Hierarchical Attribute-Based Encryption for Fine-Grained Access Control in Cloud Storage Services

Guojun Wang, Qin Liu
School of Information Science and Engineering
Central South University
Changsha, Hunan Province, P. R. China, 410083
csgjwang@mail.csu.edu.cn

Jie Wu
Dept. of Computer and Information Sciences
Temple University
Philadelphia, PA 19122, USA
jiewu@temple.edu

ABSTRACT
Cloud computing, as an emerging computing paradigm, enables users to remotely store their data into a cloud so as to enjoy scalable services on-demand. Especially for small and medium-sized enterprises with limited budgets, they can achieve cost savings and productivity enhancements by using cloud-based services to manage projects, to make collaborations, and the like. However, allowing cloud service providers (CSPs), which are not in the same trusted domains as enterprise users, to take care of confidential data, may raise potential security and privacy issues. To keep the sensitive user data confidential against untrusted CSPs, a natural way is to apply cryptographic approaches, by disclosing decryption keys only to authorized users. However, when enterprise users outsource confidential data for sharing on cloud servers, the adopted encryption system should not only support fine-grained access control, but also provide high performance, full delegation, and scalability, so as to best serve the needs of accessing data anytime and anywhere, delegating within enterprises, and achieving a dynamic set of users. In this paper, we propose a scheme to help enterprises to efficiently share confidential data on cloud servers. We achieve this goal by first combining the hierarchical identity-based encryption (HIBE) system and the ciphertext-policy attribute-based encryption (CP-ABE) system, and then making a performance-expressivity tradeoff, finally applying proxy re-encryption and lazy re-encryption to our scheme.

Categories and Subject Descriptors
E.3 [Data Encryption]: Public key cryptosystems

General Terms
Security, Algorithms, Design

Keywords
cloud computing, hierarchical attribute-based encryption, fine-grained access control, scalability

1. INTRODUCTION
With the emergence of sharing confidential corporate data on cloud servers, it is imperative to adopt an efficient encryption system with a fine-grained access control to encrypt outsourced data. Ciphertext-policy attribute-based encryption (CP-ABE), as one of the most promising encryption systems in this field, allows the encryption of data by specifying an access control policy over attributes, so that only users with a set of attributes satisfying this policy can decrypt the corresponding data. However, a CP-ABE system may not work well when enterprise users outsource their data for sharing on cloud servers, due to the following reasons:

First, one of the biggest merits of cloud computing is that users can access data stored in the cloud anytime and anywhere using any device, such as thin clients with limited bandwidth, CPU, and memory capabilities. Therefore, the encryption system should provide high performance.

Second, in the case of a large-scale industry, a delegation mechanism in the generation of keys inside an enterprise is needed. Although some CP-ABE schemes support delegation between users, which enables a user to generate attribute secret keys containing a subset of his own attribute secret keys for other users, we hope to achieve a full delegation, that is, a delegation mechanism between attribute authorities (AAs), which independently make decisions on the structure and semantics of their attributes.

Third, in case of a large-scale industry with a high turnover rate, a scalable revocation mechanism is a must. The existing CP-ABE schemes usually demand users to heavily depend on AAs and maintain a large amount of secret keys storage, which lacks flexibility and scalability.

Motivation. Our main design goal is to help the enterprise users to efficiently share confidential data on cloud servers. Specifically, we want to make our scheme more applicable in cloud computing by simultaneously achieving fine-grained access control, high performance, practicability, and scalability.

Our Contribution. In this paper, we first propose a hierarchical attribute-based encryption (HABE) model by combining a HIBE system and a CP-ABE system, to provide fine-grained access control and full delegation. Based on the HABE model, we construct a HABE scheme by making a performance-expressivity tradeoff, to achieve high performance. Finally, we propose a scalable revocation scheme by delegating to the CSP most of the computing tasks in revocation, to achieve a dynamic set of users efficiently.

2. THE HABE MODEL
The HABE model (see Figure 1) consists of a root master (RM) that corresponds to the third trusted party (TTP), multiple domain masters (DMs) in which the top-level DMs
correspond to multiple enterprise users, and numerous users that correspond to all personnel in an enterprise.

