International Journal of Hybrid Information Technology
Vol.8, No.7 (2015), pp.27-36
http://dx.doi.org/10.14257/ijhit.2015.8.7.03

A Policy Conflict Detecting Algorithm of RBAC Based on Concept Lattice Model

Daojun Han, Lei Zhang, Xiajiong Shen and Peiyan Jia

Institute of Data and Knowledge Engineering, Henan University, KaiFeng, 475004, China
E-mail:{hdj,zhanglei,shenxj,jpy}@henu.edu.cn

Abstract

Access control policy conflict detecting is an important issue in the usage of information system. To solve the problem that the expression of existing common Role-Based Access Control(RBAC) policy conflict detecting is not intuitive and the corresponding algorithm is not efficient, meanwhile, we observed that the concept lattice model has natural advantages being a data representation method and is easy to be combined with RBAC. Thus, this paper introduce the concept lattice model into RBAC policy conflict detecting algorithm, aim at the problems of jurisdiction conflict, static role conflict, and user conflict, utilizing the sub lattice on the basis of use two formal contexts to denote user-role relation and role-permission relation respectively, we provide an intuitive and efficient detecting algorithm. Experimental result shows the efficient of our algorithm.

Keywords: concept lattice; Role-Based Access Control (RBAC); policy; conflict detecting

1. Introduction

To ensure the system resources using can be under control and legal, access control is a method to explicitly allow or limit the ability and range of the access from subject to object, and it has become an important technique of information security. Nowadays, there are many kinds of access control model, such as the role based access control(RBAC), discretionary access control(DAC), mandatory access control(MAC), usage control(UCON), attribute-based access control(ABAC), and task-based access control(TBAC). In these popular models, RBAC has huge influence and widely applied because of RBAC achieving the logical separation of user and permission, convenient to manage. Furthermore, role is an important attribute, and RBAC has become the foundation of access control model like ABAC [1]. In the application process of RBAC, the policy conflict detecting is an important issue, which attracted researchers' extensive attention [2-4]. Cheng Xiangran formalized defined five kinds of RBAC policy conflict, analyzed the reason of policy conflict, bring up and take simulation test of an integrated policy conflict detecting algorithm, aiming at the conflict issue which is caused during RBAC model is applying safety principle like duty separation and minimum privilege [5]. Liu Qiang revealed a series of logical and management issues: pseudo three valued logic in authorization status, source of managerial authority, administrator's accrual synchronization, meaning of permission to leak, problem of authorization decision supported model, which provided theoretical support of raising the safety and applicability and reducing the complicacy of RBAC model[6].

In order to solve the problem that the expression of existing RBAC policy conflict detecting is not intuitive and the algorithm is not efficient, we observed that the concept lattice model has advantages being a data representation method and is easy to be combined with RBAC [7-9]. So we introduce the concept lattice model to RBAC policy

ISSN: 1738-9968 IJHIT

conflict detecting algorithm, and provide an intuitive and efficient detecting algorithm. Firstly, we build concept lattice according to the core factor of RBAC: user, role, and limit of authority, which can visually express the relationship between user and role, role and limit of authority. Then, we convert different kind of conflict to rule, make use of extension and intension in formal concept, and do conflict detecting. Experimental result shows availability of the algorithm.

2. Background

2.1 Role-Based Access Control

For access control purpose, it is much important to know what user's organizational responsibilities are, rather than who the user is. Thus, RBAC is suitable. Role-Based Access Control, RBAC, introduce the concept of role into user and permission. User is relevant to specific one or multiple roles. Role is relevant to one or multiple permissions and can be created or revoked on the base of operational need. Users who register in system can dynamically activate roles according to their own need. The RBAC has greatly simplified permission management for it implements the logical separation between user and permission by means of conferring or revoking permissions to a role instead of the user.

The CORE RBAC model was released by ANSI in 2004[10], and the main components are shown in Figure 1.

