A Proposed Security Model for Web Enabled Business Process Management System

M. S. Kandil arwaahmed1@gmail.com
Faculty of Engineering
Mansoura University

Mohamed Abu El-Soud a_m_elhady@yahoo.com
Faculty of Computer & Information Sciences
Mansoura University

A. E. Hassan arwaahmed1@gmail.com
Faculty of Engineering
Mansoura University

Abd elghafar M. Elhady a_m_elhady@yahoo.com
Faculty of Computer & Information Sciences
Mansoura University

Abstract

Business Process Management systems (BPMS) and technologies are currently used in many organizations' IT applications. This could lead to a dramatic operational efficiency improvement on their business and administrative environments. With these atmospheres, the security issue is becoming a much more important challenge in the BPMS literature. The Role-Based Access Control (RBAC) model has been accepted as a promise security model solution and standard. RBAC is able to accomplish the central administration of an organizational specific security policy. It is also able to meet the secure processing needs of many commercial and civilian government organizations. In spite of these facts, RBAC model is not reliable when applying to the BPMS without further modifications and extensions. RBAC is modified to fit with Service oriented (SRBAC), but still not reliable enough to handle BPMS.

Authors of that research proposed a security model based on SRBAC model to be more reliable when using with BPMS. Authors of that research named that proposed security model as Improved Role Based Access Control (IRBAC). The IRBAC model is directly applicable to the BPMS.

Authors defined a graphical representation and technical implementation of the IRBAC model.

This IRBAC model is tested using simple case study. The test compares between the IRBAC model and SRBAC model where IRBAC is implemented in two cases (IRBAC with caching and IRBAC with no caching). The test results show the validity and performability of the IRBAC model.

1. INTRODUCTION

Since the beginning of the shift from a functional to a process-centered view of business activities in the 80s [2], business processes play a major role in today's companies. BPMS is applied to "analyze and continually improve fundamental activities such as manufacturing, marketing, communications and other major elements of a company's operations" [3]. In other words, it is applied to engineer lean and streamlined business processes [2]. The introduction of BPMS has several benefits such as cost reduction, quality improvements and error reduction, visibility gain, and process step automation [4]. In recent years, business processes are often the target of security hazards, such as viruses, hacker attacks, or data theft [5,6].

Because business processes generate valuable information and knowledge as output, decision makers and security experts need to improve methods to secure them against external or internal attacks. These attacks could result in demand and loss of value for system and organization. These damages can be monetary loss (e.g., loss of profit due to the interruption of business activities) and/or intangible value loss (e.g., loss of reputation).

The Data stores detailed information of a organization, and Business Processes that are Performed in the organization's System should be protected. When a user connect to the system, the environment (Data/Business Processes) Created For the user should be ensured in. In order to solve the above issues, adaptive access control is necessary to make sure of the information security of Business Process.

RBAC has become a widely accepted mechanism for security management [7]. RBAC uses the assignment between users, roles and permissions to provide a more convenient access control management model. However, the traditional RBAC does not consider the user’s current environment. It merely bases on the predefined role and permission plan. Some research has combined RBAC with BPMS to achieve dynamic authorization [8,9,10,11]. Nevertheless, most of research with BPMS adopts a Model to use RBAC Methodology with BPMS. These Models have some shortages. Examples of these shortages are that some of these models didn’t present the most optimum solution of applying RBAC with BPMS. Also, they didn't present a complete implantation of this combination.

Traditional security systems with BPMS didn’t secure the system. Dey et al in [9] stated that in February 2000, a Denial of Service (DoS) attack caused access problems of Yahoo’s website, costing an estimated half a million US Dollars in just three hours. The consequence is an ever increasing amount of money on improving security (from 1999 to 2000, the number of organizations spending more than $ 1 million annually on security nearly doubled, representing 12% of all organizations in 1999 to 23% in 2000 [12]). The main problem with security - in this context information security is the lacking integration of security considerations into business processes [13].

Therefore, appropriate access control will improve the feasibility of using BPMS technology in Organizations.

Authors of that research proposed a hybrid model which modified SRBAC model to achieve a dynamic authorization security model (IRBAC).

IRBAC model is proposed in two cases. First case when IRBAC is combined with caching. And the second case when IRBAC is proposed with no caching. The proposed model is tested in the two cases and results are compared with results of SRBAC model.

This proposed model is a generic security model. This model could be added to any BPMS and handle the authorization of system’s users.

2. RELATED WORK

Access control and authorization concerns are one of the key challenges preventing BPM gaining widespread recognition. Firstly, it is not realizable to apply role based model to business process systems directly. Moreover, the inter-organization business process scenario becomes more complicated. For instance, the inherited roles might be stored remotely and permissions constraints will consequently require several remote invocations [14].

Although the concept of role has existed for a long time in systems security, the work presented by Sandhu et al in [15] has prompted a renewed interest in this approach. But proposed model that greatly simplifies security management is presented in [16]. RBAC model is now adopted in many commercial products to different degrees since access control is an important requirement of information systems. RBAC was found to be the most attractive solution for providing security characteristics in inter-organizational business systems [17]. Moreover, it would be much easier for organizations to enhance security protection from existing RBAC based policies.

David F.Ferraiolo et al in [18] and Ravi S.Sandhu et al in [15] define **Traditional RBAC Model** as a model composed of three components:

- A user is a human being belongs to an organization.
- A role is a named job function within the business process context that regards the authority and responsibility.
- A permission is an approval of actions granted to specific roles. A constraint regulates the relations between different elements.

In this model, the central notion is that permissions are associated with roles, and users are assigned to appropriate roles. This greatly simplifies management of permissions. It is suitable for simple Web applications. But in more advanced web applications such as BPMS and Service Oriented Architecture (SOA) applications, traditional RBAC is not suitable for them. Moreover traditional RBAC can not completely express dynamic characters of role according to what is mentioned in [1].

Xin Wang et al in [19] added a service element to original RBAC model and proposed a new model called Extended RBAC Model, which indicates the Web service deployed within the enterprise system and divided roles into human role and computer role. The human role indicates the tasks to be performed by human users, while the computer role indicates the tasks to be performed by Web services. This model extension addresses the SOA upgrade in this kind of progressive manner.

