Federated Time-Bound Blockchain-Based Revocation with Ciphertext-Policy Attribute-Based Encryption and Redundant Shard Reassembly

Namrata Naikwade
Department of Computer Science and Engineering, MIT Art, Design and Technology University, Pune, Maharashtra, India
namrata.naikwade@gmail.com (corresponding author)

Prashant Dhotre
Department of Computer Science and Engineering, MIT Art, Design and Technology University, Pune, Maharashtra, India
prashant.dhotre@mituniversity.edu.in

Shafi Pathan
Department of Computer Science and Engineering, MIT Art, Design and Technology University, Pune, Maharashtra, India
shafi.pathan@mituniversity.edu.in

Received: 24 July 2025 | Revised: 8 September 2025, 9 October 2025, and 22 October 2025 | Accepted: 6 November 2025

 DOI: https://doi.org/10.48084/etasr.13552

ABSTRACT

Ensuring data security, access revocation, and cost efficiency during data exchange becomes paramount in secure cloud computing. This study presents a secure data sharing framework that integrates Federated Time-Bound Ciphertext-Policy Attribute-Based Encryption (FTB-CP-ABE) along with blockchain-based revocation. On top of it, data is stored with fault-tolerant redundant sharding. The proposed FTB-CP-ABE enables data owners to define access policies and allows multiple attribute authorities to issue time-bound decryption keys in a decentralized manner. To enhance data security, encrypted data is divided into chunks (shards) using threshold secret sharing, which allows an authorized user to recover data even if some shards are lost. Blockchain is used to enforce transparent revocation and maintain audit trails, which allows logging keys that are revoked and why. In the decryption process, the user's key ID is checked against the ledger, and if the status is revoked, the system denies decryption even if the CP-ABE policy matches. The proposed multilayer framework provides auditable fine-grained access control in untrusted cloud environments.

Keywords-cloud computing; data security; access revocation; federated time-bound ciphertext-policy attribute-based encryption (FTB-CP-ABE); blockchain

I. INTRODUCTION

In a multi-user federated cloud environment, issues of data security and privacy become important, especially for sensitive data. Organizations and individuals store various types of information, such as medical records, legal documents, or multimedia files, on third-party platforms, which introduces several security challenges. Traditional encryption techniques offer confidentiality, but lack flexibility in providing fine-grained access control and revocation. In addition, centralized models not only lack scalability but also expose the system to a single point of failure. Attribute-Based Encryption (ABE) has become a promising method for fine-grained access control [1-3], implemented in two distinct forms: Key-Policy Attribute-Based Encryption (KP-ABE) [2] and Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [4]. In KP-ABE, the access policy, which is defined by a trusted authority, is embedded in the private key, which is less flexible for data owners since they do not define the access policy directly, and for dynamic access requirements, key distribution becomes complex. In CP-ABE, the ciphertext is embedded in the access policy, which increases the length of the ciphertext and

computational overhead. Since the access policy is embedded in the ciphertext, attribute revocation becomes difficult and requires re-encryption. This static nature of CP-ABE makes it difficult to apply in a dynamic environment.

To address these issues, various solutions have been proposed, such as outsourcing decryption [5] and using proxy servers [6] to minimize computational overhead. However, flexibility, access revocation, and transparency remain issues. To address these challenges, this study introduces a novel secure and privacy-enabled data storage and sharing framework integrating Federated Time-Bound CP-ABE (FTB-CP-ABE) and blockchain for efficient auditable access revocation and redundant data sharding to store files securely. To allow temporary access to data, a time duration constraint is embedded into the ciphertext and decryption keys. A federated environment allows multiple independent Attribute Authorities (AAs) to manage attributes that support decentralized operations and cross-organizational scalability. The limitations of traditional key revocation in ABE are addressed by incorporating a blockchain-based revocation mechanism [7]. Key distribution and revocation events are saved in a read-only blockchain ledger with the help of smart contracts. During the decryption process, the system verifies the user's key and time validity in the ledger, which ensures tamper-proof and transparent access revocation without requiring data re-encryption. A second layer of security is provided by implementing redundant data sharding and storing each piece separately. For data sharding, Shamir's Secret Sharing scheme ensures that the original data can still be reconstructed even if a subset of data shards is lost or unavailable. During the sharding process, a Merkle tree of shards is built, and the root hash is stored in encrypted metadata. During reconstruction, the integrity of each shard is verified. This helps in integrity verification without heavy cryptographic overhead.

