QUANTITATIVE ANALYSIS AND ENFORCEMENT OF THE PRINCIPLE OF LEAST PRIVILEGE IN ROLE-BASED ACCESS CONTROL

Chunren Lai
Department of Computer Science, University of Regina, Regina, SK, Canada S4S 0A2

Chang N. Zhang
Department of Computer Science, University of Regina, TRLabs, Regina, SK, Canada S4S 0A2

Keywords: Principle of least privilege, role based access control, system security, user-role assignment.

Abstract: Role-based access control (RBAC) models ease security administration and reduce overheads by introducing roles between users and privileges. RBAC provides the possibility to enforce the principle of least privileges that a user should be assigned just enough privileges to complete his/her job in order to prevent the possible information leaking and other wrong doing. This paper defines several concepts to quantitatively measure how well a user-role assignment meets the principle of least privilege and presents algorithms to find the perfect user-role assignment (i.e., without bringing any extra privilege) and the optimal user-role assignment (i.e., limiting any extra privilege to the minimum). The proposed approach for the enforcement of the principle of least privilege is particularly useful for automatic generation of user-role assignment in large-scale RBAC systems.

1 INTRODUCTION

Role-based access control (RBAC) ease security administration and reduce overheads by introducing roles between users and privileges (Giuri, 1997, Sandu, 1996, Ferraiolo, 2001, Bertino, 2001). One of the important features of RBAC models is that RBAC can apply constraints, enforce the principle of least privilege and other access control policies (Osborn, 2000, Ann, 2000, Ferraiolo, 1993). The principle of least privilege (Saltzer, 1975, Howard, 2003) requires that a user be assigned via roles just enough privileges to complete his/her job and receives minimum extra privileges. There are many applications that need to conform the principle of least privilege. In a task based RBAC system (Bertino, 1999, Zhang, 2003), a system security administrator often has to determine which set of roles should be assigned to a new user so that the new user can complete his/her tasks while conforming the principle of least privilege. Another application to conform the principle of least privilege is to assign a set of roles to a user to perform certain tasks in a role activation session.

Though the RBAC models enable the enforcement of the principle of least privilege, little research has been found in the literature about how to efficiently and correctly apply the enforcement of the principle of least privilege.

In this paper, we first introduce and define several concepts to quantitatively measure how well a user-role assignment meets the principle of least privilege. With respect to the target privileges required to complete a user's job, we can quickly determine whether or not there exists a perfect user-role assignment (i.e., without bringing any extra privilege except the ones required to complete the job). We then propose two algorithms to find the perfect user-role assignment if such an assignment exists, and to find out an optimal user-role assignment that has the smallest privilege leak in case that a perfect user-role assignment does not exist. The proposed analysis methods and algorithms are particularly useful in large-scale RBAC systems in which it is difficult to enforce the principle of least privilege by conventional methods.

The rest of this paper is organized as follows. Next section briefly describes the role-base access control models and the principle of least privilege. Section 3 defines several concepts to quantitatively measure the degree of how well a user-role assignment meets the principle of least privilege. In that section, we will also introduce the relationships among those concepts. In section 4, we present searching algorithms for the perfect and optimal user-role assignments. A brief conclusion is presented in section 5.

2 ROLE-BASED ACCESS CONTROL AND THE PRINCIPLE OF LEAST PRIVILEGE

There are several RBAC models proposed in the literature (Sandu, 1996, Ferraiolo, 2001, Bertino, 2001, Bertino, 1999, Zhang, 2003). Figure 1 shows the logic diagram of the RBAC96 model proposed by Sandu, 1996.

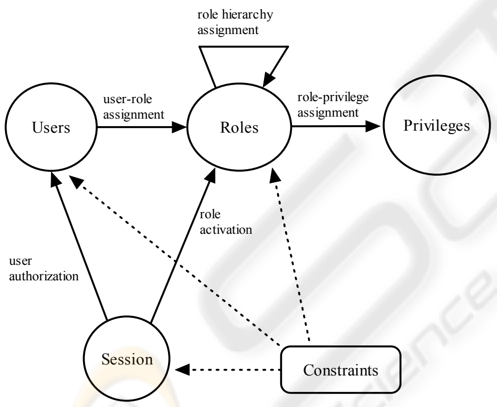

Figure 1: A typical RBAC96 model.

There are three basic sets in the RBAC96 model: U (a set of users), R (a set of roles that have hierarchical structure), and P (a set of privileges). Privileges are sometimes called permissions. There are three assignments: user-role assignment: UA $\subseteq$ U$\times$ R; role hierarchy assignment: RH $\subseteq$ R$\times$ R; role-privilege assignment: RP $\subseteq$ R$\times$ P.

