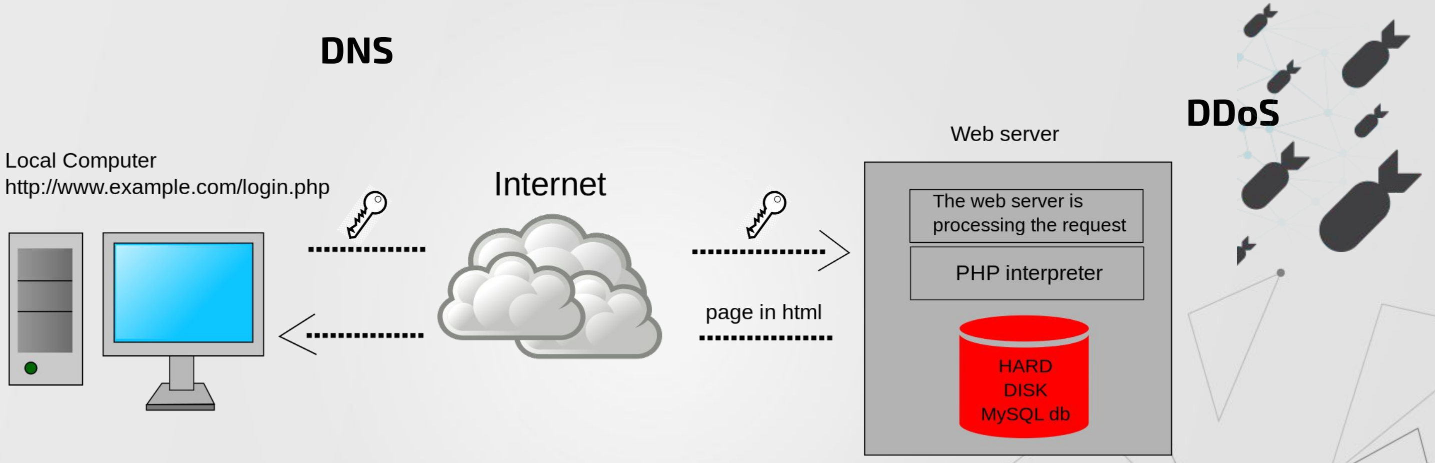
Decentralized Applications: Blockstack

Muneeb Ali   Ryan Shea   Jude Nelson   Michael J. Freedman

Traditional Internet Services

DNS

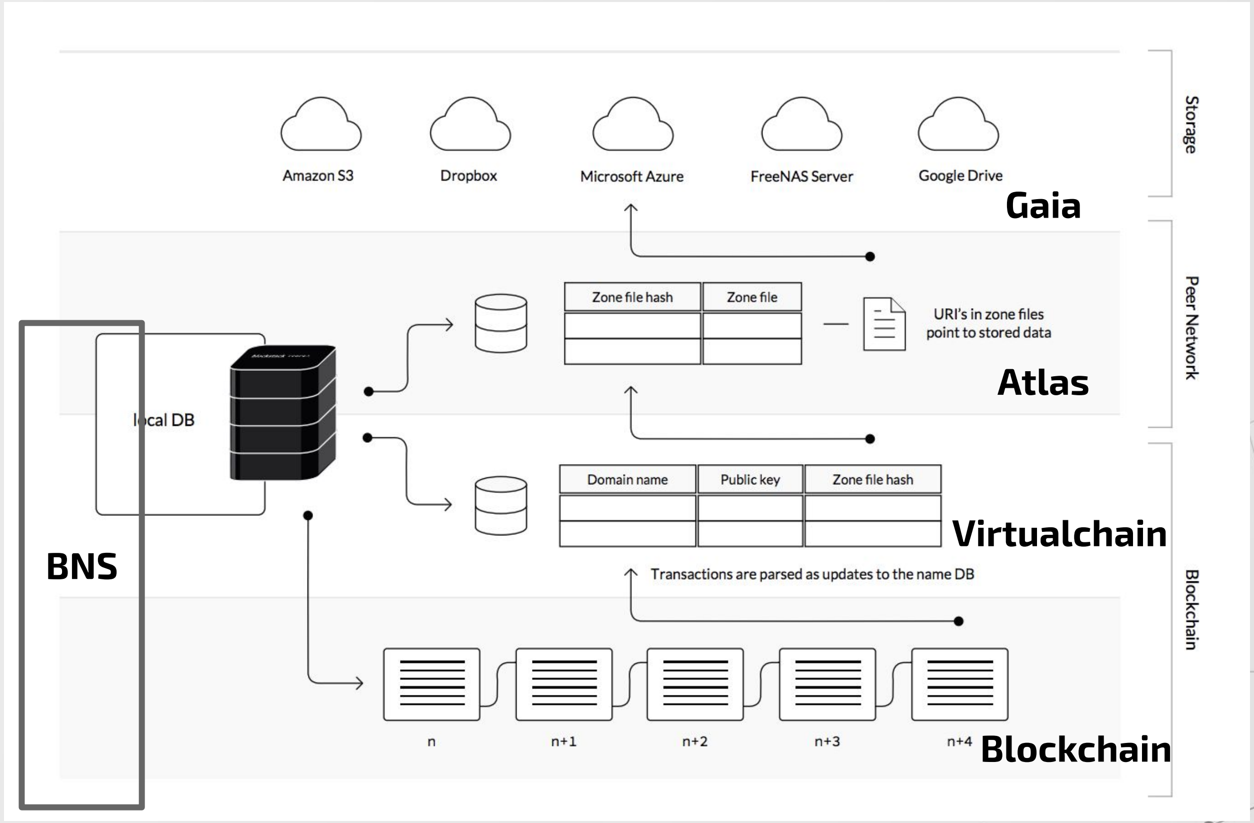
DDoS



Public-key Certificate

Untrusted

Overview of Blockstack



TCP/UDP, IP, Data Link, Physical

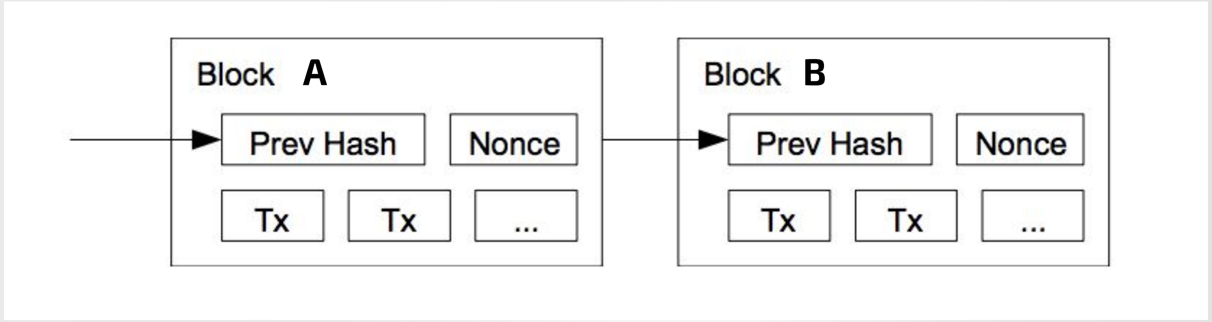
TABLE OF CONTENTS

01

Virtualchain

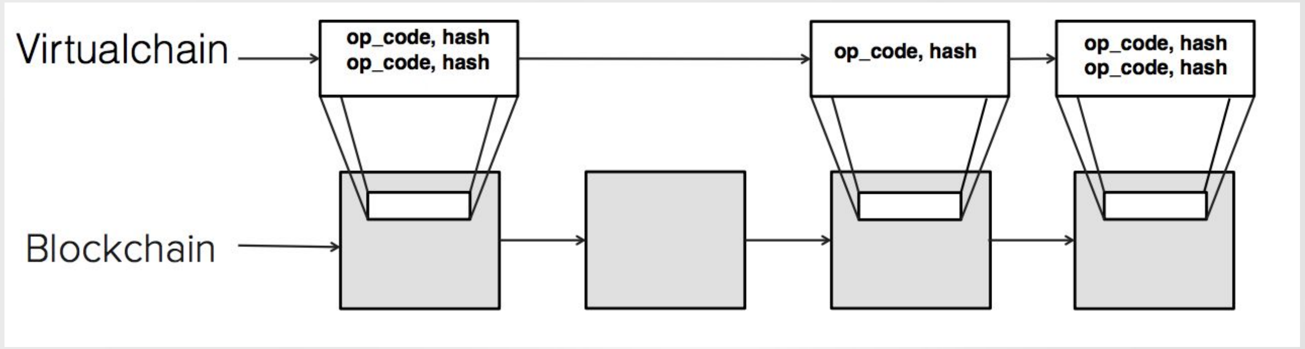
Decentralized Consensus based on Blockchain

