Deanship of Graduate Studies

Al-Quds University

Dynamic User-Oriented Role-Based Access Control Model (DUO-RBAC)

Hazem Malek Husni Kiwan

M.Sc. Thesis

Jerusalem-Palestine

1439 / 2018

Dynamic User-Oriented Role-Based Access Control Model (DUO-RBAC)

Prepared By:

Hazem Malek Husni Kiwan

B.Sc. Computer Information System from an-Najah

National University - Palestine.

Supervisor: Dr. Rashid Jayousi

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Science /Department of Computer Science Faculty of Science &Technology /Deanship of Graduate Studies /Al-Quds University.

1439 / 2018

Al-Quds University

Deanship of Graduate Studies

Computer Science Department

Thesis Approval

Dynamic User-Oriented Role-Based Access Control Model (DUO-RBAC)

Prepared By: Hazem Malek Husni Kiwan

Registration No: 21213029

Supervisor: Dr. Rashid Jayousi

Master thesis submitted and accepted, Date: 10/4/2018

The names and signatures of the examining committee members are as follows:

1-Head of Committee: Dr. Rashid Jayousi	Signature: [illegible]
2-Internal Examiner: Dr. Badie Sartawi	Signature: [illegible]
3-External Examiner: Dr. Radwan Tahboub	Signature: [illegible]

Jerusalem – Palestine

1439 / 2018

Dedication

I dedicate my thesis to my family. A special feeling of gratitude to my loving parents, whose words of encouragement and push for tenacity ring in my ears. And for my parents in law, who have never left my side and are very special. I also dedicate this dissertation to my beloved wife, and I will always appreciate her patience, encouragement and all she has done for me. I also dedicate this dissertation to my brothers, sister and friends who have supported me throughout the process.

Thank you all

Hazem Malek Husni Kiwan

Declaration:

I certify that this thesis submitted for the degree of Master, is the result of my own research, expect where otherwise acknowledged, and that this study (or any part of the same) has not been submitted for a higher degree to any other university or institution.

Signed: ….[illegible]……………………………..

Hazem Malek Husni Kiwan

Date: 10/4/2018

Acknowledgement

First of all, I would like to thank God who gave me strength and effort to complete my master thesis. I would also like to express my sincere gratitude to those who gave me the assistance and support during my master study especially my parents and my wife.

I would also like to thank my thesis advisor professors, Dr. Rashid Jayousi, who served on my thesis committee and for his continuous supports and advices at all stages of my work. My deepest gratitude and appreciation goes to Dr. Radwan Tahboub and Dr. Badie Sartawi as external and internal examiners of this thesis, and I am gratefully indebted to them for their very valuable comments on this thesis.

Another word of special thanks goes to Al-Quds University, especially for all those in the Faculty of Graduate Studies / Computer Science department.

ABSTRACT

Most researchers now trend to use role mining to generate role-based access control model from the existing user-permission assignments. User-oriented role-based access control is a type of role-based access control model, which aims to use role mining from end user perspective to generate a user-oriented RBAC model, since the user almost prefer a simple and minimum role assignments. This research is the first for generating a dynamic user-oriented role-based access control model (DUO-RBAC) for inserting a new user-permission assignments (new UPA) to the existing user-oriented RBAC model.

In a quick clarification, if there is a system which has user-permission assignments, a user-oriented RBAC model can be generated which contains new roles, each one assigns to users and permissions. Then, if we have a new users with new permissions should enter the system which has the model, we will regenerate a new model with new roles assignments to include these new users. Re-generating roles will be done by our dynamic model, with three constraints. First, there are no changes in the number of role assignments for each user in the system after the inserting process, since the user will be conflicted if he has different number of roles from time to time. Second, the permissions that each user has before the inserting process must be the same after generating the new model. Last one, will take into account that each user assign to number of roles no more than t (maximum number of roles that each user can assign), where t is predefined in the existing user-oriented RBAC model. Also, we develop a new algorithm, which based on user-oriented role mining to find the optimal way for inserting the new user permission assignments to the existing model. Our experiments applied on benchmark “Access Control” real datasets to evaluate the results and show the effectiveness of our developed algorithm of several measures. Those measures are: optimal number of roles to make the objective function minimized, optimal number of user-role assignments and generating a new model from end user perspective (keep the new generated model suitable from end-user perspective).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 Introduction

Access control is a security technique to specify who/what can view/use resources in a computing environment. The two major types of access control are: physical and logical. Systems that perform authorization identification, access approval, authentication, and accountability of entities through login credentials including passwords, personal identification numbers (PINs), biometric scans, and physical or electronic keys. Access control can be categorized into four main categories: First one is discretionary access control (DAC), by this one the user will has a complete and full control on all the programs and files into the system. Second one is mandatory access control (MAC), this one is the opposite of DAC, and it means that the access control is a policy, hardware, or software component that is used to restrict access to a resource, like password. Third one is rule-based access control (RuBAC), that mean when a request is placed to access a network or a resource, a controlling device blocks or allows access these resource based on that user's role in the organization. And last one is role based access control (RBAC), this is permission model that allows to authorize users to do their tasks and perform their actions while they are browsing the system. The permissions of users in RBAC model are fetched through role mining to generate roles based on existing user permission assignments. Then, the permissions are assigned to corresponding roles and roles are mapped to the system’s users.

The concept of Role-Based Access Control was conceived and implemented in online systems with multiple users and multiple integrated applications in 1970s. [1].

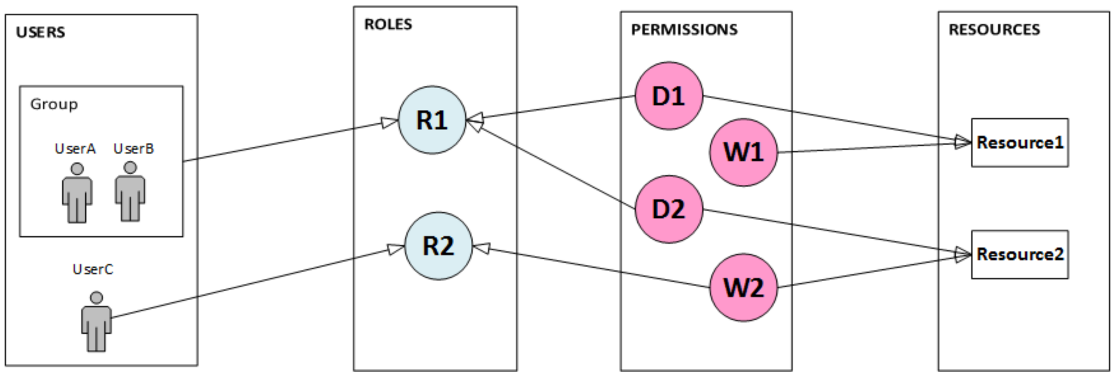


Figure 1. 1: Role-Based Access Control Model

As shown in figure 1.1, when the user accesses a content and performs an action on it (edit, delete, create or view) while browsing the system, RBAC records the event performed by the user on that particular content. It also maps the previously created role to that user and then allows or denies that user based on the permissions of that role. However, by this role the application either allow or deny the user to make this event.

“Role engineering” [2], is the task of configuring a Role-Based Access Control (RBAC) system. The term “role mining” is used to refer to automated approaches to role engineering [5]. In [3] we can find some examples of using different approaches in role engineering like “top-down” and “bottom-up”.

The key idea of role mining is to utilize the data mining technologies to discover a good role set from the existing user-permission assignments of the old access control system. The discovered roles are then applied in the system through the actions of the corresponding users. In a nutshell, the existing studies investigate the concept of role mining in different objectives, such as minimization of the administrative cost, minimization of the number of roles, minimization of the complexity of the role hierarchy structure introduced in [6], security administration introduced in [7], and user-oriented role mining.

In [4], the authors developed the only known algorithm that tailored to user-oriented role mining that explores the role mining problem from a user perspective, then generates a user-oriented role-based access control model under multiple constraints. The constraints include that users should not be assigned to roles that doesn't exceed a predefined value of "t", the generated model completely reconstructs the existing user-permissions assignments and the number of roles should be minimized. In this research, we focused on user-oriented role-based access control that is generated after using user-oriented role mining. Then, a dynamic model is generated to insert new users with their permissions (new UPA) to the existing model by using our algorithm without regenerating the model after adding the new user-permission assignments to the original dataset.

These terms have been further refined. Our contribution is to insert these new user-permission assignments to the existing user-oriented model under the same three constraints that are defined when generating user-oriented role based access control model. And the algorithm also guarantees that no other changes occur on the number of role assignments that each user has in the system before inserting the new UPA to the existing model "still having a user-oriented model".

1.2 Problem Statement

The problem studied in this research is to insert new users with their permissions (new user-permission assignments UPA) to the existing user-oriented RBAC model in the system that generated by running user-oriented exact RMP algorithm from the existing user-permission assignments.

1.3 Research Questions

- How can we build a dynamic user-oriented role-based access control model to insert the new user-permission assignments to the system without regenerating the existing model?
- How can we develop a dynamic algorithm using user-oriented role mining for the dynamic model to insert the new user-permissions assignments to the system without affecting the roles assignments for the existing users in the existing model?

    - What is the relation between the algorithm and the dynamic model?
    - How can we evaluate the results from the ones that we got from the algorithm?

Figure 1.2, shows an abstract description for the research question.

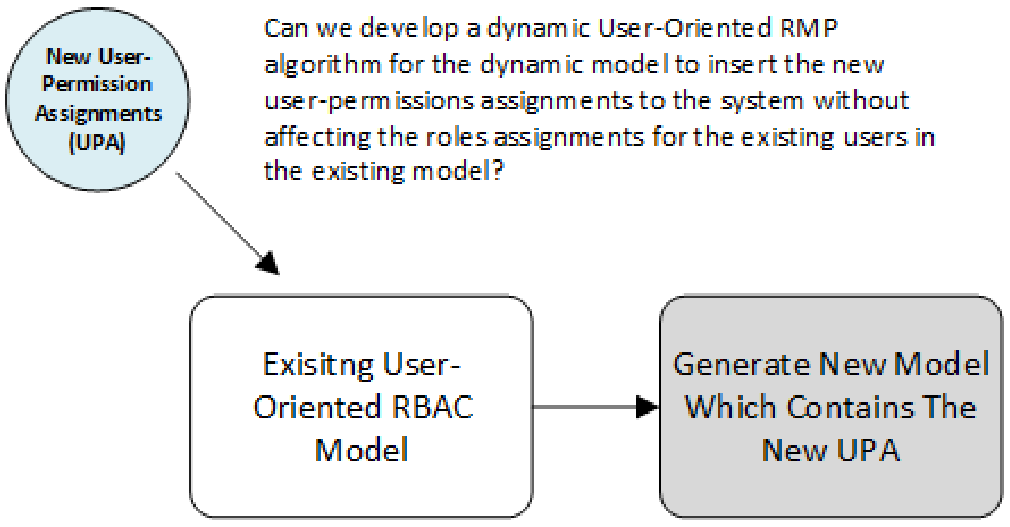

New User-Permission Assignments (UPA)
Can we develop a dynamic User-Oriented RMP algorithm for the dynamic model to insert the new user-permissions assignments to the system without affecting the roles assignments for the existing users in the existing model?
Exisitng User-Oriented RBAC Model
Generate New Model Which Contains The New UPA

Figure 1. 2: Question for generating DUO-RBAC Model

1.4 Research Motivation

There are different algorithms that use role mining to generate user-oriented role-based access control model (UO-RBAC) from an existing user-permission assignments in the system. But there aren't any dynamic algorithms that take into account whether we have new users with permissions (new user-permission assignments UPA) that join the system after the model is generated. (Don't have a dynamic model which depends on a dynamic algorithm to insert those new UPA to the generated model in the system).

