International Journal of Security and Its Applications
Vol.9, No.12 (2015), pp.357-366
http://dx.doi.org/10.14257/ijsia.2015.9.12.33

Cross-Domain Authorization Management Model for Multi-Levels Hybrid Cloud Computing

Li Na[1], Dong Yun-Wei[1], Che Tian-Wei[2], Wang Chao[3],
Gao Yang[4] and ZhangYu-Chen[3]

[1]*The School of Computer Science and Technology, Northwestern Polytechnical University, Shaanxi province, Xian, China*
[2]*The Computer Institute, Xidian University, Shaanxi province, Xian, China*
[3]*Information Engineering University, ZhengZhou, China*
[4]*The Institute of Surveying and mapping, Shaanxi province, Xian, China*

Abstract

Aimed at the security problems of the cross-cloud, cross-level and cross-domain in multi-level hybrid cloud computing, the singleness of the role establishment method, the implicit promotion of privilege and the separation of duties conflict in the traditional cross-domain authorization management models, a new cross-domain authorization management model for multi-levels hybrid cloud computing is proposed based on a novel two-tier role architecture. The two-tier role architecture which is setted in the area of arrangement can better meet the practical needs of role establishment and management. Based on that, the proposed unidirectional role mapping for cross-domain authorization can avoid the role mapping rings. Besides, by introducing attribute and condition, dynamic adjustment of privileges is realized. The model is described formally in dynamic description logic, including concepts, relations and management operations. Finally, the security of the model is analyzed and an example is presented to illustrate the effectiveness and practicality.

Keywords: *Cloud Computing; Multi-Levels Hybrid Cloud Computing; Cross-Domain Role Mapping; Authorization Management Model; Dynamic Description Logic*

1. Introduction

Cloud Computing is a new computing mode which stores computing resources in configurable shared pool of computing resources, enabling access to computing resources through some nets that are convenient, available, and on-demand. Cloud computing is one of the current hot topic in the field of information technology, which is the focus concerned by the industry, the academia and the government. As one of the five deployment patterns of cloud computing, multi-level hybrid cloud has received wide attention in recent years.

Multi-levels hybrid cloud is composed of cloud modes of different security levels, of which each cloud has different security levels and security requirements, remaining relatively independent, mutually providing interoperability and data sharing, which must bring security issues among clouds, levels and domains. Through a combination of many new and more extensive cloud services, the cloud can provide cross-level cloud authentication, authorization, access control and other security functions. Due to the characteristics of cloud service mode itself, these security issues are more complex. Though the interoperability between domain of multi-levels hybrid cloud realized the sharing of the domain resources and services. How to ensure the security of managed subjects and objects, realize interoperability between different domains safely subject

ISSN: 1738-9976 IJSIA

information sharing, make access control decision and access check, and to realize the authorization in multi-level hybrid cloud are important problems to be resolved.

The main methods of cross-domain authorization are mapping, delegation, strategy integration, *etc*. [1-10] Because role-based access control (RBAC) model is widely used in the information system, using the role mapping method to achieve cross-domain authorization and secure interoperability has got great attention with good universality and practicality. And a series of models and corresponding realizations based on cross-domain role mapping are proposed. But they are easy to cause domain shuttle, potential safety hazard such as cross-domain , implicit promotion of privilege and so on.

IRBAC2000 [1] implemented the integration of access control policies between two static domains, while dRBAC [2] focused on dynamic change of permissions in multi-domain interoperation. But these two models are subject to several security problems such as implicit promotion of privilege and separation of duties conflict. In order to ensure the security of domain asked interoperability, many scholars have explored from different angles. SHAFIQ B [3] tried to eliminate cross-domain security conflicts through multi-domain policy integration algorithm. In the work of MOHAMED S [4], cross-domain access paths were constructed to guarantee the security and effectiveness of the mapping chain. Li [5] introduced risk and confidence factors to avoid the security violation cases. Zan Yang [6] introduced the concept of domain and integrated it with roles, permissions and domains to solve cross-domain access problems and the uncertainty during role mapping based on the fuzzy theory, improving the accuracy and practicability of role mapping. Eric Freudenthal [7] provided an interoperable method based on delegation mechanism for multi-domain interoperation through third party delegation and value attribute. Hong Fan[8] introduced the concept of global role, domain role and related role to establish the mapping relationship among multi domains. Guo Dongheng [9-10] adjusted the threshold attribute and domain size of role mapping to dynamically decide the mapping capability of a role, improving the security of cross domain interoperability. However, all the above methods are proposed based on the classical RBAC with the characteristic of role management in a single domain. And the classical RBAC model has the deficiencies such as the singleness of the role establishment method, the implicit promotion of privilege and the separation of duties conflict and so on. No matter from the perspective of organizational division of function, or from the perspective of the application system of business divisions, all can't implement the authorization management well, limits the popularization and application of the model.

