Authentication and Key Agreement Based on Anonymous Identity for Peer-to-Peer Cloud

Hong Zhong, Chuanwang Zhang, Jie Cui, Yan Xu, Lu Liu

***Abstract*—Cross-cloud data migration is one of the prevailing challenges faced by mobile users, which is an essential process when users change their mobile phones to a different provider. However, due to the insufficient local storage and computational capabilities of the smart phones, it is often very difficult for users to backup all data from the original cloud servers to their mobile phones in order to further upload the downloaded data to the new cloud provider. To solve this problem, we propose an efficient data migration model between cloud providers and construct a mutual authentication and key agreement scheme based on elliptic curve certificate-free cryptography for peer-to-peer cloud. The proposed scheme helps to develop trust between different cloud providers and lays a foundation for the realization of cross-cloud data migration. Mathematical verification and security correctness of our scheme is evaluated against notable existing schemes of data migration, which demonstrate that our proposed scheme exhibits a better performance than other state-of-the-art scheme in terms of the achieved reduction in both the computational and communication cost.**

***Index Terms*—Cloud computing, data migration, elliptic curve, authentication, key agreement.**

I. Introduction

WITH the rapid development of the smart phone and mobile terminal industries, smart phones have become indispensable for people. China housed an estimation of 847 million mobile Internet users in December 2018, with 99.1 percent of them using mobile phones to surf the Internet [1]. Due to the weak storage and processing capabilities of the mobile terminals, smart phone users often prefer to store large-scale data files (video and audio files and streaming media files) in the cloud server. This has accelerated research of various perspectives in the cloud computing paradigm [2], [3]. Smartphone manufacturers are increasingly launching and deploying their own cloud computing services to provide users with convenient data storage services [4], [5].

People are now increasingly relying on hand-held devices such as smart phones, tablet etc., in an unprecedented number. It is worthy of note that one individual may own and use multiple smart devices. It is also common for people to recycle their smart devices quite frequently, given the fact that new arrivals characterize more attractive inherent features from a variety of manufacturers.

H. Zhong, CW. Zhang ,J. Cui and Y. Xu are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230039, China, the Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, Hefei 230039, China, and the Institute of Physical Science and Information Technology, Anhui University, Hefei 230039, China (e-mail:cuijie@mail.ustc.edu.cn).

L. Liu is with the School of Informatics, University of Leicester, LE1 7RH, UK (email: l.liu@leicester.ac.uk).

When people opt to use a new smart device from a different manufacturer, the data stored in the cloud server of the previous smart device provider should be transferred to the cloud server of the new smart device provider. One of the common ways of accomplishing this transfer is to log onto the original cloud server, download the data onto the smart terminal devices, log onto the new cloud server, and finally upload the data to the new server. As shown in Fig. 1, this process is very inefficient and tedious.

To this end, it is essential to develop a more efficient and secure way of data transfer from one cloud server to another. An ideal data migration model that can transfer user data directly between cloud servers is shown in Fig. 2. Such a model often imposes compatibility issues, since different cloud service providers characterize diverse user functions, mutual distrust and security risks in the process of data transmission, which make this ideal data migration model difficult to implement.

A few researches have attempted to overcome such data migration issues in the recent past. For example, in 2011, Dana Petcu [6] argued that the biggest challenge in cloud computing is the interoperability between clouds, and proposed a new approach for cloud portability. Binz et al. [7] proposed a cloud motion framework that supports the migration of composite applications into or between clouds. In 2012, Shirazi et al. [8] designed a scheme to support data portability between cloud databases.

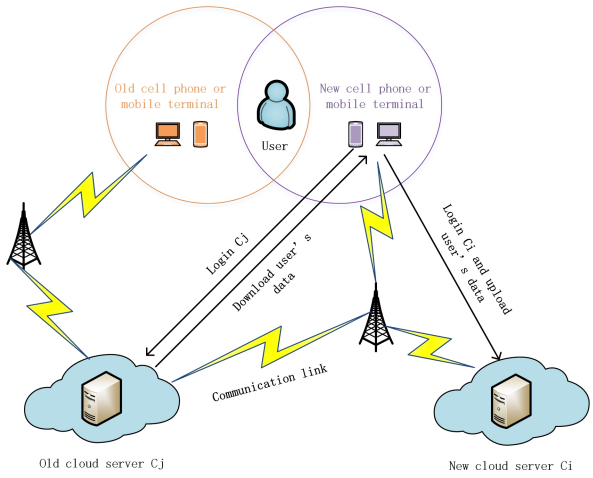


Fig. 1. Original data migration model

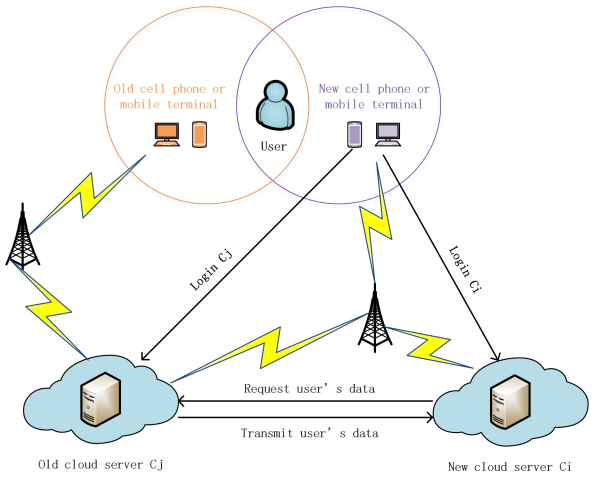


Fig. 2. Ideal data migration model

A. Our Motivations

First, we realized that the study of data migration across cloud platforms has very important practical significance. The data migration issues between clouds has many unresolved potential problems. Existing efforts in the context of cloud data migration has obvious pitfalls that restrains their efficiencies. This is to say, further research into the context of cloud data migration is an important and timely necessity, especially to facility quicker and ease data transfer between the cloud servers after users change their smartphones. Secondly, in reality, trustworthiness among multi-clouds cannot be easily achieved, particularly applications involving sensitive data transfers characterize more security constraints. For instance, achieving mutual authentication, building communication key securely and protecting the data transfer from potential attacks are some concerns to mention. Herein, authentication and key agreement mechanism can be an effective way to solve these problems. With this in mind, this paper proposes a novel authentication and key agreement scheme based on anonymous identity for peer-to-peer cloud, ultimately to facilitate easy and secure data transfer between multi-clouds.

B. Our Contributions

To our knowledge, this is the first authentication and key agreement scheme for peer cloud servers. Important contributions of this paper include the following.

- We propose a peer-to-peer cloud authentication and key agreement (PCAKA) scheme based on anonymous identity to solve the problem of trust between cloud servers. Based on the elliptic curve certificate-free cryptography, our scheme can establish secure session keys between cloud service providers to ensure session security.
- The novelty of our scheme lies in the fact that it eliminates the need for trusted authority (TA) and simplifies operations while maintaining security. In our scheme, the cloud servers enable the data owners in need of the data migration services to act as trusted third authority, so that they can verify each other and establish trusted session keys after each of the involved users performs some computation independently.
- Our scheme uses server anonymity to protect the privacy of service providers and users. It is worthy of note that both the two cloud servers involved in the migration process use anonymous identities for mutual authentication and key agreement. This strategy not only protects the identity privacy of the cloud service providers, but also makes it impossible for the involved cloud service providers to gain unnecessary information such as the brand of the old and new mobile phones belonging to the users respectively. Thus, our methodology maintains the privacy of the users by not revealing his/her personal choice.
- Our scheme provides identity traceability to trace malicious cloud servers. If the cloud service providers exhibit any errors or illegal operations in the service process, users can trace back to the real identity of the corresponding cloud server based on the anonymous identity.

C. Organization of the Rest of the Paper

In Section II, we introduce a few works related to data migration and key agreement. In Section III, the basic prerequisites and the system model of our scheme are introduced. Then, we describe our proposed PCAKA scheme in detail in Section IV. In Section V and VI, we prove the security correctness and analyze the performance of the proposed scheme respectively. Section VII concludes this paper along with outlining our future research directions.

II. Related Work

In order to realize data sharing in the cloud, a few schemes have used proxy re-encryption techniques [9]–[13]. For example, Liang and Cao [9] proposed a property-based proxy re-encryption scheme to enable users to achieve authorization in access control environments. However, Liang and Au [10] pointed out that this scheme does not have Adaptive security and CCA security features. Sun et al. [12] introduced a new proxy broadcast repeat encryption (PBRE) scheme and proved its security against selective ciphertext attack (CCA) in a random oracle model under the decision n-BDHE hypothesis. Ge and Liu [13] proposed a broadcast agent encryption (RIB-BPRE) security concept based on revocable identity to solve the key revocation problem. In this RIB-BPRE scheme, the agent can undo a set of delegates specified by the principal from the re-encryption key. They also pointed out that the identity-based broadcast agent re-encryption (RIB-BPRE) schemes do not take advantage of cloud computing, thus causes inconvenience to cloud users.

