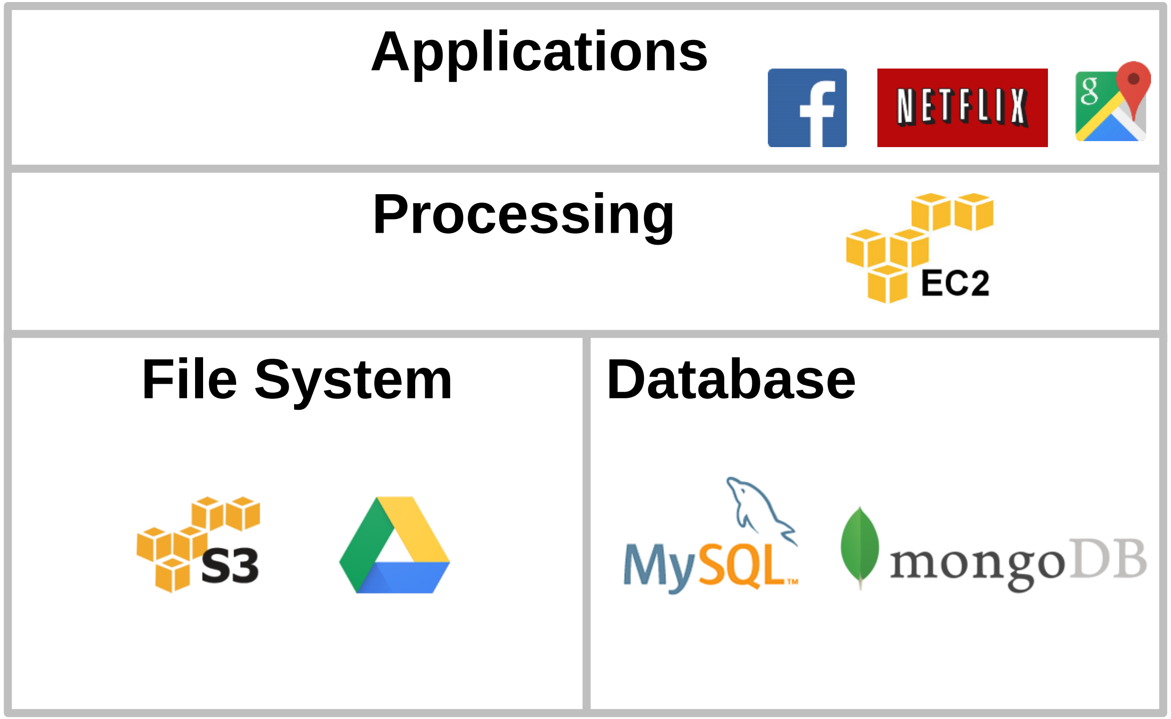
Blockchains in a Big Data World

Trent McConaghy

BIGCHAIN DB

The modern cloud application stack



Applications

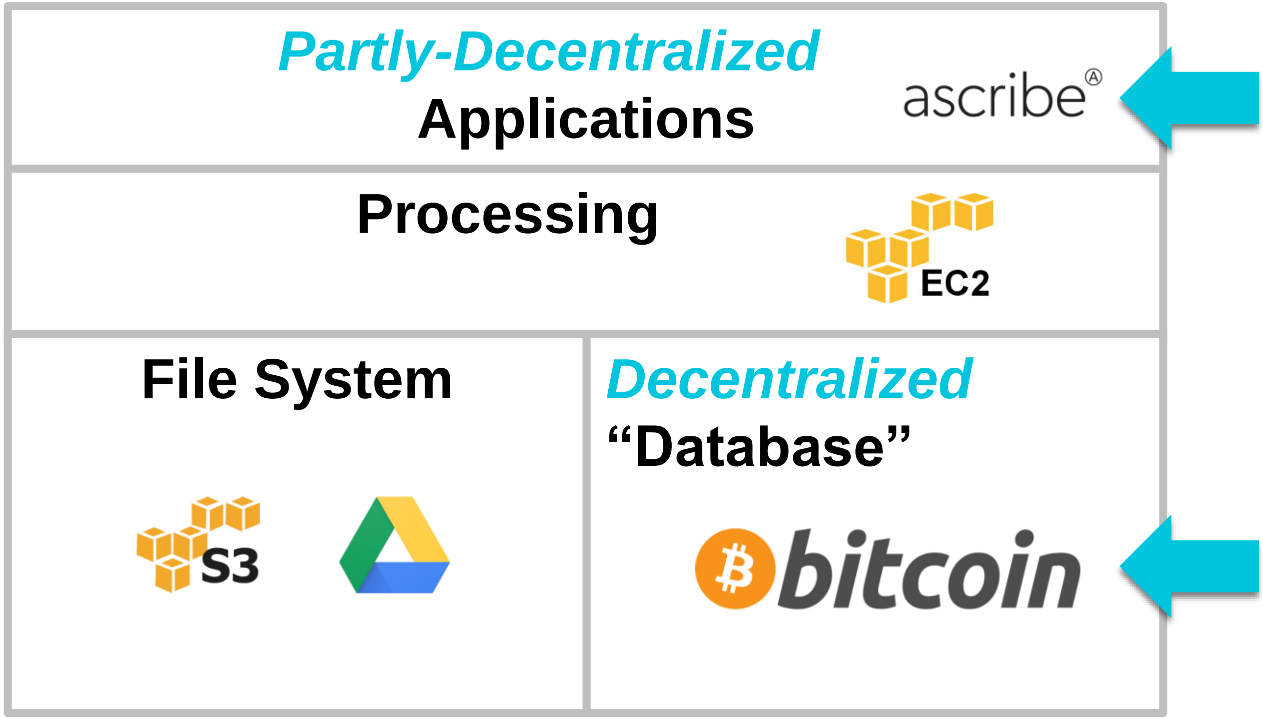
Processing

File System

Database

Along came Bitcoin…

The modern cloud application stack – with Bitcoin



Partly-Decentralized
Applications

Processing

File System

Decentralized
“Database”

Blockchain: A Special "Spreadsheet in the Sky"

What's special:

- no one owns it
- anyone can add to it
- no one can delete from it

- Writing to a blockchain is like etching in stone.
- Which allows us to issue assets, and transfer them
- Which can include art!

1.5 tx/s

50GB

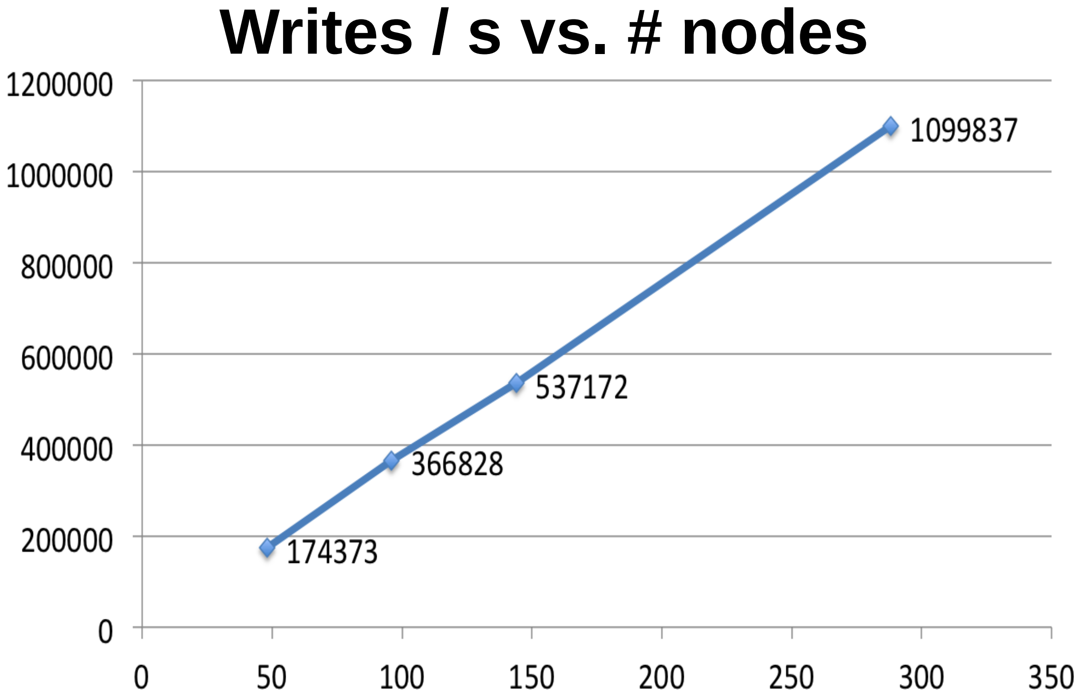
What about **planetary** scale?

Netflix uses 37% of Internet bandwidth

Netflix uses 37% of Internet bandwidth

Using a modern distributed “big data” database

Netflix uses 37% of Internet bandwidth
Using a modern distributed "big data" database



Two ways to scale up

Big data-fy blockchains

- Builds on man-decades of work
- Significant scalability hurdles

<or>

Blockchain-ify big data

- Builds on man-centuries (millennia?) of work
- Scalability challenges already resolved
- How to blockchain-ify? …

“Blockchain-ify”

Decentralization: no single entity owns or controls

Immutability: tamper-resistant

Assets: Can issue & transfer assets

Blockchain (noun): hashed-together chain of blocks (1991!)

