Constrained Role Mining using K-Map

Shalini Sharma[1], Shivam Sharma[2], Shubham Sharma[3], Vishal[4], Lopamudra Mohanty[5]

[1,2,3,4]*Student, Department of Information Technology, Inderprastha Engineering College*
[5]*Professor, Department of Information Technology, Inderprastha Engineering College*

ABSTRACT:- *Many approaches have been proposed for role mining. However, the problems solved often differ due to a lack of agreement on the formal definition of the role mining problem. We provided 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 the basic assumptions on how access-control configurations are generated, here we proposed a novel definition of the role mining problem that fulfils the requirement the real-world enterprises typically have.*

KEYWORDS:- RBAC, Role engineering, Role mining, UA, PA, UPA

I. INTRODUCTION

Looking for a competitive edge, increased security and productivity, both system vendors and implementers have been looking for the means to properly administer these rapidly expanding and costly infrastructures.

With the wide-spread use of internet, the lower technology costs, and a great need for data access and sharing in a competitive market has driven the development of new technologies and standards. Role Based Access Control (RBAC) will allow for easier administration of today's large and complex corporate environments without sacrificing the need for securing data and access to it. Role-based access control (RBAC) has been used by a lot of commercial systems. As a result, RBAC has become the necessity in many organizations for enforcing security. Basically, a role is a set of permissions that represents organizational agents that perform the required job functions within the organization. Users are assigned appropriate roles based on their qualifications.
However, the major challenge in implementing RBAC is defining a complete and accurate set of roles. This process is known as role engineering.

The two basic approaches towards role engineering are top-down and bottom-up.

Top-down role mining identifies sets of identifying attributes that should collect users with identical resource requirement.
These attributes are associated with permissions on information systems. This approach begins with defining a certain job function and then creates a role for this job function by associating needed permissions.
Bottom-up role mining identifies sets of resources that should appear together, define them as roles, and search for users who have these resources, and consequently should be assigned the roles.
Both these are used in the context of role mining, where bottom-up role mining is often abbreviated simply as role mining. Bottom-up role mining is basically 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.

The basic role mining problem is similar to the problem of database tiling proposed by Geerts et al [3]. We show how our basic RMP can be mapped to the database tiling and present an algorithm to use tiling to discover roles. The recently proposed discrete basis problem is identical to the Minimal Noise RMP, and we show the mapping between our Minimal Noise RMP to the discrete basis problem. We are using our approach to solve the Role Mining Problem by applying the concept of **K-map**.

II. LITERATURE REVIEW

(Role-based Access Control: The NIST Solution, Hazen A. Weber, October 3, 2003, Version 1.4b)

To understand the benefits of a Role-Based Access Control in a security administrative model, we have to understand some of the concepts being utilized. While there are a lot of variations and ideas behind security administration such as Mandatory Access Control, Discretionary Access Control and Role-

Based Access Control, we are focussing on Role-Based Access Control.

Role-Based Access Control (RBAC): In 1992 a model was introduced by David Ferraiolo and Rick Kuhn that

attempted to meet the requirements of the scope and created a full-fledged RBAC solution. In order to understand the elements of the RBAC model it is helpful to understand the evolution of RBAC concepts.

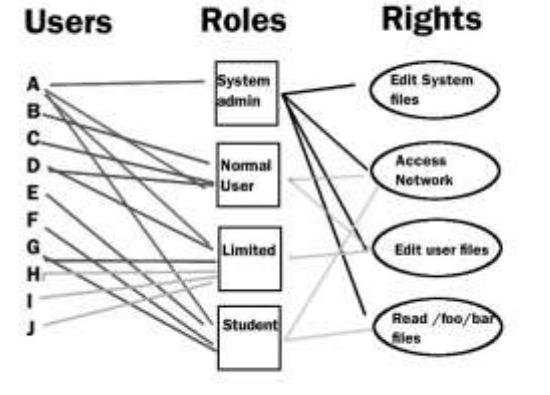

Figure 1: Representation of user-role model

RBAC model has three elements: users, roles and permissions.

User: User is any entity that requires to access a data resource or object. The user will typically not have access to resources, but instead inherit access to resources through the roles they are associated with. Users are generally the employees, network mechanisms or entities that wishes to access a resource.

Role: A role is a set of permissions based on a function. Users are assigned roles based on their position and the requirement of function in which they serve the organization. The user may have a single role associated with them or may have several roles depending on the needs of their position. Permissions (Rights): Permissions are assigned to a role and grant access to Operations. They could also be functions within a database such as insert, delete or append or could also consist of printing to a printer or accessing offline storage such as a tape drive.

III. METHODOLOGY

Our approach to solve the Role Mining Problem by applying the concept of K-map is governed by a set of rules.

1. The mapping is done based on the grouping the elements of order 2^n , where n= 0,1,2…
2. The group of highest order is mapped first.
3. The concept of map-folding of K-map is implemented in our approach.
4. The similar rows and columns can be merged together to reduce the number of rows and columns.

The Role-Mining Problem

Consider the following preliminaries

- U, ROLES are the set of users and roles.
- UA ⊆ U × ROLES, a many-to-many mapping user-to-role assignment relation.
- PRMS is the set of permissions.
- PA ⊆ ROLES × PRMS, a many-to-many mapping of role-to-permission assignments.1
- UPA ⊆ U × PRMS, a many-to-many mapping of user-to-permission assignments.
- Assigned users (ROLES) = {u ∈ U| (u, ROLES) ∈ UA}, the mapping of role R onto a set of users.
- Assigned permissions(R) = {p ∈ PRMS| (p, R) ∈ PA}, the mapping of role R onto a set of permissions.

