CERIAS Tech Report 2013-2
Role Mining on Relational Data
by Zahid Pervaiz, Arif Ghafoor, Walid G. Aref
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

Role Mining on Relational Data

Zahid Pervaiz
School of Electrical and Computer Engineering
465 Northwestern Ave
West Lafayette, IN, USA
zpervaiz@purdue.edu

Arif Ghafoor
School of Electrical and Computer Engineering
465 Northwestern Ave
West Lafayette, IN, USA
ghafoor@purdue.edu

Walid G. Aref
Department of Computer Science
305 N. University Street
West Lafayette, IN, USA
aref@cs.purdue.edu

ABSTRACT

Fine-grained access control for relational data defines user authorizations at the tuple level. Role Based Access Control (RBAC) has been proposed for relational data where roles are allowed access to tuples based on the authorized view defined by a selection predicate. During the last few years, extensive research has been conducted in the area of role engineering. The existing approaches for role engineering are top-down (using domain experts), bottom-up (role-mining), or a hybrid of both. However, no research has been conducted for role engineering in relational data. In this paper, we address this problem. The challenge is to extract an RBAC policy with authorized selection predicates for users given an existing tuple-level fine-grained access control policy. We formulate the problem for relational data, propose a role mining algorithm and conduct experimental evaluation. Experiments demonstrate that the proposed algorithm can achieve up to 400% improvement in performance for relational data as compared to existing role mining techniques.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection—*Access Controls*; H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Security, Management

Keywords

RBAC, role mining, relational data

1. INTRODUCTION

Role Based Access Control (RBAC) has emerged as a standard for enterprise resource management during the last decade. Creation of an RBAC configuration and its maintenance is a major hindrance towards widespread adoption [15].

Role-based access control has been proposed for relational data that allows roles the access to an authorized view defined by a selection predicate [27, 18, 3, 41]. Fine-grained access control for relational data allows defining tuple-level permissions for users, e.g., as in Label Based Access Control (LBAC) in IBM DB2 [42] and Virtual Private Database (VPD) in Oracle [4]. The benefit of employing RBAC is that the policy management cost is reduced in comparison to tuple-level permissions. To gain the benefits of both worlds, i.e., the fine-grained access control of tuple-level permissions and the cost reduction of role-based access control, relational role-mining is proposed that extracts new roles for an RBAC policy from the given tuple-level permissions.

Role engineering is the process to create roles for users and assign permissions to roles [7]. Role-mining is the bottom-up approach for role engineering and uses existing user-to-permission assignments to extract an RBAC policy. In the top-down approach, domain experts manually configure the policy. Both bottom-up and top-down approaches have their disadvantages. The bottom-up approach requires existing policy information and the mined roles lack semantic meaning. In contrast, the top-down approach is labor intensive and is dependent on domain experts [2]. A lot of research has been conducted in the area of role mining during the last few years [14]. However, the problem of role mining for relational data has not been explored. The contributions of this paper can be summarized as follows.

1. We define the problem of Predicate Role Mining (PRM) in the context of relational data.
2. We propose an algorithm for the approximate solution of PRM.
3. We conduct experimental evaluation of the proposed algorithm.

The rest of this paper proceeds as follows. Section 2 discusses related work. Section 3 presents the necessary background material. Section 4 formulates the problem of role mining. Section 5 introduces the new role mining algorithm. Section 6 presents the experimental results, and Section 7 concludes the paper.

2. RELATED WORK

In the top-down approach, domain experts define roles by analyzing the application's business processes and then assign permissions to derived roles based on job functions [1]. Roekle et al. propose a process-oriented approach for role

engineering [28]. Strembek and Neuman et al. introduce scenario-driven role engineering based on requirements engineering techniques. They also define functional and organizational roles [26, 32]. Functional roles are related to business functions while organizational roles are related to the organizational hierarchy. Shin et al. propose a hybrid approach using an information model with top-down and bottom-up information flows [30]. All the top-down approaches are performed manually and need domain experts.

