J Supercomput (2018) 74:5813–5845
https://doi.org/10.1007/s11227-016-1948-8

Attribute-based authentication on the cloud for thin clients

Maged Hamada Ibrahim[1] · Saru Kumari[2] · Ashok Kumar Das[3] · Vanga Odelu[4]

Published online: 2 January 2017

Abstract We propose two new authentication schemes for the cloud that support private attribute-based authentication services. The basic scheme is non-anonymous attribute-based authentication scheme. The extended scheme of the basic scheme is fully anonymous attribute-based authentication scheme to realize full anonymity and unlinkability services. In the proposed schemes, a user is authenticated by the remote server if the intersection of the set of his/her assigned attributes and the server's required attributes exceeds a satisfactory predefined level. Unlike existing attribute-based encryption and signature schemes that require the user to perform significant amount of elliptic curve bilinear pairings and modular exponentiations, and require the user to hold a significantly long decryption/signature key, in our schemes the user is not required to perform any bilinear pairings. With a fixed length private key, independent of the number of attributes, the cloud user performs only few exponentiations by

✉ Saru Kumari
saryusiirohi@gmail.com; saru@ccsuniversity.ac.in

Maged Hamada Ibrahim
mhii72@gmail.com; maged_ismail@h-eng.helwan.edu.eg

Ashok Kumar Das
iitkgp.akdas@gmail.com; ashok.das@iiit.ac.in

Vanga Odelu
odelu.vanga@gmail.com; odelu.phd@maths.iitkgp.ernet.in

1 Department of Electronics, Communications and Computers Engineering, Faculty of Engineering, Helwan University, 1, Sherif St., P.O. 11792, Helwan, Cairo, Egypt

2 Department of Mathematics, Ch. Charan Singh University, Meerut 250 005, Uttar Pradesh, India

3 Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India

4 Department of Mathematics, Indian Institute of Technology, Kharagpur 721 302, India

which he/she is able to authenticate himself/herself to the remote server and establish a session key with the server with the condition that he/she satisfies a predefined level of the server's attributes requirement. Therefore, our schemes are suitable for implementation on devices with limited resources. We provide the rigorous security of the proposed schemes and complexity analysis of our schemes. Finally, the security and performance comparisons of our schemes with the existing related schemes show that our schemes outperform other existing schemes.

Keywords Attribute-based authentication · Cloud computing · Thin clients · Smart cards · Access control · Formal security

1 Introduction

These days the disseminated open frameworks are developing extensively and also growing very fast. The grid, M2M and Fog computing, a paradigm that extends cloud computing and services to the edge of the network, frameworks are virtual associations with different independent spaces. In those frameworks, clients and asset suppliers are not in the same security area. Clients are typically recognized by their predefined personalities as well as by their qualities or attributes. Hence, the customary personality-based access control models are insufficient and, in this way, access to the framework should depend on choices for fulfillment of specific attributes' level [21].

In our schemes proposed in this paper, for a cloud user to obtain a cloud service, he/she must authenticate his/her identity to the server. He/she also must satisfy a set of server's attributes. If the intersection of the user's assigned attributes and the server's required attributes exceeds a satisfactory predefined level, the user is able to gain access to the server; otherwise, the user is denied.

In cloud confirmation, utilization of biometrics as characteristics has various critical favorable circumstances over standard validation systems. In the event that a biometric is utilized as an attribute, the identity check process is clear. The client must exhibit responsibility for biometric under the supervision of an all-around prepared administrator. In the event that impersonation assaults are recognized by the administrator (e.g., replaying the recording of a voice), the security of this stage is faulted for the utilized biometric method itself. The biometric estimation for an individual need not be kept secret. In reality, it is not in the event that it is utilized as open tokens. The verification plan should just ensure that a pernicious client cannot trick the cloud server into trusting that the client possesses a biometric property (tokens) that he/she does not.

Anonymity is one of the important services that must be available to the clients as long as they behave honestly. Clients' communication must be kept authenticated and anonymous unless malicious behaviors are detected. In this case, the accused user's clear identity must be traced and revealed by the system to solve accusations and revoke this malicious user.

Hiding the client's clear identity is not enough for realizing anonymity. To preserve clients' privacy, we need to look for solutions that prevent anyone from linking their different actions. At the risk of repeating themselves, there should be an effective

privacy-preserving solution against linkability of activities. In particular, most clients probably feel comfortable with a solution where they use a single pseudonym for all of their online shopping as long as it is guaranteed that it can never be linked to them. Although such solution may be enough for some applications, this unique pseudonym solution does not solve the real problem for other applications that require unlinkability of repeated actions [24].

1.1 Our motivation

In the recent cloud applications and organizational requirements, it is not enough that the cloud user authenticates himself/herself to the cloud server. Indeed, each server in the cloud has a defined set of attributes that must be satisfied to a certain level for the user to gain access to this server and obtain service. Hence, satisfying the server attributes is an essential part of the authentication process. Besides that a user must prove that he/she is registered in the cloud (identity authentication) and he/she also must prove that his/her assigned attributes are of the required level to access a particular server on the cloud (attribute-based). Existing attribute-based encryption schemes and attribute-based authentication/signature schemes have the major drawbacks that the user must hold a decryption/signature key of size proportional to the number of his attributes and it is required to perform a significant amount of modular exponentiations and pairings over elliptic curves, which make these schemes not suitable for thin clients applications, i.e., implementation on devices with limited resources such as smart phones and smart cards. Other computationally efficient schemes, such as U-Prove scheme [4,38], Idemix scheme [5,6,38] and HM12 scheme [43], do not provide advanced services, such as threshold access, tree-based access, possibility of revocation and some of them provide only one-show/partial anonymity.

1.2 Our contribution

The contribution of this paper is two-fold:

- First, we propose the basic scheme, which is an efficient attribute-based non-anonymous authentication scheme for the cloud to allow a cloud user, in a one-move non-interactive way with one-group-element private key and a set of unique non-secret attributes tokens, to prove his/her identity to the cloud server. The basic scheme also allows the server to check whether the registered user assigned attributes are of a satisfactory level to access this particular server (or service). If it is so, the cloud user is able to establish a session key with the server.
- Second, we provide a fully anonymous version of the non-anonymous scheme, which extends the basic scheme. Both proposed schemes also provide easy revocation of cloud users. The user in our schemes is not required to perform any bilinear pairing operations, whereas he/she performs only very few modular exponentiations and scalar multiplications on an elliptic curve. On the other hand, the complicated expensive computations are performed by the server.

1.3 Organization of the paper

The rest of this paper is organized as follows. In Sect. 2, we briefly discuss the required basic cryptographic primitives used in our proposed schemes. In Sect. 3, we briefly discuss the related existing work. Section 4 gives the design objectives, system and threat model of our schemes. In Sect. 5, we present our non-anonymous attribute-based authentication and its security and performance evaluation. We then extend our non-anonymous scheme to realize full anonymity in Sect. 6. Furthermore, security, performance evaluation and comparisons of the anonymous scheme are presented in Sect. 6. Some discussions are also presented in Sect. 6. We finally conclude the paper in Sect. 7.

2 Mathematical background

In this section, we discuss some necessary cryptographic tools and assumptions which are useful for describing and analyzing our proposed schemes in Sects. 5 and 6.

2.1 Diffie–Hellman assumptions

Let p and q be two distinct large primes such that $q|p-1$, that is, there be some integer k satisfying $p = kq + 1$. Let g be a generator of order q in Z_p.

Definition 1 *(Diffie–Hellman problem (DHP))* Let $x \in_R Z_q^*$ such that $|x| = |q|$, and $y = g^x \mod p$, where $|x|$ represents the bit length of x. Given (q, p, g, y), it is computationally infeasible to compute the discrete logarithm x.

Definition 2 *(Computational DH problem (CDHP))* Let $a, b \in_R Z_q^*$ be two large integers with $|a| = |b| = |q|$, and $A = g^a \mod p$ and $B = g^b \mod p$. Given (q, p, g, A, B), and without knowing a and b, it is infeasible to compute $g^{ab} \mod p$.

Definition 3 *(Decisional DH problem (DDHP))* Let $a, b, r \in_R Z_q^*$ be three large integers with $|a| = |b| = |r| = |q|$, and $A = g^a \mod p$ and $B = g^b \mod p$. Given (q, p, g, A, B), it is computationally infeasible to distinguish $g^{ab} \mod p$ from $g^r \mod p$ without knowing a, b and r.

2.2 Proofs of knowledge protocols

In this subsection, we review two basic proofs of knowledge protocols that are used in our schemes.

2.2.1 Proof of equality of two discrete logarithms

We review the protocol proposed in [35], which is widely used (e.g., in [9,16,17]). In this scheme, the public parameters are two large primes p, q such that $q|p-1$ and four integers $\alpha, \beta, G_1, G_2 \in Z_p^*$. The prover, say $\mathcal{P}$, wants to prove to a verifier, say

$\mathcal{V}$ that he/she knows $x \in Z_q^*$ such that $G_1 = \alpha^x \mod p$ and $G_2 = \beta^x \mod p$. The protocol works as follows.

- $\mathcal{P}$ picks $r \in_R Z_q^*$ and sends the tuple $\langle A = \alpha^r \bmod p, B = \beta^r \mod p\rangle$ to $\mathcal{V}$.
- $\mathcal{V}$ picks $c \in_R Z_q^*$ and sends c to $\mathcal{P}$.
- $\mathcal{P}$ computes $y = r + cx \mod q$ and sends y to $\mathcal{V}$.
- $\mathcal{V}$ checks validity of $\alpha^y = AG_1^c \mod p$ and $\beta^y = BG_2^c \mod p$.

There is a well-known standard way to make the above scheme non-interactive. Using a cryptographic one-way hash function $\mathcal{H}$ and setting $c = \mathcal{H}(A, B)$, the NIZKP proof of knowledge protocol Π_{LogEq} becomes as follows.

- $\mathcal{P}$ picks $r \in_R Z_q^*$, computes $c = \mathcal{H}(A, B)$, $A = \alpha^r \mod p$, $B = \beta^r \mod p$ and $y = r + cx \mod q$, and then sends the tuple $\langle A, B, y\rangle$ to $\mathcal{V}$.
- $\mathcal{V}$ computes c, and checks if $\alpha^y = AG_1^c \mod p$ and $\beta^y = BG_2^c \mod p$. $\mathcal{V}$ rejects if the checks fail; otherwise, it accepts.

In this paper, we denote this non-interactive version of the scheme by $\Pi_{LogEq} \leftarrow P_{LogEq}(\alpha, \beta, G_1, G_2, x)$.

2.2.2 Proof of knowledge of discrete logarithm

This is a well-known non-interactive protocol for proving the knowledge of a discrete logarithm. Select the public parameters as two large primes p and q such that $q|p-1$, and also select two integers $\alpha, Y \in Z_p^*$. The prover, say $\mathcal{P}$, proves to a verifier, say $\mathcal{V}$ that he/she knows $x \in Z_q^*$ such that $Y = \alpha^x \mod p$ without revealing the secret x. The protocol works as follows.

- $\mathcal{P}$ chooses $v \in_R Z_q^*$, computes $c = \mathcal{H}(\alpha, Y, T)$, $T = \alpha^v \mod p$, $r = v + cx \mod q$ and sends the tuple $\langle T, r\rangle$ to $\mathcal{V}$.
- $\mathcal{V}$ computes c and checks the validity of the condition $\alpha^r = TY^c \mod p$. $\mathcal{V}$ rejects if the check fails. Otherwise, $\mathcal{V}$ accepts it.

In this paper, we denote this non-interactive protocol as: $\Pi_{Log} \leftarrow P_{Log}(\alpha, Y, x)$.

2.3 Polynomial secret sharing

Let a dealer hold a secret $s \in Z_q$. To share this secret among a set $\mathcal{P} = \{P_1, \ldots, P_n\}$ of $n(>t)$ participants, the dealer constructs a polynomial $f(x) = \sum_{j=0}^{t} a_j x^j \mod q$, where $a_0 = s$ and $a_j \in_R Z_q$, where $j = 1, 2, \ldots, t$. $\forall i = 1, \ldots, n$, the dealer secretly delivers $f(i)$ to a participant P_i. To reconstruct the secret s, each participant P_i broadcasts $f(i)$. The participants then compute the secret s from any $t+1$ shares using the Lagrange interpolation as $s = f(0) = \Sigma_{i \in \mathcal{B}} \lambda_i f(i) \mod q$ where $\mathcal{B} \subset \mathcal{P}$, $|\mathcal{B}| = t + 1$ and λ_i is the Lagrange coefficient for the participant P_i [36].

