The addRole-EA: A New Evolutionary Algorithm for the Role Mining Problem

Simon Anderer[1], Daniel Kreppein[1], Bernd Scheuermann[1] and Sanaz Mostaghim[2]
[1]*Faculty of Management Science and Engineering, Hochschule Karlsruhe, Moltkestrasse 30, Karlsruhe, Germany*
[2]*Institute for Intelligent Cooperating Systems, Otto-von-Guericke Universität, Magdeburg, Germany*

Keywords: Role Bases Access Control, Role Mining Problem, Evolutionary Algorithm.

Abstract: Role Based Access Control is an important feature of cyber security and authorization management. The search of an optimum set of roles and their assignment to users can be modeled as an optimization problem, which is known as the Role Mining Problem (RMP) and has been shown to be NP-complete. Thus, fast heuristics are needed to search for solutions for the RMP in affordable time. This paper proposes a new evolutionary algorithm for the RMP, which is based on iteratively adding and deleting roles, while avoiding deviations from the original user-permission assignment. A range of experiments are presented which indicate that the proposed EA performs well on established benchmarking problems.

1 INTRODUCTION

In our increasingly digitized world, cyber security plays an important role to protect data and information systems. Especially, enterprise data are a likely target of external attacks. In addition, erroneous and fraudulent behavior of employees can lead to substantial damage (Verizon, 2019). To overcome these issues, access control systems manage authorization by assigning access rights to users, such that critical data is only accessible by a restricted set of selected users. While originally, permissions were directly assigned to users, nowadays they are combined to roles as additional level of access control. This approach is known as *Role Based Access Control* (RBAC). It was defined by (Sandhu et al., 1996) and is now one of the widest-spread access control models, as the grouping of permissions to roles reduces complexity, while increasing security comprehensibility and manageability, thus leading to reduced security management costs.

The task of defining roles is called *role engineering* and can be realized in two ways: top-down or bottom-up. The top-down approach consists in mostly human analysis of business functions and processes to define proper roles. Due to the large number of users and permissions and the even larger number of possible combinations, this is a very time-consuming and hardly scalable task. The bottom-up approach is based on the user-permission assignment, which, in most enterprises, is easily available (Mitra et al., 2016). At this, data-mining techniques are used to mine roles.

The corresponding optimization problem is called the *Role Mining Problem* (RMP) and is NP-complete (Vaidya et al., 2007). This paper presents an Evolutionary Algorithm for the RMP, the *addRole-EA*. Additionally, an overview of current work on the RMP is given.

The remainder of this paper is organized as follows: In Section 2, the formal framework of the RMP is presented. Section 3 reviews different solution strategies for the Role Mining Problem with more detailed focus on EA-based approaches. Section 4 illustrates the setup and structure of the addRole-EA. In Section 5, the proposed algorithm is evaluated. Section 6 and 7 conclude the paper and present paths for future research.

2 PROBLEM DESCRIPTION

In this section, the Role Mining Problem and some of its variants are presented. A first definition of the RMP was given in (Vaidya et al., 2007). In the following, the standard notation of the main elements of the RMP are introduced:

- $U = \{u_1, u_2, ..., u_m\}$ a set of $m = |U|$ users
- $P = \{p_1, p_2, ..., p_n\}$ a set of $n = |P|$ permissions

Anderer, S., Kreppein, D., Scheuermann, B. and Mostaghim, S.
The addRole-EA: A New Evolutionary Algorithm for the Role Mining Problem.
DOI: 10.5220/0010025401550166
In *Proceedings of the 12th International Joint Conference on Computational Intelligence (IJCCI 2020)*, pages 155-166
ISBN: 978-989-758-475-6

- $R = \{r_1, r_2, ..., r_k\}$ a set of $k = |R|$ roles
- $UPA \in \{0,1\}^{m \times n}$ the user-permission assignment matrix, where $UPA_{ij} = 1$ implies, that permission p_j is assigned to user u_i
- $UA \in \{0,1\}^{m \times k}$ the user-role assignment matrix, where $UA_{ij} = 1$ implies, that role r_j is assigned to user u_i
- $PA \in \{0,1\}^{k \times n}$ the role-permission assignment matrix, where $PA_{ij} = 1$ implies, that role r_i contains permission p_j.

From this, we can now give a definition of the *Basic Role Mining Problem*. Given a set of users U, a set of permissions P and a user-permission assignment UPA, find a minimal set of Roles R, a corresponding user-role assignment UA and a role-permission assignment PA, such that each user has exactly the set of permissions given by UPA:

$$|R| \rightarrow min \tag{1}$$

$$s.t. \quad \|UPA - UA \otimes PA\|_1 = 0 \tag{2}$$

where $\|.\|_1$ denotes the L_1-norm for matrices and $\otimes$ the Boolean Matrix Multiplication:

$$(UA \otimes PA)_{ij} = \bigvee_{l=1}^{k} (UA_{il} \wedge PA_{lj}) \tag{3}$$

A tuple $\langle U, P, R, UA, PA \rangle$ is called RBAC scheme. If constraint (2) holds, the corresponding RBAC scheme is said to be *0-consistent*. In further variants of the RMP, the 0-consistency-condition is relaxed, which means that the definition of roles and their assignment to users may result in deviations compared to the original UPA. Thus, we define the resulting user-permission assignment $DUPA := UA \otimes PA$. The different variants of the RMP can now be defined as in Table 1.

Table 1: Different variants of the Role Mining Problem based on (Saenko and Kotenko, 2016).

RMP-Variant	Objective	Constraint
Basic RMP	$\|R\| \rightarrow min$	$\|UPA - DUPA\|_1 = 0$
Edge RMP	$\|UA\|_1 + \|PA\|_1 \rightarrow min$	$\|UPA - DUPA\|_1 = 0$
Min. Noise RMP	$\|UPA - DUPA\|_1 \rightarrow min$	$\|R\| = k$ constant
δ-approx RMP	$\|R\| \rightarrow min$	$\|UPA - DUPA\|_1 \leq \delta$

There are several other RMP-variants considering specific business-driven aspects of role mining e.g. the interpretability or meaningfulness of roles (Xu and Stoller, 2012), (Molloy et al., 2008), administrative costs (Colantonio et al., 2008) or RBAC-reconfiguration problems (Saenko and Kotenko, 2016). All variants of the RMP have been proved to be NP-complete problems (Vaidya et al., 2007).

