International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Secured User Authentication and Data Sharing for Mobile Cloud Computing Using 2C-Cubehash and PWCC

Bala Anand Muthu[1], Sivaparthipan C.B.[2], Lakshmana Kumar R.[3], Jayanthi S.[4], Rajermani Thinakaran[5]

Submitted:11/03/2024 **Revised**: 26/04/2024 **Accepted**: 03/05/2024

Abstract: To share data securely, existing works use a public and private key for transmission. The secured data sharing along with user authentication is necessary for mobile devices. So, proper authorization is needed to access data owners' credential information from cloud computing. Therefore, this study presents a novel framework for secured data sharing and user authentication. The security-based authorization in data transfer is taken significantly. Initially, the data owner and data user register with a trusted authority, and then a hashcode is generated using 2C-Cubehash for the login. After login, the data is encrypted by primitive weierstrass curve cryptography and then uploaded by the owner into the mobile cloud. Later, with an end-user request, the possessor and permitter verify the user and give the end-user a partial private key (PPK) and smart contract. Using the two PPK, the client downloads the data. The proprietor monitors the successful transaction in the blockchain by constructing a Universally Unique Identifier-based Merkle Tree. The proposed model resulted in 3123ms processing time, 2941ms latency, and 98.03% security level. With the proposed model, secured user-authenticated data sharing as process innovation can be achieved.

Keywords: *Data Owner; Data User; Mobile Cloud Computing; Partial Private Key; User Authentication.*

1. Introduction

Cloud computing in mobile devices is a mixture of service and object-based frameworks with good efficiency and competence [1], [2], [3]. Generally, there is a gateway between the data owner (DO) and data user (DU) and the cloud acts as a medium of data exchange [4], [5]. The main focus during the data sharing is the security and user authentication to access data in the mobile cloud [6]. The low security is menaced by data loss; so, it is necessary to secure the data in the cloud [7]. User authentication confirms the user to access the data, thus ensuring the protection of the information [8], [9].

Feng et al., [10] developed an anonymous authentication model for the evaluation of the mobile cloud-sourced blockchain. The Software Guard Extension was used to periodically change the key generation, and hence achieved security and resisted attacks in the blockchain. However, this model could not detect the malicious nodes during trust evaluation. Rangwani & Om [11] implemented a secured user authentication in cloud computing based on the Elliptic Curve Cryptography (ECC) algorithm. Three-factor authentications were carried out and a security check by Automated Validation of Internet Security Protocols and Applications was conducted to produce security validation. However, the storage of transaction details was not given much importance, which led to difficulty in finding corrupted data.

While Swetha & Latha [12] employed ciphertext attribute encryption for the security of mobile-based cloud computing. Here, the key generation was updated using a splitting and re-encryption algorithm to give maximum security and low processing cost. Yet, there was no authorization for the user, making the model get attacked easily during the process. Xie et al., [13] presented cloud computing by the attribute encryption based on multi-authority for mobile cloud. A Linear Secret Sharing Scheme was injected for secured access and thus maintained user authorization and reduced the computation overheads. However, it took more time to encrypt the data, which showed a lag in the model. Zhong et al., [14] introduced data migration between the user and cloud provider with the help of ECC. A key agreement was executed to make the user authorized, hence achieving privacy in data sharing. However, the model occurred at a high computational time, leading to errors in the framework.

In most existing procedures as described above, password-based user authentication has been used, which projects as a threat to the user during data sharing. Also, authorizing a full private key in the initial stage would lead to misuse of the keys. Other methods have opted for the transmission of data using cipher text, which does not guarantee safety. The data present in the cloud should be monitored to be free from smart card attacks, replay attacks, and middle-

[1, 2, 3]*Department of Computer Science & Engineering, Tagore Institute of Engineering and Technology, Attur, Tamil Nadu, India*
[1]*ORCID ID: 0000-0002-9509-6943*
[2]ORCID ID : 0000-0002-5389-4330
[4]*Department of Electrical and Electronics Engineering, Tagore Institute of Engineering and Technology, Attur, Tamil Nadu, India.*
[3] *Faculty of Data Science and Information Technology, INTI International University, Malaysia*
ORCID ID: 0000-0002-9525-8471
** Corresponding Author Email: rajermani.thina@newinti.edu.my*

man attacks. So, to overcome these issues, a novel framework of user authentication and secured data sharing by 2C-Cubehash and primitive weierstrass curve cryptography (PWCC) has been proposed.

