An Efficient Attribute-Based Encryption and Access Control Scheme for Cloud Storage Environment

Jyun-Yao Huang, Chen-Kang Chiang, and I-En Liao

Department of Computer Science and Engineering
National Chung Hsing University, Taichung, Taiwan
allen501pc@gmail.com, s99056051@cs.nchu.edu.tw,
ieliao@nchu.edu.tw

Abstract. With the prevalence of cloud computing, many enterprise users store confidential information in the cloud servers. Therefore, the problems of data security in cloud computing are particularly important. Cloud storage service providers must offer efficient cryptography system and access control scheme to users. In recent years, some researchers proposed identity-based hierarchical key deployment model for encryption and access control in cloud computing environment. However, some of these schemes have high computing cost and do not take authentication into consideration. In this paper, we proposed a low-cost cryptography system and attribute-based access control scheme for the cloud storage environment. The simulation results and analysis show that the proposed method has lower communication and computing cost than Hierarchical Attribute-Based Encryption (HABE). Our proposed scheme can achieve the data access control via user's attribute-based rules. It also satisfies the authentication requirements by using identity-based signature in the cloud storage environment.

1 Introduction

Cloud computing is a large-scale distributed computing paradigm[1]. With the emergence of cloud computing, numerous convenient services have been provided, such as Google Gmail, Amazon EC2, Facebook and Dropbox. However, many enterprises must trust the security policies and mechanisms of cloud service providers. Cloud service providers should satisfy security requirements, such as data encryption, key management, identity authentication, and access control[2].

In recent years, a number of researchers have proposed various hierarchical architectures for the key management of cloud computing[3][4][5]. These schemes have provided some of encryption, authentication, and access control mechanisms, but not all of them. However, these proposed schemes also have a number of disadvantages. First, the computation costs of encryption and decryption are high. Second, they lack the integration of key management, encryption, authentication and access control mechanisms. Therefore, how to develop efficient encryption, authentication, and access control mechanisms is the primary issue investigated in this study.

This study proposes an efficient attribute-based encryption and access control scheme for cloud storage environments. The characteristics of the proposed scheme are described below:

J.J. Park et al. (Eds.): GPC 2013, LNCS 7861, pp. 453–463, 2013.
© Springer-Verlag Berlin Heidelberg 2013

(1) The scheme provides a method for encryption and decryption that has comparatively lower communication and computation costs. It also satisfies access control requirements by including attribute-based rules. Users who satisfy attribute-based rules can be permitted to download and decrypt data from cloud storage servers.

(2) The scheme provides identity-based signatures for authentication as users upload data and achieves non-repudiation.

(3) The scheme provides a suitable hierarchical key management method for cloud storage environments. Using hierarchical architecture, the generation and management of public and private keys does not incur server overheads.

The remainder of this study is organized as follows. In Section 2, related work on cloud computing, key management, access control are discussed. In Section 3, the proposed scheme for key management, encryption, authentication, access control, and theoretical analysis of performance are described. The experimental results and security analysis are discussed in Section 4. Section 5 provides the study conclusions.

2 Related Work

2.1 Cloud Computing

Cloud computing is a large-scale distributed computing paradigm. The National Institute of Standards and Technology (NIST) defined cloud computing as “Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction” [6][7]. Typically, cloud providers have their own cloud infrastructures or corresponding applications to provide services to their customers. The following three service models are commonly used for cloud computing: (1) Infrastructure as a Service (IaaS), which provides cloud computing infrastructures for customers. (2) Platform as a Service (PaaS), which provides both IaaSs and platform components such as operating systems or required libraries. (3) Software as a Service (SaaS), which provides applications on the cloud computing platform. The NIST [6][7] has also defined the following deployment models for cloud computing: public cloud, private cloud, hybrid cloud and community cloud.

2.2 Key Management in Cloud Computing

In previous studies of key management [3][4][8][9], how to distribute and compute public/private keys which are computed by user or device identities has been a significant issue.