Liu et al. [14] proposed a secure multi-owner data sharing scheme for dynamic groups in the cloud. Based on group signature and dynamic broadcast encryption technology, any cloud user can share their data anonymously with others. Yuan et al. [15] proposed a cloud user data integrity check scheme based on polynomial authentication tag and agent tag update technology, which supports multi-user modification to resist collusive attack and other features. Ali et al. [16] proposed

a secure data sharing cloud (SeDaSC) method using a single encryption key to encrypt files. This scheme provides data confidentiality and integrity, forward and backward access control, data sharing and other functions. Li et al. [17] proposed a new attribute-based data sharing scheme to assist mobile users with limited resources based on cloud computing.

Authentication and key agreement is a method that enables both parties to secretly calculate the session key on a public channel, which have been widely studies [18]–[31]. As early as 1993, Maurer [18] proposed that only a difference in the received signals helps achieving perfect cryptographic security, regardless of the enemy's computing power. But they have not considered the advantage of legitimate communicants. suffices for achieving perfect cryptographic security, regardless of the enemy's computing power. Lu and Lin [19] proposed a medical key negotiation scheme based on patient symptom matching. However, He et al. [32] pointed out that Lu's scheme does not provide an identity tracking and resistance modification function and further proposed a cross-domain handshake scheme applicable to medical mobile social network and developed an android app for experimental analysis. Later, Liu and Ma [20] found that He et al.'s scheme does not resist replay attack.

Tsia and Lo [21] proposed an efficient distributed mobile cloud computing service authentication scheme with multiple functions such as user anonymity. Irshad and Sher [23] improved the protocol of Tsia [21] to make the scheme suitable for practical deployment in different wireless mobile access networks. However, Jia and He [33] pointed out that Tsia et al.'s scheme does not offer resistance to impersonation attacks and man-in-the-middle attacks. Moreover, Irshad et al.'s scheme does not support perfect forward privacy. Amor and Abid [24] proposed a mutual authentication scheme for fog users and fog servers under the condition of user anonymity. Mahmood et al. [26] proposed an anonymous key negotiation protocol for smart grid infrastructure that enables smart meters to connect anonymously to utilities. But Wang and Wu [31] pointed out that Amor et al.'s protocol cannot resist stolen verifier attacks and Mahmood et al.'s protocol cannot resist man-in-the-middle attacks and impersonation attacks.

III. Preliminaries

A. Elliptic Curve

An elliptic curve $E(F_p)$ relies on a finite field F_p, where p is a large prime number. F_p can be defines as: $y^2 = x^3+\lambda x+\eta$ mod p, where $\lambda, \eta \in F_p$ and $\Delta = 4\lambda^3 + 27\eta^2 \neq 0$ mod p. We consider the point of infinity as $\mathcal{O}$. $\mathcal{O}$ and all the points in the $E(F_p)$ form the additive cyclic group G_q.

B. Difficulty Problem

The following difficulty problem remains unsolved for any probabilistic polynomial time adversary. We will use them later in the security certification process.

Elliptic Curve Computable Discrete Logarithm (ECCDL) problem: Let $\Omega \in G_q$. P is a generator of G_q. The essence is to figure out $\omega \in Z_q^*$, which is unknown and to satisfy that the condition $\omega \cdot P = \Omega$.

Elliptic Curve Computable Diffie-Hellman (ECCDH) problem: Let P is a generator of G_q and $\alpha \cdot P, \beta \cdot P \in G_q$. The essence is to figure out $(\alpha \cdot \beta) \cdot P$ is the condition that $\alpha, \beta \in Z_q^*$ are secret values.

C. System Model

Different from other traditional schemes, due to the particularity of our model, we replace the trusted authority (TA) with the users, for the generation of system parameters and partial key distribution.

As shown in Fig. 3, our scheme contains three entities including a smart phone user U and two cloud server C_i, C_j.

- U: The cell phone user, who publishes system parameters and distributes partial private keys to both the cloud servers.
- C_i or $Cloud_i$: The request data cloud server. This server verifies the validity of the user and performs mutual authentication and key negotiation with C_j.
- C_j or $Cloud_j$: The source data cloud server. This server verifies the validity of the user and performs mutual authentication and key negotiation with C_i.

In our model, users when changing their mobile devices, should first register and login to both the cloud server C_i (the new provider) and the cloud server C_j (the original mobile phone provider). The two cloud servers are now in a peer scenario. The user distributes part of the private key to both the cloud servers through a secure channel. Then, C_i and C_j exchanges related information, and C_i sends a request message to C_j to initiate the mutual authentication and key agreement process.

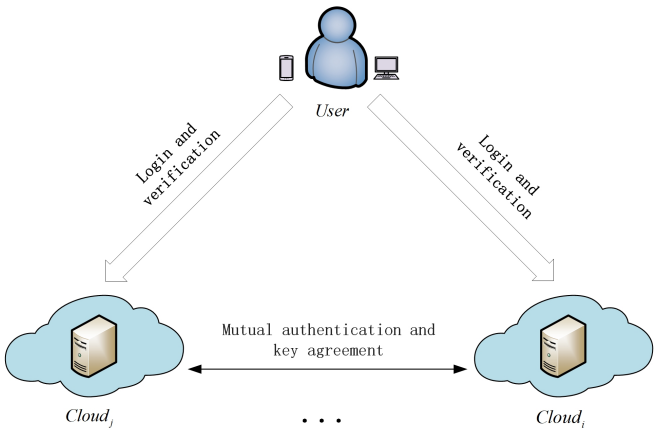


Fig. 3. System Model

IV. The Proposed PCAKA Scheme

This section describes our proposed scheme in detail. The proposed PCAKA scheme is divided into three phases.

The symbols defined below are used in the PCAKA scheme.

- ID_i: C_i's identity
- ID_j: C_j's identity
- pid_i: C_i's pseudonym
- pid_j: C_j's pseudonym

- $E_\omega()$: symmetric encryption with key ω
- ΔT: a smaller time interval
- $\Rightarrow$: a secure communication channel
- $\rightarrow$: a public communication channel

A. Initialization Phase

At this stage, U generates the primary key and system parameters as following.

1) U randomly selects two big prime numbers p, q, an elliptic curve $E(F_p)$ defined in F_p and a generator P with the order q.
2) U randomly generates its primary private key $\omega \in Z_q^*$ and computes the public key of the system, where $P_{pub} = \omega P$.
3) U chooses four one-way secure hash functions:
   a) H_1: $\{0,1\}^* \times G_q \rightarrow Z_q^*$
   b) H_2: $\{0,1\}^* \times G_q \times G_q \times \{0,1\}^* \rightarrow Z_q^*$
   c) H_3: $\{0,1\}^* \times \{0,1\}^* \times G_q \times G_q \times \{0,1\}^* \times G_q \times G_q \times \rightarrow Z_q^*$
   d) H_4: $\{0,1\}^* \times \{0,1\}^* \times G_q \times G_q \times G_q \times G_q \times G_q \times \{0,1\}^* \rightarrow Z_q^*$
4) U publishes system arguments as, $argums = \{E(F_p), p, q, P, P_{pub}, H_1, H_2, H_3, H_4\}$ and saves ω secretly.

B. Login Phase

At this stage, user logs in and assigns a key to the cloud server. As shown in Fig. 4 and Fig. 5, the user and the cloud server performs this process as bellow.

C_i Join Phase

1) User U login to the cloud server C_i.
2) C_i sends its identity ID_i to U via the secure channel.
3) U randomly selects an element $r_i \in Z_q^*$ and saves it secretly. U computes $R_i = r_i P, S_i = \omega^{-1} R_i$ and generates C_i's pseudo identity $pid_i = E_\omega(r_i, ID_i)$. Then U computes $\alpha_i = H_1(pid_i, R_i), y_i = \omega^{-1} r_i + \omega \alpha_i$. Finally, U sends $\{pid_i, R_i, y_i, S_i\}$ to C_i by a secure channel.
4) When receiving the message from U, C_i checks $y_i ? = S_i + \alpha_i P_{pub}$. If the verification fails, C_i aborts the protocol. Otherwise, C_i selects a random number $x_i \in Z_q^*$, computes $X_i = x_i P$ and saves (x_i, y_i) as its private key. At last, C_i publishes R_i and the public key (S_i, X_i).

C_j Join Phase

1) User U login to the cloud server C_j.
2) C_j sends its identity ID_j to U via the secure channel.
3) U randomly selects an element $r_j \in Z_q^*$ and saves it secretly. U computes $R_j = r_j P, S_j = \omega^{-1} R_j$ and generates C_j's pseudo identity $pid_j = E_\omega(r_j, ID_j)$. Then U computes $\alpha_j = H_1(pid_j, R_j), y_j = \omega^{-1} r_j + \omega \alpha_j$. Finally, U sends $\{pid_j, R_j, y_j, S_j\}$to C_j through a secure channel.

