Decentralizing Privacy: Using Blockchain to Protect Personal Data

Guy Zyskind, Oz Nathan, Alex 'Sandy' Pentland

The problem of protecting personal data

Data are stored centrally (Trusted Third Party model):

User perspective:

- **Security breaches**: a single point of failure
- Users don’t own their data (lack of **ownership**)
- Users can’t audit (lack of **transparency**)

The problem of protecting personal data

Data are stored centrally (Trusted Third Party model):

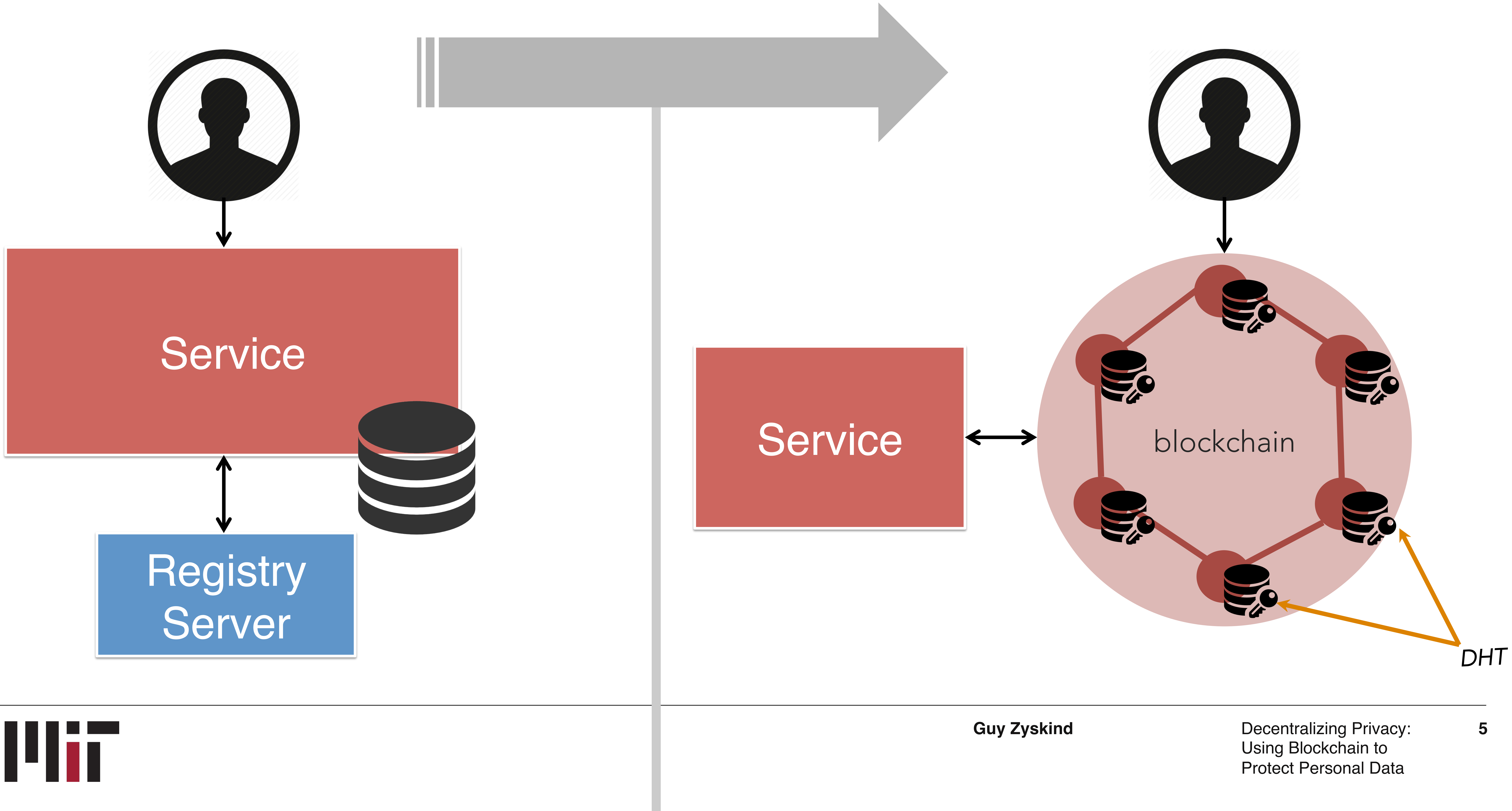
User perspective:

