An Object Oriented Role-based Access Control Model for Secure Domain Environments

Cungang Yang

Department of Electrical and Computer Engineering, Ryerson University
Toronto, Ontario, M5B 2K3, Canada (Email: cungang@ee.ryerson.ca)

(Received Aug. 22, 2005; revised and accepted Oct. 11, 2005)

Abstract

In the secure domain computing environments, it is important to keep resources and information integrity from unauthorized access. Therefore, there is a strong demand on the access control for shared resources. In the past few years, Role-based Access Control (RBAC) has been introduced and offered a powerful means of specifying access control decisions. In this paper, an Object Oriented RBAC model (ORBAC) that efficiently represents the real world is proposed. It not only supports a single secure domain authorization for an enterprise, but also supports multiple secure domains access control for E-business carried out by multiple enterprises over the Internet.

Keywords: Digital credential, e-business, least privilege, ORBAC, secure domain

1 Introduction

An important element of access control is *secure domain* which consists of a set of users and objects managed by a common security policy and defined by a single authority. With the increasing of shared resources, unauthorized access to information by illegal users also increases, it is necessary to secure data through user authentication and access control policies.

Nowadays, there are three basic access control techniques: mandatory access control (MAC), discretionary access control (DAC) and role-based access control (RBAC) [6, 7, 8]. MAC enforces access control on the basis of information security labels attached to users and objects. MAC can determine all kinds of access control between subjects and objects consistently. Security labels are granted to all subjects and objects by the system supervisor and the modification of the security labels only in accordance with the content of the object. As MAC policy is not flexible, it is not suitable to be used in commercial areas. In case of DAC, each object has an access control list. However, in a large information system there are millions of objects, and each of which is assigned to thousands of subjects. The access control lists will be enormous in size and their maintenance will be much difficult and costly. Compared with DAC and MAC, the central notion of RBAC is that users do not directly get access to enterprise objects; instead, access privileges of the objects are associated with roles, each user is assigned to one or multiple members of appropriate roles. As a result, an organization cannot only preserve access control policy appropriate to its characteristics consistently, but it can also maintain access control relationships between users and objects independently. Users can be assigned to members of roles as determined by their responsibilities and qualifications. On the other hand, they can be easily reassigned without modifying the underlying access structure. RBAC greatly simplifies the management of authorizations while providing an appropriate method for great flexibility in specifying and enforcing enterprise-specific protection policies and reducing the management costs.

In the last few years, the fundamentals of RBAC policies have been identified [8] and a number of RBAC models have been proposed to satisfy security requirements in different areas [4, 6, 7], but they are all concept models and have not been efficiently represented the real world. In this paper, a variation of RBAC model for secure domain environment is proposed. The significance of the proposed model includes (1) It extends access control from a single secure domain to multiple secure domain environments. (2) It supports a role authorization algorithm. (3) Secure domain management architectures for a single secure domain and multiple secure domains are proposed.

The rest of the paper is organized as follows. Section 2 introduces the object oriented role-based access control model for a single secure domain environment. Section 3 concerns about the model for multiple secure domain environments. Section 4 proposes the management architectures of secure domain environment. Section 5 concludes the paper.

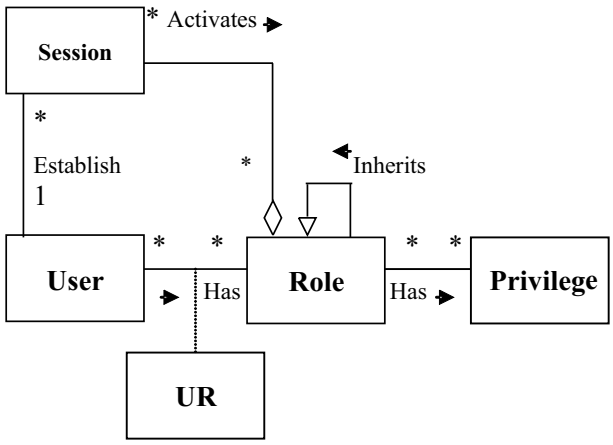

Figure 1: Class diagram of ORBAC model for single secure domain environment

2 ORBAC Model for Single Secure Domain Environment

A number of different viewpoints about RBAC have been discussed[5, 9, 10]. The abstract model defined in this section intends to capture the essential features of RBAC and fully realizes the original RBAC functions for a single secure domain environment. The class diagram of ORBAC described by United Modeling Language (UML)[11] is shown in Figure 1 and an example of ORBAC object diagram is illustrated in Figure 2.

