CTCDM: A Complete Time Constraint Based Delegation Model

Chunyan Han, Lianzhong Liu, and Zhouyang Li

***Abstract*—Time-based delegation transfers privileges between users and reduce the work load of administrators. But there are defectives on the researches of time-based delegation model: incomplete time constraints and name space explosion problem of delegation model. To solve these two problems, following researches has been done: firstly, this paper proposes a time constraint policy including time representation and the time state change algorithm; secondly, based on the time constraint policy, this paper puts forward the Complete Time Constraint Based Delegation Model(CTCDM) to solve the name explosion problem from a certain extent.**

***Index Terms*—Time constraint, delegation, model.**

I. Introduction

During the past years, RBAC [1] has gained its popularity by achieving a logical separation of users and permissions. Meanwhile, RBAC also has the ability to describe constraints, including user constraints, the role of constraints. These above two advantages make RBAC being widely used in many systems. Although this model has attained a considerable level of maturity, RBAC model is not able to satisfy these two application requirements:

1) Time constraints. In real world, permissions of access control are often changed with time, for example, many companies add time limits for temporary workers to access some specified systems.
2) Authorization with flexibility. For example, Professor Zhang will be away on official business, but during this period, he has to review exam papers on the Internet. One feasible way for him is to delegation his permission of reviewing exam papers to his assistant. You can imagine that the administrator does all the authorization under the conditions like the above. So delegation is important for authorization with flexibility. This kind of delegation belongs to the user-user delegation. In addition to user-user delegation, there is a kind of delegation named role-role delegation. A role-to-role delegation is expected in many cases. For example, N staffs with the same role R of a department all need a permission P to do a specified task. Without role-role delegation, it needs N times authorization. Now with the role-role delegation, the permission can be delegated to the role R and thus N staffs all have the permission P.

On the grounds of those above two reasons, this paper proposes a time-based delegation model in this paper.

Manuscript received August 16, 2014; revised October 8, 2014. The research was sponsored by Co-Funding Project of Beijing Municipal Education Commission under Grant No. JD100060630.

The authors are with the Beijing Key Laboratory of Network Technology, School of Computer Science and Engineering, Beihang University, China (e-mail: ChunyanHan425@yeah.net, lz_liu@buaa.edu.cn, myshell@163.com).

II. Related Work

A. Time Constraints Representation

In terms of time constraints, there are two mainly methods: discrete time point representation [2] and period time representation [3].

Discrete time point representation proposed by Hansi Qin is an extension of RCL2000 Rules [4]. G. J. Ahn put forward a method to express constraints with formalization. RCL2000 does not have the characteristic of time and Hansi Qin has extent the RCL2000 rules by adding time characteristic. The formal definition of discrete time point representation is:

The mapping function and definition of virtual to real time:

$$\forall i,j \in N, \forall t_j, t_j \in TE, t_i \prec t_j \leftrightarrow real\_time(t_i) \prec real\_time(t_j) \tag{1}$$

$$\forall i,j,k \in N, \forall t_i, t_j, t_k \in TE, t_k = t_i + t_j \leftrightarrow real\_time(t_k) = real\_time(t_i) + real\_time(t_j) \tag{2}$$

$$\forall i,j,k \in N, \forall t_i, t_j, t_k \in TE, t_k = t_i - t_j \leftrightarrow real\_time(t_k) = real\_time(t_i) - real\_time(t_j) \tag{3}$$

Assuming $\Delta t = real\_time(t_i) - real\_time(t_{i+1})$ is a constant. When time series T, real_time, Δt is determined, virtual time series can be mapped to the real time correctly.

Definition of time range: $TR = \{(t_i, t_j) | t_i, t_j \in T, i < j\}$. Time range is composed of two time points. $TRS = 2^{TR}$ is a set of TR.

Periodic time representation is proposed by Elisa Bertino. The basic idea for periodic time representation is to detect the periodic time with the constraint of the expression $\langle[BeginTime, EndTime], P\rangle$. In this expression, $BeginTime$ stands for begin time, $EndTime$ stands for end time and the P stands for periodic expression. In this method, once the role triggered event is started, the role trigger can be executed immediately or be deferred to execute. The basis of periodic time representation is the concept of calendar. A calendar is defined as a continuous interval of countable set and each time interval can be indexed by integer. Given two calendars c_1 and c_2. If every time interval in c_2 can be covered by every time interval in c_1, then c_1 is the sub calendar of c_2 expressed as $c_1 \subseteq c_2$.

