On the Definition of Role Mining

Mario Frank
mario.frank@inf.ethz.ch

Joachim M. Buhmann
jbuhmann@inf.ethz.ch

David Basin
basin@inf.ethz.ch

Department of Computer Science
ETH Zurich, Switzerland

ABSTRACT

There have been many approaches proposed for role mining. However, the problems solved often differ due to a lack of consensus on the formal definition of the role mining problem. In this paper, we provide a detailed analysis of the requirements for role mining, the existing definitions of role mining, and the methods used to assess role mining results. Given basic assumptions on how access-control configurations are generated, we propose a novel definition of the role mining problem that fulfills the requirements that real-world enterprises typically have. In this way, we recast role mining as a prediction problem.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

General Terms

Security, Management

Keywords

RBAC, Role Engineering, Role Mining

1. INTRODUCTION

Research efforts on role mining started in 2003 when data mining techniques for role engineering were proposed in [15]. This research field has expanded rapidly since then. Numerous algorithms and approaches have been developed and commercial products are making their way onto the market. As the research field matures, we feel it is time to step back and take stock of some of the central questions in role mining. In particular, *what is role mining or what should it be?* It might appear that this question is already answered given the growing body of literature on the topic. However, as we will show, there is no consensus on this fundamental question and this has a profound effect on how research in the field is carried out and how results are selected and applied.

As it is generally understood, "role engineering" [5] is the task of configuring a Role-Based Access-Control (RBAC) system, i.e., creating roles and assigning users to roles and roles to permissions. The term "role mining" is used in a more narrow sense to refer to *automated* approaches to role engineering. These terms have been further refined. For example, one may distinguish between "top-down" and "bottom-up" approaches to role engineering [7]. Top-down role engineering uses descriptions of business processes, security policies, and other business information to configure an RBAC system. The bottom-up variant uses existing direct assignments between users and permissions, such as access-control lists (ACLs). Both these terms "top-down" and "bottom-up" have also been used in the context of role mining, where "bottom-up role mining" is often abbreviated simply as "role mining". Bottom-up role mining is therefore, roughly speaking, the automated migration of access-control based on direct assignments to an RBAC configuration.

Even though a general understanding exists of what role mining is, there is still no consensus on what constitutes a good role mining solution. One can distinguish three aspects of role mining:

1. the formal problem definition,
2. the role mining algorithm, and
3. quality measures for the assessment of role mining results.

The first aspect formally defines the goal of role mining by specifying what is given, what is assumed, and what must be found. The second aspect concerns the formalization of the approach taken to solve the problem by giving an algorithm. The third aspect addresses how the results are evaluated. In general, all three of these aspects are interrelated and, ideally, they are in agreement. That is, the algorithm should solve the formulated problem in that it returns the best possible result as defined by the quality measure.

To contrast these three possibilities, consider an example from the first role mining paper [15]. The authors used an existing data-mining technique [12] that clusters the set of users so that both the intra-cluster homogeneity of the given user attributes and the inter-cluster separability is high. Afterwards, some of the clusters are identified as roles by manual inspection. Going through our list, we note that (1) a formal problem definition is lacking, although it is informally stated that role mining is a kind of automated construction of the access-control configuration. (2) the role mining algorithm maximizes the combination of an intra-cluster similarity measure and an inter-cluster dissimilarity measure.

(3) the quality assessment employed in [15] verifies for a dataset with known roles that these roles are actually discovered by the algorithm.

The literature provides numerous examples of all three aspects. For example, five formal problem definitions of role mining are given in [13, 17, 25]. In addition, many quality measures have been employed in the literature and even more algorithms and objective functions to optimize have been proposed for role mining, each of which correspond to another (implicitly given) role mining problem.

We see several problems with the current state of affairs in role mining. First, the formal problem definitions given usually only partially reflect actual real-world requirements for role mining. We will review these definitions in Section 4 and see that there is a divergence between theory and practice. The problems addressed by researchers are often not the ones faced by actual enterprises migrating to RBAC.

Second, there exists no consensus on which of the formally defined problems should be solved in practice. In fact, most of the algorithms and quality measures given in recent papers often do not fully comply with any existing formal problem definition. This indicates that, even though individual researchers have their own understanding of the problem they are solving, a consensus on the problem definition does not yet exist. In Sections 4 and 5, we review the role mining literature in this respect.

Finally, rectifying the lack of consensus does not appear to be a priority: there is no ongoing discussion in the literature aimed at reaching a consensus. We believe that this lack of consensus limits progress in role mining. It leads to researchers solving different problems and developing different algorithms and quality measures. Not only does this dilute efforts, it makes it difficult to compare approaches. For example, by employing different quality measures, every role mining method can be best! Moreover, the lack of consensus makes it difficult for new researchers to orient themselves in this comparatively young field.

Given the above, we believe that it is important to start a dialog on the objective of role mining that will hopefully lead to a consensus. This would substantially advance the entire field. Our contribution is to provide a stable basis for discussion by carrying out a careful analysis of the existing definitions and quality measures. To do this, we first identify the most important requirements for RBAC that any real-world RBAC configuration should satisfy. Second, we analyze existing problem definitions with respect to these requirements, thereby identifying shortcomings. Finally, we propose a new definition of the role mining problem that specifies solutions that fulfill these requirements.

The remainder of this paper is organized as follows. In Section 2, we introduce terminology. In Section 3, we summarize the most important requirements on role mining. Based on these requirements, we analyze the existing formal definitions, objective functions, and quality measures in Sections 4 and 5. In Section 6, we propose a novel definition of the role mining problem and explain how it reflects the role mining requirements. Finally, we draw conclusions in Section 7.

2. PRELIMINARIES

First, we will introduce the basic entities of core RBAC as defined in the RBAC standard [8]. There are:

- USERS, the set of users,
- PRMS, the set of permissions.
- ROLES, the set of roles,
- $\mathit{UA} \subseteq \mathit{USERS} \times \mathit{ROLES}$, a user-role assignment relation,
- $\mathit{PA} \subseteq \mathit{ROLES} \times \mathit{PRMS}$, a role-permission assignment relation, and
- $\mathit{UPA} \subseteq \mathit{USERS} \times \mathit{PRMS}$, a user-permission assignment relation.

As usual in the context of role mining, we neglect the notion of sessions.