However, the only two ways to insert the new user-permission assignments (UPA) to the generated model in the system without a dynamic algorithm are:

1. Add the new UPA to the original dataset which contains the existing user-permission assignments before generating the model, then rerun the algorithm on the dataset (after adding the new UPA) to generate new optimal user-oriented RBAC model. The problem of this solution is when the model is regenerated we will have a new user-role assignments for each user and different number of role assignments for some users, in this case the user will be conflicted when the number of his role assignments change from time to time. So, the main concept of user-oriented model will not be available.

2. Assign all the permissions for each new user to a new role, then assign this role for the user that contains his permissions (each user from the new users will assigned to one role which contains his permissions). The problem of this solution is that the total number of roles and the total number of user-role assignments are not optimal. So, in this case we do not have an optimal user-oriented RBAC model.

1.5 Solution

Our proposed solution is to build a dynamic user-oriented role-based access control model (DUO-RBAC), which uses a new dynamic algorithm that depends on user-oriented role mining; to insert the new users with their permissions (new user permission assignments UPA) to the existing user-oriented RBAC model by our new dynamic algorithm in the optimal way with two constraints: keeping the model user-oriented and be sure that no user (old users and new users) assign to more than t roles that predefined in the existing model.

For more clarification, we introduce an example by using a small dataset that contains user-permission assignments to describe briefly how the process will be, the main contribution of our algorithm, and the main differences between the insertion results by rerunning user-oriented exact RMP algorithm and running our dynamic algorithm.

<u>Example 1.1:</u>

Suppose we have an existing user-permission assignments (UPA) as figure 1.3 shows, then we need to generate user-oriented RBAC model which contains user-role assignments (UA), roles (R) and role-permission assignments (PA), with constrains that the generation will completely reconstruct the existing assignments and no user assigns to roles that doesn't exceed t [4], (in this case we suppose t=2).

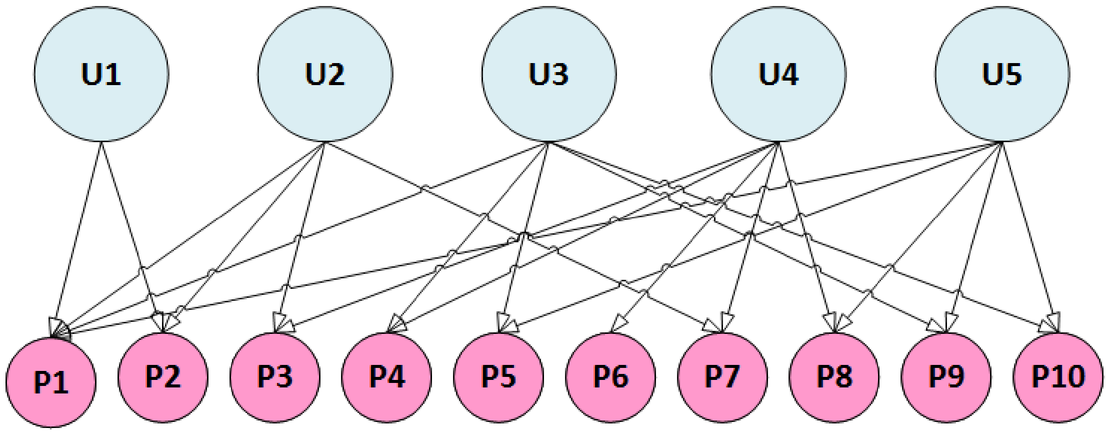


Figure 1. 3: Existing User-Permission Assignments

In this case we applied user-oriented exact RMP algorithm on the existing UPA, after that we have generate user-oriented RBAC model as figure 1.4 shows.

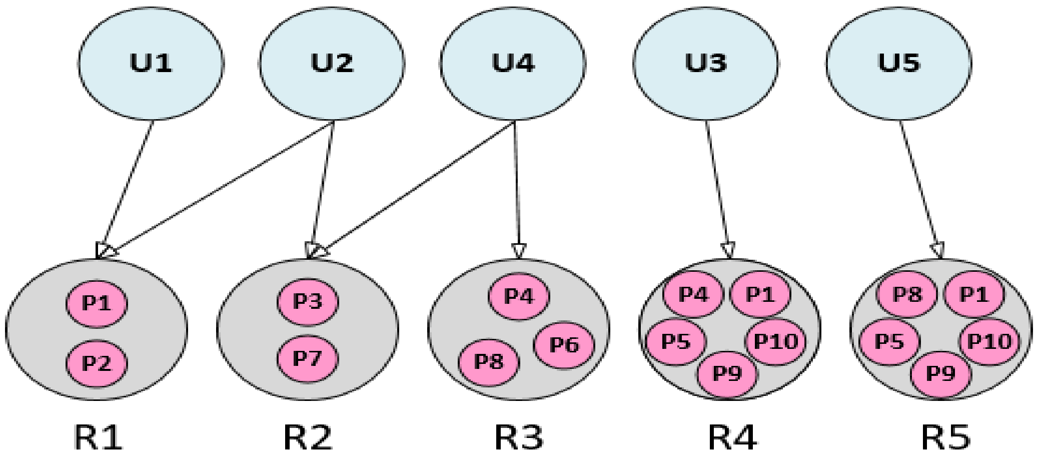


Figure 1. 4: User-Oriented RBAC Model after Running User-Oriented Exact RMP Algorithm

The figure above shows the generated user-oriented RBAC model by using user-oriented exact RMP algorithm, the model contains 5 roles, each role has a set of permissions, and each user assigns to less than 2 role (since t in this case = 2), as the following:

- U1 assigns to Role1.
- U2 assigns to Role1 and Role2.
- U3 assigns to Role4.
- U4 assigns to Role2 and Role3.
- U5 assigns to Role5.

Table 1.1 shows the number of roles (R), user-role assignments (UA) and role-permission assignments (PA) in the generated new user-oriented RBAC model.

Table 1. 1: User-Oriented RBAC Model Results

t	R	UA	PA
2	5	7	17

As figure 1.5 shows, we have a new user-permission assignments (new UPA) need to enter them to the system after the model was generated, here we will try two different ways for this process, and this is what we will describe next.

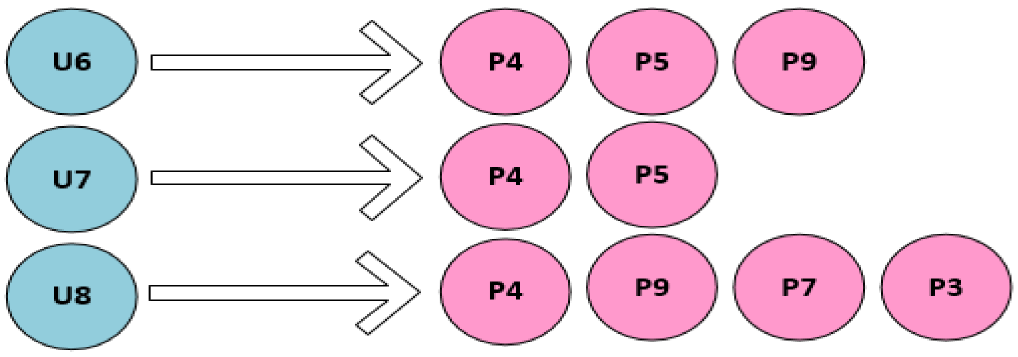


Figure 1. 5: New User-Permissions Assignments

First way to enter the new UPA to the system is by adding those new UPA to the original data (UPA), then regenerate user-oriented RBAC model by using the same algorithm which used before (user-oriented exact RMP), after that we will have a new user-oriented RBAC with an optimal number of roles (R) and user-role assignments (UA).

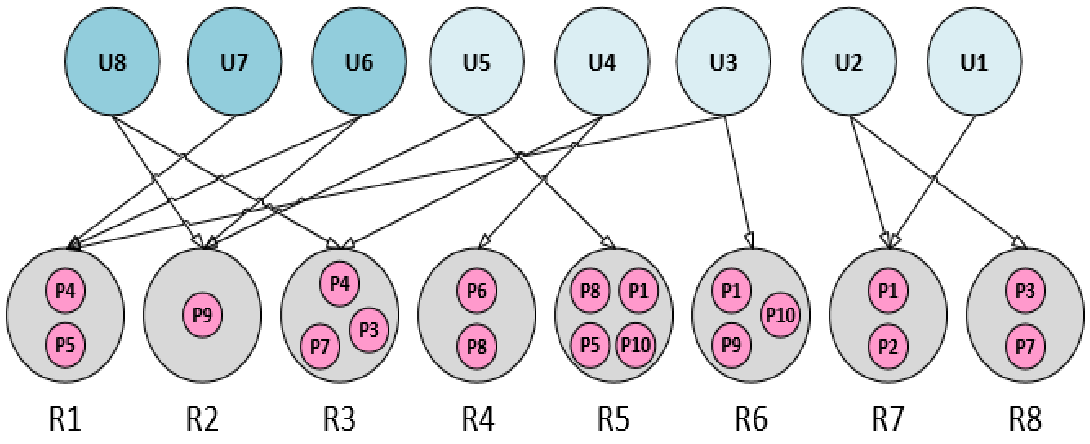


Figure 1. 6: User-Oriented RBAC Model after Rerun User-Oriented Exact RMP Algorithm New Users Dataset

Figure 1.6 shows the new user-oriented RBAC model that generated after entering the new UPA. The new model contains 8 roles, each role has a set of permissions, and each user assigns to less than 2 role, as the following:

- U1 assigns to Role7.
- U2 assigns to Role7 and Role8.
- U3 assigns to Role1 and Role6.
- U4 assigns to Role3 and Role4.
- U5 assigns to Role2 and Role5.
- U6 assigns to Role1 and Role2.
- U7 assigns to Role1.
- U8 assigns to Role2 and Role3.

Table 1.2 shows the number of roles (R), user-role assignments (UA) and role-permission assignments (PA) in the generated new generated user-oriented RBAC model.

Table 1. 2: New User-Oriented RBAC Model Results

t	R	UA	PA
2	8	14	19

The second way to enter the new UPA to the system is by using our dynamic algorithm. The algorithm entering the new UPA to the existing user-oriented RBAC model, on this process each user from the new users will have an assignments to an existing role or to a new role which generated through the process when needed, the number of the assignments for each user must be less than or equal t.