In [19], authors rely on role hierarchy which causes shortages in system performance. To access a specific service, role server could be accessed more than one time to get role which contain permissions for that user on this service, which causes more network traffic and less overall system performance. Authors divided the system operations into two types, one is performed by users and other is performed by web services. Also, Authors ignore the relation between web services and users of the system, in other words authors didn’t define how user can fire web services that perform specific functions in the system.

Another system proposed in [1] is called a Service-oriented Role Based Access Control (SRBAC) model in which, traditional protected objects are replaced by services, and a new notion of actor is introduced. An Actor is a dynamic object which is created when a user activates a role. Its condition and action may present the characters of the role activated.

In this model, Roles are organized in role Hierarchy. This causes system performance decreasing by causing more network traffic and less overall system performance as mentioned in previous model. Moreover, authors were rely on creating actor for user each time he accesses new role that contains the services he needs. This makes user has to switches among actors to manages services that spreading across more than one role. This scenario was designed to reflect the dynamic execution process of the role. They proposed that roles are dynamic continuously but in most systems, this state can exists at beginning of system building and deployment and rarely happened after that, along system life.

Authors of that research used SRBAC model after modifying it and proposed a new security model called IRBAC. The IRBAC model is the modified SRBAC that has two cases, first combines it with caching technique and second case uses no caching.

3. PROPOSED IRBAC MODEL

In this section, the IRBAC model will be presented. Several IT technologies are combined to provide a dynamic, fast, and secured mechanism for accessing system processes in the model. The implementation of the IRBAC model is presented. The IRBAC model is considered as a Generic Security model which used BPMS principles and could be applicable on any BPMS to manage the authentication and authorization of users on BPMS.

The IRBAC model rely on using the RBAC model in BPMS to improve the security of the system and provide a dynamic management environment for roles /permissions / users assignment which enable system user to adapt role and permission according to any changes happened in the system authorization.

The IRBAC model has two cases:

- First case uses caching strategy to decrease the overall response time experienced by the user when he/she is interacting with the system thus increase system Performance. Where authors utilized from the tests have been made by Kohler et al in [20] on using caching strategy in Business Process-driven Environments which results that using caching in Business Process-driven Environments decrease the response time of user requests significantly thus improve increase the overall system performance.
- The second case depends on that there are some systems has many changes happened to roles' permissions during the operating of users on the system. In this case cashing technique is not suitable with the system needs whoever it is better in performance. So IRBAC model uses no caching to meet the operational needs of these systems.

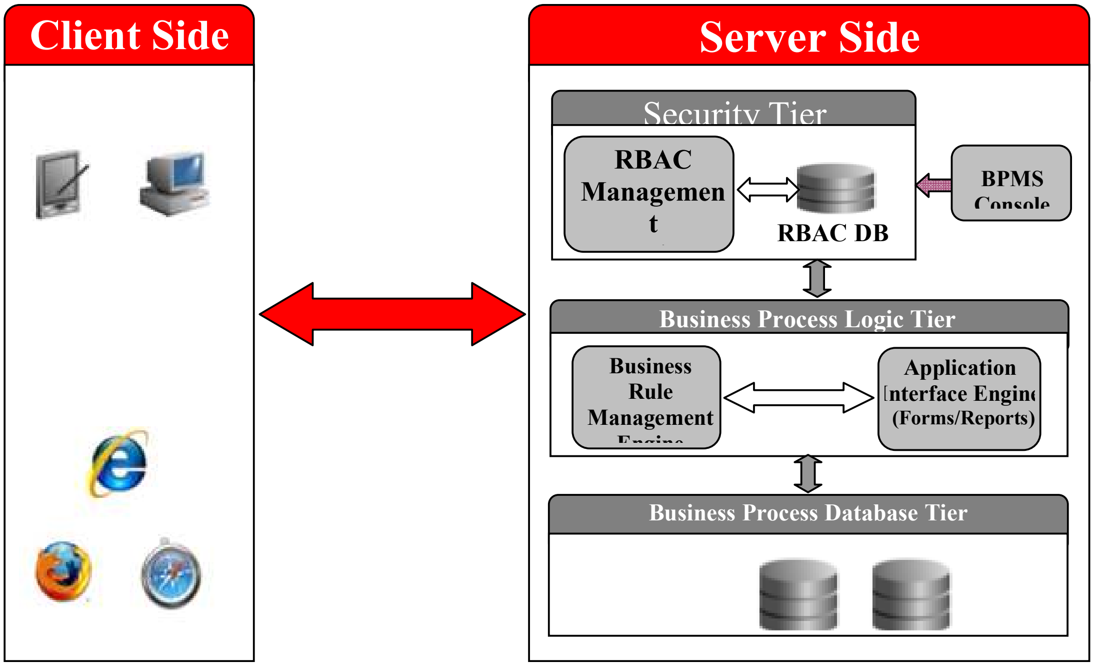

Client Side
Server Side
Security Tier
RBAC Managemen t
RBAC DB
BPMS Console
Business Process Logic Tier
Business Rule Management Engine
Application Interface Engine (Forms/Reports)
Business Process Database Tier

FIGURE 1: Proposed Model Architecture Diagram (Client / Server) N-Tier

The client / server architecture of proposed model is presented in figure (1), in which **client Side** can be computer with browser from which user can access to the BPMS .the Server Side Consists of three main components.

- **Security Tier**: which responsible on verifying the authentication and authorization of users which are dealing with the BPMS. It also detects if any changes happened to the system's processes and perform appropriate action to adapt the security tier of the BPMS.

