Wireless Networks (2022) 28:1541–1562
https://doi.org/10.1007/s11276-022-02924-y

ORIGINAL PAPER

A blockchain and smart contract-based data provenance collection and storing in cloud environment

Amrita Jyoti[1] · R. K. Chauhan[2]

Accepted: 7 February 2022 / Published online: 5 March 2022

Abstract

Data uploading needs security and privacy in the cloud. But there are some problems like centralized provenance data (PD) collection, storage, lack of security, integrity, and more time consumption. There are methods like Rabin, Knapsack, McEliece, Elagamal, and Rivest–Shamir–Adleman for the generation of keys but it increases the encryption and decryption time and less security. Therefore, the blockchain and smart contract-based data provenance (BSCDP) Architecture is proposed for providing secure storage in the cloud environment. Initially, the fingerprint biometrics and physically uncloneable functions (PUF) have been used in verification process. The combination of PUF and fingerprint biometrics is used for secure data transmission. To protect privacy and strengthen the security, the fuzzy extractor is employed. Secondly, we use an elliptic-curve key based cyclic shift transposition cryptography algorithm for enhancing the security when sharing the key. Thirdly, we introduce the blockchain and the interplanetary file system (IPFS) for PD collection, hash computation, and storing with reduced computational overhead (CO). The integrity of data is maintained by using blockchain based secure hashing algorithm-3. By arranging fuzzy based smart contracts (FSC), the data user (DU) tracks their data. FSC is employed for tracking the history of data. The data collected is directly stored in IPFS and the DU gets a hash from IPFS to retrieve the data in the future. Finally, the data verification is done by the provenance auditor. When comparing our proposed BSCDP method with existing methods, the proposed BSCDP method achieves high security in the cloud environment for 2 K users in terms of evidence insertion time (30 ms), verification time (30 ms), response time (40 ms), total change rate (5%), CO (5.8 Kb), encryption (50 ms) and decryption time (52 ms).

Keywords Blockchain · Cloud computing · Cloud service provider · Cyclic shift transportation algorithm · Data provenance · And IPFS

Abbreviations

PD	Provenance data
RSA	Rivest–Shamir–Adleman
PUF	Physically uncloneable functions
FE	Fuzzy extractor
ECCST	Elliptic-curve key based cyclic shift transposition
IPFS	InterPlanetary file system
CO	Computational overhead
SHA-3	Secure hashing algorithm-3
FSC	Fuzzy based smart contracts
DU	Data user
PA	Provenance auditor
CE	Cloud environment
CSP	Cloud service provider
PoS	Proof-of-stake
CGA	CryptoGenetic algorithm
ISA	Integrity service application
VM	Virtual machine
BPAS	Blockchain based public auditing arrangement for verification of integrity of large data storage in the cloud
DO	Data owner
MHT	Merkle hash tree
BHA	Blockchain based hybrid algorithm
ECC	Elliptic curve cryptography
AES	Advanced encryption standard
SC	Smart contracts
SO	Shifting operations
RS	Row shift

✉ Amrita Jyoti
amritajyoti2013@gmail.com

[1] Kurukshetra University, Kurukshetra, Haryana, India

[2] Department of Computer Science & Application, Kurukshetra University, Kurukshetra, Haryana, India

CS	Column shift
DS	Diagonal shifting
CoC	Chain of custody
PoO	Proof of ownership
SCFS	Self-certifying file system
RTA	Response time analysis
AS	Authentication server

List of symbols

M_i	User
I_i	Identity
B_i	Fingerprint biometric template
CH_i	Challenge
P_i	PUF outputs
FA	Auxiliary data
L_{us}	Secret key
h (.)	One way hash function
PUF_i	Physically uncloneable function of M_i
$\|\|$	Concatenation operation
$\oplus$	Exclusive-OR operation
S	Random number
a and b	Integers
P_{ad}	Padding
P_{fun}	Permutation function
R_{ate}	Rate
O_l	Output length
$U_{ser(1)}$	User
$D_{ata(1)}$	Data
$T_{ime(1)}$	Time
T_{hgpt}	Throughput
T_v	Total valid transactions
T_t	Total time in seconds
e, f	Row and column
$T_{ransaction}$	Transaction

1 Introduction

Cloud computing is a method of storing and accessing data on a single server. It gives various services for the application of storage of data and networking. Many organizations are employing this technique because of its expediency and on-demand service [1, 2]. The two main key benefits of cloud computing are user-friendly and cost-effective. The DU can store data remotely and also access cloud applications [3, 4]. This technique allows multiple DU to use a single server for both accessing and storing their data. These are the substantial factors in adopting cloud computing [5, 6]. Moreover, data security is very important in cloud computing, which provides security for cloud users. The COVID-19's rapid global spread has increased the amount of data gathered from a variety of sources. Working from home mainly depends on cloud computing applications that help employees to do their jobs quickly and efficiently. In the COVID-19 pandemic crisis, the cloud computing environment is an unsung hero. The increase in the usage of cloud computing applications leads to security risks. Today in this world crisis, cloud computing have security risks. In cloud computing, most commonly occurring security risks are loss of data, Insecure APIs and Hacked interfaces, data breach, account hijacking, and spectre and meltdown. APIs are the most convenient way to interact with most cloud services. Only a few cloud computing services are open to the public. Because these services can be accessed by third parties, there is a risk that they will be harmed by hackers. Data Breach is the process through which confidential data is accessed, viewed, or taken by a third party without authorization, resulting in the hacking of an organization's data. In cloud computing, account hijacking is a severe security concern. It is the act of hackers stealing an organization's or a user's cloud account (bank account, social media account or e-mail account,). Hackers misuse the hacked account to carry out illegal acts. Spectre and Meltdown enable applications to monitor and steal data presently being processed on a computer. It's accessible with desktop PCs, cloud, and the mobile devices. It can save your password, as well as personal information like photographs, emails, and business documents, in the memory of other programs that are now operating. Provenance is used to determine the data history of the original file [7, 8]. These PDs cover confidential information about the DU and original data [9, 10]. Most of the existing provenance services are susceptible to malicious forgery or corruption of PD. A trusted system, such as a Cloud Service Provider (CSP), normally saves the PD, which is a centralized database [11, 12].Moreover, we cannot entirely trust the CSP since it may tamper with the data. For instance, it can hide unauthorized access by altering the PD. To overcome this issue, we can store the PD in a distributed manner [13, 14].

To address the above-said problem, Blockchain is a suitable, decentralized technology. Blockchain can boost trust in the provenance of data in the cloud computing environment. It is a distributed public ledger in which all transactions are perceived and verified [15, 16]. In this decentralized architecture, each node in the network provides better efficiency and availability. This blockchain decentralized architecture has guaranteed data provenance proficiency for a cloud computing atmosphere [17]. This data provenance is cloud-based and blockchain-based, permanently recording all data operations [18].As a result, trust between the DU and the CSP can be established. Furthermore, keeping the provenance improves cloud users' trust in cyber threats. Furthermore, the storage of

huge data over the blockchain is more expensive because of its distributed nature [19, 20]. This issues can be solved by using a system like the decentralized data saving system, IPFS. Some traditional methods, such as PASS, SPROVE, Provchain, Crab, file provenance systems, and so on [21, 22], have specific problems with data integrity. To overcome this issue, we proposed a BSCDP method in the cloud environment.

1.1 Problem statement

The problems with authentication are the leakage of passwords, user IDs, and biometric information. Algorithms such as Rabin, Knapsack, McEliece, Elagamal, and RSA require more time to generate keys. The cloud storage, like the Third Party Auditor (TPA), is centralized, and the integrity is verified by requesting CSP, resulting in a high computation power. When a large amount of data is added to a blockchain network, every node has the same ledger information. As a result, the provenance of collection and storage are difficult issues to address. These problems are solved in our proposed BSCDP architecture.

1.2 Contributions

1.2.1 The contributions of the proposed BSCDP methodology are as follows:

- DU verification is the main process for secure data transmission. This is done by combining PUF and fingerprint biometric scheme. To protect privacy and strengthen security, FE is employed.
- By utilizing the ECCST algorithm, data security is provided by the generation of keys.
- The block chain based on SHA-3 maintains the integrity of data and the history of data is tracked by utilizing FSC in the block chain.
- The blockchain based IPFS system stores the data in distributed manner.

The rest of the section is summarized as follows: Sect. 2 presents recent works. Section 3 represents the architecture of the proposed BSCDP approach and briefly describes the proposed system. Section 4 presents experimental results attained by the proposed method along with its other comparative methods. At last, the overall conclusion of our work are presented in Sect. 5.

2 Related work

This existing work related to our research is discussed in this section on the basis of security, integrity, authentication, and privacy for secure storage in the cloud.

Tosh et al. [23] introduced a Block-Chain on account of the information provenance structure for the cloud to avoid vampire attacks in the integrated cloud environment. This method is utilized for auditing data objects transferred by the cloud user in a tamper-resistant manner. Additionally, the proof-of-stake (PoS) protocol was introduced on account of a consensus protocol, which includes a stake in the computing, storage, and networking resources of a cloud operator. The advantage of this method is to decrease the consensus delay. The disadvantage of this method is the decrease in safety properties.

Tahir et al. [24] introduced the CryptoGenetic algorithm (CGA) for solving problems in security. This method contains two modes of working; downloading and uploading data from storage in the cloud. Initially, by utilizing the algorithm of Caesar, cipher input data is encrypted and then completed encryption of the first level, the character of the 8-bit binary conversion is done. Then a 128-bit random key is produced and, by utilizing the required key, the binary data is encrypted. For downloading data from cloud storage, the same process is reversed. This method takes less time but has a higher level of spatial complexity.

Kumari and Kamal et al. [25] introduced a model for Integrity Service Application (ISA) with avoidance of cryptanalytic attacks. The security and integrity issues are solved by the process of hybrid verification of secret keys and authentication. To achieve security in the encryption process, a key is generated. The owner of the data generates a key and sends it to the client to download the file, which is sent by the data owner. Verification of the secret key process is done by the secret key of the client is valid with the database secret key. After verification of the secret key, further verification of the client is done with the avoidance of cryptanalytic attacks. The integrity ensures the encrypted file's integrity using the method of proof of retrievability. This method can be used for various formats of files, but this method is complex.

Li et al. [26] presented a blockchain based public auditing arrangement for verification of the integrity of large data storage in the cloud (BPAS). The data owner (DO) sends the files to the cloud and saves the tags for the corresponding files on the blockchain. Then, as a PA, select a user from the blockchain to verify its file using its public key. The proof is returned by PA for this file by generating a Merkle Hash Tree (MHT) for the hash tag saved on the blockchain. The previous hash value of the block is

estimated by utilizing the SHA-256 algorithm. On the other hand, a challenge is sent by DO to the CSP and, by using the encrypted file, the hash tag is generated by MHT as proof, which is returned by the CSP. By comparing the proof of PA and CSP, the DO checks its files quickly and efficiently. During the process of auditing, the blockchain can resist the false behavior of DO or CSP in the proposed method. This method has a low computation cost. However, security on the blockchain is not provided.