- **Security breaches**: a single point of failure
- Users don't own their data (lack of **ownership**)
- Users can't audit (lack of **transparency**)

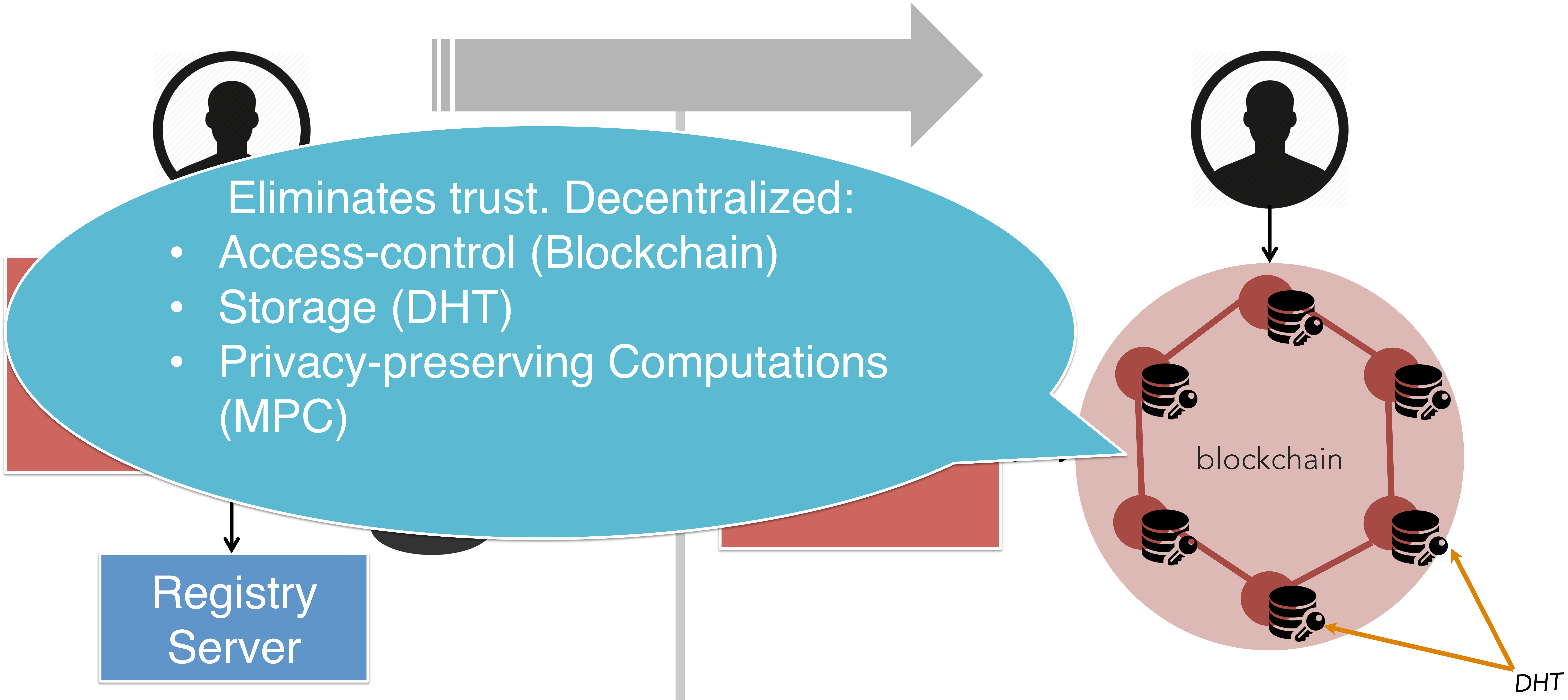
Service perspective:

