Towards Reliable Trust Management based on Structural Trust Scopes of Distributed Roles

Gang Yin[†], Ning Zhou[‡], Huaimin Wang[†]

[†] *School of Computer, National University of Defense Technology, Changsha, China*
[‡] *Institute of Electronic System Engineering of China, Beijing, China*
jack.nudt@gmail.com, whm_w@163.com, humimi74@yahoo.cn

Abstract

Role based trust management uses distributed role hierarchies (DRH) to provide flexible and scalable authorization in multi-domain environments, but DRH are inherently transitive and may easily lead to unexpected or even illegal authorization. In this paper, we propose TS-DRH, a generalized extension of DRH based on a novel trust scope model of distributed roles. TS-DRH introduces the notion of scoped roles with four kinds of structural trust scopes, and uses scoped roles to control the member scopes of senior roles and the permission scopes of junior roles, and thus helps to control the propagation of role memberships and permissions in DRH. This paper also designs rule based semantics and a compliance checking algorithm to compute authorization decisions for TS-DRH.

1. Introduction

Trust management (TM) is a delegation-based approach to access control in environments where entities in different administrative domains want to share resources [2, 3, 4, 5, 6, 7, 8, 12]. Delegation is the key mechanism in TM systems to transfer privileges between entities. Early TM systems adopt capability-based delegation to enable authorization collaboration between entities [4, 5, 7], while role-based TM systems distinguish from them by using role hierarchies on distributed roles [2, 8].

In role-based TM systems, a distributed role is a role name combined with a definition entity as its administrative domain. To support delegation, role-based TM systems allow the privileges to be transferred between roles through a chain of distributed roles, which are called distributed role hierarchies (DRH) in this paper. DRH simplify the tasks of authorization for a large amount of users that unknown to authorizers, which are suitable for large scale distributed systems such as Grid and P2P system built upon Internet.

However, DRH are inherently transitive since they are built upon the transitivity of trust relationship between entities, and thus may easily lead to unexpected or even illegal authorization. Similar to traditional role hierarchies, in DRH, the member set of a junior role contain the members of its senior roles, and a senior role inherits the permissions of all its junior roles. Such semantics of DRH are not suitable for the Internet-based applications where trust relationships between entities are dynamic and non-transitive. First, DRH may authorize unexpected users. For example, given two RT [8] credentials $A.r \leftarrow B.r_1$ and $B.r_1 \leftarrow C.r_2$, according to the semantics of RT, all members of $C.r_2$ are members of A.r. This may not be expected if A does not trust C, and it is difficult for A to resolve such problem because usually B is not controlled by A. Second, DRH may authorize users with unexpected permissions. In above example, all members of $C.r_2$ will be authorized with the permissions of A.r, and this may not be expected by B if A is a more private domain that delegate authorization tasks to B who has the responsibility to protect A.

The research for more reliable delegation mechanisms are widely studied in capability-based delegation systems [1, 7, 12], such as boolean control [4, 5], delegation depth control [7], and comprehensive constraint model on each delegation steps [1]. However, to our knowledge little work has been done to control the propagation of privileges in DRH systems, including some influential role-based TM systems [2, 8].

In this paper, we propose a novel trust model for DRH which defines different trust scopes of distributed roles, and introduce a generalized extension of DRH to control the propagation of privileges in the use of DRH. Main contributions of this paper include: (1) a role-based structural trust scope model for DRH is proposed, which can be used to control the propagation of role memberships and permissions; (2) a trust scope based DRH model TS-DRH is proposed, which introduces the notion of scoped roles and defines a credential framework for restricted DRH policies; (3) a rule based semantics for TS-DRH systems is defined and a compliance checking algorithm is designed to compute authorization decisions for TS-DRH.

The rest of this paper is organized as follows. Section 2 introduces the structural trust scope model by using sample role hierarchies of a virtual organization. Section 3 defines the formal model of TS-DRH, its semantics and the compliance checking algorithm. Section 4 compares our work with some related works and section 5 concludes this paper.

2. Trust Scopes of Distributed Roles

In this section, we introduce two kinds of structural trust scopes of distributed roles: role member scopes and role permission scopes. Similar to traditional roles [10], distributed roles can also be viewed as two kinds of sets: sets of permissions and sets of users.

In typical DRH systems such as RT [8], role hierarchies are described with a dominate relation. For example, a RT credential $A.r \leftarrow B.r_1$ means that the role $B.r_1$ dominates the role A.r. The semantics of dominate relation includes two parts: (1) all permissions of A.r are also permissions of $B.r_1$; and (2) all members of $B.r_1$ are also members of A.r. Both parts of above semantics are transitive when there are a chain of distributed roles connected with dominate relation. In decentralized environment, this semantics are not appropriate for practical applications. Consider the resource sharing in a multi-domain environment.

