Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Blockchain-enabled supervised secure data sharing and delegation scheme in Web3.0

Hongmin Gao[1], Pengfei Duan[2*], Xiaofeng Pan[1], Xiaojing Zhang[1], Keke Ye[1] and Ziyuan Zhong[1]

Abstract

Web3.0 represents the ongoing evolution of blockchain technology, placing a strong emphasis on establishing a decentralized and user-controlled Internet. Current data delegation solutions for Web3.0 predominantly rely on attribute-based encryption algorithms (ABE) but lack the essential capabilities for processing ciphertext. Additionally, the attribute-based ciphertext transformation algorithm (ABCT) falls short when it comes to verifying the transformed ciphertext provided by data proxies. The primary objective of this article is to design a fine-grained and supervised attribute-based data delegating solution tailored specifically for Web3.0. This scheme aims to enhance the ciphertext processing capabilities of existing data delegation solutions based on blockchain and ABE. Additionally, it addresses the current limitations of ABCT technology. This includes its inability to verify re-encrypted ciphertext and ensure non-repudiation of transformation results. We leverage smart contracts to ensure the automatic execution of the data delegation process and to store permanent records on the blockchain for auditing and traceability. This approach guarantees a fair distribution of interests among all stakeholders. Furthermore, we employ a commitment mechanism and digital signature to enhance the regulatory compliance of existing ABCT technology. We present a secure access control and supervised data delegation scheme for Web3.0 with blockchain along with its instantiation, emphasizing its fine-grained nature and verifiability. Finally, the evaluation results demonstrate its practicality and effectiveness.

Keywords Web3.0, Decentralized data delegation, Fine-grained data sharing, Supervised

Introduction

Data marketization refers to treating data as a resource, akin to commodities or services, enabling its free exchange, buying, and sharing in the market. It can stimulate the commercial exploitation of data, allowing organizations to generate income through data transactions [1]. This creates new revenue streams for data providers and fosters innovation and business model development. It also encourages data sharing and collaboration. Organizations can sell or share their data assets, enabling more widespread data utilization and fostering cooperation, thereby facilitating cross-industry and cross-departmental data sharing to address complex issues and challenges. It aids in more efficient utilization of data resources, as the market value of data can guide organizations in determining which data merits resource allocation for collection, storage, and analysis, as well as which data can be utilized by external parties. Furthermore, it incentivizes data providers to prioritize data quality and availability, as low-quality or unreliable data may not find favor in the market, thus impacting the market value of data [2].

Web3.0 represents the further evolution of blockchain technology, emphasizing a decentralized, distributed, and user-controlled Internet. In the Web3.0 environment, data delegating has also undergone a transformation,

*Correspondence:
Pengfei Duan
pengfeiduan@bupt.edu.cn
[1] China Mobile Information Technology Company Limited, Beijing 100037, China
[2] School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China

placing greater emphasis on user privacy, security, and data ownership. Web3.0 underscores users' rights to control their data and protect their privacy [3]. Data delegating services can offer users complete control over their data, allowing them to determine how to use, share, and safeguard their information, thereby delivering value to users. Decentralized applications (DApps) within the Web3.0 ecosystem require reliable data storage and management. Data delegation can provide stable storage solutions for DApps, assisting developers in building feature-rich applications within a distributed environment [4].

ABE is a cryptographic algorithm that seamlessly integrates precise access control with robust privacy protection [5]. It designs access control structures based on user attributes and trust relationships, creating encryption and decryption primitives. The successful decryption of ciphertext occurs exclusively when the decryptor's attributes correspond with the access structure, thereby ensuring granular control over data access. However, existing data delegation solutions based on attribute-based encryption fail to achieve flexible ciphertext data sharing and lack the capability to process ciphertext data, resulting in limited practicality [6].

Ciphertext transformation is a cryptographic technique developed to facilitate flexible sharing and delegated access to encrypted data. This method permits the data owner to empower a third party, known as a proxy, to re-encrypt the data into an alternative format. Consequently, this re-encrypted data can be decrypted by authorized entities, ensuring the original encryption and private keys remain confidential. This strategy enables data owners to maintain control over data access without compromising their private keys, thus allowing adaptable sharing and efficient management of access to the data [7]. The ciphertext transformation algorithm serves as an effective tool for data protection and access control, enabling secure and efficient sharing of encrypted data among diverse users while simultaneously diminishing the complexities and risks associated with key management.

Therefore, attribute-based ciphertext transformation (ABCT) technology enables fine-grained sharing of ciphertext data. By introducing data processors, ABCT offloads computationally intensive algorithms to resource-intensive servers, significantly enhancing the practicality of the scheme. However, it lacks the ability to verify the transformed ciphertext, allowing malicious data processors to send unrelated modified ciphertext to data requesters without performing the transformation computation, resulting in a detriment to the interests of data recipients. Conversely, legitimate data processors, even after correctly completing the re-encryption computation, can be repudiated by malicious data recipients, thereby refusing to pay the associated fees.

Motivation. Current data delegation solutions that utilize blockchain and ABE fall short in their ability to handle encrypted data. While attribute-based ciphertext transformation technology possesses the aforementioned ciphertext processing capability, it falls short in the verification of the transformation ciphertext, allowing malicious proxies to return fabricated, unrelated results to data requesters without performing the transformation computation, thereby harming the interests of data recipients. Similarly, even if honest data processors correctly execute the ciphertext transformation operation, malicious data recipients can disavow the process and levy accusations against data processors, consequently refusing to pay the associated fees. To tackle these challenges, this paper makes the following contributions:

- This paper addresses the shortcomings of current solutions in offering detailed access control and sharing options for ciphertext data within data delegation platforms. It achieves this by integrating Attribute-Based Ciphertext Transformation (ABCT) technology with smart contracts, creating a reliable data custody and secure sharing methodology specifically designed for Web3.0, characterized by its granular control features.
- To overcome the inadequacies of existing ABCT technology, which lacks the verifiability of the modified ciphertext, this invention employs hash commitment mechanisms and digital signature to enhance the verifiability of existing ABCT technology.

Related work

Decentralized content delegation

In order to solve the scalability and privacy security issues faced by the current blockchain, the use of off-chain storage is the current mainstream solution. This method not only solves the above problems, but also retains the decentralization advantages of the blockchain. In these approaches, the actual files are stored off-chain, while only timestamps and data digests are maintained on the blockchain. Among them, when storing outside the chain, these solutions usually choose Storj [8], SIA [9], Filecoin, IPFS [10], Dat [11], etc. as decentralization point-to-point system.

As open-source platforms for decentralized storage, both IPFS [10] and Dat [11] utilize blockchain technology and cryptocurrency to incentivize users for file storage and sharing. To facilitate efficient decentralized storage and content distribution, each has developed its own unique protocol. Swarm is closely integrated with the Ethereum blockchain ecosystem

and focuses on providing efficient decentralized storage solutions for DApps and data on Ethereum. Compared with Dat [11] which focuses on hosting large files, IPFS implements a more general P2P network protocol. However, due to the lack of incentive mechanism of IPFS and the lack of support for cryptocurrency, data hosting solutions based on IPFS cannot provide reliable data delegating [12].

To address these limitations, Filecoin has introduced an open-source cryptocurrency mechanism built upon IPFS, incentivizing users to contribute their unused storage space. Filecoin aims to transform the Internet's infrastructure by interconnecting computing devices and supplanting the traditional HTTP protocol with a shared file system. Moreover, numerous existing blockchain platforms have integrated IPFS to realize efficient off-chain data storage. This integration significantly influences the linkage and searchability of current online content, as noted in [13].