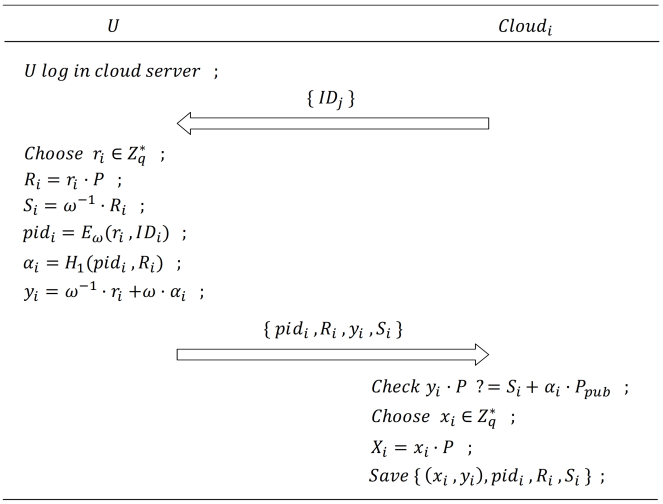


Fig. 4. The cloud i join phase

4) When receiving the message from U, C_j checks $y_j ? = S_j + \alpha_j P_{pub}$. If the verification fails, C_j aborts the protocol. Otherwise, C_j selects a random number $x_j \in Z_q^*$, computes $X_j = x_j P$ and saves (x_j, y_j) as its private key. At last, C_j publishes R_j and the public key (S_j, X_j).

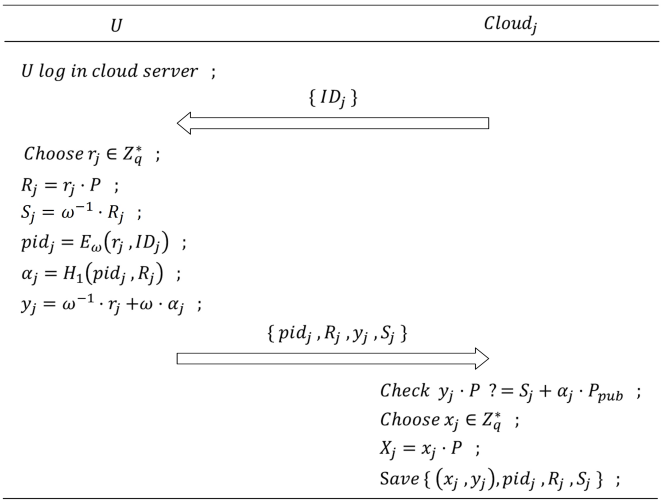


Fig. 5. The cloud j join phase

C. Cloud Handshake Phase

When C_i and C_j wants to establish a session, the two servers exchanges information $INFO_{C_i} = \{pid_i\}$ and $INFO_{C_j} = \{pid_j\}$. The following steps are then executed by C_i and C_j, as shown in Fig. 6.

1) C_i selects a random number $a_i \in Z_q^*$ and obtains the current timestamp T_i^1. Then C_i computes $A_i = a_i P, \beta_i = H_2(pid_i, A_i, S_i, T_i^1), \gamma_i = x_i(a_i + x_i \beta_i + y_i)^{-1} mod\ q$. C_i's signature is the tuple σ_i, where $\sigma_i = (\gamma_i, A_i)$. C_i computes $V_i = H_3(pid_i, pid_j, R_i, R_j, \gamma_i, A_i, X_i)$. Finally, C_i sends $\{\sigma_i, V_i, T_i^1\}$ to C_j by a public channel.

2) When receiving the message from C_i, C_j obtains the current timestamp T_j^1 firstly. Then C_j checks $T_j^1 - T_i^1? \leq \Delta T$. If this is not true, C_j aborts the session. Otherwise, C_j computes $\alpha_i = H_1(pid_i, R_i)$, $\beta_i = H_2(pid_i, A_i, S_i, T_i^1)$, $R_i^{'} = A_i + (\beta_i X_i) + (\alpha_i P_{pub}) + S_i$, $X_i^{'} = \gamma_i R_i^{'}$ and $V_i^{'} = H_3(pid_i, pid_j, R_i, R_j, \gamma_i, A_i, X_i^{'})$. Next, C_j checks $V_i^{'}$?= V_i. If this is not true, C_j aborts the protocol. Or else, C_j selects a random number $a_j \in Z_q^*$ and computes $A_j = a_j P, \beta_j = H_2(pid_j, A_j, S_j, T_j^1)$. Then C_j computes $\gamma_j = x_j(a_j + x_j\beta_j + y_j)^{-1} mod\ q$, $V_j = H_3(pid_i, pid_j, R_i, R_j, \gamma_j, A_j, X_j)$. C_j computes session key $sk_j = H_4(pid_i, pid_j, R_i, R_j, V_i^{'}, V_j, x_j X_i^{'}, T_j^1)$. Finally, C_j sends $\{\sigma_j = (\gamma_j, A_j), V_j, T_j^1\}$ to C_i through a public channel.

3) While receiving the message from C_j, C_i obtains the current timestamp T_i^2 and checks $T_i^2 - T_j^1? \leq \Delta T$. If this is not true, C_i aborts the protocol. Otherwise, C_i computes $\alpha_j = H_1(pid_j, R_j)$, $\beta_j = H_2(pid_j, A_j, S_j, T_j^1)$, $R_j^{'} = A_j + (\beta_j X_j) + (\alpha_j P_{pub}) + S_j$ and $X_j^{'} = \gamma_j R_j^{'}$. Then, C_i computes $V_j^{'} = H_3(pid_i, pid_j, R_i, R_j, \gamma_j, A_j, X_j^{'})$ and checks $V_j^{'}$?=V_j. If this is not true, C_i aborts the protocol. Or else, C_i computes the session key $sk_i = H_4(pid_i, pid_j, R_i, R_j, V_i, V_j^{'}, x_i X_j^{'}, T_j^1)$.

V. Security Proof and Analysis

A. Security Model

Based on the works of He et al. [32] and Choi et al. [34], we adopt a security model for the PCAKA scheme. The security of our proposed authentication and key negotiation (PCAKA) scheme on a peer-to-peer cloud can be defined as a game between adversary $\mathcal{A}$ and challenger $\mathcal{C}$. We indicate the kth instance of Λ by Π_Λ^k, where $\Lambda \in \{C_1, C_2, ...\}$. $\mathcal{A}$ can perform some queries on $\mathcal{C}$, and $\mathcal{C}$ will responds as follows.

- $H_k(m_k)$: $\mathcal{C}$ randomly generates a number $n_k \in Z_q^*$ when $\mathcal{A}$ carries out the query with the message m_k. Then, $\mathcal{C}$ stores tuple (m_k, n_k) in the list L_{H_k}, which is initialized as an empty set. Finally, $\mathcal{C}$ gives n_k to $\mathcal{A}$ as the return value, where $k = 1, 2, 3, 4$.
- $SymEnc(m_k, k_k, c_k)$: When $\mathcal{A}$ carries out the query with the message m_k and key k_k, $\mathcal{C}$ produces a random number c_k. Then, $\mathcal{C}$ saves the tuple (m_k, k_k, c_k) in the list L_{se}, which is initialized as an empty set. At last, $\mathcal{C}$ gives c_k to $\mathcal{A}$ as the return value.
- $ExtractSec(ID_k)$: $\mathcal{C}$ produces C_k's secret value and stores it in the list L_{cloud}^1, which is initialized as empty, when $\mathcal{A}$ executes the query with C_k's identity ID_k.
- $ExtractPar(ID_k)$: $\mathcal{C}$ produces C_k's partial private key and saves it in the list L_{cloud}^2, which is initialized as empty, when $\mathcal{A}$ carries out the query with C_k's identity ID_k.
- $Send(\Pi_\Lambda^k, m)$: When $\mathcal{C}$ receives the message m from $\mathcal{A}$'s query, it carries out the PCAKA protocol and gives $\mathcal{A}$ as a result.
- $Reveal(\Pi_\Lambda^k)$: $\mathcal{C}$ gives $\mathcal{A}$ the session key in Π_Λ^k when responding to $\mathcal{A}$'s query.
- $Corrupt(ID_k)$: When $\mathcal{A}$ executes the query with cloud's identity ID_k, $\mathcal{C}$ returns C_k's private key to $\mathcal{A}$.
- $Test(\Pi_\Lambda^k)$: After receiving the query from $\mathcal{A}$, $\mathcal{C}$ randomly selects a bit $b \in \{0, 1\}$. If $b = 1$, $\mathcal{C}$ gives $\mathcal{A}$ the session key in Π_Λ^k; If $b = 0$, $\mathcal{C}$ randomly produces a large number as the session key and gives it to $\mathcal{A}$.

When $\mathcal{A}$ carried out the above queries, it outputs a bit b', which is a guess of b. b has been produced in the $Test$ query. Now $\mathcal{A}$ is considered to have breached the security of the authenticated key negotiation of the proposed PCAKA protocol when $b^{'} = b$. We use the symbol $\mathcal{P}$ to represent the proposed PCAKA scheme. Let $Pr[E_{b'=b}]$ denotes the probability of the event that $b^{'} = b$. The advantage that $\mathcal{A}$ attacking the security of the authenticated key negotiation of the proposed PCAKA protocol $\mathcal{P}$ can define as $Adv_{\mathcal{P}}^{\mathcal{A}}(AKA) = 2|Pr[E_{b'=b}] - \frac{1}{2}|$.