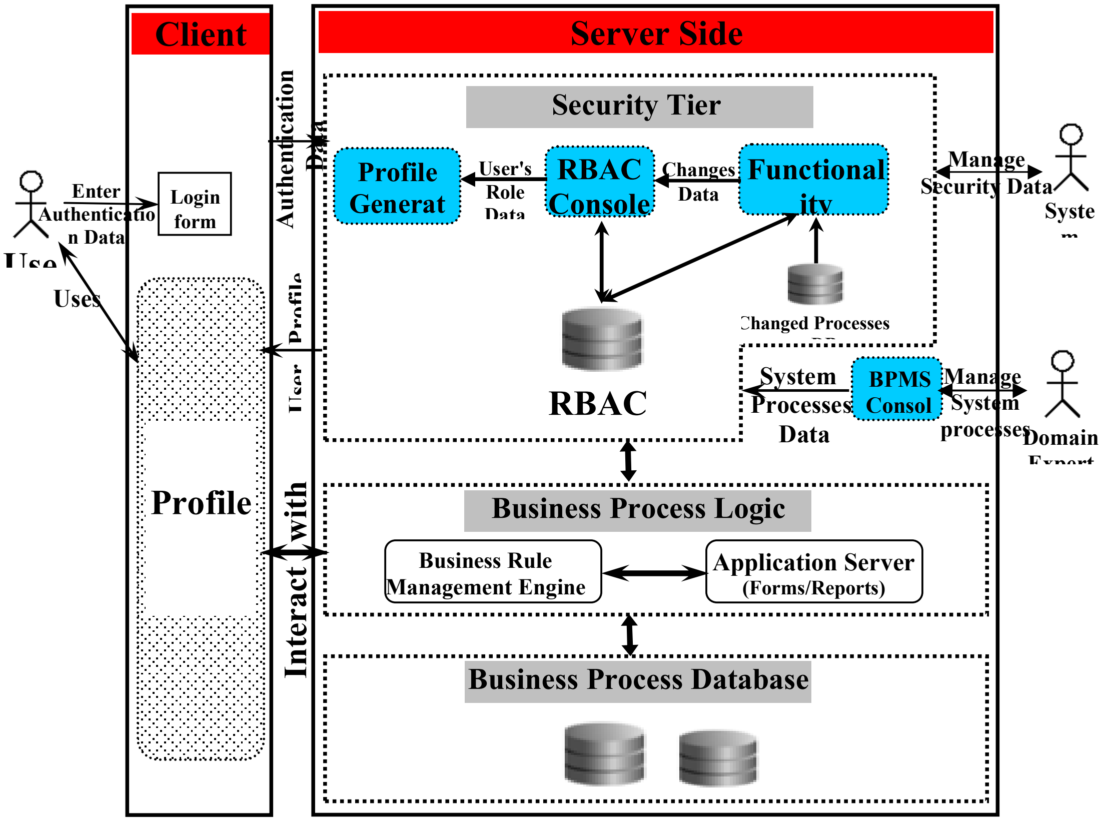
- **Business Process Logic Tier**: This maintains logic of the business system and rules which organize it in Business Rule Management Engine and Application forms and reports in Application Interface Engine.
- **Business Process Database Tier**: In which all Organization's data and information used in the system are stored in Databases.

In addition to these three tier authors design additional component called BPMS Console. By this component any system processes can be configured and/or reconfigure and the output file is delivered to the actual BPS to activate the changes.

3.1 Schematic Diagram of the Proposed Model (N-tier)

In previous section authors present a general view of the proposed model and its components in brief as client/server architecture model. Here, Authors represent the proposed model in more details as schematic diagram. as shown in figure (2), this model consists of four tiers; Client Tier, Security Tier, Business Process Logic Tier, and Business process Database Tier in addition to BPMS Console component. Authors satisfy with what they presented about Process Logic Tier, and Business process Database Tier and will focus in this section on other two Tiers which compromise the core of their work in this research.

FIGURE 2: Schematic Diagram of the Proposed Model



- **Client Tier:** through which any User of the system can access the BPMS According to his authorization where user enter his authenticated data which it send to security tier and accept his profile on his client machine . With this profile, user can deal with the system processes without any need to access security tier to get his authorization data on called processes in case of using caching technique. But in the other case, with no cashing, user profile has to connect to security tier to get the last permissions of user on the calling process.

- **Security Tier:** which is responsible for applying RBAC model to BPMS and it consists of three main components :

- **RBAC Console :** by which system administrator uses to do the following tasks:
  - Creates new Roles and/or manages existing roles.
  - Specifies the system processes' permissions such as {Insert,Update,Delete,Read,Print} to all system processes for each role.
  - Creates and/or manages user data and specifies users to their appropriate role according to their responsibilities and authorities.
  - Determines user's available processes and his permissions and path them to Profile Generator at login phase.

Authors will explain the functionality of this component, its objects, and its interaction with other components of the system in proposed security model in the following section.

- **Profile Generator**: it captures the list of all system processes and user's permissions on these processes and generates complete profile and sends it to user (client side) in caching case. But in no caching case, it captures the list of all system processes authorized to user and generate summarized profile and send it to user.
- **Functionality Adapter**: it is one of the most important components in the proposed model. Because Continuous Process Improvement is a critical feature that is must be met in BPMS. And where the proposed model was designed for running on BPMS. This leads to provide security model that can accept any changes can be happened in BPMS such as adding new processes, deleting existing processes, merging between processes and etc.

  This component is responsible on checking the system processes at login of system administrator. If any changes happened, it will update list of system processes and inform system administrator to update roles' permissions for changes processes.

3.2 Detailed explanation of IRBAC.

In this section, Authors will explain the modifications add to the IRBAC model versus the SRBAC model and how authors use caching technique to improve the BPMS performance in the first case. And how they use no caching to meet the operational needs of the BPMS in second case.

Figure (3) shows the SRBAC model, in which, access control is implemented by control the actor, which is a dynamic object that is created when a user activates a role and to maintains the role's characters and functions. When a user activates a role, an actor is created. This actor is acts as a user proxy through which user interacts with the services. A user may activate many roles, and then the user has the same number actors corresponding to these roles.