*Blockchain

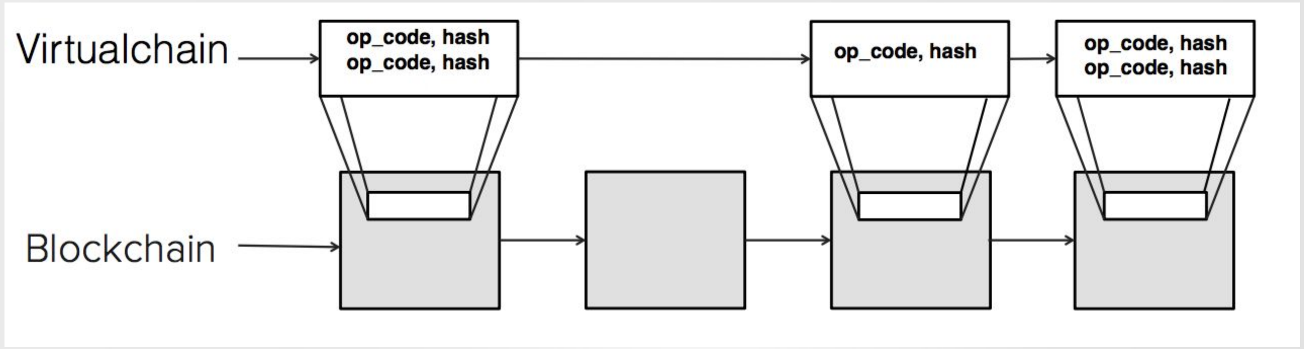


- B_n = {Hash(B_{n-1}), Nonce, Tx_1, Tx_2, …}
- Hard to change: each block contains the hash of its previous block.
- Proof-of-Work: spend CPU resources to get the nonce to add a block.
- Incentive + transaction fee: each new block has a reward and also transaction fees.
- Decentralized consensus; totally-ordered transaction log.
- Bitcoin: ~ 7TPS, ~ 10 min per block; 1 hour for 6 confirmation.