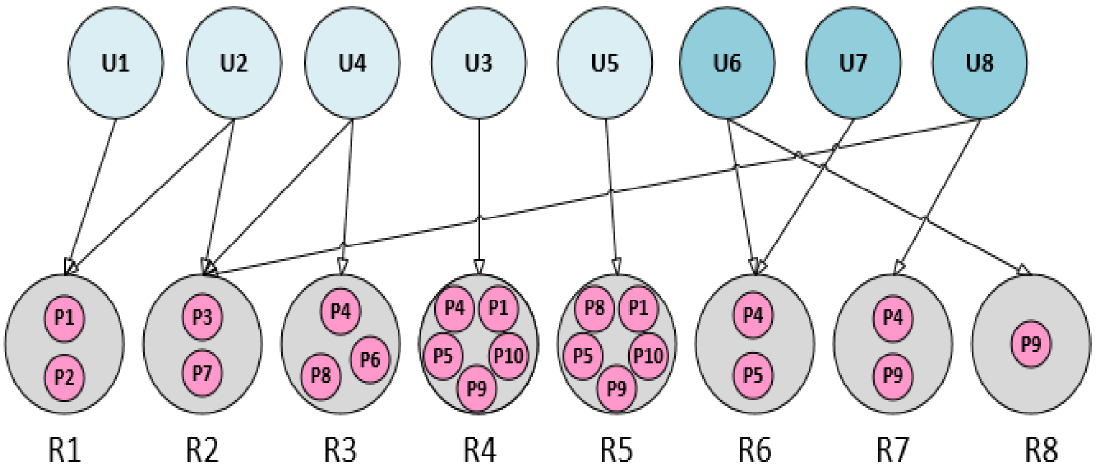


Figure 1. 7: User-Oriented RBAC Model after Inserting New Users by our Dynamic Algorithm

Figure 1.7 shows the result of the new user-oriented RBAC model which generated after inserting the new UPA to the system. The new model contains 8 roles, each role has a set of permissions each user assign to less than 2 roles, as the following:

- U1 assigns to Role1.
- U2 assigns to Role1 and Role2.
- U3 assigns to Role4.
- U4 assigns to Role2 and Role3.
- U5 assigns to Role5.
- U6 assigns to Role6 and Role8.
- U7 assigns to Role6.
- U8 assigns to Role2 and Role7.

Table 1.3 shows the number of roles (R), user-role assignments (UA) and role-permission assignments (PA) in the generated new user-oriented RBAC model.

Table 1. 3: New Dynamic User-Oriented RBAC Model Results

t	R	UA	PA
2	8	12	22

Now, after we inserted new user-permission assignments to the system by two different ways, we have two different results.

In chapter 4 we will analyze the results that we got after inserting the deleted users with their permission assignments at each dataset, then compare our results with the results that we have after running user-oriented exact RMP. Also, we will analyze the results that we got in the example to introduce our contribution by making a comparison between both results.

1.6 Research Objectives

The aims of this work is to build a dynamic user-oriented RBAC model to insert the new users with their permissions (new UPA) to the existing user-oriented RBAC model by a dynamic algorithm with the following sub objectives:

- Build and design a dynamic user-oriented role-based access control model (DUO-RBAC).
- Develop a new dynamic algorithm depends in user-oriented role mining to add the new users with their permissions (new UPA) to the existing model.
- Evaluate the results that we got after inserting the new UPA to the existing model and compare them with original results.
- Make sure that after the insertion process we will still have the following:

  - There are no changes in the number of role assignments for each user in the system after the insertion process, since the user will be conflicted if he has different number of roles from time to time. By this way we can agree that we will have a user-oriented role-based access control model (generating modal from end-user perspective).

- There are no user (from old users or new users) assigns to more than t roles (maximum number of roles that each user can assign) that was predefined in the existing model in the system.

1.7 Research Methodology

This section describes the research methodology that was followed.

- Discuss the evaluation criterion and measurement that is used to determine the efficiency of the role-mining algorithm to generate RBAC model.
- Show that there is not any algorithms take into account inserting new users with their permissions (new UPA) after generating the model.
- Show the weaknesses of the two available algorithms of inserting the new UPA to the existing model, and how it is inconsistent with the user-oriented concept.
- Building DUO-RBAC model which is used to insert the new user-permission assignments (new UPA) to the existing model in the system.

- Develop and implement a Dynamic algorithm depends on user-oriented role mining to show the results for our proposed model. The proof aims to show that the model aims to preserve the model`s user-oriented by studying and linking to previous researches.

- Do the experiment of our model and algorithm 7777on a real benchmark.

- Compare the results with the results of the best known algorithm that tailored to user-oriented role base access control model.

2. BACKGROUND AND RELATED WORKS

2.1 Introduction

In this chapter, we will walk through a literature review for previous works on role engineering, role mining, role-based access control, and user-oriented role-based access control. Also, in the last section we will provide an analysis for the previous work and conclusion.

2.2 Background

The concept of role mining and role engineering were introduced in different previous researches.

2.2.1 Role Engineering

“Role Engineering is a security-critical task for systems using role-based access control (RBAC)” [8]. Coyne, 1996 [2], introduced role engineering concept to find a correct and complete architecture structure to generate a business function and organization’s security policies. They used a top-down process oriented strategy to produce and generate roles. They also introduced role engineering which is used for role-based access control as the process of defining roles, permissions, constraints and role-hierarchies [9]. In this research, the authors applied a novel scenario-driven role engineering process to introduce and generate roles in different case studies.

The button-up approach for role engineering was used in [10] to find a set or roles R, with a set of user-role assignments UA and role-permission assignments PA.

2.2.2 Role Mining

Hung, 2012 [11] introduced the concept of role mining to enhance one of the existing access control model. The authors use an existing algorithm and apply it on the RBAC model. This algorithm depends on the data mining concept to enhance the Role-Based Access Control (RBAC) model by specifying several constraints, defining bridging entities, synthesizing a role-based access delegation model to target on specific objects, and developing domain ontologies as instantiations of the general model. Then, they applied their generated model to the New York State HIV Clinical Education Initiative (CEI) project. It also studies the behavior of the end-user to determine the perfect algorithm they used. However, In Ene, 2008 [10] the authors used a Lattic algorithm to minimize the number of the roles that each user has with the same permissions. The authors formulate several role engineering problems that introduced by [12] to develop an optimal algorithm. They defined role mining problem as a problem for discovering the optimal set of roles from existing UPA (user-permission assignments). Also, they described the relation between the role mining problem and the several problems that already identified in the data analysis and data mining.

Added weight to the role mining process was applied on [13]. This research focus on given a weight for permissions to reflect the importance of each permission in the system, they assigned each permission to a weight in a feasible ways. They also used a matrix to present the relationships between the user and the permissions in the system, then calculate the similarity and find how to define the weight for permissions based on this similarity.

There are two limitations of these researches. The first limitation is to generate a role-based access control model by using role mining for the existing user-permission assignments without taking into account if there are a new users with a new permissions needed to insert to the system after generating the model. Second, they focused on generating the model with a minimum number of role and role-permission assignments to reduce the complexity of role hierarchy and reduce the administrative cost on the system. On other hand, our study focuses on implementing a new dynamic role-based access control model based on the user-oriented concept and role mining concept. Also, this research is using a new dynamic algorithm to insert the new users with their permission to the system after generating the model.

2.3 Role-Based Access Control and User-Oriented Role-Based Access Control

2.3.1 Role-Based Access Control (RBAC)

Access control model is used and applied in different information systems with different objectives. The most popular access control model at present that is Role-based access control (RBAC) [14] [1], RBAC has been widely applied in enterprise security management products. The architecture of this model consists a set of users assigned to roles and a set of permissions assigned to those roles.

For using the new applied access control model, [15] focused on finding a new access control model for healthcare system. The authors used principles and characteristics of the traditional access control models, like Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role Based Access Control (RBAC). The access controls are used to facilitate a suitable access control model for electronic healthcare record. In this study, a new access control model for electronic health records system based on end user requirements (health authority, patient, health professional) was developed.

Also, [16] focused on improving the access control model in an existing healthcare portals. The authors present a service-oriented architecture for such e-Health portals. They also present a role-based access control module and Flexible Authorization Framework (FAF) model, study conflict resolution and interaction between the two modules.

However, the prototype had a limitations. It was limited by 4 attributes (ID, general health, sexual health and mental health), they did not provide a solution for privacy requirements because as they mentioned the healthcare industry patients health prevails over privacy requirements, and it did not take into account if there are a new users with their corresponding permissions to be added into the system after generating the model. So, they limited it with existing users before generating the model. On the other hand, our study focusses on using a data mining concept to generate a dynamic model and develop a dynamic algorithm. Then applying it on the existing model for inserting the users with their permissions to the system.

2.3.2 User-Oriented Role-Based Access Control (UO-RBAC)

The concept of user-oriented role-based access control was introduced in [4].

This research was the first and only research (as they mentioned) that used user-oriented role mining to define role mining from users' perspective. Their study depends on generate a user-oriented RBAC model by assign each user in the system to roles as few as possible; since the user does not prefer to being overwhelmed by assuming too many roles. In the fact, each user would wish to have only one role assign to him, and it provides the all necessary access privileges related to his work and function smoothly. Actually, the most organization's systems are designed that way. For example, in a healthcare system each employee carry only one role, either MANAGER, ACCOUNTANT, PATIANT or DOCTOR. So, user-oriented role mining is characterized with the fact that the maximum role assignment for each user (defined as t) should be constrained.

The authors used the concept of role mining to sparse user-role assignments, then they developed a user-oriented exact role mining problem (RMP) algorithm to generate a user-oriented role-based access control model from the existing user-permission assignments UPA in the system. They defined constraints to their algorithm while generating the model such as: using user-oriented role mining problem to finding the minimum number of roles from the candidate roles, completely reconstruct the existing user-permission assignments, and no user can have more than t roles (t defined in the beginning of their algorithm). Their experiment is conducted on a benchmark access control datasets. Then, the generated model contains total number of roles |R|, total number of role-permission assignments |PA|, and the total number of user-role assignments |UA| for each dataset, the number of direct user-permission assignments |DUPA|, and the number of edges in the reduced role hierarchy |t_reduce(RH)|. Those five main factors can be used to evaluate the feasibility of any RBAC model as mentioned in [6]. But in case of user-oriented RBAC model they used the first three factors to evaluate their model since no further exposition is needed on them. Also, they applied weighted structural complexity measure introduced in [6] as added evaluative criteria to evaluate the model.

$$\begin{aligned} &min\ \ w_r * |R| + \lambda|UA| - \lambda * t \\ s.t. &\left\{ UPA_{m\times n} = UA_{m\times k} \otimes PA_{k\times n} \right. \end{aligned} \qquad (1)$$

This objective function was developed by [4] depends on using user-oriented exact role mining problem (RMP). It can be affected by number of roles, number of user-role assignments and the weight for each role in the model. We must get the optimal number of roles and user-role assignments to make the objective function minimized. Now, the question here, how they got this objective function?

Suppose we have n permission, m users, user-permission assignments UPA_{mxn}, and positive number t, and we will use all these data to discover user role assignments and role set PA_{kxn} and user-role assignment UA_{mxk} under constraints that: total number of roles (k) minimized, user-role assignments and role-permission assignments completely reconstruct the existing user-permission assignments, and no user has more than t roles. [4] Described those constraints mathematically by the following function:

$$\min\ k$$
$$s.t. \begin{cases} UA_{m\times k} \otimes PA_{k\times n} = UPA_{m\times n} \\ \sum_j UA(i,j) \leq t,\ \forall i \\ UA \in \{0,1\}^{m\times k}, PA \in \{0,1\}^{k\times n} \end{cases} \qquad (2)$$

$$\min\ |R| + \sum_i \lambda_i \left(\sum_j UA(i,j) - t\right)$$
$$s.t. \left\{ UPA_{m\times n} = UA_{m\times k} \otimes PA_{k\times n} \right. \qquad (3)$$

“Where λ_i is the Lagrange multiplier for the constraint of

$\sum_j UA(i,j) \leq t.$

Further, we could assume that all Lagrange multipliers have the same value λ” [4]. Then we will have the objective function as shown in the equation 1. More details in [4].

