Pervasive and Mobile Computing 59 (2019) 101081

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

Offline privacy preserving proxy re-encryption in mobile cloud computing

Voundi Koe Arthur Sandor, Yaping Lin *

College of Information Science and Engineering, Hunan University, Changsha 410082, China
Hunan Provincial Key Laboratory of Dependable Systems and Networks, Changsha 410082, China

ARTICLE INFO

Article history:
Received 18 March 2019
Received in revised form 11 June 2019
Accepted 16 August 2019
Available online 20 August 2019

MSC:
00-01
99-00

Keywords:
Authentication
Authorization
Ciphertext policy attribute-based encryption
Mobile cloud computing
Proxy re-encryption

ABSTRACT

This paper addresses the always online behavior of the data owner in proxy re-encryption schemes for re-encryption keys issuing. We extend and adapt multi-authority ciphertext policy attribute based encryption techniques to type-based proxy re-encryption to build our solution. As a result, user authentication and user authorization are moved to the cloud server which does not require further interaction with the data owner, data owner and data users identities are hidden from the cloud server, and re-encryption keys are only issued to legitimate users. An in depth analysis shows that our scheme is secure, flexible and efficient for mobile cloud computing

1. Introduction

Mobile cloud computing enables mobile devices to perform heavy resource-demanding tasks thanks to the availability of cloud-based resources through heterogeneous wireless networks. An exhaustive description of mobile cloud computing (MCC) architecture can be found in [1]. As more mobile devices are increasingly being used nowadays to store and process personal and corporate data [2], there are growing concerns regarding the privacy and confidentiality of sensitive data as the device can be stolen, compromised or hacked. Moreover, applications running on the mobile device should consume the least possible amount of energy. Therefore, any security solution designed for mobile cloud data storage should be mobile device resource-friendly.

The main drawback of outsourcing data to the cloud is that sensitive data can be accessed by a breached cloud service provider (CSP), as well as by some unauthorized users, leading to a confidentiality breach. Encryption has been proposed as a solution to secure data. However, only a limited number of operations can be performed on encrypted data, with one operation of interest being the search over encrypted cloud data as described in [3]. Requiring the user to download all the encrypted data locally before attempting decryption incurs many inconveniences among which, a high data transmission overload and, due to the cloud computing principle of pay-as-you-go, high financial expenditures. Such scheme additionally poses a problem of user authentication and authorization as arbitrary ciphertexts from different data owners can directly be accessed by any user without prior authorization mean. Attribute based encryption (ABE) has been introduced in [4] as a promising solution to address the issue of access authorization and proxy re-encryption (PRE) has been proposed in [5] to allow data owners to control who has access to the data stored on cloud. The proposed scheme in [5] was not however flexible enough , as users could access the whole data set in an all-or-nothing fashion. The data owner by grouping its data in categories and requiring access only to a subset of the whole dataset could then promote a more flexible scheme.

To allow the data owner to better manage access to the different subsets of his dataset, the type-based proxy re-encryption (TB-PRE) technique was proposed in [6]. The interesting work in [7], which exhibits overall good performances over other TB-PRE schemes in [6,8,9] and [10], successfully achieves data protection integrity as well as user authentication using Boney–Lynn–Shacham (BLS) signature and Merkle hash tree. However, data access control is performed using unhidden user identity and unmasked data type at the proxy level, leading to a non-anonymity of users as the cloud could learn the association between each user and the type of data requested. Furthermore the scheme in [7] is not flexible enough as the data owner has to be constantly online during ciphertext access. Such solution obviously incurs heavy computation and communication overhead on the data owner who needs to be always available even during idle times and can even become a system performance bottleneck.

There are three major technical challenges in dealing with proxy re-encryption schemes for mobile cloud computing. First, the scheme should ensure minimal computation and communication overhead on both data user and data owner sides. Second, the scheme should ensure the legitimate user has access to the right re-encryption key, while maintaining user anonymity and data privacy. Third, the scheme must provide chosen-ciphertext attack (CCA) security, meaning that if the adversary is not amongst the intended receivers made of the data owner and authorized data users, it should not be able to obtain any useful information about the plaintext even if it proceeds with chosen ciphertext attacks.

In this paper, we implement both privacy preserving cloud-user authentication and re-encryption key access control approaches by combining the efficient multi-authority ABE scheme in [11] which we modified to suit our purpose, and the scheme in [7]. We avoid weaker re-encryption schemes wherein the proxy possesses both parties' keys simultaneously [12], by following the work in [13] entrusting the data owner to generate re-encryption keys for a stronger user authorization. In our work, the data owner no longer needs to always be online and user identities privacy as well as category privacy are guaranteed through blind decryption.

Our contributions in this paper are given below.

- We propose an offline type-based proxy re-encryption with privacy preservation such that not only data owners and data users' identities are hidden from the cloud, but also the cloud cannot learn any useful information regarding the category of data stored or the category of data to be accessed.
- We improve the flexibility of user authorization and authentication procedures by no longer needing the data owner to be constantly online, by producing ahead of time all the required parameters for the authorized users.
- In depth analysis shows that our scheme is secure, flexible and induces minimal computation and storage overhead.

The rest of this paper is organized as follows. Section 2 reviews the related works. Section 3 introduces some preliminaries while Section 4 presents the system model, the security model as well as the design goals in MCC. In Section 5, we propose an offline privacy-preserving PRE. Section 6 addresses the security of our proxy re-encryption protocol, while in depth performance analysis is given in Section 7. Section 8 discusses the different findings, limitations and possible improvements of our scheme. We finally conclude in Section 9.

2. Related works

We review in this section two categories of work: attribute based encryption and proxy re-encryption

2.1. Attribute based encryption

Sahai and Waters introduced in [4] the first attribute-based encryption (ABE) which relies on identity based encryption (IBE) and on the concept of secret sharing. Their scheme does not however offer enough access control flexibility. For fine grained access control, two flavors of ABE were proposed: key-policy ABE (KPABE) in [14] which suffers from trust in the key issuing process to the legitimate user, and ciphertext-policy ABE (CPABE) in [15] solving the key issuing problem. We focus in this paper on the CPABE scheme although it suffers from performance bottlenecks due to the use of pairings and to the increase in the number of users, as well as from the key escrow issue depicted in [16], where an authority alone possesses enough abilities to decrypt users' messages. Chase proposed in [17] the first multi-authority ABE based on CPABE and solved the performance problem of previous schemes. However, the scheme still suffers from the key escrow issue where the central authority (CA) could decrypt every ciphertext. Later on, Chase and Chow proposed in [18] a solution which removes the trusted CA and prevents attribute authorities from pooling their information on particular users. The scheme however could not totally solve the key escrow issue. Our previous work in [11] follows the work [19] by removing the CA and putting trust upon the data owner for secret parameters generation in order to solve the key escrow problem. In this paper, we improve and adapt our previous work [11] into the type-based proxy re-encryption environment in order to build our solution.