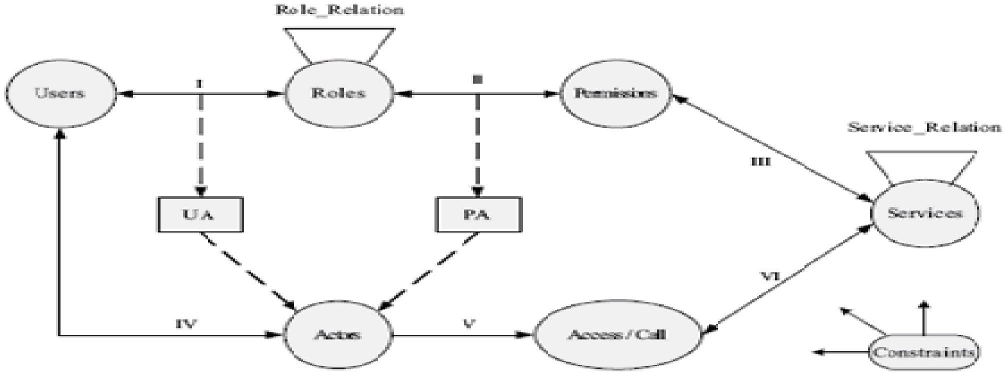


FIGURE 3: SRBAC Model [1]

In SRBAC model, authors proposed that roles are dynamic. But in real world there are two types of systems. The first type is continuously changed in role's functions at system operation stage. In this type of systems, no caching the role is more accurate even if it is less performance mechanism. The second type is rarely changed in role's functions at system operation stage. In this type, caching the user's roles in a complete profile is better in performance.

Authors of this research change the SRBAC model As in figure (4) by replacing services with processes, removing role hierarchy and assign any user to only one role which maintaining permissions on all system processes. The role can be assigned to many users. When user login to the system his role is captured and user profile is created on his client machine using caching technique. However in no caching case, when user login to the system his processes list available in his role is captured and user profile is created on his client machine and the process's permissions are checked when user is calling that process.

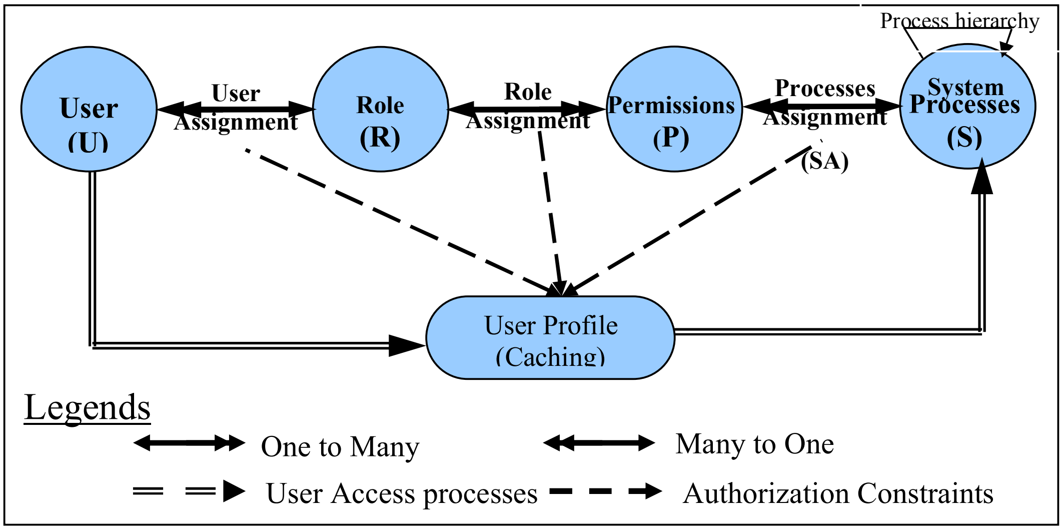


FIGURE (4) Proposed security Model

Moreover, it is better to collate all operations of user in one role and display it to his than split them into more than one role like it is in SRBAC model.

3.2 Conceptual model of IRBAC model's Security Tier Functionality.

In this section, Authors will explain the functionality of Security Tier, its objects, the relations between them, and the interaction between its components with each others and between them from one side, users and other model components on another side.

Authors of this research modify the model proposed in [1] by adding set of objects and components to meet their vision of new security model. According to figure (5), the proposed security model will contain six main objects:

- **System Processes (S):** This represents a list of system's Processes or Services which was created using BPMS Console component by Domain expert.
- **Changed Processes (CP):** This represents a list of changes happened to system's processes. This object is used by Functionality Adapter Component to update system processes(S) object with the last changes of system processes data.
- **Role (R):** This represents all the permissions of a specific type of system users on whole system's processes.
- **Permission (P):** This represents the access rights of one of system's processes for a specific Role (R).
- **User (U) :** which represents the users of BPS.
- **User Profile (UP):** This is generated by Profile Generator component. It contains all the user's permissions on the whole BPMS processes in caching case and contains list of BPMS processes available to user in no caching case.

First list of system processes or services (S) is created. Then RBAC Console is used by system administrator to perform the following steps in sequence:

1. Creates new Role and then creates permissions for all system processes and assign them to this role in **Permission Assignment** (PA) step,
2. For each permission of that role's permissions, specifies access rights of one system process in **System Processes Assignment** (SA) step.
3. Creates new users and assigns them to their appropriate Role in **User Assignment** (UA) step, where one or more user can share the same role but a user can't assign to more than one role in the system.
4. When the user login to BPMS:
   - **In caching case**, RBAC Console check user authentication. Then it captures all user authorization data which contains all constraints of the login user on that BPMS. The authorization data sent from RBAC Console to Profile Generator which uses it in creating complete User Profile. User Profile is sent to user client machine. User uses that profile which it creates on his client machine to interacts directly with the BPMS without needing to connect to the security server to capture his privileges on any service as long as his session is alive.
   - **In no caching case**, Console check user authentication. Then it captures all BPMS processes available to user and sent to Profile Generator which uses it in creating summarized User Profile and sent to user client machine. When user calls one of BPMS processes, user profile asks RBAC console to get the last permissions of user on that processes and then call that process under these permissions.