However, the only one limitation in this research is if we need to add a new users with their permissions (new user-permission assignments) to the system which already has a generated user-oriented RBAC model. In this there are only two existing solutions.

Compare to our approach, our solution is dynamic user-oriented role-based access control. We developed a dynamic algorithm depends on user-oriented role mining to add these new user-permission assignments to the existing model without using any of the two previous solutions. In this case, we will be sure that the total number of roles and the total number of user-role assignments in the model after the insertion process minimized, no user have roles more than t that predefined in the existing model, and we also keep the RBAC model suitable user perspective.

2.4 Summary

In this chapter, we studied different researches that related to the concept of role engineering [2] [9] [10], role mining [10] [11] [12] [13], and role-based access control model. Also, we discussed the different approaches that are used to explore the roles for generating RBAC model such as bottom-up approach and top-down approach [3]. Then we studied some researches that applied RBAC model in a different systems such as e-health system [15] [16]. In contrary of other researches, ours focused on user-oriented role-based access control model [4]. Then, we show the limitation of this model and how our solution is better.

3. DUO-RBAC MODEL

3.1 Introduction

This chapter describes our proposed dynamic user-oriented role-based access control model (DUO-RBAC) and dynamic algorithm depends on user-oriented role mining. DUO-RBAC is a model for inserting a new users with their permissions to the system which already has a generated user-oriented RBAC model. Our model uses the dynamic algorithm to insert those new users with their permissions to the system.
User-oriented RBAC model, is the first model that depends on user-oriented role mining. Also, this model is the first to explore the role mining problem from the end user perspective. In general, role-based access control model allows the users to have multiple roles, then he can use these roles and switch between them to access his permission in the system. In case of user-oriented RBAC, there is a maximal number of roles that each user can has in the system, and this number predefined in the model itself by the authors as t [4]. Their point is the user always prefer to have a minimal number of roles assignments to access his permission though using the system. Our model is dynamic model based on user-oriented to add a new user-permission assignments to the existing user-oriented model with constraints, these constraints will be described in this chapter. Figure 3.1, shows how DUO-RBAC model works.

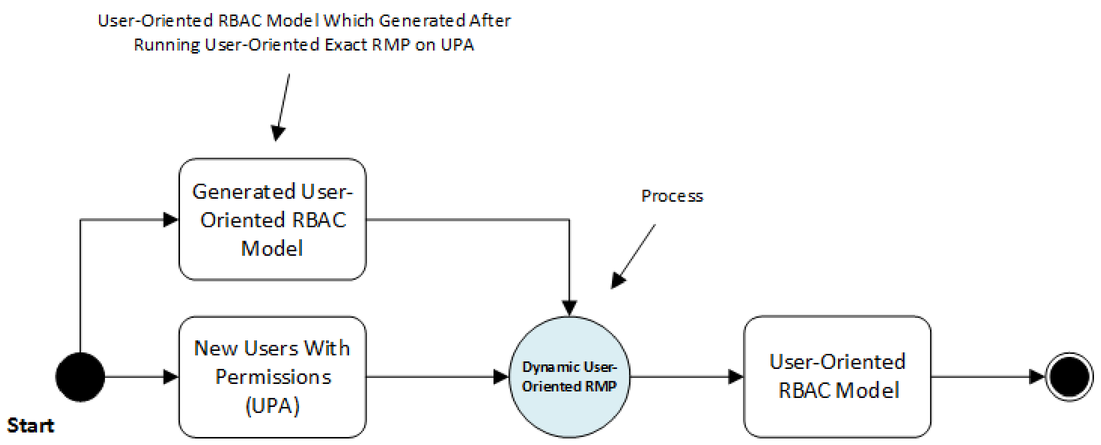

User-Oriented RBAC Model Which Generated After Running User-Oriented Exact RMP on UPA
Generated User-Oriented RBAC Model
Process
New Users With Permissions (UPA)
Dynamic User-Oriented RMP
User-Oriented RBAC Model
Start

Figure 3. 1: Data Flow Model for DUO-RBAC Model

First, we have an existing generated user-oriented RBAC model and a list of new users with their permissions (UPA) that needs to be added to the model. Then, those two sections will enter to the dynamic algorithm. Finally a new user-oriented RBAC model will be generated which contains the new users and their permissions. Figure 3.2, shows the environment model for DUO-RBAC model.

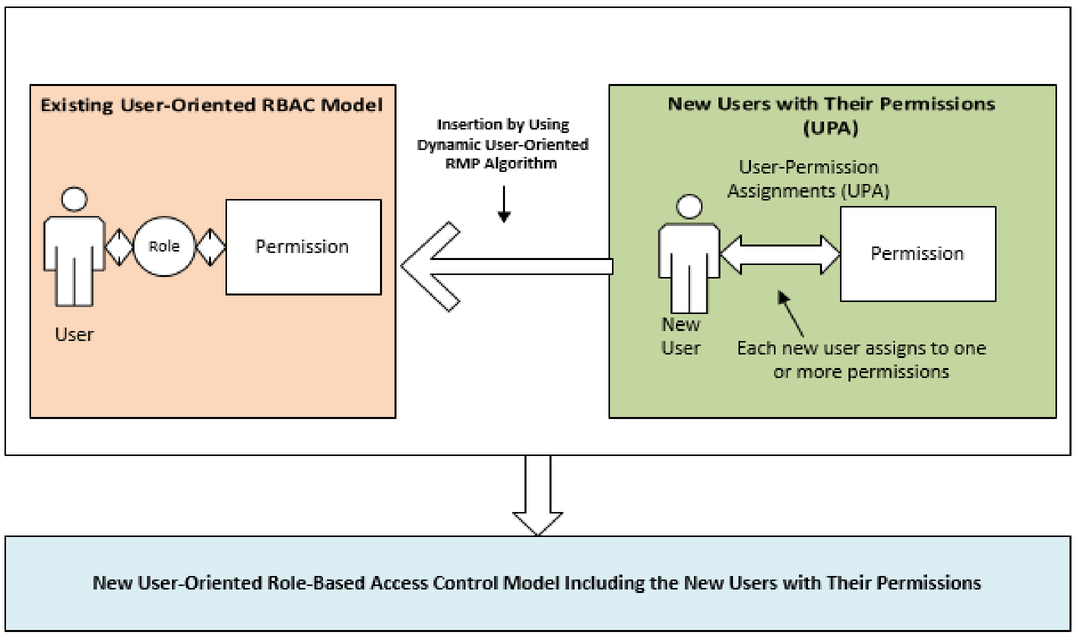

Existing User-Oriented RBAC Model
Role
Permission
User
Insertion by Using Dynamic User-Oriented RMP Algorithm
New Users with Their Permissions (UPA)
User-Permission Assignments (UPA)
Permission
New User
Each new user assigns to one or more permissions
New User-Oriented Role-Based Access Control Model Including the New Users with Their Permissions

Figure 3. 2: Environment Model for DUO-RBAC Model

Figure 3.3, shows the structure of the new user-oriented role-based access control model after inserting the new users with their permissions.

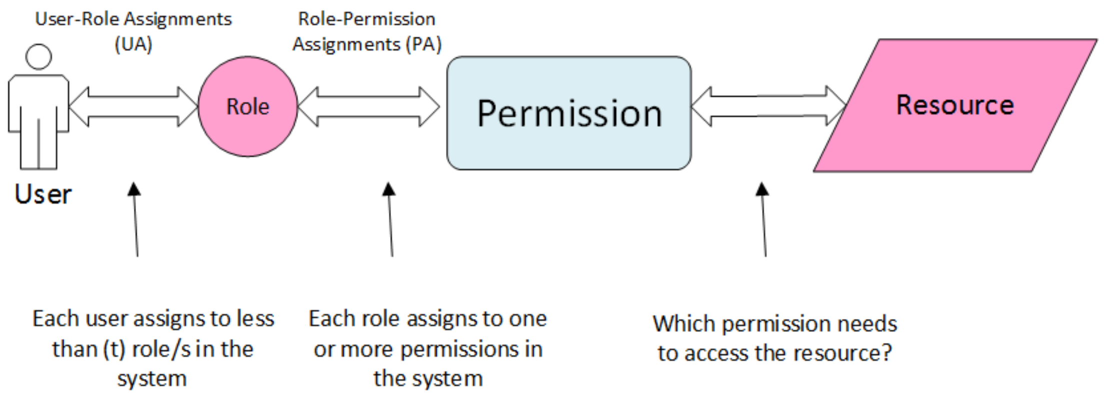

User-Role Assignments (UA)
Role-Permission Assignments (PA)
User
Role
Permission
Resource
Each user assigns to less than (t) role/s in the system
Each role assigns to one or more permissions in the system
Which permission needs to access the resource?

Figure 3. 3: Structure for New User-Oriented RBAC Model

As other role-based access control models, user-oriented RBAC model's structure include the following:

- Users (U)
- Roles (R)
- Permissions (P)
- User-Role Assignments (UA)
- Role-Permission Assignments (PA)

After the model is generated, each user in the system is assigned a number of roles that doesn't exceed t [4]. Also, each role is assigned to one or more permission in the system. The result is that the new users with their permissions are in the system without changing any existing assignments in the existing model before running our model. Through this method, each old user in the system still has the same role assignments and each new user joining the system has new assignments to an existing role or to a new role which is generated through the insertion process.

Our dynamic algorithm that we used in DUO-RBAC model is the first algorithm which depends on user-oriented role mining to insert a new users with their permissions to an existing user-oriented RBAC model. It introduces an optimal way to insert all users and their permissions to the existing user-oriented RBAC model. It also covers all possible cases when we add those new users.

3.2 Dynamic User-Oriented RMP

Dynamic user-oriented RMP algorithm depends on the concept of dynamic role generation [18] and user-oriented role mining [3] [4]. It focuses on the new UPA dataset (users with their permissions) and the existing user-oriented RBAC model. It inserts user one by one to the system with an optimal way to keep the increasing number of roles and user-role assignments (UA) perfectly. It also make sure that the constraint of users not being assigned roles that doesn’t exceed t is achieved [4].

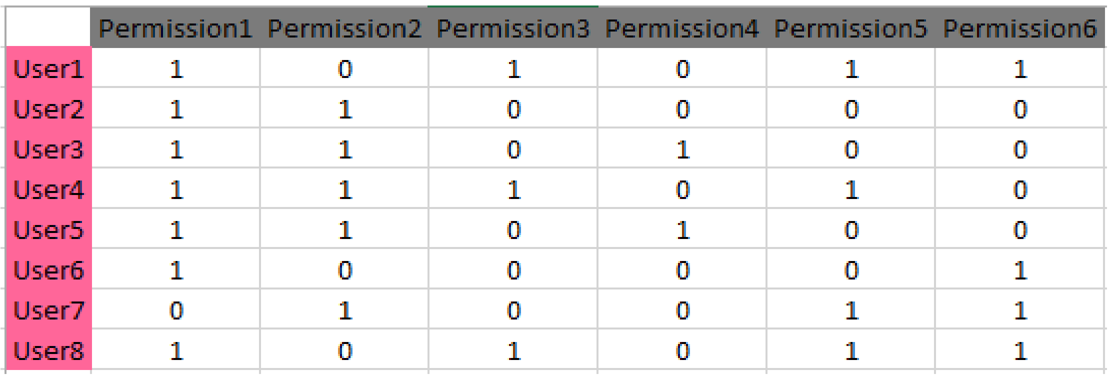
Dynamic user-oriented RMP, is an algorithm to insert a new UPA dataset to the existing user-oriented RBAC model that is generated from running user-oriented exact RMP algorithm on the old dataset, and finding the optimal final number of roles and user-role assignments the constraint of not having users assigned to roles that doesn’t exceed t [4].