2.2. Proxy re-encryption

Blaze et al. in 1998 introduced in [20] the first ElGamal-based transitive and non-collusion resistant bidirectional PRE scheme. Moreover, bidirectionality property is not very desirable in most real-world configurations. Ateniese et al. provided in [21] the first unidirectional PRE construction based on bilinear maps which is collusion resistant and non-transitive. However, their scheme only offers chosen plaintext security which is not sufficient for many practical applications. In 2007, Canetti and Hohenberger proposed a security definition against chosen ciphertext attacks (CCAs) for a PRE scheme and constructed an efficient bidirectional CCA secure PRE scheme relying on bilinear pairing. In the same year Chu and Tseng in [22] as well as Green and Ateniese in [23] proposed CCA secure unidirectional identity-based PRE schemes. However, such schemes provide an access control in an all-or-nothing fashion as stated in [9], which is not of interest in a desired fine grained access control environment. In 2008, Tang proposed in [6] a type based PRE (TB-PRE) scheme enabling a delegator to selectively delegate the decryption right to a delegatee through a chosen proxy. In 2009, a CCA secure conditional PRE was proposed by Weng et al. in both [24] and [25] to allow the ciphertext satisfying a condition set specified to be transformed by the proxy. For more flexibility, the scheme in [26] proposed a time-release proxy conditional re-encryption scheme in which a receiver cannot obtain any information about the file until a specified time arrives. [27] further introduced a conditional PRE called sender-specified PRE (SS-PRE) which enables the delegation of decryption right from a specified sender to his/her delegatee. As we focus on TB-PRE schemes, the work in [7] allows data protection integrity as well as user authentication using Boney–Lynn–Sasham (BLS) signature and Merkle hash tree. It however exposes users identities and accessed data categories and is not flexible enough as the data owner has to be constantly online during data request. This paper improves the TB-PRE scheme proposed in [7] and provides a new model for secure data distribution in proxy re-encryption for mobile cloud computing.

3. Preliminaries

In this section, we introduce some preliminaries we believe important for understanding the rest of the paper.

3.1. Bilinear map

Let G_0, G_1, G_T be multiplicative cyclic groups of prime order p. Let g be a generator of G_0, h a generator of G_1 and e, a bilinear map such that $e : G_0 \times G_1 \rightarrow G_T$. e has the properties listed as follows:

1. Bilinearity: for all $u, v \in G$ and $a, b \in Z_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$
2. Non degeneracy: $e(g, g) \neq 1$

G_0, G_1, G_T are said to be bilinear groups if the group operation in G_0, G_1 and G_T as well as the bilinear map $e : G_0 \times G_1 \rightarrow G_T$ are both efficiently computable.

3.2. Decisional Bilinear Diffie–Hellman assumption

The Decisional Bilinear Diffie–Hellman (DBDH) assumption in a bilinear group G_0 of prime order p with generator g is defined as follows: on input $g, g^a, g^b, g^c \in G_0$ and $e(g, g)^z \in G_T$, where $e : G_0 \times G_0 \rightarrow G_T$ is a bilinear map, and a, b, and $c \in Z_p$, no probabilistic polynomial time adversary can decide whether $e(g, g)^z = e(g, g)^{abc}$, that is deciding whether $z = abc$ or z is a random element $\neq abc$, with a non-negligible advantage. The assumption relies on the fact that the discrete logarithm is hard to be solved in large number field.

4. Problem statement

We consider in our system four main entities, namely, the data owner (DO), a set of attribute authorities (AA) logically playing the same role but managing disjoint sets of attributes, the cloud service provider (CSP) also considered as proxy server, and hosting the cloud user assistant (CUA) described in [11], and finally the data user (DU).

4.1. Core functionalities

In this subsection, we present the three core functionalities of our scheme: re-encryption key pre-generation, cloud based authentication and cloud based authorization. In our scheme, the data owner and all its users must share a secret hash function denoted as *Hash*, as in [28], to provide anonymity of ciphertexts re-encryption as in [29] as well as data category privacy preservation, so that the cloud operates through blind decryption. We examine the use of such hash function in the security analysis section of our paper. Re-encryption key pre-generation allows the various re-encryption keys associated each one with a legitimate user to be produced by the data owner for a specific category, prior to their secure outsourcing so that legitimate users can claim them directly from the cloud server while the data owner is offline. Re-encryption keys are uploaded to the cloud server in the form of data structures denoted as authorization structures depicted in Fig. 1. More details on the re-encryption key generation process are given in Fig. 2. Cloud based

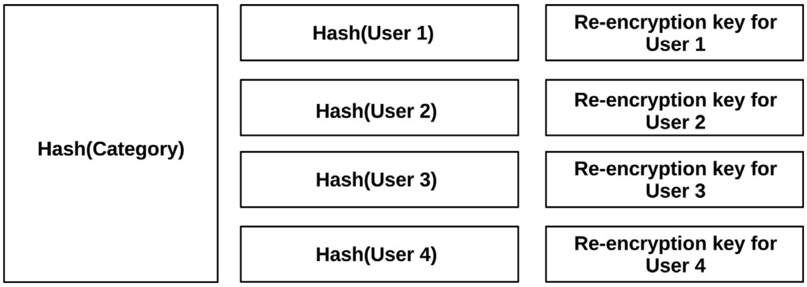
Hash(Category)
Hash(User 1)
Re-encryption key for User 1
Hash(User 2)
Re-encryption key for User 2
Hash(User 3)
Re-encryption key for User 3
Hash(User 4)
Re-encryption key for User 4

Fig. 1. Overview of an authorization structure.

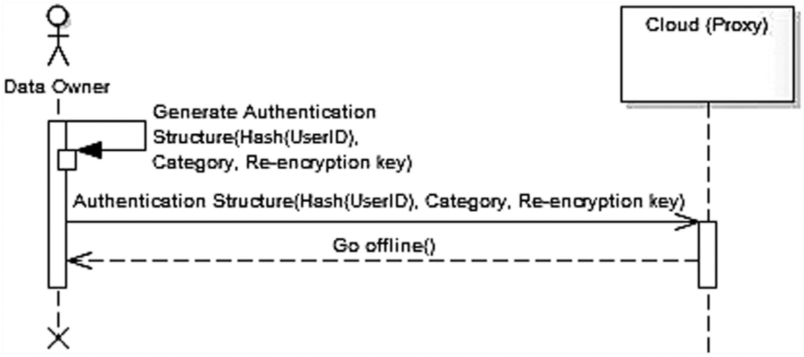
Data Owner
Cloud (Proxy)
Generate Authentication Structure(Hash(UserID), Category, Re-encryption key)
Authentication Structure(Hash(UserID), Category, Re-encryption key)
Go offline()

Fig. 2. Sequence diagram for generating an authorization structure.

authentication, our second core operation aims to verify whether a user is legitimate in accessing a given category so that the cloud can attempt to recompute the desired ciphertext belonging to that category. Cloud based authentication relies on multi-authority ABE operations for which the user will use its key obtained from both the data owner and the requested attribute authorities to decrypt a challenge ciphertext generated by the data owner and uploaded to the cloud. If the user is granted access to the given category, the cloud will search through the user authorization structure the corresponding re-encryption key to re-compute the user desired ciphertext. We provide more details on the cloud based authentication functionality in Fig. 3. Cloud based authorization, our third core functionality takes place immediately after successful user cloud based authentication. It relies on the fact that the DO can choose to generate or not re-encryption key for a given category. After successful cloud based authentication, the cloud will search the user authorization structure for the re-encryption key in order to recompute the desired ciphertext. An absence of re-encryption key means the user is not authorized to access the data even though such user has been previously authenticated. This brings an additional level of flexibility in user access control and we depict the process in more details in Fig. 4 below. All the other operations in our scheme are multi-authority ABE operations based on our previous work in [11] which complement our three above-mentioned core functionalities.