In order to resolve these problems above, a cross-domain authorization management model was proposed for multi-levels hybrid cloud computing. In the model, a two-tier role architecture was proposed to improve the model's practicability. And cross-domain unidirectional role mapping method was put forward to enhance the security of access control in multi-domain environment. The model can support dynamic and cross-domain authorization management, defined formally in dynamic description logic.

2. Dynamic Description Logic

Description logic is a decidable subset of first-order logic with strong representation and reasoning ability. But it's limited to the static knowledge representation and reasoning, can't effectively support the characterization and reasoning of dynamic knowledge. Dynamic description logic [11-14] is a formal description tool used to describe and reasoning dynamic knowledge, which is proposed based on description logic. It not only has strong ability of description, but also ensures the decidability of some reasoning problems. With the efficient inference algorithm, it has been a useful tool to represent access control model.

In 2004, Shi Zhongzhi [11] put forward Dynamic Description Logic (DDL) based on the description logic, dynamic logic and action theory, bringing a new formal logic

framework to simultaneously handle static and dynamic knowledge. Based on Shi's theory, Chang Liang [12-14] extended the expression and reason of DDL and proposed its tableau decision algorithm, providing the logical support for the modeling and reasoning of action. The concepts and symbols about DDL used in this paper are presented based on the work in literature [11-15], including concepts, relations, formulas, actions, and some theorems. Besides, "role" in DDL is defined as "relation" in this paper so as to distinct with the notion "role" in RBAC.

3. Main Idea of the Model

3.1 Two-tier Role Architecture

In order to reduce the burden of authorization management, the concept of role was introduced into the classic RBAC as a bridge of users and permissions. Authorization management in RBAC includes User-Role assignment and Role-Permission assignment. As known to all, Role-Permission assignment requires deep knowledge of application level semantics, while User-Role assignment is a personnel management function of organization level, which requires greater understanding of human side. In an RBAC system, actually it's hard to search a manager from holding the knowledge of both human side and application system at the same time.

To resolve this problem, based on the work of REN Zhiyu [15], we divide the traditional role into position role of organization level and application role of application level, and a two-tier role architecture is proposed as Figure 1. In the organization level, position role is defined based on the position and responsibility of users in the organization. While in the application level, application role is defined according to the business process of the application system.

Based on the two-tier role architecture, the authorization management will be more explicit. On the one hand, administrators of organization level holding the knowledge of human side take responsibilities for the creation of position role, user-position role assignment and the mapping relation between these two roles. After mapping relationship, the late maintenance workload is small. On the other hand, administrators of application level familiar with the information system are responsible for the creation of application role and application role-permission assignment. At the same time, both roles retain role hierarchy and role tree, and two kinds of nodes are defined: virtual node and role node, where virtual node is used as a virtual role to build the role tree. Virtual node in the position role tree is known as an organization or department, while several application systems in the application role tree.