Virtualchain

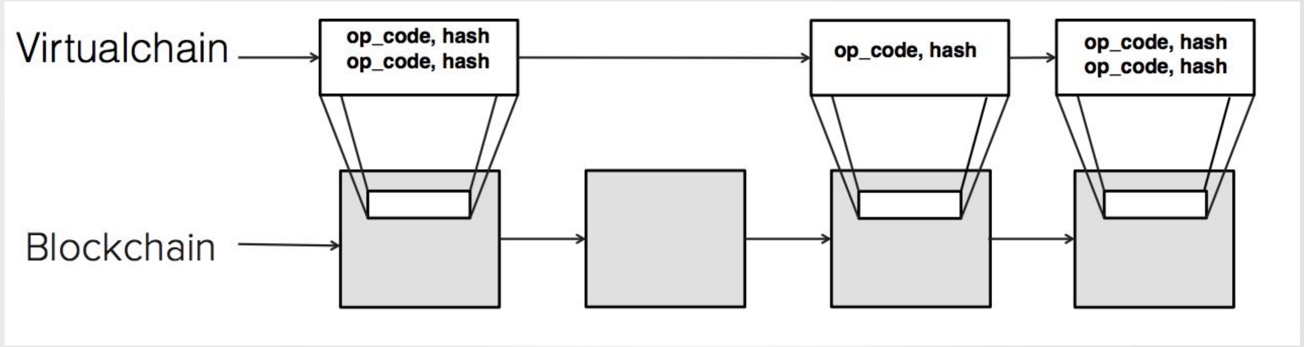


Virtualchain

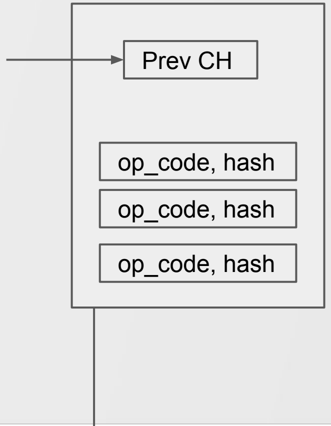


- Transaction ~ State Transition; op_code: OP_RETURN

Virtualchain



Virtualchain Block

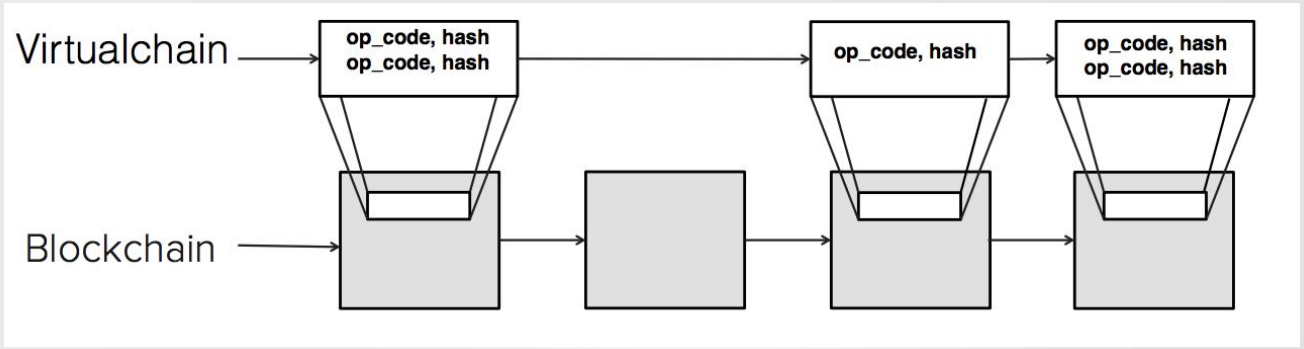


- Transaction ~ State Transition; op_code: OP_RETURN
- Consensus hash: filter invalid state transitions.

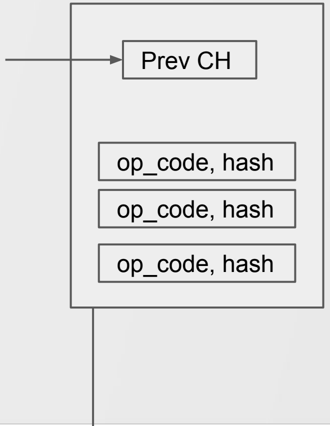
$$V_n = Merkle(tx \in b_n)$$

$$CH(n) = Hash(V_n + P_n)$$

Virtualchain



Virtualchain Block



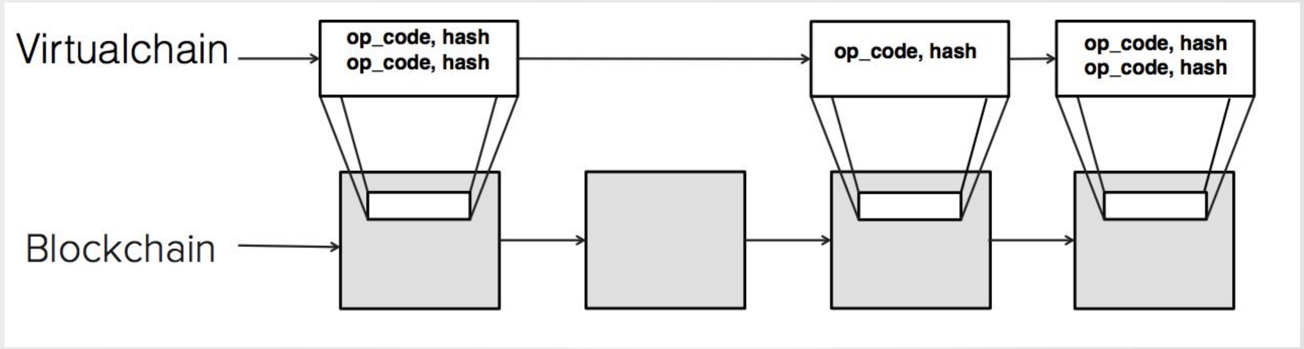
- Transaction ~ State Transition; op_code: OP_RETURN
- Consensus hash: filter invalid state transitions.

$$V_n = Merkle(tx \in b_n)$$

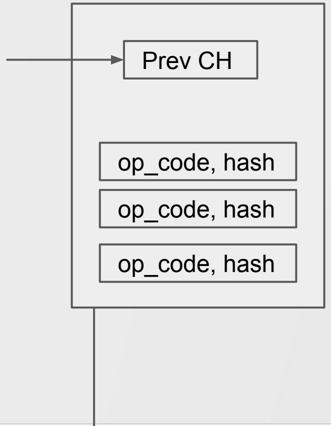
$$CH(n) = Hash(V_n + P_n)$$

- Application nodes replay their logs at each block to reach application level consensus.
- Decentralized and totally-ordered state transition log → construct state machines.

