2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications

Role Mining Using Boolean Matrix Decomposition With Hierarchy

Wei Ye, Ruixuan Li†, Huaqing Li
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, P.R.China
E-mail: csio@hust.edu.cn, rxli@hust.edu.cn, lihuaqing@smail.hust.edu.cn

***Abstract*—With the increasing adoption of role-based access control (RBAC) in business security, how to apply role mining technology to aid the process of migrating a non-RBAC system to a RBAC system has become an important problem. Numerous approaches have been proposed to use data mining techniques to discover the roles. However, the Boolean matrix decomposition is still little used in role mining, because Boolean matrix decomposition without hierarchy can not express the hierarchical relationships of the RBAC model. In this paper, we propose a new method of Boolean matrix decomposition which can clearly express the hierarchical relationships of the RBAC model. Then, we introduce the cost-utility analysis method in economics to guide the role mining. Our optimization goal is not only to minimize the administration costs, but also to maximize the utility of RBAC configuration in the meanwhile. We further propose a heuristic algorithm to find the optimal solution with the Boolean matrix decomposition. The experimental results demonstrate the effectiveness of our approach.**

***Keywords*-rbac; role engineering; role mining; cost-utility;**

I. Introduction

At present, role-based access control (RBAC) [1][2] is the most popular access control model that has been widely used in enterprise security management products. In this security model, users no longer acquire permissions directly but through roles. This method greatly simplifies the security management. However, how to build the optimal RBAC state is a challenge work.

Role engineering is introduced as a solution to build a comprehensive framework for creating the architectural structure of RBAC [3][4]. Essentially, there have two basic approaches towards role engineering: the top-down and the bottom-up. Under the top-down approach [5][6], it often starts with the analysis of business processes, security policies and other business information. Hence, this approach can well reflect the functional requirements of the organization. However, this approach is time consuming and costly [7] because there may have a large number of complex business processes and vast amounts of users and privileges in an organization. In order to overcome this shortcoming, researchers have proposed bottom-up approach to discover roles from existing user-permission assignments. This bottom-up approach can generate the architectural structure of RBAC automatically. Such a bottom-up approach is called role mining.

The best benefit of an optimal RBAC configuration is that it can greatly reduce the administration costs. Therefore, in role mining research field, the minimality is a widely used criteria for evaluating an RBAC state. Vaidya et al.[8] have formally defined the role mining problem using the notion of minimality, and try to minimize the number of roles. Zhang et al. [9] and Ene et al.[10] attempt to minimize the number of user-role assignment and permission-role assignment relations. Guo et al.[11] aim to minimize the number of roles and the edges in role hierarchy graph. Molloy et al.[14] propose the weighted structural complexity (WSC) to sum up the number of relationships.

The weighted structural complexity (WSC) simply sums up the number of relationships without explicitly revealing the economic essence hidden in the relationships between parameters. Specifically, as far as role is concerned, each new role adds to the management cost, such as creating a role, assigning permissions to roles, assigning roles to users, etc. On the other hand, if this role is not a redundant role, the result of removing it is that some permissions need to be directly assigned to users, not through RBAC configuration. This reflects the utility of the role. The more the number of direct user-permission assignment, the lower the utility of role will be. Of course, the utility of role is not simply the number of direct user-permission assignment, but the weighted sum. Our optimization goal is not only to minimize the administration costs, but also to maximize the utility of RBAC configuration in the meanwhile. This method is called cost-utility analysis. Cost-utility analysis reveals the essence of economics of the weighted structural complexity (WSC).

In the existing role mining research, Given a binary matrix UPA representing the user-permission assignment, its traditional decomposition is UA and PA matrices, where UA represents the user-role assignment and PA represents the role-permission association. This is defined as the role mining problem (RMP) from the perspective of decompositions of a binary matrix. However, this matrix decomposition does not explicitly reflect hierarchical relationships of the RBAC model. In addition to applying cost-utility analysis to role mining, the contribution of this paper is to propose a new method of matrix decomposition. The user-permissions

† Corresponding author.

978-0-7695-5022-0/13 $26.00 © 2013 IEEE
DOI 10.1109/TrustCom.2013.98

matrix is decomposed into three matrices HA, UA' and PA', where HA represents the inheritance relationship between the roles, UA' represents the user-role assignment and PA' represents the role-permission association. In this way, hierarchical relationships in the RBAC model can be clearly expressed in matrix decomposition.

In this paper, we formally define the problem of role mining with cost-utility analysis and present a heuristic algorithm to find optimal RBAC state . In order to achieve our goal, we use the theory of formal concept analysis and matrix decomposition, which have been shown to provide a solid theoretical foundation for role engineering. Our algorithm is a two-phase solution. Firstly, we get the candidate roles through formal concept analysis, and there are many redundant roles in these candidate roles. Secondly, as described in boolean matrix with hierarchy, redundant roles can be removed according to cost-utility analysis. The evaluations indicate the effectiveness of this algorithm on removing the redundant roles which bring the highest administration costs and the lowest utility.

