

# Blockchain-based System for Secure Data Storage with Private Keyword Search

DO, Hoang Giang; NG, Wee Keong. In: Services (SERVICES), 2017 IEEE World Congress on.

# Introduction

- There has been significant growth in the interest to **outsource data** as well as **operational services to clouds**.
- Traditional cloud storage has come to rely almost exclusively on large storage providers acting as trusted third parties to transfer and store data.  
( **performance, availability, security and high operation cost** )
- A decentralized cloud storage network has been introduced with **many advantages** over the datacenter-based storage.

# Decentralized Cloud Storage

Decentralized cloud storage leverages client-side encryption to maintain data security. However, the management of encrypted data poses several challenges.

## Challenges

- The data owners should have **the capability to grant permission for others to search on the remotely encrypted dataset and obtain partial but useful content.**
  - A trivial solution is that the data owner retrieves back the whole data set, filters and sends to the authorized client the useful parts of data.
  - This solution is **infeasible** due to the **heavy cost at the client-side** and **it defeats the very purpose of outsourcing data.**

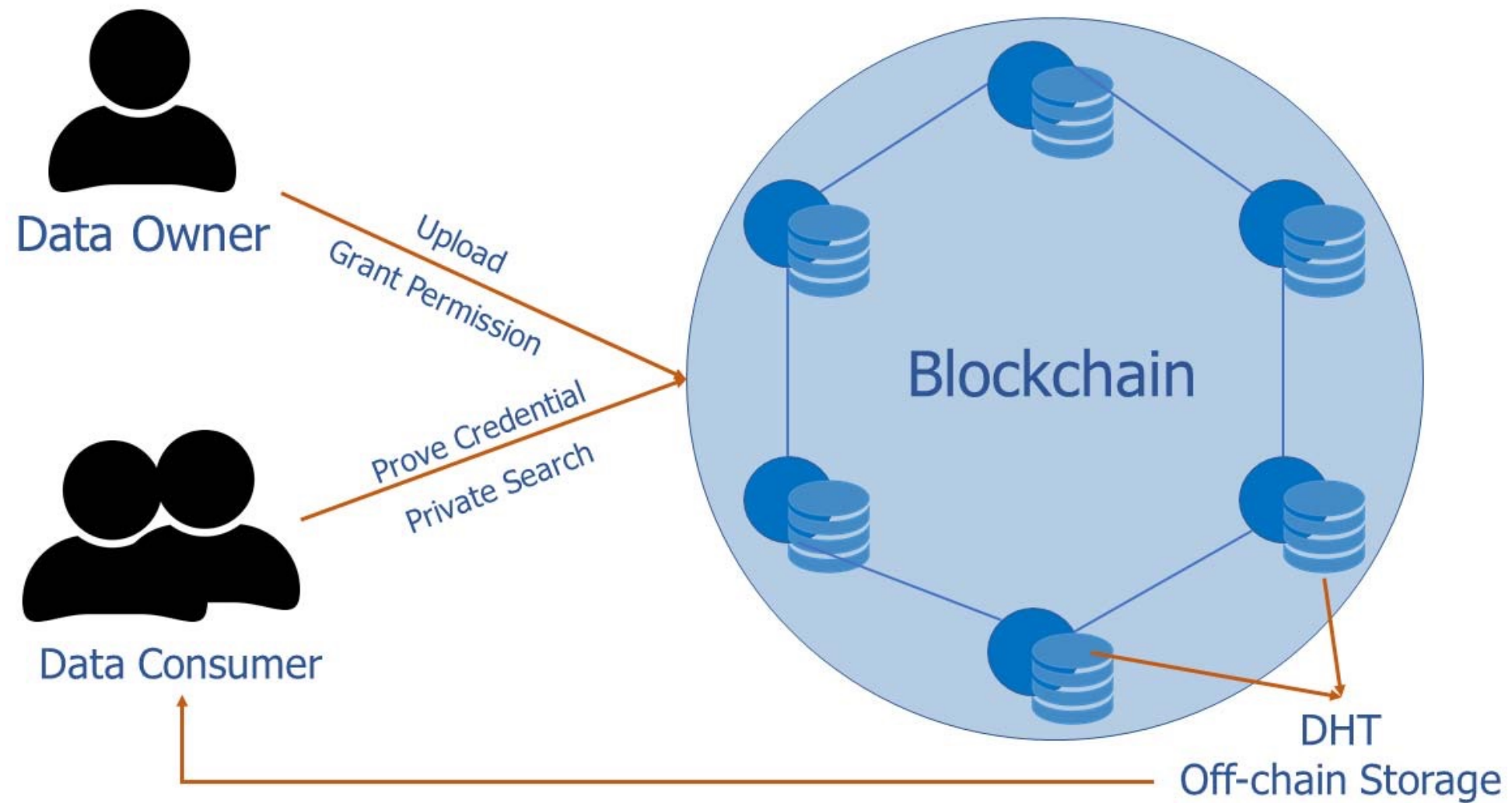
# BlockDS

BlockDS addresses the aforementioned problem : **authorized keyword searching on distributed data storage.**

## Blockchain

- Off-chain data storage access
- Permission grant and searches
- Encrypted keyword compare

