User Revocation-Enabled Access Control Model Using Identity-Based Signature in the Cloud Computing Environment

Tarun Kumar[1], Prabhat Kumar[1], Suyel Namasudra[2]*

[1] Department of Computer Science and Engineering, National Institute of Technology Patna, Bihar (India)
[2] Department of Computer Science and Engineering, National Institute of Technology Agartala, Tripura (India)

* Corresponding author: suyelnamasudra@gmail.com

Received 23 December 2023 | Accepted 26 April 2024 | Published 24 May 2024

Abstract

Nowadays, a lot of data is stored in the cloud for sharing purposes across various domains. The increasing number of security issues with cloud data raises confidentiality concerns about keeping these stored or shared data. Advanced encryption and decryption techniques in cloud computing environments can be considered useful to achieve this aspect. However, an unresolved yet critical challenge in cloud data-sharing systems is the revocation of malicious users. One of the common methods for revocation involves periodically updating users' private keys. This approach increases the workload of the Key Generation Center (KGC) as the number of users increases. In this work, an efficient Revocable Identity-Based Signature (RIBS) scheme is proposed, wherein the revocation functionality is delegated to an External Revocation Server (ERS). This proposed scheme allows only the non-revoked users to access the system resources, thus, providing restricted access control. Here, the ERS generates a secret time key for signature generation based on a revoked user list. In the proposed method, a user uses its private key and secret time key to sign a message. Furthermore, to maintain data confidentiality, symmetric encryption and Elliptic Curve Cryptography (ECC) based asymmetric encryption techniques are used before outsourcing data to the cloud server. The results illustrate that the proposed scheme outperforms some of the existing schemes by providing reduced computation costs.

Keywords

Cryptography, Digital Signature, Elliptic Curve Cryptography, Identity-Based Signature, Revocation.

DOI: 10.9781/ijimai.2024.05.001

I. Introduction

In recent years, cloud computing has been offering a promising opportunity to provide computing capacity and storage for various applications. Researchers have proposed data-sharing models utilizing cloud computing across different domains. These models have facilitated the development of third-party telematics services, such as remote diagnostics, energy consumption analysis, intelligent transportation systems, and entertainment, aimed at enhancing user safety and convenience. However, before implementing these services, the crucial challenge of securing shared data in the cloud must be addressed. Unfortunately, there have been many instances, where the confidentiality of cloud data is compromised or accessed unlawfully. Due to the presence of a large number of malicious user data, cloud services must maintain strong security to prevent unauthorized access to stored data [1]. Traditional cryptographic techniques, such as RSA (Rivest Shamir Adleman) and Advanced Encryption Standard (AES), are widely used to secure data in the cloud. However, these techniques can be vulnerable to attacks like brute-force attacks. This highlights the need for robust security measures that can provide high security [2].

A digital signature is an important part of computer security that helps to identify users, verify their authenticity, and ensure they cannot deny their actions. In a typical way, the certificate authorities are responsible for managing the certificates and checking if the signature public keys are tied to the certificates. But, there is a challenge in checking if a user (certificate) has been revoked or not. In [3], Boneh and Franklin proposed a solution by regularly updating the secret keys for all non-revoked users. This approach has a critical problem, i.e., the key managing authority needs to always be online, which can create security risks and the overhead at the key managing authority dramatically increases as more users join the system. With the rise of cloud computing, many organizations are using powerful cloud servers to do heavy computing tasks. In such systems, the user revocation process is outsourced to an external server to handle the task of updating secret keys for non-revoked users.

Please cite this article as:
T. Kumar, P. Kumar, S. Namasudra. User Revocation-Enabled Access Control Model Using Identity-Based Signature in the Cloud Computing Environment, International Journal of Interactive Multimedia and Artificial Intelligence, vol. 9, no. 1, pp. 127-136, 2024, http://dx.doi.org/10.9781/ijimai.2024.05.001

Data confidentiality is another important aspect of computer security that can protect data against unauthorized access or disclosure. This can be achieved by using Deoxyribonucleic Acid (DNA) computing [4] and ECC techniques. Here, DNA sequences can be used for the secret key generation process. DNA computing is capable of computing in parallel and providing massive storage, which makes it highly efficient for handling complex, unique, and large encryption keys. ECC is also known for its efficiency and strong security properties, making it suitable for protecting sensitive data. Encrypting the data using ECC with the DNA-based generated key before transmission ensures that the data is protected during communication and the encrypted data can only be decrypted by the intended recipient having the corresponding DNA-based generated key [5]. As a result, combining DNA computing and ECC can present a novel approach for data security and privacy of sensitive information.

Enabling secure data sharing and access control along with effective user revocation management can be achieved through the integration of Identity-Based Signature (IBS) systems and ECC-based encryption techniques. These systems play a crucial role in safeguarding sensitive information and ensuring controlled access within a given framework. The integration of IBS and ECC encryption adds an extra layer of security to the data encryption process. In the context of secure data sharing, IBS mechanisms support users' authenticity and their corresponding access privileges. By utilizing user-specific identity information as the public key, these systems facilitate secure communication and data transmission processes to authorized entities. While recent works have introduced many revocable schemes, the issue of revoking identity-based signatures remains largely unexplored. Periodically updating users' private keys in identity-based approaches is typically managed by the KGC, which increases the KGC's overhead as the number of users increases in the cloud server [6].

An advanced technique has been proposed in this paper to solve the problems of the existing schemes. Here, an efficient way to handle revoked users is explored by applying identity-based signatures. In this paper, the proposed Revocable Identity-Based Signature (RIBS) scheme delegates the revocation functionality to an external revocation server. The ERS generates a secret time key that is used in the signature generation process of non-revoked users. However, if the ERS handles all the tasks of updating keys, there can be security concerns as the cloud servers are not completely trusted. So, a user's signing process is split into two parts: one uses a private key connected to their identity (given by the KGC) and the other uses a short-term secret time key connected not only to their identity, but also to the current time (given by the ERS that updates it regularly). The ERS cannot forge a digital signature because it does not have the complete signing key. To revoke a user, the KGC just informs the ERS not to issue new short-term keys. Moreover, before uploading data to the cloud server, the data owner encrypts the data using encryption techniques. In the data access phase, the data user receives the encrypted data along with a digital signature, which ensures the origin of the data. The user also receives the public key corresponding to the secret time key that is used in the signature verification process before the data encryption process. The key contributions of the proposed scheme can be summarized below.

- The proposed scheme can manage revoked users by using identity-based signatures, thus, providing access to only non-revoked or genuine users of the system.
- In this scheme, all the revocation and key generation functionalities are delegated to an external revocation server. Thus, the overhead of the central system, i.e., the cloud server, is reduced.
- Here, the DO uses a secret time key to sign the data before uploading it to the cloud server, and the corresponding public key is used to verify the signature at the user's end. This provides access control to the system as the secret time key is provided only to the authorized entities of the system.
- Additionally, the user's signing key is composed of two key components, namely the private key generated by the KGC and the short-term secret time key generated by the ERS. Therefore, neither KGC nor ERS can forge the signature as they do not have the complete signing key.