- **Cost** (compliancy, security audits, Hiring CS PhDs…)
- Brand **reputation**
- **Simplicity**

General idea – A (verifiable) privacy-preserving decentralized cloud



General idea – Simulate TTPs with a P2P network + blockchain



A brief introduction to Bitcoin

- Proposed in 2008 in a paper by Satoshi Nakamoto (pseudonym).
- Enables parties to directly transfer a digital currency (Bitcoins) **without a *TTP*** (i.e., banks).
- Instead, a network of untrusted peers ensures the validity of *all* transactions.
- All correct transactions are publicly verifiable through a public ledger (the **blockchain**).

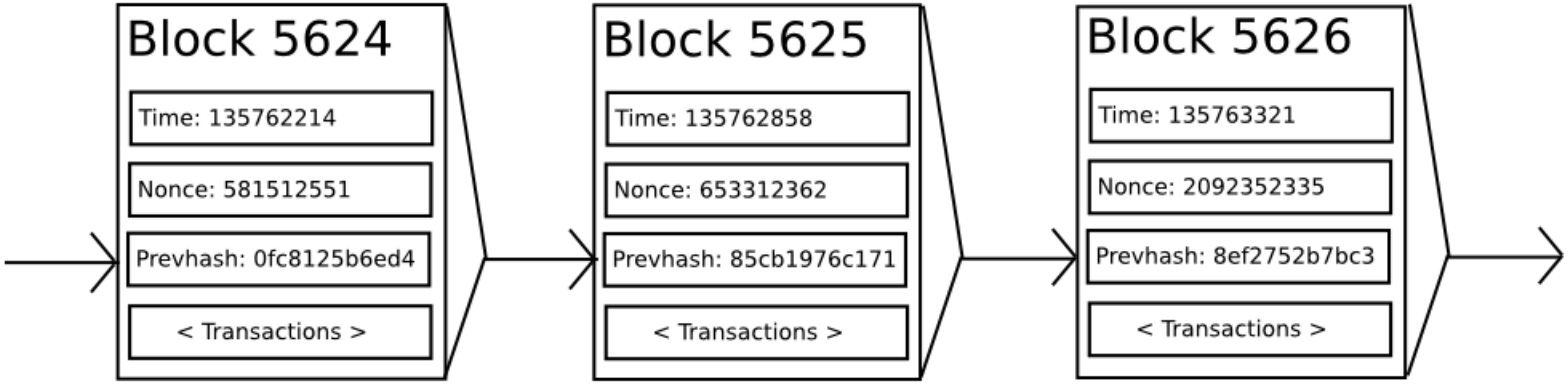
How does Bitcoin work? In a nutshell ..

Goal: Construct a public time-stamped log of all *valid* transactions without using TTPs.

How does Bitcoin work? In a nutshell ..

Goal: Construct a public time-stamped log of all *valid* transactions without using TTPs.

How? Every ~10 minutes (expected time), reach a *distributed consensus* ensuring valid mempool (floating) transactions are grouped into a block. Then, append the block to the end of the chain. The blockchain is the desired public log.



Distributed consensus mechanism

Nakamoto consensus (AKA – Proof of Work):

For every round *t* and every miner *m*:

Distributed consensus mechanism

Nakamoto consensus (AKA – Proof of Work):

For every round *t* and every miner *m*:

- Collect mempool transactions (tx's) and validate them. Construct block $b_{m,t}$

Distributed consensus mechanism

Nakamoto consensus (AKA – Proof of Work):

For every round *t* and every miner *m*:

- Collect mempool transactions (tx's) and validate them. Construct block $b_{m,t}$
- Attempt to solve a hard computational puzzle [$SHA\text{-}256(SHA\text{-}256(b_{m,t})) < target$].