Attribute-based data sharing

Recently, attribute-based encryption (ABE) has gained significant prominence due to its capability to enforce precise access control over encrypted data [14]. ABE is differentiated into two primary categories based on the generation methods of secret keys and ciphertexts: Ciphertext-Policy ABE (CP-ABE) [15] and Key-Policy ABE (KP-ABE) [16]. CP-ABE was employed for precise sharing of health information in a study by Ibraimi et al. [17]. Similarly, ABE-based data-sharing frameworks are also elaborated in the research by Chen et al. [18] and Barua et al. [19]. However, these aforementioned schemes lack the efficiency needed to share encrypted data, which is crucial in collaborative scenarios.

In 1998, Blaze et al. introduced the concept of ciphertext transformation [20], a mechanism that enables a proxy to utilize a ciphertext transformation key to transform ciphertext created by one user into a format that can be decrypted by another user. This approach enables data owners to share ciphertext with others without revealing the underlying plaintext to proxies. In the beginning, traditional PRE schemes were primarily designed for one-to-one access delegation [21]. Subsequently, in order to extend the above solution to many-to-many access delegation scenarios, Liang et al. [22] used attribute-based encryption to enable more flexible implementation of data access control. On this basis, people have proposed many ABCT solutions, such as enhancing the expressive ability of access policies [23–25], proposing enhanced security models, etc., [26, 27].

Preliminary

This section outlines key cryptographic constructs and establishes the following notations: U signifies the attribute universe, U_Θ represents the universe of attribute authorities, and *GID* denotes the universe of users' global identifiers.

Decentralized ID

The decentralized digital identity system, known as a Decentralized Identifier (DID), employs technologies such as blockchain to establish authentic ownership and control over digital identities. DID fundamentally returns individual privacy and data control to each person, enabling true ownership and control of information content. Key features of DID are as follows:

- Decentralization: DID authentication and authorization operate independently of centralized entities, putting users in direct control. This preserves user data privacy and autonomy.
- Enhanced Security: Through the utilization of distributed ledger technologies like blockchain, DID scatters users' identity data across multiple nodes, thwarting attempts to compromise identity by targeting a single point. This fortifies the security of the identity.
- Verifiability: DID employs technologies like digital signatures for identity verification, ensuring identity claims are supervised.
- Interoperability: DID supports interoperability with other identity verification systems, facilitating seamless identity authentication and authorization across various platforms.

Bilinear pairing

It is a cryptographic operation that maps two elements from two different elliptic curve groups to a third group, typically denoted as $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$, where $\mathbb{G}_1$ is a group of elliptic curve points, and $\mathbb{G}_T$ is a multiplicative group of finite field elements.

This map has two key properties: bilinearity and non-degeneracy. These properties allow bilinear pairings to be used in advanced cryptographic applications. Pairings provide a way to efficiently verify relationships between elements in different groups, a task that is difficult to accomplish in traditional elliptic curve cryptography. This makes bilinear pairings a powerful tool in the field of cryptography, enabling new types of cryptographic protocols that were previously not feasible.

q-parallel BDHE hypothesis

The q-parallel Bilinear Diffie-Hellman Exponentiation (BDHE) Hypothesis is a concept in the field of cryptographic security, particularly in the area of pairing-based cryptography. Bilinear Diffie-Hellman Exponentiation (BDHE) is an extension of the Diffie-Hellman problem, which is a fundamental concept in public-key cryptography.

Let $(e, \mathbb{G}, \mathbb{G}_T, g, p)$ be a bilinear pairing and randomly choose $a, r \in \mathbb{Z}_p$. If the adversary is given an instance:

$$\vec{y} = g, g^r, g^a, ..., g^{a^q}, g^{a^{q+2}}, ..., g^{a^{2q}},$$
$$\forall j \in [1, q] \ \ g^{r \cdot b_j}, g^{a/b_j}, ..., g^{a^q/b_j}, g^{a^{q+2}/b_j}, ..., g^{a^{2q}/b_j},$$
$$\forall j, t \in [1, q], t \neq j, \ \ g^{a \cdot r \cdot b_t/b_j}, ..., g^{a^{q+1} \cdot r \cdot b_t/b_j},$$

it is hard to distinguish $e(g, g)^{r \cdot a^{q+1}}$ from a random value $T \in \mathbb{G}_T$. Define:

$$Adv_{\mathcal{A}} = |Pr[\mathcal{A}(\vec{y}, e(g, g)^{r \cdot a^{q+1}}) = 1] - Pr[\mathcal{A}(\vec{y}, T) = 1]|$$

as the advantage of PPT adversary A in solving the decisional q-parallel BDHE hypothesis. The decisional q-parallel BDHE hypothesis holds if A cannot solve it with non-negligible advantage. The q-parallel BDHE hypothesis suggests that solving multiple instances of the BDHE problem simultaneously is computationally infeasible, making it a solid foundation for secure cryptographic protocols [28].

Attribute-based ciphertext transformation

The attribute-based ciphertext transformation (ABCT) scheme comprises the following five algorithms:

- $ABCT.Init(\lambda, U)$: The initialization procedure requires a security parameter λ and an attribute universe U, leading to the creation of a key pair (mpk, msk).
- $ABCT.KCrt(msk, S)$: The personal key creation process employs msk alongside an attribute set AS to generate a unique private key sk for user i.
- $ABCT.Encr(m, (\mathbb{M}, \pi))$: The process of securing a message involves taking a message m and an access structure $(\mathbb{M}, \pi)$, resulting in a encrypted message CTX.
- $ABCT.TransKCre(sk, (\mathbb{M}', \pi'))$: The transformation key creation algorithm operates with an individual's secret key sk and a revised access control rule $(\mathbb{M}', \pi')$ to create the transformation key tk.
- $ABCT.TransE(tk, CTX)$: The process of message transformation utilizes the transformation key tk and CTX, leading to a modified ciphertext CTX'.
- $ABCT.Decr_{or}(sk, CTX)$: The CTX decoding procedure requires the secret key sk and encrypted message to retrieve the original message m.
- $ABCT.Decr_{re}(sk, CTX', CTX)$: The altered ciphertext decoding process takes the secret key sk, modified ciphertext CTX', and the encrypted message CTX as input, outputs the message m.

In summary, these components define a system that allows for the encryption of ciphertexts based on the user's attribute set and for authorized access, as well as supports the proxy re-encryption of ciphertexts, enabling them to be forwarded or shared according to new or altered access policies.

System architecture and threat model

This section introduces the data delegating system architecture for for Web3.0 and threat model.

System architecture

From the perspective of system entities, this paper considers six categories: data owners, Data requesters, data processors, central authority, blockchains, and data custody platforms. Among these, the central authority is responsible for system initialization. It issues DID certificates based on user profiles and generates attribute-related keys from the provided attribute sets. Data owners are tasked with encrypting personal data and uploading it to a data delegating platform. Subsequently, they place data digests and related information on the blockchain for Data requesters to access. Data requesters discover data that meets their attributes by browsing on-chain data digests. They then obtain the required data by decrypting the re-encrypted ciphertext returned by data processors. Data processors are responsible for re-encrypting ciphertext data on the data delegating platform using re-encryption keys. This process enables flexible switching of data access policies while achieving one-time encryption, multiple-time sharing.