In the beginning, and before describe our algorithm, we apply a preprocessing step to reduce the size of the new dataset which contains the new user with their permissions that will enter the model, the step removes all users that have the same permissions, then replace them by one user. This step is applied on other role mining algorithm, such as [17]. They also applied one more preprocessing step by checking if there any user on the dataset has permission that no other user has. In this case they create a new role which contains those permissions then assign it to him. In our approach, we will not apply this step, because if there is a user in the new dataset has unique permission we cannot confirm that there is no user into the existing model has this unique permission. So, this step will considered as a step into our algorithm.

The following example shows how to apply the preprocessing step on the new dataset.

Example 3.1

Table 3. 1: Existing User-Permission Assignments (UPA)

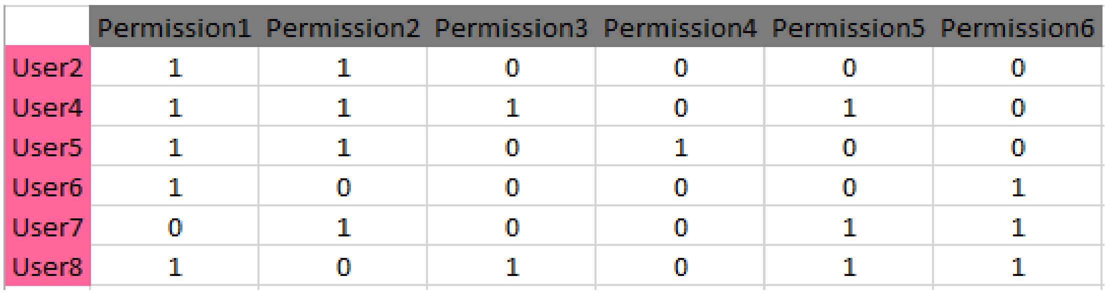


	Permission1	Permission2	Permission3	Permission4	Permission5	Permission6
User1	1	0	1	0	1	1
User2	1	1	0	0	0	0
User3	1	1	0	1	0	0
User4	1	1	1	0	1	0
User5	1	1	0	1	0	0
User6	1	0	0	0	0	1
User7	0	1	0	0	1	1
User8	1	0	1	0	1	1

If we take a look on table 3.1, we will note that User1 has the same permissions that User8 has and User3 has the same permissions that user5 has. So, in this state we will replace the two users in each case by one user, by this way we reduced the size of the dataset by deleting two users. After generating roles, the deleted users will be assigned to the same roles that the replaced user has.

After applying the preprocessing step, the remaining user-permission assignments will be as follows:

Table 3. 2: **Remaining User-Permission Assignments (UPA)**



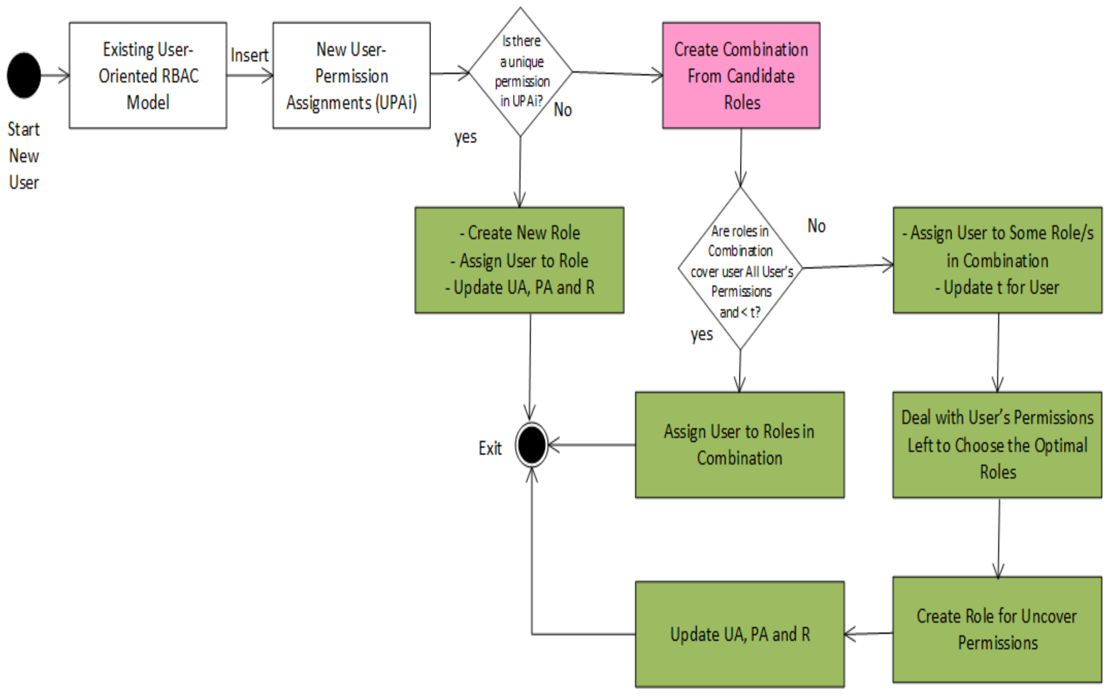
	Permission1	Permission2	Permission3	Permission4	Permission5	Permission6
User2	1	1	0	0	0	0
User4	1	1	1	0	1	0
User5	1	1	0	1	0	0
User6	1	0	0	0	0	1
User7	0	1	0	0	1	1
User8	1	0	1	0	1	1

The structure of our dynamic algorithm depends on entering the new users one by one to the model, then finding the optimal way to assign this user to an existing role or generate a new role then assigns this user to it as figure 3.4 shows below. Through this process we are making sure that all user-role assignments for each user in the existing user-oriented RBAC do not change, the total number of roles (R) and user-roles assignments (UA) do not affect the final value objective function to be minimized, and achieve the constraint that there aren't any users assigned roles that doesn't exceed t [4].

In the following an algorithm capture for the preprocessing step algorithm.

Preprocessing Step Algorithm

```
   Input: UPA
   Output: UPA'
1: If $\forall UPA_i$ in UPA s.t. $UPA_i \notin UPA'$ then
2:    UPDATE UPA' by adding $UPA_i$
3: End If
```



Start New User
Existing User-Oriented RBAC Model
Insert
New User-Permission Assignments (UPAi)
Is there a unique permission in UPAi?
No
yes
Create Combination From Candidate Roles
- Create New Role
- Assign User to Role
- Update UA, PA and R
Are roles in Combination cover user All User's Permissions and < t?
No
yes
- Assign User to Some Role/s in Combination
- Update t for User
Exit
Assign User to Roles in Combination
Deal with User's Permissions Left to Choose the Optimal Roles
Update UA, PA and R
Create Role for Uncover Permissions

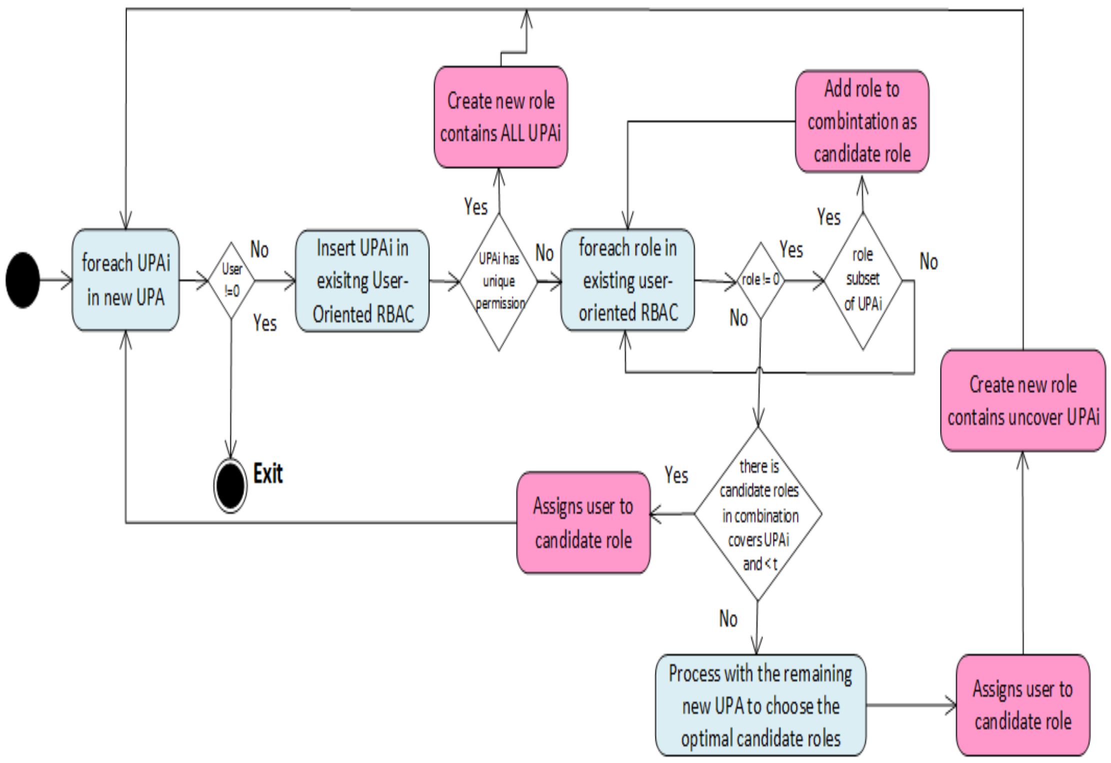
Figure 3. 4: Activity Model for Insert New User (UPA)

In the following an algorithm capture for dynamic user-oriented RMP algorithm which developed by us.

