# Python Applications with Blockstack

Jude Nelson
Blockstack

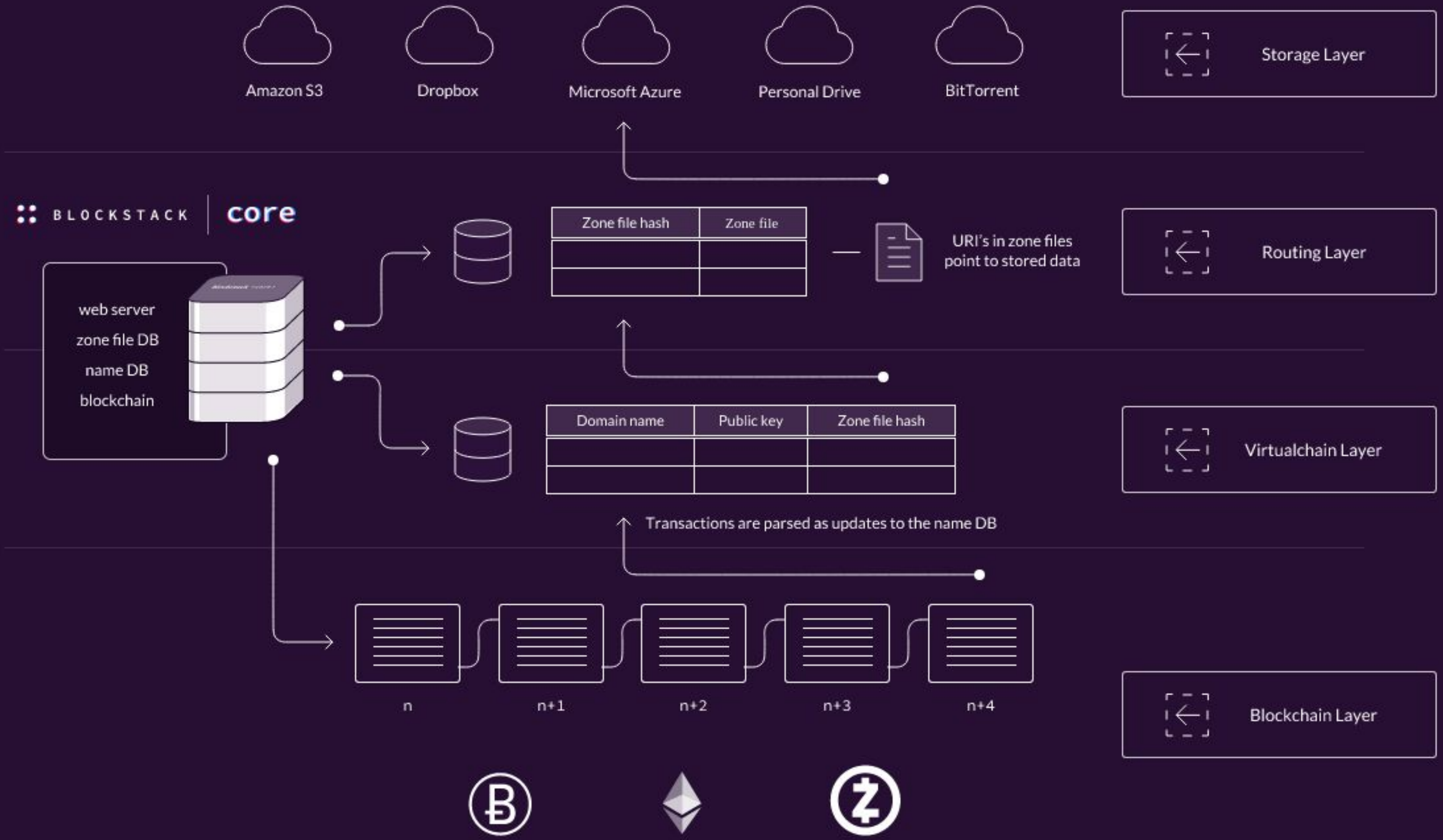
## Decentralization Goals

### Users

- Own their data
- Control where it’s stored
- Control who can access it

### Developers

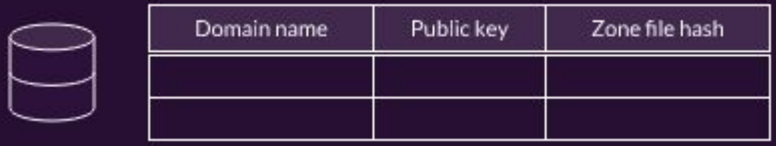
- No passwords
- No data hosting
- No VMs


