Role-based delegation with negative authorization

Hua Wang[1], Jinli Cao[2], and David Ross[3]

[1] Department of Maths & Computing, University of Southern Queensland
Toowoomba QLD 4350 Australia
wang@usq.edu.au
[2] Department of Computer Science & Computer Engineering
La Trobe University, Melbourne, VIC 3086, Australia
jinli@cs.latrobe.edu.au
[3] Engineering Faculty, University of Southern Queensland
Toowoomba QLD 4350 Australia
ross@usq.edu.au

Abstract. Role-based delegation model (*RBDM*) based on role-based access control (*RBAC*) has proven to be a flexible and useful access control model for information sharing on distributed collaborative environment. Authorization is an important functionality for *RBDM* in distributed environment where a conflicting problem may arise when one user grants permission of a role to a delegated user and another user grants the negative permission to the delegated user.
This paper aims to analyse role-based group delegation features that has not studied before, and to provide an approach for the conflicting problem by adopting negative authorization. We present granting and revocation delegating models first, and then discuss user delegation authorization and the impact of negative authorization on role hierarchies.

1 Introduction

Delegation is the process whereby an active entity grants access resource permissions to another entity in a distributed environment. Delegation is recognised as vital in a secure distributed computing environment [Abadi et al. 1993; Barka and Sandhu 2000a]. However, a conflicting secure problem may arise when one user grants permission of a role to a delegated user and another user does reject the permission to the delegated user. The most common delegation types include user-to-machine, user-to-user, and machine-to-machine delegation. They all have the same consequence, namely the propagation of access permission. Propagation of access rights in decentralized collaborative systems presents challenges for traditional access mechanisms because authorization decisions are made based on the identity of the resource requester. Unfortunately, access control based on identity may be ineffective when the requester is unknown to the resource owner [Wang et al. IEEE03]. Recently some distributed access control mechanisms have been proposed: Lampson et al. [1992] present an example on how a person can

delegate its authority to others; Blaze et al. [1999] introduced trust management for decentralized authorization; Abadi et al. [1993] showed an application of express delegation with access control calculus; and Aura [1999] described a delegation mechanism to support access management in a distributed computing environment. All these papers have not analysed the conflicting secure problem.

The National Institute of Standards and Technology developed role-based access control (*RBAC*) prototype [Feinstein, 1995] and published a formal model [Ferraiolo et al. 1992]. *RBAC* enables managing and enforcing security in large-scale and enterprise-wide systems. Many enhancements of *RBAC* models have been developed in the past decade. In *RBAC* models, permissions are associated with roles, users are assigned to appropriate roles, and users acquire permissions through roles. Users can be easily reassigned from one role to another. Roles can be granted new permissions and permissions can be easily revoked from roles as needed. Therefore, *RBAC* provides a means for empowering individual users through role-based delegation in distributed collaboration environments. However, there is little work on delegation with *RBAC*.

This paper analyses role-based delegation model based on *RBAC* and provides a solution for the conflicting problem adopting negative authorization. The remainder of this paper is organized as follows: Section 2 presents the related work associated to delegation model and *RBAC*. As the results of this section, we find that both of group-based delegation with *RBAC* and negative authorization for delegation model has never analysed in the literature. Section 3 proposes a delegation framework which includes group-based delegation. Granting authorization with pre-requisite conditions and revocation authorization are discussed. Section 4 provides an approach for the conflicting problem by adopting negative authorization and briefly discusses how to use negative authorization in delegation framework. Section 5 concludes the paper and outlines our future work.

2 Related work

The concept of delegation is not new in authorizations [Aura 1999; Barka and Sandhu 2000a; Wang et al. IEEE03; Wang et al. ACSC05], role-based delegation received attention only recently [Barka and Sandhu 2000a, 2000b; Zhang et al. 2001, 2002]. Aura [1999] introduced key-oriented discretionary access control systems that are based on delegation of access rights with public-key certificates. A certificate denotes a signed message that includes both the signature and the original message. With the certificate, the issuer delegates the rights R to someone. The systems emphasized decentralization of authority and operations but their approach is a form of discretionary access control. Hence, they can neither express mandatory policies like Bell-LaPadula model [1976], nor possible to verify that someone does not have a certificate. Furthermore, some important policies such as separation of duty policies cannot be expressed with only certificates. They need some additional mechanism to maintain the previously granted rights and the histories must be updated in real time when new certificates are issued. Delegation is also applied in decentralized trust management [Blaze et al.