In identity-based key management systems, the widely used mathematical property is bilinear maps based on an ellipse curve cryptography system [10]. Boneh and Franklin [11] proposed a security model for identity-based encryption and proposed a construction method using bilinear maps. For bilinear maps, they considered a large prime p and E to be the elliptic curve. Two groups exist; that is, a group over curve

E/F_p^2 with a large order q and denoted as G_q, and μq as a subgroup of F_q^*, which is also of the q order. They contended that a modified bilinear pairing ê: G_qx$G_q \rightarrow \mu_q$ is admissible if it possesses the following three properties:

(1) *Bilinear: For all* P, *Q*∈G_q *and a, b*∈*Z, ê(*aP, bQ*)= ê(*bP, aQ*)=ê(*P, Q*)*^{ab}. *This can be restated for all* P, Q, R∈G_q, *ê(*P+R, Q*)=ê(*P, Q*) ê(*R, Q*) and ê(*P, Q+R*)= ê(*P, Q*) ê(*P, R*).*

(2) *Non-degenerate: ê(P, P)∈Fq*, is an element of order q, and a generator of* μ_q .

(3) *Computable: Given* P, Q∈G_q, *an effective method to compute ê(*P, Q*) exists.*

They proposed the identity-based encryption method developed from admissible pairing; the security of this scheme was enhanced by computational Diffie-Hellman (CDH) and Bilinear Diffie-Hellman (BDH) assumptions. The BDH assumption is explained as follows:

Given P, aP, bP, $cP \in G_1$ for unknown random a, b, $c \in Z_p^*$, computing $r= ê(P, P)^{abc}$ is difficult when G_1 is a bilinear group. However, the method developed by Boneh and Franklin is not efficient for large networks. Jeremy Horwitz et al. [12] introduced a hierarchical concept for a two-level hierarchical ID-based encryption(HIDE). Subsequently, Gentry and Silverberg [13] proposed the practical scheme for this concept.

In recent years, Hongwei Li et al. [3] proposed an identity-based hierarchical model for cloud computing (IBHMCC). In their scheme, the cloud computing environment was composed of a hierarchical structure of nodes, and authentication was conducted using bilinear pairing. However, this scheme cannot defend against replay attacks, when attackers repeatedly transmit authentication messages to the server. Liang Yan et al. [4] adopted federated identity management combined with hierarchical identity-based cryptography (HIBC) using shared secret session key without a pre-shared secret key. However, Hongwei Li et al. [3] and Liang Yan et al. [4] did not consider the key regeneration problem when lower level PKGs failed. In a previous study, we proposed a robust and low-cost identity-based encryption method for a hierarchical key distribution model by considering PKG failures[14].

2.3 Access Control

In cloud storage service, how to retain the security of data while providing valid access ability is an issue that concerns users. For public data, data owners can use the basic access control APIs provided by cloud service providers to offer access control ability. For secret data, the data owner can use encryption and add an access control table into target data to limit the users' access abilities. This study refers to these methods as "ciphertext-policy encryption".

For cloud computing, Zhu et al. [15] provided role-based access control (RBAC)[16] to offer access control abilities to target users. However, this method involves the use of the receiver's personal information to encrypt data. To submit one file to various receivers, this method entails numerous encryption procedures. Therefore, this method is obviously unsuitable for broadcasting messages.

For access control, attribute-based access control is an alternative method. Sahai et al. [17] developed a primitive attribute-based encryption model based on the method presented by Boneh and Franklin [11]. This method used user attributes as the access control permissions because each user has unique attributes. Attribute rules were transformed into ciphertext, and the users whose secret keys satisfied the rules could decrypt the ciphertext successfully. Goyal et al. [18] provided key-policy attribute-based encryption (KP-ABE) for policy based rules. Bethencourt et al [19] developed ciphertext-policy attribute-based encryption (CP-ABE); The benefit of this method is that senders are not required to know the receivers, they simply incorporate the access rules into the encrypted data; the data is then suitably protected on the distributed system with widely unknown users. Therefore, this method has been extended to cloud computing environment.