Blockchain (noun): storage that is decentralized + immutable + assets

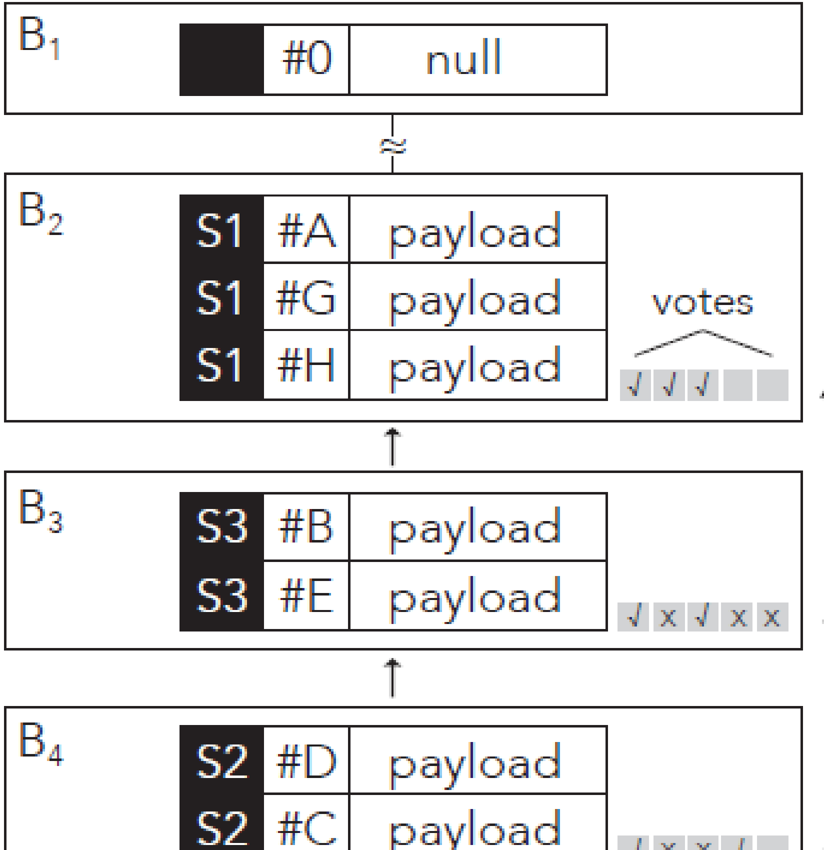
Blockchain (*adj*): decentralized + immutable + assets

How to Blockchain-ify Big Data

- **Decentralized:** each DB node is a federation node
- **Immutable:** hash on prev. blocks, append-only
- **Assets:** Interledger protocol

bigchaindb.com/whitepaper

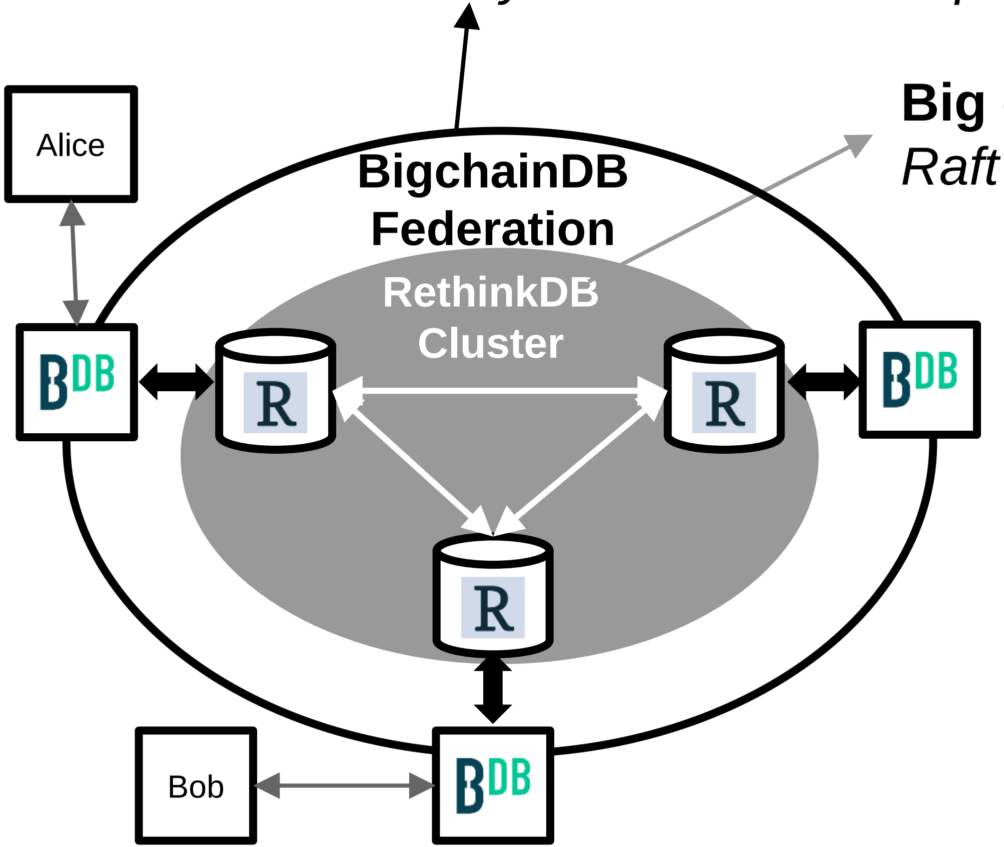
github.com/bigchaindb (AGPL)



Architecture

Blockchain consensus
Byzantine actors -> quorum

Big data consensus
Raft -> strong consistency



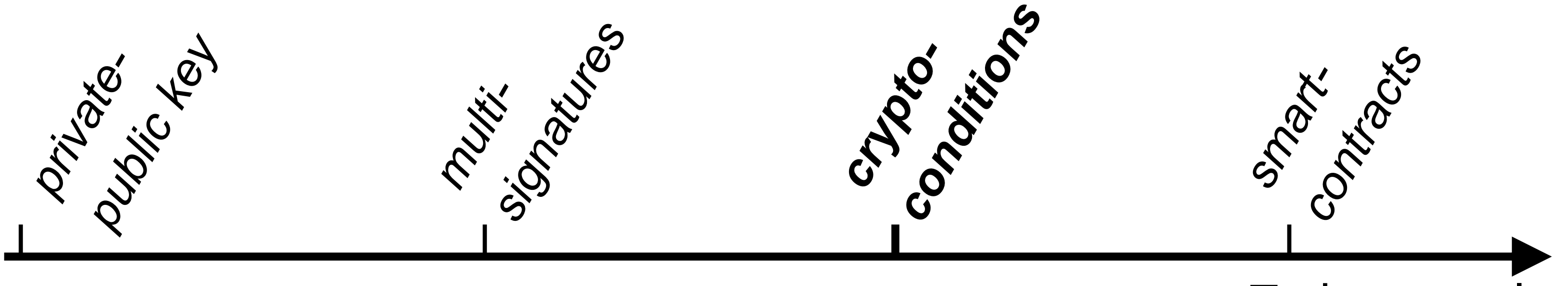
BigchainDB Interface

Database part : data

Via ReQL (JSON meets SQL)

+ Blockchain part : assets, transaction-style

Via Interledger Protocol (Crypto-conditions)



Turing completeness

BigchainDB characteristics

Throughput
>1,000,000 writes/s
~100,000 transactions/s

Latency
<100 ms

Capacity
Petabytes with each node adding 48TB

Scalability
Performance increases as nodes are added

Query
Database is fully queryable

Decentralization
Federated
non-anonymous participation

Public version of BigchainDB

iPDB

INTERPLANETARY DATABASE

iPDB

INTERPLANETARY DATABASE

- A shared global database. For everyone, everywhere
- And, a nonprofit foundation, with decentralized governance
- Powered by BigchainDB, to start
- Free except for high-volume users
- Caretakers co-operate network & co-govern foundation

IPDB Caretakers (so far)

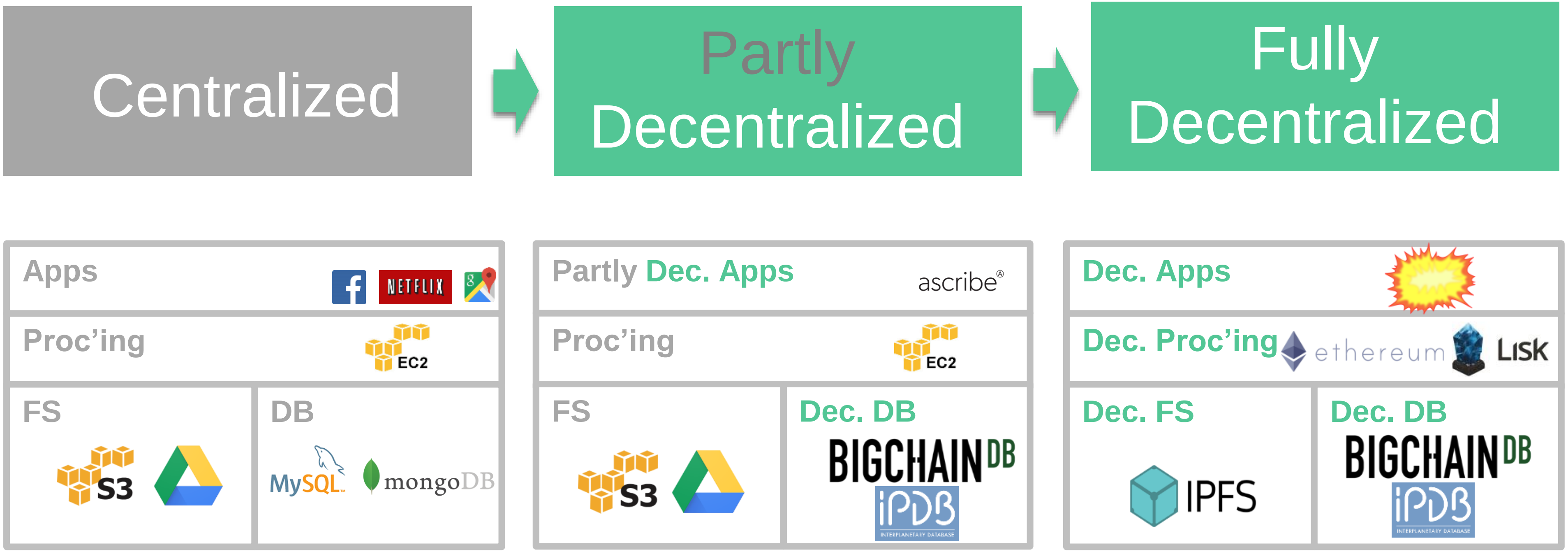
Not-for-profit

Blockstack
COALA
Dyne.org
Internet
  Archive
OpenMedia
UnMonastery

For-profit

BigchainDB
Consensys
Eris Industries
Protocol Labs (IPFS)
SmartContract.com
Synereo
Tendermint