The rest of the article is organized as: Section 2 describes the proposed framework, Section 3 evaluates the performance assessment, and Section 4 concludes the study with future work.

2. PROPOSED METHODOLOGY

The proposed framework, user authentication and secured data sharing by 2C-Cubehash and PWCC-based was illustrated in Fig. 1. While detailed explanations are as follows.

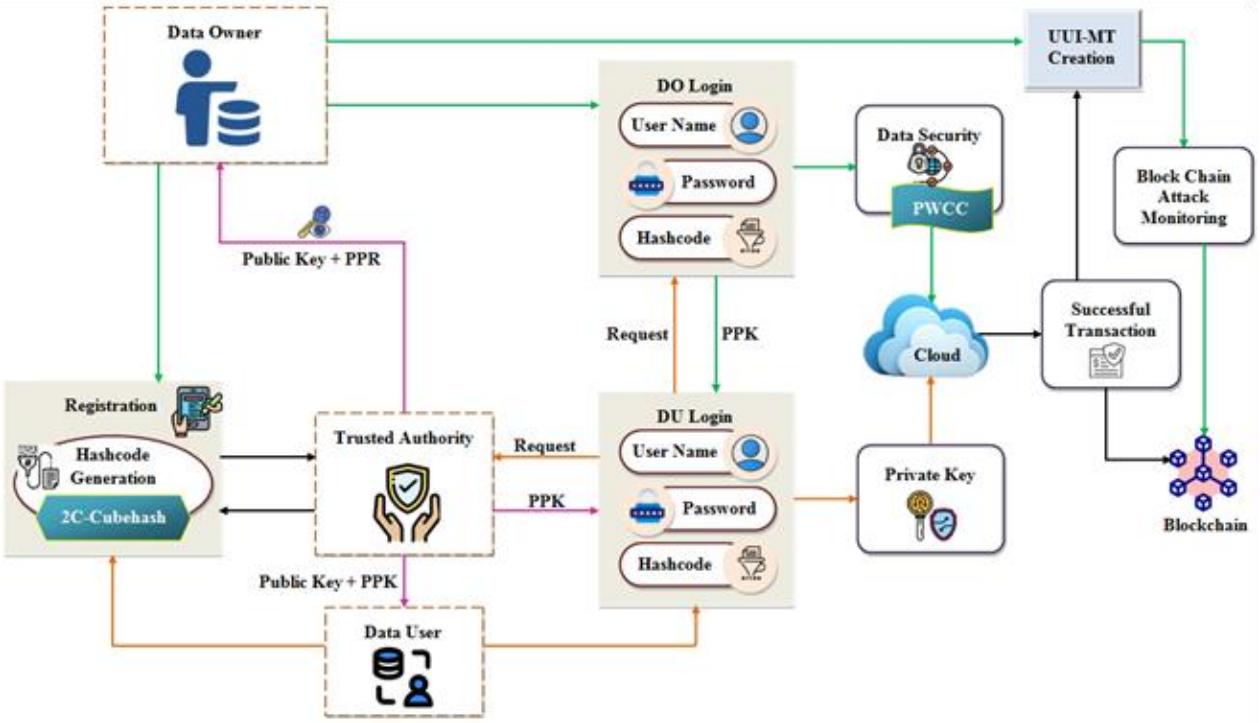

Fig. 1. Block diagram of the framework

2.1. Registration

Initially, the proposed framework starts with DO and DU registration with their respective details. Let the DO and DU details like name, address, date of birth, contact number, and age be represented as in (1) and (2).

$$K = \{K_1, K_2, K_3, .., K_o\} \quad (1)$$

$$L = \{L_1, L_2, L_3, ..., L_u\} \quad (2)$$

Where, K is the details of the o number of DO, and L is the details of the u number of DU. After registration, TA sends the public key and PPK generated by PWCC (in section 2.3) to the DO and DU.