In the recent years, the focus of research in role engineering has been in role mining using permission clustering and frequent permission sets. The Role Mining Problem (RMP) has been formally defined by Vaidya et al. and has been shown to be NP-complete [35, 36]. Vaidya et al. propose role mining algorithms using subset enumeration [37, 38]. Frank et al. view role mining as a prediction problem and define its provisioning, security, and maintainability requirements [11]. Kuhlman et al. use association rule mining and clustering to derive roles [19]. Zhang et al. perform graph optimization on user-to-permission assignments to discover an RBAC state [40]. Single linkage hierarchical clustering on permissions is utilized by Schlegelmilch and Steffens in the role mining tool ORCA [29]. Molloy et al. have proposed a hierarchical miner using formal concept analysis and an attribute miner using user semantics [22]. Frank et al. have proposed a probabilistic model for role mining that allows for the detection of wrong or missing assignments [10]. One of the issues with role mining is the presence of noise in the shape of exceptions and over- and under-assignments in user to permission assignment data. Molloy et al. use rank reduced matrix factorization to detect noise and have shown improved accuracy by mining noise-less data [25]. The attribute relevance for role mining has also been discussed by Frank et al. in their probabilistic approach using entropy-based relevance [12]. The entropy-based relevance requires the availability of training data. A weighted role mining algorithm has been proposed by Ma et al. by assigning weights to permissions [21]. Takabi et al. have proposed StateMiner that tries to find the optimal RBAC state similar to an existing RBAC state [33]. Uzun et al. extend the role mining problem to consider the separation of duty and exceptions [34]. They use Negatives Authorizations (NA) to reduce the policy cost that are also being used in our formulation. Most of the existing role mining schemes are not scalable and might not be usable for relational data. Verde et al. propose scalable techniques to reduce the computational workload and parallelize the role mining effort [39]. Colantonio et al. propose Visual Role Mining that is a graphical technique to enable quick analysis and elicitation of meaningful roles [6].

Molloy et al. have evaluated the performance of some of the approaches discussed above on different datasets [24]. The Hierarchical Miner proposed by Molloy et al. was found to give the best results [22]. However, to the best of our knowledge, role mining for relational data has not been investigated before. The general role mining techniques can be used to extract roles for relational data but the extracted roles will lack any semantic meaning [2]. From the perspective of relational data, the focus is to extract selection-predicate-based roles.

3. BACKGROUND

This section introduces fine-grained access control for relational data, role-based access control, and the role mining problem.

3.1 Fine-grained Access Control

Fine-grained access control, e.g., LBAC in IBM DB2 [42] and VPD in Oracle [4] allows to define tuple-level permissions for database users. An access control policy for a table of 12 tuples and 6 users is given in Figure 1. In this table, age and zip are the tuple attributes and the column user contains the users that are authorized to access the tuple. Given m users and n tuples, if the tuples are directly assigned to users and the number of the tuples assigned to each user is large, then user-to-permission assignments on the order of mn need to be maintained. The number of assignments required for the access control policy in Figure 1 is 36. For evaluating user queries, we assume a Truman model [27]. In this model, a user query is modified by the access control mechanism and only the authorized tuples are returned. Assume that a user, say U1, executes a range query for the tuples with age 0-20, and zip 0-30. Although 4 tuples A, B, C, and H satisfy the query predicate, only A, B, and C are returned to the user.

ID	Age	Zip	User
A	5	10	U1, U2, U5
B	15	25	U1, U2, U5
C	15	15	U1, U2, U5
D	30	30	U1, U2, U5
E	30	15	U1, U2, U6
F	40	5	U1, U2, U6
G	5	35	U3, U4, U6
H	10	20	U3, U4, U5
I	15	40	U3, U4, U6
J	30	40	U3, U4, U5
K	40	35	U3, U4, U5
L	35	25	U3, U4, U5

Figure 1: Tuple-level access control policy

3.2 Role-based Access Control

Role-based Access Control (RBAC) allows users access to resources based on the role assumed by the user in an organization. The benefit of RBAC is that the policy management is simplified in comparison to an access control policy with direct user-to-permission assignment. A core RBAC policy configuration is composed of a set of Users (U), a set of Roles (R), a set of Operations (Ops), a set of Objects (O), and a set of Permissions (P). UA is a user-to-role ($U \times R$) assignment relation, permissions are an operations-to-objects ($Ops \times O$) assignment relation, and PA is a role-to-permission ($R \times P$) assignment relation. A role hierarchy (RH) defines an inheritance relationship between roles and is a partial order on roles ($R \times R$) [8]. The benefit of RBAC is that the policy management cost is reduced, i.e, given m users and n tuples, an RBAC configuration requires the definition of policy assignments on the order of m+n as compared to mn for the case of direct assignment.

3.2.1 The Role Mining Problem

The extraction of roles by role mining requires that an existing access control policy with user-to-permission assignments is available. This problem has been defined by Molloy et al. [22] as follows.

Definition 1. (Role Mining Problem(RMP)). Given an access control policy $\rho = \langle U, P, UP \rangle$, where UP is a user-to-permission assignment relation. The role mining problem is to find an RBAC state $\langle U, R, UA, PA, RH, DUPA \rangle$ that is consistent with ρ. $DUPA$ is the direct user-to-permission assignment relation. The RBAC state is consistent with ρ if every user in the RBAC state has the same set of permissions as those of ρ.

3.2.2 Weighted Structural Complexity (WSC)

Given an access control policy ρ, many RBAC states can be consistent with ρ. WSC defines a cost measure for creating an RBAC state and can be used to select an RBAC state with minimal cost. WSC has been defined by Molloy et al. [22] as follows.