Distributed consensus mechanism

Nakamoto consensus (AKA – Proof of Work):

For every round *t* and every miner *m*:

- Collect mempool transactions (tx's) and validate them. Construct block $b_{m,t}$
- Attempt to solve a hard computational puzzle [$SHA\text{-}256(SHA\text{-}256(b_{m,t})) < target$].
- First miner to solve broadcasts the solution. All other miners **independently** validate the solution (work + all included transactions). If correct, they append it to their local copy of the blockchain.

Distributed consensus mechanism

Nakamoto consensus (AKA – Proof of Work):

For every round *t* and every miner *m*:

- Collect mempool transactions (tx's) and validate them. Construct block $b_{m,t}$
- Attempt to solve a hard computational puzzle [$SHA\text{-}256(SHA\text{-}256(b_{m,t})) < target$].
- First miner to solve broadcasts the solution. All other miners **independently** validate the solution (work + all included transactions). If correct, they append it to their local copy of the blockchain.
- Solver receives newly minted coins and tx fees.

How are transactions deemed valid? Scripts!

- Every transaction is associated with a script (actually, every tx output is associated with a script called scriptPubKey).
- Nodes validate transactions by executing the script with the arguments given in the tx by the sender (most importantly – her sig).
- Can run **arbitrary verifications** – not just financial (*smart contracts*).

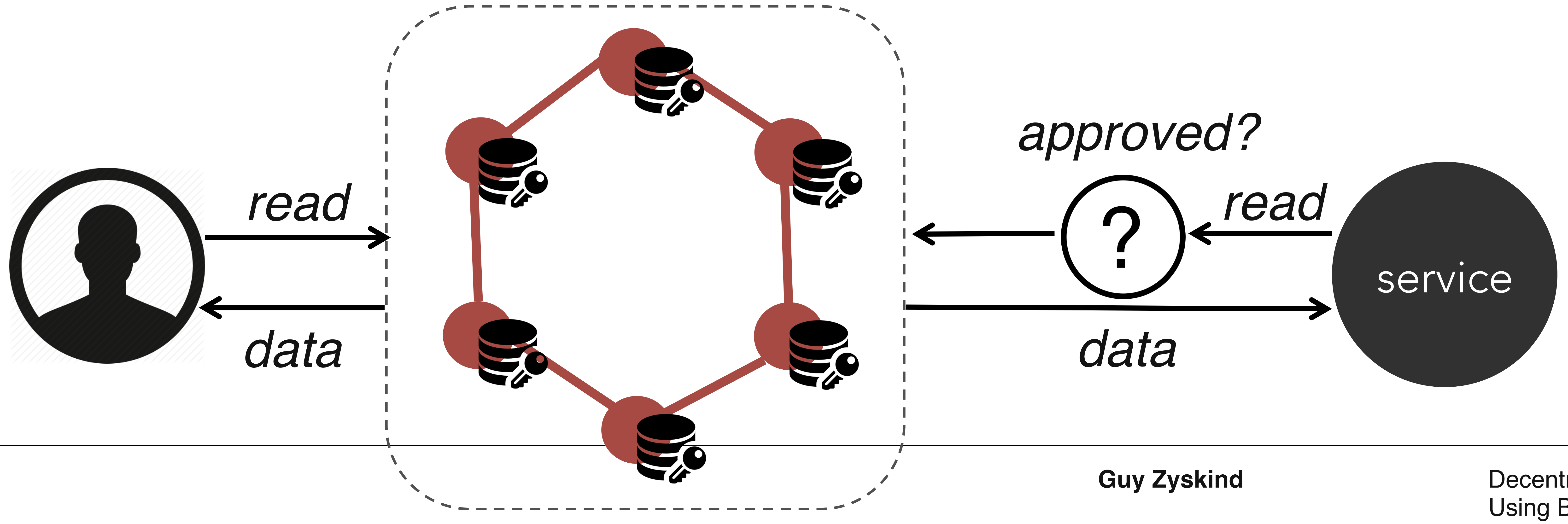
Our framework (Overview)

Goal: when a user installs a mobile app, she can control and audit what data are stored and how they are used. Access should be revokable.

