Efficient Access Control in Multimedia Social Networks

Amit sachan and Sabu Emmanuel

Abstract Multimedia social networks (MMSNs) have provided a convenient way to share multimedia contents such as images, videos, blogs, etc. Contents shared by a person can be easily accessed by anybody else over the Internet. However, due to various privacy, security, and legal concerns people often want to selectively share the contents only with their friends, family, colleagues, etc. Access control mechanisms play an important role in this situation. With access control mechanisms one can decide the persons who can access a shared content and who cannot. But continuously growing content uploads and accesses, fine grained access control requirements (e.g. different access control parameters for different parts in a picture), and specific access control requirements for multimedia contents can make the time complexity of access control to be very large. So, it is important to study an efficient access control mechanism suitable for MMSNs. In this chapter we present an efficient bit-vector transform based access control mechanism for MMSNs. The proposed approach is also compatible with other requirements of MMSNs, such as access rights modification, content deletion, etc. Mathematical analysis and experimental results show the effectiveness and efficiency of our proposed approach.

1 Introduction

Over the past few years with the advent of web 2.0 technologies, social media such as blogs, images, videos, etc. have become much popular due to their wide reach and easy accessibility. Social media depending on the type of media is generally shared

Amit Sachan
School of computer engineering, Nanyang Technological University, Singapore
e-mail: amit0009@ntu.edu.sg

Sabu Emmanuel
School of computer engineering, Nanyang Technological University, Singapore
e-mail: asemmanuel@ntu.edu.sg

over various types of social networks. In particular, the multimedia social networks (MMSNs) such as YouTube[5], Facebook[6] and, Flickr[7] etc. which allow sharing of multimedia contents, such as images, videos, audios, etc. have become much popular. MMSNs allow easy sharing and spread of multimedia contents. People can easily share any multimedia contents from anywhere in the world and the shared contents can be accessed conveniently anywhere.

Although people can easily share their multimedia contents over MMSNs, there are often various privacy, security, and legal concerns[1][2][3][4] due to online sharing of personal multimedia contents. Privacy and security concerns are about unwillingness to disclose contents to a particular group of people, unintentional disclosure of identity and sensitive information, and threat of content modification for unethical purposes. For example, a cancer patient may wish to share the pictures related to his disease only with his doctors and family members[2]. Or due to the threat of content modification, a person may wish to share his pictures only with trustworthy friends. Legal concerns are about unsuitability of content to a group of persons and restrictions associated with copyrighted contents. For example, due to legal restrictions a person may wish to share a video containing violent scenes with the persons above some specific age only. Or due to copyright issues, some contents may not be shared to the users in some countries. All privacy, security and legal issues may socially and economically harm a person. Therefore, people usually want to selectively share their multimedia contents.

Access control mechanisms[9][10][18] provide one way to selectively share the contents in MMSNs. By access control mechanisms a person can define the credentials[21] required by anybody else to access the contents shared by the person. Credentials are in the form of social relationship parameters such as friendship level, trust level, age, allowed countries, etc. [10][23][27]. Similarly, each individual has credentials associated with his profile. Access control mechanisms work by verifying the credentials of an individual accessing a content with the credentials required to access the content. If an individual has sufficient credentials to access a content, then only he is allowed to access the content.

Access control mechanisms are simple and only require comparison of the credentials. But large number of contents upload and accesses make the access control difficult in MMSNs. For instance, according to the statistics provided by Youtube [5] currently every minute about 24 hours of video is uploaded on Youtube and 2 billion videos are watched everyday on Youtube. Furthermore, large number of users (according to Facebook[6] there are more than 500 million active users are on Facebook) , and rapid growth rate of users over MMSNs(according to Nielsen online[8], facebook recorded a growth rate of 228% over the period of February, 2008 to February, 2009) makes the task of access control more difficult.

Also, as suggested in [15][20][24], users may require fine grained access control for their shared contents. For example, a user may wish to protect some words in his blog, or specific parts in a picture from specific persons. Another specific requirement for multimedia contents may be to allow the users to specify several discrete access control parameters instead of only binary access parameters (in which someone either can or cannot access a content). For example, an access value of 0.5 may refer to access to a low resolution version of the original picture. To satisfy these multimedia specific requirements, we need to treat each specific part of a content (or contents with different access parameters) as an independent content.

Thus, due to the large growth rate, large number of content uploads and accesses, fine grained access control requirements, and multimedia specific access control requirements, the access control time may grow several times in MMSNs. This necessitates an efficient access control mechanism suitable for MMSNs. However, most of the existing works on access control in social networks focus on new social network models[18], policy description[21][23], and privacy and security issues[3][24] in social networks. Authors in reference [16] discussed the issue of complexity of access control in social networks and requirements of an efficient access control solution, and proposed an efficient access control scheme. However, as we discuss in section 2, the proposed efficient control solutions is user based (no access rights are associated with the contents so if a person is allowed to access then the person can access all the contents) not the content based, which may not be suitable for access control in present day MMSNs. To the best of our knowledge there is no existing work that discusses efficient content based efficient access control mechanism (a preliminary version of this work appeared in [22]).

In this chapter, we focus on designing an efficient access control mechanism suitable for access control in present day MMSNs. For this purpose, we propose a bit-vector transform[19] based access control method. In the proposed approach we model the contents, content access rights and users in MMSNs for efficient access control. In addition, we also propose an improvement to make the proposed method more suitable when multiple contents share the identical access rights. The proposed approach is also able to provide the functionalities required in MMSNs such as multimedia content deletion and access rights modification using data in the bit-vector transform domain. The proposed access control mechanism is efficient due to efficient representation of data in the bit-vector transform domain and efficient operations on data after the bit-vector transform. Mathematical performance analysis and experimental results show effectiveness and efficiency of our proposed mechanism for MMSNs.

The rest of this chapter is organized as follows. Section 2 discusses the related works. In section 3, we present the access control model. In section 4 our proposed access rights organization based access control mechanism is discussed. In section 5 we propose a technique to make the access control more efficient when multiple contents share the same access rights. Performance analysis is in section 6. Finally, the chapter is concluded in section 7.

2 Related Works

In this section we discuss about various access control mechanisms in social networks, and requirement of efficient access control in social networks.

2.1 Access Control in Social Networks

In literature, a variety of access control mechanisms have been proposed for the social networks to address various aspects, such as new models of access control[9][10][18][21], security and privacy [3][24][25], and policy description[23], etc. have been proposed in literature.