4.2. System model

Our system model is depicted in Fig. 5 and we briefly describe in the following lines the different actors in our scheme.

1. The data owner (DO) in this work stores a local list of authorized users for which it performs pre-computations allowing it to remain offline during further steps of the system operations. We assume in this paper that when a user is revoked, its record will be removed from the authorized users' list in the DO local storage.
2. The cloud service provider (CSP) or simply cloud server in this scheme plays the additional role of proxy server as in [22,23] and [7]. The cloud server performs user authentication and user authorization for every user requesting data access.
3. The attribute authorities (AA) in our paper issue attributes to the DO for realizing encryption policy. They moreover generate blinded attribute secret keys for every data user requesting encrypted data access.
4. The data user (DU) in the scheme computes its own authentication key which will be uploaded to the cloud for authentication's purpose. DU uploads as well its hashed identity to the cloud to perform user authorization.

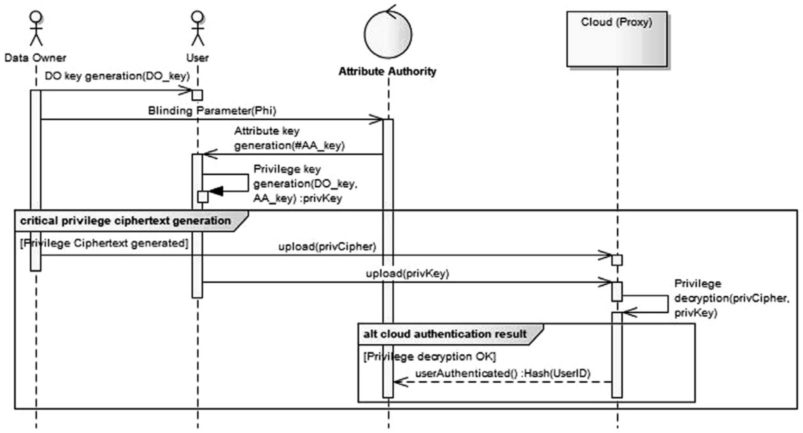
Data Owner
User
Attribute Authority
Cloud (Proxy)
DO key generation(DO_key)
Blinding Parameter(Phi)
Attribute key generation(#AA_key)
Privilege key generation(DO_key, AA_key) :privKey
critical privilege ciphertext generation
[Privilege Ciphertext generated]
upload(privCipher)
upload(privKey)
Privilege decryption(privCipher, privKey)
alt cloud authentication result
[Privilege decryption OK]
userAuthenticated() :Hash(UserID)

Fig. 3. Sequence diagram of our cloud based authentication process.

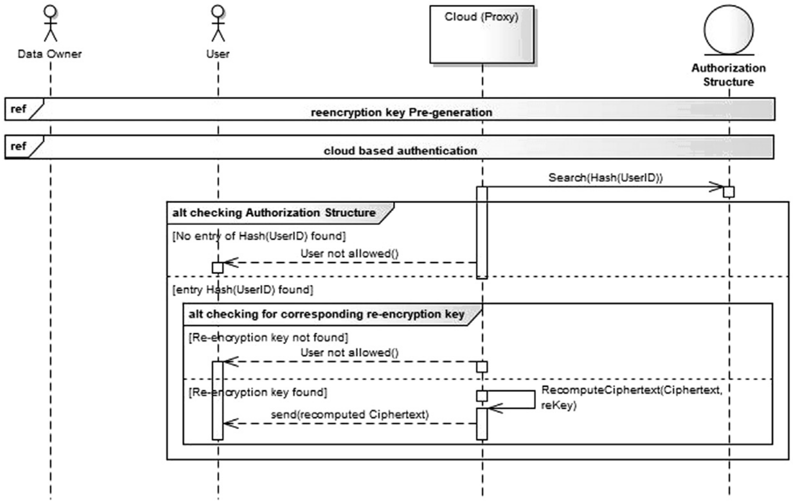
Data Owner
User
Cloud (Proxy)
Authorization Structure
ref
reencryption key Pre-generation
ref
cloud based authentication
Search(Hash(UserID))
alt checking Authorization Structure
[No entry of Hash(UserID) found]
User not allowed()
[entry Hash(UserID) found]
alt checking for corresponding re-encryption key
[Re-encryption key not found]
User not allowed()
[Re-encryption key found]
RecomputeCiphertext(Ciphertext, reKey)
send(recomputed Ciphertext)

Fig. 4. Sequence diagram of our cloud based authorization process.

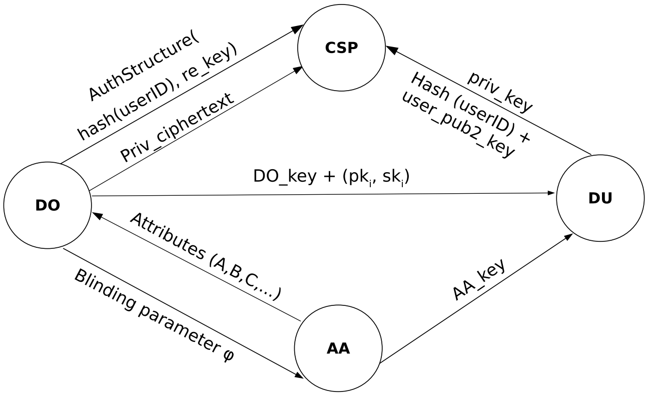
CSP
DO
DU
AA
AuthStructure(hash(userID), re_key)
Priv_ciphertext
priv_key
Hash (userID) + user_pub2_key
DO_key + (pk$_i$, sk$_i$)
Attributes (A,B,C,...)
Blinding parameter φ
AA_key

Fig. 5. System model overview of our proxy re-encryption system.

4.3. Security model

In our work, the data owner (DO) is considered a fully trusted entity, a claim which goes in line with the work of [13] in which the authors entrust the encryptor to produce re-encryption keys *reKey* for the different users in order to control the re-encryption key issuing to the legitimate user. The cloud server in our paper is considered *honest-but-curious*, meaning it will execute all the operations under its responsibility but might want to get more insights into the user authentication and authorization operations, as well as into the decryption process. The attribute authorities (AA) are as well as the cloud server, considered *honest-but-curious* meaning they will follow the desired protocol but may want to collude with data users to gain additional privileges in the user authentication and authorization processes taking place in the cloud server. Last but not least, the data user is considered untrusted in our paper as it is willing to collude with other users in order to get more privilege than granted, and to access data from a particular data owner. We furthermore prove our scheme to be indistinguishable against chosen-ciphertext attack (IND-CCA) under the DBDH assumption in the random oracle model. We rely on the security model definitions of a TB-PRE scheme in [6] and [8].