2.4 Elliptic curve and its computational problems

Elliptic curves that are most commonly used have the standard Weierstrass form $E : y^2 = x^3 + ax + b$, which are defined over a prime field $\mathbb{F}_p$, where $p > 3$ is a prime and $a; b \in \mathbb{F}_p$ such that the condition $4a^3 + 27b^2 \neq 0 \mod p$ is satisfied. The set

$E(\mathbb{F}_p)$ of all solutions of E forms an abelian group under addition modulo p, where $\mathcal{O}$ is the identity element, also known as the point at infinity or zero point. The domain parameters of the elliptic curve $E(\mathbb{F}_p)$ are the six-tuple $\langle p, q, G, a, b\rangle$ where p is a field prime and G is a generator (base point) of order q.

In the following, we present the elliptic curve (EC) version of the Diffie–Hellman assumptions [10] as these are essential for analyzing the security of our schemes.

Definition 4 *(EC discrete logarithm problem (EC-DLP))* Let $P, Q \in E(\mathbb{F}_p)$ be two points on an elliptic curve $E(\mathbb{F}_p)$. A discrete logarithm of Q to the base P is an integer k such that $Q = kP$. Given P and Q in $E(\mathbb{F}_p)$, it is computationally infeasible to compute the discrete logarithm $k \in Z_p^*$.

Definition 5 *(EC computational DH problem (EC-CDHP))* Let $P \in E(\mathbb{F}_p)$. Given P, mP and nP in $E(\mathbb{F}_p)$ for some integers $m \in Z_p^*$ and $n \in Z_p^*$, it is computationally infeasible to compute mnP.

Definition 6 *(EC decisional DH problem (EC-DDHP))* Let $P \in E(\mathbb{F}_p)$. Given P, mP and nP in $E(\mathbb{F}_p$ for some integers $m \in Z_p^*$ and $n \in Z_p^*$, it is computationally infeasible to decide whether a point $Q \in E(\mathbb{F}_p)$ is on the form nmP or a random point.

2.5 Bilinear mapping and its computational problems

Let $\mathbb{G}_1, \mathbb{G}_2$ be groups of prime order q, and let g be a generator of $\mathbb{G}_1$. A function $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ is called the bilinear map if the following conditions hold [2].

- Bilinearity: $\forall a, b \in Z_q$ and $G \in G_1$, $e(aG, bG) = e(G, G)^{ab}$.
- Non-degeneracy: $e(G, G) \neq 1$.
- Computability: There exists an efficient algorithm to compute $e(P, Q)$, $\forall P, Q \in G_1$.

Definition 7 *(Decisional Bilinear DH assumption)* Let $a, b, c, z \in_R Z_q$ for a large prime q and $G \in G_1$. The decisional bilinear DH (BDH) assumption [1,2,12,40] is that no polynomial-time adversary will be able to distinguish the tuple $\langle A = aG, B = bG, C = cG, D = e(G, G)^{abc}\rangle$ from the tuple $\langle A = aG, B = bG, C = cG, D = e(G, G)^{z}\rangle$ with more than a negligible advantage.

2.6 Mix networks

The mix networks are needed in the registration phase to realize full anonymity in our extended scheme in Sect. 6. The aim of a mix network (mixnet) is to provide anonymity for a batch of inputs by changing their appearances and removing the order of arrival information. The main component of a mixnet is the stage, also known as the mix, that performs mixing on a batch of inputs. The inputs may arrive at the stage at different times. The mixing operation involves a cryptographic transformation using either decryption or encryption that changes the appearance of inputs followed by a permutation on the batch of transformed inputs. The mixed batch is then forwarded

in parallel by the stage at a specific time to the next destination. The batching and permutation together hide the order of arrival information of the inputs. Note that if the batch size is m, by observing the mixed output batch from a stage, one can only guess the correspondence with an input with probability $1/m$. Hence, an increase in the number of inputs to the stage increases the anonymity provided by the stage.

A mixnet consists of several interconnected stages to provide the required level of robustness. Each stage performs mixing on its inputs, and the mixed batch is then forwarded to the next stage in the mixnet or directly to their destinations. The interconnection of the stages determines the mixnet topology, and based on the topology of the mixnet, there can be a cascade mixnet or a free-route mixnet [46].

In the communication model of the mixnet, multiple senders communicate anonymously with one or more receivers. Each sender may communicate with a separate receiver as in an e-mail application, or multiple senders may communicate with a single receiver as in electronic voting. It is also possible that a sender communicates with multiple receivers as in multicast applications. Further, if the mixnet is used for a two-way communication,the receiver must be able to reply to the anonymous sender [45,46].

Definition 8 *(Mixnet transformation* $(\mathcal{X})$*)* The mixnet transformation $\mathcal{X}$ takes an input batch $(x_1, \ldots, x_m)$ of size m and produces the output batch $(y_1, \ldots, y_m)$ such that $y_i = x_{\pi(i)}$ for some random permutation π on the set $\{1, \ldots, m\}$.

3 Related work

Different access control techniques have been produced by researchers to safely access resources. Access control matrix methods are not sufficient for expansive associations with numerous objects and subjects. Likewise, there is discretionary access control, which relies on the caution of the proprietor of the article who is approved to control the data asset access. Discretionary access control does not give high level of security in distributed systems and it is a proprietorship-based framework. In mandatory access control, an authority chooses who will access which data. In mandatory access control, security naming control is not sufficiently moldable and is not suitable for task execution [26,30].

In RB-AC (Role-Based Access Control) [32,34], users are assigned to appropriate roles and access rights are associated with these roles. In Task-Based Authorization Control (TB-AC) [29], authorizations are enacted or deactivated by process state or current assignment. CloudMask [42] is a system for supporting fine grained and flexible shared access control of data in cloud systems. Under this model, the organizations can enjoy the benefit to host their data in the cloud without worrying about data security.

Compared to the traditional public-key infrastructure-based or identity-based cryptosystems, attribute-based encryption (ABE) is better for those applications needed for protection of the data privacy and secrecy in a cloud environment. In ABE [25,33], given a secret key as a set of attributes A, one can decrypt a ciphertext, which is encrypted with a public key based on a set of attributes A', only if the sets A and A' sufficiently overlap. Variants of the original ABE cryptosystem appeared in subsequent publications are based on the same concept, but with other flexibilities

[1,12,28,31,40]. The main disadvantage of attribute-based encryption techniques is that a user (decryptor) requires to perform a significant number of elliptic curve pairings which are computationally expensive. Also, the decryption key length grows linearly with the number of attributes. These impediments make those systems hard for implementation on devices with limited resources.

There have been attempts to define and realize attribute-based signatures (ABS) [11,19,20,23,27,37,41]. The schemes of [37,41] are direct applications of the known transform from identity-based encryptions to identity-based signatures [11]. The proposals of [19,23] capture weaker notions, where for a verifier to be able to verify, he/she must know in advance which attributes are used to generate the signature. The authors in [20,27] treated the privacy of attributes as a crucial prerequisite of ABS and formulated schemes that provably fulfill this necessity for threshold and monotone arrangements. However, these contributions have shortcomings that make their solutions unsuitable for the scenarios outlined in our contribution. Both schemes require that the verification policy must be known to the signer before signing. In applications, where other than the signer and the verifier (e.g., signature holder), this requirement of pre-knowledge of the verification policy is a major drawback in the schemes. A case of such applications is a credential framework, where a credential owner needs to fulfill distinctive check arrangements relying on the event and it is essential for productivity and ease-of-use purposes not to require diverse certifications for every confirmation approach. The work in [22] proposed an unidentifiable and untraceable perfectly anonymous attribute-based authentication scheme. The work in [25] presented an attribute authentication scheme based on vector space, which is computationally more efficient. However, there are security flaws in these schemes concerning collusion and replay attacks. A good survey on the existing attribute-based authentication techniques is found in [39].

Typical applications of blind signatures [8] include e-cash where a bank signs coins withdrawn by users, and e-voting where an authority signs public keys that voters later use to cast their votes. Another application of blind signature schemes is anonymous credentials, where the issuing authority blindly signs a key [4]. Later, Microsoft introduced a technology, called U-Prove, to overcome the long standing dilemma between identity assurance and privacy.[1,2] Their technology uses blind signature as a central building block.

Idemix [5,6] relies on the strong RSA assumption. Unlike in U-Prove, users in Idemix need only one secret key corresponding to all credentials and it is called a master key. Optionally, a user may choose to use several master keys but that does not influence the security and privacy properties of the scheme. The user can partly randomize a credential in the verification protocol. The user then proves to the verifier using zero knowledge technique that he/she possesses a valid credential instead of releasing any information about it.

Since neither U-prove nor Idemix provides the revocation service, the recent HM12 scheme [43] aims to provide an attribute-based authentication scheme with the possi-

[1] http://www.itforum.dk/downloads/Ronny_Bjones_Uprove.

[2] http://connect.microsoft.com/site642/Downloads/DownloadDetails.aspx?DownloadID=26953.

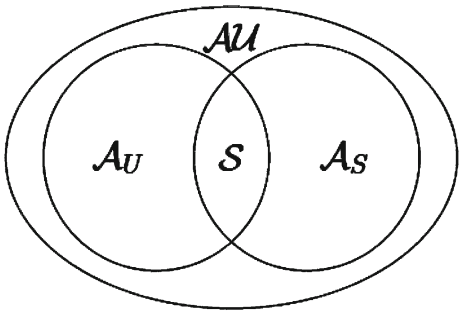

Fig. 1 Relationship among various attributes. $\mathcal{AU}$: attributes universe, $\mathcal{A}_U$: user's assigned attributes, $\mathcal{A}_S$: server's assigned attributes, $\mathcal{S}$: attributes intersection set

bility of revocation. The solution requires the revocation referee (RR) as an entity that must be referred to check whether a user is revoked.

4 System and threat model

4.1 System model

There are three main entities in our system: the registration authority (*RA*), the remote cloud server (*RS*) and the cloud user (U). It is assumed that RA follows the execution steps of the protocol word for word (honest-but-curious). The RS makes the decision whether or not U has access. We also assume that the RS can behave maliciously. However, in this case, a malicious RS may dishonestly deny access to a legitimate user. Hence, our schemes allow public verification of the user's attributes. U is also assumed malicious and he/she may play tricks trying to gain access to the server with insufficient attributes.

The *RA* publishes an attribute universe $\mathcal{AU}$. Each attribute $A_i \in \mathcal{AU}$ is assigned a unique index $i \in \mathcal{I}$, where $\mathcal{I}$ is the set of all indexes. Each attribute A_i is assigned a secret key t_i and a corresponding public elliptic curve point $t_i G$ where G is a generator point. Each U is assigned a private key x_U and an attributes satisfaction threshold k_U. Each U is assigned a subset of attributes $\mathcal{A}_U \subseteq \mathcal{AU}$ with a set of indexes $\mathcal{I}_U \subseteq \mathcal{I}$, while each remote server RS is assigned a subset $\mathcal{A}_S \subseteq \mathcal{AU}$ of attributes that is required to be satisfied by U to a k_U level. The RA assigns a set of attribute tokens $\mathcal{AT}_U$ of cardinality $|\mathcal{A}_U|$ (a unique token for each attribute). The attribute tokens bind the user's attributes to his/her identity on a threshold bases. A user U is authenticated to a remote server RS if the cardinality of the intersection $\mathcal{S} = \mathcal{A}_U \cap \mathcal{A}_S$ as shown in Fig. 1 exceeds a user's threshold k_U predefined by RA in the registration phase. If this is the case, U is able to establish a session key with RS in a one-move non-interactive way using his/her private key, the server's public key and the set of attribute tokens.

4.2 Threat model

In the threat model, we consider the following three attackers.

Outsider attacker: An attacker $\mathcal{A}$ who has the following abilities:

– $\mathcal{A}$ has the capability of eavesdropping on all the communication links in the system.
– $\mathcal{A}$ is able to record and replay messages to any entity.
– $\mathcal{A}$ has the ability to decompose and reassemble any user intercepted message (token, identity, etc.) as a new message and resend to any remote server.
– $\mathcal{A}$ observes the channels and links messages to users' identities to gain knowledge about their activities.

