Department of Computer Science

A Secure and Verifiable P2P Storage System with Encrypted Search using Blockchain

Xiaohua Jia

Department of Computer Science

City University of Hong Kong

香港城市大學

City University of Hong Kong

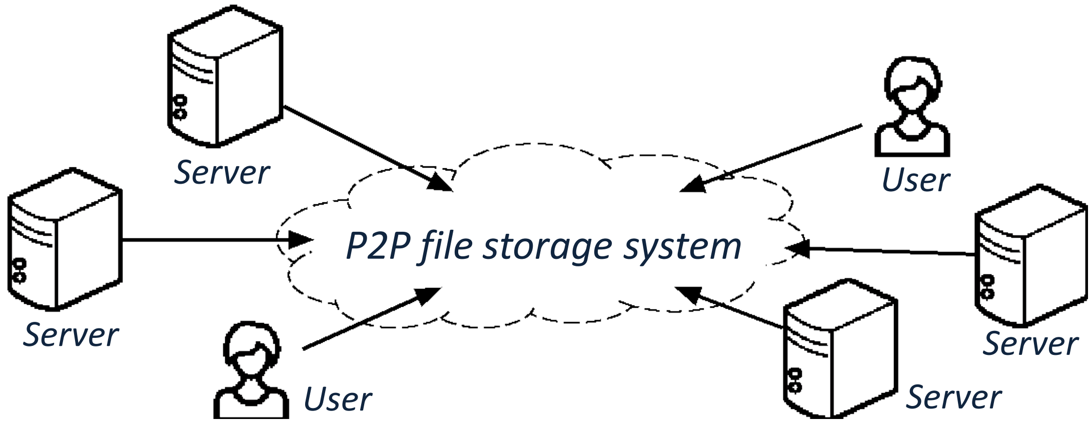
專業 創新 胸懷全球

Professional · Creative

For The World

New demand for P2P storage systems

- **Explosive growth of digital data**
  - fuelled up by e-health, e-commerce, smart cities, IoT, …
- **Mismatch between supply and demand of data storage**
  - a vast amount of under-used storages scattered all over the world
  - high demand from users looking for storage space
- **P2P storage system:**
  - utilize the unused storage space to form a huge global storage system



"centralized proprietary services are being replaced with decentralized open ones;
trusted parties replaced with verifiable computation;
inefficient monolithic services replaced with P2P algorithmic markets."

--- filecoin white paper. 2017.

Failures of traditional P2P file systems

- **Traditional P2P systems, such as BitTorrent or Gnutella, are notorious for their unfairness and lack of security**
  - people are hesitated to contribute their storage resources to or to store their data in the P2P systems
- **No enforced contribution/incentive mechanism**
  - there is no motivation to share services or resources
  - leads to the “free-rider” or the “leecher” problems
- **No enforced security and privacy protection**
  - traditional P2P systems do not offer strong protection of data
  - data stored in the P2P systems are not encrypted by default

Secure P2P file systems

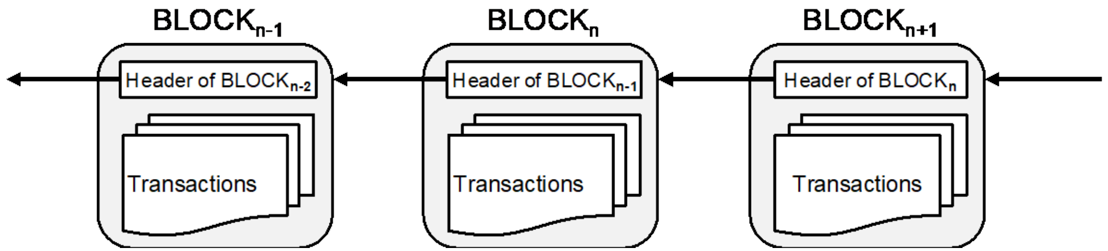
- **The advent of the blockchain brings a new P2P platform**
  - blockchain can enforce the fairness for providing resources/services and receiving respective payments, such as IPFS, Storj, Sia, etc.
- **Strong data protection in the untrusted P2P environment**
  - adopt end-to-end encryption of data, Storj and Sia
- **However, encryption makes search over encrypted data difficult**
  - files can only be accessed via their identifiers, such as in Storj

STORJ Sia Filecoin IPFS

Objective: design a blockchain-based P2P storage system with encrypted search capability

Blockchain technology and PoW

- **Blockchain: a distributed ledger (database) that records all the transactions occurred in the P2P network**
  - all participants in the network hold the same copy of the chain



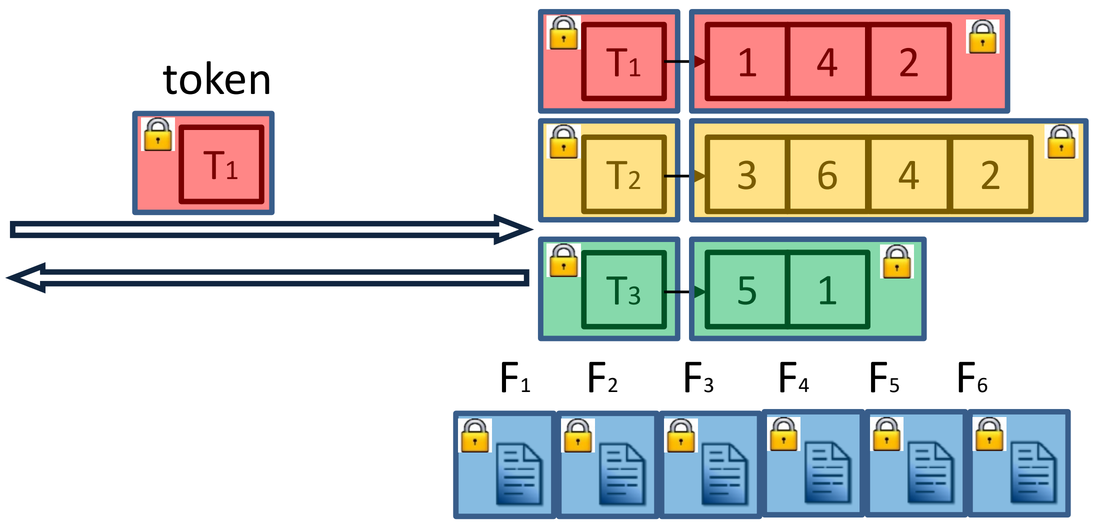
- **Proof-of-Work (PoW): a distributed consensus algorithm**
  - miners (blockchain peers) compete with each other to solve a cryptographic puzzle (Z) in order to generate a new block

$$\boxed{\text{SHA-256\^{}2}(Block_{prev} \,||\, Mr(Tx) \,||\, T \,||\, ticket) \overset{?}{<} Z}$$

- **PoW consumes massive computing resources with no useful value**
  - bitcoin’s annual carbon footprint is bigger than Switzerland’s

Searchable encryption

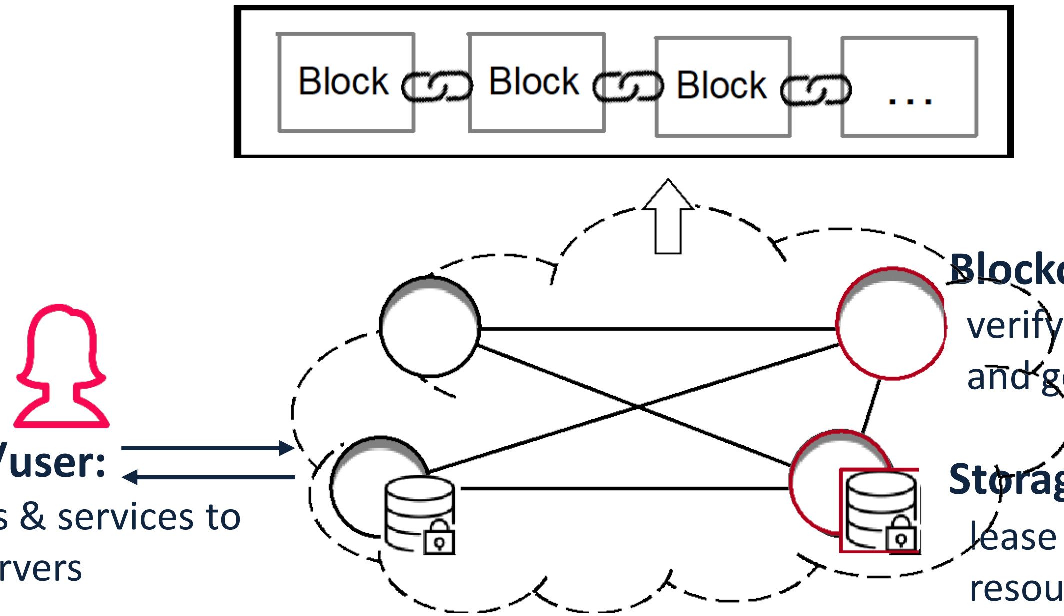
- **Searchable encryption: search encrypted files without decryption**
  - facilitates end-end encryption in client-server model
- **An example of searchable encryption**
  - data owner constructs an encrypted search index and store both the encrypted files and indexes to the server
  - user sends encrypted keyword (trapdoor) to the server to search
  - leaks no more information than search patterns