5. Cloud-based multi-authority ABE access control and offline proxy re-encryption

Our offline multi-authority ABE privacy-preserving PRE for mobile cloud computing consists of two fundamental operations realizing our three core functionalities, namely data owner key generation producing re-encryption keys, and cloud based authentication coupled with cloud based authorization. These two operations are complemented by six operations based on our previous work in [11] which are qualified as privilege operations, with the different parameters produced at each stage of the six operations, at the exception of the setup operation, bearing the name of privilege parameters. Let k be the initial security parameter and $l = 2k$. Let G_1, G_2 and G_T be bilinear groups of prime order p, and let g be a generator of G_1 and g_p be a generator of G_2. Let $e : G_1 \times G_2 \rightarrow G_T$ denote the bilinear map, we therefore assume $e(g, g_p)$ is the generator of G_T. Additionally, let $H_0 : \{0, 1\}^* \rightarrow Z_p^*$, $H_1 : \{0, 1\}^l \rightarrow Z_p$, $H_2 : G_T \rightarrow \{0, 1\}^l$ and $H_3 : G_1 \times \{0, 1\}^* \rightarrow G_1$ be families of hash functions. The eight different operations in our scheme are as follow:

- **Setup**(1^k): The setup operation produces the system public parameters and master secret key. It takes an implicit input security parameter depicted as 1^k and further chooses four random exponents α, $\beta \in Z_p$, $g \in G_1$, $g_p \in G_2$, and computes $e(g, g_p)^\alpha$, $e(g, g_p)$, g^α and $g^{1/\beta}$. Finally, the setup operation uniformly chooses $h_0 \leftarrow H_0$, $h_1 \leftarrow H_1$, $h_2 \leftarrow H_2$ and $h_3 \leftarrow H_3$ at random. The message space is set to $P = \{0, 1\}^k$ and the type space is set to $T = \{0, 1\}^*$. It publishes the public parameters as $PP = \{g, g_p, h = g^\beta, f = g^{1/\beta}, Y = e(g, g_p)^\alpha, Z = e(g, g_p), h_0, h_1, h_2, h_3, P, T\}$ and the master secret key as $MSK = \{\beta, g_p^\alpha\}$.
- **KeygenDO**$(PP, MSK, userID, categoryList)$: The data owner privilege key generation algorithm produces three kinds of user keys sent via secure channel to the data user i: DO privilege key *privKey*, public and private keys respectively pk_i and sk_i. *privKey* and pk_i are considered public while sk_i should remain secret. The algorithm takes as input PP, MSK, and the plain user identity *userID*, further hashed using the secret hash function *Hash*. The parameter *categoryList* denotes the list of categories to which a particular user has access authorization, and is used to produce the re-encryption keys $rk_{i \longrightarrow j, Hash(dataType)}$ for the user *userID*. The variable *dataType* can be obtained by iterating on *categoryList*. To produce the user i private key sk_i and the user i's public key pk_i, the DO uniformly chooses two random exponents α_i, $\beta_i \in Z_p$, and publishes $pk_i = \{g^{\alpha_i}, g_p^{\beta_i}\}$ and $sk_i = \{\alpha_i, \beta_i\}$. Furthermore to compute the DO privilege key, the DO chooses two random $r, \gamma \in Z_p$ and publishes part of the user privilege key as $DO\_key = (g^{(\alpha+r)/\beta}, g^\gamma, pk_i, sk_i)$. DO generates as well $\phi = g^{(r+\gamma)}$ that it sends to all requested AA via secure channel for attribute secret keys generation. The algorithm further computes the re-encryption key given the ith DO and the jth data user such that $reKey_{sk_i \longrightarrow pk_j, Hash(dataType)} = pk_{j,2}^{\frac{1}{H_0(Hash(dataType)+sk_{i,1})}}$, where $pk_{j,2} = g_p^{\beta_j}$, and $sk_{i,1} = \alpha_i$. The different re-encryption keys $rk_{i \longrightarrow j, Hash(dataType)}$ corresponding to the different users for a given category $Hash(dataType)$, will be aggregated into an authorization structure *authStruct* and uploaded to the cloud for user authorization purpose.
- **Encrypt**$(PP, m, T, pk_i, dataType)$: The data encryption stage is subdivided into two operations: the generation of the privilege ciphertext and the encryption of the plain data using the type-based proxy re-encryption (TB-PRE) approach as globally described in [7]. The encrypt operation takes as input the DO public key pk_i, a message M such that $M \in P$. It further takes a category *dataType*, computes $t' = Hash(dataType)$, chooses a uniform and random $p \leftarrow \{0, 1\}^k$ and computes: $v \leftarrow H_1(M \parallel p)$, $c_0 = (g^{H_0(t')} pk_{i,1})^v$, $c_1 = H_2(Y^r) \oplus (M \parallel p)$ and $c_2 = H_3(c_0, c_1)^v$. Finally, the encrypt operation returns the ciphertext $C_i = (Hash(dataType), c_0, c_1, c_2)$.
- **KeygenAA**(PP, ϕ, S): The attribute authority privilege key generation depicted in [11] produces the attribute authority privilege key *AA_key* given a set of attributes S which is further sent to the user via secure channel.
- **KeygenAggregate**$(PP, \theta, (\forall j \in N, AA_j\_key))$: The attribute authority privilege key aggregation described in our previous work in [11] relies on the cloud user assistant (CUA) to alleviate computation and communication overhead on the data user by combining the result of the *keygenAA* operation performed by each solicited AA.
- **PrivKeygenUser**(PP, DO_key, CUA_key): Detailed in our previous work in [11], the privilege data user key generation algorithm performed by the data user aims to produce the user privilege key *privKey*.

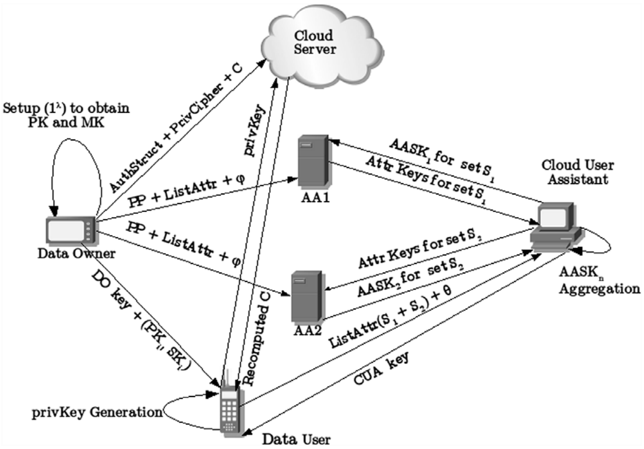

Fig. 6. Information flow among the entities in our proxy re-encryption system.