Carminati et al. in [9][10] described the access control policies using friend of a friend(FOAF) [21] scripting language. FOAF provides vocabulary to manage the access control based on the relationship between the individuals. Each relationship can be defined on a scale from 0 to a maximum value, called the domain of relationship(D)[18]. For example, the level of friendship can be defined using the values between 0 and 100. The authors in [10] defined access rule for an object with ID *oid* using the tuple (*oid*, *cset*), where *cset* is a set of conditions $\{cond_1, cond_2,...cond_m\}$ that must be fulfilled by an individual to access the object *oid*. All the conditions must be fulfilled by an individual, who wants to access *oid*. There can be more than one rule defined for an object. In this case all the rules need to be verified one by one until a valid proof, if any, is obtained. The conditions in the access rules can be trust level, friendship level, country, name of persons, etc. [10][23][27].

In [18] authors proposed a digital rights management (DRM) [12][13][14] based solution of doing the access control in social networks. In the proposed mechanism digital contents are considered as physical good. The authors proposed solutions similar to that exist in physical world. For example, in physical world a book can be rendered to only one person at a time so allow access of an e-book only to one person at a time. The proposed method is suitable for copyrighted contents. However, it may limit the distribution capability of Internet.

Ianella in [23] emphasized the need of inter operability between the contents shared between different social networks so that users need not to share their contents multiple times. But it requires same access control parameters for different social networks so that the same policy may be applied. It may be difficult as presently different existing social networks have different access policies. Furthermore, due to privacy and business related reasons social network providers may not agree for inter-operable framework.

Various authors have discussed about different types of privacy concerns in social networks and access control mechanisms to deal with the privacy concerns. Beato et al. in [24] discussed about privacy threats from the social network providers. The authors proposed an access control model that is not regulated by the social network provider but it is under the control of users. But in order to use the proposed system users must know technical details of access control mechanism and cryptographic protocols, which may be very difficult practically. Loukides and Gkoulalas-Divanis in [3] discussed privacy concerns of a user resulting from other users in the social network and social network provider. The main privacy concerns due to other users in social networks according to authors are identity disclosure, and content disclosure. Access control mechanisms mainly deal with the privacy concerns due to content disclosure.

Authors in [11] and [26] argued the use of decentralized approach to online social networking to handle privacy and inter-operability issues in present day social networks. Using decentralized system users can privately share their own data by maintaining their data on their own trusted servers. The data of other users can be accessed using FOAF vocabulary[21]. But enforcement of decentralized access control policies may be difficult and inefficient in present day social networks due to relationship based access control in present day social networks[11]. Furthermore, although decentralized approach may be more effective in terms of privacy but it will lack a good user interface and interaction (e.g. in the form of status updates) between users as provided by present day social networks. Good user interface and interaction are some of the main reasons of popularity of present day social networks over other mediums of sharing the contents such as email. To tackle these problems, Baden et al. in [28] presented a social network model called Persona using public key cryptography and attribute based encryption. Persona uses a decentralized approach yet it is able to provide the interface and interaction as provided by present day social networks. For this purpose, it uses a group based content sharing, where users in a group can access any content shared by a user in the group for the group. However, the proposed mechanism may involve a large overheads due to cryptographic operations during formation of group, or removal of any member from the group. Furthermore, using decentralized approach users have to pay for storage and computations involved as opposed to free storage and computation power provided in present day social networks, which make them much popular.

Shehab et al. in [25] discussed the privacy infringement due to third party applications on the social networks. These application require user data but users often don't know how much data is being disclosed to the third party applications. So, authors proposed a finite state machine based approach for providing data access to the third party applications.

Although various access control mechanisms exist for different privacy requirements, in this work we mainly focus on access control mechanisms for content disclosure to other users in the social networks.

From the access control models studied in this section, we found that the access control model proposed by Carminati et al. in [9][10] is more suitable for current social networks. So, we use an extension of social network model proposed by Carminati et al. as the basis, which we discuss in section 3.

2.2 *Efficient Access Control*

Various authors recently have discussed the need of finer and efficient access control requirements[20][15][16][11] in various scenarios. In [20], Tootoonchian et al. proposed an inter-operability framework for different social networks. The motivation behind that is that users are generally on different social networks and moving a content from one social network to another social network is difficult. In such cases, due to the different policies in different social networks, finer and more number of access control parameters may be needed. Gates in [15] have stated four key requirements while designing an access control system for social networks. One of the requirements according to Gates is the fine grained access control i.e. access control mechanisms should also be able to manage the access control for fine grained details inside the object. For example, a user may wish to protect some words in his blog, or specific parts in a picture from specific persons.

Authors in [16] discussed the effect of fine grained access control on rising access control time complexity. In [16] authors proposed a trusted distance classifier based scheme to efficient access control. In the scheme, for each user profile, other users are classified into three categories viz. acceptance, attestation, and rejection. Users in the acceptance zone are accepted immediately and the users in the rejection zone are rejected immediately. Requests in attestation zone require additional authorization, which is done by attesters designated by the user. The proposed scheme in [16] can reduce the access control time as it eliminates the need of comparison of attributes associated with users accessing the contents with the attributes associated with the contents. But the scheme may not be suitable for access control in present day social networks because of the following two reasons. First, the proposed access control mechanism does the access control on the basis of users not the contents but practically users may have different access control requirements for different contents. Second, for the users in attestation zone manual attestation by designated attesters is required, which may cause significant delay and inconvenience to users. The bit-vector transform based method presented in this chapter does efficient access control and it is also suitable for present day social networks.

3 Model of Access Control

In this work, we model the system by modifying the access control model proposed by Carminati et al. in [9][10]. In the access control model in [9][10], an access rights vector is associated with each content. Let the access rights vector (R^c for content c) be given by: R^c=$\{r_1$, r_2,... ,$r_M\}$, where r_1, r_2,... , and r_M are M access rights. Each access right is in the form of an interval and has a domain of all possible allowed values. If the domain of the j^{th} access right is given by D_j then $r_j \in D_j$. Let the attribute vector of an individual accessing the content be defined as A=$\{A_1$, A_2,... , $A_M\}$, where A_1, A_2,... , and A_M are M attribute values. Each attribute in the attribute vector corresponds to the respective access right in the access right vector(R^c). Note that some attributes such as trust level, friendship level, etc. in the attribute vector of an individual are not fixed and dependent upon the profile individual is visiting. Such attributes need to be obtained first during the access control process

The individual can access the content c if $A_j \in r_j$, $\forall\ j \leq M$. When the profile of an individual I_1 is visited by another individual I_2 then all the contents to which I_2 is authorized to view are identified and displayed, as shown in example 1.