Darwish et al. [27] introduced a blockchain based hybrid algorithm (BHA) to reduce the inefficiency of privacy in cloud storage. The two algorithms are elliptic curve cryptography (ECC) and Advanced Encryption Standard (AES), which is combined with blockchain for solving privacy problems. For maintaining data integrity, the cloud infrastructure is designed with a framework that encrypts the data by utilizing an algorithm followed by the technology of blockchains. This method gives high security, but time consumption is higher.

Tajammul and Praveen et al. [28] offered an algorithm for encrypting data automatically. Initially, every data is identified by the algorithm and generates a single matrix, which is utilized further for decrypt data. If the key is lost or stolen, they cannot encrypt data because only the user has the algorithm and proper coding. The encrypted data is uploaded to the cloud storage, and the key is saved on the local server for future decryption. This method reduces the time for users when encrypting data. However, the computational cost is high.

Huang et al. [29] presented a blockchain for the privacy of medical data and the availability of data among research institutions and patients. Smart contracts with zero-knowledge proof validation automatically verify the medical data of patients that meet the given requirements without knowing the patients' privacy. Verification mechanism of proxy re-encryption is utilized for encrypting medical data, which is in the form of cipher text. It can be decrypted only by approved research institutions. This method ensures the privacy of medical data. However, these conspiracy attacks are unstoppable. Table 1 illustrates the summary of existing and proposed methods.

3 Proposed BSCDP system

The proposed architecture is shown in Fig. 1. First, the fingerprint scanner in the proposed BSCDP system captures the user's fingerprint. Then the fingerprint template is converted into auxiliary encrypted data by FE, which is given to CSP, so the leakage of biometric information is avoided. After registration, the keys are generated and data is encrypted using ECCST algorithms. The PD is then stored on the blockchain and IPFS using the SHA-3 algorithm, and the hash value is generated. The hash value is stored in the blockchain and the data is stored in IPFS and it responds with an IPFS hash to retrieve data in the future. We also integrate blockchain Smart Contracts (SC) with IPFS to develop decentralized cloud storage for better DU access organizations. Our method permits DUs to track their history of data using FSC. Finally, the PA verifies the data's validity by checking the mapping file of IPF and the blockchain, as well as the data's validity for the user.

3.1 Registration phase

In the public cloud, a system of biometrics is utilized to avoid unauthorized users. User is necessary to complete the procedure of registration in between the CSP and user M_i. PUF is utilized in verification protocols. The fuzzy extractor is used to provide security in biometric data. The steps for registration are given below.

Step 1: User M_i takes an identity I_i and the fingerprint is given as input to the scanner device. Then, from the input fingerprint, the fingerprint biometric template B_i is extracted by M_i and randomly produces a random number 'S' and a challenge CH_i.

Step 2: The PUF outputs are estimated by M_i,$P_i = PUF_i(CH_i)$, then by using the process of $FE.Gen(\cdot)$, the auxiliary data FA and the secret key for the user L_{us} from the fingerprint biometric template is obtained i.e., (L_{us},FA) = FE.Gen(B_i). Then M_i evaluates $K = h(I_i \| L_{us}), CH_i^* = CH_i \oplus h(L_{us})$ and $AI_i = I_i \oplus h(L_{us} \| S)$. Finally M_i sends $\{AI_i, (CH_i^*, P), K\}$, also besides to registration request R_{req} to the CSP. The L_{us} is not disclosed to anyone because it is for the biometric of user only.

Step 3: The CSP first checks the AI_i uniqueness in the request sent by the user M_i. Then the CSP produces a unique number f for the user and a private key L_{csp} and random number e. Then CSP evaluates $F_i = h(L_{csp} \| e) \oplus K$. Finally the CSP concludes the registration by sending F_i to M_i.

Step 4: After sending F_i by CSP, M_i evaluates the secret information $U = h(I_i \| L_{us} \| S), S^* = S \oplus h(I_i)$ and $FA^* = h(I_i \| S) \oplus FA$ for future safe communication. Finally, the user saves the $\{h(.), F_i, U, S^*, FA^*\}$ into the scanner device. We didn't use passwords and store the information of biometric directly so our method is more secure.

Pseudocode-1

Input: Fingerprint

Output: Encrypted auxiliary data

Begin

Generate B_i, I_i, and CH_i

For every user

Compute: $P_i = PUF_i(CH_i)$

$$(\mathrm{L_{us}}, \mathrm{FA}) = \mathrm{FE.Gen}(B_i)$$

$$K = h(I_i \| L_{us}), CH_i^* = CH_i \oplus h(L_{us})$$

$$AI_i = I_i \oplus h(L_{us} \| S)$$

If the uniqueness of AI_i is checked

Then**,** generate a private key

Compute: $F_i = h(L_{csp} \| e) \oplus K$

Stores: $\{AI_i, (CH_i^*, P), f, L_{CSP}\}$

After receiving the secret message from the CSP

User M_i Compute: $U = h(I_i \| L_{us} \| S), S^* = S \oplus h(I_i)$

$$FA^* = h(I_i \| S) \oplus FA$$

Then store $\{h(.), F_i, U, S^*, FA^*\}$

End for

End If

3.2 ECCST based secure data sharing phase

After registration, the key is generated and data is encrypted by using the ECCST cryptography algorithm. ECCST is the combination of ECC and CST algorithm, in which keys are generated from elliptic curves over galois field and the CST performs the encryption and the decryption process.

3.2.1 Key generation

In this algorithm, two keys are generated for encryption and decryption. The ECC is a fast process for the key generation. This algorithm gives security at a high level. The key generated in ECC is exchanged by the system with the CSTA encryption method. When comparing with other methods this method gives high security in encryption and unable to decrypt by the un-authorized owners. The uploaded data bytes are extracted by the system. The key generated is shared for encryption at the destination end. CST is a cryptography technique with a symmetric key that contains varying key sizes and block sizes. By using shifting and partition operations, the plaintext is converted into a ciphertext. The DU partitions the file content into $p \times p$ matrix and then shifting operations (SO) like Row Shift (RS), secondary diagonal shift, column shift, and the primary diagonal shift is performed. The ciphertext is the output produced, which is sent into the CSP. For taking back the original file, the operation is performed in reverse at the receiver side.

Table 1 Summary of existing and proposed methods

Author	Year	Method	Authentication	Privacy	Security	Integrity
Tosh et al. [23]	2019	Blockchain-based data provenance framework	SHA-256	–	Blockcloud	–
Tahir et al. [24]	2020	CGA	–	Cryptographic algorithm	–	Cryptographic algorithm
Kumari and Kamal et al. [25]	2020	ISA	Secret key	–	Advanced encryption standarsd	Proof of retrievability method
Li et al.[26]	2020	BPAS	–	–	–	blockchain
Darwish et al.[27]	2020	BHA	–	Blockchain based ECC and advanced encryption standard	–	cloud infrastructure
Tajammul and Praveen et al. [28]	2020	AES	–	–	AES algorithm	–
Huang et al. [29]	2020	Blockchain based privacy preserving scheme	Smart contracts with zero-knowledge proof validation	–	proxy re-encryption technology	Blockchain
Proposed		BSCDP	Biometrics and PUF	Fuzzy extractor	Elliptic-Curve Key based Cyclic Shift Transposition	Blockchain based SHA-3

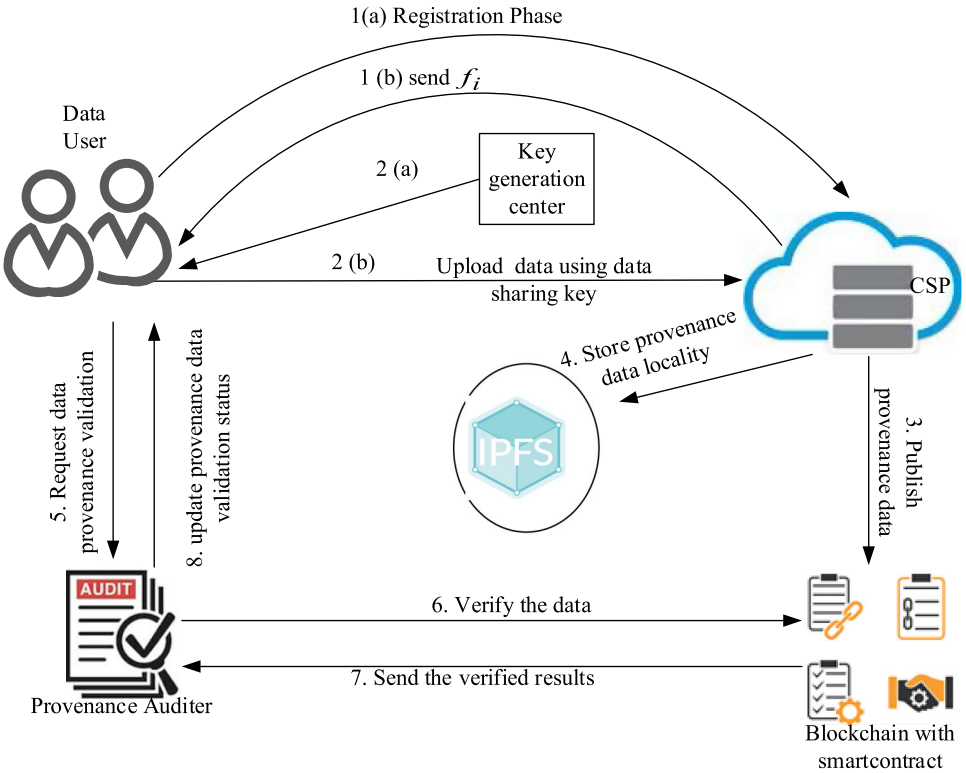

Fig. 1 Proposed architecture of secure provenance data in the cloud

3.2.2 Encryption process of ECCST

The ECC depends on a group structure indeed on an elliptic curve. The cubic equation for real numbers in an elliptic curve is determined using Eq. (1).

$$C^2 = Y^3 + aX + b \tag{1}$$

where $4a^3 + 27b^2 \neq 0$, a and b are the integers. A set of points on the elliptic curve together with an infinity point. For any point on an elliptic curve, an abelian group can be set.

The finite prime field in elliptic curve is determined in Eq. (2)

$$C^2 = Y^3 + aX + b \,(\mathrm{mod}\, P) \tag{2}$$

The elliptic curve shape can be defined by elements of a finite field (a, and b) and $4a^3 + 27b^2 \neq 0 \,(\mathrm{mod}\, P)$, where P denotes a prime number. After key generation, the key is swapped with the CST algorithm [30]. The plaintext of the input is segmented into $p \times p$ matrix where the p value is changed based on the input message size. The plaintext contents are evenly distributed into columns and rows. In this method, by utilizing shifting and partition operations, the file is encrypted. The ciphertext C (n) is the result, which is given to the CSP. The input plain text is divided into $p \times p$ matrix. Initially, SO performs Column Shift (CS) in a specific cyclical order from the lower to the upper end. Secondly, SO is applied to the resultant blocks cyclically from right to left. Thirdly SO is performed in the primary diagonal from the right lower to the left upper in a specific order. Then SO performs in the secondary diagonal from left lower to the right upper in a specific order. Finally, the obtained output is sent to the CSP. The shifting order is based upon the number of times each element is shifted in the format of the matrix. During encryption, the key and block sizes, and the key size is 2^{2P+1} are selected. Figure 2 illustrates the encryption process.