- **CloudCheck**(PP, $privKey$, $Hash(userID)$, $privCipher$, C_i, $authStruct$, pk_j): This algorithm is operated by the cloud service provider in two important operations: data user authentication followed by data user authorization. The data user authentication relies on the multi-authority decryption operation in [11] and takes as input $privKey$ and $privCipher$. if the decryption outputs $Hash(dataType)$, the user has access the category $dataType$. The decryption fails by giving as output $\perp$. The data user authorization process takes as inputs $Hash(userID)$ and $authStruct$. If the cloud cannot find $authStruct$ corresponding to $Hash(userID)$ the data authorization outputs $\perp$. If successful, the cloud will search the correspondence of $Hash(userID)$ in $authStruct$ and will output the corresponding $reKey$ or $\perp$ if nothing is found. Given a re-encryption key $reKey = rk_{sk_i \longrightarrow pk_j, Hash(dataType)}$ and the ciphertext $C_i = (Hash(dataType), c_0, c_1, c_2)$, the algorithm checks whether $e(c_0, H_3(c_0, c_1, Hash(dataType))) = e(g^{H_0(Hash(dataType))} pk_{i,1}, c_2)$ holds. If it does not hold, the algorithm returns $\perp$ and reports failure, else the algorithm computes $c'_0 = e(c_0, rk_{sk_i \longrightarrow pk_j, Hash(dataType)}) = e(g, pk_{j,2})^{v}$ and returns the re-encrypted ciphertext $C_j = (c'_0, c_1)$.
- **Decrypt**(sk_j, C_i). The decrypt algorithm performed by the data user takes as input the user private key sk_j expressed as $sk_i = \{\alpha_i, \beta_i\}$ and the ciphertext C_i while distinguishing two cases. In the first case the ciphertext is an original ciphertext and the algorithm parses $C_i = (Hash(dataType), c_0, c_1, c_2)$. It further checks if $e(c_0, H_3(c_0, c_1, Hash(dataType))) = e(g^{H_0(Hash(dataType))} pk_{i,1}, c_2)$ holds, and return $\perp$ if not. If the equation holds, the algorithm computes $K = e(c_0, h^{\frac{1}{H_0(Hash(dataType)) + sk_{i,1}}})$ and $M \parallel p = c_1 \oplus H_2(K)$. Finally, the algorithm returns M if $c_2 = H_3(c_0, c_1)^{H_1(M \| p)}$ else it returns $\perp$.
  In the second case, the ciphertext C_i is a re-encrypted ciphertext and the algorithm parses $C_i = (c_0, c_1)$. It further computes $K = c_0^{\frac{1}{sk_{i,2}}}$ and $M \parallel p = c_1 \oplus H_2(K)$. The algorithm finally returns M if $c_0 = e(g, pk_{j,2})^{H_1(M \| p)}$, else it returns $\perp$.

We give high level details on the flow of information among entities within our scheme, in Fig. 6.

6. Security analysis

We address in this section the security analysis of our offline scheme.

6.1. Key-escrow issue

We follow the work in [13] and entrust the DO to produce system wide public and master keys, data user public and private keys as well as parts of the user privilege key. As the generation of the privilege key requires both the DO and attribute authorities (AA), no single entity in our scheme possesses enough privileges to generate all the parameters on itself. Our scheme is thus key-escrow free.

6.2. Secret hash function

We are aware that our scheme seems to violate the Kerckhoffs principle as described in [30] stating that a cryptosystem should only have its key as secret. We however argue on the importance of such function in our scheme. The used secret hash function in our work allows the cloud to operate without learning any sensitive information. Sharing a secret hash function is like sharing a secret key because data owners and authorized users can both construct a keyed hash function

Table 1
Useful variables and associated description.

Variable	Description
N_{AA}	Number of requested attribute authorities
N_u	Number of users under management of each DO
N_t	Number of categories
N_c	Number of conditions

Table 2
Storage overhead in bytes.

	[7]	[26]	[27]	Our scheme
Data owner	$385 + 100 \times N_t$	$152 + 100 \times N_c$	152	$1699 + 4 \times N_u$
Data user	385	152	152	322

such as HMAC depicted in [31], with the hash algorithm being public knowledge and the key being the only secret in accordance with the Kerckhoffs principle. Furthermore, it is a common security measure to seal security computations in hardware entities generally referred as tokens. This is in fact a better security practice than to directly handle the secret keys and secret hash function to users and to ask them to configure their computing devices.

6.3. Data user revocation

In our scheme, we adopt the lazy access revocation method such that the DO can specify an attribute timestamp during the generation of the privilege ciphertext. Access to the specific category of data will be allowed within a time frame specified by the attribute timestamp. Data users are able to access the category before the expiration of the timestamp, and the DO can grant access to the non-revoked users by issuing a new privilege ciphertext for the category without recomputing the re-encryption keys or the TB-PRE ciphertext.

6.4. IND-CCA security under DBDH assumption in the random oracle model

Security in TB-PRE schemes requires the master key security property, meaning that a proxy with re-encryption keys and malicious data users without permissions or with access permission to data type t, must not be able to collude in order to access data for which they do not have access. We prove that our construction is secure against chosen ciphertext attacks (CCA) under the decisional bilinear Diffie–Hellman assumption (DBDH) in the random oracle model (See Appendix).

7. Performance analysis

We address in this section the various experiments to show the efficiency, efficacy and flexibility of our scheme. We compare our scheme with the works in [7,26] and [27]. As the original work in [7], we let the security parameter $k = 128$ by using the Barreto–Naehrig (BN) curve defined in [32] over F_p256. The group elements in G_0, G_1, G_T and Z_p can be represented respectively in 128 bytes, 33 bytes, 384 bytes and 32 bytes. We further assume that each user identity is expressed over 4 bytes and that the length of the category names is at most 64 bytes. We use SHA-256 as our secret cryptographic hash function and our code moreover derives from the CP-ABE toolkit in [33] and the pairing-based cryptography (PBC), in [34], version 0.5.12 with type F curve. We give in Table 1, some notations to be used throughout this section.

7.1. Storage

As in [7], the DO stores a users' list where each entry has 4 bytes, the public parameters weight 1409 bytes, and the master secret key 65 bytes. The user public key weights 161 bytes and the user private key 64 bytes. Each data user privilege key in our scheme has 322 bytes. We assume in [26] that the DO stores a list L of different conditions and each condition entry in L needs 100 bytes. We further give in Table 2, comparison results.

7.2. Computation

We analyze here the computations performed on both the data owner and the data user sides.

1. **Multiplication, Exponentiation and Pairing Cost**
   Let C_m denote the cost of a multiplication in G_1 and G_2, C_e the cost of an exponentiation in $G1$ and $G2$ and C_p the cost of a pairing in G_T. We give the computation comparison between our scheme and the works in [7,26] and [27], in Table 3.

Table 3
Computation cost.

textbf	[7]	[26]	[27]	Our scheme
Setup	$2(C_m + C_e + C_p)$	$3C_m + C_e + C_p$	$2C_p$	$C_m + 5(C_e + C_p)$
Key issuing	$2C_e$	C_e	C_e	$C_m + 8C_e$
Encryption	$2C_m + 3C_e$	$2C_m + 4C_e + 2C_p$	$4C_m + 6C_e + C_p$	$2C_m + 5C_e + C_p$
Decryption	$C_m + 3C_e + 4C_p$	$3C_m + 2C_e + 4C_p$	$C_m + 4C_p$	$2C_e + 4C_p$

Table 4
Encryption and decryption speed.