The remainder of this paper is organized as follows. We discuss related work in Section 2. Section 3 presents an overview of the Boolean matrix and the formal concept analysis. Section 4 formalizes the problem of role mining with cost-utility analysis. Section 5 presents a new role mining algorithm (MOF algorithm). In Section 6, we present the evaluation of our approach. Section 7 concludes the paper.

II. RELATED WORK

According to the output, the role mining algorithms can be divided into two classes. The output of the first class algorithm is a set of candidate roles with priority value. The higher a role's priority value is, the role can be chosen by administrator with more possibility. The representative algorithms of the first class are CompleteMiner (CM) and FastMiner (FM) [12]. CM finds the unique intersecting sets from generated roles, while FM only finds the intersections between pairs of initial roles. The CM and FM algorithms will generate candidate role set which has large amount of roles, and almost all of the real roles can be found in the candidate role set.

The second class algorithm outputs a complete RBAC state with hierarchical structure. There are many algorithms belonging to this class, such as ORCA [13], HierarchicalMiner (HM) [14], and GO [9]. HM restructures the lattices according to the cost decrease of the RBAC with a greedy strategy, and proposes weighted structural complexity (WSC) as a common quality measurement to RBAC state. GO reduces the number of role-user and permission-role assignments by a graph optimization method.

However, the traditional role mining approach only considers the minimality of the cost. No attention is paid to the tradeoff between cost and utility of the RBAC configuration. The traditional approach only tell you which redundant roles can be removed while we want to know removing which one is the best. Another aspect is that the traditional user permission assignment matrix decomposition can not clearly express the hierarchical relationships of the RBAC model. Hence, we can improve the decomposition method and discover roles based on boolean matrix decomposition with inheritance relationship.

In this paper, we present cost-utility ratio of the role to represent the trade-off between management cost and utility of the role. we always removing the redundant role with the maximal cost-utility ratio, so we can get the final RBAC state with the minimal cost and maximal utility. Finally, we propose Boolean matrix decomposition with inheritance relationships for the RBAC model. The experimental results are tested to show the effectiveness of our findings.

III. PRELIMINARIES

In this section, we will review the theories of boolean matrix decomposition and formal concept analysis which are the foundation of our work.

A. Boolean matrix

Definition 1 (Boolean Matrix Multiplication): A Boolean matrix multiplication between an $n \times k$ Boolean matrix A with $\mathrm{a_{il}} \in \{0, 1\}$ and a $k \times m$ matrix B with $\mathrm{b}_{lj} \in \{0, 1\}$ is $A \otimes B = C$, where C is matrix with $c_{ij} \in \{0, 1\}$ and

$$c_{ij} = \bigvee_{l=1}^{k} (\mathrm{a_{il}} \wedge \mathrm{b_{lj}}).$$

Definition 2 (Boolean Matrix Decomposition): If $C = A \otimes B$, where C, A and B are boolean matrices, $A \otimes B$ is called a decomposition of C.

Definition 3 (User Permission Assignment Matrix Decomposition): Consider an $n \times m$ binary matrix UPA, which presents the user-permission assignment. Assume Binary matrices UA and PA with dimensions $n \times k$ and $k \times m$ respectively, where UA presents a user-role assignment and PA presents a role-permission assignment. $UA \otimes PA$ is called a decomposition of UPA if

$$(UPA)_{ij} = \bigvee_{l=1}^{k} ((UA)_{il} \wedge (PA)_{lj}).$$

B. Formal concept analysis

The most common role mining approach is clustering. Nevertheless, most clustering techniques seek to find mutually-disjoint groups. That is, each user or permission can belong to only one cluster. However, in practice, we may expect a permission to be a member of two or more distinct roles.

In contrast with other methods, formal concept analysis aims at finding "concepts" in input matrix specifying a set of users and permissions. This is exactly the same problem as finding meaningful roles. In formal concept analysis, the concepts are arranged in a lattice. The relative relationships

among concepts provide semantic information in addition to the users and permissions that are associated with them. The input of formal concept analysis is called a formal context. We present the definition as follows.

Definition 4. A formal context is a triple $\mathcal{B}(G, M, I)$ where G and M are sets and $I \subseteq G \times M$ is a binary relation between G and M. The elements of G and M are called objects and attributes respectively. For $g \in G$ and $m \in M$, we write gIm when $(g, m) \in I$.

In the role mining, we view the user-permission relation as a formal context, where G is the set of all users, and M is the set of all permissions, and $(g, m) \in I$ if and only if the user corresponding to g has the permission corresponding to m.

Definition 5. A concept of the context $\mathcal{B}(G, M, I)$ is a pair(X, Y), where $X \in G$ and $Y \in M$ satisfy the following properties:

$Y = \{m \in M \mid (\forall g \in X)\ gIm\}$, i.e., Y is the set of all attributes shared by all objects in X.

$X = \{g \in G \mid (\forall m \in Y)\ gIm\}$, i.e., X is the set of all objects that share all attributes in Y.

