International Journal of Security and Its Applications
Vol. 11, No. 6 (2017), pp.1-18
http://dx.doi.org/10.14257/ijsia.2017.11.6.01

MIRBAC: A Role-Based Access Control Model for Multi-Domain Interoperability

Ting Cai[1*] and Jun-Zhan Wang[2]

[1]*College of Mobile Telecommunications, Chongqing University of Posts and Telecommunications, Chongqing, China*
[2]*Logistics Academy, PLA, Beijing, China*
ct_dolphin@163.com, wjz72506@gmail.com

Abstract

How to achieve both cross-domain authorization and access control in a multi-domain environment and ensuring local autonomy and security are hot research field of network security. Due to the centralized management, traditional access control has been unable to meet the security needs of cross-domain interoperability under a distributed environment. In this article, we introduce three types of inter-domain role relations, such as transitive mapping, non-transitive mapping and restricted access, extend the standard single-domain RBAC model to a multi-domain interoperable environment, and establish a role-based access control model based on multi-domain interoperability (MIRBAC). Compared with the prior studies, MIRBAC model supports separation of duties constraint under multi-domain environments, the security and management flexibility of inter-domain authorization is greatly improved. Moreover, based on MIRBAC model, we further research detection method of security violation during interoperability, propose a complete security conflict detection algorithm according to various conflict types caused by implementing interoperability activities, and conduct computational complexity analysis and case analysis of our proposed algorithm. Finally, we develop a prototype system based on the definitions of our proposed model to conduct experimental studies to demonstrate the feasibility and performance of our approach.

Keywords*: MIRBAC, secure interoperation, conflict detection, multi-domain environment*

1. Introduction

With the rapid development of grid computing, P2P network, cloud computing and other new distributed technologies, information system gradually evolves from closed, static and centralized system to open, dynamic and distributed system. Collaboration among different systems is becoming of vital importance since it supports building large-scale distributed environments, sharing resources and collaborating work. Business alliance, CMS (Cooperative Medical System), collaborative e-government and other typical applications are increasing, however, the security issue of sensitive resource sharing can not be ignored [1]. In recent years, a large-scale distributed environment is generally been divided into multiple highly autonomous management domains, then, how to achieve cross-domain authorization management and access control of resources during secure interoperation becomes an active area of research for network security [2].

An important issue in multi-domain secure interoperation is access control. Currently, access control models used in autonomous domains are mainly DAC, MAC and RBAC. DAC has security flaws and poor scalability, and MAC has limited applications, which lead both not widely available. RBAC in itself is neutral strategy, and is capable of

* Corresponding Author

ISSN: 1738-9976 IJSIA

expressing policies such as role hierarchy, separation of duty and least privilege, *et al* Moreover, it can implement both DAC and MAC policies by easy configuration, and has become the most widely used access control model. Therefore, study on secure interoperation based RBAC in multi-domain environments has positive practical significance.

In recent years, secure interoperation based RBAC has received considerable attention from researchers. In 1999, Kapadia *et al*, discussed the interoperable issue between two object request brokers with RBAC policy [3, 4], and first proposed an idea of using a method of dynamic role translation to establish inter-domain access control policy relations and to achieve cross-domain access. In 2000, the IRBAC 2000 [5] was presented as a model that can quickly establish a flexible policy for dynamic role translation, while the AIRBAC 2000 [6] was presented as its administration model. To achieve the direct translation of two roles from different domains, the model set up role mapping relations between them and made meaningful access control decision for secure interoperation. Shehab *et al*, proposed a protocol SERAT [7] for secure interoperation, which is based on the concept of access path and access path's constraint. Shafiq *et al*, proposed a framework [8] of integrating the heterogeneous RBAC policies into a consistent global policy though inter-domain role mapping, and focused on the issue of conflict resolution during policy integration. In 2006, the RBIAC [9] was presented as a role-based access control model for interoperability in distributed environments. The model introduced a transitive mapping relation into inter-domain role sets, and realized SSD constraint and DSD constraint in an interoperable environment. In 2009, a secure interoperation mechanism based on trust and constraint between domains was proposed [10]. The mechanism is capable of solving the violation of SSD constraint, by sending all autonomous domains' SSD relation to their trusted domains, to realize the inter-domain transition of the SSD constraint during role mappings. Ye and Guo proposed the concept of the threshold property and role mapping threshold property [11], which describe the different trust levels between different domains and achieve a more fine-grained access control.

Although the above research achievements, to some extent, solve the problem of the authorization management and access control for RBAC worked in an interoperable environment, there still exist serious problems in the following three aspects:

① In a multi-domain environment, the interoperation based RBAC in the autonomous system cannot provide the extra-domain user with effective support for security constraint.

② In a multi-domain environment, the inter-domain authorization management lacks of adequate security and flexibility.

③ In a multi-domain environment, there lacks the effective security violation detection method for secure interoperation, in particular, lacks the conflict detection algorithm that is capable of detecting all kinds of conflicts and easy to implement.

To solve these problems, in this article we first propose a MIRBAC model, which is an extended RBAC model under the secure interoperation in multi-domain environments. Our proposed model is capable of supporting various effective security constraints in a multi-domain environment, and also supporting the inter-domain restricted access and inter-domain non-transitive mapping, which enforces the security and flexibility of the inter-domain authorization management. Secondly, based on the MIRBAC model, we further study and analyze the interoperable conflict types, and present a complete detection algorithm that can detect all types of security violations. Finally, we design the real application scenarios to illustrate the MIRBAC model applies to the multi-domain security interoperability.

2. Preliminaries

This section provides prerequisite information about the RBAC model since it is being used as our basis model for achieving collaboration among multi-domain systems. Secure interoperation is being discussed in the context of RBAC since it is an issue of vital importance in multi-domain environments.

2.1. Traditional RBAC Model

In RBAC, permissions are granted to roles rather than direct to users, who obtain operating permissions though role assignments to achieve authorization. Since roles in systems have relative stability than users, and have a more intuitive understanding, thereby greatly reducing the work complexity and the workload of system administrator.

Standard RBAC model consists of four components: Core RBAC, Hierarchical RBAC, SSD (static separation of duty) constraints and DSD (dynamic separation of duty) constraints [12]. Core RBAC is the most basic model. Hierarchical RBAC joins the concept of role hierarchy based on basic model, and defines the inheritance relationships between roles. Constraint RBAC regulates some roles that can not be held by the same user, so that restrictions should be added to users when a role obtains permissions or is assigned to roles. In order to provide complete functions support for RBAC model in a multi-domain environment, we will discuss on the basis of these four components, which formal descriptions are as follows.

Definition 1 Core RBAC

- *Users*, *Roles*, *Operations*, *Objects*, *Sessions* stand for users, roles, operations, objects and sessions, respectively.
- *Permissions* = $2^{Operations \times Objects}$, the set of permissions.
- $PA \subseteq (Permissions \times Roles)$, a many-to-many mapping permission-to-role assignme -nt relation.
- $UA \subseteq Users \times Roles$, a many-to-many mapping user-to-role assignment relation.
- *User*: $Sessions \rightarrow Users$, the mapping of a session onto the corresponding user.
- *roles*: $Sessions \rightarrow 2^{Roles}$, the mapping of a session onto a set of roles. Formally: $roles(s) = \{r \mid (user(s), r) \in UA\}$.

