ATTRIBUTE-BASED MULTI-DIMENSION SCALABLE ACCESS CONTROL FOR SOCIAL MEDIA SHARING

Changsha Ma, Zhisheng Yan and Chang Wen Chen

Dept. of Comp. Sci. and Eng., State Univ. of New York at Buffalo, Buffalo, NY, 14260, USA
changsha@buffalo.edu, zyan3@buffalo.edu, chencw@buffalo.edu

ABSTRACT

Social media sharing is one of the most popular social interactions in online social networks (OSNs). Due to the diverse networking conditions and various privacy requirements of OSN users, scalable media sharing has become a promising paradigm. It allows a media data distributor to share a media content of different qualities with different data consumers. To guarantee user privacy in scalable media sharing, it is essential to design an effective scalable media access control (SMAC) mechanism. However, all existing schemes cannot support scalable media streams with more than two dimensions, which significantly limits the flexibility of OSN services. In this paper, we present the first multi-dimension SMAC (MD-SMAC) system for social media sharing, based on the proposed scalable ciphertext policy attribute-based encryption (SCP-ABE) algorithm. In the MD-SMAC system, secure and reliable access control can be performed on multi-dimension scalable media streams according to data consumers' attributes. Through the security analysis, we prove the security and reliability of the MD-SMAC system. We also conduct experiments on mobile devices to demonstrate the computation efficiency of the proposed system.

Index Terms— social media sharing, SCP-ABE, multi-dimension scalable access control, privacy

1. INTRODUCTION

Social media sharing has recently become an extremely popular social interaction in online social networks (OSNs). According to a recent survey [1], more than one billion users are sharing videos on YouTube, and they can upload on average 300 hours of video contents per minute. With such a large volume of users and media contents on modern OSNs, traditional paradigm of media sharing, which creates a single media stream for a media content, may not be able to provide the satisfactory user experience. First, transmitting a single stream without considering the heterogenous network environment of data consumers would easily cause lost or incorrect data reception, especially in mobile networks with unstable network conditions. The damaged reception will then result in significant degradation on the media consumption experience of data consumers. Second, adopting a single media stream makes it impossible to configure multiple levels of access privileges on the same content. This is undesirable in OSNs since users usually share various social relationships and have diverse privacy requirements.

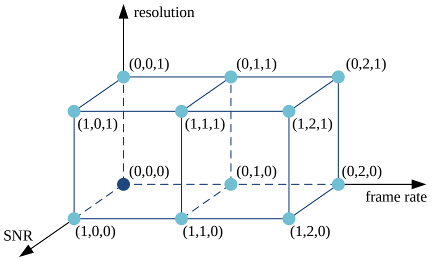


Fig. 1. 2-by-3-by-2 scalable media data structure

Recently, scalable media sharing, a new paradigm of social media sharing, is receiving increasing attention. In scalable media sharing, a data distributor uploads and shares a media content of different qualities with different data consumers [2, 3]. Specifically, the media stream is encoded into a base layer providing the basic quality and enhancement layers enhancing the quality in terms of frame rate, resolution, and signal-noise-ratio (SNR) [4]. For example, we show the data structure of a 2-by-3-by-2 scalable media stream in Fig. 1. With the base layer denoted by (0,0,0), a consumer can enjoy the basic quality with the lowest SNR, frame rate, and resolution. By receiving two more enhancement layers denoted by (1,0,0) and (1,0,1), the consumer could enjoy better quality with a higher SNR and resolution. Under such a data structure, scalable media sharing can effectively adjust the transmitted media stream according to the heterogenous network environment, which improves the media consumption experience. In addition, a data distributor can configure an access privilege for each level of media quality, which allows various levels of access privileges on the same content.

In order to manage the access behaviors of data consumers in scalable media sharing, it is essential to develop a proper scalable media access control (SMAC) mechanism. SMAC usually assigns an access key to each media layer in the scalable media stream, and then distributes the access keys to the data consumers according to their access privileges. Although

This research is supported in part by the U.S. National Science Foundation (NSF) under Grants 0915842 and 1405594; and in part by the National Natural Science Foundation of China under Grant 61550110244.

some SMAC schemes [2, 3, 5, 6] have been proposed based on this principle, none of them can support media streams with more than two dimensions. This constraint significantly limits the services provided by scalable media sharing systems. The goal of this research is to design a SMAC scheme supporting general multi-dimension scalable media streams. However, due to the large number of data consumers in OSNs, we are facing two non-trivial challenges: 1) how to ensure the cost of key distribution is affordable for data distributors who are usually equipped with resource-limited mobile devices; 2) how to securely and reliably distribute the access keys to data consumers with various access privileges.