X is also called the extent and Y is called the intent of the concept (X, Y). The set of all concepts of the context is denoted by $\mathcal{B}(G, M, I)$. A concept (X_1, Y_1) is a subconcept of (X_2, Y_2), denoted as $(X_1, Y_1) \leq (X_2, Y_2)$ if and only if $X_1 \subseteq X_2$ (or, equivalently, $Y_1 \supseteq Y_2$).

The family of these concepts obeys the mathematical axioms defining a lattice, and it is called a concept lattice or Galois lattice. The concept lattice for the running example Figure 1(a) and Figure 1(b) is given in Figure 1(c). For instance, here $(\{U_3, U_4\}, \{P_0, P_1, P_{10}, P_{11}\})$ is not a concept, because U_2, U_5 also have the permissions $\{P_0, P_1, P_{10}, P_{11}\}$. The pair $(\{U_2, U_3, U_4, U_5\}, \{P_0, P_1, P_{10}, P_{11}\})$ is a concept. In this concept lattice, each concept inherits all permissions associated with its subconcepts, and users are inherited in the inverse direction. Therefore, we can remove redundant permissions and users from each node. The result is called the reduced concept lattice and is shown in Figure 1(d). The reduced concept lattice defines a complete RBAC state.

Each concept represents a role and the lattice can be viewed as the role hierarchy. In this RBAC state, each user is assigned to exactly one role, and each permission is assigned to exactly one role and the subconcept relation corresponds to the role inheritance relation. It is clear that the reduced concept lattice provides the semantic relationships among concepts and has more meanings than just a set of permissions. Using the reduced concept lattice as the role hierarchy has the disadvantage that the role hierarchy may be extremely large. In the reduced concept lattice, some concepts don't introduce new users, new permissions, or neither. However, we can not remove all those concepts without new users or new permissions.

C. Selecting formal concepts as optimal role set

Consider decomposing a $n \times m$ binary matrix UPA into a product $UA \otimes PA$ of binary matrices UA and PA constructed from a set F of formal concepts associated to UPA. In particular, consider the concept lattice $\mathcal{B}(X, Y, I)$ associated to UPA, with $X = \{1, ..., n\}$ and $Y = \{1, ..., m\}$. Let

$$F = \{(UA_1, PA_1), ..., (UA_k, PA_k)\} \subseteq \mathcal{B}(X, Y, I)$$

UA is an $n \times k$ binary matrix with entries $UA_{il} (1 \leq i \leq n, 1 \leq l \leq k)$ defined as

$$(UA)_{il} = \begin{cases} 1 & \text{if } U_i \in UA_l \\ 0 & \text{if } U_i \notin UA_l \end{cases}$$

PA is a $k \times m$ binary matrix with entries $PA_{lj} (1 \leq l \leq k, 1 \leq j \leq m)$ defined as

$$(PA)_{lj} = \begin{cases} 1 & \text{if } P_j \in PA_l \\ 0 & \text{if } P_j \notin PA_l \end{cases}$$

That is, the columns of UA correspond to the extents UA_l of the formal concepts $\langle\{UA_l, PA_l\}\rangle$ and the rows of PA correspond to the intents PA_l of the formal concepts$\langle\{UA_l, PA_l\}\rangle$. F is then suitable to a set of factors if $UPA = UA \otimes PA$ (i.e. UPA is equal to the Boolean matrix product of UA and PA), or $UPA \approx UA \otimes PA$ (i.e. UPA is approximately equal to $UA \otimes PA$, with "$\approx$" appropriately defined).

Theorem 1: Let $F = \{(UA_1, PA_1), ..., (UA_k, PA_k)\} \subseteq \mathcal{B}(G, M, I)$ and $UPA = UA \otimes PA$, then the relation UPA corresponding to $UA \otimes PA$ is constructed as follows. UPA is a union of rectangular relations $UA_l \otimes PA_l$ corresponding to rectangles $\langle\{UA_l, PA_l\}\rangle$

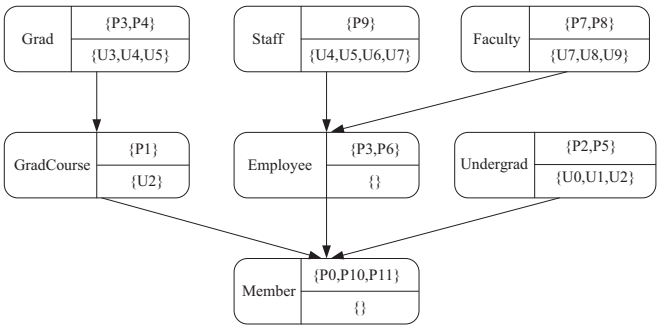
$$UPA = UA_1 \otimes PA_1 \cup \cdots \cup UA_k \otimes PA_k$$

Proof: We can regard (UA_l, PA_l) as a concept, the matrix $UA_l \otimes PA_l$ means that if the U_i gets the P_j through the concept (UA_l, PA_l), the $(UPA)_{ij} = 1$. Therefore, $(UPA)_{ij} = 1$ if and only if the U_i gets the P_j through at least one of the concept.