Given m users, n permissions and k roles (i.e., |U| = m, |PRMS| = n, |ROLES| = k), the user-to-role mapping can be represented as an m × k Boolean matrix where a 1 in cell {I, j} indicates the assignment of role j to user i. Similarly, the role-to-permission mapping can be represented as an k × n Boolean matrix where a 1 in cell {I, j} indicates the assignment of permission j to role i. Finally, the user-to permission mapping can be represented as an m × n Boolean matrix where a 1 in cell {I, j} indicates the assignment of permission j to user i.

δ-Consistency: A given user-to-role assignment UA, role-to-permission assignment PA and user-to-permission assignment UPA are δ-consistent if and only if: $M(UA) \otimes M(PA) - M(UPA)\ k1 \leq \delta$ where M(UA), M(PA), and M(UPA) denote the matrix representation of UA, PA and UPA respectively. If δ-consistency allows us to bound the degree of difference between the user-to-role assignment UA, role-to-permission assignment PA and user-to-permission assignment UPA. For UA, PA, and UPA to be δ-consistent, the user-permission matrix generated from UA and PA should be within δ of UPA.

IV. ALGORITHM TO GENERATE ROLES

The following algorithm is used to generate number of roles from UPA matrix where UPA ⊆ U × PRMS.

The algorithm takes count of a density function that is provided to randomly generate the matrix based on the value provided as density (in percentage).

Given u users, p permissions and d density

UPA_gen(u,p,d)

```
UPA ← {}
// initialising empty UPA matrix
while( i<m )
  do while( j< n)
      do randnum ← rand()/rand_max
    // generating random integer < 1
      if random <=d
          UPA ← 1
    // 1 will be substituted randomly          //as per the value of density
     end while
end  while
```

Role_gen(u,p,d)

```
UPA[u][p] ← call UPA_gen(u,p,d)
Initialise r←0
for i<=u do
   for j<=p do
      for k< 4 do
         c←0
         Keep traversing the row
         if UPA[i][j] = 1
             then rol[c] ← store the index
             c++
         else increase the row
         end if
         if c=2
             r++
             Break
          end if
      end for
   end for
end for
return r
```

V. RESULT

As per figure no. 2, we can observe the variation in number of roles with varying density and size of UPA matrix.

With increase in size of UPA matrix and high density, the number of roles decreases eventually. As the density is decreased, we can observe a relatively high number of roles.

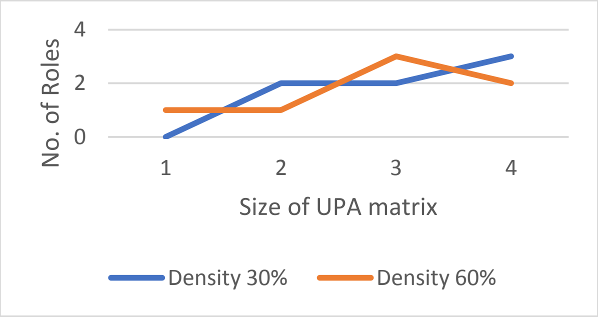

Figure 2: Variation of number of roles generated wrt density

Example

	P1	P2	P3	P4
U1	1	0	1	1
U2	1	0	1	1
U3	1	0	0	0
U4	1	0	1	1

Table 1: UPA Matrix

	R1	R2	R3
U1	1	1	0
U2	1	1	0
U3	1	0	0
U4	1	0	1

(a) User-Role Assignment

	P1	P2	P3	P4
R1	1	0	0	0
R2	0	0	1	1
R3	0	0	1	1

(b) Role-Permission Assignment

Table 2: Role Mining Problem by K-map

	P1	P2	P3	P4
U1	1	0	1	1
U2	1	0	1	1
U3	1	0	0	0
U4	1	0	1	1

Table 3: δ-approx RMP

VI. CONCLUSION

Our work states the role-mining problem for RBAC system. We have discussed the problem which people face while considering role-mining for RBAC systems. In various organisations information may be available in the form of User Permission Assignment matrix or an organisation may maintain access log. This information is needed to be converted into a suitable matrix form so that it is easy to determine the roles. We represent such information in the form of UPA matrix.

VII. REFERENCES

1. Ravi S. Sandhu, “Role-Based access control models”, George Mason University and SETA Corporation.

2. Jaideep Vaidya, Vijayalakshmi Atlur and Qi Guo,

3. “The Role Mining Problem: Finding a Minimal Descriptive Set of Roles”, MSIS Department and CIMIC Rutgers University 180 University Ave, Newark, NJ 07102.

4. Floris Geerts, Bart Goethals and Taneli Millikanian, “Tiling Databases”, Laboratory for Foundations of Computer Science School of Informatics, University of Edinburgh.

5. Mario Frank, Joachim M. Buhmann and David Basin, “On the Definition of Role Mining”, Department of Computer Science ETH Zurich, Switzerland.

6. Hazen A. Weber, “Role-Based Access Control: The NIST Solution”, October 8, 2003.

7. Carlo Blundo and Stelvio Cimato, “A Simple Role Mining Algorithm”, Dipartimento di Informatica ed Applicazioni Università degli Studi di Salerno I-84084