The principle of least privilege was first introduced by Saltzer, 1975. That is, every user and every program of a system should operate using the least set of privileges necessary to complete the job (Saltzer, 1975). Essentially this principle limits the potential damage from any accident or error, and reduces the number of interactions among privileged program to the minimum.

The advantages of the principle of least privilege in the RBAC system are obvious. In one hand, it reduces the security leak when assigning roles to new users. In the other hand, by enforcing the principle of least privilege, only necessary roles will be activated in a session and thus it minimizes the resource consumption of a system and reduces errors.

3 QUANTITATIVE ANALYSIS FOR THE ENFORCEMENT OF THE PRINCIPLE OF LEAST PRIVILEGE

An example of RBAC system is shown in figure 2 where the role-hierarchical structure and direct role-privilege assignments are indicated.

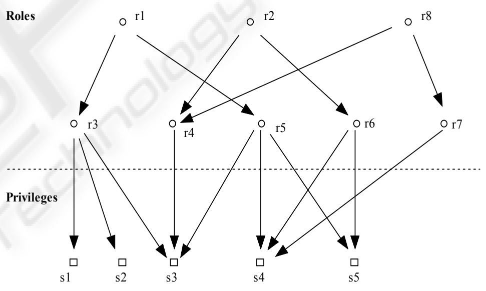

Figure 2: An example of RBAC system.

Suppose that we need to assign a new user A with a set of roles so that user A can complete a job that requires privileges s_3 and s_4. The question is that which set of roles should be assigned to user A in order to meet the principle of least privilege. To solve the problem, we first define the representation of the direct role-privilege assignments (DRPA). In general, the DRPA can be represented by an m$\times$n direct role privilege matrix, where m is the number of roles and n is the number of privileges. Each element of DRPA is denoted by $drpa[i,j]$. $drpa[i,j]=1$ if role r_i can directly access the privilege s_j, otherwise, $drpa[i,j]=0$. In RBAC, roles are organized as a hierarchical structure. For example in figure 2, role r_2 inherits the privileges from both roles r_4 and r_6, and therefore has privileges of s_3, s_4 and s_5.

By taking the role hierarchical structure into account, the DRPA can be extended into a role-privilege relationship matrix (RP) that includes direct and indirect role-privilege assignments.

Formally, the RP matrix can be defined as follows.

Definition 1: Assume there are m roles and n privileges in RBAC. A role privilege relationship matrix RP *is an* $m \times n$ *matrix where:* $rp[i,j]=1$ *if role* r_i *can access the privilege* s_j*, otherwise,* $rp[i,j]=0$.

In many RBAC systems, we need to express the relative significance of some privileges over others. For example, in most database applications, people often consider the privilege of "delete a record" action is more significant than the one of "read a record" action. The privilege weight matrix is to represent different levels of importance of the privileges. Formally, we have:

Definition 2: Assume there are n privileges in RBAC. A privilege weight array PW *is a* $1 \times n$ *matrix where its element* $pw[j]$ *is defined as the relative significance factor of the privilege* s_j *among the n privileges, where* $0 < pw[j] \le 1.0$ *for* $1 \le j \le n$.

An example of PW for the privileges in figure 2 can be:

$$PW = \begin{bmatrix} 1.0 & 0.5 & 1.0 & 1.0 & 0.5 \end{bmatrix}.$$

Definition 3: Assume there are n privileges in RBAC. A target privilege array TP *is a* $1 \times n$ *matrix where its element* $tp[j]$ *is defined by:* $tp[j]=1$ *if* s_j *is the target privilege, otherwise,* $tp[j]=0$ *for* $1 \le j \le n$.

The target privileges are the privileges that are required by a user to complete his/her job. We assume that the target privileges are s_3 and s_4 in our example, so the target privilege matrix is: $TP = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \end{bmatrix}$.

Let R be a set of l roles, i.e, $R = \{r_{k_j} \mid j = 1,\dots,l\}$ that is assigned to a user.

Definition 4: The privilege preservation degree β^R *is defined as follows:*

β^R *= (sum of weights of target privileges that* R *can access) / (sum of weights of privileges that* R *can access).*

For example in figure 2, we have: $\beta^{\{r_3\}}$ = pw[3] / (pw[1]+ pw[2]+pw[3]) = 1.0/2.5 = 0.4. Similarly, $\beta^{\{r_1\}}$ = 0.5.