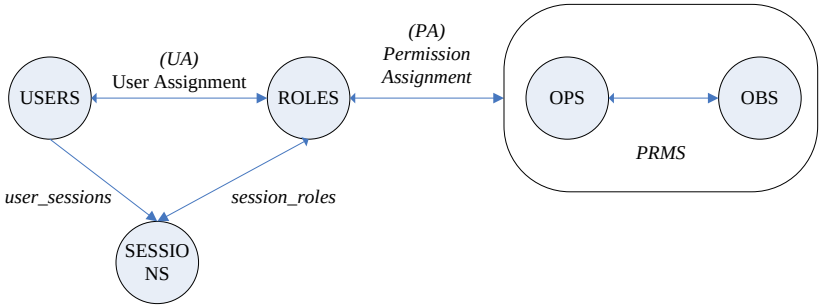


Figure 1. The Main Components of Core RBAC

CORE RBAC has the following components:

- USERS, ROLES, PRMS, and SESSIONS denote user set, role set, permission set and session set. Commonly used objects USERS, ROLES, PRMS are shorthand for U, R, P, there is P=2(OPS×OBS)
- PA P×R, which is a many-to-many permission-role relation;
- UA U×R, a many-to-many user-role relation;
- User: S→U, a function mapping each session si to a single user, user (si). Note that user(si) is constant during the session lifetime;
- Roles: S→2R, a function mapping each session si to the subset of all roles, roles (si) {r| (user (si), r) UA} (which can change over time) and there are permissions in session si.

2.2 RBAC Security Constraint and Conflict

Nowadays, the definition of RBAC security constraint and conflict is not yet complete. Aiming at the conflict issue which is caused during RBAC model is applying safety principle like duty separation and minimum privilege, Cheng Xiangran defined five kinds of RBAC policy conflict as permission, static role, dynamic role, user and role loop inheritance, the main definition is as follows [5].

Definition 1 (conflict permission constraint, permission conflict PRMS_CF) conflict permission constraint cp= (ps,n), ps={prms1, prms2,…, prms|ps|}⊆PRMS, |ps|≥n≥2, it means at most n-1 number of permissions in ps can be assigned to one role. If n=2, permissions in ps are mutually exclusion. The conflict permission set is denoted by CP= {cp1, cp2… cp|CP|}.

Definition 2 (static conflict role constraint, static role conflict SR_CF) static conflict role constraint dcr= (rs,n), cr=(rs,n), rs={r1,r2,…,r|rs|}⊆ROLES, |rs|≥n≥2, a user can own at most n-1 number of roles in rs. SCR= {scr1, scr2… scr|SCR|} is static conflict role constraint set.

Definition 3 (dynamic conflict role constraint, dynamic role conflict DR_CF) conflict user constraint dcr= (rs,n), rs={r1, r2,…,r|rs|}⊆ROLES, |rs|≥n≥2, it means one session can activate at most n-1 number of roles in rs. If n=2, roles in rs are mutually exclusion. Dynamic conflict role constraint set denoted by DCR= {dcr1, dcr2,…,dcr|DCR|}.

Definition 4(conflict user constraint, user conflict USER_CF) conflict user constraint is that cu= (us,n,r), where us={u1,u2,…,u|us|}⊆USERS, r∈ROLES, |us|≥n≥2, it means that a role r is assigned at most n-1 number of users in us. If us=USERS, conflict user constraint can express cardinality constraint. Conflict user set denoted by CU= {cu1, cu2,…,cu|CU|}.

Definition 5 (role inheritance path, role inheritance loop conflict RIC_CF) according to transitivity of inheritance, role inheritance path (n≥1) denote the transmit inheritance relationship from role ri to role rj. If exist a role inheritance path, role inheritance loop conflict is occurring.

2.3 Concept Lattice

Concept lattice is main data structure in formal concept analysis theory, and it is a common data analysis tool. Every node in concept lattice is a formal concept, it is made up with two parts: the one is extension, means instance of concept; the other is intension, means expression of concept, also means common characters of concept instance. In addition, concept lattice vividly and compactly give expression to generalization and specialization relationship between concepts.

