SUMMARY This paper proposes a security violation detection method for RBAC based interoperation to meet the requirements of secure interoperation among distributed systems. We use role mappings between RBAC systems to implement trans-system access control, analyze security violation of interoperation with role mappings, and formalize definitions of secure interoperation. A minimum detection method according to the feature of RBAC system in distributed environment is introduced in detail. This method reduces complexity by decreasing the amount of roles involved in detection. Finally, we analyze security violation further based on the minimum detection method to help administrators eliminate security violation.

Key words: RBAC, interoperation, distributed environment, security violation

1. Introduction

With the rapid development of network technology and distributed applications, information interaction and cooperation between distributed systems have become increasingly frequent. In distributed environment, how to implement secure interoperation between independent systems is one of the key problems in distributed access control.

Role-Based Access Control (RBAC) is the most popular access control model at present. Compared with traditional access control models, Discretionary Access Control (DAC) and Mandatory Access Control (MAC), RBAC is a model with greater generality and flexibility, and can even express DAC and MAC strategies [1]. RBAC has been extensively applied in electronic businesses and large information systems.

To fulfill the requirements of distributed access control, extension of RBAC model has been further analyzed to solve problems of secure interoperation between systems in distributed environment [2]. Kapadia et al. proposed IRBAC, a model of access control for interoperation between distributed applications, to realize secure trans-domain operation through role mapping [3]. Joshi et al. proposed XML Role-Based Access Control (XRBAC) to constitute mapping information between domains, and provided description of role mapping [4]. Role mapping mechanism associated distributed applications together, implemented access control for interoperation between them, and therefore established secure interoperation [5]. However, the process of introducing role mapping mechanism may also introduce new security violation and result in illegal interoperation, and interoperation security would be breached [6]. When low permission gets equivalence with high permission thru transitive relations, such as mapping and hierarchy relation, a typical security violation, Cyclic Inheritance Conflict, mainly occurring in RBAC based interoperation [7]. [8] further discussed methods to eliminate security violation based on IRBAC model, but neither [8] nor IRBAC proposed methods to detect security violation. [9] proposed to collect all role information of each distributed application, merge duplicate roles, decompose and refactor ambiguous roles, establish new role relations, and compute security violation based on this information. However, it is quite difficult to compute in real distributed environment with such a centralized method.

This paper formalizes definition of RBAC based secure interoperation in distributed environment, proposes an optimized method to detect security violation according to the features of distributed environment, further analyzed types of security violation, and provides more information to eliminate security violation.

The rest of this paper is organized as follows. Section 2 reviews the traditional RBAC model. Secure interoperation in RBAC is described in Sect. 3. Our minimum security violation detection algorithm is specified in Sect. 4. Section 5 presents the simulation experiment and result analysis of security violation detection. We further analyze the security violation with case study in Sect. 6.

2. Traditional RBAC Models

In RBAC, access privileges are granted to roles, rather than users. Users acquire access privilege through their role association. Since roles in the system are relatively more stable than users and have intuitive meanings, the work of system security administrators is greatly simplified and their workload is greatly reduced.

The NIST RBAC model is defined in terms of four model components: core RBAC, hierarchical RBAC, static separation of duty (SSOD) in constraint RBAC, and dynamic separation of duty (DSOD) in constraint RBAC [11]. Core RBAC is the most basic model; hierarchical RBAC adds the concept of role hierarchy into the basic model with the hierarchical relations among roles; constraint RBAC is same with the MAC except that there are constraints to determine the acceptability of various components of RBAC and only acceptable values are permitted; Consol-
idated model RBAC combines the hierarchical RBAC and constrained RBAC [14],[15].

This paper focuses on core RBAC and hierarchical RBAC, both of which are formalized as follows:

**Definition 1 Core RBAC**
- Users, Roles, Operations, Objects and Sessions denote the set of all users, the set of roles, the set of operations, the set of accessed objects and the set of sessions respectively.
- $Permissions = 2^{(Operations \times Objects)}$, it denotes the set of all permissions.
- $PA \subseteq (Permissions \times Roles)$, it's a many-to-many mapping from the set of permissions to the set of roles, which denotes permissions conferred to the role.
- $UA \subseteq (Users \times Roles)$, it's a many-to-many mapping from the set of users to the set of roles, which denotes the role associated to the user.
- $user: Sessions \rightarrow Users$, this function returns the only user related with the specified session; users and sessions have a one-to-many relation.
- $roles: Sessions \rightarrow 2^Roles$, this function returns roles related with the session, i.e. $roles(s) = \{r|(user(s), r) \in UA\}$, users may establish sessions to selectively activate the set of roles he is associated with.

**Definition 2 Role Hierarchies**
- $RH \subseteq (Roles \times Roles)$, it's a partial order relation on the set of roles, denoted as $\geq$.
- Hierarchical $roles: Sessions \rightarrow 2^Roles$, i.e. $roles(s) \subseteq \{r|\exists r' \geq r|(user(s), r') \in UA\}$.

3. RBAC-Based Secure Interoperaton

3.1 Basic Principles of Secure Interoperation

One aim of distributed access control is to prohibit security violated interoperation between distributed systems, and any secure interoperation should conform to the basic rules of Autonomy and Security (AS):
- Rule of Autonomy. In a distributed environment, any distributed system is independent and autonomic, i.e. any operation permitted in an independent system should also be permitted in secure interoperation.
- Rule of Security. Interoperation between distributed applications should not cause security violation in any independent system, i.e. any operation prohibited in an independent system should also be prohibited in secure interoperation.