β^R represents the degree in which how the role set R preserves the target privileges with respect to all the privileges that the role set R can access. If β^R =1, it means that the role set R can and only can access target privileges. If β^R <1, it means that the role set R can access at least one non-target privilege. A privilege leaking occurs when β^R <1.

Definition 5: Fulfillment degree γ^R *is defined as follows:* γ^R *= (sum of weights of target privileges that* R *can access) / (sum of weights of all target privileges).*

For the same example, we have, $\gamma^{\{r_3\}}$ = (pw[3]) / (pw[3]+pw[4]) = 1.0/2.0 = 0.5. Similarly, $\gamma^{\{r_1\}}$ = 1.0 and $\gamma^{\{r_4, r_7\}}$ = 1.0. If γ^R =1, it means that the role set R can access all the target privileges.

Definition 6: The overall satisfaction degree φ^R *is defined as follows:*

$\varphi^R = \beta^R \bullet \gamma^R$

If φ^R =1, it means that the role set R fully satisfies the principle of least privilege. If φ^R <1, it means that either R is not able to access all target privileges or R can access at least one non-target privilege.

Definition 7: A role set R *is perfect if* φ^R *=1.0*

According to the above definitions, we have:

Theorem 1: A single role r_k *is perfect if* $rp[k,j] = tp[j]$ *for all* $1 \le j \le n$.

Proof: If $rp[k,j] = tp[j]$ for all $1 \le j \le n$, it means that role r_k can and only can access target privileges, so we have $\beta^{\{r_k\}} = 1.0$ and $\gamma^{\{r_k\}} = 1.0$ according to definition 4 and definition 5, and thus $\varphi^{\{r_k\}} = 1.0$. According to Definition 7, we can conclude that the single role r_k is perfect.

In the example of figure 2, the single role r_8 is perfect.

Definition 8: Target privilege accessibility array TPA is an $m \times 1$ *matrix and defined by:* $tpa[i] = \vee_{j=1}^{n}(rp[i,j] \wedge tp[j])$, *for* $1 \le i \le m$, *where* $\wedge$ *is logic AND operation, and* $\vee$ *is logic OR operation.*

TPA defines the property of whether a role can access any target privilege. If $tpa[k]$=1, it means that the role r_k can access at least one target privilege. If $tpa[k]$=0, then the role r_k cannot access any target privilege.

Definition 9: Non-target privilege accessibility array NTPA is an $m \times 1$ *matrix and defined as:*

$$ntpa[i] = \vee_{j=1}^{n}(rp[i,j] \wedge (1 - tp[j]))$$

NTPA defines whether a role can access any non-target privilege. If $ntpa[k]$=1, it means that the role r_k can access at least one non-target privilege. If $ntpa[k]$=0, then the role r_k cannot access any non-target privilege.

Theorem 2:

Let $S = \vee_{i=1}^{m}(tpa[i] \wedge (1 - ntpa[i]))$. *If* S *=0, then there does not exist a role set* R *that is perfect.*

Proof: If S =0, then all $tpa[i] \wedge (1 - ntpa[i])$=0 for $1 \le i \le m$, so we have $tpa[i]$=0 or $ntpa[i]$=1 for any role r_i. It means that any role either cannot access any target privilege or can access at least one non-target privilege. That is, any role set R formed from any combinations of roles r_i ($1 \le i \le m$) is not perfect. It concludes that there is not perfect role set.

Theorem 2 implies that S=1 is the necessary condition for a role set to be perfect.

A perfect role set has an important property that is described by the following theorem. This property provides the guidelines to find the perfect role set efficiently.

Theorem 3: If a role set R *is perfect, then every role* r_k *in* R *satisfies:* $ntpa[k]$ *=0.*

Proof: If the role set R is perfect, then β^R=1.0 according to Theorem 1. From β^R=1.0, we know that none of the roles in β^R can access any non-target privilege. According to the Definition 9, we have $ntpa[k]$=0 for every role r_k in β^R.

Definition 10: A role r_w *(*$1 \le w \le m$*) is a must-in role of the target privilege* s_v *(i.e.,* $tp[v] = 1$, $1 \le v \le n$*) if* r_w *is the only one role that can access the target privilege* s_v. *A must-in role is marked as* $\hat{r}^v$.