The assignment relation between entities is conveniently represented by a Boolean matrix where a 1 at the (i, j)-th entry indicates an assignment of i to j and a 0 indicates no assignment. We will denote the matrix representation of a relation A as $M(A)$. When there are m users, n permissions, and k roles, then $M(\mathit{UA})$ is a $m \times k$ Boolean matrix, $M(\mathit{PA})$ is a $k \times n$ matrix, and $M(\mathit{UPA})$ is a $m \times n$ matrix. We will use k, m, and n with this meaning throughout the paper.

The Boolean product $C = A \otimes B$ of two matrices $A \in \{0,1\}^{m \times k}$ and $B \in \{0,1\}^{k \times n}$ is $c_{ij} = \bigvee_{l=1}^{k} (a_{il} \wedge b_{lj})$. The Boolean matrix product will prove useful since, given a user-role assignment relation UA and a role-permission assignment relation PA, the matrix representation of the equivalent user-permission assignment relation UPA can be expressed as $M(\mathit{UPA}) = M(\mathit{UA}) \otimes M(\mathit{PA})$.

We use $\mathit{RC} = (\mathit{ROLES}, \mathit{UA}, \mathit{PA})$ to denote an *RBAC configuration.* If RC is the output of a role mining algorithm, we will sometimes call it the *role mining result.* There exist methods that, in addition to a user-role assignment UA and role-permission assignment PA, output a direct assignment relation DUPA. For outputs including DUPA, we will use the same notation for an RBAC configuration: $\mathit{RC} = (\mathit{ROLES}, \mathit{UA}, \mathit{PA}, \mathit{DUPA})$. We will be explicit about which kind of configuration is intended when this distinction is necessary. Direct assignments DUPA are not required for RBAC [8]. However, this relation is frequently used to express assignments that do not fit into the role structure. Moreover, by defining a new role for each of the individual direct assignments, one can, in principle, map such a configuration to standard RBAC (even though such specialized roles are usually not desirable).

Role mining approaches that take business information, also called "top-down information", together with UPA as input are called hybrid role mining methods. Top-down information often includes, for instance, locations, department affiliations, or task descriptions of the users. Alternatively it might refer to the confidentiality levels or the criticality of permissions or to security policies. Generally speaking, top-down information complements the data on USERS, PRMS, and UPA. The representation of top-down information can be of different forms. It can be of a categorical nature, a ranking type, relational, or numerical. For example, the location of users at one of the enterprise branch offices could be modeled as categorical data (e.g. $l_i = p$, if user i has location p). Rank-type top-down information can be found in military domains. One example of relational top-down information is a separation-of-duty policy represented by a binary $n \times n$ matrix S, where the entry $S_{ij} = 1$

indicates that permissions i and j should never be jointly assigned to the same role. Finally, the budget ceiling of a user (e.g. $b_i = \$10,000$) is an example of numerical top-down information. Throughout this paper, we will denote any kind of top-down information by TDI. Depending on what kind of top-down information is available, TDI can have specific representations and values. Examples of hybrid role mining algorithms along with different types of top-down information can be found in [3, 10, 19].

We will often use the notion of consistency as defined in [25] (in [1] the term "complete" is used instead):

Definition 1 (δ-Consistency). *A user-role assignment UA, role-permission assignment PA, and user-permission assignment UPA are δ-consistent if and only if*

$$\|M(UA) \otimes M(PA) - M(UPA)\| \leq \delta \ ,$$

where $M(UA)$, $M(PA)$, and $M(UPA)$ denote the matrix representation of UA, PA, and UPA respectively and $\|.\|$ is the L_1 norm for matrices with $\|M\| = \sum_i^m \sum_j^n |M_{ij}|$.

If an RBAC configuration is 0-consistent with the direct assignments UPA, it is usually just called "consistent".

3. REQUIREMENTS FOR ROLE MINING

In order to reason about what role mining should achieve, it is necessary to first specify what real-world enterprises require of RBAC configurations. Role mining requirements are then those that favor such configurations. This transition step from real-world requirements to an abstract problem space enables us to analyze and compare different problem definitions with respect to the same frame of reference. In the following, we briefly explain the main requirements for role mining. Since we aim to achieve a consensus on the goal of role mining, we restrict ourselves to the most fundamental requirements, which we expect most researchers and practitioners to agree with.

Requirement 1: Provisioning. An RBAC configuration must enable the users to carry out their tasks.

On a technical level, this means that an RBAC configuration must provide users with the privileges they need to perform actions on the resources they require.

Requirement 2: Security. An RBAC configuration must provide security.

This means that the configuration should conform to the security policies of the enterprise such that no user can access resources that he is not authorized to access, that separation of duty constraints are fulfilled, etc. As a result, if there are errors in the direct assignment UPA, role mining should be able to detect them in order to prevent these errors from being migrated to the RBAC configuration.

From a more practical perspective, RBAC configurations should comply to audit requirements and simplify security audits. The configuration should, for instance, make it easy to answer questions like: "Why does x have access to y?" This is closely related to Requirement 3.

Requirement 3: Maintainability. An RBAC configuration should be as easy to maintain as possible.

Maintainability lowers administration costs and helps administrators to work with the system. Here, three main aspects come into play: **minimality**, **interpretability**, and **generalization ability**.

First, most researchers consider a small RBAC configuration easier to maintain than a large one. Such a configuration has, for instance, few roles or few assignments.

Second, a configuration that is easy to interpret is easier to maintain than one where the roles are artificial and unintuitive. Interpretability can be achieved if the RBAC configuration is in agreement with the business processes and with the business properties of the users, such as their associated departments, locations, or tasks. Ideally, a user's roles correspond to the business roles that he works in and the two entities can be identified. In this case, one could even use the name of the business role (e.g. "receptionist, head office") as the name of the access-control role. This would enable system administrators and business employees to speak in the same language.

Finally, a configuration that allows administrators to easily add new users is easier to maintain than one where roles must be added or modified whenever new employees are added to the system. Configurations with a high generalization ability are comprised of roles that are stable under employee fluctuations. This substantially increases maintainability since the administrative effort to add new users or to move them within the enterprise is minimal if such actions can be made without requiring the RBAC configuration to be modified. Roles that generalize well are usually in agreement with the business processes of the enterprise in that they endow the user with necessary permissions independent of who is engaged in these processes.

Minimality, interpretability, and generalization ability are all interrelated. As already noted, interpretability increases maintainability. Moreover, it simplifies security audits because provisioning is better supported by meaningful roles. Having roles that conform to business processes requires that the roles contain the permissions needed to carry out these processes, but not more (i.e. least privilege). Since roles that are interpretable and generalize well usually reflect the business processes, they also fulfill the provisioning requirement. Moreover, security requires that the risk that future insecure assignments are granted based on the current RBAC configuration is low. This requires that administrators understand the role configuration, thereby avoiding errors. Hence, an intuitive and maintainable configuration increases security.