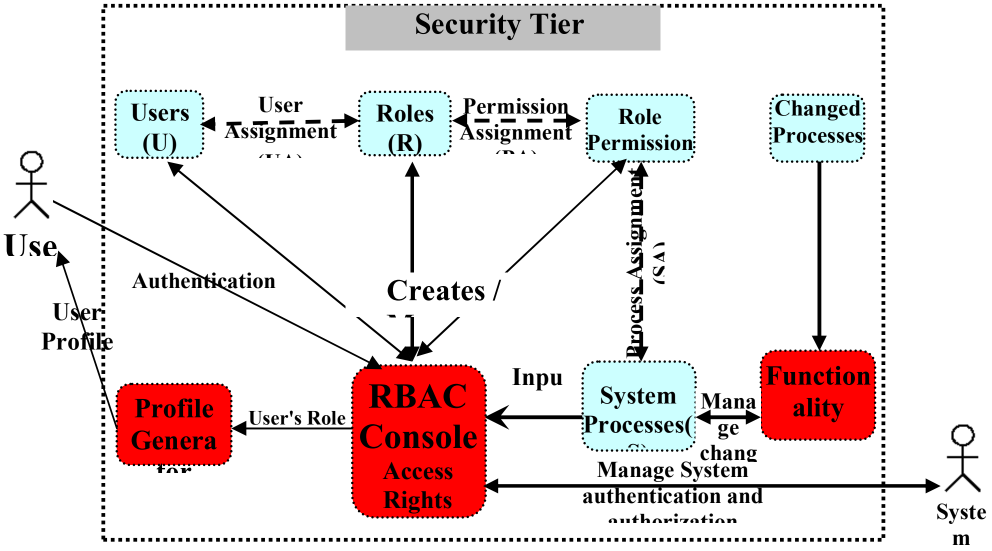


FIGURE 5: Conceptual Model of Proposed Security Model

But what if any changes happened in BPMS processes after BPMS had been deployed. How can these changes deployed to the running system? Authors use plug and play mechanism to do that. Where domain expert uses BPMS Console component to specify the changes happened to system processes. BPMS Console creates change processes object that maintain these changes. Then change processes object is plug into the BPMS. When system administrator login to the system, the Functionality Adapter component in the security tier of BPMS check the change processes object and executes all changes to the system.

3.3 Proposed Security Model Analysis.

According to scenario of model from authors point of view, there are three types of users will deal with the proposed model. These users are:

- **System Domain Expert:** He is a person who specifies and manages BPMS processes.
- **System Administrator:** is a person who creates/ manage users accounts, system's roles and permissions, and assign users to roles.
- **System user:** is a person who uses or operate BPMS.

In the following section authors will demonstrate the proposed model analysis by explain the model flow chart and the model use case.

3.3.1 Proposed Security Model Flow Chart

Figure (6) shows the proposed model flow chart. The user logs in to the System by entering his username/password. These data is checked by RBAC Management Engine. If authentication data is correct, Profile Generator component generates the profile of the user according to his type. If user is Domain Expert, the Profile Generator creates BPMS Domain Expert profile which contains BPMS Console of BPMS. If user is system Administrator, the Profile Generator creates BPMS System Administrator profile which contains RBAC Management Engine of BPMS. If user is Regular System User, the Profile Generator creates BPMS User profile which contains BPMS's processes available to the login user and his permissions on these processes according to Role he belongs to in caching case. However the profile generator creates only a list of all system processes available to user and displays it to him in summarized profile on his machine.

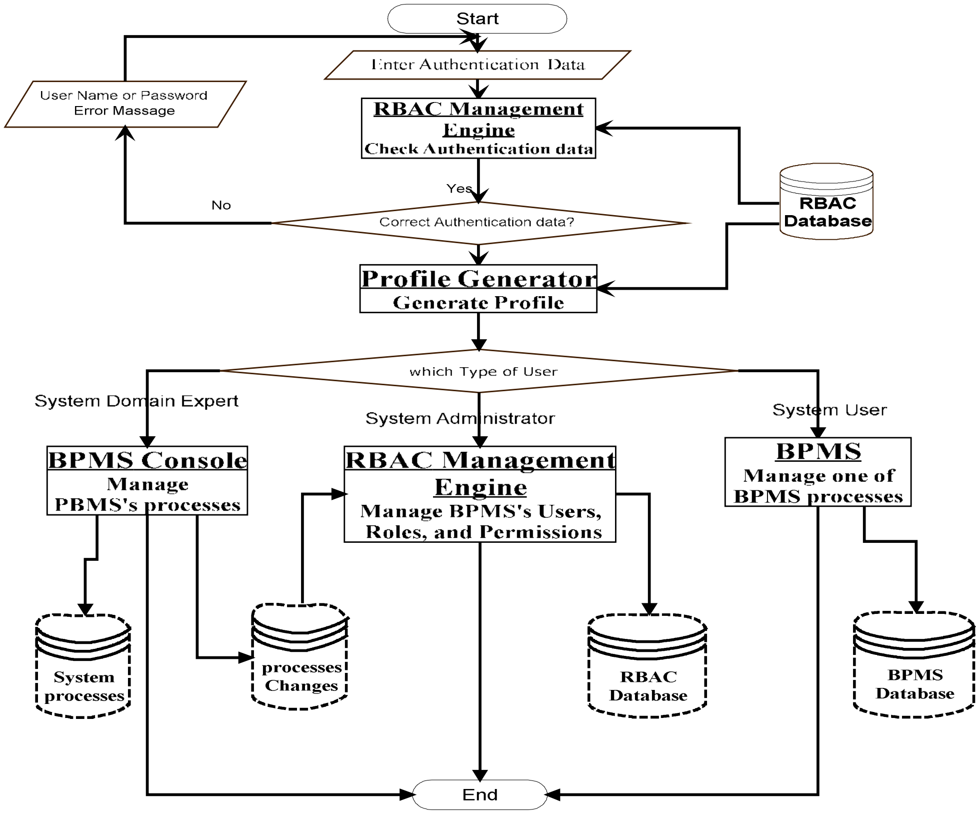


FIGURE 6: System Flow Chart of Proposed Security Model

3.3.2 Proposed Security Model Case Study

According to the authors' vision of the IRBAC model, use case consists of seven actors (System domain expert, System Administrator, System User, system processes DB, BPMS Database, and RBAC Database, and Business Process System) and nine use cases (Manage system processes, Manage role and specify access permissions, Manage User and assign them to appropriate Role, Authenticate to System, Create profile, ask for system process, check user permission, and call process under specific permissions) as shown in figure(7).

System Domain Expert uses "Manage all system's processes to create all system's processes at first or to modify these processes or add new processes next. This operation's data is store in system processes DB, which delivers to actual system to deploy the changes in system processes by updating system's processes which stored in BPMS Database of the actual system. Then system administrator use “Manage role and specify access permissions” to create new roles and specify access rights -permissions- for each process in the system to the created role. This operation read all system processes from BPMS DB and stores all role data and its permissions in RBAC DB. Then system administrator can create users’ accounts and assign user to his appropriate role according to the permissions specified to this user and roles permissions and store all these data in RBAC DB.