In the model, a *user* is a human being or an autonomous agent, a *role* is a collection of privileges to perform a certain task, a *privilege* is an access mode that can be exercised on objects. A user can be assigned to a number of different roles, and a role can have multiple users. A role may have multiple privileges, and the same privilege can be associated to different roles. Moreover, a *role hierarchy* is introduced to reflect inheritance of authority and responsibility among the roles and it is defined by: If $r_i \leftarrow r_j$, then role r_i inherits the privileges of role r_j , role r_i is a *direct parent role* of role r_j and role r_j is a *direct child role* of role r_i. Furthermore, the inheritance relationship can be transitive, that is, if $r_i \leftarrow r_j$ and $r_j \leftarrow r_k$, then $r_i \leftarrow r_k$, role r_i is an *indirect parent role* of role r_k and role r_k is an *indirect child role* of role r_i . In the role hierarchy, direct and indirect parent roles are also defined as *parent roles*, direct and indirect child roles are defined as *child roles*. For instance, role B in Figure 2 is a direct parent role of role D and role D is a direct child role of role B, role B is a indirect parent role of role F and role F is an indirect child role of role B, role B is a parent role of role D and role F, role F is a child role of role B and role D.

In the object diagram, each user is assigned a role set according to his/her responsibility and authority, it is called *assigned role set*. In Figure 2, the assigned role set of user X is {A, B} and the assigned role set of user Y

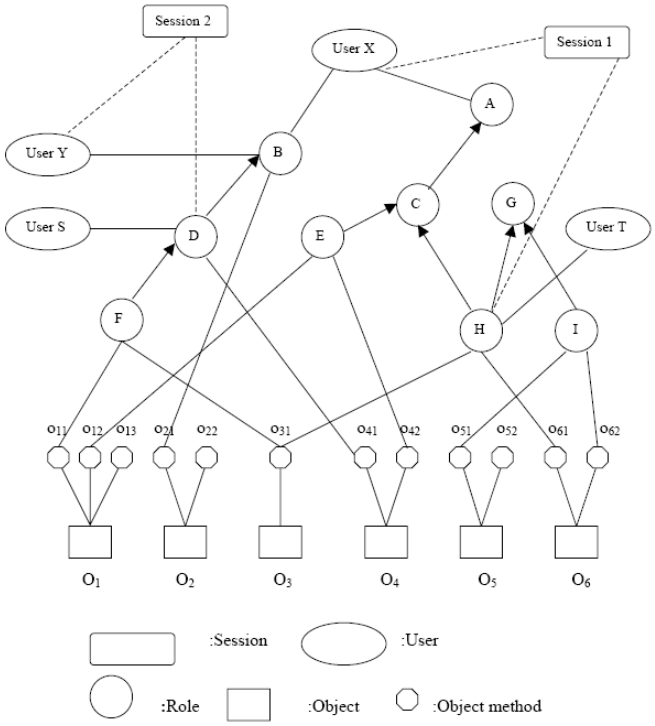

Figure 2: An example of ORBAC object diagram in a single secure domain

is {B}. Privileges are represented by object and its object methods. Any role may invoke one or a number of object methods and therefore directly have one or multiple privileges. Furthermore, a role may also indirectly inherit the privileges of their child roles. For instance, the privileges of role D is {((Object 4, $o41$)} and its inherited privileges from role F is {(Object 1, o11), (Object 3, $o31$)}. In addition, for each role L, if a user's assigned role set including role L then he/she belongs to the *authorized user set* of role L. In Figure 2, the authorized user set of role B is {X, Y} and the authorized user set of role A is {X}.

Association class UR deals with the relationships between a user and the roles he/she applies by using a role authorization algorithm. The algorithm automatically searches the role hierarchy from each of the applied roles to its direct or indirect parent roles, in a large enterprise, since a role hierarchy could be very large and complex, breadth-first is used as the searching method.