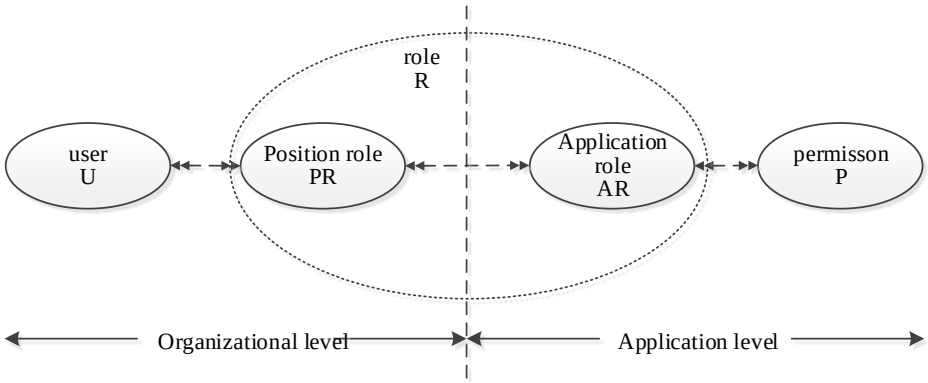


Figure 1. Two-Tier Role Architecture of Position Role and Application Role

2.2. Cross Domain Authorization Management based on Unidirectional Role Mapping

Unidirectional role mapping method was used to realize the cross-domain authorization management and maintain the independence of the internal policies in a single domain. In order to maintain the independence of the internal security policies, the position role is divided into internal role and mapping role. The former role is used to authorization in a single domain, which is only allowed to associate with the application role and permissions in the internal domain. While the latter is specifically designed to cross-domain role mapping.

To accord with the start and end point of the role mapping, the mapping position role includes In-role and Out-role. The former is at the start point of the mapping relationship, used to obtain the permission mapped from external domain. While the latter is at the end point, used to map the permissions in the internal domain to other domains. The set of In-role and that of Out-role are disjoint. Users assigned In-role can obtain permissions from the external domain, but it's not allowed for users assigned Out-role. The Out-role can associate with application role to obtain the permissions and then map them to other domains, but it's not allowed for In-role.

Through the division of the position roles into internal role and mapping role, the model can protect the security policy from external influence of other domains. And by dividing mapping position role into In-role and Out-role, the model can ensure the cross-domain permission isolation and make the permission flow unidirectional. Meanwhile, through role division and unidirectional role mapping, the model can implement static and dynamic separation of duty (SSoD, DSoD). Besides, trust, condition and attribute can be introduced into the model to realize dynamic authorization, improving the flexibility, security and extensibility.

4. DDL-Based Description of the Model

The proposed cross-domain authorization management model for multi-level hybrid cloud computing is shown as Figure 2. The concepts, relations and operations are described in dynamic description logic (DDL), where operations are expressed as actions in DDL.

Definition 1: Concepts

(1) *USER*、*S*、*ROLE*、*PRMS*、*OPE*、*RES* represent the set of users, sessions, roles, permissions, operations and resources respectively. As the same with classic RBAC, permission is a tuple consisted of operation and resource.

(2) *AROLE* and *PROLE* are respectively application roles and position roles. $ROLE \equiv AROLE \sqcup PROLE$.

(3) *CARD* means cardinal number, and $CARD \sqsubseteq N$. It mainly means the maximum number of role assignment, which represents the maximum number of position role assigned to users or that of application role mapped to application role.

(4) *ATTR* means attributes. Users, roles and permissions all have attributes. Attributes can be used in user-role assignment, role activation and access process.

(5) *SYS* means system states, which are system environmental parameters, such as time, location and IP address, *etc*.

(6) *CON* means condition, representing the constraints when users access resources.

The concepts above use the name of management domain as the subscript, indicating that the concept belongs to different management domains. The instance of concepts also uses the name of management domain as the subscript, showing the concept instances belonging to different management domains.