Secure interoperations implemented through role mapping between several RBAC systems should also conform to the AS basic rules. As illustrated in Fig. 1, the role hierarchy is denoted by solid lines, and role mapping between different

[Fig. 1] (1) Secure interoperation between RBAC systems (2) Security violation of interoperation between RBAC systems.

RBAC systems is denoted by dash lines. In Fig. 1 (1) mapping $RM_1$ shows that users associated with role B in Domain 1 have the permission of role Y in Domain 2. Through comparison, we can find in Fig. 1 (1) that interoperations related with role mapping $RM_1$ and $RM_2$ are secure, while in Fig. 1 (2) the establishment of role mapping resulted in interoperations violating AS basic rules:
- Through role mapping $RM_1$ and $RM_2$, in the RBAC system of Domain 2, the lower role Y has the permission of the higher role X, which is contradictory to the hierarchy of role X and Y.
- Through role mapping $RM_1$ and $RM_3$, as well as the hierarchy between role X and Y, in RBAC system of Domain 1, a hierarchy is established between role B and D, which does not exist in independent RBAC systems.

To meet the requirement of AS basic rules, we extended basic concepts of RBAC, and introduced the definition of secure interoperation based on RBAC.

**Definition 3 Model of secure interoperation based on RBAC**
- Role system $R = (Roles, I)$, wherein, $Roles$ denotes all sets of roles in RBAC system, $I$ denotes hierarchies on role sets.
- Permitted access in the role system: if $R$ is thought of as an acyclic directed graph of the point set $Roles$ and the edge set $I$, then role relation $(r', r)$ is permitted in $R$ if and only if there is a directed path in $R$ from $r'$ to $r$, denoted as $(r', r) \in R$, i.e. $(r', r) \in I^*$, wherein $I^*$ denotes the transitive closure of the hierarchy $I$.
- Independent role system: $R_i = (Roles_i, I_i)$, $i = 1, 2, \ldots, n$ and $Roles_i \cap Roles_j = \emptyset$, $i \neq j$.
- Permitted access between independent role systems: binary relations on $\cup_{i=1}^n Roles_i$, namely $RM = \{(r, r') | r \in Roles_i \land r' \in Roles_j \land i \neq j\}$, i.e. permitted interoperation between independent role systems. The mapping relation is transitive.
- Global role system: $R_0 = (Roles_0, RR)$, $Roles_0 = \cup_{i=1}^n Roles_i$, $RR = \cup_{i=1}^n I_i \cup RM$.
- Permitted access in the global role system: access is permitted in the global role system if and only if there's a directed path from $r'$ to $r$ existing in $RR$, denoted as
Based on the definition of Global Roles System, ought to conform to the autonomy rule of AS, and therefore if security violated interoperation exists in the global role system, it definitely violates the security rule of AS, i.e. \(3r, r' \in \text{Roles}_s, (r, r') \notin I_i \Rightarrow (r, r') \in RR, \) then interoperation in \(R_0\) is secure.

By its definition, the global role system \(R_0\) surely conforms to the autonomy rule of AS, and therefore if security violated interoperation exists in the global role system, it violates the security rule of AS, i.e. \(3r, r' \in \text{Roles}_s, (r, r') \in RR \& (r, r') \in I_i. \) Method of detecting security violation is discussed in the following.

4. Security Violation Detection for RBAC Based Interoperation

An abstract algorithm of security violation detection for interoperation was introduced in [12], and it discussed the problem of security violation detection for interoperation without the constraint of a specific access control model; the main idea is to centralize regional security strategies and interoperation information in all independent distributed system to form a global security strategy of interoperation, and then check that regional security strategy is in accordance with the global security strategy; if there is inconformity, then security violation exists. Such a security violation detection method is called global detection method. On the basis of [12], security violation detection for RBAC based interoperation is discussed by [10]. The relation among roles is expressed by graph [13], and security violation is detected based on combined global security strategies.

4.1 Global Detection Method

Based on the definition of Global Roles System, ought to satisfy the Autonomy rule of the AS principle. The Warshall algorithm is chosen to calculate the transitive closure of the relation matrix, since it is widely used in the estimation of connectivity in the simple directed graph.

**Algorithm 1 Global Detection Method**

Input: Role system \(R_i=\langle \text{Roles}_i, I_i \rangle, i=1,2,\ldots,n\) and binary relation \(RM\) defined on \(\cup_{i=1}^n \text{Roles}_i.\)

Output: Whether security violation exists.

(1) Construct relation matrix \(M(I_i)\) for each independent role system \(R_i=\langle \text{Roles}_i, I_i \rangle, i=1,2,\ldots,n\) respectively, wherein, \(\forall r, r' \in \text{Roles}_i, c[r,r']=\begin{cases} 1, (r, r') \in I_i \\ 0, (r, r') \notin I_i \end{cases}.\)

(2) Computer the relation matrix \(M(I_i^+)\) of transitive closure with Warshall algorithm for each role relation \(I_i, i=1,2,\ldots,n\) in independent systems respectively.

