Audit Recommendation for Privacy Protection in Personal Health Record Systems

Zhong Han, Yuqing Sun*, Yuan Wang
School of Computer Science and Technology
Shandong University
Jinan, China
hanzhong15@163.com, sun_yuqing@sdu.edu.cn, wangyuan_2010@yeah.net

***Abstract*—Personal Health Record (PHR) systems store a large amount of users' health information, which is very sensitive for users. So privacy protection is an important issue for PHR systems. Although information technology audit can find out improper accesses to users' sensitive data that violate their privacy policies, it is difficult for inexperienced users to use audit commands. This paper presents an audit recommendation framework to help users appropriately audit their sensitive records so as to have a good understanding of their privacy situation. For an audit requester, we analyze doctors' historical queries to find the similar users of the requester. Then we analyze the audit commands of those similar users and make some recommendations to the requester. Experiments are performed to verify our method.**

Keywords—audit; recommendation; personal health record systems

I. INTRODUCTIONON

Personal Health Record systems are user-centered electronic medical systems which store users' personal health information, such as Microsoft HealthVault and Google Health [1]. The information in these systems includes medical diagnoses, clinical symptoms, medication records, insurance information and other data from medical devices such as wireless electronic weight scales. It is convenient for users and doctors to communicate with each other. Users can check and share their health information. Doctors can know well about the users' health situations so as to make better treatment options. Since PHR systems store a large amount of users' sensitive information, privacy protection is essential. Some PHR systems allow users to set their own privacy policies [2]. However, users may not know how to set the privacy policies without affecting doctors' diagnostic. Users may worry about their health privacy if the policies are too loose. But setting too strict policies may affect the treatment in some emergency circumstances. In this case, general users often assign more access rights to their doctors so as not to affect the diagnostic.

However, assigning much too access rights to doctors may lead privacy disclosure. In order to solve this problem, audit technology is adopted to help users to check their privacy situation. The advantage of audit is that users can discover and track inappropriate queries and those related doctors. So audit mechanism can constrain doctors' behavior to some extent. Hence, users can assign more access rights to doctors to allow all necessary accesses. Some of current PHR systems provide audit function for users, such as HealthVault allows users to view the audit trails in their HealthVault account at any time. But the audit function proposed by HealthVault is simple that users can't specify flexible audit goals. It is difficult for users to find the privacy disclosure in a large number of log entries. So providing more flexible and efficient audit functions for users is necessary. However, those inexperienced users don't know how to audit efficiently, especially in big data and complex access cases. For example, a user who has a heart disease doesn't want others to know the fact. He may want to know whether some uncorrelated persons have viewed his heart disease related records. But he doesn't know how to do an audit to achieve this goal. So audit recommendations are necessary to help him to appropriately audit his records of heart disease.

This paper proposes an audit recommendation framework to solve this problem. The purpose of the framework is to recommend appropriate incomplete audit commands to those inexperienced users, help the users appropriately audit their sensitive records so as to have a good understanding of their privacy situation. The incomplete audit commands here specify the audit commands that ignore the unimportant fields. For an audit requested user, we first find users who have similar diseases with the requester by analyzing doctors' historical queries. Then we analyze the audit commands that have been submitted by the similar users and obtain the related incomplete audit commands. At last, we calculate the recommendation priority of those incomplete audit commends and recommend several ones with high priority to the audit requester.