3 RELATED WORK

The Role Mining Problem and its different variants are well-studied problems. Many solution techniques have been applied in the last years e.g. clustering techniques, problem transformation, permission grouping etc. Subsection 3.1 gives a short overview of the applied methods. Subsequently, in subsection 3.2, evolutionary algorithms in the context of role mining are explained in more detail.

3.1 General Solution Strategies

One common approach consists in mapping the RMP to other problems in data mining. Lu et al. consider the different variants of the RMP as Matrix Decomposition Problems and use greedy algorithms to solve the corresponding binary integer programming problems (Lu et al., 2008). Huang et al. propose a pre-processing step to reduce the input UPA by removing similar users and users whose permission set corresponds to the union of other users' permission sets. Furthermore, they show the equivalence of the RMP and the Set Cover Problem and use greedy algorithms (GA_{Basic}, GA_{Edge}, MR_{Basic}) as solution strategy for the latter (Huang et al., 2015). Vaidya et al. map the RMP to the Minimum Tiling problem and use greedy algorithms to mine tiles, which are then converted into roles (Vaidya et al., 2007). In (Ene et al., 2008) the equivalence of the Basic-RMP and Minimum Biclique Cover Problem is shown and again greedy algorithms are applied as solution strategy. From this, the *Role Minimization* (RM), *Edge Concentration* (EC) and *Lattice Postprocessing* (LP) algorithms are derived. In addition, a set of real-world datasets, the *HP-Labs* benchmark set, is described, which now constitutes the standard benchmark for role mining.

Zhang et al. use graph optimization (GO) to mine roles (Zhang et al., 2007). They iteratively merge two roles depending on their intersection to improve the RBAC scheme. In (Zhang et al., 2010) roles are iteratively merged, created or deleted from the underlying graph to reduce the cost function. Dong et al. propose the Network-Clique Finding Model to map the Role Mining Problem onto problems of graph theory, which enables the use of graph optimization techniques for its solution (Dong et al., 2014).

One of the first role mining tools is *ORCA* (Schlegelmilch and Steffens, 2005). Roles are obtained by clustering permissions, thus obtaining a hierarchy of permission clusters. One drawback of this approach consists of the fact that overlapping of permissions is only allowed among roles that are hierar-

chically related. The work of (Kuhlmann et al., 2003) is also based on a clustering approach. Kumar et al. add an additional cardinality constraint to role mining, which limits the number of permissions in each role to a maximum number and propose the *Constrained Role Miner* (CRM), which is also based on permission clustering (Kumar et al., 2010). Molloy et al. propose the *HierarchicalMiner* (HM), which is based on formal concept analysis by using a reduced concept lattice as initial role hierarchy and then heuristically prunes it to obtain improved role concepts (Molloy et al., 2008).

Algorithms like the *Simple Role Mining Algorithm* (SMA) (Blundo and Cimato, 2010), *CompleteMiner* (CM) and *FastMiner* (FM) (Vaidya et al., 2010) or *Pair Count* (PC) (Molloy et al., 2009) are based on the concept of permission grouping. A set of candidate roles is obtained by grouping permissions to roles. This is often done by intersecting the permission sets of two or more users. Subsequently, the roles in the candidate set are ranked by different prioritization functions and then assigned to users. Zhang et al. use permission utilization counts to be able to also consider top-down information in their *Data-Driven Role Evolution approach* (DDRE) (Zhang et al., 2013). Lu et al. consider different sets of candidate roles. One set (IT) consists exactly of the user-permission-assignment defined by UPA. Hence, this corresponds to the Discrete Basis Problem. For the other set (INT), all intersections of two user permission sets are added. In both cases greedy algorithms are applied to obtain the final RBAC scheme (Lu et al., 2014).

A more detailed description of the different algorithms for the RMP can be found in (Mitra et al., 2016).

3.2 Evolutionary Algorithms in Role Mining

Even if evolutionary algorithms seem to be a straightforward approach in solving the Role Mining Problem, not much research has been conducted in this direction.

Hu et al. apply evolutionary algorithms in the RBAC context. However, instead of mining roles to improve RBAC schemes, the EA is used for insider threat detection by mining role-action mapping rules connecting roles to processes. Based on these mappings, discrepancies in process execution can be detected, which results in a reduction of internal fraudulent behaviors in enterprises (Hu et al., 2006).

Suganthy and Chithralekha examine the structural change in organizations including employees changing their compartment, new employees joining or existing employees leaving the organization (Suganthy and Chithralekha, 2018). From this, they derive theoretical methods for role evolution to adapt the outdated RBAC scheme to the new organization structure, comprising the creation of new roles, deletion of existing roles, merging and splitting of roles and delegation/revocation of permissions, which implies extending/reducing existing roles by a set of permissions. Although not having been directly implemented by the authors, these methods provide a good basis of evolutionary operators for evolutionary algorithms in the RBAC context.

Saenko and Kotenko are the first to apply EAs for Role Mining (Saenko and Kotenko, 2011). This approach assumes exactly $k = m$ roles at all times of the optimization process. This is due to the observation that the Role Mining Problem has a trivial solution in which each of the m users gets assigned only one role, that contains all of the user's permissions given by UPA. Each individual is represented by three chromosomes: the matrices $Chr[X] := UA^\top \in \{0,1\}^{m\times m}$ and $Chr[Y] := PA \in \{0,1\}^{m\times n}$, and an an additional vector $Chr[Z] \in \{0,1\}^m$. Note that the UA matrix is quadratic and transposed, such that the columns of both matrices correspond to roles. For an example see Figure 1.

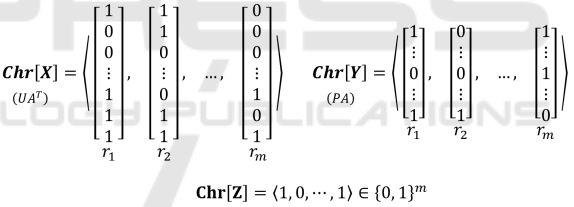

Figure 1: Representation of the individuals in (Saenko and Kotenko, 2011).

The entries of Z describe which role is active in the RBAC scheme corresponding to the individual. In the evolutionary loop, crossover is done by applying one-point-crossover to Z, UA and PA, while mutation on these elements is based on bit flip. A weighted sum of the number of active roles $\|Chr[Z]\|_1$ and the number of deviations between $DUPA = UA \otimes PA$ and the original UPA is used as fitness function. Finally, this approach is evaluated on synthetic data based on randomly generated binary matrices.