(3) For the global role system \(R_0=\langle \text{Roles}_0, RR \rangle,\) construct a relation matrix \(M(RR)\), wherein, \(\forall r, r' \in \text{Roles}' \), \(d[r,r']=\begin{cases} 1, (r, r') \in RR \\ 0, (r, r') \notin RR \end{cases}.\)

(4) Using the Warshall algorithm compute the Matrix of a Relation of the transitive closure of role relations \(RR\) in the Global Roles System, let \(M(RR^+)\) denotes the matrix of a relation.

(5) If \(\exists x \in N, \exists r, r' \in \text{Roles}_s, r \neq r' \& ct[r,r'] \neq dt[r,r'],\) i.e. \((r, r') \notin I_k \& (r, r') \in RR^+,\) then a secure violated interoperation exists in global role system \(R_0=\langle \text{Roles}_0, RR \rangle.\)

The correctness of Algorithm 1 has been proved in [9]. Whereas, [9] did not take the equivalence of the role mappings into consideration. \(3r \in \text{Roles}_s, r' \in \text{Roles}_j, i \neq j, (r, r') \in RM \& (r', r) \in RM,\) it denotes equivalence of role mapping and results in a secure violation, \(r, r' \in RR^+, (r, r) \notin I_i^+ \& (r', r') \notin I_j^+\) based on above calculation. But \((r, r) \in I_i \& (r', r') \in I_j\) has no effect in every independent role system, so it should be ignored in calculation.

Although the global detection method is effective, there are many difficulties when applying the method to real distributed environment.

- In distributed environment, it is difficult to centralize all roles and hierarchies of all independent role system to compute a global security strategy.
- The amount of roles in global security strategy will increase with the addition of independent role systems or roles, and the computational scale of global detection method enlarges when the amount of roles in global security strategy increases. Therefore such method will bring large computational load for complex distributed environment.

In view of the problems of the global detection method, we improved its computational method and introduced the minimum detection method.

4.2 Minimum Detection Method

First, minimum global role system is defined here as a new concept:

**Definition 4 Minimum global role system:**

\(R' = \langle \text{Roles}', RR' \rangle,\) it's the minimum global role system, \(\text{Roles}' = \{r | \exists r', (r, r') \in RM \& (r', r) \in RM \}, RR' = RM \cup \{(r', r') | (r, r') \notin I_i^+ \& (r', r') \in \text{Roles}' \} .\)

In the minimum global role system \(R',\) the role set \(\text{Roles}'\) consists of roles with mapping relations in each independent role system. The role relation \(RR'\) is a binary relation on \(\text{Roles}',\) and it includes permitted access relation between roles in the same independent role system in addition to the mapping relation between roles of different role systems. We compute security violation within the minimum global role system.

**Algorithm 2 Minimum Detection Method**

Input as Algorithm 1.

Output: Whether security violation exists.

The construction of the minimum role system is not consid-
ered in the algorithm and in the time complexity.

(1) Construct relation matrix \( M(I_i) \) for each independent role system \( R_i = \langle \text{Roles}_i, I_i \rangle, i = 1, 2, \ldots, n \) respectively, wherein, \( \forall r, r' \in \text{Roles}_i, c[r, r'] = \begin{cases} 1, & (r, r') \in I_i \\ 0, & (r, r') \notin I_i \end{cases} \).

(2) Compute the relation matrix \( M(I_i) \) of transitive closure with Warshall algorithm for each role relation \( I_i, i = 1, 2, \ldots, n \) in independent systems respectively, wherein, \( \forall r, r' \in \text{Roles}_i, ct[r, r'] = \begin{cases} 1, & (r, r') \in I_i^+ \\ 0, & (r, r') \notin I_i^+ \end{cases} \).

(3) For the minimum role system \( R' = \langle \text{Roles}', RR' \rangle \), construct a relation matrix \( M(RR) \), wherein, \( \forall r, r' \in \text{Roles}', d[r, r'] = \begin{cases} 1, & (r, r') \in RR' \\ 0, & (r, r') \notin RR' \end{cases} \).

(4) Compute the relation matrix \( M(RR') \) of transitive closure with Warshall algorithm for the role relation \( RR' \) in the minimum global role system, wherein, \( \forall r, r' \in \text{Roles}', dt[r, r'] = \begin{cases} 1, & (r, r') \in RR'^+ \\ 0, & (r, r') \notin RR'^+ \end{cases} \).

(5) If \( \exists x \in N, \exists r', r' \in \text{Roles}_x, r \neq r' \land ct[r, r'] \neq dt[r, r'] \), i.e., \( (r, r') \notin I_i^+ \land (r, r') \in RR'^+ \), then security violated interoperation exists in the minimum global role system \( R' \).

Now we have to prove that the results of the minimum detection method and global detection method are consistent for the same interoperation, i.e. security violated interoperation exists in the minimum global role system \( R' \), if and only if the same security violated interoperation exists in the global role system \( R_0 \).

**Prove:** Since \( RR' \subseteq RR \Rightarrow RR'^+ \subseteq RR^+ \), if security violated interoperation exists in the minimum global role system \( R' \), i.e., \( (r, r') \notin I_i^+ \land (r, r') \in RR'^+ \), then it surely exists in the global role system \( R_0 \), i.e., \( (r, r') \notin I_i^+ \land (r, r') \in RR^+ \). Assume that security violated interoperation exists in the global role system \( R_0 \), i.e. \( (r, r') \notin I_i^+ \land (r, r') \in RR^+ \), but does not exist in the minimum global role system, i.e. \( (r, r') \notin RR^+ \), it can be deduced that \( (r, r') \in RR^+ - RR^+ \), and therefore roles \( r, r' \) cannot be associated with each other by any mapping relation, and we have \( (r, r') \in I_i^+ \), which is contradictory to our assumption. Thus any security violated interoperation existing in the global role system \( R_0 \) exists in the minimum global role system \( R' \). Therefore, \( (r, r') \notin I_i^+ \land (r, r') \in RR^+ \Rightarrow (r, r') \notin I_i^+ \land (r, r') \in RR^+ \), the results of the minimum detection method and the global detection method are consistent.