II. RELATED WORK

A. Query Auditing

The Health Insurance Portability and Accountability Act (HIPAA) released by United States demands that users have rights to know when and by whom their health information is viewed. Query auditing is a security mechanism to conform this rule. In order to implement query auditing, database keeps query logs which record all queries and other related information. For a given query log, query auditing analyzes the queries in the log to determine whether some legal queries have acquired users' sensitive information. Auditors can make further analysis on the basis of auditing results to find out the persons who have disclose the sensitive information. An audit

* Corresponding author.

framework for electronic medical record (EMR) systems has been presented in [3] depending on the security rule of HIPAA. If a patient's privacy leaked after treating in a hospital, the auditor of the hospital can find out the suspects who may have disclosed the user's privacy. The audit framework presents an audit expression which is close to a SQL statement:

AUDIT *audit list* FROM *table list* WHERE *condition list.*

where *audit list* is a list of privacy attributes, *table list* is a list of tables of the attributes in audit list and *condition list* is a list of conditions that describe the target patients of the audit. Literature [4] proposes an audit method which can analyze any complex SQL statements. The audit method presented in [5] can detect privacy disclosures caused by the combining of multiple SQL queries. Literature [6] presents an audit method under the environment of XML data query. Literature [7] discusses the audit method after setting wrong access control policies. Literature [8] presents an audit method under the security rule of data retention. The existing literatures have discussed audit methods in different ways. The methods are mainly applicable to EMR systems in which the audit functions are provided to professional auditors. Although the existing audit methods can be applied to PHR systems, they are not appropriate for those inexperienced users. So we present an audit recommendation framework to help users to appropriately audit their health privacy in PHR systems.

B. Recommender Systems

This paper is also related to recommender systems. Recommender systems produce a list of items for users of their interests, which are widely used in areas such as electronic commerce and social network. There are four kinds of implementation approaches of recommender systems [9]: content-based filtering, collaborative filtering, social network based filtering and network structure based filtering. Collaborative filtering methods are widely used in recommender systems. The idea of collaborative filtering is that if two users have similar evaluations to some certain items, they are similar users who share similar interests, so the evaluations of them on other items should also be similar. This paper adopts this idea, that is, if doctors have visited two users' health indicators in similar ways, the two users may have same diseases, so they may concern same health privacy and have same audit requirements. But the existing recommendation approaches can't be applied directly to PHR systems because the methods of finding similar users in PHR environments are different with those in traditional recommender environments. In this paper, we calculate the similarity between two users through doctors' historical queries, which is different to the similarity calculation methods in recommender systems.

III. AUDIT RECOMMENDATION FRAMEWORK

A. Concepts and Notions

In a PHR system, we call a person user who uses the system to store and manage his health information and call a person doctor who accesses users' health information for treatment purposes. Let $\mathcal{U}$ and $\mathcal{D}$ be the set of all users and all doctors in a PHR system, respectively, $|\mathcal{U}|$ and $|\mathcal{D}|$ represent the number of elements in these two sets.

Definition 1(Health indicator):A health indicator is one of the items of clinical results, physical test results and medication records, such as cholesterol, blood pressure, blood type, and so on. Let O be the set of all health indicators.

Definition 2(access transaction): An access transaction records a data access in a PHR system. Each access transaction is denoted as a tuple of the form <*doctor*, *user*, *objects*, *time*>, where $doctor \in \mathcal{D}$, $user \in \mathcal{U}$ and $objects \subseteq O$.

For example, an access transaction of <Bob, Alice, {cholesterol, blood pressure}, 2012/01/01> specifies that a doctor Bob has accessed a user Alice's health indicators of cholesterol and blood pressure on the day of 2012/01/01.

Definition 3(accessed matrix): An accessed matrix describes access behaviors of doctors. Each user has an accessed matrix and we obtain a total accessed matrix through all users' accessed matrixes.

For a user $u_k \in \mathcal{U}(1 \le k \le |\mathcal{U}|)$, his accessed matrix B_k describes the accessed situation of his health indicators by doctors, the size of B_k is $|\mathcal{D}_k| \times |O|$ where $|\mathcal{D}_k|$ denotes the number of doctors who have accessed u_k. $B_k(i, j)=1$ if $d_i \in \mathcal{D}_k$ has accessed u_k, and 0 otherwise.

The total accessed matrix B describes the accessed situation of all users, the size of B is $|\mathcal{U}| \times |O|$, we call each row of B as a accessed vector, $B(i, j)$ is the number of doctors who have accessed u_i's health indicator o_j.