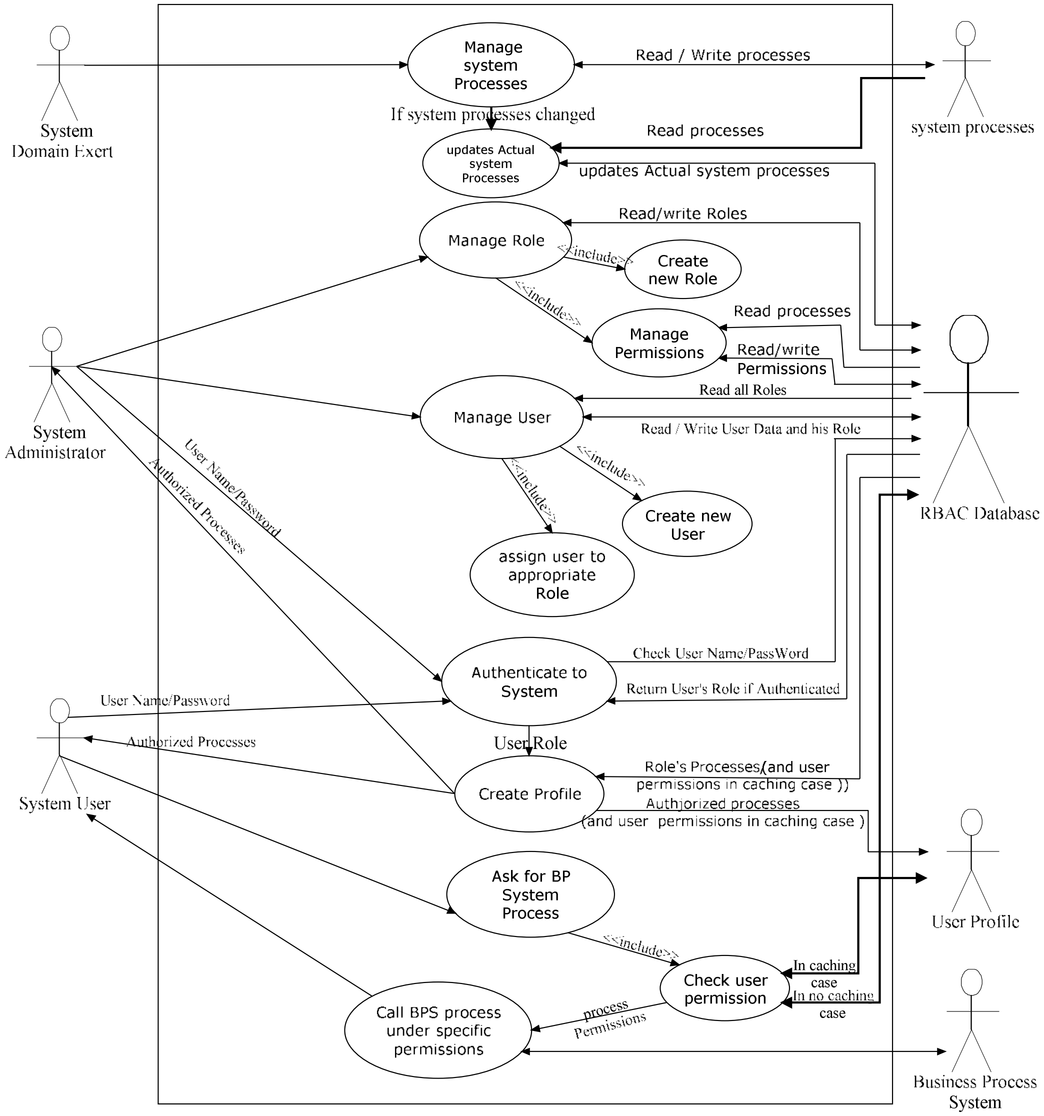


FIGURE 7: Use Case Diagram of proposed Model

Finally, for system user there are to cases:

- **In caching case :** when the user logs in to the system he enter his user name/password and the system perform Authentication check and specifies his role and what system’s processes available to him with his permissions by accessing BPMS DB and RBAC DB and create complete user profile and deliver it to him. When user needs to perform one of system processes, he selects the process from his profile displayed to him. User profile check user permission on the selected process and call this process under user’s permissions on that process.

- **In no caching case:** when the user logs in to the system he enter his user name/password and the system perform Authentication check and specifies his role and what system’s processes available to him by accessing BPMS DB and RBAC DB and create summarized user profile and deliver it to him. When user needs to perform one of system processes, he selects the process from his profile displayed to him. User profile check user permission on the selected process by getting it from RBAC DB and call this process under user's permissions on that process.

4. Proposed model Simulation and Validation

In this validation, authors compare between the performance in the two cases (with caching and no caching) and SRBAC model performance that was mentioned previously as a similar model to the IRBAC model and it was described well. This performance comparison has made on a small case study that simulate the IRBAC model in the two cases and SRBAC model.

Authors will propose the following three scenarios for two models:

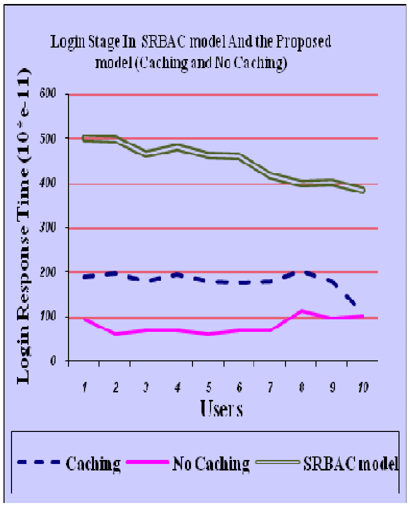
- For SRBAC model, after user authentication to one of his roles, an actor is created. The services of that role that he can access will appear to him in this actor with role's permissions. When he want to access any services from list of services appear to him, the actor will check the access rights of on that service and deal with that service. When user wants to access another role he belongs to, he must authenticate again, but this time to the other role which will create a new actor for that role. Through the new actor, user can deals with the system with another manner.
- For the proposed model, there are two scenarios:
    - With caching technique, after user authentication, the Profile Generator will check the all processes available to user and his permissions on these processes and caching all of them together in complete profile generated to user on his client terminal. When user wants to access specific process, his profile which was cached on his client terminal get access rights of that service from list of access rights stored within the profile which is generated at login without need to connect to the security server to access the process permissions, then connect to application server to get the process under his permissions.
    - With no caching technique, after user authentication, the Profile Generator will check the all processes available to user display a list of all of them in profile generated to user on his client terminal. When user wants to access specific process, his profile checks calling process's permissions for that user from security server. Then connect to application server to get the process under his permissions.