# System Model



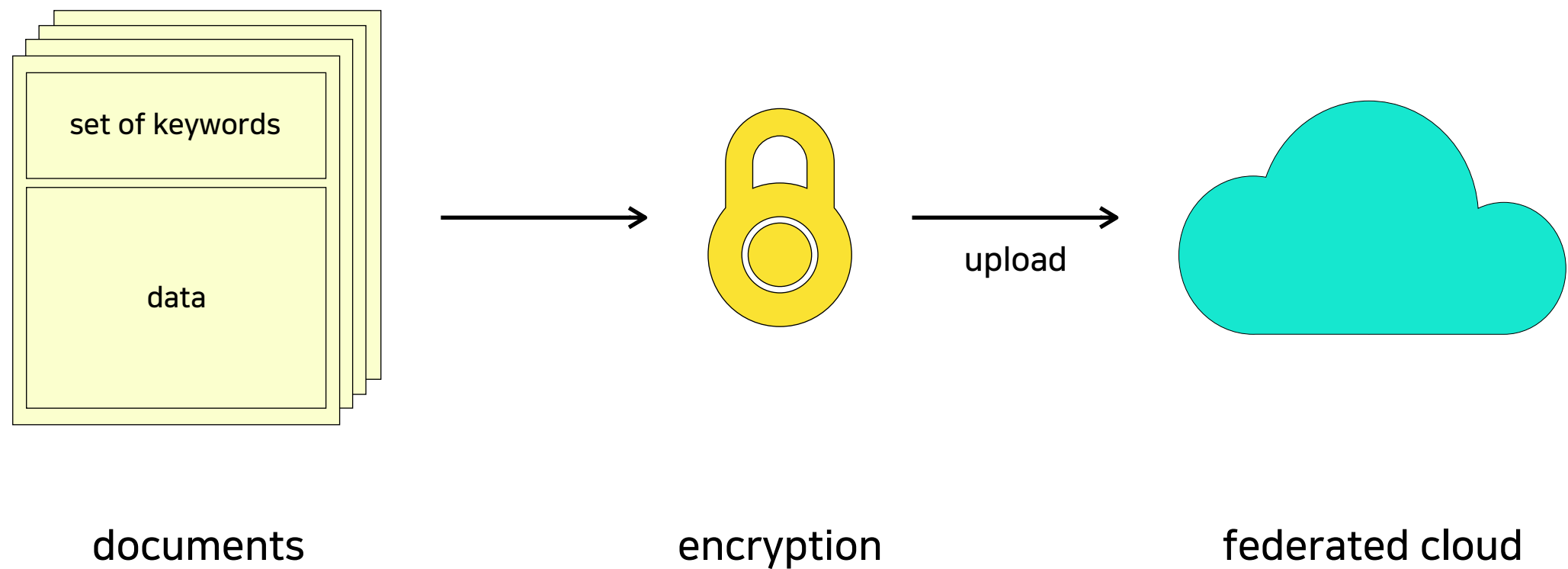
# Data Owner

- Data owner poses a data set of document.
- Data owner wishes to outsource the dataset to save the cost of data storage and maintenance.
- Data owner wants to be **able to grant the permission for other clients to search on its outsourced dataset.**

# Data Consumer

- A data consumer is a **subscriber for the dataset** provided by the data owner.
- If a data consumer subscribes for the dataset, **it is granted the permission.**
- It is able to interact with a blockchain node **to prove its credential** and **to perform keyword search on the outsourced data.**

# Client-side Encryption

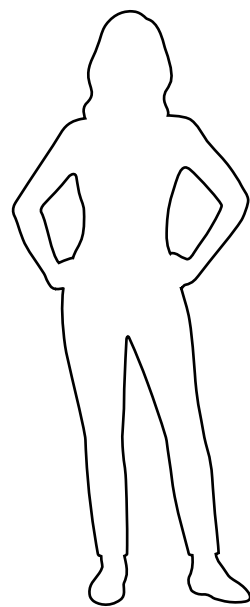




# Set Up

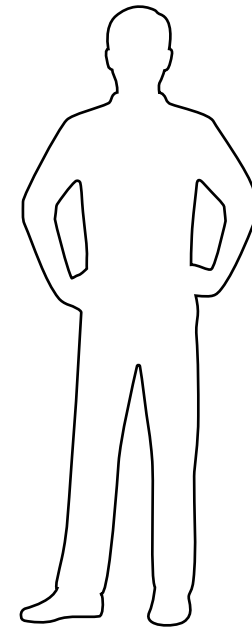
- The uploaded data include the **encrypted documents** and the **encrypted keyword tags** that facilitate the keyword search process.
- The **encrypted dataset is stored distributedly by the cloud consortium**, while the **encrypted keyword tags are maintained by the blockchain**.
- The encrypted keyword tags are used for **indexing and providing a means to access the matching encrypted documents**.

# Permission



**Data owner**

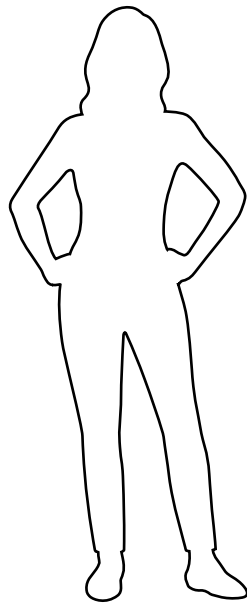
permission



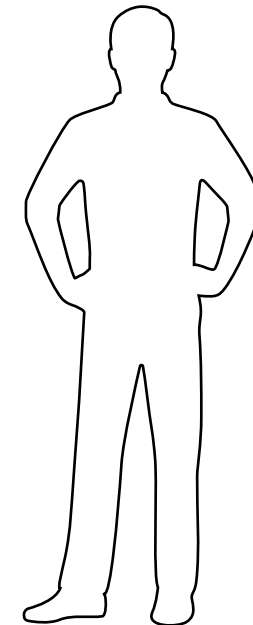
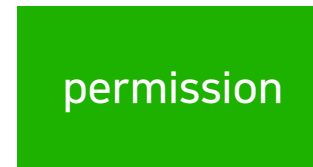
**Data consumer**

