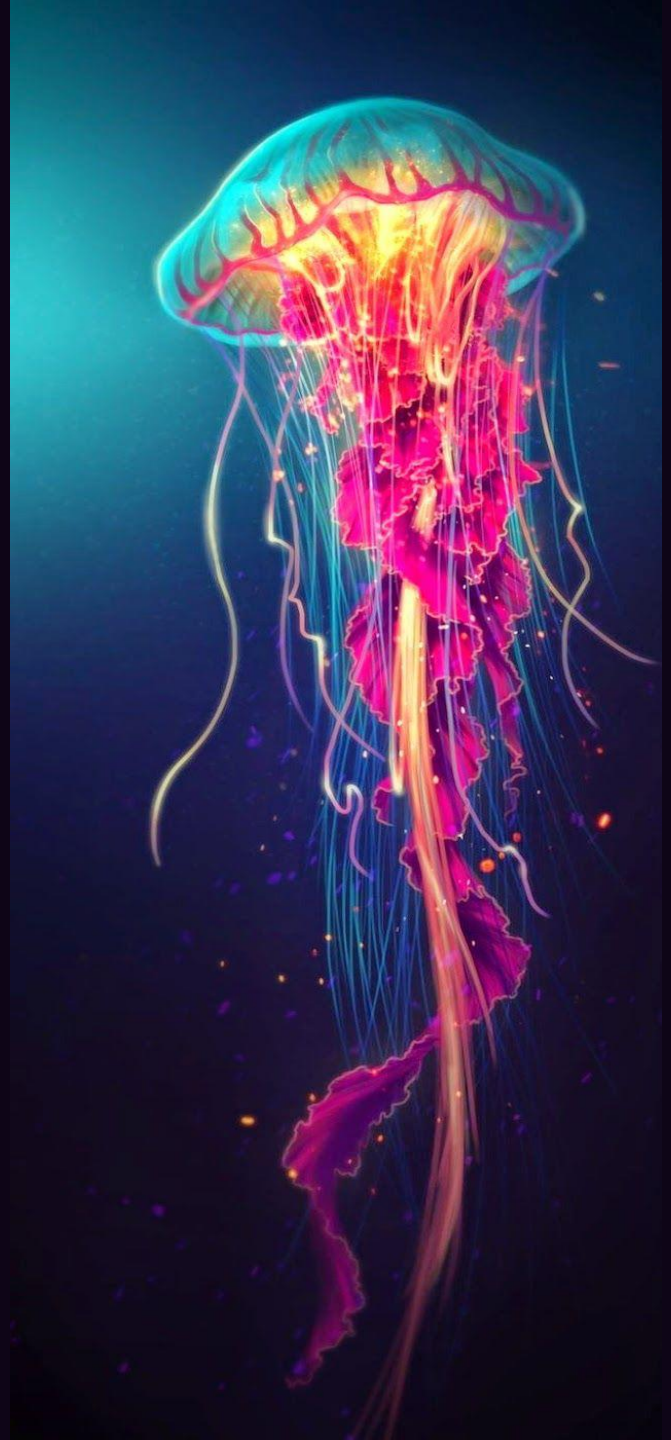
Average price is $\left(\frac{5+3}{2}\right) = 4$ OCN/10 drops.

He'd get (4)(100) = 400 OCN.

Recall that he spent 5 OCN/10 drops, or 500 OCN for 1000.

So, he'd lose money. He doesn't sell. He's a late adopter.

Instead, he waits; he thinks others might invest.



Conclusion

AI → Data silos → data crisis.

A thoughtful token design
has thoughtful building blocks.

For Ocean, this includes:

- **Curated Proofs Markets (CPMs)**
- **Token Curated Registry**

Trent McConaghy
@trentmc0

ocean