Definition 2 Role Hierarchies

- $RH = (Roles \times Roles)$, is a partial order on Roles, written as $\geq$.
- In role inheritance, there is roles : *roles*: $Sessions \rightarrow 2^{Roles}$, that is, $roles(s) = \{ r \mid (\exists r' \geq r), (user(s), r') \in UA \}$.

Definition 3 Constraint RBAC

- $\forall (rs,n) \in SSD, \ \forall t \subseteq rs: |t| \geq n \Rightarrow \bigcap_{r \in t} assigned\_user(r) = \phi$, refers to in static separation of duty constraint, there is no user can simultaneously be assigned to the collection of role pairs in SSD.
- $\forall (rs,n) \in SSD, \ \forall t \subseteq rs: |t| \geq n \Rightarrow \bigcap_{r \in t} authorized\_user(r) = \phi$, it implies the role inheritance of SSD in role hierarchy can be defined as: if an SSD relation is specified for a role *r*, then the user assigned to *r* is granted the role of the SSD as an authorized user of constraint.
- $\forall rs \in 2^{ROLES}, n \in N, (rs,n) \in DSD \Rightarrow n \geq 2 \wedge |rs| \geq n$, refers to in dynamic separation of duty constraint, there is no user can simultaneously be assigned to the collection of role pairs in DSD, and there is: $\forall s \in SESSIONS, \forall rs \in 2^{ROLES}, n \in N, (rs,n) \in DSD, t \in rs, t \subseteq session\_roles(s) \Rightarrow |t| < n$.

2.2. Secure Interoperation

In a multi-domain system, security interoperation must takes into account two principles of autonomy and security [13]. In collaboration, the principle of autonomy states that if an access is permitted by an individual system, it must be also permitted under secure interoperation. Likewise, the principle of security states that if an access is denied by an individual system, it must be also denied during secure interoperation. In a RBAC based collaborative system, violations of security policy may be caused by adding role inheritance relations from multi-domain joint. As stated by Barka and Sandhu [14], these constraint violations can be detected by checking relevant security policies and discoveries previously, so that the security risk can be avoided in advance. Such secure interoperable properties include modal conflict, cyclic inheritance, privilege escalation and separation of duty (SoD) [15].

Now, before giving their definition, let us first make the following statement. To differentiate roles, users and permissions among domains, we use the formal format as below, where D_i denotes a domain name. If a role r_k belongs to a domain D_i, and then written as r_{ik}. Likewise, u_{it} represents a user u_t in domain D_i, and p_{iw} a permission p_w in domain D_i.

Definition 4 Modal Conflict

Modal conflict refers to the inconsistent description on policy, which occurs when two or more policies with the opposite signs action on same subject, object and measure. Depending on the type of policy violation, modal conflict can be divided into three types: authorization policy conflict, coercive policy conflict and forced authorization policy conflict [16]. Modal conflict subsequent to discuss in MIRBAC model under interoperation refers to that there is a dominance relation between two inter-domain roles, where a restrict access relation correspondence exists.

Definition 5 Cyclic Inheritance

There are two roles with inheritance relations in an autonomous domain, if the junior role can dominate the senior role through inter-domain role mapping, then we call cyclic inheritance conflict occurs.

We describe the conflict using an example. In a multi-domain RBAC system, the cyclic inheritance refers to the problem that a user u_{it} assigned to role r_{ik} in domain D_i is authorized for the permissions of another local role r_{ij} such as $r_{ij} \mapsto r_{ik}$ (see Subset. 3.1 for definition of $\mapsto$), even though u_{it} is not directly assigned to r_{ij} in the role hierarchy of domain D_i as shown in Figure 1(a).

Definition 6 Privilege Escalation

There are two roles with no inheritance relations in an autonomous domain, however, if there are dominance relations between them during interoperability, then we call privilege escalation conflict occurs.

Specifically, privilege escalation refers to the problem that a user u_{it} assigned to role r_{ij} in domain D_i is authorized for the permissions of another local role r_{ik} such as $r_{ij} \triangleright r_{ik}$ (see Subset. 3.1 for the definition of $\triangleright$), even though u_{it} is not directly assigned to role r_{ik} in the role hierarchy of domain D_i (Figure 1(b)).

Definition 7 Separation of Duty Property

If the user in a session can get or activate two mutually exclusive roles with SoD constraint, then it violates the separation of duty properties, as shown in Figure 1(c). Detection of the SoD conflict is based on two properties:

① Roles r_k and r_m are mutually exclusive if neither one inherit the other directly or indirectly.

② If roles r_k and r_m are mutually exclusive, then there is no other role inherits both of them.

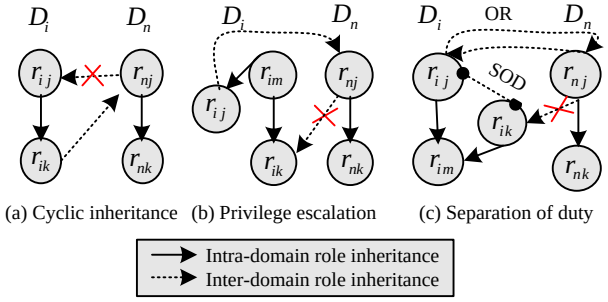

Figure 1. Secure Interoperable Properties

3. The MIRBAC Model

Based on the ANSI INCITS 359-2004, we define our proposed access control model. The MIRBAC model is proposed for multi-domain environments under secure interoperation, which effectively increases the security and flexibility for cross-domain authorization by introducing three types of inter-domain role relationships such as transitive mapping, non-transitive mapping and restricted access. In addition, we provide the model analysis through comparison (see Subset. 3.2).

3.1. Model Definition

3.1.1. Basic Concepts

To support secure interoperation in multi-domain environments, MIRBAC model establishes across-domain authorization policy mainly through three inter-domain role associations such as transitive mapping, non-transitive mapping and restricted access, whose definitions as below.

Definition 8 Transitive Mapping

Transitive mapping refers to the one-way mapping relationship between local roles and foreign roles, which explicitly maps foreign roles to local roles. To this effort, it can obtain the privileges of local roles and their junior roles. Meanwhile, the seniors of foreign roles will correspondingly inherit all of the privileges, written as $\mapsto_T$.

Currently, almost all of the RBAC based security interoperation support transitive mapping relation. As shown in Figure 2, the role r_{2c} in domain D_2 is transitive mapped to the role r_{1c} in domain D_1, written as $r_{2c} \mapsto_T r_{1c}$, then r_{2c} has the permissions of r_{1c} and its junior role r_{1e}. At the same time, role r_{2a} that is the senior role of r_{2c} in domain D_2 can also implicitly acquire the access permissions of r_{2c} and r_{1e}. Visible, transitive mapping relations in inter-domain roles are essentially the same as inheritance relations in intra-domain roles, but the former is capable of greatly reducing the workload of inter-domain role mapping management.

Definition 9 Non-transitive Mapping

Non-transitive mapping refers to the one-way mapping relationship between local roles and foreign roles, which explicitly maps foreign roles to local roles and gets the permissions of local roles and their junior roles. However, unlike transitive mapping, in non-transitive mapping, the seniors of roles in foreign domains cannot implicitly acquire the access permissions of local roles and their juniors, written as $\mapsto_{NT}$.