The must-in roles are the roles that must be included in the user-role assignment. The existence of must-in roles can significantly reduce the searching time for the user-role assignment to conform the principle of least privilege. The following algorithm searches the must-in roles for each target privilege. If the must-in role does not exist for a target privilege, it is set to be ϕ.

```
Algorithm 1: find must-in roles for each target privilege.
    Input: $RP$, $TP$
    Output: must-in roles for each target privilege
    Step 1: For j=1 to n, do
    Step 1.1: If $tp[j] \ne 1$, then goto step 1
    Step 1.2: $\hat{r}^j = \phi$
    Step 1.3. k = $\sum_{i=1}^{m} rp[i,j]$; if $k \ne 1$, then goto step 1
    Step 1.4: For i=1 to m, do
        Step 1.4.1: if $rp[i,j] = 1$, then $\hat{r}^j = r_i$
    Step 2: Halt;
```

4 ALGORITHMS TO FIND USER-ROLE ASSIGNMENT MEETING THE PRINCIPLE OF LEAST PRIVILEGE

4.1 Algorithm to Find a Perfect User-role Assignment

```
Algorithm 2: find perfect role set $R$.
    Input: $RP$, $TP$, $PW$
    Output: A perfect role set $R$ or reporting "no solution".
    Step 1: Calculate $TPA$, $NTPA$, $S$
```

Step 2: If S =0, then report "No perfect role set solution", and goto step 7.

Step 3: For *i*=1 to m, do

Step 3.1: If $rp[i,j]=tp[j]$ for all $1\le j\le n$, then output $R=\{r_i\}$, and goto step 7.

Step 4: Assume *q* is the number of roles that satisfy: $tpa[i_z]=1$ and $ntpa[i_z]=0$, where $1\le z\le q$.

Step 5: For *b*=2 to *q*, do

Step 5.1: Pick up *b* roles from the *q* roles (i.e., r_{i_1}, r_{i_2}, …, r_{i_q}) to form the role set R_b.

5.1.1: If φ^{R_b} =1, then output the role set $R=R_b$, and goto Step 7.

Step 5.2: Repeat step 5.1 until no more combinations of *b* roles.

Step 6: Reporting "No perfect role set solution"

Step 7: Halt;

The strategy of Algorithm 2 to find a perfect role set R is: (1) If S=0, then there is no perfect role set based on Theorem 2, and the algorithm exits immediately; (2) If S=1, then step 3.1 checks whether any single role is perfect based on the Theorem 1. If such a single role is found, output the result and exit. (3) If no single role is a perfect role set, then consider any combinations of appropriate roles according to Definition 7. This is done in step 5.

For the example in figure 2, the single role r_8 is found as a perfect role set.

4.2 Algorithm to Find Optimal Role Set that Meets the Principle of Least Privilege

In the case that perfect role-assignment does not exist, we first search all possible role sets that have a target privilege fulfillment γ^R=1.0, and then sort them by the value of overall satisfaction degree φ^R. Based on our definition of φ^R, the bigger the value of φ^R is, the better the role set R meets the principle of least privilege.

The role set $\tilde{R}$ with the maximum φ value is called the optimal role set. We assume that the role constraints C should be taken into consideration when searching the role set.

Algorithm 3: find the role set that conforms the principle of least privilege

Input: RP, TP, PW, $DRPA$, C (role constraints)

Output: Either an optimal role set R that can access all target privileges, satisfies the constraints C and conforms the principle of least privilege, or reporting "no solution".

Step 1: k=0; find must-in roles for each target privilege based on algorithm 1;

Step 2: For each j that satisfies $tp[j]$=1, do

Step 2.1: k=k+1; set role set $\vec{R}_k=\phi$; $n[k]$ =0;

Step 2.2: If $\hat{r}^j\ne\phi$, then $\vec{R}_k=\hat{r}^j$, $n[k]$ =1, and continue step 2;

Step 2.3: For i=1 to m do

Step 2.3.1 If $drpa[i,j]$=1 and role r_i satisfies the constraints C, then add r_i to $\vec{R}_k$, and set $n[k]$ = $n[k]$+1;

Step 2.4: If $n[k]$ =0, then reporting "No solution" and goto step 7.

Step 3: Pick one role from each role set $\vec{R}_i$ (where $1\le i\le k$), and combine those roles to form a new role set $T\tilde{R}_j$ (where $1\le j\le\prod_{i=1}^{k}n[i]$).

Step 4: Set $\varphi_{\max}$ =0 and Z=0;

Step5: For j=1 to $\prod_{i=1}^{k}n[i]$, do

Step 5.1: If role set $T\tilde{R}_j$ does not satisfy the role constraints C, then goto step 5.