Note that minimality cannot always be jointly achieved with interpretability and generalization ability. The smallest system configuration is often not identical to the configuration with the highest interpretability and generalization ability. Hence, one must prioritize. Which aspect is more important? As long as administrators can understand the configuration and the configuration can be used with new users without many modifications, it is not problematic if the configuration is larger than the minimized configuration could, in principle, be. If, in turn, minimization comes at the price of low interpretability or low generalization ability, then the system is harder to administer. Hence, finding roles that are interpretable and generalize well should be favored over minimizing the configuration's size. We will therefore give minimality a lower priority, when we analyze problem definitions in terms of the maintainability of their solutions.

4. EXISTING DEFINITIONS

4.1 Review of the literature

In this section we review the existing formal definitions of the role mining problem. We restrict our attention to those definitions that are formally defined in the sense that they have a well-defined mathematical meaning. By distinguishing between problem definition, algorithm, and quality assessment as explained in the introduction, we can also investigate the goals formulated in the literature that are not formally defined but rather implicitly given (see Section 5).

The first three formal definitions of the role mining problem are proposed in [25].

Definition 2 (Basic Role Mining Problem (RMP)). *Given a set of users U, a set of permissions PRMS, and a user-permission assignment UPA, find a set of roles, ROLES, a user-role assignment UA, and a role-permission assignment PA, consistent with UPA that minimizes the number of roles k.*

Definition 3 (δ-approx RMP). *Given a set of users U, a set of permissions PRMS, and a user-permission assignment UPA, find a set of roles, ROLES, a user-role assignment UA, and a role-permission assignment PA, δ-consistent with UPA that minimizes the number of roles k.*

Note that Definition 2 is a special case of Definition 3 with $\delta = 0$.

Definition 4 (min-noise RMP). *Given a set of users U, a set of permissions PRMS, a user-permission assignment UPA, and the number of roles k, find a set of k roles, ROLES, a user-role assignment UA, and a role-permission assignment PA, minimizing*

$$\|M(UA) \otimes M(PA) - M(UPA)\|,$$

where $M(UA)$, $M(PA)$, and $M(UPA)$ denote the matrix representation of UA, PA, and UPA respectively.

In subsequent work [17], the problem is defined as minimizing the number of assignments of a consistent RBAC configuration.

Definition 5 (min-edge RMP). *Given a set of users U, a set of permissions PRMS, and a user-permission assignment UPA, find a set of roles, ROLES, a user-role assignment UA, and a role-permission assignment PA, consistent with UPA and minimizing $|UA| + |PA|$.*

A variant that also takes the role hierarchy of an RBAC configuration into account is considered in [13].

Definition 6 (Role Hierarchy Mining Problem). *Given a set of users U, a set of permissions PRMS, and a user-permission assignment UPA, find a set of roles, ROLES, a user-role assignment UA, a role-permission assignment PA, and a complete role hierarchy, $RH = G(V, E)$, that are consistent with UPA and that minimize $k + |E|$.*

In the same paper, a variant of Definition 6 is proposed that also takes predefined roles as input. The goal is then to find roles that are "close" to the deployed ones (as in [26]). Since we would like to focus on situations where no deployed roles are known at the time of role mining, we do not further investigate this variant.

For so-called hybrid role mining methods, which take both business information and a direct access-control system *UPA* as input, we could not find a formal problem definition in the literature. However, several methods for hybrid role mining have been proposed in [3, 10, 19]. Some authors also denote by hybrid role mining those bottom-up methods that are followed by, or used together with, the manual inspection of an expert who knows the business properties (i.e. in [11]). However, this naming convention is not consistent with the use of role mining for fully automated methods. We advocate instead using a term like "machine-assisted role engineering".

We analyze the above definitions in the next section and relate them to associated quality measures in Section 5.

4.2 Analysis

The existence of multiple definitions already suggests that there may be not a universal one. Given this, how do the above definitions differ and in which situations should they be used? Moreover, what advantages and shortcomings do they share? We provide orientation.

The above definitions all aim for a notion of minimality of the RBAC configuration. They vary in just two aspects: How configuration size is measured (number of roles, number of assignments, size of the hierarchy, etc.), and whether the RBAC configuration is required to be consistent with the given direct assignments *UPA*.

We first consider the question of whether an RBAC configuration should be 0-consistent with *UPA*. There seems to be no consensus on this question in the literature, as can be seen in the bottom two rows of Table 1 given in Section 5. Ultimately, the question boils down to two mutually exclusive assumptions on the properties of *UPA*:

1. *UPA* grants each user exactly the permissions he needs and no more.

2. Some of the assignments in *UPA* might be either unnecessary or missing.

Clearly, under the first assumption, consistency should be a part of the definition of role mining. Under the second assumption, consistency should be dropped.

The second assumption appears to be more realistic. *UPA* is influenced by human factors. For example, a user may require a privilege that he originally did not posses but which he needs to perform a special, exceptional task. Once given additional permissions, users often keep them because they do not view this as a problem. More concretely, a user may change his position within the enterprise and keep some of his old permissions to help his successor during the transition period. Later, he may neglect to have these permissions removed. Errors may also arise because administrators simply make mistakes and wrongly assign privileges.

In the remainder of this paper, we denote all assignments in *UPA* that are modified due to such exceptions or mistakes as *exceptional assignments* and we denote the actions that lead to exceptional assignments as *modification processes.* Given that *UPA* can be modified by so many processes, we advocate using definitions that do not require RBAC configurations to be 0-consistent. Our discussions with practitioners indicate that they share the second assumption. Dropping the notion of consistency reduces the set of compatible existing formal definitions to Definitions 3 and 4.

Another problem is that the definitions above that do *not* aim for a consistent solution, require additional information, which is usually not available at the time the problem is to be solved. For Definition 3, one must know the number of assignments δ that can be either dropped or additionally given. Knowing δ requires knowledge of the percentage of exceptionally given or missing assignments in *UPA*. For Definition 4, the number of roles k is needed as input. Both δ and k are difficult to know. The upper-bound of k is the Boolean rank or Schein-rank of $M(UPA)$, which would conform to a consistent role mining solution and cannot be efficiently found [18] (this problem is equivalent to Definition 2, which is shown to be NP-hard in [25]). If a small δ is allowed, k can be considerably smaller than the Boolean rank. For an estimate of k in $O(|USER|^2)$-time, based on visual inspection, one could, for instance, employ the method proposed in [31].