Virtualchain

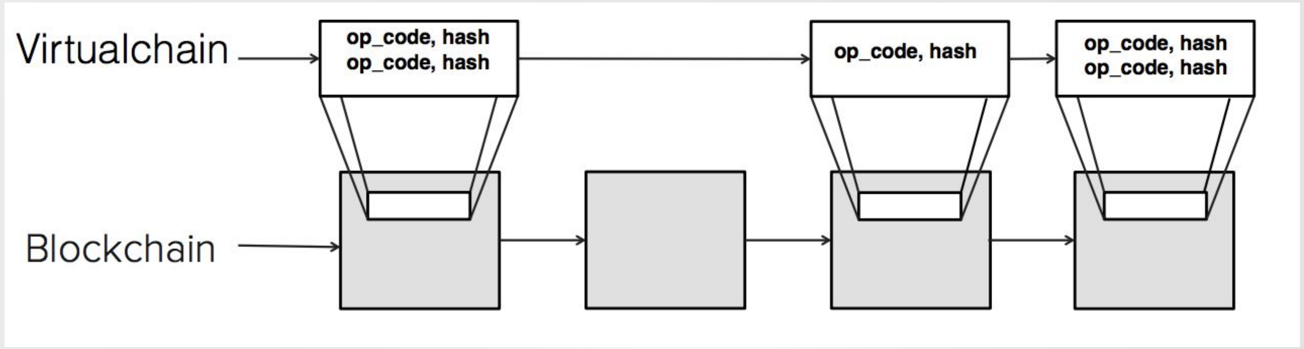


Virtualchain Block

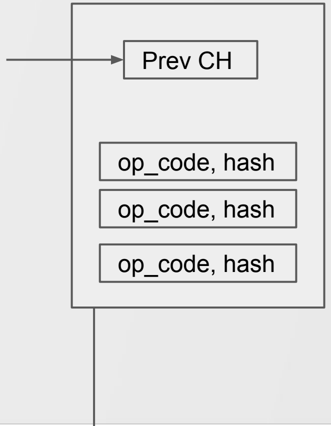


- Transaction ~ State Transition; op_code: OP_RETURN
- Consensus hash: filter invalid state transitions.
  - $$V_n = Merkle(tx \in b_n)$$
    $$CH(n) = Hash(V_n + P_n)$$
- Application nodes replay their logs at each block to reach application level consensus.
- Decentralized and totally-ordered state transition log → construct state machines.
- Join-at-most-once of fork*-consistency.

Virtualchain



Virtualchain Block



- Transaction ~ State Transition; op_code: OP_RETURN
- Consensus hash: filter invalid state transitions.

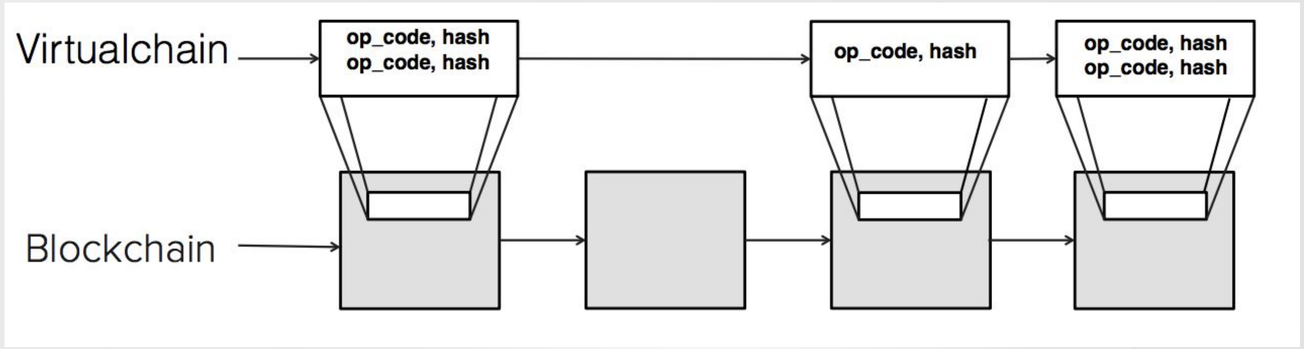
$$V_n = Merkle(tx \in b_n)$$

$$CH(n) = Hash(V_n + P_n)$$

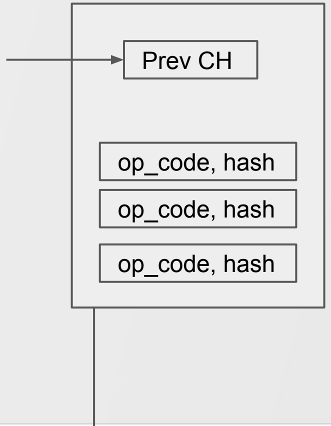
fast queries

- Application nodes replay their logs at each block to reach application level consensus.
- Decentralized and totally-ordered state transition log → construct state machines.
- Join-at-most-once of fork*-consistency.

Virtualchain



Virtualchain Block



- Transaction ~ State Transition; op_code: OP_RETURN
- Consensus hash: filter invalid state transitions.

$$V_n = Merkle(tx \in b_n)$$

$$CH(n) = Hash(V_n + P_n)$$

fast queries

- Application nodes replay their logs at each block to reach application level consensus.
- Decentralized and totally-ordered state transition log → construct state machines.
- Join-at-most-once of fork*-consistency.
- Why?

02

Atlas Network

Decentralized key-value Storage

Atlas Network

Decentralized peer-to-peer network for content discovery. (key-value storage)

Atlas Network

Decentralized peer-to-peer network for content discovery. (key-value storage)

- Alternative: Distributed Hash Table
  - structured/unstructured,
  - routing tables with a subset of peers.

Atlas Network

Decentralized peer-to-peer network for content discovery. (key-value storage)

- Alternative: Distributed Hash Table
  - structured/unstructured,
  - routing tables with a subset of peers.
- Problem:
  - Sybil attacks: malicious users claims multiple identities to compromise the whole network.
  - Churn/partition: inconsistent states
  - Variable latency

Atlas Network

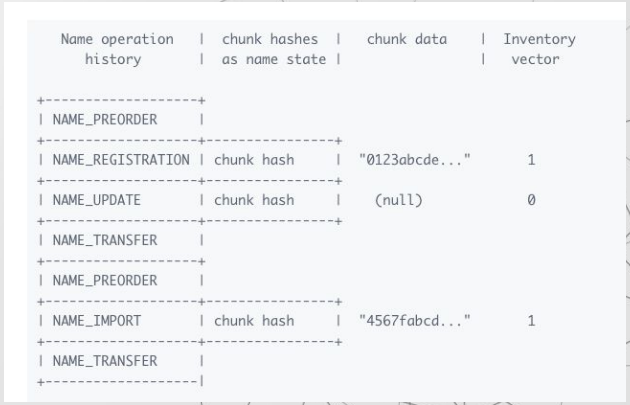
Decentralized peer-to-peer network for content discovery. (key-value storage)

- Alternative: Distributed Hash Table
  - structured/unstructured,
  - routing tables with a subset of peers.
- Problem:
  - Sybil attacks: malicious users claims multiple identities to compromise the whole network.
  - Churn/partition: inconsistent states
  - Variable latency
- Assumption:
  - value size is fairly small. (Each zone file is less than 4KB, 25M zone files → 100G)
  - a full(trustworthy) index of data is available to the network.

Atlas Network

Decentralized peer-to-peer network for content discovery. (key-value storage)

- Alternative: Distributed Hash Table
  - structured/unstructured,
  - routing tables with a subset of peers.
- Problem:
  - Sybil attacks: malicious users claims multiple identities to compromise the whole network.
  - Churn/partition: inconsistent states
  - Variable latency
- Assumption:
  - value size is fairly small. (Each zone file is less than 4KB, 25M zone files → 100G)
  - a full(trustworthy) index of data is available to the network.
- Mechanism:
  - 100% state replicas, unstructured overlay network.
  - K-regular random graph: select neighbors, fetch missing pairs, propagate new pairs.