In (Saenko and Kotenko, 2012), the authors present an improved version of their EA. It is based on a new representation of individuals and the omission of passive roles. Each individual gets assigned one chromosome $Chr := \langle r_1, r_2, ..., r_s \rangle$, where each r_i corresponds to one role. Further, $r_i := \langle L_i^X, L_i^Y \rangle$, where L_i^X is the list of users and L_i^Y the list of permissions of role r_i, implying that each permission in L_i^Y is as-

signed to each user in L_i^X. For crossover, the one-point-crossover method is adapted to the new representation of the individuals (see Figure 2).

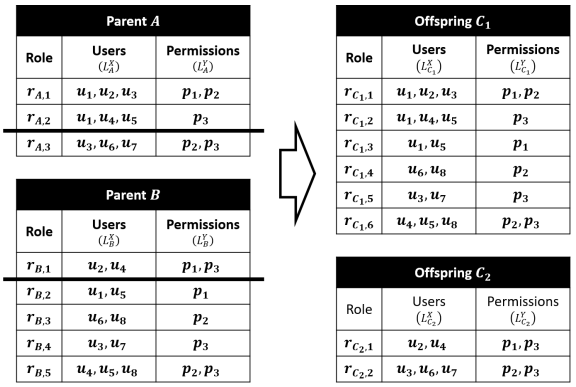

Figure 2: One-Point-Crossover adapted to the new representation of individuals in (Saenko and Kotenko, 2012).

Mutation is realized by adding, deleting or changing values in L_i^X and L_i^Y according to the given probability of mutation. The fitness function remains as in (Saenko and Kotenko, 2011). The authors show that the new representation of the individuals and especially the variable-length list of roles (containing only roles that are active in the sense of (Saenko and Kotenko, 2011)) result in a performance gain.

In (Kotenko and Saenko, 2015) the developed EAs are applied to the problem of finding optimal access control schemes in virtual local networks.

In (Saenko and Kotenko, 2016) (Saenko and Kotenko, 2017a) and (Saenko and Kotenko, 2017b) the *RBAC Scheme Reconfiguration Problem*, which is based on dynamically changing access control policies, is defined. Given an access control policy UPA_{init}, a corresponding RBAC scheme $\langle U, P, R_{init}, UA_{init}, PA_{init} \rangle$ and a new policy UPA_{new}, the Reconfiguration Problem aims at finding a new RBAC scheme $\langle U, P, R_{new}, UA_{new}, PA_{new} \rangle$ that on the one hand satisfies the requirement of the new policy as good as possible (4), while on the other hand has minimal differences to the initial RBAC scheme (5):

$$\|UPA_{new} - UA_{new} \otimes PA_{new}\|_1 \rightarrow min \quad (4)$$

$$\|UA_{new} - UA_{init}\|_1 + \|PA_{new} - PA_{init}\|_1 \rightarrow min. \quad (5)$$

At this, EAs are used for mining the initial RBAC scheme $\langle U, P, R_{init}, UA_{init}, PA_{init} \rangle$ as well as the reconfigured RBAC scheme $\langle U, P, R_{new}, UA_{new}, PA_{new} \rangle$. Furthermore, a new representation of individuals is presented, where the columns of UA and PA are interpreted as binary numbers (see Figure 3).

In (Saenko and Kotenko, 2018), the authors extend their work on RBAC to other domains of access control (VPN- and VLAN-problems) and evaluate their methods in a real-world case study on data derived from enterprise resource planning systems.

$$\boldsymbol{Chr}[X] \coloneqq UA^T = \left\langle \begin{bmatrix}0\\0\\1\\0\\1\end{bmatrix}, \begin{bmatrix}1\\0\\0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\1\\1\\1\\0\end{bmatrix} \right\rangle \mapsto \boldsymbol{Chr}[X]_{01} = (5, 18, 14)$$

$$\boldsymbol{Chr}[Y] \coloneqq PA = \left\langle \begin{bmatrix}1\\0\\1\\0\\1\\0\end{bmatrix}, \begin{bmatrix}1\\0\\0\\0\\0\\1\end{bmatrix}, \begin{bmatrix}0\\1\\0\\0\\1\\1\end{bmatrix} \right\rangle \mapsto \boldsymbol{Chr}[Y]_{01} = (42, 33, 19)$$

Figure 3: Representation of the individuals in (Saenko and Kotenko, 2017a).

In (Dong et al., 2011) a definition of the δ-*approx Least Privilege Mining Problem* is given. This is, given a set of users, a fixed set of roles in PA and a user-permission-assignment UPA, to find a user-role-assignment UA such that $\|UPA - UA \otimes PA\|_1 \leq \delta$. The authors apply an EA as solution strategy. In this approach, a role is defined by its users and permissions as in (Saenko and Kotenko, 2012), while the combination of different roles constitutes the chromosome of an individual. Subsequently, a series of crossover, mutation and selection operations, which are not further specified by the authors, are applied iteratively to improve the solution quality.

Du and Chang also use EAs to solve the Role Mining Problem (Du and Chang, 2014). Firstly, a set of candidate roles CR is created based on (Xu and Stoller, 2012), in which a role is again defined by the contained permissions and the set of users to which the role is assigned. Choosing a role r_i to be activated in an RBAC scheme implies assigning each of its contained permissions to each user in r_i. Secondly, an initial population of RBAC schemes is created randomly. Each individual has assigned two binary vectors S_i and H_i, where:

$$S_i := \langle s_i^{(1)}, s_i^{(2)}, ..., s_i^{(|CR|)} \rangle$$
$$H_i := \langle h_i^{(1)}, h_i^{(2)}, ..., h_i^{(|CR|)} \rangle.$$

At this, $s_i^{(j)} = 1$ means that role j is activated in the RBAC scheme corresponding to individual i, while $h_i^{(j)} = 0$ implies deactivation of role j in individual i in the next generation (only if possible without creating deviations compared to the original UPA). In the evolutionary loop, new individuals are created by one-point crossover and bit flip mutation of the H_i-vectors and the corresponding update of S_i. A 2-tuple consisting of a weighted sum to describe the structural complexity of an RBAC scheme and an interpretability-measure to describe the meaningfulness of the con-

tained roles (Xu and Stoller, 2012) constitutes the fitness function.