2.2. Hashcode Generation

After the registration, the hashcode is generated by TA using 2C-Cubehash for DO and DU. Cubehash is strong in synchronization but has fixed permutation issue that leads to collision in hash code. So, 2C-Cubehash, which uses a compliment process to create a hashcode with no collision is used as follows:

2.2.1. ASCII Conversion

First, the data K and L are converted into the American Standard Code for Information Interchange (ASCII). Here, the ASCII converts characters and numbers into 8-bit code. Let the ASCII converted values K^A for DO and L^A for DU to be represented in (3) and (4).

$$K^A = \{K_1^A, K_2^A, K_3^A, .., K_o^A\} \quad (3)$$

$$L^A = \{L_1^A, L_2^A, L_3^A, ..., L_u^A\} \quad (4)$$

2.2.2. XOR operation

Now, K^A and L^A are given into Exclusively-OR (XOR), which performs the logical operation. In this operation, if the input value is the same, then the output will be zero, or else the output will be one. Let the outputs from XOR be K^X and L^X.

2.2.3. 2' Compliment-Cubehash (2C-Cubehash)

In this step, the values K^X and L^X are given as input to 2C-Cubehash to convert the binary form into hashcode. Here, at first, 2's compliment step is executed in which first, the binary numbers with value 1 are converted to 0 and vice versa. In the second step, 1 is added to each input to give the outputs. K^C and L^C.

2.2.3.1. Cubehash

At last, the outputs K^C and L^C are given to the cubehash algorithm to generate the hashcode. The cubehash has three parameters: α - the number of bytes in the input, β - the number of compression iterations, and γ - the total number of bits present in the input. The cubehash output for r rounds are represented as in (5) and (6). Where K^H and L^H are the hashcodes generated for DO and DU, respectively.

$$K^H = \left(\frac{\gamma}{\beta}\right) + r - \alpha\left(K^C\right) \tag{5}$$

$$L^H = \left(\gamma * \beta^{-1}\right) + r - \alpha\left(L^C\right) \tag{6}$$

2.3. Uploading Data

After the generation of the hashcode, to securely upload the data into the mobile cloud, the DO logs in using the username, password, and hashcode K^H. Normally, to encrypt data, ECC is used, which encrypts the data into large messages and leads to side attacks. So, to overcome this issue, PWCC, which locates every input into the curve, is used for the encryption. The following is the explanation for the PWCC process.

2.3.1. Primitive Weierstrass (PW) Equation

Let M be the area of the curve, where encryption takes place. Now, for the primitive values of (ω_1, ω_2), the PWC equation $[\varphi(M)]$ is determined as in (7).

$$\phi(M) = \left(\frac{1}{M^2}\right) + \sum_{(\omega_1,\omega_2)} \left[\frac{1}{(M-\omega_1)^2} - \frac{1}{\omega_2^2}\right] \tag{7}$$

2.3.2. Key Generation

The DO uses PWC to generate the public key (ρ) and PPK (η_1, η_2). First, a constant (η_1, η_2) with range $[1 to (q-1)]$, which represents PPK, is selected from the curve. And q is the maximum limit in the curve. Hence, the key is generated with respect to the point x from the curve as given in (8).

$$\rho = \frac{x}{(\eta_1 + \eta_2)^{-1}} \tag{8}$$

2.3.3. Data Encryption

Now, the data V from the owner is pointed as an affine point v in the PWC. Thus, two encrypted forms E_1 and E_2 are generated with respect to the public key ρ and PPK (η_1, η_2) as shown in (9) and (10). These encrypted forms E_1 and E_2 are thus uploaded to the mobile cloud.

$$E_1 = (\eta_1) \times (v) \tag{9}$$

$$E_2 = (v + \eta_2) + \left(\frac{1}{(x * \rho)^{-1}}\right) \tag{10}$$

2.4. Downloading Data

The hashcode L^H created for DU (L) along with the username and password used to log in. Now, the DU sends a request to the DO for the purpose of accessing the data. The DO (K) checks the user, accepts the request, and creates a smart contract χ_1 with respect to an agreement Θ as given in (11).

$$\chi_1 = L + \Theta + K \tag{11}$$

Now, the DO (K) sends the smart contract χ_1 and PPK η_1 to the data user as represented in (12).

$$K \xrightarrow{\chi_1 + \eta_1} L \tag{12}$$

After getting the PPK (η_1), the DU sends a request to the TA to get another PPK (η_2). TA accepts the request and creates a smart contract (χ_2) with an agreement Θ between the TA (Z) and user L as in (13).

$$\chi_2 = Z + \Theta + L \tag{13}$$

The TA sends the PPK (η_2) and the smart contract (χ_2) to the data user L as given in (14).

$$Z \xrightarrow{\chi_2, \eta_2} L \tag{14}$$

Thus, the DU gets both the PPK and uses these keys to decrypt the data as represented in (15).

$$v = \eta_2 - (x + \eta_1) \tag{15}$$

Hence, after decrypting the data, the original data V is downloaded by the DU from the mobile cloud.

