Mining Stable Roles in RBAC

Alessandro Colantonio[1], Roberto Di Pietro[2], Alberto Ocello[3],
and Nino Vincenzo Verde[4]

[1] Engiweb Security, Roma, Italy
alessandro.colantonio@eng.it
and
Università di Roma Tre, Roma, Italy
colanton@mat.uniroma3.it
[2] Università di Roma Tre, Roma, Italy and
UNESCO Chair in Data Privacy, Tarragona, Spain
dipietro@{mat.uniroma3.it,urv.cat}
[3] Engiweb Security, Roma, Italy
alberto.ocello@eng.it
[4] Università di Roma Tre, Roma, Italy
nverde@mat.uniroma3.it

Abstract. In this paper we address the problem of generating a candidate role-set for an RBAC configuration that enjoys the following two key features: it minimizes the administration cost; and, it is a stable candidate role-set. To achieve these goals, we implement a three steps methodology: first, we associate a weight to roles; second, we identify and remove the user-permission assignments that cannot belong to a role that have a weight exceeding a given threshold; third, we restrict the problem of finding a candidate role-set for the given system configuration using only the user-permission assignments that have not been removed in the second step—that is, user-permission assignments that belong to roles with a weight exceeding the given threshold. We formally show—proof of our results are rooted in graph theory—that this methodology achieves the intended goals. Finally, we discuss practical applications of our approach to the role mining problem.

1 Introduction

Role-based access control (RBAC) [1] is a well known and recognized flexible security model for enterprise access control management. Central to the model is the *role* concept; a role is just a collection of access permissions, and users are assigned to roles based on duties to fulfil [9]. The main reason for the adoption of RBAC within many medium to large-size organizations is its simplicity. In particular, the RBAC model offers several benefits in terms of simplified access control administration, improved organizational productivity, and security policy enforcement. However, the overhead associated to role definition is often the main obstacle toward its adoption. Indeed, the first step in setting up an RBAC scheme is the role definition. To this aim, the *role engineering* discipline has been introduced. It refers to the set of methodologies and tools to define roles and to assign permissions to roles according to the actual needs of the company [5]. Existing role engineering approaches are usually classified in two categories: the *top-down* and the *bottom-up* approaches. The former carefully decomposes

D. Gritzalis and J. Lopez (Eds.): SEC 2009, IFIP AICT 297, pp. 259–269, 2009.

the business processes into elementary components, identifying which system features are necessary to carry out specific tasks. The latter, based on the analysis of existing access controls permissions, elicits a set of roles that correctly describe the existing user-permission assignments. Since this approach can be easily automated, it has attracted many researchers. In the literature, bottom-up approaches are usually referred to as *role mining*.

A recently addressed problem is the analysis of the effort incurred by administrators when managing the set of roles elicited by role mining algorithms. To this aim, [2,3] introduces the *administration cost function*. An *optimal candidate role-set* is a set of roles that correctly describes the existing permissions in such a way its administration cost is minimized. When new users, new permissions, or new user-permission assignments are added, there is the need to re-compute the optimal candidate role-set, that could lead to change the role-set in use. In particular, roles could be *unstable*, in the sense that the introduction of few users or few permissions could drastically change the optimal candidate role-set. Unstable roles could be difficult to be managed as they frequently change. Conversely, a role is *stable* if it is not greatly affected by the introduction of new users, new permissions or new user-permission assignments. That is why, when dealing with automated role mining algorithms, the stability of generated roles is a fundamental property that is worth investigating.

Contributions. The main contribution of this paper is to address the problem of finding a core of roles that is both stable and minimizes the cost function of the corresponding RBAC configuration. We model this problem with graphs, introducing a three-steps methodology that is able to prune user-permission assignments that lead to unstable roles. This way, we are able to build a core of roles which have the required characteristics. These results have been formally proven. In addition, relevant practical applications of the proposed methodology are shown.

Roadmap. The remainder of this paper is organized as follows: Section 2 offers an overview of previous approaches to the role mining problem. Section 3 sums up the concepts and the definitions used in this paper. In Section 4 the proposed model is discussed. Section 5 illustrates some of the possible applications of our approach. Finally, in Section 6 conclusions and some possible extensions of the work are presented.