1999; Li et al. 2000]. Blaze et al [1999] identified the trust management problem as a distinct and important component of security in network services and Li et al [2000] made a logic-based knowledge representation for authorization with tractable trust-management in large-scale, open, distributed systems. Delegation was used to address the trust management problem including formulating security policies and security credentials, determining whether particular sets of credentials satisfy the relevant policies, and deferring trust to third parties. Other researchers have investigated machine to machine and human to machine delegations [Wang et al. WISE01; Abadi et al. 1993]. For example, Wang et al [WISE01] proposed a secure, scalable anonymity payment protocol for Internet purchases through an agent which provided a higher anonymous certificate and improved the security of consumers. The agent certified re-encrypted data after verifying the validity of the content from consumers. The agent is a human to machine delegation which can provide new certificates. However, many important role-based concepts, for example, role hierarchies, constraints, revocation were not mentioned.

Zhang et al [2001, 2002] proposed a rule-based framework for role-based delegation including *RDM2000* model. *RDM2000* model is based on *RBDM0* model which is a simple delegation model supporting only flat roles and single step delegation. Furthermore, as a delegation model, it does not support group-based delegation.

This paper focuses exclusively on a role-based delegation model which supports group-based delegation and provides a solution of the conflicting problem with negative authorization in the role-based delegation model. We will extend our previous work and propose a delegation framework including delegation granting and revocation models, group-based delegation. To provide sufficient functions with the framework, this project will analyse how does original role assignment changes impact delegation results. This kind of group-based delegation and negative authorization within delegation framework have not been studied before.

3 Delegation framework

In this section we propose a role -based delegation model called RBDM which supports role hierarchy and group delegation by introducing the delegation relation.

3.1 Basic elements and components

RBAC involves individual users being associated with roles as well as roles being associated with permissions (Each permission is a pair of objects and operations). As such, a role is used to associate users and permissions. A user in this model is a human being. A role is a job function or job title within the organization associated with authority and responsibility. As shown in Figure 1, the relationships between users and roles, and between roles and permissions are many-to-many.

Many organizations prefer to centrally control and maintain access rights, not so much at the system administrator's personal discretion but more in accordance with the organization's protection guidelines [David et al. 1993]. RBAC is being considered as part of the emerging SQL3 standard for database management systems, based on their implementation in Oracle 7 [Sandhu 1997]. Many RBAC practical applications have been implemented [Barkley et al. 1999, Sandhu 1998].

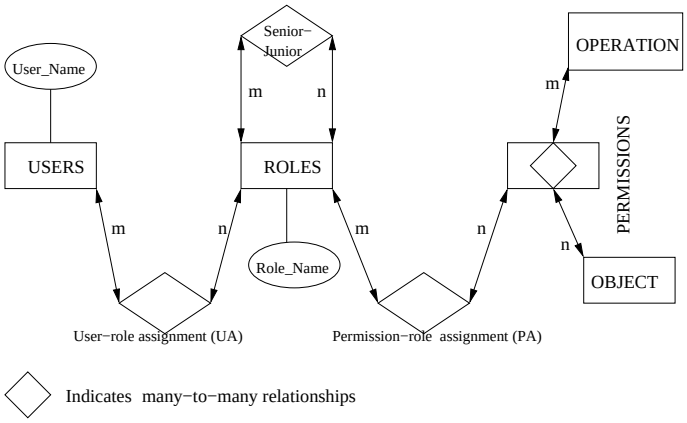

Fig. 1. RBAC relationship

A session is a mapping between a user and possibly many roles. For example, a user may establish a session by activating some subset of assigned roles. A session is always associated with a single user and each user may establish zero or more sessions. There may be hierarchies within roles. Senior roles are shown at the top of the hierarchies. Senior roles inherit permissions from junior roles. Let $x > y$ denote x is senior to y with obvious extension to $x \geq y$. Role hierarchies provide a powerful and convenient means to enforce the principle of least privilege since only required permissions to perform a task are assigned to the role.