Amazon S3
Dropbox
Microsoft Azure
Personal Drive
BitTorrent
Storage Layer
BLOCKSTACK core
web server
zone file DB
name DB
blockchain
Zone file hash
Zone file
URI's in zone files point to stored data
Routing Layer
Domain name
Public key
Zone file hash
Virtualchain Layer
Transactions are parsed as updates to the name DB
n
n+1
n+2
n+3
n+4
Blockchain Layer

## Blockchain Layer

- Global, consistent input tape to a replicated state machine
- Peers validate and serialize inputs
- Anyone can append
- Tokens are the append rate-limiter

## Virtualchain Layer



| Domain name | Public key | Zone file hash |
|---|---|---|
| | | |
| | | |

- Embed app-specific inputs in blockchain
- Application validates, not blockchain
- Blockstack is one of many apps
- No PoW → *Fork*-consistent* RSM (!!)
- DCCL 2016

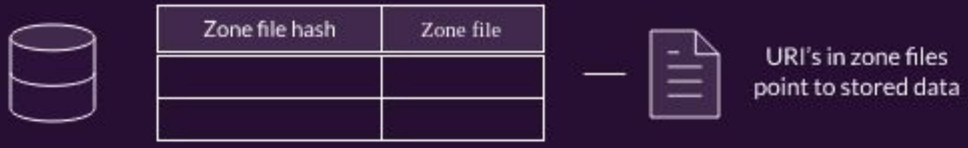
# Blockstack's Virtualchain

| Domain name | Public key | Zone file hash |
|---|---|---|
| | | |
| | | |

- Name registry database RSM
- DB of `(name, public key, 20-byte payload)`
- Payload = hash of a DNS zone file

Nodes that see the **same blockchain** and follow **the same validation rules** derive the **same DB**

## Routing Layer



- Zone file hash → Zone file → Off-chain data
- Atlas Protocol:  BitTorrent-like zone file replication

```
$ORIGIN judecn.id
$TTL 3600
pubkey TXT
"pubkey:data:04cabba0b5b9a871dbaa11c044066e281c5feb57243c7d2a452f06a0d708613a46ce
d59f9f806e601b3353931d1e4a98d7040127f31016311050bedc0d4f1f62ff"
_file URI 10 1 "file:///home/jude/.blockstack/storage-disk/mutable/judecn.id"
_https._tcp URI 10 1 "https://blockstack.s3.amazonaws.com/judecn.id"
_http._tcp URI 10 1 "http://node.blockstack.org:6264/RPC2#judecn.id"
_dht._udp URI 10 1 "dht+udp://fc4d9c1481a6349fe99f0e3dd7261d67b23dadc5"
```