	[7]	[26]	[27]	Our scheme
Encryption	$t_p + 2t_e$	$t_p + 5t_e$	$t_p + 4t_e$	$t_p + t_{me} + 5t_e$
Decryption	$5t_p + 2t_{me} + 2t_e$	$3t_p + 2t_e$	$6t_p$	$2t_p + 2t_{me} + 2t_e$

2. **Encryption and Decryption time**
   We denote t_p, t_e, and t_{me}, as being respectively the time for computing a bilinear pairing, the time for operating an exponentiation and the time for computing a multi-exponentiation in the bilinear group. Table 4 gives a comparison between our scheme and the works in [7,26] and [27] in terms of speed of encryption and decryption.

8. Discussion

Our scheme successfully achieves a privacy preserving proxy re-encryption protocol keeping the data owner offline after the key generation stage. Although our scheme exhibits more computation and more storage overhead on the DO, its benefits however are numerous. Our scheme allows the early generation of all the necessary parameters so that the DO does not intervene in subsequent data access control, rendering our scheme more flexible than the schemes in [7,26] and [27]. Our scheme is further more mobile-friendly, as it provides overall less computation cost and storage overhead on the data user side, as well as more security over the works in [7,26] and [27] relying on trusted third party entities and exposing sensitive data.

9. Conclusion

In this paper, we address the issue of having the data owner being constantly online in order to issue re-encryption keys to authorized users in a type based proxy re-encryption configuration, and in a mobile cloud environment. Our scheme relies on the cloud server to authenticate every data user and check that it is authorized to access a specific data from a given category. To furthermore keep user anonymity and data privacy, we use a secret hash function to be shared only between a data owner and its users with the cloud having no knowledge of it. We finally show that our scheme, while increasing the payload on data owner's side for flexibility, greatly reduces the computation and storage overhead on the data user side, and can be considered as a mobile device-friendly protocol.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Voundi Koe Arthur Sandor: Conceptualization, Data Curation, Formal analysis, investigation, Methodology, Software, Visualization, writing-original draft, writing-review & editing. **Yaping Lin:** Funding Acquisition, Project Administration, Resources, Validation.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (Project No. 61872131).

Appendix. Security model and security proof of our offline proxy re-encryption scheme

A.1. Security model

We first mention in this section the security model of a type-based proxy re-encryption scheme (TB-PRE) depicted in [25] and [35]. We consider the following oracles which model the ability of an adversary as well as which are provided to the adversary A by a challenger C simulating an environment running TB-PRE:

- Uncorrupted key generation oracle $O_{ukg}(i, t, pk_i, pk_j)$: Given user identity i and category $t \in T$, The challenger C chooses a security parameter k and runs the algorithm KeygenDO(i, t, pk_i, pk_j) to generate a key pair (pk_i, sk_i). C returns pk_i to A and inputs (pk_i, sk_i) in Table T_k. Furthermore, given two public keys pk_i, pk_j such that $(pk_i, pk_j) \in T_k$, and a type t, C runs the KeygenDO sub-algorithm ReKeyGen(pk_i, pk_j, t) and returns the re-encryption key $rk_{i \leftarrow j,t} \leftarrow$ ReKeyGen(sk_i, pk_j, t) such that the re-encryption key is sealed into an authorization structure AuthStruct$^{(t)} \leftarrow (Hash(t), rk_{i \leftarrow j,t})$, where sk_i is the secret key corresponding to the public key pk_i. If pk_i or pk_j are not in T_k, C returns $\perp$ for $rk_{i \leftarrow j,t}$ We have $(pk_i, rk_{i \leftarrow j,t}) \leftarrow KeygenDO(i, t, pk_i, pk_j)$.
- Corrupted key generation oracle $O_{ckg}(i, t)$: Given user identity i and category $t \in T$, The challenger C chooses a security parameter k and runs the algorithm KeygenDO (i, t, pk_i, pk_j) to generate a key pair (pk_i, sk_i). C returns the key pair (pk_i, sk_i) to A and inputs (pk_i, sk_i) in Table T_k. Furthermore, given two public keys pk_i, pk_j such that $(pk_i, pk_j) \in T_k$, and a type t, C runs the KeygenDO sub-algorithm ReKeyGen(pk_i, pk_j, t) and returns the re-encryption key $rk_{i \leftarrow j,t} \leftarrow$ ReKeyGen(sk_i, pk_j, t) such that the re-encryption key is sealed into an authorization structure AuthStruct$^{(t)} \leftarrow (Hash(t), rk_{i \leftarrow j,t})$, where sk_i is the secret key corresponding to the public key pk_i. If (pk_i, sk_i) or (pk_j, sk_j) are not in T_k, C returns $\perp$ for $rk_{i \leftarrow j,t}$. We have $(pk_i, sk_i, rk_{i \leftarrow j,t}) \leftarrow KeygenDO(i, t, pk_i, pk_j)$.
- Ciphertext re-encryption oracle $O_{cro}(C_i, pk_i, pk_j, t)$: Given two public keys pk_i and pk_j, a type t and the original ciphertext C_i, C runs the algorithm CloudCheck$(i, C_i, pk_j, t, ReKeyGen(sk_i, pk_j, t))$, and outputs the re-encrypted ciphertext C_j that it returns to A, where sk_i is the secret key corresponding to the public key pk_i.
- Decryption oracle $O_{dec}(pk_i, C_i, t)$: Given a public key pk_i, a type t and a ciphertext C_i, the challenger C returns the plaintext $m \leftarrow Decrypt(sk_j, C_i, t)$, where sk_j is the secret key corresponding to the public key pk_j.

In our scheme, we work in the static corruption model where the adversary decides the corrupted users before the game starts. A public key is good if output by O_{ukg}, meaning the user is legitimate and bad if output by O_{ckg}, meaning the user is malicious or the user private key is known to or has been corrupted by the adversary. Furthermore, as the work in [7], we only consider in this work ciphertexts supporting re-encryption because our scheme does not generate ciphertexts which cannot be re-encrypted, which leads us to focus only on the security of original ciphertext also known as second level ciphertext. We give below the definition of the semantic security for our unidirectional single hop TB-PRE Π_x scheme under chosen ciphertext attacks (CCA). Our scheme security is further defined on the decisional bilinear Diffie–Hellman (DBDH) assumption.

A.1.1. Definition (CCA-security for Our TB-PRE)

A TB-PRE scheme is semantically secure against an adaptively chosen ciphertext attack according to the work in [36] if for any given polynomial time TB-PRE-CCA adversary$A$, Adv$^{CCA-A}_{TB-PRE}(k)$ is negligible.

A.1.2. Complexity assumptions

Our scheme security is based on the decisional bilinear Diffie–Hellman (DBDH) assumption. We first define the DBDH problem below.

Let $(p, g, G, G_T, e) \leftarrow BSetup(1^k)$. The DBDH problem is defined as follows: Given (g, g^a, g^b, g^c, T) for $a, b, c \in Z_p^*$ and $T \in G_T$, decide if $T = e(g, g)^{abc}$. An algorithm A has advantage ϵ in solving DBDH problem if $|Pr[A(g, g^a, g^b, g^d, g^{\frac{1}{d}}, g^{bc}, g^{dc}, e(g, g)^{ac}) = 0] - Pr[A(g, g^a, g^b, g^d, g^{\frac{1}{d}}, g^{bc}, g^{dc}~, T) = 0]| \leq \epsilon$, with the probability over the random choice of $a, b, c \in Z_p^*$, random choice of T in G_T, random choice of $g \in G^*$ and finally, on the random bits of A.