Non-transitive mapping is capable of controlling the authorization scope for foreign roles effectively, by limiting inter-domain relations to specific foreign roles, which increases the flexibility of inter-domain authorization. Meanwhile, the non-transitive mapping also meets special inter-domain authorization requirements. For instance, in

Figure 2, a SSD relation is specified for roles r_{1b} and r_{1c} in domain D_1, and $r_{2c} \overset{\mapsto}{}_T\ r_{1c}$ indicates that role r_{2a} can implicitly gain access permissions of role r_{1c}. If role r_{2b} in domain D_2 is mapped to role r_{1b} in domain D_1 under inter-domain interoperability, and if we adopt transitive mapping, then r_{2a} that is the senior of r_{2b} will implicitly obtain the access permissions of r_{1b}, which leads to the violation of SSD relations between roles r_{1b} and r_{1c} in domain D_1. Therefore, here we must put a non-transitive mapping relation between role r_{2b} and role r_{1b}, written as $r_{2b} \overset{\mapsto}{}_{NT}\ r_{1b}$.

Definition 10 Restricted Access

Restricted access prevents foreign roles from obtaining local roles' access permissions through the direct inter-domain mapping, or indirectly through other transitive mapping, explicitly or implicitly, written as $\overset{\mapsto}{}_R$.

To meet the security needs, restricted access denies access relations for some specific inter-domain roles during interoperation. For instance, in Figure 2, there is $r_{2a} \overset{\mapsto}{}_T\ r_{1c}$, then role r_{2a} can implicitly gain access permissions of role r_{1c}. Since a SSD relation is specified for roles r_{1b} and r_{1c} in domain D_1, D_1 can explicitly deny r_{2a} to acquire the access permissions of local role r_{2a} by introducing the restricted access relations as $r_{2a} \overset{\mapsto}{}_R r_{1b}$, where the function is equivalent to negative authorization between domains.

Inter-domain transitive mapping and non-transitive mapping role relationships build an interoperable bridge for the independent autonomous RBAC systems, and establish partial orders on the set of roles separated originally, which extend inter-domain role inheritance to multi-domain environments. In this article, we call both role inheritance relations and partial orders as role dominance relations.

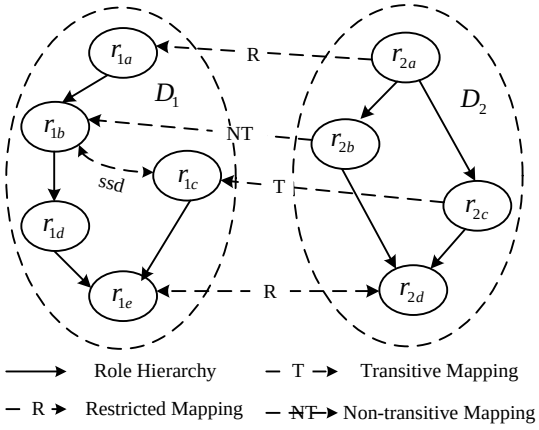

Figure 2. Inter-domain Role Relations in MIRBAC Model

Definition 11 Role Dominance

In a multi-domain environment, if a role r_1 explicitly or implicitly gains the access permissions of another role r_2 through inter-domain role hierarchies and cross-domain role mappings, then we call role r_1 dominant role r_2, written as $r_1 \triangleright r_2$.

Clearly, the role dominance expands role inheritance in a multi-domain environment. It covers the inheritance relationships between the roles from each independent RBAC system, and also contains the partial orders between roles obtained through the cross-domain mappings. All the role dominance relations in Figure 2 are as follows.

Intra-domain role dominance: $r_{1a} \triangleright r_{1b}$, $r_{1a} \triangleright r_{1d}$, $r_{1a} \triangleright r_{1e}$, $r_{1b} \triangleright r_{1d}$, $r_{1b} \triangleright r_{1e}$, $r_{1d} \triangleright r_{1e}$, $r_{1c} \triangleright r_{1e}$, $r_{2a} \triangleright r_{2b}$, $r_{2a} \triangleright r_{2c}$, $r_{2a} \triangleright r_{2d}$, $r_{2b} \triangleright r_{2d}$, $r_{2c} \triangleright r_{2d}$.

Inter-domain role dominance: $r_{2a} \triangleright r_{1c}$, $r_{2a} \triangleright r_{1e}$, $r_{2b} \triangleright r_{1b}$, $r_{2b} \triangleright r_{1d}$, $r_{2b} \triangleright r_{1e}$, $r_{2c} \triangleright r_{1c}$, $r_{2c} \triangleright r_{1e}$.

3.1.2. Basic Elements

For intra-domain, we give the formal definitions of MIRBAC model as below:

- $USERS_i$, $ROLES_i$, $PRMS_i$, stands for users, roles and permissions in domain D_i, respectively.
- $RH_i \subseteq ROLES_i \times ROLES_i$, is a set of role inheritance relations in domain D_i. Only if all permissions of r2 are also permissions of r1, and all users of r1 are also users of r2, written as $r_1 \succ r_2$, meanwhile, define direct role inheritance relations as $\succ\!\!\succ$.
- $UA_i \subseteq USERS_i \times ROLES_i$, a many-to-many mapping user-to-role assignment relation in domain D_i.
- $PA_i \subseteq ROLES_i \times PRMS_i$, a many-to-many mapping permission-to-role assignment relation in domain D_i.
- $SSD_i \subseteq (2^{ROLESi} \times M)$, is a collection of pairs (rs, m) in SSD, where each rs is a role set in the domain D_i, and m is a nature number ≥2, with the property that any user can only get a maximum of m -1 roles from rs.
- $DSD_i \subseteq (2^{ROLESi} \times M)$, is a collection of pairs (rs, m) in DSD, where each rs is a role set in domain D_i, and m is a nature number ≥2, with the property that no user can active more than m roles in the same session.

For inter-domain, in a multi-domain environment that consists of n autonomous domains, we extend the aforementioned definitions as below:

- $USERS = \bigcup_{i=1}^{n} USERS_i$, $ROLES = \bigcup_{i=1}^{n} ROLES_i$, $PRMS = \bigcup_{i=1}^{n} PRMS_i$, stands for all user set, all role set and all permission set in the multi-domain environment, respectively.
- $SSD = \bigcup_{i=1}^{n} SSD_i$, $DSD = \bigcup_{i=1}^{n} DSD_i$, stands for the all static separation of duty (SSD) constrain set and all dynamic separation of duty (DSD) constrain set in the multi-domain environment, respectively.
- $ISSESSIONS = \bigcup_{i=1}^{n} \bigcup_{j=1, j\neq i}^{n} ISESSIONS_{i,j}$, stands for the set of all interoperability sessions under multi-domain environment.
- $UA = \bigcup_{i=1}^{n} UA_i$, $PA = \bigcup_{i=1}^{n} PA_i$, $RH = \bigcup_{i=1}^{n} RH_i$, represents the collection of the user to role assignment relation, role-to-permission assignment relation and inheritance relation in multi-domain environments, respectively.
- $TM = \bigcup_{i=1}^{n} \bigcup_{j=1, j\neq i}^{n} TM_{i,j}$, $NM = \bigcup_{i=1}^{n} \bigcup_{j=1, j\neq i}^{n} NM_{i,j}$ and $RM = \bigcup_{i=1}^{n} \bigcup_{j=1, j\neq i}^{n} RM_{i,j}$, represent the set of transitive mapping relation, non-transitive mapping relation and restricted access relation for the cross-domain roles in a multi-domain environment, respectively.
- $RD = ROLES \times ROLES$, stands for all set of role dominance relation in multi-domain environments. Formal definition: $RD = \{(r_i, r_j) | r_i \triangleright r_j\}$.
- *global_junior_roles*: $ROLES \rightarrow 2^{ROLES}$, the mapping of a role onto a set of the dominated roles under interoperability in multi-domain environments, which include both of local role and foreign role. Formal definition: $global\_junior\_roles(r) = \{r' \in ROLES \mid (r, r' \in RD)\}$.
- *global_authorized_users*: $ROLES \rightarrow 2^{USERS}$, stands for the user collections of both intra-domain users and inter-domain users under interoperability in a multi-domain environment. Formal definition: $global\_authorized\_users(r) = \{u \in USERS | (\exists r' \triangleright r)\ ((u, r') \in UA)\}$.
- *global_authorized_roles*: $USERS \rightarrow 2^{ROLES}$, stands for the set of users' global authorization role in an interoperable multi-domain environment, which consists of the three parts: intra-domain role that directly assigned to users, intra-domain role

obtained form the role inheritance and extra-domain role from the inter-domain mapping. Formal definition: *global_authorized_roles*(u) = {$r \in ROLES$ | ($\exists r' \triangleright r$) (($u, r'$) $\in UA$)}.

- *global_authorized_permissions*:$ROLES \rightarrow 2^{PRMS}$, the mapping of roles onto a set of all permissions in an interoperable multi-domain environment. Formal definition: *global_authorized_permissions*(r) = {$p \in PRMS | (\exists r' \triangleright r)$ ($(p, r') \in PA$)}.
- *global_user*(*isession*: *ISESSIONS*)$\rightarrow$*USERS*, a mapping of interoperable sessions onto a corresponding user, where is an one-to-many relation between the users and interoperable sessions.
- *user_issessions*(*user*:*USER*)$\rightarrow 2^{ISESSIONS}$, a mapping of a user onto the collection of interoperable sessions.
- *issessions_roles*(*isessioins*: *ISESSIONS*) $\rightarrow 2^{ROLES}$, the mapping of interoperable sessions onto the active roles that include the authorized roles of local domain and the foreign roles through mapping relations.

3.2. Model Comparison

Combined with existing research results such as IRBAC 2000 [5], SEART [7], B.Shafig 05 [8], RBIAC [9], L.Deng 09 [10] and CX.Ye 12 [11], we analyze and compare MIRBAC model regarding the following aspects, as shown in Table 1.

Table 1. The MIRBAC against Relevant Research Results

No.	Comparison criterion	IRBAC 2000	SEART	B. Shafiq 05	RBIAC	L. Deng 09	CX. Ye 12	MIRBAC
1.	Support core RBAC	Y	Y	Y	Y	Y	Y	Y
2.	Support hierarchical RBAC	Y	Y	Y	Y	Y	Y	Y
3.	Support static separation of duties	N	N	Y	Y	Y	Y	Y
4.	Support dynamic separation of duties	N	N	N	Y	N	Y	Y
5.	Support inter-domain transitive mapping	Y	Y	Y	Y	N	Y	Y
6.	Support inter-domain non-transitive mapping	Y	N	N	N	N	N/A	Y
7.	Support role restricted access	N	Y	N	N	N	N/A	Y
Y = Yes, N = No and N/A = Not applicable								

By comparison, MIRBAC model is capable of providing extensive support for the standard RBAC components, and a flexible inter-domain authorization mode. Specifically, the main advantages are mainly shown as three aspects:

① A complete extension of standard RBAC model: MIRBAC model makes a complete extension for a multi-domain environment based on ANSI RBAC model, and it supports four components used in autonomous domains such as core RBAC, hierarchical RBAC, static separation of duty and dynamic separation of duty under a multi-domain interoperable environment, which fully guarantee the integrity of inter-domain security policy in multi-domain interoperability.

② Increase the flexibility of inter-domain authorization management: Compared with the current studies, MIRBAC model increases the flexibility of inter-domain authorization management, by introducing two types of inter-domain role relations as transitive mapping and non-transitive mapping. Specifically, it mainly reflects in two aspects: First, since inter-domain transitive mapping has the same function as intra-domain role

inheritance, it greatly reduces the complexity of authorization management through the transitive inheritance of permissions, which helps reduce the burden of inter-domain authorization management. Second, the non-transitive mapping is capable of meeting the special inter-domain authorization needs such as SoD constraints, by limiting a foreign role's senior role to mapping onto the local role.

③ Improve the security of inter-domain authorization management: MIRBAC model promotes safety in two ways: firstly, the inter-domain restricted access relation can explicitly limit foreign roles accessing local roles, which implement the protection of autonomous domains for their intra-domain sensitive permissions. Moreover, it also increases the feedback function of authorized management, that is, when the user of foreign roles performs malicious acts, the autonomous domains are capable of providing active secure protection to restricting their access to inter-domain roles. Secondly, non-transitive mapping can not only limit the seniors of foreign roles to accessing the local roles mapped by its juniors, but also avoid role mapping shuttling among multiple domains and control the spread of the scope of permissions.

4. Security Violation Detection Based on MIRBAC Model

We describe formal definitions for security conflicts such as modality conflict, cyclic inheritance conflict, privilege escalation conflict and violation of SSD and DSD. We also propose a complete detection algorithm for MIRBAC model according to the feature of secure interoperation in multi-domain environments, and further analyze computational complexity of our proposed algorithm.

4.1. Conflict Description

In this section, combined with Subsection 2.2, we redefine the following five conflict types of MIRBAC model under interoperability in multi-domain systems.

Definition 12 Modal Conflict

To detect a modality conflict for roles r_1 and r_2, we check if the proposition $(r_1 \triangleright r_2) \wedge i \neq j \wedge (r_1, r_2) \in RM$ is true, formally:

$$r_1 \in ROLES_i, r_2 \in ROLES_j, (r_1 \triangleright r_2) \wedge i \neq j \wedge (r_1, r_2) \in RM.$$

Definition 13 Cyclic Inheritance Conflict

To detect a cyclic inheritance conflict for roles r_1 and r_2, we check if the proposition $(r_1 \succeq r_2) \wedge (r_2 \triangleright r_1)$ is true, formally:

$$\exists (r_1, r_2 \in ROLES_i), (r_1 \succ r_2) \wedge (r_2 \triangleright r_1) = 1.$$

Definition 14 Privilege Escalation Conflict

To detect privileges escalation for a role r_1 against a role r_2, we check if the proposition $\neg(r_2 \succeq r_1) \wedge \neg(r_1 \succeq r_2) \wedge (r_2 \triangleright r_1) \vee (r_1 \triangleright r_2)$ is true, formally:

$$\exists (r_1 , r_2 \in ROLES_i), \neg(r_2 \succeq r_1) \wedge \neg(r_1 \succeq r_2) \wedge (r_2 \triangleright r_1) \vee (r_1 \triangleright r_2) = 1.$$

Definition 15 Violation of SSD Conflict

Assume that *ssd* is an element of SSD constraint, where *ssd* = <*rs*, *m*>. If there exist a user's authorization role set to intersect with the role set *rs* of *ssd*, and the number of elements in the intersection is not less than *m*, then we identify the violations of SSD conflict.