A. Attribute-Based Encryption

In [8], a revocable lattice-based CP-ABE (RL-ABE) scheme was designed to provide security against collusion and quantum attacks based on the Ring Learning with Errors (R-LWE) problem. This scheme not only allows efficient attribute revocation but also enables dynamic update of user access rights. In [9], a CP-ABE scheme supported shared decryption, allowing authorized users to decrypt data independently. When an authorized user is unavailable, semi-authorized users collaborate to perform the decryption operation. In addition, an integrated access tree structure is designed to improve efficiency in the decryption process. Attribute-Based Encryption with Dynamic Attributes Support (ABE-DAS) [10] was proposed to enhance data security in a hybrid cloud environment. Sensitive operations are performed on private clouds, and general operations are managed by public clouds, where privacy risks and data leakage issues are inherent. ABE-DAS was designed to address these issues, providing fine-grained access control with dynamic attribute support to improve data confidentiality and system adaptability.

In [11], a Revocable and Searchable Attribute-Based Encryption (RSABE) scheme was proposed, specifically designed for mobile cloud storage to address issues such as attribute revocation, keyword search, and computational efficiency for mobile devices with limited resources. In this scheme, cloud servers look after secret key and ciphertext updates, as well as handle keyword search without requiring a continuous online authority. The decryption operation is outsourced to reduce computational overhead on the client-side. In [12], KP-ABS-UT was proposed to address the scenario in which the attribute authority is not trusted in traditional ABE. To avoid any forgery by the authority, user keys are divided into a part provided by the authority and a self-chosen secret component. The signer is identified by a tracing entity, which ensures traceability and accountability. In [13], CP-ABE was extended for IoT data security in cloud-based environments. A ciphertext-policy hiding CP-ABE scheme was proposed for fine-grained access control, and a white-box traceable CP-ABE variant was designed with built-in accountability features to address key abuse issues.

B. Blockchain-Based Revocation in ABE

In [14], a blockchain-based fine-grained access control scheme utilized CP-ABE for cloud storage environments. This system eliminated reliance on a single trusted authority for user revocation and key escrow by implementing a two-authority-based key generation mechanism. The confidentiality of outsourced data is ensured by allowing real-time attribute-level access revocation rather than traditional time-based mechanisms. This system makes it difficult for attribute authorities or cloud service providers to breach the data. In [15], CP-ABE with dynamic attributes was proposed for multi-user access control in untrusted cloud environments. This system provides secure data sharing without involving a cloud provider in the decryption process. A blockchain-based decentralized ledger is used to record security events such as key generation, policy assignment, and access revocation. A set of cryptographic protocols is designed to maintain the privacy of operations involving secret keys. In [16], the focus was on distributed data storage and retrieval, designing a hybrid multi-cloud model to enhance cloud efficiency and provide a secure environment. The data was split into chunks and encrypted before being stored in the cloud. In [17], a hierarchical multi-expressive blockchain was presented for healthcare systems. This study stated that healthcare systems need diverse privacy requirements and multiple trust domains. Thus, to enforce local policies, domain-specific blockchains and a proxy blockchain layer were proposed for cross-domain interoperability, allowing secure data sharing among mutually untrusted stakeholders. CP-ABE was used for fine-grained access control, and smart contracts were used to ensure decentralized policy enforcement. In [18], blockchain and Attribute-based Searchable Encryption (ABSE) were combined for data authorization control. Proxy encryption and decryption were used to hide policy and attribute revocation with minimum computational cost. Encrypted data is stored on the cloud, and metadata is maintained on the blockchain. This system enforces access behavior rules through smart contracts.

II. SYSTEM DESIGN

The proposed architecture embeds a blockchain-based access control model, where access revocation events are logged into an immutable chain. Every issued key has a smart contract that contains the key ID and its status, and a valid time

window where the key expires or an admin manually revokes access. Figure 1 shows the model design, which covers the following key points.