Although the concept of a user can be extended to include intelligent autonomous agents, machines, even networks, we limit a user to a human being in our model for simplicity.

Figure 2 shows the role hierarchy structure of *RBAC* in an example of a problem-oriented system *POS* which has two projects.

The following Table 1 expresses an example of user-role assignment in *POS*.

There are two sets of users associated with role r:

Original users are those users who are assigned to the role r;

Delegated users are those users who are delegated to the role r.

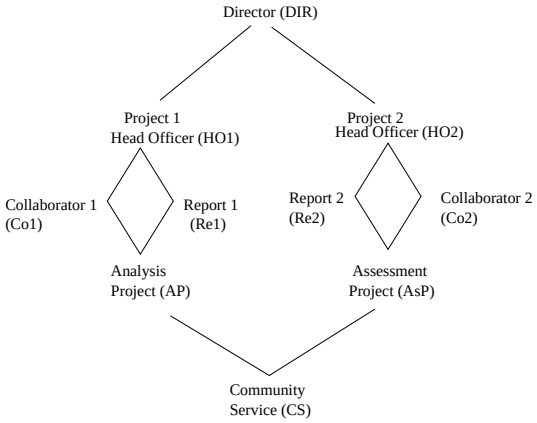

Fig. 2. Role hierarchy in *POS*

RoleName	UserName
DIR	Tony
HO1	Christine
HO2	Mike
Co1	Richard
Re1	John
CS	Ahn

Table 1. User-Role relationship

The same user can be an original user of one role and a delegated user of another role. Also it is possible for a user to be both an original user and a delegated user of the same role. For example, if Christine delegates her role HO1 to Richard, then Richard is both an original user (explicitly) and a delegated user (implicitly) of role Co1 because the role HO1 is senior to the role Co1. The original user assignment (UAO) is a many-to-many user assignment relation between original users and roles. The delegated user assignment (UAD) is a many-to-many user assignment relation between delegated users and roles.

We have the following components for RBDM model:

U, R, P and S are sets of users, roles, permissions, and sessions, respectively.

1. $UAO \subseteq U \times R$ is a many-to-many original user to role assignment relation.
2. $UAD \subseteq U \times R$ is a many-to-many delegated user to role assignment relation.
3. $UA = UAO \cup UAD$.
4. Users: $R \Rightarrow 2^U$is a function mapping each role to a set of users. $Users(r) = \{u|(u,r) \in UA\}$ where UA is user-role assignment.
5. $Users(r) = Users\_O(r) \cup Users\_D(r)$
   where
   $Users\_O(r) = \{u|\exists r' > r, (u,r) \in UAO\}$

$$Users\_D(r) = \{u | \exists r' > r, (u, r) \in UAD\}$$

3.2 Role-Based Delegation

The scope of our model is to address user-to-user delegation supporting role hierarchies and group delegations. We consider only the regular role delegation in this paper, even though it is possible and desirable to delegate an administrative role.

A delegation relation (DELR) is existed in the role-based delegation model which includes three elements: original user assignments UAO, delegated user assignment UAD, and constraints. The motivation behind this relation is to address the relationships among different components involved in a delegation. In a user-to-user delegation, there are five components: a delegating user, a delegating role, a delegated user, a delegated role, and associated constraints. For example, ((Tony, DIR), (Christine, DIR), Friday) means Tony acting in role DIR delegates role DIR to Christine on Friday. We assume each delegation is associated with zero or more constraints. The delegation relation supports partial delegation in a role hierarchies: a user who is authorized to delegate a role r can also delegate a role r' that is junior to r. For example, ((Tony, DIR), (Ahn, Re1), Friday) means Tony acting in role DIR delegates a junior role Re1 to Ahn on Friday. A delegation relation is one-to-many relationship on user assignments. It consists of original user delegation (ORID) and delegated user delegation (DELD). Figure 3 illustrates components and their relations in a role-based delegation model.