The subsequent sections of this paper are organized as follows. The Related Work section presents a comprehensive overview of existing research works. Following this, the Problem Statement section discusses the system model and key definitions of the proposed work. The methodology of the proposed work is given in detail in Section IV. Subsequently, the Performance Analysis section discusses the outcomes derived from the proposed work. Finally, the Conclusion and Future Work section summarizes the main findings of this work and outlines future research directions.

II. Related Work

The related work in data sharing, access control, and user revocation management encompasses various approaches, including identity-based signature systems, DNA encryption, and their integration for enhanced security.

Liu et al. [7] introduced a two-factor access control framework that incorporates user secret keys and security devices for accessing web-based cloud services. While this framework focuses on access control, it supports multiple layers of authentication in securing data-sharing environments. Yang et al. [8] proposed a smart card framework for multimedia cloud usage, employing a Role-Based Access Control (RBAC) model for authentication and authorization. This work provides insights into the role of smart card technologies in securing data access. Jia et al. [9] proposed a method to outsource the user revocation process to the cloud server. Instead of an immediate revocation approach, this scheme uses a periodic time-key update approach for revocation. This scheme minimizes the communication and computation costs for the key generation process. Bai et al. [10] suggested a smart card authentication framework by using ECC, wherein a single smart card serves for accessing multiple applications. By incorporating the revocation technique of [11], Tsai et al. [12] introduced the first revocable IBS scheme. In this scheme, the authors partitioned the private key of each non revoked user into two separate keys. Here, the authors verified the security strength of this scheme using standard security models. Building upon this foundation, Hung et al. [13] proposed an enhanced RIBS scheme with increased security. Sun et al. [14] later introduced an efficient RIBS scheme without pairing operations, although no formal security proof was provided by this scheme. Wei et al. [15] suggested a forward-secure RIBS scheme using the Complete Subtree (CS) method, where the KGC maintains a binary tree with each node representing a user. However, in the aforementioned RIBS schemes [12]-[15], the KGC not only issues the initial identity key for each registered user, but also periodically renews the time update keys for non-revoked users. This approach faces two challenges: (i) maintaining the KGC online poses security risks, and (ii) when the number of system users grows, the computational and communication overheads at the KGC increase rapidly. Consequently, the KGC can evolve into a security and performance bottleneck for the entire cryptosystem. Another Revocable Certificateless Public Key Encryption (RCL-PKE) scheme proposed by Ma et al. [16] addresses critical issues of the existing RCL-PKC systems. The framework of this system consists of three main entities: Private Key Generator (PKG), Cloud Revocation Agents (CRAs), and users. The PKG generates system

TABLE I. Summary of Existing Schemes

Scheme	Advantage	Disadvantage
Liu et al. [7]	This scheme provides multi-factor access control and authentication for data-sharing environments.	It cannot prevent the revoked users from accessing the system.
Tsai et al. [12]	It provides access control that solves user revocation problems.	The communication overhead of the KGC increases as the KGC is kept online always.
Hung et al. [13]	Here, a revocable access control model is proposed with improved security strength.	Both computation and communication costs increase as the number of users present in the system increases.
Yang and Lin [27]	This scheme provides confidentiality and key management by using the RSA algorithm.	The long key size used in RSA reduces the efficiency of the system in terms of storage and communication costs.
Qin et al. [28]	It provides a certificateless signing scheme and solves the key escrow issue. Here, the computation at the user end is reduced.	Pairing-based operations of this scheme result in high computation overhead. Additionally, it does not address the user revocation issue.
Liao et al. [30]	This scheme solves the issues of digital certificate management and key escrow using ECC-based operations.	It cannot revoke malicious users from the system.

parameters and distributes a secret master time key to each CRA. The PKG issues partial identity keys to users, while each CRA updates users' time update key shares based on the revocation list. Users then generate their public keys using a secret value and system parameters. By outsourcing revocation functionality to the CRA, it reduces the overhead at the PKG and introduces a dependency on the CRAs. Moreover, the use of multiple CRAs increases complexity in management and raises concerns about the overall security and reliability of the system. Yang et al. [17] introduced another secure and efficient ID-based signature scheme for the Internet of Things (IoT) environment. Though this scheme outperforms many other existing RIBS schemes in terms of computational performance, it incurs comparatively more computational cost as it is based on pairing-based operations. These pairing-based operations are very expensive computationally as compared to ECC-based scaler-point multiplication operations. Both the schemes of [16] and [17] utilize pairing-based operations for internal mathematical foundations.

In recent years, numerous schemes have been proposed by employing DNA computing to address security-related concerns, such as authentication, access control, encryption, and decryption. For instance, Adleman [18] introduced a DNA computing technique to efficiently solve the traveling salesman problem, showing the efficacy of DNA computing in solving this problem more efficiently than traditional methods. Sohal and Sharma [19] suggested a symmetric key cryptographic technique, namely Binary DNA (BDNA), that uses the concept of DNA sequences. BDNA employs a combination of DNA sequence, substitution cipher, and XOR operations to generate a secure and robust encryption key. Murugan and Thilagavathy [20] proposed a secure cloud storage model based on DNA computing and Morse code. Their approach involves storing encrypted data in a zigzag pattern to enhance data security. Addressing these limitations remains a significant challenge in the advancement of secure DNA-based computing applications [21], [22]. Apart from the above-mentioned schemes, many other schemes [23]-[30] present in the literature discuss efficient access control mechanisms for cloud computing environments. Yet, there is a need for further research to explore the potential of access control techniques for other security-related aspects in cloud computing environments [31]-[35]. Table I summarizes a few existing schemes along with their advantages and disadvantages.

III. Problem Statement

This section mainly represents the system model, design goals, and system definition of the proposed scheme in detail.

A. System Model

This work involves five different participants or entities: Key Generation Center, External Revocation Server, Data Owner (DO), Data User (DU), and Cloud Service Provider (CSP).

The key generation center plays a crucial role in setting up the system, verifying the identity of participants, and issuing key pairs. Upon an entity's request to join the system, the KGC issues the required key pair components. The external revocation server issues and updates the users' time update keys according to the Revoked User List (RUL). If a user is in the RUL, then, the ERS does not issue the secret time key for the user. A cloud service provider functions as a semi-trusted entity with significant storage and computation capabilities. It securely stores the encrypted data collected from DOs and manages access requests from data users. Data owners, including various application service providers, share data through the cloud. To ensure data confidentiality, these providers encrypt the shared data with a signature before uploading it to the cloud. The provider encrypts the data by using a DNA-based symmetric encryption algorithm as given in [36]. The provider generates a DNA-secret key and encrypts the requested data using the DNA-based symmetric encryption process. Data users can query the shared ciphertext from the cloud service provider. The workflow of the proposed model is shown in Fig. 1.

B. Design Goals

The main design goals of this work are presented below:

- **Message Confidentiality**: Data providers encrypt the shared data with a signature to ensure data confidentiality. If the user does not have a valid secret time key, which is updated periodically for each non-revoked user, then, the user cannot access the data.
- **Access Control**: By updating the secret time key and signing key of each non-revoked user, the proposed scheme restricts the revoked users from accessing the system resources.
- **Reduced Workload**: The proposed model outsources the key update process to an external server to reduce the workload of the key generation center.