Example 1: Let an individual I_1 has 3 contents, c_1, c_2 and c_3 in his profile with access rights vector R^{c_1}, R^{c_2} and R^{c_3} given as (with access rights as friendship level(F),

trust level(T) and age(AG), each having domain 0 to 100):
R^{c_1}={F=(40, 100], T=(30, 80], AG=(0, 100]}
R^{c_2}={F=(24, 60], T=(63, 100], AG=(18, 100]}
R^{c_3}={F=(40, 80], T=(70, 100], AG=(16, 100]}

Access right vector R^{c_1} states that the content c_1 can be accessed by a person having friendship level between 40 and 100, trust level between 30 and 80, and age between 0 and 100. Similarly, R^{c_2} and R^{c_3} can be interpreted. Let an individual I_2 with the attribute vector A={50, 40, 25} visits the profile of I_1. Now, the attributes 50, 40, and 25 are within the ranges (40, 100], (30, 80], and (0, 100] respectively. Therefore I_2 can access c_1. But for c_2 and c_3, trust level=40 of the individual I_2 is not within the access rights ranges (63, 100] and (70, 100] respectively. So, I_2 cannot access the contents c_2 and c_3 and only c_1 will be displayed.

The model presented above uses only a single range for each access right but access rights in present day social networks may be discrete and may consist of multiple discrete ranges. For example, a user may allow access of a content in few different regions, which may not be represented with a single range. So, to reflect the existing social networks, instead of directly using the same model in [9][10], we use a social network model in which access rights can have multiple discrete ranges. However, in the rest of this chapter, for the sake of simplicity, we first present the case of all access rights with a single range and then provide the extensions required to accommodate the multiple access rights ranges.

4 Proposed Efficient Access Control Mechanism

In this section, we first present our proposed bit-vector transform based access rights organization and an access control mechanism using the data after the organization. Then we also propose the mechanisms(in the bit-vector transform domain) for deletion of contents from the profile and modification of access rights associated with a content.

4.1 Access Rights Organization

Let each content has M access rights associated with it. We organize the access rights associated with the contents into an M dimensional space. As shown in figure 1, each dimension corresponds to a particular access right and divided into several elementary ranges. Each elementary range has a bit-vector associated with it. The bit-vectors are string of bits '0' or '1'. Number of bits in the bit-vectors is equal to the number of contents in the profile of the user. Each bit in a bit-vector corresponds to a particular content. Thus, if a user has N content items in his profile then the length of bit-vectors will be N bits; and the n^{th} bit in the bit-vector will correspond to the n^{th} content. For example, figure 3(e) represents the dimension for friendship level(F) in example 1. There are 5 elementary ranges viz. 0 to 24, 24 to 40, 40 to 60, 60 to 80, and 80 to 100. Since there are 3 contents in example 1 so in figure 3(e) each elementary range has a 3 bits length bit-vector (shown in rectangular box) associated with it.

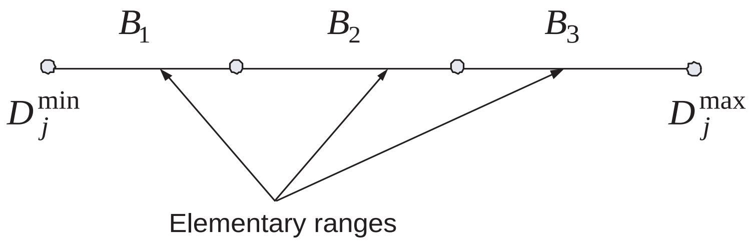

Fig. 1 Illustration of elementary ranges and bit-vectors

Initially, each access right's dimension is assumed to contain only one elementary range in the entire domain D_j= (D_j^{min}, D_j^{max}] of the j^{th} access right. D_j^{min} and D_j^{max} are the minimum and maximum values respectively in the j^{th} access right's domain. No bit-vector is assumed to be present for the initial elementary range(the first bit in a bit-vector is inserted when the first content is inserted). Two steps viz. access right's range insertion and modification of bit-vectors, are required for access rights' organization. In the first step, the range of access rights in the contents is inserted along the respective access right's dimension. In the second step, existing bit-vectors are modified by appending a new bit to them.

1) Access Right's Range Insertion: Let the range of the j^{th} access right for the n^{th} content be [a_j^n, b_j^n]. We call *AR_Insert*(a_j^n) and *AR_Insert*(b_j^n) for the insertion of (a_j^n, b_j^n]. The process *AR_Insert*(x_j) is as defined in algorithm 1. During the insertion process the point x_j is inserted in the numerical order if it was not already present along the j^{th} access right's dimension. If x_j is inserted then the same bit-vector as the elementary range in which it is inserted is assigned to both the elementary ranges formed after insertion. The same process is used for all M dimensions.

Figure 2 (a) shows the initial range of friendship dimension in example 1. To insert the access right's range (which is (40, 100]) in the first content initially algorithm 1 is called with the parameter x_j=40. Since x_j=40 was not present along the access right's range in figure 2 (a) so x_j=40 inserted in the numerical order between 0 and 100 as shown in figure 2 (b). Since there was no bit-vector present for initial elementary range (0, 100] so no bit-vector is assigned to newly formed elementary ranges viz. (0, 40] and (40, 100]. Next algorithm 1 is called with the parameter x_j=100. Since 100 is already present along the access right's dimension in figure 2 (b) so we need not to insert it again. The access right's dimension remains same as in figures 2 (b) and (c) after calling the algorithm with x_j=100.

Algorithm 1 $AR\_Insert(j,\, x_j)$

```
Input: $j$: A numerical value between 1 and $M$ do determine the access right dimension.
$x_j$: A numerical value to be inserted along the $j^{th}$ access right's dimension.
Output: Modified $j^{th}$ access right's dimension.
Procedure:
Search $x_j$ along the $j^{th}$ dimension.
if $x_j$ is already present along the $j^{th}$ access right's dimension then
    Exit.
else
    1. Insert $x_j$ in numerical order along the $j^{th}$ access right's dimension.
    2. Assign the same bit-vector as previous elementary range to both the newly generated ele-
    mentary ranges formed due to the insertion of $x_j$ (as in figures 3(b) after insertion of points
    $x_j$=24 and $x_j$=60 in figure 3(a)).
    3. Return the modified $j^{th}$ access right's dimension.
end
```

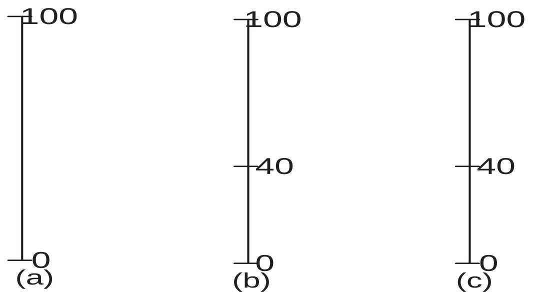

Fig. 2 (a). Initial friendship(F) access right dimension. (b). Access right dimension after insertion of the end point 40 in the first content.(c). Access right dimension after insertion of the end point 100 in the first content.