## Storage Layer

Amazon S3 Dropbox Microsoft Azure Personal Drive BitTorrent

- **I/O does not touch the blockchain**
- Storage providers = dumb hard drives

**Write**

- Sign data
- Upload to zone file URLs

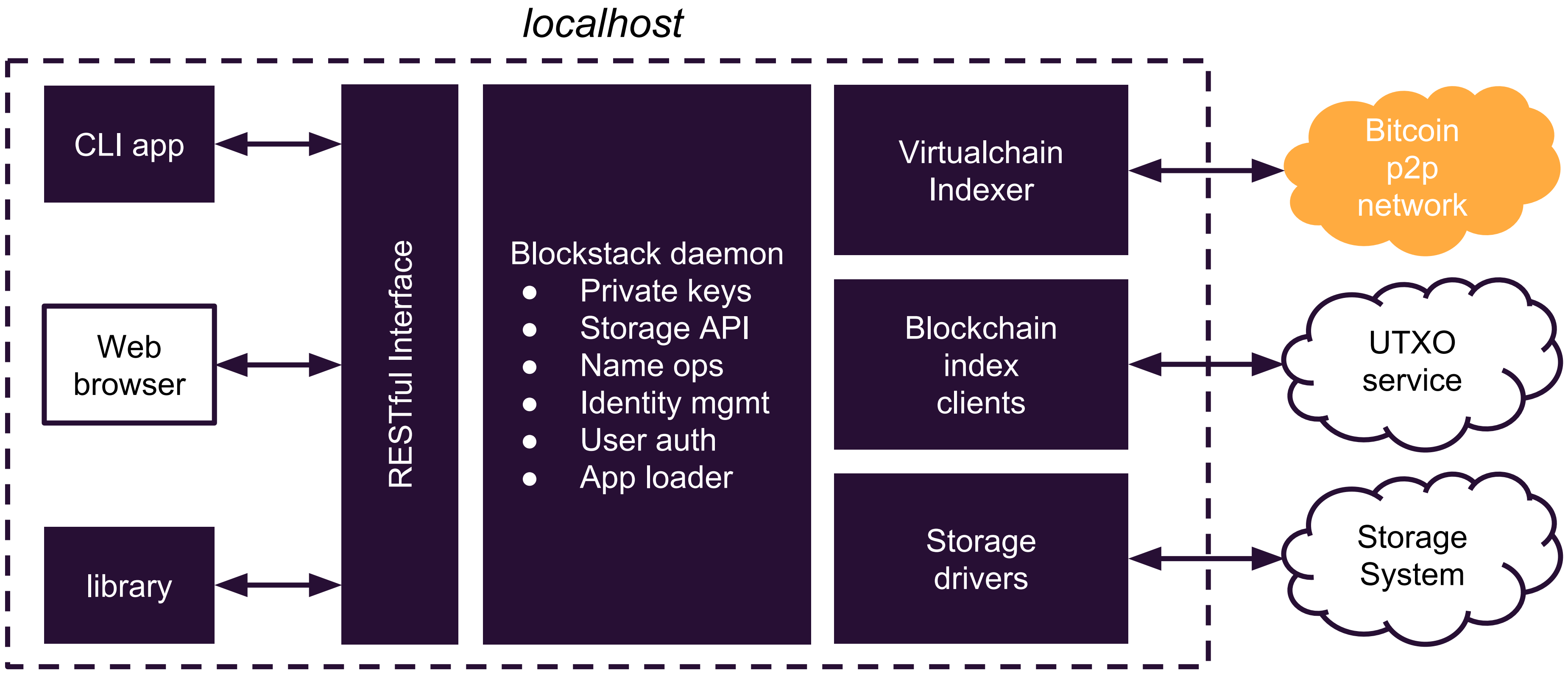
**Read**

- Get zone file
- Resolve URLs
- Verify signature
- Check fresh

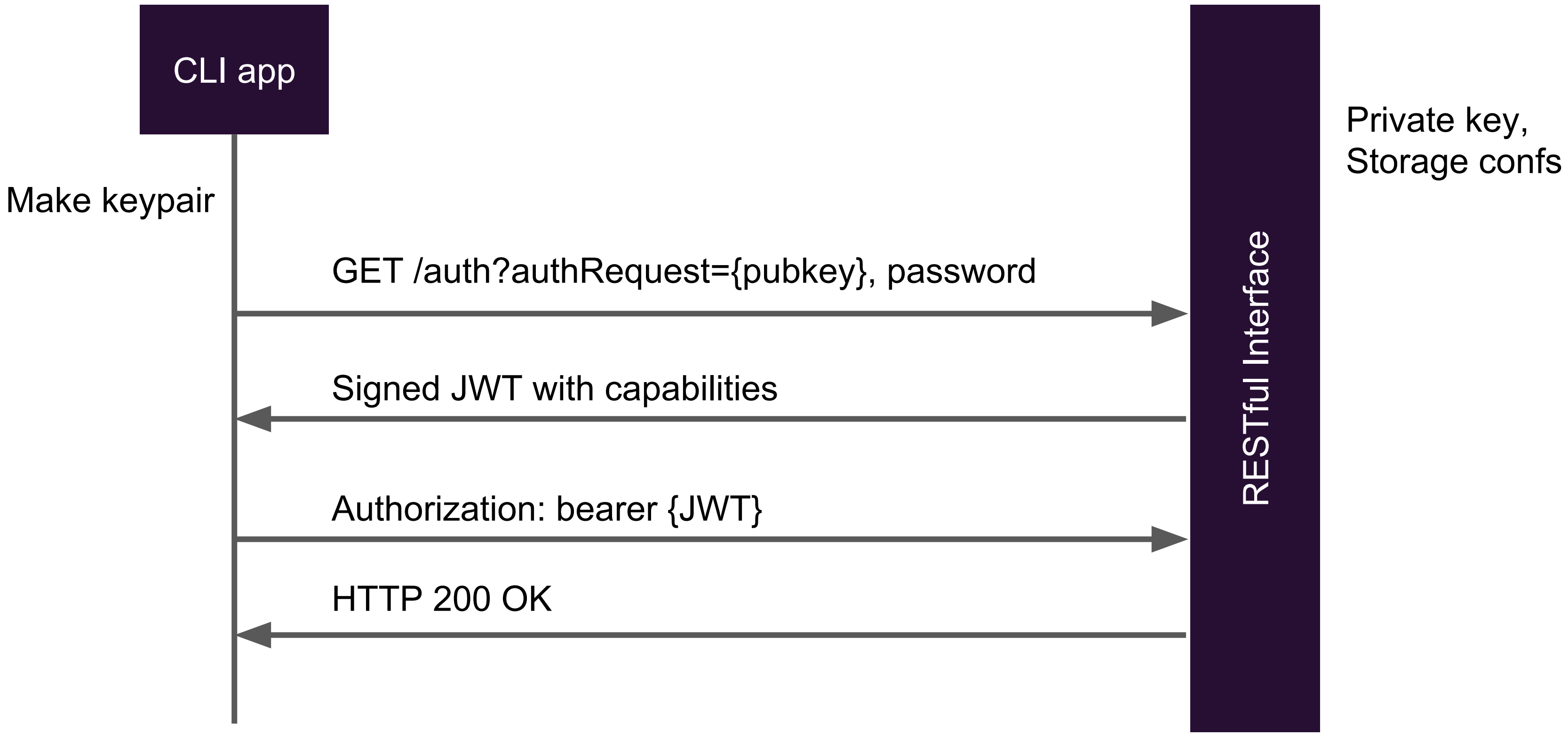