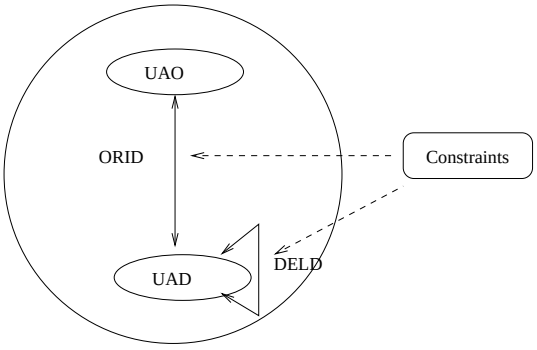

Fig. 3. Role-based delegation model

From the above discussions, the following components are formalized:

1. $DELR \subseteq UA \times UA \times Cons$ is one-to-many delegation relation. A delegation relation can be represented by $((u, r), (u', r'), Cons) \in DELR$, which means

the delegating user u with role r delegated role r' to user u' who satisfies the constraint $Cons$.
2. $ORID \subseteq UAO \times UAD \times Cons$ is an original user delegation relation.
3. $DELD \subseteq UAD \times UAD \times Cons$ is a delegated user delegation relation.
4. $DELR = ORID \cup DELD$

In some cases, we may need to define whether or not a user can delegate a role to a group and for how many times, or up to the maximum delegation depth. We only analyze one-step group delegation in this paper which means the maximum delegation path is 1. The new relation of group delegation is defined as delegation group relation (DELGR) which includes: original user assignments UAO, delegated user assignments UAD, delegated group assignments GAD, and constraints. In a user-group delegation, there are five components: a delegating user (or a delegated user), a delegating role, a delegated group, a delegated role, and associated constrains. For example, ((Tony, DIR), (Project 1, DIR), 1:00pm–3:00pm Monday) means Tony acting in role DIR delegates role DIR to All people involved in Project 1 during 1:00pm–3:00pm on Monday. A group delegation relation is one-to-many relationship on user assignments. It consists of original user group delegation (ORIGD) and delegated user group delegation (DELGD). Figure 4 illustrates components and their relations in role-based delegation model.

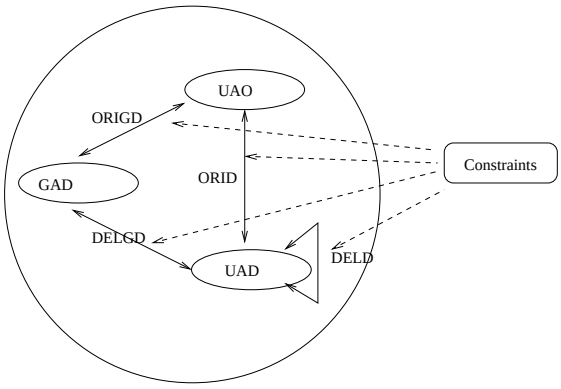

Fig. 4. Role-based group delegation model

We provide elements and functions in group delegation:

1. G is a set of users.
2. $DELGR \subseteq UA \times GA \times Cons$ is one-to-many delegation relation. A delegation relation can be represented by $((u, r), (G, r), Cons) \in DELR$, which means the delegating user u with role r delegated role r to group G who satisfies the constraint $Cons$.
3. $ORIGD \subseteq UAO \times GAD \times Cons$ is a relation of an original user and a group.

4. $DELGD \subseteq UAD \times GAD \times Cons$ is a relation of a delegated user and a group.
5. $DELGR = ORIGD \cup DELGD$

4 Delegation Authorization

This section analyses delegation authorization and provides an approach for the conflicting secure problem with negative authorization.