A.2. Security analysis

We first denote in this subsection the following interesting lemma.

Lemma 1. *For the events E_1, E_2 and G defined on some probability space, we consider that the event $S1 \bigvee \neg G$ occurs if and only if $S2 \bigvee \neg G$ occurs. In other words, $|Pr(E_1) - Pr(E_2)| \leq Pr(G)$. The security of our offline TB-PRE is summarized in the following theorem.*

Theorem 1 (*TB-PRE-CCA Security*)**.** *Our scheme is TB-PRE-CCA secure in the random oracle model if solving the DBDH problem is hard.*

Proof. We define an incremental sequence of games beginning by the real attack game denoted as Game G_0 up to Game G_{12}, clearly showing that the adversary A cannot break the scheme. Let E_i be the event that $b = b'$ in Game G_i, with b being the bit involved in the challenge phase and b' the output of A in the guess phase. We have:

- Game G_0 relates to the real attack and therefore $|Pr[E_0] - 1/2|= \text{Adv}^{CCA-A}_{TB-PRE}(k)$.
- In Game G_1, all the hash functions are replaced by random oracles. As we work in the random oracle model, we have $Pr[E_1] = Pr[E_0]$.
- In Game G_2, we change O_{h0} and the challenge phase by guessing the target message type t^*. The probability of guessing the right t^* is at least $\frac{1}{q_{h0}}$, with q_{h0} being the maximum number of queries to O_{h0}, therefore having $Pr[E_2] \geq \frac{1}{q_{h0}} Pr[E_1]$.
- In Game G_3, we change O_{h1} and the challenge phase by defining $h^* = h$, where $h \overset{R}{\leftarrow} Z_p^*$. Game G_3 and Game $G2$ are indistinguishable if the adversary never queries O_{h1} with $m_b \parallel h^*$. Therefore, $|Pr[E_3] - Pr[E_2]| \leq \frac{q_{h1}}{p}$, where q_{h1} is the maximum number of queries to O_{h1}.
- In Game G_4, we modify our TB-PRE decryption oracle concerning the following case. The adversary computes $c_2 = H_3(c_0, c_1)^v$ without knowing v. We have $|Pr[E_4]-Pr[E_3]| \leq q_{Tdec}.\epsilon_{BF}$, where ϵ_{BF} is the probability of the adversary A to break the scheme BF in [], and q_{Tdec} is the maximum number of queries to O_{Tdec}.
- In Game G_5, we change the TB-PRE decryption oracle by using the re-encryption decryption oracle. Because of the correctness of TB-PRE, this change is purely conceptual. Therefore $Pr[E_5] = Pr[E_4]$.
- In Game G_6, we define $g' = g_{+1}$ which is chosen randomly from G. The challenger C has knowledge of a value g_{-1} satisfying $e(g_{=1}, g_{-1}) = e(g, g)$ but is unaware of the value of $log_g g_{+1}$. However, the change is purely conceptual. Therefore $Pr[E_6] = Pr[E_5]$.
- In Game G_7, we change the re-encryption decryption oracle as follow. The re-encryption output with the challenge ciphertext are recorded in Table T_{re}. Due to restrictions in the security model, the change is purely conceptual. Therefore, $Pr[E_7] = Pr[E_6]$.
- In Game G_8, we modify the re-encryption key generation oracle by using random oracles. We therefore have $Pr[E_8] = Pr[E_7]$
- In Game G_9, we change the uncorrupted key generation and the corrupted key generation by using the DBDH problem input. We therefore have $|Pr[E_9] - Pr[E_8]| \leq q_{okg}.\epsilon_{DBDH}$.
- In Game G_{10}, we change the challenge phase by using random values (X, Y, V, Z). Using the DBDH assumption, we have $|Pr[E_1 0] - Pr[E_9]| \leq \epsilon_{DBDH}$. Furthermore, $Pr[E_1 0] = \frac{1}{2}$ due to the randomness of (A, B, C, T).

By combining the different games above, we prove our scheme secure in the random oracle model using the Lemma 1, under assumptions that the DBDH problem is hard. We hence complete our proof.

References

[1] A. Abolfazli, S. Sanaei, Z. Sanaei, M.H. Shojafar, M. Gani, Mobile cloud computing: The-state-of-the-art, challenges, and future research, Encycl. Cloud Comput. (FEBRUARY) (2015) 24–32.

[2] C. white paper, Cisco visual networking index: Global mobile data traffic forecast update the cisco® visual networking index (VNI) global mobile data traffic forecast update, Cisco (2016) 2016–2021, URL https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html.

[3] T. Peng, Y. Lin, X. Yao, W. Zhang, An efficient ranked multi-keyword search for multiple data owners over encrypted cloud data, IEEE Access (2018) 1, http://dx.doi.org/10.1109/ACCESS.2018.2828404.

[4] A. Sahai, B. Waters, Fuzzy identity based encryption, Eurocrypt '05 (2005) 457–473, URL http://eprint.iacr.org/2004/086.

[5] M. Blaze, G. Bleumer, M. Strauss, Divertible protocols and atomic proxy cryptography, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), Vol. 1403, 1998, pp. 127–144, http://dx.doi.org/10.1007/BFb0054122.

[6] Q. Tang, Type-based proxy re-encryption and its construction, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), in: LNCS, vol. 5365, 2008, pp. 130–144, http://dx.doi.org/10.1007/978-3-540-89754-5_11.

[7] J. Zhang, Z. Zhang, H. Guo, Towards secure data distribution systems in mobile cloud computing, IEEE Trans. Mob. Comput. 16 (11) (2017) 3222–3235, http://dx.doi.org/10.1109/TMC.2017.2687931.

[8] B. Libert, D. Vergnaud, Unidirectional chosen-ciphertext secure proxy re-encryption, IEEE Trans. Inform. Theory 57 (3) (2011) 1786–1802, http://dx.doi.org/10.1109/TIT.2011.2104470.

[9] J.W. Seo, D.H. Yum, P.J. Lee, Proxy-invisible CCA-secure type-based proxy re-encryption without random oracles, Theoret. Comput. Sci. 491 (2013) 83–93, http://dx.doi.org/10.1016/j.tcs.2012.11.026.

[10] J. Qiu, G.H. Hwang, H. Lee, Efficient conditional proxy re-encryption with chosen-ciphertext security, in: Proc. - 2014 9th Asia Jt. Conf. Inf. Secur. AsiaJCIS 2014, 2014, pp. 104–110, http://dx.doi.org/10.1109/AsiaJCIS.2014.11.

[11] V.K.A. Sandor, Y. Lin, X. Li, Efficient decentralized multi-authority attribute based encryption for mobile cloud data storage, J. Netw. Comput. Appl. (2019) http://dx.doi.org/10.1016/j.jnca.2019.01.003, URL http://www.sciencedirect.com/science/article/pii/S1084804519300037.

[12] Wikipedia, Proxy re-encryption — Wikipedia{, } The Free Encyclopedia (2017). URL https://en.wikipedia.org/w/index.php?title=Proxy{_}re-encryption{&}oldid=765960567.