Consider A_{in} as input file. Firstly, we divide the file A_{in} into the format of $p \times p$ matrix. Then, the CS operation is performed and the operation is done based on Eq. (3).

$$A'_{e,f} = A_{e+shift(e,p_b)\mathrm{mod}P_{b,f}} \tag{3}$$

where e, f signifies the row and column and the key value is depended by $shift(e, p_b)$ it changes from 0 to 9. The number of elements cyclic shift denotes the key values and mod operation denotes the modular arithmetic. RS is performed and it is given in Eq. (4).

$$A'_{e,f} = A_{e+shift(e,p_b)\mathrm{mod}P_{b,}} \tag{4}$$

Then the Diagonal Shifting (DS) operation is performed, i.e., diagonal elements are shifted from the upper to right lower. It is represented in Eq. (5).

$$A'_{e,f} = A_{e+shift(e,p_b)\mathrm{mod}P_{b,f+shift(e,p_b)\mathrm{mod}P_b}} \tag{5}$$

Then a secondary DS operation is performed. This is given in Eq. (6).

$$A'_{e,f} = A_{(e-1)\mathrm{mod}P_{b,f}} \tag{6}$$

Then, in a particular order, output is found, which is given in Eq. (7).

$$A'_{e,f} = A_{(e+(P_{b-1})),f} \tag{7}$$

By changing the output into the format of ASCII, the encrypted text is obtained. Finally, the obtained output is sent to the CSP.

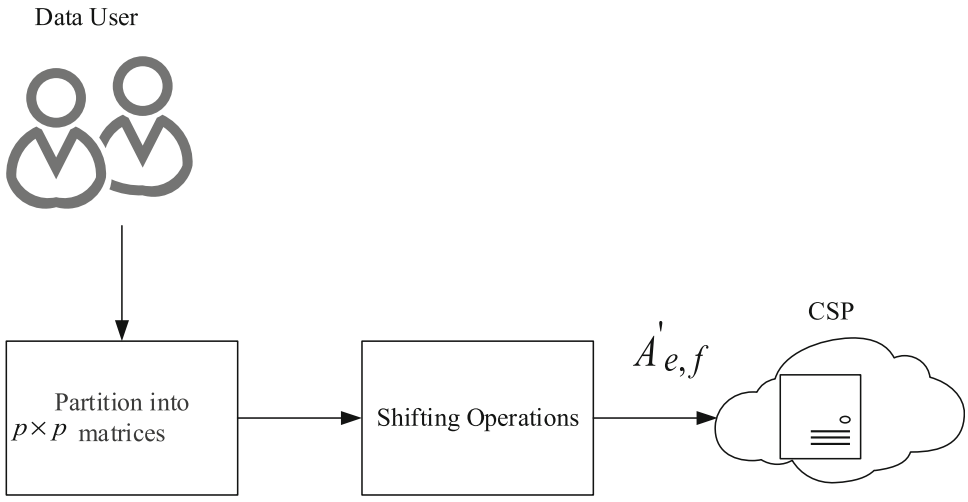

Fig. 2 Process of encryption [31]

Algorithm 1: Encryption

Input: The file into the matrix format

Output: Cipher text

Start

Elliptic curve real number is determined in Eqn. (1)

Key is generated by using Eqn. (2)

For encryption process

Perform CS operation using Eqn. (3)

Row shift operation using Eqn.(4)

Diagonal shift operation using Eqn. (5)

Secondary diagonal shift operation using Eqn. (6)

End for

Output is found using Eqn. (7)

End

3.2.3 Decryption process of ECCST

Decryption is the inverse of the encryption process. Therefore, by using the key of decryption, the ciphertext is changed into original plaintext, and by using QR-code plaintext is send to DO, and operations of partition and shifting are performed in the inverse order. Based on key size and block size, the CST improves data security. Based on the application's security requirements, the key size varies between as small, medium, and high. For transmission of data, the DU constantly fixes the key size that can be utilized for several DOs.

The output is changed into the format of ASCII for receiving the encrypted file. Next, in a specific order, row and column is shifted. Then, the matrix is shifted diagonally and the secondary diagonal shift operation is performed. Finally the decrypted file is found.

Algorithm 2: Decryption

Input: Cipher text

Output: Plain text

Start

Convert output into ASCII format for getting encrypted file

For round

Shift the row operation using Eqn. (4)

Shift the column operation using Eqn. (3)

shift the matrix diagonally

Perform secondary diagonal shifting using Eqn. (6)

End For

Obtain decrypted file

End

The ECC generates a key pair, which is a private key and a public key. The Fig. 3 illustrates the decryption process. A public key is given to CSTA and the data is encrypted. The DU uploads the encrypted data with the key to the CSP. The authorized owner can only download the data by utilizing the private key. The architecture flow is shown in Fig. 4 and ECCST flowchart in Fig. 5.

3.3 Provenance data collection and storing phase

It mainly contains two steps, they are (i) PD collection and (ii) storing PD in IPFS. The blockchain is utilized to store the PD hash value and IPFS is for saving the PD. The FSC track the history of PD which is provided in blockchain.

3.3.1 Provenance data in the blockchain

After the encryption process, the PD is given to the blockchain and IPFS. The main challenging issues in existing methods are PD collection and storage. For solving these challenges, we proposed an FSC based blockchain

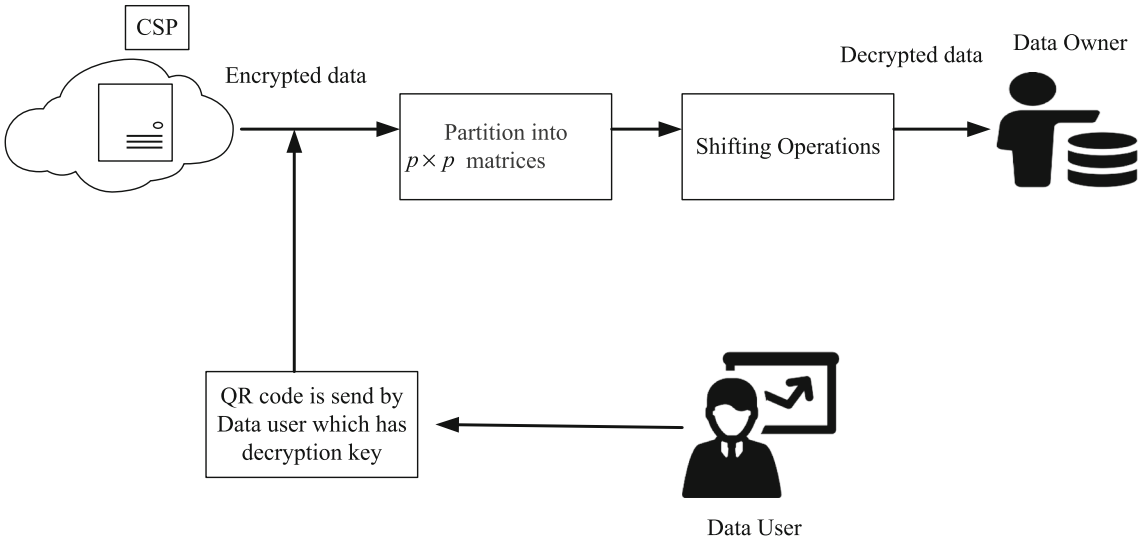

Fig. 3 Process of decryption [31]

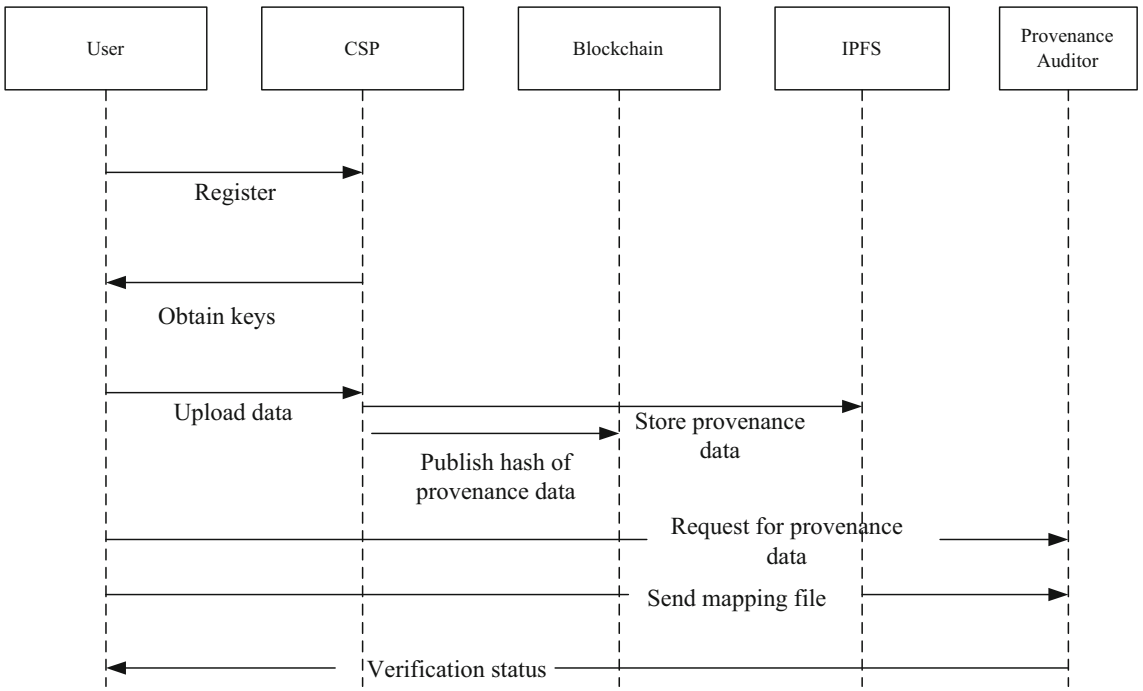

Fig. 4 Architecture flow

method to collect PD and storing. The SHA-3 algorithm calculates the hash value for PD and saved in blockchain. In addition to this saving, the blockchain will reply with a blockchain ID (log of proof, hash value of user ID). The IPFS network stores the PD and IPFS will reply with an IPFS hash. SHA-3 utilizes the sponge construction domain extender, which works on the fixed permutation by adjusting exchange specific security assets for improving productivity, producing variable range of outputs [32]. Some important explanations in the cloud are,

- Chain of custody (CoC): Digital evidence is explained as the process of sequential documenting and verifying the data handling history. CoC is continued in our proposed work because every action in PD is saved in the blockchain.
- SC: SC is a program in computer, which is utilized for finding the data history automatically. In this research, rules of fuzzy are used to optimize SC.
- Data provenance: The ownership history is recorded and storing data provenance is done by the blockchain.