The blockchain establishes a decentralized and trustworthy environment for data circulation among users. Deploying smart contracts on the blockchain ensures that the data exchange among parties is meticulously recorded, offering a robust foundation for subsequent accountability and traceability. Web3.0-oriented data delegating platforms emphasize decentralization, granting users better control over their data. Data is no longer concentrated on a single centralized server but distributed across different nodes on the network. This approach enhances user data sovereignty. Moreover,

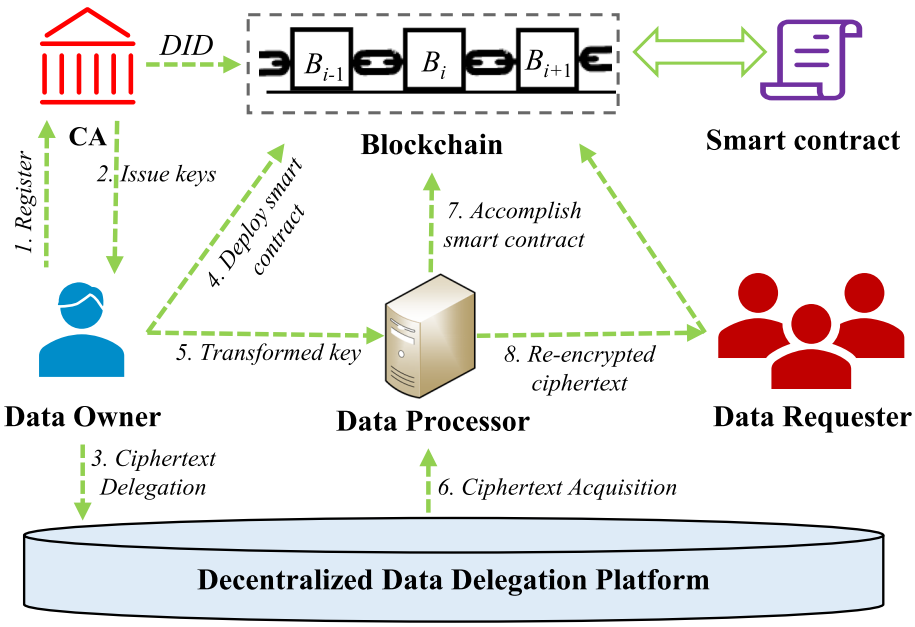

Fig. 1 The System Model

Web3.0-oriented data delegating platforms offer improved data privacy and security. The system model is depicted in Fig. 1.

This paper utilizes IPFS, the InterPlanetary File System, as the decentralized platform for data delegation. IPFS is a novel and decentralized protocol aimed at establishing a peer-to-peer approach for storing and sharing hypermedia within a distributed file system. IPFS was created to surmount the limitations of the traditional HTTP protocol, thereby enhancing the web by increasing its speed, security, and openness. It has the following features: decentralization, content addressing, and integration with Blockchain. IPFS represents a significant step towards a more distributed web where users have greater control over their data and where information can be disseminated more efficiently and robustly. As such, it's an important part of the conversation about the future of internet technology, particularly in the realms of web decentralization and blockchain.

Threat model

In this paper, we consider both the central authority and attribute authorities as reliable entities, issuing keys exclusively to authorized users. The secret key issuance process occurs after the establishment of secure communication channels. Consequently, only authorized, legitimate users have access to the secret keys. The proxy, while honest, is assumed to be curious and might attempt to decipher the ciphertext content. Furthermore, we assume the possibility that system users, intrigued by the stored data, could collude with unauthorized parties in an attempt to illegitimately extract information. To assess the security of our proposed scheme, we identify two distinct types of adversaries: the first seeks to differentiate between ciphertexts, while the second endeavors to collude using their secret keys in an attempt to decrypt data that they are unable to decrypt individually.

Scheme description

First, we introduce the usage of DID. DID fundamentally restores individual privacy and data control to users, facilitating genuine ownership and command over information content. The essential stages of DID encompass the following:

- DID Establishment: Users create a unique and immutable DID on their devices or on a blockchain. A DID may encompass certain metadata, such as public keys and authorization details.
- DID Registration: Users register their self-created DID with decentralized DID registration entities or on the blockchain. This process is decentralized, devoid of any centralized institution controlling DID registration.
- DID Verification: When it becomes necessary to verify a user's DID information, validation can be conducted using the public key carried by the DID itself. Given the tamper-proof nature of DIDs, the associated public keys are also considered trustwor-

thy. Moreover, this public key can be employed for encrypting and decrypting the information carried by the DID.

- DID Authorization: DID owners can grant other users access to their DID information, with this authorization data also being included in the DID's metadata. While accessing a DID, other users must undergo authorization validation to obtain the DID's information.

Through the above steps, this scheme achieves decentralized, secure, and controllable identity verification. Next, we introduce the details of our solution, including DID registration, system initialization, user key creation, data encryption, data request, transformation key creation, ciphertext transformation, ciphertext decryption, supervision.

DID registration

Each user initiates the decentralized digital identity (DID) registration process by submitting identity information (such as personal details, ID photos, biometric data, etc.) to an authoritative institution. After verification, the authoritative institution issues a DID key pair to the user. Subsequently, the institution uploads the user's DID information onto the blockchain. In this context, a DID serves as a unique identity identifier created by the authoritative institution for each user, commonly represented as an encrypted hash value.

System initialization

The CA conducts system initialization by executing an initialization algorithm *ABCT.Init*(λ, U), generating a key pair (*sk*, *pk*), and distributing the public keys to individual users while retaining the private keys. The specific process of system initialization is shown in Algorithm 1.

Algorithm 1 System Initialization

Input: λ, U
Output: PP
Randomly select $a,\ b \in Z_p^*$;
Randomly select $h, C, \{t_U\} \in G$;
Set $msk = h^a$, $ehh = e(h,h)^a$, $HB = h^b$;
Return: $PP = (e, G, G_T, h, C, \{t_U\}, HB, ehh)$.

User key creation

Each user submits their attribute set *AS* to the central authority, which then runs *ABCT.KCrt*(*msk*, *AS*) algorithm to generate attribute-based secret keys for each user and subsequently dispatches them. The specific procedure is shown in Algorithm 2.

Algorithm 2 User Key Creation

Input: msk, AS
Output: sk
Randomly choose $s \in Z_p^*$;
Set $SK1 = h^{a+bs}$, $SK2 = h^s$;
For each attribute x in attribute set AS, compute $SK_x = t_x^s$;
Return: $sk = (SK1, SK2, \{SK_U\})$.

Data encryption

Data owners autonomously formulate access policies for their data. They generate ciphertext *CTX* for data *msg* by executing an encryption algorithm *ABCT.Encr*$(msg, (\mathbb{M}, \pi))$ and subsequently upload it to the data delegating platform. Subsequently, data owners employ a commitment mechanism to provide the commitment value CMT_{msg} for data *msg*, sign data *msg* using the DID private key to obtain a signature Sig_{msg}, and upload commitment value CMT_{msg}, and signature sig_{msg} to the blockchain along with data abstract *abstr*. To compute the commitment value CMT_{msg} for data *msg*, we utilize the Pedersen commitment scheme. The process of generating the signature here can be likened to the RSA digital signature algorithm. Ultimately, the data owner uploads the data abstracts *abstr*, commitment value CMT_{msg}, and signature Sig_{msg} to the blockchain. The detailed procedure is outlined in Algorithm 3.