**Consistency**

- Get zone file
- Resolve URLs
- Verify signature
- Check fresh

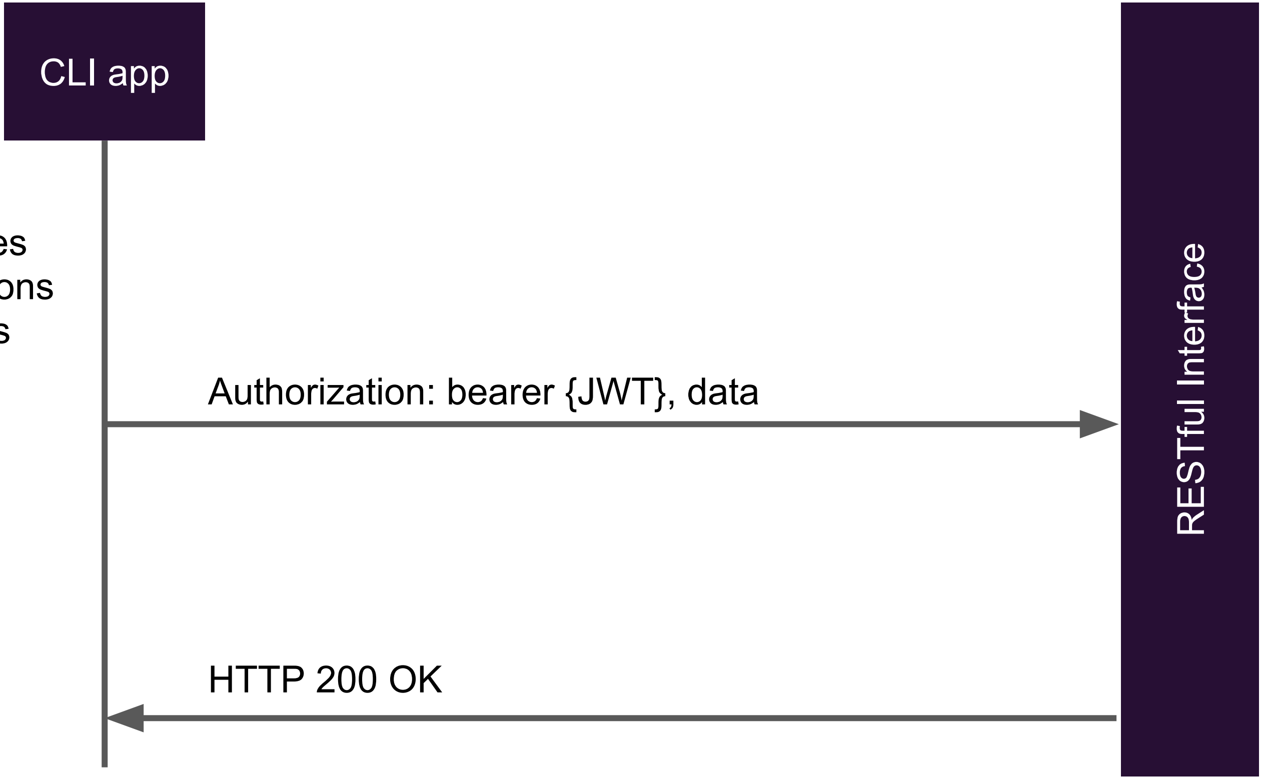
# Applications



# Storage Programming Model: Authentication


CLI app
RESTful Interface
Private key,
Storage confs
Make keypair
GET /auth?authRequest={pubkey}, password
Signed JWT with capabilities
Authorization: bearer {JWT}
HTTP 200 OK