In recent years, a number of studies have consolidated CP-ABE[19] and HIDE[13] for cloud computing. In 2010, Jie Wu et al. [5] proposed using hierarchical attribute-based encryption (HABE); This reduces the overheads of key generation procedures through hierarchical architecture, and this architecture is used to manage domain access conveniently. When one user wishes to share secret data, this scheme can use an access tree structure consisting of attribute sets to verify receivers' permissions. However, this scheme only provides encryption, signature authentication functions are lacking. Additionally, the scheme had higher decryption computation and communication costs.

3 Proposed Scheme

The objective of this study was to improve HABE[5] by reducing encryption, decryption, and signature authentication costs. The architecture proposed in this study is shown in Fig. 1.

As shown in Fig. 3, each cloud computing server possesses a public key and private key generator and is considered a PKG. “Root PKG” possesses different PKGs to “Domain PKG,” and each domain PKG has users and attribute data. For security reasons, attribute data are stored on one server under the domain PKG to separate users and the attribute data.

This study features various key notations; thus, a key notations list is provided in Table 1.

In this study, unique identifiers are used for each PKG, user, and attribute. The public keys generated using these unique identifiers comprise the following concepts:

(1) *If the public key under level i for the specific domain PKG is composed of an upper PKG public key PKi-1 and self-identifier IDi, it is denoted as PKi = (PKi-1,IDi).*

(2) *When one user with his identifier IDu add to PKG under level i including public key PKi, his public key is denoted by PKu = (PKi, IDu).*

If some data attributes with identifier ID_a stored in the PKG under level i, its' public key is denoted as PK_a= (PK_i,ID_a).

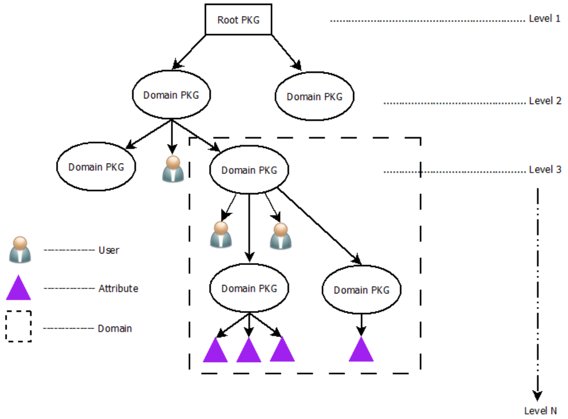


Fig. 1. System architecture

Table 1. Key notations list

Notation	Meaning	Purpose
PK_i	The public key of the *i*th domain PKG	To generate private keys for each sub-domain PKG
PK_a	The public key of various attribute data.	To generate private keys that satisfy a number of user attribute rules.
PK_u	The public key of a number of users.	To generate private keys for a number of users.
SK_i	The private key of the *i*th level domain PKG	To generate private keys for a number of users.
$SK_{i,u}$	The private key of user *u* for domain PKG under level *i*	Decryption
$SK_{i,u,a}$	The private key of user *u* for domain PKG under level *i*, which satisfies attribute *a*	Decryption

3.1 PKG Setup

Root PKG Setup: the root PKG setup is conducted using the following steps:

(1) *Select a random number* $r_0 \in Z_q$ *with a large prime* q, *two groups* G_1 *and* G_2 *with order* q, *and bilinear map* $\hat{e}: G_1 \times G_1 \rightarrow G_2$.

(2) *Generate* $P_0 \in G_1$ *randomly and compute* $Q_0 = r_0 P_0 \in G_1$.