Given a context as triples K= (G, M, R), G is objects set, M is attribute set, R is a binary relation between G and M. There is only one ordered set corresponding with K, and a lattice structure is generating according to the ordered set. The lattice L constructed by context (G, M, R) is a concept lattice. Each node in lattice L is an ordered pair (named formal concept or concept), denoted as (X, Y), X∈P (G) is extension of concept (P (G) is power set of G); Y∈P (M) is intension of concept. Every ordered pair is complete about relationship R, has two characters.

$$(1)X= \{x \in G \mid y \in Y, xRy\};$$

$$(2)Y= \{y \in M \mid x \in X, xRy\}。$$

In context K, we define two mapping f：P(G)→P(M) and g：P(M)→P(G),

$$\forall G_1 \subseteq G : f(G_1) = \{m \mid \forall x \in G_1(xRm)\}$$

$$\forall M_1 \subseteq M : g(M_1) = \{x \mid \forall m \in M_1(xRm)\}$$

They are called Galois connection between P (G) and P (M). For two-tuples (G1, M1) ∈P (G) ×P (M), if satisfy G1=g (M1) and M1=f (G1), then this two-tuples is a formal concept of information table K. For given formal concept C=(G1,M1), G1 is extension of formal concept C, denoted by Extension(C), M1 is intension of formal concept C, denoted by Intension(C). All formal concept sets of K are denoted by CS (K).

An ordered relationship can be built between those concept lattice nodes. Given H1=(X1, Y1) and H2=(X2, Y2), then H1<H2 Y1 Y2, lead order signify H1 is father node

or direct generalization of H2. Hasse diagram of lattice can be created by the ordered relationship: if H1<H2 and there is no other element H3 meet H1<H3<H2, then exist an edge from H1 to H2. Table 1 show a formal context, 1 in row u and line m means uRm, in which u is object and m is attribute. There is G={1,2,3,4} and M={a, b, c, d, e}, and R describe elements in G have attribute set in M. Figure 2 shows a concept lattice created from K1, which represented by Hasse diagram.

Table 1. A Context K1

	A	*B*	*c*	*d*	*e*
1		1		1	1
2	1	1			1
3	1		1		1
4	1	1	1	1	1

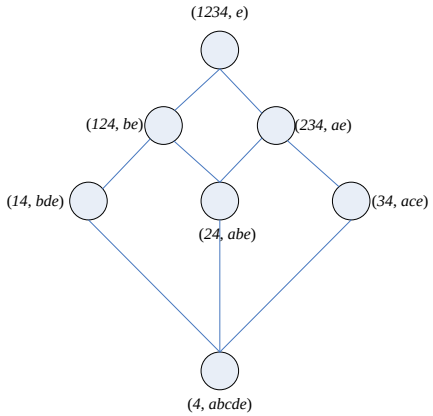


Figure 2. A Concept Lattice And Its Hasse Diagram Corresponding With K1

3. Policy Conflict Detecting Algorithm Based on Concept Lattice Model

3.1. Concept Lattice Express RBAC

Concept lattice can express generalization and specialization relationship among formal context, also, extension and intension set of formal concept describe the mapping relationship between objects and attribute. Those two characters of concept lattice are familiar with RBAC: (1) according to permissions included, some roles have inclusion relations among themselves; (2) mapping relations exist between role and permission as well as role and user. In this way concept lattice model can closely integrated with RBAC and do some research and operation in RBAC on the basis of concept lattice model [8-9].

It is known that the core factors in RBAC are user, role, and permission from the analysis of the above. Role plays an important role like as the uses of bridge and it make the user and permission logical disjunction. We can use two formal contexts K1 and K2 to denote user-role relation and role-permission relation respectively. Assume there exist user set Suser, role set Srole and permission set Sright in system.

We construct K1=(G1,M1,R1), G1={x|x∈Suser}, M1={x|x∈Srole},R1={(x,y)|P(x,y), x∈Suser, y∈Srole }, where P(x,y) means user x has role y.

In a similar way, we construct K2=(G2,M2,R2), G2={x|x∈Srole}, M2={x|x∈Sright}, R2={(x,y)|Q(x,y), x∈Srole, y∈Sright}, where Q(x,y) means role x has permission y.

