ICIC Express Letters
Volume **10**, Number **9**, September **2016**

ICIC International ©2016 ISSN 1881-803X
pp. 2205–2210

ON THE DEFINITION OF PERMISSION WEIGHT

Xiaopu Ma[1,*], Baoping Wang[1], Li Zhao[2], Ying Li[1] and Wenpeng Zhang[1]

[1]School of Software
[2]School of Computer and Information Technology
Nanyang Normal University
No. 1638, Wolong Road, Wolong District, Nanyang 473061, P. R. China
*Corresponding author: mapxiao@nynu.edu.cn

Received January 2016; accepted April 2016

Abstract. *There are many approaches proposed for role mining in the process of migrating the system from non role-based access control model to role-based access control model. However, a problem related to this which has not been addressed adequately is how to discover roles based on weight because the traditional methods often treat each permission evenly. In this paper, we provide a detailed analysis of the requirements for permission. Then we formally define the weight of permission based on the requirements. To support our claim, we present an algorithm based on the simple definition of permission weight to assess the role mining results. Experiments on performance study prove the superiority of the algorithm.*

Keywords: Role-based access control, Role engineering, Permission attribute, Permission weight

1. Introduction.

Role-Based Access Control (RBAC) model has several key benefits rather than other methods in protecting system security [1]. Therefore, RBAC has become the norm access control model in enterprise security management and enterprise management products. Hence, how to facilitate the non RBAC system to RBAC system has become the major challenge in implementing the advantages of RBAC. As a solution to constructing RBAC system, role engineering is introduced [2].

Essentially, RBAC system can be constructed using two basic approaches: one is top-down and the other is bottom-up. In top-down approach, roles can be generated by allocating the needed permissions to create roles for each business process based on analyzing each business function. In this approach, it is time consuming and costly because there may be dozens of business processes and tens of thousands of users in an organization. In bottom-up approach, RBAC system can take business information, also called "top-down information", together with the pre-existing user to permission assignment relationships as input to generate roles by data mining techniques. This approach has raised significant interests because much of its process can be automated or semi automated to generate potential or candidate roles [3].

In role mining field, the typical algorithms, though effective in certain situations, are not sufficient to describe the complex access control policies in today's collaborative environments. Take role mining approach for example: because the traditional method treats the permission evenly, it does not consider any other characteristic that a user may demonstrate. Further, the traditional role mining method generally does not take into account the characteristics of resources; nor does it capture any security relevant information of the environment. Hence, a certain number of roles may not be identified by the traditional role mining techniques.

In this paper, we focus on how to introduce the different attributes to define the permission weight and how to mine roles based on permission weight. The rest of the paper is organized as follows. We discuss related work in Section 2. Section 3 and Section 4 propose our notation about how to define the permission weight and how to mine roles based on permission weight. A summary of our experimental results on simulated data is discussed in Section 5. Finally, Section 6 provides some insight into our ongoing and future work.

2. Related Work.

According to the different principles, we can divide the role mining algorithm into four approaches.

1) In the first approach, the algorithm can generate a set of candidate roles and then give every role a priority value. The typical algorithms are CompleteMiner (CM) and FastMiner (FM) where the CM algorithm finds the unique intersection sets from generated roles, while the FM algorithm only finds the intersection between pairs of initial roles [4].

2) In the second approach, the algorithm will generate a complete RBAC state using Weighted Structural Complexity (WSC) as the common quality measurement. The representative algorithms are HierarchicalMiner (HM) [5] and GO [6].

3) In the third approach, the algorithm will take the different constraints into account in role mining. According to the different constraints, the role mining algorithm based on constraints can be divided into six categories.