Definition 16 Violation of DSD Conflict

Assume that *dsd* is an element of DSD constraint, where *dsd* = <*rs*, *m*>. If there exist a session's active role set to intersect with the role set *rs* of *dsd*, and the number of elements in the intersection is not less than *m*, then we identify the violations of DSD conflict.

4.2. A Complete Conflict Detection Algorithm

MIRBAC model integrates multiple RBAC systems into a whole system using inter-domain role mappings, which are capable of interacting with each domain, and keeping autonomy and security for a single RBAC system. In a multi-domain environment, we divide the relationships between any two roles into four categories: direct role inheritance, inter-domain transitive mapping, inter-domain non-transitive mapping and non-direct partial order relations, according to the direct partial orders between two roles. Next, we use four integers 3, 2, 1, 0 to represent the abovementioned four relations, respectively. In particular, there is a direct role inheritance relation between a role and itself. To this effort, by mapping onto the set of {3, 2, 1, 0}, ROLES×ROLES, the direct partial order relationship of any two roles in a multi-domain environment, is able to form a direct role partial order relationship matrix RPR.

We assume that, if a role dominant another role is represented by 1, otherwise with 0. Thus the role dominance relationship matrix RDR can be constructed. Similarly, we can define direct role inheritance relationship matrix RIR and restricted access relationship matrix RAR. RIR+, which is the transitive closure of RIR, is the complete role inheritance relationship matrix of system.

Similarly, we use the following rules to construct direct user-to-role assignment matrix UAS, user authorization role matrix UAH and session active role relationship matrix SAR, where the rules are: If a role is assigned directly to the user, then the corresponding element in matrix UAS is 1, otherwise is 0; If a user owns access permissions to the role, then the corresponding element in matrix UAH is 1, otherwise is 0; If a role is activated in the session, then the corresponding element in matrix SAR is 1, otherwise is 0.

Algorithm: Complete detection algorithm

Input: $USER_i$, $ROLES_i$, UA_i, RH_i, SSD_i and DSD_i in RBAC domains; inter-domain role relations set: *TM*, *RM*, *NM*; interoperation sessions set: *ISESSIONS*.

Output: TRUE or FALSE.

Step 1. Construct relationship matrices *RPR*, *RIR*, *RAR* and *UAS*, respectively;

Step 2. Compute the transitive closure of *RIR* with Warshall algorithm, and generate a complete role hierarchy relationship matrix RIR^+;

Step 3. Compute role dominance relationship matrix *RDR*, follows:

$$\forall r_i, r_j, r_k \in ROLES,\ RPR(r_i, r_k)=3 \wedge RPR(r_k, r_j)=3 \Rightarrow RPR(r_i, r_j) = 3;$$

$$\forall r_i, r_j \in ROLES,\ RPR(r_i, r_k)>0 \Rightarrow RDR(r_i, r_j) = 1;$$

$$\forall r_i, r_j, r_k \in ROLES,\ RPR(r_i, r_k)>0 \wedge RPR(r_k, r_j)>1 \Rightarrow RDR(r_i, r_j) = 1.$$

Step 4. $\exists\ r, r' \in ROLES$, if there is $RDR(r, r')=1 \wedge RMR(r, r')=1$, then the modal conflict occurs;

Step 5. $\exists\ r, r' \in ROLES$, if there is $RDR(r, r') = 1 \wedge RHR^+(r, r') = 1$, then the cyclic inheritance conflict occurs;

Step 6. $\exists\ r, r' \in ROLES$, if there is $RDR(r, r')=1 \wedge RHR^+(r, r')=0 \wedge RHR^+(r, r')=0$, then the privilege escalation conflict occurs;

Step 7. Compute user authorization role matrix *UAH*, according to *UAS* and *RDR*, wherein

$$\forall u \in USERS, \forall r \in ROLES, UAH(u,r) = \begin{cases} 1 & \exists r' \in ROLES, RDR(r',r) = 1 \wedge \\ & UAS(u,r')\ or\ UAS(u,r)=1 \\ 0 & \text{others} \end{cases};$$

Step 8. $\exists\ ssd=(rs,m) \in SSD, \exists u \in USERS, \exists r' \in ROLES$, if $\sum_{r \in rs} RDR(r,r') \geq m \vee \sum_{r \in rs} UAH(u,r) \geq m$ is true, then the violation of SSD conflict occurs;

Step 9. Construct interoperation session active role relationship matrix *SAR*, wherein

$$\forall is \in ISSESSIONS, \forall r \in ROLES, SAR(is,r) = \begin{cases} 1 & r \in isession\_roles(is) \\ 0 & \text{others} \end{cases};$$

Step10. $\exists dsd=(rs,m)\in DSD$, $\exists is\in ISESSIONS$, if $\sum_{r\in rs} SAR(r,r') \geq m$ is true, then the violation of DSD conflict occurs;

Step11. If cannot detect any conflict, then returns FALSE, otherwise returns TRUE.

Since the violation of DSD constraints occurs when creating interoperable sessions or activating roles, and can be automatically detected by every autonomous domain, without needing to understand other domains' security policies. According to the model support functions defined in Appendix, the violation of DSD conflict can be avoided by effectiveness validation mechanisms of functions as *CreateInteroperationSession*() and *AddInteroperationActiveRole*(). Hence, the Algorithm can be simplified the steps of S9 and S10.

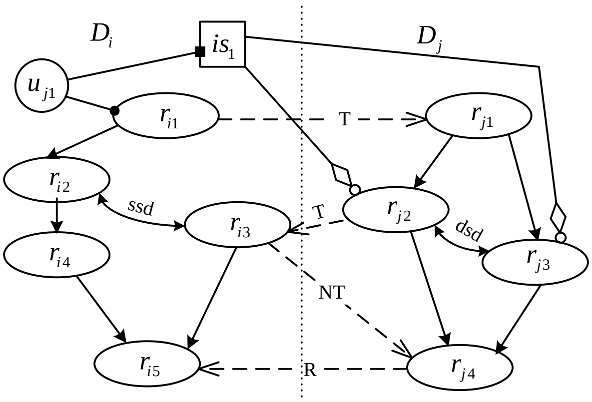

Figure 3. A Multi-domain Security Violation during Interoperability

RPR =

	r_{i1}	r_{i2}	r_{i3}	r_{i4}	r_{i5}	r_{j1}	r_{j2}	r_{j3}	r_{j4}
r_{i1}	3	3	0	0	0	2	0	0	0
r_{i2}	0	3	0	3	0	0	0	0	0
r_{i3}	0	0	3	0	3	0	0	0	1
r_{i4}	0	0	0	3	3	0	0	0	0
r_{i5}	0	0	0	0	3	0	0	0	0
r_{j1}	0	0	0	0	0	3	3	3	0
r_{j2}	0	0	2	0	0	0	3	0	3
r_{j3}	0	0	0	0	0	0	0	3	3
r_{j4}	0	0	0	0	0	0	0	0	3