2 Related Work

The term “role mining” was first introduced by Kuhlmann et al. [10] who applied existing data mining techniques to implement a bottom-up approach. They presented a clustering technique similar to the well known *k*-means clustering, which required a prior definition of the number of roles. In [13] it is described the first algorithm explicitly designed for role engineering, that is based on hierarchical clustering. Another approach to the role mining is explained by Vaidya et al. [17], that is based on the analysis of all possible intersections among permissions possessed by the users. Only recently researchers have started to formalize the role-set optimality concept, defining “interestingness” metric for roles [18, 11, 12]. Indeed, the importance of role completeness and role management efficiency resulting from the role engineering process has been evident from the earliest papers on the subject. However, the problem of identifying the role interestingness is only partially addressed.

Colantonio et al. [2,3], introduced the administration cost concept, proposing a metric to evaluate "good" collection of roles, namely role-sets which are easily administrable. This approach makes it possible to easily mine administrable roles. Vaidya et al. [15, 16] also studied the problem of reducing the administration effort but, in this case, the cost is simply represented by the number of roles which cover all permissions possessed by the users. They defined this problem as the Basic Role Mining Problem (*basicRMP*). The authors proposed a branch and bound method, and then a greedy heuristic, to build a set of roles by including, at each step, the role that covers the largest possible set of previously uncovered user-permission assignments. They also demonstrated that *basicRMP* is NP-complete. As shown in [6], not only is the *basicRMP* problem equivalent to the minimum clique covering, but it can be reduced to many other NP-complete problems, like binary matrices factorization [14,11] and tiling database [7] to cite a few.

Our approach is slightly different from the other ones. Though we preserve the existence of a general cost function, that is useful to identify the best possible roles, we also introduce a weight metric aimed at excluding the presence of unstable roles.

3 Background

In this section we introduce the fundamental concepts of the graph theory and some formal definitions of the RBAC standard that will be used later on.

Graphs Theory. A *graph* G is an ordered pair $G = \langle V, E \rangle$, where V is the set of vertices, and E is a set of unordered pairs of vertices. The endpoints of an edge $\langle v, w \rangle \in E(G)$ are the two vertices $v, w \in V(G)$. Given a subset S of $V(G)$, the subgraph *induced* by S is the graph whose vertex set is S, and whose edges are the members of $E(G)$ whose two endpoints are both in S. We denote with $G[S]$ the subgraph induced by S. A *bipartite graph* is a graph where the set of vertex can be partitioned into two subsets V_1 and V_2, such that for every edge $\langle v_1, v_2 \rangle \in E(G)$, $v_1 \in V_1$ and $v_2 \in V_2$.

A *clique* is a subset S of $V(G)$, such that the graph $G[S]$ is a complete graph, namely for every two vertices in S there exists an edge connecting the two. A *biclique* in a bipartite graph, also called *bipartite clique*, is a set of vertices $B_1 \subseteq V_1$ and $B_2 \subseteq V_2$, such that $\langle b_1, b_2 \rangle \in E$ for all $b_1 \in B_1$ and $b_2 \in B_2$. In the rest of the paper we will say that a set of vertices S induce a biclique in a graph G if $G[S]$ is a complete bipartite graph. In the same way, we will say that a set of edges induce a biclique if their endpoints induce a biclique. A *maximal* clique or biclique is a set of vertices, that induces a complete subgraph and is not a subset of the vertices of any larger complete subgraph.

A *clique cover* of G is a collection of cliques $C_1, \dots, C_k$, such that for each edge $\langle u, v \rangle \in E$ there is some C_i that contains both u and v. A *minimum clique partition* (MCP) of a graph is the smallest collection of cliques such that each vertex is a member of exactly one of the cliques. It is a partition of the vertices into cliques. Similar to the clique cover, a *biclique cover* of G is a collection of biclique $B_1, \dots, B_k$ such that for each edge $\langle u,v\rangle \in E$ there is some B_i that contains both u and v. We say that B_i covers $\langle u,v\rangle \in E$ if B_i contains both u and v. Thus, in a biclique cover, each edge of G is covered at least by one biclique. A *minimum biclique cover* (MBC) is the smallest collection of bicliques that covers the edges of a given bipartite graph.