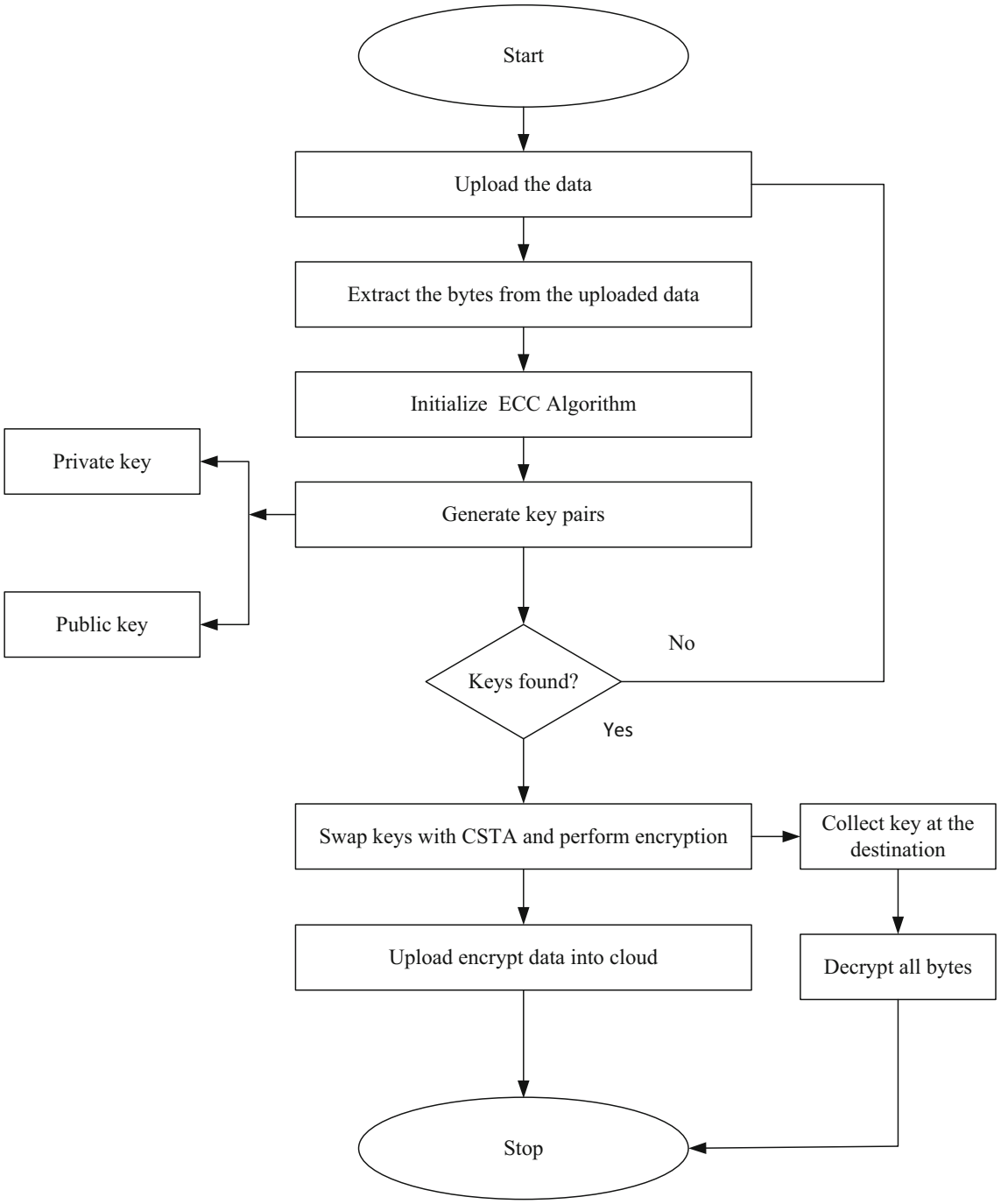

Fig. 5 ECCST flow chart

In our research data, every modification is saved in the blockchain and tracked by the FSC.

- Proof of ownership (PoO): PoO is explained as digital evidence of ownership where different users can control the data during its lifetime. When the data ownership is altered then the present owner employ the data for storing PoO in the cloud. The change in ownership is saved in the blockchain as the history of the data.

IPFS is a storage system of peer-to-peer file for connecting all devices for computing in the files of the same system. It is the storage system of decentralized files that operates in the distribution of different technologies in the web, such as the Self-Certifying File System (SFS), Git, etc. The system of peer-to-peer file allows the data for recovering and saving the file from the network. Since the data is saved as content-addressable in IPFS therefore, once some data is saved in IPFS it responds with a hash for

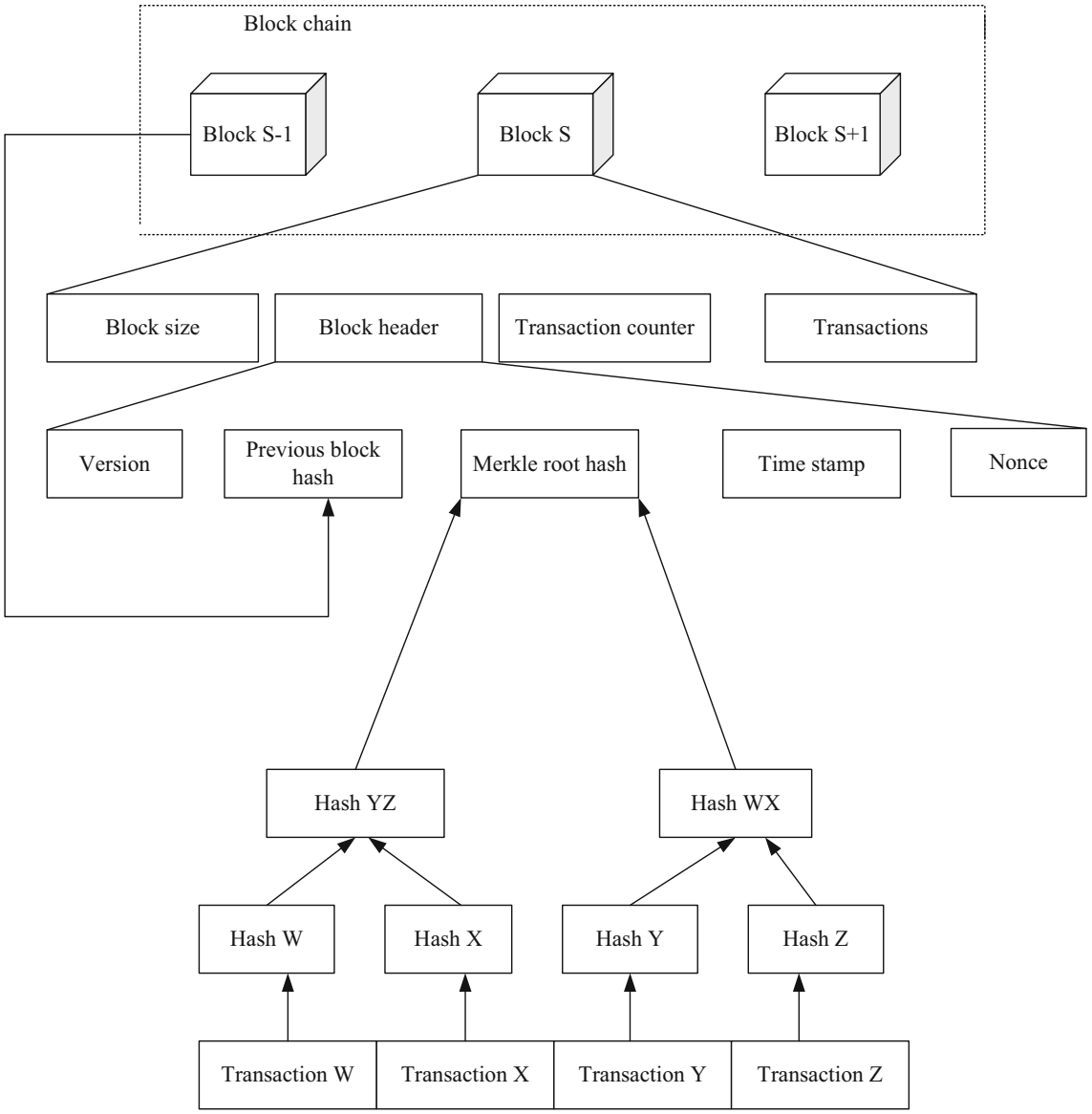

Fig. 6 Structure of blockchain

saving the data. For retrieving data in future, this hash can be utilized. Here, the exact position of the file can be found by using a specific file address. The Blockchain ID (log of proof, hash value of user ID) and IFPS hash are saved in the mapping file. Fig.6 illustrates the structure of the blockchain.

Initially, the PD of the hash value is calculated by utilizing the SHA-3 algorithm and it is stored in the blockchain. In SHA-3, every block hash value is calculated using Eq. (8).

$$H_{ash=S_{po}}[P_{fun}, P_{ad}, R_{ate}](T_{ransaction}, O_l) \tag{8}$$

where $T_{ransaction}$ denotes the transaction with the function of padding P_{ad}, P_{fun} signifies the permutation function, R_{ate} denotes rate and O_l signifies the output length. By using Eq. (6), the hash value is produced by the construction of the sponge procedure in SHA-3. Let us assume DU $U_{ser(1)}$ saves data $D_{ata(1)}$ at $T_{ime(1)}$ time in the cloud. Then the block is formed for data $D_{ata(1)}$ and SHA-3 generates a hash value. From the time of block creation, for every transaction, FSC tracks the data modified in the data $D_{ata(1)}$. In the blockchain, every modification is saved as evidence and distributed in the network of the blockchain. The log of proof contains the ID of the user who made the data transaction, accessing time, IP address, and details of hardware (file deletion, Virtual Machine (VM) logs). Figure 7 illustrates the representation of FSC. FSC keeps track of all important activities done in PD. Therefore, all proof is saved and collected in the blockchain.