Example 1. ARG is an international Aids research group. ELab is an epidemical lab. NAI is a national Aids institute. ComA and ComB are two companies. ARG allows its scientists to run the tasks at ARG, and allows NAI professors and Aids researchers in ComA to run the aids tasks. ELab allows ARG scientists to run the aids tasks at ELab. ComA allows the Aids researchers in ComB to use the permissions of the Aids researchers in ComA. Above security requirements can be modeled by a DRH structure, as shown in figure 1. The policies can be defined by five RT credentials.

(1) ARG.scientist←NAI.prof

(2) ARG.scientist←ComA.aids-r

(3) ELab.aids-task←ARG.scientist

(4) ARG.task←ARG.scientist

(5) ComA.aids-r←ComB.aids-r

These credentials define the backbone DRH structures, and can not disable the flow of privileges that do not expected by entities. For example, ELab may not want ComA and

ComB to run its Aids tasks. ARG may not want ComB to run its tasks, but allow ComA to do so. However, according to the DRH semantics of RT, the members of ComB.aids-r should be allowed to run the tasks controlled by ELab.aids-task and ARG.task.

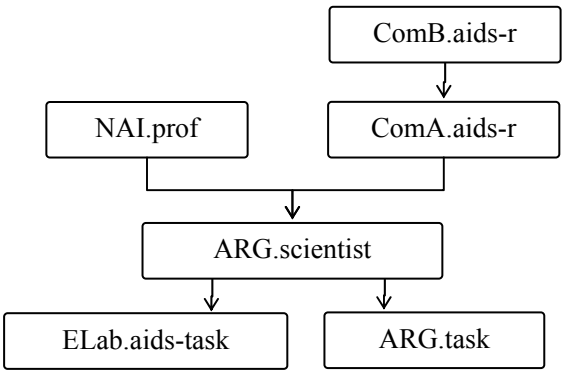

Figure 1. Sample DRH for a virtual organization

DRH does not control the flow of permissions and memberships. Given the credential $A.r \leftarrow B.r_1$, the members of $B.r_1$ are usually composed of the entities directly assigned with $B.r_1$ by B and the entities from different domains by authorization processes with delegation. We consider four kinds of role members:

1. Members directly assigned with $B.r_1$, which is called the role scope of $B.r_1$, and denoted as $B.r_1.s_r$.

2. Members of $B.r_1$ and its senior roles defined by domain B, which are called the affiliation scope of $B.r_1$. We use $B.r_1.s_a$ to denote the affiliation scope of $B.r_1$;

3. Members of $B.r_1$ and its senior roles defined by B and other specified domains, which are called domain scopes of $B.r_1$. We use $B.r_1.s_{ds}$ to denote a domain scope of $B.r_1$, where ds is a set of domains.

4. Members of $B.r_1$ and all its senior roles, which is called the entire scope of $B.r_1$, and denoted as $B.r_1.s_e$.

Figure 2 shows the typical member scopes of $B.r_1$, which are orderly denoted as $B.r_1.s_r$, $B.r_1.s_a$, $B.r_1.s_{\{D\}}$ and $B.r_1.s_e$ respectively. Consider example 1, to prevent the Aids researchers of ComB to run the tasks and allow all NAI professors to run the tasks, ARG can define its credentials as follows:

(6) ARG.scientist←ComA.aids-r.s_a

(7) ARG.scientist←NAI.prof.s_e

According to the definition of ComA.aids-r.s_a, the Aids researchers of ComB can not obtain the membership of ARG.scientist based on above credentials while the NAI professors can use the role ARG.scientist.