grant permission for a data consumer to search on the encrypted dataset

# Permission



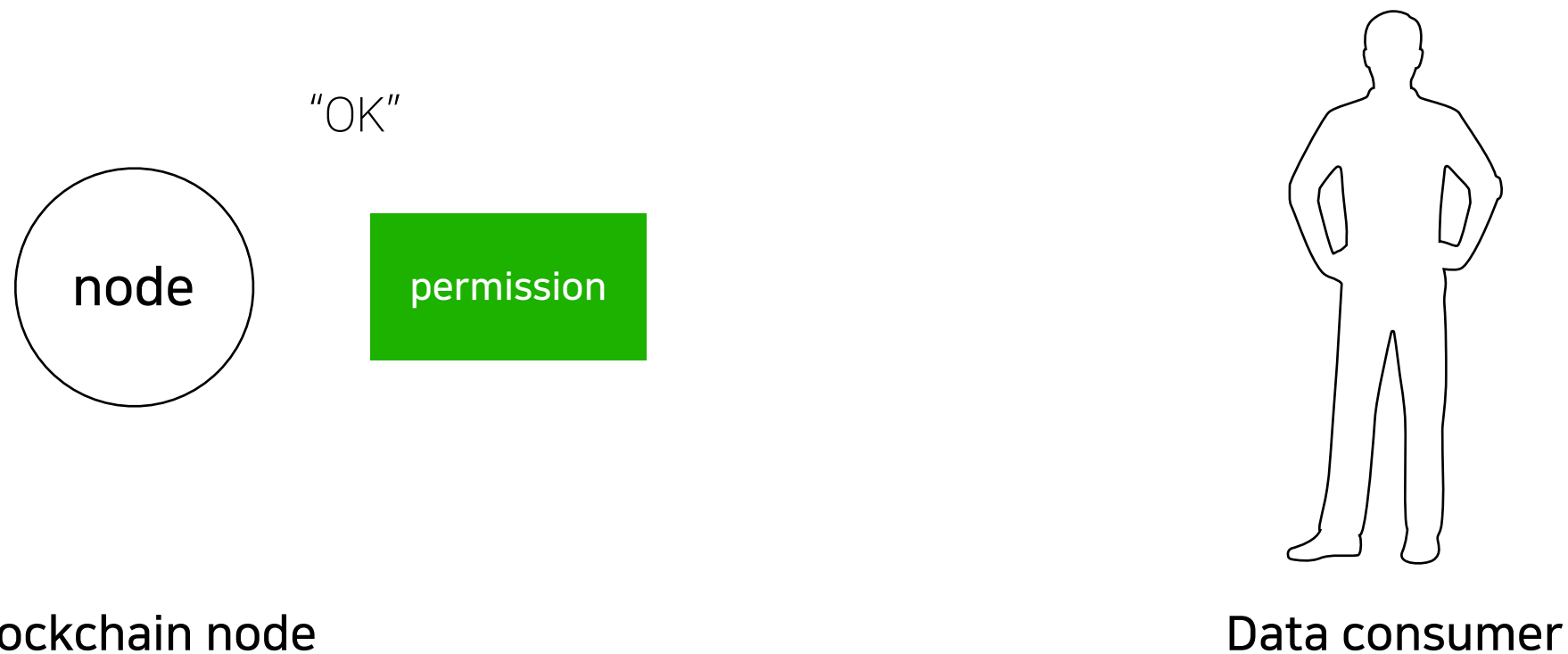
**Data owner**



**Data consumer**

grant permission for a data consumer to search on the encrypted dataset

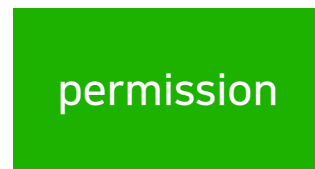
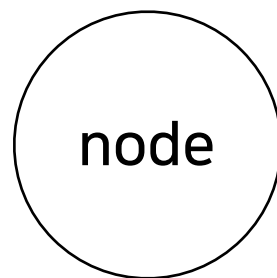
# Permission



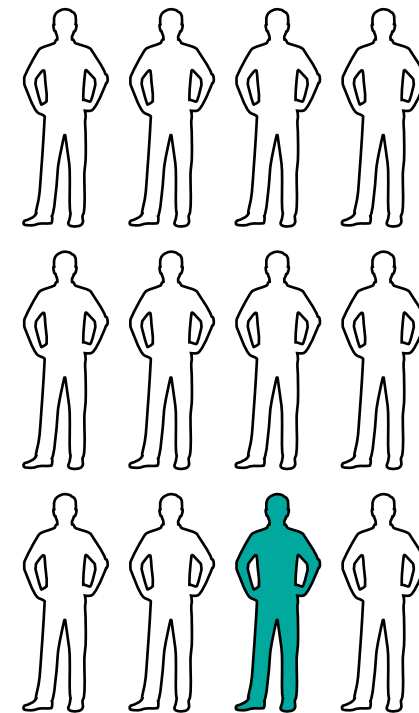
prove to the blockchain nodes that he has a particular permission for searching

# Privacy Policy

"Oh, You're one of the subscribers"



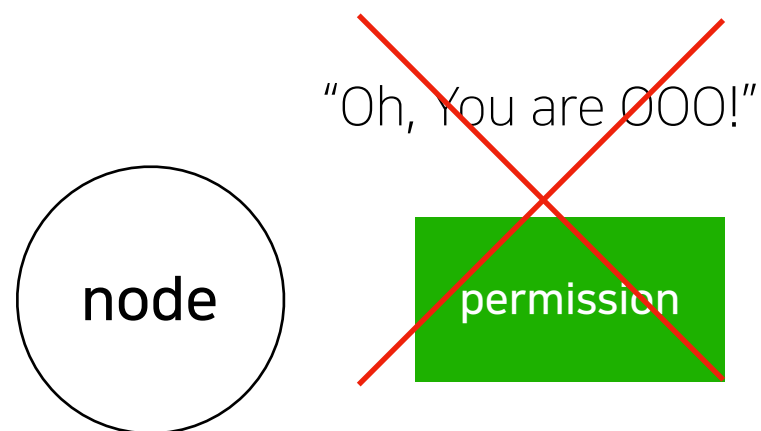
**Blockchain node**



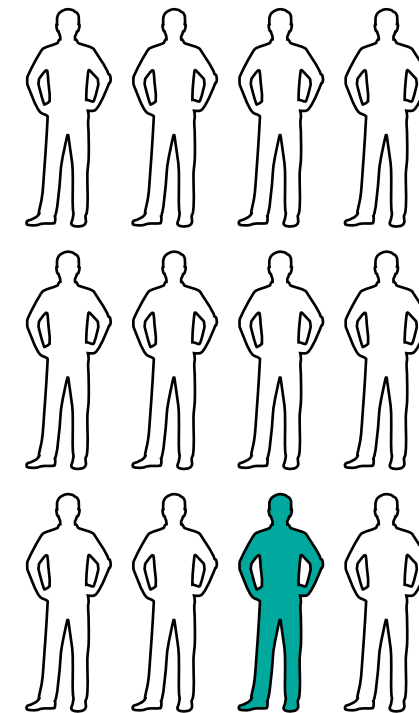
**Subscribers**

The blockchain can only verify that he is one of the subscribers.

# Privacy Policy



Blockchain node



Subscribers

But the blockchain **cannot** extract his identify from the proof.

# Strength

- Instead of having to download the whole dataset, **Data consumer should have the capability to search for a keyword to retrieve only the necessary documents from the federated cloud.**

# BlockDS System

BlockDS consists of three main components

## 1. Distributed Data Storage

- The component provides a decentralized cloud storage platform.
- The data are **distributedly stored by the federated cloud.**

## 2. Anonymous Access Control

- The anonymous access control component allows **the data consumer to anonymously convince the blockchain nodes of the possession of a certificate issued by the data owner.**
- The blockchain nodes **do not know** who is the data consumer, and how many times he has shown the credential proof.