One main drawback of the presented algorithms is the violation of the 0-consistency property as the proposed mutation and crossover methods are designed in a way that causes deviations between the $DUPA = UA \otimes PA$ and the original user-permission assignment UPA. Furthermore, instead of the commonly used HP-Labs benchmark set, random matrices are used for evaluation, which complicates performance comparison.

4 A NEW EA FOR THE RMP

In this section, a new evolutionary algorithm for the Role Mining Problem, the *addRole-EA*, is presented. It is based on a new technique for addition and consequential deletion of roles to UA and PA, such that the 0-consistency property is fulfilled at all times.

At first, a pre-processing procedure, which is applied to initial input data of the EA to reduce the problem dimension, is presented. Subsequently, the main aspects of the addRole-EA are outlined. This comprises the representation of individuals, the addRole-method as primary method of the algorithm as well as the typical evolutionary components like fitness, mutation and crossover.

To provide an overview of the proposed role-mining algorithm, its components and their order, the operational course of the addRole-EA in Pseudo-Code is given below:

```
Input:  Users U, Permissions P,
        User-Permission Assignment UPA

Output: User-Role Assignment UA,
        Role-Permission Assignment PA

Begin addRole-EA
   Pre-Processing of initial UPA;

   Initialization();
   Evaluation of Fitness();

   Repeat (Until stopping condition met)
       Crossover();
       Mutation();
       Evaluation of Fitness();
       Replacement();
   End Repeat.

   Post-Processing of resulting UA and PA.
End.
```

4.1 Pre-processing

Simplification of input data, in our case, means deletion of redundant data in the UPA matrix, resulting in the reduction of the initial number users and permissions without effecting the solution of the corresponding RMP (see (Huang et al., 2015)). For this, we apply a four-step procedure:

(PP1): Deletion of Empty Rows and Columns.
Deletion of permissions that are assigned to no user respectively users that have an empty permission set. This equals the deletion of columns and rows of the UPA matrix that contain only zero-elements.

(PP2): Aggregation of Permissions.
Aggregation of permissions that are assigned to exactly the same users. This corresponds to the deletion of all columns, which are identical copies of another column, except for one representative.

(PP3): Aggregation of Users 1.
Aggregation of all users that have exactly the same permission sets. This corresponds to the deletion of all rows, which are identical copies of another row, except for one representative.

(PP4): Aggregation of Users 2.
Deletion of users whose permission set is equal to the union of other users' permission sets.

By applying the inverse of (PP1-4) to the resulting UA and PA matrices in a post-processing step, the proposed solutions can easily be extended to the original user and permission sets.

4.2 Chromosome Encoding

Each individual is represented by two chromosomes: its UA matrix and its PA matrix. In real-world use cases, the number of users and permissions is usually very large, whereas the number of roles per user and the number of permissions per role is quite low. Thus, the UA and PA matrices are of high dimension, but rather sparsely populated. This leads to a disproportionally high occupation of memory space, if the classical representation as binary matrices is utilized. As a consequence, we use binary sparse matrices to represent the pre-processed UPA as well as the corresponding UA and PA matrices in the addRole-EA. Unlike ordinary binary matrices, binary sparse matrices only store the position of the one-elements in each row, as illustrated in Figure 4. Thus, the zero-elements, which constitute the majority of the matrices' elements, are omitted, resulting in huge savings of memory space.

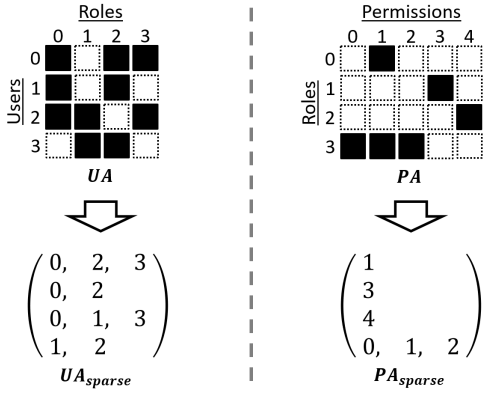

Figure 4: Representation of individuals in the addRole-EA.

4.3 Initialization

The initialization of the individuals is conducted in two steps. In the first step, an initial population of N_{pop} identical individuals are generated. We start from n different roles, each containing exactly one of the n permissions. Thus, $PA = I_n$, where I_n denotes the $n \times n$ identity-matrix. To ensure the 0-consistency property of the initial population, we set $UA = UPA$ (see Figure 5). In a second step, we use a series of mutation operations, which are explained in Section 4.7, to ensure diversity among the population. The selection of the mutation operations is conducted randomly, based on the uniform distribution.

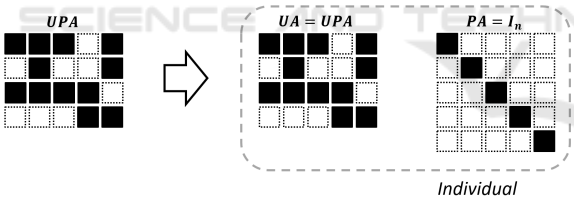

Figure 5: Initial individual before mutation.

4.4 The addRole-method

The principle method of the addRole-EA is the addRole-method, in which a new role is added to the UA and PA of a given individual. An example of the operating principle of the addRole-method can be found in Figure 6.

The addition of the new role to the PA matrix can easily be done by attaching a new row to the PA matrix, containing the permissions of the new role (aR-1). For the addition of the new role to the UA matrix, the permissions of the new role are compared to the permission sets of the users defined by the UPA matrix. If all permissions of the new role are contained in a user's permission set, the new role is assigned to that user (aR-2). As a result of adding new roles to an individual, an old role may become obsolete, in the sense that it can be removed from a user's role list, while the remaining roles of the user still cover all of the permissions contained in his permission set. If this holds for all users in the individual, the old role is deleted from its UA and PA matrix (aR3-5).