Apply concept lattice construction algorithm on K1 and K2, and denote core factor in RBAC by concept lattice model, then we obtain CS (K1) and CS(K2). We can acquire the contain relation of user-role and role-permission from CS(K1) and CS(K2) by using the relationship between extension and intension and partial order organization in formal concept, which is easy for quick search of corresponding information.

3.2 Policy Conflict Analysis and Detecting Algorithm

During the process of applying user and permission, that is, apply one or some roles to a user and apply one or some permissions to an role, adding role inheritance and role activated, we can implement policy conflict detecting of the system current session state. In this section, we will introduce detecting idea for different conflicts, then provide corresponding detecting algorithm.

3.2.1 Policy Conflict Analysis and Detecting Idea: Aiming at five kinds of conflict definitions in section 2.2 and analyzing the practical significance of conflict style, we provide a total detecting idea based on use concept lattice model to represent RBAC model, which is shown as Table 2.

Table 2. Total Conflict Detecting Idea Based On Use Concept Lattice Model to Represent RBAC

Conflict style	Conflict explanation	Detecting idea based on concept lattice model
Definition 1: permission conflict *PRMS_CF*	If a role gets more than *n*-1 permissions in *ps* directly or indirectly, permission conflict is occurring.	According to the method of concept lattice explanation for role-permission, the detecting method of permission conflict is to analyze node and get the sub-lattice in $CS(K_2)$, then apply the detect method to system.
Definition 2: static role conflict *SR_CF*	For static conflict role constraint *scr* = (*rs*,*n*), if there are more than *n*-1 roles are assigned to one role in *rs*, static role conflict is occurring.	According to the method of concept lattice explanation for user-role, the detecting method of static role conflict is similar with permission conflict. That is, analyze nodes of concept lattice in *CS* (K_1), then get sub-lattice and use it in detect method.
Definition 3: dynamic role conflict *DR_CF*	For dynamic conflict role constraint *dcr*= (*rs*,*n*), if there are more than *n*-1 roles are activated by one session, dynamic role conflict is occurring.	In a session, according to concept lattice explanation of role, if the smallest extension is null, we can get a sequence of smallest intension and get the union set to intersect with *rs*.
Definition 4: user conflict *USER_CF*	For every conflict user set *cu*= (*us*,*n*,*r*), if there are more than *n*-1 users can get role *r* in *us*, user conflict is occurring.	In a session, we find the biggest concept node relevant to *r* in *CS* (K_1), then explain it according to extension of concept node. If the number of biggest node extension is small than *n*-1, then it does not exist.
Definition 5: role inheritance loop conflict *RIC_CF*	If a role inherits itself indirectly, role inheritance loop conflict is occurring.	Role inheritance loop conflict can be detect when building concept lattice *CS* (K_2) by constrains of set union operate. If exist inheritance loop, it is a formal concept in essence.

Conflict styleConflict explanation Detecting idea based on concept lattice model

Definition 1: permission conflict PRMS_CF If a role gets more than n-1 permissions in ps directly or indirectly, permission conflict is occurring. According to the method of concept lattice explanation for role-permission, the detecting method of permission conflict is to analyze node and get the sub-lattice in CS(K2), then apply the detect method to system.

Definition 2: static role conflict SR_CF For static conflict role constraint scr = (rs,n), if there are more than n-1 roles are assigned to one role in rs, static role conflict is occurring. According to the method of concept lattice explanation for user-role, the detecting method of static role conflict is similar with permission conflict. That is, analyze nodes of concept lattice in CS (K¬1), then get sub-lattice and use it in detect method.

Definition 3: dynamic role conflict DR_CF For dynamic conflict role constraint dcr= (rs,n), if there are more than n-1 roles are activated by one session, dynamic role conflict is occurring. In a session, according to concept lattice explanation of role, if the smallest extension is null, we can get a sequence of smallest intension and get the union set to intersect with rs.

Definition 4: user conflict USER_CF For every conflict user set cu= (us,n,r), if there are more than n-1 users can get role r in us, user conflict is occurring. In a session, we find the biggest concept node relevant to r in CS (K¬1), then explain it according to extension of concept node. If the number of biggest node extension is small than n-1, then it does not exist.