Algorithm 3 Data Encryption

Input: message msg, access control rule $(\mathbb{M}, \pi)$
Output: encrypted message CTX
Set $\vec{\mu} = \{r, y_2, ..., y_n\} \in Z_p^n$ where $y_2, ..., y_n \in Z_p$;
Set $\lambda_j = \vec{\mu}\mathbb{M}_j, j \in [1, l]$ for each row $\mathbb{M}_j$ of $\mathbb{M}$;
Calculate: $ctx_{A,j} = HB^{\lambda_j} t_{\pi(j)}^{-r_j}$, where $r_j \in Z_p$;
Calculate: $ctx_{B,j} = h^{r_j}$, where $r_j \in Z_p$;
Calculate: $ctx = msg \cdot ehh^r$;
Return: $CTX = ((\mathbb{M}, \pi), C^r, ctx, \{ctx_{A,j}, ctx_{B,j}\}, h^r)$.

Data request

data requesters identify the specific data they intend to request by perusing the on-chain data attributes. Subsequently, they deploy a data request smart contract that incorporates an updated data access control rule $(\mathbb{M}', \pi')$. In this context, $\mathbb{M}'$ represents a matrix of

$\widetilde{l} \times \widetilde{n}$, and π' refers to a one-way mapping function that projects each row of $\mathbb{M}'_j$ onto certain attribute. The data request smart contract is shown below.

Algorithm 4 Data Request Smart Contract

procedure PAYMENT FUNCTION
  The data recipient declares an access control rule $(\mathbb{M}', \pi')$. Then data recipient pays fee and sets limitation time T.
  if the limitation time is valid and the proxy finishes the ciphertext transformation **then**
    distribute the fee to data owner and the proxy;
    emit a success event;
  else
    refund the amount to the data recipient;
    Emit a failure event;
  end if
end procedure

Transformation key creation

Upon learning of a new data request, the data owner locally executes the transformation key creation algorithm *ABCT.TransKGen*(*sk*, $(\mathbb{M}', \pi')$) to generate the transformation key *tk*. Subsequently, the owner employs the data processor's DID private key for encrypting *tk* and sends the generated ciphertext CTX_{tk} to the data processor. The process here draws parallels to RSA private key encryption. The specific procedure is shown in Algorithm 5.

Algorithm 5 Transformation Key Creation

Input: secret key sk, transformed access control rule $(\mathbb{M}', \pi')$
Output: transformation key tk
Randomly select $Y \in G_T$ and compute $tkA = (h^{a+bs})^Y C^{\delta}$, $tkB = (h^s)^Y$, $tkC = h^{\delta}$, where $\delta \in Z_p^*$;
For each attribute x in attribute set AS, compute $\{SK_x^Y\}$;
Set $\widetilde{\vec{\mu}}$ be a random vector and the first element is $\widetilde{r}$, compute $\widetilde{\lambda_j} = \widetilde{\vec{\mu}}\widetilde{\mathbb{M}}_j$;
Compute $tk_{A,j} = HB^{\widetilde{\lambda_j}} t_{\widetilde{\pi}(j)}^{-\widetilde{r_j}}$, $tk_{B,j} = h^{\widetilde{r_j}}$, where $\widetilde{r_j}$ is randomly picked from Z_p;
Return:
$tk = (tkA, tkB, tkC, Yehh^{\widetilde{r}}, h^{\widetilde{r}}, \{SK_x^Y\}, \{tk_{A,j}, tk_{B,j}\})$.

Ciphertext transformation

The data processor decrypts CTX_{tk} with their own DID public key to obtain the transformation key *tk*, retrieves the ciphertext data *CTX* from the decentralized data custody platform, and locally performs the ciphertext transformation algorithm *ABCT.TransE*(*tk*, *CTX*) to generate the modified ciphertext *CTX*′. Subsequently, it sends the modified ciphertext to the data requester. In concrete instantiation, the data processor decrypts CTX_{tk} using their own DID public key to obtain the transformation key *tk*, which can be analogous to RSA public key decryption. The specific calculation process is shown as Algorithm 6.

Algorithm 6 Ciphertext Transformation

Input: transformation key tk, encrypted message CTX
Output: the modified ciphertext CTX'
Consider a set W that consists of those indices w for which $\pi(w)$ is in the attribute set AS and select elements ele_w from the group Z_p^*, making the sum of the products of ele_w and $\mathbb{M}_w$ for all w in the set W equal to the vector $(1, \cdots, 0)$;
Compute $A = e(tkA, h^r)/e(tkC, C^r)$;
Compute $B = e(tkB, ctx_{A,w})e(tk_{A,w}, ctx_{B,w})$;
Set $\alpha = A/\prod B^{ele_w}$ for each w in W.
Return: $CTX' = (\alpha, Yehh^{\widetilde{r}}, h^{\widetilde{r}}, \{tk_{A,j}, tk_{B,j}\})$.

Ciphertext decryption

The data requester decrypts the modified ciphertext *CTX*′ using the *ABCT.Decr*$_{re}$(*sk*, *CTX*′, *CTX*) to generate the original message and verifies the commitments and signatures generated by the data owner, thereby confirming whether the data processor has correctly executed the re-encryption process. The specific step is shown as Algorithm 7.

Algorithm 7 Ciphertext Decryption

Input: user's private key sk, encrypted message CTX, and modified ciphertext CTX'
Output: the message msg
Consider a set W' that consists of those indices w' for which $\pi'(w')$ is in the attribute set AS and select elements $ele_{w'}$ from the group Z_p^*, making the sum of the products of $ele_{w'}$ and $\mathbb{M}'_{w'}$ for all w' in the set W' equal to the vector $(1, \cdots, 0)$;
Compute $A' = e(SK1, h^{\widetilde{r}})$;
Compute $B' = e(SK2, tk_{A,w})e(SK_w^Y, tk_{B,w})$;
Set $\alpha' = A'/\prod B'^{ele_{w'}}$ for each w' in W';
Compute $Y' = Yehh^{\widetilde{r}}/\alpha'$ and $msg' = ctx/\alpha^{1/Y'}$;
Return: msg'.

Supervision

In order to verify the correctness of the ciphertext transformation operation and to prevent malicious users from illegally accusing honest processors, we use the hash commitment mechanism and digital signature technology to generate commitment CMT_{msg} and signature values Sig_{msg} respectively in the data encryption stage. These are stored

on the blockchain along with the ciphertext *CTX*. Taking advantage of the decentralized and non-tamperable characteristics of the blockchain, we ensure that the storage results on the chain cannot be tampered with. When data users doubt the correctness of the decryption results, the CA can verify the on-chain storage results to supervise the data sharing process. In this paper, the hash commitment functions can be constructed using standard cryptographic collision-resistant hash functions like SHA-1. Specifically, when the message *msg* becomes the input for SHA-1, it can yield an output of fixed length. Due to the collision-resistant nature of the hash function, the attacker cannot find a different message to make the hash result consistent.

In all, the flow chart of fine-grained supervised attribute-based content delegation scheme is shown in Fig. 2. In this paper, the encrypted data would be stored on the IPFS. The integration of IPFS with blockchain technology offers a powerful combination that enhances data integrity and security, optimizes storage efficiency, and boosts decentralization. This synergy leverages IPFS for distributed and efficient data storage, reducing the burden on blockchain networks. It ensures immutable and verifiable record-keeping, thereby enhancing the authenticity of data. The content-based addressing of IPFS, coupled with the blockchain's verification processes, guarantees accurate data retrieval. This combination not only improves scalability and cost-effectiveness but also empowers users with greater control over their data privacy.