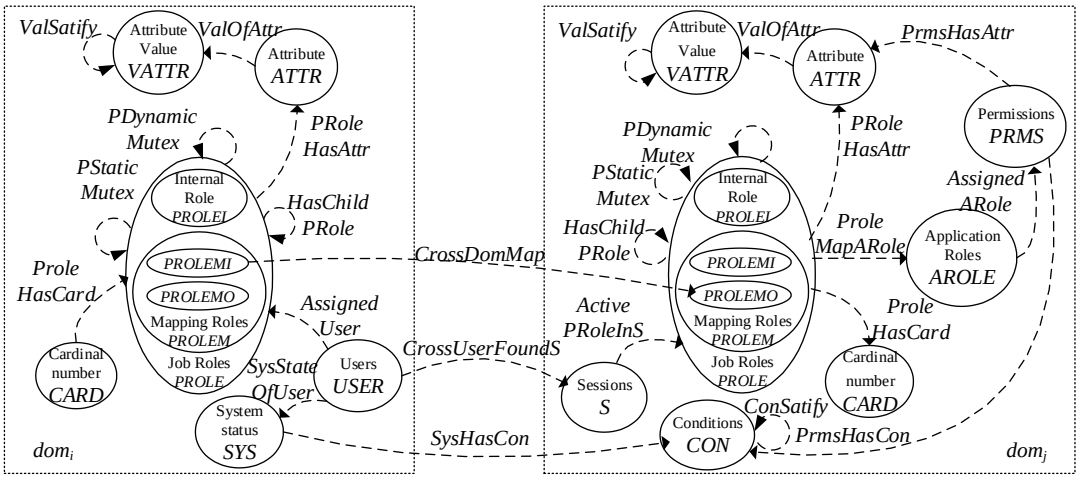


Figure 2. Cross-Domain Authorization Management Model for Multi-Levels Hybrid Cloud Computing

Definition 2: Intra-domain atomic relations

Intra-domain atomic relations are mainly includes:

(1) *PStaticMutex*, *PDynamicMutex* are static and dynamic mutually exclusive relationship between position roles respectively. *AStaticMutex, ADynamicMutex* are static and dynamic mutually exclusive relationship between application roles respectively.
(2) *HasChildPRole*, *HasChildARole*, *HasChildPrms* are hierarchy relationship of position roles, application roles and relationship between father and son of permission respectively.
(3) *AssignedUser* means user-position role assignment. *AssignedARole* is the direct assignment of application role to permission, which only includes explicit permission of the application role.
(4) *PRoleMapARole* is the mapping relationship of position role and application role. Only when both are role nodes, the relationship exists from position role to application role.
(5) *UserHasAttr, PRoleHasAttr*, *PrmsHasAttr* means that the user, position role and permission have some attributes respectively.
(6) *ValOfAttr* means the tuple relationship between attribute and attribute values. *ValSatify* means the satisfied relationship among attribute values.
(7) *SysHasCon, PrmsHasCon* represent the real-time condition of current state and the condition satisfied when using permissions respectively. *ConSatify* is the satisfied relationship among conditions.
(8)) *UserFoundS* is the relationship between user and session, where a user can creates multiple sessions, while a session can map to only one user. Besides, only one position role can be activated in a session.
(9) *ActivePRoleInS* is activated position role in a given session.

Definition 3: Cross-domain atom relations

Cross-domain authorization mainly includes the following atom relations:

(1) *CrossDomMap* is cross-domain role mapping relationship, which maps In-role in the initial domain to Out-role in the target domain. It makes the In-role in the initial domain to obtain the permission of Out-role in the target domain. This relation is not transitive and symmetry. It is unidirectional relation mentioned above.

(2) $CrossUserFoundS$ is the relationship from user in the initial domain to the session in the target domain.

(3) $CrossPStaticMutex$, $CrossPDynamicMutex$ are cross-domain static and dynamic mutually exclusive relationship of In-role in the initial domain. It is the extended relationship of static and dynamic mutually exclusive relationship of Out-role in the target domain.

Definition 4: Complex relations

Based on definition 7 and definition 8, complex relations can be composed. The following are some examples.

(1) $UserHasPrms \equiv AssignedUser \circ PRoleMapARole \circ ARoleHasPrms$

(2) $PRoleHasPrms \equiv PRoleMapARole \circ ARoleHasPrms$

(3) $PRoleHasPrms \equiv PRoleMapARole \circ ARoleHasPrms$

(4) $ActivePrmsInS \equiv ActivePRoleInS \circ PRoleMapARole \circ ARoleHasPrms$

Due to space limitation, only parts of relations are presented in definition 2,definition 3, definition 4 and Figure 2.

Definition 5: Cross-domain permission management operations

Due to space limitation, only some key operations referred to cross-domain authorization are presented. It's assumed that i , j means the name of management domain, and $i \neq j$.

There are several operations in the model, such as assigning position role to a user, establishing cross-domain unidirectional role mapping, establishing cross-domain session of access resource and so on. Take the third operation above as an example, which is defined as follows.