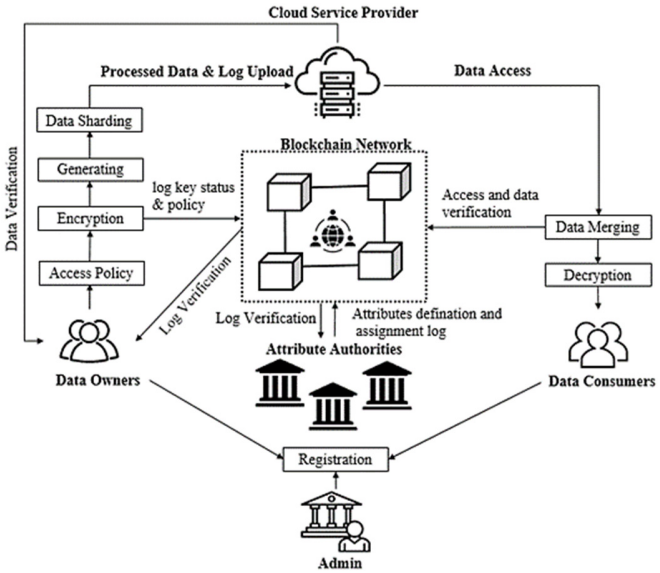

Fig. 1. Blockchain-based federated time-bound CP-ABE data sharing framework for secure data sharing and transparent data access revocation.

- Attribute Authority (AA): Defines, manages, and distributes keys for each user. Multiple AAs work together in a decentralized way to avoid a single point of failure. Each AA manages a specific set of attributes, and each event is logged on an immutable blockchain, where key assigning, revocation, and update are implicitly executed through smart contracts.
- Cloud Service Provider (CSP): Stores and manages encrypted data and metadata. The cloud is publicly available, but access is restricted through authentication-based protocols.
- Blockchain: Stores important events such as key assigning time, key status, key expiration time, and revocation events. Before the decryption process, a user's key ID is checked against the blockchain ledger. Unauthorized or expired keys are rejected implicitly without requiring re-encryption or key redistribution, which enhances the ability of the system to scale without any tamper issues.
- A Data Owner generates the data, defines access policies, encrypts and divides data into multiple fragments (shards), generates hashes, and uploads the processed data to CSP.
- A Data Consumer can request access to data, but can decrypt only those who have the right attributes to satisfy the access policy and time conditions. Before accessing data, the user needs to get authenticated by the admin in the registration phase.

AAs handle definition, update, revocation, and key distribution independently. Smart contracts are used to implicitly record these operations on the blockchain, which guarantees data transparency. The blockchain eliminates the single point of failure issue due to its decentralized nature. The system contains two main processes: the data owner process, which includes data sharding, hash generation, access policy definition, and encryption, and the data consumer process, which includes data decryption and merging. Each process is described in detail, and each notation used is defined in Table I.

TABLE I. NOTATIONS AND DESCRIPTION

Notation	Description
AAs	Federated Attribute Authorities
G, GT	Cyclic groups of prime order P, with bilinear map e: $G \times G \rightarrow GT$
g	Generator of group G
$e(g,g)$	Bilinear pairing
PK	Public Key
MK	Master secret Key
U	User
$Attr(U)$	Set of user attributes
$Attr_{secret}$	Secret random for a specific attribute
$Attr_{public}$	Public parameter for specific attribute
sec	Session secrete
SK_U	User's secret key
P	Access policy
$[T_{start}, T_{end}]$	Time window for key validity
$H()$	Cryptographic hash function that maps attributes/time to group elements
$Issuer_{id}$	Id of AA that has issued a key
Key_{id}	Key id
U_{id}	User id
$Sign$	Digital signature of issuer (AA or DO)