Definition 2. (Weighted Structural Complexity). Given a weight scheme $W = \langle w_r, w_u, w_p, w_h, w_d \rangle$, where w_r, w_u, w_p, w_h, $w_d \in \mathbb{Q}^+ \cup \{\infty\}$, the Weighted Structural Complexity (WSC), denoted by $wsc(\gamma, W)$, of an RBAC state γ is computed as follows.

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

where $t_reduce(RH)$ is the transitive reduction of a role hierarchy.

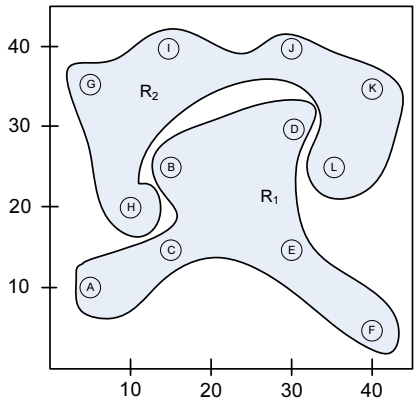


Figure 2: Role mining on relational data

Example 1. The two-dimensional representation of the tuples in Figure 1 is given in Figure 2 after role mining. R_1 and R_2 are the two roles that were extracted after role mining ($W : w_r = w_u = w_p = w_h = w_d = 1$) on user-to-permission assignments given on the user column. The role R_1 is assigned to users U_1 and U_2, while role R_2 is assigned to users U_3 and U_4. Users U_5 and U_6 are directly assigned to the authorized tuples. The policy management cost with direct user-to-permission assignment was 36 while with the RBAC policy the cost has been reduced to $|R| + |UA| + |PA| + |DUPA|$ (2+4+12+12) = 30.

3.3 RBAC for Relational Data

For the relational model, we assume that the conjunction of the selection predicates on the attributes of relational schema defines a role [5]. We denote the role for relational data as a Predicate Role (PR). An RBAC configuration for relational data is composed of a set of Users (U) and a set of Predicate Roles (PR). RA is a user-to-predicate role ($U \times PR$) assignment relation. Each predicate role defines a hyper-rectangle in the tuple space and all the tuples enclosed by this hyper-rectangle are authorized to the user assigned to the predicate role. In practice, when a user executes any query, the tuples satisfying the conjunction of the query predicate and the authorized predicate (defined by PR) are returned [27, 3, 41]. For predicate roles, the tuple membership is implicitly defined by the selection predicates. So, the role-to-tuple assignment and the role hierarchy are not required to be maintained.

4. PREDICATE ROLE MINING (PRM)

In this section, the predicate role mining problem and the weighted structural complexity for relational data have been defined. Let t_i be a tuple in Table T with d attributes $A_1, A_2, \ldots, A_d$. Tuple t_i can be expressed as a d-dimensional vector $\{v_1^{t_i}, \ldots, v_d^{t_i}\}$, where v_i is the value of the i^{th} attribute. Let D_{A_i} be the domain of the attribute A_i, then $t_i \in D_{A_1} \times \ldots \times D_{A_d}$. A predicate role PR in the attribute domain space can be defined as a d-dimensional hyper-rectangle.

4.1 The Predicate Role Mining Problem

Definition 3. (Predicate Role Mining Problem(PRMP)). Given an access control policy $\rho_r = \langle U, T, UT \rangle$ on relational data, where UT is a user-to-tuple assignment relation. The predicate role mining problem is to find an RBAC state for the relational data $\langle U, PR, RA, DT, NA \rangle$ that is consistent with ρ_r, where DT is the direct user-to-tuple assignment relation, and NA is a direct user-to-tuple assignment relation that defines a negative authorization.

Conflict resolution between positive and negative authorizations is done by applying the *denials-take-precedence* policy under which the NA assignment overrides the authorization of a predicate role, i.e., even if access to a tuple is authorized by a predicate role, if NA exists for that a user and a tuple, then the access is denied. The relational RBAC state is consistent with ρ_r, if every user in the RBAC state has the same set of permissions as those in ρ_r. The formation of predicate roles can result in access to unauthorized tuples to users and the definition of NA allows to ensure the consistency of relational RBAC state.

Definition 4. (Relational Weighted Structural Complexity). Given a weight scheme $W_r = \langle w_{pr}, w_{ra}, , w_{dt}, w_{na} \rangle$, where w_{pr}, w_{ra}, w_{dt}, $w_{na} \in \mathbb{Q}^+ \cup \{\infty\}$, the Weighted Structural Complexity(WSC), denoted by $wsc(\gamma_r, W_r)$, of a relational RBAC state γ_r is computed as follows.