# Storage Programming Model: Writes

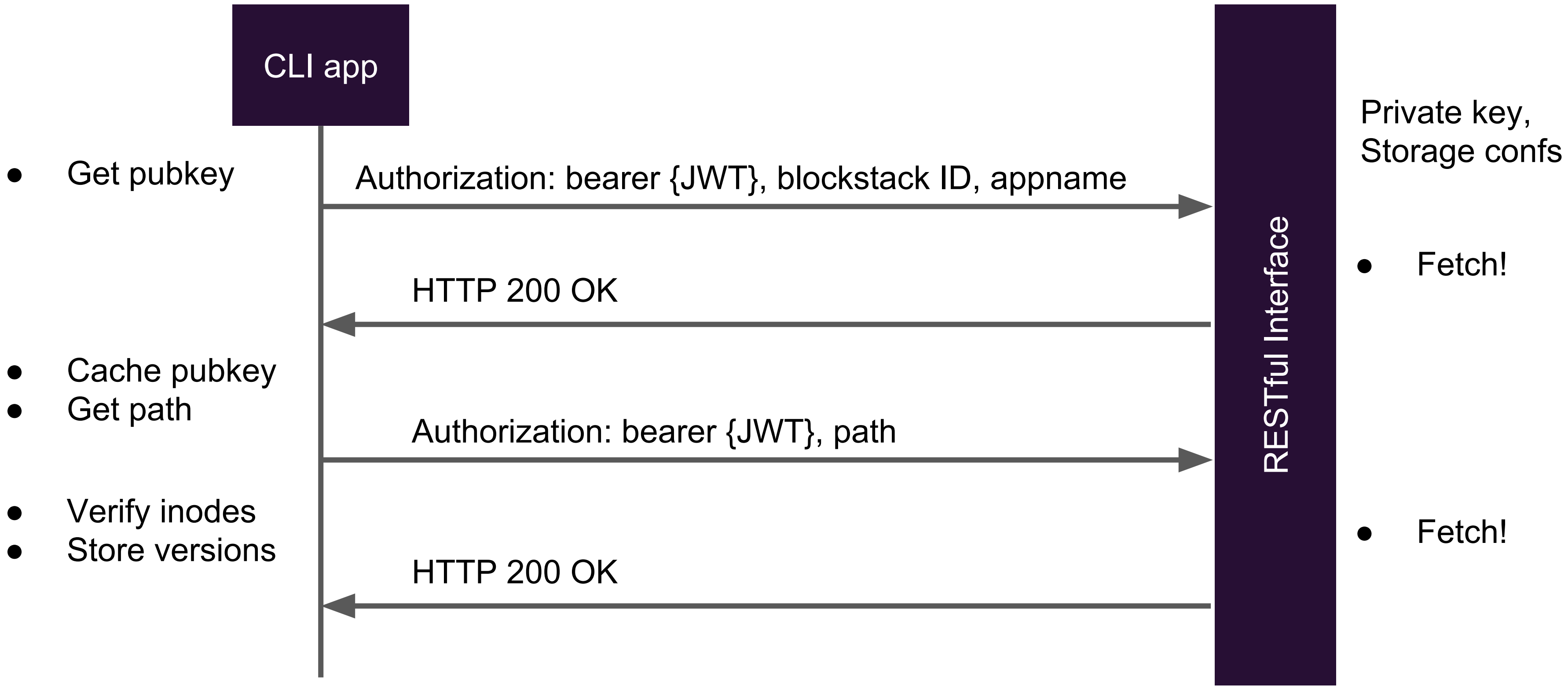


- Make inodes
- Store versions
- Sign inodes

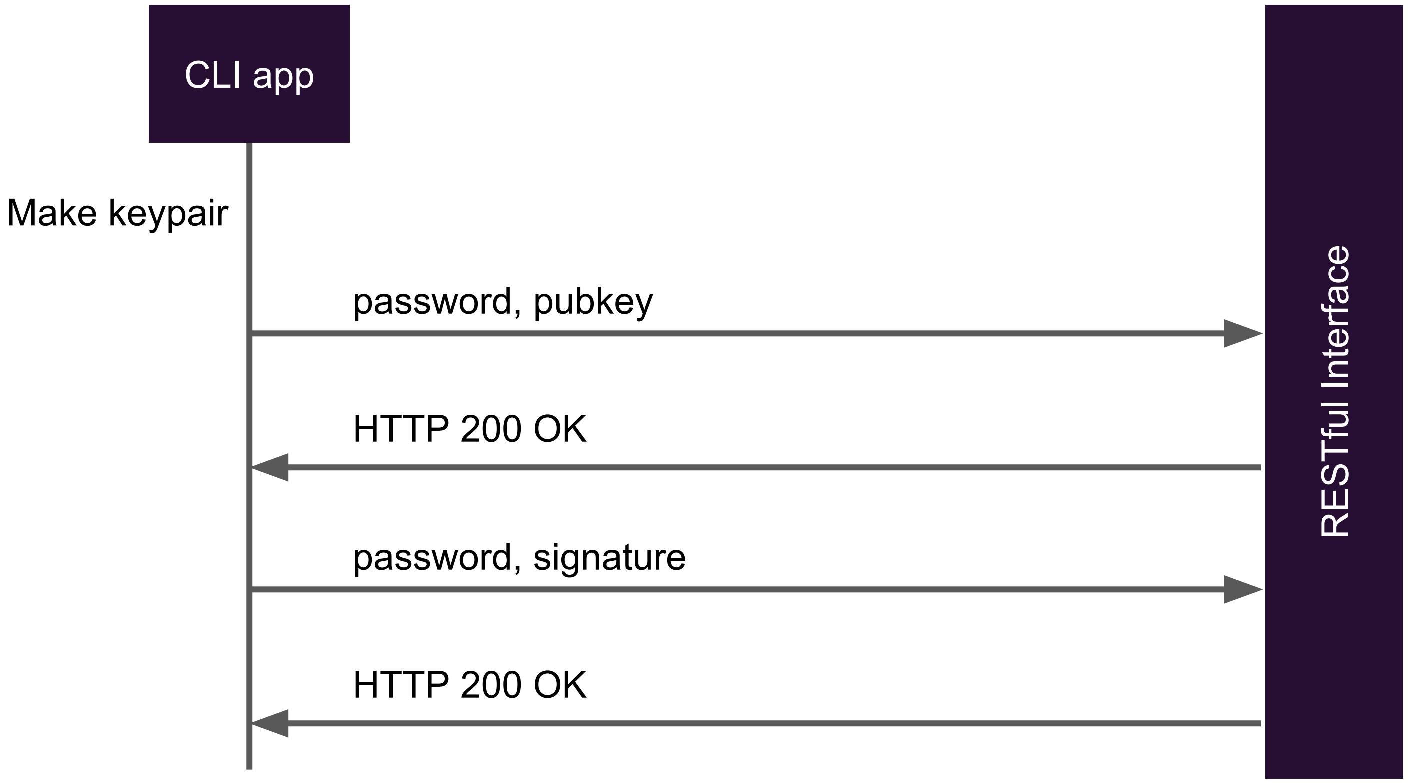
Private key,
Storage confs

- Replicate!

## Storage Programming Model: Reads



# Storage Programming Model: Key Discovery



Private key,
Storage confs

- Sign!
- Replicate!