Dynamic User-Oriented RMP Algorithm

```
   Input: UPA, UPAi, PA, UA, t
   Output: PA, UA
1: If $\exists$P' in UPAis.t.P'$\notin$ (P in PA) then
2:   $\forall$ 'P' in UPAi, r $\leftarrow$ {r, P}
3:   Update PA by adding r
4:   Update UA
5: Else
6:   $\forall$ $PA_j$ s.t.$PA_j$/$UPA_i$ = $\emptyset$, CRoles $\leftarrow$ {CRoles, $PA_j$}
7:   $\forall$CRoleSet = {$CRoles_1$ TO $CRoles_t$) s.t. $CRoles_1 \cap CRoles_2$
     $\cap \ldots \cap$ CRolest = $\emptyset$, Combinations $\leftarrow$ {Combinations, CRoleSet}
8:   If $\exists$$Combinations_i$ s.t. $UPA_i$/$Combinations_i$ = $\emptyset$ AND
     |$Combinations_i$| < t then
9:     Update UA by adding $Combinations_i$
10:  Else
11:    $\forall$$UPA_i$ in UPA, s.t. $UPA_i$/$Combinations_i$
12:      CRoles $\leftarrow$ {$Combinations_i$, UPA}
         in $UPA_i$/$Combinations_i$ s.t. |P| is min AND |CRoles| < t
13:    Update PA by adding r
14:    Update UA by adding CRoles
15:  End If
16: End If
```

Figure 3.5 shows our structural model for dynamic user-oriented RMP.



foreach UPAi in new UPA
User !=0
No
Yes
Exit
Insert UPAi in exisitng User-Oriented RBAC
UPAi has unique permission
Yes
Create new role contains ALL UPAi
No
foreach role in existing user-oriented RBAC
role != 0
Yes
No
role subset of UPAi
Yes
Add role to combintation as candidate role
No
Create new role contains uncover UPAi
there is candidate roles in combination covers UPAi and < t
Yes
Assigns user to candidate role
No
Process with the remaining new UPA to choose the optimal candidate roles
Assigns user to candidate role

Figure 3. 5: Structural Model for Dynamic User-Oriented RMP

As shown in figure 3.5, new users with their permissions enter the algorithm one by one. The inputs of our algorithm are:

- UPA: all new users and their permissions.
- UPAi: the current user and their permissions.
- t: the maximum number of roles that each user can has.
- UA: user-role assignments in user-oriented RBAC model.
- PA: role-permission assignments in user-oriented RBAC model.

And the outputs of our algorithm are:

- UA: updated user-role assignments after inserting new users.
- PA: updated role-permission assignments after inserting new users.

The first step of our algorithm is to compare users' permissions with role-permission assignments, and check if the user has a permission that is not covered by any role. In this case, the algorithm creates a role which contains those all permissions and assigns this role to user, this step also applied in [4].

If all user's permissions are covered by roles, then our algorithm creates a combination of candidate roles. It focuses on the existing role-permission assignments and checks if there is a role assigned to permissions and the new user partially or entirely has those permissions (role is not assigned to any other permissions), then add this role-permission assignments to a combination as a candidate role. After that, we will have a combination of candidate roles which the new user may be assigned.

After we have a combination of candidate roles, the algorithm checks if we can create a set of candidate roles from the combination. In this case we have two constraints: The number of candidate roles that doesn't exceed t, and those candidate roles cover completely all permissions for the new users. If those two constraints are applied, the algorithm assigns the new user to this candidate role without creating any new roles for the new user.

The last step is applied if none of the previous steps cover the current user. This step is applied if one of the following two cases is valid. First, if we cannot find a set of candidate roles from the combination to completely cover all new user's permissions (the user still has uncovered permissions). Second, if the number of candidate roles in the set that completely cover all new user's permissions that doesn't exceed t. In these two cases the algorithm creates a new role to include the uncovered permissions, the user should be assigned to some of the candidate roles in the set, but the question here: What candidate roles the algorithm will choose? To answer this question, first we studied some of strategies that are used to choose candidate role such as, [4], [10], and [17].

After we have tried each strategy separately on our algorithm, we found that all of them do not work well in our case. So, we used an alternative strategy: we choose to make the selection of candidate roles dynamically, it depends on what uncovered permissions will be left to utilize the created role which includes these left permissions on the assignments for the remaining users. Using this method, the new user is guaranteed to have all the uncovered permissions, and we reduce the amount of role creation in the system. Finally we will assign the user to the created roles and to the candidate roles that have been chosen from the list.

3.3 Summary

In this chapter, we explained dynamic user-oriented RBAC model (DUO-RBAC) to insert new users with their permissions (UPA) to the existing user-oriented

RBAC model in the system. We described (by using diagrams) how the model works, and the final structure of the model that will be generated after inserting the new users with their permissions.

Also, we explained the algorithm that we used in the model, which is dynamic user-oriented RMP, we defined the preprocess step before the algorithm begin, and defined that the main two concepts of our algorithm are: dynamic role generation and dynamic candidate roles selection. After that, we viewed the activity model for insert new user to the existing model, the pseudo code, and structure model for our algorithm.

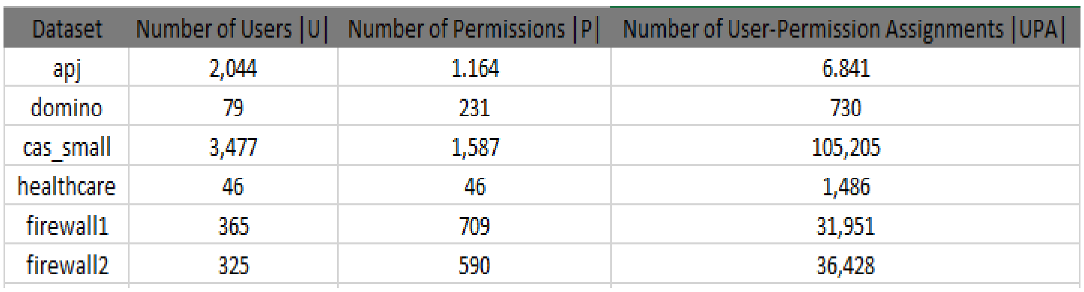
4. EXPERIMENTS AND ANALYSIS OF RESULTS

4.1 Experiments

4.1.1 Introduction

In this research, our experiments applied on benchmark “Access Control” real datasets, those datasets are apj, domino, cas_small, healthcare, firewall1 and firewall2. All those datasets are collected by [10] [4]. Table 4.1 shows the data description for each dataset that includes number of users, number of permissions and number of user-permission assignments.

Table 4. 1: Datasets Description



Dataset	Number of Users |U|	Number of Permissions |P|	Number of User-Permission Assignments |UPA|
apj	2,044	1.164	6.841
domino	79	231	730
cas_small	3,477	1,587	105,205
healthcare	46	46	1,486
firewall1	365	709	31,951
firewall2	325	590	36,428

The cas_small and apj are the authentication for users to access HP network on their profile on Cisco firewalls. The healthcare was obtained from US Veteran’s Administration. The feirewall1 and firewall2 are the result of running an analysis algorithm on Checkpoint firewalls. The domino graph is from a set of user and access profiles for a Lotus Domino server. More details in [4].

Also, to validate the results after the experiments, we depend on an objective function mentioned in chapter 2. As a result, the final RBAC model that generated after inserting new users with their permissions must be user-oriented. So, to evaluate our model we used this objective function to make sure that the number of user-role assignments and number of roles are optimal under constraints of no user assigns to roles that doesn't exceed t [4], and this what the objective function achieve. Also, the role assignments for the existing users must not change.

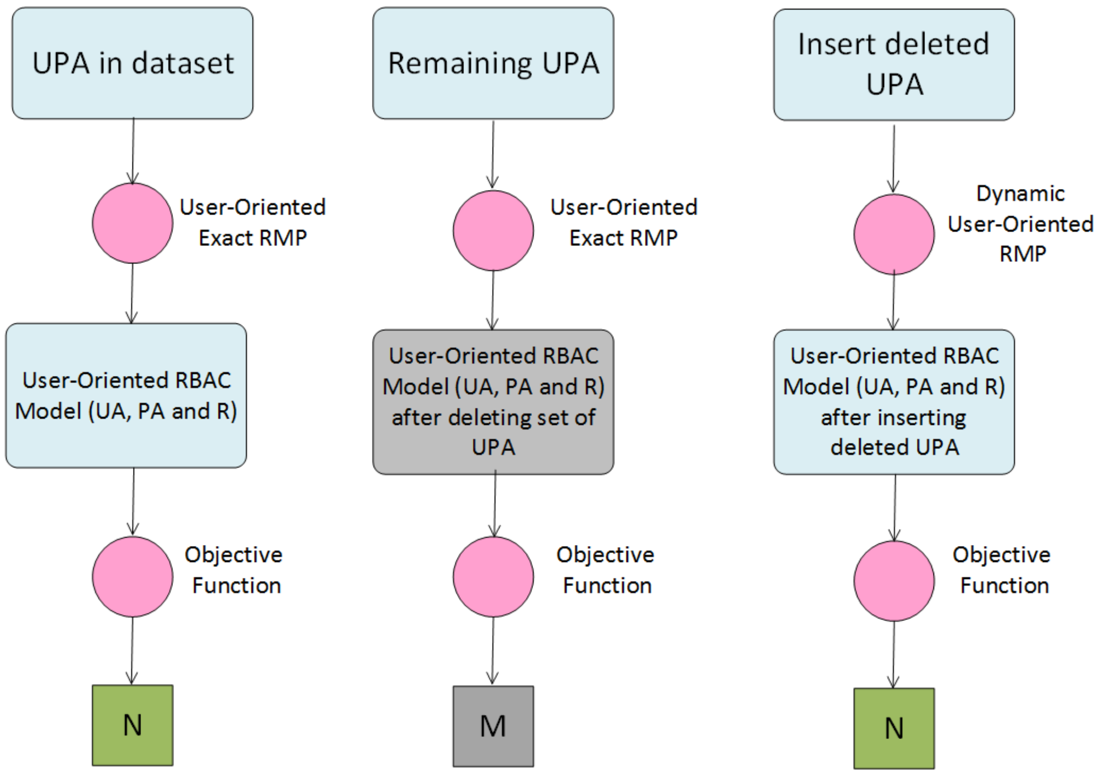

UPA in dataset
User-Oriented Exact RMP
User-Oriented RBAC Model (UA, PA and R)
Objective Function
N
Remaining UPA
User-Oriented Exact RMP
User-Oriented RBAC Model (UA, PA and R) after deleting set of UPA
Objective Function
M
Insert deleted UPA
Dynamic User-Oriented RMP
User-Oriented RBAC Model (UA, PA and R) after inserting deleted UPA
Objective Function
N

Figure 4. 1: Experiments Process

As shown in figure 4.1, our experiments consists of three steps.

First step we applied user-oriented exact RMP on the existing real datasets to generate user-oriented RBAC model which contains roles, user-role assignments and role permission assignments as in [4]. After that, use these components to get the optimal results by using the objective function.

Second step, we removed random number of users with their permissions from those datasets to have remaining data less than the original, then running user-oriented exact RMP on those remaining data to generate its user-oriented RBAC model and get the value from the objective function. The result of the objective function in this step should be optimal in case user-oriented, because we used user-oriented exact RMP, after that we are now having a user-oriented RBAC model, and also we have a new users and their permission (the removed users from datasets).

Final step, inserting the new users (removed users) and their permissions to the existing user-oriented RBAC model to generate the final model. Note that, to evaluate the results of our dynamic model and our dynamic algorithm that depends on user-oriented role mining, the insertion process for the deleted user-permission assignments should be under the following constraints. First one, user-role assignments in the final model should be almost the same as in the model that generated in the first step (before removing users), so, if there are a small changes in the total number of roles or the total number of user-role assignments, the reflection of those changes should not affect the final value of the objective function to be minimized. Second one, the number of user-role assignments for each existing user in the model must still the same as before inserting new users. Last one, be sure that each new user will have role assignments no more than t that predefined in the existing model.

4.1.2 Implementation and Results

As we mentioned before, we must have a user-oriented RBAC model to insert the new user with their permissions (UPA) to the model. So, the first step we did and before running our algorithm is generate the user-oriented RBAC model by running user-oriented exact RMP on the existing datasets. In this research, we used C# as a language to implement user-oriented exact RMP algorithm and to get the same results in [4].