To tackle these challenges, we propose a scalable ciphertext policy attribute-based encryption (SCP-ABE) algorithm, which is able to efficiently encrypt multi-dimension scalable data. Based on this algorithm, we design the multi-dimension scalable media access control (MD-SMAC) system for scalable social media sharing. In the MD-SMAC system, the access keys created for multi-dimension scalable media streams are encrypted under SCP-ABE according to the corresponding access policies. The access keys can only be decrypted by the data consumers if their attributes satisfy the corresponding access policies. A data consumer with a higher access privilege owns more desirable attributes and thus can decrypt more access keys. Therefore, by storing the SCP-ABE ciphertext on the OSN server, a data distributor can securely and reliably delegate the server to distribute the access keys to the data consumers. This also shifts the key distribution cost to the powerful servers. Through formal security analysis, we prove the security and reliability of the proposed MD-SMAC system. Furthermore, we conduct experiments on mobile devices to demonstrate its computation efficiency.

To summarize, the contributions of this research are: 1) a novel SCP-ABE algorithm that is able to efficiently encrypt multi-dimension scalable data (Section 3); 2) the first MD-SMAC system that supports secure and reliable access control for multi-dimension scalable social media stream (Section 4).

2. RELATED WORK

In scalable media sharing, key distribution cost is relatively high due to two reasons. First, each data consumer may need multiple access keys to decrypt a media stream, since the stream could be composed of multiple media layers. Second, the large number of data consumers in OSNs can result in numerous access keys. Suppose the media stream is encoded into M layers, the number of access keys distributed among N consumers is as large as $O(MN)$.

In order to reduce the cost of key distribution, some related works propose to send only one access key to each data consumer. Specifically, the access key is divided into N_s key segments in such a way that the segments of lower-level access keys can be generated from the corresponding segments of higher-level access keys through one-way hash chains [6]. To prevent degradation of the security level, the length of the access key should be $N_s - 1$ times longer than the original one. Therefore, such a strategy can reduce the key distribution cost to $O(N_sN)$. From the results shown in [5], the best performance of a secure scheme needs $N_s = mn$ key segments for the m-by-n-by-k ($k = max(m, n, k)$) scalable media data structure. Thus, the key distribution cost is reduced from $O(MN)$ to $O(mnN)$. In OSNs, however, N is much larger than M, which indicates the key distribution cost of such a strategy is still too large to be affordable by data distributors.

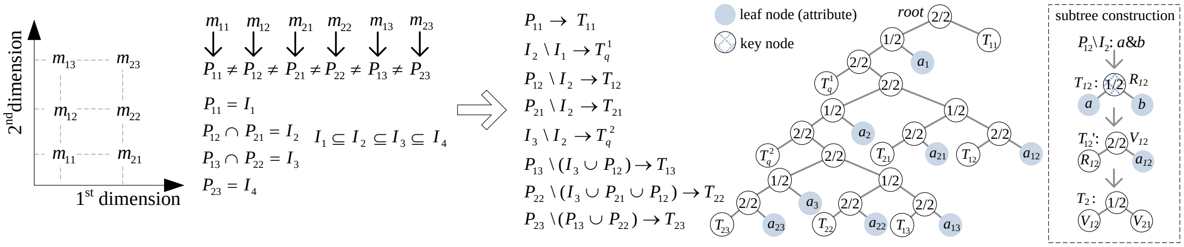
In this research, we propose to release the key distribution from the data distributor side, and delegate it to the resourceful server side. Since the server may not be trusted by the data distributors, such a mechanism should keep the access keys confidential to the server. In addition, the mechanism should guarantee that only the data consumers with the desired access privileges can obtain the corresponding access keys. Only a few existing schemes adopt a similar strategy. In [2], a multi-message ciphertext policy attribute-based encryption (MCP-ABE) scheme is proposed to encrypt multiple messages into one ciphertext according to the attribute-based access policies. Then a data distributor can set a scalable access policy for multiple access keys and encrypt them under MCP-ABE. The ciphertext can be securely stored on the server, and distributed to data consumers by the server. Whether or not a consumer can decrypt the access keys, and how many access keys the consumer can decrypt, are determined by the individual's attributes. Despite the secure and reliable key distribution, this scheme only supports one-dimension scalability, and hence is not able to provide multi-dimension scalable access control. Similarly, an algorithm that is able to encrypt two-dimension (2D) scalable data based on the 2D scalable access policies is proposed in [3]. However, it cannot encrypt general scalable media stream with more than two dimensions. To solve the issue, we instead propose the SCP-ABE algorithm to encrypt multi-dimension scalable data, and accordingly design the MD-SMAC system where secure and reliable access control can be performed on MD scalable social media streams based on data consumers' diverse attributes.