Challenges in the design of blockchain-based P2P storage systems

- **Support efficient search capability over encrypted files**
  - on-chain encrypted search and on-chain verification of search results
- **Enable file updates with forward-security**
  - efficient file update with strong security: forward-security
- **incorporate the data auditing into proof-of-work (PoW)**
  - make some useful value of PoW

Framework of a blockchain-based P2P storage system



Data owner/user: outsource files & services to the storage servers

Blockchain peers: verify the correctness of transactions and generate new blocks

Storage servers: lease out computing/storage resources to users

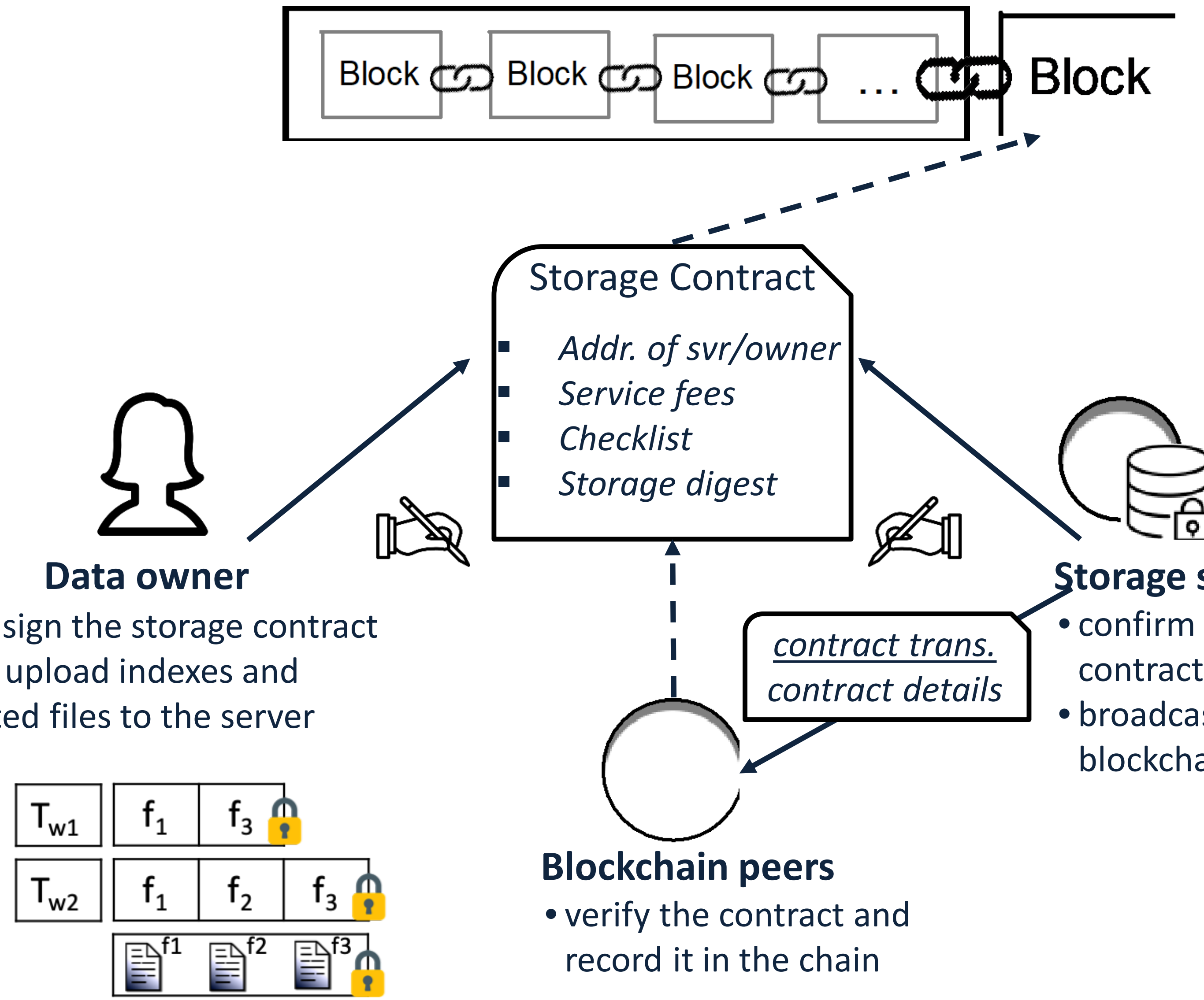
P2P storage network

A secure and fair platform for people to lease computing resources and for users to receive services

Framework of a blockchain-based P2P storage system (cont'd)

- **Blockchain P2P network consists of storage servers and peers**
  - storage servers can be peers
- **Data owners/users interact with storage servers via transactions**
  - data owners bind with servers via smart contracts
  - data and search indexes are stored off-chain at storage servers
  - all operations between owner/user and server are via transactions
    - contract transactions, data search/update transactions, etc
- **Peers verify correctness of transactions and generate new blocks to the blockchain**

Signing a storage contract



Data owner

- draft & sign the storage contract
- build & upload indexes and encrypted files to the server

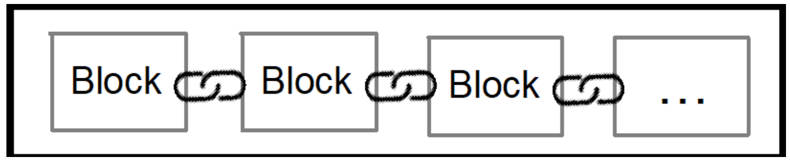
Storage server

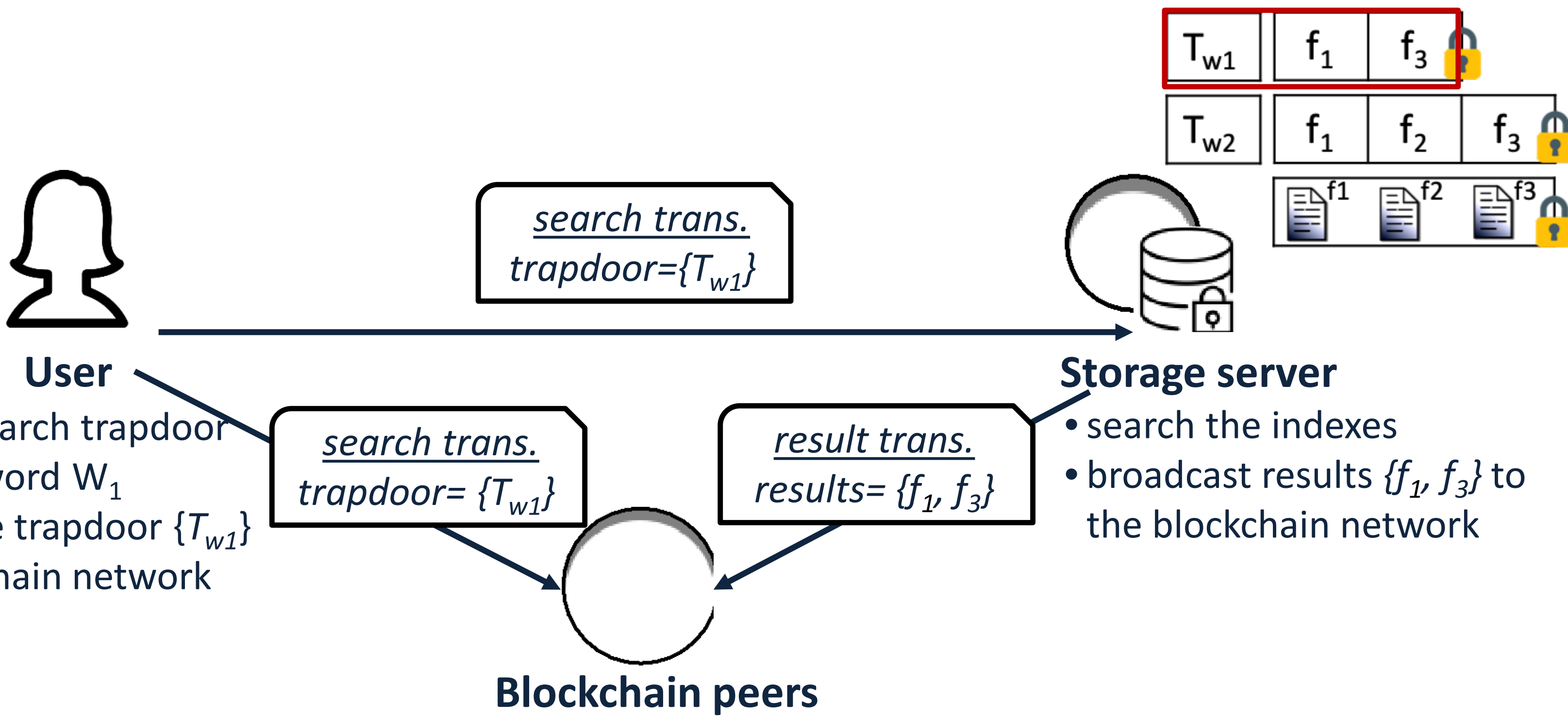
- confirm & sign the storage contract
- broadcast the contract to blockchain network