(3) *Select three hash functions for encryption,* H_1: *{0,1}*$^*\rightarrow G_1$, H_2: $G_2 \rightarrow$*{0,1}*^{n}, H_A:*{0, 1}*$\rightarrow Z_q$, *where* n *is any positive integer.*

(4) *Generate system parameters* $<G_1, G_2, \hat{e}, Q_0, P_0, H_1, H_2, H_A>$ *for the lower level domain PKGs.*

Domain PKG Setup: Suppose the public and private keys of domain PKG in level i+1 are generated by the upper level domain PKG or root PKG in level *i*. Then, the key generation is proceeded by domain PKG or root PKG in level *i* using the following steps:

(1) Generate two distinct random r_{i+1} and $r'_{i+1} \in Z_q$.

(2) Compute public key $P_{i+1} = H_1(PK_{i+1}) \in G_1$ and secret $Q_{i+1} = r_{i+1}P_0 \in G_1$.

(3) Generate the private key $SK_{i+1} = Q_0 + r'_{i+1}P_{i+1}$.

(4) Send system parameters $< G_1, G_2, \hat{e}, Q_0, P_0, H_1, H_2, H_A >$ to the sub-domain PKG.

3.2 User Addition

Assume that one user *u* with user public key PK_u who satisfies some data attributes *a* wants to join PKG_i (at level *i*) to access data. Then the PKG_i generate secret keys for the user using the following steps:

(1) Set $r_u = H_A(PK_u) \in Z_q$, $P_a = H_1(PK_a)$ and generate $r_i \in Z_q$.

(2) Generate two private keys $SK_{i,u} = r_i r_u P_0$ and $SK_{i,u,a} = SK_i + r_i r_u P_a$.

(3) Pass $SK_{i,u}$, $SK_{i,u,a}$ to user *u* under a secret channel.

3.3 Data Encryption and Decryption

Data Encryption: Assume that some user *u* in level *t* wants to upload file *f* to cloud servers. Then, he must proceeds the following steps:

(1) We use conjunctive clause(*CC*) to represent all attribute data in some server. Give disjunctive normal form(DNF) $A = \bigvee_{i=1}^{N} (CC_i) = \bigvee_{i=1}^{N} (\bigwedge_{j=1}^{n_i} a_{ij})$, where $N \in Z^+$ is the number of conjunctive clauses, $n_i \in Z^+$ is the number of attributes at the *i*th conjunctive clause (CC_i), and a_{ij} is the *j*th attribute at CC_i .

(2) Set $P_t = H_1(PK_t) \in G_1$ and $P_{a_{ij}} = H_1(PK_{a_{ij}}) \in G_1$, where $1 \leqq i < N$ and $1 \leqq j \leqq n_i$.

(3) Randomly select $\alpha \in Z_q$ and obtain the lowest common multiple $n_A = \{n_1, n_2, ..., n_N\}$. Compute $U_0 = \alpha P_0, U_t = \alpha P_t$, $U_1 = \alpha \sum_{j=1}^{n_1} P_{a_{1j}}$, $\ldots, U_N = \alpha \sum_{j=1}^{n_N} P_{a_{Nj}}$. Then, compute $V = f \oplus H_2\left(\hat{e}\left(\alpha Q_0, n_A P_0\right)\right)$ and $C_f = [U_0, U_t, U_1, \ldots, U_N, V]$.

(4) Output $CT = (A, C_f)$ to the target server.

Decryption: When user *u* wishes to obtain file *f* from one server in level *t*, they can pass the parameters $SK_{t,u}$ and $SK_{t,u,a_{ij}}$ to the server to decrypt *CT*. The server decryption procedure is run as follows to get file *f*:

$$V \oplus H_2\left(\frac{\hat{e}\left(U_0, \frac{n_A}{n_i}\sum_{j=1}^{n_i} SK_{t,u,a_{ij}}\right)}{\hat{e}\left(Q_t, n_A U_t\right)\hat{e}\left(SK_{t,u}, \frac{n_A}{n_i}U_i\right)}\right) = V \oplus H_2\left(\frac{\hat{e}\left(Q_0, n_A \alpha P_0\right)\hat{e}\left(Q_t, n_A U_t\right)\hat{e}\left(SK_{t,u}, \frac{n_A}{n_i}U_i\right)}{\hat{e}\left(Q_t, n_A U_t\right)\hat{e}\left(SK_{t,u}, \frac{n_A}{n_i}U_i\right)}\right) =$$

$$V \oplus H_2\left(\hat{e}\left(\alpha Q_0, n_A P_0\right)\right) = f$$

3.4 Signature

If user *u* in level *i* wishes to add a signature to data *f*, they should use the following procedure:

(1) Compute $P_m = H_1(ID_u \,\|\, f)$.

(2) Use the current time string T_C to compute $t = H_A(T_C)$.

(3) Randomly generate $\beta \in Z_q$.

(4) Compute $\mu = \beta SK_{i,u} + t\beta P_{\mathrm{m}} P_u$.

(5) Output signature $<\mu, \beta SK_{i,u}, t\beta P_m P_0> = <\mu, A_i, B_i>$.

When the server receives the signature $s' = <\mu', A_i', B_i'>$ from user *u* and a previous signature $s'' = <\mu'', A_i'', B_i''>$ from the same user exists, the user can perform the following two steps to verify the signature:

(1) If $B_i' \neq B_i''$ proceed to Step 2 or reject the data to prevent a “replay attack.”

(2) The verification is correct if $\hat{e}(P_0, \mu') = \hat{e}(P_0, A_i')\hat{e}(P_u, B_i')$.

When servers with ID_i in level *i* also create signatures, they follow similar procedures for authentication.

3.5 Theoretical Performance Analysis

The theoretical performance analysis was based on the assumption that a number of users in level *i* wish to access data on the server. To conduct clear comparisons, the notations used are explained below.

(1) C_{BM}: The time costs of bilinear map $\hat{e}$ computations.

(2) C_h: The time costs of the hash function.

(3) C_{xor}: The time costs of exclusive OR computation.

(4) $C_{parameter}$: The time costs of generating the required communicative parameters.

This study assumed that some user in some server at level *i*, uploads one file which satisfied *a* attribute rules (i.e., conjunctive clauses, CC). The computation costs of encryption and decryption are shown in Table 2 , and the required number of communication parameters is shown in Table 3.

Table 2 and Table 3 show that HABE uses iterative bilinear map operations because the generated keys are based on a hierarchical architecture. Thus, its computation costs are as high as the located levels of target data. For encryption, HABE also uses various conjunctive clauses at various levels, from root PKG to the current server, to generate permission parameters; therefore, it should generate 10 permission parameters when at level 10.

In the proposed scheme, because the keys are not generated based on hierarchical architecture, the decryption procedure requires minimal bilinear map operations and parameters.

Table 2. Comparison of computation costs of the encryption and decryption

Method	Computation	
	Encryption	Decryption
HABE	$1C_{BM} + 1C_h + 1C_{xor}+(a*i+1)C_{parameter}$	$(i+1)C_{BM}+ 1C_h + 1C_{xor}$
Proposed	$1C_{BM} + 1C_h + 1C_{xor}+ (a+2)C_{parameter}$	$3C_{BM} + 1C_h + 1C_{xor}$

Table 3. Comparison of parameters of the encryption

Method	Required number of parameters for encryption
HABE	$a * i + 1$
Proposed	$a + 2$

4 Experimental Results and Security Analysis

4.1 Experiment Environment

Under the proposed environment, the MIRACL [20] library and elliptic curve $y^2 = x^3 + 1$ were used to design the experiment. This study assumed that the number of attributes was 100 and the key size was 1024 bits. A number of emulations were adopted and the effluence of the levels of the proposed hierarchical cloud architecture was considered. Our hardware is based on AMD X4 640 processor, 4GB memory, 1TB(7200 rpm) hard disk and the software environment is based on C++ programming language and with Linux 2.6.32.