C. System Definitions

The proposed scheme comprises nine algorithms, namely system initialization, registration, time key update, signature generation, data encryption, data storage phase, data access phase, signature verification, and data decryption, which are described below. The KGC maintains a RUL that contains the identities of revoked users and the RUL is updated periodically. All the symbols used in this work are described in Table II, as well as in the text.

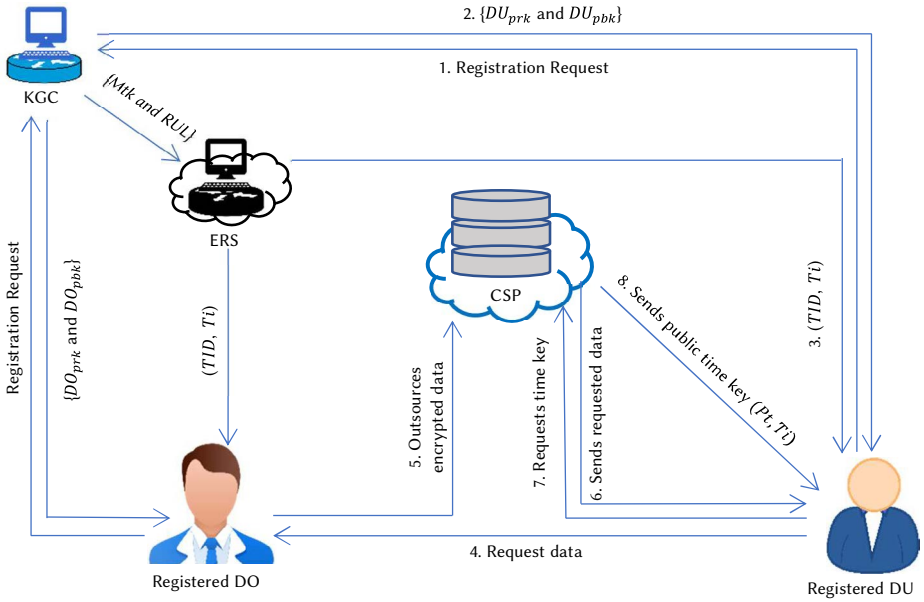

Fig. 1. System model of the proposed scheme, including all the entities.

TABLE II. Symbols and Their Description

Description	Symbol	Description	Symbol
KGC's Master key	KGC_{msk}	DU's private key	DU_{prk}
Secret time key at time *Ti*	(*TID, Ti*)	DO's private key	DO_{prk}
Hash operation	*Hash*()	DO's public key	DO_{pbk}
KGC's private key	KGC_{prk}	DU's public key	DU_{pbk}
KGC's public key	KGC_{pbk}	Signature	(m, s_1, s_2)
XOR operation	+	Hash operation	*h*()
Login Details	*LD*	DNA secret key	D_{SK}
DU's identity	DU_{ID}	DNA-encrypted ciphertext (CT)	CT_{SYMM}
DO's identity	DO_{ID}	Data file	*PT*
Master time key	*mtk*	Revoked and non-revoked User List	*RUL, nRUL*
Encrypted key	E_{Key}	Hash of CT	H_{CT}

System_initialization {$q \rightarrow (C(q), PP, KGC_{msk}, mtk, KGC_{prk}, KGC_{pbk})$}: The KGC performs this *System_initialization* algorithm to set up the system initially by choosing a prime number q as the security parameter. The algorithm then generates all the necessary Public Parameters (*PP*), including an elliptic curve $C(q)$ for the system, and the initial keys, such as KGC_{msk}, *mtk*, KGC_{prk} and KGC_{pbk}.

Registration {(*ID, PP*)→(*prk, pbk*)}: When an entity sends a registration request, the KGC performs this *Registration* algorithm by taking entity's Identity (*ID*) and *PP* as inputs and generates private key (*prk*) and public key (*pbk*) of the entity.

Time_Key_Update:{(*PP, mtk, ID, Ti*) → (*SID, Ti*)}: The ERS executes this algorithm to update the users' secret time key and to generate user's signing key (*SID, Ti*).

Encryption {$(D_{SK}, PT) \rightarrow CT_{SYMM}$}: The DO uses the *Encryption* algorithm to encrypt the plaintext data (*PT*) file and generates CT_{SYMM} by using D_{SK}.

Sign_Generation {$(PP, CT_{SYMM}, DO_{prk}, (TID, Ti)) \rightarrow (m, s_1, s_2)$}: The DO generates the signature (m, s_1, s_2) by using CT_{SYMM}, DO_{prk}, and (*TID, Ti*) as inputs.

Data_Storage {$(DU_{pbk}, D_{SK}) \rightarrow (E_{Key}, H_{CT})$}: The ECC encryption is used by the *DO* to encrypt D_{SK} using users' DU_{pbk}. This generates the encrypted secret key E_{Key} and H_{CT}.

Data_Access {$(REQ, DO_{ID}, DU_{ID}) \rightarrow Enc_{pt}$}: The CSP retrieves (*Pt, Ti*) from the request *REQ*. It encrypts (*Pt, Ti*) by using the users' DU_{pbk} and sends Enc_{pt} to the user.

Signature_Verification {$((m, s_1, s_2), DO_{pub}, PP) \rightarrow (Accept/Reject)$}: The user verifies the signature by using the Signature_Verification algorithm. The user takes signature (m, s_1, s_2), DO's public key (DO_{pub}), and *PP* as inputs and verifies the signature.

Decryption {$(D_{SK}, CT_{SYMM}) \rightarrow PT$}: The DU performs the *Decryption*() by using D_{SK} to decrypt CT_{SYMM} and generates *PT*.

IV. Proposed Methodology

In the beginning, the KGC initializes the system by selecting all the necessary parameters of the proposed scheme. The registration process starts with the registration requests received by the KGC, which registers all the entities, including DOs and DUs. The KGC generates login credentials, private-public key pair, and other necessary elements for all registered users. Once they are registered, the entities can request the DO to access data of their interest. If the DU is authorized, the KGC sends the requested data file to the DU along with an identity-based signature. The entire process is divided into many phases which are discussed below in detail.

A. Phase 1: System Initialization

During this phase, the central entity KGC sets up the system initially by choosing all the necessary parameters. The KGC generates its own master secret key, master time key, and private-public key pair. These keys are required to generate the necessary keys for DOs and DUs, when they register themselves. The CSP keeps its private key KGC_{msk} and *mtk* securely to maintain the system's security. The KGC sends *mtk* to the ERS. Here, *mtk* is used to generate a secret time key for each non-revoked user. This phase focuses on the initial setup and key management aspects of the cloud security system. The steps of the system setup phase executed by the KGC are as follows:

Step 1: The *KGC* selects a security parameter *q* and generates all public parameters like the elliptic curve and the generator point *P*.

Step 2: In the second step, the master key KGC_{msk} and master time key *mtk* are randomly selected by the KGC.

Step 3: Then, the *KGC* selects the private key KGC_{prk} and computes the corresponding public key $KGC_{pbk} = KGC_{prk} \cdot P$.