From scenarios that has been stated, there are two stages will be take in consideration in validation process .first stage is at login stage, and the second stage is at process calling.

Our test contains 10 users that are connecting to BPMS which consists of 40 processes. Each one of user can access only 30 processes with different permissions. Then 10 times of process call has been performed and measure the response time for each process call in the SRBAC model and the proposed model with and without caching and drew statically graphs which illustrate the results. In SRBAC model, Authors proposed that login user has three roles and the 10 processes he needs to access spread across these roles. Then to make 10 process calls across three roles, he needs to login to each role separately and make process call to required processes in this role.

Figure (8) shows the response time of login stage in the SRBAC model and the proposed model with and without caching. Figure (9) shows the response time of process calling in the SRBAC model and the proposed model with and without caching.

The results show that the proposed model without caching is better than the SRBAC model and proposed model with caching in login stage where the average response time for the proposed model no caching is (80.29*10e-11 s) but it is (112.76*10e-11s) in the proposed model with caching and average of three times login for 10 user of the SRBAC model is (447.65*10 e-11s). Whereas the proposed model with caching and SRBAC model is better than the proposed model without caching after login stage, along session life between user and BPMS. The average of response time of the proposed model with caching is (2.76*10e-11s) and it is (2.1*10e-11s) in SRBAC model, but it is (44.51*10e-11s) in the proposed model without caching for each process calling.



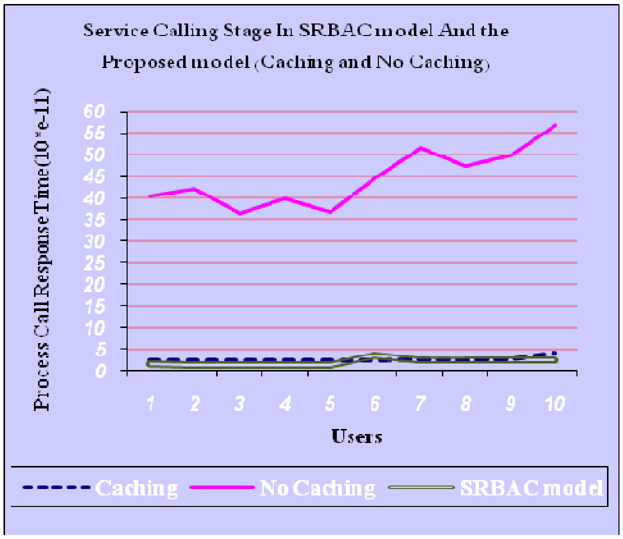


FIGURE 8: Login stage Response time

FIGURE 9: Process calling Response time

Authors combines the two stages (login & process access) in complete scenario for the proposed model with (caching & no caching) and SRBAC model and make 10 users perform all scenarios. The results can be seen in figure (10). The average of total response time using proposed model with caching is (163.07*10e-11s) whereas it is (1144.7*10e-11s) when using the proposed model without caching and it is (454.59*10e-11s) when using the SRBAC model.

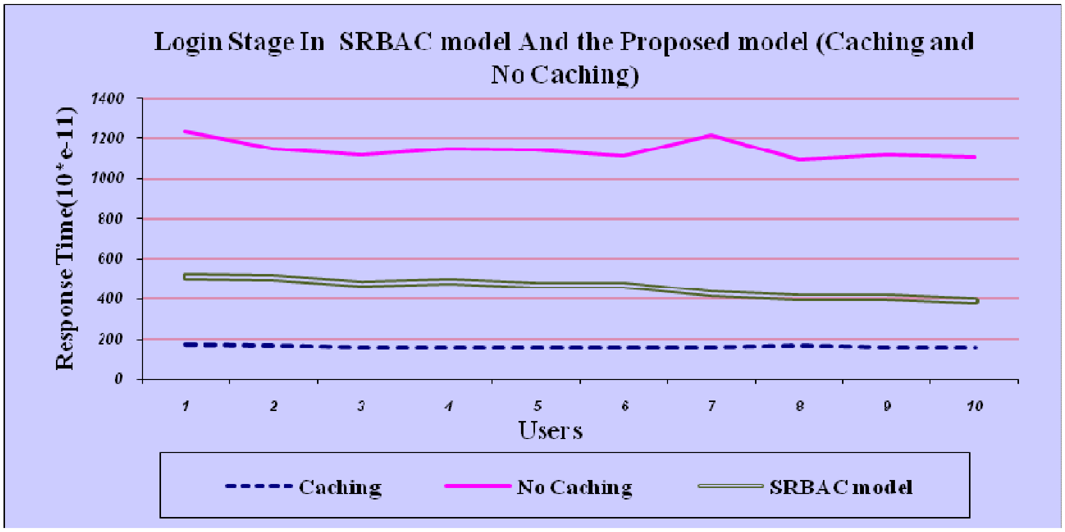


FIGURE 10: Total Response time of proposed model and SRBAC model

From results that have been reached, Authors conclude that when system's roles are rarely changing, the proposed model with caching is the best solution for managing user's authorization. But, when system's roles are continuously dynamic, the proposed model without caching is better solution whoever is the lowest in performance but it grantees that roles' permissions are up-to-date when user calls BPMS processes.

5. CONCLUSION

In this paper, authors proposed a generic security model (IRBAC) which modified SRBAC model to achieve a dynamic authorization security model when applying on any BPMS. The IRBAC model is more reliable when directly applied on the BPMS.