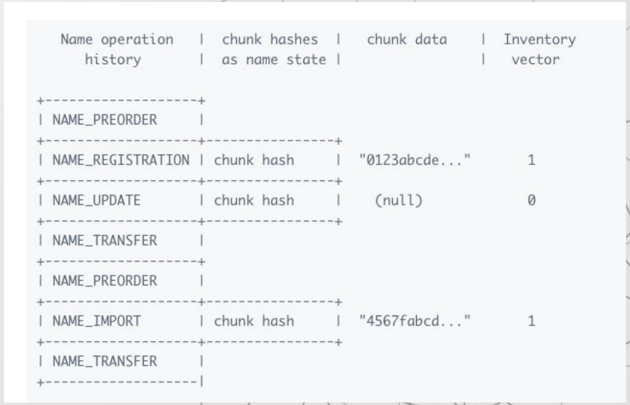


```
  Name operation    |  chunk hashes  |   chunk data    |  Inventory
     history        |  as name state |                 |   vector

+-------------------+
| NAME_PREORDER     |
+-------------------+----------------+
| NAME_REGISTRATION | chunk hash     |  "0123abcde..."       1
+-------------------+----------------+
| NAME_UPDATE       | chunk hash     |    (null)             0
+-------------------+----------------+
| NAME_TRANSFER     |
+-------------------+
| NAME_PREORDER     |
+-------------------+----------------+
| NAME_IMPORT       | chunk hash     |  "4567fabcd..."       1
+-------------------+----------------+
| NAME_TRANSFER     |
+-------------------|
```

Atlas Network

Decentralized peer-to-peer network for content discovery. (key-value storage)

- Alternative: Distributed Hash Table
  - structured/unstructured,
  - routing tables with a subset of peers.
- Problem:
  - Sybil attacks: malicious users claims multiple identities to compromise the whole network.
  - Churn/partition: inconsistent states
  - Variable latency
- Assumption:
  - value size is fairly small. (Each zone file is less than 4KB, 25M zone files → 100G)
  - a full(trustworthy) index of data is available to the network.
- Mechanism:
  - 100% state replicas, unstructured overlay network.
  - K-regular random graph: select neighbors, fetch missing pairs, propagate new pairs.
- How could the node accept a pair from its peers? (Note: peers are not trusted)

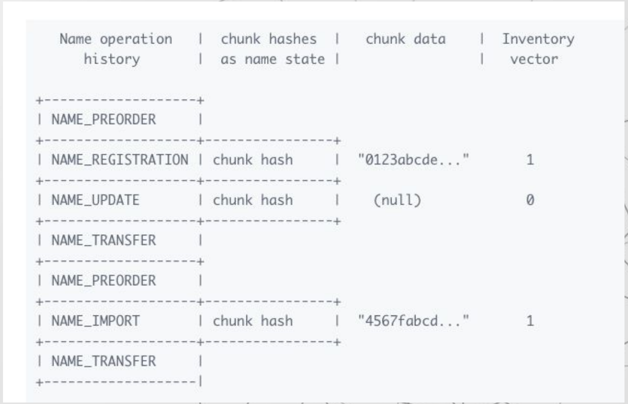


```
  Name operation    |  chunk hashes  |   chunk data    |  Inventory
     history        |  as name state |                 |   vector

+-------------------+
| NAME_PREORDER     |
+-------------------+----------------+
| NAME_REGISTRATION | chunk hash     |  "0123abcde..."       1
+-------------------+----------------+
| NAME_UPDATE       | chunk hash     |    (null)             0
+-------------------+----------------+
| NAME_TRANSFER     |
+-------------------+
| NAME_PREORDER     |
+-------------------+----------------+
| NAME_IMPORT       | chunk hash     |  "4567fabcd..."       1
+-------------------+----------------+
| NAME_TRANSFER     |
+-------------------|
```

Atlas Network

Decentralized peer-to-peer network for content discovery. (key-value storage)

- Alternative: Distributed Hash Table
  - structured/unstructured,
  - routing tables with a subset of peers.
- Problem:
  - Sybil attacks: malicious users claims multiple identities to compromise the whole network.
  - Churn/partition: inconsistent states
  - Variable latency
- Assumption:
  - value size is fairly small. (Each zone file is less than 4KB, 25M zone files → 100G)
  - a full(trustworthy) index of data is available to the network.
- Mechanism:
  - 100% state replicas, unstructured overlay network.
  - K-regular random graph: select neighbors, fetch missing pairs, propagate new pairs.
- How could the node accept a pair from its peers? (Note: peers are not trusted)
- How to recover? How to bootstrap a new node?