$wsc(\gamma_r, W_r) = w_{pr} * |PR| + w_{ra} * |RA| + w_{dt} * |DT| + w_{na} * |NA|$

Example 2. The two-dimensional representation of the tuples in Figure 1 is given in Figure 3 after predicate role mining. PR_1, PR_2, PR_3, PR_4 are four predicate roles that were extracted using PRM ($W_r : w_{pr} = w_{ra} = w_{dt} = w_{na} = 1$) on the user-to-tuple assignments. The roles PR_1 and PR_4 are assigned to users U_1 and U_2 while PR_2 and PR_3 are assigned to users U_3 and U_4. Predicate roles PR_1 and PR_3 are assigned to user $U5$ and roles PR_2 and PR_4 are assigned to user $U6$. The policy management cost with direct

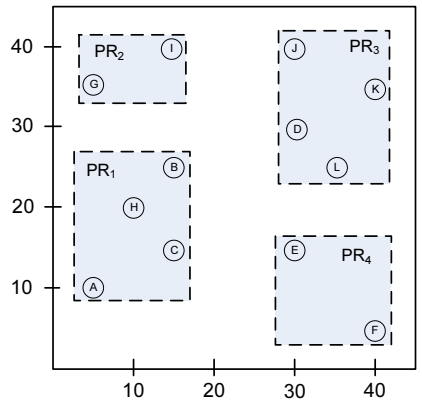


Figure 3: Predicate role mining on relational data

user-to-tuple assignment is 36 while with that of the RBAC state after role mining it is 30. The RBAC policy cost after PRM has been reduced to $|PR| + |RA| + |DT| + |NA|$ (4+12+4+4) = 24.

4.2 Predicate role cost definitions

Definition 5. (User Role Cost) The user role cost is the minimum number of assignments required for a User U_j for tuples inside a predicate role PR_i to ensure a consistent γ_r. Formally,

$$C^{U_j}_{PR_i} = \begin{cases} |DT^{U_j}_{PR_i}| + RA^{U_j}_{PR_i}, & \text{if } |DT^{U_j}_{PR_i}| \leq |NA^{U_j}_{PR_i}| + 1 \\ |NA^{U_j}_{PR_i}| + RA^{U_j}_{PR_i}, & \text{otherwise} \end{cases} \quad (1)$$

where $RA^{U_j}_{PR_i} = 1$, if $|DT^{U_j}_{PR_i}| > |NA^{U_j}_{PR_i}|$ otherwise 0.

The intuition for the user assignment in Equation 1 is that the role is assigned to a user only if the negative authorization cost is less than the direct user-to-tuple assignment cost.

Definition 6. (Predicate Role Cost (PRC)) The relational RBAC policy cost for a predicate role PR_i is the minimum number of assignments required for all users authorized to access tuples inside PR_i for a consistent γ_r and is denoted by PRC_{PR_i}

$$PRC_{PR_i} = RC_{P_i} + \sum_{j=1}^{|U|} C^{U_j}_{PR_i} \quad (2)$$

where $RC_{P_i} = 1$ if $\sum_{j=1}^{|U|} RA^{U_j}_{PR_i} > 0$ otherwise 0.

$RC_{P_i} = 1$, means that the predicate role is only created when at least one user is assigned to the predicate role. The sum of cost for all predicate roles is equal to the $wsc(\gamma_r, W_r)$ as given in Definition 4, i.e.,

$$wsc(\gamma_r, W_r) = \sum_{i=1}^{|PR|} PRC_{PR_i} \quad (3)$$

Example 3. Consider the predicate role PR_1 in Figure 3. The user role cost for user $U1$ is $C^{U_1}_{PR_1} = 2$ ($DT^{U_1}_{PR_1} = 4$ and $NA^{U_1}_{PR_1} = 2$). The predicate role cost for PR_1 is (sum of user role costs for users $U1 - U6$ plus 1 for role creation cost = 2+2+1+1+1+0+1) is 8. The total cost for the sum of all predicate roles $PR_1 - PR_4$(8+4+8+4) is 24, which is the $wsc(\gamma_r, W_r)$ computed in Example 2.

5. PRM ALGORITHM

In this section, a new algorithm for predicate role mining is presented. The general problem of role mining has been shown to be NP-complete [36, 23]. The predicate role mining problem is a space-partitioning problem and similar problems for multidimensional partitioning have been shown to be NP-hard [20]. We provide a greedy algorithm that tries to minimize the cost of a relational RBAC policy.

The new Predicate Role Mining (PRM) algorithm is similar to the kd-tree construction [13]. PRM starts with the whole tuple space as one predicate role and then roles are recursively split into two halves if the sum of predicate role cost by creating new roles does not increase as compared to the existing predicate role. To split a predicate role, two decisions need to be made; i) Choosing a split value along each dimension, and ii) Choosing a dimension along which to split. The split value is chosen along the median in each dimension and then the dimension having the minimum predicate role cost $PRC_{CR^1_i} + PRC_{CR^2_i}$ is selected.