Device corruption attacker: In addition to his/her capabilities as an outsider attacker, he/she is capable of utilizing the private key of the compromised device to decrypt eavesdropped messages or forge messages. This represents a full corruption/control of the attacked user.
RA and RS corruption attacker: In addition to his/her capabilities as an outsider attacker, he/she is able to steal and manipulate *RA*'s and/or *RS*'s databases.

5 The proposed non-anonymous attribute-based authentication scheme

In this section, we present our basic scheme, called non-anonymous attribute-based authentication scheme. The entities incorporated in our scheme are a registration authority (*RA*), a remote cloud server (*RS*) and a cloud user amongst the set of cloud users in the system (U). The RA has its own public/private key pair (pk_{RA}, sk_{RA}), which allows authenticated transmission to the remote servers, where sk_{RA} is its private key and each remote server RS knows the public key pk_{RA} of RA. For describing the basic scheme, we use the notations listed in Table 1.

5.1 Description of the basic scheme

In the following subsections, we give the full descriptions of the phases of our non-anonymous attribute-based authentication scheme.

5.1.1 Initialization phase by the RA

The system public/private parameters are initialized by RA as follows.

Step 1. Given a security parameter κ, *RA* determines the size of the prime field groups and the underlying EC $E(\mathbb{F}_p)$. *RA* establishes a bilinear group $\mathbb{G}_1$ and a generator G for $\mathbb{G}_1$ of prime order q. In addition, *RA* defines the bilinear map $e\colon \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$.
Step 2. *RA* defines the attributes universe $\mathcal{AU} = \{A_1, \ldots, A_{|\mathcal{AU}|}\}$ and assigns each attribute A_i a unique index $i \in \mathcal{I}$. For simplicity and without loss of generality, let the set of indexes $\mathcal{I} = \{1, \ldots, |\mathcal{AU}|\}$.
Step 3. For each attribute A_i, *RA* picks $t_i \in_R Z_q$ and computes an elliptic curve point $t_i G$.

The RA's master private key is the tuple $\langle t_1, \ldots, t_{|\mathcal{AU}|}\rangle$, whereas its public key is the tuple $\langle t_1 G, \ldots, t_{|\mathcal{AU}|} G\rangle$.

Table 1 Notations used in our proposed schemes

Notation	Meaning
q, p	Two primes where p defines the field and q defines the order of the field
G	Elliptic curve base-point (generator)
t_i	Master private key for attribute A_i
U	Cloud thin client/user
RA	Registration authority
RS	Remote cloud server
id_U	U's clear identity
bid_U	U's blinded identity
k_U	Satisfaction level (threshold)
x_U	U's private key
r_U	A random integer in Z_q unique for user U and known only to RA
$a_U(x)$	A random polynomial unique for user U with its free term $r_U x_U$
$\mathcal{AU}, \mathcal{A}_U, \mathcal{A}_S$	Set of attributes universe, users' attributes and server's attributes
$\mathcal{A}$	Adversary
$\mathcal{I}, \mathcal{I}_U$	Set of all attributes indexes, the set of the user U's attributes indexes
$\mathcal{S}$	Attributes intersection set, $\mathcal{S} = \mathcal{A}_U \cap \mathcal{A}_S$
$\mathcal{H}$	Cryptographic one-way hash function: $\{0,1\}^* \rightarrow \{0,1\}^L$
$AT_U^{(i)}$	Attribute token for U corresponding to attribute A_i
$BAT_U^{(i)}$	Blinded version of attribute token $AT_U^{(i)}$
$\mathcal{AT}_U$	Set of all attributes tokens of user U
$\mathcal{BAT}_U$	Set of all blinded attributes tokens of user U
$\mathcal{ID}$	Set of all users' clear identities
$\mathcal{BID}$	Set of blind identities of at least two users
$\mathcal{X}$	A robust mixnet
$x \in_R S$	x is randomly chosen from the set S
TS_U	Current timestamp of user U
ΔT	Maximum transmission delay

5.1.2 Registration phase

We introduce two versions of the registration phase. The first version does not require any computations on the user's side. On the other hand, it has the key-escrow problem, where the RA knows the user's private key. The second version does not allow the RA to know the user's private key. However, there are few computations required to be performed by the user in the second version. The choice of which of the two versions

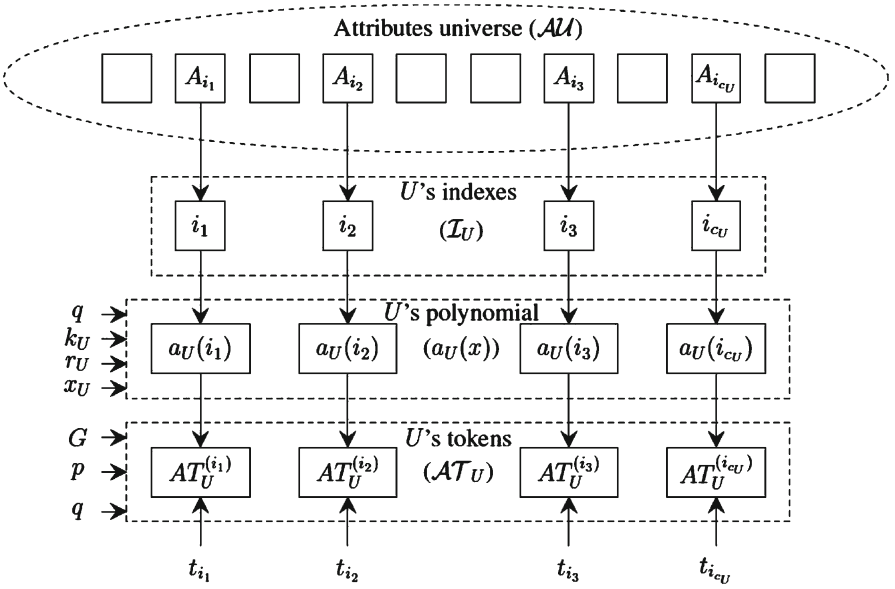

Fig. 2 Attributes tokens generation by RA for user U

to apply is left to the organization and the specific application. The generation of the attributes tokens for a user U is illustrated in Fig. 2.

Registration phase with key-escrow: The sequence diagram of this phase is shown in Fig. 3. A user U is registered on the system by RA as follows.

Step 1. RA picks $x_U \in_R Z_q$ as U's private key and a random secret integer $r_U \in_R Z_q$, and then computes $id_U = e(G, G)^{x_U r_U}$ as U's public identity.

Step 2. RA defines the subset of attributes $\mathcal{A}_U \subseteq \mathcal{AU}$, the corresponding subset of indexes $\mathcal{I}_U \subseteq \mathcal{I}$ for U, and the attributes satisfaction threshold k_U. RA sets up a random $(k_U - 1)$-degree polynomial, $a_U(x) = \sum_{i=0}^{k_U-1} a_i x^i$, where $a_0 = x_U r_U$ and each other $a_{i\neq 0} \in_R z_q$.

Step 3. For each attribute $A_i \in \mathcal{A}_U$, RA computes the attribute token as an elliptic curve point $AT_U^{(i)} = (a_U(i)/t_i)G$ and sets $\mathcal{AT}_U = \{AT_U^{(i_1)}, \ldots, AT_U^{(i_{c_U})}\}$. RA then sends $Tuple_U = \langle \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G, G)^{r_U}, id_U, x_U \rangle$ to U, where c_U is the cardinality of $\mathcal{A}_U$. RA also sends a signed version of id_U to each remote server RS.

Step 4. Finally, U stores $Tuple_U$.

We denote $\mathcal{ID} = \{id_1, \ldots, id_n\}$ as the set of the users' clear identities.

Remark 1 We emphasize that r_U picked by RA is kept secret from any entity including U. Note that the attribute tokens are not secret. However, we emphasis that these tokens are useless to anyone who does not know the corresponding private key x_U. Each attribute token $AT_U^{(i_j)}$ is associated with its unique index $i_j \in \mathcal{I}_U$ in the attributes universe $\mathcal{AU}$.

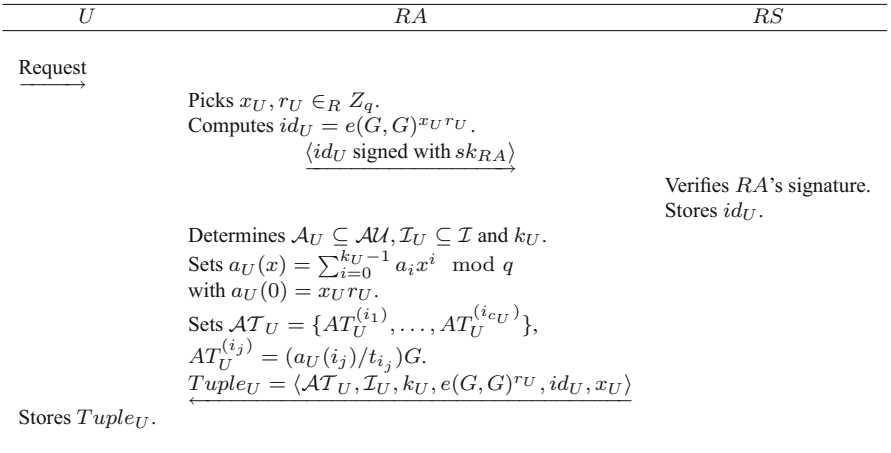

Fig. 3 The registration phase with key-escrow for the basic scheme

Registration phase without key-escrow: The sequence diagram of this phase is illustrated in Fig. 4. A user U needs to contact the *RA* for registration. This registration process is as follows.

Step 1. U picks $x_U \in_R Z_q$ as his/her own private key and computes $e(G,G)^{x_U}$. U also prepares a proof of knowledge of x_U to the base $e(G,G)$ as $\Pi_{Log} \leftarrow P_{Log}(e(G,G), e(G,G)^{x_U}, x_U)$. U then sends $\langle \Pi_{Log}, e(G,G)^{x_U}\rangle$ to RA.

Step 2. RA runs $V_{Log}(e(G,G), e(G,G)^{x_U}, \Pi_{Log})$ to verify Π_{Log}. RA aborts if the verification fails. RA defines the subset of attributes $\mathcal{A}_U \subseteq \mathcal{AU}$, the corresponding subset of indexes $\mathcal{I}_U \subseteq \mathcal{I}$ for U and the attributes satisfaction threshold k_U. RA then picks $r_U \in_R Z_q$ and sets up a random $(k_U - 1)$-degree polynomial $a_U(x)$ with its free term $a_U(0) = r_U$. For each attribute $i \in \mathcal{I}_U$, RA also computes the elliptic curve point $AT'^{(i)}_U = (a_U(i)/t_i)G$. RA calculates U's identity $id_U = e(G,G)^{r_U x_U}$. RA sends $\langle AT'^{(i_1)}_U, \ldots, AT'^{(i_{c_U})}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, id_U\rangle$ to U. Simultaneously, RA also sends a signed version of id_U to each remote server RS.

Step 3. U computes his/her attribute tokens $AT_U^{(i_1)}, \ldots, AT_U^{(i_{c_U})}$, where each $AT_U^{(i_j)} = x_U AT'^{(i_j)}_U$, and sets $\mathcal{AT}_U = \{AT_U^{(i_1)}, \ldots, AT_U^{(i_{c_U})}\}$. Finally, U stores $Tuple_U = \langle \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, id_U, x_U\rangle$.

5.1.3 Authentication phase

The sequence diagram of the authentication phase is given in Fig. 5. U authenticates a message m to *RS* using his/her tuple $Tuple_U$ as follows.

Step 1. U selects a random string r, generates the current timestamp TS_U and computes the hash value $h = \mathcal{H}(TS_U, m, r)$. U then calculates $z = h^{x_U}$. U computes NIZK proof of knowledge as $\Pi_{LogEq} \leftarrow P_{LogEq}(h, e(G,G)^{r_U}, z,$

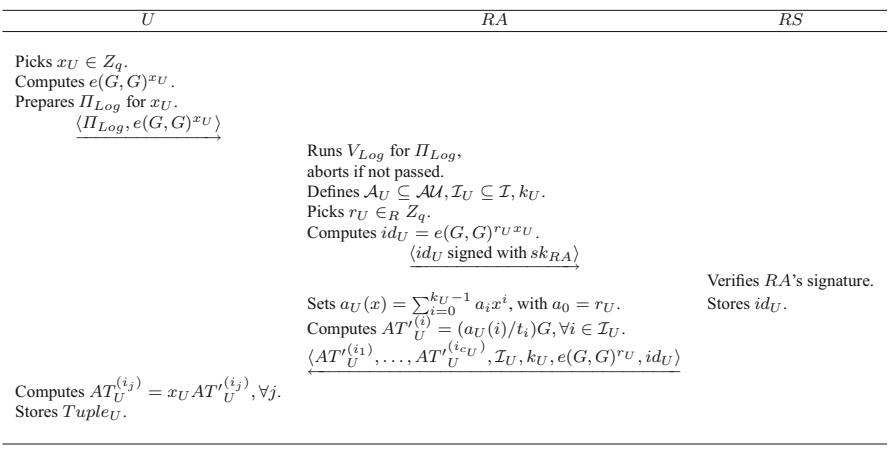

Fig. 4 The registration phase without key-escrow for the basic scheme

$id_U, x_U)$, which proves that $\log_h(z) = \log_{e(G,G)^{r_U}}(id_U) = x_U$. U then parses σ_U as $\langle r, z, id_U, \Pi_{LogEq} \rangle$. σ_U is U's signature on m. After that U sends the message $\langle m, \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, \sigma_U, TS_U \rangle$ to the remote server RS.

