The RBAC System Based on Role Risk and User Trust

Caiyuan Jin, Aodong Shen*, Wenxue Yu
School of Computer Science and Engineering, Southeast University in Nanjing, Jiangsu Province, China.

* Corresponding author. Tel.: +86 02583794249; email: shen.list@seu.edu.cn
Manuscript submitted October 15, 2015; accepted January 6, 2016.
doi: 10.17706/ijcce.2016.5.5.374-380

Abstract: The risk problem of access control model is always the key point of security management. In this paper, we introduce the risk factor to control the risk of role under the proposed threshold when building RBAC policy based on role mining. At the same time, a user should meet some demands to activate roles. So we propose the following principal. A user could activate a role when his trust is higher than the trust threshold of the candidate role. This method makes role authority more reasonable and it avoids of the abuse of permissions. Experiments on performance indicate that the proposed algorithm can not only build the complete RBAC system to avoid roles' high risk caused by the high complexity of permissions in roles, but also meet the least privilege when a user activates a role.

Key words: RBAC, role mining, role risk, user trust.

1. Introduction

Nowadays role based access control RBAC [1] is one of the most popular access control models. In the model, users do not grant permissions directly but perform permissions by activating roles. This model realizes the separation between users and permissions to simplify the security management.

In order to build and maintain RBAC model, role mining [2] is of great concern. Role mining finds roles from system configuration by using data mining methods. It decreases the cost of distributing roles manually and makes the RBAC system more concise.

Normally, the present role mining algorithms aim to find the minimal state of role state [3] and ignore that the role itself as an entity in access control model may cause risk. When the allocation of permissions between roles is improper it would reduce the security level of the model. Also, if the activation policy is improper when a user activates a role, it would cause the role's misuse and abuse.

So in this paper we introduce the role risk to make sure the produced roles' risks are all under control when mining roles. This decreases the whole system's risk. And then we introduce user trust and user trust threshold in role in order to make sure the proper role is activated and meet the least privilege [4] in RBAC.

There are little related works. Ref. [4] and [5] divide the risk system may cause into four matrices. They are role similar matrices, compliance to policy matrix, trust matrix and abuse and misuse matrix. However, they only consider the risk while mining roles and do not take role activation problem into consideration. Baracaldo and Joshi [6] mainly considers the inside threats including permissions' abuse and misuse and users' trust. This method is based on the well-formed RBAC policy and it cannot take building risk into consideration. Ref. [7] combines the role risk and role inheritance and introduces risk band by the inheritance of risk.

The rest of this paper is as follows. The second section introduces the preliminaries used in the paper, including how to compute permissions' weight. The third section is about the calculation rules of role risk based on permissions' weight. In section four, we introduce the notation of user trust and role's user trust threshold and give the role activation algorithm. In section five, we draw the conclusion and introduce the future work.

2. Preliminary

In this paper, we use the standard RBAC model [8] including $RBAC_0$, $RBAC_1$, $RBAC_2$ and $RBAC_3$. And in a RBAC model, there are always the following elements.

Definition 1: The RBAC model includes the following elements:

- U, P, R represent user, permission and role respectively;
- $PA \subseteq P \times R$, a many-to-many mapping of permission to role assignments;
- $UA \subseteq U \times R$, a many-to-many user to role assignment relationships;
- $RH \subseteq R \times R$, the inheritance of roles to roles;
- $P(R) = \{p \in P | (p, R) \in PA\}$, the mapping of role R onto a set of permissions.
- $U(R) = \{u \in U | (u, R) \in UA\}$, the mapping of role R onto a set of users.
- $U(P) = \{u \in U | (u, P) \in UPA\}$, a set of users who have the permission P.

Role mining results to build a complete RBAC model based on the input of user-permission matrix. The definition of role mining is as below.

Definition 2: Given an access control configuration $\rho =< U, P, UPA >$, where U is a set of all users, P is a set of all permissions and $UPA \subseteq U \times P$ is the user-permission relation. We want to find an RBAC state <R, UA, PA, RH> that is consistent with ρ.

Our role mining algorithm is based on ORCA [9]. In ORCA, it treats each permission as an original permission cluster and gets its user members. Then it finds the pairs of clusters with a maximal overlap among their members and merges them to produce a new permission cluster. Repeat until the clusters are stable.

Since permission is represented by user sets and users can be described using permission sets, we can give a measure of similarity between users and permissions based on this.

Definition 3: The similarity between the two permissions is defined as follow,

$$\mathrm{sim}(p_i, p_j) = \frac{|U(p_i) \cap U(p_j)|}{|U(p_i) \cup U(p_j)|} \tag{1}$$

where $|\cdot|$ represents size of the set. This measure is based on a statistical similarity measure called Jaccard co-efficient [10], where 0 implies no similarity between two permissions and 1 represents an exact match between these two permissions.