The PRM algorithm is listed in Algorithm 1. In Line 1, the whole tuple space is added to the set of candidate roles. The **for** loop in Line 2 selects a predicate role and then the role is split along the median. The dimension having the lowest role cost is selected to split the candidate role in Lines 3-4. If after the split the role cost increases, then we add the predicate role to the RBAC state in Lines 8-9. Otherwise, the new roles are added to candidate role set in Line 6.

Algorithm 1: PRM

```
   Input  : T, ρ_r, W_r
   Output: γ_r
1  Initialize Set of Candidate Roles(CR ← T)
2  for (CR_i ∈ CR) do
3      Split CR_i into CR_i^1 and CR_i^2 along median in each
       dimension
4      Select dimension with least PRC_{CR_i^1} + PRC_{CR_i^2}
5      if (PRC_{CR_i^1} + PRC_{CR_i^2} ≤ PRC_{CR_i}) then
6          Create new predicate roles and add to CR
7      else
8          Add predicate role to PR
9          Update γ_r
10 return (γ_r)
```

LEMMA 1. *The time complexity of PRM is $\mathcal{O}(dmnlgn)$.*

PROOF. Assume that there are m users and n tuples in a relational table. The time complexity is derived by multiplying the height of the kd-tree with the work performed at each level. The height of the kd-tree for the median split is lgn. At each level, the policy cost for each user is calculated over all tuples, which leads to dmn, where d is the

number of attributes in relational data. Then, the total time complexity is $\mathcal{O}(dmnlgn)$. □

The PRM algorithm generates a hierarchical kd-tree along with the predicate roles as the leaf nodes of the tree. There are two approaches to define hierarchical relationships among predicate roles:

1. An intermediate node's hyper-rectangle might enclose smaller leaf-node's hype-rectangles for predicate roles assigned to any user. If the assignment of user to larger hyper-rectangle reduces the policy cost then we remove the user-to-role assignments for the smaller roles and update DT and NA accordingly.

2. Cluster the predicate roles to generate role hierarchy. However, with this approach, assignments for relationships between roles will need to be maintained.

However, in our experiments in Section 6 we observed only minor cost improvement after replacing smaller roles with larger predicate roles.

6. EXPERIMENTAL EVALUATION

The experiments are carried out on three datasets for the empirical evaluation of the proposed algorithm for predicate role mining. The first dataset, Normal, is generated synthetically from the normal distribution for the purpose of the visual representation of the generated predicate roles. 2000 tuples with two attributes are randomly selected (from a Normal distribution with $\mu = 50$, $\sigma = 10$, and cardinality = 100). The second dataset is the Adult dataset from the Machine Learning Repository at the University of California, Irvine [9]. The Adult dataset has 45222 tuples with the following schema: Age, Work class, Education, Marital status, Occupation, Relationship, Race, and Gender. The third dataset is the Census dataset [31] from IPUMS. This dataset is extracted for Year 2001 using attributes: Age, Gender, Marital status, Race, Birth place, Language, Occupation, and Income. The size of the Census dataset is about 1.2 million tuples.

6.1 Random Policy Generator

The basic assumption for predicate role mining is that the tuple-level permissions for users have a semantic relationship. If a user is working on Project 1, then the user will have access to most of the tuples related to Project 1. In order to generate tuple-level permissions for the three datasets, two tuples are randomly selected from the tuple space and a bounding box of these two tuples is formed [17]. For the Normal dataset, we select three such bounding boxes (with cardinality BC between $200 \leq BC \leq 500$) for each user. Then, we assign all the unique tuples inside the bounding boxes to the user as a tuple-level permission. The three randomly selected bounding boxes for the first four users are given in Figure 4. For the Adult and Census datasets, we use five bounding boxes ($500 \leq BC \leq 2000$) and ten bounding boxes ($2000 \leq BC \leq 10,000$), respectively. The bounding boxes are used to assign user-to-tuple level permissions. The summary of the direct assignment policy generation for the datasets is given in Table 1.