Step 2. On the reception of the message $\langle m, \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, \sigma_U, TS_U \rangle$ from user U, RS first checks the validity of the received timestamp TS_U by the condition $|TS^*_U - TS_U| < \Delta T$, where ΔT is the maximum transmission delay and TS^*_U is the time when the message $\langle m, \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, \sigma_U, TS_U \rangle$ is received by RS. If it is valid, RS parses σ_U as $(r, z, id_U, \Pi_{LogEq})$ and aborts if $id_U \notin \mathcal{ID}$. RS then computes $h = \mathcal{H}(TS_U, m, r)$ and runs the verification algorithm $V_{LogEq}(h, e(G,G)^{r_U}, z, id_U, \Pi_{LogEq})$. RS rejects m and aborts on failure of the verification.

Step 3. Using $\mathcal{I}_U$, RS determines the attributes intersection set $\mathcal{S}$ with U, i.e., $\mathcal{S} = \mathcal{A}_U \cap \mathcal{A}_S$. RS aborts in case $|\mathcal{S}| < k_U$. RS picks any k_U attribute tokens in the received message of the corresponding attributes found in $\mathcal{S}$. RS then calculates $\Pi_{i \in \mathcal{S}} e(AT^{(i)}_U, t_i G)^{\lambda_i}$ and checks equality with id_U, where λ_i is the Lagrange coefficient. If the equality does not hold, RS aborts. Otherwise, RS accepts m.

To prove the correctness of the computations on RS's side, we note that

$$\begin{aligned}
\Pi_{i \in \mathcal{S}} e(AT^{(i)}_U, t_i G)^{\lambda_i} &= \Pi_{i \in \mathcal{S}} e((a_U(i)/t_i)G, t_i G)^{\lambda_i} \\
&= \Pi_{i \in \mathcal{S}} e(G,G)^{a_U(i)\lambda_i} \\
&= e(G,G)^{\Sigma_{i \in \mathcal{S}} a_U(i)\lambda_i} \\
&= e(G,G)^{x_U r_U} \\
&= id_U.
\end{aligned}$$

U	RS
Picks a random string r.	
Generates current timestamp TS_U.	
Computes $h = \mathcal{H}(TS_U, m, r)$ for a message m.	
Computes $z = h^{x_U}$.	
Prepares Π_{LogEq}.	
Parses σ_U as $\langle r, z, id_U, \Pi_{LogEq}\rangle$.	
$\langle m, \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, \sigma_U, TS_U\rangle \longrightarrow$	
	Checks the validity of received timestamp TS_U.
	Aborts if it is not valid. Otherwise,
	parse σ_U as $\langle r, z, id_U, \Pi_{LogEq}\rangle$.
	Aborts if $id_U \notin \mathcal{ID}$.
	Computes $h = \mathcal{H}(TS_U, m, r)$.
	Runs $V_{LogEq}(h, e(G,G)^{r_U}, z, id_U, \Pi_{LogEq})$.
	Aborts on failure of V_{LogEq} and rejects m.
	Using $\mathcal{I}_U$ determines $\mathcal{S} = \mathcal{A}_U \cap \mathcal{A}_S$.
	Aborts in case $\lvert\mathcal{S}\rvert < k_U$.
	Picks any k_U attributes in $\mathcal{S}$.
	Aborts in case $\Pi_{i\in\mathcal{S}} e(AT_U^{(i)}, t_i G)^{\lambda_i} \neq id_U$.
	Accepts m.

Fig. 5 The authentication phase for the basic scheme

5.1.4 Revocation

To revoke a user U with identity id_U, the RA instructs the servers to remove this id_U from their databases.

5.2 Analysis of the basic scheme

In this section, we analyze the security of our basic non-anonymous scheme. We then evaluate the efficiency of the basic scheme and compare it with related existing non-anonymous schemes.

5.2.1 Security analysis

In the following, we show that our basic scheme prevents several known attacks.

- *Impersonation attack*: An adversary $\mathcal{A}$ who tries to masquerade a registered user U using id_U and attributes tokens of U will not be able to generate a valid proof of knowledge without knowing the corresponding private key x_U of that user U. Therefore, $\mathcal{A}$ fails to generate a valid z and Π_{LogEq}. Hence, $\mathcal{A}$ cannot generate a valid message $\langle m, \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, \sigma_U, TS_U\rangle$ on behalf of U and, as a result, the user impersonation attack is eliminated in our basic scheme.
- *Databases insertion attack*: An adversary $\mathcal{A}$ who is able to gain access to RA and RS databases, he/she may insert a valid id_A created with some key x_A and a random integer r_A to become a registered user in the system. $\mathcal{A}$ can generate a fake tuple $\langle \mathcal{I}_A, k_A, e(G,G)^{r_A}, id_A\rangle$. However, $\mathcal{A}$ cannot create his/her own set of

valid attribute tokens without the knowledge of the RA's master private key. Thus, $\mathcal{A}$ cannot succeed in the databases insertion attack.

- *Stolen smart device/card attack*: In case, an attacker is able to steal the user U's smart device/card, he/she can use the tuple $Tuple_U$ to impersonate U. However, using id_U and attributes tokens of U, the attacker will not be able to generate a valid proof of knowledge without knowing the corresponding private key x_U of that user U (see the registration phase without key-escrow in Sect. 5.1.2). Hence, our basic scheme is secure against the stolen smart device/card attack.
- *Replay attack*: Our scheme is essentially a signature scheme, where x_U is the signing key and id_U is the verification key of user U. Assume that an adversary $\mathcal{A}$ intercepts the message $\langle m, \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, \sigma_U, TS_U\rangle$ and resends it after sometime. However, the validity of the attached timestamp TS_U in the message will fail. $\mathcal{A}$ thus fails to launch the replay attack against our basic scheme.
- *Man-in-the-middle attack*: Suppose an adversary $\mathcal{A}$ intercepts the message $\langle m, \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, \sigma_U, TS_U\rangle$. Let $\mathcal{A}$ generates its own timestamp TS_A and a fake message m' and compute $h' = \mathcal{H}(TS_A, m', r')$. However, adversary $\mathcal{A}$ cannot compute the modified z as $z' = (h')^{x_U}$ without knowing the private key x_U of the user U. As a result, $\mathcal{A}$ cannot create another valid message, say $\langle m', \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, \sigma_U, TS_A\rangle$. Hence, the basic scheme has the ability to protect against the man-in-the-middle attack.
- *Users collaboration attack*: Malicious users in the system may attempt to collaborate and combine their tokens and keys to gain access to a remote server. Note that each user U is assigned a unique random polynomial $a_U(x)$ with random coefficients independent of any other user U''s polynomial $a_{U'}(x)$ from the finite field Z_q. The random integer r_U as part of the free term in the polynomial $a_U(x)$ ensures that U cannot maliciously delegate his/her tokens to be used by another user U'. The secret shares $a_U(i)$ and $a_{U'}(i)$ are inconsistent as they belong to different random polynomials and, thus, the quantities $(a_U(i)/t_i)G$ and $(a_{U'}(i)/t_i)G$ are totally inconsistent. Hence, the users collaboration attack does not help to gain access to a remote server in our basic scheme.
- *Users-server collaboration attack*: Several malicious users may collaborate with a malicious remote server to disclose the privacy of other users or deceive other servers. Malicious users may reveal their private keys x_U's. In our basic scheme, each user's private parameters are completely independent of those for any other users in the system. On the other hand, the servers have nothing in common except the set of identities $\mathcal{ID}$, which is not a secret. From the BDH assumption (Definition 7), the revealed information does not allow the collaborated entities to reveal any information about the system master keys and, hence, the security of the rest of the users and servers is preserved. This shows that our basic scheme is also secure against the users–server collaboration attack.
- *Non-repudiation*: Given a transcript of a valid transmitted message with the signature $\sigma_U = \langle r, z, id_U, \Pi_{LogEq}\rangle$, a user U cannot later deny the transmission of this message as this cannot be generated correctly without the knowledge of the private key x_U of U. The identity id_U is viewed as the registered verification key of U. Recall that $\Pi_{LogEq} \leftarrow P_{LogEq}(h, e(G,G)^{r_U}, z, id_U, x_U)$ is of the form

(A, B, y), where $A = h^w$, $B = e(G, G)^{r_U w}$, $c = \mathcal{H}(A, B)$ and $y = w + cx_U$ for a random $w \in Z_q$. The verification of the transcript is as follows.

- Using r, TS_U and m, compute $h = \mathcal{H}(TS_U, m, r)$.
- Parse Π_{LogEq} as (A, B, y).
- Compute $c = \mathcal{H}(A, B)$
- Check if $h^y = Az^c$ and $(e(G, G)^{r_U})^y = B(id_U)^c$.
- In case, both checks are valid, U is deemed to be the signer.

Therefore, our proposed scheme supports the non-repudiation service and it may be also used as an attribute-based digital signature scheme.

5.2.2 Complexity evaluation

In this section, we evaluate the basic scheme for storage, computation and communication overheads. Let $|x|$ denote the bit-length of x and let n be the number of registered users in the system.

Storage overhead: Consider the registration phase key-escrow phase. The RA stores the master secret parameters $(t_1, \ldots, t_{|\mathcal{AU}|})$. Each of these parameters is of size $|q|$ bits. Therefore, the storage cost for these parameters is $|\mathcal{AU}||q|$ bits. RA also stores the identities as the set $\mathcal{ID}$, where each identity id_U is of the form $e(G, G)^{x_U r_U}$ which is of $|p|$ bits and the set $\mathcal{ID}$ requires $n|p|$ bits. The total storage required by RA is then $(|\mathcal{AU}|)|q|+n|p|$ bits. Note that during the registration phase without key-escrow phase, the storage complexity of RA is same as that for registration phase key-escrow phase.

RS stores the set $\mathcal{ID}$, which requires $n|p|$ bits. RS also stores $\mathcal{A}_S$. Each element in $\mathcal{A}_S$ is of the form $t_i G$, which is an EC point of size $2|p|$ bits. Since an attribute token is an EC point, it requires $2|p|$ bits. The total storage overhead for RS is then $(n + 2|\mathcal{A}_S|)|p|$ bits. Similarly, we have computed the storage cost for a user U. The storage overheads for RA, RS and each user U are summarized in Table 2.

Computation overhead: During the registration phase with key-escrow phase, RA computes id_U for each user U, which requires one modular exponentiation. Also, RA computes $|\mathcal{A}_U|$ attribute tokens for each user U, and each is of the form $(a_U(i)/t_i)G$. These require $|\mathcal{A}_U|$ scalar multiplications and one modular exponentiation. RA further performs one modular multiplication to compute $r_U x_U$, and k_U modular multiplications and $k_U - 1$ modular additions to compute each $a_U(i)$. Therefore, RA needs $|\mathcal{A}_U|$ scalar multiplications, one modular exponentiations, $(1 + k_U|\mathcal{A}_U|)$ modular multiplications and $|\mathcal{A}_U|(k_U + 1)$ modular additions. Note that modular multiplications/additions are very efficient as compared to those for scalar multiplications and exponentiations. Similarly, the computation overheads for RS and each user U are computed and summarized in Table 2.