4.1 Authorization models

The delegation authorization goal imposes restrictions on which role can be delegated to whom. We partially adopt the notion of prerequisite condition from Wang et al. [ADC03] to introduce delegation authorization in the delegation framework.

A prerequisite condition CR is an expression using Boolean operators $'\wedge'$ and $'\vee'$ on terms r and $\bar{r}$ where r is a role and $'\wedge'$ means "and", $'\vee'$ means "or", for example, $CR = r_1 \wedge r_2 \vee r_3$.

The following relation authorizes user-to-user delegation in this framework:

$$Can\_delegate \subseteq R \times CR \times N$$

where R, CR, N are sets of roles, prerequisite conditions, and maximum delegation depth, respectively. For group-based delegation mentioned last section, $N = 1$. The meaning of $(r, cr, n) \in Can\_delegate$ is that a user who is a member of role r (or a role senior to r) can delegate role r (or a role junior to r) to any user whose current entitlements in roles satisfy the prerequisite condition CR without exceeding the maximum delegation depth n.

There are related subtleties that arise in RBDM concerning the interaction between delegating and revocation of user-user delegation membership and the role hierarchy.

Definition 1 A user-user delegation revocation is a relation $Can-revoke \subseteq R \times 2^R$, where R is a set of roles. ◇

The meaning of Can-revoke (x, Y) is that a member of role x (or a member of an role that is senior to x) can revoke delegation relationship of a user from any role $y \in Y$, where Y defines the *range of revocation*. Table 2 gives the Can-revoke relation in Figure 2.

RoleName	Role Range
HO1	[Co1, CS]

Table 2. Can-revoke

There are two kinds of revocations [Wang et al. ADC03]. The first one is weak revocation; the second one is strong revocation.

Definition 2 A user U is an explicit member of a role x if $(U, x) \in UA$, and that U is an implicit member of role x if for some $x' > x, (U, x') \in UA$. $\diamond$

Weak revocation only revokes explicit membership from a user and do not revoke implicit membership. On the other hand, strong revocation requires revocation of both explicit and implicit membership. Strong revocation of $U's$ membership in x requires that U be removed not only from explicit membership in x, but also from explicit (implicit) membership in all roles senior to x. Strong revocation therefore has a cascading effect up-wards in the role hierarchy. For example, suppose there are two delegations $((Tony, DIR), (Ahn, AP), Friday)$ and $((John, Re1), (Ahn, AP), Friday)$ and Tony wants to remove the membership of AP from Ahn on Friday. With weak revocation, the first delegation relationship is removed, but the second delegation has not yet removed. It means that Ahn is still a member of AP. With strong revocation two delegation relationships are removed and hence Ahn is not a member of AP.

4.2 An approach for the conflicting problem

In the real world of access control, there are two well-known decision policies [Bertino et al. 1997]:

a. Closed policy: This policy allows access if there exists a corresponding positive authorization and denies it otherwise.

b. Open policy: This policy denies access if there exists a corresponding negative authorization and allows it otherwise.

It is quite popular to apply closed policy in centralize management system. However, in uncentralized environment, the closed policy approach has a major problem in that the lack of a given authorization for a given user does not prevent this user from receiving this authorization later on. Bertino et al. [1997] proposed an explicit negative authorization as blocking authorizations. Whenever a user receives a negative authorization, his positive authorizations become blocked. Negative authorization is typically discussed in the context of access control systems that adopt open policy. The introduction of negative authorization brings with it the possibility of conflict in authorization, an issue that needs to be resolved in order for the access control model to give a conclusive result. The types of conflicts brought about by the negative authorization are beyond this paper. Negative authorization is rarely mentioned in $RBAC$ literature, mainly because $RBAC$ Models such as RBAC96 and the proposed NIST standard model are based on positive permissions that confer the ability to do something on holders of the permissions.