Decentralization of the Cloud



Centralized → Partly Decentralized → Fully Decentralized

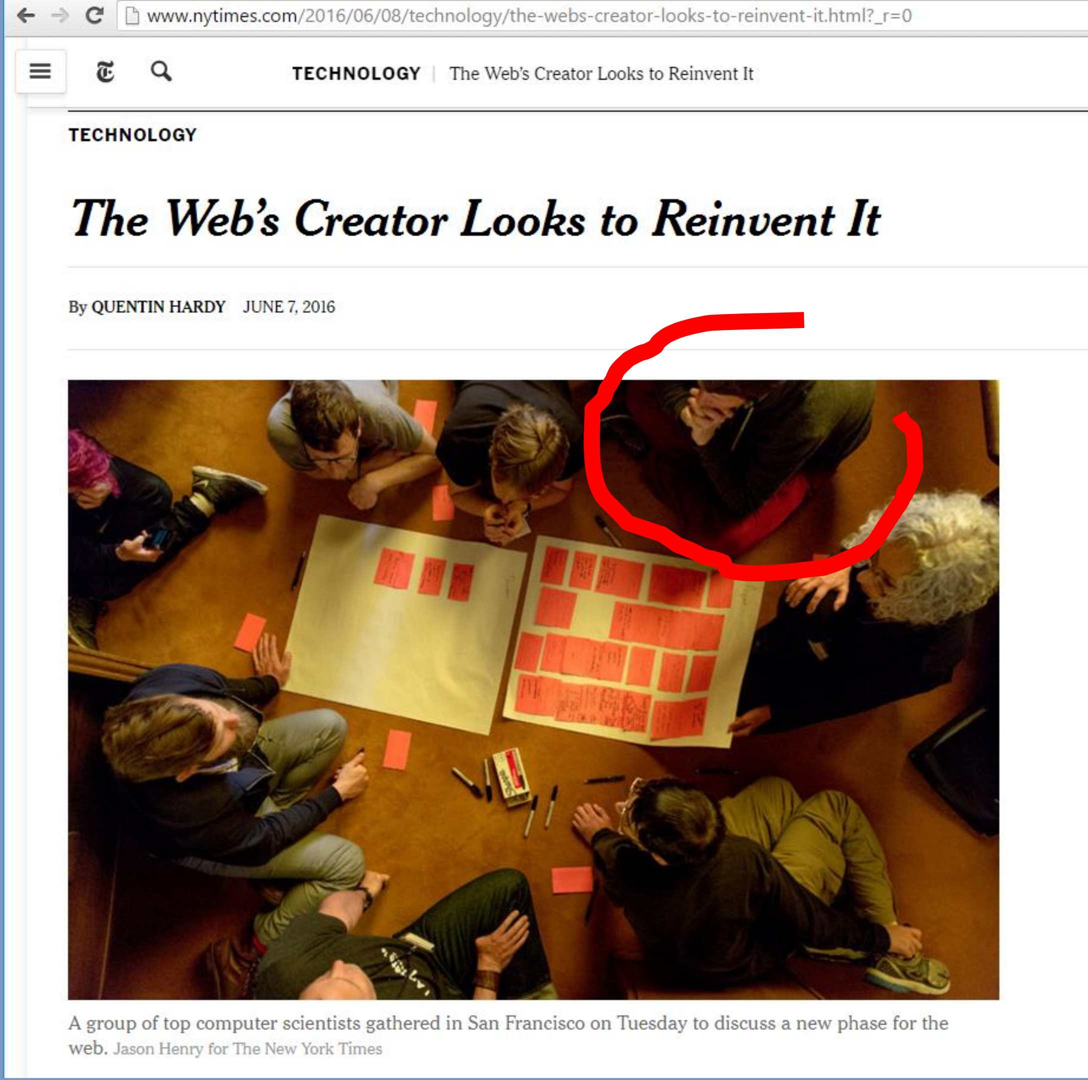
Centralized	Partly Decentralized	Fully Decentralized
Apps	Partly Dec. Apps	Dec. Apps
Proc'ing	Proc'ing	Dec. Proc'ing
FS / DB	FS / Dec. DB	Dec. FS / Dec. DB

The internet is getting upgraded, driven by the winds of blockchain.

Old + new guard are joining forces!

How to have lasting upgrade? New protocols.

W3C Blockchain, Coala IP, Copyright Hub / LCC, OMI, Interledger, IPLD, Web of Trust, Estonia e-identity

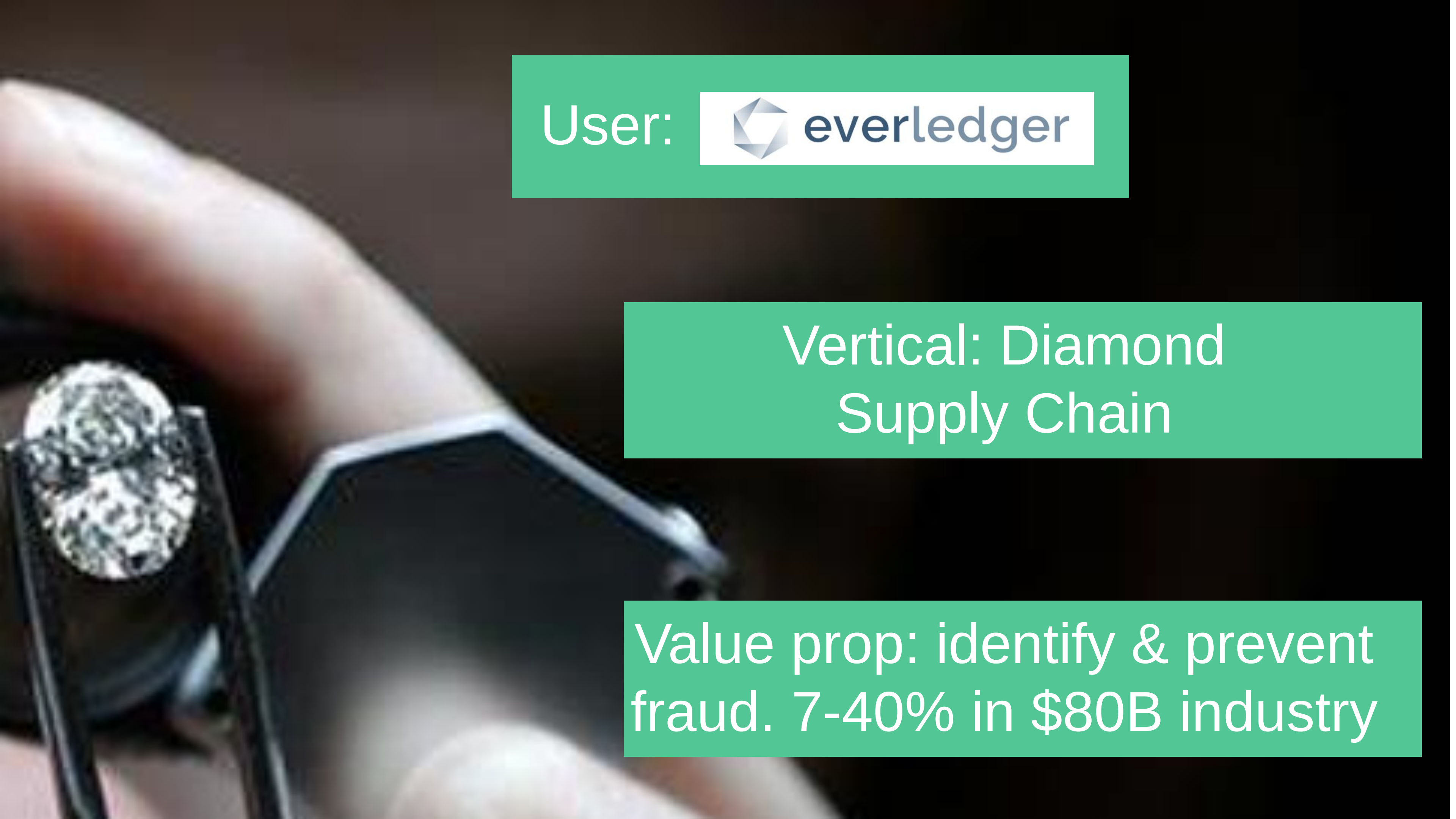

www.nytimes.com/2016/06/08/technology/the-webs-creator-looks-to-reinvent-it.html?_r=0
TECHNOLOGY | The Web's Creator Looks to Reinvent It

TECHNOLOGY

The Web's Creator Looks to Reinvent It

By QUENTIN HARDY JUNE 7, 2016

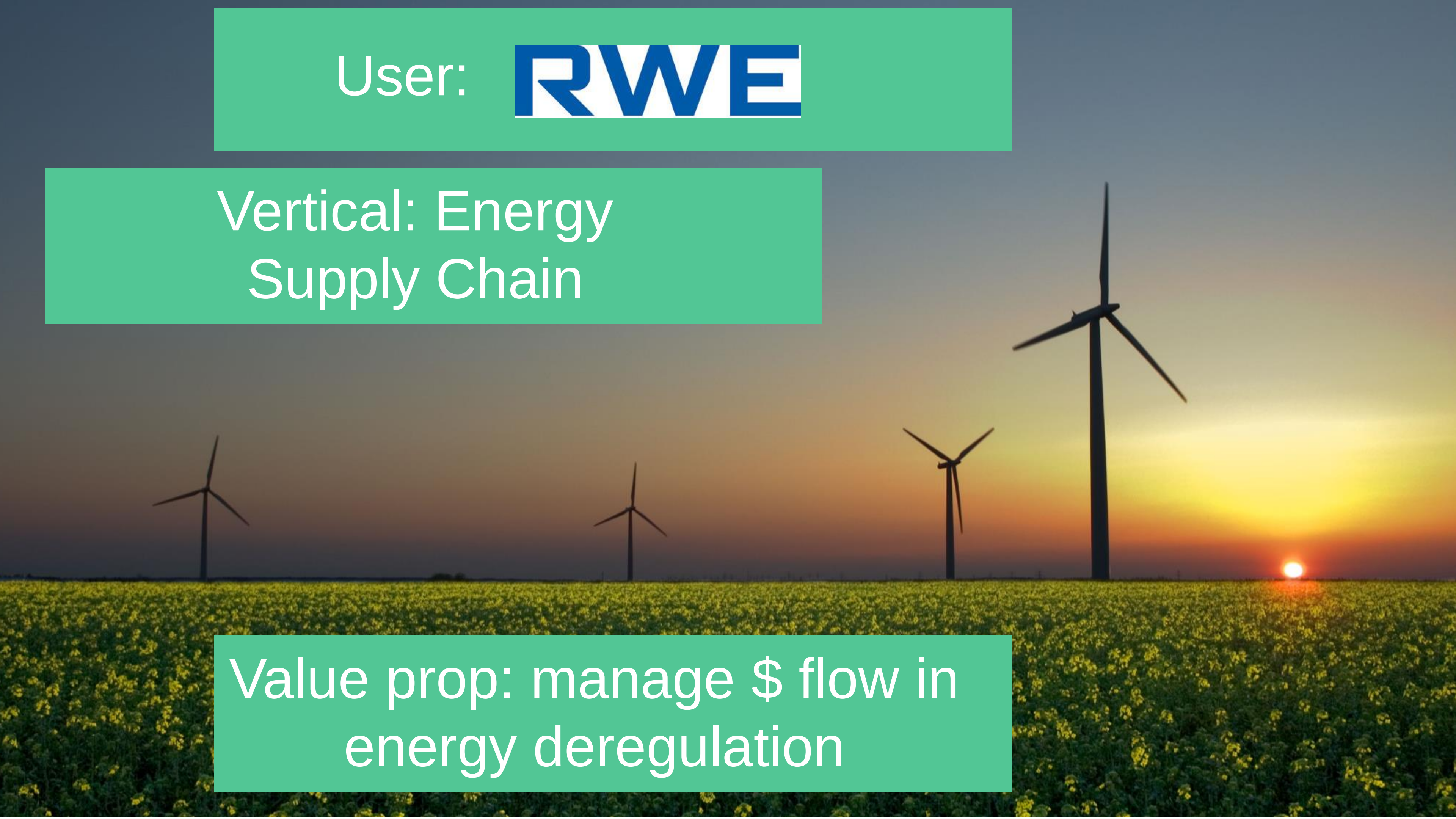
A group of top computer scientists gathered in San Francisco on Tuesday to discuss a new phase for the web. Jason Henry for The New York Times