Definition 1. ***(AKA-secure)*** *We define that PCAKA protocol $\mathcal{P}$ is a secure authenticated key agreement protocol if $Adv_{\mathcal{P}}^{\mathcal{A}}(AKA)$ is negligible for any PPT adversary $\mathcal{A}$.*

$\mathcal{A}$ is considered to have breached the mutual authentication of the PCAKA protocol $\mathcal{P}$ when it fakes a legal login information or a response information. Let $E_{i \to j}$ and $E_{j \to i}$ represents the events that $\mathcal{A}$ faking a legal login information and a response information. The advantage that $\mathcal{A}$ attacking the mutual authentication of the protocol $\mathcal{P}$ is defined by the symbol $Adv_{\mathcal{P}}^{\mathcal{A}}(MA)$, where $Adv_{\mathcal{P}}^{\mathcal{A}}(MA) = Pr[E_{i \to j}] + Pr[E_{j \to i}]$.

Definition 2. ***(MA-secure)*** *We define that PCAKA protocol $\mathcal{P}$ is a secure mutual authenticated protocol if $Adv_{\mathcal{P}}^{\mathcal{A}}(MA)$ is negligible for any PPT adversary $\mathcal{A}$.*

B. Security Proof

In this section, we prove that our proposed PCAKA protocol is able to provide the security of the authenticated key agreement. We assume that the four hash functions in our scheme include four random oracles [35].

Lemma 1. *For the proposed PCAKA scheme, no polynomial adversary can use a non-negligible probability to fake a legal login information or the corresponding response information.*

Proof. Suppose the adversary $\mathcal{A}$ can use a non-negligible probability ϵ to fake a legal login information or the corresponding response information. Now we demonstrate how the challenger $\mathcal{C}$ can use a non-negligible probability to settle the ECCDH problem. □

Suppose, a random instance $(P, Q_1 = mP, Q_2 = nP)$ of ECCDH problem in G_q. If $\mathcal{C}$ can solve the problem, then $\mathcal{C}$ can figure out nmP. $\mathcal{C}$ randomly chooses the request cloud C_I and further considers the responder cloud C_J as the challenge cloud, whose identity are ID_I, ID_J respectively. During the beginning of the game, $\mathcal{C}$ randomly produces four numbers $r_I, \alpha_I, r_J, \alpha_J \in Z_q^*$. Then $\mathcal{C}$ sets $P_{pub} = Q_1 - r_I \cdot \alpha_I \cdot P - r_J \cdot \alpha_J \cdot P$ and sends the arguments $argums = \{p, q, E(F_p), P, P_{pub}, H_1, H_2, H_3, H_4\}$ to $\mathcal{A}$. Challenger $\mathcal{C}$ interacts with adversary $\mathcal{A}$ as follows:

- $H_k(m_k)$: When $\mathcal{A}$ carries out the query with message m_k, $\mathcal{C}$ checks if the list L_{H_k} has the tuple (m_k, n_k).

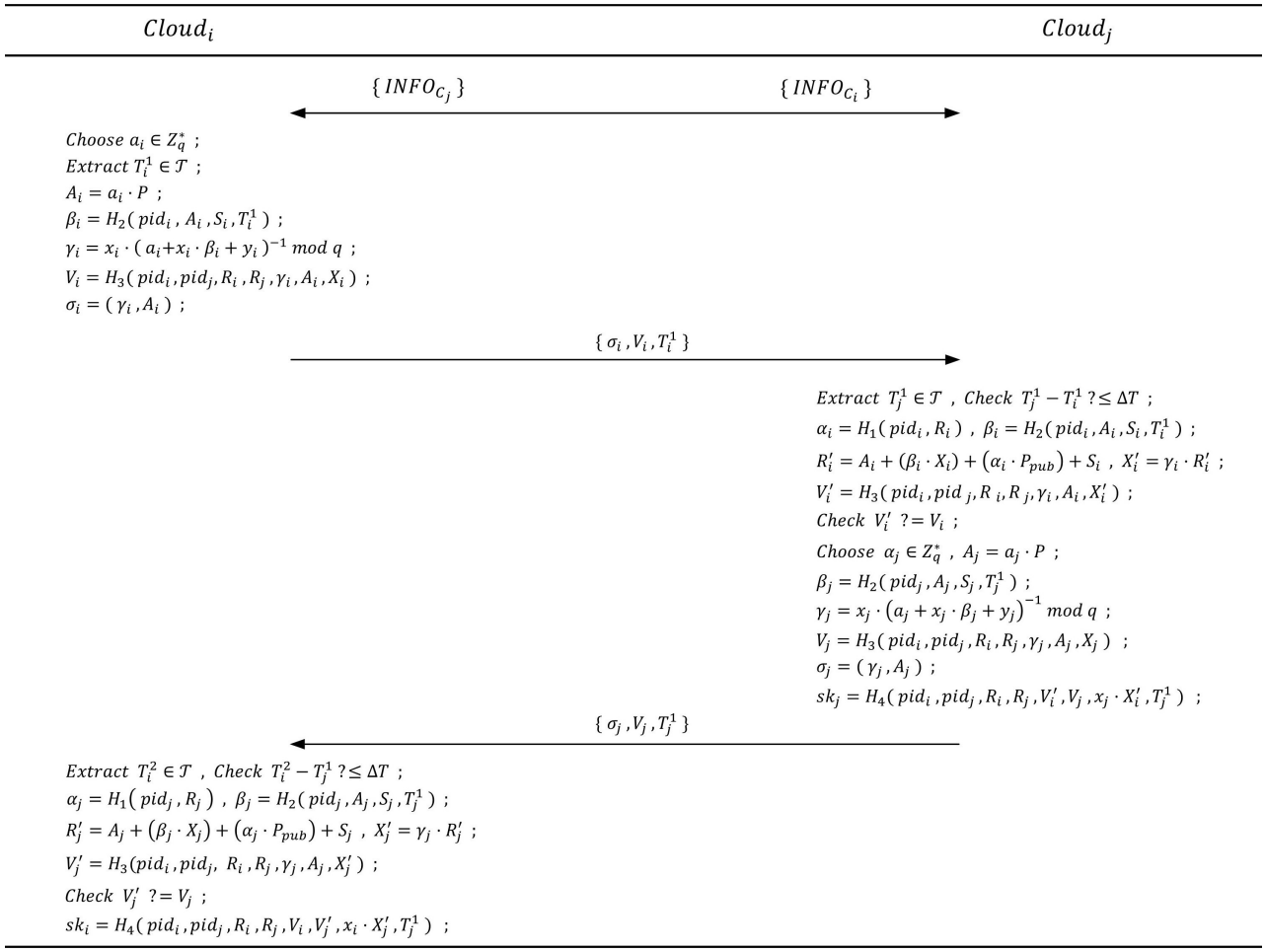


Fig. 6. The cloud handshake phase

If it exists, $\mathcal{C}$ returns n_k to $\mathcal{A}$; Otherwise, $\mathcal{C}$ randomly produces a number $n_k \in Z_q^*$, inserts the tuple (m_k, n_k) in to L_{H_k}, where $k = 1, 2, 3, 4$. Finally, $\mathcal{C}$ gives n_k to $\mathcal{A}$ as the return value.

- $SymEnc(m_k, k_k, c_k)$: Upon receiving the symmetric encryption query on message m_k and key k_k, $\mathcal{C}$ checks if the list L_{se} has the tuple (m_k, k_k, c_k). If it exists, $\mathcal{C}$ returns c_k to $\mathcal{A}$; Otherwise, $\mathcal{C}$ randomly produces a string $c_k \in Z_q^*$, inserts the tuple (m_k, n_k) in to L_{H_k}, where $k = 1, 2, 3, 4$. Finally, $\mathcal{C}$ gives n_k to $\mathcal{A}$ as the return value.
- $ExtractSec(ID_k)$: Upon receiving the extract query with cloud identity ID_k, $\mathcal{C}$ checks if the list L_{cloud}^1 has the tuple (ID_k, x_k, X_k). If it exists, $\mathcal{C}$ returns x_k to $\mathcal{A}$; Otherwise, $\mathcal{C}$ randomly selects a number $x_k \in Z_q^*$, computers $X_k = x_k P$. Finally, $\mathcal{C}$ stores the new tuple in L_{cloud}^1 and returns it to $\mathcal{A}$.
- $ExtractPar(ID_k)$: Upon receiving the extract query with cloud identity ID_k, $\mathcal{C}$ checks if the list L_{cloud}^2 has the tuple (ID_k, pid_k, R_k, y_k). If it exists, $\mathcal{C}$ returns y_k to $\mathcal{A}$; Otherwise, $\mathcal{C}$ executes as follows:
  - If $ID_k = ID_I$, $\mathcal{C}$ selects random numbers $r_I, pid_I \in Z_q^*$. And then, $\mathcal{C}$ inserts the tuple $(r_I, \perp, pid_I)$ into the list L_{se}. $\mathcal{C}$ computes $R_I = r_I P$ and sets $y_I = \perp$. Finally, $\mathcal{C}$ stores (pid_I, R_I, α_I) and $(ID_I, pid_I, R_I, \perp)$ into L_{H_1} and L_{cloud}^2 respectively.
  - If $ID_k = ID_J$, $\mathcal{C}$ selects random numbers $r_J, pid_J \in Z_q^*$. Now, $\mathcal{C}$ inserts the tuple $(r_J, \perp, pid_J)$ into the list L_{se}. $\mathcal{C}$ computes $R_J = r_J P$ and sets $y_I = \perp$. Finally, $\mathcal{C}$ stores (pid_J, R_J, α_J) and $(ID_J, pid_J, R_J, \perp)$ into L_{H_1} and L_{cloud}^2 respectively.
  - Otherwise, $\mathcal{C}$ selects random numbers $r_k, pid_k \in Z_q^*$. Now, $\mathcal{C}$ inserts the tuple $(r_k, \perp, pid_k)$ into the list L_{se}. $\mathcal{C}$ selects a random number $\alpha_k \in Z_q^*$, computes $R_k = r_k P - \alpha_k P_{pub}$ and sets $y_k = r_k \alpha_k$. Finally, $\mathcal{C}$ stores (pid_k, R_k, α_k) and (ID_k, pid_k, R_k, y_k) into L_{H_1} and L_{cloud}^2 respectively.
- $Send(\Pi_\Lambda^k, m)$: When receiving the query of message m, $\mathcal{C}$ responds as follows:
  - If $m = (\sigma_i, V_i)$: The query is message m, which is from C_i to C_j.
    - If $C_i = C_I$, $\mathcal{C}$ terminates the session.
    - If $C_i \neq C_I, C_j = C_J$, $\mathcal{C}$ terminates the session.
    - If $C_i \neq C_I, C_j \neq C_J$, $\mathcal{C}$ operates according to the protocol's specification.
  - If $m = (\sigma_j, V_j)$: The query is message m, which is from C_j to C_i.
    - If $C_j = C_J$, $\mathcal{C}$ terminates the session.