# BlockDS System

## 3. Private Keyword Search

- The component provides an elegant mechanism for a data consumer **to identify the specific encrypted data.**
- Rather than downloading the whole data set, the data consumer **interacts with blockchain nodes and the blockchain content to get the fingerprint of the specific data content based on the keywords.**
- The data consumer **retrieves the encrypted documents from the distributed off-chain data storage.**

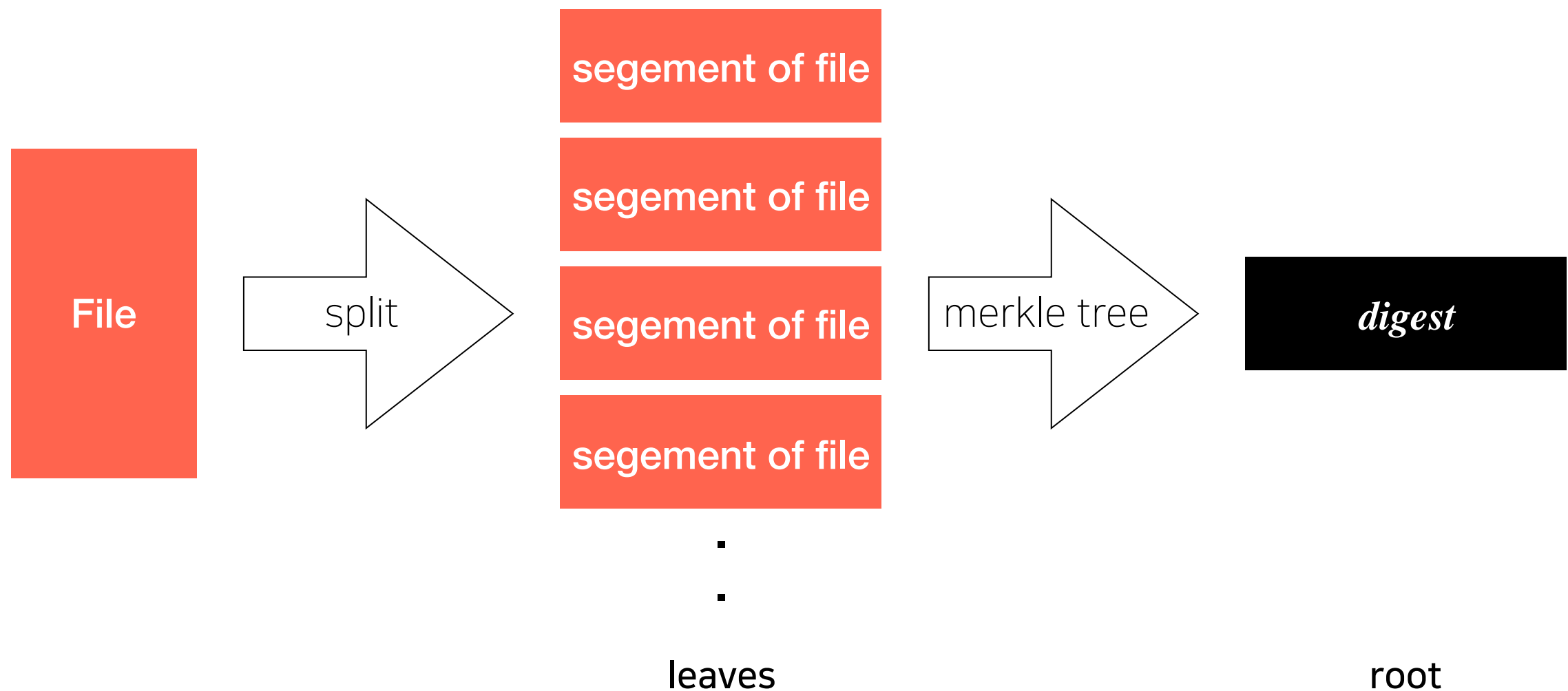
# Distributed Encrypted Data Storage

- BlockDS allows data owner to outsource a large amount of data to the system.
- The system **shards the data content and distributes the shards across peer nodes** to reduce the impact of content delivery on any given node.
- In order to maintain the access to the encrypted files, BlockDS leverages **decentralized off-chain distributed hash-table (DHT)** that is accessible through the **blockchain, which stores references to the data but not the data themselves.**
- DHTs have been widely used to coordinate and maintain metadata about peer-to-peer systems.

# Proof-of-Retrievability

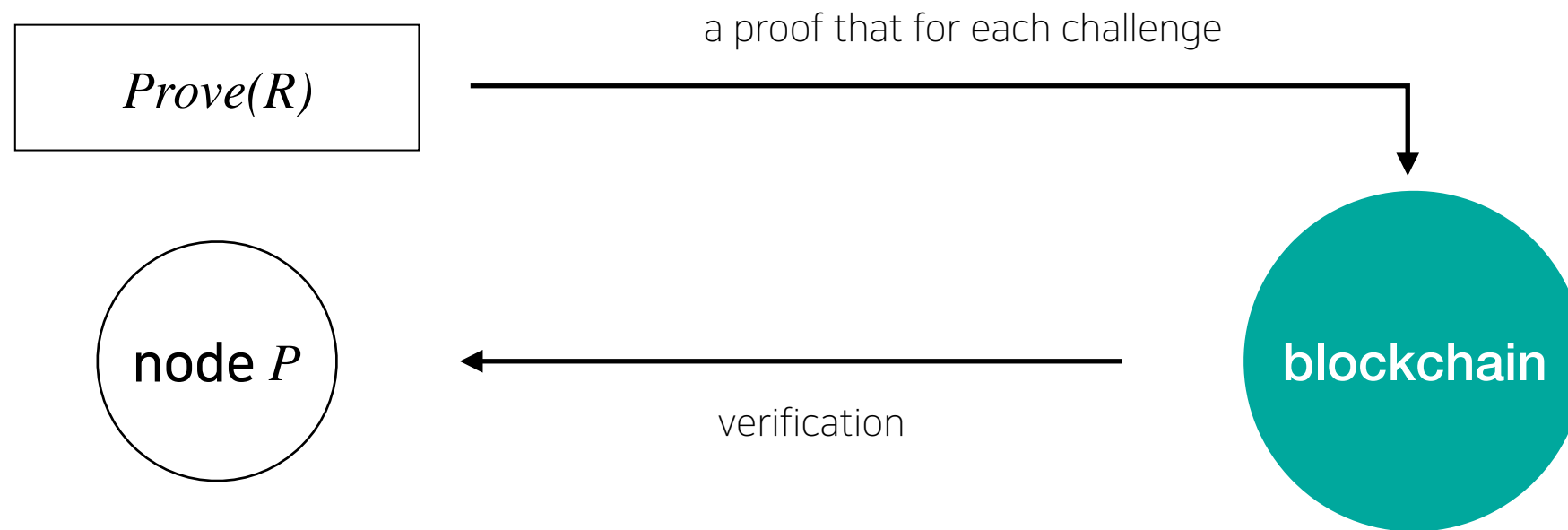
- The system leverages distributed Proof-of-Retrievability (*PoR*) to maintain the integrity of the storage.
- We consider a particular file  $F$  consist of  $n$  data segments :  $F = (F_1, F_2, \dots, F_n)$ .
- A basic *PoR* takes the form of a **challenge-response protocol** in which a node  $P$  demonstrates its possession of a file  $F$  and the fact that it can be correctly retrieved.
- To audit  $P$ 's possession of  $F$ ,  $P$  receives a random challenge  $c$  at regular basis; it produces a response  $r$ , which it can be publicly verified without possessing  $F$ .