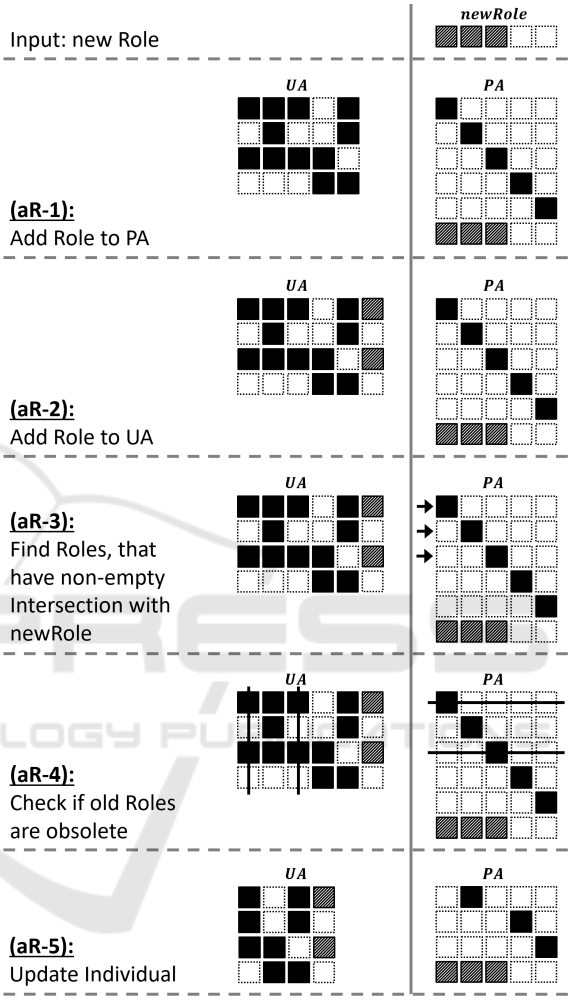

Figure 6: The addRole-method (example).

4.5 Fitness Function

The fact, that new roles are only assigned to users whose permission sets completely cover the permissions contained in the new role and old roles are only deleted if they became obsolete by adding the new role, ensures the 0-consistency property at all time of the EA. Thus, the constraint of the Basic Role Mining Problem $\|UPA - UA \otimes PA\|_1 = 0$ is always fulfilled and no penalty costs are required, when modeling the fitness function, which is therefore equivalent to the total number of roles $fitness = |R|$.

4.6 Replacement

Based on the values of the previously defined fitness function, next generations' individuals are chosen by the elitist selection scheme. This implies the survival of the best $elitismRate \cdot N_{pop}$ individuals, while the remaining $(1 - elitismRate) \cdot N_{pop}$ individuals are chosen randomly to maintain diversity.

4.7 Mutation

In each evolutionary step, a new role is created for each individual and then added to its *UA* and *PA* matrix using the addRole-method. The creation of roles is based on the permission grouping concept (see Section 3.1). To reduce the search space, only such permissions are grouped to roles that form a subset of at least one user's permission set. For this, the following role-creation methods are available:

(M1): Intersection of Permission Sets.
This role-creation method creates a new role by intersecting the permission sets of different users. For this, a random subset of all users is selected for permission set intersection. As a result, the new role can be assigned to at least all of the selected users. To avoid the creation of empty roles, the number of selected users is limited by $usersForIntersection_{max}$.

(M2): Merging of Two Roles.
Creating one new role from two old roles may directly decrease the total role number. To ensure that the new role is assigned to at least one user, this role-creation method selects a random user and two random roles of this user's role set to derive a new role. The role is then created by the union of the permissions of the two selected roles.

(M3): Role Setminus other Role.
Contrarily, reducing the number of permissions contained in a role, increases the probability that this role can be assigned to more users. For this, two roles r_1, r_2 of one user are selected randomly. If they have non-empty intersection, all permissions of r_2 are removed from r_1 to create the new role.

(M4): Permission Set of a User.
Whenever a user's permission set has little intersection with the permission sets of other users, it may be reasonable, to assign one role to this user that contains all of his permissions given by the user-permission assignment. Therefore, a new role is created that equals the permission set of a randomly chosen user.

(M5): Permission Set Setminus Union of Roles.
In practice, RBAC schemes are often constructed by distributing a set of predefined roles to users. In addition, special permissions which are not covered by an explicit role are given to selected users like managers or administrators. Thus, this role-creation method tries to gather the permissions, that cannot be covered by regular roles (roles derived from (M1-4)), in an extra role. For this, a user and a subset of this user's role set are selected randomly. Then, a new role is created containing all permissions of the user's permission set, except the ones contained in one of the selected roles.

An example of the different role-creation methods for mutation is illustrated in Figure 7.

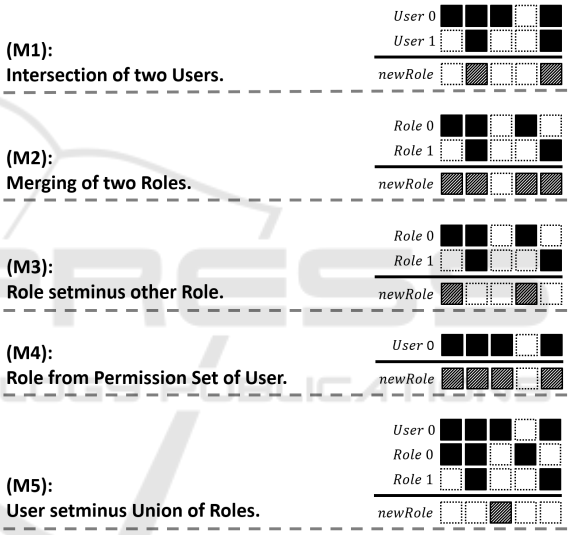

Figure 7: The role-creation methods (example).

Mutation-Method in Pseudo-Code:

```
Input:  Individual
Output: mutated Individual

Begin Mutation
   Choose role-creation method from (M1-5)
   randomly based on uniform distribution;

   Create new Role r based on chosen method;

   Mutate Individual by addition of r to
   individual using addRole-method;
End.
```

4.8 Crossover

The crossover method of the addRole-EA consists in reciprocal exchange of roles between individuals. At

first, two parent individuals $parent_1$ and $parent_2$ are selected for crossover. From these, two child individuals $child_1$ and $child_2$ are derived as copies of the corresponding parent. For each of the child individuals, a set of roles from the role set of the other parent is selected by one of the following role-selection methods and added to its *PA* and *UA* matrix by the addRole-method. As a consequence, roles are not only exchanged, but old roles that became obsolete are deleted from the UA and PA matrix of the child individuals. The main difference compared to the mutation method consists in the fact, that no new roles are created, but taken from already existing roles in the *PA* matrix of the other individual. There are three different methods for crossover role-selection:

(C1): Random-role Selection.
At this, a random set of roles is chosen from each parent individual's role set and then added to the opposed child individual by the addRole-method.

(C2): User-role Set Selection 1.
Instead of choosing random roles, one user is selected randomly. The roles for exchange are then determined by the roles assigned to this user in the respective parent individual.