  - If $C_j \neq C_J, C_i = C_J$, $\mathcal{C}$ terminates the session.
  - If $C_j \neq C_J, C_i \neq C_I$, $\mathcal{C}$ operates according to the protocol's specification.
- $Reveal(\Pi_\Lambda^k)$: When receiving the query, $\mathcal{C}$ checks if $\Pi_\Lambda^k = \Pi_{C_I}^K$ or $\Pi_\Lambda^k = \Pi_{C_J}^K$. If yes, $\mathcal{C}$ aborts the session. Otherwise, $\mathcal{C}$ gives the session key of Π_Λ^k to $\mathcal{A}$ as the return value.
- $Corrupt(ID_k)$: When receiving the query, $\mathcal{C}$ looks up the tuple (ID_k, x_k, X_k) and (ID_k, pid_k, R_k, y_k) from the list L_{cloud}^1 and L_{cloud}^2 respectively. At last, $\mathcal{C}$ returns (x_k, X_k, R_k, y_k) to $\mathcal{A}$.

Now, $\mathcal{A}$ outputs a legal login message σ_i or respond message σ_j of its correspondent. If $(C_i, C_j) \neq (C_I, C_J)$, $\mathcal{C}$ terminates the game. Otherwise, $\mathcal{C}$ randomly selects a tuple $(*, pid_i, pid_j, R_i, R_j, \gamma_i, A_i, X_i^{'}, *)$ or $(*, pid_i, pid_j, R_i, R_j, \gamma_j, A_j, X_j^{'}, *)$ form the list L_{H_3}. Then $\mathcal{C}$ outputs $X_i^{'}$ or $X_j^{'}$ as the solution of the ECCDH problem. If $\mathcal{C}$ can solve the ECCDH problem with the probability $\epsilon^{'}$, the system needs to satisfy the following events.

- E_1: $\mathcal{C}$ does not terminate any $ExtractSec$ query.
- E_2: $\mathcal{C}$ does not terminate while responding to $Send$ query.
- E_3: $\mathcal{C}$ outputs a legitimate login message or its responder's message.
- E_4: $(\Pi_{C_i}^k, \Pi_{C_j}^k) = (\Pi_{C_I}^K, \Pi_{C_J}^K)$.
- E_5: $\mathcal{C}$ selects a right tuple from the list L_{H_3}.

Let $q_{es}, q_{send}, q_{H_i}, q_{ins}$ denotes the times of $ExtractSec$ queries, $Send$ queries, $Hash$ queries and instance $\Pi_{C_i}^k$(or $\Pi_{C_j}^k$). n denotes the number of cloud service providers registered by users in the system. Then we obtain:

$$Pr[E_1] \geq (1 - \frac{2}{q_{es}+1})^{q_{es}}$$

$$Pr[E_2|E_1] \geq (1 - \frac{2}{q_{send}+1})^{q_{send}}$$

$$Pr[E_3|E_1 \wedge E_2] \geq \epsilon$$

$$Pr[E_4|E_1 \wedge E_2 \wedge E_3] \geq \frac{1}{nq_{ins}}$$

$$Pr[E_5|E_1 \wedge E_2 \wedge E_3 \wedge E_4] \geq \frac{2}{q_{H_3}}$$

Therefore, the probability $\epsilon^{'}$ that $\mathcal{C}$ can solve the ECCDH problem is calculated as below.

$$\begin{aligned} \epsilon^{'} &= Pr[E_1 \wedge E_2 \wedge E_3] \wedge E_4 \wedge E_5] \\ &= Pr[E_1] \cdot [Pr[E_2|E_1] \cdot Pr[E_3|E_1 \wedge E_2] \cdot \\ &Pr[E_4|E_1 \wedge E_2 \wedge E_3] \cdot Pr[E_5|E_1 \wedge E_2 \wedge E_3 \wedge E_4] \\ &\geq 2\epsilon \cdot \frac{(1 - \frac{2}{q_{es}+1})^{q_{es}} \cdot (1 - \frac{2}{q_{send}+1})^{q_{send}}}{n \cdot q_{ins} \cdot q_{H_3}} \end{aligned} \tag{1}$$

This is the opposite of the difficulty of the ECCDH problem. Thus we obtain the conclusion that any PPT adversary $\mathcal{C}$ can not fake a legal login information or the corresponding response information with a non-negligible probability.

Theorem 1. *When the ECCDH problem is hard, the proposed PCAKA protocol $\mathcal{P}$ is MA-secure.*

Proof. According to lemma 1, we know that there no polynomial adversary can fake a legal login information or a corresponding response information while the ECCDH problem is hard. Hence, we confirm that the PCAKA protocol is MA-secure. □

Theorem 2. *The proposed PCAKA protocol $\mathcal{P}$ is AKA-secure if the underlying ECCDH problem is hard.*

Proof. We assume that a PPT adversary $\mathcal{A}$ can correctly guess b with a non-negligible probability ϵ during the $Test$ query. A challenger $\mathcal{C}$ solves the ECCDH problem with a non-negligible probability as follows. □

Let E_{SK} represents the event that $\mathcal{A}$ acquire the right session key about C_i and C_j. We can get $Pr[E_{SK}] \geq \frac{\epsilon}{2}$, due to the probability that $\mathcal{A}$ guesses a right b is at least $\frac{1}{2}$. Let E_{Test_i} and E_{Test_j} represent the event that $\mathcal{A}$ uses the $Test$ query to C_i and C_j and obtains their session key, respectively. If $\mathcal{A}$ can forge a legal login message, then $\mathcal{A}$ can break the $C_i - to - C_j$ authentication. This event is denoted by $E_{i \to j}$. Thus, we obtain the below.

$$\begin{aligned} Pr[E_{SK}] &= Pr[E_{SK} \wedge E_{Test_i}] + Pr[E_{SK} \wedge E_{Test_j}] \\ &= Pr[E_{SK} \wedge E_{Test_i}] + Pr[E_{SK} \wedge E_{Test_j} \wedge E_{i \to j}] \\ &+ Pr[E_{SK} \wedge E_{Test_j} \wedge \neg E_{i \to j}] \\ &\leq Pr[E_{SK} \wedge E_{Test_i}] + Pr[E_{i \to j}] \\ &+ Pr[E_{SK} \wedge E_{Test_j} \wedge \neg E_{i \to j}] \end{aligned} \tag{2}$$

That is to say,

$$\begin{aligned} &Pr[E_{SK} \wedge E_{Test_i}] + Pr[E_{SK} \wedge E_{Test_j} \wedge \neg E_{i \to j}] \\ &\geq \frac{\epsilon}{2} - Pr[E_{i \to j}] \end{aligned} \tag{3}$$

Because $E_{Test_j} \wedge \neg E_{i \to j}$ and E_{Test_i} are equivalent, we get

$$Pr[E_{SK} \wedge E_{Test_i}] \geq \frac{\epsilon}{4} - \frac{Pr[E_{i \to j}]}{2} \tag{4}$$

Thus, the probability that $\mathcal{A}$ breaking the authenticated key agreement is

$$\begin{aligned} &Pr[sk_i = H_4(*, *, *, *, (x_i \cdot x_j) \cdot P, *) | x_i, x_j \in Z_q^*] \\ &\geq \frac{\epsilon}{4} - \frac{Pr[E_{i \to j}]}{2} \end{aligned} \tag{5}$$

According to the above, we know that $Pr[E_{i \to j}]$ is negligible and ϵ is non-negligible. Thus, $\frac{\epsilon}{4} - \frac{Pr[E_{i \to j}]}{2}$ is non-negligible. Namely, the adversary $\mathcal{A}$ can solve the ECCDH problem. This conclusion contradicts with the difficulty of the ECCDH problem. Thus, we conclude that PCAKA protocol is AKA-secure on the premise that ECCDH problem is hard.

C. Security Analysis

In this section, we analyze the security characteristics of the PCAKA scheme under the above "Security Model".