Definition 5: role inheritance loop conflict RIC_CF If a role inherits itself indirectly, role inheritance loop conflict is occurring. Role inheritance loop conflict can be detect when building concept lattice CS (K2) by constrains of set union operate. If exist inheritance loop, it is a formal concept in essence.

3.3.2 Policy Conflict Detecting Algorithm: 1 Sub-lattice Definition: There is a core application, i.e., solving the sub-lattice of one node and denote by SubLattice(x) during the process of policy conflict detecting based on concept lattice model. It satisfies the following definition.

Definition 6 sub-lattice x: visits from a node x in CS (K) to the lower node until the least element. All these visited nodes can form a lattice L0, in which node x is the greatest element, called sub-lattice x, denoted by *SubLattice*(x).

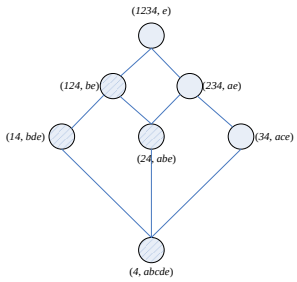


Figure 3. A Concept Lattice and Sublattice(X) Example

Node with shadow is *SubLattice(*x) in Figure 3, where x is (124, be).

Refer to breadth-first calendar calculation method, the sub-lattice building algorithm is explained as Table 3.

Table 3. The Sub-Lattice Building Algorithm

Input: lattice *L*, node *X*
Output: lattice L_0

Begin
NodeSet :={*X*}, where *NodeSet* is the node set of L_0;
PartialOrderSet:=$\varnothing$, where *PartialOrderSet* is the partial order set of L_0;
IF *X* is the greatest element of *L*， THEN L_0:=*L*, return L_0;
Else get the sub nodes of *X*， obtain set Children(*X*)， *NodeSetTemp*:= Children(*X*)；
Label 1： IF *NodeSetTemp* is not empty
THEN for $\forall Y \in$ *NodeSetTemp*
IF $Y \notin$ *NodeSet*；
THEN add *Y* to *NodeSet*；
Add (*X*,*Y*) to *PartialOrderSet*
Obtain Children(*Y*)， *NodeSetTemp*:= Children(*Y*)， turn to *Label* 1；
ELSE return L_0；
End

4. Conflict Detecting Algorithm

(1)Policy Conflict Detecting Algorithm

Theorem 1: in a session, for the node X satisfied Intension(X) =ps. If Children(X) is not empty, roles in sub concept of X can't be assigned role set.

Proof: The proof process can be divided into two steps.

Firstly, we prove there exist X in CS (K2). Because K2=(G2,M2,R2), G2={x|x∈Srole}, M2={x|x∈Sright},R2={(x,y)|Q(x,y), x∈Srole, y∈Sright}, it also means K2 describe the incidence relation of role and permission. All role-permission relations can be described by formal concept for the completeness of concept lattice. So node X exists in CS (K2).

Secondly, we prove roles in sub concept of X cannot be assigned role set. For any node in Y SubLattice(X), there is Y SubLattice(X) according to definition and construction algorithm of sub-lattice. Because Intension(X) =ps, ps Intension(Y). So roles in Y cannot be assigned roles.

□

Theorem 1 explain that, if we construct SubLattice(X) by using elements in ps as intension set retrieval node X, elements in these sets cannot be assigned roles during authorization.

According to theorem 1, the permission conflict detecting algorithm can be describe as follows, in which DenyRoleSet is forbidden assign roles set and cannot assign any role in the set for user during authorization.

1) Select node X in CS (K2), satisfy Intension(X) =ps; DenyRoleSet: =;
2) According to sub-lattice information constructed in advance, get SubLattice(X);
3) If Extension(X) =, make FatherNodeSet(X) be the father node set of node X;

(2)Static Role Conflict Detecting Algorithm

According to the definitions of static role conflict and policy conflict, there is similarity among them. On the basis of user-role relation in CS(K1) and refer to permission conflict detecting algorithm, the static role conflict can be described as below, in which StaticUserSet is forbidden use user set in the session.