Operation: Establishing cross-domain session of access resource

$$
\begin{aligned}
&Create\_CrossSession(user_i,\ s_j,\ prole_j)\\
&\equiv (USER(user_i) \wedge PROLEMO(prole_{dom_j}) \wedge UserHasCrossPRole(user_i,\ prole_{dom_j}))?;\\
&\quad \neg(PDynamicMutex(prole_{dom_j}, \exists UserActiveCrossPRole^{-}.user_i)?;\\
&\quad \neg(ADynamicMutex(\exists PRoleMapARole^{-}.prole_j, \exists UserActiveARole^{-}.user_i))?;\\
&\quad ConSatify(\exists UserHasCon^{-}.user_i, \exists PrmsHasCon^{-}.(\exists PRoleHasPrms^{-}.prole_j))?;\\
&\quad ValSatify(\exists ValOfAttr.(\exists UserHasAttr^{-}.user_i),\\
&\qquad (\exists ValOfAttr.(\exists PrmsHasAttr^{-}.(\exists AssignedARole^{-}.(\exists PRoleMapARole^{-}.prole_j)))))?;\\
&\quad (\{\neg S(s_j), \neg UserFoundS(user_i, s_j), \neg ActivePRoleInS(s_j, prole_j)\},\\
&\qquad \{S(s_j), UserFoundS(user_i, s_j), ActivePRoleInS(s_j, prole_j),\\
&\qquad ActiveARoleInS(s_j, PRoleMapARole^{-}.prole_j),\\
&\qquad ActivePrmsInS(s_j, \exists AssignedARole^{-}.(\exists PRoleMapARole^{-}.prole_j)),\\
&\qquad UserActiveCrossPRole(user_i, prole_j),\\
&\qquad UserActiveCrossARole(user_i, \exists PRoleMapARole^{-}.prole_j),\\
&\qquad UserActiveCrossPrms(user_i, \exists PRoleMapARole^{-}.prole_j)\});\\
&\quad UpdateValOfUserAttr(user_i))
\end{aligned}
$$

The steps are as follows: ① check whether $prole_j$ is assigned to $user_i$ through cross-domain operations; ② check whether $prole_j$ is not dynamic mutually exclusive with user's active position role in the target domain j ; ③ check whether the application role mapped from $prole_j$ is not dynamic mutually exclusive with user's active application role

in the target domain; ④ check whether the current state of users satisfy the conditions of access permissions; ⑤ check whether the attribute values of users in this session are satisfied with the conditions of access permissions; ⑥ create session for the user $user_i$, and activate the corresponding position roles, application roles and permissions; ⑦ update the user's attributes.

5. Security Analysis of the Model

According to security and autonomy principles of secure interoperation [16], security in cross-domain authorization management should abide by the following principles:

Principle C1: the cross-domain role mapping shouldn't affect the intra-domain permissions of users.

Principle C2: the static mutually constraints won't be violated because of the cross-domain role mapping between the role in the initial domain and that in the target domain.

Principle C3: the dynamic mutually constraints won't be violated because of the cross-domain role mapping.

The method to ensure the satisfication of above principles is to check whether the static or dynamic duty of separation may be violated because of the cross-domain role mapping. In other words, the model should ensure that the following formulas mustn't be satisfied during the cross-domain unidirectional role mapping.

f1: $PROLE(prole_i) \wedge PRMS(prms_i) \wedge PRoleHasCrossPrms(prole_i, prms_i)$

f2: $PStaticMutex(prole_j, prole_j') \wedge UserHasCrossPRole(user_i, prole_j) \wedge UserHasCrossPRole(user_i, prole_j')$

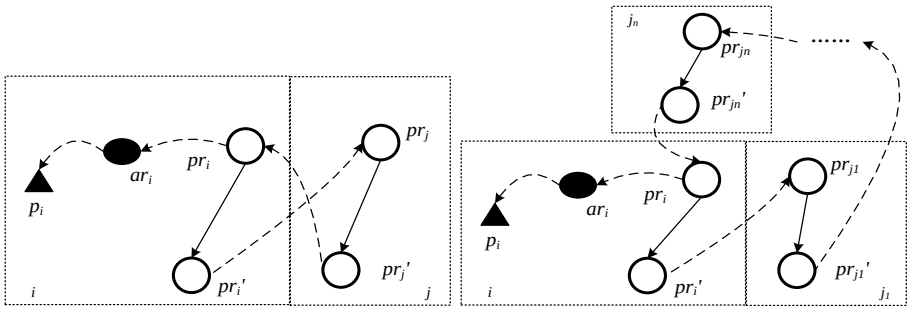
f3: $PDynamicMutex(prole_j, prole_j') \wedge UserActiveCrossPRole(user_i, prole_j) \wedge UserActiveCrossPRole(user_i, prole_j')$

Take formula f1 as an example, and the process of checking its satisfaction is shown as follows.