Our framework (Overview)

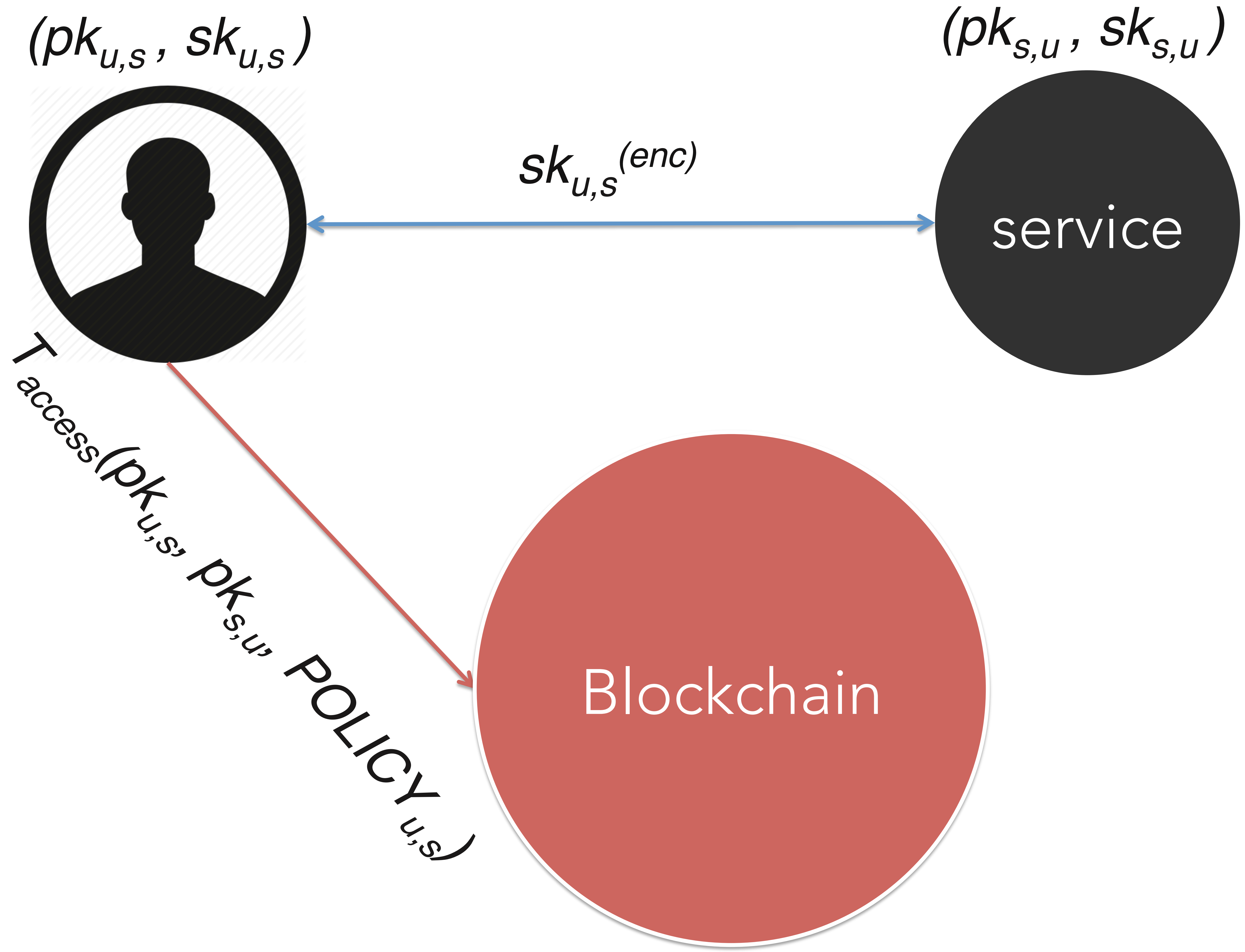
Goal: when a user installs a mobile app, she can control and audit what data are stored and how they are used. Access should be revokable.