### 4.3 Comparison of Computational Complexity

In distributed environment, mapping information among roles can be saved in systems outside those independent role systems. The advantage of the minimum detection method is that it works without all the information of roles or their hierarchies of each independent role system, so the amount of roles involved in computation is less than the amount of total roles of all independent role systems, and hence the computational complexity is reduced. The time complexities of both methods are compared in the following.

\( q \) denotes steps in the algorithm, \( T_1^q \) and \( T_2^q \) denote the time complexity of corresponding steps of each method respectively. The amount of roles of independent role system \( R_i (i = 1, 2, \ldots, n) \), global role system \( R_0 \) and minimum global role system \( R' \) is denoted as \( m_i \), \( m_0 \), \( m' \) respectively. One-pass traverse of the relation matrix is enough in order to construct relation matrices or compare relation matrices, and their time complexity is \( O(m^2) \); however, the time complexity of computing the relation matrix of transitive closure with Warshall algorithm is \( O(m^3) \). Therefore, total time complexities of these two methods are:

\[
(1) \quad T_{\text{total}1}(m_0) = \sum_{q=1}^{s} T_1^q = O\left(\sum_{i=1}^{n} m_i^2\right) + O\left(\sum_{i=1}^{n} m_i^3\right) + O(m_0^3) + O(m_0^2) + O(m_0^3)
\]

\[
(2) \quad T_{\text{total}2}(m_0) = \sum_{q=1}^{s} T_2^q = O\left(\sum_{i=1}^{n} m_i^2\right) + O\left(\sum_{i=1}^{n} m_i^3\right) + O(m_0^2) + O(m_0^3) + O(m_0^3)
\]

Since the computing processes of step 1 and 2 of both methods are the same, only time complexities of other steps are considered, which can be deduced from formula (1) and (2):

\[
(3) \quad T'_{\text{total}1}(m_0) = O(m_0^3)
\]

\[
(4) \quad T'_{\text{total}2}(m_0) = O(m_0^3)
\]

It can be found in formula (3) and (4) that detection time of security violation for interoperation is proportional to the cube of total amount of roles involved in computation. The less the amount of roles involved in role mapping is, the less the computing time of the minimum detection method is than the global detection method.

### 5. Simulation Experiment and Result Analysis of Security Violation Detection

We performed simulation experiments of both security violation detection methods on PC with Pentium 2.4 GHz, 1 G RAM and Windows XP.

In simulated environment, the global role system has following parameters:

(1) \( n \) denotes the amount of independent role systems.
(2) \( m' \) denotes the amount of roles involved in interoperation.
(3) \( m_0 \) denotes total amount of roles in all independent role system.
(4) \( t \) denotes the computing time of security violation detection.

The experiment simulated the global detection method and the minimum detection method through adjustment of \( n \), \( m' \) and \( m_0 \), recorded time spent on detection, and analyzed data of the experiment.
WANG et al.: SECURITY VIOLATION DETECTION FOR RBAC BASED INTEROPERATION IN DISTRIBUTED ENVIRONMENT

Table 1 Data of simulation experiment for the global detection method.

<table>
<thead>
<tr>
<th>m0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>15</td>
<td>1</td>
<td>16</td>
<td>1</td>
<td>15</td>
<td>1</td>
<td>16</td>
<td>1</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>200</td>
<td>63</td>
<td>78</td>
<td>62</td>
<td>63</td>
<td>78</td>
<td>63</td>
<td>62</td>
<td>63</td>
<td>62</td>
<td>63</td>
</tr>
<tr>
<td>300</td>
<td>188</td>
<td>218</td>
<td>204</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>400</td>
<td>532</td>
<td>531</td>
<td>516</td>
<td>546</td>
<td>532</td>
<td>531</td>
<td>531</td>
<td>516</td>
<td>531</td>
<td>531</td>
</tr>
<tr>
<td>500</td>
<td>1000</td>
<td>985</td>
<td>1015</td>
<td>1094</td>
<td>984</td>
<td>1016</td>
<td>1078</td>
<td>1031</td>
<td>1016</td>
<td>1000</td>
</tr>
<tr>
<td>600</td>
<td>1688</td>
<td>1765</td>
<td>1688</td>
<td>1797</td>
<td>1734</td>
<td>1734</td>
<td>1782</td>
<td>1765</td>
<td>1688</td>
<td>1875</td>
</tr>
<tr>
<td>700</td>
<td>2687</td>
<td>2828</td>
<td>2750</td>
<td>2703</td>
<td>2797</td>
<td>2688</td>
<td>2797</td>
<td>2734</td>
<td>2703</td>
<td>2688</td>
</tr>
<tr>
<td>800</td>
<td>4015</td>
<td>3938</td>
<td>4015</td>
<td>3985</td>
<td>3984</td>
<td>3922</td>
<td>3953</td>
<td>3969</td>
<td>3953</td>
<td>4016</td>
</tr>
<tr>
<td>900</td>
<td>5609</td>
<td>5641</td>
<td>5843</td>
<td>5609</td>
<td>5579</td>
<td>5812</td>
<td>5578</td>
<td>5781</td>
<td>5625</td>
<td>5578</td>
</tr>
<tr>
<td>1000</td>
<td>7812</td>
<td>7734</td>
<td>7719</td>
<td>7641</td>
<td>7625</td>
<td>7625</td>
<td>7672</td>
<td>7672</td>
<td>7797</td>
<td>8391</td>
</tr>
</tbody>
</table>