The conditions that the search will be ended are:

- If user X applies for a role R, for any direct or indirect parent role of R, S, if S belongs to the authorized user set of role R then user X is authorized the role R.
- If user X is applying for a role R, for any direct or indirect parent role of R, S, if S does not belong to the authorized user set of role R then user X cannot be authorized the role R.

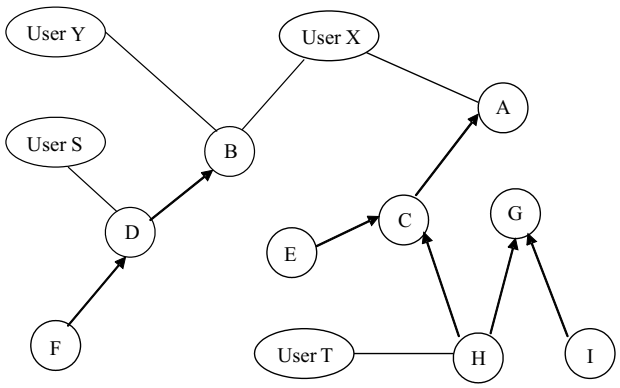

Figure 3: An example of role authorization algorithm

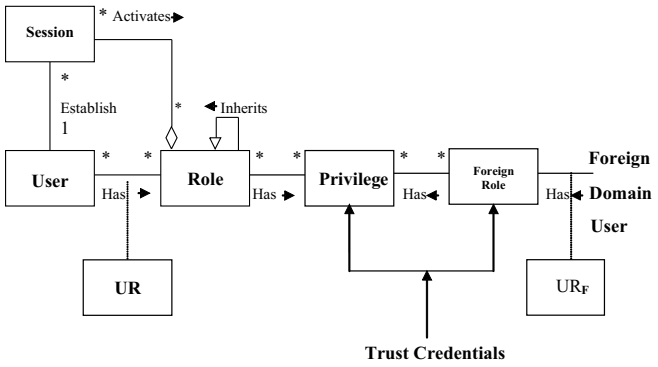

Figure 4: Class diagram of ORBAC model for multiple secure domain environment

Algorithm 1: Role Authorization Algorithm (*Applied role R, User X*)

Input:

X: identification of the user.

Applied role R: the role that user applies for.

V: the role set including role R and its parent roles in the role hierarchy.

Q: a FIFO queue.

Mark []: indicator of a visited/unvisited role.

Di-Parent [r]: direct parent role of role r.

Output:

authorized role R to the user or refuse message.

Begin

For each $u \in V - \{R\}$

$Mark[u] = 0$;

$Mark[R] = 1$;

$Q \leftarrow \{R\}$;

While $Q <> 0$ *do*{

Remove R from Q;

if X $\in$ *the authorized user set of role R* {

output R;

}

For eachv $\in$ *Di-Parent[R]*{

if mark[v]=0

put v on Q ;

}

}

output refuse message

}

Analysis of the algorithm 1:

For the applied role R, each direct or indirect parent role of R, S, is places in the queue once, assume the number of S is L, the computation complexity of the algorithm will be O(L).

Example 1. *In Figure 3, assume user X applies for role* $\{H\}$ *to obtain privileges, suppose the authorized user set of role A is* $\{X\}$*, the authorization user set of role D is* $\{S\}$*, the authorization user set of role H is* $\{T\}$*. Assume breadth-first search method is used and the searching path for role H will be* $H \rightarrow G$ *and* $H \rightarrow C \rightarrow A$*, when the search procedure reaches role A, user X belongs to the authorized user set of role A, so role* $\{H\}$ *is authorized to user X. If user T applies for role D, the searching path will be* $D \rightarrow B$*, user T does not belong to the authorized user set of role D and role B, so his/her application will be refused.*

In the object diagram, session *is a mapping of a user to possible many roles [2], after the roles are authorized to a user by the role authorization algorithm, they are called* authorized roles *for the user. A user may have one or more than one sessions activated at the same time and each session may have a different combination of authorized roles. For instance, in Figure 2, in session 1, the authorized role for user X is* $\{H\}$*; in session 2, the authorized role for user Y is* $\{D\}$*.*