The weight of permission means how important the permission is. The weight can be computed by similarity. Intuitively, permission which has more users is more basic and the similarity with other permission is higher. So we can use the reciprocal of similarity to describe the notion of weight.

Definition 4: The weight of permission can be seen as the function of the similarity between permissions, the function is as follows.

$$w_{p_i} = \gamma \times \frac{n-1}{\sum_{j=1, j \neq i}^{n} \mathrm{sim}(p_i, p_j)} + (1 - \gamma) \times w_0 \tag{2}$$

w_{p_i} means the weight of permission p_i and w_0 is the initial weight of permission p_i preset by the system based on the knowledge of comprehensive effect of all factors on permission p_i; γ is parameters used to

adjust the relative importance about the weight of permission p_i corresponding to the similarity and the initial weight. If we have no prior knowledge, we can set γ to 1.

For example, Table 1 gives the user-permission matrix as system configuration. Each row represents a user and each column represents permission. If a user possesses permission, the corresponding place is 1. Else the corresponding place is 0. According to formula 2, the weight of each permission is shown in Table 2. In Table 2, the weight of permission P_1 is low because P_1 has the largest number of users. P_3 and P_5 have less number of users and their weights are higher. This corresponds to the above description.

Table 1. User-Permission Matrix

	P_1	P_2	P_3	P_4	P_5
U_1	1	1	0	1	0
U_2	1	1	0	1	0
U_3	1	0	0	0	0
U_4	1	0	1	0	1
U_5	1	1	1	1	1
U_6	1	1	1	1	1

Table 2. Weight of Each Permission

P_1	P_2	P_3	P_4	P_5
1.714	2.0	2.182	2.0	2.182

Table 3. Roles and Risks

<P_2,P_4>	<P_1,P_2,P_4>	<P_3,P_5>
0	0.135	0.0

3. Role Mining Based on Role Risk

Role itself has risk and ORCA only considers the maximal overlap of user members. In order to solve the risk caused by ORCA, we introduce the notion of role risk factor to control the risk of role while role mining and make the risk of the whole RBAC state controllable.

Since role is cluster of permissions, the risk of role can be measured by the complexity of permissions included in the role.

As we all know, standard deviation is a measure of the complexity of data sets. So we can use standard deviation of permissions' weight in role to measure the complexity of a role.

Definition 5: The role risk can be defined by the standard deviation of permissions' weights in a role,

$$r(R) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(w_{p_i} - \mu\right)^2} \tag{3}$$

where n is the number of permissions in the role r. μ is the average weight of permissions in role r.

If we want to control the risk of roles while role mining, the risk threshold should be set. While role mining, when a new role is going to be generated, if the new role's risk is larger than the risk threshold, the new role would not be generated. In this paper, we compute the standard deviation of weights of all permissions to get the risk threshold.

Definition 6: The risk threshold is defined as follows,

$$r = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(w_{pi} - \mu_0)^2} \tag{4}$$

where N is the total number of permissions, μ_0 is the average weight of all permissions.

In general, it is reasonable that the risk of role is critical to determine this role would be created. So, the threshold is important to affect the final results as a ruler. Combined the risk of role, the whole algorithm of role mining based on role risk is depicted in Algorithm 1.

The algorithm groups permissions into clusters, associates any person with the cluster that has all the permissions in the cluster, and keeps those clusters with a significant set of persons. The algorithm first treats each permission as a role (Step 2). And then find the pairs of roles with a maximal overlap among their members (Step 4). Cluster them into a temporary role. If the risk of the clustered role is over the risk threshold, do not add the role into the cluster (Step 5). Otherwise, add the clustered role into the cluster and add the role and the pairs into RH, role inheritance. Repeat the above steps until the cluster and RH stable (Step 6).

According to Table 1, roles based on the algorithm 1 are shown in Table 3. As show in Table 3, all roles mined are under risk threshold.

Algorithm 1: Role mining based on role risk

Input:
User-permission matrix: UPA
Output:
Roles: the set of all roles in RBAC
RH：Role inheritance relationship