Blockchain peers

- verify the contract and record it in the chain

Search transaction





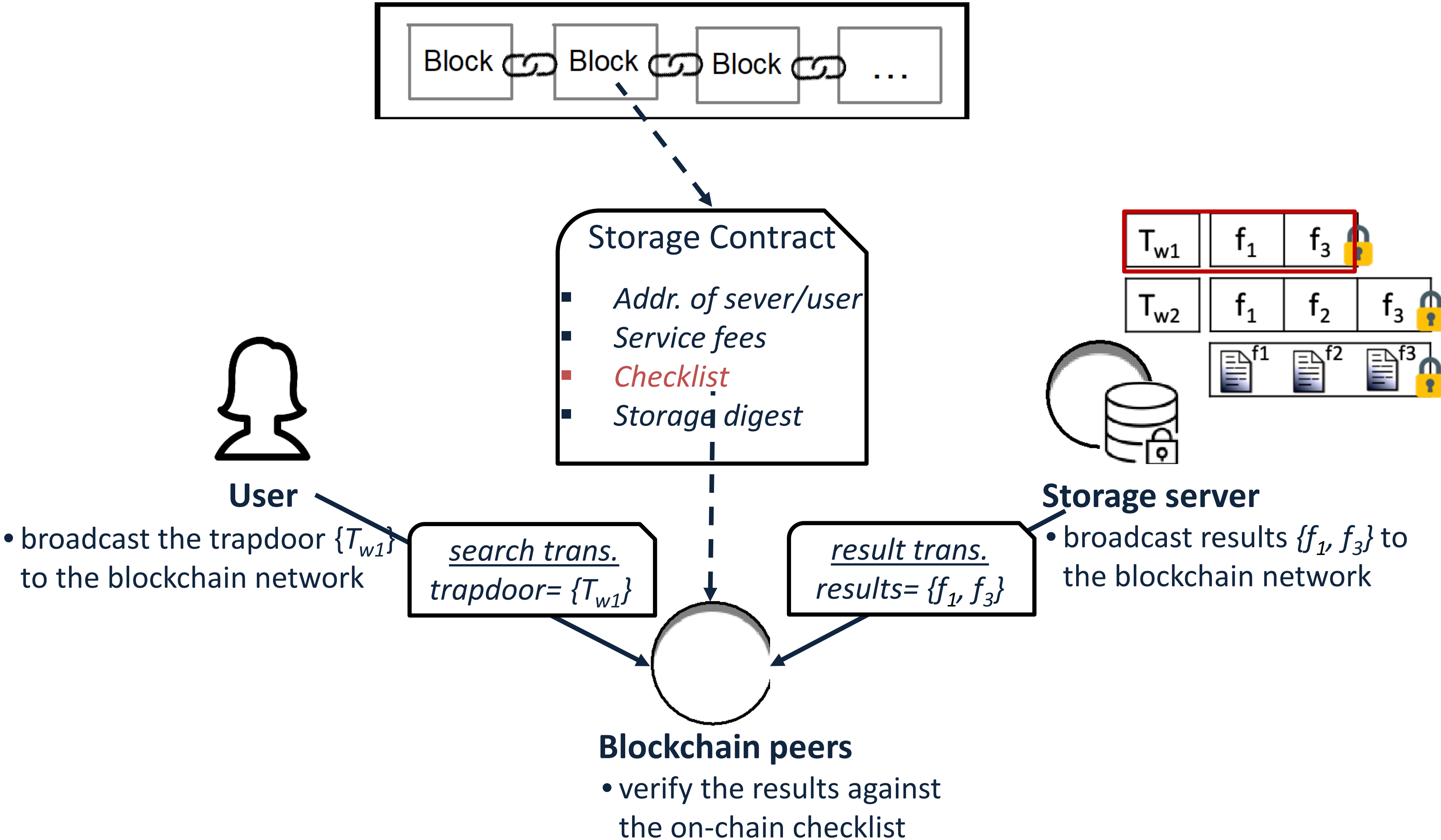
User

- generate a search trapdoor $\{T_{w1}\}$ for keyword W_1
- broadcast the trapdoor $\{T_{w1}\}$ to the blockchain network

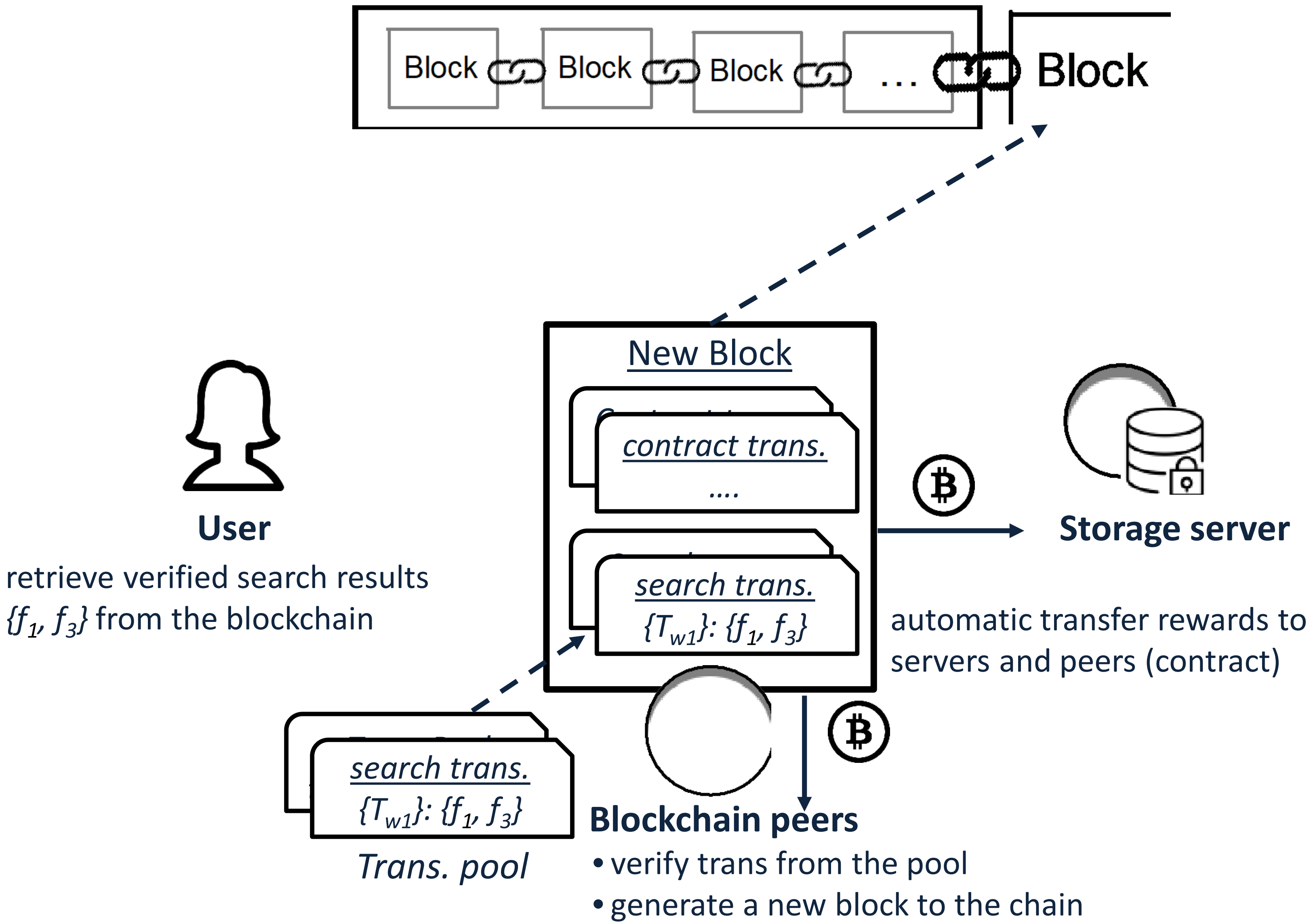
Storage server

- search the indexes
- broadcast results $\{f_1, f_3\}$ to the blockchain network