6.2 Results

Molloy et al. evaluated the existing role mining algorithms and found that the Hierarchical Miner (HM) [22] gives the lowest RBAC policy cost [24]. The HM algorithm uses a concept lattice that is reduced and pruned to generate a role hierarchy. For the generation of a concept lattice, we use Colibri-Java, an implementation of the Formal Concept Analysis in Java [16]. It has built-in functions to derive the reduced concept lattice but the state space for the concept lattice grows exponentially as the number of users and permissions increase. Instead of pruning the reduced lattice, we just estimate the cost of HM from the reduced lattice by using the pruning rules for HM. The pruning rules are applied to estimate the policy cost as follows:

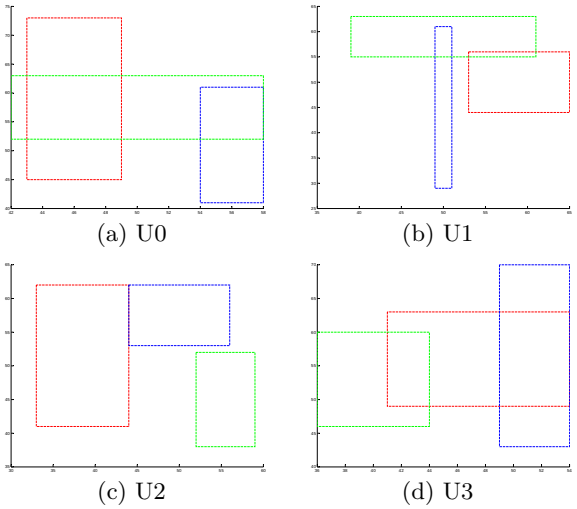

(a) U0 (b) U1 (c) U2 (d) U3

Figure 4: Assignment of random permissions to users

Table 1: Dataset and assigned permissions

	Users	Tuples	UT	Density
Normal	20	2000	15513	0.387
Adult	300	45222	1451715	0.107
Census	500	1192206	21654616	0.036

1. Roles are created for nodes in the reduced lattice having both new users and new permissions and UA and PA costs are evaluated for each node. We assume an RH cost of one for each role.

2. A node with only roles will be mapped to existing roles according to the Lower Neighbors (LN). The UA cost (UA*LN) for each node is evaluated.

3. A node with only permissions will also be mapped to existing roles according to the Upper Neighbors. The PA cost (PA*UN) for each node is evaluated.

4. The nodes with no new roles and permissions are ignored for $W : w_r = w_u = w_p = w_h = w_d = 1$.

We iterate over all nodes and sum the total cost according to the applicable pruning rule. We compare our results of the PRM algorithm with that of estimated HM cost for the first dataset only. The weight scheme used is $W_r : w_{pr} = w_{ra} = w_{dt} = w_{na} = 1$ for PRM and $W : w_r = w_u = w_p =$

$w_h = w_d = 1$ for HM. The results for predicate role mining for the Normal dataset are given in Table 2. The first row gives the cost for the Direct Assignment Policy (DAP), the second row gives the RBAC cost of policy generated by the hierarchical miner, and the third row shows the cost of relational RBAC policy generated by the predicate role mining algorithm. The PRM algorithm generates 160 predicate roles and there are 1108 assignment relations for 20 users. The total cost for the relational RBAC policy is 2927 that is about five times less than the direct assignment cost and three time less than the estimated HM cost. The visual representation of the generated predicate roles is given in Figure 5. In this figure, the blue rectangles (darker) are predicate roles generated by the PRM algorithm while the red rectangles are the candidate roles that are created but are not assigned to any user due to the high NA (negative authorization) cost. The policy cost reported for PRM for all the three datasets is without the hierarchy step.

Table 2: Results for the Normal dataset

	|PR|	|RA|	|PA|	|DT|	|NA|	Total
DAP	0	0	0	15513	0	15513
HM	11	183	9480	0	0	9685
PRM	160	1108	0	940	719	2927

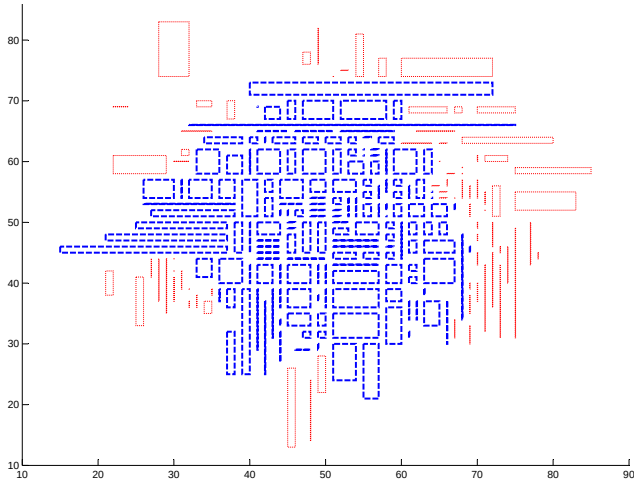
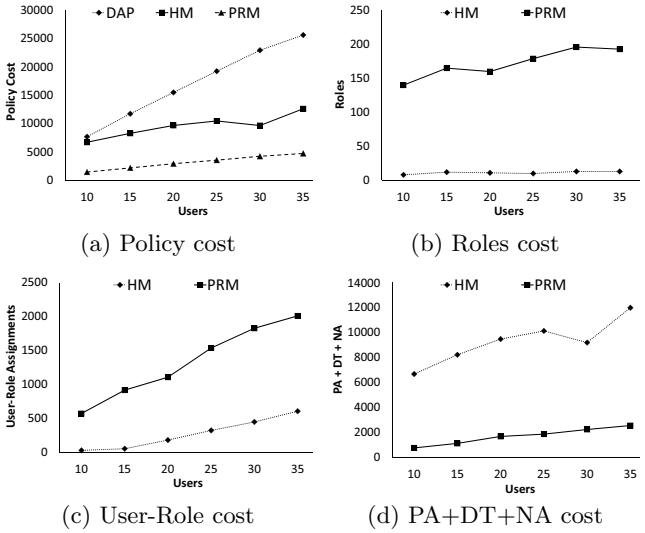