1. Initial parameters:
   Roles $= \emptyset$, RH $= \emptyset$
2. For each permission p, define role r
   $P(r) = \{p\}$;
   $U(r) = \{u | < u, p > \in UPA | < p, r > \in PA\}$;
   Add r to Roles, Roles $=$ Roles $\cup \{r\}$;
3. For pair <$r1$, $r2$> that not selected in Roles, define:
   $U(< r1, r2 >) = U(r1) \cap U(r2)$;
   $P(< r1, r2 >) = P(r1) \cup P(r2)$;
4. Find the pairs in Roles with a maximal overlap among their members. If there are more than one pair, we choose the pair in which two roles have the most similar role weight. Now do the following operation between two roles in the pair.
   $r = \{U < r1, r2 >, P < r1, r2 > | \max(|U < r1, r2 > |)\}$;
5. if risk(r) >= r
   do not generate this role, namely do not add it to Roles.
   Mark these two roles in order to ignore the pair generated by the two roles.
   else
   Roles $=$ Roles $\cup \{r\}$;
   RH $=$ RH $\cup \{< r, r1 >, < r, r2 >\}$;
6. Repeat Step 3 to Step 5 until the Roles is stable;

4. User Trust and Role Activation

In order to meet least privilege, the proper role should be activated by user with proper activation policy that can promise roles wouldn't be misused or abused.

In this paper we introduce user trust and trust threshold of a role to verify if the user has the permission to activate the role. Although the input is only the user-permission matrix, which has little information that can be used, the matrix is widely used and easy to get. So there is still value to research.

Definition 7: We use the max weight of permissions the user possesses to indicate the highest authority the user can exercise.

$$t(u) = \max\{w_p | \forall p \in P, < u, p > \in UPA\} \quad (5)$$

It is important to avoid the abuse of permissions according to the user's trust. Once one user's trust is higher than the maximal value, the risk would be introduced into the system. However, existed algorithms seem to consider the problem not at all. So, the trust threshold is critical to active some roles for one user. The next definition is described to explain the trust threshold as follows:

Definition 8: We use the min weight of permissions in role to be the trust threshold.

$$t(r) = \min\{w_p | \forall p \in P, < r, p > \in PA\} \quad (6)$$

The activated algorithm we propose first selects a set of roles which contain the permission the user apply for and the trust threshold of which is below the user's trust. And then we find the least risk role to assign to user. It can not only promise the correct and efficient manage of roles and users but can also make the risk less which caused when a user activates a role. Algorithm 2 shows the detail of role activation algorithm.

The role activation algorithm first scans the whole role cluster. And for each role, if the permission applied by user is in the role and the user's trust is over the role's trust threshold, add the role to the candidate roles that the user may activates (Step 2). If the candidate roles are null, that means the user cannot activate any role (Step 3). Otherwise, for each role in candidate roles we find the role which has the minimum trust threshold for the user to activate (Step 4).

Algorithm 2: Role Activation Algorithm

Input:
RBAC: the whole RBAC state
$r(R)$: the risk of role R
$t(R)$: user trust threshold for each role R
U: the user applying for permission
P: the permission the user is applying for
$t(U)$: the user trust for user U
Output:
Role: the role that the user can activate
1. canRoles $= \emptyset$
2. for each $r \in R$
   If $P \in r$ && $t(U) > t(r)$;
   canRoles = canRoles $\cup \{r\}$;
3. if canRoles $== \emptyset$
   return null; that means there are no roles current user can activate
4. for each $r \in$ canRoles, find the min$\{t(r)\}$, Role=r

Return Role;

According to the algorithm 2, we can find that the user's trust is higher than the threshold of trust for each activated role. At the same time, activated roles meet the least privilege as common. Some experiments have executed to validate the algorithm, for simple, we depict one of the serials as following.

Table 4. User Trust for Each User

U_1	U_2	U_3	U_4	U_5	U_6
2.0	2.0	1.714	2.182	2.182	2.182

Table 5. Trust Threshold for Each Role

$<P_2,P_4>$	$<P_1,P_2,P_4>$	$<P_3,P_5>$
2.0	1.714	2.182

For the user-permission matrix given by Table 1 and roles given by Table 3, the user trust for each use and the trust threshold for each role are shown in Table 4 and Table 5. From the tables we can see, the algorithm proposed in this paper can activate the proper role and avoid the abuse and misuse of roles.

5. Conclusion and Future Work

We propose a method to assess and control role risk while building RBAC model based on role mining. This method can make sure the mined roles more secure and stable. Because of this, the security level of the model is improved. The proposed method can avoid the abuse and wrong use of permissions. The generated roles are qualified tested by the computed threshold in the configuration.

After that, the role activation algorithm is proposed which is based on the user trust and trust threshold to make sure the user can be trusted. It is sure that each permission is assigned to the proper user with the qualified privilege.

Table 6 gives the comparison results of various methods with our presented method from the mean value, the maximum value of risk. The dataset is computed by the FastMiner method used 60 users and 200 permissions as inputs.