RAR =

	r_{i1}	r_{i2}	r_{i3}	r_{i4}	r_{i5}	r_{j1}	r_{j2}	r_{j3}	r_{j4}
r_{i1}	0	0	0	0	0	0	0	0	0
r_{i2}	0	0	0	0	0	0	0	0	0
r_{i3}	0	0	0	0	0	0	0	0	0
r_{i4}	0	0	0	0	0	0	0	0	0
r_{i5}	0	0	0	0	0	0	0	0	0
r_{j1}	0	0	0	0	0	0	0	0	0
r_{j2}	0	0	0	0	0	0	0	0	0
r_{j3}	0	0	0	0	0	0	0	0	0
r_{j4}	0	0	0	0	1	0	0	0	0

RIR =

	r_{i1}	r_{i2}	r_{i3}	r_{i4}	r_{i5}	r_{j1}	r_{j2}	r_{j3}	r_{j4}
r_{i1}	1	1	0	0	0	0	0	0	0
r_{i2}	0	1	0	1	0	0	0	0	0
r_{i3}	0	0	1	0	1	0	0	0	0
r_{i4}	0	0	0	1	1	0	0	0	0
r_{i5}	0	0	0	0	1	0	0	0	0
r_{j1}	0	0	0	0	0	1	1	1	0
r_{j2}	0	0	0	0	0	0	1	0	1
r_{j3}	0	0	0	0	0	0	0	1	1
r_{j4}	0	0	0	0	0	0	0	0	1

RIR^+ =

	r_{i1}	r_{i2}	r_{i3}	r_{i4}	r_{i5}	r_{j1}	r_{j2}	r_{j3}	r_{j4}
r_{i1}	1	1	0	1	1	0	0	0	0
r_{i2}	0	1	0	1	1	0	0	0	0
r_{i3}	0	0	1	0	1	0	0	0	0
r_{i4}	0	0	0	1	1	0	0	0	0
r_{i5}	0	0	0	0	1	0	0	0	0
r_{j1}	0	0	0	0	0	1	1	1	1
r_{j2}	0	0	0	0	0	0	1	0	1
r_{j3}	0	0	0	0	0	0	0	1	1
r_{j4}	0	0	0	0	0	0	0	0	1

Figure 4. Relation Matrices: RPR, RAR and RIR

We further elaborate on the creation and computation of the Algorithm through an example. Let's assume the multi-domain access control policy depicted in Figure 3 that

allows collaboration between domain D_i and domain D_j. Domain D_j has the following elements: session is_1, roles r_{i1}, r_{i2}, r_{i3}, r_{i4} and r_{i5}. A SSD relation is specified for r_{i2} and r_{i3}, that is $ssd = <\{r_{i2},r_{i3}\}, 2>$. Domain D_j has the following elements: user uj_1, roles r_{j1}, r_{j2}, r_{j3} and r_{j4}. A DSD relation is specified for r_{j2} and r_{j3}, that is $dsd = <\{r_{j2},r_{j3}\}, 2>$. For inter-domain, there are relations as $r_{i1} \mapsto_T r_{j1}$, $r_{j2} \mapsto_T r_{i3}$, $r_{i3} \mapsto_{NT} r_{j4}$ and $r_{j4} \mapsto_R r_{i5}$.

The relationship matrices RPR, RAR and RIR are listed in Figure 4, which can be constructed easily by following the step S1 in Algorithm. Next, we compute the transitive closure of *RIR* with the Warshall algorithm, and generate a complete role inheritance relationship matrix RIR^+ (Figure 4). According to the step S3 in Algorithm, the role dominance relationship matrix *RDR* generates (see Figure 5).

After computing with steps S4 and S5 in Algorithm, we conclude there are no modal conflict and cyclic inheritance conflict. Next, in step S6, there is $RIR^+(r_{i1}, r_{i3}) = 0$ and $RDR(r_{i1}, r_{i3}) = 1$, where occurs privilege escalation conflict. Continue steps S7 and S8, there is $RDR(r_{i1}, r_{i2}) = RDR(r_{i1}, r_{i3}) = 1$, where occurs the violation of SSD constraints. Moreover, as shown in Figure 3, the set of active roles in session is_1 is (r_{j2}, r_{j3}), where occurs the violation of DSD ($dsd = <\{r_{j2}, r_{j3}\}, 2>$) constraints through the detection of steps S9 and S10.

$$RDR = \begin{array}{c} \\ r_{i1} \\ r_{i2} \\ r_{i3} \\ r_{i4} \\ r_{i5} \\ r_{j1} \\ r_{j2} \\ r_{j3} \\ r_{j4} \end{array} \begin{array}{c} \begin{array}{ccccccccc} r_{i1} & r_{i2} & r_{i3} & r_{i4} & r_{i5} & r_{j1} & r_{j2} & r_{j3} & r_{j4} \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Figure 5. Relation Matrix: RDR

4.3. Computational Complexity Analysis

In a distributed environment, inter-domain role mapping is usually managed distributed by each individual domain. While, conflict detection requires to implement under a global policy view from a trusted third party. Following, we analyze the computational complexity for our proposed conflict detection algorithm.

$$\begin{aligned} T_{total} &= \sum_{q=1}^{11} T^{(q)} = O(|ROLES|^2 + |USERS| \times |ROLES|) \\ &+ O(|ROLES|^2) + O(|ROLES|^3) + O(|ROLES|^2) \\ &+ O(|ROLES|^2) + O(|USERS| \times |ROLES|^2) \\ &+ O(|SSD| \times |ROLES|^2 + |SSD| \times |USERS| \times |ROLES|) \\ &+ O(|ISESSIONS| \times |ROLES|) + O(|ROLES|^2) \\ &+ O(|DSD| \times |ISESSIONS| \times |ROLES|) \end{aligned} \quad (1)$$

Let q denotes a step in the algorithm, and $T^{(q)}$ represents the time complexity of computing where the algorithm in the corresponding step. To construct various relevant roles matrices and compare corresponding elements of two matrices, traverse is the only needed operation, where the time complexity is $O(|ROLES|^2)$. Compute the relation matrix of transitive closure with Warshall algorithm, and the time complexity is $O(|ROLES|^3)$. Moreover, the time complexity of calculating user's authorization role matrix *UAH* is $O(|USERS| \times |ROLES|^2)$, judging the violation of SSD constraint is $O(|SSD| \times |ROLES|^2 + |SSD| \times |USERS| \times |ROLES|)$ and judging the violation of DSD constraints is $O(|DSD| \times$

|*I*SESSIONS|×|ROLES|). Thus, the total time complexity of our proposed algorithm is as Equation 1.

Under normal circumstances, the number of roles in every autonomous RBAC system is much smaller than user numbers. Therefore, the time complexity of Algorithm can be simplified as Equation 2.

$$\begin{aligned} T_{total}' \approx & O(|USERS|\times|ROLES|^2) \\ & + O(|SSD|\times|USERS|\times|ROLES|) \\ & + O(|DSD|\times|ISESSIONS|\times|ROLES|) \end{aligned} \tag{2}$$

If we use security mechanism of autonomous domains, to judge and avoid the violation of DSD conflicts, our proposed algorithm can further reduce steps S9 and S10. To this effort, the time complexity of Algorithm can be further simplified as Equation 3.

$$T_{total}'' \approx O(|USERS|\times|ROLES|^2) + O(|SSD|\times|USERS|\times|ROLES|) \tag{3}$$

From the above calculation and analysis, the computational complexity of our complete detection algorithm shows as a cubic polynomial.