Step 4: Finally, the *KGC* sends *mtk* to the *ERS*.

B. Phase 2: Registration

To initiate data transmission, any entity must first register with the KGC. This involves sending a registration request to the KGC containing the entity's identity (for example, D_{ID}, i.e., *ID* of DU). Then, KGC creates a user profile and sends login details and public-private key pairs as the registration request reply. Once registered, the DU can request data from the DO and a secret time key from the ERS. By following the same procedure, a DO can also register itself with the KGC. The steps of the registration phase (for example, DU registration phase) are mentioned below.

To join the system, an entity sends a registration request to the *KGC* that includes the entity's identity (e.g., D_{ID}, representing the identity of *DU*). The KGC, in turn, creates a user profile ($DU_{ID} = h(D_{ID} + KGC_{msk})$) and inserts DU_{ID} into *nRUL* list. KGC sends public-private key pairs as a reply to the registration request. The same registration procedure applies, when the DO registers itself with the KGC.

Step 1: In the first step, the entity selects a random nonce *r*.

Step 2: Here, the entity computes $DU_{prk} = DU_{ID} + r + KGC_{msk}$ and $DU_{pbk} = DU_{prk} * P$ for the data user *DU*.

Step 3: Then, it sends DU_{ID} and (DU_{prk}, DU_{pbk}) key pair to the *DU*.

Step 4: At last, the *KGC* sends the updated *RUL* and *nRUL* to the *CSP*.

By following the same procedure, the *CSP* also registers the data owners and provides DO_{prk} and DO_{pbk} key pair.

C. Phase 3: Time Key Update

Once registered, the *DU* can request data from the *DO*, as well as the secret time key from the *ERS*. Each entity (i.e., data user) needs to request for secret time keys from the ERS before starting data access. Here, to upload or store data to the cloud server, the *DO* needs to obtain its secret time key. The *DO* uses this key to encrypt the symmetric encryption key. The *DU* uses this key to verify the signature and to decrypt the encrypted symmetric key. The time key update phase as shown in Algorithm 2 provides access control to the system by providing the secret time key only to the authorized entities of the system.

Upon receiving a user's time key update request, the ERS first authenticates whether the user is a legal user by verifying its DU_{ID}, and extracts the DU's real identity D_{ID}. Before issuing the keys, the ERS again initiates the verification process by checking if the user is present in the RUL. If the user is found in the *RUL*, the ERS denies the time key request. If the user is not listed in the *RUL*, the ERS executes the algorithm and creates a new secret time key (*TID*, *Ti*) for the user. This operation involves some input parameters, namely public parameters (*PP*), the master time key (*mtk*), the user's identity (DU_{ID}), and the current time (*Ti*)). The user's signing key (*SID*, *Ti*) is composed of two components: *SID*, $Ti = (DU_{prk}$ and *TID*, *Ti*). Upon receiving an update request from a user DU_{ID} at the time *Ti*, the *ERS* executes the algorithm as follows for the user:

Step 1: The ERS verifies DU_{ID} by checking if $DU_{ID} == h(D_{ID} + KGC_{msk})$.

Step 2: Then, it verifies if the user is a valid user by checking $DU_{ID} \in RUL$.

Step 3: In the third step, the ERS computes $h(DU_{ID})$.

Step 4: Here, the ERS computes $(TID, Ti) = mtk \cdot h(DU_{ID}) \cdot Ti$

Algorithm 1: Registration

Input: D_{ID}

Output: DU_{ID}, *RUL*, private key, and public key

1. Start
2. If $D_{ID} \neq$ Registered entity
3. Compute $DU_{ID} = h(D_{ID} + KGC_{msk})$
4. Insert DU_{ID} into *nRUL*
5. Else
6. Decline request
7. If $DU_{ID} \in nRUL$
8. Select *r*
9. Compute $DU_{prk} = DU_{ID} + r + KGC_{msk}$
10. Compute $DU_{pbk} = DU_{prk} * P$
11. Else
12. Invalid user
13. Send DU_{ID} private-public key pair to the user
14. Sends updated *nRUL* to the CSP
15. Stop

Algorithm 2: Generate time key

Input: DU_{ID}, *PP*, *mtk*, *Ti*

Output: (*SID*, *Ti*)

1. Start
2. Verify DU_{ID}
3. Compute $DU_{ID} = h(D_{ID} + KGC_{msk})$
4. If $DU_{ID} == DU_{ID}$
5. Check if $DU_{ID} \in RUL$
6. Compute $h(DU_{ID})$
7. Compute $(TID, Ti) = mtk \cdot h(DU_{ID}) \cdot Ti$
8. Compute $Pt = (TID, Ti) * P$
9. Generate $(SID, Ti) = \{DU_{prk}, (TID, Ti)\}$
10. Sends (*SID*, *Ti*) and *Pt* to the user
11. Else
12. Invalid user
14. Stop

Algorithm 3: Search time key from the CSP_{List}

Input: DO_{ID}, DU_{ID}

Output: Enc_{pt}

1. Start
2. Verify DU_{ID}
3. If $DU_{ID} \in$ Authorized user's list
4. Goto step 7
5. Else
6. Stop
7. for $i = 1$ to U
8. If $DO_{ID} \in$ Authorized user's list
9. Search DO within CSP_{List}
10. Retrieve DO's (Pt, Ti)
11. Computes $Enc_{pt} = Enc_{ECC}(DU_{pbk}, (Pt, Ti))$
12. Sends Enc_{pt} to the DU
11. Else
12. Invalid user
13. End for
14. Stop

Step 5: In the fifth step, the public key (Pt) corresponding to the secret time key (TID, Ti) is computed as $Pt = (TID, Ti) * P$.

Step 6: In the sixth step, the ERS generates the user's signing key as $(SID, Ti) = (DU_{prk}$ and $(TID, Ti))$.

Step 7: Finally, the ERS sends(SID, Ti) and Pt to the user.

D. Phase 4: Signature Generation

To sign a message CT_{SYMM}, the signer (i.e., DO) runs this algorithm with the inputs PP, CT_{SYMM}, DO_{prk}, and $\{TID, Ti\}$, and outputs the signature (m, s_1, s_2). At first, to sign a message CT_{SYMM}, the DO computes the hash of CT_{SYMM} and selects an integer k from the interval $(1, r - 1)$. The further steps of the signature generation phase (for example, DO's signature generation for message CT_{SYMM}) are as follows:

Step 1: In the first step, the DO generates a random number k.

Step 2: Here, the DO computes the hash of $CT_{SYMM} = h(CT_{SYMM})$.

Step 3: In this step, the DO computes $m = (k \times P)_x \bmod r$, where $(k \times P)_x$ is the x coordinate of the point $k \times P$.

Step 4: Here, $s_1 = k^{-1}(h(CT_{SYMM}) + DO_{prk} * m) \bmod r$ is computed.

Step 5: In the last step, the DO computes s_2 as

$s_2 = k^{-1}(h(CT_{SYMM}) + (TID, Ti) * m) \bmod r$, where (TID, Ti) is the DO's secret time key.

Step 6: The combined signature is (m, s_1, s_2).