A. System Initialization

Trusted authority generates global parameters. First, bilinear groups G, GT of prime order p are chosen, and then a generator g is selected ($g \in G$). Finally, random exponents are selected, with $\alpha, \beta \in Z_p$. So the public key can be defined as:

$$PK = \{G, GT, g, h = g^{\beta}, f = g^{\alpha}, e(g,g)^{\alpha}\} \quad (1)$$

and the master key as:

$$MK = \{\beta, g^{\beta}\} \quad (2)$$

Each AA maintains a set of attributes $\{Attr_1, Attr_2, \ldots Attr_n\}$, with each attribute $Attr \in \{Attr_1, Attr_2, \ldots Attr_n\}$. For user U, various AAs issue sets of attributes $Attr(U)$, and for each attribute, AA chooses a secret random value $Attr_{secret}$ that is used to compute a public parameter $Attr_{public} = g^{Attr_{secret}} \in G$, which is used during decryption and keeps $Attr_{secret}$, which is used for key generation. To support time-bound key generation, each AA defines a hash function $H(Attr)$ that maps attribute names into group elements. A hash function $H(T_{start}, T_{end})$ is generated separately to bind time to the user's secret key. Here, $Attr_{secret}$ provides collusion resistance, making it harder for a user to combine keys improperly, and $H(T_{start}, T_{end})$ provide temporal constraints when issuing attributes.

B. Users Key Generation and Issuance

Key generation and issuance are critical phases in which secret keys are distributed to users, each tied to attributes such

as the user receiving and the valid time window. The user can decrypt the ciphertext only if they match the access policy and the key has not expired or been revoked. Each AA has the attribute list $Attr(U)$ of the respective user and valid time window $[T_{start}, T_{end}]$. A random value $rand \in Z_p$ is selected to ensure key uniqueness. A root key component is calculated by using a hash of the valid time window bound to the key.

$$K_c = g^{\alpha}.H(T\_start, T\_end\ )^{rand} \quad (3)$$

Here, K_c is a root key component and g^{α} comes from a master key already known to the AA.

An attributes component is calculated for each attribute $attr \in Attr(U)$:

$$attr_c = g^{Attr_{secrete}}.H(attr)^{rand} \forall\ att \in Attr(U) \quad (4)$$

$Attr_{secret}$ is the secret for specific attribute generated by the AA, and the name of each attribute gets mapped into group G by $H(attr)$. Finally a secret key for the user is generated:

$$SK_U = (K_c, attr_c | attr \in Attr(U), [T_{start}, T_{end}] \quad (5)$$

C. Access Policy Definition and Encryption

The data owner defines access policy P. A random session secret is selected ($sec \in Z_p$), which is used to protect message M using exponentiation. The message is encrypted using pairing. Assuming $e(g,g)^a$, the encrypted message components are computed: $C_o = M.e(g,g)^{a.sec}$ and $C_1 = g^{sec}$. Attribute parameters and policy are encoded for each attribute node n in an access policy tree. Let $attr_n$ be an attribute at node n. A random session secret is selected for each node $sec_n \in Z_p$ such that $\sum sec_n = sec$. The public component is computed using (6), which will be used to reconstruct the encryption secret.

$$C_{1n} = g^{sec_n} \quad (6)$$

The component C_{2n} is calculated using (7), binding $attr_n$ to the ciphertext using a public value of the attribute $Attr_{public_n = g^{Attr_{secrete_n}}}$ and the random secret share sec_n , which ensures that unless the user possesses a valid key bound to $Attr_{secret_n}$ the respective user attribute cannot cancel $g^{Attr_{secret_n}{}^{sec_n}}$:

$$C_{2n} = Attr_{secrete_n}{}^{sec_n} = g^{Attr_{secrete}{}^{sec_n}} \quad (7)$$

Finally a time window is appended to the ciphertext $[T_{start}, T_{end}]$. The final ciphertext can be defined as:

$$CT = \left\{ \begin{array}{c} Policy: P, \\ Time\ Window: [T_{start}, T_{end}], \\ C_o = M.e(g,g)^{a.sec \in GT}, \\ C_1 = g^{sec}, \\ C_{1n} = g^{sec_n \in G}, \\ C_{2n} = Attr_{secret_n}{}^{sec_n} = g^{Attr_{secret}{}^{sec_n \in G}} \end{array} \right\} \quad (8)$$