- The first class constrained role mining algorithm satisfies a cardinality condition that no role contains more than a given number of permissions, such as Constrained Role Miner algorithm (CRM) [7].
- The second class constrained role mining algorithm satisfies a cardinality condition that no role contains more than a given number of users. In this class, Hingankar and Sural propose a biclique cover method to generate a set of roles that limits the maximum number of users for a role [8].
- The third class constrained role mining algorithm satisfies a cardinality condition that no user contains more than a given number of roles. The representative algorithm of the third class Role Priority based Approach (RPA) and Coverage of Permissions based Approach (CPA) [9].
- The fourth class constrained role mining algorithm satisfies a cardinality condition that no permission belongs to more than a given number of roles. The typical algorithm is post-processing framework and concurrent processing framework proposed by P. Harika et al. [10].
- The fifth class constrained role mining algorithm satisfies the separation of duty or exception constraint that is proposed by H. Lu et al. [11]. In this algorithm, when the user obtains the negative permissions, the user cannot activate the negative permissions for ever; when the role obtains the negative permissions, the role cannot activate the negative permissions for ever.
- The sixth class constrained role mining algorithm satisfies more than one constraints. For example, Ma et al. propose a role mining algorithm to generate roles based on permission cardinality constraint and user cardinality constraint [12]. R. Li et al. propose a role mining algorithm to generate roles based on the cardinality constraints of roles and permissions [13].

4) In the fourth approach, the algorithm can generate role based on weight. The representative algorithm is WRM (Weighted Role Mining). However, it does not consider different characteristics to reflect the permission weight [14, 15].

3. Problem Statement and Preliminaries.

In this paper, we follow the basic definitions in NIST standard, which is the most widely known as formal description of RBAC model [16].

Definition 3.1. **(RBAC Model)** *The RBAC model contains the following components:*

- *Users, Perms, Roles, the set of users, permissions and roles respectively;*
- $UA \subseteq Users \times Roles$*, a many to many relationship between users and roles;*
- $PA \subseteq Perms \times Roles$*, a many to many relationship between permissions and roles;*
- $UPA \subseteq Users \times Perms$*, a many to many relationship between users and permissions;*
- $auth\_perms(r) = \{p \in Perms | (p, r) \in PA\}$*, the mapping of role* r *onto a set of permissions;*
- $auth\_perms(u) = \{p \in Perms | (u, p) \in UPA\}$*, the mapping of user* u *onto a set of permissions;*
- $auth\_users(p) = \{u \in Users | (u, p) \in UPA\}$*, the mapping of permission* p *onto a set of users.*

In practice, the weight of permission is a value attached to a permission representing its importance. Furthermore, a role is a collection of permissions and permission can be split into operations and resources. Operation and resource carry the function and action information of permission. Such a split approach can provide good interpretability of a permission. Depending on the domain, there could be any variable ranging from the types of operations to the types of resources. Hence, for defining the permission weight, we are concerned with four types of attributes.

Definition 3.2. (**Permission Attribute**) *The attribute affects the permission weight that can be included in the following types:*

- *User Attributes (UAt): A user is an entity that takes operation on resources. Each user has associated attributes which define the identity and characteristics of the user. Such attributes may include the user's identifier, name, organization, location, department affiliations, task description, and so on. Here, we can use* UAt_i *to describe the attributes of user* u_i*;*
- *Operation Attributes (OAt): An operation can carry the function and action information of permission, such as add, create, and write. Here, we can use* OAt_i *to describe the permission to operation mapping, which gives the ith operation attribute associated with permission* p_i*;*
- *Resource Attributes (RAt): A resource is an entity that is operated upon by a user. As with users, resources have attributes that can be leveraged to make access control decisions. A Microsoft Word document, for example, may have attributes such as title, subject, date, and author. Resource attributes can often be extracted from the "metadata" of the resource. Here, we can use* RAt_i *to describe the permission to resource mapping, which gives the ith resource attribute associated with permission* p_i*;*
- *Environment Attributes (EAt): These attributes describe the operational, technical, and even situational environment or context in which the information access occurs. For example, attributes such as current date and time, the current virus or hacker activities, and the network's security level, are not associated with a particular subject nor a resource, but may nonetheless be relevant in applying an access control policy. Here, we can use* EAt_i *to describe the permission to environment mapping, which gives the ith environment attribute associated with permission* p_i*.*

According to the above permission attribute definition, we can define the permission weight based on these attributes as follows.