All definitions share the notion of minimality. As discussed in Section 3, small configuration sizes are beneficial since they can improve maintainability. However, if one tries to minimize the number of roles or the number of assignments (either for consistent RBAC configurations or RBAC configurations with a given approximation error δ), the roles and role assignments to users are then defined such that the roles cover as many of the given direct assignments as possible without granting extra permissions. As a result, the roles can end up quite synthetic and unintuitive. As Vaidya et. al. note [26]: “Minimality is a good notion since it allows one to formally define the problem” but “such a role discovery process can only serve as a guideline to security administrators to launch RBAC.”

In summary, each of the above definitions runs contrary to at least one of the basic requirements for role mining as given in Section 3. The notion of consistency is not reasonable and setting k or δ in advance is dangerous. Setting δ too low would result in migrating some of the exceptional assignments to the RBAC configuration. This runs contrary to the principle of least-privilege and thus does not fulfill Requirement 2 for security. Setting δ too high would result in an RBAC configuration that does not provide all users their needed permissions. This would violate Requirement 1 for provisioning. Finally, the goal of minimality contradicts the requirement for intuitive roles that generalize well and that correspond to the business properties of the enterprise. Role mining is, after all, not just a compression problem!

5. EXISTING QUALITY MEASURES

As pointed out in the introduction, there exist three aspects that explicitly or implicitly determine the goals of role mining: The formal problem definition, the algorithm employed, and the quality measure used for assessment. A thorough analysis of the role mining problem should consider all three aspects.

In addition to formal problem definitions, many quality measures and algorithms for bottom-up role mining have been introduced in the literature. Often, the corresponding publications do not include a formal problem definition (see Table 1). However, in many of these cases, the quality criterion given provides an implicit problem definition. Note, in this regard, that many of the given quality measures are contradictory and thus correspond to different (implicit) problem definitions. In this section, we survey all existing quality measures for role mining and the most prominent algorithms. A summary of the most prevalent concepts is provided in Table 1.

5.1 Definition vs. algorithm vs. assessment

First, we would like to compare these three aspects and explain why it is beneficial to distinguish them.

Definition vs. assessment.

The problem definition and the quality measure used for assessment are related in a simple way: an adequate quality measure quantifies if (or how well) the defined problem was solved. This relation is undirected: Associated to each given measure is a hypothetical problem that one tries to solve. Hence, it is desirable to have a common definition of the problem since it enables one to pinpoint an associated quality measure for role mining solutions that can be used to compare role mining approaches.

Why should one distinguish the two concepts? First, for some of the problems defined, it is computationally intractable to evaluate *if* they were solved. For instance, checking if a role mining result solves the basic RMP from Definition 2 is NP-complete [25]. For computationally intractable problems, one cannot even efficiently check *how good* the result is because evaluating how close one is to the solution requires first knowing the solution. For such problems, one can only compare pairs of RBAC configurations in order to see which one is closer to the solution (in the case of basic RMP, this would correspond to comparing the number of roles). Such a comparative assessment already deviates from the original problem definition. Moreover, there are far more quality measures than problem definitions (compare columns 2 and 3 in Table 1) thus making it convenient to group the measures in a distinct category.

Definition vs. algorithm.

Some role mining approaches are not defined in terms of an algorithm, but rather by defining an objective function or cost function that must be optimized. Since the optimization of a cost function is again an algorithm, we do not further distinguish the two cases. The term “algorithm”, as we use it here, shall also cover such approaches.

If a role mining algorithm is based on optimizing an objective function, this function determines the problem being solved. However, often there are deviations between the problem one wants to solve and the algorithm used. First, in most cases, the problems defined are NP-hard (for Definitions 2-4 this was shown in [25]). Therefore the algorithm used often employs a heuristic, which does not actually solve the given problem but rather a computationally tractable approximation. The algorithm thus solves the approximating problem and not the original problem. Second, some problem definitions do not directly suggest a way to solve the problem. The definition that we propose provides an example of this, as will be discussed in Section 6.1.

Algorithm vs. assessment.

There is a natural quality measure for role mining algorithms that optimize an objective function $f(RC)$. For a given RBAC configuration RC', one can simply compute the function's value $f(RC')$ (or the difference of $f(RC')$ from the optimum, if known) to quantify the solution's quality. This is often used in practice (see Table 1). However, since the problem definition and the algorithm often deviate, and

	formal definition	solution algorithm	quality measure
size of RBAC configuration (number of roles, no. of assignments, etc.)	[13, 17, 25]	[13, 17, 21, 24, 29, 26]	[2, 13, 17, 26, 29]
linear combination of size measures (*wsc* or "costs" with specified weights)		[1, 3, 4, 6, 16, 19, 20, 23]	[1, 3, 4, 6, 16, 19, 20, 23]
comparison with original roles (if known)		[13, 23, 26]: situation where some deployed roles are given	[9, 13, 15, 20, 22, 23, 26] [27, 30]
likelihood		[9, 10, 22]	
agreement with top-down information		[3, 10, 19]	[3, 9, 10, 11, 19]
0-consistency with *UPA*: required	[13, 17, 25]	[1, 2, 3, 4, 6, 13, 17] [23, 26, 29]	
0-consistency with *UPA*: not required	[25]	[9, 10, 16, 19, 21, 22] [27, 30]	

Table 1: Usage statistics for the most prevalent concepts used for the definition, algorithms, and solution assessment of the role mining problem.

since the quality measure should refer to the problem definition rather than to the algorithm, this might not always be the best choice.

Given possible deviations between the three aspects, we put forth that research on role mining approaches should ideally answer all three questions. What is the problem? What algorithm is used to solve it? And how is the solution assessed?

5.2 Measures corresponding to existing definitions

There are obvious measures that evaluate how well the five problems defined in Section 4.1 are solved. For Definitions 2 and 3 this is the number of roles, for Definition 4 the approximation error δ, for Definition 5 the number of assignments, and for Definition 6 the number of roles plus the size of the role hierarchy. These measures are employed in the papers that give the problem definitions [13, 17, 25]. An advantage of these definitions is that solutions can be assessed so easily. However, the notion of minimality, which is the key criterion for these definitions, should not be the primary requirement for RBAC configurations as explained in Section 4.2

5.3 Weighted Structural Complexity

The most frequently used quality criterion is the *weighted structural complexity* (*wsc*) [16]. This measures the size of an RBAC configuration as a linear combination of a set of individual measures of the size (such as number of roles, number of assignments, etc.).