Role-Based Access Control and Definitions. We now sum up the main concepts of the RBAC model [1] that will be used in the rest of the paper. In particular, the entities of interest are: $PERMS$, that is the set of access permissions; $USERS$, namely the set of all system users; $ROLES$, that is the set of all the roles; $UA \subseteq USERS \times ROLES$, that is the set of user-role assignments; $PA \subseteq PERMS \times ROLES$, that is the set of permission-role assignments. Given a role, the function $\mathrm{assigned\_users}\colon ROLES \to 2^{USERS}$ identifies all the assigned users, and the function $\mathrm{assigned\_perms}\colon ROLES \to 2^{PERMS}$ identifies all the assigned permissions. In addition to the RBAC standard entities, the set $UP \subseteq USERS \times PERMS$ identifies permission to user assignments. In an access control system it is represented by entities describing access rights (e.g., access control lists).

To formally describe the role mining problem we need other definitions:

Definition 1 (System Configuration). *Given an access control system, we refer to its* configuration *as the tuple $\varphi = \langle USERS, PERMS, UP\rangle$, that is the set of all existing users, permissions, and the corresponding relationships between them within the system.*

A system configuration represents the user authorization state before migrating to RBAC, or the authorizations derivable from the current RBAC implementation.

Definition 2 (RBAC State). *An RBAC* state *is a tuple $\psi = \langle ROLES, UA, PA\rangle$, namely an instance of all the sets characterizing the RBAC model.*

An RBAC state is used to obtain a system configuration. Indeed, the role engineering goal is to find the "best" state that correctly describes a given configuration. In particular we are interested in the following:

Definition 3 (Candidate Role-Set). *Given an access control system configuration φ, a* candidate role-set *is the RBAC state ψ that "covers" all possible combinations of permissions possessed by users according to φ, namely a set of roles whose union of permissions matches exactly with the permissions possessed by the user. Formally: $\forall u \in USERS, \exists R \subseteq ROLES : \bigcup_{r\in R} \mathrm{assigned\_perms}(r) = \{p \in PERMS \mid \langle u,p\rangle \in UP\}$.*

Definition 4 (Cost Function). *Let Φ, Ψ be respectively the set of all possible system configurations and RBAC states. The* cost function *is defined as* $\mathrm{cost}\colon \Phi \times \Psi \to \mathbb{R}^+$. *It represents an administration cost estimate for the state ψ used to obtain the configuration φ.*

The administration cost concept was first introduced in [2]. Leveraging the cost metric makes it possible to find candidate role-sets which lead to the lowest possible effort for the administration of the resulting RBAC state.

Definition 5 (Optimal Candidate Role-Set). *Given a configuration φ, an* optimal candidate role-set *is the corresponding configuration ψ that simultaneously represents a candidate role-set for φ and minimized the cost function* $\mathrm{cost}(\varphi,\psi)$.

The main goal of role mining algorithm is thus finding optimal candidate role-sets. Considering the users and the permissions associated with a role, we define a weight function.

Definition 6 (Role Weight). *Let r be a given role, P_r be a set of permissions and U_r be a set of users associated to r. We indicate with $w(r)$ the weight of r, where $w(r)$ is the function defined as*

$$w(r) = c_u|U_r| \oplus c_p|P_r|,$$

where the operator "$\oplus$" is associative with respect to multiplication, and c_u and c_p are real numbers.