User: everledger

Vertical: Diamond Supply Chain

Value prop: identify & prevent fraud. 7-40% in $80B industry



User: RWE

Vertical: Energy
Supply Chain

Value prop: manage $ flow in
energy deregulation

User: BenBen

Vertical:
Land registry

Value prop: less-corruptible land titles; land ownership as stepping stone

User: Tangent90

Vertical: Medical Journals / Supply Chain

Value prop: government-mandated transparent $ flow

User: RECRUIT

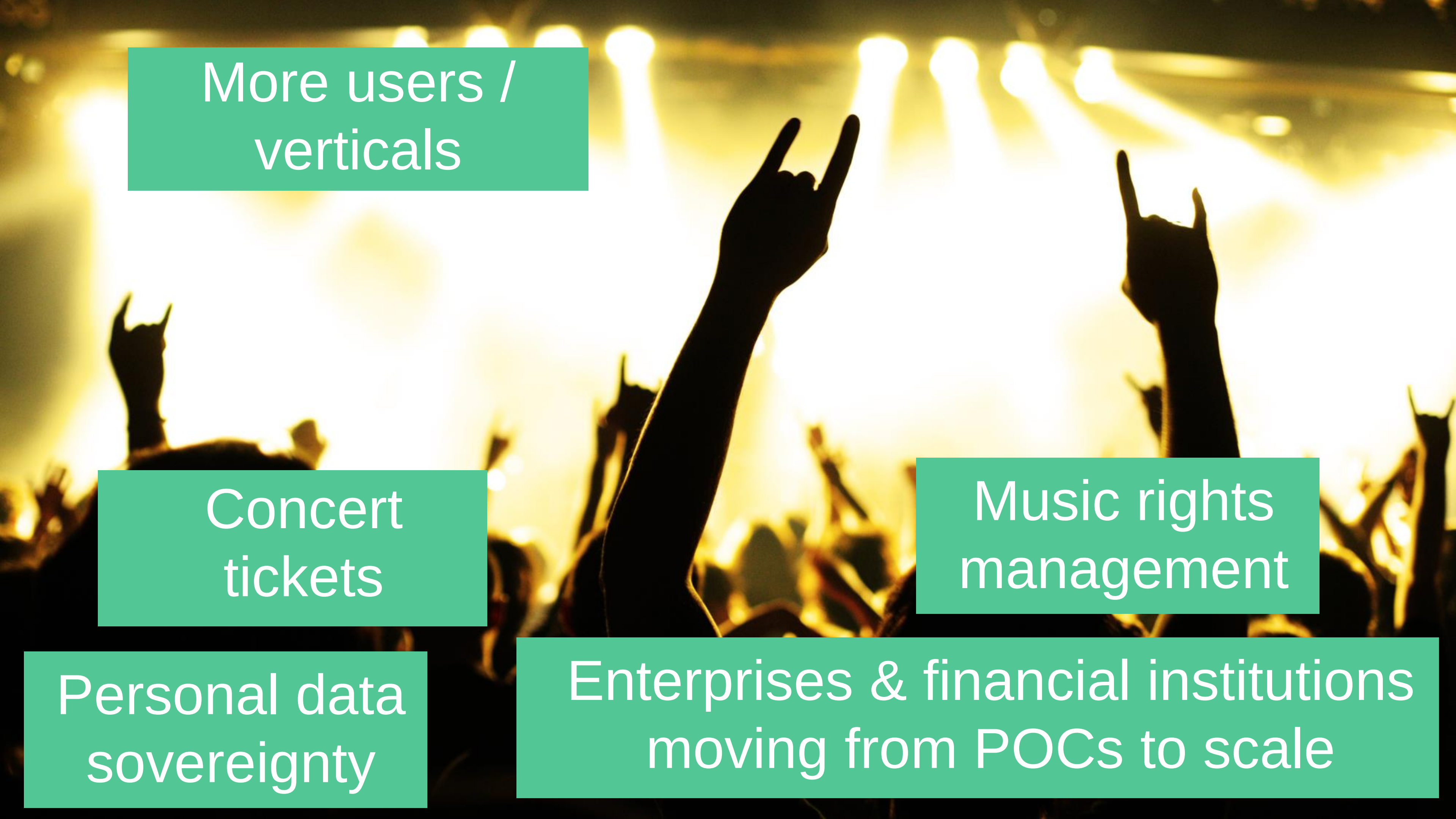
Vertical: Education credentials

Value prop: reduce fraudulent degrees, lower HR friction

User: ascribe.io
(incl. 5000 artists, 25 orgs)

Verticals: Art Supply Chain, Intellectual Property

Value Props: secure provenance, IP mgmt.



More users / verticals

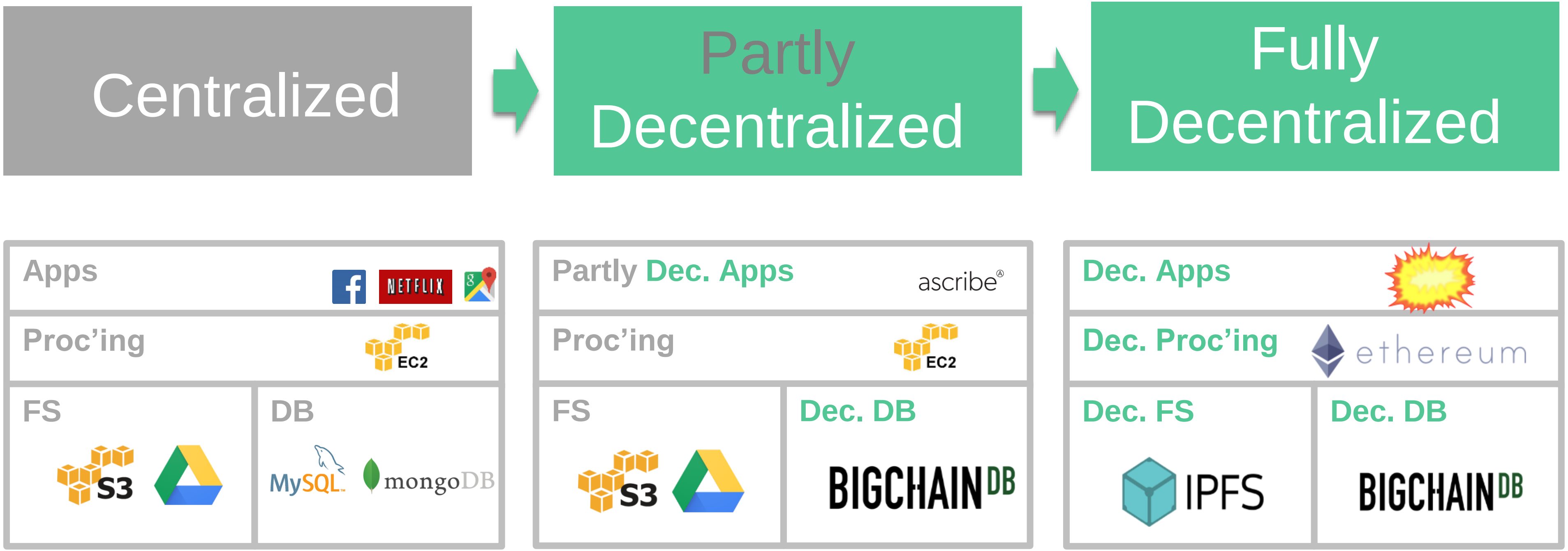
Concert tickets

Music rights management

Personal data sovereignty

Enterprises & financial institutions moving from POCs to scale

Decentralization of the Cloud



Centralized → Partly Decentralized → Fully Decentralized

Centralized	Partly Decentralized	Fully Decentralized
Apps	Partly Dec. Apps	Dec. Apps
Proc'ing	Proc'ing	Dec. Proc'ing
FS DB	FS Dec. DB	Dec. FS Dec. DB

Big data, meet blockchain.

- Blockchain DBs merges best of both worlds: blockchain and big data
- BigchainDB is the world’s first instance of a blockchain DB
- It’s a part of a broader evolution: the re-decentralization of the web. Along with decentralized processing, FS, ..

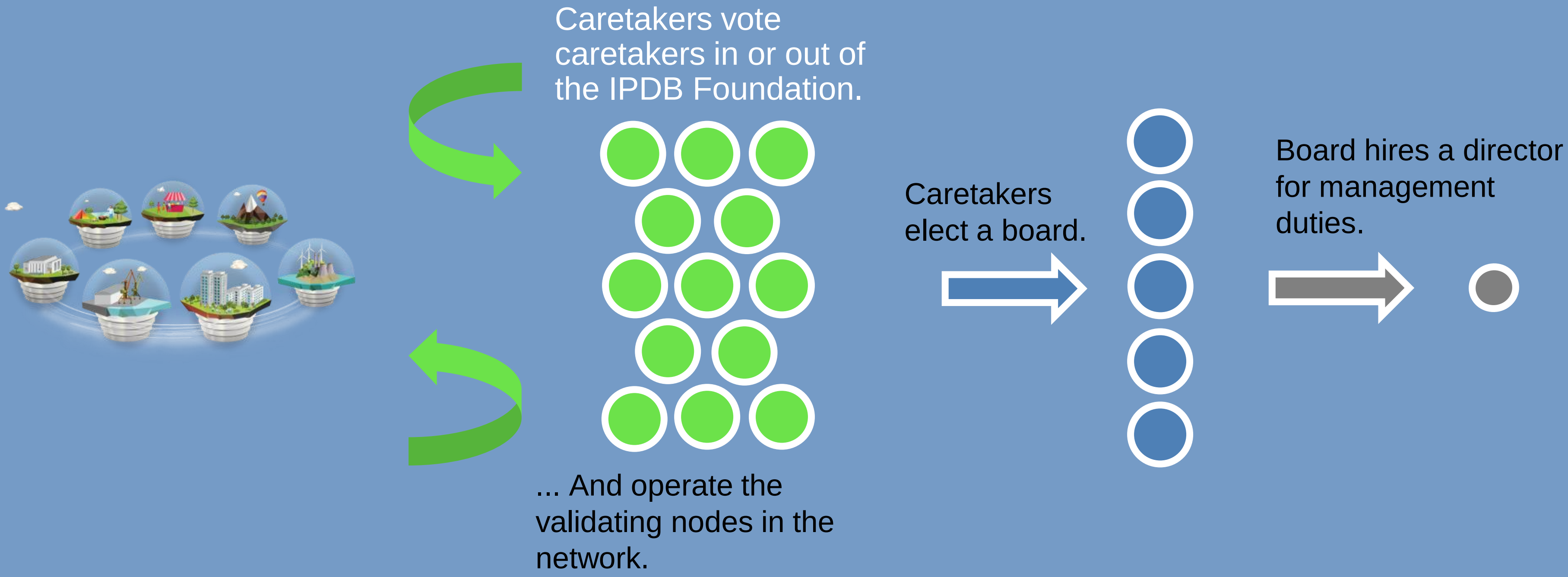
More info:

- bigchaindb.com
- github.com/bigchaindb
- github.com/bigchaindb/org/blob/master/engjob.md

@trentmc0

APPENDIX: IPDB

IPDB Governance: caretakers at the heart



Caretakers vote caretakers in or out of the IPDB Foundation.