We also evaluate the computation time required by a user U on smart card and mobile device. Let $t_h, t_{mm}, t_{exp}, t_{ma}, t_{ecsm}$ and t_{pair} denote the execution times needed for one SHA-1 hash function, one modular multiplication, one modular exponentiation, one modular addition, one elliptic curve scalar multiplication and one bilinear pairing, respectively. From the experimental results [7,15], the computation times are mapped to the hashing time as the time unit, and these are provided in Table 3. Using the

Table 2 Storage and computation overheads

Entity	Storage cost (in bits)	Computation cost				
		Mod mult	Mod add	Mod exp	Scalar mult	Pairing
Storage and computation costs of our scheme (with key-escrow)						
RA	$\|\mathcal{AU}\|\|q\| + n\|p\|$	$1 + k_U\|\mathcal{A}_U\|$	$\|\mathcal{A}_U\|(k_U - 1)$	1	$\|\mathcal{A}_U\|$	–
RS	$(n + 2\|\mathcal{A}_S\|)\|p\|$	k_U	–	$4 + k_U$	–	k_U
U	$\|q\| + 2(\|\mathcal{A}_U\| + 1)\|p\|$	1	1	3	–	–
Storage and computation costs of our scheme (without key-escrow)						
RA	$\|\mathcal{AU}\|\|q\| + n\|p\|$	$4 + k_U\|\mathcal{A}_U\|$	$\|\mathcal{A}_U\|(k_U - 1)$	4	$\|\mathcal{A}_U\|$	–
RS	$(n + 2\|\mathcal{A}_S\|)\|p\|$	k_U	–	$4 + k_U$	–	k_U
U	$\|q\| + 2(\|\mathcal{A}_U\| + 1)\|p\|$	2	2	5	$\|\mathcal{A}_U\|$	–

Note: mod mult: modular multiplication; mod add: modular addition; mod exp: modular exponentiation; scalar mult: elliptic curve point multiplication; pairing: bilinear pairing operation

Table 3 Computation cost of different cryptographic operations mapped to the computation time of one-way hashing operations

Notation	Description	Cost
t_h	Execution time of one hash invocation	t_h
t_{mm}	Execution time of one modular multiplication	$2.5t_h$
t_{ma}	Execution time of one modular addition	$0.3t_h$
t_{exp}	Execution time of one modular exponentiation	$600t_h$
t_{inv}	Execution time of one modular inverse	$200t_h$
t_{ecsm}	Execution time of one scalar multiplication	$72.5t_h$
t_{pa}	Execution time of one point addition	$13t_h$
t_{pair}	Execution time of one elliptic curve pairing	$1550t_h$

Table 4 Concrete evaluation of the computation cost mapping to the time taken by one SHA-1 hash operation as the time unit

	With key-escrow	Without key-escrow
RA	$(602.5 + (71.7 + 2.8k_U)\lvert\mathcal{A}_U\rvert)t_h$	$(2410 + (71.7 + 2.8k_U)\lvert\mathcal{A}_U\rvert)t_h$
RS	$(2400 + 2152.5k_U)t_h$	$(2400 + 2152.5k_U)t_h$
U (registration)	–	$1202.8t_h$
U (authentication)	$1802t_h$	$1802t_h$
Total cost	$(4804.5 + 71.7\lvert\mathcal{A}_U\rvert +$ $(2152.5 + 2.8\lvert\mathcal{A}_U\rvert)k_U)t_h$	$(7814.8 + 71.7\lvert\mathcal{A}_U\rvert +$ $(2152.5 + 2.8\lvert\mathcal{A}_U\rvert)k_U)t_h$

Table 5 Communication overhead

Communication mode	Communication cost (in bits)
$RA \rightarrow U$	$2(\lvert\mathcal{A}_U\rvert + 1)\lvert p\rvert + \lvert q\rvert$
$U \rightarrow RA$ (with key-escrow)	–
$U \rightarrow RA$ (without key-escrow)	$2\lvert p\rvert + \lvert q\rvert$
$RA \rightarrow RS$	$\lvert p\rvert$ per user
$U \rightarrow RS$	$\lvert m\rvert + (2\lvert\mathcal{A}_U\rvert + 5)\lvert p\rvert + \lvert q\rvert$

information in Tables 2 and 3, Table 4 shows the computation costs mapping to the time required by one hashing operation (using SHA-1 hashing algorithm) as the time unit.

Communication overhead: We assume that the bit-length of the order of the elliptic curve is $|q| = 160$ bits, while the bit-length of the field prime is $|p| = 512$ bits. The bit-length of each communication mode is computed, and these are shown in Table 5.

5.2.3 Performance comparison

We compare our non-anonymous basic scheme with the original ABE scheme presented in [25]. Using Table 2, from Table 6 we see that our scheme offloads the

Table 6 Comparison of our basic scheme (without key-escrow) and the ABE scheme

	Our scheme (without key-escrow)			ABE scheme [25]		
	RA	*RS*	*U*	*RA*	*RS*	*U*
Mod mult	$4 + k_U\lvert\mathcal{A}_U\rvert$	k_U	2	$\lvert\mathcal{A}_U\rvert k_U$	1	$k_U + 2$
Mod add	$\lvert\mathcal{A}_U\rvert(k_U - 1)$	–	2	$\lvert\mathcal{A}_U\rvert(k_U - 1)$	–	–
Mod exp	4	$4 + k_U$	5	–	1	$k_U + 1$
Mod inv	–	–	–	–	–	1
Scalar mult	$\lvert\mathcal{A}_U\rvert$	–	$\lvert\mathcal{A}_U\rvert$	$\lvert\mathcal{A}_U\rvert$	$\lvert\mathcal{A}_S\rvert$	–
Pairing	–	k_U	–	–	–	k_U

note: mod inv: modular inverse

computation burden from the users to the servers in an efficient way. In the authentication phase, it requires the user to perform three modular exponentiations in our scheme, while in the ABE scheme [25], it requires the user to perform k_U parings in addition to $k_U + 1$ modular exponentiations. Unlike the ABE scheme [25], the computations required by the user in the authentication phase of our scheme are independent of the attributes sizes or the threshold k_U. Moreover, the size of the private key in our basic scheme is short, and it is also independent of the sizes of the attributes.

6 The proposed fully anonymous attribute-based authentication scheme

In this section, we extend our basic non-anonymous scheme presented in Sect. 5.1 to realize full anonymity and unlinkability services. We call this version as the extended scheme. We show how to realize these services against channel observers, remote servers (RS's), registration authority (RA), and collaboration of RA and all RS's. We employ/plug a mixnet $\mathcal{X}$ (Definition 8) in the registration phase. The mixnet $\mathcal{X}$ receives the blinded identities from at least $h(\geq 2)$ users, privately shuffles the received identities, and then delivers the shuffled version to the RA. In this case, the RA is not able to cluster any set of blinded identities for any particular user.

6.1 Description of the extended scheme

This scheme extends the basic non-anonymous scheme to allow U to anonymously conduct up to ℓ unlinkable sessions with the RS, while his/her transactions are kept anonymous and unlinkable. The initialization phase is the same as that for the basic non-anonymous scheme. We only modify the registration and authentication phases as described below. In this case, the registration without the key-escrow is mandatory.

6.1.1 Registration phase

This phase is illustrated in Fig. 6. A user U approaches to the RA for registration purpose. The registration process has the following steps:

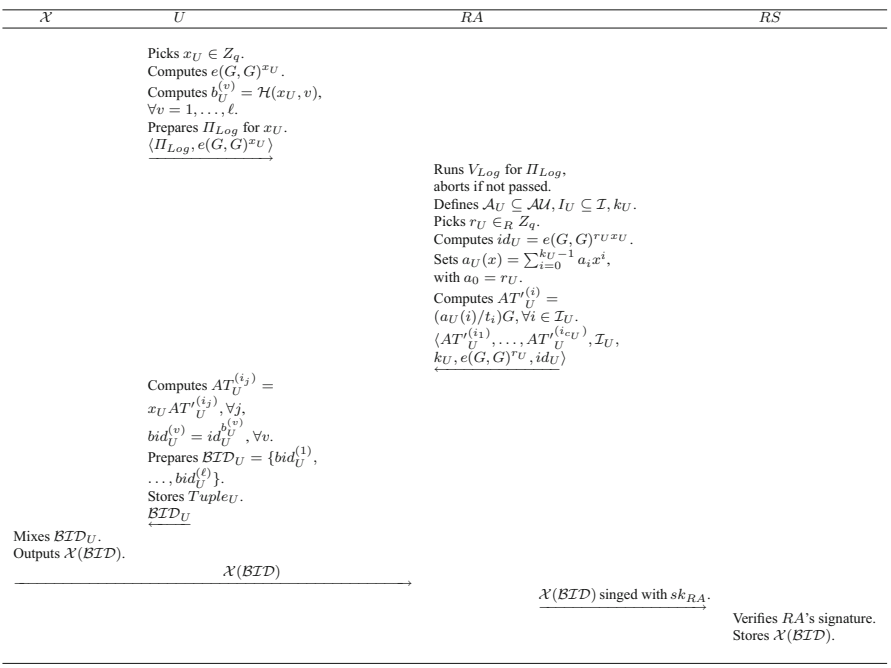

Fig. 6 The registration phase of the anonymous scheme

Step 1. U picks $x_U \in_R Z_q$ as his/her own private key, and computes $e(G,G)^{x_U}$, the blinding parameters $b_U^{(v)} = \mathcal{H}(x_U, v)$, for $v = 1, \ldots, \ell$. U prepares a proof of knowledge of x_U to the base $e(G,G)$ as $\Pi_{Log} \leftarrow P_{Log}(e(G,G), e(G,G)^{x_U}, x_U)$. U the sends the message $\langle \Pi_{Log}, e(G,G)^{x_U} \rangle$ to RA.

Step 2. After receiving the message, RA runs $V_{Log}(e(G,G), e(G,G)^{x_U}, \Pi_{Log})$ to verify Π_{Log}. RA aborts, if the verification fails. RA defines the subset of attributes $\mathcal{A}_U \subseteq \mathcal{AU}$, the corresponding subset of indexes $\mathcal{I}_U \subseteq \mathcal{I}$ for U and the attributes satisfaction threshold k_U. RA then picks $r_U \in_R Z_q$ and sets up a random $(k_U - 1)$-degree polynomial $a_U(x)$ with its free term $a_U(0) = r_U$. RA also computes U's identity $id_U = e(G,G)^{r_U x_U}$. After that for each attribute $A_i \in \mathcal{A}_U$, RA computes the elliptic curve point $AT'^{(i)}_U = (a_U(i)/t_i)G$, and sends the message $\langle AT'^{(i_1)}_U, \ldots, AT'^{(i_{c_U})}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, id_U \rangle$ to U.

Step 3. After receiving the message, U computes his/her attribute tokens as $\mathcal{AT}_U = \{AT^{(i_1)}_U, \ldots, AT^{(i_{c_U})}_U\}$, where $AT^{(i_j)}_U = x_U AT'^{(i_j)}_U$ and the set $\mathcal{BID}_U = \{bid^{(1)}_U, \ldots, bid^{(\ell)}_U\}$ as the blinded versions of his/her identities, where $bid^{(v)}_U = id^{b^{(v)}_U}_U$. U sends $\mathcal{BID}_U$ as an input to the mixnet $\mathcal{X}$.

Step 4. The output $\mathcal{X}(\mathcal{BID})$ is delivered to RA, where $\mathcal{BID}$ is the set of identities of at least two users. U stores the tuple $Tuple_U = \langle \mathcal{AT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U}, id_U, x_U \rangle$. Finally, RA sends a signed version of $\mathcal{X}(\mathcal{BID})$ to all remote servers.

6.1.2 Authentication phase

A user U can anonymously authenticate a message m to the RS using his/her tuple $Tuple_U$ using the following steps:

Step 1. U picks a random string r, generates the current timestamp TS_U and then computes $h = \mathcal{H}(TS_U, m, r)$ and $z = h^{x_U}$.

Step 2. U picks a new unused index $v \in \{1, \ldots, \ell\}$, and computes $b_U^{(v)} = \mathcal{H}(x_U, v)$, $bid_U^{(v)} = id_U^{b_U^{(v)}}$ and $e(G, G)^{r_U b_U^{(v)}}$. U prepares NIZK proof of knowledge $\Pi_{LogEq} \leftarrow P_{LogEq}(h, e(G, G)^{r_U b_U^{(v)}}, z, bid_U^{(v)}, x_U)$ and blinds each of his/her attribute token $AT_U^{(i)}$ as $BAT_U^{(i)} = b_U^{(v)} AT_U^{(i)}$. After that U prepares $\mathcal{BAT}_U = \{BAT_U^{(i_1)}, \ldots, BAT_U^{(i_{c_U})}\}$ and parses σ_U as $\langle r, z, bid_U^{(v)}, \Pi_{LogEq}\rangle$. U then sends the message $\langle m, \mathcal{BAT}_U, \mathcal{I}_U, k_U, e(G, G)^{r_U b_U^{(v)}}, \sigma_U, TS_U\rangle$ to the remote server RS.