Definition 3.3. (**Permission Weight**) *The weight of permission* p_i *is defined as*

$$\omega_{p_i} = \mathrm{F}\left(UAt_1, \ldots, UAt_n, OAt_1, \ldots, OAt_n, RAt_1, \ldots, RAt_n, EAt_1, \ldots, EAt_n\right)$$

where $\omega_{p_i} \in [0, 1]$. Since the role is a set of permissions, we must define weight for all roles. This can be described as follows. For any $r_i \in Roles$ $(i = 1, 2, \ldots, m)$, suppose $auth\_perms(r_i) = \{p_1, p_2, \ldots, p_k\}$, we define the weight of r_i as follows.

Definition 3.4. (**Role Weight**) *The weight of r_i is defined as*

$$\omega_{r_i} = \mathrm{F}(\omega_{p_1}, \omega_{p_2}, \ldots, \omega_{p_k}) \qquad where \qquad \mathrm{F} : R^k \to [0,1]$$

4. Algorithms.

In data mining field, there are many algorithms to generate frequent item sets on database containing transaction, such as FP-growth, Apriori, and Eclat. In constructing RBAC system, we regard permissions as items, pre-existing user permission relationship assignments as database and each user permission relationship as transaction. We can analogously use this idea for extracting frequent permission sets based on weight and then define each permission set as a role. This process of the Apriori algorithm based on permission weight to generate the roles can be described as follows.

In the first stage, we generate the candidate permission set Y (Y has q permission, $q = 1, 2, \ldots, k$, where k is the number of permission in pre-existing user permission assignments) if the count number of permission set Y in pre-existing user to permission relationship assignments is greater than or equal to any of the k-upper bound of the permission set Y.

Here, the count number of permission set Y is the number of users which possess permission set Y presented in the user to permission assignments. The k-upper bound of the permission set Y is defined as follows.

Definition 4.1. (k-Support Bound of Y) *The k-upper bound of the permission set Y is defined as*

$$B(Y,k) = \left\lceil \frac{wminsup \times numUsers(all)}{\sum_{p_i \in Y} \omega_{p_i} + \sum_{j=1}^{k-q} \omega_{p_j}} \right\rceil$$

where $numUsers(all)$ is the total number of users in the pre-existing user-permission assignments, $wminsup$ is the weighted support threshold, $\sum_{p_i \in Y} \omega_{p_i}$ is the sum permission weight in Y, and $\sum_{j=1}^{k-q} \omega_{p_j}$ is the maximum sum permission weight for the remaining permissions besides Y in pre-existing user permission assignment relationships. The more details to generate the candidate permission sets are based on weight described in [17].

In the second stage, we generate the frequent permission sets based on the candidate permission sets as roles if the candidate permission sets satisfy the following equation

$$\sum_{p_i \in Y} \omega_{p_i} \times \frac{numUsers(Y)}{numUsers(all)} \geq wminsup$$

where $numUsers(Y)$ is the number of users containing permission set Y in pre-existing user to permission assignments.

5. Experimental Results.

In this section, we will implement the proposed method of obtaining the roles based on permission weight on an Intel(R) Core(TM) i5-4590 @CPU 3.30GHz machine with 4GB memory to evaluate how well our method performance. From above, we can derive the user attributes, operation attributes, resources attributes, and environment attributes can affect the permission weight; however, if we cannot get these attributes, we can define the weight of permission simply based on the user similarity and permission similarity as follows.