Example 1: Let n = 4, m = 5, k = 4, $UPA = \{(UA_1, PA_1), (UA_2, PA_2), (UA_3, PA_3), (UA_4, PA_4)\}$. $UA_1 = \{U_1, U_2, U_3\}, PA_1 = \{P_1, P_2\}$, $UA_2 = \{U_3\}$, $PA_2 = \{P_3, P_4\}$, $UA_3 = \{U_2, U_4\}, PA_3 = \{P_1, P_5\}$, $UA_4 = \{U_1\}, PA_4 = \{P_2\}$.

$$UA = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} PA = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

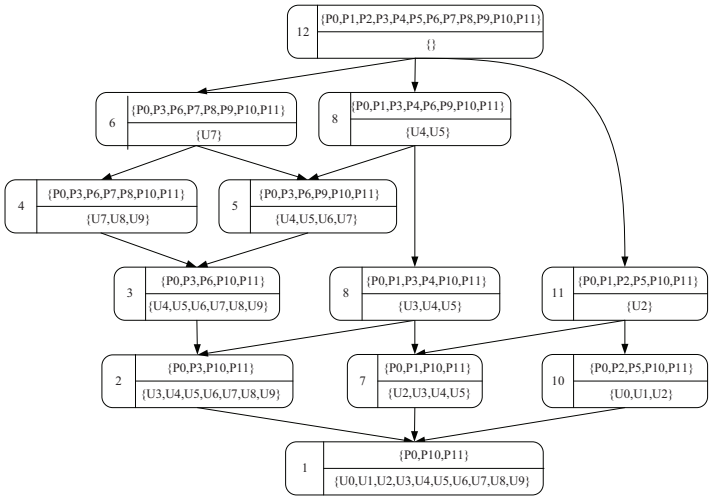
$$UPA = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$



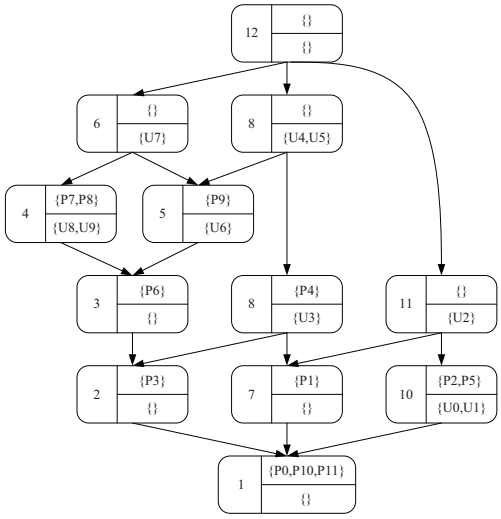
(a) Original role hierarchy

User	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
U0	1	0	1	0	0	1	0	0	0	0	1	1
U1	1	0	1	0	0	1	0	0	0	0	1	1
U2	1	1	1	0	0	1	0	0	0	0	1	1
U3	1	1	0	1	1	0	0	0	0	0	1	1
U4	1	1	0	1	1	0	1	0	0	1	1	1
U5	1	1	0	1	1	0	1	0	0	1	1	1
U6	1	0	0	1	0	0	1	0	0	1	1	1
U7	1	0	0	1	0	0	1	1	1	1	1	1
U8	1	0	0	1	0	0	1	1	1	0	1	1
U9	1	0	0	1	0	0	1	1	1	0	1	1

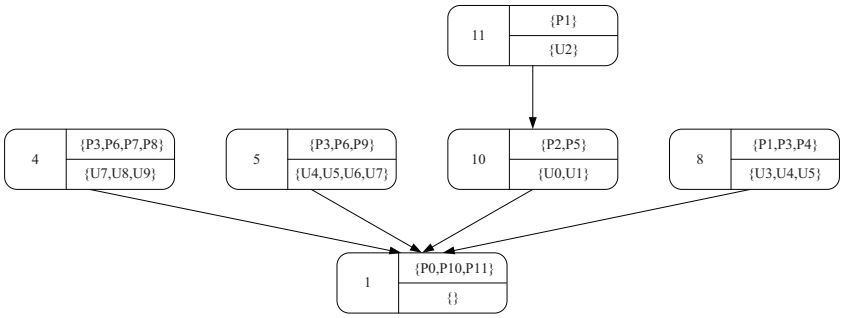
(b) User-permission relation



(c) The concept lattice



(d) Reduced lattice



(e) Pruned role hierarchy

Figure 1: Running example

For the relation $UPA = UA \otimes PA$, we have $UPA = \bigcup_{l=1}^{4} UA_l \otimes PA_l$. In a matrix setting, this means that it is built as a union of binary matrices $UA_1 \otimes PA_1, UA_2 \otimes PA_2, UA_3 \otimes PA_3$, and $UA_4 \otimes PA_4$. We have

$$UPA = UA_1 \otimes PA_1 \cup UA_2 \otimes PA_2 \cup UA_3 \otimes PA_3 \cup UA_4 \otimes PA_4$$

$$= \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \cup \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\cup \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} \cup \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

IV. Role Mining with Cost-Utility Analysis

A. Minimal information loss

Definition 6 (Minimal Information Loss): Given an access control configuration $\rho = \langle U, P, UP \rangle$, where U is a set of all users, P is a set of all permissions and $UP \subseteq U \times P$ is the user-permission relation, we want to find an RBAC state $\langle R, UA, PA, RH, DUPA \rangle$ that is consistent with ρ.