2.4.1. Blockchain

Blockchains are used for the storage of transaction information and these transactions are updated in the form of blocks. Thus, the b number of successful transactions τ is given in (16).

$$\tau = [\tau_1, \tau_2, \tau_3, ..., \tau_b] \tag{16}$$

The following pseudocode is for Uploading and Downloading data with PWCC keys.

Input: Data V ***Block Chain***

Output: Successful Transaction τ

Begin

Initialize parameters V , K^H , L^H

While login

//Uploading *data*

For $[\varphi(M)]$

Generate key $\rho = \dfrac{x}{(\eta_1 + \eta_2)^{-1}}$

Encrypt and upload data $E_1 = (\eta_1) \times (v)$, E_2

End for

//Downloading *data*

Smart contract creation

For K owner

$$\chi_1 = L + \Theta + K$$

End for

For Z authority

$$\chi_2 = Z + \Theta + L$$

End for

Obtaining smart contract, partial private keys (η_1, η_2)

Decrypt and download data $v = \eta_2 - (x + \eta_1)$

End While

Successful transaction τ

End

2.5. Data Monitoring

DO handles the transaction history by constructing a Universally Unique Identifier - Merkle Tree (UUI-MT), which monitors the attack happening in the blockchain containing transaction details. In Merkle Tree (MT), two different data could end up in the same root, which causes the collision of data. So, to prevent this collision, UUI, which identifies the data in the system uniquely, is used along with MT as described in Fig. 2.

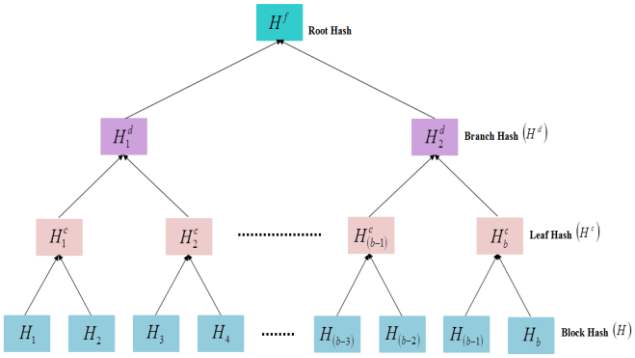

Fig. 2. The framework of UUI-MT

2.5.1. Universally Unique Identifier

First, let the UUI (J) for b successful transactions are represented as in (17).

$$J = \{J_1, J_2, J_3, .., J_b\} \tag{17}$$

Now, this identifier (J) is added to the respective transaction to uniquely identify the transaction details. The UUI-added transaction details τ^U are equated in (18).

$$\tau^U = \{[\tau_1 + J_1], [\tau_2 + J_2], [\tau_3 + J_3], ..., [\tau_b + J_b]\} \tag{18}$$

2.5.2. Merkle Tree

The hashcode H is generated in the MT for each τ^U transaction as given in (19).

$$H = (H_1, H_2, H_3, ..., H_b) \tag{19}$$

Where, H_1 is the hash generated by the Merkle tree for the $[\tau_1 + J_1]$ transaction, and so on. After creating the hash value, the MT divides the transactions into leaf hash H^c, branch hash H^d, and root hash H^f. It is derived as in (20), (21) and (22).

$$H^c = \{(H_1 + H_2), (H_3 + H_4), ..., (H_{(b-1)} + H_b)\} \tag{20}$$

$$H^d = [(H_1 + H_2 + H_3 + H_4), ..., (H_{(b-3)} + H_{(b-2)} + H_{(b-1)} + H_b)] \tag{21}$$

$$H^f = (H_1 + H_2 + H_3 + H_4 + , , , + H_{(b-1)} + H_b) \tag{22}$$

Thus, the root hash H^f created in UUI-MT is checked with the blockchain, and if the hash does not match with the blockchain hash, then an attack has happened and the owner deletes that particular data in the cloud.