The role weight concept can be used as an indicator of the stability of a role. If a role r has a limited weight, it could be unstable, in that if a new user or a new permission is introduced it could drastically change the configuration of the role. In this case, it could be better to manage this role in a simpler way, namely breaking down the role in many single-permission roles which are easier to manage. The main idea is thus identifying the user-permission assignments that can belong only to roles with a limited weight. We manage these assignments with single-permission roles, restricting the role mining problem to the remaining user-permission assignments only. In this way, the elicited roles are *representative* and *stable*. Representative in that they are used by several users, or they cover several permissions, or both. Stable because they are not greatly affected by the introduction within the system of new users or new permissions. Once a set of roles that minimize the cost function has been found, introducing a new user or permission may change the system equilibrium whenever roles with limited weight exist. In particular, a new RBAC state could be necessary in place of the current one. This translates in higher administration cost, which is something that RBAC administrators tend to avoid. So, roles with a consistent weight are preferable, since they are more stable and less affected by the modifications of the existing user-permission assignments.

Due to space limitation, proofs of lemmas and theorems are not reported in this paper, but they can be found in [4].

4 Problem Modeling

The configuration $\varphi = \langle USERS, PERMS, UP\rangle$ can be represented by a bipartite graph $G = \langle V, E\rangle$, where the vertex set V is partitioned into disjoint subsets $USERS$ and $PERMS$, and two vertices $u \in USERS$ and $p \in PERMS$ are connected by an edge if and only if $\langle u, p\rangle \in UP$. A biclique coverage of the graphs G univocally identifies a candidate role-set $\psi = \langle ROLES, UA, PA\rangle$ for the configuration φ [6]. Indeed, every biclique identifies a role, and the vertices of the biclique identify the users and the permission assigned to this role.

Starting from the graph G, it is possible to construct a new undirected unipartite graph G', where the edges of G become the vertices of G': two vertices in G' are connected by an edge if and only if the endpoints of the corresponding edges of G induce a biclique of G. Formally: $G' = \left\langle E,\ \{\langle e_1, e_2\rangle \mid e_1, e_2 \text{ induce a biclique in } G\}\right\rangle$. That is, the vertices of a clique in G' correspond to a set of edges of G, where the endpoints induce a biclique in G. The edges covered by a biclique of G induce a clique in G'. Thus,

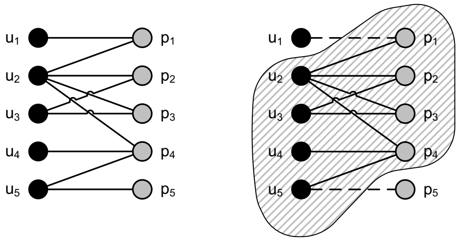

Fig. 1. An example of user-permission assignments and the subgraph to be considered

every biclique edge cover of G corresponds to a collection of cliques of G' such that their union contains all of the vertices of G'. From such a collection, a clique partition of G' can be obtained by removing any redundantly covered vertex from all but one of the cliques it belongs to. Similarly, any clique partition of G' corresponds to a biclique cover of G.

It is known that finding a clique partition of a graph $G = \langle V, E \rangle$ is equivalent to finding a coloring of its complement $\overline{G} = \langle V, (V \times V) \setminus E \rangle$ [6]. Thus, any coloring of $\overline{G'}$ identifies a candidate role-set of the given access control system configuration $\varphi = \langle USERS, PERMS, UP \rangle$, from which we have generated the graph G. Thus, finding a proper coloring for $\overline{G'}$ means finding a candidate role-set that covers all possible combinations of permissions possessed by users according to φ; namely, a set of roles such that the union of related permissions matches exactly with the permissions possessed by the users.

In this paper, we face the general problem of generating a candidate role-set that is stable and contextually minimizes the administration cost function. We split this problem in three steps:

1. Define a weight-based threshold.
2. Catch the user-permission assignments that do not belong to a role with a weight exceeding the given threshold.
3. Restrict the problem to find a set of roles that minimizes the administration cost function, including only the user-permission assignments that have not been identified in the second step.