4.2 Experiment Analysis

Experiment 1. (Effects on the number of attributes): In this experiment, the effects on the number of attributes were considered. The experiment results are shown in Fig. 2.

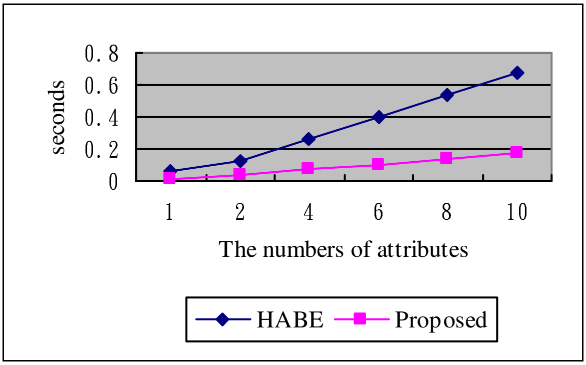
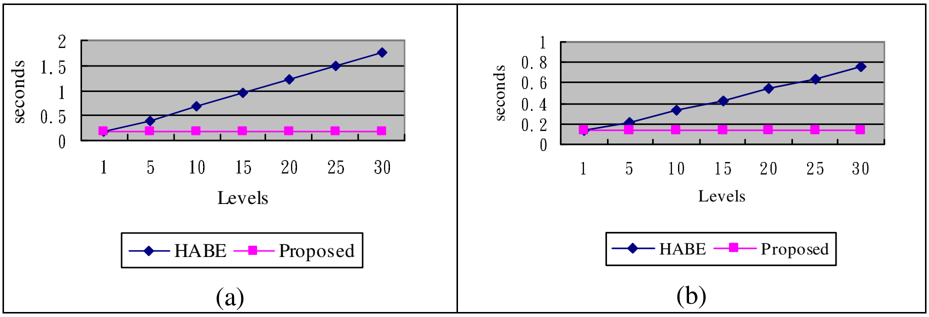


Fig. 2. Simulation result for the effects on the number of attributes

In Fig. 3, the encryption costs are significantly higher for a greater number of attributes. By contrast, the costs of the proposed scheme were lower than that of HABE, because unlike HABE, the proposed scheme does not involve iterative decryption and encryption operations, so its cost is lower than HABE

Experiment 2. (Effects on the numbers of levels): The encryption and decryption time costs for various levels were considered. The experiment results are shown in Fig. 3(a) and Fig. 3(b)



(a)

(b)

Fig. 3. (a) Simulation result for encryption at various levels (b) Simulation result for decryption at various levels

In Fig. 3(a) , as mentioned previously, HABE uses iterative key generation for encryption, resulting in higher time and communication costs; additionally, the computation costs increase with more levels. As shown in Fig. 3(b), the proposed scheme uses non-iterative key generation and considers less iterative bilinear operations; thus, the time costs do not increased with additional levels.

4.3 Security Analysis

This study considered the following general attack techniques to conduct security analysis:

(1) **Guessing Attack:** When one attacker attempts to crack the cloud server key using brute force. According to the CDH assumption[11], "given the points P, aP, $bP(a, b \in Z_q^*)$ at additive G_1 , compute abP is difficult." The key generation method of the proposed scheme was derived from $SK_{i,u} = r_i r_u P_0$ and $SK_{i,u,a} = SK_i + r_i r_u P_a$, which was also based on this assumption. Thus, the proposed method is secure.
(2) **Man-in-the-Middle Attack**: When one attacker intercepts the connection between the sender and receiver, they obtain the encrypted data and send fake data to the receiver. However, the scheme proposed in this study can prevent this attack using signature verification.
(3) **Replay Attack:** When one attacker uses the same signature and then sends data to the receiver repeatedly to busy it with verifications. However, this attack is nullified because the proposed scheme includes a timestamp in the signature and adopts the time verifier in Step 1 of signature verification.