3. THE SCP-ABE ALGORITHM

Based on the CP-ABE algorithm [8], we propose the SCP-ABE algorithm to encrypt multi-dimension scalable data according to the corresponding attribute-based access policies. Specifically, SCP-ABE is composed of six sub-algorithms including system setup, access tree construction, encryption, user key generation, delegation, and decryption. In this section, we first introduce the preliminary, i.e. the bilinear map, and then proceed to describe each sub-algorithm.

3.1. Preliminary

Similar as CP-ABE, the proposed SCP-ABE algorithm is built based on the bilinear map. Let G_0 and G_1 be two multiplicative cyclic groups of prime order p. Let g be a generator of G_0 and e be a bilinear map, $e : G_0 \times G_0 \rightarrow G_1$. Then e has the following properties:


2nd dimension
1st dimension
m_{13} m_{23}
m_{12} m_{22}
m_{11} m_{21}
m_{11} m_{12} m_{21} m_{22} m_{13} m_{23}
$P_{11} \neq P_{12} \neq P_{21} \neq P_{22} \neq P_{13} \neq P_{23}$
$P_{11} = I_1$
$P_{12} \cap P_{21} = I_2$
$P_{13} \cap P_{22} = I_3$
$P_{23} = I_4$
$I_1 \subseteq I_2 \subseteq I_3 \subseteq I_4$
$P_{11} \rightarrow T_{11}$
$I_2 \setminus I_1 \rightarrow T_q^1$
$P_{12} \setminus I_2 \rightarrow T_{12}$
$P_{21} \setminus I_2 \rightarrow T_{21}$
$I_3 \setminus I_2 \rightarrow T_q^2$
$P_{13} \setminus (I_3 \cup P_{12}) \rightarrow T_{13}$
$P_{22} \setminus (I_3 \cup P_{21} \cup P_{12}) \rightarrow T_{22}$
$P_{23} \setminus (P_{13} \cup P_{22}) \rightarrow T_{23}$
leaf node (attribute)
key node
root
2/2
1/2
T_{11}
a_1
T_q^1
a_2
T_q^2
T_{21}
a_{21}
T_{12}
a_{12}
a_3
T_{23}
a_{23}
T_{22}
a_{22}
T_{13}
a_{13}
subtree construction
$P_{12} \setminus I_2$: $a\&b$
T_{12}: R_{12}
a b
T_{12}': V_{12}
R_{12} a_{12}
T_2:
V_{12} V_{21}

(a) (2D) scalable data structure & access policy

(b) access tree construction

Fig. 2. An indication of SCP-ABE access tree construction

- *Bilinearity*: for all $u, v \in G_0$ and $a, b \in Z_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- *Non-degeneracy*: $e(g, g) \neq 1$.

3.2. System Setup

The setup algorithm chooses a bilinear group G_0 of prime order p with generator g, and two random exponents $\alpha, \beta \in Z_p$. The public key PK and the master key MK are then returned as: $\{PK = G_0, g, h = g^\beta, f = g^{1/\beta}, e(g,g)^\alpha\}$, $\{MK = \beta, g^\alpha\}$.

3.3. Access Tree Construction

The access tree construction algorithm builds a scalable access tree according to the attribute-based access policies. In SCP-ABE, the access policy associated with a higher level access privilege contains all the attributes in the access policy for a lower level access privilege. Suppose that the data is a 3-D scalable media stream, and the access policy of the media layer denoted by (i, j, k) $(i = 0, 1; j = 0, 1, 2; k = 0, 1$ for the example shown in Fig. 1) is P_{ijk}, then we have:

$$P_{ijk} \subseteq P_{pql}, i \leq p, j \leq q, k \leq l \tag{1}$$

Based on (1), we can further conclude that:

$$\begin{aligned} &P_{ijk} \subseteq P_{pql} \cap P_{mng}, \text{ if} \\ &i \leq min(p, m), j \leq min(q, n), k \leq min(l, g) \end{aligned} \tag{2}$$