(C3): User-role Set Selection 2.
This method again exchanges the roles of one user. Selection of the user is carried out by comparing the number of assigned roles to this user in the parent individuals, choosing the user with the highest difference.

Crossover-Method in Pseudo-Code:

```
Input:  Individual parent_1, parent_2
Output: Individual child_1, child_2

Begin Crossover
   Choose role-selection method from (C1-3)
   randomly based on uniform distribution;

   Initialize child_1 (copy of parent_1);
   Initialize child_2 (copy of parent_2);

   Select set of roles R2 from parent_2
   by chosen method;
   Select set of roles R1 from parent_1
   by chosen method;

   Add R2 to child_1 by addRole-method;
   Add R1 to child_2 by addRole-method;
End.
```

5 EVALUATION

The evaluation of the proposed approach is conducted in two steps. At first, the pre-processing procedure, defined in Section 4.1, is applied to the HP-Labs benchmark set. In a second step, these benchmarks are used to evaluate the performance of the addRole-EA. For more detailed information on the different benchmark instances see (Ene et al., 2008).

5.1 Pre-procession of Benchmark Data

In the first step of the addRole-EA, redundancies are removed from the input data, as outlined in Section 4.1. To examine the degree of redundancy, the four steps of the pre-processing procedure (PP1-4) are applied to the HP-Labs benchmarks. The results can be seen in Table 2, where the first line corresponding to each benchmark shows the original data sizes. As (PP2) only reduces the number of permissions, while (PP3) only influences the number of users, the results of those two steps are summarized in one line in the table. It is visible, that the application of (PP1) has no influence on the benchmark data. However, in real-world cases, it remains important to detect users without permissions or permissions that are assigned to no user.

Table 2: Pre-Processing of benchmark data.

Benchmark	Users	Prms.	Assign.	Occup.
America Large	3,485	10,127	185,294	0.53%
(PP1)	3,485	10,127	185,294	0.53%
(PP2 & 3)	432	1,354	18,779	3.21%
(PP4)	430	1,354	18,719	3.22%
America Small	3,477	1,587	105,205	1.91%
(PP1)	3,477	1,587	105,205	1.91%
(PP2 & 3)	259	349	6,035	6.68%
(PP4)	225	349	5,011	6.38%
APJ	2,044	1,164	6,841	0.29%
(PP1)	2,044	1,164	6,841	0.29%
(PP2 & 3)	564	578	2,089	0.64%
(PP4)	475	578	1,588	0.58%
EMEA	35	3,046	7,220	6.77%
(PP1)	35	3,046	7,220	6.77%
(PP2 & 3)	34	263	1,278	14.29%
(PP4)	34	263	1,278	14.29%
Healthcare	46	46	1,486	70.23%
(PP1)	46	46	1,486	70.23%
(PP2 & 3)	18	19	120	35.09%
(PP4)	16	19	98	32.24%
Domino	79	231	730	4.00%
(PP1)	79	231	730	4.00%
(PP2 & 3)	23	38	156	17.85%
(PP4)	20	38	146	19.21%
Firewall 1	365	709	31,951	12.35%
(PP1)	365	709	31,951	12.35%
(PP2 & 3)	90	86	935	12.08%
(PP4)	71	86	616	10.09%
Firewall 2	325	590	36,428	19.00%
(PP1)	325	590	36,428	19.00%
(PP2 & 3)	11	11	58	47.93%
(PP4)	10	11	51	46.36%

The results show that the pre-processing procedure can significantly reduce the size of the input data. Furthermore, comparing the lower bound of roles given in Table 3 to the remaining number of users after applying all pre-processing steps, reveals, that for *EMEA*, *Domino* and *Firewall 2* the optimum number of roles can already be attained by defining one role for each of the remaining users containing the corresponding permissions.

For the other benchmarks, Table 3 indicates that the difference between the number of remaining users and the role lower bound leaves little room for proper evaluation of role mining algorithms. Thus, future research should focus on the creation and provision of new and more conclusive benchmarks.

Table 3: Number of users after pre-processing compared to role lower bounds based on (Ene et al., 2008).

Benchmark	Users after (PP1-4)	Role Lower Bound	Δ
America Large	430	390	40
America Small	225	172	53
APJ	475	453	22
EMEA	34	34	0
Healthcare	16	14	2
Domino	20	20	0
Firewall 1	71	64	7
Firewall 2	10	10	0

5.2 Performance Evaluation

For performance evaluation, the addRole-EA was tested 20 times on each of the eight benchmark instances of the HP-Labs benchmark set. Each run was terminated if one of the two stopping conditions was met: either a total number of iterations $iterations\_total_{max}$ was reached or no improvement of the current best solution could be attained for a given number of iterations $iterations\_without\_improvement_{max}$.

The values of the relevant parameters were determined by parameter performance tests based on different parameter value sets. These tests were performed on one medium-sized instance of the HP-Labs benchmarks, namely *Firewall 1* as well as the largest instance *America Large*. As the derived parameters also showed good performance on the other benchmark instances they were adopted for the final test setup as shown in Table 4.

In Table 5 and Table 6 the different mutation and crossover methods are examined. For this, the number of cases in which the application of one mutation or crossover method on one individual led directly to an improvement of the fitness function is divided by the method's total application count. The positive rates in Table 5 show that all set operations performed by the different mutation methods lead to the creation of roles that contribute to the reduction of the total role number during the evolutionary process. Table 6 shows that the exchange of roles during crossover can further decrease the total number of roles.

Table 4: Test parameters.

Population size N_{pop}	20
Mutation Rate	1.0
Crossover Rate	0.1
Elitism Rate	0.7
$usersForIntersection_{max}$	5
$iterations\_total_{max}$	100,000
$iterations\_without\_improvement_{max}$	10,000

Table 5: Evaluation of mutation.

Benchmark	(M1)	(M2)	(M3)	(M4)	(M5)
America Large	1.81%	1.61%	2.18%	7.71%	4.34%
America Small	1.89%	3.71%	3.76%	7.21%	3.49%
APJ	1.05%	0.73%	1.76%	4.26%	1.15%
EMEA	7.84%	5.33%	1.85%	19.03%	4.15%
Healthcare	2.52%	1.78%	1.54%	2.97%	1.35%
Domino	1.36%	1.21%	1.58%	4.08%	1.91%
Firewall 1	2.70%	2.30%	3.34%	7.77%	3.20%
Firewall 2	1.12%	1.04%	0.77%	1.99%	0.85%

Table 6: Evaluation of crossover.