Figure 1: A three-level HABE model

The RM, whose role closely follows the root private key generator (PKG) in a HIBE system, is responsible for the generation and distribution of system parameters and domain keys. The DM, whose role integrates both the properties of the domain PKG in a HIBE system and AA in a CP-ABE system, is responsible for delegating keys to DMs at the next level and distributing keys to users. Specifically, we enable the leftmost DM at the second level to administer all the users in a domain, just as the personnel office administers all personnel in an enterprise, and not to administer any attribute. Notice that other DMs administer an arbitrary number of disjoint attributes, and have full control over the structure and semantics of their attributes.

In the HABE model, we first mark each DM and attribute with a unique identifier (ID), but mark each user with both an ID and a set of descriptive attributes. Then, as Gentry et al [1], we enable an entity’s secret key to be extracted from the DM administering itself, and an entity’s public key, which denotes its position in the HABE model, to be an ID-tuple consisting of the public key of the DM administering itself and its ID, e.g., the public key of DM1 with ID1 is in the form of (PK1, ID1), the public key of user U with IDa is in the form of (PKa, IDa), and the public key of attribute a with IDa is in the form of (PKa, IDa), where PK1, PK0, and PKa are assumed to be the public keys of the DMs that administer DM1, U, and a, respectively.

3. CONSTRUCTION

Based on the proposed HABE model, we construct the HABE scheme using the bilinear map [1]. As Muller et al [2] sacrificing the expressivity of access structure to achieve better performance, we also use disjunctive normal form (DNF) policy. We assume that all attributes in one conjunctive clause are administered by the same DM. The HABE scheme consists of the following five algorithms:

Setups(K) → (params, MK0) : The RM first picks mk0 ∈ Zq, and then chooses groups G1 and G2 of order q, a bilinear map e : G1 × G1 → G2, two random oracles H1 : {0, 1}∗ → G1 and H2 : G2 × {0, 1}n → G2. The system parameters params = (q, G1, G2, e, n, P0, Q0, H1, H2) will be publicly available, while MK0 = (mk0) will be kept secret.

CreateDM(params, MKi, PKi+1) → (MKi+1) : To generate the master key for DMi+1 with PKi+1, the RM or DMi first picks a random element mk+1 ∈ Zq, and then computes SKi+1 = SKi + mk+i+1Pk+i+1 where P+i+1 = H1(PKi+1) ∈ G1, and Qi+1 = mk+i+1P0 ∈ G1, finally sets MKi+1 = (mk+i+1, SKi+1, Q-tuple(e) where Q-tuple(e) = (Q-tuple, Qi+1), and gives the random oracle H2 : {0, 1} → Zq that is chosen by the RM and shared in a domain. Here, we assume that SKi is an identity element of G1, and Q-tuple = (Q0).

CreateUser(params, MKi, PKu, PKa) → (SKi,u, SKi,u,a) : To generate a secret key for user U with PKu on attribute a with PKa, DMi first checks whether U is eligible for a, and a is administered by itself. If so, it first stores mk+i = H1(PKi) ∈ Zq, SKi,u = mk+iSKi, P0 ∈ G1, and SKi,u,a = SKi + mk+iSKi, where P0 = H1(PKi) ∈ G1, and then gives Q-tuple(e). Otherwise, it outputs “NULL”.

Encrypt(params, A, {PKa}|1 ≤ i ≤ N, 1 ≤ j ≤ ni), f) → (CT) : Given a DNF access control policy A = N i=1 (CCi) = N i=1 fi ≈ 0, (A ∧ ai), where N ∈ Z+ is the number of conjunctive clause in A, ni ∈ Z+ is the number of attributes in the i-th conjunctive clause CCi, and ai is the j-th attribute in CCi. Let DMi, with (IDi1, . . . IDit) be the DM at level ti, administrator all attributes in CCi, where IDik for 1 ≤ k < ti are IDs of DMi−k’s ancestors. The sender:

1. For 1 ≤ i ≤ N: Computes Pi = H1(IDi1, . . . IDit) ∈ G1 for 1 ≤ j ≤ ti, and P auj = H1(IDi1, . . . IDit, IDauj) ∈ G1 for 1 ≤ j ≤ ni.