Solution: Store access-policies to personal data on the blockchain. Then, let the blockchain nodes moderate access to a DHT.



Sign up (*or* user downloads the app)

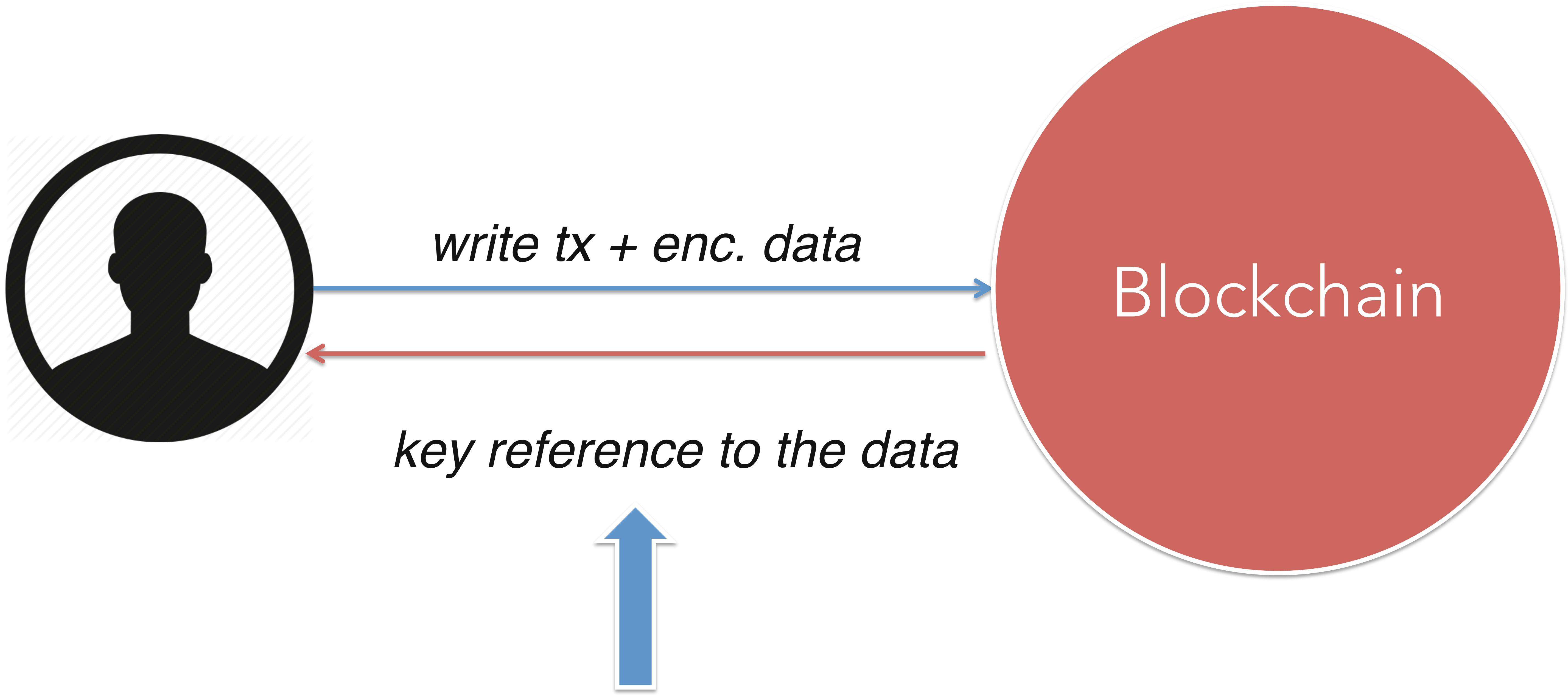
- User *u* and service *s each* generate a signing key pair.
- A symmetric encryption key is generated and shared over a secure channel.
- The user approves the list of permissions $POLICY_{u,s}$
- *u* sends $T_{access}(pk_{u,s}, pk_{s,u}, POLICY_{u,s})$ to the blockchain.
- **Also used for uninstall/modify.**



Storing & loading data

Storing data:

- Send $T_{data}(E_v, 'w')$. Nodes verify sig against policy.
- Set $k=SHA\text{-}256(E_{u,s}(v))$



(key stored on **blockchain**, data stored on **DHT**).