Proof: the formula f1 can't be satisfied.

The formula f1 mainly checks whether the permission of the position roles in the local domain will increase because of cross-domain mapping. The main idea is that, suppose the position role $prole_i$ didn't own the permission $prms_i$ in the internal domain, and then check whether the formula f1 is satisfied after the role mapping.

After role mapping, the reason why there are changes on the permissions of the position role in the initial domain is that, the cross-domain role mapping and role hierarchy form a circle of permission flow, bringing the implicit promotion of privilege of the position roles in the initial domain. Because the cross-domain role mapping operation $Creat\_CrossMap$ requires that the start point of the mapping should be In-role and the end point should be Out-role, the circle relies on both cross-domain role mapping and role hierarchy in the intra-domain. Figure 3 illustrates that cross-domain role mapping and role hierarchy, construst a mapping chain from the start point in the initial domain to itself, where Figure 3(a) is an example between two domains and Figure 3(b) among multiple domains.



(a)role mapping between two domain (b) role mapping among multi-domain

Figure 3. Cross-Domain Role Mapping and Role Hierarchy Forming Circle of Permission Flow

Take Figure 3(a) as an example to prove the satisfaction of formula f1. First, suppose that the role pr_i' in domain i is related to its senior role pr_i through the cross-domain role mapping $CrossDomMap(pr_i', pr_j)$, $CrossDomMap(pr_j', pr_i)$, the role hierarchy $HasChildPRole(pr_i, pr_i')$ between role pr_j and pr_j'. Then, check whether the permissions of role pr_i' will increase through the mapping chain $pr_i' \mapsto pr_j \mapsto pr_j'$, obtaining the permissions of its senior role pr_i.

The state of Figure 3(a) can be expressed by the following formulas:

$A = \{ PROLE(pr_i)$, $PROLE(pr_i')$, $AROLE(ar_i)$, $PRMS(p_i)$, $PROLE(pr_{j1})$,
$PROLE(pr_{j1}')$, $HasChildPRole(pr_i, pr_i')$, $HasChildPRole(pr_j, pr_j')$,
$PRoleHasPrms(pr_i, p_i)$, $\neg PRoleHasPrms(pr_i', p_i)$, $CrossDomMap(pr_i', pr_j)$,
$CrossDomMap(pr_j', pr_i) \}$

Based on $CrossDomMap(pr_i', pr_j)$, $CrossDomMap(pr_j', pr_i)$ and $HasChildPRole(pr_i, pr_i')$, though role pr_i' is associated with pr_i through operation $Creat\_CrossMap$, there aren't permissions transmitting in the mapping chain $pr_i' \mapsto pr_j \mapsto pr_j'$ because of the following reasons.

- Because cross-domain role mapping is unidirectional, pr_j can't gain excessive permissions of outer domain, and can only transmit permission in domain j to pr_i'. Therefore, it can't transmit the permissions of any outer domain, including the initial domain, to role pr_i'.
- According to the definition of the relation between position roles, there is no permission inheritance relation between a position role and its subordinate role. So role pr_j can't inherit the permissions of pr_j'.

Based on the analysis above, the cross-domain unidirectional role mapping won't make the permissions of position role in the intra domain increase.

The conclusion is effective for multiple domains as Figure 3(b) as well. Based on the discussion above, it's easy to know that formula f1 is not tenable and the principle C1 is satisfied.

The formula f2 and formula f3 can be proved in the same way.

6. An Example of Cross-Domain Authorization

An example is present to illustrate the application of the cross-domain authorization management model for multi-level hybrid cloud computing.

Suppose that a school *uni* cooperates with a company *com* to do some research. During the cooperation, there are two position roles named $administrator_{com}$ and $developer_{com}$ in the company. For better resource sharing, cooperation and secure resource access, a new role named as $parner_{uni}$ is introduced into the position role set of the school, used to map the roles of outer domain.

The position role $parner_{uni}$ is mapped to application role $user_{uni}$ in *uni* through the relation $PRolrMapARole(parner_{uni}, user_{uni})$. And the position role $developer_{com}$ in *com* is mapped to $parner_{uni}$ in the school through cross-domain mapping relations $CrossDomMap(developer_{com}, parner_{uni})$.