2. Picks a random element r ∈ Zq, sets nk to be the lowest common multiple (LCM) of ni1, . . . , niN, and computes U0 = rP0, U12 = rP12, . . . , Ut = rPt, U1 = r ni j=1 Paj, . . . , Un = rNnj = rPnNj, and V = f ⊕ H2(e(Q(u), rnaP1)).

Decrypt(params, CT, SKi,u,a, {SKi−k,u,a}|1 ≤ k ≤ ni), Q-tuple(e) → (f) : User U, whose attributes satisfy CCi, computes V ⊕ H2(e(Ui1, naQ1j)) to recover f. Observe that:

V ⊕ H2(e(Ui1, naQ1j)) = V ⊕ H2(e(Q(u), rnaP1)) = V ⊕ H2(e(Q(u), rnaP1)) = V ⊕ H2(e(Q(u), rnaP1)) as required.

Remark. To achieve better performance, we enable user U to send the value of Q-tuple(e) to the CSP before decrypting data, so that the CSP can help to calculate the value of H2(e(Ui1, naQ1j)). Given this value, U executes the bilinear map operations for two times to recover the file.
5. REVOCATION

Performance. The HABE scheme eliminates the on-line
inquiry for authenticated attribute public keys. When there is
only the first level DM to administer all attributes, a user
needs to execute one bilinear map and O(N) number of point
multiplication operations to output a ciphertext of O(N)
length, and O(1) bilinear map operations to recover a file.

4. SECURITY INTUITION

Recall that a confidential file $f$ is encrypted in the form of
$f \oplus H_2(\hat{e}(Q_0, r_n P_0))$. Therefore, an adversary $\mathcal{A}$ needs to
construct $\hat{e}(Q_0, r_n P_1) = \hat{e}(U_0, SK_I)^{a_k}$ to decrypt $C_f$. $\mathcal{A}$
can request any user key of his choice other than possessing
a sufficient set of attribute keys to decrypt $C_f$.

For ease of presentation, we have the following assump-
tions: $\mathcal{A}$ has requested attribute secret keys for user $U$ on
all but one of the attributes $a_1, \ldots, a_{(k-1)}, a_{(k+1)}, \ldots, a_{n}$,
in $CC_i$, and for user $U'$ on the missing attribute $a_k$.

The only occurrence of $SK_I$ is in the user attribute secret
key, therefore adversary $\mathcal{A}$ has to use the attribute secret
keys requested for $U$ and $U'$ for the bilinear map, yielding:

$$\hat{e}(U_0, \frac{a_k}{a_k} \sum_{j=1, j \neq k}^{n} SK_{it, a, ij} + \frac{a_k}{a_k} SK_{it, a', a_k} + \alpha)$$

$$\hat{e}(U_0, SK_I)^{a_k} = \hat{e}(Q_1, \ldots, Q_{n-1}); U_{it, i, ij}^{a_k} \hat{e}(r P_0, \alpha)$$

$$\hat{e}(mk_{ij} mk_{it} P_0, r P_{ij}^{a_k} \hat{e}(mk_{ij} mk_{it} P_0, r \sum_{j=1, j \neq k}^{n} P_{ij})^{a_k}$$

for some $\alpha$. To obtain $\hat{e}(U_0, SK_I)^{a_k}$, the last four elements
have to be eliminated. However, the values of the last two
elements are unknown to the adversary, and cannot be con-
structed. Therefore, $\mathcal{A}$ cannot recover the file.

5. REVOCATION

Inspired by Yu et al [3], we make slight alterations into
our HABE scheme, and apply proxy re-encryption and lazy
re-encryption into our scheme.

Modifications in keys: We enable each attribute $a$ with
ID$_a$ to be bound to a version number, which increases by one
whenever a user associated with $a$ is revoked. Therefore, an
attribute public key is the form of PK$_{a}^{t} = (v'_a, \text{PK}_a, \text{ID}_a)$,
where $t \in \mathbb{Z}_q$ is the version number of the attribute public
key, and $v'_a \in \{0, 1\}^*$ is a string corresponding to $t$.