... And operate the validating nodes in the network.

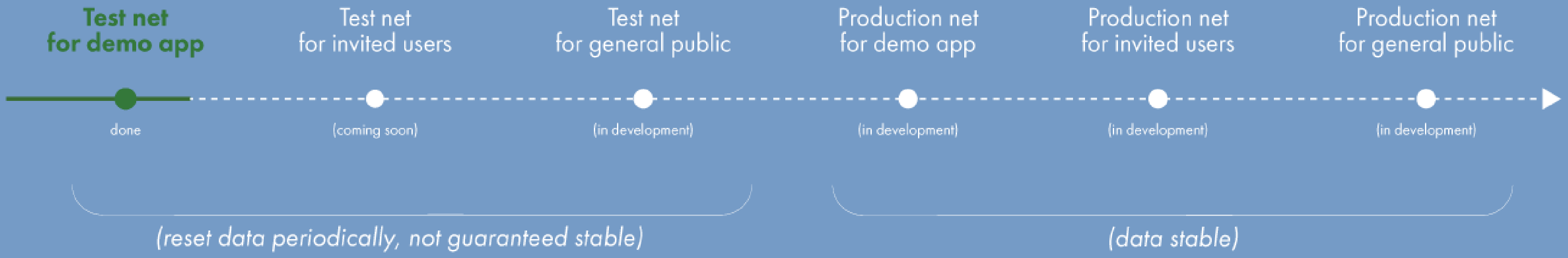
Caretakers elect a board.

Board hires a director for management duties.

Yes, this could be a DAO.

But not yet. Walk before we run.

IPDB Roadmap



Test net
for demo app
done

Test net
for invited users
(coming soon)

Test net
for general public
(in development)

Production net
for demo app
(in development)

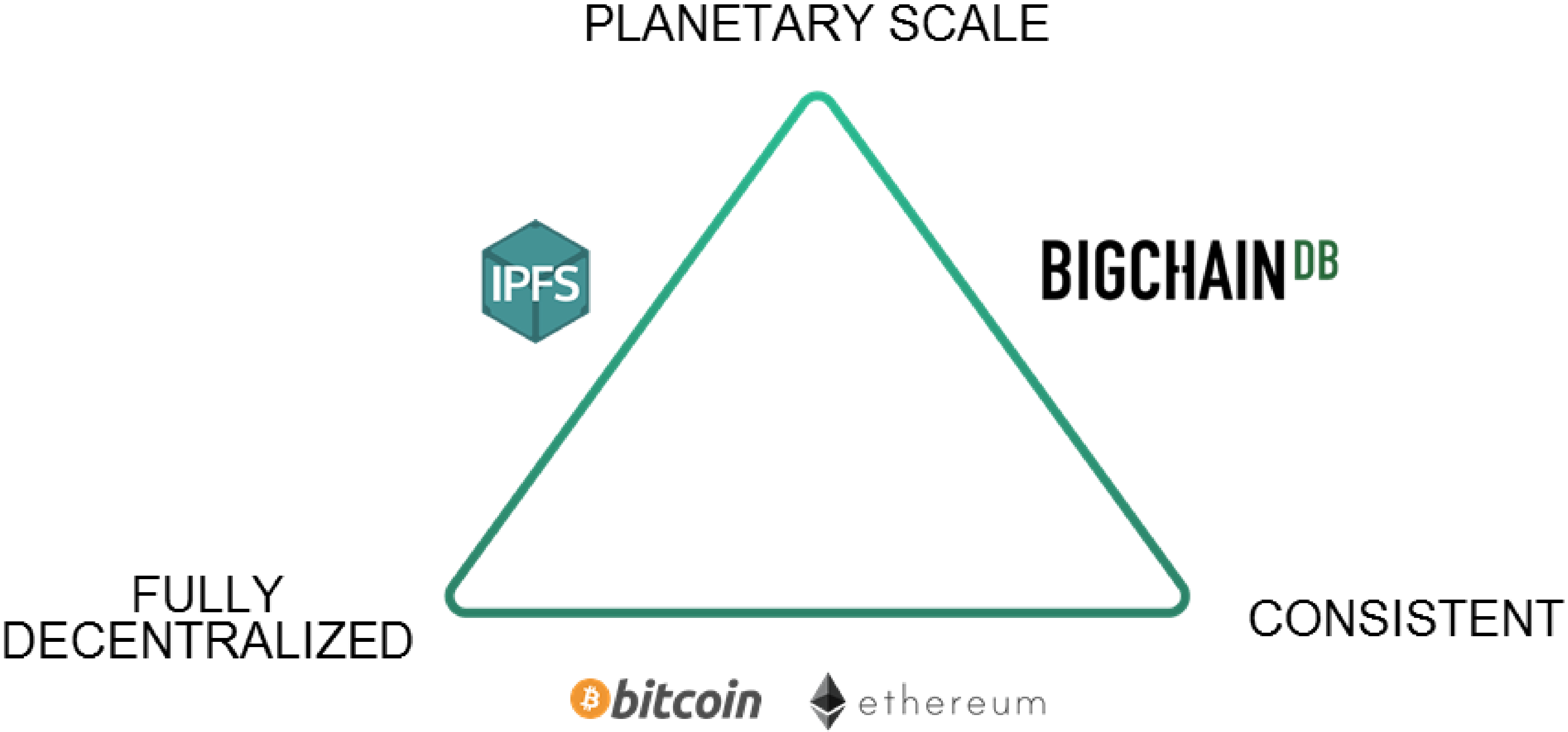
Production net
for invited users
(in development)

Production net
for general public
(in development)

(reset data periodically, not guaranteed stable)

(data stable)