Table 6. The Comparison Results

Method	Threshold	Mean	Maximum
Algorithm1	0.283	0.192	0.280
ORCA	0.283	0.144	0.551
Fast Miner	0.283	0.248	0.352

Given the same threshold value, the different mean value and maximum value can be shown from Table 6. It demonstrates that the original ORCA method can produce the lowest and highest risk roles based on mining but the lower mean risk. And then, the roles with similar risk value would be mined by the Fast Miner method as the threshold, therefore the results would be likely to dangerous. The appropriate mean value shows the mined role set are lower risk introduced based on our method. Various experiments have exercised to show that the groups of roles and users are feasible and valuable. The risk role and user trust will be efficient to mine RBAC with the high confidence level.

Especially, the user-permission matrix is the original input of our algorithm. It is simple and reasonable because there is less or none operation logs in a new access system and the configuration is given by manager. The configuration may have some errors or mistakes and this may cause the role mined is not suitable.

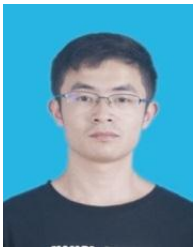
However, in the future, with the increase of operation logs, we can analyze these logs to optimize these roles. Also users may have some attributes such as age, position. The information can also be used to improve the correctness of role mining.

Acknowledgements

This work was granted by Prospective Study Project in Jiangsu Province (Grant No. BY2014127-11), Specialized Research Fund for the Doctoral Program of Higher Education (Grant No. 20130092120035) and the Natural Science Foundation for the Youth of Jiangsu (Grant No. BK20150650).

References

[1] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., *et al*. (1996). Role-based access control models. *Computer*, *29(2)*, 38-47.
[2] Mario, F., Buhmann, J. M., & Basin, D. (2010). On the definition of role mining. *Proceedings of the 15th ACM Symposium on Access Control Models and Technologies* (pp. 35-44).
[3] Molloy, I., Chen, H., Li, T., *et al*. (2008). Mining roles with semantic meanings. *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies* (pp. 21-30).
[4] Ahmed, S., & Osborn, S. L. (2014). A system for risk awareness during role mining. *Proceedings of the 19th ACM Symposium on Access Control Models and Technologies* (pp. 181-184).
[5] Ahmed, S. (2014). *Application of Risk Metrics for Role Mining*.
[6] Baracaldo, N., & Joshi, J. (2012). A trust-and-risk aware RBAC framework: Tackling insider threat. *Proceedings of the 17th ACM Symposium on Access Control Models and Technologies* (pp. 167-176).
[7] Nissanke, N., & Khayat, E. J. (2004). Risk based security analysis of permissions in RBAC. *Proceedings of the 2nd International Workshop on Security In Information Systems* (pp. 332-341).
[8] Ferraiolo, D. F., & Kuhn, D. R. (2009). Role-based access controls. arXiv preprint arXiv:0903.2171.
[9] Schlegelmilch, J., & Steffens, U. (2005). Role mining with ORCA. *Proceedings of the Tenth ACM Symposium on Access Control Models and Technologie*s (pp. 168-176).
[10] Vaidya, J., Atluri, V., Guo, Q., *et al*. (2008). Migrating to optimal RBAC with minimal perturbation. *Proceedings of the 13th ACM Symposium on Access Control Models and Technologies* (pp. 11-20).

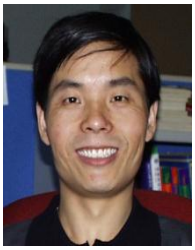


Caiyuan Jin was born in Suzhou, Jiangsu province on Feb. 27, 1991. Jin is a master student in Southeast University in Nanjing, Jiangsu, China majoring in computer science and engineering. Before that, Jin received the bachelor's degree of computer science at Nanjing Normal University in Nanjing, Jiangsu China in 2013.

He is now working in the Wenxue Yu's lab of image processing. His main research interests include data mining and role mining based on RBAC and access control model.

Aodong Shen was born in Jiangsu, China. Shen is a lecturer with the Image Processing Lab at College of Computer Science and Engineering, Southeast University, Jiangsu, China. He received the bachelor of electronic engineering degree in Xi'an Electronic and Science University, China in 2000. In 2003, Shen received the master degree of signal and communication engineering in Xi'an Electronic and Science University. In 2010, Shen received the doctor degree of biomedical engineering in Southeast University. Shen has published more than 10 papers.

His main research interests include image processing and access control model.



Wenxue Yu was born in Shandong, China. Yu is an associate professor with the Image Processing Lab at College of Computer Science and Engineering, Southeast University, Jiangsu, China. He received his master degree of experimental mechanics in Southeast University in 1997. In 2000, Yu received the doctor degree of biomedical engineering in Southeast University. In 2003, he became an associate professor of the College of Computer Science and Engineering, Southeast University. Yu has published more than 50 papers.

His main research interests include image processing and analysis, radiation therapy and computer aided diagnosis and treatment.