Mutual Authentication.

According to lemma 1, no polynomial probability time adversary $\mathcal{A}$ can fake a legal login or response information. Therefore, cloud service providers participating in the negotiation can authenticate each other by verifying the signed messages. So, the proposed PCAKA protocol supports

the mutual authentication.

Session Key Agreement.

According to the protocol specification, both the parties involved in the key negotiation process construct a session key using their own known information (without disclosing private information). For example, C_i finds out $R_j^{'} = A_j + (\beta_j \cdot P) + (\alpha_j \cdot P_{pub}), X_j^{'} = \gamma_j \cdot R_j^{'}$ by the message (σ_j, V_j, T_j^1) received from C_j and the known information. Thus the session key $sk_i = H_4(pid_i, pid_j, R_i, R_j, x_i \cdot X_j^{'}, T_j^1)$ is calculated. In the same way, C_j figure out $sk_j = H_4(pid_i, pid_j, R_i, R_j, x_j \cdot X_i^{'}, T_j^1)$. According to section 4,however,$X_i = X_i^{'}(X_j = X_j^{'})$, $x_i \cdot X_j^{'} = x_i \cdot x_j \cdot P = x_j \cdot X_i^{'}$. We can obtain $sk_i = sk_j$. Thus, the proposed PCAKA scheme supports session key negotiation.

Identity Anonymity.

The two parties participating in the cloud handshakes and interacts with anonymous identities $pid_i = E_\omega(r_i, ID_i)$ and $pid_j = E_\omega(r_j, ID_j)$ in the PCAKA protocol. For them, anonymity protects the privacy of their identities when interacting with data on public channels. The adversary $\mathcal{A}$ can not extract the $ID_i(ID_j)$ from the $pid_i(pid_j)$. Thus, the proposed PCAKA protocol supports cloud anonymity.

Identity Traceability.

When $Cloud_i$ (or $Cloud_j$) uses an anonymous identity pid_i (or pid_j) to send error messages or illegal information, user U can use ω to extract the real identity ID_i(or ID_j). Therefore, the PCAKA scheme supports identity tracking.

Perfect Forward Secrecy.

Suppose that the adversary $\mathcal{A}$ can access the current private keys (x_i, y_i) and (x_j, y_j) of the cloud servers, respectively. However, the random numbers x_i and x_j are generated by C_i and C_j, respectively, and are updated with the process of building the session key each time. In addition, in order to obtain the previous x_i and x_j, $\mathcal{A}$ needs to extract them from the previous X_i and X_j, so that $X_i = x_i \cdot P, X_j = x_j \cdot P$. That means $\mathcal{A}$ needs to be dealt with the ECCDL problem. Therefore, the PCAKA scheme provides perfect forward secrecy.

Replay attack.

Timestamps (T_i^1, T_j^1, T_i^2) are used in the authentication process of the PACAKA protocol. Communications from both the side generate fresh random numbers (a_i, a_j) and compute $A_i = a_i \cdot P, A_j = a_j \cdot P$, so as to embed the immutable parameter γ_i, γ_j. The authentication message V_i, V_j contains the parameters γ_i, γ_j. Due to the freshness of a_i and a_j, both parties during the conversation can judge whether the message is being replayed by checking the correctness of the received message. Thus, the PCAKA scheme can withstand replay attacks.

Man-in-the-middle attack.

Based on the above security analysis, the proposed PCAKA protocol provides mutual authentication between C_i and C_j. That is to say, no adversary can deceive either side. Thus, our scheme can resist man-in-the-middle attack.

Impersonation attack.

Based on the above analysis, we know that no PPT adversary $\mathcal{A}$ can forge a legal login information or a corresponding response message, if it doesn't have the secure key of C_i or C_j. Thus, the PCAKA protocol can also resist impersonation attack.

Tampering attack.

According to our proposed protocol, γ_i is the core of the validation to C_i. The verification is that C_j checks if $V_i^{'} ? = V_i$, where $V_i^{'} = H_3(pid_i, pid_j, R_i, R_j, \gamma_i, A_i, X_i^{'})$. However, if $\mathcal{A}$ do not have the secure key (x_i, y_i) of C_i, it can not modify γ_i. Thus, it will not pass the verification of C_j, and it is the same case for γ_j. So, the PCAKA also provides immunity to tampering attack.

D. Security Comparison

In this section, we compare the security performance of Hsieh et al.'s scheme [27], Odelu et al.'s scheme [28], Li et al.'s scheme [29] and our scheme. Let S1, S2, S3, S4, S5, S6, S7, S8, S9 denote mutual authentication, session key agreement, identity anonymity, identity traceability, perfect forward security, resistance of replay attack, resistance of mam-in-the-middle attack, resistance of impersonation attack and resistance of tampering attack, respectively. The comparisons are shown in Table I.

According to Amin and Biswas [36], Hsieh et al.'s scheme does not provide identity anonymity and it cannot provide defense against impersonation attack. We find that the Odelu et al.'s scheme does not provide identity anonymity and identity traceability. We also find that Li et al.'s scheme does not achieve identity traceability. However, our scheme can provide all of the security requirements in the Table I.

TABLE I
SECURITY COMPARISONS

	Hsieh et al.'s scheme [27]	Odelu et al.'s scheme [28]	Li et al.'s scheme [29]	our scheme
S1	✓	✓	✓	✓
S2	✓	✓	✓	✓
S3	× [36]	×	✓	✓
S4	✓	×	×	✓
S5	✓	✓	✓	✓
S6	✓	✓	✓	✓
S7	✓	✓	✓	✓
S8	× [36]	✓	✓	✓
S9	✓	✓	✓	✓

✓: The security requirement is satisfied.
× : The security requirement is not satisfied.

VI. PERFORMANCE ANALYSIS

In this section, we compare and analyze the schemes proposed by Hsieh et al. [27], Odelu et al. [28], Li et al.

[29] against our proposed scheme from the perspectives of computational cost and communication cost.

We select a bilinear mapping $e : G_1 \times G_1 \rightarrow G_2$ for the aforementioned existing three schemes. G_1 is the additive cyclic group of prime order q, which is generated by an elliptic curve $E(F_p)$. G_2 is the multiplicative group of prime order q, which is generated by an elliptic curve $E(F_p)$.

A. Computation Cost

To analyze the computation costs of the four schemes, we use a few uniform basic cryptographic operations. The run time of the operations used in this analysis include the following:

- T_{hp}: The time taken to execute a hash-to-point operation.
- T_b: The time taken to execute a bilinear pairing operation.
- T_{pm}: The time taken to execute a point multiplication operation in G_1.
- T_{me}: The time taken to execute a modular exponentiation operation.
- T_{pa}: The time taken to execute a point addition operation.
- T_h: The time taken to execute a general hash operation.
- T_{mul}: The time taken to execute a multiplication operation in G_2.

The run time for some of the base operations, in reference to [29], are shown in Table II.

TABLE II
THE RUNTIME OF FUNDAMENTAL OPERATION (MS)

Operation	The user	The server
T_{hp}	30.40	5.18
T_b	32.55	5.02
T_{pm}	11.85	2.04
T_{me}	3.05	0.53
T_{pa}	0.10	0.02
T_h	0.225	0.015
T_{mul}	0.05	0.003

In Hsieh et al.'s protocol, the session sponsor side (the user) needs to carry out one hash-to-point operation, seven point multiplication operations, one point addition operation and eight general hash operations. Thus, the sponsor needs $T_{hp} + 7T_{pm} + T_{pa} + 8T_h \approx 115.25(ms)$. The responder side (the server) needs to carry out one hash-to-point operation, two bilinear pairing operations, five point multiplication operations, one point addition operation and four general hash operation. Thus, the responder needs $T_{hp}+2T_b+5T_{pm}+T_{pa}+3T_h \approx 25.485(ms)$.

Similarly, the sponsor of Odelu et al.'s protocol needs $2T_{pm} + T_{pa} + 2T_{me} + 6T_h \approx 31.25(ms)$, and the responder needs $2T_b + 2T_{pm} + T_{me} + T_{pa} + 6T_h \approx 14.76(ms)$.

The sponsor of Li et al.'s protocol needs $9T_h + 2T_{me} \approx 8.125(ms)$, and the responder needs $T_b+T_{hp}+T_{mul}+4T_{me}+5T_h \approx 12.398(ms)$.

In our proposed PCAKA scheme, the sponsor needs to carry out five point multiplication operations, six hash operations, three point addition operations and one modular exponentiation operation. Thus, the sponsor needs $5T_{pm} + 3T_{pa} + T_{me} + 6T_h \approx 10.88(ms)$. Because our proposed PCAKA scheme is symmetrical, the responder performs the same operations as the initiator. Therefore, the responder also needs $5T_{pm} + 3T_{pa} + T_{me} + 6T_h \approx 10.88(ms)$. The computation cost comparison results are shown in Table III, Fig. 7 and Fig. 8.

The session sponsor for our PCAKA scheme is the cloud server, while the session sponsor for the other three schemes is the user or smart meter. Therefore, the data presented in Fig. 7 will be significantly different. We found that the computational overhead of the session sponsor in Li et al.'s scheme is lower than that of the sponsor in our scheme. This is because of the fact that Li et al. used only the computationally trivial hash operation and modular exponentiation operation when designing operations for the session sponsor. However, in our scheme, although the session sponsor is the cloud server, it used the elliptic curve point multiplication operation with high computational overheads, so the sponsor's computational overhead is still greater than that of the Li et al's scheme.