B. Delegation Model

In terms of delegation models, most delegation models are based on the PBDM [5] portfolio which contains three models: PBDM0. PBDM1 and PBDM2. They are all permission level delegation which means delegation unit is permission not role. Permission level delegation satisfies the Principle of Least Privilege [6]. PBDM0 model extends RBAC96 model by including user-user delegation. Delegation user has two types of roles: regular roles and delegation roles. The mechanism of PBDM0 is that delegation user creates a delegation role first and then assigns roles or permissions of regular roles to delegation role. At last, delegation user delegates the DTR to the delegated user. Based on the PBDM0, PBDM1 adds a type of role to the model: delegable role to restraint the behavior of the delegation users. In PBDM0, delegation user can delegate any role or any permission which may cause security threat to the system. In PBDM1, user can only delegate permissions or roles of delegable role. As to PBDM2 model, it is a role-role delegation model. There are four types of roles in PBDM2: regular roles, fixed delegable roles, temporary delegable roles and delegation roles. The mechanism of PBDM2 is: 1) delegation user first creates a delegation role. 2) delegation user assigns the roles or permissions of fixed delegable roles to the delegation role. 3) delegation user delegates the delegation role to the temporary delegable role which belongs to the delegated user.

In recent years, there are many delegation models to achieve the specified goals or to enhance the previous delegation model. Reference [7] explores a secure delegation scheme that could keep access control information hidden through network transmission. Reference [8] proposes an improved delegation model where the various users in a delegation chain may perform supervision on the delegate to exercise the delegated permission. Reference [9] presents both static and dynamic delegation in the context of the Role Graph Model. Reference [10] enables a role member to create delegations based on the dynamic needs of collaboration; in the meantime, a delegation chain can be verified by anyone without the participation of role administrators.

C. Summary of Related Work

Below is the summary of the time constraints and delegation models mentioned above:

1) Discrete time point applies to the time constraints of simple expression. Although complex time constraints can be expressed, it is not intuitive. Period representation is suitable for the periodic time constraint, but still unable to express some periodic time constraints such as every other day of year 2008.
2) Although PBDM0, PBDM1 and PBDM2 are permission level delegation with flexibility, it has the name space explosion problem: when initiate a delegation, delegation role needs to be created. Besides, there may be tremendous delegable roles. Management of delegation roles and delegatables is a tedious for delegation user.
3) The recent delegation models do not solve the problems mentioned in (2).

So this paper will put emphasis on the solution of these two problems.

III. Complete Time Constraint Based Delegation Model

A. Complete Time Constraints

Representation of time constraint in this paper is the combination of discrete time representation and periodic time representation. Meanwhile, this paper also add limit of times and limit of total time constraints on it and periodic time representation is also modified in order to support more semantics.

Under the premise of the concept $Calendar$, time constraint in this paper is in the format of four-tuple as following. This paper will give the explanation of every items of this tuple in Table I.

$$tc = \langle [t_s, t_e], \Delta T_0, N, P' \rangle \tag{4}$$

TABLE I: Meaning of Each Element in Time Constraint

Item	Meaning
$[t_s, t_e]$	$[t_s, t_e]$ means time interval. t_s is the begin time and t_e is the end time ($t_s \geq t_e$). Meanwhile, the value of t_s or t_e can be ∞ which means start time or end time is arbitrary.
ΔT_0	ΔT_0means the total time limit in the time interval $[t_s, t_e]$ and it is optional. When ΔT_0 is null, it indicates that there is no total time limit on time constraint.
N	N means the max times of using a specified permission.
P'	P' means the periodic time and this paper use $\sum_{i=1}^{n} O_i \cdot C_i \rhd interval \cdot C_i$ to express it. Below are the explanations of symbol $\rhd$,O_i,C_i and $interval$ (1) The symbol $\rhd$ separates the first part of the expression, identifying the set of starting points of the intervals it represents, from the specification of the duration of each interval in terms of calendar. (2) C_i is time unit such as year, month, week and so on. (3) O_i is the enumeration of the available time in a specified time unit. There are two representation of this element: a. enumerates all the available time in a specified time unit. b. only enumerates the start time of a specified time unit and use $interval$ to describe the periodicity. (4) $interval$ is the interval of specified time unit and it is related to O_i. When using (3) a to describe O_i, $interval$ becomes meaningless and the value of it is -1. Only when using (3) b to describe O_i, $interval$ can work.