3. Results and Discussion

The performance analysis of the proposed work is carried out in this session to check the stability of the model. The implementation of the proposed work is done in the PYTHON platform.

3.1. Performance Evaluation

Here, the performance of the proposed work is evaluated by comparing it with the existing works. Fig. 3 represents the comparative analysis of the proposed PWCC with existing techniques, such as ECC, Rivest Shamir Adleman (RSA) algorithm, Data Encryption Standard (DES), and ElGamal. The proposed work obtained a security level of 98.03% and an attack level of 1.97%, whereas the existing work obtained an average of 91.4125% for security level and 8.5875% for attack level. The use of Weierstrass to locate every data into the curve gave higher security and a lower attack level compared to the existing works.

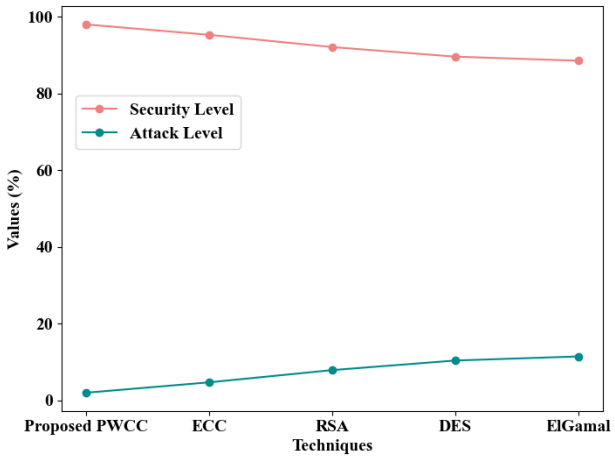

Fig. 3. Graphical Comparison of proposed PWCC

The Encryption Time, Decryption Time, and Key Generation Time of the proposed PWCC are compared with the existing techniques as given in Fig. 4. The proposed work took 1137ms to encrypt and 1273ms to decrypt the data; but the existing models took 3402.25ms and 3795.25ms to encrypt and decrypt the data. Also, the key generation time was 273ms for the proposed PWCC, whereas the existing models obtained the key in 564.75ms. This shows that PWC needed less time to encrypt, decrypt, and generate keys than other works.

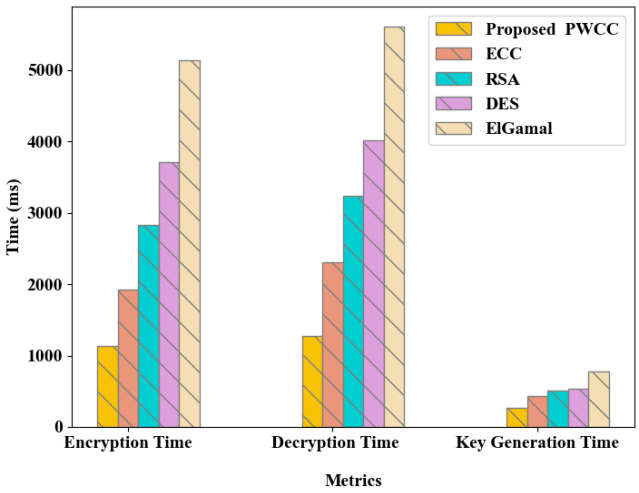

Fig. 4. Comparative Analysis

The proposed PWCC model is compared with the existing techniques as shown in Table 1 with respect to memory usage during encryption and decryption. It is noted that 137245452kb and 139456981kb of memory were used during encryption and decryption. However the existing models used higher memory than the proposed work because the input data could not end up in the existing curves. Thus, the proposed model shows better performance than the existing frameworks.