On the responder side of the session, both Hsieh et al.'s scheme and Odelu et al.'s scheme used a lot of hash-to-points and bilinear pairing operations that require an intense computation, so their schemes characterize high computational overheads. In Li et al. scheme, although their computationally intensive operations contains only one bilinear pairing operation and one hash-to-point operation, the responder side still comprises high computational overheads. Our scheme only uses five point multiplication and other low-computational operations, and so the responder characterize the lowest computational overhead among the studied four schemes.

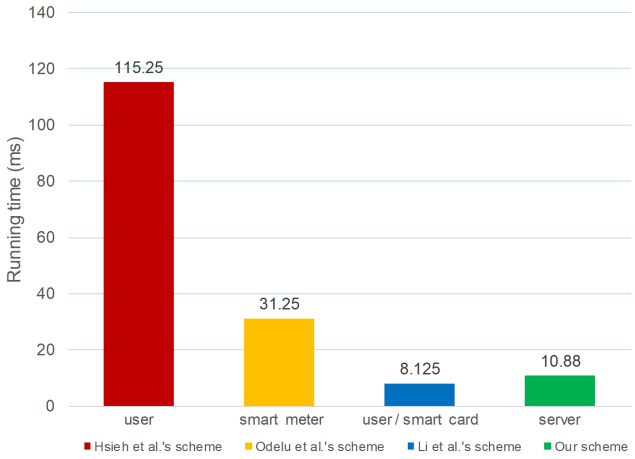


Fig. 7. Computational costs comparison: Sponsor side.

B. Communication Cost

In order to compare the communication overhead incurred by the four schemes during the key negotiation process, we set the length of p as 512-bits and the length of q as 160-bits.

Let $|G_1|$, $|G_2|$ and $|Z_q|$ represent the size of an element in G_1, G_2 and Z_q^*, respectively, and $|H|$ denotes the length of the result of the general hash operation. In our scheme, G_q is equivalent to G_1. That is to say, $|G_q| = 1024(bits), |G_1| = 1024(bits), |G_2| = 1024(bits), |Z_q| = 160(bits), |H| =$

TABLE III
COMPUTATION COST COMPARES (MS)

	Hsieh et al.'s scheme [27]	Odelu et al.'s scheme [28]	Li et al.'s scheme [29]	Our scheme
Sponsor	$T_{hp}+7T_{pm}$ +T_{pa} +$8T_h$ (115.25)	$2T_{pm}+T_{pa}+2T_{me}$ +$6T_h$ (31.25)	$2T_{me}+9T_h$ (8.125)	$5T_{pm}+3T_{pa}+T_{me}$ +$6T_h$ (10.88)
Responder	$T_{hp}+2T_b+5T_{pm}$ +T_{pa}+$3T_h$ (25.485)	$2T_b+2T_{pm}+T_{me}$ +$T_{pa}+6T_h$ (14.76)	$T_b+T_{hp}+T_{mul}$ +$4T_{me}+5T_h$ (12.398)	$5T_{pm}+3T_{pa}+T_{me}$ +$6T_h$ (10.88)

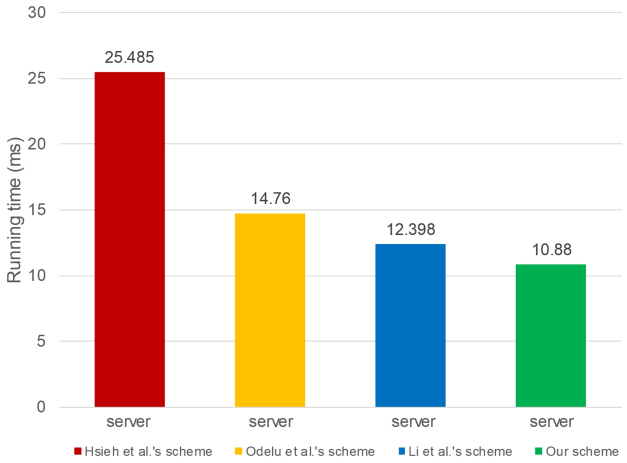


Fig. 8. Computational costs comparison: Responder side.

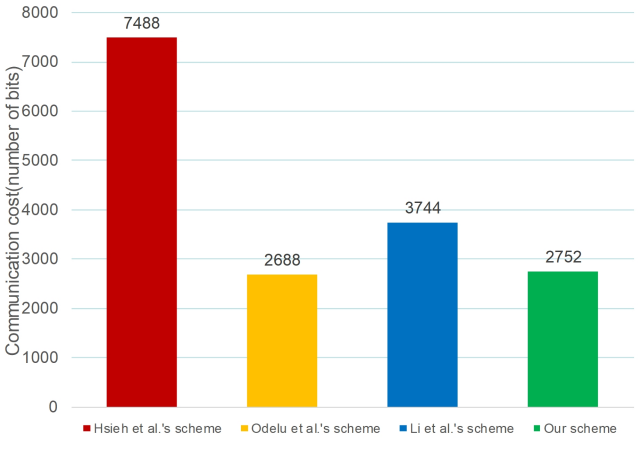


Fig. 9. Communication cost comparison.

$160(bits)$. $|T|$ represents the length of the timestamp. We also assume that $|T|=32(bits)$.

In Hsieh et al.'s protocol, the session sponsor side (the user) needs to send the message $(xAuth_i, C_m, R_i, B_{ij}, M_i, Auth_{ij})$, and the responder side (the server) needs to send the message $(Auth_{ji}, W_j, R_j)$. $xAuth_i, C_m, R_i, B_{ij}, M_i, W_j, R_j \in G_1$ and $Auth_{ij}, Auth_{ji} \in Z_q^*$. So, Hsieh et al.'s communication cost is $7|G_1|+2|H|=7\times 1024+2\times 160=7488(bits)$.

In Odelu et al.'s protocol, the session sponsor side (the smart meter) needs to send the message (T_1, C_1, A_1, A_3), and the responder side (the server) needs to send the message (g_2, A_2). A_1, A_2, A_3 are the outputs of general hash operation. $T_1, g_2 \in G_2$ and $C_1 \in Z_q^*$. So, Odelu et al.'s communication cost is $2|G_2|+|Z_q|+3|H|=2\times 1024+160+3\times 160=2688(bits)$.

In Li et al.'s protocol, the session sponsor side (the smart card) needs to send the message $(F_{ui}, k_{ui}, B_{ui}, d_{tui}, t, D_{ui})$, and the responder side (the server) needs to send the message (D_{sj}, k_{sj}). t is the timestamp. D_{sj}, D_{ui} are the outputs of general hash operation. $k_{ui}, B_{ui}, k_{sj} \in G_2$ and $F_{ui}, d_{tui} \in Z_q^*$. So, Li et al.'s communication cost is $|T|+3|G_1|+2|Z_q|+2|H|=32+3\times 1024+2\times 160+2\times 160=3744(bits)$.

In our scheme, the sponsor side (the server) needs to send the message $(pid_i, \gamma_i, A_i, T_i^1)$, and the responder side (the server) needs to send the message $(pid_j, \gamma_j, A_j, T_j^1)$. pid_i and pid_j are the outputs of general hash operation. $\gamma_i, \gamma_j \in Z_q^*$ and $A_i, A_j \in G_q$. T_i^1 and T_j^1 are the timestamps. Therefore, our scheme's communication cost is $2|H|+2|Z_q|+2|G_q|+2|T|=2\times 160+2\times 160+2\times 1024+2\times 32=2752(bits)$. The communication cost comparison results are shown in Fig. 9.

VII. CONCLUSION

This paper proposed a novel scheme to transfer user data between different cloud servers based on a key agreement protocol. Through the mathematical analysis and comparative evaluation presented in this paper, the advantages of our scheme are proved from three aspects: security performance, calculation costs and communication costs. Our proposed scheme can efficiently solve the primary problem of trust during data migration between cloud servers and further can provide anonymity for the identity of cloud servers. On the premise of protecting the privacy of cloud service providers, our proposed scheme indirectly protects the privacy of users. In addition, the identity traceability provided by our proposed scheme also enables users to effectively constrain the cloud service providers.

As a future work, we plan to explore and develop a protocol that allows multiple users to share data across different cloud servers, with the motivation of enhancing the efficiency of data sharing among multiple users.

ACKNOWLEDGEMENTS

The work was supported by the National Natural Science Foundation of China (No. U1936220, No. 61702005, No. 61872001), the Special Fund for Key Program of Science and Technology of Anhui Province, China (No. 18030901027), the Open Fund for Discipline Construction, Institute of Physical Science and Information Technology, Anhui University. The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this paper.

REFERENCES

[1] C. I. network information center, "The 44th china statistical report on internet development," http://www.cnnic.net.cn/hlwfzyj/hlwxzbg/hlwtjbg/201908/P020190830356787490958.pdf, 2019.

[2] B. Li, J. Li, and L. Liu, "Cloudmon: a resource-efficient iaas cloud monitoring system based on networked intrusion detection system virtual appliances," *Concurrency and Computation: Practice and Experience*, vol. 27, no. 8, pp. 1861–1885, 2015.