D. Data Hashing for Verifying Correctness and Sharding

- Data Hashing: A Merkle Tree is integrated to ensure the integrity of an encrypted file. The encrypted data CT, is divided into n fixed-size blocks ($CT = (B_1, B_2, \dots \dots B_n)$) and the hash function SHA-256 is ysed to compute the hash of each block B_i, that is $L_i = H(B_i)$, where $(L_1, L_2, \dots \dots L_n)$ are leaf nodes of the Merkle Tree. An internal tree is built by hashing adjacent pairs and computing the parent $L_{ij} = H(L_i || L_j)$. This process is repeated until one root is left, $M_{root} = H_t$, where the M_{root} is a Merkle root and H_t is the final hash. If the number of leaves is odd, then the last hash is duplicated to make it even.
- Data Sharding: To implement Shamir's secret sharing, the whole encrypted file CT is converted into number S in the field Z_p, and then a random polynomial $f(x)$ of degree $k-1$ is chosen:

  $$f(x) = S + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k-1} mod p \quad (9)$$

  Calculate n shares

  $$S_i = (x_i f((x_i)) \quad (10)$$

  Each cloud node gets $S_i = (x_i f((x_i))$ and H_t is stored on blockchain.
- Data Integrity Verification: For data correctness verification, k shards are collected and Lagrange interpolation is used to reconstruct $s = f(0)$. s is converted into an encrypted file CT', which is then split into blocks $(B_1', B_2', \dots, B_n')$. Each block is hashed using $L_i' = H(B_i')$. M_{root}' is reconstructed from $[L_i']$. If M_{root}' matches H_t, which is previously saved on blockchain, then the data is not tampered, else the data is manipulated.

E. Blockchain

Blockchain contains a sequence of blocks ($B_1, B_2, \dots . B_n$), each containing a transaction list $tr[]$. $prev\_hash$ contains the hash of the previous block B_{i-1} , a random number rn represents a proof-of-work condition, and t_stamp represents the time when the block was created. The issued key transaction is represented as:

$$tr_{key_{issue}} = H(Key_{id} | U_{id} | Attr(U) | T_{start} | T_{end} | Issuer_{id} | Sign) \quad (11)$$

The key revocation transaction is represented as:

$$tr_{revoke} = H(Key_{id} | U_{id} | Attr(U) | reason | T_{end} | T_{revoke} | Sign) \quad (12)$$

A smart contract defines the AccessRegistry with some functions. When AA issues a key to a user, it calls a function $KeyRegister(U_{id}, Attr(U), Key_{id}, [T_{start}, T_{end}])$ that logs user id, user attributes, key id, and key validity with a timestamp. During the key revocation process, $KeyRevoke(Key_{id})$ logs the revocation of the user key, which implicitly invalidates the key for decryption. Data owner calls $PolicyRegister(File_{id}, Hash(P), H_t)$ to log policy and Merkle root.

F. Data Reconstruction

In the reconstruction process, k out of n shards are collected to rebuild the original encrypted file CT. Lagrange interolation is used to reconstruct $CT = f(0)$ from k shares $(x_1, y_1), (x_2, y_2), (x_k, y_k)$ as:

$$CT = f(0) = \sum_{j=1}^{k} y_j \cdot \lambda_j(0) \quad (13)$$

where:

$$\lambda_j(0) = \prod_{1 \le m \le k, m!=j} (0 - x_m) / (x_j - x_m) modp \quad (14)$$

where x_j is the j^{th} share of the x-value and x_m is the other share.

G. Data Decryption

In the decryption process, the first user's secret key SK_U is validated against the blockchain to ensure that it has not been revoked. Key time validity is checked:

$$if (T_{start}^{U} \le T_{start} \; AND \; T_{end}^{U} \ge T_{end}) ? \text{Continue} : \text{Abort}$$

Then, access policy satisfaction is checked:

$$if \; attr(U) \ne P ? Abort : Continue$$

If both conditions are passed, then compute $e(g,g)^{a.sec}$ using pairing $Y = e(C_1, K_c)$, which can be expanded as:

$$e(g^{sec}, g^{\alpha} \bullet H(T_{start} || T_{end})^{rand}) \quad (15)$$

which can be further expanded as:

$$e(g,g)^{\alpha.sec} . e(g^{sec}, H(T_{start} || T_{end})^{rand}) \quad (16)$$

Here Y contains the desired $e(g,g)^{\alpha.sec}$, which contains an extra time-bound term, $e(g^{sec}, H(T_s || T_e)^{rand}$, that needs to be cancelled.