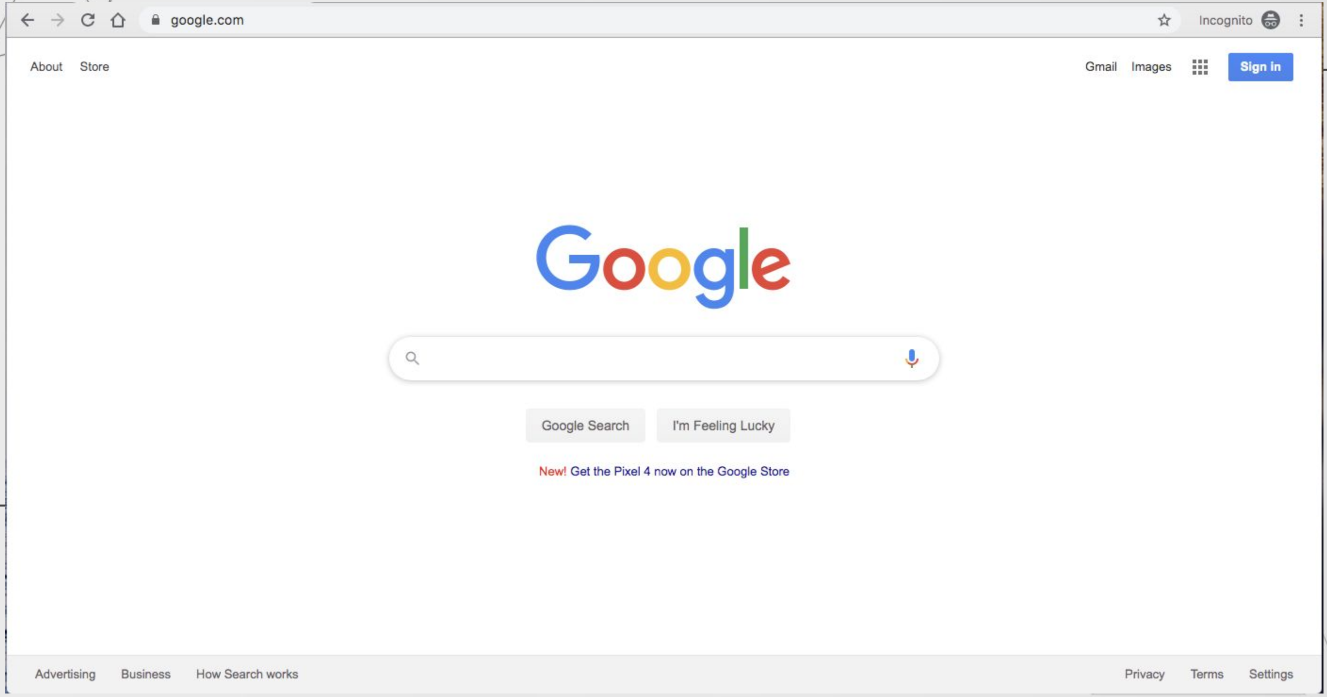
```
  Name operation    |  chunk hashes  |   chunk data     |  Inventory
     history        |  as name state |                  |   vector

+-------------------+
| NAME_PREORDER     |
+-------------------+----------------+
| NAME_REGISTRATION | chunk hash     |  "0123abcde..."       1
+-------------------+----------------+
| NAME_UPDATE       | chunk hash     |    (null)             0
+-------------------+----------------+
| NAME_TRANSFER     |
+-------------------+
| NAME_PREORDER     |
+-------------------+----------------+
| NAME_IMPORT       | chunk hash     |  "4567fabcd..."       1
+-------------------+----------------+
| NAME_TRANSFER     |
+-------------------|
```

03

BNS

Blockchain Naming System

What happens when you enter a URL in a browser?

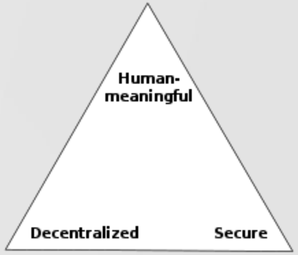


$$\begin{tikzcd}\textbf{DNS} \arrow[d] \\ \textbf{BNS}\end{tikzcd}$$

BNS

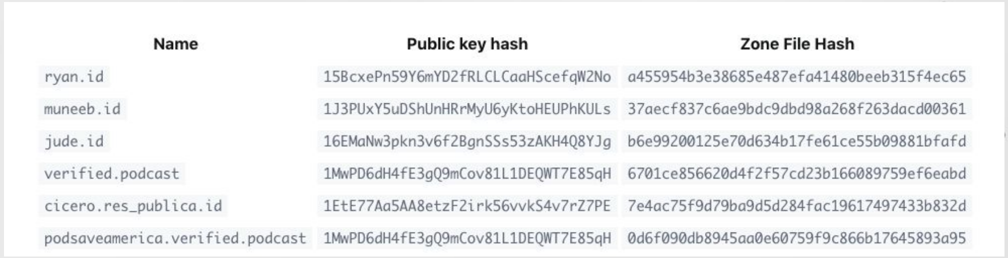
Bind human-readable names to discovery data without central points of failure and control.

- Name
  - unique: "I am Alice", "I am Alice, too"
  - human-readable: 1A1zP1eP5QGefi2DMPTfTL5SLmv7DivfNa
  - no centralized control: e.g. Domain Name Server.
  - → Zooko's Triangle
- Use **virtualchain** to maintain the mapping between human-readable names and pointers to **Atlas Network** that maps to discovery information.
- Zone file: stored in Atlas layer, contain routing information
- Name in virtualchain



Zone file hash
Zone file

$ORIGIN zhoutao_.id.blockstack
$TTL 3600
_http._tcp IN URI 10 1
"https://gaia.blockstack.org/hub/142cczgGrxDLhWZi
b3uJhDXA9Ndvkx5KwV/profile.json"