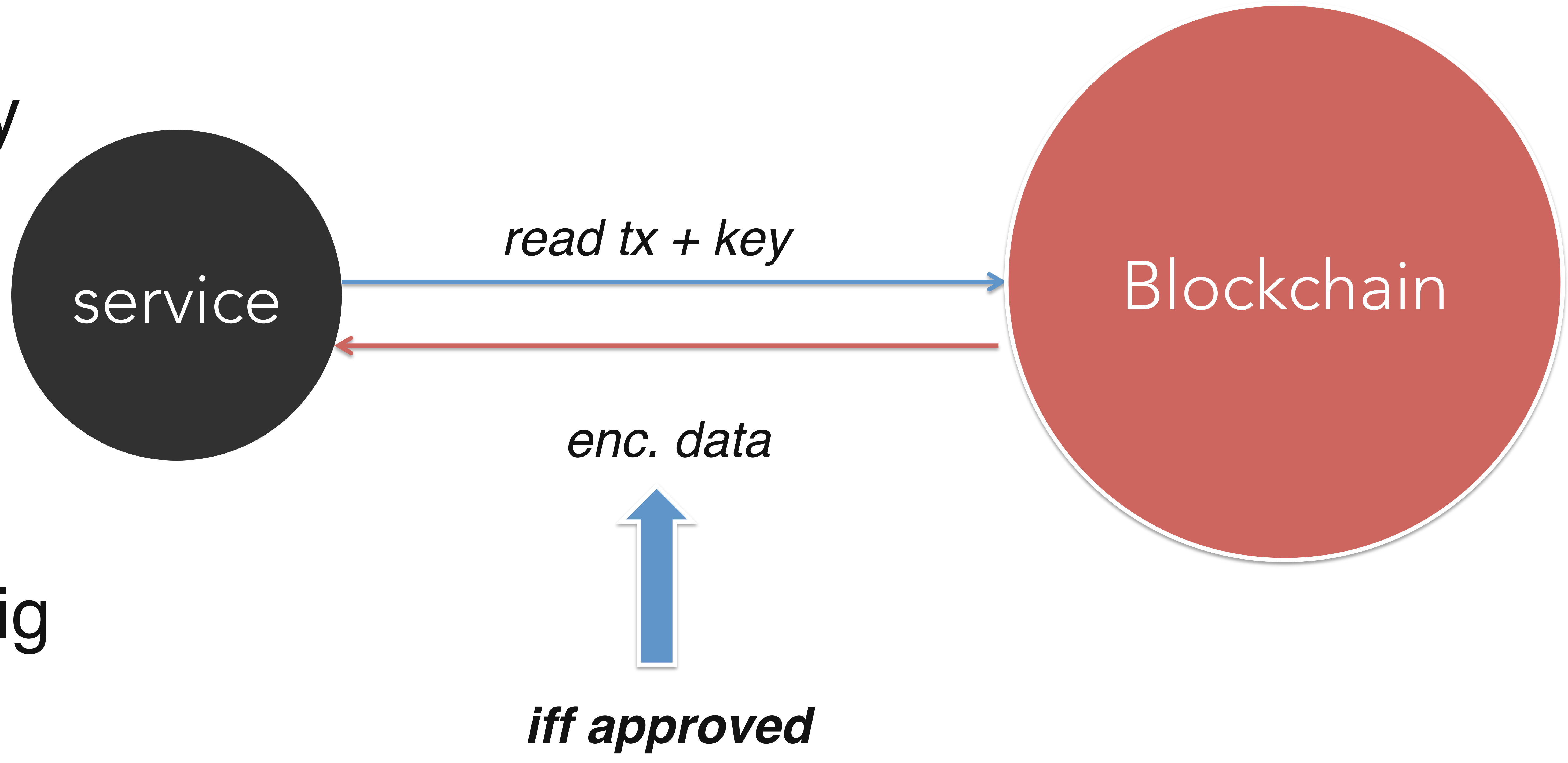
Storing & loading data

Storing data:

- Send $T_{data}(E_v, 'w')$. Nodes verify sig against policy.
- Set $k = SHA\text{-}256(E_{u,s}(v))$

Reading data:

- Send $T_{data}(k, 'r')$. Nodes verify sig against policy.
- Return $v \leftarrow DHT[k]$



Security and privacy analysis

- An adversary controlling any number of DHT nodes cannot compromise privacy (because of encryption).

adversary **A**

Security and privacy analysis

- An adversary controlling any number of DHT nodes cannot compromise privacy (because of encryption).
- An adversary controlling less than 50% of the computational power of miners cannot compromise privacy or resiliency (Nakamoto consensus & forging digital signatures)

adversary **A**

Security and privacy analysis

- An adversary controlling any number of DHT nodes cannot compromise privacy (because of encryption).
- An adversary controlling less than 50% of the computational power of miners cannot compromise privacy or resiliency (Nakamoto consensus & forging digital signatures)
- An adversary controlling the service can learn as much as permissions allows, but **traceable.** An improved model follows.

adversary **A**

Security and privacy analysis

- An adversary controlling any number of DHT nodes cannot compromise privacy (because of encryption).
- An adversary controlling less than 50% of the computational power of miners cannot compromise privacy or resiliency (Nakamoto consensus & forging digital signatures)
- An adversary controlling the service can learn as much as permissions allows, but **traceable.** An improved model follows.
- In general, an adversary can't learn anything, as that implies forging either the user or the service's sig.

adversary **A**

Adding secure computation

Problem: Malicious services could read the raw data and store them. In addition, encrypted DHTs are mainly useful for random search.

Adding secure computation

Problem: Malicious services could read the raw data and store them. In addition, encrypted DHTs are mainly useful for random search.

Solution: Instead of direct-access, use secure MPC. The network already exists!

Adding secure computation

Problem: Malicious services could read the raw data and store them. In addition, encrypted DHTs are mainly useful for random search.

Solution: Instead of direct-access, use secure MPC. The network already exists!

Implementation (sketch):

- Instead of encrypting data, secret-share them. Policies now allow services to compute functions over private data, **but they can't obtain the raw data.**
- T_{access} and T_{data} require small modifications. Off-chain network needs to be extended from storage only to MPC.

Security and privacy analysis (MPC)

Key observations:

- An adversary controlling the service **can never reveal the raw data**. Specifically, if *x* are the secret shared data, the service can only obtain *f(x)*.

- **Privacy** and **resiliency** follows feasibility results of secure MPC [BGW87] (unconditionally secure and resilient against a dishonest minority. Better bounds exist with computational assumptions).

Reputation and trust

- Bitcoin's PoW is an expensive way to reach distributed consensus.
- It weighs each node based on its computational power ($trust_n \propto resources(n)$)
- Instead, approximate *trust* (or *reputation*) by node's honesty. For example:

$$trust_n^{(i)} = \frac{1}{1 + e^{-\alpha(\#good - \#bad)}}$$

* A plausible example; this is an open question requiring significant research.

Conclusions and future work

- Blockchains are a practical tool for removing TTPs from the equation.
- Can be used to govern **access-control** and ensure **transparency**.
- Personal data are not stored centrally (DHT); third-parties run MPC protocols on the network ***without accessing raw data directly***.
- **Future work:** making secure MPC scalable for large *n* & high dimensional data; formalizing atomicity of operations; ease-of-use (a parser).