Definition 4(similar user): For a user, his similar users are those who have similar diseases with him. We calculate the similarity between users through their accessed matrixes.

Definition 5(audit command): An audit command is submitted by a user to audit the security situation of his sensitive information. Each audit command is denoted as a tuple of the form < *audit_user*, $time_1$, $time_2$, *objects* >, where $audit\_user \in \mathcal{U}$ and $objects \subseteq O$, specifies that a user *audit_user* audits the security situation of his health indicators in *objects*.

For example, an audit command of <Alice，2012/01/01，2012/01/10，{cholesterol}> specifies that Alice submits an audit command to audit the security situation about her cholesterol during 2012/01/01 to 2012/01/10.

B. The Audit Recommendation Framework

As shown in Figure 1, the audit recommendation framework consists of two parts: similar users finding and audit recommendation. In the first part, we find out similar users of an audit requester. For each disease, there is a set of health indicators related to it, doctors who treat the disease for users would focus on the users' health indicators of the set. If two users have a same disease, doctors would focus on the same set of their health indicators. So we compute two users' similarity through the accessed situation of their health indicators by doctors. We use a weight vector of health indicators during the similarity calculation so as to acquire more accuracy results. In the second part, we first abstract the

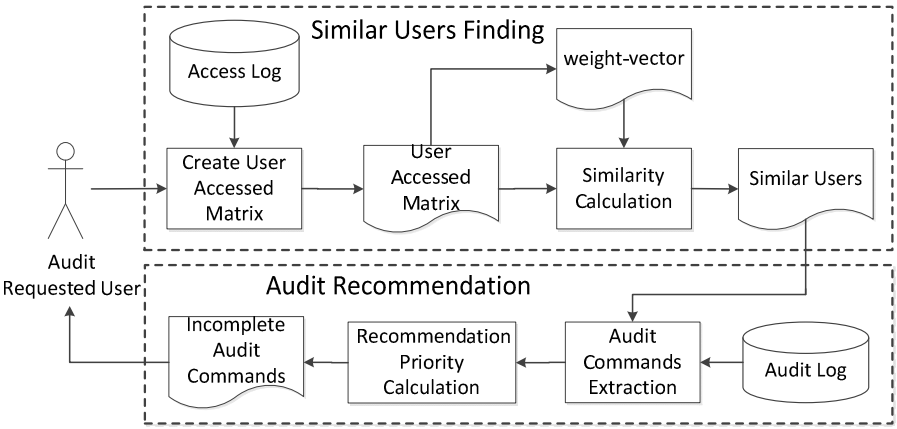

Fig. 1. An overview of the Audit Recommendation Framework.

audit commands that submitted by the similar users. Then we acquire some related incomplete audit commands and calculate their recommendation priority. At last, we recommend several incomplete audit commands with high priority to the audit requester.

IV. SIMILAR USERS FINDING

PHR systems record users' illness history, but they may be more concerned about the diseases that they have in recent time. For an audit requested user, we first set a time window [*recentBegin*, *currentTime*] which specifies a recently period of time during which the user has a sick. To get the *recentBegin* time, we select a time threshold and divide the access transactions that related to the user into groups. The time interval between any two records in each group is less than the time threshold. The *recentBegin* is the time of the earliest record in the recent group and the *currentTime* is current time. The time window can also be set by the audit requester. Let U, D and T, respectively, denote the set of related users, doctors and access transactions within the window [*recentBegin*, *currentTime*].