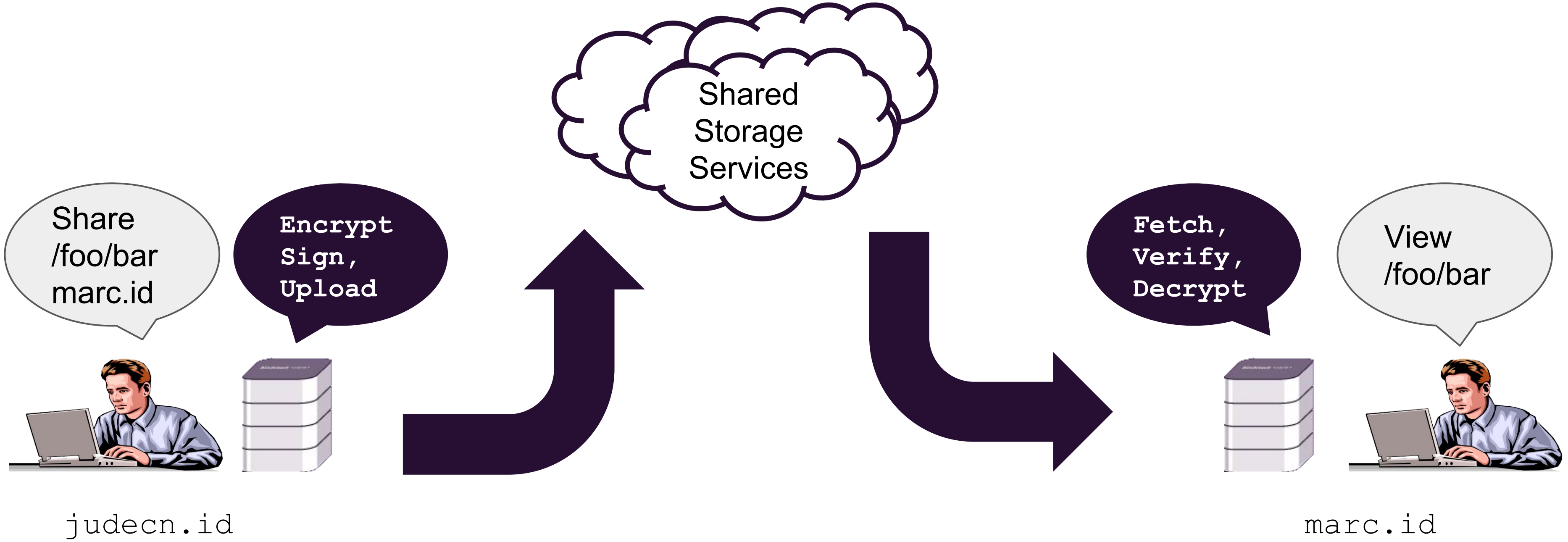
## Getting Started with Blockstack

- `brew install gmp libffi openssl # OS X`
- `virtualenv /tmp/blockstack`
- `source /tmp/blockstack/bin/activate`
- `pip install --upgrade pip`
- `git clone https://github.com/blockstack/blockstack-core`
- `cd blockstack-core`
- `git checkout rc-0.14.2`
- `./setup.py build && ./setup.py install`

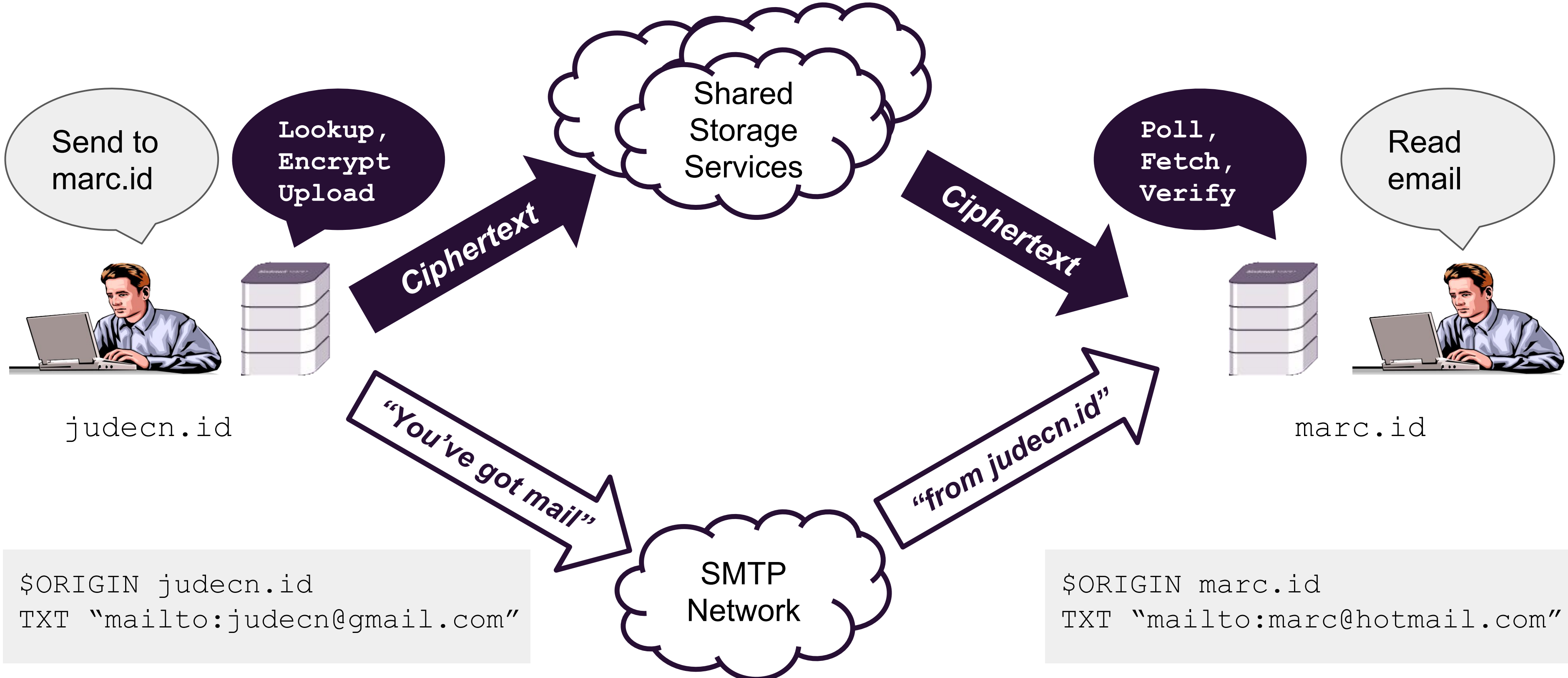
## Getting Started with Blockstack

- `blockstack --debug setup`
- `blockstack --debug api start`
- `ps aux | grep blockstack`
- `tail -n 30 ~/.blockstack/api_endpoint.log`
- `cd demos/python-filesharing`