Figure 5: Visual representation of the extracted predicate roles

In the next experiment for the Normal dataset, we vary the number of users and generate random user-to-tuple assignments for 10, 15, 20, 25, 30, and 35 users. The policy cost for the generated policies by HM and PRM is given in Figure 6. Observe in Figure 6(a) that PRM gives the lowest policy cost as compared to DAP and HM. The cost comparison for the creation of roles, user-to-role assignments, and role-to-permission assignments between HM and PRM is given in Figure 6(b), Figure 6(c), and Figure 6(d), repectively. The HM forms an RBAC policy with a fewer number of roles in Figure 6(b) as there are no spatial constraints for the formation of roles. However, this requires a high cost to maintain the role-to-permission assignments as given in



(a) Policy cost (b) Roles cost

(c) User-Role cost (d) PA+DT+NA cost

Figure 6: Varying the number of users for the Normal dataset

Figure 6(d). On the contrary, more predicate roles are made by PRM with a lower DT and NA cost as compared to the PA cost of HM.

The results for the Adult dataset are given in Table 3. The PRM algorithm generates 7994 predicate roles for 300 users. The relational RBAC cost for the policy generated by PRM is 376545 that is about four times less than the direct assignment cost.

The predicate role mining results for the Census dataset are given in Table 4. In this case also, PRM reduces the policy cost by a factor of about three as compared to the direct assignment cost.

The cost savings after hierarchy step is given in Table 5 for all three datasets.

7. CONCLUSIONS

Lot of research has been conducted in the area of role mining during the last few years. However, the problem of role engineering for relational data has not been investigated. We formulate the problem of predicate role mining for relational data and define the relational weighted structural complexity for a relational RBAC policy. To extract predicate roles from existing user-to-tuple permission assignments, we propose a predicate role mining algorithm and conduct experimental evaluation. Experiments demonstrate that the proposed algorithm can achieve up to 400% improvement in performance. In future work, we plan to work on hardness results of the predicate role mining problem and performance guarantees for the PRM algorithm.

8. COMMENTS

Study following changes

1. Implement another RM algorithm, Zhang or any other
2. Apply role mining to cluster PRM and check cost saving
3. Investigate hybrid PRM and compare with Comment 2

Table 3: Results for the Adult dataset

	|PR|	|RA|	|PA|	|DT|	|NA|	Total
DAP	0	0	0	1451715	0	1451715
PRM	7994	173390	0	122541	72620	376545

Table 4: Results for the Census dataset

	|PR|	|RA|	|PA|	|DT|	|NA|	Total
DAP	0	0	0	21654616	0	21654616
PRM	182942	2843769	0	2703658	1440781	7171150

Table 5: Cost saving after hierarchy step

	Cost	Cost Saving	% Improvement
Normal	2927	317	10.83
Adult	376545	25761	6.84
Census	7171150	486986	6.79

4. Generate Hierarchical by bottom up approach too
5. Add bottom up approach too with benefits (doesnt require to store complete dataset)

9. REFERENCES

[1] V. Atluri. Panel on role engineering. In *Proceedings of the 13th ACM symposium on Access control models and technologies*, pages 61–62. ACM, 2008.

[2] E. Bertino, S. Calo, H. Chen, N. Li, T. Li, J. Lobo, I. Molloy, and Q. Wang. Some Usability Considerations in Access Control Systems. In *Proc. Symposium On Usable Privacy and Security (SOUPS)*, 2008.

[3] E. Bertino and R. Sandhu. Database security-concepts, approaches, and challenges. *Dependable and Secure Computing, IEEE Transactions on*, 2(1):2–19, 2005.

[4] K. Browder and M. Davidson. The virtual private database in oracle9ir2. *Oracle Technical White Paper, Oracle Corporation*, 500, 2002.

[5] S. Chaudhuri, T. Dutta, and S. Sudarshan. Fine grained authorization through predicated grants. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 1174–1183. IEEE, 2007.