Search result verification



Generating new blocks to the blockchain



Key techniques

- **Efficient verification of search results on the blockchain**
  - design an efficient and secure on-chain checklist
- **Efficient file updates with forward security**
  - design a file index structure that can efficiently support both file update and search operations
- **Data integrity auditing: a useful proof-of-work (PoW)**
  - design a data auditing protocol that is incorporated into PoW

Verifying search results in P2P systems

- **Verification of correctness usually requires a third-party authority to make an unbiased judgement**
  - storage servers may not return a complete or accurate set of search results for saving computational cost or other reasons
  - vice versa, users may mis-accuse the honest servers in order to deny the payment
- **However, there is no central authority in the P2P network**
  - rely on peers to verify search results in the blockchain network

An efficient on-chain checklist

- **Store pre-defined search results on the blockchain for verification**
  - verification of search results can be done by any peers
- **Explore the incremental set hash technique to reduce the size of the checklist on blockchain**
  - incremental set hash can map multi-sets of arbitrary sizes to fixed length strings
  - generate one hash digest (4 bytes) for each pre-defined search result

kwd	checklist
w_1	f_1, f_3
w_2	f_1, f_2, f_3

Compressed checklist →

kwd	compressed checklist
$H(w_1)$	$H(f_1 \cup f_3)$
$H(w_2)$	$H(f_1 \cup f_2 \cup f_3)$

A secure on-chain checklist

- **However, simply using this on-chain checklist would leak result distribution, leading to inference attacks**

kwd	compressed checklist
$H(w_1)$	$H(f_1 \cup f_3)$
$H(w_2)$	$H(f_1 \cup f_2 \cup f_3)$
$H(w_3)$	$H(f_1 \cup f_2 \cup f_3)$

attacker's view
kt6gUXGWgL
TRxZDzVYjV
TRxZDzVYjV

Match them up

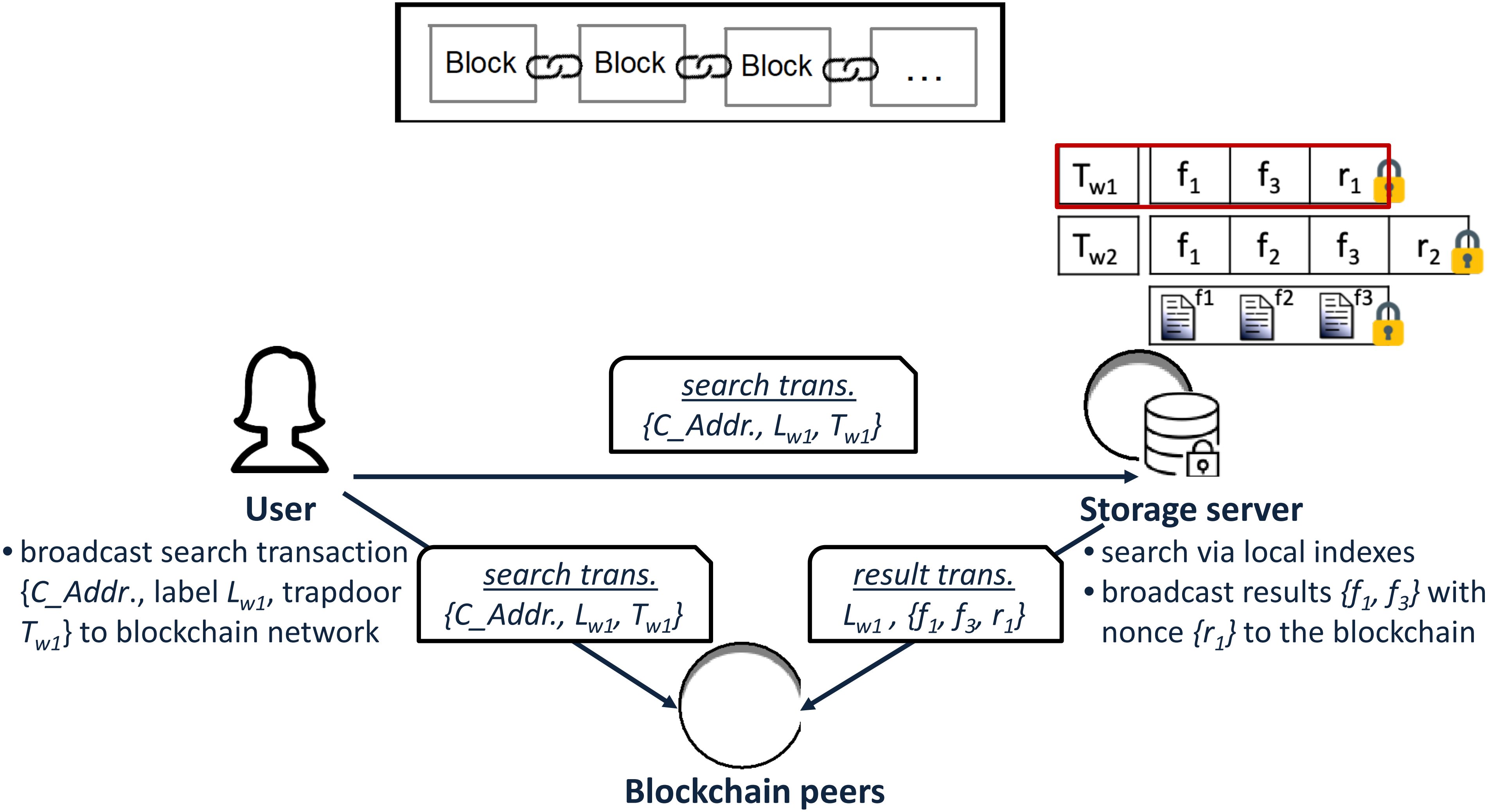
auxiliary info.	
f_1, f_3	w_1
f_1, f_2, f_3	w_2
f_1, f_2, f_3	w_3

- **Embed random masks (nonce r) into compressed checklist to hide the result distribution**

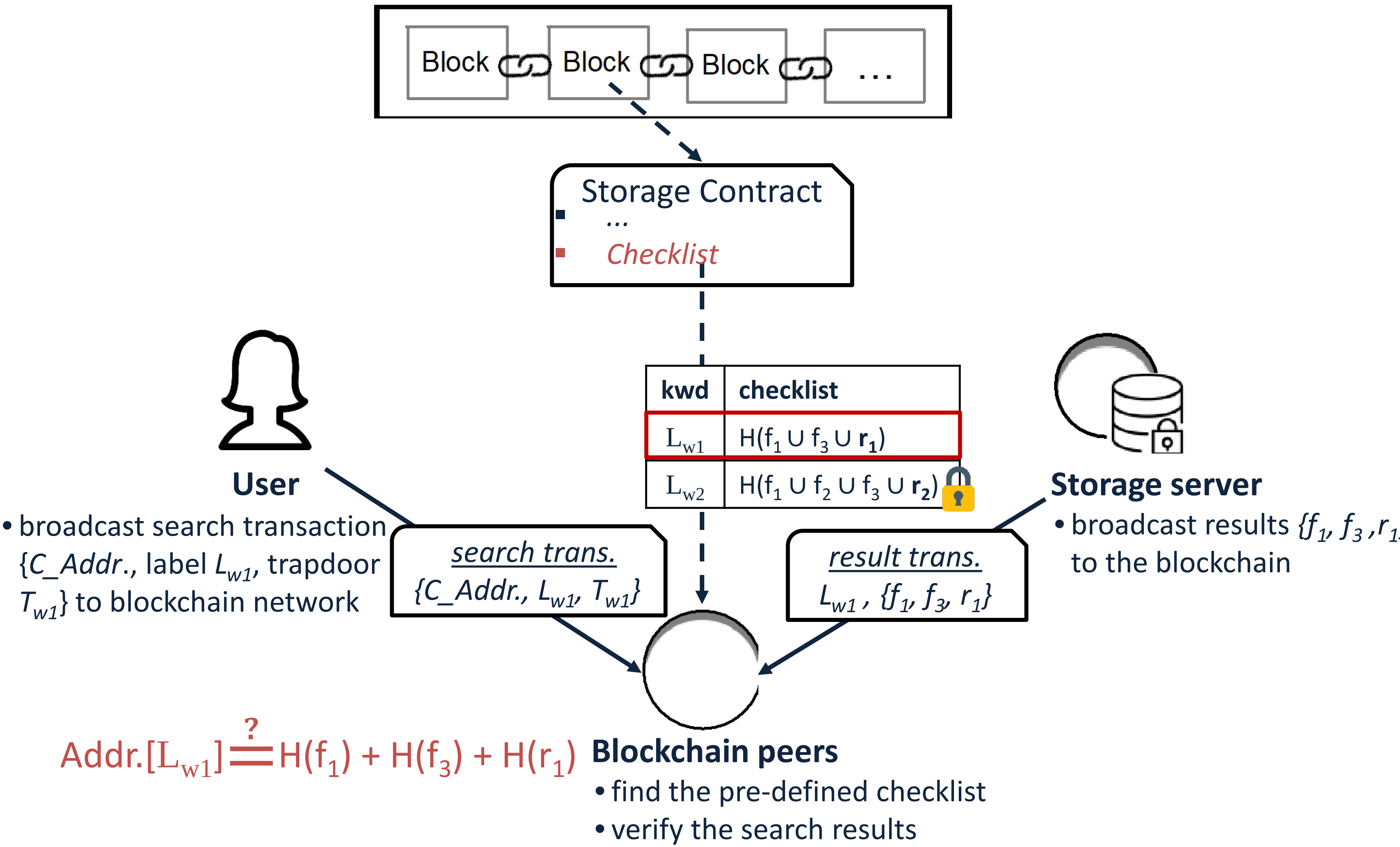
kwd	compressed checklist
$H(w_1)$	$H(f_1 \cup f_3 \cup \mathbf{r_1})$
$H(w_2)$	$H(f_1 \cup f_2 \cup f_3 \cup \mathbf{r_2})$
$H(w_3)$	$H(f_1 \cup f_2 \cup f_3 \cup \mathbf{r_3})$