Table 1. Memory usage analysis

Method	Memory usage on Encryption(kb)	Memory usage on Decryption(kb)
Proposed PWCC	137245452	139456981
ECC	158754246	161324756
RSA	168724563	169923654
DES	171475623	171542368
ElGamal	192687451	192275698

From Fig. 5, the Hash Code Generation Time for the proposed 2C-Cubehash is compared with the existing hash generation models like Cubehash, Secure Hash Algorithm-512 (SHA512), HAVAL, and Message Digest-5 (MD5). Only 314ms was needed by the proposed work to generate the hash, whereas existing works took a mean of 675.5ms. Utilizing 2-C into the input value resulted in generating a hash value quickly and making the proposed work distinct.

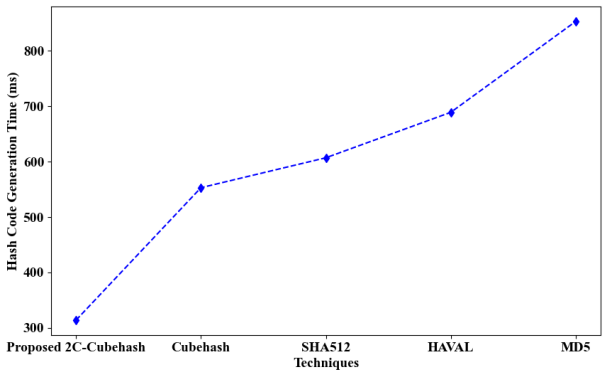

Fig. 5. Graphical Representation with respect to Hash Code Generation

In response to the comparative analysis of the proposed UUI-MT with the existing MT, Verkle Tree (VT), Radix Tree (RT), and Sparse Merkle Tree (SMT), the proposed model gave better results than other works as shown in Fig. 6(a) and 6(b). This is because uniquely identifying the transaction and creation of root hash achieved a response time of 3571ms, processing time of 3123ms, and latency of 2941ms. But, the existing works achieved an average of 5429.25ms response time, 5044.75ms processing time, and 4406.75ms latency, which is higher than the proposed framework.

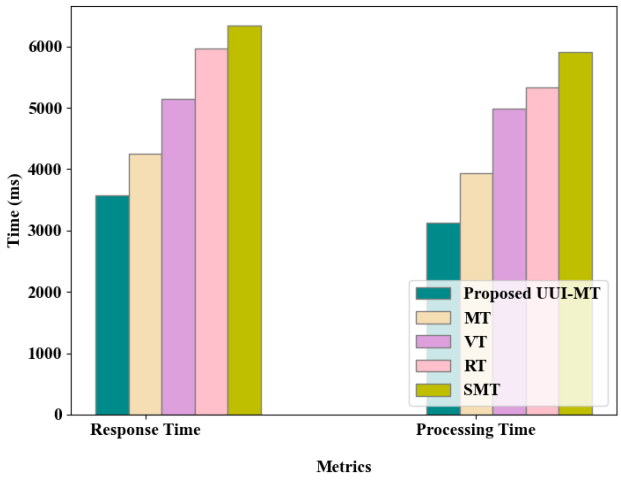

Fig. 6(a). Comparison result

Table 2 describes the comparison of the proposed model and the existing technique with respect to Waiting Time and Turn Around Time (TAT). Creation of the MT based on the hash created for unique data performed the monitoring with a low waiting time of 2356ms and low TAT of 2851ms than the current models. This proves that the data attack is checked within a short time regularly.

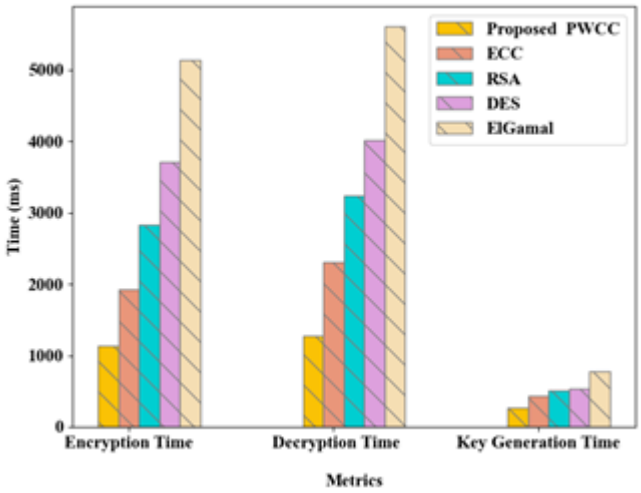

Fig. 6(b). Latency comparison

Table 2. Waiting time and Turn Around Time evaluation

Method	Waiting Time (ms)	Turn Around Time (ms)
Proposed UUI-MT	2356	2851
Merkle Tree	2942	3217
Verkle Tree	3471	3978
Radix Tree	4011	4562
Sparse Merkle Tree	4612	5153