1. Select node X in CS (K1), satisfy Intension(X) =rs ; StaticRoleSet:=;
2. According to sub-lattice information constructed in advance, get SubLattice(X);

3. If SubLattice(X), for any node Y SubLattice(X), StaticUserSet: = StaticUserSet∪Extension(Y).

(3) Dynamic Role Conflict Detecting Algorithm

On the basis of definition of dynamic role conflict and the user-role relation described by CS (K1) in one session, detecting algorithm is as follows by means of concept lattice's completeness:

1) Select the least element X, TempResult: =;

2) If Extension(X), turn to 5);

3) If Extension(X) =, make FatherNodeSet(X) be father nodes set of node X;

4) For node Y FatherNodeSet(X), TempResult: = Intension(Y) ∪TempResult ;

5) TempResult: = Intension(X) ∩rs. If ||TempResult||≥n, there exist dynamic role conflict in the session. Otherwise there is no conflict.

(4) User Conflict Detecting Algorithm

According to the definition of user conflict and the user-role relation explained by CS(K1), we can find the first concept node X obtain role r by means of concept intension set ranking. If ||Extension(X) ||>n-1, there exist user conflict.

(5) User Inheritance Loop Detecting Analysis

User inheritance loop detecting can be eliminate after analysis, also means two roles of loop inheritance is one concept.

Proof: assume there exist role inheritance loop, then Intension(R) Intension (Rk, 1) Intension (Rk, 2) … Intension (Rk,n) Intension(R). The expression is false on basis of set inclusion definition.

That is to say that role inheritance loop is commonplace and can be detected and eliminated directly when using concept lattice model to express RBAC.

4. Experimental Analysis

A human resource management system is mainly due with employee, cheek, and payment. In this testing environment, the number is 56, 146, and 23 for user, permission, and role respectively. For role loop inheritance can be detected directly, also means that if there exist roles in inheritance sequence are the same formal concept during the role inheritance, we consider it is role loop inheritance.

In this test, we mainly test the permission conflict situation. We build five cases whose conflict permission constraint amount is 3, 5, 10, 20, 30 respectively, and aiming at each cases, we test the average probabilities of permission conflict if the amount of permission assignments is 100, 200, 300, 400, and 500 respectively.

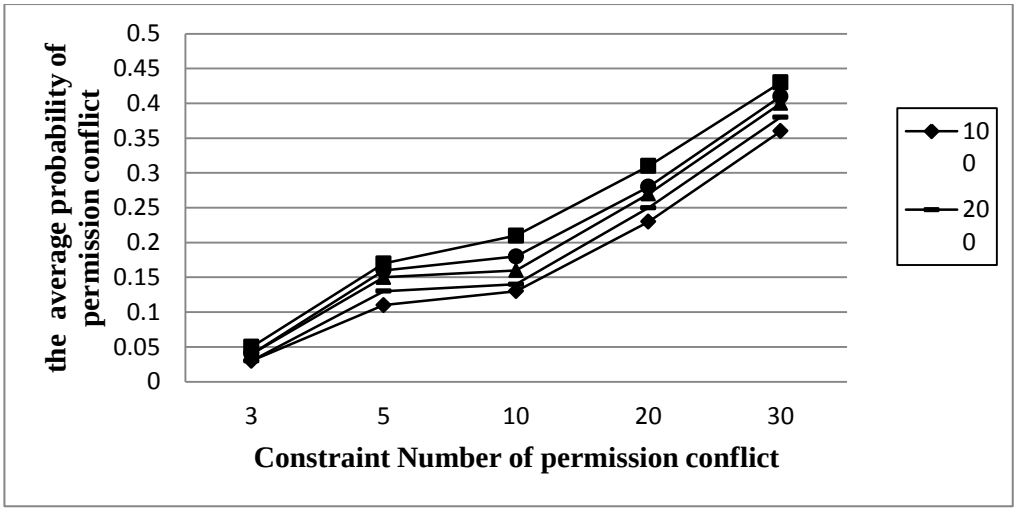


Figure 4. Relation of Permission Conflict Probability and Conflict Constraint

From the experimental results in Figure 4, we can find that, the probability of permission conflict is rising along with the increase of permission assignment amount. If the number of permission assignment is identical, permission conflict probability also increase as long as the number of conflict permission constraint is rising. To be sure that the sub-lattice X of two concept lattices which is used in the test of permission conflict based on concept lattice model can be constructed in advance and repeated used. Therefore, the average probability of permission conflict during permission assignment does not contain the time complexity of algorithm.