Security proof

In the security proof, we show that the proposed approach is confidential and collusion-resistant. Besides, the central authority can supervise the shared results to prevent deceptive behavior by malicious processors and users.

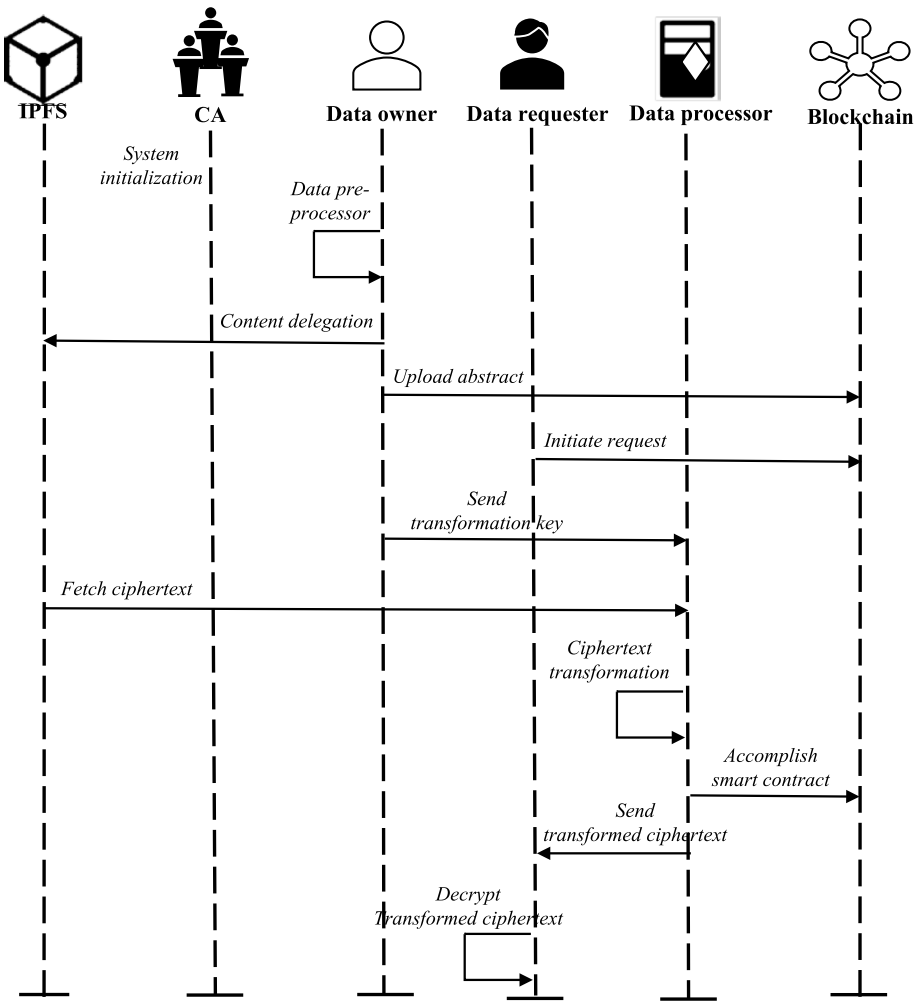

Fig. 2 Flow Chart of Our Scheme

Scheme rightness

In this part, we proof the rightness of the ciphertext decryption algorithm, the proof process is shown as Algorithm 8.

Algorithm 8 The proof of scheme rightness

$$
\begin{aligned}
A &= e(tkA, h^r)/e(tkC, C^r) \\
&= e((h^{a+bs})^Y, h^r)/e(h^\delta, C^\delta); \\
B &= e(tkB, ctx_{A,w})e(tk_{A,w}, ctx_{B,w}) \\
&= e(h^{sY}, HB^{\lambda_w} t_{\pi(w)}^{-r_w})e(tk_{A,w}, ctx_{B,w}) \\
&= e(h^{sY}, HB^{\lambda_w} t_{\pi(w)}^{-r_w})e(HB^{\widetilde{\lambda_w}} t_{\widetilde{\pi}(w)}^{-\widetilde{r_w}}, h^{r_w}) \\
\alpha &= \prod A/B^{ele_w} = ehh^{rY} \\
A' &= e(SK1, h^{\widetilde{r}}) = e(h^{a+bs}, h^{\widetilde{r}}) \\
B' &= e(SK2, tk_{A,w})e(SK_w^Y, tk_{B,w}) \\
&= e(h^s, HB^{\widetilde{\lambda_w}} t_{\widetilde{\pi}(w)}^{-\widetilde{r_w}})e(t_w^{sY}, h^{\widetilde{r_w}}) \\
\alpha' &= A'/\prod B'^{ele_{w'}} = egg^{\widetilde{r}} \\
Y' &= Yegg^{\widetilde{r}}/\alpha' = Y \\
msg' &= ctx/\alpha^{1/Y'} = msg \cdot ehh^r/ehh^{rY \cdot 1/Y'} = msg
\end{aligned}
$$

Collusion resistance

In this part, we give the evidence to demonstrate that our scheme is collusion-resistant, meaning that even if a data requester colludes with another legitimate data requester, they cannot compute the secret key *sk*. Because in the *TransKGen* algorithm of our scheme, the *tk* generated in *TransKGen* algorithm is perturbed by a random value, which is protected by the difficult assumption problem used at the bottom. Because when data requesters collude with a user, they cannot decrypt any part of the *tk*. Therefore, the proposed solution in this paper is collusion-resistant under the premise that the difficult assumption problem is established.

Confidentiality

Our scheme's confidentiality guarantees that adversaries are unable to obtain any portion of the plaintext information. Below, we present our proof of this confidentiality.

For the confidentiality of the ciphertext *CTX*, we first assume that an adversary Alice can destroy the confidentiality of the ciphertext of our scheme with a probability of γ. At the same time, another adversary Bob can solve the assumed difficulty problem on which our scheme relies with a probability of δ. In the initialization phase, Bob first constructs two lists L_{sk} and L_{rk}, which are used to store attribute-related secret keys and ciphertext transformation keys respectively.

To generate the public parameters, Bob selects a random value g_x for each attribute x in the attribute space and calculates the public key *PP*. In order to simulate the private key of the attribute set provided by Alice, Bob adds the generated $SK1$, $SK2$, $\{SK_x\}$ to the list L_{sk}. Using the similar representation, Bob generates the transformation key *tk* and adds it to the list L_{tk}. Alice selects messages msg_1 and msg_2 of equal length and sends them to Bob. Bob randomly selects one of the messages and uses an encryption algorithm to generate the corresponding ciphertext and returns it to Alice. After receiving the ciphertext, Alice guesses with α probability which message this encrypted ciphertext was generated from. If the final correctness is consistent with 1/2, it means that as long as our underlying choice assumes that the difficult problem is indistinguishable to Bob, Alcie cannot destroy the confidentiality of the encrypted ciphertext in our scheme.

For the modified cipherext CTX', we assume that there is an adversary Alice who can destroy the confidentiality of the modified ciphertext of our scheme with α probability, and an adversary Bob who can solve the underlying hypothetical difficult problem on which our scheme relies with β probability. Simply put, this difficult problem involved in our scheme ensures that the adversary cannot tell whether the element *ele* is a random element from G_T or a pair of two group elements. In this proof stage, the early stages are basically consistent with the confidentiality proof of the ciphertext *CTX*.