For ease of presentation, we use the 2D-scalable (M-by-N) data structure to describe the algorithm. However, note that the process is applicable for multi-dimension scalable data structure. Suppose that the access policy corresponding to each data segment m_{ij} $(i = 1, 2, ..., M; j = 1, 2, ..., N)$ is P_{ij}. According to (1) and (2), we hence have $P_{11}(I_1) \subseteq P_{12} \cap P_{21}(I_2)$, $P_{12} \subseteq P_{13}$, and $P_{21} \subseteq P_{22}$ for the 2-D scalable data structure shown in Fig. 2. Furthermore, we also have $I_2 \subseteq P_{13} \cap P_{22}(I_3)$. Similarly, we have $I_3 \subseteq I_4$. Then the access tree T is constructed as follows.

a. To begin with, we define the *referee* for m_{ij} $(i = 1, 2, ..., M; j = 1, 2, ..., N)$ as the data segment(s) in the path from m_{ij} to m_{11} that is (are) nearest to m_{ij}. The referee for m_{11} is defined as itself. In addition, we divide the data segments into $M + N - 1$ groups according to their distances to m_{11} in the data structure. Let $U_i (i = 1, ..., M + N - 1)$ be the union access policy of members in group G_i, and I_i be the common access policy of members in group G_i. In the example shown in Fig. 2 (a), the referees of $m_{12}, m_{21}, m_{13}, m_{22}$, and m_{23} are (m_{11}), (m_{11}), (m_{12}), (m_{12}, m_{21}), and (m_{13}, m_{22}), respectively. There are four groups, i.e., $G_1 = (m_{11})$, $G_2 = (m_{12}, m_{21})$, $G_3 = (m_{13}, m_{22})$, and $G_4 = (m_{23})$.
b. Construct the sub-tree T_q^i $(i = 1, 2, ..., M + N - 3)$ according to $I_{i+1} \setminus I_i$.
c. Construct T_{11} according to P_{11}, and T_{ij} $(i = 2, ..., M | j = 1; j = 2, ..., N | i = 1)$ according to $P_{ij} \setminus (I_k \cup P'_{ij})$, where I_k is the common access policy of members in group G_k that P_{ij} belongs to, and P'_{ij} is the referee(s) of P_{ij}. The root R_{ij} of T_{ij} is called key node.
d. Let R_{ij} $(i = 2, ..., M | j = 1; j = 2, ..., N | i = 1)$ and an additional attribute a_{ij} be the children of the and gate; build the subtree T'_{ij}, whose root is V_{ij}.
e. Let all V_{ij} in the group G_i $(i = 2, 3, ..., M + N - 2)$ be the children of the $1/|G_i|$ gate; build the group subtree T_i. Let T_{M+N-1} be T'_{MN}, and T_1 be T_{11}.
f. The whole access tree is constructed group by group from bottom to up. Starting from $i = M + N - 2$, for group G_i, let an and gate be the mother of the root of T_i and an or gate whose children are a selected attribute a_i and the root of T_{i+1}. Then let the and gate and T_q^{i-1} be the children of a new and gate. Let the new constructed tree be T_i, and recursively do the previous steps until all groups are added in the tree. The root of the access tree will be an and gate.

Then we choose a polynomial p_x for each tree node x in the access tree $\mathcal{T}$ in a manner as: (1) From the bottom to up, set the degree d_x of the polynomial p_x to be $d_x = k_x - 1$, where k_x is the threshold value of node x; (2) For the root node $root$, choose a random $s \in Z_p$ and set $p_{root}(0) = s$, and randomly choose other points of polynomial p_{root}; (3) For any other node x, set $p_x(0) = p_{parent(x)}(index(x))$, and randomly choose other points of p_x. The access tree construction process for Fig. 2 (a) is summarized in Fig. 2 (b).

In the access tree structure, we have added some additional attributes except for the attributes in the access policies. Specifically, $a_i (i = 1, 2, ..., M + N - 2)$ is used to enable users to perform decryption starting from the level where G_i locates. For example, if the attributes of a user only conform to P_{11}, the user takes the user key corresponding to attributes in P_{11} and a_1 as input of the decryption algorithm, and performs decryption operation starting from the level that G_1 locates. If the user has attributes that conform to P_{12}, he or she can instead perform decryption starting from the level that G_2 locates. Furthermore, $a_{ij}(i = 2, ..., M|j = 1; j = 2, ..., N|i = 1)$ is used to guarantee the uniqueness of each key node, which is essential to enforce the access policies. For example, without a_{12} and a_{21}, $p_{R_{12}}(0)$ will be equal to $p_{R_{21}}(0)$ since their mother node has the degree of zero. As a result, a user who can access $p_{R_{12}}(0)$ would also be able to access $p_{R_{21}}(0)$ even without the required attributes.