The comparison of the proposed framework with the existing works has been given in Table 3. In the proposed work, the public key and PPK have been generated by the PWCC model, which gives a security level of 98.03%. Other works like the Seamless Secure Anonymous Authentication Scheme (SSAAS) gave a low-security level of 85% due to no updation in the private key. The data has been secured and regularly monitored by the creation of UUI in the Merkle Tree, which checks attacks regularly for 3123ms time. Whereas, models like ECC and Compromise-Resilient Anonymous Mutual Authentication Scheme (C-RAMAS) took 6848ms and 4010ms to complete the process. The proposed model obtained a latency of 2941ms, which shows that the model executed the work in a shorter time than the existing technique that obtained an average latency of 4053.68ms. Hence, the proposed model achieved effective results in secured authentication and data sharing in the mobile cloud.

Table 3. Comparative analysis of the proposed model with existing works

Study	Method	Latency (ms)	Security Level (%)	Processing Time (ms)
Proposed	PWCC	2941	98.03	3123

work				
Zhu et al., [15]	RISKCOG	3237.7	93.77	3237.7
Deebak et al.,[16]	SSAAS	4100	85	-
Ghaffar et al., [17]	ECC	-	90	6848
Olakanmi & Odeyemi [18]	C-RAMAS	3560	-	4010
Vivekanandan et al., [19]	ECC	5317	92	5120

4. Conclusion and Future Work

This work proposed a secured user authentication and data sharing with the help of 2C-Cubehash and PWCC techniques. The proposed framework undergoes several processes like partial private key generation, smart contract, unique user identification, and hashcode generation. After this execution, the proposed framework resulted in a 98.03% security level, ensuring that the proposed model securely shared the data. The proposed model also achieved low encryption of 1173ms, decryption of 1273ms, and key generation time of 273ms, which are lower than other works that make the proposed work to achieve the target more readily. Thus, it can be concluded that the proposed work authorized the user and securely transferred the data between the owner and the user.

Even though the proposed work obtained secured data encryption and decryption, sharing big data might be difficult to control. So, in the future, big data handling will be concentrated to achieve secured big data sharing.

Author contributions

Bala Anand Muthu, **Sivaparthipan C.B.**, **Lakshmana Kumar R.**, **Jayanthi S.**, and **Rajermani Thinakaran** contributed to the design and implementation of the research, to the analysis of the results and to the writing of the manuscript.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] L. Devadass, S. S. Sekaran, & Thinakaran, R. (2017). Cloud computing in healthcare. International Journal of Students' Research in Technology & Management, 5(1), 25-31.

[2] N. D. Sarier, (2021) "Multimodal biometric authentication for mobile edge computing", *Information Sciences*, *573*, pp. 82–99, https://doi.org/10.1016/j.ins.2021.05.036

[3] S. A. Chaudhry, T. Shon, F. Al-Turjman, & M. H. Alsharif, (2020) "Correcting design flaws: An improved and cloud assisted key agreement scheme in cyber physical systems", *Computer Communications*, 153, pp. 527–537, https://doi.org/10.1016/j.comcom.2020.02.025

[4] M., Mamdouh, A. I., Awad, A. A. M., Khalaf, & H. F. A. Hamed, (2021) "Authentication and Identity Management of IoHT Devices: Achievements, Challenges, and Future Directions", *Computers and Security*, 111, pp. 1–24, https://doi.org/10.1016/j.cose.2021.102491

[5] V. Krishnasamy, & S. Venkatachalam, (2021) "An efficient data flow material model based cloud authentication data security and reduce a cloud storage cost using Index-level Boundary Pattern Convergent Encryption algorithm", *Materials Today: Proceedings*, 81(2), pp. 931–936, https://doi.org/10.1016/j.matpr.2021.04.303

[6] W. Wang, H. Huang, L. Xue, Q. Li, R. Malekian, & Y. Zhang, (2021) "Blockchain-assisted handover authentication for intelligent telehealth in multi-server edge computing environment", *Journal of Systems Architecture*, 115, pp. 1–12, https://doi.org/10.1016/j.sysarc.2021.102024