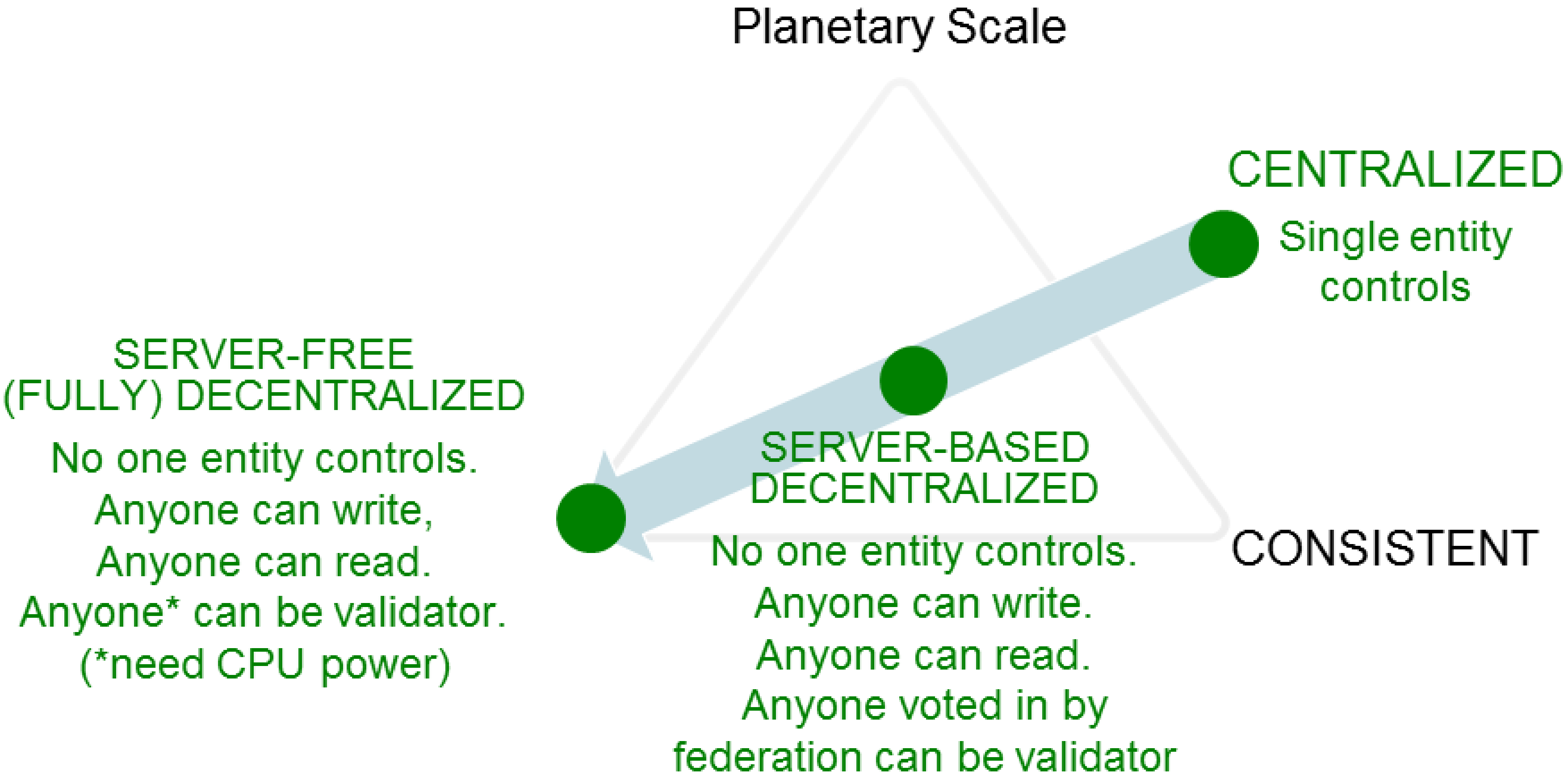
APPENDIX: TRADEOFFS



PLANETARY SCALE

FULLY DECENTRALIZED

CONSISTENT



Planetary Scale

CENTRALIZED

Single entity controls

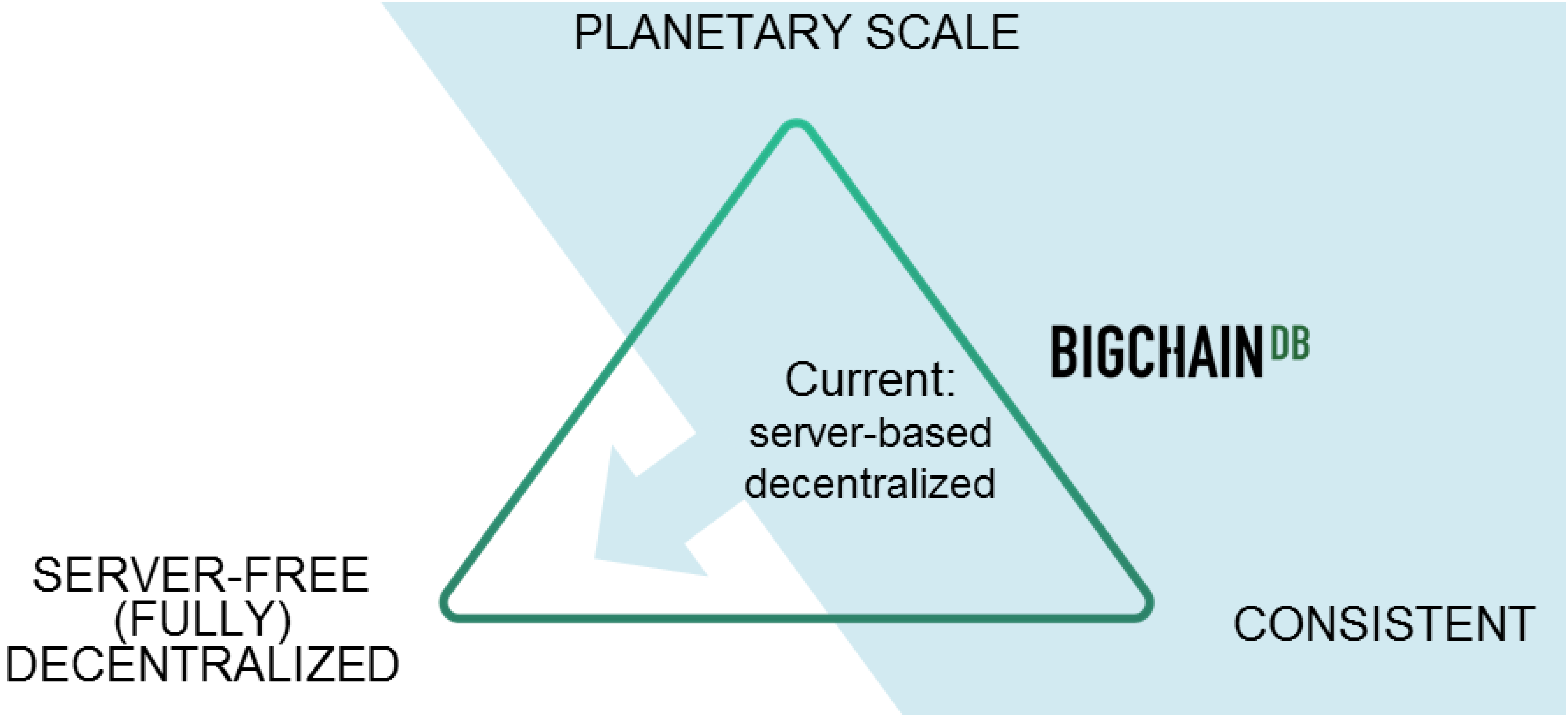
SERVER-BASED DECENTRALIZED

No one entity controls.
Anyone can write.
Anyone can read.
Anyone voted in by federation can be validator

SERVER-FREE (FULLY) DECENTRALIZED

No one entity controls.
Anyone can write,
Anyone can read.
Anyone* can be validator.
(*need CPU power)