In order to further explain the meaning of each item in time constraint expression, this paper use two examples to make it clear.

Example 1: a school stipulates that time for 08 undergraduates logging in system to select courses is September and December from 2008 and 2012. Obviously, the privilege is periodic for students, so the expression is:

$$\langle [2008{,}2012], null, null, \{2008{,}2009{,}2010{,}2011{,}2012\} \cdot years \rhd 4 \cdot year + \{9{,}12\} \cdot months \rhd 2 \cdot month \rangle$$

In terms of the expression of P', following expression can also be used:

$$\{2008\} \cdot year \rhd 1 \cdot year + \{9{,}12\} \cdot month \rhd (-1) \cdot month$$

Example 2: a company stipulates that temporary workers can only log in the system from 2012 to 2015, max total time online for temporary workers 15 hours, moreover, max times of accessing the system for these workers is 10 and they can't be online for more than 2 hours every time. The expression of time constraint is:

$$\langle [2012,2015], 15, \left(10,2, \sum_{i=1}^{10} \Delta T_i\right), null\rangle$$

Theoretically, when time changes, the system should check that whether sessions with time constraints are valid. But the downside to it is obvious, especially when constraints are tremendous and the computation granularity is small, the efficiency of the system is difficult to guarantee. This paper introduces riggers assisted with timers and counters to reduce dependence on real-time monitoring.

Triggers are used in multiple disciplines. Database triggers are a kind of method provided to programmers and data analyst to ensure data integrity and it is a special procedure associated with table events. The execution of it is not by the program calls, nor the manual start, but by the event to trigger, such as when table operations like insert, delete and update.

Triggers in this paper and database triggers have a similarity: they are both triggered by event. There are two events of this trigger: user starts performing an action with time constraint; (2) user finishes an action with time constraint. For each permission with time constraint, there is at least one corresponding trigger. If time constraint defines the total time limit, then there will be a timer to record the time of each operation; If time constraint defines the times, then there will be a counter to record operation frequency.

Three states for triggers are defined: ready, active and invalid. This paper uses t_c to represent current time, t_s to represent start time and t_e to represent end time.

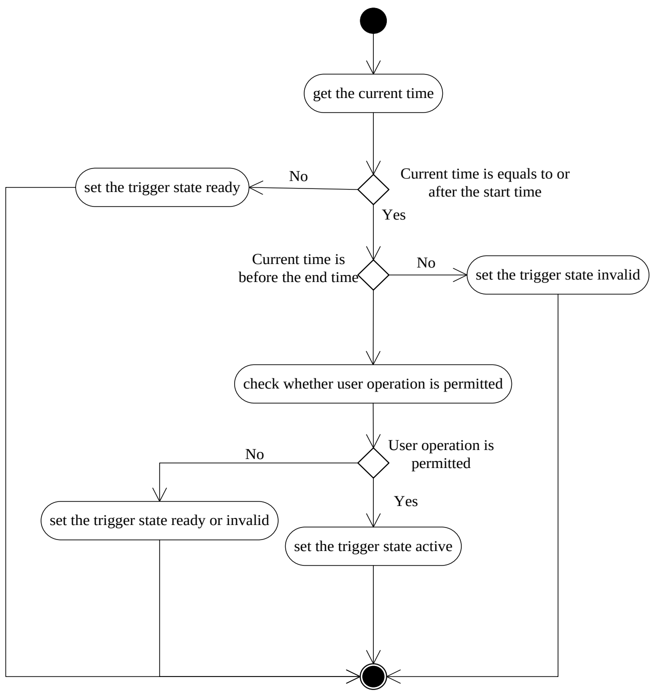
get the current time
Current time is equals to or after the start time
No
set the trigger state ready
Yes
Current time is before the end time
No
set the trigger state invalid
check whether user operation is permitted
User operation is permitted
No
set the trigger state ready or invalid
Yes
set the trigger state active

Fig. 1. Trigerstate change algorithm.