[13] X.A. Wang, F. Xhafa, Z. Zheng, J. Nie, Identity based proxy re-encryption scheme (ibpre+) for secure cloud data sharing, in: 2016 Int. Conf. Intell. Netw. Collab. Syst., IEEE, 2016, pp. 44–48, http://dx.doi.org/10.1109/INCoS.2016.83, URL http://ieeexplore.ieee.org/document/7695147/.

[14] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: Proc. 13th ACM Conf. Comput. Commun. Secur. - CCS '06, 2006, p. 89,http://dx.doi.org/10.1145/1180405.1180418, URLhttp://portal.acm.org/citation.cfm?doid=1180405.1180418.

[15] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute-based encryption, in: Proc. - IEEE Symp. Secur. Priv., 2007, pp. 321–334, http://dx.doi.org/10.1109/SP.2007.11.

[16] S. Wang, K. Liang, J.K. Liu, J. Chen, J. Yu, W. Xie, Attribute-based data sharing scheme revisited in cloud computing, IEEE Trans. Inf. Forensics Secur. 11 (8) (2016) 1661–1673,http://dx.doi.org/10.1109/TIFS.2016.2549004, URLhttp://ieeexplore.ieee.org/document/7448433/.

[17] M. Chase, M. Chase, Multi-authority attribute based encryption, in: Theory Cryptogr. 4th Theory Cryptogr. Conf., Vol. 4392, 2007, pp. 515–534, http://dx.doi.org/10.1007/978-3-540-70936-7, URLhttp://www.springerlink.com/index/10.1007/978-3-540-70936-7.

[18] M. Chase, S.S. Chow, Improving privacy and security in multi-authority attribute-based encryption, Proc. 16th ACM Conf. Comput. Commun. Secur. - CCS '09 (2009) 121,http://dx.doi.org/10.1145/1653662.1653678, URLhttp://portal.acm.org/citation.cfm?doid=1653662.1653678.

[19] S. Yu, C. Wang, K. Ren, W. Lou, Achieving secure,scalable ,and fine-grained data access control in cloud computing.pdf, IEEE Infocom (2010) 1–9,http://dx.doi.org/10.1109/INFCOM.2010.5462174,arXiv:arXiv:1011.1669v3.

[20] M. Blaze, G. Bleumer, M. Strauss, Divertible protocols and atomic proxy cryptography, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), Vol. 1403, 1998, pp. 127–144,http://dx.doi.org/10.1007/BFb0054122.

[21] G. Ateniese, K. Fu, M. Green, S. Hohenberger, Improved proxy re-encryption schemes with applications to secure distributed storage, ACM Trans. Inf. Syst. Secur. 9 (1) (2006) 1–30,http://dx.doi.org/10.1145/1127345.1127346, URLhttp://portal.acm.org/citation.cfm?doid=1127345.1127346.

[22] C.-K. Chu, W.-G. Tzeng, Identity-based proxy re-encryption without random oracles, in: Inf. Secur., in: LNCS, vol. 4779, 2007, pp. 189–202, URL http://dx.doi.org/10.1007/978-3-540-75496-1{_}13.

[23] M. Green, G. Ateniese, Identity-based proxy re-encryption, Appl. Cryptogr. Netw. Secur. 4521 LNCS (2007) 288–306,http://dx.doi.org/10.1007/978-3-540-72738-5_19, URLhttp://link.springer.com/10.1007/978-3-540-72738-5{_}19.

[24] Tadapaneni, N. R. (2018). Cloud Computing: Opportunities And Challenges. International Journal of Technical Research and Applications. SSRN Electronic Journal. http://dx.doi.org/10.2139/ssrn.3563342

[25] J. Weng, R.H. Deng, X. Ding, C.-K. Chu, J. Lai, Conditional proxy re-encryption secure against chosen-ciphertext attack, in: Proceedings of the 4th International Symposium on Information, Computer, and Communications Security, in: ASIACCS '09, ACM, New York, NY, USA, 2009, pp. 322–332,http://dx.doi.org/10.1145/1533057.1533100, URLhttp://doi.acm.org/10.1145/1533057.1533100.

[26] J. Weng, Y. Yang, Q. Tang, R.H. Deng, F. Bao, Efficient conditional proxy re-encryption with chosen-ciphertext security, in: P. Samarati, M. Yung, F. Martinelli, C.A. Ardagna (Eds.), Information Security, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 151–166.

[27] C.-I. Fan, J.-C. Chen, S.-Y. Huang, J.-J. Huang, W.-T. Chen, Provably secure timed-release proxy conditional reencryption, IEEE Syst. J. 11 (2017) 2291–2302,http://dx.doi.org/10.1109/JSYST.2014.2385778.

[28] P. Zeng, K.R. Choo, A new kind of conditional proxy re-encryption for secure cloud storage, IEEE Access 6 (2018) 70017–70024,http://dx.doi.org/10.1109/ACCESS.2018.2879479.

[29]W. Zhang, S. Xiao, Y. Lin, T. Zhou, S. Zhou, Secure ranked multi-keyword search for multiple data owners in cloud computing, in: 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2014, pp. 276–286,http://dx.doi.org/10.1109/DSN.2014.36.

[30] J. Shao, P. Liu, G. Wei, Y. Ling, Anonymous proxy re-encryption, Secur. Commun. Netw. 5 (5) (2012) 439–449,arXiv:https://onlinelibrary.wiley.com/doi/pdf/101002/sec.326,http://dx.doi.org/101002/sec.326, URLhttps://onlinelibrary.wiley.com/doi/abs/101002/sec.326.

[31] Arora, R., Parashar, A., & Transforming, C. C. I. (2013). Secure user data in cloud computing using encryption algorithms. International journal of engineering research and applications, 3(4), 1922-1926.

[32] A. Kerckhoffs, La cryptographie militaire, J. des Sci. Mil. IX (1883) 5–83, URLhttp://www.petitcolas.net/fabien/kerckhoffs/.

[33] H. Krawczyk, M. Bellare, R. Canetti, Hmac: Keyed-hashing for message authentication (1997).

[34] P.S. Barreto, M. Naehrig, Pairing-friendly elliptic curves of prime order, in: Lect. Notes Comput. Sci. (Including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), in: LNCS, vol. 3897, 2006, pp. 319–331,http://dx.doi.org/10.1007/11693383_22.

[35] B. W. a. role) John Bethencourt, Amit Sahai (advisory role), Advanced Crypto Software Collection (2006). URLhttp://acsc.cs.utexas.edu/cpabe/.

[36] Ben Lynn, PBC Library - Pairing-Based Cryptography - About. URLhttps://crypto.stanford.edu/pbc/.

[37] B. Libert, D. Vergnaud, Unidirectional chosen-ciphertext secure proxy re-encryption, IEEE Trans. Inform. Theory 57 (3) (2011) 1786–1802, http://dx.doi.org/10.1109/TIT.2011.2104470.

[38] X. Jia, J. Shao, J. Jing, P. Liu, Cca-secure type-based proxy re-encryption with invisible proxy, in: 2010 10th IEEE International Conference on Computer and Information Technology, 2010, pp. 1299–1305,http://dx.doi.org/10.1109/CIT.2010.234.