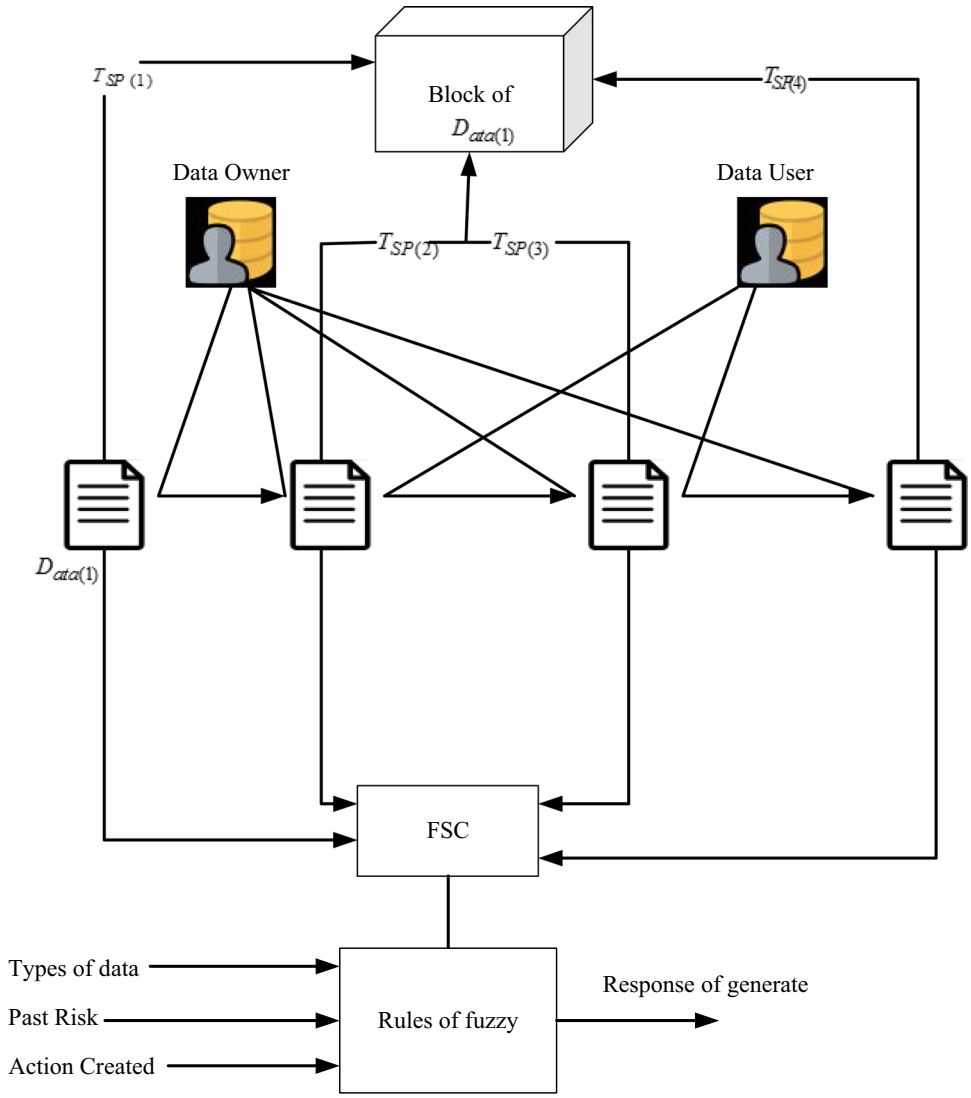

Fig. 7 The architecture of FSC

Table 2 Fuzzy rules of FSC

Types of data	Previous risk	Action performed	Value of fuzzy	Report generation
Non-delicate	L	Read	0–0.5	N
delicate			0–0.5	N
Non- delicate	H	Read	0–0.5	N
delicate			0.51–0.1	Y
Non- delicate	L	Edit	0–0.5	N
delicate			0.51–0.1	Y
Non- delicate	H	Edit	0–0.5	N
delicate			0.51–0.1	Y
Non- delicate	L	Delete	0.51–0.1	Y
delicate			0.51–0.1	Y
Non- delicate	H	Delete	0.51–0.1	Y
Delicate			0.51–0.1	Y

Table 2 shows the rules of fuzzy arranged in FSC. Here, the previous risk indicates the change made in data during former access. The FSC is worked on basis of data delicate level. If the information is non-delicate and the previous risk is low, the access login proof is removed and the result is not produced. Else the result made is taken as important proof and saved in the blockchain. Here 'L' denotes low, 'Y' denotes yes, 'N' denotes no, and 'H' denotes high.

3.4 Request for data provenance

After entering the mapping file, it consists of an IPFS hash and Blockchain ID (log of proof, hash value of the user ID). The DU request for the data validity to the PA and along with this request, DU also sends the mapping file to the PA.

Pseudocode 2: Provenance data Collection

Input: Data of the user

Output: digital evidence

Start

For all $U_{ser(i)} \in U_{ser}$

Create FSC for users

End for

For every data

$U_{ser(1)}$ saves in cloud $D_{ata(1)}$

Create $D_{ata(1)}$ block

By utilizing Eqn. (8) Calculate Hash ($D_{ata(1)}$).

Track $D_{ata(1)}$ and update the evidence

End for

For every $D_{ata(1)}$ transaction

Store action made, source IP time stamp, etc.

If (fuzzy rules are violated) // FSC

Generate report

Else

Report is not generated

End if

End for

End

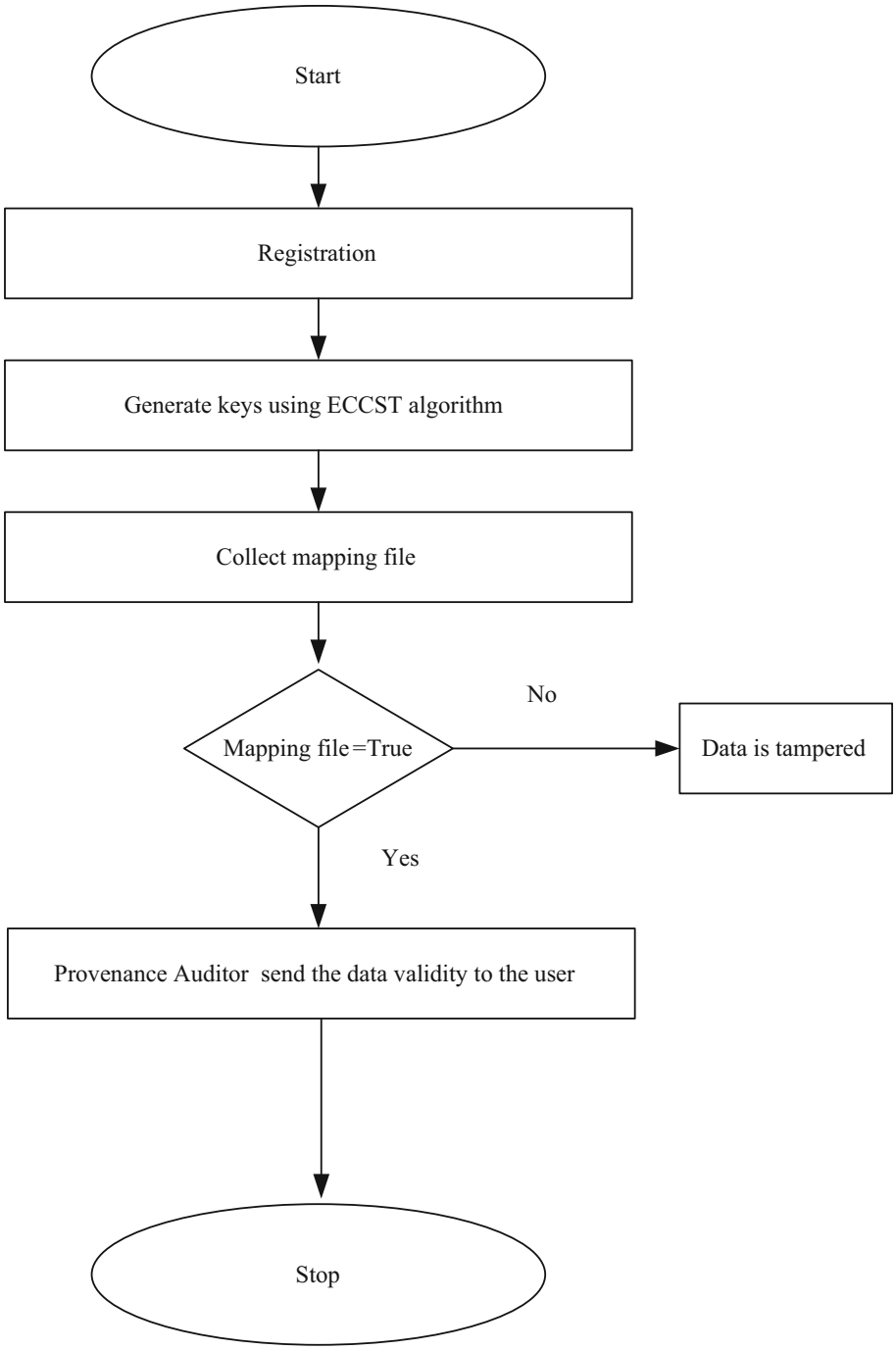

Fig. 8 Flow chart for provenance data

3.5 Verification phase

After accessing the mapping file, the data is requested by PA from the blockchain and IPFS. Then the data saved in the IPFS is cross-checked and the data validity to the DU is updated by the PA. Fig. 8 shows the flow chart for provenance data.

Algorithm 3 Provenance algorithm

Input: Mapping file

Output: data is tampered or not

By sending its mapping file validation for PD can request by user-validation (Report)

Validate by Provenance auditor

Validate-Report

For all request entry in **Report**{

If (Valid (Blockchain id=true)

R= Blockchain - hash (Blockchain_id)

S= IPFS value (IPFS- hash)

If (R= SHA3(Y))

False; data is tampered

True; All good

End for

End If

End If

Pseudocode 3: data provenance

Input: Data

Output: Provenance data

Begin

Registration using biometrics and PUF

Generate keys using ECCST algorithm

Collect mapping file (blockchain ID and IPFS hash)

IF (mapping file=True)

P.A send the data validity to the user

Else

Data is tampered

End if

End

4 Experimental results and setup

The proposed Blockchain and Smart contract-Based data provenance architecture using IPFS in a CE will be tested under MATLAB and its performance is to be compared with existing methods.

4.1 Experimental setup

Our proposed BSCDP method can be simulated using MATLAB by using 100 K users in the environment of the cloud. The parameters in the remaining are scheduled in Table 3. The number of rounds is selected as 24 because it is resistance to differential cryptanalysis attacks. The block size is taken as 576, which represents the security strength.

Table 3 Simulation parameters

Parameter	Value
Number of users	100
Number of Keys generated	300
Number of controllers	1
Number of AS	1
SHA-3	
Number of rounds	24
Block size	576 bits
Customized contract	FSC
Word size	64 bits
Maximum handles	2048
Cloud	
Number of VMs	24
Average bandwidth	1000,000 MB
Average random access memory	512 MB
Simulation time	100 s

The customized contract is taken as FSC because the history of data can be tracked. The average bandwidth signifies the capacity of network connection, which is represented as bits per second. The average random access memory is used to obtain the highest possible average access performance and to reduce the entire total cost of the entire memory system. These are the criteria for selecting simulation parameters.

4.2 Quantitative evaluation

Our proposed BSCDP is compared with BHA [27], ISA [25], and CGA [24] existing methods for the metrics such as evidence insertion time, evidence verification time, response time, total change rate, and Computational overhead.