Step 3. On the reception of the message $\langle m, \mathcal{BAT}_U, \mathcal{I}_U, k_U, e(G, G)^{r_U b_U^{(v)}}, \sigma_U, TS_U\rangle$ from U, RS first checks the validity of the received timestamp TS_U by the condition $|TS_U^* - TS_U| < \Delta T$, where ΔT is the maximum transmission delay and TS_U^* is the time when the message is received by the RS. If it is valid, RS parses σ_U as $(r, z, bid_U^{(v)}, \Pi_{LogEq})$ to ensure that $bid_U^{(v)} \in \mathcal{BID}$. RS then computes $h = \mathcal{H}(TS_U, m, r)$ and runs the verification algorithm $V_{LogEq}(h, e(G, G)^{r_U b_U^{(v)}}, z, bid_U^{(v)}, \Pi_{LogEq})$. RS rejects m and aborts it on failure of the verification.

Step 4. Using $\mathcal{I}_U$, RS determines the attributes intersection set $\mathcal{S}$ with U, which is $\mathcal{S} = \mathcal{A}_U \cap \mathcal{A}_S$. RS the session aborts if $|\mathcal{S}| < k_U$. After that RS picks any k_U tokens from $\mathcal{BAT}_U$ of the corresponding attributes found in $\mathcal{S}$. RS further computes $\Pi_{i\in\mathcal{S}} e(BAT_U^{(i)}, t_i G)^{\lambda_i}$ and checks if matches with $bid_U^{(v)}$, where λ_i is the Lagrange coefficient. If the equality does not hold, RS aborts the session. Otherwise, RS accepts m as a legitimate message (Fig. 7).

To verify correctness on the RS's side, we have

$$\begin{aligned}
\Pi_{i\in\mathcal{S}} e(BAT_U^{(i)}, t_i G)^{\lambda_i} &= \Pi_{i\in\mathcal{S}} e(b_U^{(v)}(AT_U^{(i)}), t_i G)^{\lambda_i} \\
&= \Pi_{i\in\mathcal{S}} e((b_U^{(v)} a_U(i)/t_i)G, t_i G)^{\lambda_i} \\
&= \Pi_{i\in\mathcal{S}} e(G, G)^{b_U^{(v)} a_U(i)\lambda_i} \\
&= e(G, G)^{b_U^{(v)} \Sigma_{i\in\mathcal{S}} a_U(i)\lambda_i} \\
&= e(G, G)^{b_U^{(v)} x_U r_U} \\
&= id_U^{b_U^{(v)}} \\
&= bid_U^{v}.
\end{aligned}$$

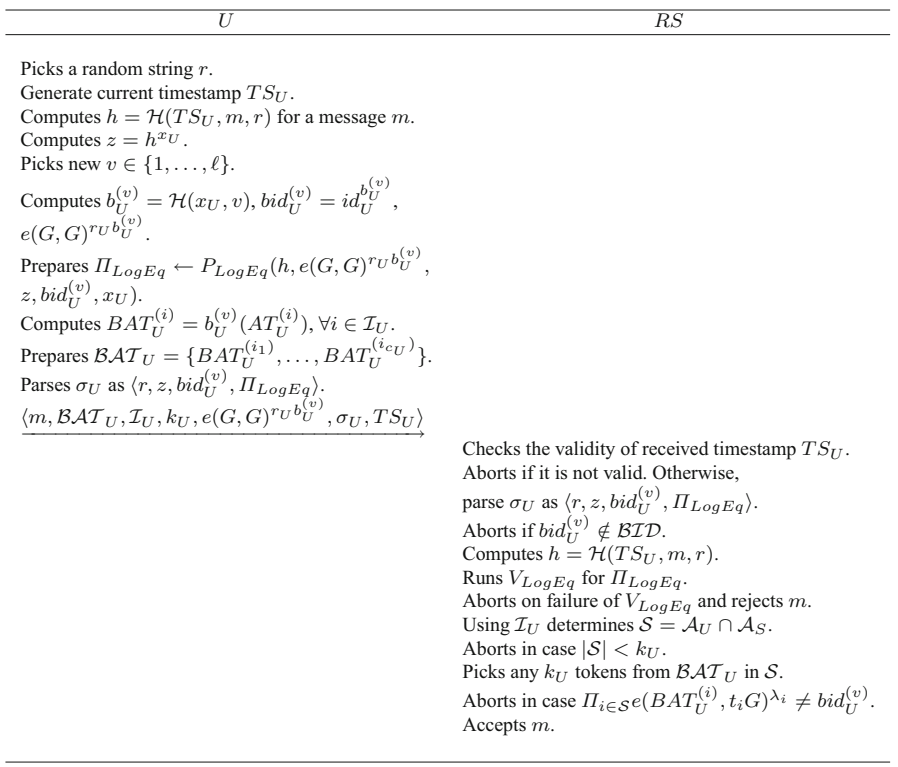

Fig. 7 The authentication phase of the extended scheme

6.1.3 Revocation service

Revocation of a user U is performed according to either of the following two cases:

> **Case 1:** RA wants to revoke a user with a clear identity $id_U^* \in \mathcal{RL}$, where $\mathcal{RL}$ is a revocation list.
> **Case 2:** A remote server RS wants RA to revoke a user with a blinded identity $bid_U^{(v)}$.

In either of the above two cases, once a user U's blinded identities are expired, U must approach RA for renewing his/her blinded identities. Before RA accepts to renew U's blinded identities, RA must perform a revocation check for U. To preserve anonymity and unlinkability, we emphasize that revocation must be done without allowing U to reveal his private key x_U.

Revocation (Case 1): The following steps are involved:

Step 1. U sends his/her clear identity id_U and a proof of knowledge of x_U to the base $e(G,G)^{r_U}$ as $\Pi_{Log} \leftarrow P_{Log}(e(G,G)^{r_U}, id_U, x_U)$.
Step 2. RA then needs to ensure that id_U is in her database for U. After that RA runs V_{Log} for Π_{Log}. RA aborts and revokes U if the verification fails.
Step 3. Finally, RA checks $\mathcal{RL}$. If $id_U \in \mathcal{RL}$, U is revoked.

Revocation (Case 2): In this case, a remote server RS wants to revoke a user with a blinded identity $bid_U^{(v)}$. RS first sends a revocation request to RA, and the transcript of the message and the signature $\langle m, r, z, bid_U^{(v)}, \Pi_{LogEq}\rangle$ he/she received from some malicious U. U approaches RA for renewing his/her blinded identities. The revocation check is performed as follows.

Step 1. U sends his/her clear identity id_U and a proof of knowledge of x_U to the base $e(G, G)^{r_U}$ as $\Pi_{Log} \leftarrow P_{Log}(e(G, G)^{r_U}, id_U, x_U)$.

Step 2. RA first ensures that id_U is in his/her database for U. After that RA runs V_{Log} for Π_{Log}. RA aborts and revokes U if the verification fails. RA then computes and sends $h = \mathcal{H}(TS_U, m, r)$ to U, where TS_U, m and r are in the transcript sent by RS for revocation.

Step 3. U computes h^{x_U} and a NIZK proof of discrete log equality $\Pi_{LogEq} \leftarrow P_{LogEq}(h, e(G, G)^{r_U}, z, id_U, x_U)$. U sends h^{x_U} and Π_{LogEq} to RA.

Step 4. RA runs the verification algorithm V_{LogEq} for Π_{LogEq}. If it fails, RA revokes U and aborts. Otherwise, RA checks if $z = h^{x_U}$, where z is in the transcript given by RS. If the equality holds, U is deemed to be the signer. RA revokes U and aborts. Otherwise, U is not the signer of m.

6.2 Analysis of the extended scheme

In this section, we first provide the formal security analysis under the standard model for our extended scheme only as it is the extended version of the basic scheme presented in Sect. 5.1. After that we also compare the performance of the extended scheme with the relevant existing schemes.

6.2.1 Formal security using random oracle model

The Abdalla et al.'s Real-Or-Random (ROR) model [47,48] is taken for our formal security analysis. There are three participants in the extended scheme, which are user U, RA and RS.

Participants: We have the oracles Π_U^t, Π_{RA}^u and Π_{RS}^v, which are the instances t, u and v of U, RA and RS, respectively.

Partnering. The partner of Π_U^t of U is Π_{RS}^v of RS and vice versa. We call Π_{RS}^v is the partner ID pid_U^t of Π_U^t. Note that the partial transcript of all communicated messages between U and RS is treated as the unique session identity sid_U^t for current session in which Π_U^t participates.

Freshness. We say Π_U^t or Π_{RS}^v is fresh, if attacker $\mathcal{A}$ does not know their secrets in each session.

Adversary. $\mathcal{A}$ can fully control all the communications in the network. Thus, $\mathcal{A}$ can read, modify all exchanged messages, fabricate new messages and inject them into the network. Furthermore, the following queries can be accessed by $\mathcal{A}$ [48]:

– $Execute(\Pi^t, \Pi^v)$: It is executed by $\mathcal{A}$ in order to obtain the messages exchanged between two honest participants. This is modeled as an eavesdropping attack.

- $Send(\Pi^t, m)$: It is queried by $\mathcal{A}$ to send a message, say m to a participant instance Π^t and also to receive a response message. This models as an active attack.
- $Test(\Pi^t)$: This query models the semantic security of the authenticated message following the indistinguishability in ROR [47]. A coin c is first flipped before the beginning of the experiment, and its value is only known to $\mathcal{A}$. This value is used to decide the output of the $Test$ query. If $\mathcal{A}$ executes this query and does not know the secrets in each session, Π^t returns 1 in case $c = 1$ or 0 when $c = 0$; otherwise, it outputs null ($\perp$).

Semantic security of the session key. $\mathcal{A}$ can query several $Test$ queries to either Π^t_U or Π^v_{RS}. The output of $Test$ must be consistent with respect to the random bit c. At the end, $\mathcal{A}$ returns a guessed bit c' and wins the game in case $c' = c$. If $Succ$ is an event that $\mathcal{A}$ can win the game, the advantage in breaking the semantic security of our extended scheme, say $\mathcal{S}$, is $Adv^{ake}_{\mathcal{S}} = |2.Pr[Succ] - 1|$. In the ROR sense, $\mathcal{S}$ is secure if $Adv^{ake}_{\mathcal{S}} \leq \psi$, for sufficiently small real number $\psi > 0$.

Random oracle. All the participants and $\mathcal{A}$ have access to a collision-resistant one-way cryptographic hash function $\mathcal{H}(\cdot)$. As in [48], $\mathcal{H}(\cdot)$ is modeled by a random oracle, say $HOracle$.

Theorem 1 *If $\mathcal{A}$ be an adversary running in polynomial time t against our extended scheme $\mathcal{S}$ in random oracle, then*

$$Adv^{ake}_{\mathcal{S}} \leq \frac{q_h^2}{|Hash|} + 2(Adv^{DLP}(t) + Adv^{BDHP}(t)),$$

where q_h, $|Hash|$, $Adv^{DLP}(t)$ and $Adv^{BDHP}(t)$ are the number of $HOracle$ queries, the range space of $\mathcal{H}(\cdot)$, the advantage of $\mathcal{A}$ in breaking the discrete logarithm problem (DLP), and the advantage of $\mathcal{A}$ in breaking the decisional bilinear Diffie–Hellman problem (BDHP), respectively.