Definition 5.1. (User Similarity) *The similarity between the ith user and the jth user is defined as*

$$sim(u_i, u_j) = \frac{|auth\_perms(u_i) \bigcap auth\_perms(u_j)|}{|auth\_perms(u_i) \bigcup auth\_perms(u_j)|}$$

Definition 5.2. (Permission Similarity) *The similarity between the ith permission and the jth permission is defined as*

$$sim(p_i, p_j) = \frac{|auth\_users(p_i) \bigcap auth\_perms(p_j)|}{|auth\_users(p_i) \bigcup auth\_users(p_j)|}$$

Definition 5.3. (The Simple Permission Weight) *The simple weight of permission p_i is defined as*

$$\omega_{p_i} = 1 - \left(\alpha \times \frac{\sum_{j=1, j\neq i}^{numPerms(all)} sim(p_i, p_j)}{numPerms(all) - 1} \right.$$
$$\left. + \beta \times \frac{\sum_{m=1 \wedge u_m \in auth\_users(p_i)}^{numUsers(all)} \sum_{n=1, n\neq m}^{numUsers(all)} sim(u_m, u_n)}{|auth\_users(p_i)| \times |numUsers(all) - 1|} + \gamma \times \omega_0 \right)$$

where $numPerms(all)$ is the total number of permissions in the user permission assignments, ω_0 is the initial weight of permission p_i, and α, β, γ $(\alpha+\beta+\gamma = 1)$ are parameters used to adjust the relative importance to the weight of permission corresponding to the different factors. We mainly consider five weight schemes $W1$: $\alpha = 0$, $\beta = 1$, $\gamma = 0$; $W2$: $\alpha = 1/3$, $\beta = 2/3$, $\gamma = 0$; $W3$: $\alpha = 1/2$, $\beta = 1/2$, $\gamma = 0$; $W4$: $\alpha = 2/3$, $\beta = 1/3$, $\gamma = 0$; and $W5$: $\alpha = 1$, $\beta = 0$, $\gamma = 0$.

Figure 1(a) shows the number of roles in the different minimum weighted support under the different weight schemes when there are 50 users and 20 permissions. It can be seen that when the minimum weighted support threshold is low, there is a large number of roles. As the minimum weighted support threshold increases, the number of roles will decrease. Furthermore, we can find that the number of roles will increase when we increase the importance of the similarity between permissions to the permission weight. For example, at the same minimum weighted support threshold 0.4, the numbers of roles are 65, 65, 65, 83, 121 under parameters $W1$, $W2$, $W3$, $W4$, $W5$ respectively. It may mean that the similarity between permission can impact more on the weight of each permission. Figure 1(b) shows the average search time in the different minimum weighted support thresholds under the different parameters. It can be seen that when the minimum weighted support threshold is larger, the searching time will decrease.

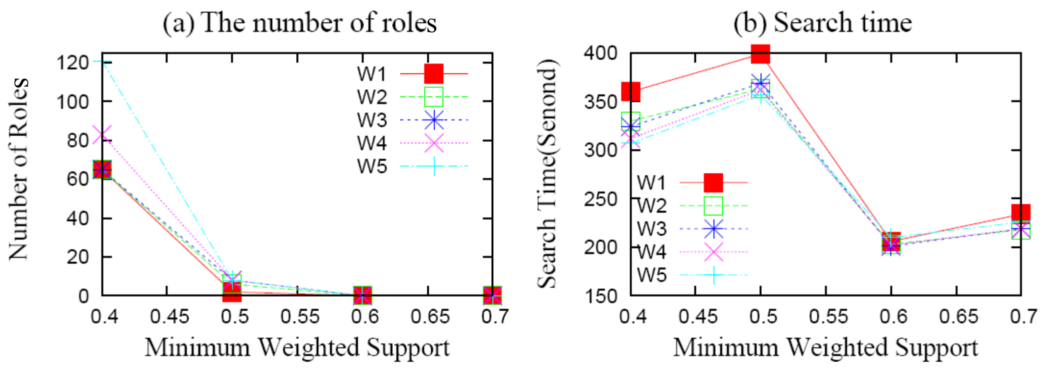

FIGURE 1. Performance analysis under 50 users and 20 permissions under parameters $W1$, $W2$, $W3$, $W4$, $W5$

6. **Conclusions and Future Work.** While many role mining approaches have been proposed recently, most of them did not consider the nature and importance of permissions. In this paper, we define the weight of permission based on the different attributes and also propose an algorithm to mine roles based on weight. As a result, the proposed approach can generate the different roles that can consider the different importance of

permissions. For the future work, we will try to find more meaningful information that is available to make the weight of permissions more accurately.