3.4. Encryption

The encryption algorithm encrypts the data segments. Let L be the set of leaf nodes in $\mathcal{T}$, K be the set of key nodes V_{ij} $(i = 1, 2, ..., M; j = 1, 2, ..., N)$, and m_{ij} be the corresponding data segment, the ciphertext is given as

$$CT = (\mathcal{T}, \forall i \in L : E_i = g^{p_i(0)}, E_i' = H(att(i))^{p_i(0)}$$

$$\forall R_{ij} \in K : \tilde{C_{ij}} = m_{ij} e(g,g)^{\alpha(p_{R_{ij}}(0)+s)}, C_{ij} = h^{p_{R_{ij}}(0)+s})$$

3.5. User Key Generation

Taking a set of attributes S as input, the user key generation algorithm outputs a SCP-ABE user key. Specifically, the algorithm selects a random $r \in Z_p$ and random $r_x \in Z_p$ for every attribute x in S. Note that S always includes a_i $(i = 1, 2, ..., M + N - 2)$, and a_{ij} $(i = 2, ..., M|j = 1; j = 2, ..., N|i = 1)$. The user secret key is computed as

$$SK = \{D = g^{(\alpha+r)/\beta},$$
$$\forall x \in S : D_x = g^r \cdot H(attri_x)^{r_x}, D_x' = g^{r_x}\}$$

3.6. Delegation

Given a user secret key SK with the attribute set S, the delegation algorithm creates a new user secret key $\tilde{SK}$ with the attribute set $\tilde{S} \subseteq S$. Specifically, the algorithm selects a random number $\tilde{r} \in Z_p$ and also $\tilde{r_x} \in Z_p$, $\forall x \in \tilde{S}$. Then $\tilde{SK}$ is created as

$$\tilde{SK} = \{\tilde{D} = Df^{\tilde{r}},$$
$$\forall x \in \tilde{S} : \tilde{D_x} = D_x g^{\tilde{r}} \cdot H(attri_x)^{\tilde{r_x}}, \tilde{D_x}' = D_x' g^{\tilde{r_x}}\}$$

3.7. Decryption

The decryption algorithm employs three types of input, i.e., several encrypted data segments, a secret key SK of a user, and the public key PK. Suppose that the highest level of data segment a user can obtain is m_{pq} $(1 \leq p \leq M, 1 \leq q \leq N)$. Starting from T_{pq}, the user needs to perform the following computation for each leaf node x in T_{pq} and in the path from R_{pq} to the root.

$$\begin{aligned} F_x &= \frac{e(D_x, E_x)}{e(D_x', E_x')} \\ &= \frac{e(g^r \cdot H(attri_x)^{r_x}, g^{p_x(0)})}{e(g^{r_x}, H(attri_x)^{p_x(0)})} \\ &= \frac{e(g^r, g^{p_x(0)}) \cdot e(H(attri_x)^{r_x}, g^{p_x(0)})}{e(g^{r_x}, H(attri_x)^{p_x(0)})} \\ &= e(g,g)^{rp_x(0)} \end{aligned}$$

Then the user recursively computes the corresponding values of non-leaf nodes including the key nodes, in a bottom-up manner using polynomial interpolation technique [8], until it reaches the root and obtains:

$$F_{root} = e(g,g)^{rp_{root(0)}} = e(g,g)^{rs}$$

In this process, the user also obtains:

$$F_{R_{ij}} = e(g,g)^{rp_{R_{ij}}(0)} (i = 1, 2, ..., p, j = 1, 2, ..., q)$$

Furthermore, the user compute $K_{ij} = F_{R_{ij}} \cdot F_{root} = e(g,g)^{r(p_{R_{ij}}(0)+s)}$. Each data segment m_{ij} $(i = 1, 2, ..., p, j = 1, 2, ..., q)$ can then be decrypted as following.