[3] J. Cui, H. Zhou, H. Zhong, and Y. Xu, "Akser: attribute-based keyword search with efficient revocation in cloud computing," *Information Sciences*, vol. 423, pp. 343–352, 2018.

[4] J. Cui, H. Zhong, W. Luo, and J. Zhang, "Area-based mobile multicast group key management scheme for secure mobile cooperative sensing," *Science China Information Sciences*, vol. 60, no. 9, p. 098104, 2017.

[5] J. Cui, H. Zhou, Y. Xu, and H. Zhong, "Ooabks: Online/offline attribute-based encryption for keyword search in mobile cloud," *Information Sciences*, vol. 489, pp. 63–77, 2019.

[6] D. Petcu, "Portability and interoperability between clouds: challenges and case study," in *European Conference on a Service-Based Internet*. Springer, 2011, pp. 62–74.

[7] T. Binz, F. Leymann, and D. Schumm, "Cmotion: A framework for migration of applications into and between clouds," in *2011 IEEE International Conference on Service-Oriented Computing and Applications (SOCA)*. IEEE, 2011, pp. 1–4.

[8] M. N. Shirazi, H. C. Kuan, and H. Dolatabadi, "Design patterns to enable data portability between clouds' databases," in *2012 12th International Conference on Computational Science and Its Applications*. IEEE, 2012, pp. 117–120.

[9] X. Liang, Z. Cao, H. Lin, and J. Shao, "Attribute based proxy re-encryption with delegating capabilities," in *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, 2009, pp. 276–286.

[10] K. Liang, M. H. Au, J. K. Liu, W. Susilo, D. S. Wong, G. Yang, Y. Yu, and A. Yang, "A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing," *Future Generation Computer Systems*, vol. 52, pp. 95–108, 2015.

[11] P. Xu, T. Jiao, Q. Wu, W. Wang, and H. Jin, "Conditional identity-based broadcast proxy re-encryption and its application to cloud email," *IEEE Transactions on Computers*, vol. 65, no. 1, pp. 66–79, 2015.

[12] M. Sun, C. Ge, L. Fang, and J. Wang, "A proxy broadcast re-encryption for cloud data sharing," *Multimedia Tools and Applications*, vol. 77, no. 9, pp. 10 455–10 469, 2018.

[13] G. Chunpeng, Z. Liu, J. Xia, and F. Liming, "Revocable identity-based broadcast proxy re-encryption for data sharing in clouds," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[14] X. Liu, Y. Zhang, B. Wang, and J. Yan, "Mona: Secure multi-owner data sharing for dynamic groups in the cloud," *IEEE transactions on parallel and distributed systems*, vol. 24, no. 6, pp. 1182–1191, 2012.

[15] J. Yuan and S. Yu, "Efficient public integrity checking for cloud data sharing with multi-user modification," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 2121–2129.

[16] M. Ali, R. Dhamotharan, E. Khan, S. U. Khan, A. V. Vasilakos, K. Li, and A. Y. Zomaya, "Sedasc: secure data sharing in clouds," *IEEE Systems Journal*, vol. 11, no. 2, pp. 395–404, 2015.

[17] J. Li, Y. Zhang, X. Chen, and Y. Xiang, "Secure attribute-based data sharing for resource-limited users in cloud computing," *Computers & Security*, vol. 72, pp. 1–12, 2018.

[18] U. M. Maurer, "Secret key agreement by public discussion from common information," *IEEE transactions on information theory*, vol. 39, no. 3, pp. 733–742, 1993.

[19] R. Lu, X. Lin, X. Liang, and X. Shen, "A secure handshake scheme with symptoms-matching for mhealthcare social network," *Mobile Networks and Applications*, vol. 16, no. 6, pp. 683–694, 2011.

[20] X. Liu and W. Ma, "Cdaka: a provably-secure heterogeneous cross-domain authenticated key agreement protocol with symptoms-matching in tmis," *Journal of medical systems*, vol. 42, no. 8, p. 135, 2018.

[21] J.-L. Tsai and N.-W. Lo, "A privacy-aware authentication scheme for distributed mobile cloud computing services," *IEEE systems journal*, vol. 9, no. 3, pp. 805–815, 2015.

[22] J. Xu, D. Zhang, L. Liu, and X. Li, "Dynamic authentication for cross-realm soa-based business processes," *IEEE Transactions on services computing*, vol. 5, no. 1, pp. 20–32, 2010.

[23] A. Irshad, M. Sher, H. F. Ahmad, B. A. Alzahrani, S. A. Chaudhry, and R. Kumar, "An improved multi-server authentication scheme for distributed mobile cloud computing services." *TIIS*, vol. 10, no. 12, pp. 5529–5552, 2016.

[24] A. B. Amor, M. Abid, and A. Meddeb, "A privacy-preserving authentication scheme in an edge-fog environment," in *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*. IEEE, 2017, pp. 1225–1231.

[25] V. Odelu, A. K. Das, S. Kumari, X. Huang, and M. Wazid, "Provably secure authenticated key agreement scheme for distributed mobile cloud computing services," *Future Generation Computer Systems*, vol. 68, pp. 74–88, 2017.

[26] K. Mahmood, X. Li, S. A. Chaudhry, H. Naqvi, S. Kumari, A. K. Sangaiah, and J. J. Rodrigues, "Pairing based anonymous and secure key agreement protocol for smart grid edge computing infrastructure," *Future Generation Computer Systems*, vol. 88, pp. 491–500, 2018.

[27] W.-B. Hsieh and J.-S. Leu, "An anonymous mobile user authentication protocol using self-certified public keys based on multi-server architectures," *The Journal of Supercomputing*, vol. 70, no. 1, pp. 133–148, 2014.

[28] V. Odelu, A. K. Das, M. Wazid, and M. Conti, "Provably secure authenticated key agreement scheme for smart grid," *IEEE Transactions on Smart Grid*, vol. 9, no. 3, pp. 1900–1910, 2016.

[29] W. Li, L. Xuelian, J. Gao, and H. Y. Wang, "Design of secure authenticated key management protocol for cloud computing environments," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[30] J. Cui, X. Zhang, H. Zhong, J. Zhang, and L. Liu, "Extensible conditional privacy protection authentication scheme for secure vehicular networks in a multi-cloud environment," *IEEE Transactions on Information Forensics and Security*, 2019.

[31] J. Wang, L. Wu, K.-K. R. Choo, and D. He, "Blockchain based anonymous authentication with key management for smart grid edge computing infrastructure," *IEEE Transactions on Industrial Informatics*, 2019.

[32] D. He, N. Kumar, H. Wang, L. Wang, K.-K. R. Choo, and A. Vinel, "A provably-secure cross-domain handshake scheme with symptoms-matching for mobile healthcare social network," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 633–645, 2016.

[33] X. Jia, D. He, N. Kumar, and K.-K. R. Choo, "A provably secure and efficient identity-based anonymous authentication scheme for mobile edge computing," *IEEE Systems Journal*, 2019.

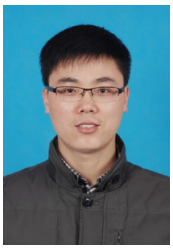
[34] K. Y. Choi, J. Y. Hwang, D. H. Lee, and I. S. Seo, "Id-based authenticated key agreement for low-power mobile devices," in *Australasian Conference on Information Security and Privacy*. Springer, 2005, pp. 494–505.

[35] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of cryptology*, vol. 13, no. 3, pp. 361–396, 2000.

[36] R. Amin and G. Biswas, "Design and analysis of bilinear pairing based mutual authentication and key agreement protocol usable in multi-server environment," *Wireless Personal Communications*, vol. 84, no. 1, pp. 439–462, 2015.

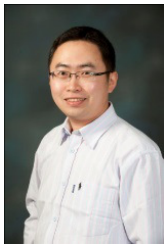
Hong Zhong was born in Anhui Province, China, in 1965. She received her PhD degree in computer science from University of Science and Technology of China in 2005. She is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). She has over 120 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network and Service Management, IEEE Transactions on Big Data and IEEE Internet of Things Journal), academic books and international conferences.

Chuanwang Zhang was born in Anhui Province, China , in 1996. He is now a graduate student in the School of Computer Science and Technology, Anhui University. His research focuses on cloud computing security and edge computing security.



Jie Cui was born in Henan Province, China, in 1980. He received his Ph.D. degree in University of Science and Technology of China in 2012. He is currently an associate professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). He has over 90 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network and Service Management, IEEE Transactions on Emerging Topics in Computing, IEEE Transactions on Circuits and Systems and IEEE Internet of Things Journal), academic books and international conferences.

Yan Xu is currently an associate professor of School of Computer Science and Technology at Anhui University. She received the BS and MS degrees from Shandong University in 2004 and 2007, respectively, and the PhD degree from University of Science and Technology of China in 2015. Her research interests include information security and applied cryptography.



Lu Liu is the Professor of Informatics and Head of School of Informatics in the University of Leicester, UK. Prof Liu received the Ph.D. degree from University of Surrey, UK and MSc in Data Communication Systems from Brunel University, UK. Prof Liu's research interests are in areas of cloud computing, service computing, computer networks and peer-to-peer networking. He is a Fellow of British Computer Society (BCS).