In particular, we introduce a pruning operation on the vertices of $\overline{G'}$, that corresponds to identifying the user-permission assignments that can only belong to roles with a limited weight. These assignments are managed with single-permission roles, namely roles composed by only one permission. Using the pruning operation, we force the assignments to be assigned to roles with a higher weight. An example is shown in Figure 1: by creating single-permission roles for $u_1 p_1$, we are preventing the assignments $u_2 p_1$ for a role with a limited weight—composed by the users u_1 and u_2, and covering the permission p_1. The same happens for $u_5 p_5$ and $u_5 p_4$. Moreover, we will show that the portion of the graph that survives after the pruning operation is representable as a graph $\overline{G'}$ with a limited degree. Since the third step corresponds to coloring $\overline{G'}$, this property can be leveraged using many different coloring algorithms which make assumptions on the degree of the graph and available in the literature. The choice of which algorithm to use depends on the definition of the administration cost function.

4.1 Decomposition of the Bipartite Graph G

As seen in the previous section, any biclique coverage of G identifies a candidate role-set $\psi = \langle ROLES, UA, PA \rangle$ for the configuration φ. Moreover, any coloring of $\overline{G'}$ identifies a biclique coverage of G. If the graph G is not connected, it is possible to consider any connected component as a separate problem. Indeed, the union of the solutions of each component will be the solution of the original graph, as proved in the following lemma:

Lemma 1. *A biclique cannot exist across two or more disconnected components of a bipartite graph G.*

Since a biclique corresponds to a role, the previous lemma states that a role r, composed by a set of users U_r and a set of permissions P_r, cannot exist if all the users in U_r do not have all the permissions in P_r. If this is the case, we introduce some user-permission relations that are not in $\varphi = \langle USERS, PERMS, UP \rangle$. This lemma has an important implication, that is:

Theorem 1. *If G is disconnected, the union of the biclique coverage of each component of G is a biclique coverage of G. Moreover, if the biclique coverage of each component is optimal, their union will be an optimal biclique coverage for G.*

The main consequence of the theorem is that, if the graph G is disconnected, we can study each component independently. Therefore, we can use the union of the biclique coverage of the different components to build a biclique coverage of G. As we will see, we can use this result to limit the degree of $\overline{G'}$ when the bipartite graph G is disconnected.

4.2 Degree of $\overline{G'}$

In our model, the role mining problem corresponds to finding a proper coloring for the graph $\overline{G'}$. Indeed, this identifies a candidate role-set of the given access control system configuration. Depending on the cost function used, the optimal coloring could be different. For instance, if the cost function is defined as the total number of roles, the optimal coloring is the one which uses the minimum number of colors needed. However, if the cost function is more complex, the optimal coloring could be different. In this section we will analyze the degree of the graph $\overline{G'}$, highlighting what it represents and how this information can be used to simplify the role mining problem.

The graph $\overline{G'}$ is the graph composed by the same vertices of G', but $E(\overline{G'})$ is the complement of $E(G')$. The graph G' is built from the bipartite graph G, where each edge of G becomes a vertex of G', and two vertices of G' are connected by an edge if and only if the endpoints of the corresponding edges of G induce a biclique. Thus, a vertex $v \in G'$ univocally determines an edge $e \in G$. The degree of $v \in G'$ is the number of edges of G that induces a biclique together with e. Since $\overline{G'}$ is the complement of G', the degree of a vertex $v' \in \overline{G'}$ is the number of edges of G that do *not* induce a biclique together with the edge $e \in G$ identified by v'. According to the definition of the degree of a graph, the degree of $\overline{G'}$ is the maximum degree of its vertices. Formally, if e is an edge of the bipartite graph G, indicating with $b(e)$ the number of edges that do *not induce biclique* together with the edge e, then:

$$\Delta(\overline{G'}) = \max_{e \in E(G)} b(e).$$

To understand the meaning of the pruning operations we introduced, it is useful to describe the graphs in terms of RBAC semantic. A vertex of $\overline{G'}$ is a user-permission relation existing in the set *UP*. An edge in $\overline{G'}$ between two vertices v_1 and v_2 exists if the corresponding user-permission relations cannot be in the same role, because the user of v_1 does not have the permission in v_2, or the user of v_2 does not have the permission in v_1, or both. Thus, if a vertex of $\overline{G'}$ has a high degree, it means that this vertex cannot be colored using the same colors of a high number of other vertices; in other words, this user-permission relation cannot be in the same role together with a high number of other user-permission relations.