3 ORBAC Model For Multiple Secure Domain Environments

Multiple Secure domains consist of a set of users and objects that are managed by different security policies and defined by multiple authorities. In the multiple secure environments, users may want to access objects located outside their administration domains. On the other hand, with the development of the Internet and e-business, there are many requirements that users in a secure domain would like to share objects with users in other secure domains. The large distributed system, like Internet, is quickly becoming the largest marketplace, allowing commerce and business between parties who are physically distant and do not know each other, such applications involve every aspect of e-business. In the ORBAC model of Figure 4, the access control model is extended from a single secure domain environment to multiple secure domain environments and assume that all those different secure domains are based on the ORBAC model and interconnected on the Internet. In each single secure domain, some privileges can be accessed by the users from other secure domains, they are foreign privileges. Those users from other secure domains are called *foreign domain*

users. To establish the relationships between users in different domains, some trustworthy need to be established. The traditional approach of using authentication to differentiate between classes of clients is no longer sufficient, as knowledge of a user's identity will often not sufficient to determine whether a user is authorized a privilege, credentials make it feasible to manage trust establishment efficiently, our goal is therefore to explore the use of digital credentials [1, 2, 3] to solve this problem. Digital credentials are the online counterparts of paper credentials that people use in their daily life, such as a driver's license, ACM membership card, VISA card, etc. They are signed by authorized parties and can be made verifiable and unforgeable. Trust credentials contains one or more than one digital credentials that should be provided by foreign domain users if they want to get some foreign privileges, such as the certificate of doctor or the certificate of nurse. For instance, if some services in a domain only can be accessed by the foreign domain user who is a certificated nurse, the certificate of nurse should be provided, if some service only can be accessed by the foreign domain user who is a certificated doctor, the certificate of doctor should be provided.

Based on trust credentials, the privileges of a single secure domain is divided into three groups:

1) Privileges that can be accessed by foreign domain users without the constraints of trust credentials
2) Privileges that can not be access by foreign domain users.
3) Privileges that can be accessed by foreign domain users under different constraints of trust credentials.

Only privileges in group (1) and group (3) are foreign privileges, in order to simplify the administration of those foreign privileges, foreign role is introduced. Foreign role represents a group of foreign domain users who are authorized to get access to foreign privileges, foreign domain users do not directly get access to foreign privileges; instead, foreign privileges are associated with foreign roles, each foreign domain user is assigned to one or multiple members of appropriate foreign roles and each foreign role is assigned to one or multiple foreign domain users. Each foreign role can be assigned multiple foreign privileges and each foreign privilege can be assigned to multiple foreign roles.

This idea greatly simplifies the management of authorizations while providing an appropriate method for great flexibility in specifying and enforcing multiple secure domain access control and reduces the management costs.

There are two kinds of trust credentials, *authentication credentials* and *authorization credentials*. Authentication credentials are the digital credentials which constraint foreign domain users to get foreign roles. For example, if a user wants to view a web page in which only some authentication credential holder can access, it is sufficient to prove that he/she is actually an authentication credential holder, foreign domain users can be assigned to members of foreign roles as determined by authentication credentials, for instance, in Figure 5, foreign role H is authorized to the group of foreign domain users who owns the C1 authentication credential and foreign role I is authorized to the group of foreign domain users who owns the C2 authentication credential.