E. Phase 5: Data Encryption

Here, the requested data file is encrypted by the DO by using the symmetric encryption algorithm used in [36]. The DO generates a DNA secret key D_{SK}. It encrypts the requested data PT using a symmetric encryption process and the D_{SK} to generate the ciphertext CT_{SYMM}. Then, it sends CT_{SYMM} and D_{SK} for further process, i.e., Data Storage Phase.

F. Phase 6: Data Storage

In this phase, the DO encrypts D_{SK} and generates encrypted key E_{Key} using the ECC encryption algorithm before outsourcing the complete encrypted data $\{CT_{SYMM}, E_{Key}, (m, s_1, s_2)\}\}$ to the CSP for sharing. To encrypt a message, e.g., D_{SK}, the DO selects a secret nonce n and encrypts it by using DU_{pbk}. It also includes the signature (m, s_1, s_2) corresponding to the CT_{SYMM}, which provides the authenticity and integrity of the ciphertext. The detailed steps are shown below:

Step 1: The DO generates a random nonce n.

Step 2: In the second step, the DO computes $E_{Key} = Encpt_{ECC}(DU_{pbk}, D_{SK}) = [(n \cdot P), (D_{SK} + n \cdot DU_{pbk} + Pt, Ti)]$.

Step 3: Here, the computes $H_{CT} = h(CT_{SYMM}, E_{Key})$.

Step 4: In the fourth step, CT_{SYMM}, E_{Key}, and H_{CT} are combined.

Step 5: Finally, the DO uploads $\{CT_{SYMM}, E_{Key}, H_{CT}, (m, s_1, s_2)\}$ to the CSP

G. Phase 7: Data Access

During the data access phase, the DU tries to access the received data $\{CT_{SYMM}, E_{Key}, H_{CT}, (m, s_1, s_2)\}$ from the CSP. However, to perform the signature verification and data decryption, the DU requires the DO's public key component related to its secret time key, which is embedded into s_2 component of the signature. Therefore, the DU can perform the signature verification and data decryption processes only after getting (Pt, Ti) from the CSP. When a DU sends a data access request to the CSP, the CSP verifies the DU's authenticity before providing the DO's public key component related to its secret time key (Pt, Ti) to the user. Here, the CSP searches for the DO_{ID} in the CSP's authorized user list. Then, the CSP searches the (Pt, Ti) in the CSP's list (CSP_{List}) and provides the (Pt, Ti) in the encrypted form to the requested DU. Otherwise, the CSP shows an invalid user. The detailed steps are shown below:

Step 1: At first, the DU sends a request $\{REQ, DO_{ID}, DU_{ID}\}$ to the CSP.

Step 2: The CSP verifies DU_{ID}'s authenticity.

Step 3: Here, (Pt, Ti) is retrieved by the CSP using Algorithm 3.

Step 4: In this step, the CSP computes $Enc_{pt} = Enc_{ECC}(DU_{pbk}, (Pt, Ti))$.

Step 5: The CSP sends Enc_{pt} to the DU

Step 6: Then, the DU computes $(Pt, Ti) = (Enc_{pt})$.

Step 5: Finally, the DU sends (Pt, Ti) for the sign verification process.

H. Phase 8: Signature Verification

To authenticate a signature (m, s_1, s_2) associated with the message CT_{SYMM} corresponding to the DO's identity DO_{ID} and time period Ti, the verifying entity, i.e., the DU executes this algorithm. It produces an outcome of "Accept" or "Reject" based on the validity of the provided signature. This algorithm takes signature (m, s_1, s_2), DO's public key (DO_{pub}), and PP as inputs to generate output as "Accept" or "Reject". The steps to verify a signature (m, s_1, s_2) for a message $\{CT_{SYMM}\}$ are as follows:

Step 1: In the first step, the DU computes the hash of $CT_{SYMM} = h(CT_{SYMM}) = e$.

Step 2: Then, m is checked by the DU regarding its validity to be an x-coordinate on the curve.

Step 3: In the third step, the DU computes $w = s_1^{-1} \bmod r$

Step 4: Then, the DU computes $u1 = (e \times w) \bmod r$ and $u2 = (r \times w) \bmod r$.

Step 5: In the fifth step, the DU computes $(x1, y1) = u1 \times P + u2 \times DO_{pbk}$.

Step 6: Then, it computes $(x2, y2) = u1 \times G + u2(Pt, Ti)$.

TABLE IV. NUMBER OF CRYPTOGRAPHIC OPERATIONS USED IN THE PROPOSED SCHEME

Schemes	Initial Key Extraction	Time Key Update	Sign Generation	Sign Verification
Jia et al. [9]	$T_{sm} + T_{hash}$	$T_{sm} + T_{hash}$	$2T_{sm}$	$3T_{bl} + 2T_e$
Tsai et al. [12]	$3T_{sm}$	$3T_{sm}$	$4T_{sm}$	$4T_{bl}$
Hung et al. [13]	$3T_{sm}$	$3T_{sm}$	$5T_{sm}$	$4T_{bl} + T_e$
Proposed	T_{sm}	$T_{sm} + T_{hash}$	$2T_{sm} + T_{hash}$	$2T_{sm} + T_{hash}$

Step 7: In the final step, the DU verifies that $m \equiv x1 \ mod \ r$ and $m \equiv x2 \ mod \ r$. If the conditions hold, the signature is valid, and the algorithm outputs "Accept".

I. Phase 9: Data Decryption

Here, the user verifies the signature using Phase 8 (Signature Verification). If the received signature matches the derived signature, the decryption process starts. At first, the user decrypts the E_{Key} by using his/her own private key. Then, the user decrypts CT_{SYMM} using the corresponding decryption process mentioned in [36]. Here, the user performs the Decryption Algorithm by using the symmetric key D_{SK} generated from the E_{Key}. The data decryption process of the proposed scheme has three steps which are represented below:

Step 1: The user first computes $DCpt\ (E_{Key}, DU_{prk})$.

Step 2: In the second step, D_{SK} is computed as $D_{SK} = Dcpt\ (Encpt_{ECC}\ (Du_{pbk}, D_{SK})) = (D_{SK} + n.Du_{pbk} + Pt, Ti) - (n.P.Du_{pbk} + Pt, Ti) = D_{SK}$

Step 3: Finally, the DU computes $Dcpt\ (Encpt_{ECC}.\ (D_{SK}, CT_{SYMM})) = PT$

V. PERFORMANCE ANALYSIS

This section validates the performance of the proposed scheme by giving the experimental environment and results and discussion.

A. Experimental Environment

The performance evaluation of the proposed scheme is conducted by using a cloud simulation environment developed using CloudSim 3.0.3. The experiments are executed on an HP Pro 200 G4 PC with the following specifications: Intel Core i5-10210U processor, Windows 10 Operating System, 8 GB RAM, and 2 TB HDD. Furthermore, the simulation tool, i.e., CloudSim, is configured with Apache Commons Math 3.6.1, and the HP Pro 200 G4 PC is equipped with Java version 8 for seamless compatibility and execution. Here, fifty data centers are considered to develop a cloud computing environment, which is heterogeneous. In this heterogeneous cloud environment, there are 5000 physical nodes, 4 GB memory capacity, 1 GB/s bandwidth, and four types (1. 3000 MIPS, 3 GB, 2. 4000 MIPS, 4 GB, 3. 5000 MIPS, 5 GB, and 4. 6000 MIPS, 6 GB) of VMs are considered.