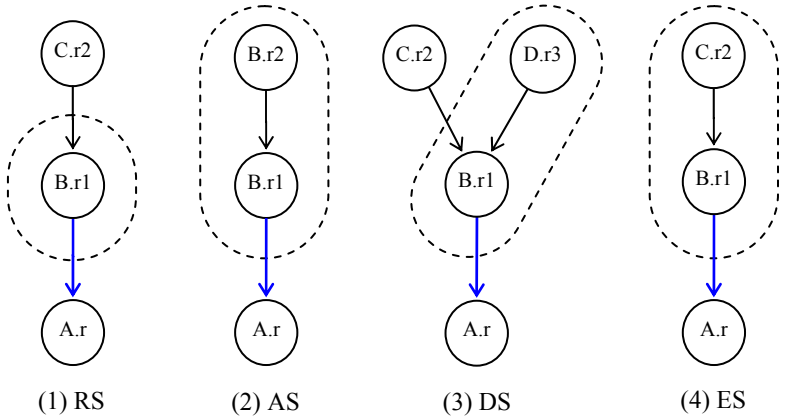

Figure 2. Different Member Scopes in DRH

Consider the credential $A.r \leftarrow B.r_1$, one can conclude that the members $B.r_1$ can use all the permissions of $A.r$. The permissions of a role may be directly assigned to roles, or inherited by its junior roles. Similar to role member scopes, a distributed role have four kinds of role permission scopes: $A.r.s_r$, $A.r.s_a$, $A.r.s_{\{C\}}$, $A.r.s_e$, as shown in figure 3, where $A.r.s_r$ denotes the permissions directly assigned to $A.r$; $A.r.s_a$ denotes the permissions of $A.r$ and its junior roles within domain A; $A.r.s_{\{C\}}$ denotes the permissions of $A.r$ and its junior roles defined in A and C; $A.r.s_e$ denotes the permissions of $A.r$ and all its junior roles.

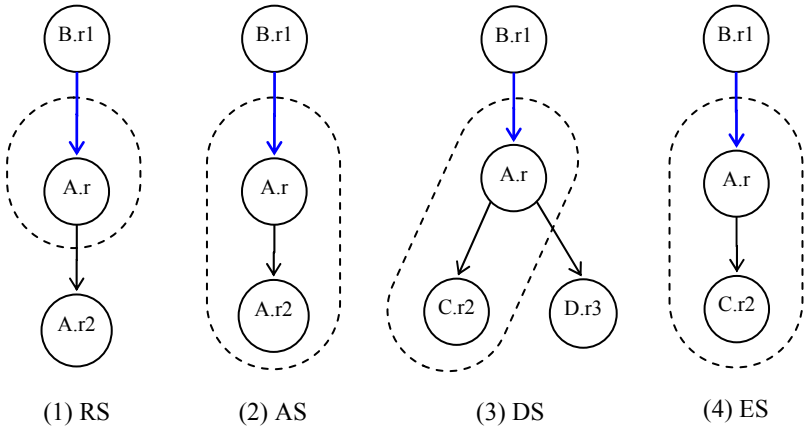

Figure 3. Different Permission Scopes in DRH

Now reconsider example 1, ARG can control the flow of permissions from ELab to ComA, and allow the professors in NAI to run the Aids tasks in ELab:

(8) ARG.scientist.$s_a \leftarrow$ComA.aids-r.s_a

(9) ARG.scientist.$s_e \leftarrow$NAI.prof.s_e

Given a distributed role hierarchy $A.r.ts_1 \leftarrow B.r_1.ts_2$, the trust scopes add the following new semantics: the members of $B.r_1.ts_2$ are authorized with the permissions of $A.r.ts_1$. Trust scopes give the senior roles and junior roles different semantics and thus support various security requirements.

3. Formal Model

In this section, we give a formal model of a trust scope-based DRH (TS-DRH) and its semantics. TS-DRH uses scoped roles to control the flow of role memberships and permissions. TS-DRH is built upon the following sets:

- E, P and N are the sets of all entities, permissions and role names respectively;
- $R \subseteq E \times N$ is the set of all roles;
- $DS \subseteq \wp(E)$ is the set of all domain scopes;
- $TS = \{s_r, s_a, s_e\} \cup DS$ is the set of all trust scopes;
- $SR \subseteq R \times TS$ is the sets of all scoped roles;

Users, authorizers and domains are entities in TS-DRH. Given an entity A, a role name r and a role trust scope ts, we use $A.r$ and $A.r.ts$ to denote a role and a scoped role defined by A. The trust scopes s_r, s_a and s_e are use to denote the role scopes, affiliation scopes and entire scopes of a given role. Each domain scope is denoted with a set of domains. For example, a domain scope $s_{\{A., B, C\}} \in DS$ represents the scope including the domains A, B and C. Now we introduce the *comply* relation on scoped roles, which defines the relations between different scoped roles, and will be used to compute the authorization decisions.