CONSISTENT



PLANETARY SCALE

BIGCHAIN DB

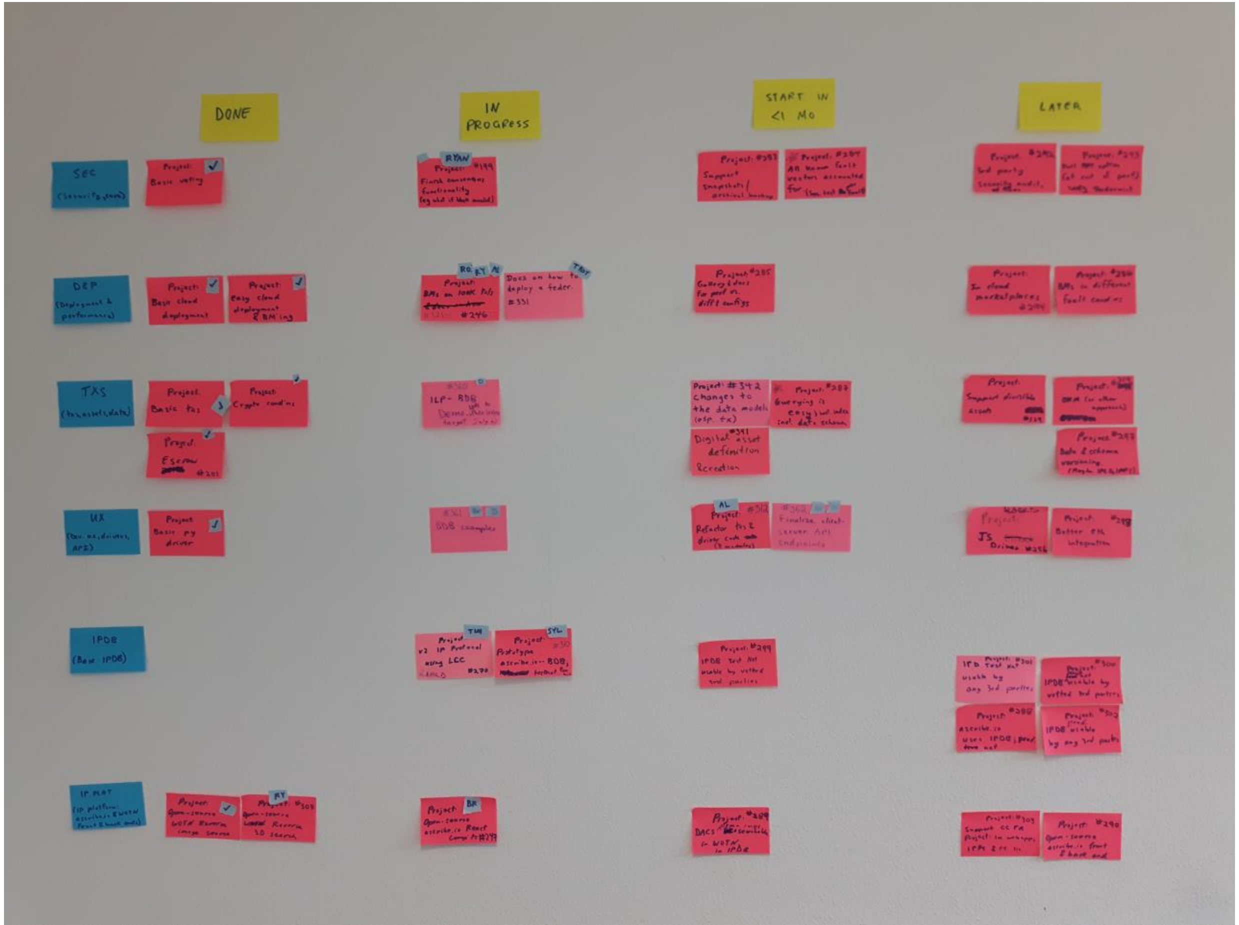
Current:
server-based
decentralized

SERVER-FREE
(FULLY)
DECENTRALIZED

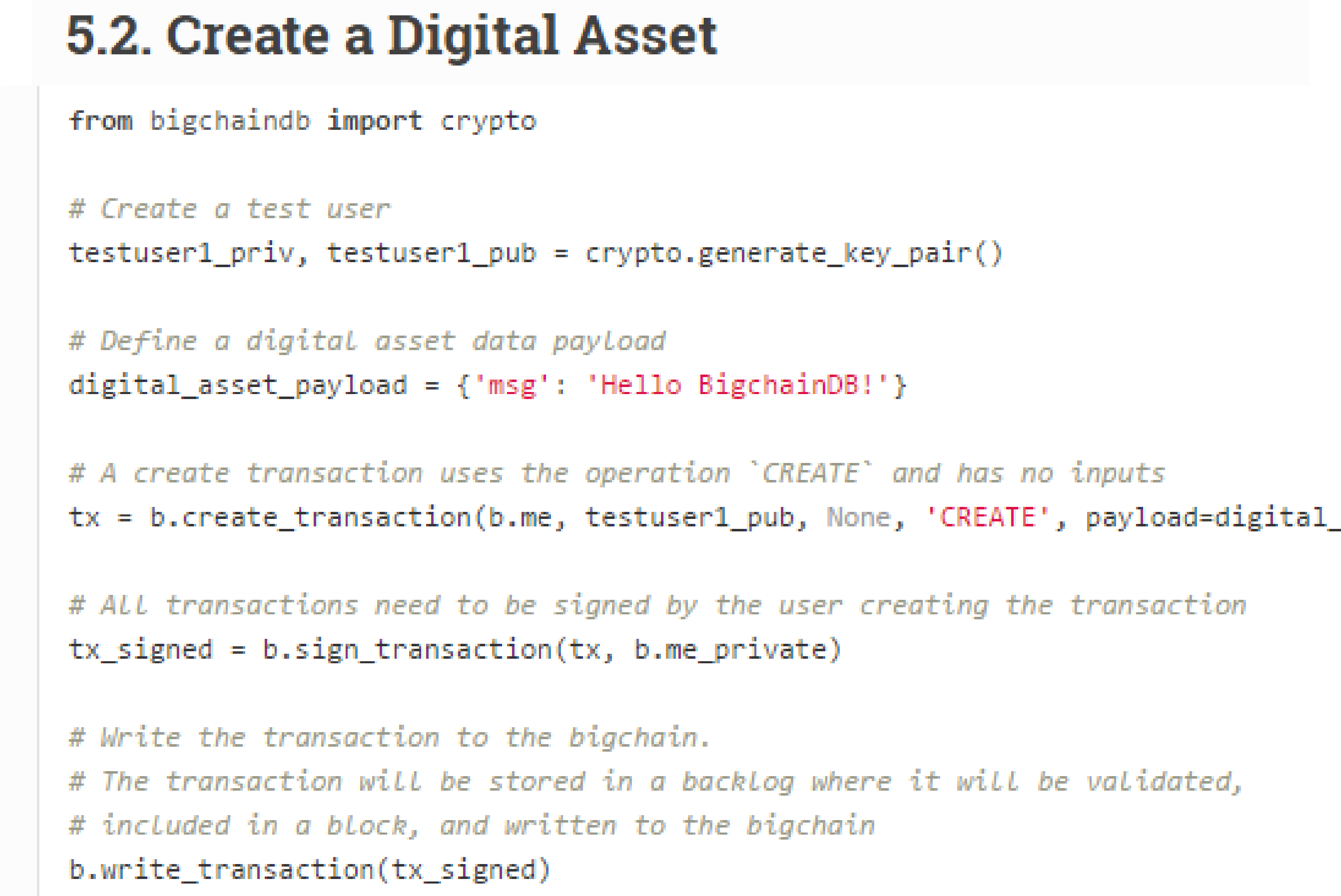
CONSISTENT

APPENDIX: ROADMAP

https://github.com/bigchaindb/org/blob/master/ROADMAP.md


DONE
IN PROGRESS
START IN <1 MO
LATER
SEC
DEP
(Deployment & performance)
TXS
UX
IPDB
(Base IPDB)
IP PLAT
Project: Basic voting
Project: Basic cloud deployment
Project: Easy cloud deployment & BM'ing
Project: Basic txs
Project: Crypto conditions
Project: Escrow #201
Project: Basic py driver
RYAN
Project: #199 Finish consensus functionality
Docs on how to deploy a feder. #331
Project: BMs on 100K Tx/s #246
TROY
ILP - BDB Demo
BDB examples
Project: #283 Support snapshots / archival backup
Project: #285
Project: #342 changes to the data models (esp. tx)
#341 Digital asset definition
Project: #287
AL
Project: #312 Refactor tx & driver code
Finalize client-server API endpoints
Project: #299
Project: #289
Project: Support divisible assets
Project: #288 Data & schema versioning
JS Driver #256
Project: #298 Better ETH integration
TIM
Project v2 IP Protocol using LCC #270
SYL
BR
RY
Project: #303
Project: #290

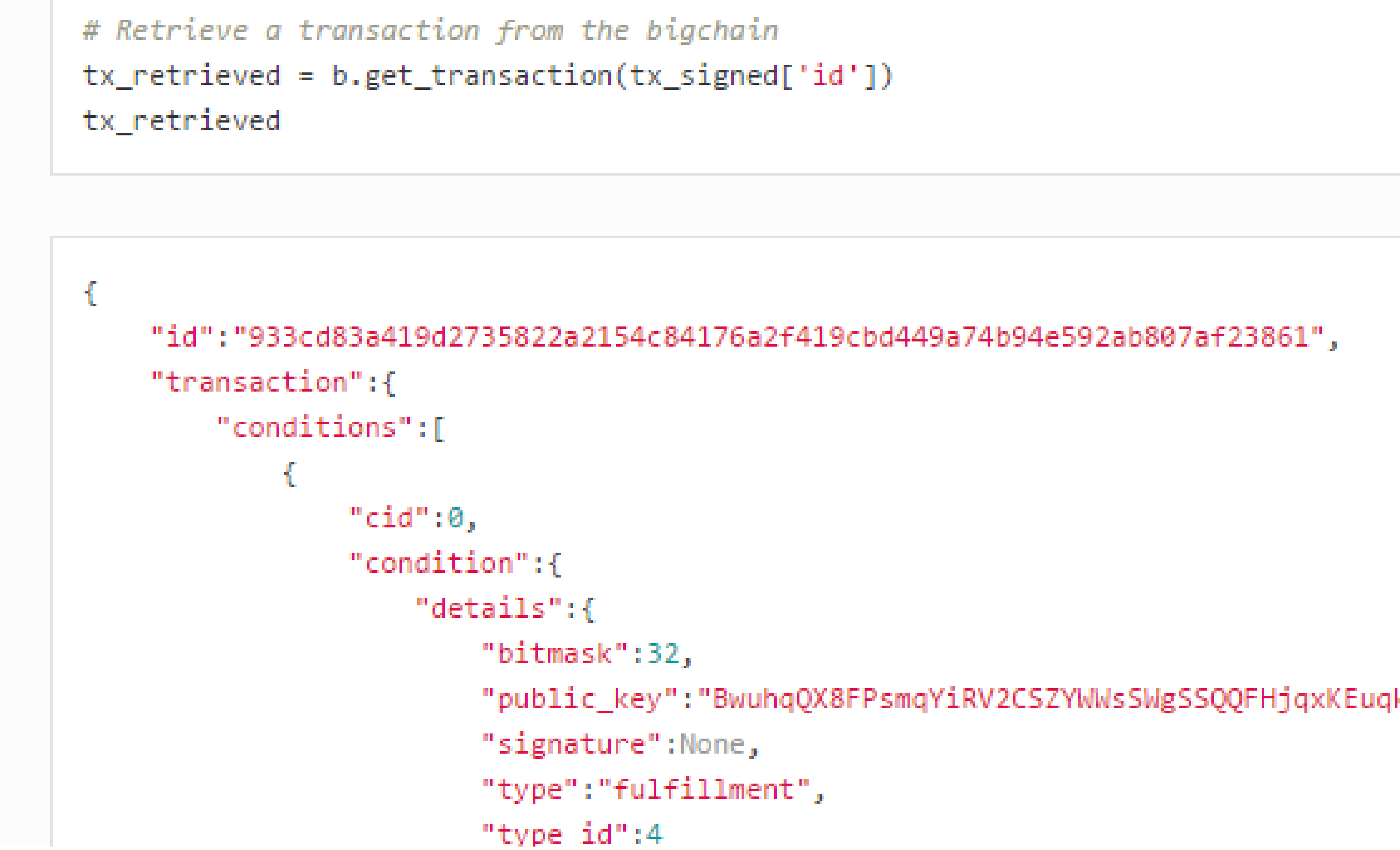
APPENDIX: USAGE



5.2. Create a Digital Asset

```
from bigchaindb import crypto

# Create a test user
testuser1_priv, testuser1_pub = crypto.generate_key_pair()

# Define a digital asset data payload
digital_asset_payload = {'msg': 'Hello BigchainDB!'}

# A create transaction uses the operation `CREATE` and has no inputs
tx = b.create_transaction(b.me, testuser1_pub, None, 'CREATE', payload=digital_

# All transactions need to be signed by the user creating the transaction
tx_signed = b.sign_transaction(tx, b.me_private)

# Write the transaction to the bigchain.
# The transaction will be stored in a backlog where it will be validated,
# included in a block, and written to the bigchain
b.write_transaction(tx_signed)
```

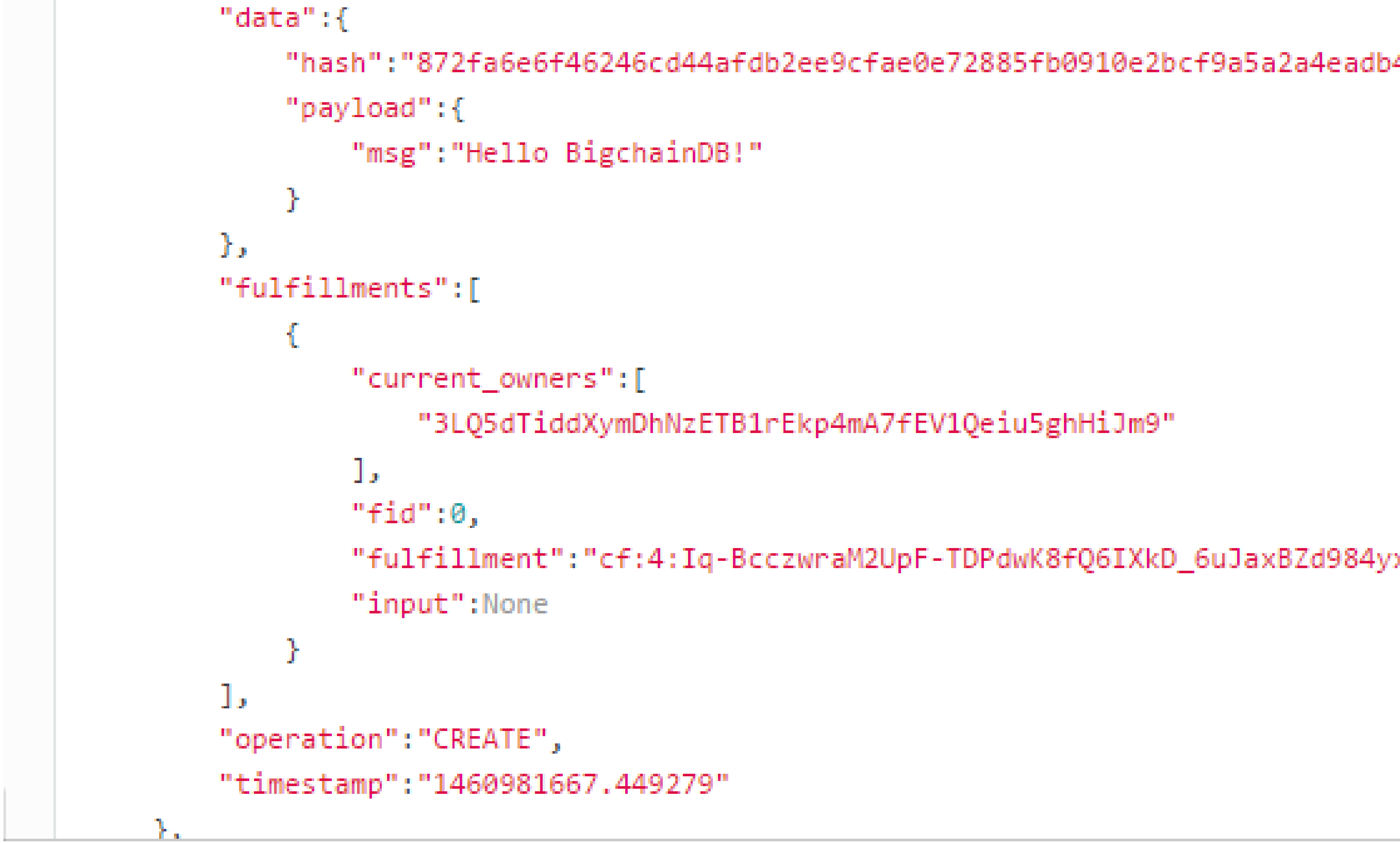
5.3. Read the Creation Transaction from the DB



```
# Retrieve a transaction from the bigchain
tx_retrieved = b.get_transaction(tx_signed['id'])
tx_retrieved
```