B. Results and Discussion

In this subsection, a performance evaluation of the proposed scheme is presented by mainly focusing on computation cost. The proposed scheme is compared to some RIBS schemes proposed by Jia et al. [9], Tsai et al. [12], and Hung et al. [13]. The comparison among schemes is performed by considering computation costs for initial key extraction (registration and key generation in the case of the proposed scheme), time key update, signature generation, and signature verification.

Apart from these, a few other parameters like execution time for registration, time-key generation, and searching algorithms are also considered to analyze the performance of the proposed scheme. A number of experiments have been carried out to evaluate the performance of the proposed approach in comparison to other existing techniques. The experiments involve calculating the execution time for registration and searching operations for different numbers of users, ranging from 1 to 70 users as shown in Figures 3-5. Each experiment is repeated 20 times in different scenarios like for varying numbers of users and the average value is calculated to obtain accurate results. The experiments are repeated 20 times to check the stability of the results across repetitions. The performance metrics showed consistent behavior during these repetitions.

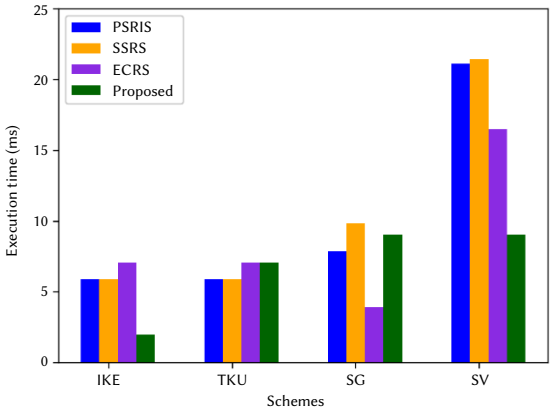

Fig. 2. Execution time (ms) in Initial Key Extraction (IKE), Time Key Update (TKU), Sign Generation (SG), and Sign Verification (SV).

To analyze the computational cost, a few operations, namely T_{sm}, T_{hash}, T_e, and T_{bl} as described in Table III are considered. As shown in Table IV, for the computation cost in the initial key extraction, both Tsai et al. [12] and Hung et al. [13] schemes require $3T_{sm}$(5.91ms), while Jia et al. [9] protocol requires $T_{sm} + T_{hash}$ (7.071ms). The proposed protocol requires T_{sm} (1.970ms) in this process. Regarding the time key update, Jia et al. [9] scheme requires $T_{sm} + T_{hash}$ (7.071ms), the schemes of Tsai et al. [12] and Hung et al. [13] need $3T_{sm}$ (5.91ms), while the proposed protocol requires $T_{sm} + T_{hash}$ (7.071ms). In the signing process, the scheme of Tsai et al. [28] requires $4T_{sm}$ (7.88ms), the scheme of Hung et al. [13] needs $5T_{sm}$ (9.85ms), and Jia et al. [9] scheme requires only $2T_{sm}$ (3.94ms). However, the proposed protocol requires 2 T_{sm} + T_{hash} (9.041 ms), which is a little more than the schemes of Tsai et al. [28] and Jia et al. [9]. For the verification process, although Jia et al. [6] scheme involves evaluating three bilinear maps for each signature, some of them can be pre-computed. Therefore, Jia et al. [9] scheme requires $3T_{bl}$+ $2T_e$ (16.472ms), while Tsai et al. [12] scheme requires $4T_{bl}$(21.1ms), and Hung et al. [13] scheme requires $4T_{bl} + T_e$ (21.431ms). Here, the proposed protocol requires $2T_{sm} + T_{hash}$ (9.041 ms) to verify the signature.

TABLE III. EXECUTION TIME OF CRYPTOGRAPHIC OPERATIONS USED IN THE PROPOSED SCHEME

Operation Notation	Description	Time Cost (ms)
T_{sm}	Time cost of scalar multiplication	1.970
T_{hash}	Time cost of hash function	5.101
T_e	Time cost of exponential operation	0.331
T_{bl}	Time cost of bilinear pairing operation	5.270

The execution time of all phases is presented in graphical form in Fig. 2. In this figure, the schemes of Tsai et al. [12], Hung et al. [13], and Jia et al. [9] are represented by the names Provably Secure Revocable ID-Based Signature (PSRIS), Strongly Secure Revocable System (SSRS), and Efficient Cloud Revocation Server (ECRS), respectively. Here, the proposed scheme (in Fig. 2) takes comparatively slightly more time in the sign generation and verification processes. It is crucial to note that the proposed scheme generates two separate signatures, which are combined to get the actual signature. Significantly, the proposed scheme minimizes the impact on overall performance.

The execution time for registration refers to the total time required to complete the registration algorithm. Fig. 3 illustrates that the proposed approach is more efficient in terms of registration time, when compared to the existing schemes, specifically PSRIS, SSRS, and ECRS. The proposed scheme employs only one point multiplication in the entire registration and key generation processes, resulting in a less time-consuming process than the existing schemes. In contrast, PSRIS and SSRS use three point multiplication operations, while ECRS uses one hash operation and one point multiplication in the registration process. Overall, the proposed scheme supports low data encryption time compared to the existing works.

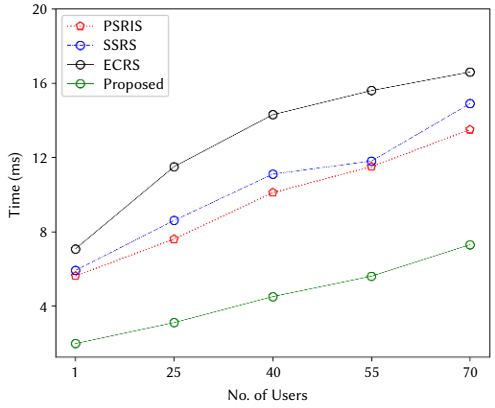

Fig. 3. Execution time for the registration process.

Fig. 4 shows the time to generate a time key for the user. In this phase, PSRIS and SSRS schemes take almost the same time to generate the time key. The proposed scheme and ECRS also take almost the same amount of time in the time key update phase, which is more than that of PSRIS and SSRS. This is because the proposed scheme uses the hash of the user's identity, i.e., DU_{ID}, to generate the time key that results in a higher execution time.

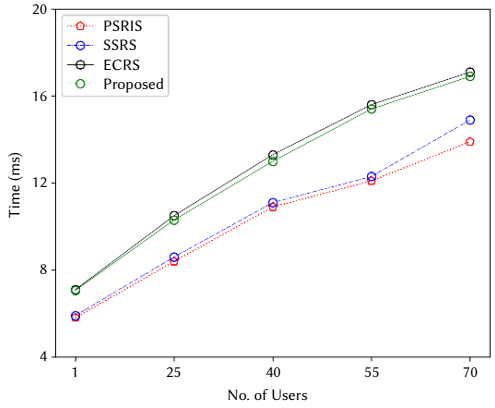

Fig. 4. Time-key generation time.