For every attribute node n that satisfies the policy P, compute:

$$A_n = e(C_{1n}, K_c) \quad (17)$$

$$B_n = e(C_{2n}, H(attr)^{rand}) \quad (18)$$

This pairing result cancels the extra time binding part of the key. Finally, the original message is extracted:

$$M = C_0 / \, e(g,g)^{\alpha.sec} \text{ where } C_0 = M . e(g,g)^{\alpha.sec} \quad (19)$$

$$M = M . e(g,g)^{\alpha.sec} / \, e(g,g)^{\alpha.sec} \quad (20)$$

III. RESULT ANALYSIS

The proposed method was evaluated and compared with existing blockchain and ABE schemes. The simulation of the proposed scheme was performed on a Windows machine equipped with an Intel Core i5 processor running at 3.60 GHz and 8 GB of RAM. Textual data files, such as PDF and Word and Excel files, were used for data sharing.

To ensure performance reliability, the scheme was executed 10 times, and the results were averaged. In addition, the Standard Deviation (SD) along with 95% Confidence Interval (CI) was calculated. Figures 2 and 3 depict error bars representing the CIs. Figure 2 depicts a comparative analysis between B-CP-ABE [14], RVWABE-CA [19], CP-ABE-NDN [20], and the proposed scheme, which shows the encryption time required for a specific number of attributes. B-CP-ABE achieved an encryption time of 3615±42 ms and CP-ABE-NDN had 3528±38 ms, showing that both have a steep growth exceeding 3500ms for 50 attributes, which indicates that they are not scalable. In contrast, RVWABE-CA shows linear growth. The proposed scheme maintains an encryption time of 486±15 ms for 50 attributes due to its lightweight nature.

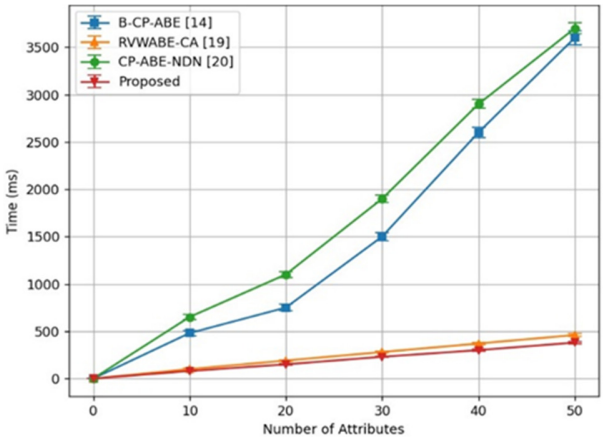

Fig. 2. Encryption time required for various numbers of attributes.

Similarly, Figure 3 shows a comparative analysis based on the decryption time and the number of attributes. CP-ABE-NDN and B-CP-ABE have decryption times of more than 4500 ms and 3500 ms, respectively, while RVWABE-CA shows linear growth. The proposed model divides decryption responsibilities between the cloud and the user, which reduces overhead on the user side. A heavy decryption operation is executed on the cloud side, and so, the computation time for decryption remains linear on the user side (maintaining 612±22 ms). These results highlight the scalability and lightweight nature of the proposed model, which makes it suitable for environments with complicated policies, time constraints, and low resources, such as IoT and edge computing systems.

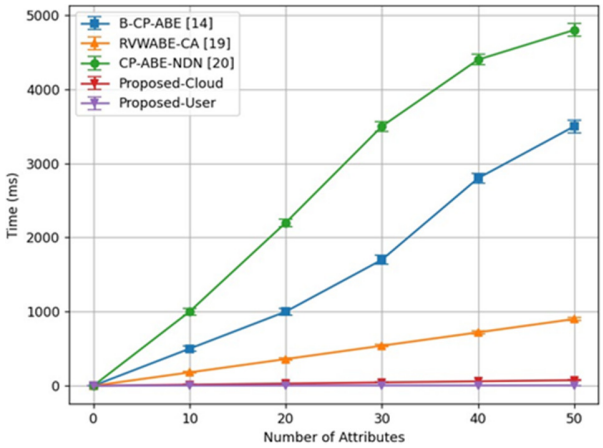

Fig. 3. Decryption time required for various numbers of attributes.