5.2 Experiment Analysis of the Minimum Detection Method

(1) Set the amount of independent role systems \( n = 10, 20, 30, 40, 50 \), and randomly set the amount of roles in each independent role system as well as role relations.

(2) Set the total amount of roles in all independent role systems \( m_0 = 1000 \).

(3) Set the total amount \( m' \) of roles involved in interoperation from 100 to 1000 with an increment of 100 each, and randomly generate role mapping relations.

(4) Do the experiment for 10 times for each \( n, m' \), record computing time of the minimum detection method, and compute the mean time.

In Table 2, the first row is the amount of roles involved in interoperation \( m' \), the first column is the amount of independent role systems \( n \), and other data are means of computing time. Following two conclusions can be drawn from experiment data in Table 2:

(1) As illustrated in Fig. 3, we can find from upward curves that for each \( n \), time spent on computation is prolonged as \( m' \) increases, and the amount of roles involved in interoperation \( m' \) can be thought of as the main parameter affecting the computing time of the minimum detection method.

(2) As illustrated in Fig. 3, we can find from the horizontal curves that for each \( m' \), time spent on computing is reduced as \( n \) increases. When \( m' \) is much less than \( m_0 \), the effect is great; with the increase of \( m' \), the effect becomes less.

5.3 Analysis of Comparison Experiment of Both Methods

(1) Set the amount of independent role systems \( n = 10 \), and randomly set the amount of roles in each independent role system as well as role relations.

(2) Set the total amount of roles in all independent role systems \( m_0 = 1000 \).

(3) Set the total amount \( m' \) of roles involved in interoperation from 100 to 1000 with an increment of 100 each, and randomly generate role mapping relations.

(4) Do the experiment for 10 times for each \( m' \), record the time of computation which is differently implemented.
Table 2: Data of simulation experiment for the minimum detection method.

<table>
<thead>
<tr>
<th></th>
<th>100</th>
<th>200</th>
<th>300</th>
<th>400</th>
<th>500</th>
<th>600</th>
<th>700</th>
<th>800</th>
<th>900</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>270.9</td>
<td>312.5</td>
<td>596.8</td>
<td>810.9</td>
<td>1339.1</td>
<td>1978.2</td>
<td>2906</td>
<td>4148.5</td>
<td>5845.3</td>
<td>7901.7</td>
</tr>
<tr>
<td>20</td>
<td>116.1</td>
<td>156.2</td>
<td>343.1</td>
<td>598.4</td>
<td>1051.6</td>
<td>1748.4</td>
<td>2684.4</td>
<td>3950.1</td>
<td>5559.3</td>
<td>7606.2</td>
</tr>
<tr>
<td>30</td>
<td>58.3</td>
<td>106.3</td>
<td>242.2</td>
<td>551.6</td>
<td>1020.4</td>
<td>1704.7</td>
<td>2675</td>
<td>3918.7</td>
<td>5532.8</td>
<td>7548.4</td>
</tr>
<tr>
<td>40</td>
<td>44.1</td>
<td>96.8</td>
<td>225</td>
<td>543.8</td>
<td>1006.3</td>
<td>1687.5</td>
<td>2640.7</td>
<td>3904.6</td>
<td>5523.5</td>
<td>7526.5</td>
</tr>
<tr>
<td>50</td>
<td>37.7</td>
<td>79.6</td>
<td>218.9</td>
<td>528.1</td>
<td>996.9</td>
<td>1678.3</td>
<td>2634.4</td>
<td>3896.9</td>
<td>5498.4</td>
<td>7515.7</td>
</tr>
</tbody>
</table>

Fig. 3: Mean of simulation experiment data of the global detection method.

Table 3: Comparison of computing time of security violation between global detection method and minimum detection method.