# Setup



Node  $P$  computes a Merkle tree whose leaves are segments of the file  $F$  and whose root is *digest*.

# Prove and Verify



node have ownership of the file  $F$

nodes do not have ownership of the file  $F$

Node  $P$  outputs a proof that for each challenge index  $r_i$  in  $R$ ,  $F$  contains  $F_{r_i}$  and the accompanying path  $\pi_{r_i}$  in the Merkle tree.

The validation process verifies the Merkle path for each segment against *digest*.

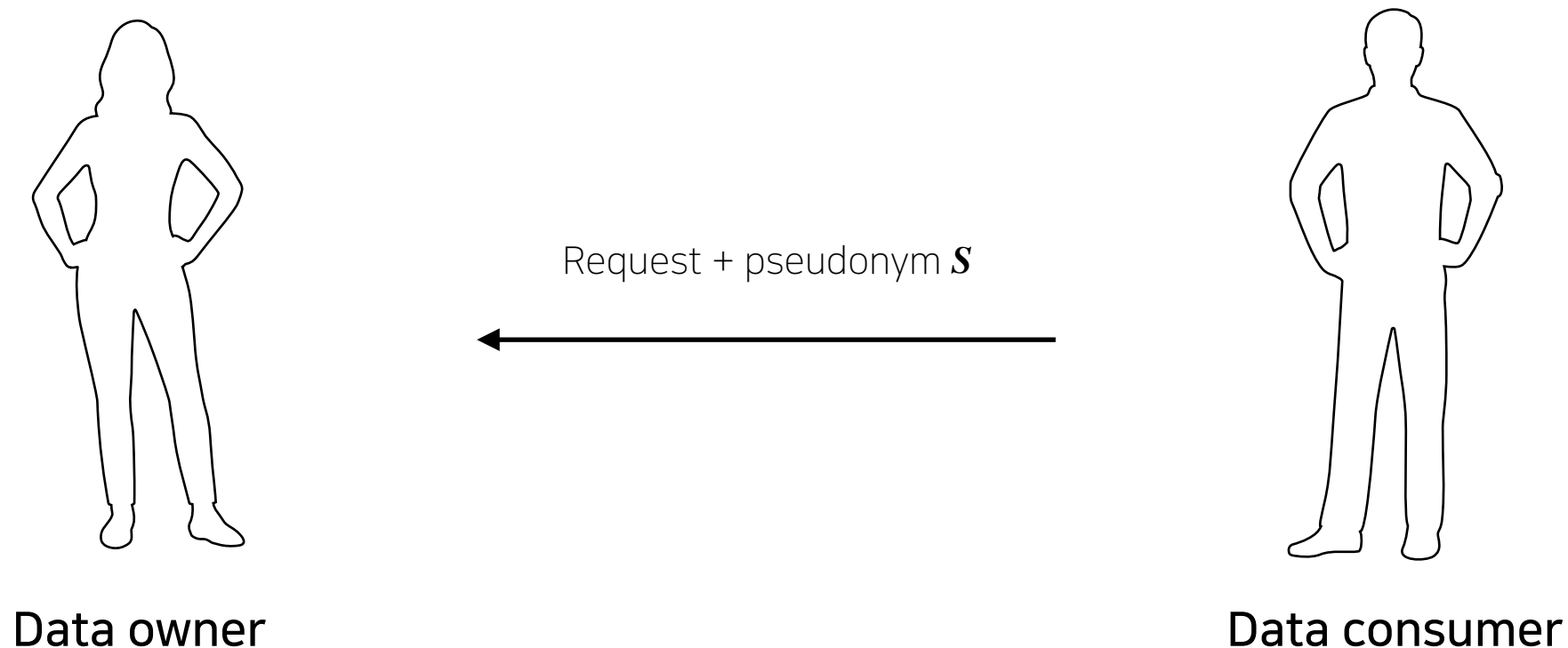
# Effect of PoR

- PoR provides a strong guarantee, namely that with overwhelming probability, if  $P$  provides correct responses,  $F$  can be retrieved completely from  $P$ .
- The correctness of the proof can be verified by every other nodes so that the **integrity** of the data storage is maintained.