When authorization with time constraints is set up, the system will create a trigger for the authorization and then get the current time t_c. The system will set an initial state for the trigger according to the current time t_c.Every time a user to perform an operation with time constraint the trigger is triggered. When the state of the trigger condition is active, the user can perform the corresponding operations. Activity diagram of state change of trigger is shown in Fig. 1.

In Fig. 2, checking whether user operation is permitted is under the precondition that $t_s \preccurlyeq t_c \preccurlyeq t_e$. t_c satisfying the condition $t_s \preccurlyeq t_c \preccurlyeq t_e$ does not indicate that user operation is permitted. For example, a school stipulates that time for 08 undergraduates logging in system to select courses is September and December from 2008 and 2012.So when months except September and December from 2008 to 2009 is not available for 08 undergraduates to select courses. In that period time, the state of the trigger is ready. Algorithm for checking whether user operation is permitted in this further circumstance is in Table II.

TABLE II: TIME STATE CHANGE ALGORITHM

```
Input: current time $t_c$, time constraint $tc$,counter $counter$, timer $timer$
Output: state of the trigger
Procedure:
/*total time limit is defined in the time constraint*/
if($ct.\Delta T_0$ isnotnull)
/* get the total operation time before */
$\Delta T =$ timer.$value$
if($\Delta T \geq \Delta T_0$ ) setTriggerState(INVALID);
else set Trigger State(ACTIVE);
/* times limit is defined in the time constraint */
if(ct.$Nisnotnull \ \cap trigger.state != INVALID$)
 /* get the total operation times before */
$n = counter.value$;
if($n \geq N$) setTriggerState(INVALID);
else  set Trigger State(ACTIVE);
/* period time is defined in the time constraint*/
if(Pisnotnull)
for each $O_i \cdot C_i \rhd interval \cdot C_i$ in $P$
if($interval < 1$ )
/*Below operation will get the value of current time under a specified time unit. For example, if the current time is 2014-6-5， then the corresponding time under time unit year is 2014*/
$O_i^{'}$ = getSpecTimeFromCurrentTime($t_c, C_i$);
 if($O_i^{'} in O_i$)  setTriggerState(ACTIVE);
else  set Trigger State(READY);
else if (interval $\geq$ 1)
/*Below operation will calculate the time interval between current time and start time under a specified time unit. For example, when periodic time expression is $\{2008\} \cdot year \rhd 1 \cdot year + \{9,12\} \cdot month \rhd (-1) \cdot month$ and the current time is 2014-6-5，then the interval of time unit year is 6（2014-2008）*/
$totalInterval$= get Total Interval Of Spec Time Unit($t_c, O_i, C_i$);
if(total Interval modinterval equals 0 ) set Trigger State(ACTIVE);
else set Trigger State(READY);
```

B. Delegation Model

In the last section this paper describes the time constraints of delegation. In this section this paper will put forward the Complete Time Constraint Based Delegation Model (CTCDM).

CTCDM contains two parts: user-user delegation and role-role delegation. User-user delegation is the improvement of PBDM1 and it can solve the name space explosion problem of PBDM1 and PBDM2 from a certain extent. The solution of name space explosion problem relies on the time constraints of this model. Role-role delegation is an improvement of PBDM2 model.

In order to solve name space explosion problem caused by the agent role, this paper removes the delegatable roles in PBDM1, Roles are divided into common roles(CR) and delegation roles(DTR). For each permission in each CR, it contains three variables: CPDF (Can Permission Delegated Flag), HDF (Has Delegated Flag) and SDC (Steps of Delegation Chain). Table III will illustrate these three variables.