Table II shows the average time required for the execution of each blockchain function and its purpose in the proposed system. The *registerPolicy()* function takes ~14 ms, showing a fast and efficient write operation. The *getKeyStatus()* and *isRevoked()* take ~12 ms each, while *getMerkleRoot()* takes 19 ms, since it retrieves metadata integrity for verifying reconstructed data. *getPolicyHash()* checks the consistency of

the policy in ~16 ms. These results demonstrate the lightweight interactions suitable for real-time access control and verification.

TABLE II. EXECUTION TIME REQUIRED FOR BLOCKCHAIN FUNCTIONS

Function	Time (ms)	Purpose
registerPolicy(fileID, policyHash)	14	Write a policy on blockchain
getKeyStatus(keyID)	12	Verify if a key exists and is valid
isRevoked(keyID)	12	Verify if a key has been revoked
getMerkleRoot(fileID)	19	Get Merkle root for file integrity check
getPolicyHash(fileID)	16	Verify if a policy hash matches the stored one

IV. CONCLUSION

The proposed system integrates Time-Bound Ciphertext-Policy Attribute-Based Encryption (TB-CP-ABE) with blockchain for fine-grained data access control with transparency in decentralized environments. Heavy decryption operations are offloaded to the cloud and time constraints are embedded in keys, ensuring fine-grained and lightweight access control. Merkle Tree is used for data integrity verification, and Shamir's secret sharing is implemented for data distribution. Experimental results show that the proposed scheme requires less encryption and decryption time compared to existing ABE-based schemes, even with an increase in the number of attributes, highlighting its ability to scale in resource-constrained environments. The proposed scheme can be extended by integrating with IoT edge nodes for secure access control and focusing on dynamic policy updates without re-encrypting the data, allowing the system to change access control requirements in real time.

REFERENCES

[1] X. Wang *et al.*, "Attribute-Based Access Control Encryption," *IEEE Transactions on Dependable and Secure Computing*, vol. 22, no. 3, pp. 2227–2242, May 2025, https://doi.org/10.1109/TDSC.2024.3481497.

[2] A. Thakur, V. Ranga, and R. Agarwal, "Attribute-Based Encryption Scheme for Secure and Efficient Access in Blockchain," in *2024 IEEE International Conference for Women in Innovation, Technology & Entrepreneurship (ICWITE)*, Bangalore, India, Feb. 2024, pp. 653–658, https://doi.org/10.1109/ICWITE59797.2024.10502721.

[3] X. H. Liu, X. Y. Huang, W. Wu, and J. T. Ning, "Key-Policy Attribute-Based Encryption Based on SM9," *Journal of Computer Science and Technology*, vol. 40, no. 1, pp. 267–282, Jan. 2025, https://doi.org/10.1007/s11390-024-3726-z.

[4] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," in *2007 IEEE Symposium on Security and Privacy (SP '07)*, Berkeley, CA, USA, May 2007, pp. 321–334, https://doi.org/10.1109/SP.2007.11.

[5] Y. Liao, G. Zhang, and H. Chen, "Cost-Efficient Outsourced Decryption of Attribute-Based Encryption Schemes for Both Users and Cloud Server in Green Cloud Computing," *IEEE Access*, vol. 8, pp. 20862–20869, 2020, https://doi.org/10.1109/ACCESS.2020.2969223.

[6] X. Yang, S. Lian, Y. Wang, and C. Chen, "Attribute-Based Proxy Re-Encryption with Constant Size in Cloud Computing," in *2025 6th International Conference on Computing, Networks and Internet of Things (CNIOT)*, Shanghai, China, May 2025, pp. 1–5, https://doi.org/10.1109/CNIOT65435.2025.11070748.

[7] X. Qin, Y. Huang, Z. Yang, and X. Li, "A Blockchain-based access control scheme with multiple attribute authorities for secure cloud data sharing," *Journal of Systems Architecture*, vol. 112, Jan. 2021, Art. no. 101854, https://doi.org/10.1016/j.sysarc.2020.101854.