This consideration has an important aftermath: if a user-permission relation cannot be in the same role together with a high number of other user-permission relations, it will belong to a role with few user-permission relations, and we can estimate its maximal weight.

4.3 Pruning of the Given Access Control System Configuration

The main idea of our approach is pruning those user-permission relations which belong only to roles with a weight lower than a fixed threshold. If a role is composed only by few user-permission relations, its weight will be limited, and its administration cost could be acceptable even if we create for it a few single-permission roles. Moreover, when a change of the access control configuration happens, there is the need to recalculate the optimal candidate role-set. But a role with a limited weight is unstable, in the sense that the introduction of only one new user-permission assignment could drastically change the configuration of the role, according also to the specific cost function considered. Leveraging these observations, we will prune the given access control system configuration, creating single-permission roles for the pruned assignments and restricting the role mining to the rest of the assignments.

Suppose that for each $e \in E(G)$ there are at least d other edges, where the relative endpoints induce a biclique together with the endpoints of e. Every edge of G will not be in biclique with a maximum of $|E(G)| - d - 1$ other edges. That is:

Lemma 2. *If $\forall e \in E(G)$ there are at least d other edges, such that the endpoints of each one induce a biclique together with the endpoints of e, then:*

$$\Delta(\overline{G'}) \leq |E(G)| - d - 1$$

Thus, having chosen a suitable value for d, the idea is to prune the graph $\overline{G'}$ deleting the vertices that have a higher degree than $|E(G)| - d - 1$. Indeed, as it will be proven in the following Theorem 2, the user-permission assignments relative to these vertices will belong to roles with a limited weight, that could be administered using single-permission roles.

Theorem 2. *The pruning operation will prune only user-permission assignments that cannot belong to any role r, such that $w(r) > (d+1)(c_U \oplus c_P)$.*

Note that the pruning operation of the vertices of $\overline{G'}$ can be executed directly on the edge of G: it corresponds to the pruning of the edges $e \in E(G)$ such that $b(e) \geq |E(G)| - d - 1$. The pruning on G, rather than on $\overline{G'}$, is more convenient, because in this way we have to work directly with a smaller graph $\overline{G'}$. Once we have a graph $\overline{G'}$, with a maximum degree of $\Delta(\overline{G'})$ we can use a coloring algorithm to find a candidate role-set for the access control system configuration from which we have built the graph G. Many coloring algorithms known in the literature make assumptions on the degree of the graph. Without our pruning operation, the degree of the graph $\overline{G'}$ could be high, up to $|E(G)|-1$. This is the case when there exists a user-permission assignment that must be in a role alone. Note also that, when the graph G is disconnected in two or more components, any edge of one component is not in a biclique together with all the other edges belonging to different components. This involves a high degree for $\overline{G'}$, but for the argument given in Section 4.1 we can split the problem considering the different components distinctly, and then join the results of each component.

5 Applications of Our Approach to the Role Mining Problem

Having a bound for $\Delta(\overline{G'})$ makes it possible to use many known algorithms to color a graph. Indeed, finding a coloring for $\overline{G'}$ corresponds to finding a candidate role-set for the given access control system configuration. The choice of which algorithm to use depends on what we are interested in. For example, a company could be interested in obtaining only no more than a given number of roles, and to manage the others user-permission assignments with single-permission roles. This could happen when the company has just started a migration process to the RBAC model, and the administrators are not experts of role management. The *naive approach* presented in the following makes it possible to obtain no more than $\Delta+1$ roles (without considering the single-permission roles). The the algorithm choice can also depend on the particular class of the obtained graph. For instance, if the given access control system configuration induces a dense graph G, it is possible to obtain Δ/k roles, where $k=O(\log\Delta)$, covering almost all the existent user-permission assignments. We will show how this is possible by adopting a *randomized approach* based on [8].