2) Modification of Bit-vectors: Each bit in a bit-vector corresponds to a particular content. Thus, if $n-1$ contents are present then bit-vectors will be of length $n-1$ bits, which needs to be changed to n bits while inserting a new content. Bit-vectors of all the elementary ranges are modified by appending a bit equal to 0 or 1 to them. We call the algorithm 2, $BV\_Modify(j,a_j^n,b_j^n)$, to modify the bit-vectors. The algorithm appends a bit equal to 1 in the bit-vectors corresponding to all the elementary ranges between a_j^n and b_j^n and 0 to rest of the bit-vectors. Thus, in the bit-vector transform domain if the n^{th} bit for a content is 1 in an elementary range then the elementary range will be within the respective access right's range in the n^{th} content. This is used as a basis in access control mechanism in section 4.2.

Figure 3(a) shows the friendship access right's dimension after insertion of initial bit-vectors in figure 2(b). Figures 3(b) and 3(d) show access right's range insertion steps for the second and third content respectively. 3(c) and 3(e) show bit-vector modification step after the insertion of second and third content respectively.

```
Algorithm 2 $BV\_Modify(j, a_j^n, b_j^n)$
Input: $a_j^n$, $b_j^n$: End points of the $j^{th}$ access right's range in the $n^{th}$ content.
Output: Modified $j^{th}$ access rights dimension.
Procedure:
1. Append a bit equal to 0 to the LHS of all the bit-vectors present between $D_j^{min}$ and $a_j^n$ along the
   $j^{th}$ access right's dimension.
2. Append a bit equal to 1 to the LHS of all the bit-vectors present between $a_j^n$ and $b_j^n$ along the
   $j^{th}$ access right's dimension.
3. Append a bit equal to 0 to the LHS of all the bit-vectors present between $b_j^n$ and $D_j^{max}$ along the
   $j^{th}$ access right's dimension.
4. Return the modified $j^{th}$ access right's dimension.
```

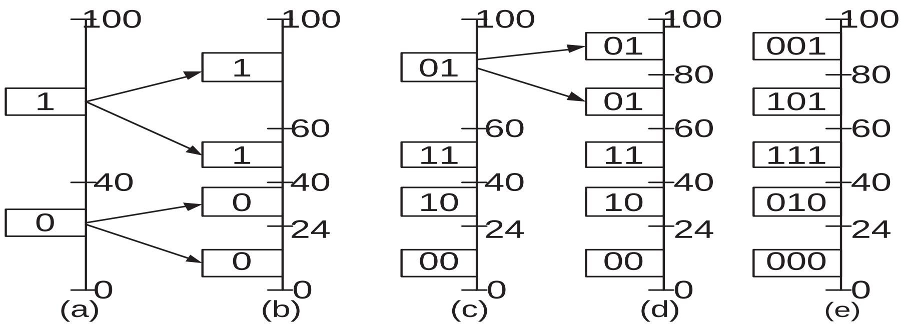

Fig. 3 Illustration of the Content Insertion Process

If multiple discrete ranges are present for an access right associated with a content then insertion process is modified slightly. Let d number of discrete ranges viz. ($a_{j_1}^n$, $b_{j_1}^n$], ($a_{j_2}^n$, $b_{j_2}^n$],..., ($a_{j_d}^n$, $b_{j_d}^n$] are present for the j^{th} access right. In the first step, all the end points of all the ranges are inserted along the j^{th} dimension using algorithm 1. For the bit-vector modification step, initially algorithm 2, $BV\_Modify(a_{j_1}^n, b_{j_1}^n)$ is called (for the first range). For doing bit-vector modification using all other ranges, the bit-vector modification step is slightly modified. In the modified algorithm for the $i^{th}(1< i\leq d)$ range, we don't insert a new bit but change the n^{th} bit to 1 in all the bit-vectors present between $a_{j_i}^n$ and $b_{j_i}^n$. This ensures that the size of the bit-vectors remain n bits if n contents are inserted, irrespective of the number of ranges.

4.2 Access Control Mechanism

Let the attribute vector of an individual accessing the contents be given by A=$\{A_1$, A_2,...$A_M\}$. Initially, for each value of j, a search is made for A_j along the j^{th} dimension. Let the elementary range which contains A_j be given by E_j and respective bit-vector be given by B_j. The contents for which the j^{th} access right's range contains E_j, their corresponding bit will be equal to 1 in B_j(see section 4.1). Thus, the j^{th} access right's range will also contain A_j as $A_j \varepsilon E_j$.

A content(say n^{th}) can be accessed by an individual if he satisfies all M access rights conditions to access the content. Thus, the n^{th} bit must be equal to 1 in all M bit-vectors searched over all access right's dimensions. This information can be obtained by taking an AND operation between the bit-vectors obtained for all access right dimensions(if a particular bit is 1 in all bit-vectors then that bit will also be equal to 1 in the bit-vector obtained after AND operation). As shown in equation 1, final bit-vector B is obtained by performing AND operation between all the bit-vectors.

$$B = B_1 \wedge B_2 \wedge ... \wedge B_M \tag{1}$$

Finally, we scan the final bit-vector B to identify the contents bits corresponding to which is 1. To identify the n^{th} ($1 \le n \le N$) bit, we perform an AND operation of B with a bit vector containing n^{th} bit 1 and other bits 0. If the result is 1 then the n^{th} content can be accessed.

Example 2. Consider the individual I_2 in example 1 accesses the contents shared by I_1. The attribute vector of I_2 is A=\{50, 40, 25\}. Figure 4 shows friendship (F), trust (T) and age dimensions after the insertion of all three contents in example 1. The first attribute(having value 50), belongs to the elementary range 40 to 60 in figure 4(a). Hence, B_1=111 in this case. Similarly, for other two access rights bit-vectors can be calculated (using figures 4(b) and 4(c)) and are given as: B_2=001 and B_3=111. In this case, B=111$\wedge$001$\wedge$111=001. Since, only the bit corresponding to c_1 is 1 therefore I_2 can only access the content c_1, which is in accordance with example 1.

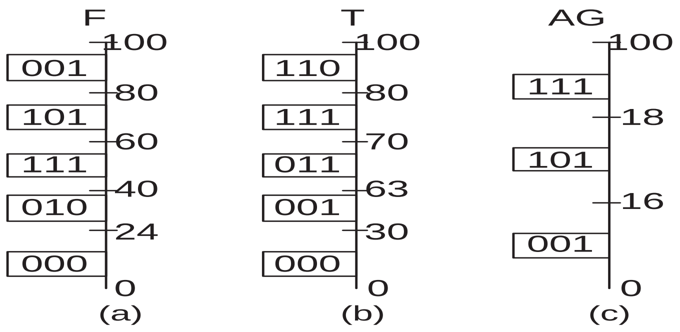

Fig. 4 (a). Friendship dimension (F), (b). trust dimension (T), and (c). age dimension (AG) after insertion of contents in example 1.