Definition 7 (Weighted Structural Complexity). *Given $W = (w_r, w_u, w_p, w_h, w_d) \in \mathbb{R}^5$, the weighted structural complexity $wsc(RC, W)$ of an RBAC configuration $RC = (ROLES, UA, PA, DUPA)$ is*

$$\begin{aligned} wsc(RC, W) &= w_r|R| + w_u|UA| + w_p|PA| \\ &\quad + w_h|t_{reduce}(RH)| + w_d|DUPA| \, . \end{aligned}$$

Here, RH is the role hierarchy and $t_{reduce(RH)}$ denotes the transitive reduction of RH, which is the minimal set of relationships that encodes the same hierarchy as in RH. Note that, in contrast to the original definition in [16], we have dropped some of the constraints on the weights W (such as being natural numbers) since these parameters must be specified externally anyway. Moreover real weights are also reasonable.

With appropriate weights, $wsc(RC, W)$ equals the quality measures given in Section 5.2. Hard constraints, such as consistency, can be imposed by setting the corresponding weights (w_d) to a very large number (in the literature this is often formalized as ∞). Another quality measure that is similar to $wsc(RC, W)$ is the cost function defined in [1]. There, the two terms $|t_{reduce}(RH)|$ and $|DUPA|$ are dropped and, instead, an additional term $\sum_k c(r_k)$ is introduced that penalizes or rewards specific roles r_k with a function $c(r)$ that must be predefined by the role miner.

Note that the quality criterion *wsc* could be used to formally define the role mining problem. The problem to minimize $wsc(RC, W)$ is similar to the min-edge RMP in Definition 5. The difference is that the edge measure $|UA| + |PA|$ in the min-edge RMP is replaced by *wsc*.

In [16], a decision problem based on *wsc* is defined that asks whether there is an RBAC configuration with a given *wsc* that is consistent with a given *UPA*:

Definition 8 (*wsc* Decision Problem). *Given a set of users U, a set of permissions $PRMS$, a user-permission assignment UPA, an integer c, and weights W, does an RBAC configuration exist that is consistent with ρ and has $wsc(\rho, W) \leq c$?*

The high flexibility of *wsc* due to the unspecified weights W is, at the same time, its biggest shortcoming. There are several variants of how to set the weights (of *wsc* and costs) in the literature [1, 3, 4, 6, 16, 19, 20, 23], and, moreover, multiple weight configurations are considered in the same publications. However, to the best of our knowledge, there is little discussion on which weights should be used in practice.

The problem is that if the weights W are not specified, then $wsc(RC, W)$ fails to answer the question of what is a good quality criterion. It is obvious that the quality criterion could be any linear combination of the individual measures. We have just shifted the question of how the original measures should be defined to the question of how the weights should be chosen. Thus, $wsc(RC, W)$ defines an entire family of quality measures. Moreover, the full space of possible quality measures is not covered since all combinations of nonlinear terms are neglected. Finally, the same considerations on the notion of minimality apply, as discussed in Section 4.2.

5.4 Comparison with original roles

For experiments with artificially created data, we found four variants of a quality measure that compares the resulting set of roles with the ones that were used to create the data. The method used in [15] checks whether all original roles used to construct the data are discovered by the role mining method. The fraction of roles that equal the original ones is used for assessment in [27] and [28].

A relaxed version of this comparison of the computed set of roles with the original roles is proposed in [20]. There, for each role discovered, the maximal Jaccard-Coefficient over all pairings with original roles is taken. The final distance to the original role-set R_{orig} is then

$$d(R, R_{orig}) = avg_{r_i \in R}\left(\max_{r_j \in R_{orig}} (Jaccard(r_i, r_j)) \right) . \quad (1)$$

In the context of role mining, the Jaccard coefficient was first used as an optimization criterion for role mining scenarios where some deployed roles are given in advance[13, 26].

In [22], a measure based on the average Hamming distance between the original and the mined roles is proposed. In contrast to (1), where the $\max_{r_j \in R_{orig}}$ operation holds for each computed role r_i (such that each role is compared to the most similar original role), in [22] a single permutation of all roles is taken that gives a one-to-one mapping between the discovered roles and the original roles. This prevents any two discovered roles r_i and $r_{i'}$ from being compared to the same original role r_j.

As we will see later, these four measures could serve as quality criteria that would correspond to the new definition of the role mining problem as advocated in Section 6.1. The limitation of such measures is clear: they do not work if the real roles are not known. In real life this is always the case!

5.5 Generalization error

The generalization error is a quality measure that is often used for assessing supervised learning methods for prediction [14]. In [10, 22], it is used to assess the results of role mining algorithms. Here we briefly describe how it is computed. In Section 6.4, we propose this measure to serve as an evaluation method for our definition of the role mining problem and investigate it further there.

To compute the generalization error, one must randomly split the available dataset UPA along the users. The larger fraction UPA_1 is given to the role mining algorithm that is to be assessed and a smaller fraction UPA_2 is kept secret from the algorithm. After mining an RBAC configuration RC_1 based on UPA_1, one tests how good this configuration generalizes to the remaining users in UPA_2. To do this, a small fraction of permissions is revealed (say 10%) and used to determine which of the roles from RC_1 best suit them. The remaining fraction of permissions (here, 90%) are then predicted by these roles. The generalization error is the deviation of the predicted permissions from the actual permissions. This deviation is computed, for example, as the fraction of wrongly predicted permissions.

6. A NEW DEFINITION OF ROLE MINING

In this section we propose a novel definition of the role mining problem. First, we explain the assumptions underlying our definition. Then we give a general definition that covers the bottom-up role mining problem as well as hybrid role mining. The bottom-up problem will be a special case of the general problem, where the input differs but the goal remains the same. Afterwards, we show that a solution to the problem, as it is defined here, fits the requirements described in Section 3. Moreover, we describe how the problem could be approached and propose quality measures for assessing solutions.

6.1 Assumptions and Problem Definition

The input of the role mining problem is a set of users $USERS$, a set of permissions $PRMS$, a user-permission assignment relation UPA, and, depending on its availability, top-down information TDI. Our problem definition is based on three assumptions about the generation process of UPA and we begin by motivating these assumptions.