The difference is that Alice will select two messages of the same length, msg_1 and msg_2, and send them to Bob. Immediately afterwards, Bob locally generates an attribute set *AS*, and generates *sk* and *tk* through the *ABCT.KCrt* algorithm and the *TransKGen* algorithm. Subsequently, Bob randomly selects one of msg_1 and msg_2, together with the access control rule *AP* as the input of the encryption algorithm, and obtains the ciphertext *CTX*. Finally, use the ciphertext *CTX* and the transformation key *tk* as the input of the ciphertext transformation algorithm to obtain the modified ciphertext CTX' and return it to Alice. After obtaining CTX' generated by Bob, Alice guesses with α probability which message the re-encrypted ciphertext was generated from. If the final correctness is consistent with 1/2, it means that as long as our underlying choice assumes that the difficult problem is indistinguishable to Bob, Alcie cannot destroy the confidentiality of the modified ciphertext in our scheme. Proof completed.

Regulatorability

To ensure the integrity and correctness of ciphertext transformation and to protect against false accusations by malicious entities, our method incorporates hash commitment and digital signature technologies during the encryption process. We generate a hash commitment (CMT_{msg}) and digital signatures (Sig_{msg}), which are stored on the blockchain along with the ciphertext (*CTX*). The decentralized and immutable characteristics

of the blockchain ensure the security of these stored records. Digital signatures play a crucial role in affirming the integrity of data, ensuring it remains unaltered since its signing. Any slight alteration in the data will result in the failure of signature verification, thereby safeguarding data integrity. Concurrently, hash commitments serve to maintain this integrity by generating a unique data digest (hash value). This hash value, acting as the data's distinct fingerprint, undergoes significant changes even with minor modifications to the original data. Furthermore, hash commitments enable a submitter to commit to specific data for a recipient while withholding the immediate disclosure of the data itself, thus ensuring the privacy of the data until the submitter decides to disclose the original content.

In the event of disputes over decryption results, the Certification Authority (CA) can authenticate the data stored on the blockchain, enabling monitoring and verification of the data sharing process. Our hash commitment functions utilize collision-resistant cryptographic hash functions, such as SHA-1, which yield a consistent output length for any input message, rendering it highly impractical for attackers to generate a different message yielding the same hash result.

Implementation and evaluation

Performance analysis

To verify the performance of our scheme, we simulate our solution utilizing the Charm framework [29] and evaluate its performance on a personal computer running Ubuntu 18.04, equipped with an Intel Core i7-8700@3.20 GHz and 8 GB RAM.

To simulate the scheme we propose in this paper, we used a total of four cipher curves for testing, as shown in Table 1 below.

Our evaluation of the scheme's performance included measuring the running times of various algorithms. Figure 3 presents the computation times for the *ABCT.Init* and *ABCT.KCrt* algorithms. For the *ABCT.Init* algorithm, when employing the SS512 curve, the computational time registers at 29ms for a set of 10 attributes, escalating to 212ms as the attribute set expands to 100. In the case of the *ABCT.KCrt* algorithm, we noted a linear increase in computational cost proportional to the size of the attribute set. Significantly, the *ABCT.KCrt* algorithm demonstrates greater efficiency on the SS512 curve compared to its performance on the MNT224, MNT201, and MNT159 curves.

Table 1 The definition of curves

Curve	Definition
SS512	Symmetric pairing curve with 512-bit base field
MNT224	Asymmetric pairing curve with 224-bit base field
MNT201	Asymmetric pairing curve with 201-bit base field
MNT159	Asymmetric pairing curve with 159-bit base field

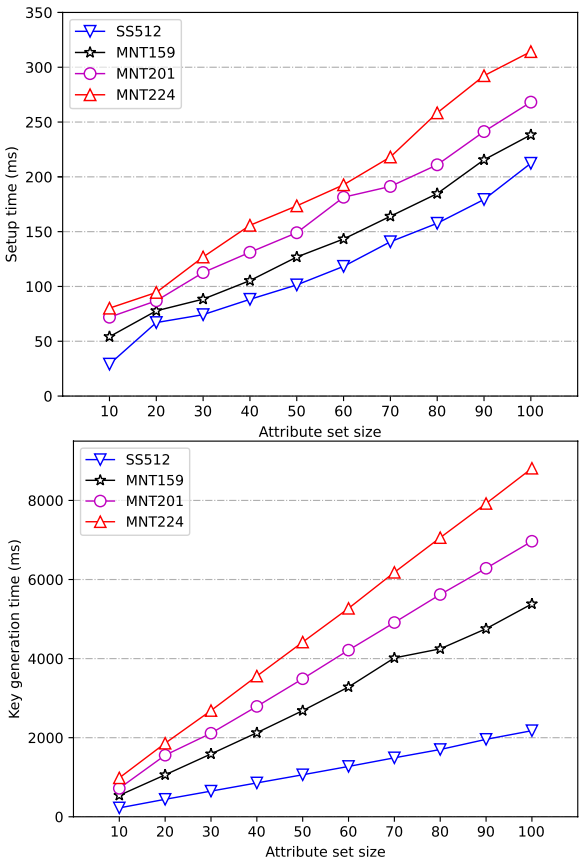

Fig. 3 Computation time of *ABCT.Init* and *ABCT.KCrt* algorithms

For the *ABCT.Encr* algorithm, the calculation cost using SS512 curve is 210ms when the number of attributes is 10 and the calculation cost is 2026ms when the number of attributes is 100. For the *ABCT.TransKCre* algorithm, we can know that the computational cost of *ABCT.TransKCre* algorithm keeps the same trend with the *ABCT.KCrt* algorithm of our proposed scheme and *ABCT.TransKCre* algorithm based on SS512 curve has better computational performance than those based on MNT224 curve, MNT201 curve and MNT159 curve. The comparison results are given in Fig. 4.

The computational costs of the *ABCT.TransE* and *ABCT.Decr* algorithms in our scheme are demonstrated to increase with the attribute count, as illustrated in Fig. 5. For the *ABCT.TransE* algorithm, employing the SS512 curve, the computational time is 210ms for an attribute set size of 10, and it escalates to 2026ms for 100 attributes. Similarly, the *ABCT.Decr* algorithm exhibits a linear increase in computational cost proportional to the