5. Simulation and Experimental Study

We developed a prototype (Role-base Secure Interoperation Prototype, RSIP) system for experiment. Our test platform of the RSIP system is running on the Windows Server 2003 Professional SP2 running on 2.6GHz Intel Pentium 4 processor, with 512MB of RAM. Development platform is Visual Studio 2005, where the background database using SQL Server 2005. The overall design of the RSIP system shown in Figure 6, the figure shows a framework of secure interoperation under multi-domain environments consisting of three autonomous domains. The core components in RSIP system are the overall Secure Inter-operation Management Center (SIMC) and the Secure Inter-operation Management Agent (SIMA) for each autonomous domain. Wherein, SIMA component mainly responsible for the completion of autonomous RBAC policy management, inter-domain authorization management and interoperability session management functions, and provides an autonomous policy administrator with a graphical management interface; SIMC component is mainly to complete conflict detection and build access paths for inter-domain roles, which provide security for the implementation of interoperability in multi-domain environments.

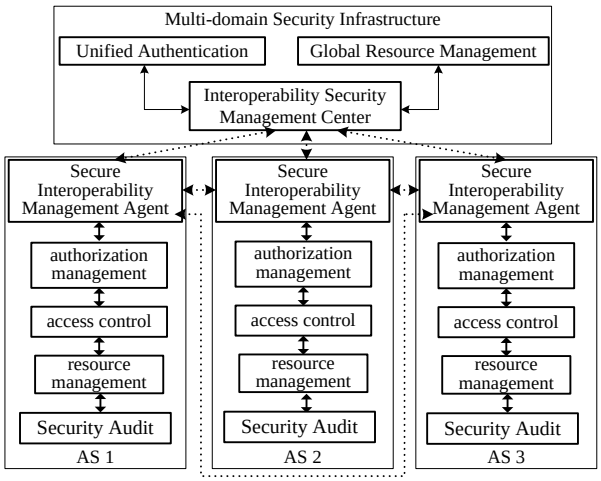

Figure 6. The Overall Design of RSIP System

Next step, we check the applicability and efficiency of the MIRBAC model during multi-domain environments. First, we generated access control requests from multi-domain users using the NetworkX python package. Wherein, NetworkX is a modeling tool of graph theory and complex network developed by Python language, and provides functions to dynamically generate user requests and correspondence privileges. Secondly, we simulated the role assignments between domains using RBAC simulator, and then used function to detect role assignment relationships. We performed a series of simulations on different data sets, more specifically, experiment simulated 4 m×n-scale multi-domain interoperable scene, where m represents the number of hosted domains, n represents the number of characters contained in each hosted domain. Test specific parameters set: 50 × 100, 200 × 100, 5 × 1000, 20 × 1000, and the division of 50 × 100, 200 × 100 called as group A of data, and of 5 × 1000, 20 × 1000 called as group B of data. Single hosted domain system, for example, by randomly generating the required number of roles assigned their role relationship, such as SSD and DSD, to evaluate the memory consumption and time overhead to detect conflicts of MIRBAC model when it implements the access control decision functions. Performance test as shown in Table 2.

Table 2. Memory Usage and Identified Violations (5000 Random Role Assignments)

Domains × Roles	Role assignments	SSD	DSD	Inter -domain	RIR^+(KB)	RDR (KB)	Role violation	SSD violation	DSD violation
50×100	5368	86	109	265	10,359	39,789	4553	8	12
200×100	19,936	38	35	64	39,927	114,308	4876	1	0
5×1000	5980	252	245	77	10,976	48,659	3944	50	25
20×1000	20,365	128	153	21	40,382	164,875	4591	3	6

The results indicate that two conclusions: first, for systems formed by single hosted domains with the same number of roles, the greater the number of inter-domain role assignment, the smaller the number of SSD and DSD. Second, for the same size system, the fewer the number of single hosted domains, more the role assignment relations between domains, and the memory consumption accordingly is greater. Easy to see, the performance evaluation of "A" system is superior to "B" system, which shows MIRBAC model is more suitable for a collaborative environment with multiple administrative domains.

To further evaluate performance, on one hand, we make detailed comparison of the calculation time model used to express its data structure and access control strategy, as shown in Figure 7. One the other hand, Figure 8 shows the decision time required for our proposed model (MIRBAC) and the standard model (RBAC). Our evaluation consider the worst case: Maximum_Decision_Time = $T(RIR^+)$+ $T(RDR)$ + $T(m\_violation())$+ $T(ci\_violation())$ + $T(pe\_violation())$ + $T(ssd\_violation())$ + $T(dsd\_violation())$, wherein, *m_violation*(), *ci_violation*(), *pe_violation*(), *ssd_violation*() and *dsd_violation*() stand for modal conflict, cyclic inheritance conflict, privilege escalation conflict and SoD conflict, respectively.

Calculated, the maximum decision time regarding the worst case ranges approximately from 248 to 1142 ms, and the average decision time for MIRBAC model to perform access control policy will be far less than the value in this range. Experiments show that, MIRBAC model has good adaptability and performance feasibility in a multi-domain interoperable environment.

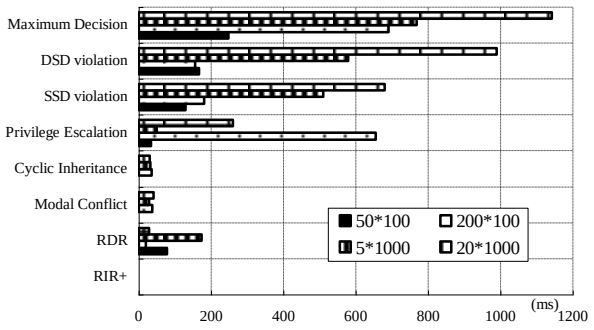

Figure 7. Time of Computations for MIRBAC Model

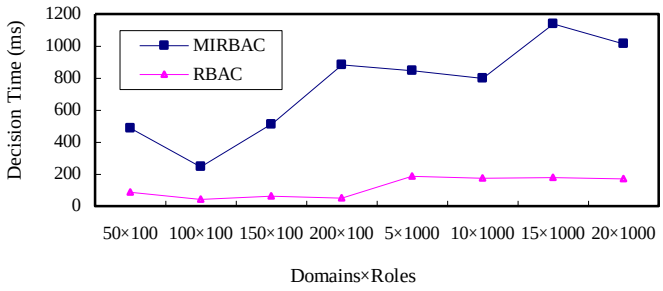

Figure 8. Comparison of MIRBAC and RBAC

6. Conclusions

In this paper, we demonstrated the security violation detection based on MIRBAC model in multi-domain systems. We proposed a role-based access control model for multi-domain interoperability (MIRBAC), which applied an extended redefined ANSI INCITS 359-2004 to express relevant definitions and functions. By comparison, MIRBAC model is capable of supporting inter-domain transitive mapping, inter-domain non-transitive mapping and role restricted access, and effectively enhances the security and flexibility of cross-domain authorization. To the best of our knowledge, existing studies lack the methods and strategies of conflict detection for multi-domain secure interoperation, let alone a complete detection algorithm for all types of security conflicts. Hence, we devised a complete violation detection algorithm based on MIRBAC model to perform security verification. Through experimental studies, we concluded that our proposed access control model has good adaptability and efficiency during collaboration in a multi-domain environment.