The access control mechanism is also valid if multiple ranges are possible for an access right as the bit corresponding to a content is 1 in the bit-vectors corresponding to all the ranges present for the content.

4.3 Modification of Access Rights

The access rights associated with the contents can be modified by the content owners. These changes must be reflected in the bit-vector transformed domain. Let the user changes a range in the j^{th} access right associated with the n^{th} content from $[a_j, b_j]$ to $[a'_j, b'_j]$. Then algorithm 3, $AR\_Modification(n, j, a_j, b_j, a'_j, b'_j)$, is used to reflect the change in bit-vector transform domain. In the algorithm, we first change the n^{th} bit in every bit-vector between a_j and b_j to 0. After that the algorithm calls $AR\_Insert(a'_j)$ and $AR\_Insert(b'_j)$ (algorithm 1) to insert the points a'_j, and b'_j if they are not already present. Then it sets n^{th} bit in all the bit-vectors between a'_j, and b'_j to 1. Finally, the algorithm removes a_j and/or b_j if no other content shares the end points a_j and/or b_j using algorithm 4. This is to reduce the search time and storage space by removing any redundant point along the access right's dimension.

Algorithm 3 $AR\_Modification(n, j, a_j, b_j, a'_j, b'_j)$

```
Input: $a_j$ and $b_j$: Original end points of the $j^{th}$ access right's range for the $n^{th}$ content.
$a'_j$ and $b'_j$: New end points of the $j^{th}$ access right's range for the $n^{th}$ content.
Output: Modified $j^{th}$ access right's dimension.
Procedure:
1. Make the $n^{th}$ bit to 0 in all the bit-vectors between $a_j$ and $b_j$.
2. Call $AR\_Insert(a'_j)$ and $AR\_Insert(b'_j)$.
3. Set the $n^{th}$ bit to 1 in all the bit-vectors between $a'_j$ and $b'_j$.
4. Call $delete\_point(j, a_j)$.
5. Call $delete\_point(j, b_j)$.
6. Return $j^{th}$ access right's dimension.
```

Algorithm $delete\_point(j, x_j)$ first obtains the bit-vectors for the elementary ranges just before and after x_j along the j^{th} access right's dimension. If both are equal this implies that no content has x_j as an end point for the j^{th} access right. Thus, the point x_j can be deleted.

4.4 Deletion of a Content

In social networks, a user can delete a content from his profile. The algorithm 5, $delete\_content(n)$, is used to delete the n^{th} content from the user's profile. Two steps are required. In the first step, the n^{th} bit in all the bit-vectors is made 0. Next, if no

Algorithm 4 $delete\_point(j, x_j)$

```
Input: $x_j$: The point to be deleted from the $j^{th}$ access right's range.
Output: Modified $j^{th}$ access right's dimension.
Procedure:
if $x_j = D_j^{min}$ or $x_j = D_j^{max}$ then
    Exit.
else
    Let the bit-vectors for the elementary ranges just before and after $x_j$ be given by $B_{low}$ and $B_{high}$ respectively.
if $B_{low} = B_{high}$ then
    Delete $x_j$ from the $j^{th}$ access right's range.
Return $j^{th}$ access right's dimension.
```

other content shares the end points a_j and/or b_j then remove a_j and/or b_j from the j^{th} dimension. Algorithm 4 is used to remove a_j and/or b_j.

Algorithm 5 $delete\_content(n)$

```
Input: $n$: ID of the content to be deleted.
Output: Modified version of all $M$ access rights' dimensions.
Procedure:
for $j=1$ to $M$ do
    1. Make the $n^{th}$ bit in each bit-vector equal to 0.
    2. Let the range of $j^{th}$ access right in the $n^{th}$ content be given by $[a_j, b_j]$.
    3. Call $delete\_point(j, a_j)$.
    4. Call $delete\_point(j, b_j)$.
end
```

4.5 Efficient Data Representation

In the bit-vector transform domain, end points of elementary ranges along each access right dimension are present in the increasing order of magnitude. This suggests use of efficient data structures such as binary search trees(BST), AVL trees, red black trees[30] etc. to represent each access right dimension. As AVL trees are more search efficient[29][30] so we use an AVL tree to represent each access right's dimension. A separate AVL tree is designed for each access right's dimension. In this section, we briefly describe the procedure for insertion and searching of data in AVL trees.

There are two types of nodes in each AVL tree: internal nodes and leaf nodes. As shown in figure 5, the internal nodes(in circles) in each AVL tree stores the points along the access right's dimension. Whereas, leaf nodes(in rectangles) store the bit-vector corresponding to an elementary range determined by the values stored in internal nodes. For example, in figure 5(b) the bit-vectors 00, 10, 11, and 01(from left to right) corresponds to elementary ranges 0 to 24, 24 to 40, 40 to 60, and 60 to 100 respectively.

Two steps(corresponding to access rights insertion and bit-vector modification steps in section 4.1) are required for the insertion of access rights in an AVL tree. The first step is access right's range insertion and the second step is bit-vector modification. During the access right's range insertion process, the end points of the access right's ranges are inserted one by one using the normal insertion process in AVL trees [30]. During the bit-vector modification step, an in-order traversal(recursive processing of the tree in the order: the left sub-tree, root, and right sub-tree) in the AVL tree is performed. If the range of the j^{th} access right is given by $[a_j, b_j]$ then all the bit-vectors in between a_j and b_j are appended with a bit equal to 1 and rest of the bit-vectors are appended with a bit equal to 0. Figure 5(a), 5(b), and 5(c) illustrates an AVL tree designed for the access right dimension shown in figure 3(a) 3(c), and 3(e) respectively.

For the access control mechanism discussed in section 4.2, a search along each access right's dimension is required. Following algorithm(algorithm 6) is used to search along the bit-vector along the j^{th} dimension(let the j^{th} attribute in the attribute vector of a person accessing the content be given by A_j).