[7] D. V. K. Vengala, D. Kavitha, & A. P. S. Kumar, (2023) "Three factor authentication system with modified ECC based secured data transfer: untrusted cloud environment", *Complex and Intelligent Systems*, *9*(3), pp. 2915–2928, https://doi.org/10.1007/s40747-021-00305-0

[8] T. A. Alkhdour, M. A Almaiah, A. I. Ali, A. B Lutfi, M. A Alrawad, & T. T. Tin, (2024). "Revolutionizing Healthcare: Unleashing Blockchain Brilliance Through Fuzzy Logic Authentication*"* *Journal of Theoretical And Applied Information Technology*, 29, 102(4).

[9] X. Lu, Z. Pan, & H. Xian, (2020) "An efficient and secure data sharing scheme for mobile devices in cloud computing", *Journal of Cloud Computing*, *9*(1), pp. 1–13, https://doi.org/10.1186/s13677-020-00207-5

[10] W. Feng, Z. Yan, L. T. Yang, & Q. Zheng, (2022) "Anonymous Authentication on Trust in Blockchain-Based Mobile Crowdsourcing. *IEEE Internet of Things Journal*, *9*(16), pp. 14185–14202, https://doi.org/10.1109/JIOT.2020.3018878

[11] D. Rangwani, & H. Om, (2021) "A Secure User Authentication Protocol Based on ECC for Cloud Computing Environment", *Arabian Journal for Science and Engineering*, 46(4), pp. 3865–3888, https://doi.org/10.1007/s13369-020-05276-x

[12] M. Swetha, & M. Latha, (2023) “Security on mobile cloud computing using cipher text policy and attribute based encryption scheme”, *Materials Today: Proceedings*, 80, pp. 3059–3063, https://doi.org/10.1016/j.matpr.2021.06.462

[13] M. Xie, Y. Ruan, H. Hong, & J. Shao, (2021) “A CP-ABE scheme based on multi-authority in hybrid clouds for mobile devices”, *Future Generation Computer Systems*, 121, pp. 114–122, https://doi.org/10.1016/j.future.2021.03.021

[14] H. Zhong, C. Zhang, J. Cui, Y. Xu, & L. Liu, (2022) “Authentication and Key Agreement Based on Anonymous Identity for Peer-to-Peer Cloud”, *IEEE Transactions on Cloud Computing*, 10(3), pp. 1592–1603, https://doi.org/10.1109/TCC.2020.3004334

[15] T. Zhu, Z. Qu, H. Xu, J. Zhang, Z. Shao, Y. Chen, S. Prabhakar, & J. Yang, (2020) “RiskCog: Unobtrusive real-time user authentication on mobile devices in the wild”, *IEEE Transactions on Mobile Computing*, 19(2), pp. 466–483, https://doi.org/10.1109/TMC.2019.2892440

[16] B. D. Deebak, F. Al-Turjman, & L. Mostarda, (2020) “Seamless secure anonymous authentication for cloud-based mobile edge computing”, *Computers and Electrical Engineering*, 87, pp. 1–13, https://doi.org/10.1016/j.compeleceng.2020.106782

[17] Z. Ghaffar, S. Ahmed, K. Mahmood, S. H. Islam, M. M. Hassan, & G. Fortino, (2020) “An improved authentication scheme for remote data access and sharing over cloud storage in cyber-physical-social-systems”, *IEEE Access*, *8*, pp. 47144–47160, https://doi.org/10.1109/ACCESS.2020.2977264

[18] O. O. Olakanmi, & K. Odeyemi, (2021) “Compromise-resilient anonymous mutual authentication scheme for n by m-times ubiquitous mobile cloud computing services”, *Computers and Security*, 108, pp. 1–14, https://doi.org/10.1016/j.cose.2021.102369

[19] M. Vivekanandan, V. N. Sastry, & U. Srinivasulu Reddy, (2021) “Blockchain based Privacy Preserving User Authentication Protocol for Distributed Mobile Cloud Environment”, *Peer-to-Peer Networking and Applications*, 14(3), pp. 1572–1595, https://doi.org/10.1007/s12083-020-01065-3