Suppose $zhang_{com}$ is a user in the company, who was assigned role $developer_{com}$, and then obtained the outer-domain role $parner_{uni}$ through cross-domain role mapping. The example is shown as Figure 4.

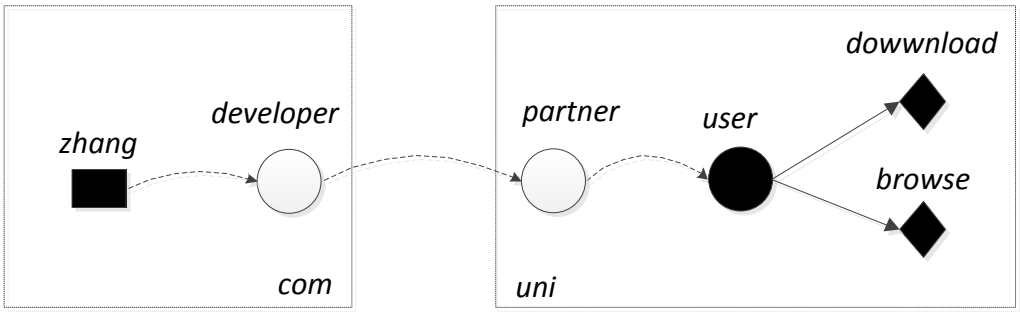


Figure 4. An Example of Cross Domain Role Mapping and Authorization

The process of cross-domain access by user $zhang_{com}$ is presented as follows:

$$
\begin{aligned}
&Create\_CrossSession(zhang_{com}, s_{uni}, partner_{uni}) \\
&\quad \equiv (USER(zhang_{com}) \wedge PROLER(partner_{uni}) \wedge UserHasCrossPRole(zhang_{com}, partner_{uni}))?; \\
&\quad \neg(PDynamicMutex(zhang_{com}, \exists UserActiveCrossPRole^{-}.zhang_{com}))?; \\
&\quad \neg(ADynamicMutex(\exists PRoleMapARole^{-}.partner_{uni}, \exists UserActiveARole^{-}.zhang_{com}))?; \\
&\quad ConSatify(\exists UserHasCon^{-}.zhang_{com}, \exists Prms HasCon^{-}.download_{uni})?; \\
&\quad ValSatify(\exists ValOfAttr.(\exists UserHasAttr^{-}.zhang_{com}), \\
&\qquad (\exists ValOfAttr.(\exists Prms HasAttr^{-}.(\exists AssignedARole^{-}.(\exists PRoleMapARole^{-}.?partner_{uni}))))?; \\
&\quad (\{\neg S(s_{uni}), \neg UserFoundS(zhang_{com}, s_{uni}), \neg ActivePRoleInS(s_{uni}, partner_{uni})\}, \\
&\qquad \{S(s_{uni}), UserFoundS(zhang_{com}, s_{uni}), ActivePRoleInS?s_{uni}(partner_{uni})? \\
&\qquad ActiveARoleInS(s_{uni}, PRoleMapARole^{-}.partner_{uni})?, \\
&\qquad ActivePrmsInS?(s_{uni}, \exists AssignedARole^{-}.(\exists PRoleMapARole^{-}.?partner_{uni})), \\
&\qquad UserActiveCrossPRole(zhang_{com}, partner_{uni})?, \\
&\qquad UserActiveCrossARole(zhang_{com}, \exists PRoleMapARole^{-}.?partner_{uni}), \\
&\qquad UserActiveCrossPrms(zhang_{com}, \exists PRoleMapARole^{-}.?partner_{uni}) \\
&\quad UpdateValOfUserAttr(zhang_{com})
\end{aligned}
$$

Based on the current state, the attribute of the user $zhang_{com}$ can satisfy the condition of executing the operation $Create\_CrossSession(zhang_{com}, s_{uni}, partner_{uni})$. So the request is allowed and $zhang_{com}$ can access the requested resources.

From the example above, it is not difficult to conclude that the cross-domain authorization management model for multi-levels hybrid cloud computing is effective and feasible in resolving the authorization of cross-domain environment.