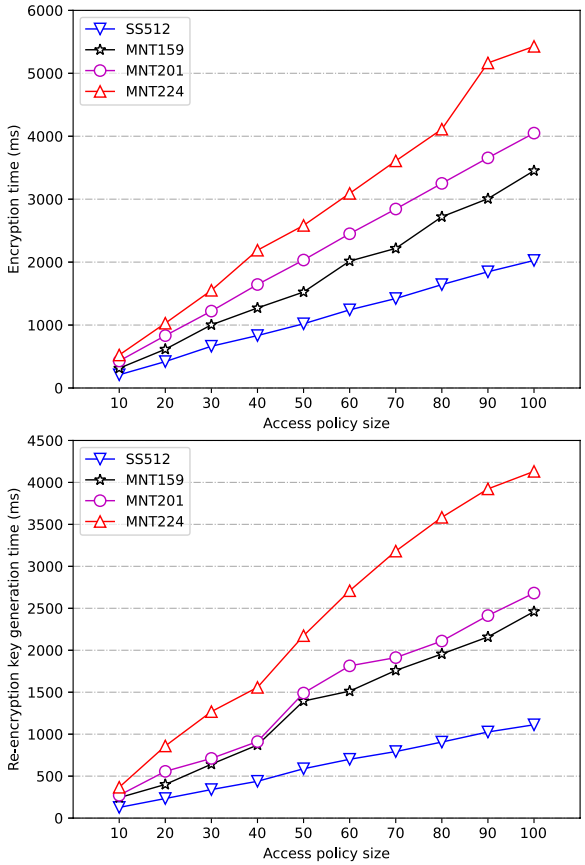

Fig. 4 Computation time of *ABCT.Encr* and *ABCT.TransKCre* algorithms

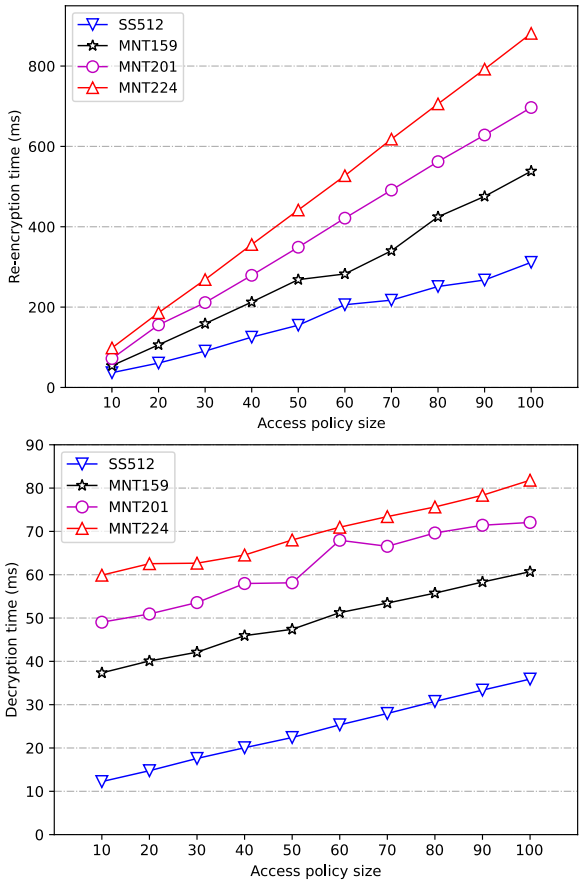

Fig. 5 Computation time of *ABCT.TransE* and *ABCT.Decr* algorithms

attribute count. Notably, the *ABCT.Decr* algorithm's performance is more efficient on the SS512 curve compared to its counterparts based on the MNT224, MNT201, and MNT159 curves. Overall, the simulation results for our proposed scheme are deemed suitable for most practical applications.

Simulation

In our simulation, the data owners and data requesters are registered on the Hyperledger Fabric 1.4 blockchain. Additionally, the system employs IPFS as a decentralized data delegation platform. Detailed computer specifications are presented as follows: Ubuntu 18.04 64-bit, Intel Core i7-7700@3.60GHz, Samsung DDR4-3200 8GB RAM, and a DT01ACA200 2TB HDD. Given that the transformed ciphertext produced by the *ABCT.TransE* algorithm requires storage on the blockchain, we implemented the evidence chaincode in Go and deployed it on the blockchain. To evaluate the blockchain's performance, we tested its throughput using Hyperledger Caliper 0.3.0, an excellent blockchain benchmarking tool. We set the total transaction count to 20 and the transaction sending rate to 10, recording the throughput for

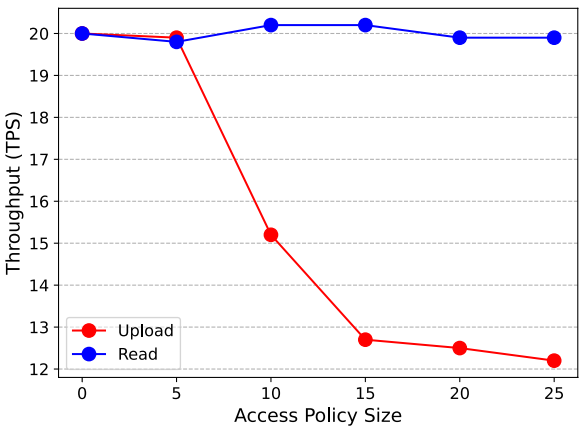

Fig. 6 Throughput of Blockchain

uploading and reading CTX' under varying access policies. The results are shown in Fig. 6.

It can be seen that the throughput remains unaffected by the increasing size of access control rule when reading CTX', consistently maintaining a rate of approximately

20 TPS. This throughput rate is consistent with that observed when initiating an empty transaction without ciphertext reading. However, while uploading *CTX′*, the throughput shows a downward trend. This decline is attributed to the increase in access control rule size, which correspondingly enlarges the ciphertext size and, as a result, augments the transaction payload. When the access control rule size increases to 25, the transaction throughput drops to around 12 TPS.

We constructed a decentralized data delegation platform using IPFS0.5.0 to store the encrypted data *CTX* and assessed its performance. The upload and download time for files of varying sizes were evaluated, as depicted in Fig. 7. Notably, the download time increases gradually, with a 1GB file downloading in just 2.23 seconds. For file sizes up to 100MB, the upload time remains consistently below 3 seconds. However, for file sizes of 500MB and 1GB, the upload times rise to 16.84 seconds and 37.82 seconds, respectively.

Conclusion

Web3.0 represents a significant evolution of blockchain technology, heralding the next generation of the Internet with a primary focus on decentralization, distribution, and user autonomy. Within this context, data delegation solutions for Web3.0 place paramount importance on user privacy, security, and data ownership. However, it is worth noting that existing data delegation solutions grounded in blockchain and attribute-based encryption (ABE) currently face limitations in handling encrypted data. While attribute-based ciphertext transformation (ABCT) algorithm possesses the inherent capability to process ciphertext, it falls short in verifying transformed ciphertext provided by potentially malicious data processors. In response to this challenge, this paper introduces an innovative approach that integrates ABCT technology with smart contracts to devise a trusted data delegating and fine-grained secure sharing scheme tailored specifically for Web3.0.

This novel scheme incorporates a commitment mechanism and provides digital signature, greatly enhancing the verifiability and equity of the existing ABCT technology. It not only enables precise access control for one-to-many data but also empowers the ciphertext re-encryption. Moreover, by leveraging the inherent features of blockchain-based smart contracts, this approach ensures a comprehensive audit trail throughout the entire process and automates execution, thereby safeguarding the equitable distribution of interests among all stakeholders. This development holds immense significance in enhancing the practicality of the solution, broadening the impact of data, and increasing its intrinsic value. In the future, we envisage further enhancements through the utilization of public chain incentive mechanisms for the design of a data hosting solution tailored to the Web3.0 ecosystem.

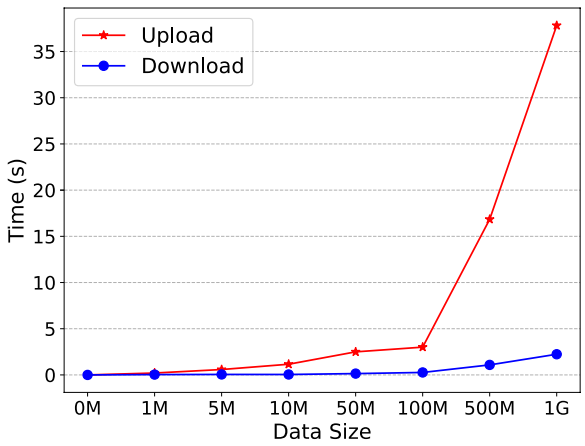

Fig. 7 IPFS performance under different file sizes

Authors' contributions
Conceptualization, H. Gao and P. Duan; methodology, X. Pan and P. Duan; software, X. Zhang and P. Duan; validation, K. Ye and and P. Duan; formal analysis, P. Duan; investigation, Z. Zhong and P. Duan; writing—original draft preparation, H. Gao and P. Duan; writing—review and editing, X. Pan and K. Ye; supervision, X. Zhang.