# Sample App:  Encrypted File-sharing



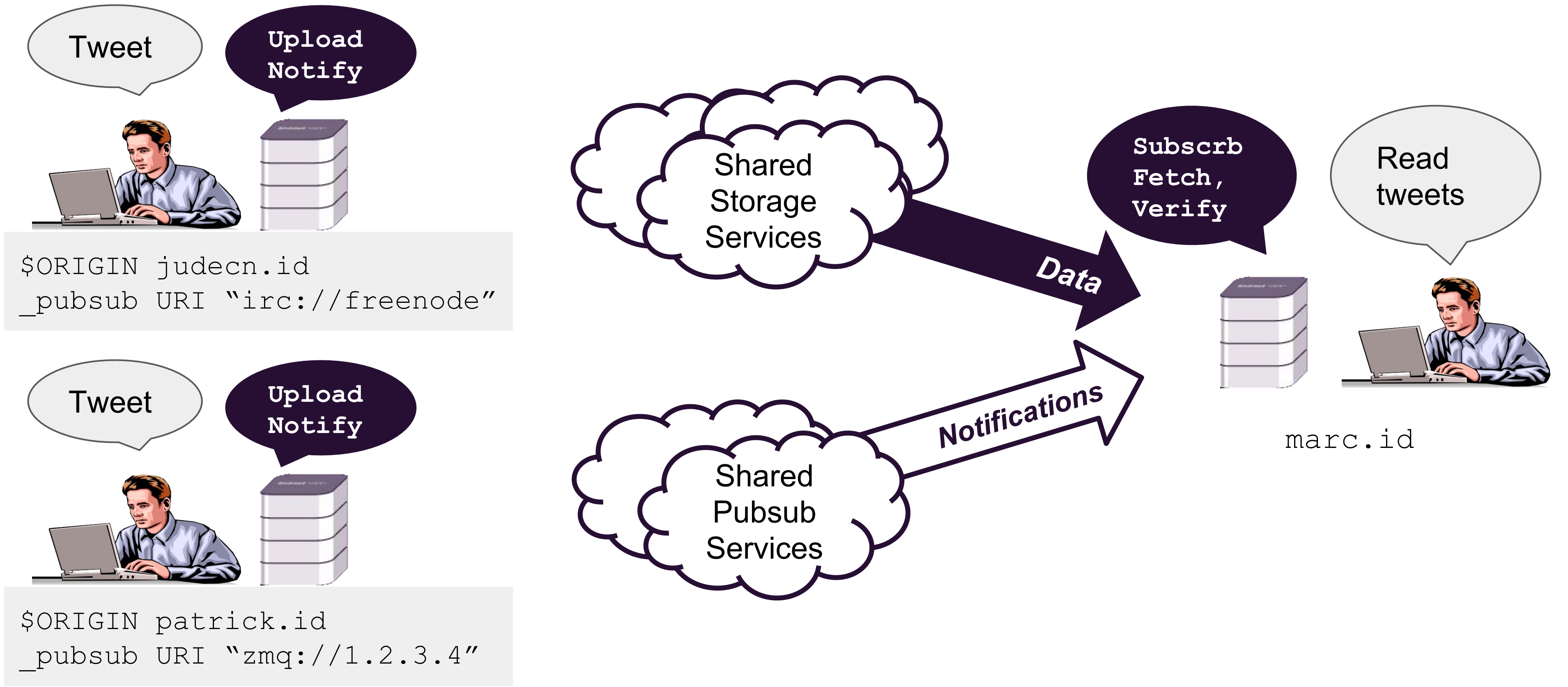
# Sample App: Private Email sans PGP



```
$ORIGIN judecn.id
TXT "mailto:judecn@gmail.com"
```

```
$ORIGIN marc.id
TXT "mailto:marc@hotmail.com"
```

# Sample App:  Decentralized Twitter



```
$ORIGIN judecn.id
_pubsub URI “irc://freenode”
```

```
$ORIGIN patrick.id
_pubsub URI “zmq://1.2.3.4”
```

## Design Patterns

- Reuse legacy Web for availability
  - Identify services in zone file
  - Bulk data via storage service
  - Push notifications via pubsub service
- Use Blockstack for authenticity
  - Authenticates user with pubkey
  - Sign/verify ALL content
  - Auto-managed keyrings

## Towards No-Server Applications

- Single-user functionality is client-side
- What about cross-user functionality?
  - Search indexes
  - Analytics
  - Karma
- Edge computing:  a possible solution?
  - Crawl users asynchronously
  - Save to developer's storage

## State of the System

- In production for 2+ years
- Over 70,000 names registered
- Peer-reviewed
- Apps by end of Q1 2017
- Open source

https://github.com/blockstack