$$\begin{aligned} & \frac{\tilde{C_{ij}}}{e(C_{ij}, D)/K_{ij}} \\ &= \frac{m_{ij} e(g,g)^{\alpha(p_{R_{ij}}(0)+s)}}{e(h^{p_{R_{ij}}(0)+s}, g^{(\alpha+r)/\beta})/e(g,g)^{r(p_{R_{ij}}(0)+s)}} \\ &= \frac{m_{ij} e(g,g)^{(\alpha+r)(p_{R_{ij}}(0)+s)}}{e(g^{\beta(p_{R_{ij}}(0)+s)}, g^{(\alpha+r)/\beta})} \\ &= \frac{m_{ij} e(g,g)^{(\alpha+r)(p_{R_{ij}}(0)+s)}}{e(g,g)^{\beta(p_{R_{ij}}(0)+s)\cdot(\alpha+r)/\beta}} \\ &= m_{ij} \end{aligned}$$

4. THE PROPOSED MD-SMAC SYSTEM

Based on the proposed SCP-ABE algorithm, we design the MD-SMAC system to provide access control for multi-dimension scalable media sharing. As shown in Fig. 3, there are four communication parties in the system, i.e. the data distributor, the data consumer, the OSN sever, and the attribute authority (AA) who authorizes attributes and assigns SCP-ABE user secret keys. In practice, the AA can be run by a trusted third party, or alternatively by the data distributor. In this section, we first present the media sharing process in the MD-SMAC system, and then analyze how secure and reliable access control can be achieved in the system.

4.1. Media Sharing in MD-SMAC

The entire operation process of MD-SMAC is illustrated in Fig. 3. To begin with, a data distributor uploads and encodes the social media content into a multi-dimension scalable media stream composed of multiple media layers. For each media layer, the distributor configures the corresponding access policy by selecting the desirable attributes of data consumers. The selection of attributes should ensure that the access policies satisfy (1) and (2). The distributor also chooses an access key for each media layer, and encrypts the layer by the access key under a standard encryption algorithm such as DES and AES. The encrypted media stream is hence generated.

Then the data distributor cooperates with the AA to set up the SCP-ABE parameters by running the SCP-ABE setup algorithm. The distributor proceeds to run the SCP-ABE access tree construction algorithm to build the SCP-ABE access tree, and the SCP-ABE encryption algorithm to encrypt the access keys under the access tree. The resulted ciphertext and the encrypted media stream are together sent to the OSN servers.

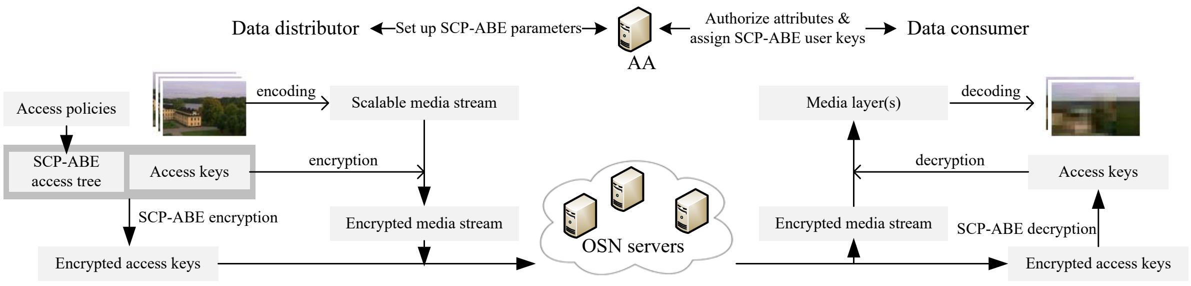


Fig. 3. Multi-Dimensional Scalable Social Media Sharing And Access Control

The encrypted media stream and access keys stored on the servers are accessible for all data consumers. In order to decrypt the media stream, a data consumer needs to first decrypt the access keys by running the SCP-ABE decryption algorithm using his or her SCP-ABE user key assigned by the AA. To generate the SCP-ABE user key for the consumer, the AA first authenticates the attributes of the consumer, and runs the SCP-ABE user key generation algorithm based on the authenticated attributes. In the case that the AA is not reachable for all consumers, the SCP-ABE user keys can also be delegated by other consumers with higher or equal access privileges by running the SCP-ABE delegation algorithm. Finally, the media stream can be decrypted by the decrypted access keys.

4.2. Secure and Reliable Access Control

In this section, we prove that the proposed MD-SMAC system provides secure access control. This means the access keys are securely encrypted, so that a data consumer without an access privilege cannot decrypt the access keys. Furthermore, we prove that the MD-SMAC system provides reliable access control, which indicates that a data consumer cannot obtain the access keys that require higher access privileges than what the consumer owns. Since the main security challenge in SCP-ABE is to resist collusion attacks as discussed in previous ABE schemes [7, 8], we prove the system security and reliability based on the analysis of collusion resistance.