# Anonymous Access Control

- Anonymous access control component allows a data consumer to obtain a **credential** from the data owner so that at some later points of time, **he is able to construct a non-interactive proof for his credential.**
- The blockchain nodes accept the **request only if the attached proof is valid.**

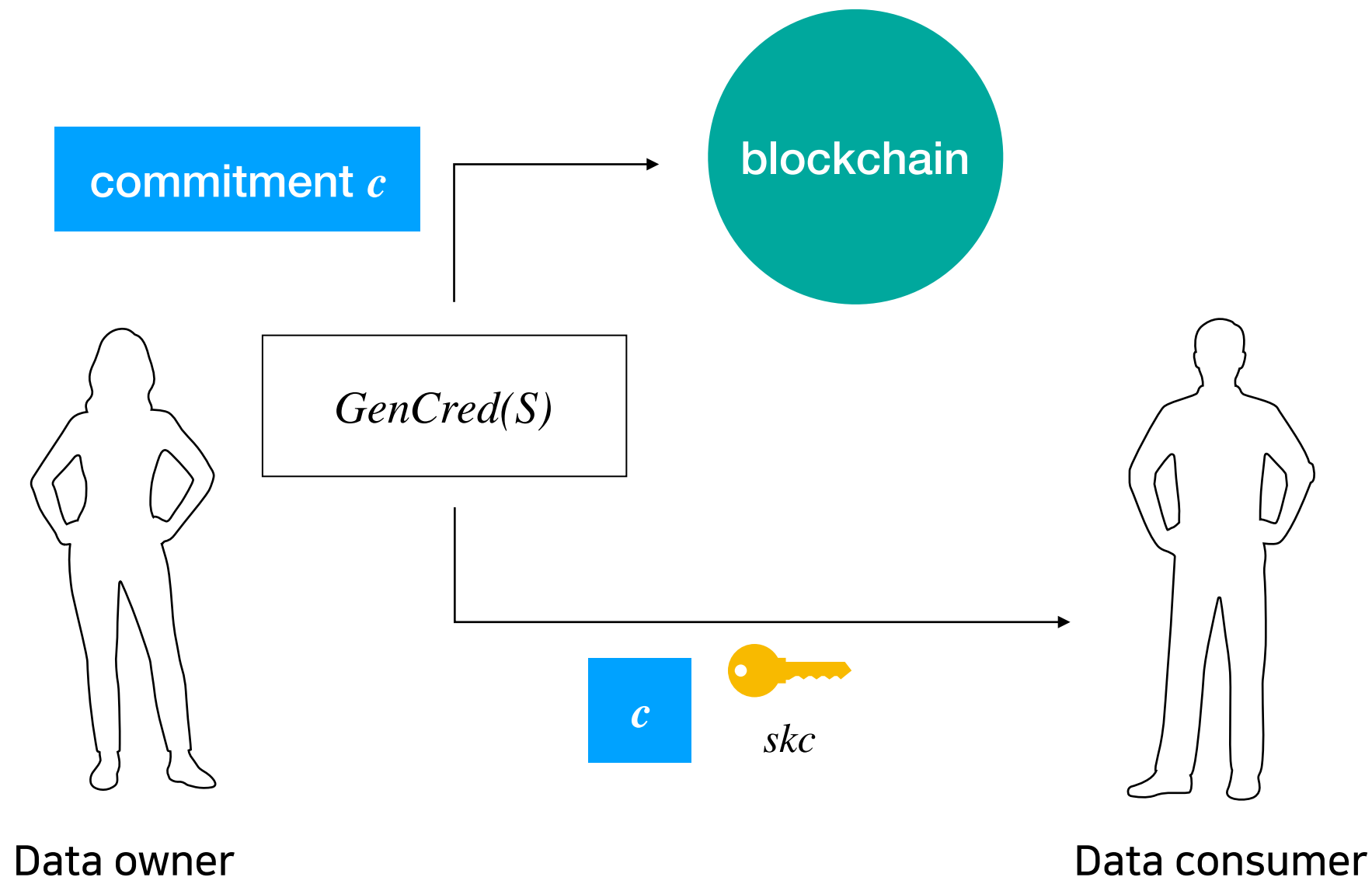
# Anonymous Access Control Process



When the data consumer wishes to obtain a credential for data access, he sends a request to data owner together with his pseudonym  $S$ .