attacker's view
XGkt7gUW8A
LKGM8EUnGd
IgDwwF64cl

Search transaction with result verification



Verifying search results with on-chain checklist



File updates in blockchain-based systems

- **Blockchain is “append-only” and the data stored on the chain cannot be modified**
  - store encrypted indexes off the chain
    - make the blockchain light-weighted
- **Two issues for file updates: efficiency and security**
  - the efficiency of search often conflicts with the efficiency of update in encrypted search
  - achieve forward-security for file updates

Index structures for encrypted search

secure index structures

Direct index (file index)	Inverted index (keyword index)

File IDs	**Keywords**
$H(f_1)$	$Enc(w_1)$; $Enc(w_2)$
$H(f_2)$	$Enc(w_2)$

- file-update time complexity: sub-linear
- kwd-search time complexity: linear

Keywords	**File IDs**
$H(w_1)$	$Enc(f_1)$
$H(w_2)$	$Enc(f_1)$; $Enc(f_2)$

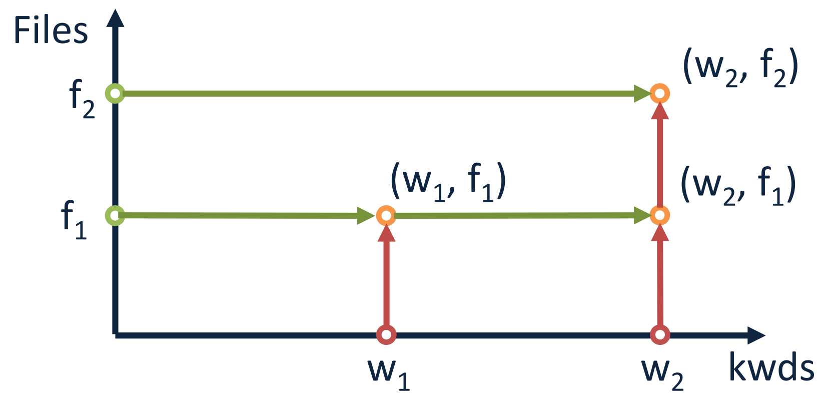
- kwd-search time complexity: sub-linear
- file-update time complexity: linear

How to design an index structure efficient for both search and update operations?

File indexes vs. keyword indexes

- **Relationship between file indexes & keyword indexes**

kwd	file IDs
w_1	f_1
w_2	f_1, f_2



- **To make both search and update efficient, we need**
  - keep both file and keyword indexes (dual indexes)
    - optimal search and update complexity: O(1)

Building encrypted dual indexes

- Extract each keyword-file pair (w, f) from the original data set
- Assign an index pointer (ptr) to each keyword-file pair

Original DB

Kwd	file IDs
w_1	f_1
w_2	f_1, f_2

Indexing pointer

KV pairs

ptr_1	w_1	f_1
ptr_2	w_2	f_1
ptr_3	w_2	f_2

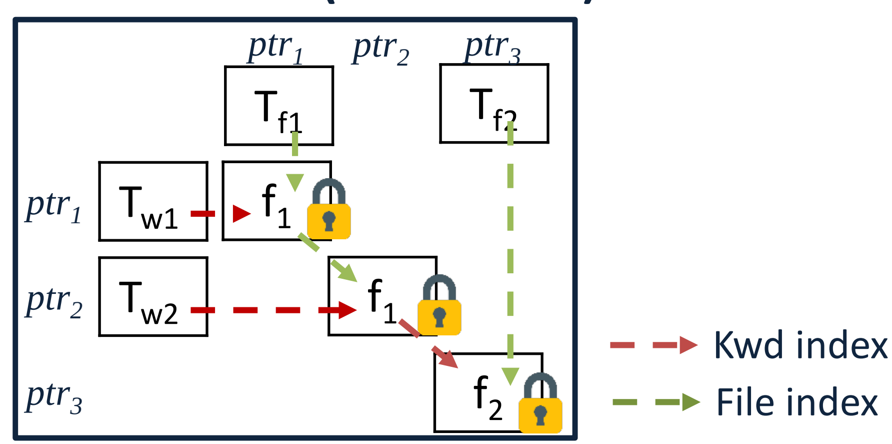
- Build the dual indexes (kwd index and file index) and store at the server-side
- Keep the local kwd and file states (pointers to the dual-index entries)

Local states (pointers)