7. Conclusion

Due to the security problems of the cross-cloud, cross-level and cross-domain in multi-levels hybrid cloud computing, the singleness of the role establishment method, the implicit promotion of privilege and the separation of duties conflict, a cross-domain authorization management model for multi-levels hybrid cloud computing was proposed. Based on the role splitting, a novel two-tier role architecture of position role and application role was introduced into the model to satisfy the practical needs of organization level and application level at the same time. Unidirectional role mapping was presented to realize the secure cross-domain authorization and interoperation. Attributes, conditions and other dynamic elements are introduced to provide dynamic and fine-grained authorization with better adaptability. The dynamic description logic was used to describe and characterize the model. The security of the model was analyzed with the reasoning function of dynamic description logic, showing that the model satisfies the principle of autonomy and security. Finally, the feasibility, practicality and effectiveness of the model are illustrated through an example of cross-domain authorization.

References

[1] A. Kapadia, J. Al-Muhtadi , CAMPBELL D, *et al*. "IRBAC 2000:Secure Interoperability Using Dynamic Role Translation[R]". Chicago: University of Illinois, **(2000)**.
[2] E. Freudenthal, T. Pesin, L. Port , " dRBAC: Distributed role-based access control for dynamic coalition environment[C]". In Proceedings of the 22nd International Conference on Distributed Computing Systems", Piscataway, NJ: IEEE Press, **(2002)**, pp. 411-420.
[3] B. Shafiq , J B D Joshi, E. Benino, " Secure interoperation in a multi domain environment employing RBAC policies[J]", IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 11, **(2005)** pp. 1557-1577.
[4] S. Mohamed, B. Elisa, G. Arif, "SERAT: secure role mapping technique for decentralized secure interoperability[C]", In Proceedings of ACM Symposium on Access Control Models and Technologies. New York: ACM Press, **(2005)**, pp. 159-167.
[5] L. Ruixuan, H. Jingwei, T. Zhuo. "R2BAC:a risk-based multi-domain secure interoperation model [J]" Journal of Communications, vol. 29, no. 10, **(2008)**, pp. 58-69.
[6] Z. Yang, L. Yang, X-yang Luo, *et al*. "Model of Domain based RBAC and Supporting Technologies[J]", JOURNAL OF COMPUTERS, vol. 8, no. 5, **(2013)**, pp. 1220-1229.
[7] E. Freudenthal, T. Pesin, *et al*. "dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments[C]", Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS'02). IEEE Computer Society, **(2002)**.
[8] L. Junguo, H. Fan, Y. Qiuwei *et.al*., "Toward Security Analysis of the dRBAC Model[J]. Journal of Chinese Computer Systems vol. 28, no. 7, **(2007)**, pp. 1177- 1180.
[9] G. Dongheng. "Research on multi-domain security interoperation[D]", Chong Qing: Chong Qing University Master Degree Dissertion.**(2013)**.
[10] Y. Chunxiao, G. Dongheng, "Research on secure interoperation in multi-domain environment[J]" Journal of Computer Applications vol. 32, no. 12, **(2012)**, pp. 3422-3425.
[11] S. Zhongzhi, D. Minkai, J. Yuncheng, " Logic Basis of Semantic Web [J]", Science in China, vol. 34, no. 10, **(2004)**, pp. 1123-1138.
[12] C. Liang, S. Zhongzhi, C. Limin, "Fmaily of Extended Dynamic Description Logics[J]",Journal of Software, vol. 21, no. 01,**(2010)**, pp. 1-13.
[13] C. Liang, S. Zhongzhi, Q. Lirong, "A Tableau Decision Algorithm for Dynamic Description Logic[J]", Chinese Journal of Computers, vol. 31, no. 06, **(2008)**, pp. 896-909.
[14] C. Liang, C. Limin, "Action Theory Based on the Dynamic Description Logic DDL[D]". Computer Science, vol. 38, no. 7, **(2011)**, pp. 203-208.
[15] R. Zhiyu, C. Xingyuan, S. Dibin, "Cross-domain authorization management model based on two-tier role mapping", Joumal of Computer Applications, vol. 33, no. 9, **(2013)**, pp. 2511-2515.
[16] L. Gong and X. Qian, "Computational issues in secure interoperation[C]", IEEE Transactions on Software and Engineering, vol. 22, no. 1, **(1996)**, pp. 43-52.