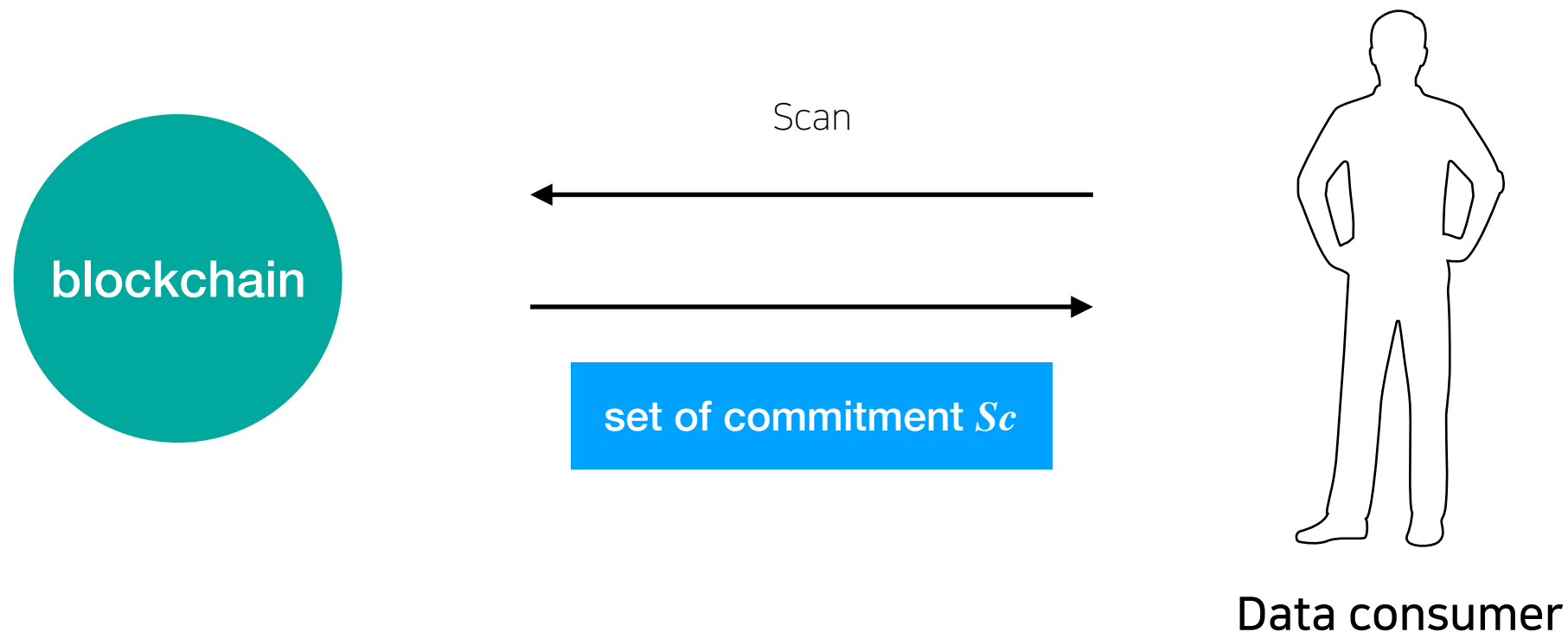


# Anonymous Access Control Process



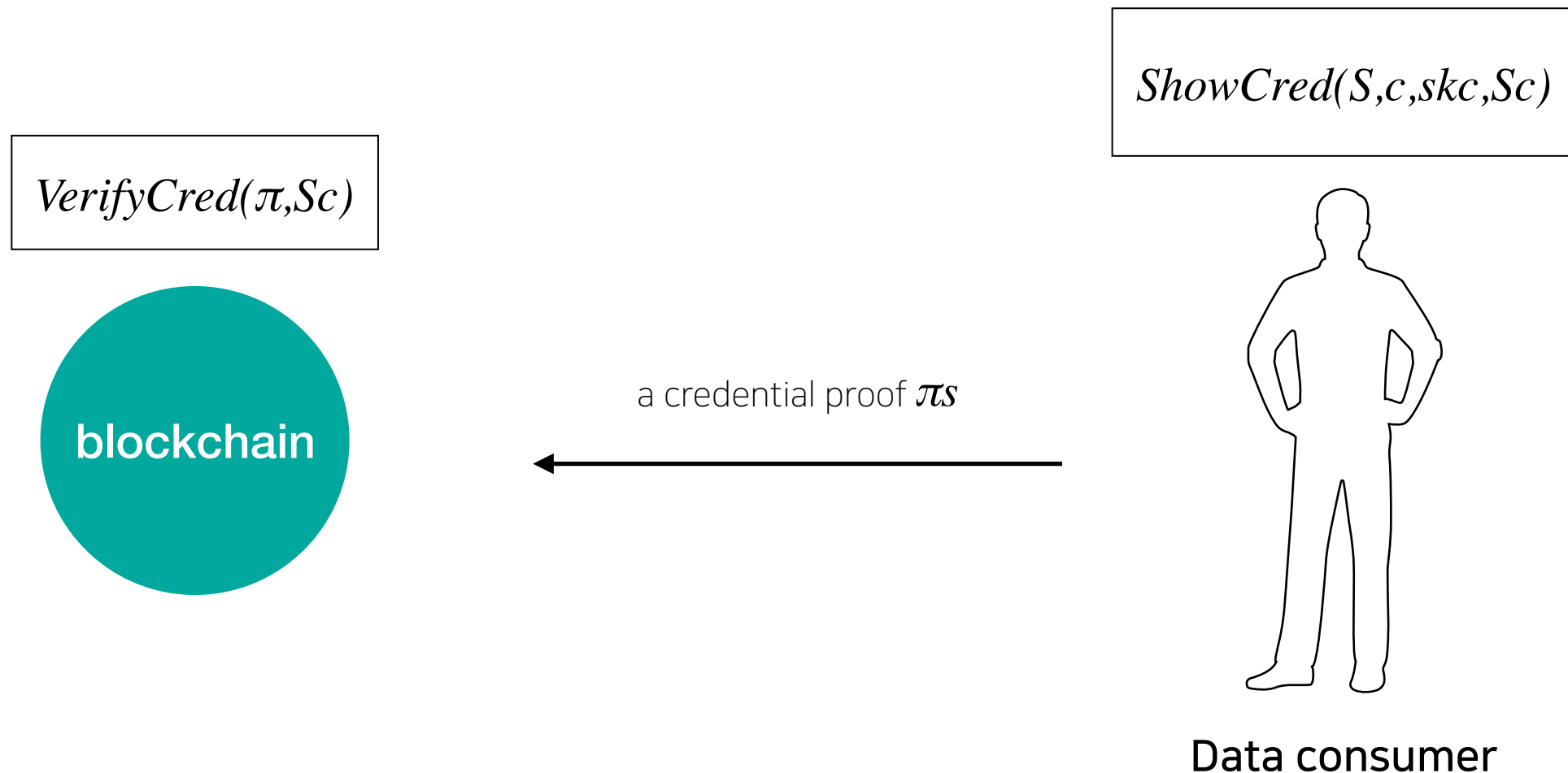
At this point, data owner runs *GenCred* routine on the input  $S$  to generate a digital commitment  $c$  and its secret key  $skc$ .

# Anonymous Access Control Process



When the data consumer wishes to show his credential, he first scan through the blockchain to obtain the set  $S_c$  consisting of all credential issued by the data owner.