$DUPA \subseteq U \times P$ is the direct user-permission assignment relation. If $\langle U_l, P_i \rangle \in DUPA$, and the weight of the permission P_i is W_{p_i}, minimizing

$$Loss = \sum_{\langle U_l, P_i \rangle \in DUPA} W_{p_i}.$$

In the following paper, we consider all the permission's weight are equal(W_{p_i} = 1).

B. Cost-utility analysis

Essentially, the problem of role mining with cost-utility analysis can be viewed as a multi-objective optimization problem that aims at trade-off conflicting objectives. One classical way is to compute a weighted sum of all the objective functions. We define the following multi-objective optimization function.

Definition 7 (Multi-objective Optimization Function): Given $W = (w_r, w_u, w_p, w_h, w_l)$ where $w_r, w_u, w_p, w_h, w_l \in [0, +\infty]$, the multi-objective optimization function of an RBAC state, which is denoted as $WSC(\gamma, W)$, is computed as

$$WSC(\gamma, W) = w_r * |R| + w_u * |UA| + w_p * |PA| + w_h * |tr(RH)| + w_d * |DUPA|$$

where $|\cdot|$ denotes the size of the set or relation. The goal is to minimize the WSC.

A transitive reduction is the minimal set of relationships that describes the same hierarchy. For example, $tr(\{(r1, r2), (r2, r3), (r1, r3)\}) = \{(r1, r2), (r2, r3)\}$, as $(r1, r3)$ can be inferred.

It is possible to adjust the weights of WSC to limit the RBAC states to meet different objectives. For example, by setting w_h to ∞, we can force a flat RBAC state since each role inheritance relation costs ∞.

This is a particular case, by setting $w_d = 0$, we can not control the number of the direct user-permission assignment. That means, if we set $w_d = 0$, the final RBAC state will have direct user-permission assignments. By setting $w_d = \infty$, we do not allow direct user-permission assignments.

The WSC was often used as a measurement for the goodness of an RBAC state. However, we must ensure that all parameters and settings are the same during comparisons.

There is a contradictory statement in the StateMiner[15], StateMiner algorithm do not allow direct user permission assignments while setting $w_d = 0$. This is internally contradictory because we can not control the number of the direct user permission assignment by setting $w_d = 0$. As a result, the StateMiner may have smaller WSC value while other algorithms set $w_d = 1$ or $w_d = \infty$, because it do not control the number of the direct user permission assignment.

Definition 9 (Cost-utility Ratio of Role):
Given an optimal role set, the cost-utility ratio of the role is defined as

$$CUR = \frac{w_r * |R| + w_u * |UA| + w_p * |PA| + w_h * |tr(RH)|}{w_l * |Loss|}.$$

If we set $w_r = w_u = w_p = w_h = w_l = 1$, we can simplify the formula as

$$CUR = \frac{|R| + |UA| + |PA| + |tr(RH)|}{|Loss|}.$$

Each additional role will increase the management cost, such as creating a role, assigning permissions to roles, assigning roles to user, etc. We use the cost-utility ratio of the roles (CUR) to represent the trade-off between management cost and utility of the roles. It can be understood as the management cost for each unit of the role utility. In fact, we do not directly calculate the value of the cost-utility ratio of the role. We calculate the difference between cost and utility for the role. This is because whether sorting by the ratio or sorting by the difference, the result of the sort is the same.

Definition 10 (Multi-objective Optimization Function): Given $W = (w_r, w_u, w_p, w_h, w_l)$ where $w_r, w_u, w_p, w_h, w_l \in [0, +\infty]$, the multi-objective optimization function of an RBAC state, which is denoted as $MOF(\gamma, W)$, is computed as

$$MOF(\gamma, W) = w_r * |R| + w_u * |UA| + w_p * |PA| + w_h * |tr(RH)| - w_l * |Loss|$$

where $|\cdot|$ denotes the size of the set or relation. The goal is to minimize the MOF.

C. Boolean matrix decomposition with hierarchy

With the traditional Boolean matrix decomposition method, we decompose $UPA = UA \otimes PA$. However, this

traditional decomposition method can not explicitly reflect the inheritance relationship of the RBAC model. So we present a new matrix decomposition approach .