4.2.1 Evidence insertion time analysis (EIT)

Figure 9 depicts the insertion time analysis. It is the analysis of the time required for the inserting biometrics of the user and the owner in the cloud server. When the user number increases, the data regarding the user of biometric also increases. Therefore, our proposed BSCDP decreases the insertion time when compared with existing methods such as BHA, ISA, and CGA. The proposed BSCDP has an insertion time of 30 ms when compared with existing methods CGA, ISA, and BHA of 38 ms, 50 ms, and 60 ms for the 2 K users. The number of keys varies from 10 to 40, the insertion time is minimum for our proposed BSCDP when compare with existing CGA, ISA, and BHA methods. The proposed BSCDP has an insertion time of 1300 ms

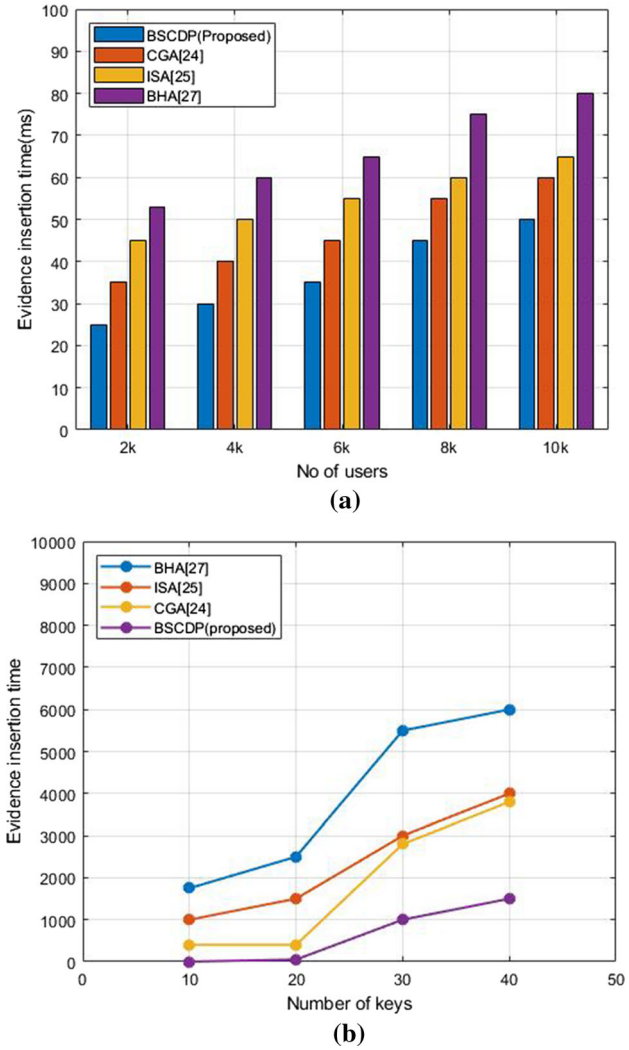

Fig. 9 Insertion time analysis **a** Number of users **b** Number of keys

when compared with existing methods of CGA, ISA, and BHA of 3900 ms, 4000 ms, and 6000 ms for the 40 keys. Therefore, the insertion of biometric information requires less time because of the PUF and FE.

4.2.2 Encryption time (ET) and decryption time (DT) analysis

Figure 10 illustrates the time analysis of encryption and decryption (a) Number of users (b) File size. By utilizing ECCST, the encryption and decryption time is minimum for the proposed BSCDP. Therefore, without time consumption, the ECCST algorithm improves security when compared with existing methods such as BHA, ISA, and CGA. The proposed BSCDP has an encryption time of 50 ms when compared with existing methods CGA, ISA, and BHA of 100 ms, 150 ms, and 250 ms for the 2 K users. The proposed BSCDP has a decryption time of

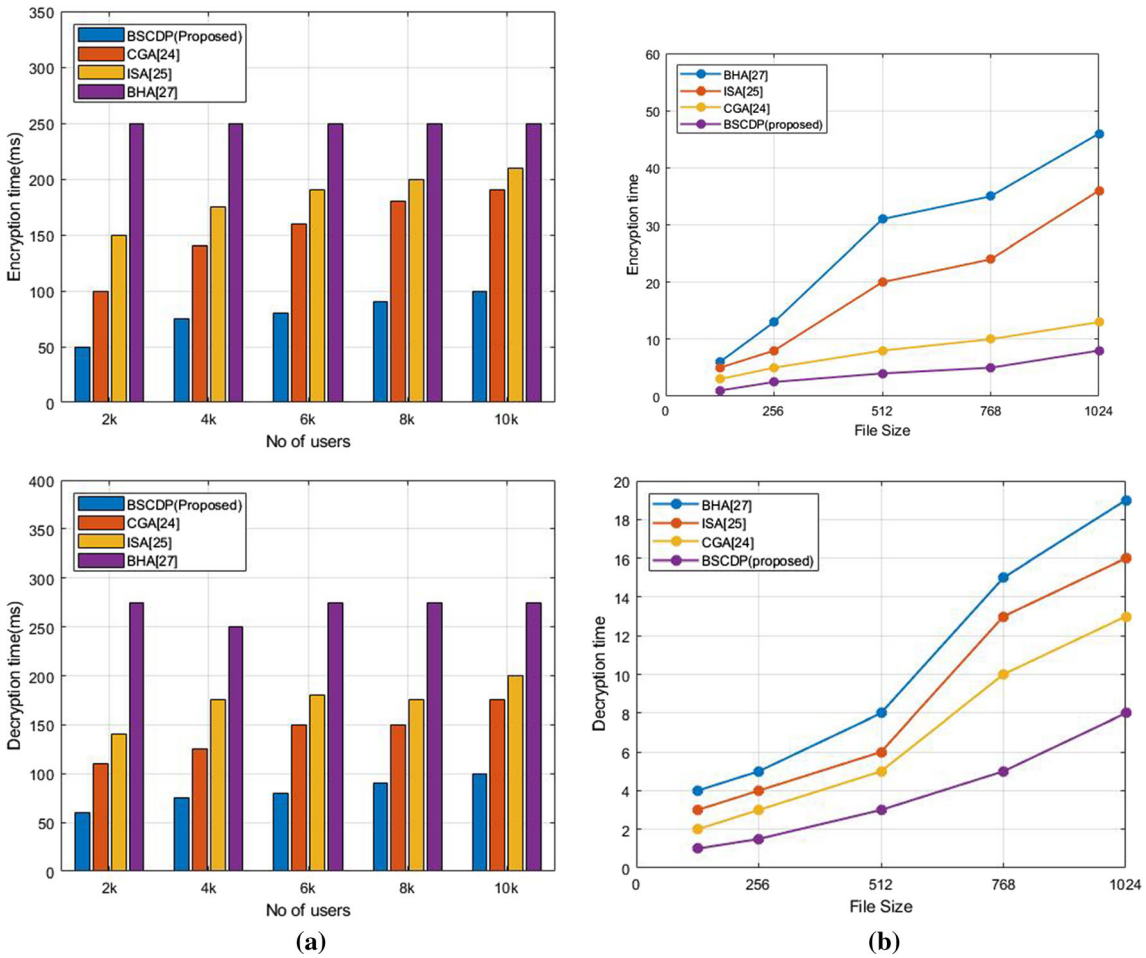

Fig. 10 Encryption and decryption time analysis **a** Number of users **b** File size

60 ms when compared with existing methods, CGA, ISA, and BHA, of 118 ms, 150 ms, and 270 ms for 2 K users respectively. Table.5 provides a comparative analysis of encryption time and decryption time with the number of users. The proposed BSCDP has an encryption time of 3 s when compared with existing methods CGA, ISA, and BHA of 5 s, 8 s, and 12 s for the file size 256. The proposed BSCDP has a decryption time of 1.8 s when compared with existing methods CGA, ISA, and BHA of 3 s, 4 s, and 4.9 s for the file size 256.

4.2.3 Response time analysis (RTA)

The time taken by the cloud server to process a request made by the user and to generate a response corresponding to the request is known as RTA. Figure 11 shows the analysis of response time. A huge amount of data can be stored by using a decentralized file on the IPFS system. As a result, any user can immediately request data from the cloud and receive a response. The data can be quickly collected by PA from the IPFS. Hence, the response time is minimal in the proposed BSCDP when compared with existing methods, BHA, ISA, and CGA. The proposed BSCDP has an RTA of 40 ms when compared with existing methods, CGA, ISA, and BHA, of 55 ms, 63 ms, and 77 ms for 2 K users respectively. The proposed BSCDP has an RTA of 700 ms when compared with existing methods, CGA, ISA, and BHA of 800 ms, 1150 ms, and 1550 ms for file size 15.

4.2.4 Total change rate (TCR) analysis

Figure 12 represents the result obtained in the analysis of the TCR. It is the ratio of the changed data to the actual data saved in the cloud. The increase in the total change rate occurs when the intruder changes the data. Data from the authorized DU is only permitted for an efficient and reliable cloud system; otherwise, it is denied. The total change rate is minimal for the proposed BSCDP because it uses FSC, which can track the history of data when compared with existing methods like BHA, ISA, and CGA. The proposed BSCDP has a total change rate of 5% when

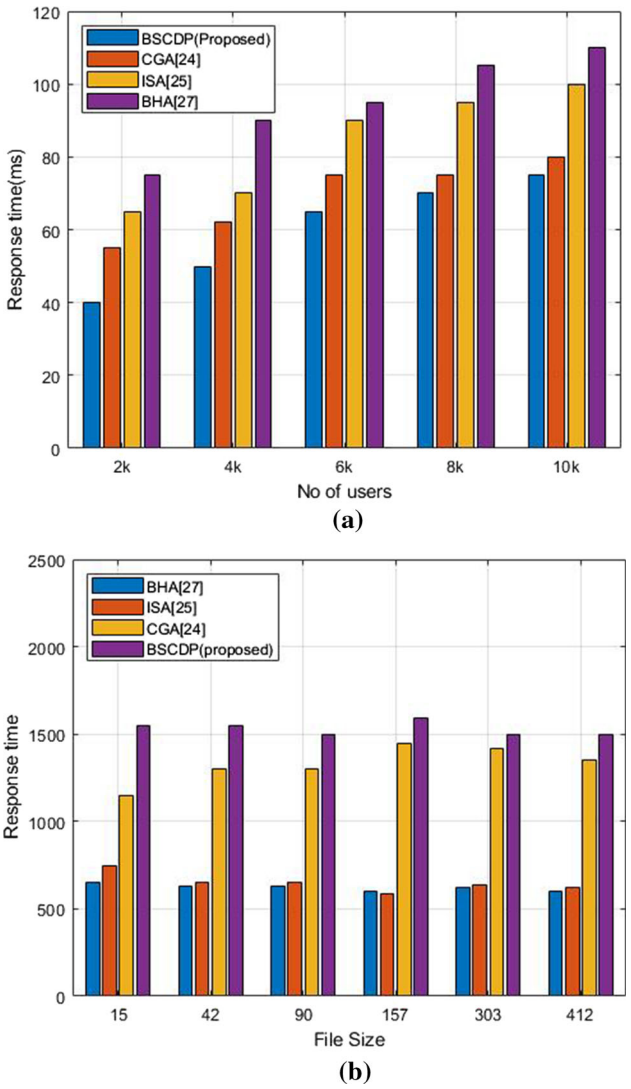

Fig. 11 Response time analysis **a** Number of users **b** File size

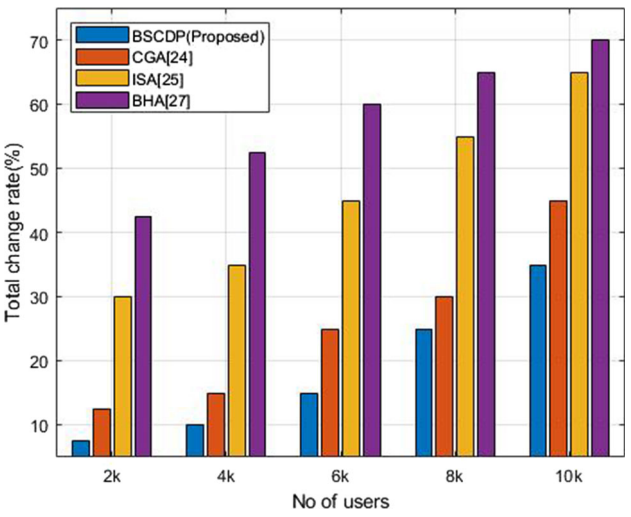

Fig. 12 Total change rate analysis

compared with existing methods of CGA, ISA, and BHA of 13%, 30%, and 43% for 2 K users respectively.