# Anonymous Access Control Process

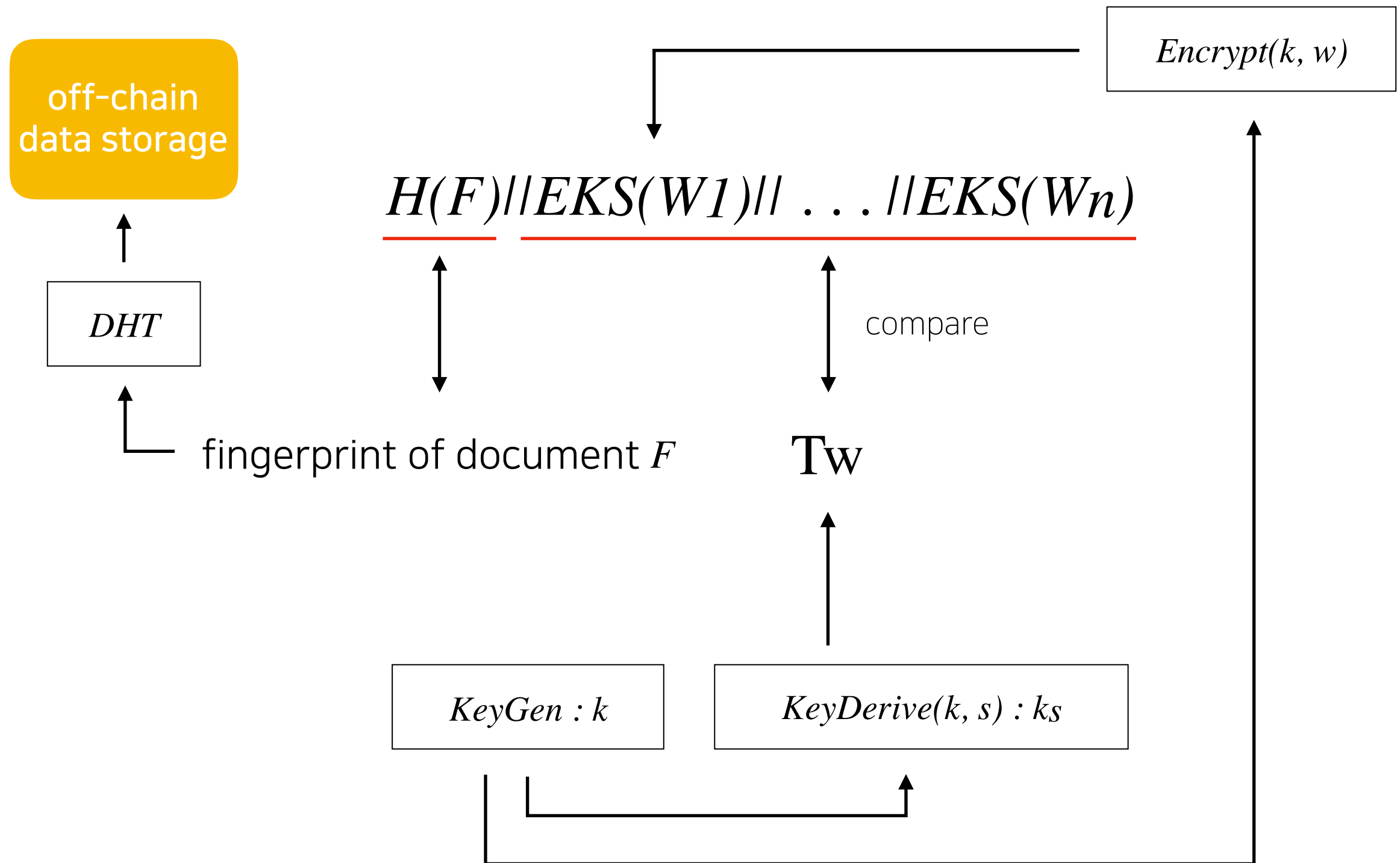


He then runs the *ShowCred* routine to generate a credential proof, and broadcast it to the blockchain nodes for verification.

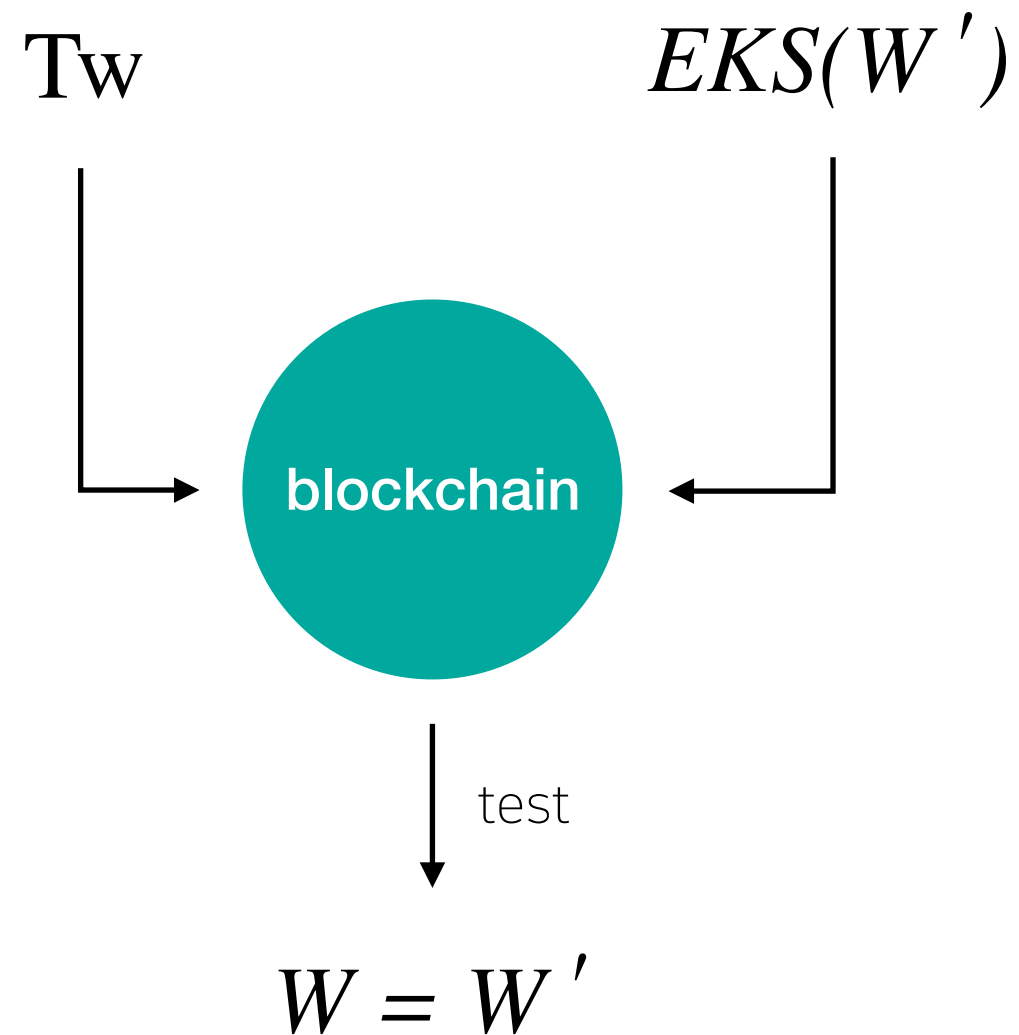
# Private Keyword Search

- The component allows the data consumer to identify the specific encrypted data that he is interested in without the requirement of downloading the whole dataset.
- The **meta-data ( the fingerprint )** of the encrypted documents are **stored in the blockchain**, and **only the authorized clients are able to search via it**.
- The blockchain does **not store actual content**, however, **it maintains the access key data** so that the data consumers are able to link to the DHT using blockchain.
- The data owner appends to the access key a list of EKS ciphertext of each keyword and stores it in the permissioned blockchain.

# Private Keyword Search



# Role of Blockchain node



# Conclusion

- The BlockDS system enables outsourcing of data storage to a fluid distributed network of service providers.
- The system makes use of blockchain technology to enforce the data integrity via PoR scheme.
- The processes of anonymous credential grant, proving credential, and private keyword search are also performed with the help of blockchain.
- **Future work** should address more complex requirements such as **credential revocation, boolean keyword search**, etc.