In the beginning, there are many ambiguous steps that we faced (while implementing user-oriented exact RMP) that are not clarified (used as black boxes), in which may cause the results to un-match the ones in [4], and these are the following:

First step when choosing the candidate role. In [4] said that we chose the candidate role which matches as whole the most number of roles, but it didn't clarify the case of multi-equal-matching, which candidate to take?! We tried a small sample that resulted with this same case, and found out that there are two possible results, of course in bigger samples like the benchmarks, it will cause more than that, because while applying the algorithm on each datasets always we have more than candidate roles at each iteration. Though theoretically all of them are correct.

Second step is choosing a (t-1) role. In the algorithm it is written that after any iteration, we take the set of permissions that has only one role to be assigned with (its own t = t-1), we take that role and see if the left uncovered permissions are shared with any other role/s, then we assign them to a single role and continue. Just like the previous issue, if there are multi set of permissions with 1 role left to assign, which role should we take?!

Final step, when the t-1 role doesn't have any matches with the left roles it’s written that we should remove all the assignments from it, and return it to the original permissions, and then assign them in one single role for that user. So the missing point here, should we do only that?, or also look for the assignments that are removed, and also remove them and start all-over again (taking those steps back), because by removing those assignment it might cause a difference in selecting the candidate role in the first place, (choosing another candidate in that step which was cancelled), or should we go forward? What we did was going forward, we did not take back steps, and we couldn't say it’s the reason that giving us the different results, because as we mentioned in the first two points, there are many possible solutions based on decision making.
After fixed the previous ambiguous steps, we got the same results as ones in [4]. The following are the results of generating user-oriented RBAC model for each dataset.

Table 4. 2: Americas small

t	R	UA	UA+PA
2	259	3477	25229
4	256	3722	23610
6	249	3890	22194
8	246	4269	20119

Table 4. 3: fire1

t	R	UA	UA+PA
2	90	365	7100
4	85	454	6890
6	84	600	6897
8	80	1516	6638

Table 4. 4: fire2

t	R	UA	UA+PA
2	11	325	1499

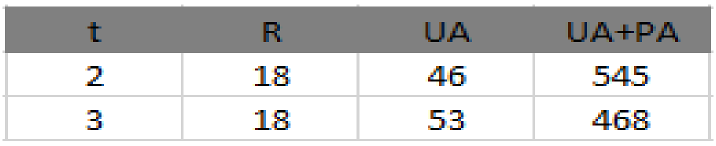
Table 4. 5: domino

t	R	UA	UA+PA
2	23	79	716

Table 4. 6: apj

t	R	UA	UA+PA
2	564	2044	5565
3	497	2218	5221
4	485	2277	5096

Table 4. 7: healthcare



t	R	UA	UA+PA
2	18	46	545
3	18	53	468

Tables above specify the number of roles (R), user-role assignments (UA), role-permission assignments (PA) and the summation of UA and PA in different values of t at each dataset after generating user-oriented RBAC model by running user-oriented exact RMP algorithm.

Also, it illustrated that the number of roles (R) inversely proportional with the value of t, since the minimum number of R was in case that t is maximized, and the maximum number of R was in case that t minimized. On the other hand, the number of user-role-assignments (UA) directly proportional with the value of t, since the minimum number of UA was in case that is minimized and the maximum number of UA was in case that t is maximized.

After that, we implement our algorithm by using the same language as before (C#). In the following the process of how we did the experiments by using dynamic user-oriented RMP, and the results that we got at each dataset.

First step we did is deleting a random number users with their permission assignments from each the original dataset. We deleted 50 users from apj, 15 and 22 users from domino, 65 and 81 users from cas_small, 10 and 14 users from healthcare, and 25 and 36 users from both firewall1 and firewall2, as well as deleting all their permission assignments. Second step is running user-oriented exact RMP algorithm at each dataset that contains the remaining user-permission assignments (the remaining data after the deletion process from each dataset), then we will have a new generated user-oriented RBAC model for each dataset. Third step is using our dynamic algorithm to insert the deleted users and their permissions assignments to the existing user-oriented RBAC model to have final model that covers all users-permission assignments as in the original datasets. In the following the results of number of roles (R), user-role assignments (UA) and role-permission assignments (PA) for each dataset after inserting the deleted users and their permission assignments to the existing model.

Table 4. 8: **Americas small**

t	Deleted Users	R	UA	UA+PA
4	65	251	3729	25249
8	65	244	3281	20193
4	81	256	3729	25249
8	81	245	4112	21029

Table 4. 9: **apj**

t	Deleted Users	R	UA	UA+PA
2	50	563	2050	5596
3	50	495	2221	5240
4	50	480	2284	5149

Table 4. 10: **domino**

t	Deleted Users	R	UA	UA+PA
2	15	23	79	716
2	22	23	79	716

Table 4. 11: **fire1**

t	Deleted Users	R	UA	UA+PA
2	25	90	365	7100
6	25	80	605	6936
2	36	90	366	7102
6	36	82	605	6938

Table 4. 12: **fire2**

t	Deleted Users	R	UA	UA+PA
2	25	10	342	1519
2	36	11	331	1500

Table 4. 13: **healthcare**

t	Deleted Users	R	UA	UA+PA
2	10	18	46	545
3	10	16	53	481
2	14	18	46	545
3	14	15	58	490

Tables above show the number of roles (R), user-role assignments (UA), role-permission assignments (PA) and the summation of UA and PA in different values of t at each dataset after generating the dynamic user-oriented RBAC model (DUO-RBAC) by running our dynamic algorithm.

Also, the tables show that the number of roles (R) inversely proportional with the value of t, since the minimum number of R was in case that t is maximized, and the maximum number of R was in case that t minimized. On the other hand, the number of user-role-assignments (UA) directly proportional with the value of t, since the minimum number of UA was in case that is minimized and the maximum number of UA was in case that t is maximized.

4.2 Analysis of Results

4.2.1 Introduction

In the beginning, we will introduce our main contribution from developing dynamic user-oriented RBAC model using our dynamic algorithm. Our main contribution is inserting new users with their permission assignments (new UPA) to the system with user-oriented RBAC model without any changing on the number of role assignments for each user in the system, this mean that the number user-role assignments for each user in the system must still the same after inserting the new UPA to the system.

Also, the total number of roles (R) and total number of user-role assignments (UA) almost similar or better than the total number roles (R) and total number of user-role assignments (UA) before inserting new UPA. So, after applying the total number of R and UA on the objective function, the final result will be minimized. However, changing the number of role-permission assignments (PA) after the insertion process is not important, since our model is user-oriented and the user does not care about the changes in role-permissions assignments while he still has the same permission to access the system, but he cares if his role assignments number changed because the user assigned directly to role/s and he make his activities in the system through his roles.

4.2.2 Analysis of Example Results

Table 4. 14: Compare the Results after Inserting the New UPA Using Two Algorithms

	Dynamic User-Oriented RMP	User-Oriented RMP
Nuber of t	2	2
Number of roles (R)	8	8
Number of User-Role Assignments (UA)	12	14
Number of Role-Permission Assignments (PA)	22	19

Table 4.17, shows a comparison between the results after inserting new UPA by rerun user-oriented exact RMP algorithm and after inserting new UPA by our dynamic algorithm. If look on the table we will find that the two algorithm used t = 2 (the maximum number of role assignments for each user in the system), that mean if we are looking to figure 4.5 and figure 4.6 we will see that each user assigned to roles less than 2, also the two algorithm are completely reconstruct the user-permission assignments in the system, this means that the all users in the system must have the same permissions that he has before inserting the new user-permission assignments to the system (there are no permissions lost or added to his original permissions). Now, we have three observations while comparing between the two results that we had and

check differences between them. The first observation is the total number of roles (R) that generated in the model, the second observation is the total number of user-role assignments in the model and the third observation is find if there is any changed of role assignments for each user in the system.

The total number of roles that returned by running user-oriented exact RMP algorithm is 18, and the total number of roles that returned by running our dynamic algorithm on the existing user-oriented RBAC model is 18, by this case the two results are equal. The total number of user-role assignments (UA) that returned by running user-oriented exact RMP algorithm is 14, and the total number of user-role assignments (UA) that returned by running our dynamic algorithm on the existing user-oriented RBAC model is 12.

However, the total number of user-role assignments that returned by our algorithm is less than the total number of user-role assignments that returned by the other one, here we demonstrate that results of our algorithm in this case better than the other algorithm, since in case user-oriented is better to have a total number of user-role assignments as little as possible.

The last observation is to find out if there is any changing in the assignments for each user in the system after inserting then new user-permission assignments (new UPA) to the system. In case rerunning user-oriented exact RMP algorithm, figure 4.3 shows that user3 has one role assignment which is role4, and user5 also has one role-assignment which is role5, but figure 4.5 shows that the number of role-assignments for user3 become 2 which are role1 and role6, also the number of role-assignments for user5 become 2, that’s mean the number of role assignments for both users after inserting the new user-permission assignments are changed from 1 to 2. In case running our dynamic algorithm on the existing user-oriented RBAC model, and if we are looking to figure 4.3 and figure 4.6 we will see that all existing users have the same number of role assignments after inserting the new user-permission assignments to the system without any changes.

So, our algorithm make sure that no changes in the number of role assignments for each user in the system after the insertion process, but there are changes in the role assignments for some users in the system when we applied the other algorithm, which is bad from term user-oriented because each time we are inserting a new user-permission assignments to the system the user will has a different role assignments. And to make the generated model user-oriented, user does not prefer to change his role-assignment number from time to time.

4.2.3 Analysis of Benchmark Results

After analyzing the results that we got in the example. In this section we will analyze the results that we got after inserting new user-permission assignments by running our dynamic algorithm and rerunning user-oriented exact RMP algorithm in the benchmark.

This experiment is centered on deleting some users with their permissions from each benchmark file, then applying user-oriented exact RMP algorithm on the remaining user-permission assignments at each file to get user-oriented RBAC model, after that inserting the deleted users with their permissions to the model by running our dynamic algorithm. Figure 4.1 shows the experiment process. In the following a comparison between the results that we got after inserting the deleted users with their permissions to the model for each file in the benchmarks and the results in [4].

Table 4. 15: Americas small

		Dynamic User-Oriented RMP		User-Oriented Exact RMP	
t	Deleted Users	R	UA	R	UA
4	65	251	3729	256	3722
8	65	244	3281	246	4269
4	81	256	3729	256	3722
8	81	245	4112	246	4269

Table 4. 16: apj

		Dynamic User-Oriented Exact RMP		User-Oriented Exact RMP	
t	Deleted Users	R	UA	R	UA
2	50	563	2050	564	2044
3	50	495	2221	497	2218
4	50	480	2284	485	2277

Table 4. 17: domino

		Dynamic User-Oriented RMP		User-Oriented Exact RMP	
t	Deleted Users	R	UA	R	UA
2	15	23	79	23	79
2	22	23	79	23	79

Table 4. 18: fire1

		Dynamic User-Oriented RMP		User-Oriented Exact RMP	
t	Deleted Users	R	UA	R	UA
2	25	90	365	90	365
6	25	80	605	84	600
2	36	90	366	90	365
6	36	82	605	84	600

Table 4. 19: fire2

		Dynamic User-Oriented RMP		User-Oriented Exact RMP	
t	Deleted Users	R	UA	R	UA
2	25	10	342	11	325
2	36	11	331	11	325

Table 4. 20: healthcare

		Dynamic User-Oriented Exact RMP		User-Oriented Exact RMP	
t	Deleted Users	R	UA	R	UA
2	10	18	46	18	46
3	10	16	53	18	53
2	14	18	46	18	46
3	14	15	58	18	53

The tables above show a comparison between the total number of roles and the total number of user-role assignments after inserting the deleted user with their permissions at each file in the benchmark with different t. In some files our algorithm returned the same results as tables 4.17 where deleted users=15, 4.18 where deleted users=25, and 4.20 where deleted users=10 in case where t=2, or better than the other algorithm as shown in table 4.20 where deleted users=10 and t=3, table 4.18 where deleted users=25 and t=6, and table 4.16 where deleted users=50 and t=4, while in other files the results are too closed without a huge effectiveness for the results on the value of the objective function. So, after we will apply our results on the same objective function, the values are approximately equal. Also, by our algorithm, now we are make sure that each user in the system after the insertion process will still has the same number of role assignments as before without any changes.