As we previously discussed a delegation relation $((u1, r1), (u', r'), Cons1) \in DELR$, which means the delegating user $u1$ with role $r1$ delegated role r' to user u' who satisfies the constraint $Cons1$. What will happen if there is another delegation $((u2, r2), (u', \neg r'), Cons2) \in DELR$ which means the delegating user $u2$ with role $r2$ rejected to delegate role r' to user u' who satisfies the constraint $Cons2$. We analyse the solution of this conflicting problem with role hierarchy. We may use one of the following policies:

1. Denial takes precedence (DTP): Negative authorizations are always adopted when conflict exists.
2. Permission takes precedence (PTP): Positive authorizations are always adopted when conflicts exists.

These two policies are too simple for enterprise collaborations since it is not an efficient solution. In enterprise environment, role hierarchy is a very important feature since a senior role has all permissions of its junior roles. It means a senior role is more powerful than a junior role. Therefore, some differences with negative authorization between a senior role and its junior role are necessary. A practical solution for the above conflicting problem is:

1. Role r' can delegate to user u' if $r1$ is senior to $r2$,
2. Role r' cannot delegate to user u' if $r1$ is junior to $r2$.

For the security reason, we suggest using DTP policy for two roles without hierarchy relationship when a conflicting problem happens. We summarize the above discussion for the conflicting problem.

1. Role r' can delegate to user u' if $r1$ is senior to $r2$,
2. Role r' cannot delegate to user u' if either $r1$ is junior to $r2$ or there is no hierarchy relationship between $r1$ and $r2$.

5 Conclusions and future work

This paper has discussed role-based delegation model and negative authorization for a solution of conflicting secure problems which may easy arise in distributed environment. We have analysed not only delegating framework including delegating authorization and revocation with constraints, but also group-based delegation. To provide a practical solution for the conflicting problem, we have analysed role hierarchies, the relationship of senior and junior role, and positive and negative authorizations. The work in this paper has significantly extended previous work in several aspects, for example, the group-based delegation and negative authorizations. It also begins a new direction of negative authorizations.

The future work includes develop algorithms based on the framework and solution proposed in this paper and the delegating revocation model including constraints.

References

1. Abadi, M., Burrows, M., Lampson, B., and Plotkin, G. 1993. A calculus for access control in distributed systems. *ACM Trans. Program. Lang. Syst.* 15, 4(Sept.), 706-734.
2. Al-Kahtani, E. and Sandhu, R. 2004. Rule-Based RBAC with Negative Authorization, *20th Annual Computer Security Applications Conference*, Tucson, Arizona, 405-415.