A Naive Approach. It is known that any graph with maximum degree Δ can be colored with $\Delta+1$ colors by choosing an arbitrary ordering of the vertices and then coloring them one at time, labeling each vertex with a color not already used by any of its neighbors. In other words we can find $\Delta(\overline{G'})+1$ roles which cover all the user-permission assignments survived after the pruning. With the pruning operation, we are disregarding some user-permission assignments; this is the cost that we have to pay in order to have a Δ degree graph $\overline{G'}$. But the user-permission assignments that we are disregarding will belong to roles with a limited weight. So, it is better to create single-permission roles for those assignments.

The value $\Delta(\overline{G'})+1$ is not the optimum, it is only an upper bound for the chromatic number of $\overline{G'}$. If the coloring is optimal, namely if it uses the minimum number of colors needed, the candidate role-set found will be the smallest possible in cardinality. This problem is NP-hard and corresponds to *basicRMP*. If the degree of the graph is known, we can use many algorithms to approximate the solution.

A Randomized Approach. Using the randomized approach of [8] it is possible to generate Δ/k roles covering all the user-assignments which survive after the pruning, with $k=O(\log\Delta)$. This is a good result when minimizing the number of roles.

The hypothesis about which graph to color are basically two: it must be a Δ-regular graph, with $\Delta \gg \log n$, and it must have girths of length at least 4. The former is not a problem, because we can add some null nodes to the pruned graph $\overline{G'}$ and the relative edges obtaining a Δ-regular graph. The latter is more complex and we next discuss how to deal with this hypothesis. Every vertex of $\overline{G'}$ is a user-permission assignment in the given access control configuration; this means that it corresponds to an edge of G. Two vertices of $\overline{G'}$ are connected by an edge if the corresponding edges of G do not induce a biclique. Thus, a girth of length 3 in $\overline{G'}$ means that there are three edges of G, and every pair of this, does not induce a biclique. Having chosen two edges of G, let A be the event "these two edges induce a biclique" and let $\Pr(A) = p$. If $\overline{A}$ is the complement of A, then $\Pr(\overline{A}) = 1 - p$. Having chosen three edges, if B is the event "these three edges do not induce a biclique", $\Pr(B) = \Pr(\overline{A})^3$, indeed every unordered pair of the chosen triplet of edges must induce a biclique. Thus, $\Pr(B) = (1-p)^3$. This is also the probability that, having chosen three vertices in $\overline{G'}$, they compose a girth of length 3. In other words, the probability to have a girth of length 3 in $\overline{G'}$, depends on the number of edges of the graph G. Therefore, it depends on how many user-permission assignments exist in the given access control configuration. Indeed, if the number of edges of G is close to the maximal number of edges, the probability p will be very high, and $\Pr(B)$ will be close to 0. However, suppose that $\overline{G'}$ is not completely free of girths of length 3, but there are only a few of such girths. We can still use the randomized approach by removing an appropriate edge for such girths, hence breaking these girths. Note that removing an edge from $\overline{G'}$ corresponds to forcing two edges of G, which do not induce a biclique, to induce a biclique. This means that with this operation we are adding some user-permission assignments not present in the given access control configuration. The roles obtained can then be sanitized by removing those users that do not have all the permissions of the role, and managing these users in other ways, i.e. creating some single-permission roles (which roles will not be numerous since we have shown that, in general, we will only obtain few girth of length 3).

6 Conclusions and Future Works

In this paper we have proposed a general method to elicit roles using a bottom-up role engineering approach, with the objective to limit the presence of unstable roles and to minimize the administration cost function. We have proposed a three steps methodology that, leveraging the graph modeling of this role mining problem, achieves the intended results supporting them with formal proofs. A further contribution is to show the applications of the proposed approach. Possible extensions of this work could address: refining the weight function including, for instance, access logs or business information; defining the optimal value of the pruning parameter d.

Acknowledgment

This work was partly supported by: The Spanish Ministry of Science and Education through projects TSI2007-65406-C03-01 "E-AEGIS" and CONSOLIDER CSD2007-00004 "ARES", and by the Government of Catalonia under grant 2005 SGR 00446.