Definition 11 (Role-role Hierarchy Matrix): Let $K = |R|$, and $RH \subseteq R \times R$ is a role-role inheritance relationship. We assume that the sets of users, permissions and roles can all be ordered, e.g., we will speak of the i^{th} role, for $i \in \{1, ..., N\}$. For notational convenience, we encode RH as a binary matrix. We represent RH as $x \in \{0,1\}^{K \times K}$. In this representation, $x_{ij} = 1 (x_{ij} = 0)$ indicates that role r_i is (not) the immediate junior to role r_j. We can also set $x_{ii} = 1$ for the RH matrix, and call this matrix RH'.

The traditional UA and PA matrix cannot reflect the inheritance relationship between the roles. In the concept lattice, each concept inherits all permissions associated with its subconcepts, and users are inherited in the inverse direction. Therefore, we can remove redundant permissions and users from each node. The result is called the reduced concept lattice. The i^{th} column of the UA matrix and the i^{th} row of the PA matrix correspond to the i^{th} concept. Hence, we can remove redundant permissions and users from UA and PA . The result is called UA and PA hierarchy matrix.

Definition 12 (Permission-role Matrix Decomposition): Given a role-permission assignment PA, a $k \times k$ role-role hierarchy matrix RH' with $a_{il} \in \{0, 1\}$, and a $k \times m$ matrix PA' with $b_{lj} \in \{0, 1\}$, in role-role hierarchy matrix RH', the set of all junior role of R_i is called Ω_i. A Boolean matrix multiplication between RH' and PA' is $PA = RH' \odot PA'$ where PA is a matrix with $c_{ij} \in \{0, 1\}$ and

$$c_{ij} = \bigcup_{R_x \in \Omega_i} \left(\bigcup_{l=1}^{k} (a_{il} \cap b_{lj}) \right).$$

Definition 13 (User-role Matrix Decomposition): Given a user-role assignment UA, a $k \times k$ role-role hierarchy matrix RH' with $a_{il} \in \{0, 1\}$, and a $k \times m$ matrix $(UA')^T$ with $b_{lj} \in \{0, 1\}$, in role-role hierarchy matrix RH', the set of all junior role of R_i is called Ω_i. A boolean matrix multiplication between RH' and $(UA')^T$ is $(UA)^T = RH' \odot (UA')^T$ where $(UA)^T$ is a matrix with $d_{ij} \in \{0, 1\}$ and

$$d_{ij} = \bigcup_{R_x \in \Omega_i} \left(\bigcup_{l=1}^{k} (a_{il} \cap b_{lj}) \right).$$

Definition 14 (UPA Matrix Decomposition): Given a user-permission assignment UPA, a user-role assignment UA, a role-permission assignment PA, a role-role hierarchy matrix RH', a permission-role hierarchy matrix PA', and a user-Role hierarchy matrix UA', the matrix decomposition of UPA can be defined as

$$UPA = \left(RH' \odot (UA')^T \right)^T \otimes (RH' \odot (PA')).$$

Example 2: For instance, we have

$$UPA = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix},$$

$$UA = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \; PA = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$

Here, $UPA = UA \bigotimes PA$. We can continue to decompose UA and PA matrices. We decompose the UA matrix into RH' and UA' matrices. The PA matrix is decomposed into RH' and PA' matrices. The decomposed RH, UA', PA' and RH' are as follows.

$$RH = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, UA' = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$PA' = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, RH' = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

V. Multi-objective Optimization Algorithm

In this section, we present a greedy algorithm MOF to find a set of roles satisfying the goals of role mining with cost-utility analysis. In the first phase, we generate the reduced concept lattice using the deployed configuration, which produces an RBAC state. In the second phase, we prune this reduced concept lattice and select the final RBAC state. In order to do this, we use a greedy algorithm to heuristically optimize the lattice.

Once we get the reduced concept lattice, we should decide which roles are appropriate and which ones should be removed through using cost-utility analysis. Removing each role reduces the cost of creating the role and the associated relationships. However, we need to add back some relationships so that user-permission assignment relation and the inheritance relation remain correct. We use a general pruning rule in each iteration: removing role r from the reduced concept lattice when the decrease of the value of the MOF is maximum after removing that role. MOF algorithm as shown in Algorithm 1 is a greedy algorithm. It iterates over all of the roles in each iteration. It stops when no more operations can be performed. In Algorithm 1, the "Δ" symbol means the change of parameters before and after the redundant role was removed.

VI. Experimental Results

A. Computational complexity

As the MOF algorithm has two phases, its computational complexity depends on the complexity of the generation of the reduced concept lattice and the complexity of the pruning process. The time complexity for generating the reduced concept lattice is linear in the size of the concept lattice. Because every concept in the reduced concept lattice is equivalent to a role, the size of the lattice equals the number of roles. The number of roles in the worst case is the number of permissions when every role only has one permission. We use n as the symbol for the number of permissions. As the algorithm iterates n times during pruning process, in the worst case the computational complexity is $O(n^2)$ where n is the number of permissions in the UPA.