Assumption 1: Exceptions exist. In Section 4.2, we saw that it is reasonable to assume that there are exceptions in the relation UPA, which arise from modification processes. We formalize this assumption by defining an unknown relation UPA', without exceptions. The relation UPA' is then perturbed by modification processes, leading to the observable relation UPA. In Section 4.2, we gave examples of different modification processes that can lead to perturbations. We abstain from making further assumptions on them such as, for instance, the fraction of exceptions δ. We consider the perturbations as unknowns and simply assume their existence.

In practice, one hopes that the perturbations do not fully obscure the data's regularities and thus role mining is still feasible. For given data, the fraction of perturbed assignments influences the difficulty of the problem, but does not change the goal of role mining. Some of the perturbation processes might be deterministic, whereas others might be random. In order to account for random modifications, we assume that UPA is a random variable that is drawn from a probability distribution $p(UPA|UPA')$ that is conditioned on UPA'. This formulation includes the case where modifications are due to a deterministic function $UPA = f(UPA')$, which is expressed by simply setting $p(f(UPA')|UPA') = 1$.

Assumption 2: An underlying RBAC configuration exists. The second assumption is that UPA' was induced by an unknown RBAC configuration $RC^* = (ROLES^*, UA^*, PA^*)$, where "induced" means that $UPA' = UA^* \otimes PA^*$. This is not a strong assumption. We believe that most researchers and practitioners involved in role mining actually make this assumption implicitly. To search for roles implicitly assumes that they are there to be found. Said another way, searching for roles in direct assignments between users and permissions only makes sense if one assumes that the data could, in principle, be organized in such a structured way. Of course, one might try to find a role structure in assignments that are completely random (i.e., each individual assignment is an exception and there are no structural dependencies between them). But what one finds then are random roles without any business semantics; synthetic sets of permissions will emerge that are found only because exceptional assignments are randomly aggregated in a way that mimics structure.

Assumption 3: TDI influences RC^*. The third assumption is that a relationship exists between RC^* and the top-down information TDI. We assume that RC^* reflects the security policies of the enterprise in the sense that RC^* complies to these policies. Moreover, the business properties of the enterprise influence the generation of RC^* and therefore RC^* reflects these properties. For role mining, not all top-down information might be available. In practice, the role miner has only a subset of this information, for example, the affiliations of users to organizational units. Sometimes one has no TDI at all. Since the unknown part of TDI might still have influenced RC^*, we only assume that RC^* was influenced by *partially* given TDI.

The generation process of *UPA*, under Assumptions 1-3, is sketched in Figure 1. The entities that are input for role mining are drawn in black boxes and the entities that are unknown are gray.

Given the above assumptions on *UPA*'s generation process, we propose the following definition of the role mining problem.

Definition 9 (Inference RMP). *Let a set of users USERS, a set of permissions PRMS, a user-permission relation UPA, and, optionally, part of the top-down information TDI be given. Under Assumption 1-3, infer the unknown RBAC configuration* $RC^*=(ROLES^*, UA^*, PA^*)$.

Our definition, together with the assumptions on *UPA*'s generation process, provides a unified view of bottom-up and hybrid role mining. In both cases, the RBAC configuration RC^* must be discovered. In both cases, RC^* is assumed to induce *UPA* (modulo perturbations) and in both cases RC^* is assumed to be influenced by top-down information. The two cases only differ in terms of the availability of top-down information TDI. In hybrid role mining, a non-zero fraction of all top-down properties that influenced RC^* is available. When no TDI is given, the problem reduces to bottom-up role mining. Note that, in such cases, the goal still remains the same: the solution of Problem 9 solves the bottom-up problem as well as the hybrid role mining problem. Only the input differs. The assumption that RC^* is (partially) influenced by TDI is also reasonable for the pure bottom-up role mining problem. Whether TDI actually influences RC^*, does not depend on the availability of such data for the role miner. Available TDI is desirable since it can provide additional evidence of what RC^* might be and thus can be used in role mining.

There are two challenges faced when using this definition in practice. First, it does not provide an objective function to optimize. That is, the problem definition does not itself indicate how to solve the problem. Hence, some creativity is required to devise an algorithm or an objective function for this problem. Second, there is no obvious quality measure that can be easily computed. To validate that the problem was solved, one must know the hidden underlying RBAC configuration RC^*. Except for experimental scenarios with artificially created data, this information is rarely accessible. However, these challenges do not invalidate the problem definition. In Section 6.3, we explain how solutions could be found and we propose a quantitative quality measure that can be even used in the case where RC^* is unknown.

6.2 Relation to RBAC requirements

We believe that Definition 9 is a good candidate for the actual problem that researchers in role mining should be

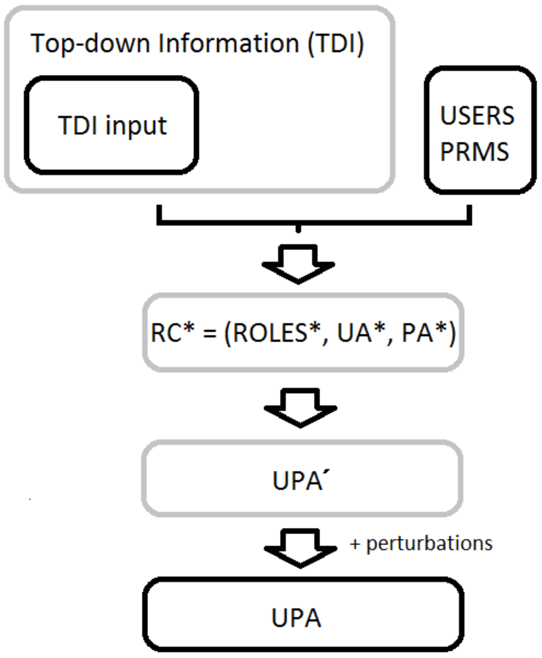

Figure 1: Scheme of the assumed generation process of the direct user-permission assignment *UPA*. Grey entities are unknown and black ones are given as input for role mining. For pure bottom-up role mining, no top-down information is given at all.

solving. We support this by showing that a solution to Problem 9, which is the RBAC configuration RC^*, fits well the requirements on role mining given in Section 3. Intuitively, since RC^* is generated by the security policies and the business properties of the enterprise, RC^* complies with the security policies and reflects the business properties and thereby naturally fulfills the above requirements.