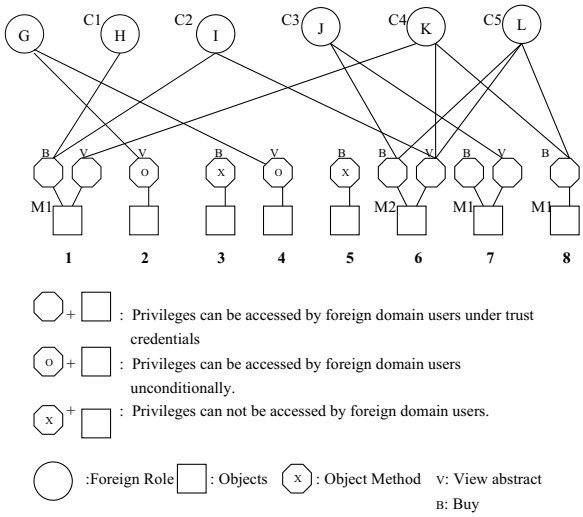

Figure 5: An example of ORBAC object diagram in multiple secure domains

The other trust credential constraint is called *authorization credential*, it constraints whether or not the foreign domain users can get access to a foreign privilege after they have held foreign roles, the authorization credentials, for example, M1 and M2 in Figure 5 are Visa or Master card credential, so if a foreign domain user wants to buy the medical web pages 1 or web page 7, a M1 or M2 authorization credential should be submitted.

Following example briefly introduces the basic elements in ORBAC object diagram for multiple secure domains.

Example 2. *Consider Figure 5 where privileges of object 3 and privileges of object 5 can not be authorized to foreign domain users, privilege (Object 2, V) and (Object 4, V) are authorized to foreign role G without authentication or authorization credentials, any foreign domain user can get access to them unconditionally. Privilege (Object 1, B) is authorized to foreign role H under the authentication credential C1; Privilege (Object 1, B) is authorized to foreign role I under the authentication credential C2; privileges (Object 6, B) is authorized to foreign role J under the authentication credential C3 and authorization credential M2, privileges (Object 6, B) is authorized to foreign role L under the authentication credential C5 and the authorization credential M2; privilege (Object 7,V) is authorized to foreign role J under the authentication credential C3 but without authorization credentials, privilege (Object 8, B) is authorized to foreign role K under the authentication credential C4 and the authorization credential M1; privilege (Object 8, B) is authorized to foreign role L under the authentication credential C5 and the authorization*

credential M1.

The association class UR_F determines the mapping from the authentication credentials to foreign role and return the foreign role to the foreign domain user. The main function of the class is to call the foreign role authorization algorithm, accept applied foreign privilege from foreign domain users, for each applied foreign privilege, searches all the foreign roles which is authorized the foreign privilege using breadth-first search method, for instance, in Figure 5, the foreign roles of the foreign privilege {Object 7, V} is J and the authentication credential for privilege {Object 7, V} is C3, if the foreign authentication credentials a user provided contain C3, the user will be authorized the foreign role J to the user, otherwise, his/her application will be refused. The foreign role authentication algorithm is shown as follows.

Algorithm 2: Foreign Role Authorization Algorithm

Input:

Applied foreign privilege M: the required foreign privilege M.

Authentication credentials of the foreign domain user H: the authentication credentials the foreign domain user provided.

Output:

Authorized foreign role R to H or refuse message.

```
{
For each foreign role R of the foreign privilege M
    {
    if authentication credentials of R ⊆ H {
        foreign role R is authorized;
          output foreign role R;
        }
      }
output refuse message;
}
```

4 Secure Domain Management Architecture

4.1 Single Secure Domain Management Architecture

In this section, two management architectures for a single secure domain, architecture 1 and architecture 2, are presented. Their collaboration diagrams are shown separately in Figure 6 and Figure 7.

The main functions of each element in the architecture, domain security manager, role server, client and server have been described [12, 13].

Domain security manager: Creates and maintains the ORBAC secure role hierarchy, etc.

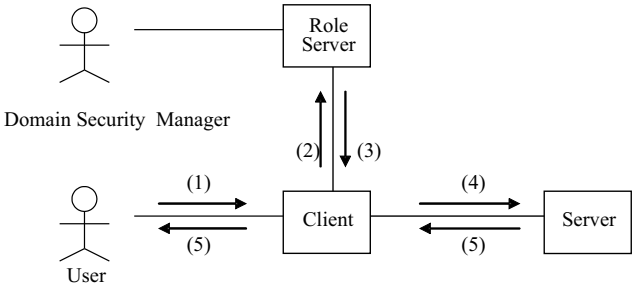

Figure 6: Collaboration diagram for management architecture 1

Role Server: Automatically authorizes roles to users.

Client: Accepts the applied roles from users who want to get some privileges for tasks and returns the authorized roles back to them.