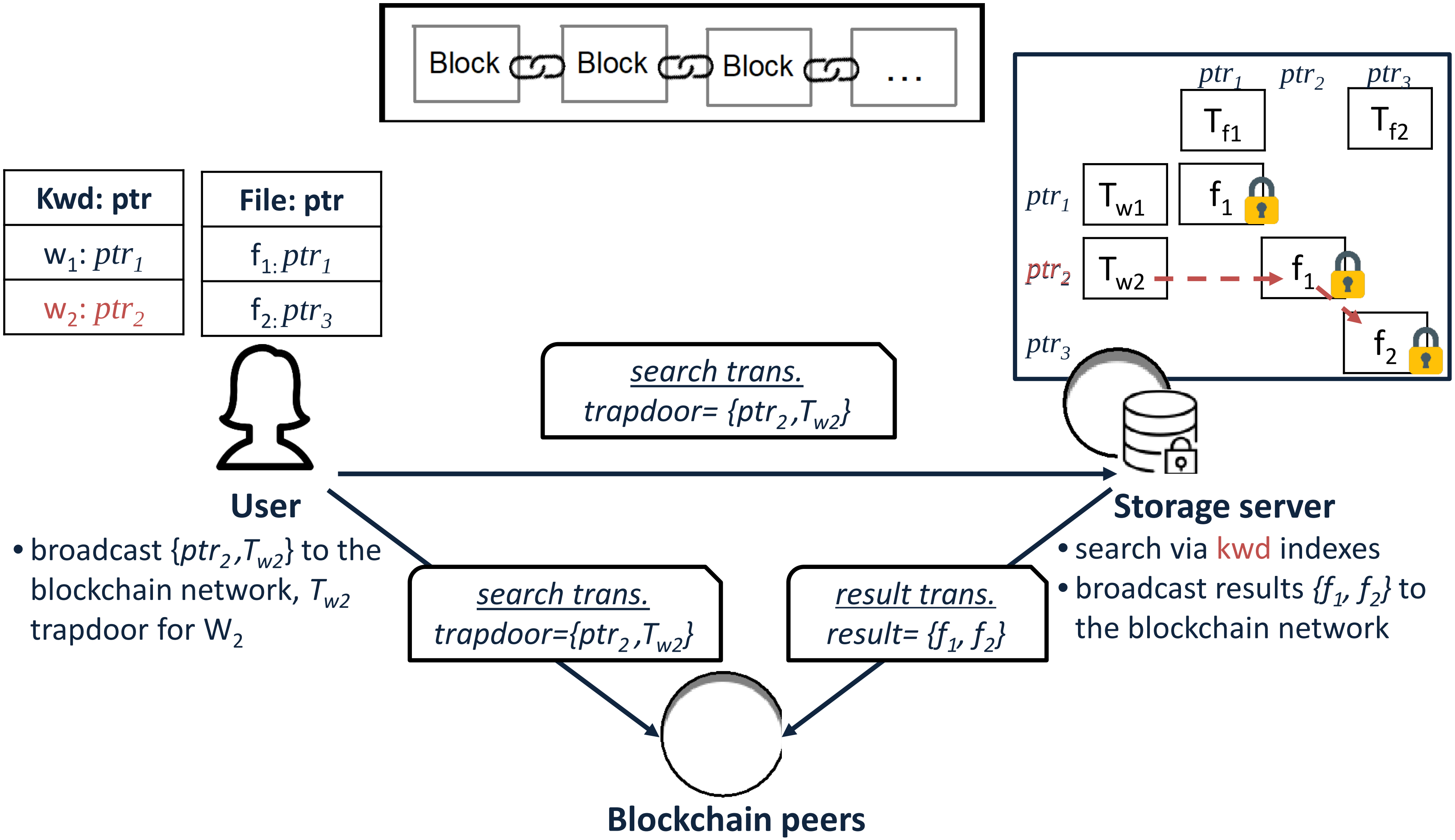
Kwd: ptr
w_1: ptr_1
w_2: ptr_2

File: ptr
$f_{1:}$ ptr_1
$f_{2:}$ ptr_3

Dual indexes (server-side)

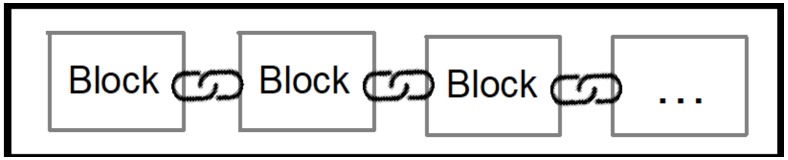


Search by using dual indexes



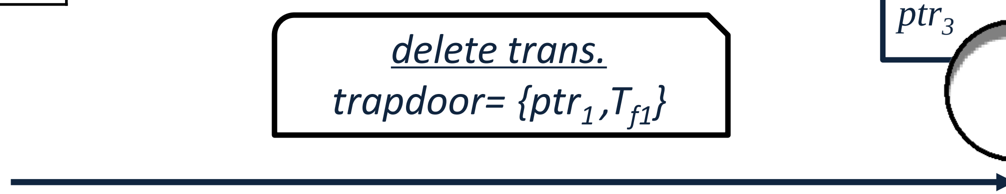
O(1) complexity for encrypted search

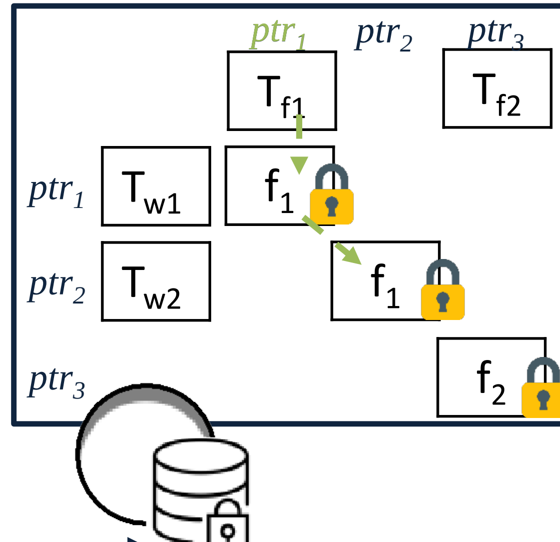
Deleting files from dual indexes



Kwd: ptr
w_1: ptr_1
w_2: ptr_2

File: ptr
f_1: ptr_1
f_2: ptr_3



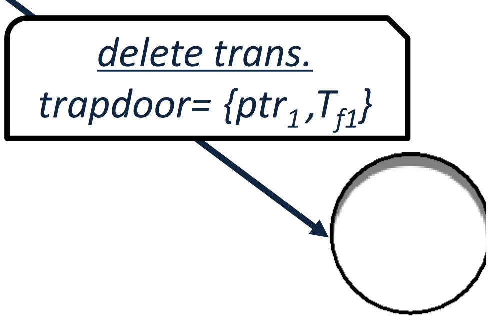


User

- generate a delete trapdoor {T_{f1}} for file f_1
- broadcast {ptr_1,T_{f1}} to the blockchain network

Storage server

- search via file indexes
- delete the file-pointers from indexes

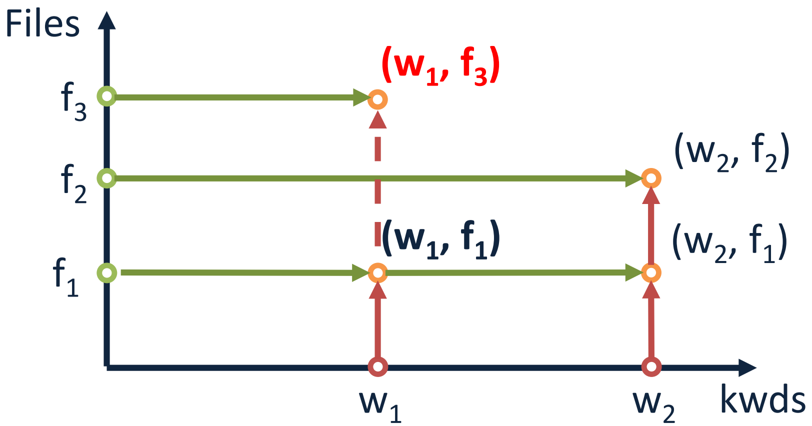


Blockchain peers

O(1) complexity for file delete

Forward security

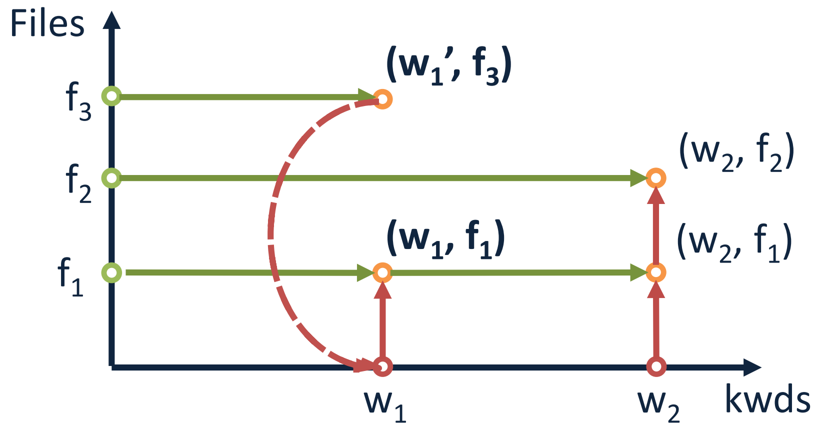
- **Forward-security** is a security property, requiring that the newly added files shall not have any link to the previous search results
- **Example:** Suppose to add the new file f_3 with pair (w_1, f_3), and w_1 was searched before, the server shall not learn that f_3 contains a previously searched kwd w_1

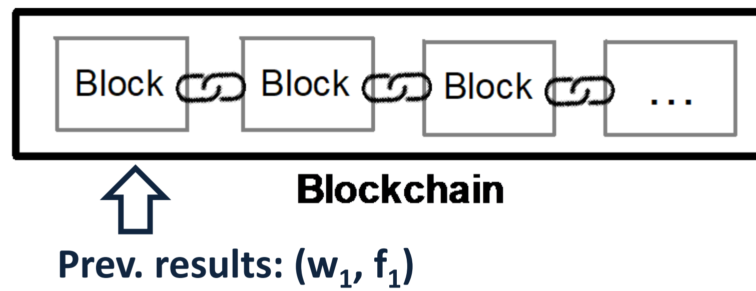


However, by directly adding new entry (w_1, f_3) to the indexes, it will reveal that f_1 and f_3 share the same kwd w_1!