To explain this in more detail, consider the generation process of *UPA*, sketched in Fig. 1. Top-down information influences the generation of RC^* in that the administrators account for TDI when setting up the access-control system and when changing the configuration. The RBAC configuration RC^* underlying the direct assignments thus reflects TDI. As a consequence, if a role mining solution results in RC^*, then this configuration reflects the business processes and the users' business features and complies with the security policies. When RC^* reflects TDI, then RC^* satisfies Requirements 1-3. Requirement 1, provisioning, is satisfied because when the RBAC configuration reflects the business processes, users can take part in these processes by being assigned to the roles that satisfy the provisioning requirement. RC^* satisfies Requirement 2, security, since it complies with the security policies of the enterprise. Requirement 3, maintainability, is satisfied because when the RBAC configuration reflects the business features of the users and reflects the business processes, the roles are intuitive for humans that know these processes and features and the roles also generalize well. This makes the RBAC configuration easy to maintain. Hence, by defining the goal of role mining as inferring RC^*, a solution of the problem satisfies these fundamental role mining requirements.

6.3 Associated algorithms and quality measures

The best way to search for the hidden RBAC configuration RC^* is to search for the configuration with the highest probability given UPA, namely $p(RC|UPA)$. This can be done using the maximum likelihood principle. We took this approach, for instance, in [22]. For a given UPA, maximum likelihood denotes the RBAC configuration RC_{max} that maximizes the function $p_L(UPA|RC)$, where $p_L(UPA|RC)$ is the probability that UPA would be generated if RC were the underlying RBAC configuration. Maximum likelihood assumes that the RBAC configuration that makes the generation of UPA most likely has the highest probability to be the real underlying RBAC configuration. For a given UPA, this assumption holds if the prior probabilities $p(RC)$ of all possible RBAC configurations RC are equal. One can see this by employing Bayes rule:

$$p(RC|UPA) \quad = \quad \frac{p_L(UPA|RC)p(RC)}{p(UPA)}\,. \qquad (2)$$

Maximum likelihood requires that, to compute the value of $p_L(UPA|RC)$, one must first define the likelihood function $p_L(a|b)$, which requires modeling of the generation process of the RBAC configuration. We derived such a probabilistic model in [9] from the logical structure of $RBAC$ and used different variants of it [9, 10, 22] for role mining. One must also account for the assumption that $p(RC)$ is the same for all RC. Some RBAC configurations might have a higher probability than others per se, that is, without considering a particular UPA. If such prior knowledge is available, then it must be modeled too. In [9], for instance, a so-called Dirichlet process is taken as $p(RC)$.

Hybrid role mining requires that TDI is available in the first place. Moreover, not all available TDI might be useful for role mining. Consider, for example, the users' home address or gender. TDI that contradicts the role structure present in UPA could be even misleading since it usually renders the underlying optimization problems ill-conditioned. One must therefore understand how (and if) a particular kind of TDI influences RC^*. This must be modeled with a probability distribution $p(RC|UPA, TDI)$ that includes TDI.

To the best of our knowledge, such a model has not been discussed in the literature. All hybrid role mining approaches to date [3, 10, 19] use deterministic cost functions $f(TDI, RC)$ to reward RBAC configurations that agree with TDI. The approach that we present in [10] optimizes the likelihood $p(RC|UPA)$ for given UPA and, thereby, takes TDI into account via a linear combination of the log-likelihood and a deterministic cost function $f(TDI, RC)$. The selection process of a particular kind of TDI is based on an information-theoretic measure of relevance. However, a full probabilistic model including TDI has not been given. Moreover, as with many other role mining methods, the work presented in [10] lacks a formal definition of the problem. Nevertheless, the method described there could be used to approximately solve the problem that we propose here.

As already pointed out, the obvious quality measure that corresponds to our definition is the comparison with the hidden RBAC configuration RC^* underlying the given data UPA. If RC^* is not known, as in real-world scenarios, it cannot be compared with the discovered solution RC. However, in experiments with artificially created data this is possible. Two major design decisions are required to assess solutions. First, one must define a similarity measure $d(RC_1, RC_2)$. Counting the number of correctly recovered roles as done in [15, 27, 28] has the disadvantage that tiny deviations to an original role are as severe as a full disagreement. We therefore recommend using more relaxed measures such as the Jaccard-coefficient (used in [20]) or the Hamming distance (used in [22]), both discussed in Section 5.4. Second, the data used for experiments must be generated so that it has properties similar to those of real-world data. Simulating real-world access-control configurations poses a substantial challenge. Failure to do so would limit the significance of experiments with synthetic data.

Since in the real world, the original RBAC configuration RC^* is unknown, assessment is much more challenging there. However, even in this case one can quantitatively assess how close the solution is to RC^*. As will be explained in the next section, the best measure for carrying out this assessment is the generalization error as described in Section 5.5.

6.4 Role mining as a prediction problem

We advocate the generalization error as the appropriate quality measure for solutions of the problem as defined in Definition 9. In the following, we explain why the most predictive role configuration, i.e. the one with lowest generalization error, is RC^*.

Problem Definition 9 assumes that UPA was induced by RC^*. If we split UPA in two parts, UPA_1 and UPA_2 (as done when computing the generalization measure), this assumption still holds for both parts: RC^* underlies both UPA_1 and UPA_2. RC^* should thus be the best predictor for assignments in both UPA_1 and UPA_2. One can try to discover RC^* based on one part and then use the discovered roles to predict the permission assignments of the other. The closer the RBAC configuration is to RC^*, the lower the generalization error will be. As a consequence, the solution that generalizes the best is also the best solution for the role mining problem defined here.

Given the above, the generalization error can be used to recast the proposed problem definition in terms of this quality measure. Namely:

> Given $USERS$, $PRMS$, UPA, and, optionally, TDI, find the RBAC configuration RC^* that minimizes the empirical generalization error.

Hence, ultimately, role mining is a prediction problem.

7. CONCLUSION

We have carried out a detailed analysis of the existing definitions, algorithms, and assessment methods for role mining. As we have seen, there is a lack of consensus on goals and this leads to very different approaches to the problem. We have also shown how existing definitions fail to account for some of role mining's practical requirements. This has motivated us to propose a new definition of the role mining problem. Our problem definition is based on three assumptions that we carefully justified and its solution fulfills the most fundamental requirements for role mining. We additionally proposed approaches suitable to solving the problem and explained methods to validate solutions.