[6] A. Colantonio, R. Di Pietro, A. Ocello, and N. Verde. Visual role mining: A picture is worth a thousand roles. *Knowledge and Data Engineering, IEEE Transactions on*, 24(6):1120–1133, 2012.

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

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

[9] A. Frank and A. Asuncion. UCI Machine Learning Repository, 2010.

[10] M. Frank, D. Basin, and J. Buhmann. A class of probabilistic models for role engineering. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 299–310. ACM, 2008.

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

[12] M. Frank, A. Streich, D. Basin, and J. Buhmann. A probabilistic approach to hybrid role mining. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 101–111. ACM, 2009.

[13] J. Friedman, J. Bentley, and R. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):209–226, 1977.

[14] L. Fuchs, G. Pernul, and R. Sandhu. Roles in information security–a survey and classification of the research area. *computers & security*, 30(8):748–769, 2011.

[15] M. Gallaher, A. OÓČonnor, and B. Kropp. The economic impact of role-based access control. *Planning report 02-1*, 2002.

[16] D. Götzmann. Formal concept analysis implemented in java. `http://code.google.com/p/colibri-java/`.

[17] T. Iwuchukwu. *Anonymization techniques for large and dynamic data sets*. PhD thesis, The University of Wisconsin-Madison, 2008.

[18] G. Kabra, R. Ramamurthy, and S. Sudarshan. Redundancy and information leakage in fine-grained access control. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 133–144. ACM, 2006.

[19] M. Kuhlmann, D. Shohat, and G. Schimpf. Role mining-revealing business roles for security

administration using data mining technology. In *Proceedings of the eighth ACM symposium on Access control models and technologies*, page 186. ACM, 2003.

[20] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 25–25. IEEE, 2006.

[21] X. Ma, R. Li, and Z. Lu. Role mining based on weights. In *Proceeding of the 15th ACM symposium on Access control models and technologies*, pages 65–74. ACM, 2010.

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

[23] I. Molloy, H. Chen, T. Li, Q. Wang, N. Li, E. Bertino, S. Calo, and J. Lobo. Mining roles with multiple objectives. *ACM Transactions on Information and System Security (TISSEC)*, 13(4):36, 2010.

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

[25] I. Molloy, N. Li, Y. Qi, J. Lobo, and L. Dickens. Mining roles with noisy data. In *Proceeding of the 15th ACM symposium on Access control models and technologies*, pages 45–54. ACM, 2010.

[26] G. Neumann and M. Strembeck. A scenario-driven role engineering process for functional RBAC roles. In *Proceedings of the seventh ACM symposium on Access control models and technologies*, page 42. ACM, 2002.

[27] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy. Extending query rewriting techniques for fine-grained access control. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 551–562. ACM, 2004.

[28] H. Roeckle, G. Schimpf, and R. Weidinger. Process-oriented approach for role-finding to implement role-based security administration in a large industrial organization. In *Proceedings of the fifth ACM workshop on Role-based access control*, page 110. ACM, 2000.

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

[30] D. Shin, G. Ahn, S. Cho, and S. Jin. On modeling system-centric information for role engineering. In *Proceedings of the eighth ACM symposium on Access control models and technologies*, page 178. ACM, 2003.

[31] B. Steven, A. Trent, G. Katie, G. Ronald, B. S. Matthew, and M. S. Integrated Public Use Microdata Series: Version 5.0 [machine-readable database], 2010.

[32] M. Strembeck. Scenario-Driven Role Engineering. *IEEE Security & Privacy*, 8(1):28–35, 2010.

[33] H. Takabi and J. Joshi. StateMiner: An efficient similarity-based approach for optimal mining of role hierarchy. In *Proceeding of the 15th ACM symposium on Access control models and technologies*, pages 55–64. ACM, 2010.

[34] E. Uzun, V. Atluri, H. Lu, and J. Vaidya. An optimization model for the extended role mining problem. *Data and Applications Security and Privacy XXV*, pages 76–89, 2011.

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

[36] J. Vaidya, V. Atluri, and Q. Guo. The role mining problem: A formal perspective. *ACM Transactions on Information and System Security (TISSEC)*, 13(3):27, 2010.

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

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

[39] N. Verde, J. Vaidya, V. Atluri, and A. Colantonio. Role engineering: from theory to practice. In *Proceedings of the second ACM conference on Data and Application Security and Privacy*, pages 181–192. ACM, 2012.

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

[41] H. Zhang, I. Ilyas, and K. Salem. Psalm: Cardinality estimation in the presence of fine-grained access controls. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*, pages 505–516. IEEE, 2009.

[42] P. Zikopoulos, G. Baklarz, L. Katsnelson, and C. Eaton. *IBM DB2 9 new features*. McGraw-Hill, Inc., 2007.