Step 5.2: Calculate $\varphi^{T\tilde{R}_j}$;

Step 5.3: If $\varphi^{T\tilde{R}_j}>\varphi_{\max}$, then $\varphi_{\max}=\varphi^{T\tilde{R}_j}$ and Z=j;

Step 6: If Z>0 then Output the role set $T\tilde{R}_Z$ (i.e., $T\tilde{R}_Z$ is the role set that can access all target privileges, satisfies the constraints C and conforms the principle of least privilege), otherwise reporting "No solution".

Step 7: Halt.

In the role hierarchical structure, high-level roles inherit the privileges from all their low-level roles, and all the roles that we are searching for should be in the lowest level, otherwise they may bring extra non-target privileges and thus can violate the principle of least privilege. That is, for any target privilege s_j, the target role r_i should satisfy $drpa[i, j]$=1. Step 2 builds a temporary role set $\vec{\vec{R}}_i$ from each target privilege (where $1 \le i \le k$, k is the number of the target privileges), and set the temporary role set as must-in role if the must-in role exists for the target privilege. Step 3 generates all possible role sets $T\tilde{R}_j$ by picking up one role from each $\vec{\vec{R}}_i$. Step 5 sorts the overall satisfaction degree $\varphi^{T\tilde{R}_j}$. The combined role set $T\tilde{R}_Z$ with the biggest value of $\varphi_{\max}$ is the role set that we are searching for. Because of the role constraints C, it is possible that there does not exist such a role set that can access all target privileges and satisfy the role constraints C.

In a large-scale RBAC system, it requires quite amount of computation in the step 3 and step 5 of the Algorithm 3, and it is difficult or impossible to enforce the principle of least privilege based on intuitive observations or conventional approaches for user-role assignments. The Algorithm 3 can thus be used for the automatic generation of user-role assignment that conforms the principle of least privilege.

5 CONCLUSION

The principle of least privilege is important to many RBAC applications. In this paper, we introduced and defined the concepts to quantitatively measure the enforcement of the principle of least privileges. Two algorithms to find the perfect and optimal user-role assignments that meet the principle of least privilege are presented. The proposed approach for the enforcement of least principle is particularly useful for automatic generation of user-role assignment in large-scale RBAC systems in which it is difficult to enforce the principle of least privilege based on intuitive observations or conventional approaches for user-role assignments.

REFERENCES

Ahn, G., and Sandhu, R., 2000. Role-based authorization constraints specification. ACM Transactions on Information and System Security, Vol. 3 No. 4, November 2000, pp 207-226.

Bertino, E., Bonatti, P. A., and Ferrari,E., 2001. TRBAC: A temporal role-based access control model. ACM Transactions on Information & System Security, Vol. 4, No. 3, Aug.2001, pp 191-233.

Bertino, E., Ferrari, E., and Atluri, V., 1999. The specification and enforcement of authorization constraints in workflow management systems. ACM Transactions on Information and System Security, Vol. 2, No. 1, 1999, pp 65-104.

Ferraiolo, D. F., Gilbert, D. M., and Lynch, N., 1993. An examination of federal and commercial access control policy needs. In Proceedings of NISTNCSC National Computer Security Conference, Baltimore, MD, September 1993, pp 107-116.

Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R., Chandramouli, R., 2001. Proposed NIST standard for role-based access Control. ACM Transactions on Information and System Security, Vol. 4, No. 3, August 2001, pp 224-274.

Giuri, L., 1997. Role-based access control: A natural approach. In Proceedings of the 1st ACM Workshop on Role-Based Access Control, ACM, 1997, Pages II, pp 33-37.

Howard, M., and LeBlanc, D., 2003. Writing secure code. Microsoft Press, 2003.

Osborn, S., Sandhu, R., and Munawer, Q., 2000. Configuring role-based access control to enforce mandatory and discretionary access control policies. ACM Transactions on Information and System Security, Vol. 3, No. 2, May 2000, pp 85-106.

Sandhu, R., Coyne, E. J., Feinstein, H. L., Youman, C. E., 1996. Role-based access control models. IEEE Computer, Vol. 29, No. 2, IEEE Press, February 1996, pp 38-47.

Saltzer, J. H., and Schroeder, M.D., 1975. The protection of information in computer systems. Proceedings of the IEEE, Vol. 63, No. 9, September 1975, pp 1278-1308.

Zhang, C. N. and Yang, C., 2003. Integrating object-oriented role-based access control model with mandatory access control principles. The Journal of Computer Information Systems, Vol. 43, No. 3, 2003, pp 40-49.