Algorithm 6 $Search(j, A_j)$

```
Input: $j$: Dimension to search.
$A_j$: A numerical value to be searched along $j^{th}$ dimension.
Output: Bit-vector corresponding to $A_j$.
1. Set the root node as current node.
2. Compare the value $A_j$ with the value stored in the current node. If $A_j$ is greater than the value
   stored in the current node then assign the right child of the current node as current node. Else
   assign the left child of the current node as current node.
3. If current node is a leaf node then bit-vector stored in the node is the bit-vector required. Else
   go to step 2.
```

5 Grouping of Contents With Same Access Rights

Practically in present social networks, a number of contents shared by a user may have the identical access rights vector. This fact may be used to do the access control more efficiently (as compared to bit-vector transform proposed in section 4). In this section, we present an algorithm to do the access control more efficiently by identifying the contents with identical access rights vector. The algorithm works efficiently by deriving the smaller length bit-vectors after identification of contents with the identical access rights vectors. The proposed algorithm consists of two steps. First the grouping of contents with similar access rights vector and smaller

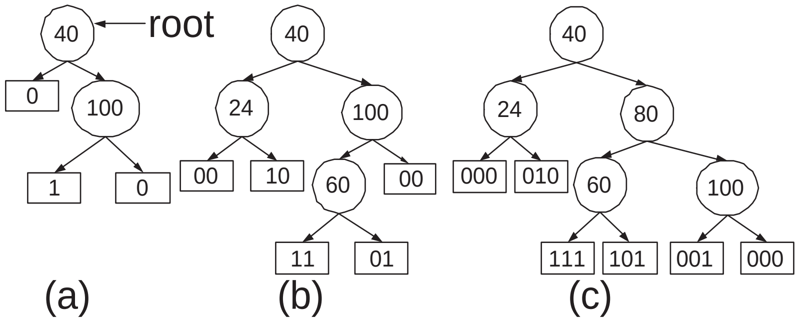

Fig. 5 Insertion process in AVL tree

length bit-vector derivation, and second efficient access control using the smaller length bit-vectors derived. Next, we discuss both these steps in detail.

5.1 Grouping of Contents

Let a user has previously $n-1$ (n>1) number of contents in his profile and these contents are divided into p ($p \leq n-1$) different groups such that contents in each group have identical access rights vector (i.e. all M access rights ranges have identical end point values for all the contents in a group). To do the access control more efficiently in such cases, instead of using $n-1$ length bit-vectors we use p length bit-vectors. i^{th} bit in a bit-vector represents the i^{th} group of contents. To identify the access rights vector and contents in each group we maintain a table of the form table 1 mapping the group IDs with the rights vectors and content IDs. Following process is used when n^{th} content is added to the profile.

Table 1 Mapping of group ID with access rights vector and content IDs

group ID	Access rights vector	*content IDs*
1	R_1=$\{r_{11}, r_{12},..., r_{1M}\}$	1, 3, 8, 10, 11, 15
2	R_2=$\{r_{21}, r_{22},..., r_{2M}\}$	2, 12, 16
3	R_3=$\{r_{31}, r_{32},..., r_{3M}\}$	4, 7, 14
4	R_4=$\{r_{41}, r_{42},..., r_{4M}\}$	5, 6, 9, 13, 17

Let the j^{th} access right's range in the n^{th} content be given by [a^n_j, b^n_j]. Initially we obtain the bit-vector at the points a^n_j and b^n_j. The i^{th} bit's value (0 or 1) in these bit-vectors give information on whether a^n_j and/or b^n_j are present in the j^{th} access right's range in the i^{th} group of contents. In order to find all such groups (with bits=1), we take an AND operation between these two bit-vectors obtained at a^n_j and b^n_j. Let B be the resultant bit-vector obtained after taking the AND operation. If the i^{th} bit of B is equal to 1 then both a_j^n and b_j^n will be within (not necessarily exactly identical) the access rights' ranges in all the contents in the i^{th} group of contents. Note that there can be multiple bits=1 in B, each position of a bit=1 implies that all the access rights' ranges in the n^{th} content are within the access rights' ranges of the group at the bit position. Finally, we need to find the group whose access rights' ranges exactly match the access rights' ranges of the n^{th} content. For this purpose, we compare the access rights ranges in the n^{th} content with the access rights ranges of all group for which bit is 1 in B. If any such group is found then the content is added to that group and no further comparisons are required (as access rights associated with each group are different and at most one group's access rights can exactly match the access rights associated with the content). In this case no change is required in the p length bit-vectors associated with the elementary ranges. Else if no such group is found then a new group containing only the n^{th} content is created. And new elementary ranges are inserted and a new $(p+1)^{th}$ bit is added to each bit-vector according to algorithms 1 and 2. The same process is applied every time a new content is added to the profile.

5.2 Access Control After Grouping of Contents

Let the attribute vector of an individual accessing the contents be given by $A=\{A_1, A_2,...A_M\}$. Initially, we follow the same process as discussed in section 4.2 to obtain the resultant bit-vector B. Let there be total p groups of contents then the resultant bit-vector obtained will consist of p bits, each bit corresponds to a particular group of contents. The value of the bit for a group of contents is 1 if all the contents in the group can be accessed by the individual. So, we scan the bits in B. Then in the mapping table between group IDs and content IDs, go to the groups respective to bits=1 in B and then present all the content in the groups to the individual accessing the contents.

The method presented is efficient as the bit-vectors, after identification of contents with identical access rights vector, consist of less number of bits. Hence the number of comparisons during AND operation and scanning of bit-vectors will be low.

6 Performance Analysis

In this section, we analyze both mathematical as well as experimental performance of our proposed method in terms of time required to access the contents, storage space, and insertion time. For experimental performance analysis, we use relational database(RDB) model as reference for storing the access rights. This is because most of the literature[9][10] and existing social networks [17] use relational database for storing the access rights. In RDB model, access rights are stored in tabular format in a relational database(DB). Each row in the table corresponds to a particular content and each column corresponds to a particular access right. In this section, we first present the reason for efficient access control and then we present the mathematical and experimental analysis.

The proposed method of access rights organization using bit-vector transform outperforms the RDB model of access control because of following reasons. Firstly, the maximum possible elementary ranges along each access right's dimension cannot be more than maximum possible values in the domain of that access right. Thus, in our proposed arrangement the search space is limited to the number of values in the domain of the access right. Secondly, the search time along each access rights dimension is further reduced as each dimension can be represented using AVL tree structure. For example, If there are 1000 contents with a person. To verify the credential of an individual accessing the content(assuming a single access right), 2000 comparisons are needed (assuming only single range with each access right) if we use RDB based access control model. If the domain of that particular access contain 100 values then our method requires 100 comparisons if we do not represent access rights with an AVL tree. Only 8 to 10 comparisons are needed for the same purpose when we represent the dimension using an AVL tree. In case of multiple access rights associated with each content, our method requires M-1 more AND operation between the bit-vectors obtained for each access right's dimension, where M is the number of access rights associated with each content. But still our method remains much more efficient. As discussed in mathematical and experimental analysis, the relative gain becomes even more for the case of multiple ranges per access right.

6.1 Mathematical Performance Analysis

To analyze the mathematical performance, we assume that there are N number of content present and each content is defined with M access rights. For ease of understanding, we derive the expressions for access rights with single range only. We discuss the effect of multiple discrete ranges on performance in section 6.2.