Appendix

I. *AddTransitiveMapping*(), this function establishes a transitive relationship between two roles r and r' that come from two different domains. Only if the two roles belong to different domains, and there is no transitive mapping relation between r and r' before mapping, then the function is effective.

II. *DeleteTransitiveMapping*(), this function delete the transitive relationship between two roles r and r' that come from two different domains. Only if the two roles belong to different domains, and the transitive mapping relation has already existed before mapping, the function is effective.

III. *AddNonTranitiveMapping*(), this function establishes a non-transitive relationship between two roles *r* and r' that come from two different domains. Only if the two roles belong to different domains, and there is no non-transitive mapping relation between *r* and r' before mapping, then the function is effective.

IV. *DeleteNonTransitiveMapping*(), this function delete the non-transitive relationship between two roles *r* and r' that come from two different domains. Only if the two roles belong to different domains, and the non-transitive mapping relation has already existed before mapping, the function is effective.

V. *AddRestrictedAccess*(), this function establishes a restricted access relationship between two roles *r* and r' that come from two different domains. Only if the two roles belong to different domains, and there is no restricted access relation between *r* and r' before mapping, then the function is effective.

VI. *DeleteRestirctedAccess*(), this function delete the restricted access relationship between two roles *r* and r' that come from two different domains. Only if the two roles belong to different domains, and the restricted access relation has already existed before mapping, the function is effective.

VII. *CreateInteroperationSession*(), this function creates a new interoperable session *isession* for a user *user*, and creates an active set of roles *ars*. Only if interoperable session *isession* has not been created, and the active role set *ars* meets *DSD* relations, the function is effective.

VIII. *DeleteInteroperationSession*(), this function deletes the interoperable session *isession* initiated by the user *user*, if and only if the *isession* belongs to the user *user*, the function is effective.

IX. *AddInteroperationActiveRole*(), this function adds a role $role$ to an interoperable session *isession* of the user *user*, if and only if *role* is not an active role in current interoperable session *isession*, and the *role* on the original active role set does not violate *DSD* constraint in interoperable session *isession*, the function is effective.

X. *DeleteInteroperationActiveRole*(), this function deletes the active role *role* in interoperable session *isession* of the user *user*.

XI. *CheckInteroperationAccess*(), this function returns a Boolean value, indicating that whether a user *user* is able to perform the specific cross-domain permission *p* in a interoperable session *isession*. Specifically, if and only if permission *p* be isassigned to an active role in the interoperable session *isession*, then the user *user* can perform permissions *p* in session *isession*.

Acknowledgments

This paper is supported by the following foundations or programs, including National Natural Science Foundation of China (61105093), Scientific Research Programs in Higher Education of Chongqing Institute of Higher Education (CQGJ15203B), Quality Improvement Projects in Higher Education of Chongqing Education Science "Twelfth Five Year Plan" (2015-GX-086), Science and Technology Research Project of Chongqing Education Committee of China (KJ1602203).

References

[1] L. Gong and X. Qian, “Computational issues in secure interoperation”, Software Engineering, IEEE Transactions on, vol. 22, no. 1, **(1996)**, pp. 43-52.
[2] S. Dawson, S. Qian and P. Samarati, “Providing security and interoperation of heterogeneous systems”, Distributed and Parallel Databases, vol. 8, no. 1, **(2000)**, pp. 119-145.
[3] D. Ferraiolo, D. R. Kuhn and R. Chandramouli, “Role-based access control”, Artech House, **(2003)**.
[4] A. Kapadia, “Secure Interoperability Using Dynamic Role Translation”, Chicago: University of Illinois, **(1999)**.

[5] A. Kapadia, J. Al-Muhtadi and R. H. Campbell, "IRBAC 2000: Secure Interoperability Using Dynamic Role Translation", University of Illinois, Chicago, https://www.cs.indiana.edu/~kapadia/papers/irbac2000.pdf (Accessed 18 August 2015), **(2000)**.
[6] A. Kapadia, J. Al-Muhtadi and R. H. Campbell, "The A-IRBAC 2000 model: Administrative interoperable role-based access control", University of Illinois, Chicago, **(2000)**.
[7] M. Shehab, E. Bertino and A. Ghafoor, "SERAT: Secure role mapping technique for decentralized secure interoperability", Proceedings of the tenth ACM symposium on Access control models and technologies, ACM, **(2005)**, pp. 159-167.
[8] B. Shafiq, J. B. D. Joshi and E. Bertino, "Secure interoperation in a multi-domain environment employing RBAC policies", Knowledge and Data Engineering, IEEE Transactions on, vol. 17, no. 11, **(2005)**, pp. 1557-1577.
[9] D. Wu, "Research on role based interoperation access control in distributed environment", Unpublished PhD thesis, Zhejiang University, Zhejiang, China, **(2006)**.
[10] L. L. Den, Z. Y. Xu and Y. P. He, "Trust-based constraint-secure interoperation for dynamic mediato-free collaboration", Journal of Computer, vol. 4, no. 9, **(2009)**, pp. 862-872.
[11] C. X. Ye and D. H. Guo, "Research on secure interoperation in multi-domain environment", Journal of Computer Applications, vol. 32, no. 12, **(2012)**, pp. 3422-3425.
[12] R. Sandhu, D. Ferraiolo and R. Kuhn, "The NIST model for role-based access control: towards a unified standard", ACM workshop on Role-based access control, **(2000)**.
[13] Y. D. Xu, Q. G. Shen and J. Zhou, "PonderTalk network policy modality conflict detection system", Journal of Military Communications Technology, vol. 33, no. 4, **(2012)**, pp. 42-47.
[14] E. Barka and R. Sandhu, "A role-based delegation model and some extensions", 23rd National Information Systems Security Conference, October, **(2000)**, pp. 396-404.
[15] A. Gouglidis, I. Mavridis and V. C. Hu, "Security policy verification for multi-domains in cloud systems", International Journal of Information Security, vol. 13, no. 2, **(2014)**, pp. 97-111.
[16] A. O. Diaconescu, C. Alain and A. R. McIntosh, "The co-occurrence of multi-sensory facilitation and cross-modal conflict in the human brain", Journal of neurophysiology, vol. 106, no. 6, **(2011)**, pp. 2896-2909.

Authors

Ting Cai, received her B.Eng. in computer science and technology (2006) and M.Sc. in computer application (2009) from the School of Computer Science, Wuhan University of Science & Technology, China. Now she is an associate professor of computer science at College of Mobile Telecom., Chongqing University of Posts and Telecom., China. Her current research interests include Internet Computing, Network Security Structure and Control Technology.

Jun-Zhan Wang, received his Bachelor of Management in 2005 from Military Economy College of the CPCA, China. Now he is a master student in Logistics Academy, PLA, China. His current research interests include different aspects of Information Management and Information Security.