3. Aura, T. 1999. Distributed access-rights management with delegation certificates. *Security Internet programming.* J. Vitec and C. Jensen Eds. Springer, Berlin, 211-235.
4. Barka, E. and Sandhu, R. 2000. A role-based delegation model and some extensions. *In Proceeings of 16th Annual Computer Security Application Conference*, Sheraton New Orleans, December, 2000a, 168-177.
5. Barka, E. and Sandhu, R. 2000. Framework for role-based delegation model. *In Proceedings of 23rd National Information Systems Security Conference*, Baltimore, October 16-19, 2000b, 101-114.
6. Barkley J. F., Beznosov K. and Uppal J. 1999, Supporting Relationships in Access Control Using Role Based Access Control, *Fourth ACM Workshop on RoleBased Access Control*, 55-65.
7. Bell D.E., La Padula L.J. 1976. Secure Computer System: Unified Exposition and Multics Interpretation, *Technical report ESD-TR-75-306*, The Mitre Corporation, Bedford MA, USA.
8. Bertino, E. P. Samarati, P. and S. Jajodia, S. 1997. An Extended Authorization Model for Relational Databases, *In IEEE Transactions On Knowledge and Data Engineering*, Vol. 9, No. 1, 145-167.
9. Blaze, M. Feigenbaum, J., Ioannidis, J. and Keromytis, A. 1999. *The role of trust management in distributed system security. Security Internet Programming.* J. Vitec and C. Jensen, eds. Springer, Berlin, 185-210.
10. David F. F., Dennis M. G. and Nickilyn L. 1993. An examination of federal and commercial access control policy needs, *In NIST NCSC National Computer Security Conference*, Baltimore, MD, 107-116.
11. Feinstein, H. L. 1995. Final report: NIST small business innovative research (SBIR) grant: role based access control: phase 1. Technical report. *SETA Corporation.*
12. Ferraiolo, D. F. and Kuhn, D. R. 1992. Role based access control. *The proceedings of the 15th National Computer Security Conference*, 554-563.
13. Lampson, B. W., Abadi, M., Burrows, M. L., and Wobber, E. 1992. Authentication in distributed systems: theory and practice. *ACM Transactions on Computer Systems* 10 (4), 265-310.
14. Li, N. and Grosof, B. N. 2000. A practically implementation and tractable delegation logic. *IEEE Symposium on Security and Privacy.* May, 27-42.
15. Sandhu, R. 1997. Rational for the *RBAC*96 family of access control models. *In Proceedings of 1st ACM Workshop on Role-based Access Control*, 64-72.
16. Sandhu R. 1998. Role activation hierarchies, *Third ACM Workshop on RoleBased Access Control*, Fairfax, Virginia, United States, ACM Press, 33-40.
17. Sandhu R. 1997. Role-Based Access Control, *Advances in Computers*, Academic Press, Vol. 46.
18. Wang, H., Cao, J., and Zhang, Y. 2005. A flexible payment scheme and its role based access control, *IEEE Transactions on Knowledge and Data Engineering*(IEEE05), Vol. 17, No. 3, 425-436.
19. Wang, H., Cao, J., Zhang, Y., and Varadharajan, V. 2003. Achieving Secure and Flexible M-Services Through Tickets, In B.Benatallah and Z. Maamar, Editor, IEEE Transactions Special issue on M-Services. *IEEE Transactions on Systems, Man, and Cybernetics. Part A*(IEEE03), Vol. 33, Issue: 6, 697- 708.
20. Wang, H., Zhang, Y., Cao, J., and Kambayahsi, J. 2004. A global ticket-based access scheme for mobile users, special issue on Object-Oriented Client/Server Internet Environments. *Information Systems Frontiers*, Vol. 6, No. 1, pages: 35-46. Kluwer Academic Publisher.

21. Wang, H., Cao, J., and Zhang, Y. 2002. Formal Authorization Allocation Approaches for Role-Based Access Control Based on Relational Algebra Operations, *The 3nd International Conference on Web Information Systems Engineering (WISE'2002)*, Singapore, pages: 301-310.
22. Wang, H., Sun, L., Zhang, Y., and Cao, J. 2005. Authorization Algorithms for the Mobility of User-Role Relationship, *Proceedings of the 28th Australasian Computer Science Conference (ACSC05)*, Australian Computer Society, 167-176.
23. Wang, H., Cao, J., Zhang, Y. 2003. Formal authorization approaches for permission-role assignment using relational algebra operations, *Proceedings of the 14th Australasian Database Conference*(ADC03), Adelaide, Australia, Vol. 25, No.1, 125-134.
24. Wang, H., Cao, J., Zhang, Y. 2001. A Consumer Anonymity Scalable Payment Scheme with Role Based Access Control, *Proceedings of the 2nd International Conference on Web Information Systems Engineering (WISE01)*, Kyoto, Japan, 73-72.
25. Yao, W., Moody, K., and Bacon, J. 2001. A model of OASIS role-based access control and its support for active security. *In Proceedings of ACM Symposium on Access Control Models and Technologies (SACMAT)*, Chantilly, VA, 171-181.
26. Zhang, L., Ahn, G., and Chu, B. 2001.A Rule-based framework for role-based delegation. *In Proceedings of ACM Symposium on Access Control Models and Technologies (SACMAT 2001)*,Chantilly, VA, May 3-4, 153-162.
27. Zhang, L., Ahn, G., and Chu, B. 2002. A role-based delegation framework for healthcare information systems. *In Proceedings of ACM Symposium on Access Control Models and Technologies (SACMAT 2002)*. Monterey, CA, June 3-4, 125-134.