Security proof: Using the example shown in Fig. 2, suppose P_{11} is b_0 *and* b_1. A data consumer with the specific access privilege thus should have both b_0 and b_1. Now we analyze whether two users who have no access privilege but together own b_0 and b_1 can obtain m_{11} (the access key) by collusion. Specifically, we assume that one user owns attribute b_1, while the other user just owns attribute b_0. With b_1, the first user is assigned by the AA with the secret key shown in (3). Therefore, the first user is able to compute $F_{b_1} = e(g,g)^{r_a p_{b_1}(0)}$ for the leaf node b_1 in T_{11}. With b_0, the second user owns the secret key as in (4), and hence can compute $F_{b_0} = e(g,g)^{r_b p_{b_0}(0)}$ for the leaf node b_0 in T_{11}.

$$\begin{aligned} SK_a = \{ & D_a = g^{(\alpha+r_a)/\beta}, \\ & D_1 = g^{r_a} \cdot H(b_1)^{r_1}, D'_1 = g^{r_1}\} \end{aligned} \tag{3}$$

$$\begin{aligned} SK_b = \{ & D_b = g^{(\alpha+r_b)/\beta}, \\ & D_2 = g^{r_b} \cdot H(b_0)^{r_2}, D'_2 = g^{r_2}\} \end{aligned} \tag{4}$$

The decryption of m_{11} requires K_{11}, which is computed from $F_{R_{11}}$ and F_{root}. However, the computation of $F_{R_{11}}$ either needs F_{b_1} combined with $e(g,g)^{r_a p_{b_0}(0)}$ or F_{b_0} combined with $e(g,g)^{r_b p_{b_1}(0)}$. Therefore, the collusion through combining F_{b_1} and F_{b_0} will not be able to decrypt m_{11} .

Reliability proof: With Fig. 2, now we suppose one user can access m_{13}, m_{12}, and m_{11}, while the other user can access m_{21} and m_{11}. We analyze whether the two users can access m_{23} and all the lower level access keys through collusion. Since the decryption of an access key needs the corresponding attributes, the success of such a collusion would require that the attributes of the two users satisfy the access policy of the target access key. In other words, the derivation of m_{23} from m_{13} and m_{21} needs $P_{23} \subseteq P_{12} \cup P_{21}$. Besides, we should have $P_{12} \cup P_{21} \subseteq P_{22} \subseteq P_{23}$. Then the success of collusion requires $P_{23} = P_{12} \cup P_{21}$, which means there is no T'_{22} or T'_{23} in the access tree and accordingly no K_{22} or K_{23} is used to encrypt the media stream. However, to avoid that m_{22} and m_{23} are transmitted in plain text, they have to be encrypted. Therefore, such a collusion is also infeasible.

5. PERFORMANCE EVALUATION

Although MD-SMAC system securely and reliably shifts the access key distribution cost to the server, it introduce certain computation on user side (distributors and consumers). In this section, we evaluate the computation cost of the MD-SMAC system to demonstrate that MD-SMAC can be efficiently implemented on typical mobile equipments.

5.1. Efficiency of Applying SCP-ABE

We first compare the cost of the proposed SCP-ABE algorithm and the classical CP-ABE algorithm [8] in the system.

Specifically, we assume the media stream can be scalable from 1 to 3 dimensions. In addition, the access policy corresponding to a layer has at least k $(k = 1, 2)$ more attributes than that corresponding to a lower-level layer. By varying the total number of media layers in the stream, we show the number of attributes to be computed by SCP-ABE and CP-ABE for scalable access control. Note that different number of attributes may be needed for a given number of media layers under different scalability dimensions. We therefore measure the average number of attributes (over 1D to 3D scalability) as well as the upper bound and the lower bound.

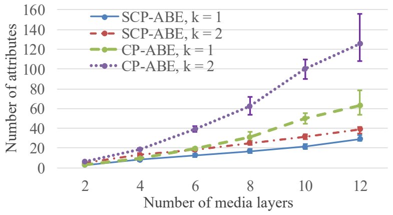


Fig. 4. Comparison between SCP-ABE and CP-ABE

The results are presented in Fig. 4. We can see that the computation performance of SCP-ABE is superior to CP-ABE, and the superiority becomes more significant as the number of media layers increases. This can be explained as follows. The access policy with a higher level access key should contain all the attributes in the access policy that is associated with a lower level access key. If we apply an existing ABE algorithm to encrypt each access key based on the corresponding access policy respectively, the computation power will be wasted for repeatedly computing on the overlapped attributes. As the number of required access keys increases, the redundancy will become more significant. However, such redundant computations can be avoided in the proposed SCP-ABE algorithm, since the access tree in SCP-ABE is constructed in a scalable manner.