We see several directions for future research. As we previously indicated, there is currently no model that describes the influence of top-down information on the generation of access-control configurations. Such a model would allow us to develop hybrid role mining methods that better reflect the interrelationships between top-down information and the roles that are to be discovered. Designing such a model is a difficult, but worthwhile, task. Another direction is developing foundations for the risk analysis of access-control configurations. The ability to compute the risk of fraud, the risk that a user is lacking a permission that he needs, or the risk that an administrator does not understand a role and commits an error, would enable one to reason about RBAC configurations from a different perspective.

Acknowledgments.
This work was partially supported by the Zurich Information Security Center. It represents the views of the authors.

8. REFERENCES

[1] A. Colantonio, R. Di Pietro, and A. Ocello. A cost-driven approach to role engineering. In *SAC '08*, volume 3, pages 2129–2136, Fortaleza, Brazil, 2008.

[2] A. Colantonio, R. Di Pietro, and A. Ocello. Leveraging lattices to improve role mining. In *SEC '08*, volume 278, pages 333–347, 2008.

[3] A. Colantonio, R. Di Pietro, A. Ocello, and N. V. Verde. A formal framework to elicit roles with business meaning in RBAC systems. In *SACMAT '09*, 2009.

[4] A. Colantonio, R. Di Pietro, A. Ocello, and N. V. Verde. Mining stable roles in RBAC. In *SEC '09*, volume 297, pages 259–269, 2009.

[5] E. J. Coyne. Role engineering. In *RBAC '95*, page 4, New York, NY, USA, 1996. ACM.

[6] A. Ene, W. Horne, N. Milosavljevic, P. Rao, R. Schreiber, and R. E. Tarjan. Fast exact and heuristic methods for role minimization problems. In *SACMAT '08*, pages 1–10, New York, NY, USA, 2008.

[7] P. Epstein and R. Sandhu. Engineering of role/permission assignments. In *ACSAC '01*, page 127, Washington, DC, USA, 2001. IEEE Computer Society.

[8] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. Proposed NIST standard for role-based access control. *ACM Trans. Inf. Syst. Secur.*, 4(3):224–274, 2001.

[9] M. Frank, D. Basin, and J. M. Buhmann. A class of probabilistic models for role engineering. In *CCS '08*, pages 299–310, New York, NY, USA, 2008. ACM.

[10] M. Frank, A. P. Streich, D. Basin, and J. M. Buhmann. A probabilistic approach to hybrid role mining. In *CCS '09*, pages 101–111, New York, NY, USA, 2009. ACM.

[11] L. Fuchs and G. Pernul. Hydro — hybrid development of roles. In *ICISS '08*, pages 287–302, Berlin, Heidelberg, 2008. Springer-Verlag.

[12] J. Grabmeier and A. Rudolph. Techniques of cluster algorithms in data mining. *Data Mining and Knowledge Discovery*, 6(4):303–360, 2002.

[13] Q. Guo, J. Vaidya, and V. Atluri. The role hierarchy mining problem: Discovery of optimal role hierarchies. In *ACSAC '08*, pages 237–246, Washington, DC, USA, 2008. IEEE Computer Society.

[14] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer Series in Statistics. Springer, 2001.

[15] M. Kuhlmann, D. Shohat, and G. Schimpf. Role mining – revealing business roles for security administration using data mining technology. In *SACMAT '03*, pages 179–186, New York, NY, USA, 2003. ACM.

[16] N. Li, T. Li, I. Molloy, Q. Wang, E. Bertino, S. Calo, and J. Lobo. Role mining for engineering and optimizing role based access control systems. Technical report, November 2007.

[17] H. Lu, J. Vaidya, and V. Atluri. Optimal Boolean matrix decomposition: Application to role engineering. In *ICDE '08*, pages 297–306, Washington, DC, USA, 2008. IEEE Computer Society.

[18] G. Markowsky. Ordering d-classes and computing Schein rank is hard. *Semi-group Forum, 44*, pages 373–375, 1992.

[19] I. Molloy, H. Chen, T. Li, Q. Wang, N. Li, E. Bertino, S. Calo, and J. Lobo. Mining roles with semantic meanings. In *SACMAT '08*, pages 21–30, New York, NY, USA, 2008. ACM.

[20] I. Molloy, N. Li, T. Li, Z. Mao, Q. Wang, and J. Lobo. Evaluating role mining algorithms. In *SACMAT '09*, pages 95–104, New York, NY, USA, 2009. ACM.

[21] J. Schlegelmilch and U. Steffens. Role mining with ORCA. In *SACMAT '05*, pages 168–176, New York, NY, USA, 2005. ACM.

[22] A. P. Streich, M. Frank, D. Basin, and J. M. Buhmann. Multi-assignment clustering for Boolean data. In *ICML '09*, pages 969–976, New York, NY, USA, 2009. ACM.

[23] H. Takabi and J. Joshi. StateMiner: An efficient similarity-based approach for optimal mining of role hierarchy. In *CCS '09, Poster Session*, 2009.

[24] R. Thion. Découverte automatisée de hiérarchies de rôles pour les politiques de contrôle d'accès. *INFORSID'07*, pages 139–154, May 2007.

[25] J. Vaidya, V. Atluri, and Q. Guo. The role mining problem: finding a minimal descriptive set of roles. In *SACMAT '07*, pages 175–184, New York, NY, USA, 2007. ACM.

[26] J. Vaidya, V. Atluri, Q. Guo, and N. Adam. Migrating to optimal RBAC with minimal perturbation. In *SACMAT '08*, pages 11–20, New York, NY, USA, 2008. ACM.

[27] J. Vaidya, V. Atluri, and J. Warner. Roleminer: mining roles using subset enumeration. In *CCS '06*, pages 144–153, New York, NY, USA, 2006. ACM.

[28] J. Vaidya, V. Atluri, J. Warner, and Q. Guo. Role engineering via prioritized subset enumeration. *IEEE Transactions on Dependable and Secure Computing*, 99, 2008.

[29] D. Zhang, K. Ramamohanarao, and T. Ebringer. Role engineering using graph optimisation. In *SACMAT '07*, pages 139–144, New York, NY, USA, 2007. ACM.

[30] D. Zhang, K. Ramamohanarao, T. Ebringer, and T. Yann. Permission set mining: Discovering practical and useful roles. In *ACSAC '08*, pages 247–256, Washington, DC, USA, 2008. IEEE Computer Society.

[31] D. Zhang, K. Ramamohanarao, S. Versteeg, and R. Zhang. Rolevat: Visual assessment of practical need for role based access control. In *ACSAC '09*, pages 13–22, Los Alamitos, CA, USA, 2009. IEEE Computer Society.