4.2.5 Computational overhead (CO) analysis

The amount of bandwidth used to complete a particular task, like authentication, reading, creation of biometrics, editing in the cloud system, is called CO. Figure 13 illustrates the analysis of CO. In the presence of TPA, a centralized file is utilized for the existing method, which increases the CO. But, the blockchain using IPFS reduces the overall CO because of the decentralized file system when comparing the proposed BSCDP with existing methods like BHA, ISA, and CGA. The proposed BSCDP has a CO of 5.8 KB when compared with existing methods, CGA, ISA, and BHA, of 8 KB, 9 KB, and 10 KB for 2 K users respectively.

4.2.6 Evidence verification time analysis (EVT)

Figure 14 illustrates the verification time analysis. It is defined as the time required for the PA to verify the information collected from the blockchain and IPFS. For better analysis, PA verifies the data. Besides, this SHA-3 algorithm is used for hash computation without time consumption. So, the proposed BSCDP attains minimum verification time compared with existing methods such as BHA, ISA, and CGA. The proposed BSCDP has an Evidence Verification time of 30 ms when compared with existing methods of CGA, ISA, and BHA of 38 ms, 50 ms, and 60 ms for 2 K users respectively. The proposed BSCDP has an Evidence Verification time of 0.2 ms when compared with existing methods of CGA, ISA, and BHA of 0.67 ms, 0.86 ms, and 1 ms for a block size of 288 respectively.

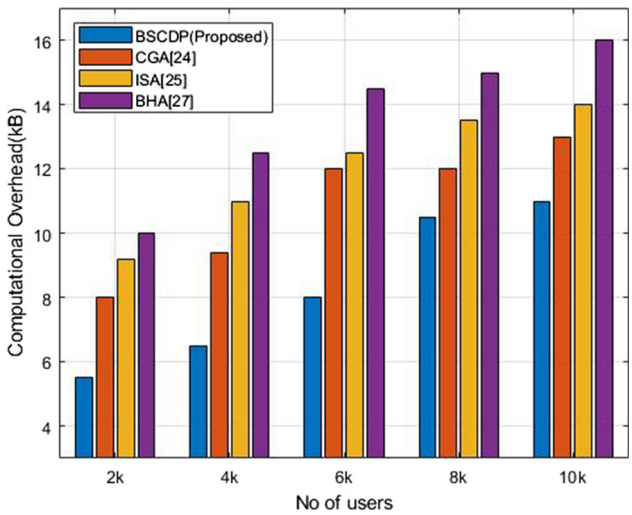

Fig. 13 Computational overhead analysis

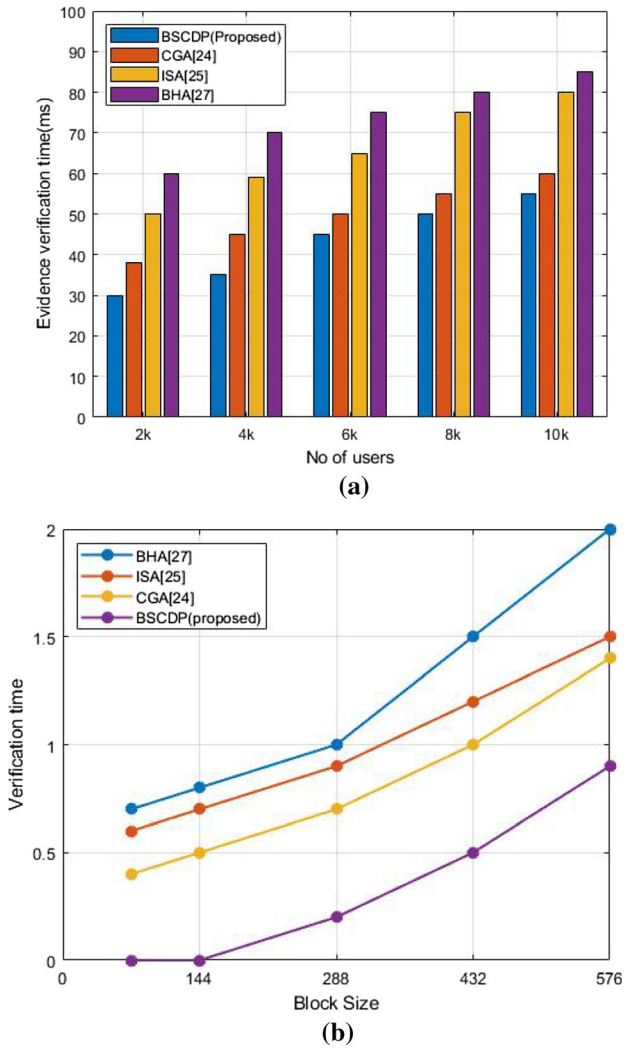

Fig. 14 Verification time analysis **a** Number of users **b** Block size

4.2.7 Throughput analysis

The period at which the valid transactions are done by blockchain, which is expressed as transactions per second (TPS).

$$T_{hgpt} = \frac{T_v}{T_t} \tag{9}$$

where T_{hgpt} denotes throughput, T_v signifies the total valid transactions, and T_t denotes total time in seconds.

When compared to existing BHA, ISA, and CGA methods, our proposed BSCDP has the highest transaction throughput. Our proposed BSCDP transaction has been linearly increased from 2 to 12 TPS when compared with the existing BHA, ISA, and CGA methods. The throughput value for our proposed BSCDP is 12tps when compared with the existing BHA (5tps), ISA (7tps), and CGA (10tps) methods. Figure 15 shows the analysis of throughput.

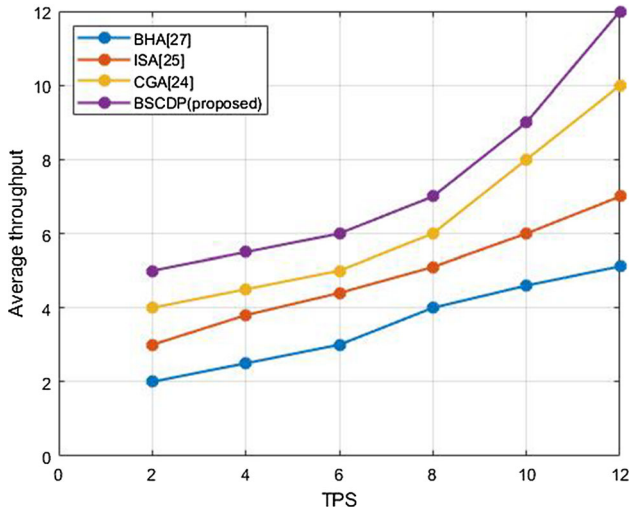

Fig. 15 Throughput analysis

Table 4 provide the time analysis and Table 5 provide comparative analysis of proposed with existing methods.

4.2.8 Complexity analysis

A key challenge with blockchain-based is the space and time complexity without efficient verification, Unauthorized users are also permitted into the system, increasing the system's vulnerability. The space complexity includes the storage and search overhead in the blockchain. The time complexity involves the time taken for getting a response from the cloud to the user. We suggest the BSCDP Architecture, which is helpful in reducing space and time complexity to achieve faster response time and improve storage in the cloud environment.

For algorithm 1, assume that total number of transaction of the basic statement is $\mathrm{f(n)} = \sum_{i=1}^{n} 3 = 3n$, thus the time complexity of the algorithm is $T = O(f(n))$=O (n). For algorithm 2, the basic statement associated with n is same as algorithm 1. So, it's time complexity is also O (n). For algorithm 3, the total number of transaction of the basic statement is $\mathrm{f(n)} = 1 + 1 = O(n^2)$. So its time complexity is O (n).

The space complexity for algorithm 1 and algorithm 2 is O (n). The hash value is stored in each block and hash value's length is fixed in algorithm 3, so, the spacecomplexity $= \mathrm{O}(\mathrm{nlog}(n))$.

4.2.9 Discussion

Our proposed system consists of authentication, privacy, security, and integrity. For authentication, we utilized PUF and fingerprint biometrics for secure data transmission. FE is used to protect privacy and strengthen security. When sharing a key, the ECCST cryptography algorithm is used

Table 4 Average time analysis

Methods	EIT (%)	EIT (%)	EDT (%)	RT (%)	EVT (%)
BHA	55.5	9.09	9.04	12	25
ISA	25	22.2	70.07	31	16.6
CGA	25.64	7.33	20	6.25	31.97
BSCDP (proposed)	23.07	2.56	8	5	10

Table 5 Comparative analysis with existing methods

Methods	TCR (%)	CO (%)	Throughput (%)
BHA	11.6	4.6	15.6
ISA	22	2.14	14.28
CGA	40	8.33	20
BSCDP (proposed)	9.2	1.59	25.83

to increase security. We introduced the blockchain and IPFS for PD collection, hash computation, and storing with reduced CO. The integrity of data is maintained by using a blockchain based on SHA-3. By arranging FSC, the DU tracks their data. FSC is used to keep track of data's history. The data collected is directly stored in IPFS and the DU gets a hash from IPFS to retrieve the data in the future. Finally, the data verification is done by the PA.

Each person's biometrics are unique. The fingerprint is the most important biometric because of its convenience, stability, and uniqueness. A fingerprint, like a password, is difficult to copy and store. However, if it is lost, the user's fingerprint information is leaked, which leads to a great loss for the users. Many researchers have tried to save the fingerprint by giving it directly to the device, which leads to the most risk. To overcome this risk, the one-way function method is used to store the fingerprint. Because a slight change in the fingerprint makes the output less secure, these methods are impractical. However, in our proposed BSCDP method, the FE extracts the auxiliary information and the secret key from the fingerprint template. PUF and biometrics are used for verification, while FE is used to protect privacy and strengthen security. Algorithms such as Rabin, Knapsack, McEliece, Elagamal, and RSA require more time to generate keys. The ECCST algorithm, which we propose as a BSCDP method, is faster and more secure. Cloud storage, such as TPA, is centralized, and the integrity is checked by requesting CSP, resulting in a high computation power. When a large amount of data is added to a blockchain network, every node has the same ledger information. As a result, the provenance of collection and storage are difficult issues to address.