[8] S. Zhao, R. Jiang, and B. Bhargava, "RL-ABE: A Revocable Lattice Attribute Based Encryption Scheme Based on R-LWE Problem in Cloud Storage," *IEEE Transactions on Services Computing*, vol. 15, no. 2, pp. 1026–1035, Mar. 2022, https://doi.org/10.1109/TSC.2020.2973256.

[9] N. Chen, J. Li, Y. Zhang, and Y. Guo, "Efficient CP-ABE Scheme With Shared Decryption in Cloud Storage," *IEEE Transactions on Computers*, vol. 71, no. 1, pp. 175–184, Jan. 2022, https://doi.org/10.1109/TC.2020.3043950.

[10] P. Kanchanadevi, L. Raja, D. Selvapandian, and R. Dhanapal, "An Attribute Based Encryption Scheme with Dynamic Attributes Supporting in the Hybrid Cloud," in *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, India, Oct. 2020, pp. 271–273, https://doi.org/10.1109/I-SMAC49090.2020.9243370.

[11] S. Wang, D. Zhang, Y. Zhang, and L. Liu, "Efficiently Revocable and Searchable Attribute-Based Encryption Scheme for Mobile Cloud Storage," *IEEE Access*, vol. 6, pp. 30444–30457, 2018, https://doi.org/10.1109/ACCESS.2018.2846037.

[12] H. Hong and Z. Sun, "An efficient and traceable KP-ABS scheme with untrusted attribute authority in cloud computing," *Journal of Cloud Computing*, vol. 5, no. 1, Dec. 2016, Art. no. 2, https://doi.org/10.1186/s13677-016-0052-1.

[13] J. Li, Y. Zhang, J. Ning, X. Huang, G. S. Poh, and D. Wang, "Attribute Based Encryption with Privacy Protection and Accountability for CloudIoT," *IEEE Transactions on Cloud Computing*, vol. 10, no. 2, pp. 762–773, Apr. 2022, https://doi.org/10.1109/TCC.2020.2975184.

[14] P. Sharma, R. Jindal, and M. D. Borah, "Blockchain-based cloud storage system with CP-ABE-based access control and revocation process," *The Journal of Supercomputing*, vol. 78, no. 6, pp. 7700–7728, Apr. 2022, https://doi.org/10.1007/s11227-021-04179-4.

[15] I. Sukhodolskiy and S. Zapechnikov, "A blockchain-based access control system for cloud storage," in *2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, Moscow, Russia, Jan. 2018, pp. 1575–1578, https://doi.org/10.1109/EIConRus.2018.8317400.

[16] M. Ali, N. Q. Soomro, H. Ali, A. Awan, and M. Kirmani, "Distributed File Sharing and Retrieval Model for Cloud Virtual Environment," *Engineering, Technology & Applied Science Research*, vol. 9, no. 2, pp. 4062–4065, Apr. 2019, https://doi.org/10.48084/etasr.2662.

[17] V. Malamas, P. Kotzanikolaou, T. K. Dasaklis, and M. Burmester, "A Hierarchical Multi Blockchain for Fine Grained Access to Medical Data," *IEEE Access*, vol. 8, pp. 134393–134412, 2020, https://doi.org/10.1109/ACCESS.2020.3011201.

[18] L. Yan, L. Ge, Z. Wang, G. Zhang, J. Xu, and Z. Hu, "Access control scheme based on blockchain and attribute-based searchable encryption in cloud environment," *Journal of Cloud Computing*, vol. 12, no. 1, Apr. 2023, Art. no. 61, https://doi.org/10.1186/s13677-023-00444-4.

[19] X. Li, H. Wang, S. Ma, M. Xiao, and Q. Huang, "Revocable and verifiable weighted attribute-based encryption with collaborative access for electronic health record in cloud," *Cybersecurity*, vol. 7, no. 1, Mar. 2024, Art. no. 18, https://doi.org/10.1186/s42400-024-00211-1.

[20] Z. Wu, Y. Zhang, and E. Xu, "Multi-Authority Revocable Access Control Method Based on CP-ABE in NDN," *Future Internet*, vol. 12, no. 1, Jan. 2020, Art. no. 15, https://doi.org/10.3390/fi12010015.