5.2. Computation Cost on Mobile Devices

We now present the experimental results of the computation time on the user side. In particular, we implement the user side algorithms on two mobile devices, i.e., Google Nexus 4 (1.5GHz quad-core Snapdragon S4 Pro with Krait CPUs, Android 5.0), and Lenovo T430 (2.6GHz Intel i5 CPU, Ubuntu 14.04), and then measure their computation performance.

To begin with, the computation cost of distributor side mainly comes from SCP-ABE encryption, which involves pairings, exponentiations, and multiplications computations on G_0 and G_1. The computation cost linearly increases with the number of attributes. We measure the average computation time per attribute on the smart phone and on the laptop[1], and the results are 60ms and 35ms, respectively.

In terms of consumer side computation, it mainly includes pairings and multiplications computations on G_1 resulted from SCP-ABE decryption. The computation cost of SCP-ABE decryption varies with different access tree structures. For simplification, we fix the access tree structure and thus are able to consider the computation cost to grow with the number of attributes in a linear way, and test the average computation time per ten attributes on devices. The results are 48ms on smart phone and 25ms on laptop.

Fig. 5 shows the computation time on the user side with various number of attributes in the access policies. We can see that the computation performance of the MD-SMAC system on the consumer side is less than one second even if the number of attributes is large. This satisfies the requirement of general mobile applications [9]. Although the computation cost on the distributor side can reach a few seconds when the number of attributes is relatively large, it is still computationally acceptable. This is because the distributor only needs to encrypt the access keys once before sharing all the contents. A new SCP-ABE encryption is needed only when the access keys are leaked or the access policies are changed. Therefore, we conclude that MD-SMAC can be efficiently implemented in real-world systems.

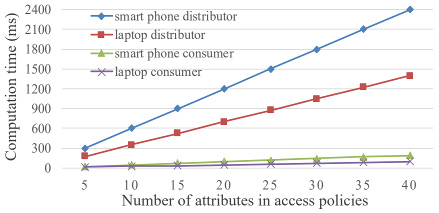


Fig. 5. Computation cost on mobile devices

6. CONCLUSION

In this paper, we have designed a MD-SMAC system for social media sharing based on the proposed SCP-ABE algorithm. We have proved that MD-SMAC achieves secure and reliable access control on social media contents encoded in multi-dimension scalable format. Through the mobile terminal implementation, we have also demonstrated the computation efficiency of the system.

7. REFERENCES

[1] http://expandedramblings.com/index.php/youtube-statistics/

[2] Wu, Y.; Zhuo, W.; Deng, R., "Attribute-Based Access to Scalable Media in Cloud-Assisted Content Sharing Networks," *IEEE Trans. Multimedia,* vol.15, no.4, pp.778-788, 2013.

[3] Ma, C.; Chen, C. W., "Secure media sharing in the cloud: Two-dimensional-scalable access control and comprehensive key management," *IEEE ICME*, 2014.

[4] Schwarz, H.; Marpe, D.; Wiegand, T., "Overview of the Scalable Video Coding Extension of the H.264/AVC S-tandard," *IEEE Trans. Circuits and Syst. Video Technol.*, vol.17, no.9, pp.1103-1120, 2007.

[5] Zhu, X.; Chen, C. W., "A collusion resilient key management scheme for multi-dimensional scalable media access control," *IEEE ICIP*, 2011.

[6] Zhu, B.B.; Feng, M.; Li, S., "An efficient key scheme for layered access control of MPEG-4 FGS video," *IEEE ICME*, pp. 443-446, 2004.

[7] Goyal, V.; Pandey, O.; Sahai, A.; Waters B., "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data," *ACM CCS*, pp. 89-98, 2006.

[8] Bethencourt, J.; Sahai, A.; Waters, B., "Ciphertext-Policy Attribute-Based Encryption," *IEEE Symposium on Security and Privacy*, pp. 321-334, 2007.

[9] http://www.infoq.com/news/2014/03/mobile_app_performance_benchmark

[1] As in [8], operations are conducted using a 160-bit elliptic curve group based on the curve $y^2 = x^3 + x$ over 512-bit finite field.