<table>
<thead>
<tr>
<th>$m'$</th>
<th>$\left(\frac{m_0}{m'}\right)^3$</th>
<th>$\frac{t'}{t}$</th>
<th>$\frac{t}{t'}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>8000</td>
<td>1</td>
<td>7585.5</td>
</tr>
<tr>
<td>100</td>
<td>1000</td>
<td>7.5</td>
<td>1011.4</td>
</tr>
<tr>
<td>150</td>
<td>296.2963</td>
<td>24.6</td>
<td>308.35366</td>
</tr>
<tr>
<td>200</td>
<td>125</td>
<td>58.8</td>
<td>129.0051</td>
</tr>
<tr>
<td>250</td>
<td>64</td>
<td>115.4</td>
<td>65.73224</td>
</tr>
<tr>
<td>300</td>
<td>37.03704</td>
<td>199.8</td>
<td>37.96547</td>
</tr>
<tr>
<td>350</td>
<td>23.32362</td>
<td>321.6</td>
<td>23.58675</td>
</tr>
<tr>
<td>400</td>
<td>15.625</td>
<td>507.5</td>
<td>14.9468</td>
</tr>
<tr>
<td>450</td>
<td>10.97394</td>
<td>722.9</td>
<td>10.49315</td>
</tr>
<tr>
<td>500</td>
<td>8</td>
<td>979.2</td>
<td>7.74663</td>
</tr>
<tr>
<td>550</td>
<td>6.01052</td>
<td>1304</td>
<td>5.8171</td>
</tr>
<tr>
<td>600</td>
<td>4.62963</td>
<td>1674.2</td>
<td>4.53082</td>
</tr>
<tr>
<td>650</td>
<td>3.64133</td>
<td>2121.8</td>
<td>3.57503</td>
</tr>
<tr>
<td>700</td>
<td>2.91545</td>
<td>2638.5</td>
<td>2.87493</td>
</tr>
<tr>
<td>750</td>
<td>2.37037</td>
<td>3226</td>
<td>2.35136</td>
</tr>
<tr>
<td>800</td>
<td>1.95313</td>
<td>3907.5</td>
<td>1.94127</td>
</tr>
<tr>
<td>850</td>
<td>1.62833</td>
<td>4657.4</td>
<td>1.6287</td>
</tr>
<tr>
<td>900</td>
<td>1.37174</td>
<td>5562</td>
<td>1.36381</td>
</tr>
<tr>
<td>950</td>
<td>1.16635</td>
<td>6518.3</td>
<td>1.16372</td>
</tr>
<tr>
<td>1000</td>
<td>1</td>
<td>7385.5</td>
<td>1</td>
</tr>
</tbody>
</table>

Fig. 4: Comparison between theoretical value and computed value of time on computing security violation.

6. Analysis of Security Violation

The detection method discussed above just find out if there is security violation in interoperations. Further analysis of matrixes in detecting course is also important to assist administrators to deal with security violation.

6.1 Analysis of Security Violation

When analyzing security violation, we first have to understand the characteristic of security violation. According to basic principles of secure interoperation, there are generally two types of violation:

- **Non-associated Violation** There exist two roles in one...
independent role system which do not have access relation but have permitted access relation in the global role system, i.e. \( \exists r, r', (r, r') \in I_1 \land (r, r') \in I_2 \land (r, r') \in RR \).

- **Associated Violation** There exist two roles with access relation in one independent role system, but the one in the lower hierarchy has access to the one in the higher hierarchy, i.e. \( \exists r, r', (r, r') \in I_1 \land (r', r') \in I_2 \land (r, r') \in RR \).

\[
\text{if}(r' \neq r' \& \& \text{ct}(r, r') \neq \text{dt}(r, r')) \{
\text{if}(\text{dt}(r, r') \neq \text{dt}(r', r)) \{
\text{No associated violation in } r \rightarrow r';
\} \text{else}
\text{Associated violation in } r \rightarrow r';
\}
\]

//Examine all roles which can arrive at \( r' \) for each \( r'' \in \text{Roles} \) { if(\( r'' \in \text{Roles} \)) add to the set of temporary roles } Find \( IRM \) which cause security violation: \( IRM = \{(r, r') | r, r' \in \text{TempRoles} \land (r, r') \in RM \} \)

On the basis of the minimum detection method, we can analyze the type of security violation with the algorithm above, and have the role mapping relation which causes security violation. To keep the autonomy of each independent role system, we should eliminate security violation, to the greatest extent through modification of role mapping. For non-associated violation, we only have to consider elimination of relations of roles, but for associated violation the relation of roles shall be modified with prudence so as to avoid lower roles accessing higher roles while not undermining legitimate relations.

### 6.2 Case Analysis

Now we take the interoperation in Fig. 1 (2) as an example to introduce method of security violation detection.

According to the minimum detection method, firstly the relation matrices \( M(I_1), M(I_2) \) of independent role system \( R_1, R_2 \) are constructed respectively, and then relation matrices \( M(I_1^+), M(I_2^+) \) of transitive closures of \( I_1, I_2 \) are computed.

\[
M(I_1) = \begin{bmatrix}
A & B & C & D \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
A & B & C & D \\
1 & 0 & 0 & 0
\end{bmatrix}
\]

\[
M(I_2) = \begin{bmatrix}
X & Y & Z \\
0 & 0 & 1 \\
0 & 0 & 0 \\
1 & 0 & 0 \\
0 & 0 & 0 \\
Y & 0 & 0 & 0 \\
Z & 0 & 0 & 0 \\
Y & 0 & 0 & 0 \\
Z & 0 & 0 & 0 \\
\end{bmatrix}
\]

Then construct relation matrix \( M(RR') \) of the minimum global role system \( R' \), and compute relation matrix \( M(RR'^+) \) of transitive closure of \( RR' \):

\[
M(RR') = \begin{bmatrix}
A & B & D & X & Y & Z \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
A & 0 & 1 & 1 & 1 & 1 \\
B & 0 & 1 & 1 & 1 & 1 \\
X & 0 & 1 & 1 & 1 & 1 \\
Y & 0 & 1 & 1 & 1 & 1 \\
Z & 0 & 0 & 0 & 0 & 0 \\
A & B & D & X & Y & Z \\
A & 0 & 0 & 0 & 0 & 0 \\
B & 0 & 0 & 0 & 0 & 0 \\
X & 0 & 0 & 0 & 0 & 0 \\
Y & 0 & 0 & 0 & 0 & 0 \\
Z & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

We can find through comparison that there exists security violation in the minimum global role system \( R' \).