A. Create the Accessed Matrix

For a user $u_k \in U$, we extract a subset T_k from T and let the second term of the access transactions in T_k is u_k. Then we construct u_k's accessed matrix B_k and the size of B_k is $|D_k| \times |O|$, where D_k is the set of doctors who have accessed u_k. Each row in B_k represents a doctor's access situation on u_k's health indicators. We only consider once even if a doctor accessed a health indicator for many times. We construct a total accessed matrix B by all users' accessed matrixes, the size of B is $|U| \times |O|$, each row in B represents a user's accessed situation by doctors. We obtain the matrix B as follows:

$$B(k, j) = \sum_{i=1}^{|D_k|} B_k(i, j) \tag{1}$$

B. Weight Vector of Health Indicators

Different health indicators should have different weights when calculating the similarity between users. We use $\boldsymbol{w}$ to represent a weight vector which marks the weights of all health indicators. The weight vector $\boldsymbol{w}$ can be used to adjust the influence of each health indicator during the similarity calculation. If there is a health indicator that has been accessed by doctors for most users, the health indicator may not be sensitive and should be assigned a smaller weight. But if the health indicator has been accessed by doctors for only a few users, it may be a special one and should be assigned a bigger weight. The weight of each health indicator $o_j \in O$ in $\boldsymbol{w}$ can be computed as follows:

$$w_j = \frac{1}{\sum_{i=1}^{|U|} isAccessed(B(i, j))} \tag{2}$$

where for each health indicator o_j, we calculate the number of users whose o_j have been accessed by doctors. Then take the reciprocal of the number as the weight of the health indicator o_j. If the result of denominator equals 0, the result of w_j is 0. The function in (2) is as follows:

$$isAccessed(B(i, j)) = \begin{cases} 0 & if(B(i, j) = 0) \\ 1 & else \end{cases} \tag{3}$$

C. Similarity Calculation

For an audit requested user u_i, we calculate the similarity between u_i and other users. We get a revised accessed matrix M in which we consider the influence of different health indicators. The matrix M is constructed through the accessed matrix B and the weight vector $\boldsymbol{w}$:

$$M(i, j) = B(i, j) \times w_j \tag{4}$$

We use cosine similarity formula to calculate the similarity between u_i and u_j [12]:

$$sim(u_i, u_j) = \frac{\mathbf{M_i} \mathbf{M_j^T}}{(\mathbf{M_i} \mathbf{M_i^T})(\mathbf{M_j} \mathbf{M_j^T})} \tag{5}$$

where $\mathbf{M_i}$ and $\mathbf{M_j}$, respectively, denote the i-th row vector and the j-th row vector in matrix M. We specify a threshold r, the user whose similarity with u_i is bigger than r is a similar user of u_i.

V. AUDIT RECOMMENDATION

As the time fields in an audit command have so many possible selections, we ignore them when making a recommendation. An audit command is called as an incomplete audit command if we ignore its time fields and the first term. Such as <{cholesterol}> is an incomplete audit command related to an audit command of < Alice，2012/01/01, 2012/01/10，{cholesterol} >. For an audit requested user, we first extract and find appropriate incomplete audit commands from the audit histories of his similar users', then recommend them to the requester to help him set appropriate audit commands.

Assume that we are recommending audit commands to u_i. Let $v_k \in S_i (1 \le k \le |S_i|)$ be a user in S_i, where S_i is the set of u_i's similar users. Let A_k be a set of audit commands that submitted by v_k within the window [*recentBegin*, *currentTime*]. Then the set of all audit commands of u_i's similar users can be expressed as $A=A_1+A_2+A_3+\ldots+A_{|Si|}$. Then we partition A into subsets as $A=A_1'+A_2'+A_3'+\ldots A_p'$, where in A_j' there exist only one kind of audit commands that have same *objects*. Let a_j' be the incomplete audit command related to the audit commands in A_j'. So the total number of incomplete audit commands related to A is p. Then we construct a matrix C with a size of $|S_i| \times p$, where $C(k,j)$ represents the number of audit commands that related to a_j' and submitted by v_k. As everyone's audit habits may be different, so we deal with the matrix C to obtain a revised matrix C_* as follows:

$$C_*(k,j) = \frac{C(k,j)}{\sum_{j=1}^{p} C(k,j)} \tag{6}$$

where the denominator represents the total number of audit commands that have been submitted by v_k. Each row in C_* can be understood as a similar user's interesting degrees to all the incomplete audit commands. We calculate the recommendation priority of every incomplete audit commands for u_i as follows:

$$P_{i,j} = \frac{\sum_{u \in S_i} (sim(u_i,u) \times C_*(getRowNum(u), j))}{\sum_{u \in S_i} sim(u_i,u)} \tag{7}$$

where the function *getRowNum*(u) returns the row number of u in matrix C_*. The main idea to calculate $P_{i,j}$ is that sum all the similar users' interesting degrees on a_j' and also consider the similarity during the calculation. The greater the final result of $P_{i,j}$ is, the more attention on a_j' would be paid by u_i. Then, we recommend several incomplete audit commands with high priority to u_i.

VI. EXPERIMENTS

We perform experiments on a set of simulated data to verify our methods. In order to verify the correctness of the similar users finding part of our framework, we perform 4 experiments where we select four typical users and calculate their similarity with other users. The results are expressed in Figure 2 to Figure 5. Then we verify the correctness of the audit recommendation part of our framework according to the similarity calculation results of Figure 2 to Figure 5 and the accuracy calculation results of recommendation are expressed in Figure 6.

According to the diseases introduction on a professional medical website we find 28 diseases and their related indicators. Then we construct 141 users and randomly assign 1-2 diseases to each user. We predicate the accessed situation of those 141 users depending on the diseases that assigned to them. As doctors tend to cooperate with each other when they work and the optimal number of cooperation is 6 [10, 11]. So for a user and one of his diseases, we use a random number between 5 and 7 to denote the number of doctors who have accessed his health indicators that related to the disease. We first construct an original accessed matrix by the 141 users and their diseases. Then we obtain a revised accessed matrix by a weight vector and the original accessed matrix. We use these two matrixes to calculate the similarity between users in the following experiments. We call the calculation method as *accessed-matrix based method* (ABM) when using the original accessed matrix and *revised accessed-matrix based method* (RABM) when using the revised accessed matrix during the similarity calculation.

We select four typical diseases: AIDS, diabetes, pneumonia and flu to perform experiments, where AIDS stands for special diseases, diabetes and pneumonia stand for serious diseases and flu stands for common diseases. Then we select four users from the 141 users and each one of the four users has one or two diseases of the four diseases. We use both ABM and RABM to get the four users' similar users. We found that the similarity calculation results are more conform to our actual expectations when using RABM. In the following several figures, each figure represents the similarity calculation results of one of the four users with other users. The dotted lines represent the similarity between the user and other users when using ABM, the solid lines represent the results when using RABM. Here we say two users are similar users if their similarity is bigger than 0.8.

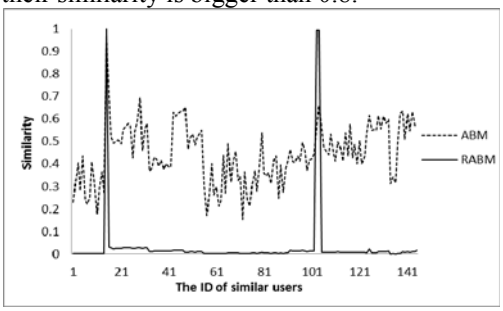

Fig. 2. Similarities of a user who have AIDS with other users

AIDS is a special disease and the number of users who have AIDS is so small, but it is a very sensitive privacy for those users. In Figure 2, for a user who has AIDS, we found one similar user who has only AIDS with ABM, and found another two users who have both AIDS and gastritis when using RABM. The latter two users should also be similar users because users who have AIDS would take it as a more important privacy than most of other diseases. The latter two users would pay more attention to AIDS related records than gastritis related records, so their audit commands should also

be valuable for the audit requester.