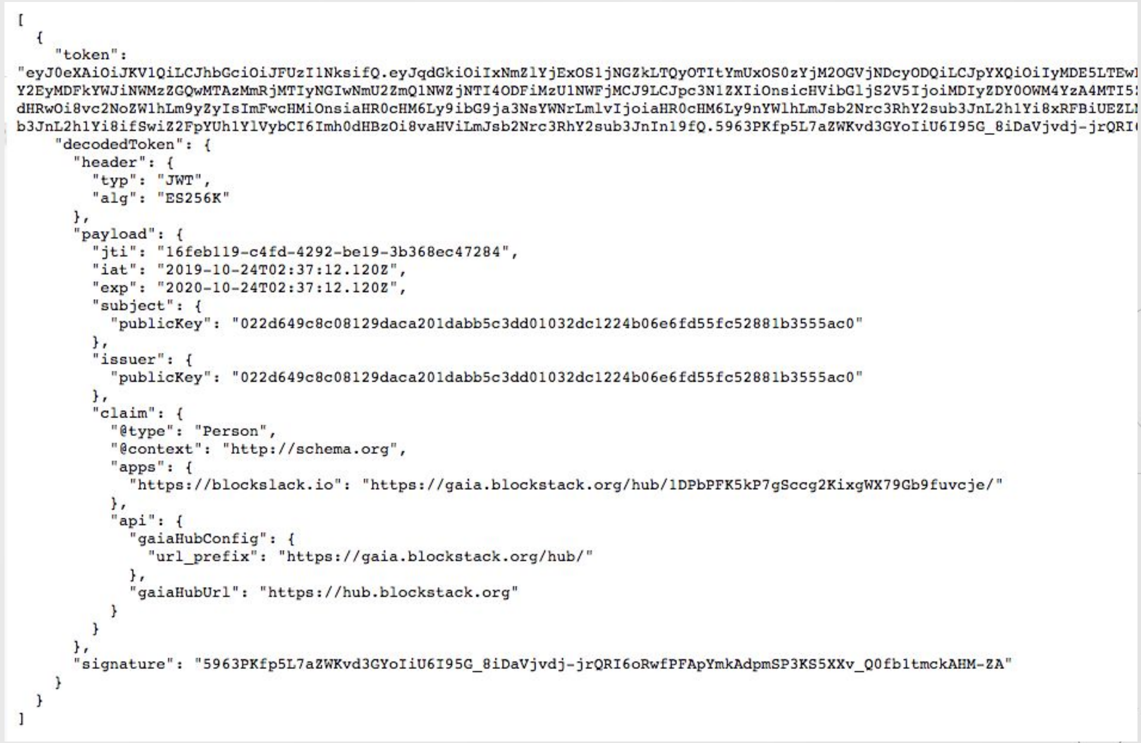


Name	Public key hash	Zone File Hash
ryan.id	15BcxePn59Y6mYD2fRLCLCaaHScefqW2No	a455954b3e38685e487efa41480beeb315f4ec65
muneeb.id	1J3PUxY5uDShUnHRrMyU6yKtoHEUPhKULs	37aecf837c6ae9bdc9dbd98a268f263dacd00361
jude.id	16EMaNw3pkn3v6f2BgnSSs53zAKH4Q8YJg	b6e99200125e70d634b17fe61ce55b09881bfafd
verified.podcast	1MwPD6dH4fE3gQ9mCov81L1DEQWT7E85qH	6701ce856620d4f2f57cd23b166089759ef6eabd
cicero.res_publica.id	1EtE77Aa5AA8etzF2irk56vvkS4v7rZ7PE	7e4ac75f9d79ba9d5d284fac19617497433b832d
podsaveamerica.verified.podcast	1MwPD6dH4fE3gQ9mCov81L1DEQWT7E85qH	0d6f090db8945aa0e60759f9c866b17645893a95

BNS

Bind human-readable names to discovery data without central points of failure and control.

- profile.json



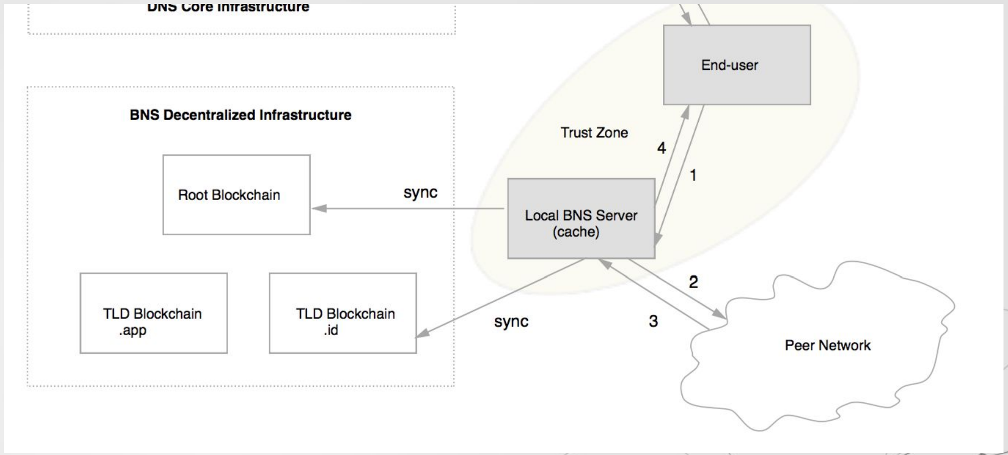
```
[
  {
    "token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJFUzI1NksifQ.eyJqdGkiOiIxNmZlYjExOS1jNGZkLTQyOTItYmUxOS0zYjM2OGVjNDcyODQiLCJpYXQiOiIyMDE5LTEw
Y2EyMDFkYWJiNWMzZGQwMTAzMmRjMTIyNGIwNmU2ZmQ1NWZjNTI4ODFiMzU1NWFjMCJ9LCJpc3N1ZXIiOnsicHVibGljS2V5IjoiMDIyZDY0OWM4YzA4MTI5
dHRwOi8vc2NoZW1hLm9yZyIsImFwcHMiOnsiaHR0cHM6Ly9ibG9ja3NsYWNrLmlvIjoiaHR0cHM6Ly9nYWlhLmJsb2Nrc3RhY2sub3JnL2h1Yi8xRFBiUEZL
b3JnL2h1Yi8ifSwiZ2FpYUh1YlVybCI6Imh0dHBzOi8vaHViLmJsb2Nrc3RhY2sub3JnIn19fQ.5963PKfp5L7aZWKvd3GYoIiU6I95G_8iDaVjvdj-jrQRI
    "decodedToken": {
      "header": {
        "typ": "JWT",
        "alg": "ES256K"
      },
      "payload": {
        "jti": "16feb119-c4fd-4292-be19-3b368ec47284",
        "iat": "2019-10-24T02:37:12.120Z",
        "exp": "2020-10-24T02:37:12.120Z",
        "subject": {
          "publicKey": "022d649c8c08129daca201dabb5c3dd01032dc1224b06e6fd55fc52881b3555ac0"
        },
        "issuer": {
          "publicKey": "022d649c8c08129daca201dabb5c3dd01032dc1224b06e6fd55fc52881b3555ac0"
        },
        "claim": {
          "@type": "Person",
          "@context": "http://schema.org",
          "apps": {
            "https://blockslack.io": "https://gaia.blockstack.org/hub/1DPbPFK5kP7gSccg2KixgWX79Gb9fuvcje/"
          },
          "api": {
            "gaiaHubConfig": {
              "url_prefix": "https://gaia.blockstack.org/hub/"
            },
            "gaiaHubUrl": "https://hub.blockstack.org"
          }
        }
      },
      "signature": "5963PKfp5L7aZWKvd3GYoIiU6I95G_8iDaVjvdj-jrQRI6oRwfPFApYmkAdpmSP3KS5XXv_Q0fb1tmckAHM-ZA"
    }
  }
]
```

BNS

Bind human-readable names to discovery data without central points of failure and control.