- In role system \( R_1 \), \( \forall (r, r') \in [(A, D), (B, D)] \), \( (r, r') \in I_1 \wedge (r, r') \in RR'^+ \).

- In role system \( R_2 \), \( (Y, X) \in I_2 \wedge (Y, X) \in RR'^+ \).

According to the method introduced in the previous section, we can determine:

Role relation \( (A, D), (B, D) \) may cause non-associated violation, and role mapping relations \( [(B, X), (Y, B), (Y, D)] \) can be modified, such as getting rid of role mapping \( (B, X) \).

### 7. Case Studies

To demonstrate the application of the role based secure interoperation and security violation detection method, we performed case studies with realistic scenarios in State Street Corporation (SSC). SSC is a global corporation providing IT financial services, whose IT development and support is also distributed in the whole world. Because the collaboration and communication is going to globalization, the security becomes more and more important in the interoperation among different regions.

#### 7.1 Interoperation Environment

SSC IT staffs are playing many different roles in software development and support process. Table 4 is a brief list of the roles and responsibilities.

SSC global collaborative process covers multidomains, such as US team, China team and Europe team etc and each one has its own environment and applications
Table 4 Parameters for security violation detection.

<table>
<thead>
<tr>
<th>Roles</th>
<th>Responsibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Sponsor (PS)</td>
<td>Propose the product and approve the initiation.</td>
</tr>
<tr>
<td>Product Manager (PDM)</td>
<td>Communicate with the project sponsors and outline the goals (vision) for the product.</td>
</tr>
<tr>
<td>Project Manager (PM)</td>
<td>Develop and maintain a detailed project plan; Monitor project execution and performance.</td>
</tr>
<tr>
<td>Development Manager (DM)</td>
<td>Manage and coordinate all of development works.</td>
</tr>
<tr>
<td>Business Analyst (BA)</td>
<td>Work on business documentation; Take charge of all issues at the business level; Clarify the business requirement to the team.</td>
</tr>
<tr>
<td>Architect</td>
<td>Propose the system architecture and the corresponding technique.</td>
</tr>
<tr>
<td>Develop Team Leader (DTL)</td>
<td>Organize all the development activities. Take all responsibilities for technical issues.</td>
</tr>
<tr>
<td>Developer (DEV)</td>
<td>Work on implementation, unit test and documents.</td>
</tr>
<tr>
<td>Quality Assurance Manager (QAM)</td>
<td>Approve the test plan, test case and test result. Approve the release before systems go to UAT or production.</td>
</tr>
<tr>
<td>Quality Assurance Leader (QAL)</td>
<td>Organize all the QA activities. Monitor the execution in the whole QA cycle.</td>
</tr>
<tr>
<td>Quality Assurance (QA)</td>
<td>Draft test case and submit the test report.</td>
</tr>
<tr>
<td>Tester</td>
<td>Do the execution and record the test result.</td>
</tr>
<tr>
<td>Deployment Support (DS)</td>
<td>Take charge of the deployment work in DEV, QA, UAT and production environment.</td>
</tr>
<tr>
<td>Database Administrator (DBA)</td>
<td>Database management and support.</td>
</tr>
<tr>
<td>System Administrator (SA)</td>
<td>Manage and support the various system environments; Back up data and disaster recovery.</td>
</tr>
<tr>
<td>User Acceptance Test (UAT)</td>
<td>Execute the transactions that the business users would run. Highlight any potential areas of negative impacts on the business processes.</td>
</tr>
<tr>
<td>Project Management Officer (PMO)</td>
<td>Manage the overall resources of projects; Monitor the process for all projects. Contribute to project strategy, policy and procedure.</td>
</tr>
<tr>
<td>Configuration Manager (CM)</td>
<td>Administer the configuration management repository. Manage and record all release versions according to the configuration management procedures.</td>
</tr>
</tbody>
</table>

generally. Interoperation often happens across domains that each one maintains its own security settings such as user, role and permission etc to protect applications and information, thus there are many interoperations in the global collaborative development. Two typical situations are listed as following:

(1) Several domains share user identity information, such as employee id, name, division and office call etc in the same identity data source.

(2) There is no any intersection of user identity information across different domains.

Roles mapping is easier way to interoperate across multi-domains in above two situations. It can keep logical independency within domain while building links to the others. Role is also more flexible to map, because the amount of user or permission is huge in a global enterprise.

7.2 Building Interoperation

SSC senior managers identify human resources in different domain on the specific situation of each project. Generally, China team is responsible for most of DEV works and US team plays other roles in the whole project. The interoperation can be built as below:

Scenario One: At requirement analysis stage, China team requests use case from BA in US team. QA in US team requests delivery for functional testing at validation stage. RBAC based secure interoperation can be built simply as below, according to Definition 3. Domain 1 denotes the environment and applications in China, and domain 2 denotes those in US. Secure interoperation is defined as below:

(1) $\text{Roles}_1 = \{\text{DEV}\}$, $\text{Roles}_2 = \{\text{BA}, \text{QA}, \text{UAT}, \text{CM}\}$

(2) $\text{RM} = \{(\text{DEV}, \text{BA}), (\text{QA}, \text{DEV})\}$

The secure interoperation denotes that

(1) Users in domain 1 assigned as the role DEV can have access permission of the role BA in domain 2.