Proof We follow the similar proof as presented in [48–51]. We have defined a sequence of four games, say $Game_i$, where $i = 0, 1, 2, 3$. We denote $Succ_i$ as an event wherein $\mathcal{A}$ can guess the bit c in the game $Game_i$ correctly.
$Game_0$: This game corresponds to the real attack by $\mathcal{A}$ against the extended scheme $\mathcal{S}$ in random oracle model. Since the bit c is selected before $Game_0$ starts, by definition, we have

$$Adv^{ake}_{\mathcal{S}} = |2.Pr[Succ_0] - 1|. \tag{1}$$

$Game_1$: This game simulates $\mathcal{A}$'s eavesdropping attacks by querying $Execute(\Pi^t, \Pi^v)$ oracle. At the end, $\mathcal{A}$ queries the $Test$ oracle. Suppose $\mathcal{A}$ eavesdrops the message $\langle m, \mathcal{BAT}_U, \mathcal{I}_U, k_U, e(G, G)^{r_U b_U^{(v)}}, \sigma_U, TS_U \rangle$. Due to the difficulty of solving intractable DLP and BDHP, $\mathcal{A}$ cannot obtain the secrets x_U, $b_U^{(v)}$ and r_U. Therefore, the chance of winning $Game_1$ for $\mathcal{A}$ is not increased by eavesdropping. Hence, we have

$$Pr[Succ_0] = Pr[Succ_1]. \tag{2}$$

$Game_2$: The game $Game_1$ is converted to $Game_2$ by adding the simulations of the $Send$ as well as $HOracle$ oracles. Note that $Game_2$ models as an active attack. The

aim of $\mathcal{A}$ in this game is to deceive a participant into accepting a modified message. $\mathcal{A}$ can make several $HOracle$ queries to verify the existence of the hash collisions. In the message $\langle m, \mathcal{BAT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U b_U^{(v)}}, \sigma_U, TS_U\rangle$ in each session, computation of h involves the random number r, the timestamp TS_U as well as the message m . Due to randomness, there is no collision if $\mathcal{A}$ queries $Send$ oracle. With the birthday paradox results, we have

$$|Pr[Succ_1] - Pr[Succ_2]| \leq \frac{q_h^2}{2|Hash|}. \tag{3}$$

$Game_3$: This game is translated from $Game_2$. $Game_3$ is modeled as an active attack wherein $\mathcal{A}$ can compromise the secrets of U in the message $\langle m, \mathcal{BAT}_U, \mathcal{I}_U, k_U, e(G,G)^{r_U b_U^{(v)}}, \sigma_U, TS_U\rangle$. Since $z = h^{x_U}$, due to difficulty of solving DLP, it is computationally infeasible for $\mathcal{A}$ to derive x_U. Without x_U, $\mathcal{A}$ cannot modify z. Furthermore, $\mathcal{A}$ cannot modify $e(G,G)^{r_U b_U^{(v)}}$ without knowing either r_U or $b_U^{(v)}$ due to the hardness of BDHP. Then, we have

$$|Pr[Succ_2] - Pr[Succ_3]| \leq Adv^{DLP}(t) + Adv^{BDHP}(t). \tag{4}$$

In the game $Game_3$, all the random oracles are simulated. Hence, $\mathcal{A}$ is only left with guessing the bit c for winning the game after querying the $Test$ oracle. Thus

$$Pr[Succ_3] = \frac{1}{2}. \tag{5}$$

From (1) and (2), we have

$$\begin{aligned} \frac{1}{2}.Adv_{\mathcal{S}}^{ake} &= |Pr[Succ_0] - \frac{1}{2}| \\ &= |Pr[Succ_1] - \frac{1}{2}|. \end{aligned} \tag{6}$$

Applying the triangular inequality, and (3) and (4), we have the following:

$$\begin{aligned} |Pr[Succ_1] - Pr[Succ_3]| &\leq |Pr[Succ_1] - Pr[Succ_2]| \\ &\quad + |Pr[Succ_2] - Pr[Succ_3]| \\ &\leq \frac{q_h^2}{2|Hash|} + (Adv^{DLP}(t) + Adv^{BDHP}(t)). \end{aligned}$$

Thus, using (5), we get

$$|Pr[Succ_1] - \frac{1}{2}| \leq \frac{q_h^2}{2|Hash|} + (Adv^{DLP}(t) + Adv^{BDHP}(t)). \tag{7}$$

Finally, using (6) and (7), we have the required result:

$$Adv_{\mathcal{S}}^{ake} \leq \frac{q_h^2}{|Hash|} + 2(Adv^{DLP}(t) + Adv^{BDHP}(t)).$$

6.2.2 Anonymity and unlinkability

This scheme is an extension of our basic non-anonymous scheme presented in Sect. 5.1. Therefore, the extended scheme is secure against the attacks explained in Sects. 5.2.1 and 6.2.1. Additionally, our extended scheme preserves anonymity and unlinkability due to the following reason. To realize unlinkability, U generates ℓ independent pseudonyms/blinded identities $\mathcal{BID}_U$ for himself/herself. Each pseudonym $bid_U^{(v)}$ is computed independently of any other pseudonym for the same user or any other user. In the registration phase, U computes his/her blinded identities using blinding parameters derived from his/her private key x_U as $b_U^{(v)} = \mathcal{H}(x_U, v)$ for indexes, $v = 1, \ldots, \ell$. Since RA does not know x_U, RA cannot compute the blinding parameters generated using (indexed) hashing of the unknown private key x_U. Therefore, RA is unable to compute the blinded identities $\mathcal{BID}_U$ and cannot link them to U's clear identity. Note that U delivers his/her blinded identities mixed by the mixnet $\mathcal{X}$. The blinded identities in $\mathcal{BID}$ of at least $h(\geq 2)$ users are securely mixed by $\mathcal{X}$ and the output $\mathcal{X}(\mathcal{BID})$ is delivered to RA. Since $\mathcal{X}$ is a secure mixnet, RA cannot link the received mixed blinded identities $\mathcal{X}(\mathcal{BID})$ of at least two users to any particular user.

6.2.3 Additional complexity

The additional computations required for the extended scheme over the basic non-anonymous scheme (Tables 4, 6) are as follows. In the registration phase, a user U requires to perform additional ℓ hash operations $\mathcal{H}(x_U, 1), \ldots, \mathcal{H}(x_U, \ell)$. U also requires to perform ℓ exponentiations to compute $\mathcal{BID}_U$. In the authentication phase, U needs to perform one hash operation to compute $b_U^{(v)}$, one modular exponentiation to compute $bid_U^{(v)}$ (note that $b_U^{(v)}$ and $bid_U^{(v)}$ are computed on demand; therefore, U is not required to store all his blinded identities) and c_U elliptic curve scalar multiplications to blind his/her tokens to generate $\mathcal{BAT}_U$. The storage requirement of U is the same as that for the basic non-anonymous scheme (Table 2). The RS requires to store additional $\ell|p|$ bits for each user instead of only $|p|$ bits in the basic non-anonymous scheme.

6.3 Performance comparison with related schemes

In this section, we compare our extended anonymous scheme with the well-known existing U-Prove scheme [4,38], Idemix scheme [5,6,38] and the recent HM12 scheme [43].

6.3.1 Comparisons of features, functionalities and services

In U-Prove scheme [4,38], though the authentication phase is interactive, a user is required to perform $|\mathcal{D}|$ exponentiations and multiplications, where $|\mathcal{D}|$ is the number of required attributes. This is in addition to the exponentiations required by the proofs of knowledge. Therefore, U-Prove requires at least $|\mathcal{D}|$ exponentiations, while our extended anonymous scheme requires only a fixed few number of exponentiations. Note that the number of exponentiations performed by a user in U-Prove is proportional to the size of the disclosed attributes. For a large attributes universe, these computations are unaffordable by thin clients. Moreover, U-Prove credential is revealed every time it is used. A U-Prove credential consists of a private key, a public key, a set of attributes, and a signature by the credential issuer. The signature is jointly computed by the issuer and the user agent in the course of a three-turn interactive protocol, called the issuance protocol, where the issuer sees the attributes but not the public key nor the signature itself. Therefore, the issue of a credential can be linked to show the credential only on the basis of the disclosed attribute information. It is worth mentioning that the system [3] is substantially different from U-Prove and it is not well suited for use on the Web since it requires the set of relying parties to be known at system setup time.

The Idemix [5,6,38] is based on the RSA assumption, where the magnitude of the modulus n, the private and public keys are in the order of 1024 bits, and p and q are each of 512 bits. These parameters are very large compared to elliptic curve private key of 160 bits for q and 512 bits for p in our basic and extended schemes. In addition to the large bit-length of the public/private parameters, like U-prove, the user must perform a number of exponentiations equal to all the disclosed attributes. Moreover, each exponentiation is performed over an 1024-bit composite modulus and 1024-bit exponent. The complexity of one exponentiation is much higher than that required by elliptic curves over prime fields (almost triple the time). As in U-Prove, the user must first know what attributes to disclose to the server and based on this information, he/she computes the proofs of knowledge. Therefore, the scheme cannot be made non-interactive. The important advantage of the Idemix over U-Prove is that the Idemix is unlinkable for the same set of $\mathcal{D}$. Also, the secret key of the user is fixed in size.

The revocation service is a serious problem in both U-prove and Idemix since neither of these schemes was built with the possibility to revoke malicious users or at least such service was not straight forward in these schemes. In U-prove, to make revocation possible the verifier must return to the issuer to ensure that the contacting prover is not revoked. Idemix on the other hand provides credential expiration. The HM12 scheme [43] focused on building an attribute-based authentication scheme with easier revocation service. However, the HM12 scheme requires an extra trusted entity called Revocation Referee (RR), which must be contacted every session to decide whether a user is revoked. The incorporation of the RR complicates the network architecture and communications among entities. Moreover, the computation and communication complexities of the HM12 scheme are high compared to the U-Prove scheme. In the ABE scheme and its variants, once a user is authenticated to the key generator and receives the decryption key with attributes, it is impossible to revoke the user and prevent him/her from accessing/decrypting files unless the files are re-encrypted

Table 7 Comparison of features, functionalities and services between our non-anonymous scheme and existing related schemes

	U-Prove	Idemix	HM12	ABE	Our scheme
Anonymous	Yes	Yes	Yes	No	Yes
Unlinkable	Partial	Yes	Yes	No	Yes
Threshold access	No	No	No	Yes	Yes
Non-interactive	No	No	No	Yes	Yes
Fault tolerant	No	No	No	Yes	Yes
Non repudiation	Yes	Yes	Yes	No	Yes
Smart card computations	Low	High	High	High	Low
Fixed size private key	No	Yes	No	No	Yes
Simple key management	No	Yes	No	No	Yes
Use secret credentials/tokens	Yes	No	Yes	Yes	No
User perform pairings	No	No	No	Yes	No
Vulnerable to smart card theft	Yes	Yes	Yes	Yes	No
Support access trees	No	No	No	Yes	Yes
Provide revocation service	No	No	Requires RR	No	Yes

with a new public key. Therefore, such existing schemes do not provide the revocation service. On the other hand, revocation in our proposed schemes is easy. The RA simply instructs the servers to remove the user's identity from their databases, and once the identity is removed, the user is simply revoked.

Finally, Table 7 shows a comparison of different functionalities, features and services between our proposed extended non-anonymous scheme and related existing schemes.

6.3.2 Comparison of computation costs

The suitability for thin clients, specially smart cards, is governed by the cryptographic operations performed using the master private key held by the user. Other computations on non-secret parameters or temporarily picked secret parameters can be offloaded to more computationally powerful devices (e.g mobile phones, tablets or PC's). Based on this assumption, in our schemes, the only computations performed by the user in the authentication phase using his/her private key x_U is the generation of the signature σ_U, which needs computation of $z = h^{x_U}$ and Π_{LogEq}.

We use the experimental results for different cryptographic primitives on different brands smart cards technologies [14]. For example, according to the experimental results reported in [14], on oberthur ID-One Cosmo v7.0-A, a modular exponentiation takes 186 ms, while Π_{LogEq} takes 529 ms. Therefore, the generation of σ_U in our schemes takes 715 ms. They also evaluated the computation costs for the U-Prove, Idemix and HM12 schemes. Based on those results, we have constructed Table 8 for comparison of the computation times of Idemix, U-Prove and our schemes.

Table 8 Comparison of computation time (in ms) on different brands of smart cards

	Idemix	U-Prove	HM12	Our schemes
Oberthur ID-One Cosmo V7.0-A	4519	837	2540	715
Gemalto TOP IM GX4	9433	1618	6016	1970
Gemalto .NET V2+	7270	1295	3312	811
MultOS ML2-80K-65	4219	827	2509	831
MultOS ML3-36K-R1	4208	633	1467	371

From Table 8, the U-Prove scheme computationally beats the Idemix and HM12 schemes on almost all smart card technologies. U-Prove is more efficient than our schemes on Gemalto TOP IM GX4 and MultOS ML2-80K-65. However, our schemes are computationally more efficient than U-Prove on all other brands of smart cards as shown in this table.