- Namespace:
  - blockstack, id.blockstack, zhoutao_.id.blockstack
  - different virtualchain or blockchain
- Operations:
  - preorder
  - register
  - update
  - transfer
  - revoke
- Price Function
  - land grabs: stop people from registering a lot of unused namespace or names.
  - name length, non-alphabetic characters, etc.
- Public key in BNS

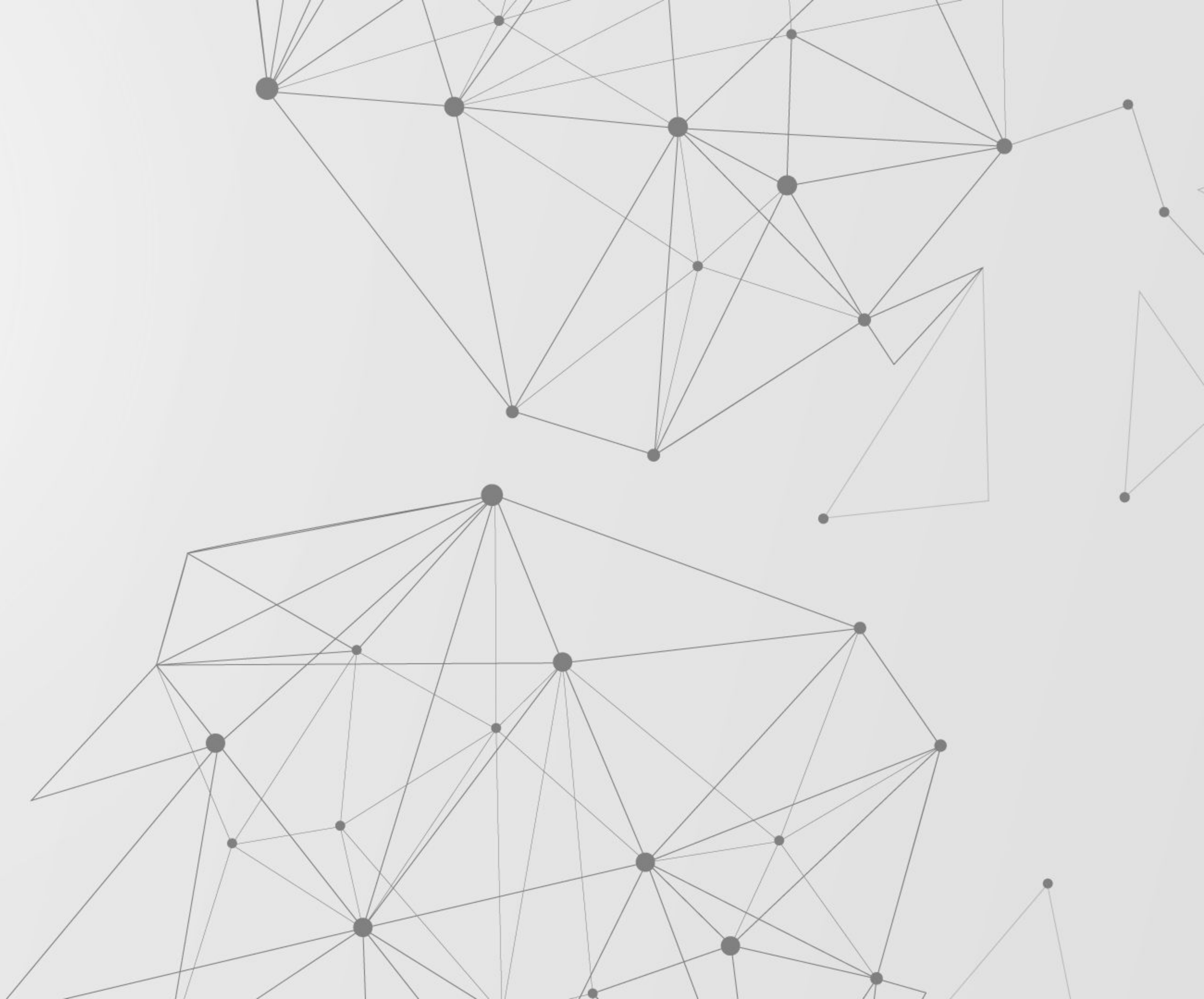
two phase commit



04

Gaia

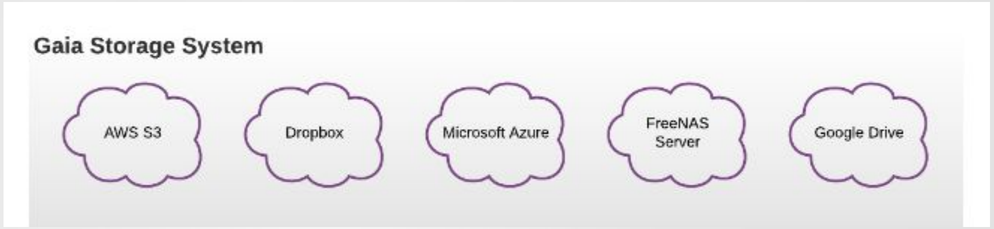
Decentralized Storage System



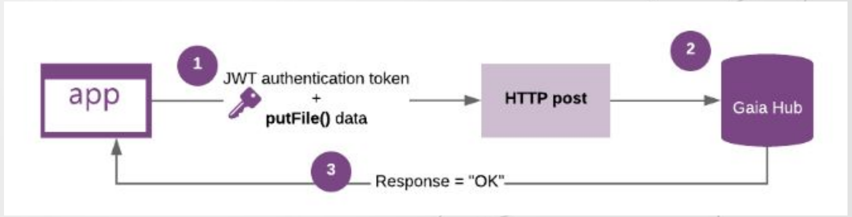
Gaia

Decentralized storage system that gives users the control over their own data.

- Traditional storage: central server, remote cloud → out of users' trust zone.
- Reuse the existing cloud provider and infrastructure but treat them as dumb driver and users decide where to store the data.
- Gaia Hub:



- Write to Gaia: sign / encryption



- Read from Gaia
  - fetch the zonefile for alice.id.
  - Read her profile URL from her zonefile.
  - Fetch Alice's profile.
  - Verify that the profile is signed by alice.id's key
  - Read the gaiaHubUrl (e.g. https://gaia.alice.org/) out of the profile
  - Fetch the file from https://gaia.alice.org/data.txt.

Gaia

Decentralized storage system that gives users the control over their own data.

- Performance
  - Storage overhead: 5% from encryption
  - CPU overhead: signing/encryption for write, decryption for read
- Scalability:
  - storage layer is good
  - Atlas is scalable
  - Bottleneck is virtualchain
    - pack multiple application transaction into a single blockchain transaction
- How to do sharing?

Is the decentralized world realizable?

THANKS

Any questions?