(2) Users in domain 2 assigned with role QA can have access permission of role DEV in domain 1.

Scenario Two: At testing stage, QA in US team requests delivery for functional testing from DTL in China team. DEV in China team requests specific testing report from QA in US team. Secure interoperation is defined as below:

(1) $\text{Roles}_1 = \{\text{DTL}, \text{DEV}\}$, $\text{Roles}_2 = \{\text{QA}\}$

(2) $I_1 = \{(\text{DTL}, \text{DEV})\}$, $I_2 = \emptyset$

(3) $\text{RM} = \{(\text{QA}, \text{DTL}), (\text{DEV}, \text{QA})\}$

7.3 Security Violation Discussion

When the interoperation is performed with role mapping, a global session is created to calculate roles set which includes all the roles generated by transitive relations, such as mapping and hierarchy relation. Then security violation can be found:

(1) In the scenario one, global session has such relation $\text{QA} \rightarrow \text{DEV} \rightarrow \text{BA}$ which means users assigned as the role QA in domain 2 can have permission of the role BA.

(2) In the scenario two, global session has such relation $\text{DEV} \rightarrow \text{QA} \rightarrow \text{DTL}$ which means users assigned as the role DEV in domain 1 can have permission of the role DTL.

The roles, DEV in Scenario One and QA in Scenario Two, are assigned too many permission that causes security violation.

7.4 Detection Performance Discussion

As demonstrated above, interoperation can happen in a single project process, also can be in multi-project processes, such as projects under the same business or projects in different business etc. In SSC, there are more than 400 applications to support business and operation. Although the total set of the roles which are possibly involved in the application development is large, actually most of application development only involve the part of the whole set. According to our statistics, the amount of roles is 10 30 in the most of applications development process. According to realistic
scenarios exampled above in most of development projects, BA, DEV and QA are often participated in interoperation to access codes and documents etc. Naturally, the roles BA, DTL, DEV and QA are involved in roles mapping. So the amount of roles in interoperation is 14. Table 5 shows the parameters for security violation detection according to above analysis.

Actually, most of enterprise solutions don’t maintain roles in distributed domains physically, but just logically. For example, an enterprise LDAP tree to keep all the roles and many branches to reflect configuration of applications. Above statistic comes from the enterprise LDAP tree in SSC.

As discussed in Sects. 4 and 5, we chose the critical value from parameter list in Table 5 and got such evaluation result on \( t' \):

\[
\frac{4000}{1600} < \frac{t}{t'} < \left( \frac{12000}{400} \right)^3 \Rightarrow 15.6 < \frac{t}{t'} < 27000
\]

Since the roles mapping is not applied arbitrarily in realistic scenarios, minimum method is more effective than global method to detect security violation in role based secure interoperation model.

8. Conclusions

This paper has discussed methods of security violation detection for RBAC based interoperation in distributed environment. Requirements of secure interoperation in distributed environment are analyzed in detail, and the definition of RBAC based secure interoperation are introduced and formalized on the basis of Core RBAC and Hierarchical RBAC. According to features of RBAC based interoperation in distributed environment, methods of security violation detection are discussed intensively. Finally the case study in SSC global collaborative development process can prove that minimum security violation detection method has better performance than global one in realistic enterprise applications.

In future research, we intend to carry out our work in following aspects:

(1) Further improve the method of solving security violation, and provide users with best overall solution strategy, rather than simply compute the security of interoperation.

(2) In RBAC models more broadly, e.g. Constraint RBAC and Symmetric RBAC, define RBAC based secure interoperation and provide corresponding solutions to security violations.

(3) Design and implement the system architecture of interoperation between RBAC systems in integrated distributed environment.

References

Xinyu Wang received the B.E. degree in the department of Computer Science and Engineering from Zhejiang University (China) in 2002. From 2002 to 2007, he was a research assistant in the Zhejiang University, where he got the D.E. degree from the college of Computer Science. Since the June 2007, he has been the lecturer in the same university. His primary research interests include Software engineering, distributed software architecture and distributed computing. He is a member of IEEE CS.

Jianling Sun received the B.E. and D.E. degrees of Computer Application from Zhejiang University (China) in 1990 and 1993 respectively. Since October 1993, he has been a faculty member of the College of Computer Science and Engineering at Zhejiang University. His current research interests include database systems, software engineering, and software architecture.

Xiaohu Yang received his Ph.D. degree of computer science at Department of Computer Science and Engineering, Zhejiang University in 1993. Since 1994, he has been a faculty member and associate professor of Department of Computer science and Engineering at Zhejiang University. His research interests include software engineering, legacy system reengineering, and software technology for financial services.

Chao Huang received the B.E. degree in the department of Computer Science and Engineering from Zhejiang University (China) in 2005. He is a research assistant and Ph.D. candidate in Zhejiang University now. His primary research interests include distributed software architecture, and enterprise security architecture.

Di Wu received the B.E. degree in the department of Computer Science and Engineering from Zhejiang University (China) in 2001. He received his Ph.D. degree of computer science at Department of Computer Science and Engineering, Zhejiang University in 2006. Since 2007, he has been a software engineer in the State Street Technology Zhejiang. His primary research interests include distributed software architecture, distributed computing and enterprise security architecture.