Server: Checks the user's authorized roles and decides whether or not permit him/her to get the privileges.

The general procedure of the architecture 1 is shown as follows:

1) User submits his/her user id and applies for role R via Client.

2) Client sends the user's id and his/her applied role R to role server.

3) Role server receives them and the role authorization algorithm is called to determine whether or not authorizes role R to the user, if R is authorized, a session will be created and the authorized roles are returned back to the user.

4) User gets the authorized role R and accesses the server.

5) Server authorizes privileges to the user according to the authorized role R.

The general procedure of architecture 2 is described as below:

1) User submits his/her user id, required privileges and applies for role R via Client.

2) Client sends the user's id, required privileges and his/her applied role R to role server.

3) Role server receives them and the role authorization algorithm is called to determine whether or not assign role R to the user, if R is authorized, a session will be created, then Role server submits the authorized role R and the user's required privileges to access the server.

4) Server authorizes privileges to the user according to the authorized role R.

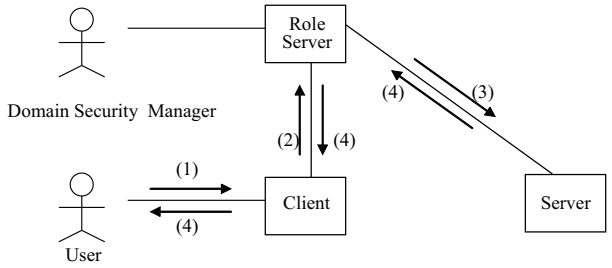

Figure 7: Collaboration diagram for management architecture 2

4.2 Secure Multiple Domain Management Architecture

In this section, a collaboration diagram for multiple secure domain management architecture is present. Assume that all single secure domains are based on the ORBAC model, a user in a single secure domain not only can be authorized the roles and get access to the objects in his/her domain, but also he/she can be authorized foreign roles and get access to the objects in other secure domains. In Figure 8, user in domain N gets access to the objects in domain M and user in domain M gets access to the objects in domain N.

In multiple secure domain environments, more functions of each element in the architecture, domain security manager, role server, client and server are shown as below:

Domain security manager: Creates and maintains the trust credential constraints and define foreign roles and foreign privileges, etc.

Role Server: Besides authorizing roles to user in its secure domain, role server also automatically authorizes those foreign privileges to foreign domain users. Automatically authorizes foreign roles to foreign domain users.

Client: Accepts the applied foreign domain privilege from users and returns the authorized foreign privilege back to them.

Server: Checks the foreign domain user's authorized foreign roles and decides whether or not permit him/her to get the foreign privileges.

The general procedure of the multiple secure domain management architecture is shown as below, assume the user in domain M applies for foreign privilege in domain N:

1) User in domain M applies for foreign privilege in domain N, role server of domain N replies him/her with the required authentication credentials (if exist), then he/she submits the required authentication credentials to the role server.

2) Role server calls foreign role authentication algorithm to assigns him/her a foreign role R, then searches all the authorization credentials (if exist) for his/her required privilege and returns them to the user.

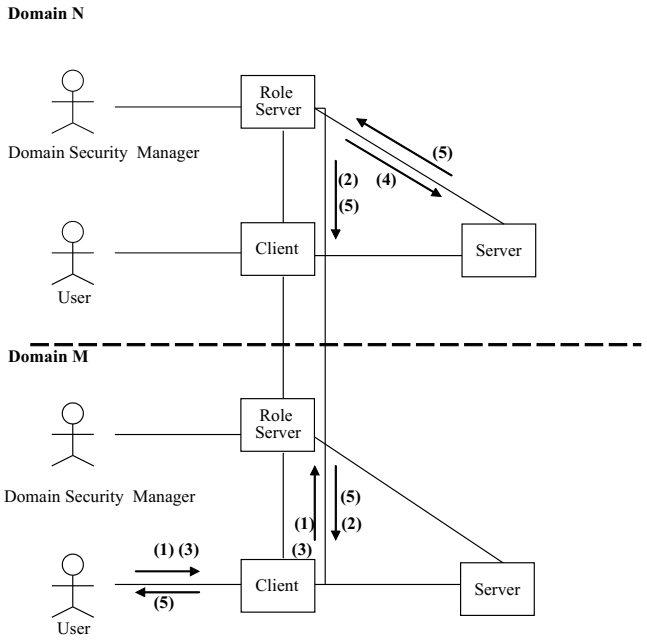

Figure 8: Collaboration diagram for multiple secure domain management architecture

3) User selects the required authorization credentials and sends them to role server.