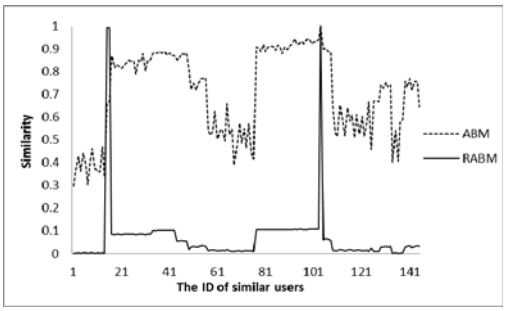

Fig. 3. Similarities of a user who have AIDS and pneumonia with other users

In Figure 3, for a user with both AIDS and pneumonia, we found many similar users with ABM and only found three similar users with RABM. This is because the indicators of pneumonia impact the similarity calculation so much when using ABM.

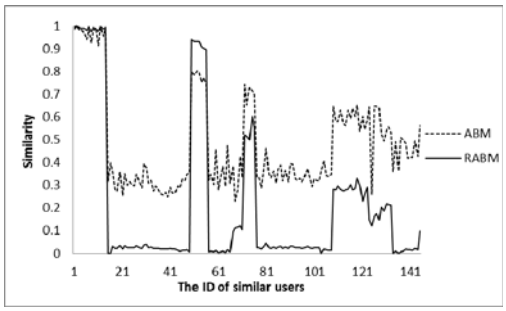

Fig. 4. Similarities of a user who have diabetes with other users

In Figure 4, we found users who have diabetes with both methods. But we can find some similar users who have both diabetes and hepatitis when using RABM. This is because RABM can decrease the influence of the common indicators of the two diseases during the similarity calculation.

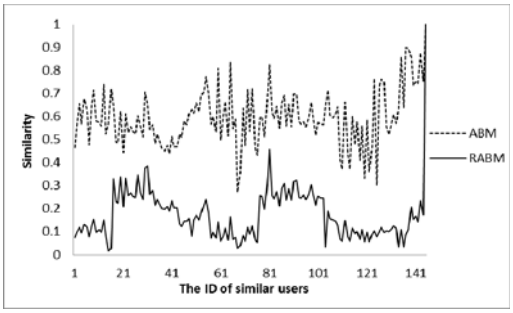

Fig. 5. Similarities of a user who have flu with other users

Flu is a fairly common disease and its related indicators are common for many diseases. So flu related indicators would be accessed more than other indicators. We found many similar users who have no flu when using ABM. But the results tend to be correct when we use RABM. This is because RABM can decrease the influence of the common indicators during the similarity calculation.

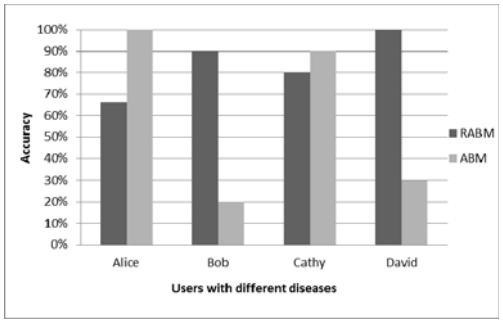

Fig. 6. Recommendation accuracy for users with different diseases where Alice has AIDS, Bob has both AIDS and pneumonia, Cathy has diabetes and David has flu.

We find the main indicator groups of every disease on a professional medical website. A main indicator group of a certain disease is a set of indicators that we can use to confirm whether someone has got the disease. For the selected similar users in Figure 2 to Figure 5, we simulate their audit logs on the basis of diseases they have. Then we calculate the recommendation priority of every incomplete audit commands obtained from their audit logs. In order to verify the accuracy of our methods, we select the number of recommendations depending on the diseases of the audit requester has, that is, we use the total number of main indicator groups of the user's all diseases as the recommendation number. Then we divide the number of right recommendations by the number of all recommendations to obtain the accuracy. Figure 6 is the accuracy analysis of the audit recommendation part and the four groups are corresponding to Figure 2 to Figure 5. In the Group of Alice, we can see that the recommendation accuracy is better when using ABM. This is because there are two users who have both AIDS and gastritis are selected as similar users when using RABM and the two users have done some audits about pneumonia. In the Group of Cathy, the accuracy of the two methods is almost the same because the similar users who have two diseases are found with RABM affect the accuracy, but the influence is small. In the Groups of Bob and David, RABM displays better than ABM does, this is because RABM can adjust the influence of different health indicators to gain more accurate results. When we use RABM to calculate the similarities, the weights of AIDS related indicators are increased in the Group of Bob, and the weights of flu related indicators are decreased in the Group of David.