Benchmark	(C1)	(C2)	(C3)
America Large	7.22%	6.33%	6.26%
America Small	11.66%	10.20%	10.31%
APJ	5.46%	5.24%	5.30%
EMEA	8.07%	6.39%	6.31%
Healthcare	7.75%	7.36%	7.38%
Domino	5.34%	5.04%	5.15%
Firewall 1	8.25%	7.69%	7.54%
Firewall 2	4.94%	4.60%	4.76%

The results of the conducted tests, in terms of the resulting number of proposed roles, are displayed in Table 7. At this, $addRole_{best}$ denotes the best result attained in each instance, while $addRole_{avg.}$ denotes the average number of resulting roles across all test runs and $addRole_{SD}$ denotes the corresponding standard deviation. Furthermore, the results of the addRole-EA are compared to the results attained by different role mining algorithms found in literature (see Section 3). $\Delta_{avg.}$ denotes the deviation between the average results obtained from the applica-

Table 7: Evaluation of addRole-EA in comparison with other role-mining algorithms based on (Mitra et al., 2016).

Algorithm	America Large	America Small	APJ	EMEA	Healthcare	Domino	Firewall 1	Firewall 2
RM	422	206	455	**34**	**14**	**20**	65	**10**
EC	928	258	471	104	15	27	75	**10**
LP	**400**	193	454	**34**	**14**	**20**	66	**10**
HM	—	428	542	106	17	27	91	**10**
GO	—	225	475	**34**	16	**20**	71	**10**
GA_{Basic}	495	193	461	**34**	15	**20**	66	**10**
GA_{Edge}	907	275	479	115	16	28	79	**10**
MR_{Basic}	412	196	454	**34**	**14**	**20**	66	**10**
CM	—	2672	764	674	31	62	278	21
CRM	—	199	**453**	**34**	**14**	**20**	66	**10**
PC	—	1778	779	242	24	64	248	14
ORCA	—	1587	1164	3046	46	231	709	590
IT	—	—	—	**34**	16	**20**	71	**10**
INT	—	—	—	43	**14**	21	65	**10**
$addRole_{best}$	**400**	**184**	**453**	**34**	**14**	**20**	**64**	**10**
$addRole_{avg.}$	401.85	187.15	453.1	34	14	20	64.95	10
$addRole_{SD}$	0.93	1.71	0.30	0.00	0.00	0.00	0.22	0.00
$\Delta_{avg.}$	$+0.46\%$	-3.03%	$+0.02\%$	0.00%	0.00%	0.00%	-1.54%	0.00%

tion of the addRole-EA $addRole_{avg.}$ and the best solution found by the other role-mining algorithms. It is noticeable, that in each benchmark instance, the currently best solution is attained in at least one run. For *America Small* and *Firewall 1*, the best solution can even be improved by nine, respectively one role. The average values attained by the addRole-EA range from -3.03% to $+0.46\%$ of the best solution known in literature. This emphasizes the high performance of the addRole-EA.

6 CONCLUSION

In this paper, a new evolutionary algorithm for the Role Mining Problem was proposed. It is mainly based on a new method for adding roles to the chromosomes of an individual and the implicit deletion of roles that became obsolete due to the new role. This ensures the 0-consistency of the created RBAC scheme and the initial user permission assignment at all times of the algorithm. Furthermore, methods for the creation of new roles (mutation) and the exchange of existing roles between individuals (crossover) were presented. Evaluation shows, that the proposed algorithm can compete with current state-of-the-art algorithms in terms of the number of generated roles.

7 FUTURE WORKS

The current implementation of the addRole-EA contains many components which are based on random decisions. Role-creation methods during mutation or role-selection methods for role exchange during crossover are chosen randomly. Hence, the extension of the addRole-EA by more specific operators like the consideration of similarity measures between users or roles or the application of prioritization functions for more focused addition of roles are subject to further investigation. The analysis of the HP-Labs benchmark set shows, that the contained benchmark instances leave little space for detailed evaluation of role-mining algorithms. Thus, new standardized benchmarks have to be developed.

Choosing an evolutionary approach for solving the Role Mining Problem has many advantages. Its scalability allows for easy application in enterprise-driven use cases. In this context, further requirements, like the consideration of role hierarchies, compliance constraints or the semantic quality of roles, gain focus. Thus, these business aspects have to be included in a future version of the addRole-EA. In addition, academic benchmarks covering those features have to be derived. Furthermore, as evolutionary algorithms are able to handle dynamic events, it is desirable to adapt our approach to the dynamic surroundings of enterprises by the development of real-time-event-handling methods for common events like permission-requests, employees joining or leaving the company or employees changing departments within one enterprise.

REFERENCES

Blundo, C. and Cimato, S. (2010). A simple role mining algorithm. In Shin, S. Y., Ossowski, S., Schumacher, M., Palakal, M. J., and Hung, C.-C., editors, *Proceedings of the 2010 ACM Symposium on Applied Computing - SAC '10*, pages 1958–1962, New York, New York, USA. ACM Press.

Colantonio, A., Di Pietro, R., and Ocello, A. (2008). A cost-driven approach to role engineering. In Wainwright, R. L. and Haddad, H. M., editors, *Proceedings of the 2008 ACM symposium on Applied computing - SAC '08*, pages 2129–2136, New York, New York, USA. ACM Press.

Dong, L., Wu, J., Gong, C., and Pi, B. (2014). A network-cliques based role mining model. *Journal of Networks*, 9(8):2079–2088.

Dong, L. J., Wang, M. C., and Kang, X. J. (2011). Mining least privilege roles by genetic algorithm. *Applied Mechanics and Materials*, 121-126:4508–4512.

Du, X. and Chang, X. (2014). Performance of ai algorithms for mining meaningful roles. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 2070–2076. IEEE.

Ene, A., Horne, W., Milosavljevic, N., Rao, P., Schreiber, R., and Tarjan, R. E. (2008). Fast exact and heuristic methods for role minimization problems. In Ray, I. and Li, N., editors, *Proceedings of the 13th ACM symposium on Access control models and technologies - SACMAT '08*, pages 1–10, New York, New York, USA. ACM Press.