Acknowledgment. This work is supported by the Joint Fund for National Natural Science Foundation of China and Henan Province for Fostering Talents under Grant No. U1304619. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] D. F. Ferraiolo, R. Sandhu, S. Gavrila et al., Proposed NIST standard for role-based access control, *ACM Trans. Information and System Security*, vol.4, no.3, pp.224-274, 2001.

[2] E. J. Coyne, Role engineering, *ACM Workshop on Role-based Access Control*, p.4, 1996.

[3] D. Ferraiolo, J. Cugini and R. Kuhn, Role-based access control (RBAC): Features and motivations, *Proc. of the 11th Annual Computer Security Applications Conference*, pp.241-248, 1995.

[4] A. Baumgrass, M. Strembeck and S. R. Ma, Deriving role engineering artifacts from business processes and scenario models, *Proc. of the 16th ACM Symposium on Access Control Models and Technologies*, pp.11-20, 2011.

[5] I. Molloy, H. Chen, T. Li, Q. Wang, N. Li, E. Bertino, S. Calo and J. Lobo, Mining roles with semantic meanings, *Proc. of the 13th ACM Symposium on Access Control Models and Technologies*, Colorado, USA, pp.21-30, 2008.

[6] D. Zhang, K. Ramamohanarao and T. Ebringer, Role engineering using graph optimisation, *Proc. of the 12th ACM Symposium on Access Control Models and Technologies*, Antipoles, France, pp.139-144, 2007.

[7] R. Kumar, S. Sural and A. Gupta, Mining RBAC roles under cardinality constraint, *Proc. of the 6th International Conference on Information Systems Security*, pp.171-185, 2010.

[8] M. Hingankar and S. Sural, Towards role mining with restricted user-role assignment, *Proc. of the 2nd Wireless VITAE Conference*, pp.1-5, 2011.

[9] J. C. John, S. Sural, V. Atluri and J. S. Vaidya, Role mining under role-usage cardinality constraint, *Proc. of the 27th Conference on Information Security and Privacy*, pp.150-161, 2012.

[10] P. Harika, M. Negajyothi, J. C. John, S. Sural, J. Vaidya and V. Atluri, Meeting cardinality constraints in role mining, *IEEE Trans. Dependable and Secure Computing*, vol.12, no.1, pp.71-84, 2015.

[11] H. Lu, J. Vaidya, V. Atluri and Y. Hong, Constraint-aware role mining via extended boolean matrix decomposition, *IEEE TDSC*, pp.655-669, 2012.

[12] X. Ma, R. Li, H. Wang and H. Li, Role mining based on permission cardinality constraint and user cardinality constraint, *Security and Communication Networks*, vol.8, no.13, pp.2317-2328, 2015.

[13] R. Li, H. Li, X. Gu, Y. Li, W. Ye and X. Ma, Role mining based on cardinality constraints, *Concurrency and Computation: Practice and Experience*, vol.27, no.12, pp.3126-3144, 2015.

[14] X. Ma, R. Li and Z. Lu, Role mining based on weights, *Proc. of the 15th ACM Symposium on Access Control Models and Technologies*, pp.65-74, 2010.

[15] R. Li, W. Wang, X. Ma, X. Gu and K. Wen, Mining roles using attributes of permissions, *International Journal of Innovative Computing, Information and Control*, vol.8, no.11, pp.7909-7923, 2012.

[16] X. Ma, J. Wang, L. Zhao and R. Li, Mutual exclusion role constraint mining based on weight in role-based access control system, *International Journal of Innovative Computing, Information and Control*, vol.12, no.1, pp.91-101, 2016.

[17] C. H. Cai, W. C. Fu, C. H. Cheng and W. W. Kwong, Mining association rules with weighted items, *Proc. of the 1998 International Symposium on Database Engineering and Applications*, pp.68-77, 1998.