IRBAC model is compared with SRBAC in two cases. First case when IRBAC is combined with caching. And the second case when IRBAC is proposed with no caching.

Authors of that research presented a client/server N-tier architecture diagram of the proposed model. The client side represents the computer with browser from which system user interacts with the BPMS. The Server side consists of three tiers. First tier represents security tier and is responsible on manage the authentication and authorization of the BPMS. Second tier is business process logic tier which maintains all business logic of the BPMS and consists of business rule management engine and Application Interface Engine. Last tier is database tier, in which all BPMS data is maintained and managed.

Then authors presented a schematic diagram of the proposed model. It displayed the three types of users that deal with the security model, what component of the model user interacts with and interaction between all system components. The security tier, consists of three components. First component is RBAC Console which responsible on managing the authentication and authorization of all BPMS users on the system. Second component is profile generator component which captures all system processes available to login user and his permissions and creates his profile and send it to his machine (client side). When BPMS processes are changed, the functionality component is responsible on applying all changes on the actual system at system administrator login. In addition to these components, authors displayed DBPM Console component which enable BPMS domain expert from managing all system processes.

The modifications that made on the SRBAC model which modifications lead to a more reliable security model is presented.

Analysis of the proposed security model is done. That analysis is presented by presenting the proposed model system flow chart and use cases. This analysis presents how the three types of users (BPMS Domain expert, System Administrator, System User) interact with the system through the proposed security model.

Finally implementation of the proposed model using a simple case study is done. The case study is the Cultural Affairs System of Mansoura University. This model is implemented entirely in PHP language and MySQL. The performance of the proposed model in tested in two cases and compared the SRBAC model. This comparison had made in two stages. First stage is at login stage which appeared that the proposed model with no caching is better than SRBAC model figure(8). Second stage is at process calling stage which appeared that the proposed model with caching is better than SRBAC. But SRBAC is better than the proposed model with no caching figure(9).

Then authors combined the two stages in figure (10) which showed that the proposed model with caching technique is better solution for managing authorization of system’s users.

In the future, authors of that research will apply the proposed model on another real BPMS case study in details.

6. REFERENCES

1. Xu Feng ,Lin Guoyuan , Huang Hao , Xie Li;"Role-based Access Control System for Web Services"; In Proceedings of the 4th IEEE International Conference on on Computer and Information Technology ,2004

2. Ateniese, G., Camenisch, J., and Madeiros, B. de, “Untraceable RFID tags via insubvertible encryption”, Proceedings of the 12 ACM conference on Computer and communications security, November, pp.92-101, 2005.

3. Barkley, J., Beznosov, K., and Uppal, J., “Supporting Relationship in Access Control Using Role Based Access Control”, Proceedings of ACM Role-Based Access Control Workshop, Fairfax, Virginia, USA, pp. 55-65, 1999.

4. Bernardi, P., Gandino, F., Lamberti, F., Montrucchio, B., Rebaudengo, M., and Sanchez, E.R., “An Anti-Counterfeit Mechanism for the Application Layer in Low-Cost RFID Devices”, In International Conference on Circuits and Systems for Communications, IEEE, July, pp.207-211, 2006.

5. T. Neubauer, M. Klemen, and S. Biffl. Secure Business Process Management: A Roadmap. In Proceedings of the First International Conference on Availability, Reliability and Security ARES, pages 457–464. IEEE Computer Society, 2006.

6. T. Neubauer and J. Heurix : Objective Types for the Valuation of Secure Business Processes. In Proceedings of the Seventh IEEE/ACIS International Conference on Computer and Information Science, page 231. IEEE Computer Society, 2008.

7. M. Wu and Y. Fong : Applying Role-Based Access Control in Combining the Chinese and Western Medicine Systems. In Proceedings of the 19th International Conference on Systems Engineering . IEEE Computer Society, 2008.

8. Chen, G., and Kotz, D., “A Survey of Context-Aware Mobile Computing Research”, Technical Report 2000-381, Dept. of Computer Science, Dartmouth College, Hanover, N.H, 2000.

9. Dey, A. K., and Abowd, G. D., “Towards A Better Understanding of Context and Context-awareness”, GVU Technical Report GITGVU-99-22, pp.304-307, 1999.

10. Schilit, B. N., Adams, N., and Want, R., “Context-Aware Computing Applications”, In Proceedings Workshop on Mobile Computing Systems and Applications, IEEE, pp.85-90, December, 1994.

11. Wolf, R., Keinz, T., and Schneider, M., “A Model for Context-dependent Access Control for Web-based Services with Role-based Approach”, Proceedings of the 14th International Workshop on Database and Expert Systems Applications, September, pp.209-214, 2003.

12. Heiko, K., and Hartmut, P., “RFID Security”, Information Security Technical Report, December, Volume 9, Issue 4, pp.39-50, 2004.

13. Li, Y.Z., Jeong, Y.S., Sun, N., and Lee, S.H., “Low-Cost Authentication Protocol of the RFID System Using Partial ID”, In Computational Intelligence and Security, IEEE, pp.1221-1224, November, 2006.

14. M. Sloman and E. Lupu. Security and management policy specification. Network, IEEE, 16(2):10–19, 2002.

15. R. Sandhu, E. Coyne, H. Feinstein, and C. Youman. Rolebased access control models. IEEE Computer, 29(2):38–47, 1996.

16. T. Neubauer, M. Klemen, and S. Biffl. Secure Business Process Management: A Roadmap. In ARES’06, pages 457– 464, 2006

17. C. Yang. Designing secure e-commerce with role-based access control. International Journal of Web Engineering and Technology, 3(1):73–95, 2007.

18. David F. Ferraiolo, John F. Barkley, and D. Richard Kuhn. A role based access control model and reference implementation within a corporate intranet. In ACM Transactions on Information Systems

19. Xin Wang, Yanchun Zhang, Hao Shi ;" Access Control for Human Tasks in Service Oriented Architecture "; in IEEE/ the Fourth International Conference on Computer and Information Technology (CIT’04);2004 IEEE Computer, 29(2):38–47, 1996.

20. Mathias Kohler and Andreas Schaad . ProActive Access Control for Business Process-driven Environments.in IEEE/ Annual Computer Security Applications Conference 156 .2008.