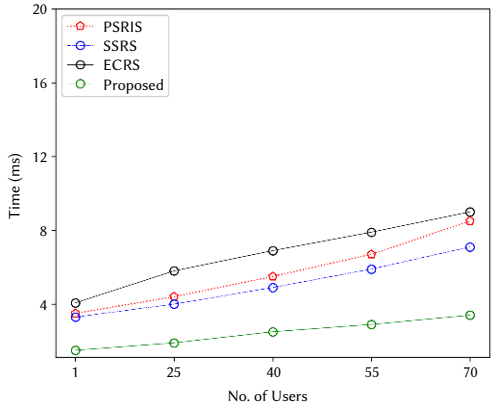

Fig. 5. Key searching time.

The next experiment is performed to compare the time for searching the secret time key (Pt, Ti) from the *CSP*. From Fig. 5, it can be seen that the proposed scheme has better performance than the other existing schemes, namely PSRIS, SSRS, and *ECRS*. In the proposed scheme, the *CSP* searches for the specified *DO*'s public key related to its time key (Pt, Ti) in the CSP_{List}, whenever the *CSP* receives an access request from the DU. In the CSP_{List}, (Pt, Ti) is stored along with the *DO*'s identity, i.e., DO_{ID}. Therefore, the *CSP* searches for DO's (Pt, Ti) by using his/her DO_{ID} that reduces search time in the proposed scheme. However, as the number of users with access requests increases in the system, the search time also increases, resulting in a linearly increasing graph in Fig. 5. In the existing schemes, the keys are stored and accessed by using a different approach, which takes more time as compared to the proposed scheme to search the keys. Thus, the proposed scheme facilitates faster access times, establishing its practicality for real-world applications.

VI. Conclusions and Future Work

This work has addressed critical aspects of secure data sharing, access control, and user revocation management by incorporating ECC-based techniques alongside identity-based signature systems. This scheme also uses DNA-based cryptography for symmetric key generation and encryption. The proposed scheme introduced a novel approach by delegating revocation functionality to an external revocation server, which generates a short-time secret time key that is used in the signature generation process of non-revoked users. Here, a user's signing key is generated by using two secret components: a long-term private key and a short-term secret time. This allows only authorized users to access the system resources. Additionally, the key generation center can efficiently revoke a user by just instructing the ERS not to issue a new short-term key. Moreover, in the proposed scheme, before uploading data to the cloud server, the data owner encrypts the data using ECC and symmetric encryption techniques. The experimental results show that the proposed scheme outperforms some existing schemes. Currently, this work can provide efficient access control by limiting access to only non-revoked system users. An extension of this work can be done by providing a mechanism to authenticate system entities before initiating data transmission.

References

[1] F. J. Abdullayeva, "Internet of things-based healthcare system on patient demographic data in Health 4.0," *CAAI Transactions on Intelligence Technology*, vol. 7, no. 4, pp. 644–657, 2022.

[2] L. D. Sharma, J. Rahul, A. Aggarwal, and V. K. Bohat, "An improved cardiac arrhythmia classification using stationary wavelet transform decomposed short duration QRS segment and Bi-LSTM network," *Multidimensional Systems and Signal Processing*, vol. 34, pp. 503-520, 2023.