B. Experimental evaluation

To study the performance of the algorithm, we implement MOF by Java and run it on five datasets, including university, healthcare, Domino, Firewall 1 and Firewall 2. The university datasets is used to evaluate HierarchicalMiner by Molloy et al[14]. The other datasets obtained from researchers at HP Labs and used for evaluation in [10], which were from the US Veteran's Administration. We use nine main role mining algorithms to compare with MOF algorithm.

Table 1 shows the weighted structural complexity (WSC) of the original RBAC state and the states generated by MOF algorithm. We use two weight schemes, $W_1 : w_r = w_u = w_p = w_h = 1$ and $W_2 : w_r = w_u = 1, w_p = w_h = 2$. The scheme W_1 assumes that the cost of adding each element (a role or a relationship) to the RBAC state is 1. The WSC thus measures the cost of creating the RBAC state. The scheme W_2 assumes that actions related to permissions are more expensive. We can see that MOF algorithm with the scheme W_2 has smaller WSC than the HierarchicalMiner and is closer to the optimal solution.

In Table 2, we present the results from evaluating the nine algorithms with five datasets. We use the scheme $W_3 : w_r = w_u = w_p = w_h = w_l = 1$. We can see that MOF algorithm with the scheme W_3 has smaller WSC than other algorithms on average. The main reason is that the MOF algorithm using cost-utility analysis can calculate pruning which brings more benefits.

VII. Conclusion and Future Work

While there are many role mining approaches have been proposed recently, none of them pay attention to the tradeoff between cost and utility of the RBAC configuration. It may fail to find the optimal RBAC state. In the meanwhile, none of them considered decomposing the user-permission assignment matrix with inheritance relationships between roles. In this paper, we formally define the problem of role mining with cost-utility analysis and present a heuristic

Algorithm 1 MOF algorithm for role mining

Input:
Input: user-permission assignment UPA
Input: weight factors, W = $(w_r, w_u, w_p, w_h, w_l)$
Output:
UA, PA, RH, UA' , PA' matrix ← Zero matrix
$MOF_{max} = 0$
Create concept lattice. The set of all concepts of the context is denoted by $B(G, M, I)$. $B(G, M, I) = \{(A_1, B_1), (A_2, B_2), ..., (A_k, B_k)\}$;
UA matrix ← $\{(A_1, A_2, ..., A_k,)\} matrix$
PA matrix ← $\{(B_1, B_2, ..., B_k,)\} matrix$
Create reduced concept lattice $\gamma = \langle R, UA', PA', RH\rangle$. The set of all concepts of the context is denoted by $B'(G, M, I)$. $B'(G, M, I) = \{(A'_1, B'_1), (A'_2, B'_2), ..., (A'_k, B'_k)\}$;
RH matrix ← Compute hierarchy relationship
UA' matrix ← $\{(A'_1, A'_2, ..., A'_k,)\}$ $matrix$
PA' matrix ← $\{(B'_1, B'_2, ..., B'_k,)\}$ $matrix$
while $MOF_{max} >\doteq 0$ **do**
  for each(A_i, B_i) **do**
    $B_i(G, M, I)$=$B(G, M, I)$- $\{(A_i, B_i)\}$;
    UA_{temp} matrix ← UA matrix - $\{A_i\}$
    PA_{temp} matrix ← PA matrix - $\{B_i\}$
    UA'_{temp} matrix ← remove the i^{th} column of the UA' matrix and add back corresponding edges so that the inheritance relation remains correct.
    PA'_{temp} matrix ← remove the i^{th} row of the PA' matrix and add back corresponding edges .
    RH_{temp} matrix ← remove the i^{th} row and the i^{th} column of the RH, add back corresponding edges.
    $\Delta|R| \leftarrow |R|_{before} - |R|_{after}$
    $\Delta|UA| \leftarrow \parallel UA'_{before} - UA'_{temp} \parallel_1$
    $\Delta|PA| \leftarrow \parallel PA'_{before} - PA'_{temp} \parallel_1$
    $\Delta|RH| \leftarrow \parallel RH'_{before} - RH'_{temp} \parallel_1$
    $\Delta|Loss| \leftarrow Loss_{before} - Loss_{after}$
    $MOF_i = w_r * \Delta|R| + w_u * \Delta|\text{UA}| + w_h * \Delta|\text{RH}| + w_p * \Delta|\text{PA}| - w_l * \Delta Loss$;
    $FindMax[k] \leftarrow MOF_i$
  end for
  $MOF_{max} = max(FindMax[k])$
  if $MOF_{max} >= 0$ **then**
    $B(G, M, I) = B(G, M, I) - \{(A_{max}, B_{max})\}$
    $UA_{new}matrix \leftarrow UAmatrix - \{A_{max}\}$
    $PA_{new}matrix \leftarrow PAmatrix - \{B_{max}\}$
    UA'_{new} matrix ← remove the max^{th} column of the UA' matrix and add back corresponding edges.
    PA'_{new} matrix ← remove the max^{th} row of the PA' matrix and add back corresponding edges.
    RH_{new} matrix ← remove the max^{th} row and the max^{th} column of the RH , add back edges.
  end if
end while
return UA', PA', RH