Modifications in algorithms: First, we enable the Create-
DM algorithm to uniformly and randomly generate a
hash function $H_{mk_i}: \{0, 1\}^{t_q} \rightarrow \mathbb{Z}_q$ for DM$_i$, where $H_{mk_i}$ is
a random oracle. Second, we construct another algorithm Create-
Attribute($PK_{a}^{t} = (v'_a, \text{PK}_a, \text{ID}_a)$, which is executed by DM$_i$
whenever it receives a request for $P_{it}^{t}$, and outputs $H_{mk_i}(PK_a^{t})P_0 \in \mathbb{G}_1$.
Therefore, the first step in the Encrypt algorithm turns into
requesting $P$-values of all attributes in $A$ from the DMs.

When a user is revoked, denoted V, it is imperative to
update public keys of attributes in $S_V$, and attribute secret
keys for remaining users who possess at least one attribute
in $S_V$, and re-encrypt data whose access structure specifies
at least one attribute in $S_V$, where the set $S_V$ contains all
attributes associated with $V$. If all these tasks are performed
by the DMs themselves, it would introduce a heavy comput-
ing overhead and may also require the DMs to always be
online. Therefore, we get the idea to take advantage of the
abundant resources in a cloud by delegating to CSPs most
of the computing tasks in revocation.

The main process is as follows: First, for each attribute $a$ in $S_V$, we enable the DM to update $PK_a^{t}$, by adding each version number to 1, and compute the the
PRE key with $\text{Pkey}^{t+1}_a = H_{mk_i}(PK_a^{t+1}) - H_{mk_i}(PK_a^{t})$. After
sending the update messages to the CSP, the DMs can go
off-line. Second, for every attribute in $S_V$, the CSP stores a
new attribute public key and PRE key received in a proper
position in an attribute history list (AHL). Whenever an ac-
cess request is received from a user, denoted $U$, it first checks
whether all attributes in the data access structure are the
latest ones. If so, it returns the data directly. Otherwise, it
re-encrypts the data, and tells the user to update his overdue
attribute secret keys, using related PRE keys, as follows:

Data Re-encryption. Suppose $CT' = (a', [U_0, U_{1j}, \ldots, U_{i1}, U_i, \ldots, U_{N, 1}], U_{N, 2}, \ldots, U_{Ni}, U_N, V')$ is the original cipher-
text. The CSP inquires the AHL and re-encrypts $CT'$ to $CT$
by setting $V = V'$, $U_0 = U_0'$, and $U_i = U'_i + \sum_{a \in S_V} \wedge CC_i$
($\text{Pkey}_{a}^{t+1} \ldots + \text{Pkey}'_{a}U_{a}$, for $1 \leq i \leq N$, where $t'$ and $t$
are the version numbers of $a$'s public key in $A'$ and
the latest version number of $a$'s public key in AHL, respectively.

Key Update. For every attribute $a$ in $S_V$, which is
a set of overdue attributes associated with user $U$, $U$ up-
dates corresponding attribute secret key $SK'_{i, u, a}$ to $SK_{i, u, a}$
by setting $SK'_{i, u, a} = SK_{i, u, a} + P_{key}'_{a}$, in which $P_{key}'_{a}$,
taken from the CSP, is set to be $P_{key}'_{a}^{t+1} \ldots + P_{key}'_{a}U_{a}$, where $t'$
and $t$ are the version number of $a$'s public key for $U$ and the latest version number of $a$'s public key in AHL, respectively.

6. CONCLUSION AND FUTURE WORK

In this paper, we construct a scheme, which has several
traits: (1) high performance; (2) fine-grained access control;
(3) scalability; (4) full delegation. Our HABE scheme, which
is also collusion resistant, can be proven to be semantically
secure against adaptive chosen plaintext attacks under the
BDH assumption and the random oracle model [1].

In future work, we will work towards designing a more
expressive scheme, which can be proved to have full security
under the standard model, with better performance.

7. ACKNOWLEDGMENTS

This work is supported by National Natural Science Foun-
dation of China under Grant Nos. 90718034 & 60773013,
the Program for New Century Excellent Talents in University
under Grant No. NCET-06-0686, and the Program for
Changjiang Scholars and Innovative Research Team in Univer-
sity under Grant No. IRT0661.

8. REFERENCES

Cryptography. In Proceedings of ASIACRYPT 2002,
pages 548-566.

Attribute-Based Encryption. In Proceedings of ICISC

Secure, Scalable, and Fine-grained Data Access Control
2010, pages 534-542.