6.4 Discussions

6.4.1 Access control and access trees

In access tree constructions, a set of descriptive attributes are used to label the users. The servers are also assigned to tree access structures in which each non-leaf node of the tree represents a threshold gate while leaf nodes are associated with the attributes. For example, a tree with AND and OR gates can be represented using 2 of 2 and 1 of 2 threshold gates, respectively. If there is an assignment of attributes from the tokens to nodes of the tree such that the tree is satisfied, then only a user is able to authenticate to a server.

We review the access structure technique presented in [12], which can be straightforwardly applied to our authentication schemes. Let $\mathcal{T}$ be an access structure tree. Each non-leaf node of the tree represents a threshold gate and it is described by its children and a threshold value. Let n_y and $k_y (0 < k_y \leq n_y)$ be the number of children of a node and its threshold value, respectively. When $k_y = 1 (k_y = n_y)$, the threshold gate is an OR gate (an AND gate). Let $parent(y)$ denote the parent of the node y in the tree. The function $att(y)$ is defined only if y is a leaf node and denotes the attribute associated with the leaf node y in the tree. The ordering between the children of every node is defined by the access tree $\mathcal{T}$. The function $index(y)$ returns a number associated with the node y, where the index values are uniquely assigned to nodes in the access structure for a given key in an arbitrary manner. Let $\mathcal{T}$ be an access tree with root r. The subtree of $\mathcal{T}$ rooted at the node y is denoted by $\mathcal{T}_y$. Thus, $\mathcal{T}$ is the same as $\mathcal{T}_r$. If a set of attributes $\mathcal{AT}$ satisfies the access tree $\mathcal{T}_y$, $\mathcal{T}_y(\mathcal{AT})$ is recursively defined as follows. If y is a non-leaf node, evaluate $\mathcal{T}_{y^*}(\mathcal{AT})$ for all children y^* of node y. $\mathcal{T}_y(\mathcal{AT})$ returns 1 if and only if at least k_y children returns 1. If y is a leaf node, $\mathcal{T}_y(\mathcal{AT})$ returns 1 if and only if $att(y) \in \mathcal{AT}$.

In our schemes, for every node y in the tree the degree of the polynomial $a(y)$ is set to be $k_y - 1$ (one less than the threshold of that node). Now, set $a_r(0) = y$ and $k_r - 1$ other points of the polynomial a_r randomly to define it completely for the root node r. For any other node y, set $a_y(0) = a_{parent(y)}(index(y))$ and choose $k_y - 1$ other points randomly to completely define a_y. Finally, for each leaf node y the user is given the attribute token point $AT_U^{(i)} = (x_U r_U a_y(0)/t_i)G$.

6.4.2 Safeguarding RA's master keys

The secrecy of the master keys $(t_1, \ldots, t_{|\mathcal{AU}|})$ is very important for preserving the security of our schemes. One method to safeguard the master keys is to employ threshold cryptographic techniques, where the RA may apply a verifiable secret sharing scheme over elliptic curves [18] to share these master keys on a number of servers. The servers use the shares of the master keys as private inputs to jointly compute the public parameters and the user's attribute tokens on a threshold bases in a fault-tolerant manner.

7 Conclusion

In this paper, we have proposed two new authentication schemes for the cloud that support private attribute-based access to remote cloud servers by thin clients. We have first proposed a non-anonymous scheme as the basic scheme. We have then extended the non-anonymous scheme to realize anonymity and unlinkability services against channel observers, remote servers, registration authority and any collaboration of these entities. We have shown that the proposed schemes are secure and can withstand challenging adversarial attacks through both the rigorous formal and informal security analysis. In our proposed schemes, a user requires to hold one fixed-length secret key in the size of one group element in addition to a number of non-secret attributes tokens as points on an elliptic curve. The security and performance comparisons among our schemes and other related attribute-based schemes show that our schemes are computationally efficient and secure, and also provide additional services as compared to other existing schemes.

Acknowledgements The authors would like to thank the anonymous reviewers and the Editor for providing constructive and generous feedback.

References

1. Bethencourt J, Sahai A, Waters B (2007) Ciphertext-policy attribute-based encryption. In: IEEE Symposium on Security and Privacy (SP'07). Oakland, California, USA, pp 321–334
2. Boneh D, Franklin M (2001) Identity-based encryption from the Weil pairing. In: Advances in Cryptology-CRYPTO 2001. Santa Barbara, California, USA, pp 213–229
3. Brands S, Demuynck L, De Decker B (2007) A practical system for globally revoking the unlinkable pseudonyms of unknown users. In: Information Security and Privacy (ACISP'07). Townsville, Australia, pp 400–415
4. Brands SA (2000) Rethinking public key infrastructures and digital certificates: building in privacy. MIT Press, Cambridge

5. Camenisch J, Lysyanskaya A (2001) An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Advances in Cryptology (EUROCRYPT'01). Innsbruck (Tyrol), Austria, pp 93–118
6. Camenisch J et al (2010) Specification of the identity mixer cryptographic library. Technical report, Tech Rep
7. Cao X, Kou W, Du X (2010) A pairing-free identity-based authenticated key agreement protocol with minimal message exchanges. Information Sciences 180(15):2895–2903
8. Chaum D (1983) Blind signature system. In: Advances in cryptology (Crypto'83). Santa Barbara, California, USA, pp 153–153
9. Chaum D, Pedersen TP (1992) Wallet databases with observers. Advances in Cryptology (CRYPTO'92), Santa Barbara, California, USA, pp 89–105
10. Cohen H, Frey G, Avanzi R, Doche C, Lange T, Nguyen K, Vercauteren F (2005) Handbook of elliptic and hyperelliptic curve cryptography. CRC Press, Boca Raton
11. Gentry C, Silverberg A (2002) Hierarchical ID-based cryptography. In: 8th International Conference on the Theory and Application of Cryptology and Information Security, Advances in cryptology (ASIACRYPT'02). Queenstown, New Zealand, pp 548–566
12. Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: 13th ACM conference on Computer and Communications Security, Alexandria, VA, USA, pp 89–98. ACM
13. Guttman B, Roback EA (1995) An introduction to computer security: the NIST handbook. DIANE Publishing, USA
14. Hajny J, Malina L, Martinasek Z, Tethal O (2013) Performance evaluation of primitives for privacy-enhancing cryptography on current smart-cards and smart-phones. In: 8th International Workshop on Data Privacy Management and Autonomous Spontaneous Security (DPM/SETOP'13), Leuven, Belgium, Lecture Notes in Computer Science, vol 8247, pp 17–33
15. Huang JJ, Juang WS, Fan CI, Liaw HT (2013) Robust and privacy protection authentication in cloud computing. Int J Innov Comput Inf Control 9(11):4247–4261
16. Ibrahim MH (2009) Resisting traitors in linkable democratic group signatures. Int J Netw Secur 9(1):51–60
17. Ibrahim MH (2015) AATCT: anonymously authenticated transmission on the cloud with traceability. Int J Adv Comput Sci Appl 6(9):251–259
18. Ibrahim MH, Ali IA, Ibrahim II, El-sawi AH (2003) A robust threshold elliptic curve digital signature providing a new verifiable secret sharing scheme. In: 46th IEEE Midwest Symposium on Circuits and Systems, Cairo, Egypt, vol 1, pp 276–280
19. Khader D (2007) Attribute Based Group Signatures. IACR Cryptology ePrint Archive, p 159
20. Khader D (2008) Authenticating with Attributes. IACR Cryptology ePrint Archive
21. Khan AR (2012) Access control in cloud computing environment. ARPN J Eng Appl Sci 7(5):613–615
22. Kiyomoto S, Fukushima K, Tanaka T (2009) Design of anonymous attribute authentication mechanism. IEICE Trans Commun 92(4):1112–1118
23. Li J, Kim K (2008) Attribute-Based Ring Signatures. IACR Cryptology ePrint Archive, p 394
24. Lindell Y (2010) Anonymous authentication. J Priv Confid 2(2):35–63
25. Liu J, Wang J, Zhuang Y (2012) Fuzzy attribute authentication scheme based on vector space. J Comput Eng Appl 48(19):4–7
26. Lu S, Jiang H (2006) RTFW: An Access Control Model for Workflow Environment. In: 10th IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD'06). Southeast University, Nanjing, China, pp 1–5
27. Maji HK, Prabhakaran M, Rosulek M (2008) Attribute-Based Signatures: Achieving Attribute-Privacy and Collusion-Resistance. IACR Cryptology ePrint Archive, p 328
28. Nabeel M, Bertino E, Kantarcioglu M, Thuraisingham B (2011) Towards privacy preserving access control in the cloud. In: 7th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom'11). Orlando, Florida, USA, pp 172–180
29. Oh S, Park S (2003) Task-role-based access control model. Inf Syst 28(6):533–562
30. Pfleeger CP, Pfleeger SL (2002) Security in computing. Prentice Hall Professional Technical Reference
31. Raykova M, Zhao H, Bellovin SM (2012) Privacy enhanced access control for outsourced data sharing. In: 16th International Conference on Financial Cryptography and Data Security. Divi Flamingo Beach, Bonaire, pp 223–238

32. Rostad L, Edsberg O (2006) A study of access control requirements for healthcare systems based on audit trails from access logs. In: 22nd IEEE Annual Computer Security Applications Conference (ACSAC'06). Miami Beach, Florida, USA, pp 175–186
33. Sahai A, Waters B (2005) Fuzzy identity-based encryption. Advances in Cryptology (EUROCRYPT 2005). Aarhus, Denmark, pp 457–473
34. Sandhu RS, Coyne EJ, Feinstein HL, Youman CE (1996) Role-based access control models. IEEE Comput 29(2):38–47
35. Schnorr CP (1991) Efficient signature generation by smart cards. J Cryptol 4(3):161–174
36. Shamir A (1979) How to share a secret. Communications of the ACM 22(11):612–613
37. Shaniqng G, Yingpei Z (2008) Attribute-based signature scheme. In: 2nd IEEE International Conference on Information Security and Assurance (ISA'08). Hanwha Resort Haeundae, Busan, Korea, pp 509–511
38. Vullers P, Alpár G (2013) Efficient selective disclosure on smart cards using idemix. In: 3rd IFIP Working Conference on Policies and Research in Identity Management. Royal Holloway, UK, pp 53–67
39. Yang H, Oleshchuk V (2015) Attribute-based authentication schemes: a survey. Int J Comput 14(2):86–96
40. Yang K, Jia X (2012) Attributed-based access control for multi-authority systems in cloud storage. In: IEEE 32nd International Conference on Distributed Computing Systems (ICDCS). Macau, China, pp 536–545
41. Yang P, Cao Z, Dong X (2008) Fuzzy Identity Based Signature. IACR Cryptology ePrint Archive, p 2
42. Zhou M, Mu Y, Susilo W, Au MH, Yan J (2011) Privacy-preserved access control for cloud computing. In: IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'11). Changsha, China, pp 83–90
43. Jan H, Lukas M (2012) Unlinkable Attribute-Based Credentials with Practical Revocation on Smart-Cards. In: 11th International Conference on Smart Card Research and Advanced Applications (CARDIS 2012). Graz, Austria, pp 62–76
44. Wan Z, Liu JE, Deng RH (2012) HASBE: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing. IEEE Trans Inf Forensics Secur 7(2):743–754
45. Diaz C, Preneel B (2004) Taxonomy of mixes and dummy traffic. 19th IFIP International Information Security Conference, Toulouse, France, pp 217–232
46. Sampigethaya K, Poovendran R (2007) A survey on mix networks and their secure applications. Proceedings of the IEEE 94(12):2142–2181
47. Abdalla M, Fouque P, Pointcheval D (2005) Password-based authenticated key exchange in the three-party setting. In: 8th International Workshop on Theory and Practice in Public Key Cryptography (PKC'05). Les Diablerets, Switzerland, pp 65–84
48. Chang CC, Le HD (2016) A provably secure, efficient and flexible authentication scheme for ad hoc wireless sensor networks. IEEE Trans Wirel Commun 15(1):357–366
49. Das AK, Kumari S, Odelu V, Li X, Wu F, Huang X (2016) Provably secure user authentication and key agreement scheme for wireless sensor networks. Security and Communication Networks
50. Wazid M, Das AK, Kumari S, Li X, Wu F (2016) Design of an efficient and provably secure anonymity preserving three-factor user authentication and key agreement scheme for TMIS. Security and Communication Networks
51. Wazid M, Das AK, Kumari S, Li X, Wu F (2016) Provably secure biometric-based user authentication and key agreement scheme in cloud computing. Security and Communication Networks