1) Access Time Required: Let after insertion of N contents, j^{th} access right's dimension is divided into n_j number of elementary ranges. The complexity for the searching step will be $O(log_2(n_j))$ for the j^{th} access right's dimension. If there are N contents then each bit-vector will be N bits long. It can be represented using $\lceil N/k \rceil$ integers in computer, where k is the number of bits in the data format(e.g. integer or long) using which bit-vectors represented in computers(N bit long bit-vectors are implemented using a linked list containing $\lceil N/k \rceil$ elements). Further, $M-1$ AND operations will be required to get the final bit-vector. Thus, the time required for the second step will of O(($M-1$)*$\lceil N/k \rceil$). Finally, the bit-vector obtained is scanned to find the bits which are equal to 1 for finding the contents which a person can access. This step requires N number of AND operations. Thus, the total time required can be given as $k_1 * \sum_{j=1}^{M} (log_2(n_j))$+ k_2*((M-1)*$\lceil N/k \rceil$+N). Here, k_1 and k_2 are arbitrary system dependent constants.

2) Space Complexity: The space complexity is given by the space required to store the bit-vectors and end points of elementary ranges. Each bit-vector consists of N bits and there n_j number of bit-vectors along the j^{th} access rights dimension. Thus, the space required to store the bit-vectors will be $\sum_{j=1}^{M} n_j * N$ bits or if bits are represented in integer format then the space required will be $\sum_{j=1}^{M} n_j * \lceil N/k \rceil * k$. To store the end points of elementary ranges, we require to store n_j integers along the j^{th} access right's dimension. Thus, the space required to store elementary ranges will be $\sum_{j=1}^{M} n_j * k$. In addition, $2*N*M*k$ bits of space is required to provide an interface for the access rights. Thus, the total space required will be $\sum_{j=1}^{M} n_j * (\lceil N/k \rceil * k + k) + 2*N*M*k$ bits.

3) Insertion Time complexity: The first step of the insertion process is to insert the both end points of all access rights. The time required for this step is $k_1*2*\log(n_j)$ for the j^{th} access right's dimension. The second step is the bit-vector modification. Let the range of j^{th} access right be given by $[a_j, b_j]$. Let the number of elementary ranges between a_j and b_j be given by n'_j. Then the time required for the second step would be $k_2*\sum_{j=1}^{M} n'_j$. Thus, the overall time complexity would be $2*k_1*\sum_{j=1}^{M}\log(n_j) + k_2*\sum_{j=1}^{M} n'_j$.

6.2 Experimental Performance Analysis

All the experiments were performed on Intel(R) core(2) 2.40 GHZ 32-bit CPU with 2 GB RAM. We perform the experiments for access time required for the number of concurrent profile accesses in the system, assuming that on an average 1000 contents are present in each user's profile. For storage space and insertion time complexity, we perform experiments for up to 10000 contents(N) in the user's profile. For the experiments purpose, we assume that the number of access rights associated with the contents are between 7 and 15. The number of values in the domain of access rights is randomly chosen in between 10 and 100. The value of k is 32 in our case as we represent bit-vectors using integer format. We compare the performance after considering both single ranged and multiple ranged access rights. For multi-ranged access rights we assume that 3 access rights out of all access rights can have multiple discrete ranges. To show the effect of multi-ranged access rights, we do the experiments by assuming up to 4 ranges for these access rights.

1) Access Time Required: Figure 6 shows the comparison os access time performance of our approach with RDB based approach for the case of single range and multi-ranged access rights. As shown in figure 6, our approach outperforms the RDB based approach for all the values of concurrent content accesses. It can also be observed that, in contrast to RDB approach, our approach takes almost same time in both single-ranged and multi-ranged access rights cases. This is because in our method number of *AND* operations required are same for both the cases. The only difference is searching along the access right dimensions. The search time along each access right's dimension also becomes almost same as the number of elementary ranges(n_j) reaches to maximum possible elementary ranges in access rights' domain. Thus, the proposed mechanism is more suitable for present social networks, which may have access rights in the form of multiple discrete ranges.

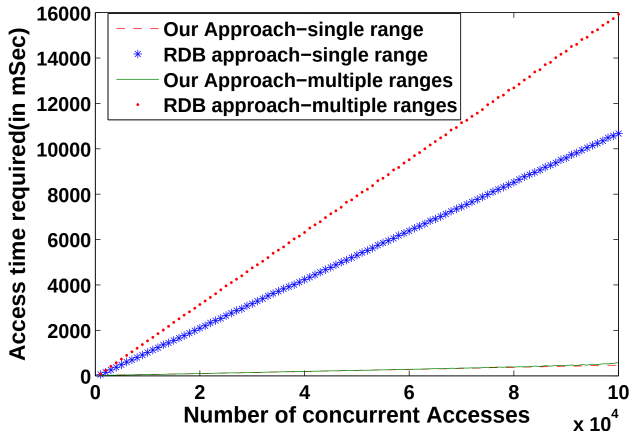

Fig. 6 Access control time

2) Space Complexity: The storage space is given by the space required to store the bit-vectors, end points of elementary ranges and access rights in the original RDB form(required for providing an interface to users). Figure 7 compares the performance in terms of storage space required to store N contents in a profile. Although, the storage of access rights in bit-vector transform takes lesser time but additional space is required to store the access rights in the original RDB form along with the contents. Overall overhead for the case of single-ranged access rights is about 60-70%. The overhead is relatively smaller for the case of multi-ranged access rights due to finite possible access rights' ranges.

3) Insertion Time Complexity: Figure 8 shows the insertion time complexity for insertion of N^{th} content when $N-1$ contents are already present. As derived in section 6.1, the insertion time depends on n_j and n'_j, it does not depend on N. So, the curve initially rises until maximum elementary ranges are reached then it becomes almost constant. The overhead due to the insertion is very less. For most of the values of N, it is much less than the time required to do the access control with RDB approach. In particular, we observe that it is about 3 times to 30 times less for N=1000 to N=10000(see figures 6 and 8). In case of multi-ranged access rights, the insertion time is proportional to the number of ranges. However, we need to insert the contents only once. And in general, every content in MMSNs is accessed several times so the overall effect of insertion complexity on the system performance would be small.