TABLE III: VARIABLES IN PERMISSION

Variables	Range	Explanations
CPDF	$\{0,1\}$	This variable means whether the permission can be delegated. When CPDF is 0, this permission can not be delegated to anyone. When CPDF is 1, this permission can be delegated.
HDF	$\{0,1\}$	This variable indicates that whether the permission has been delegated. When HDF is 1, it means the permission has been delegated or otherwise when HDF is 0. In addition, this variable is related to variable CPDF. When CPDF is 0, the value of HDF is always 0. When CPDF is 1, the value of HDF can be 0 or 1.
SDC	$(x\|x \in \mathbb{N})$	Steps of delegation chain. Take an example to explain this variable. There is a permission p_1 and its owner is u_1.u_1 has delegated p_1 to u_2. Then u_2 delegates $p_1$to u_3. For u_1, SDC is 0. For u_2, SDC is 1. For u_3, SDC is 2.

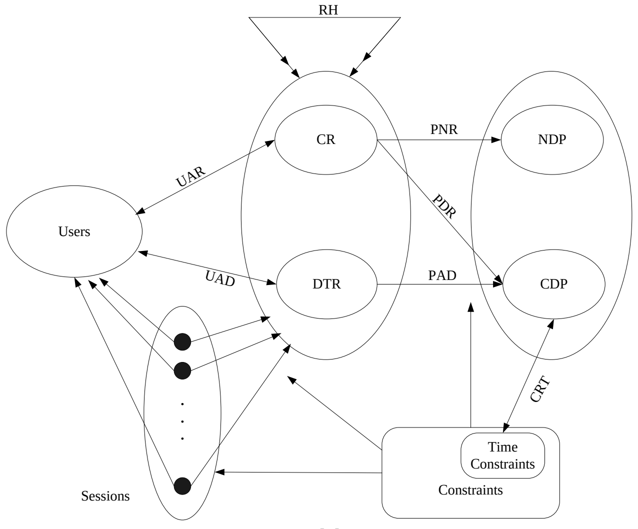

Fig. 2. User-user delegation in CTCDM.

Users can only delegate permissions whose CPDF value is 1. When a user does a delegation operation, he or she only needs to create DTR in the first time. DTR is a set of all delegated permissions of one user. Each element in DTR can be expressed like below:

$$\langle P, DU, TC\rangle \tag{5}$$

P refers to the permission that has been delegated. DU means the delegated user and TC means the time constraint of this delegation operation. When the permissions and delegated users are chosen and a user finishes the operation of delegation, $\langle P, DU, TC\rangle$ will be automatically saved in DTR. $\langle P, DU, TC\rangle$ is visible for delegated user. When user wants to withdraw the delegated permissions or the delegated permissions is due, related $\langle P, DU, TC\rangle$ will be erased from DTR. Fig. 2 is the user-user delegation model.

Below is the sets and functions of user-user delegation:

Sets:Users, CR, UAR, DTR, UAD, $Sessions$, NDP, PNR, CDP, PDR ,PAD, RH, $Constraints$, $TimeConstraints$, CRT.

1) $Users$: User set containing delegation users and delegated users.
2) CR: common roles containing all the roles a user holds.
3) $UAR \subseteq Users \times CR$: many to many mapping relationship of $Users$ and CR.
4) DTR:delegated roles.
5) $UAD \subseteq Users \times DTR$: one to one mapping relationship of $Users$and DTR.
6) $Sessions$: this set has the same meaning with the $Sessions$ of RBAC.
7) NDP : permissions that can not be delegated to anyone(No delegate Permission).
8) $PNR \subseteq CR \times NDP$: many to many mapping relationship of CR and NDP.
9) CDP: permissions that can be delegated（Can delegate Permission）.$CDP \cap NDP = \emptyset$.
10) PDR ⊆ CR × CDP: many to many mapping relationship of CR and CDP.
11) PAD ⊆ DTR × CDP : many to many mapping relationship of DTR and CDP..
12) RH: role hierarchy and its meaning is the same with RH in RBAC.
13) Constraints: constraints of CTDM. Constraints can take effect on UAD or PAD. This paper will describe these constraints in detail in the next section.
14) Time Constraints ⊆ Constriants: time constraints of CTDM which is a subset of Constriants.
15) CRT ⊆ CDP × Time Constraints: many to one mapping relation of CDP and Time Constraints.