The performance time in our case does not extremely matter since we trend to user-oriented more than performance wise. Although, the performance of our algorithm is good.

4.3 Summary

In this chapter, we made the experiments to inserting a new users with their permissions to system which has a user-oriented RBAC model by running our dynamic algorithm that depends on user-oriented role mining, then validate the returned results by comparing them with the results in [4].

Our experiments based on using a benchmark access control dataset. Also, we introduced an example in chapter 1 to prove the changes on the number of roles assignments for some users and how our solution kept the same number of role assignments for all users the same.

5. SUMMARY AND FUTURE WORK

This chapter concludes the research. A review of the importance of the model is presented with focus on the main contribution, result, limitations and possible future work.

In this research, a dynamic user-oriented role-based access control model (DUO-RBAC) was designed, and a dynamic algorithm that depends on user-oriented role mining was developed. The DUO-RBAC is a complete model aimed to insert the new user-permission assignments (new UPA) to the existing model by using the dynamic algorithm to the existing model under two constraints which are makes our designed model more efficient than the existing ways.

The rest of this chapter presents the conclusion, including contribution, summary of results and limitations. It also presents potential research areas for future work.

5.1 Contribution

Our dynamic model depends on using the concept of role mining to find the optimal way for inserting this new UPA to the existing model by running our dynamic algorithm. Unlike the available ways for inserting new UPA, our algorithm keep the number of role assignments for each user in the system after the insertion process the same as they are before the insertion process, also the user will keeping has the same roles without any changes, by this way the concept of user-oriented achieved and the model is still suitable from end-user perspective.

In this research, we discussed the only two ways to insert the new user-permission assignments (UPA) to the generated model and determined the limitations for each one in different case.

Compare to our work, the developed algorithm achieved the optimal total number of roles and total user-role assignments in the generated model after the insertion process. On other hand, the total number of role-permission assignments not matter of us, since our study aimed to generate an RBAC model from end-user perspective and not for reducing the administrative cost on generating model.

5.2 Results

Our work is evaluated by generating our dynamic user-oriented RBAC model after inserting the new users with their permissions to system which has a user-oriented RBAC model by our running dynamic user-oriented RMP algorithm. Our experiments based on using a benchmark access control datasets.

The dynamic model that generated at each dataset contains a new number of roles, user-role assignments and role-permission assignments. To validate our returned results, we made a comparison between them and the results in [4]. The number of user-role assignments in the dynamic model at each dataset was less than the original one, and the number of roles are equal or more than the original one. By using this method, the evaluating function still has the same value at each case which is the optimal value in case user-oriented. Also, our experiment make sure that each user in the system still has the same number of role assignments which achieve and keep the model on the system suitable to term user-oriented.

In addition, we introduced an example to show the changes on the number of roles assignments for some users and how our solution kept the same number of role assignments for all users the same.

5.3 Limitations

Our work focused on using user-oriented role mining to generate role-based access control model from end-user perspective. In this case we focus on the total number of roles and the total number of user-role assignments, but we do not care about the total number of role-permission assignments. The total number of role-permissions assignments that returned in the dynamic model was more than the ones that returned in the original modal, by using this method, the administrative cost and the complicity if role hierarchy do not minimized.

Performance wise, the time for insertion process of the new user-permission assignments to the existing system by using the dynamic algorithm to generate the dynamic model is more than the time that needed to insert the new user-permission assignments by regenerate the model after insert the new UPA to the original dataset (introduced in chapter 1). But if we use regenerating method, the concept of user-oriented will not achieved in the model since there will be changes on the number of role assignments for some users in the system.

5.4 Future Work

In the future work, can design another dynamic model for inserting new UPA to the existing model in case the existing model in the system not a user-oriented model but it was for reducing the administration cost. A dynamic algorithm will developed by using the concept of role mining, but the constraints will be focused on the total number of roles (R) and the total number of role-permission assignments (PA).

REFERENCES

[1] Ravi S. Sandhu, Edward J. Coynek, Hal L. Feinsteink and Charles E. Youmank, (1995). Role-Based Access Control Models. IEEE Computer, Volume 29, Number 2, pages 38-47.

[2] E.J. Coyne, (1996). Role engineering. In Proc. of the ACM Workshop on Role-Based Access Control.

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

[4] Haibing Lu, Yuan Hong, Yanjiang Yang, Lian Duan, Nazia Badar, (2015). Towards User-Oriented RBAC Model. Journal of Computer Security - Data and Applications Security, Volume 23 Issue 1, Pages 107-129.

[5] Mario Frank, Joachim M. Buhmann, David Basin, (2010). On the Definition of Role Mining. SACMAT’10, Proceedings of the 15th ACM symposium on access control models and technologies.

[6] Ian Molloy, Hong Chen, Tiancheng Li, Qihua Wang, Ninghui Li, Elisa Bertino, (2008). Mining Roles with Semantic Meanings. SACMAT’08, Estes Park, Colorado, USA.

[7] Martin Kuhlmann, Dalia Shohat, Gerhard Schimpf, (2003). Role Mining - Revealing Business Roles for Security Administration using Data Mining Technology. SACMAT'03, Como, Italy.

[8] Mario Frank, David Basin, Joachim M. Buhmann, (2008). A Class of Probabilistic Models for Role Engineering. CCS'08, Alexandria, Virginia, USA.

[9] Gustaf Neumann and Mark Strembeck, (2002). A Scenario-driven Role Engineering Process for Functional RBAC Roles. SACMAT'02, Monterey, California, USA.

[10] Alina Ene, William Horne, Nikola Milosavljevic, Prasad Rao, Robert Schreiber, Robert E. Tarjan, (2008). Fast Exact and Heuristic Methods for Role Minimization Problems. SACMAT Estes Park, Colorado.

[11] Xuan Hung Le, Terry Doll, Monica Barbosu, Amneris Luque, Dongwen Wang, (2012). An enhancement of the Role-Based Access Control model to facilitate information access management in context of team collaboration and workflow. Journal of Biomedical Informatics, Volume 45, Issue 6, Pages 1084–1107.

[12] Jaideep Vaidya, Vijayalakshmi Atluri, Qi Guo, (2007). The Role Mining Problem: Finding a Minimal Descriptive Set of Roles. SACMAT'07, Sophia Antipolis, France.

[13] Xiaopu Ma, Ruixuan Li, Zhengding Lu, (2010). Role Mining Based on Weights. SACMAT’10, Pittsburgh, Pennsylvania, USA.

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

[15] Randike Gajanayake, Renato Iannella, Tony Sahama, (2012). Privacy Oriented Access Control for Electronic Health Records. DUMW2012, Lyon, France.

[16] Shuo Lu, Yuan Hong, Qian Liu, Lingyu Wang, Rachida Dssouli, (2007). Access Control in e-Health Portal Systems. IEEE, Innovations in Information Technology. IIT '07. 4th International Conference.

[17] I. Molloy, H. Chen, T. Li, Q. Wang, N. Li, E. Bertino, S. B. Calo, and J. Lobo, (2010). Mining roles with multiple objectives. ACM Trans. Inf. Syst. Secur.., 13(4):36.

[18] Jaideep Vaidya, Vijayalakshmi Atluri, Janice Warner, (2006). RoleMiner: Mining Roles using Subset Enumeration. CCS’06, October 30–November 3, 2006, Alexandria, Virginia, USA.

بناء نموذج التحكم في الصلاحيات القائم على الدور لكل مستخدم بشكل ديناميكي من وجهة نظر المستخدم (DUO–RBAC)

إعداد: حازم مالك حسني كيوان

إشراف: د.رشيد جيوسي

الملخص

أغلب الأبحات في الوقت الحالي تتجه الى استخدام استكشاف الأدوار لتوفير نموذج للتحكم في الصلاحيات القائم على الدور لكل مستخدم من خلال مجموعة من الصلاحيات المعينة مسبقا لكل مستخدم في النظام. نموذج التحكم في الصلاحيات القائم على الدور لكل مستخدم من وجهة نظر المستخدم بشكل عام هو أحد أنواع التحكم في الصلاحيات القائم على الدور لكل مستخدم والذي يهدف لاستخدام استكشاف الأدوار من وجهة نظر وتوقع المستخدم لانتاج نموذج ملائم لما يتوقعه المستخدم بشكل تقريبي, حيث أن المستخدم بشكل عام دائما ما يفضل امتلاك عدد قليل وبسيط من الأدوار خلال استخدامه النظام. هذا البحث هو الأول من نوعه والذي يوفر نموذج للتحكم في الصلاحيات القائم على الدور لكل مستخدم بشكل ديناميكي من وجهة نظر المستخدم (DUO–RBAC) لادخال مجموعة من المتسخدمين الجدد والصلاحيات لكل منهم للنموذج الموجود مسبقا في النظام.

اعادة استكشاف الأدوار لكل مستخدم والذي سوف يتم عن طريق النموذج الديناميكي الذي قمنا بتطويره سيعتمد بشكل أساسي على ثلاث شروط: الأول, أن نكون واثقين أنه لا يوجد أي تغير بعدد الأدوار التي يمتلكها كل مستخدم بعد عملية ادخال المستخدمين الجدد الى النظام, والثاني هو أن الصلاحيات التي يمتلكها المستخدم قبل عملية الأدخال يجب أن تكون نفس الصلاحيات بعد عملية الأدخال من غير أي تغيير, أما الثالث فيجب أن نأخذ بعين الأعتبار أن كل مستخدم لا يمكن أن يمتلك عدد أدوار أكثر من t وهو العدد الأقصى من الأدوار التي يمكن لكل مستخدم

في النظام أن يمتلكها, وتكون معرفة في النموذج الموجود بالنظام والذي سيتم ادخال المستخدمين الجدد عليه.

وأيضا, لقد قمنا بتطوير خوارزمية جديدة تعتمد على استكشاف الأدوار من وجهة نظر المستخدم لايجاد أفضل طريق لادخال المستخدمين الجدد مع الصلاحيات الخاصة بكل منهم للمنوذج الموجود في النظام.

من جهة أخرى, لقد قمنا بعمل التجارب الخاصة بنا على مجموعة من البيانات الحقيقية (benchmark) والخاصة بالتحكم والوصول (access control) لتقييم وأختبار النتائج التي حصلنا عليها واظهار مدى فعالية وانتاجية النموذج الذي قمنا بتطويره من عدة نواحي.