VII. CONCLUSIONS

This paper presents an audit recommendation framework to help users appropriately audit their sensitive records so as to have a good understanding of their privacy situation. We use doctors' historical queries to calculate the similarity between two users. A weight vector of health indicators is used to adjust the influence of health indicators during the similarity calculation, which can improve the accuracy of the results. Then we analyze the audit commands that have been submitted by similar users and find some appropriate incomplete audit commands for an audit requester. We consider the similarities between the audit requester and other users during the calculation of recommendation priority. At last, we perform experiments on a set of simulated data to verify our methods. In the future, we will consider more about

the influence of time factor during the similarity calculation.

ACKNOWLEDGMENT

Part of this work is supported by the National Natural Science Foundation of China (61173140), the National Science & Technology Pillar Program (2012BAF10B03-3), the Science Foundation of Shandong Province (Y2008G28), and the Independent Innovation Foundation of Shandong University (2010JC010).

REFERENCES

[1] A. Sunyaev, D. Chomyi, C. Mauro, and H. Krcmar, "Evaluation Framework for Personal Health Records: Microsoft HealthVault vs. Google Health," Proc. 43rd Hawaii International Conference on System Sciences, Kauai, Hawaii, 2010, pp.1-10.

[2] I. Carrión, J.L. Fernández-Alemán and A. Toval, "Assessing the HIPAA Standard in ractice: PHR Privacy Policies," Proc. 33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2011, pp. 2380-2383.

[3] R. Agrawal, R. Bayardo, C. Faloutsos, J. Kieman, R. Rantzau and R. Srikant. "Auditing compliance with a hippocratic database," Proc. 30th International Conference on Very Large Data Bases, Toronto, Canada, 2004, pp.516-527.

[4] K. Raghav and R. Ravi. "Efficient auditing for complex sql queries," Proc. ACM's Special Interest Group on Management Of Data, Athens, Greece, 2011, pp.697-708.

[5] R. Motwani, S.U. Nabar, D. Thomas. "Auditing sql queries, " Proc. 24th International Conference on Data Engineering, Cancun, Mexico, 2008, pp.287-296.

[6] B. Stefan and H. Rita. "Information disclosure by answers to xpath queries," Journal of Computer Security, vol.17, pp.69-99, January 2009.

[7] D. Fabbri, K. LeFevre and Q. Zhu, "Policyreplay: misconfiguration -response queries for data breach reporting," VLDB Endowment, vol.3, pp.36-47, September 2010.

[8] W. Lu, G. Miklau. "Auditing a database under retention restrictions," Proc. 25th International Conference on Data Engineering, Washington, DC, USA, 2009, pp.42-53.

[9] G.X. Wang, H.P. Liu. "Survey of personalized recommendation system," Computer Engineering and Applications, vol.48, pp.66-76, 2012.

[10] Y. Chen, B. Malin. "Detection of anomalous insiders in collaborative environments via relational analysis of access logs," Proc. first ACM conference on Data and application security and privacy, New York,USA, 2011, pp.63-74.

[11] Y. Chen, S. Nyemba, and B. Malin. "Detecting anomalous insiders in collaborative information systems," IEEE Trans. Dependable Sec. Comput., vol.9, pp.332-344, May 2012.

[12] B. Sarwar, G. Karypis, J. Konstan, et al. "Item-based collaborative filtering recommendation algorithms," Proc. 10th International Conference on World Wide Web, New York, 2001, pp.285-295.