Forward secure search in blockchain

- Only the keywords that are searched before may cause the forward-security problem: utilize prev. search results in the chain

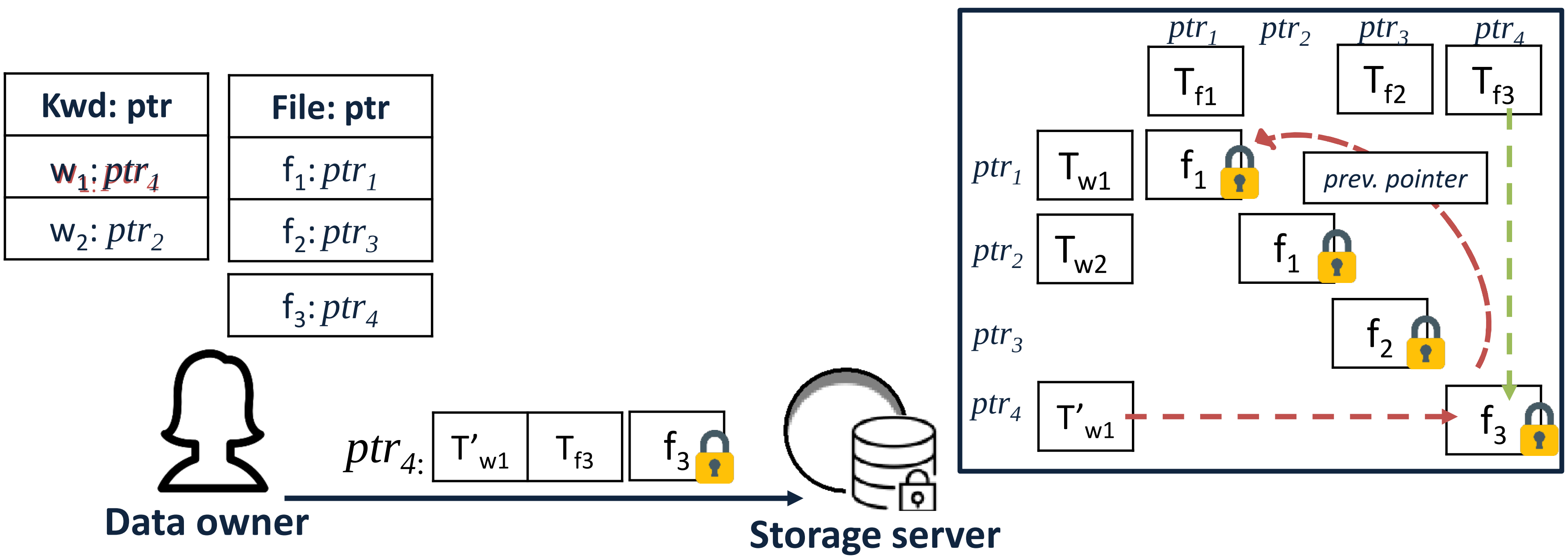




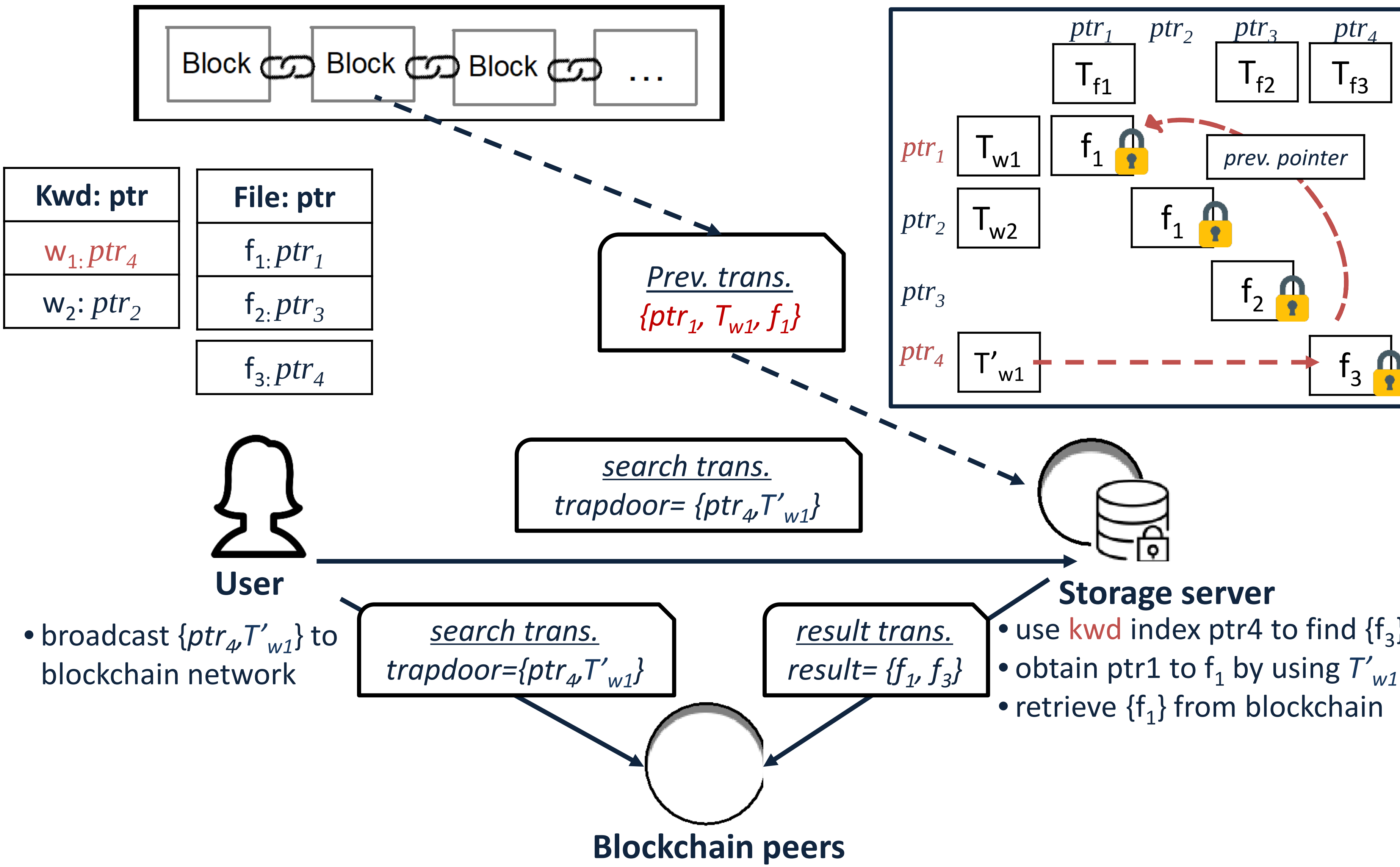
- **Add an entry with new trapdoor w_1' to the front of the kwd index chain**
  - the new entry points to the previous trapdoor (kwd) entry
- **Use the new trapdoor w_1' to search for kwd w_1**
  - update the local states to the new trapdoor
- **Retrieve the previously searched results from the blockchain**
  - previously searched results are recorded in the blockchain

Update indexes for adding a new file

- **Example:** when add a new kwd-file pair (w_1, f_3), i.e., w_1: $f_3 \rightarrow f_1$
  - add new pointer ptr_4 (f_3: ptr_4) and (w_1: ptr_1) -> (w_1: ptr_4) in the local states
  - generate a new trapdoor T'_{w1} for w_1 and request the server to add a new index-entry for T'_{w1}, which further links to f_1 with the old trapdoor T_{w1}
- Need to enter a new contract with the updated checklist