5. Conclusion

Access control policy conflict detecting is the core of information system usage. Concept lattice as a common method of data expression and analysis, has its natural advantages, and can be combined with RBAC easily. In this study, we introduce concept lattice model to RBAC policy conflict detecting algorithm for the detecting method is not intuitive and the algorithm is not efficient. We analyze permission conflict, static role conflict, user conflict and use two formal contexts to express user-role and role-permission relations respectively. Then, provide an intuitional and efficient detecting algorithm. The future study will make use of concept lattice model and surround the aspect of conflict detecting during automation authorization.

Acknowledgements

We thanks for the support of National Natural Science Foundation of China (61272545, 61402149) and Scientific and technological project of Henan Province (142102210390, 14A520026).

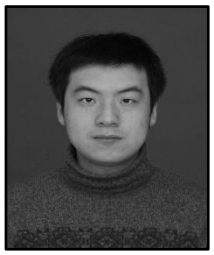
References

[1] HAN Daojun, GAO Jie, ZHAI Haoliang, LI Lei. Research Development of Access Control Model [J]. Computer Science, 37(11):29-33(**2010**)
[2] Yuan Chunyang, He Yeping, He Jianbo, Zhou Zhouyi. Formal Specification and Proof of the RBAC with Constraints of Conflict [J]. Journal of Computer Research and Development. 43(suppl.):498-508(**2006**)
[3] H.T. Pham, N.-T. Truong, and V.-H. Nguyen, Analyzing RBAC security policy of implementation using AST[C], Proceedings of the 2009 International Conference on Knowledge and Systems Engineering (KSE '09), Oct'2009

[4] Fuchs, L., Pernul, G., Sandhu, and R.: Roles in information security: A survey and classification of the research area [J]. Computer and Security. (**2011**)
[5] CHENG Xiangran, CHEN Xingyuan, ZHANG Bin, YANG Yan. Research on RBAC Policy Conflict and Its Detection Algorithm [J]. Computer Engineering, 36(18):135-137(**2010**)
[6] LIU Qiang, WANG Lei, HE Lin. Research on a Series of Problems in RBAC Model [J]. Computer Science, 39(11):13-18(**2012**)
[7] JIAO Suyun, LIU Yanheng, WEI Da. Dynamic policy access model based on classification concept lattice [J]. Journal on Communications, 32(2):27-33(**2011**).
[8] HAN Daojun, HOU Yane, JIA Peiyan. Roles Acquisition Based on Concept Lattice Model [J]. Computer Science, 39(12):162-166. (**2012**)
[9] JIA Xiaoming, HAN Daojun, Wang Baoxiang. Roles Evaluation Based on Concept Lattice in RBAC [J]. Journal of Henan University (Natural Science). 41(3):308-311(**2011**)
[10] ANSI, ANSI INCITS 359-2004 for Role Based Access Control (**2004**).

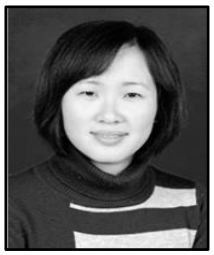
Authors

Daojun Han, he received Ph.D. degree in 2011. He is an associate professor at the School of Computer and Information Engineering, Henan University, Kaifeng, China. His major study fields include information security, knowledge discovery and spatial data process.



Lei Zhang, he received the M. Sc. degree from Henan University, Kaifeng, China, in 2006. He is a Ph.D. candidate of Harbin Institute of Technology, Harbin, China. His research interests include formal concept analysis, data mining, and information security.

Xiajiong Shen, he received Ph.D. degree in 2006. He is a professor at the School of Computer and Information Engineering, Henan University, Kaifeng, China. His major study fields include software engineering, knowledge discovery and spatial data process.



Peiyan Jia, she received the M. Sc. degree from Henan University, Kaifeng, China, in 2006. Her research interests include formal concept analysis, data mining, and information security.