Funding
Not applicable.

Availability of data and materials
No datasets were generated or analysed during the current study.

Declarations

Ethics approval and consent to participate
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 20 November 2023 Accepted: 13 December 2023
Published online: 22 January 2024

References

1. Hannila H, Silvola R, Harkonen J, Haapasalo H (2022) Data-driven begins with data; potential of data assets. J Comput Inf Syst 62(1):29–38
2. Zhao L, Zhong L, Liu J, Zeng X, Zhang J (2023) A regulatable mechanism for transacting data assets. IEEE Internet Things J 10(24):21615-21632
3. Yang S, Li M (2023) Web3. 0 data infrastructure: Challenges and opportunities. IEEE Netw 37(1):4–5
4. Guan C, Ding D, Guo J, Teng Y (2023) An ecosystem approach to web3. 0: a systematic review and research agenda. J Electron Bus Digit Econ 2(1):139–156
5. Rasori M, La Manna M, Perazzo P, Dini G (2022) A survey on attribute-based encryption schemes suitable for the internet of things. IEEE Internet Things J 9(11):8269–8290

6. Ge C, Susilo W, Liu Z, Baek J, Luo X, Fang L (2023) Attribute-based proxy re-encryption with direct revocation mechanism for data sharing in clouds. IEEE Trans Dependable Secure Comput:1-12
7. Lin Z, Zhou J, Cao Z, Dong X, Choo KKR (2023) Generalized autonomous path proxy re-encryption scheme to support branch functionality. IEEE Trans Inf Forensic Secur 18:5387-5440
8. Wilkinson S, Boshevski T, Brandoff J, Buterin V (2014) Storj a peer-to-peer cloud storage network
9. Vorick D, Champine L (2014) Sia: Simple decentralized storage (2014) White paper available at https://sia.tech/sia.pdf, Retrieved May, 2014, 8:2018
10. Benet J (2014) Ipfs-content addressed, versioned, p2p file system. https://doi.org/10.48550/arXiv.1407.3561
11. Ogden M, McKelvey K, Madsen MB, et al (2017) Dat-distributed dataset synchronization and versioning. Open Sci Framework 10(2.2) https://doi.org/10.31219/osf.io/nsv2c
12. Politou E, Alepis E, Virvou M, Patsakis C, Politou E, Alepis E, Virvou M, Patsakis C (2022) Implementing content erasure in ipfs. Priv Data Prot Challenges Distrib Era 26:151–163
13. Politou E, Alepis E, Patsakis C, Casino F, Alazab M (2020) Delegated content erasure in ipfs. Futur Gener Comput Syst 112:956–964
14. Sahai A, Waters B (2005) Fuzzy identity-based encryption. In: Proceedings of the Annual international conference on the theory and applications of cryptographic techniques. Springer Berlin Heidelberg, pp 457–473
15. Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. In: Proceedings of the IEEE symposium on security and privacy. IEEE, Berkeley, pp 321–334
16. Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the ACM conference on Computer and communications security. CCS, Alexandria, pp 89–98
17. Ibraimi L, Asim M, Petković M (2009) Secure management of personal health records by applying attribute-based encryption. In: Proceedings of the international workshop on wearable, micro, and nano technologies for personalized health. IEEE, Oslo, pp 71–74
18. Chen D, Chen L, Fan X, He L, Pan S, Hu R (2014) Securing patient-centric personal health records sharing system in cloud computing. China Commun 11(13):121–127
19. Barua M, Liang X, Lu R, Shen X (2011) Peace: An efficient and secure patient-centric access control scheme for ehealth care system. In: Proceedings of the IEEE Conference on Computer Communications Workshops. IEEE, Shanghai, pp 970–975
20. Blaze M, Bleumer G, Strauss M (1998) Divertible protocols and atomic proxy cryptography. In: Proceedings of the International conference on the theory and applications of cryptographic techniques. Springer, Berlin Heidelberg, pp 127–144
21. Ateniese G, Fu K, Green M, Hohenberger S (2006) Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans Inf Syst Secur 9(1):1–30
22. Liang X, Cao Z, Lin H, Shao J (2009) Attribute based proxy re-encryption with delegating capabilities. In: Proceedings of the international symposium on information, computer, and communications security. Asia CCS, Sydney, pp 276–286
23. Liang K, Au MH, Liu JK, Susilo W, Wong DS, Yang G, Yu Y, Yang A (2015) A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing. Futur Gener Comput Syst 52:95–108
24. Liang K, Susilo W (2015) Searchable attribute-based mechanism with efficient data sharing for secure cloud storage. IEEE Trans Inf Forensic Secur 10(9):1981–1992
25. Ge C, Susilo W, Wang J, Huang Z, Fang L, Ren Y (2016) A key-policy attribute-based proxy re-encryption without random oracles. Comput J 59(7):970–982
26. Liang K, Au MH, Susilo W, Wong DS, Yang G, Yu Y (2014) An adaptively cca-secure ciphertext-policy attribute-based proxy re-encryption for cloud data sharing. In: Proceedings of the Information Security Practice and Experience International Conference. Springer, Fuzhou, pp 448–461
27. Ge C, Susilo W, Liu Z, Xia J, Szalachowski P, Fang L (2020) Secure keyword search and data sharing mechanism for cloud computing. IEEE Trans Dependable Secure Comput 18(6):2787–2800
28. Rouselakis Y, Waters B (2015) Efficient statically-secure large-universe multi-authority attribute-based encryption. In: Proceedings of the Financial Cryptography and Data Security International Conference. pp 315–332
29. Akinyele JA, Garman C, Miers I, Pagano MW, Rushanan M, Green M, Rubin AD (2013) Charm: a framework for rapidly prototyping cryptosystems. J Cryptographic Eng 3:111–128

Publisher's Note

Hongmin Gao received the PH.D. degree from BeiJing University of Posts and Telecommunications in 2022. He now works in China Mobile Information Technology Co., Ltd. His current research interests include blockchain, cryptography, and network and information security.

Pengfei Duan received the B.E. degree in 2018 and the M.E. degree in 2021. He is currently pursuing a Ph.D. degree in cyberspace security at BUPT, China. His research interests include privacy protection, and information security.

Xiaofeng Pan graduated with a master's degree from Institute of Computing Technology, Chinese Academy of Sciences. He now is the blockchain architect of China Mobile Information Technology Co., Ltd. His current research direction is blockchain technology application and architecture.

Xiaojing Zhang received the M.S. degree in computer science and technology from Beijing University of Posts and Telecommunications in 2011. He now is the deputy general manager of China Mobile Information Technology Co., Ltd. His current research interests include blockchain architecture design and application technology innovation.

Keke Ye received the M.S. degree from Beijing University of Posts and Telecommunications in 2011. He now works in China Mobile Information Technology Co., Ltd. His current research interests include blockchain application and related integration technology.

Ziyuan Zhong received the M.S. degree from Beijing University of Posts and Telecommunications in 2014. He now works in China Mobile Information Technology Co., Ltd. His current research interests include blockchain, cryptography, and database.