These problems are solved in our proposed BSCDP architecture. The blockchain and IPFS are for PD collection, hash computation, and storing with reduced CO. The data's integrity is maintained by employing a blockchain based on the SHA-3 algorithm. By arranging FSC, DU tracks its own data. The FSC standard is used to track the history of data. The performance analysis is given in Figs. 7, 8, 9. 10, 11, and 12. The proposed BSCDP has an insertion time of 30 ms, encryption time of 50 ms, decryption time of 60 ms, encryption time of 3 s, RTA of 40 ms, total change rate of 5%, CO of 5.8 KB, and Evidence Verification time of 30 ms. The proposed BSCDP has a throughput of 5tps when compared with existing methods such as CGA, ISA, and BHA of 4tps, 3.8tps, and 2tps for TPS of 2 respectively. From the analysis, it is clear that our proposed BSCDP has a minimum insertion time, verification time, encryption, and decryption time, RTA, total change rate, and CO.

5 Conclusion

People's dependence on Cloud Computing applications and other technologies has risen because of the present COVID-19 pandemic. The pandemic problem has an impact on every industry, including tourism, healthcare, education, and others. This crisis may result in a permanent shift toward working from home. The working from home increased the amount of data gathered from a variety of sources. Working from home mainly depends on cloud computing applications that help employees to do their jobs quickly and efficiently. The increase in the usage of cloud computing applications leads to security risks. In this paper, a BSCDP using IPFS and blockchain technology is proposed for data integrity, privacy, and security. The combination of PUF and fingerprint biometrics is used for secure data transmission. To protect privacy and strengthen security, the FE is employed. For the security of data, the ECCST algorithm is utilized and in this paper, every file's key size is varied so that the intruder has difficulty in hacking the file. For every block, we create a MHT by utilizing SHA-3 to improve the integrity of the data. FCS is added to the system to track data activities, and by using FE, privacy is protected and security is strengthened. Therefore, the proposed work improves the privacy and security of data in the cloud environment. The IPFS system is used for storing PD and its hash is used for retrieving

data in the future. Finally, PA verifies the data. When comparing our proposed BSCDP method with existing methods, the proposed BSCDP method achieves high security in the Cloud Environment (CE) for 2 K users in terms of evidence insertion time 30 ms, verification time 30 ms, response time 40 ms, total change rate 5%, CO 5.8 KB, encryption 50 ms and decryption time 52 ms. The analysis revealed that the proposed BSCDP approach has a high level of security in terms of verification time and performance investigation. Scalability affects factors like block size, and block interval time. Which may lead to reducing security, for overcoming this, the future work may focus on resolving scalability issues in healthcare, finance etc.

Acknowledgements Not applicable.

Author contributions All the authors have participated in writing the manuscript and have revised the final version. All authors read and approved the final manuscript.

Funding There is no funding for this study.

Declarations

Conflict of interest Authors declares that they have no conflict of interest.

Consent to participate There is no informed consent for this study.

Consent for publication Not Applicable.

Ethical approval This article does not contain any studies with human participants and/or animals performed by any of the authors.

References

1. Varghese, B., & Buyya, R. (2018). Next generation cloud computing: New trends and research directions. *Future Generation Computer Systems, 79*, 849–861.
2. Hassan, H., El-Desouky, A. I., Ibrahim, A., El-Kenawy, E. S. M., & Arnous, R. (2020). Enhanced QoS-based model for trust assessment in cloud computing environment. *IEEE Access, 8*, 43752–43763.
3. Li, P., Li, J., Huang, Z., Gao, C. Z., Chen, W. B., & Chen, K. (2018). Privacy-preserving outsourced classification in cloud computing. *Cluster Computing, 21*(1), 277–286.
4. Aceto, G., Persico, V., & Pescapé, A. (2020). Industry 4.0 and health: Internet of things, big data, and cloud computing for healthcare 4.0. *Journal of Industrial Information Integration*, *18*, 100129.
5. Alsmadi, D., & Prybutok, V. (2018). Sharing and storage behavior via cloud computing: Security and privacy in research and practice. *Computers in Human Behavior, 85*, 218–226.
6. Razaque, A., Amsaad, F., Hariri, S., Almasri, M., Rizvi, S. S., & Frej, M. B. H. (2020). Enhanced grey risk assessment model for support of cloud service provider. *IEEE Access, 8*, 80812–80826.
7. Subramanian, N., & Jeyaraj, A. (2018). Recent security challenges in cloud computing. *Computers & Electrical Engineering, 71*, 28–42.
8. Yang, Y., Liu, X., Guo, W., Zheng, X., Dong, C., & Liu, Z. (2020). Multimedia access control with secure provenance in fog-cloud computing networks. *Multimedia Tools and Applications, 79*(15), 10701–10716.
9. Cui, H., Deng, R. H., & Li, Y. (2018). Attribute-based cloud storage with secure provenance over encrypted data. *Future Generation Computer Systems, 79*, 461–472.
10. Namasudra, S., Devi, D., Kadry, S., Sundarasekar, R., & Shanthini, A. (2020). Towards DNA based data security in the cloud computing environment. *Computer Communications, 151*, 539–547.
11. Siddiqui, M. S., Rahman, A., & Nadeem, A. (2019). Secure data provenance in IoT network using bloom filters. *Procedia Computer Science, 163*, 190–197.
12. Gireesha, O., Somu, N., Krithivasan, K., & VS, S. S. (2020). IIVIFS-WASPAS: An integrated Multi-Criteria Decision-Making perspective for cloud service provider selection. *Future Generation Computer Systems, 103*, 91–110.
13. Lnenicka, M., & Komarkova, J. (2019). Developing a government enterprise architecture framework to support the requirements of big and open linked data with the use of cloud computing. *International Journal of Information Management, 46*, 124–141.
14. Wilczyński, A., & Kołodziej, J. (2020). Modelling and simulation of security-aware task scheduling in cloud computing based on blockchain technology. *Simulation Modelling Practice and Theory, 99*, 102038.
15. Casino, F., Dasaklis, T. K., & Patsakis, C. (2019). A systematic literature review of blockchain-based applications: Current status, classification and open issues. *Telematics and Informatics, 36*, 55–81.
16. Ge, C., Ma, X., & Liu, Z. (2020). A semi-autonomous distributed blockchain-based framework for UAVs system. *Journal of Systems Architecture, 107*, 101728.
17. Abramson, W., Hall, A. J., Papadopoulos, P., Pitropakis, N., & Buchanan, W. J. (2020, September). A distributed trust framework for privacy-preserving machine learning. In *International Conference on Trust and Privacy in Digital Business* (pp. 205–220). Springer, Cham.
18. Zhu, L., Wu, Y., Gai, K., & Choo, K. K. R. (2019). Controllable and trustworthy blockchain-based cloud data management. *Future Generation Computer Systems, 91*, 527–535.
19. Muzammal, M., Qu, Q., & Nasrulin, B. (2019). Renovating blockchain with distributed databases: An open source system. *Future generation computer systems, 90*, 105–117.
20. Wang, M., & Zhang, Q. (2020). Optimized data storage algorithm of IoT based on cloud computing in distributed system. *Computer Communications, 157*, 124–131.
21. Kaur, H., Alam, M. A., Jameel, R., Mourya, A. K., & Chang, V. (2018). A proposed solution and future direction for blockchain-based heterogeneous medicare data in cloud environment. *Journal of medical systems, 42*(8), 1–11.
22. Tasnim, M. A., Al Omar, A., Rahman, M. S., & Bhuiyan, M. Z. A. (2018, December). Crab: Blockchain based criminal record management system. In *International conference on security, privacy and anonymity in computation, communication and storage* (pp. 294–303). Springer, Cham.
23. Tosh, D., Shetty, S., Liang, X., Kamhoua, C., & Njilla, L. L. (2019). Data provenance in the cloud: A blockchain-based approach. *IEEE consumer electronics magazine, 8*(4), 38–44.
24. Tahir, M., Sardaraz, M., Mehmood, Z., & Muhammad, S. (2021). CryptoGA: A cryptosystem based on genetic algorithm for cloud data security. *Cluster Computing, 24*(2), 739–752.
25. Kumari, P. S., & Kamal, A. N. B. (2020). Integrity service application model with prevention of cryptanalytic attacks. *Materials Today: Proceedings, 33*, 3877–3883.

26. Li, J., Wu, J., Jiang, G., & Srikanthan, T. (2020). Blockchain-based public auditing for big data in cloud storage. *Information Processing & Management, 57*(6), 102382.
27. Darwish, M. A., Yafi, E., Al Ghamdi, M. A., & Almasri, A. (2020). Decentralizing privacy implementation at cloud storage using blockchain-based hybrid algorithm. *Arabian Journal for Science and Engineering*, 1–10.
28. Tajammul, M., & Parveen, R. (2020). Auto encryption algorithm for uploading data on cloud storage. *International Journal of Information Technology, 12*(3), 831–837.
29. Huang, H., Zhu, P., Xiao, F., Sun, X., & Huang, Q. (2020). A blockchain-based scheme for privacy-preserving and secure sharing of medical data. *Computers & Security, 99*, 102010.
30. Bal, P. K., & Pradhan, S. K. (2020). Multi-level authentication-based secure aware data transaction on cloud using cyclic shift transposition algorithm. In *Advances in Intelligent Computing and Communication* (pp. 384–393). Springer, Singapore.
31. Ajaykumar, N., Sarvagya, M., & Parandkar, P. (2020). A novel security algorithm ECC-L for wireless sensor network. *Internet Technology Letters, 3*(3), e150.
32. Debnath, S., Chattopadhyay, A., & Dutta, S. (2017, November). Brief review on journey of secured hash algorithms. In *2017 4th International Conference on Opto-Electronics and Applied Optics (Optronix)* (pp. 1–5). IEEE.

Amrita Jyoti Ph.D. research scholar Kurukshetra University, Kurukshetra, Haryana, India. She received her B.Tech. degree in Information Technology from Kurukshetra University, India in 2003 and M.Tech. in Computer Science & Engineering from the Uttar Pradesh Technical University Lucknow, India in 2011. In 2005, she joined the Department of Computer Science & Engineering, ABES Engineering College, Uttar Pradesh, India as a lecturer and became an Associate Professor in 2015. She is the author of a Book ''Title: Data compression'' which has covered all the techniques of compression in text, audio and video. Her current research area include Software Engineering, Software Testing, Data compression, Data structure, JAVA, Cloud Computing and Blockchain. She published many research papers in National and International Journals (SCI/ SCIE/ SCOPUS) and presented many papers in various National and International Conferences.

R. K. Chauhan is the oldest founder faculty member as well as senior most professors in the department of Computer Science & Applications in Kurukshetra University, Kurukshetra. He obtained the doctor of philosophy in computer science from the Kurukshetra university. Under his supervision, fourteen scholars have been awarded Ph.D. degree on different areas of computer science and applications and three scholars are pursuing their research work. He has also guided more than fifty M.Tech. desertion. He has affended and participated in numerous professional conferences meetings and has published more than hundred research papers in national and International journals. He was awarded 6 merit certificate for best research paper in Dec 1998 by Institution of engineers(India). He held the position of chairman of Department of Computer Science and Applications from Dec 2007 to Dec 2010. His research area includes Advance Database, Data Mining & Warehousing, Mobile Computing, Ad-hoc Networks and Software Engineering. He has been the member of various academic and administrative bodies of Kurukshetra University and other University.