4) Role Server gets access to the server along with the user's authorized foreign role R and his/her required foreign privileges.

5) Server authorizes him/her required privileges according to the authorized foreign role R.

5 Conclusion and Future Work

In this paper, an objected oriented RBAC model (ORBAC) for secure domain is presented. The driving motivation of it is to simplify security policy administration. The main contributions of this paper are the object oriented role-based access control model and management architecture for a single and multiple secure domain environments.

In the role-based access control model, it is assured that users access to objects based on roles to which users belong. However, illegal information flow among objects may occur, as this is the confinement problem pointed out in the basic access control model. In order to create a secure role hierarchy so that no indirect privilege authorization (illegal information flow) exists and guarantee the user role authorization methods, more research work on it is much required.

References

[1] M. Blaze, J. Feignbaum, and A. D. Keromytis, "KeyNote: Trust management of public-key infrastructures," *Security Protocals, 6th International Workshop*, Cambridge UK, pp. 59-63, 1998.

[2] M. Blaze, J. Feignbaum, and J. Lacy, "Decentralized trust management," *IEEE Symposium on Security and Privacy*, pp. 17-28, Oakland, CA, May 1996.

[3] M. Blaze, J. Feignbaum, J, Ioannidis, and A. Keromytis, "The KeyNote trust management system version 2," *Internet Draft RFC 2704*, Sept. 1999.

[4] D. Ferraiolo, J. Cugini, and D. R. Kulin, "Role based access control: Features and motivacation," in *Annual Computer Security Applications Conference,* IEEE Computer Society Press, pp. 241-248, 1995.

[5] *Integrity in Automated Information Systems*, National Computer Security Center, pp. 79-91, Sept. 1991.

[6] T. Jaeger and Frederquegiraud, "A role-based access control model for protection domain derivation and management," *Second ACM Workshop on Role-Based-Access-Control*, pp 95-108, Fairfax, Virginia, USA, Nov. 1997.

[7] R. Sandhu and V. Bhamidipati, "The ARBAC97 model for role-based administration of roles: Preliminary description and outline," *Second ACM Workshop on Role-Based-Access-Control*, Fairfax, Virginia, USA, pp. 41-54, Nov. 1997.

[8] R. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, "Role based access control models," *IEEE Computer*, vol. 29, no. 2, pp. 38-47, Feb. 1996.

[9] R. Sandhu, D. Ferraiodo and R. Kulin, "The NIST model for role-based access control: Towards a unified standard," *Fifth ACM Workshop on Role-Based Access Control*, Berlin, Germany, pp. 47-64, July 2000.

[10] S. Sborn and Y. Guo, "Modeling users in role-based access control," *Fifth ACM workshop on Role-Based Access Control, Berlin, Germany*, pp. 31-38, July 26-27, 2000.

[11] *UnitedModeling Language*, UML resource center, Http://www.rational.com/UML/

[12] C. N. Zhang and C. Yang, "Specification and enforcement of object-oriented RBAC model," in *2001 IEEE Canadian Conference on Electrical and Computer Engineering*, Toronto, pp. 65-77, May 2001.

[13] C. N. Zhang and C. Yang, "An object-oriented RBAC model for distributed system," in *2001 IEEE/IFIP Conference on Software Architecture WICSA 2001*, Netherland, pp. 24-32, Aug. 2001.

Cungang Yang received the M.S degree in computer science from Jilin University, China. He completed his Ph.D degree in computer science in 2003 at University of Regina, Canada. In 2003, he joined the Ryerson University as an assistant professor in the Department of Electrical and Computer Engineering. His research areas include security and privacy, enhanced role-based access control model, information flow control, web security and secure wireless sensor networks.