DOI: https://doi.org/10.1007/s11045-023-00875-x.
[3] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in J. Kilian, (eds) *Advances in Cryptology-CRYPTO 2001. CRYPTO 2001. Lecture Notes in Computer Science*, vol. 2139. Springer, 2001. DOI: https://doi.org/10.1007/3-540-44647-8_13
[4] Y. Wang, Q. Han, G. Cui, and J. Sun, "Hiding messages based on DNA sequence and recombinant DNA technique," *IEEE Transactions on Nanotechnology*, vol. 18, pp. 299-307, 2019.
[5] L. D. Sharma, V. K. Bohat, M. Habib, A. M. Al-Zoubi, H. Faris, and I. Aljarah, "Evolutionary inspired approach for mental stress detection using EEG signal," *Expert Systems with Applications*, vol. 197, 2022. DOI: https://doi.org/10.1016/j.eswa.2022.116634
[6] Q. Qian, Y. Jia, and R. Zhang, "A lightweight RFID security protocol based on elliptic curve cryptography," *International Journal Network Security*, vol. 18, no. 2, pp. 354-361, 2016.
[7] J. K. Liu, M. H. Au, X. Huang, R. Lu and J. Li, "Fine-grained two-factor access control for web-based cloud computing services," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 3, pp. 484-497, 2016.
[8] T. C. Yang, N. W. Lo, H. T. Liaw, and W. C. Wu, "A secure smart card authentication and authorization framework using in multimedia cloud," *Multimedia Tools and Applications*, vol. 76, pp. 11715-11737, 2017.
[9] X. Jia, D. He, S. Zeadally, and L. Li, "Efficient revocable ID-based signature with cloud revocation server," *IEEE Access*, vol. 5, pp. 2945-2954, 2017.
[10] T. D. P. Bai, K. M. Raj, and S. A. Rabara, "Elliptic curve cryptography-based security framework for internet of things (IoT) enabled smart card," in *Proceedings of the World Congress on Computing and Communication Technologies* (WCCCT), IEEE, Tiruchirappalli, India, 2017.
[11] Y. M. Tseng and T. T. Tsai, "Efficient revocable id-based encryption with a public channel," *The Computer Journal*, vol. 55, no. 4, pp. 475-486, 2012.
[12] T. T. Tsai, Y. M. Tseng, and T. Y. Wu, "Provably secure revocable ID-based signature in the standard model," *Security and Communication Networks*, vol. 6, no. 10, pp. 1250-1260, 2013.
[13] Y. H. Hung, T. T. Tsai, Y. M. Tseng, and S. S. Huang, "Strongly secure revocable id-based signature without random oracles," *Information Technology and Control*, vol. 43, no. 3, pp. 264-276, 2014.
[14] Y. Sun, F. Zhang, L. Shen, and R. Deng, "Revocable identity-based signature without pairing," in *Proceedings of the* 5th *International Conference on Intelligent Networking and Collaborative Systems*, IEEE, Xi'an, China, 2013, pp. 363-365.
[15] J. Wei, W. Liu, and X. Hu, "Forward-secure identity-based signature with efficient revocation," *International Journal of Computer Mathematics*, vol. 94, no. 7, pp. 1390-1411, 2017.
[16] M. Ma, G. Shi, X. Shi, M. Su, and F. Li, "Revocable certificateless public key encryption with outsourced semi-trusted cloud revocation agent," *IEEE Access*, vol. 8, pp. 148157-148168, 2020.
[17] X. Yang, J. Wang, T. Ma, C. Chen, and C. Wang, "A secure and efficient ID-based signature scheme with revocation for IOT deployment," *in Proceedings of the Sixth International Conference on Advanced Cloud and Big Data (CBD)*, IEEE, Lanzhou, China, 2018, pp. 202-207.
[18] L. M. Adleman, "Molecular computation of solutions to combinatorial problems," *Science*, vol. 266, no. 5187, pp. 1021-1024, 1994.
[19] M. Sohal and S. Sharma, "BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 1, pp. 1417-1425, 2022.
[20] A. Murugan and R. Thilagavathy, "Cloud storage security scheme using DNA computing with morse code and zigzag pattern," *in Proceedings of the IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, IEEE, Chennai, India, 2018, pp. 2263-2268.
[21] B. Wang, Y. Xie, S. Zhou, C. Zhou, and X. Zheng, "Reversible data hiding based on DNA computing," *Computational Intelligence and Neuroscience*, vol. 2017, 2017. DOI: http://dx.doi.org/10.1155/2017/7276084
[22] S. Namasudra and P. Roy, "Size based access control model in cloud computing," in *Proceedings of the International Conference on Electrical, Electronics, Signals, Communication and Optimization*, IEEE, Visakhapatnam, India, 2015, pp. 1-4.
[23] A. T. Ehis, "Optimization of security information and event management (SIEM) infrastructures, and events correlation/regression analysis for optimal cyber security posture," *Archives of Advanced Engineering Science*, 2023. DOI: https://doi.org/10.47852/bonviewAAES32021068
[24] V. S. Gaur, V. Sharma, and J. McAllister, "Abusive adversarial agents and attack strategies in cyber-physical systems," *CAAI Transactions on Intelligence Technology*, vol. 8, no. 1, pp. 149-165, 2023.
[25] B. M. Ahmad, S. M. Ahmed, and D. E. Sylvanus, "Enhancing phishing awareness strategy through embedded learning tools: A simulation approach," *Archives of Advanced Engineering Science*, 2023. DOI: https://doi.org/10.47852/bonviewAAES32021392
[26] D. Zhang, M. Shafiq, L. Wang, G. Srivastava, and S. Yin, "Privacy-preserving remote sensing images recognition based on limited visual cryptography," *CAAI Transactions on Intelligence Technology*, vol. 8, no. 4, pp. 1166-1177, 2023.
[27] J. H. Yang and P. Y. Lin, "A mobile payment mechanism with anonymity for cloud computing," *Journal of Systems and Software*, vol. 116, pp. 69-74, 2016.
[28] Z. Qin, J. Sun, A. Wahaballa, W. Zheng, H. Xiong, and Z. Qin, "A secure and privacy-preserving mobile wallet with outsourced verification in cloud computing," *Computer Standards & Interfaces*, vol. 54, pp. 55-60, 2017.
[29] A. J. L. Rivero, M. E. Beato, C. M. Martínez, and P. G. C. Vázquez, "Empirical analysis of ethical principles applied to different AI uses cases," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 7, no. 7, pp. 105-104, 2022.
[30] Y. Liao, Y. He, F. Li, and S. Zhou, "Analysis of a mobile payment protocol with outsourced verification in cloud server and the improvement," *Computer Standards & Interfaces*, vol. 56, pp. 101-106, 2018.
[31] S. Das, S. Namasudra, S. Deb, P. M. Ger, and R. G. Crespo, "Securing IoT-based smart healthcare systems by using advanced lightweight privacy-preserving authentication scheme," *IEEE Internet of Things Journal*, vol. 10, no. 21, pp. 18486-18494, 2023.
[32] S. Khasim and S. S. Basha, "An improved fast and secure CAMEL based authenticated key in smart health care system," *Cloud Computing and Data Science*, vol. 3, no. 2, pp. 77-91, 2022.
[33] M. An, Q. Fan, H. Yu, B. An, N. Wu, H. Zhao, X. Wan, J. Li, R. Wang, J. Zhen, Q. Zou, and B. Zhao, "Blockchain technology research and application: A literature review and future trends," *Journal of Data Science and Intelligent Systems*, 2023. DOI: https://doi.org/10.47852/bonviewJDSIS32021403
[34] G. Zhang, X. Chen, L. Zhang, B. Feng, X. Guo, J. Liang, and Y. Zhang, "STAIBT: Blockchain and CP-ABE empowered secure and trusted agricultural IoT blockchain terminal," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 7, no. 5, pp. 66-75, 2022.
[35] G. Thakur, P. Kumar, C. M. Chen, A. V. Vasilakos, Anchna, and S. Prajapat, "A robust privacy-preserving ECC-based three-factor authentication scheme for metaverse environment," *Computer Communications*, vol. 211, pp. 271-285, 2023.
[36] T. Kumar, S. Namasudra, and P. Kumar, "Providing data security using DNA computing in the cloud computing environment," *International Journal of Web and Grid Services*, vol. 19, no. 4, pp. 463-486, 2023.

Tarun Kumar

Tarun Kumar is a research scholar in the Department of Computer Science and Engineering at the National Institute of Technology Patna, Bihar, India. He is also an Assistant Professor in the School of Computing Science and Engineering, Galgotias University, Greater Noida, India. He has 16 years of experience in academics. His research interests are cloud computing and DNA computing. He has published several papers in peer reviewed journals and international conferences. He also organized and attended several workshops.

Prabhat Kumar

Prabhat Kumar is a Professor in the Computer Science and Engineering Department at the National Institute of Technology Patna, India. He is a senior member of IEEE, a Professional member of ACM, a life member of CSI, IAENG, ISTE, and a global member of the internet society. He has more than 130 publications in reputed international journals, conference proceedings, and book chapters. He has supervised six PhD scholars, 25 M. Tech., scholars, and has an Indian patent on his name. His research area includes wireless sensor networks, internet of things, data analytics, software engineering, e-governance, and many more.

Suyel Namasudra

Suyel Namasudra has received Ph.D. degree from the National Institute of Technology Silchar, Assam, India. He was a post-doctorate fellow at the International University of La Rioja (UNIR), Spain. Currently, Dr. Namasudra is working as an assistant professor in the Department of Computer Science and Engineering at the National Institute of Technology Agartala, Tripura, India. Before joining the National Institute of Technology Agartala, Dr. Namasudra was an assistant professor in the Department of Computer Science and Engineering at the National Institute of Technology Patna, Bihar, India. His research interests include blockchain technology, cloud computing, IoT, AI, and DNA computing. Dr. Namasudra has edited 6 books, 5 patents, and 80 publications in conference proceedings, book chapters, and refereed journals like IEEE TCE, IEEE TII, IEEE T-ITS, IEEE TSC, IEEE TCSS, IEEE TCBB, ACM TOMM, ACM TOSN, ACM TALLIP, FGCS, CAEE, and many more. He is the Editor-in-Chief of the Cloud Computing and Data Science (ISSN: 2737-4092 (online)) journal. Dr. Namasudra has served as a Lead Guest Editor/Guest Editor in many reputed journals like IEEE TBD (IEEE, IF: 7.2), ACM TOMM (ACM, IF: 3.144), MONET (Springer, IF: 3.426), CAEE (Elsevier, IF: 3.818), CAIS (Springer, IF: 4.927), CMC (Tech Science Press, IF: 3.772), Sensors (MDPI, IF: 3.576), and many more. He has also participated in many international conferences as an organizer and session chair. Dr. Namasudra is a member of IEEE, ACM, and IEI. He has been featured in the list of the top 2% scientists in the world in 2021, 2022, and 2023. His h-index is 36.