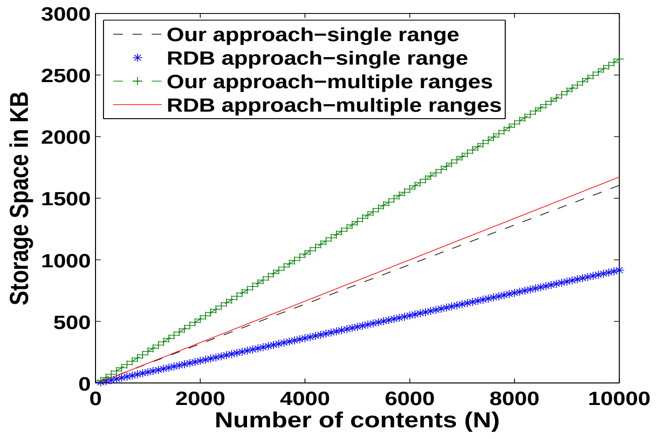

Fig. 7 Storage Space Required

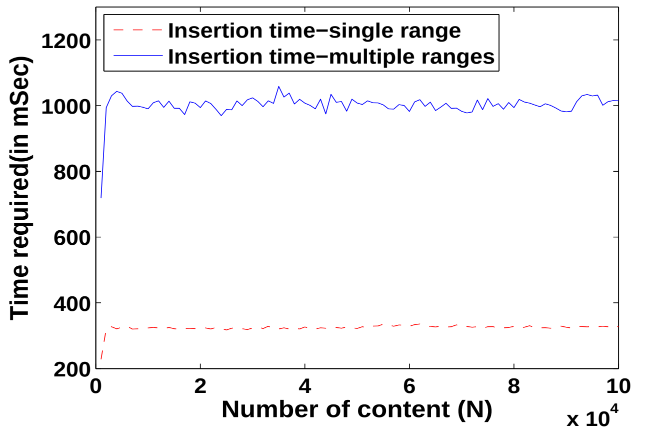

Fig. 8 Insertion Time Complexity

7 Conclusion

Efficient access control in MMSNs is an important requirement with the fast rate of proliferation of MMSNs and new innovative access control requirements. In this chapter, we presented an efficient access control mechanism for MMSNs. The proposed mechanism is suitable and scalable for present social networks as it can easily handle fine grained access control and other functionalities required in existing MMSNs. Experimental results show that our technique is about 30 times more efficient than the existing access control mechanisms. The overhead due to insertion of access rights and storage space is also small. Thus, our proposed bit-vector transform based access control mechanism turns out to be a good choice for storing the access rights associated with the contents in MMSNs.

Acknowledgment

Thanks to the Agency for Science, Technology and Research (A-STAR), Singapore for supporting this work under the project ”Digital Rights Violation Detection for Digital Asset Management” (Project No: 0721010022).

References

1. V. Kolovski, Y. Katz, J. Hendler, D. Weitzner, and T. Berners-Lee. Towards a policy-aware web. *Semantic Web and Policy Workshop at the 4th International Semantic Web Conference*, 2005.
2. J. Williams. Social networking applications in health care: threats to the privacy and security of health information. *ICSE Workshop on Software Engineering in Health Care*, 2010.
3. G. Loukides and A. Gkoulalas-Divanis. Privacy challenges and solutions in the social web. *ACM Crossroads*, 16(2):14–18, 2009.
4. A.C. Squicciarini, M. Shehab and F. Paci. Collective privacy management in social networks. *18th international conference on World wide web.*, pages 521–530 2009.
5. YouTube fact sheet. Available at: http://www.youtube.com/t/fact_sheet.
6. Facebook statistics. Available at: http://www.facebook.com/press/info.php?statistics.
7. Flickr. www.flickr.com/
8. Neilsen online technical report 2009 Available at: http://en-us.nielsen.com/.
9. B. Carminati, E. Ferrari, and A. Perego. Enforcing access control in web-based social networks. *ACM Transactions on Information and System Security*, 13(1):191–233, 2009.
10. B. Carminati, E. Ferrari, and A. Perego. Rule-based access control for social networks. In *On the Move to Meaningful Internet Systems: OTM Workshops*. pages 1734–1744, 2006.
11. B. Carminati, E. Ferrari, and A. Perego. Privacy-Aware Access Control in Social Networks: Issues and Solutions. *Privacy and Anonymity in Information Management Systems*, pages 181–195, 2010.
12. R. Iannella. Digital rights management (DRM) architectures. *D-Lib Magazine*. 7(6), 2001.
13. E. Diehl. A four-layer model for security of digital rights management. *8th ACM workshop on Digital rights management.* pages 19–28. 2008.
14. A. Sachan, S. Emmanuel and M.S. Kankanhalli. Efficient Aggregate Licenses Validation in DRM. *15th Database Systems for Advanced Application(DASFAA)*. pages 313–319. 2010.
15. C. Gates. Access control requirements for web 2.0 security and privacy. *IEEE Web 2.0 Privacy and Security Workshop*. 2007.
16. V. Wilfred, A. Bader and M. Muthucumaru. An Access Control Scheme for Protecting Personal Data. *Sixth Annual Conference on Privacy, Security and Trust*. 2008.
17. W. Graham. Reaching users through facebook: A guide to implementing facebook athenaeum. *Code4Lib Journal*, (5), 2008.
18. S. Grzonkowski, B. Ensor, S. R. Kruk, A. Gzella, S. Decker, and B. McDaniel. A DRM solution based on social networks and enabling the idea of fair use. *Proceedings of Media in Transition*, 2007.
19. A. Sachan, S. Emmanuel and M.S. Kankanhalli. Efficient License Validation in MPML DRM Architecture. *9th ACM Workshop on Digital Rights Management*. pages 73–82. 2009.

20. A. Tootoonchian, K. K. Gollu, S. Saroiu, Y. Ganjali, and A. Wolman. Lockr: social access control for web 2.0. In *First ACM workshop on Online social networks*. pages 43–48. 2008.
21. D. Brickley and L. Miller. FOAF vocabulary specification. 2005.
22. A. Sachan, S. Emmanuel and M.S. Kankanhalli. An Efficient Access Control Method for Multimedia Social Networks. *2nd ACM Multimedia Workshop on Social Media (WSM)*. 2010.
23. R. Iannella. A framework for the policy-oriented web in social networks. *7th International Workshop for Technical, Economic and Legal Aspects of Business Models for Virtual Goods*. 2009.
24. M. K. F. Beato and K. Wouters. Enforcing access control in social networks. *HotPets* pages 1–10. 2009.
25. M. Shehab, A. Squicciarini and G.J. Ahn. Beyond User-to-User Access Control for Online Social Networks. *Information and Communications Security* pages 174–189. 2008.
26. C.A. Yeung, I. Liccardi, K. Lu, K. Seneviratne and T. Berners-Lee. Decentralization: The future of online social networking. *W3C Workshop on the Future of Social Networking* 2009.
27. G. Governatori and R. Iannella. Modelling and reasoning languages for social networks policies. *13th IEEE international conference on Enterprise Distributed Object Computing*. 2009.
28. R. Baden, A. Bender, N. Spring, B. Bhattacharjee and D. Starin. Persona: an online social network with user-defined privacy *ACM SIGCOMM*. pages 135–146. 2009.
29. B. Pfaff Performance analysis of BSTs in system software. *ACM SIGMETRICS Performance Evaluation Review*. 32(1):410–411, 2004.
30. A. Andersson. General balanced trees. *Journal of Algorithms*, 30(1):1–18, 1999.