Hu, N., Bradford, P. G., and Liu, J. (2006). Applying role based access control and genetic algorithms to insider threat detection. In Menezes, R., editor, *Proceedings of the 44th annual southeast regional conference on - ACM-SE 44*, pages 790–795, New York, New York, USA. ACM Press.

Huang, H., Shang, F., Liu, J., and Du, H. (2015). Handling least privilege problem and role mining in rbac. *Journal of Combinatorial Optimization*, 30(1):63–86.

Kotenko, I. and Saenko, I. (2015). Improved genetic algorithms for solving the optimisation tasks for design of access control schemes in computer networks. *International Journal of Bio-Inspired Computation*, 7(2):98–110.

Kuhlmann, M., Shohat, D., and Schimpf, G. (2003). Role mining - revealing business roles for security administration using data mining technology. In Ferrari, E. and Ferraiolo, D., editors, *Proceedings of the eighth ACM symposium on Access control models and technologies - SACMAT '03*, pages 179–186, New York, New York, USA. ACM Press.

Kumar, R., Sural, S., and Gupta, A. (2010). Mining rbac roles under cardinality constraint. In Jha, S. and Mathuria, A., editors, *Information Systems Security*, volume 6503 of *Lecture Notes in Computer Science*, pages 171–185. Springer Berlin Heidelberg, Berlin, Heidelberg.

Lu, H., Vaidya, J., and Atluri, V. (2008). Optimal boolean matrix decomposition: Application to role engineering. In *2008 IEEE 24th International Conference on Data Engineering*, pages 297–306. IEEE.

Lu, H., Vaidya, J., and Atluri, V. (2014). An optimization framework for role mining. *Journal of Computer Security*, 22(1):1–31.

Mitra, B., Sural, S., Vaidya, J., and Atluri, V. (2016). A survey of role mining. *ACM Computing Surveys*, 48(4):1–37.

Molloy, I., Chen, H., Li, T., Wang, Q., Li, N., Bertino, E., Calo, S., and Lobo, J. (2008). Mining roles with semantic meanings. In Ray, I. and Li, N., editors, *Proceedings of the 13th ACM symposium on Access control models and technologies - SACMAT '08*, pages 21–30, New York, New York, USA. ACM Press.

Molloy, I., Li, N., Li, T., Mao, Z., Wang, Q., and Lobo, J. (2009). Evaluating role mining algorithms. In Carminati, B. and Joshi, J., editors, *Proceedings of the 14th ACM symposium on Access control models and technologies - SACMAT '09*, pages 95–104, New York, New York, USA. ACM Press.

Saenko, I. and Kotenko, I. (2011). Genetic algorithms for role mining problem. In *2011 19th International Euromicro Conference on Parallel, Distributed and Network-Based Processing*, pages 646–650. IEEE.

Saenko, I. and Kotenko, I. (2012). Design and performance evaluation of improved genetic algorithm for role mining problem. In *2012 20th Euromicro International Conference on Parallel, Distributed and Network-based Processing*, pages 269–274. IEEE.

Saenko, I. and Kotenko, I. (2016). Using genetic algorithms for design and reconfiguration of rbac schemes. In Unknown, editor, *Proceedings of the 1st International Workshop on AI for Privacy and Security - PrAISe '16*, pages 1–9, New York, New York, USA. ACM Press.

Saenko, I. and Kotenko, I. (2017a). Administrating role-based access control by genetic algorithms. In Unknown, editor, *Proceedings of the Genetic and Evolutionary Computation Conference Companion on - GECCO '17*, pages 1463–1470, New York, New York, USA. ACM Press.

Saenko, I. and Kotenko, I. (2017b). Reconfiguration of rbac schemes by genetic algorithms. In Badica, C., El Fallah Seghrouchni, A., Beynier, A., Camacho, D., Herpson, C., Hindriks, K., and Novais, P., editors, *Intelligent Distributed Computing X*, volume 678 of *Studies in Computational Intelligence*, pages 89–98. Springer International Publishing, Cham.

Saenko, I. and Kotenko, I. (2018). Genetic algorithms for solving problems of access control design and reconfiguration in computer networks. *ACM Transactions on Internet Technology*, 18(3):1–21.

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

Schlegelmilch, J. and Steffens, U. (2005). Role mining with orca. In Ferrari, E. and Ahn, G.-J., editors, *Proceedings of the tenth ACM symposium on Access control models and technologies - SACMAT '05*, pages 168–176, New York, New York, USA. ACM Press.

Suganthy, A. and Chithralekha, T. (2018). Role-evolution in role-based access control system. *International Journal of Emerging Research in Management and Technology*, 6(7):223–227.

Vaidya, J., Atluri, V., and Guo, Q. (2007). The role mining problem. In Lotz, V. and Thuraisingham, B., editors, *Proceedings of the 12th ACM symposium on Access control models and technologies - SACMAT '07*, pages 175–184, New York, New York, USA. ACM Press.

Vaidya, J., Atluri, V., Warner, J., and Guo, Q. (2010). Role engineering via prioritized subset enumeration. *IEEE Transactions on Dependable and Secure Computing*, 7(3):300–314.

Verizon (2019). Data breach investigations report 2019. *Computer Fraud & Security*, 2019(6):4.

Xu, Z. and Stoller, S. D. (2012). Algorithms for mining meaningful roles. In Atluri, V., Vaidya, J., Kern, A., and Kantarcioglu, M., editors, *Proceedings of the 17th ACM symposium on Access Control Models and Technologies - SACMAT '12*, pages 57–66, New York, New York, USA. ACM Press.

Zhang, D., Ramamohanarao, K., and Ebringer, T. (2007). Role engineering using graph optimisation. In Lotz, V. and Thuraisingham, B., editors, *Proceedings of the 12th ACM symposium on Access control models and technologies - SACMAT '07*, pages 139–144, New York, New York, USA. ACM Press.

Zhang, D., Ramamohanarao, K., Versteeg, S., and Zhang, R. (2010). Graph based strategies to role engineering. In Sheldon, F. T., Prowell, S., Abercrombie, R. K., and Krings, A., editors, *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research - CSIIRW '10*, pages 1–4, New York, New York, USA. ACM Press.

Zhang, W., Chen, Y., Gunter, C., Liebovitz, D., and Malin, B. (2013). Evolving role definitions through permission invocation patterns. In Conti, M., Vaidya, J., and Schaad, A., editors, *Proceedings of the 18th ACM symposium on Access control models and technologies - SACMAT '13*, pages 37–48, New York, New York, USA. ACM Press.