Table I: The MOF algorithm results

	W={1,1,1,1}					W={1,1,2,2}				
	R	UA	PA	RH	WSC	R	UA	PA	RH	WSC
Original	32	799	35	19	885	32	799	35	19	885
HierarchicalMiner	21	498	67	19	605	21	505	65	20	696
MOF	21	498	67	19	605	21	498	67	19	691

Table II: Minimal WSC for $W =< 1, 1, 1, 1, 1 >$

	MOF	HM	GO	CM	PC	DM	HPr	HPe	ORAC
University	605	605	615	620	619	683	894	564	1773
Healthcare	138	146	136	164	148	325	298	202	223
Domino	417	423	476	495	501	553	723	693	659
Firewall 1	1072	1456	2254	2678	2258	4689	2932	1810	22101
Firewall 2	945	959	981	995	992	998	1562	1299	30789
Average	635	718	892	990	903	1450	1282	914	11109

algorithm to find optimal RBAC state. In order to achieve our goal, we use the theory of formal concept analysis, and propose an innovative method to decompose the user-permission assignment matrix into UA matrix, PA matrix and inheritance relationships matrix. Our algorithm is a two-phase solution. Firstly, we get the candidate roles through formal concept analysis. Secondly, as described in boolean matrix, redundant roles can be removed according to optimization function. We carry out the experiments and the results show the proposed approach has superior performance to traditional algorithms in finding the optimal RBAC state. For the future work, we will try to find the optimal RBAC state with available user-attribute information to make the role more meaningful, such as the different types of operations. They could be used to further refine the meaningful roles.

VIII. Acknowledgements

This work is supported by National Natural Science Foundation of China under grants 61173170 and 60873225, National High Technology Research and Development Program of China under grant 2007AA01Z403, and Innovation Fund of Huazhong University of Science and Technology under grants 2013QN120, 2012TS052 and 2012TS053.

References

[1] D.F. Ferraiolo, R. Sandhu, S. Gavrila, D.R. Kuhn, and R. Chandramouli. Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security*, 4(3):224–274, 2001.

[2] R.S. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.

[3] E.J. Coyne. Role engineering. In *Proceedings of the first ACM Workshop on Role-based access control*, page 4, 1996.

[4] M. Frank, J.M. Buhmann, and D. Basin. On the definition of role mining. In *Proceeding of the 15th ACM symposium on Access control models and technologies*, pages 35–44, 2010.

[5] P. Epstein and R. Sandhu. Engineering of role/permission assignments. In *Annual Computer Security Applications Conference (ACSAC)*, pages 127–136, 2001.

[6] A. Kern, M. Kuhlmann, A. Schaad, and J. Moffett. Observations on the role life-cycle in the context of enterprise security management. In *Proceedings of the seventh ACM symposium on Access control models and technologies*, pages 43–51, 2002.

[7] I. Molloy, N. Li, T. Li, Z. Mao, Q. Wang, and J. Lobo. Evaluating role mining algorithms. In *Proceedings of the 14th ACM symposium on Access control models and technologies*, pages 95–104, 2009.

[8] J. Vaidya, V. Atluri, and Q. Guo. The role mining problem: finding a minimal descriptive set of roles. In *Proceedings of the 12th ACM symposium on Access control models and technologies*, pages 175–184, 2007.

[9] D. Zhang, K. Ramamohanarao, and T. Ebringer. Role engineering using graph optimisation. In *Proceedings of the 12th ACM symposium on Access control models and technologies*, pages 139–144, 2007.

[10] A. Ene, W. Horne, N. Milosavljevic, P. Rao, R. Schreiber, and R.E. Tarjan. Fast exact and heuristic methods for role minimization problems. In *Proceedings of the 13th ACM symposium on Access control models and technologies*, pages 1–10, 2008.

[11] Q. Guo, J. Vaidya, and V. Atluri. The role hierarchy mining problem: Discovery of optimal role hierarchies. In *Annual Computer Security Applications Conference (ACSAC)*, pages 237–246, 2008.

[12] J. Vaidya, V. Atluri, and J. Warner. RoleMiner: mining roles using subset enumeration. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 144–153, 2006.

[13] U Steffens J Schlegelmilch. Role mining with orca. In *Proceedings of the tenth ACM symposium on Access control models and technologies*, pages 168–176, 2005.

[14] I. Molloy, H. Chen, T. Li, Q. Wang, N. Li, E. Bertino, S. Calo, and J. Lobo. Mining roles with semantic meanings. In *Proceedings of the 13th ACM symposium on Access control models and technologies*, pages 21–30, 2008.

[15] H. Takabi and J. B. D. Joshi. Stateminer: An efficient similarity-based approach for optimal mining of role hierarchy. In *In Proceedings of the 16th ACM Symposium on Access Control Models and Technologies*, pages 55–64, 2010.