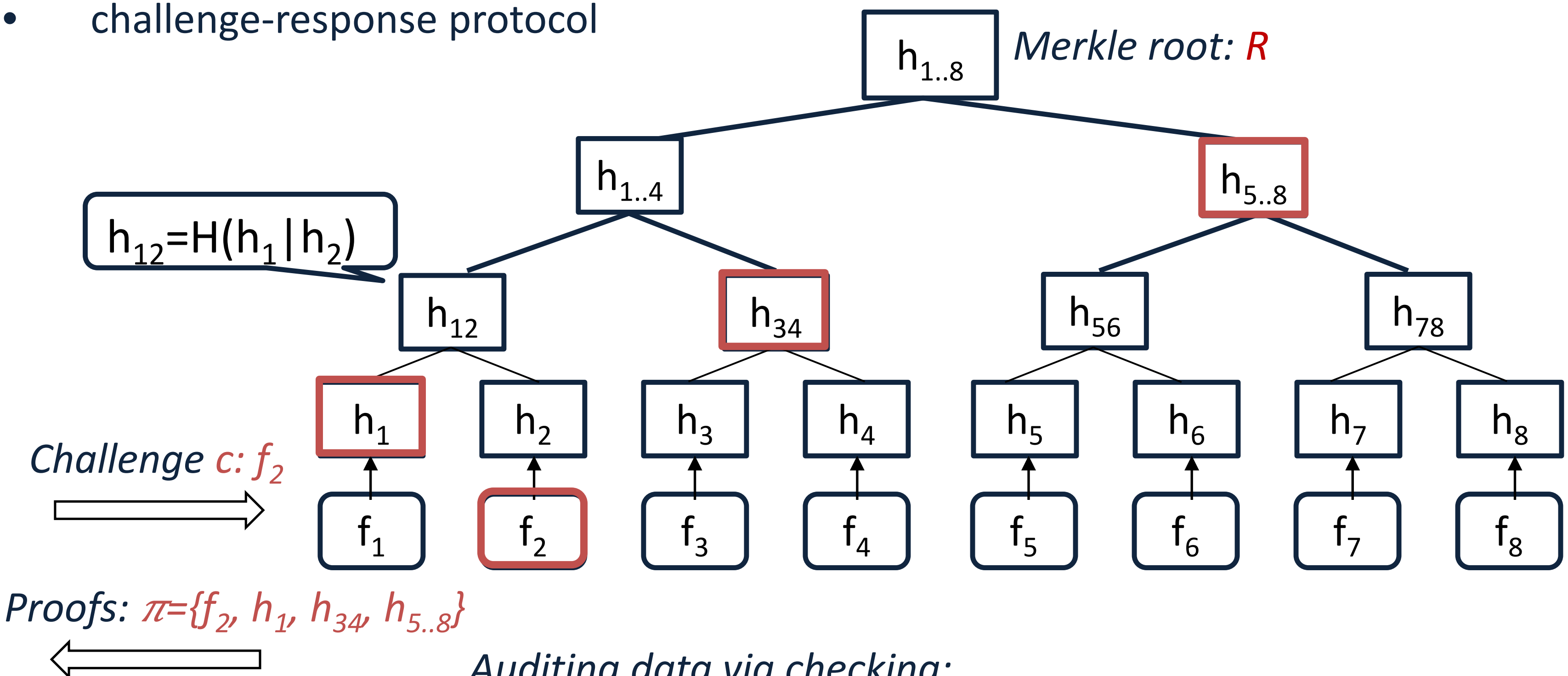


Search for new files via dual indexes



Data auditing: A useful PoW

- **Data auditing is to check the integrity of data stored at servers**
  - ensure the data at the servers are not missing, corrupted, …
- **Merkle Hash Tree for data auditing**
  - challenge-response protocol



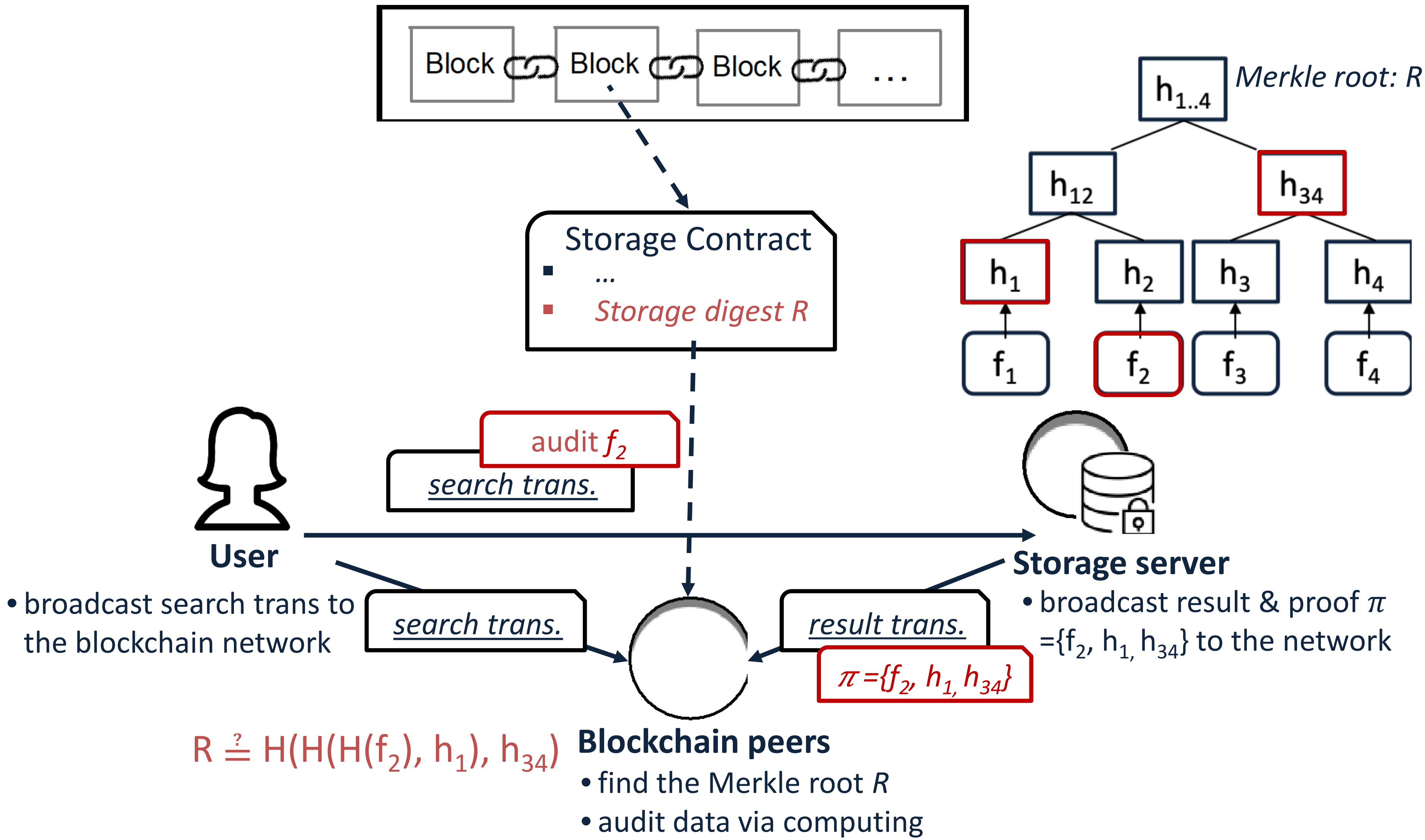
Auditing data via checking:

$$R \stackrel{?}{=} H(H(H(H(f_2), h_1), h_{34}), h_{5..8})$$

Design strategies

- **An auditing challenge is generated from each search transaction**
  - the storage server generates an auditing position (a file ID) out of the incoming search transaction
  - it broadcasts the auditing proof, together with search results
- **Data auditing is part of the PoW by the peers**
  - blockchain peers can verify auditing proofs and the search results
- **Ensure the integrity of data throughout the life time of the system**

Data auditing within the blockchain



A new consensus protocol

- **Verification of a search result transaction includes:**
  - verifying the search results, and
  - auditing the integrity of the stored file
- **Peers compete with each other to generate new blocks**

$$\boxed{\text{SHA-256}^\wedge 2(\underbrace{pk}_{\textit{peer ID}} \,||\, \underbrace{T}_{\textit{ticket}} \,||\, \underbrace{Mr(Tx)}_{\textit{transitions}} \,||\, \underbrace{H(\pi)}_{\textit{file-proofs}} \,||\, \underbrace{Block_{pre}}_{\textit{Prev. block}}) \overset{?}{<} Z \times \underbrace{B_{stc}}_{\textit{peer's stake}}}$$

$Mr(Tx)$: the Merkle-tree root of validated transactions in the new block

$H(\pi)$: the hash value of validated file-proofs

B_{stc} : the peer's stake (amount of deposit it has in the system)

A hybrid method of proof-of-stake and proof-of-work

- **Proof-of-stake gives more advantage to peers with higher stake, reducing the average time for generating a new block**
  - a trade-off between randomness and deterministic in block mining
  - increase the throughput of generating new blocks
- **Peers perform data auditing as a useful PoW**
- **The longest chain rule still holds the global consensus among the peers**

Summary

A secure and verifiable blockchain-based P2P storage system

- **support secure search over encrypted files**
  - off-chain storage of files and search indexes:
    - make the blockchain light-weighted
  - on-chain verification of search results
- **support file updates with forward security**
  - both search and update operations are in sub-linear complexity
  - leverage the property of blockchain to preserve the forward-security for file updates
- **data auditing as a useful PoW**
  - ensure data integrity in the P2P systems
  - a hybrid method of proof-of-stake and proof-of-work:
    - increase the throughput of the blockchain

Department of Computer Science

THANK YOU

csjia@cityu.edu.hk

香港城市大學
City University of Hong Kong
專業 創新 胸懷全球
Professional · Creative
For The World