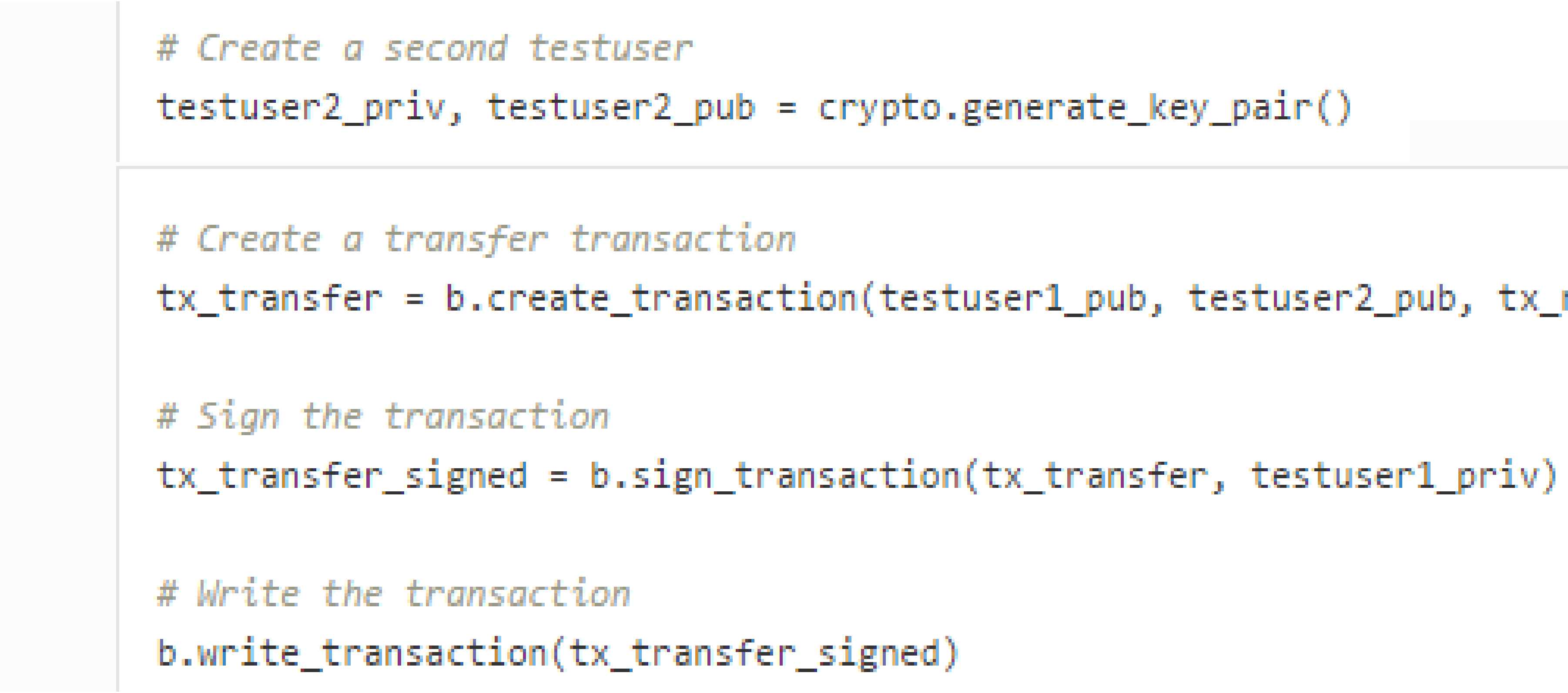
```
{
    "id":"933cd83a419d2735822a2154c84176a2f419cbd449a74b94e592ab807af23861",
    "transaction":{
        "conditions":[
            {
                "cid":0,
                "condition":{
                    "details":{
                        "bitmask":32,
                        "public_key":"BwuhqQX8FPsmqYiRV2CSZYWWsSWgSSQQFHjqxKEuq
                        "signature":None,
                        "type":"fulfillment",
                        "type_id":4
```

5.3. Read the Creation Transaction from the DB



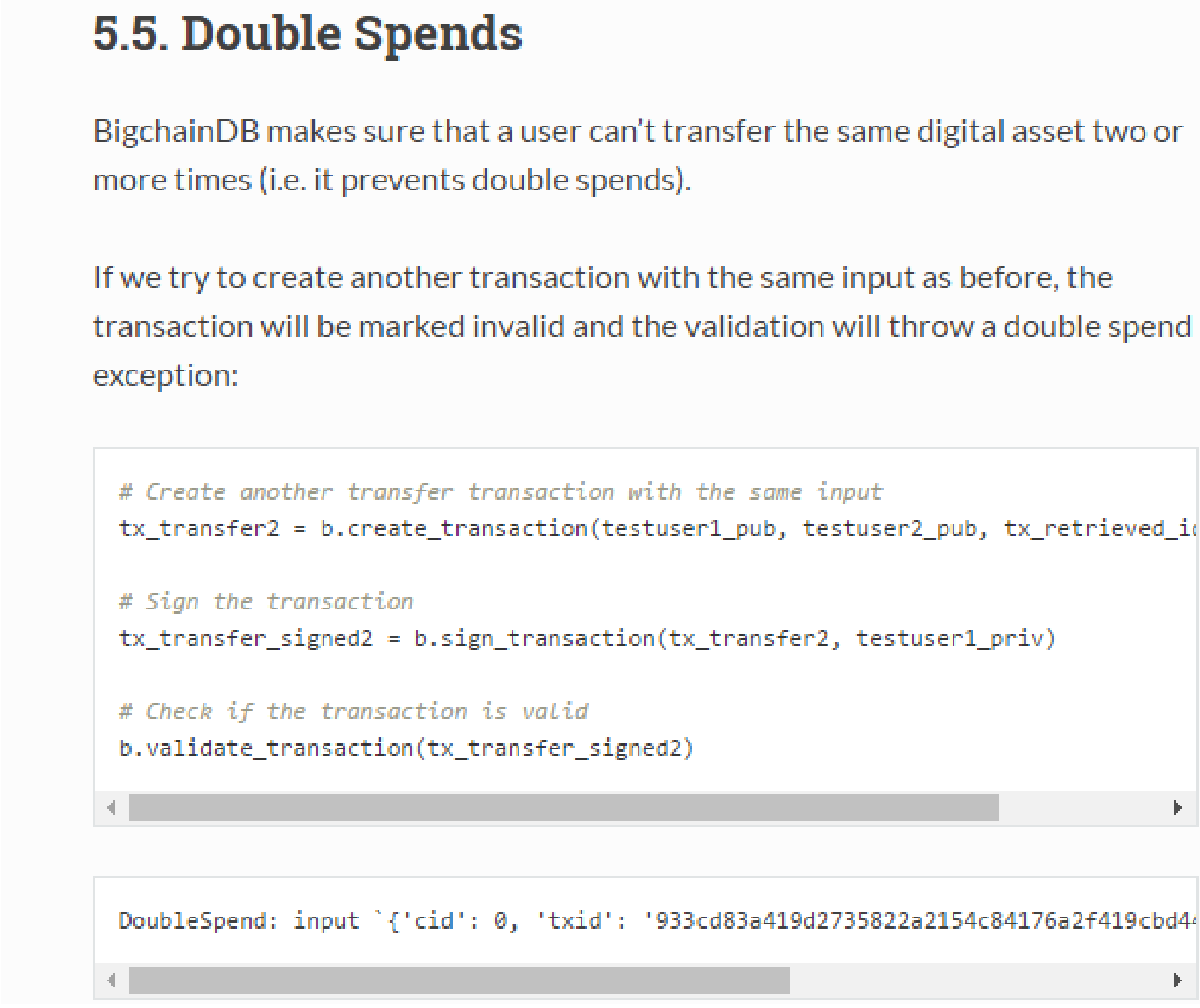
```
    "data":{
        "hash":"872fa6e6f46246cd44afdb2ee9cfae0e72885fb0910e2bcf9a5a2a4eadb
        "payload":{
            "msg":"Hello BigchainDB!"
        }
    },
    "fulfillments":[
        {
            "current_owners":[
                "3LQ5dTiddXymDhNzETB1rEkp4mA7fEV1Qeiu5ghHiJm9"
            ],
            "fid":0,
            "fulfillment":"cf:4:Iq-BcczwraM2UpF-TDPdwK8fQ6IXkD_6uJaxBZd984y
            "input":None
        }
    ],
    "operation":"CREATE",
    "timestamp":"1460981667.449279"
},
```

5.4. Transfer the Digital Asset



```
# Create a second testuser
testuser2_priv, testuser2_pub = crypto.generate_key_pair()
```

```
# Create a transfer transaction
tx_transfer = b.create_transaction(testuser1_pub, testuser2_pub, tx_

# Sign the transaction
tx_transfer_signed = b.sign_transaction(tx_transfer, testuser1_priv)

# Write the transaction
b.write_transaction(tx_transfer_signed)
```



5.5. Double Spends

BigchainDB makes sure that a user can't transfer the same digital asset two or more times (i.e. it prevents double spends).

If we try to create another transaction with the same input as before, the transaction will be marked invalid and the validation will throw a double spend exception:

```
# Create another transfer transaction with the same input
tx_transfer2 = b.create_transaction(testuser1_pub, testuser2_pub, tx_retrieved_i

# Sign the transaction
tx_transfer_signed2 = b.sign_transaction(tx_transfer2, testuser1_priv)

# Check if the transaction is valid
b.validate_transaction(tx_transfer_signed2)
```

```
DoubleSpend: input `{'cid': 0, 'txid': '933cd83a419d2735822a2154c84176a2f419cbd4
```

	Traditional blockchains	Big Data	BIGCHAIN DB
Decentralized	☑		☑
Immutable	☑		☑
Assets	☑		☑
Scale: Throughput, Capacity, Latency		☑	☑
Query Capabilities		☑	☑