Functions of this model are:

1) $Users(r,t) : DTR \rightarrow 2^{Users}$, a function mapping a delegated role to a set of users which are assigned to this role at a specified time point t.
2) $Users\_O(r,t)$: $DTR \rightarrow 2^{Users}$, a function which returns all the users who are the original users of a specified delegated role at a specified time point t.
3) $Users\_D(r,t)$: $DTR \rightarrow 2^{Users}$, a function which returns all the users owning a specified DTR through a delegation at a specified time point t.
4) $User(s_i,t) : Sessions \rightarrow Users$, a function which returns all the users belonging to session s_i at a specified time point t.

5) $Roles(s_i, t)$: $Sessions \rightarrow 2^{CR}$, a function which returns all the roles belonging to session s_i at a specified time point t.
6) $Permissions(s_i, t)$:$Sessions \rightarrow 2^{NDP+CDP}$, a function which returns all the permissions belonging to session s_i at a specified time point t.
7) $role\_u(u, t)$:$Users \rightarrow 2^{CR}$, a function which returns all the roles user u holds at a specified time point t.
8) $canDelegatePermission\_u(u, t)$: $Users \rightarrow 2^{CDP}$, a function which returns all permissions that can be delegated user u holds at a specified time point t.
9) $noDelegatePermission\_u(u, t)$: $Users \rightarrow 2^{NDP}$, a function which returns all permissions that can not be delegated user u holds at a specified time point t.
10) $prior(u, r)$: $Users \times DTR \rightarrow Users \times DTR$, a function returns all the delegations prior to the delegation (u, r).
11) $path(u, r$: $Users \times DTR \times \left((u_{0,} r_0), \dots (u_{i,} r_i)\right)$, a function returns the delegation path of delegation (u, r).
12) $TC(u, p, t)$: $Users \times DTR \times CDP \times T \rightarrow TimeConstraints$, a function returns the time constraint of a specified delegation user u holds at a specified time point t.

Role-role delegation is an improvement of PBDM2. Similar to user-user delegation in this article, role-role delegation reduce the number of roles types. Role types in role-role delegation are divided into three categories: common roles(CR), temporary delegated roles(TDBR) and delegation roles(DTR). Fig. 3 is the role-role delegation model.

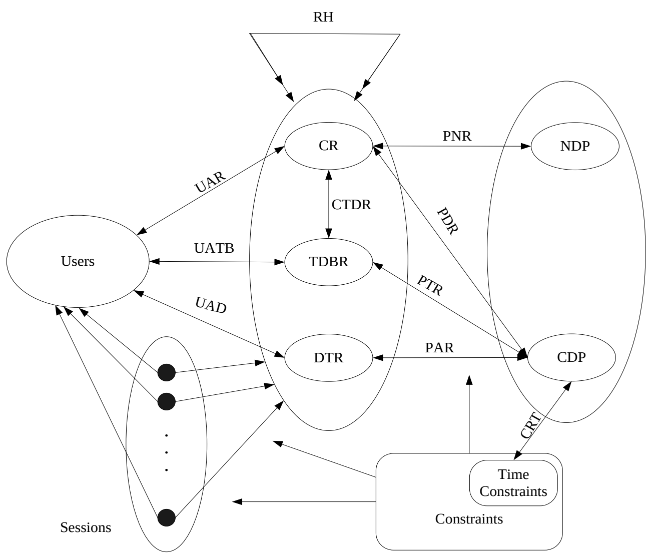

Fig. 3. Role-role delegation in CTCDM.

Semantics of most elements in role-role delegation is similar to user-user delegation. The biggest difference between them is the TDBR. When the administrator initiates a delegation, he first choose some permissions that can be delegated and then delegate these permissions to the TDBR belonging to the delegated roles.

Below are the explanations of the elements in role-role delegations (elements which have the same meaning with user-user delegation is out of our discuss).

1) $TDBR$: owned by delegated roles and it contains all the delegated roles.
2) $UATB \subseteq Users \times TDBR$: many to many mapping relations of $Users$ and $TDBR$.
3) $CTDR \subseteq CR \times TDBR$: many to many mapping relations of CR and $TDBR$.
4) $PTR \subseteq TDBR \times CDP$: many to many mapping relations of $TDBR$ and CDP.