References

1. American National Standards Institute (ANSI) and InterNational Committee for Information Technology Standards (INCITS): ANSI/INCITS 359-2004, Information Technology – Role Based Access Control (2004)
2. Colantonio, A., Di Pietro, R., Ocello, A.: A cost-driven approach to role engineering. In: Proceedings of the 23rd ACM Symposium on Applied Computing, SAC 2008, Fortaleza, Ceará, Brazil, vol. 3, pp. 2129–2136 (2008)
3. Colantonio, A., Di Pietro, R., Ocello, A.: Leveraging lattices to improve role mining. In: Proceedings of the IFIP TC 11 23rd International Information Security Conference, SEC 2008. IFIP International Federation for Information Processing, vol. 278, pp. 333–347. Springer, Heidelberg (2008)
4. Colantonio, A., Di Pietro, R., Ocello, A., Verde, N.V.: Mining stable roles in RBAC, TR 01-01-09. Tech. rep., Università degli Studi di Rome Tre (2009), `http://www.dsi.uniroma1.it/~dipietro/TR01-01-09.pdf`
5. Coyne, E.J.: Role engineering. In: RBAC 1995: Proceedings of the first ACM Workshop on Role-based access control, p. 4. ACM, New York (1996)
6. Ene, A., Horne, W., Milosavljevic, N., Rao, P., Schreiber, R., Tarjan, R.E.: Fast exact and heuristic methods for role minimization problems. In: Proceedings of the 13th ACM Symposium on Access Control Models and Technologies, SACMAT 2008, pp. 1–10 (2008)
7. Geerts, F., Goethals, B., Mielikäinen, T.: Tiling databases. In: Suzuki, E., Arikawa, S. (eds.) DS 2004. LNCS, vol. 3245, pp. 278–289. Springer, Heidelberg (2004)
8. Grable, D.A., Panconesi, A.: Fast distributed algorithms for brooks-vizing colorings. J. Algorithms 37(1), 85–120 (2000)
9. Jajodia, S., Samarati, P., Subrahmanian, V.S.: A logical language for expressing authorizations. In: SP 1997: Proceedings of the 1997 IEEE Symposium on Security and Privacy, p. 31. IEEE Computer Society, Washington (1997)
10. Kuhlmann, M., Shohat, D., Schimpf, G.: Role mining – revealing business roles for security administration using data mining technology. In: Proceedings of the 8th ACM Symposium on Access Control Models and Technologies, SACMAT 2003, Como, Italy, pp. 179–186 (2003)
11. Lu, H., Vaidya, J., Atluri, V.: Optimal boolean matrix decomposition: Application to role engineering. In: Proceedings of the 24th IEEE International Conferene on Data Engineering, ICDE 2008, pp. 297–306 (2008)
12. Rymon, R.: Method and apparatus for role grouping by shared resource utilization (2003); United States Patent Application 20030172161
13. Schlegelmilch, J., Steffens, U.: Role mining with ORCA. In: Proceedings of the 10th ACM Symposium on Access Control Models and Technologies, SACMAT 2005, pp. 168–176 (2005)
14. Siewert, D.J.: Biclique covers and partitions of bipartite graphs and digraphs and related matrix ranks of $\{0, 1\}$ matrices. Ph.D. thesis, The University of Colorado at Denver (2000)
15. Vaidya, J., Atluri, V., Guo, Q.: The role mining problem: finding a minimal descriptive set of roles. In: Proceedings of the 12th ACM Symposium on Access Control Models and Technologies, SACMAT 2007, pp. 175–184 (2007)
16. Vaidya, J., Atluri, V., Guo, Q., Adam, N.: Migrating to optimal RBAC with minimal perturbation. In: Proceedings of the 13th ACM Symposium on Access Control Models and Technologies, SACMAT 2008, pp. 11–20 (2008)
17. Vaidya, J., Atluri, V., Warner, J.: RoleMiner: mining roles using subset enumeration. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 144–153 (2006)
18. Zhang, D., Ramamohanarao, K., Ebringer, T.: Role engineering using graph optimisation. In: Proceedings of the 12th ACM Symposium on Access Control Models and Technologies, SACMAT 2007, pp. 139–144 (2007)