5 Conclusions

This study proposed a low-cost cryptography system and attribute-based access control scheme for cloud storage environments. The simulation results and analysis showed that this method incurs lower communication and computing costs compared to HABE. The proposed scheme can achieve data access control through user attribute-based rules. The scheme also satisfies the authentication requirements by including identity-based signatures in the cloud storage environment.

However, this method was only implemented in a simulation environment because of resource limitations. In the future, we aim to implement this method in a true cloud storage environment for verification.

References

1. Foster, I., Zhao, Y., Raicu, I., Lu, S.: Cloud computing and grid computing 360-degree compared. In: Grid Computing Environments Workshop (GCE 2008), Austin, Texas, pp. 1–10 (2008)
2. Alliance, C.S.: Security guidance for critical areas of cloud computing version 3.0., https://cloudsecurityalliance.org/research/security-guidance/ (accessed July 20, 2012)
3. Li, H., Dai, Y., Tian, L., Yang, H.: Identity-based authentication for cloud computing. In: Proceedings of the 1st International Conference on Cloud Computing (CloudCom 2009), Beijing, China, pp. 157–166 (2009)
4. Yan, L., Rong, C., Zhao, G.: Strengthen cloud computing security with federal identity management using hierarchical identity-based cryptography. In: Jaatun, M.G., Zhao, G., Rong, C. (eds.) Cloud Computing. LNCS, vol. 5931, pp. 167–177. Springer, Heidelberg (2009)

5. Wang, G., Liu, Q., Wu, J.: Hierarchical attribute-based encryption for fine-grained access control in cloud storage services. In: Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS 2010), New York, NY, USA, pp. 735–737 (2010)
6. Grance, P.M.T.: The nist definition of cloud computing (15 ed.) National Institute of Standards and Technology (NIST), http://csrc.nist.gov/groups/SNS/cloud-computing (accessed July 20, 2012)
7. Grance, P.M.T.: The NIST Definition of Cloud Computing (Draft). National Institute of Standards and Technology (NIST), http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_cloud-definition.pdf (accessed July 20, 2012)
8. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
9. Ramgovind, S., Eloff, M., Smith, E.: The management of security in cloud computing. Information Security for South Africa (ISSA). University of Johannesburg, Johannesburg, South Africa, pp.1–7 (2010)
10. Miller, V.S.: Use of elliptic curves in cryptography. In: Williams, H.C. (ed.) CRYPTO 1985. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg (1986)
11. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, p. 213. Springer, Heidelberg (2001)
12. Horwitz, J., Lynn, B.: Toward hierarchical identity-based encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 466–481. Springer, Heidelberg (2002)
13. Gentry, C., Silverberg, A.: Hierarchical ID-based cryptography. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 548–566. Springer, Heidelberg (2002)
14. Huang, J.-Y., Liao, I.-E., Chiang, C.-K.: Efficient identity-based key management for configurable hierarchical cloud computing environment. In: IEEE 17th International Conference on Parallel and Distributed Systems (ICPADS 2011), Tainan, Taiwan, pp. 883–887 (December 2011)
15. Tianyi, Z., Weidong, L., Jiaxing, S.: An efficient role based access control system for cloud computing. In: IEEE 11th International Conference on Computer and Information Technology (CIT), pp. 97–102 (2011)
16. Tsai, W.-T., Shao, Q.: Role-based access-control using reference ontology in clouds. In: 2011 10th International Symposium on Autonomous Decentralized Systems (ISADS), pp. 121–128 (2011)
17. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
18. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, New York, NY, USA, pp. 89–98 (2006)
19. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: Proceedings of the 2007 IEEE Symposium on Security and Privacy, SP 2007, Washington, DC, USA, pp. 321–334 (2007)
20. CertiVox.: MIRACL Crypto SDK, http://certivox.com/index.php/solutions/miracl-crypto-sdk/