Below are the functions of role-role delegation：

1) $Cr\_tdbr(cr, t)$, $CR \rightarrow 2^{TDBR}$, a function which returns all temporary delegated roles of a specified common role at time point t.
2) $TDBRs(s_i, t)$: $Sessions \rightarrow 2^{TDBR}$, a function which returns all the temporary delegated roles belonging to session s_i at a specified time point t.
3) $Permissions(s_i, t)$: $Sessions \rightarrow 2^{NDP+CDP}$, a function which returns all the permissions belonging to session s_i at a specified time point t..
4) $TC(r, p, t)$: $TDBR \times CDP \times T \rightarrow TimeConstraints$, a function returns the time constraint of a specified delegation user u holds at a specified time point t.

IV. Conclusion

Complete Time Constraint Based Delegation Model is on the basis of PBDM1 and PBDM2 models and it is divided into two parts: user-user delegation and role-role delegation. In order to solve name space explosion problem caused by the agent role in PBDM1 and PBDM2, delegatable roles are removed. In addition, delegation users no longer need to create delegation roles every time they initiate a delegation. Also, time constraint is of great significance to the delegation model. In the real world, Permissions or roles one person often change with time. There are mainly two methods of time representation: discrete time representation and period time representation. However, these two methods are not able to express complex time constraints with flexibility. So this paper extends the time representation by combining them together with adding the times limit and total time limit. What's more, this paper also modifies the expression of period time to support more time constraints. Next, the Complete Time Delegation Model will be implemented in a real system developed by our lab.

References

[1] E. J. Coyne, H. L. Feinstein, C. E. Youman *et al.*, "Role-based access control models," *Computer*, vol. 29, no. 2, pp. 38-47, 1996.
[2] J. Huang, S. Qing, and H. Z. Wen, "Timed role-based access control," *Journal of Software*, no. 2, pp. 1944-1954, 2003.
[3] B. Elisa, C. Bettini, E. Ferrari, and P. Samarati, "An access control model supporting periodicity constraints and temporal reasoning," *ACM Transactions on Database Systems (TODS)*, vol. 23, no. 3, pp. 231-285, 1998.
[4] L. H. Zhang, G.-J. Ahn, and B.-T. Chu, "A rule-based framework for role based delegation," in *Proc. the Sixth ACM Symposium on Access Control Models and Technologies*, ACM, 2001, pp. 153-162.
[5] X. H. Zhang, S. Oh, and R. Sandhu, "PBDM: a flexible delegation model in RBAC," in *Proc. the Eighth ACM Symposium on Access Control Models and Technologies*, ACM, 2003, pp. 149-157.
[6] Fred. B. Schneider, "Least privilege and more," *Computer Systems*, pp. 253-258, Springer New York, 2004.
[7] G. X. Zhou, M. Demirer, C. Bayrak, and L. C. Wang, "Enable delegation for RBAC with Secure Authorization Certificate," *Computers & Security*, vol. 30, no. 8, pp. 780-790, 2011.
[8] W. C. R. Lui, L. C. Hui, and S.-M. Yiu, "Delegation with supervision," *Information Sciences*, vol. 177, no. 19, pp. 4014-4030, 2007.

[9] H. Wang and S. L. Osborn, "Static and dynamic delegation in the role graph model," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 10, pp. 1569-1582, 2011.

[10] T. Roberto, D. F. Yao, and W. H. Winsborough, "Independently verifiable decentralized role-based delegation," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 40, no. 6, pp. 1206-1219, 2010.

Chunyan Han was born in Fujian Province, China in 1990. She is now pursuing a mater degree in computer science and technology in Beihang University and she has gained the bachelor degree in Beijing University of Posts and Telecommunications.

She has published a paper named “Time Constraints of Access Control” in 2nd International Conference on Precision Mechanical Instruments and Measurement Technology, ICPMIMT 2014. Her interests are database tuning and access control.

Liu Lianzhong was born in 1958. He received a M.S. degree from Huazhong University of Science and Technology, Wuhan, China, in 1985.

He is currently a professor of the School of Computer Science and Engineering, Beihang University. His research interests include social complex networks, database security, identity and access manager, analysis and design of large information system.

Li Zhouyang was born in China in 1991. He graduated from Beijing Jiaotong University. He is now pursuing a master’s degree in computer science and technology in Beihang University. His research interests include web development, database security and design of information system.

He has participated in a development of train control system called CBTC in his junior year. He is now doing a CRM project and researching on social network.