Definition 1. The comply relation $\hookrightarrow$ is a one to one relation on SR. Given sr_1, $sr_2 \in SR$, $sr_1 \hookrightarrow sr_2$ means sr_1 represents a role with more restricted trust scope than sr_2, and read as sr_1 complies sr_2. Given $A.r \in R$, $\hookrightarrow$ has the following properties:

- $A.r.s_r \hookrightarrow A.r.s_a \hookrightarrow A.r.s_e$;
- $A.r.s_{\{A\}} \hookrightarrow A.r.s_a$ and $A.r.s_a \hookrightarrow A.r.s_{\{A\}}$;
- $A.r.ts \hookrightarrow A.r.ts \hookrightarrow A.r.s_e$, $ts \in TS$;
- $A.r.s_{ds1} \hookrightarrow A.r.s_{ds2}$, where $ds_1 \subseteq ds_2$, ds_1, $ds_2 \in DS$;

Affiliation scopes and entire scopes are special cases of domain scopes. For example, given a role $A.r$, $A.r.s_{\{A\}}$ is identical to $A.r.s_a$, and all trust scopes of $A.r$ comply the entire scope $A.r.s_e$.

3.1 Trust Scope-based DRH Credentials

TS-DRH uses three kinds of credentials to define authorization policies. A TS-DRH system also contains the set of access control queries and the authorization relation. The semantics of the authorization relation will be defined in section 3.2.

Definition 2. A TS-DRH system α is a 5-tuple (PRA, URA, DRH, ACQ, $\Rightarrow$), which are defined as follows:

- $A.r \leftarrow p \in PRA \subseteq R \times P$ means A assigns p to $A.r.s_e$;
- $A.r.ts \leftarrow u \in URA \subseteq SR \times E$ means A assigns $A.r.s_r$ to u and u can use the permissions in $A.r.ts$;
- $A.r.ts_1 \leftarrow B.r_1.ts_2 \in DRH \subseteq SR \times SR$ means all members in $B.r_1.ts_2$ are members of $A.r.ts_2$ and can use the permissions in $A.r.ts_1$;

where ACQ is the set of all access control queries and $\Rightarrow$ is an authorization relation. The tuple $\lambda(x, y, p) \in ACQ \subseteq E \times E \times P$ represents an access control query whether x can authorize p to y. Given $\lambda(x, y, p) \in ACQ$, $\alpha \Rightarrow \lambda(x, y, p)$ means that x can allow y to use the permission p in TS-DRH.

3.2 Semantics

Now we define the semantics of TS-DRH based on several sorted predicates, and the authorization decisions are computed by a trust scope-based compliance checking algorithm TSCCA, as shown in figure 4.

Definition 3. The predicates defining the semantics of TS-DRH includes permit, pra, m, m-ps, m-us, ura-ps, ura-us. The basic three predicates are the following:

- x.pra(p, r): entity x assigns permission p to role $x.r$.
- x.ura-ps(y, r, ts): entity x assigns entity y with the permissions in $x.r.ts$.
- x.ura-us(y, r, ts): entity x assigns entity y with the membership of $x.r.ts$.

where x, y are entities, p is a permission, ts is a trust scope and r is a role name. The intuitionistic meaning of x.permit(y, p) is that x allows y to access the resource controlled by p. The meanings of other predicates are defined by the compliance checking algorithm TSCCA.

Definition 4. Given a set Σ of credentials, the credential base B_Σ for C is a rule set that can be constructed by the following transformation process:

- Given $A.r \leftarrow p$, then add pra(p, r) into B_Σ;
- Given $A.r.ts \leftarrow u$, then add A.ura-ps(u, r, ts) and A.ura-us(u, r, s_r) into B_Σ;
- Given $A.r.ts_1 \leftarrow B.r_1.ts_2$, then add A.ura-ps(x, r, ts_1)$\leftarrow B$.ura-ps(x, r_1, ts), c_1 into B_Σ; add A.ura-us(x, r, ts_2)$\leftarrow B$.ura-us(x, r_1, ts), c_2 into B_Σ; where $c_1 = B.r_1.ts \hookrightarrow B.r_1.s_{\{A\}}$, $c_2 = B.r_1.ts \hookrightarrow B.r_1.s_{\{A\}}$.

Given a credential base and an authorization query, TSCCA computes the authorization decision by proving the authorization query. If the query x.permit(y, p) is proved, then x should allow y to use p. TSCCA uses depth-first-search processes to construct delegation chains from x to y: delegation chains of permissions and delegation chains of role memberships.

(α1) x.permit(y, p) $\leftarrow$ x.pra(p, r), x.m(y, r, s_r, s_r).

(α2) x.m(y, r, ps, us) $\leftarrow$ x.m-ps(y, r, ps), x.m-us(y, r, us).

(α3) x.m-ps(y, r, ps_1) $\leftarrow$ x.ura-ps(y, r, ps_2), $x.r.ps_1 \hookrightarrow x.r.ps_2$.

(α4) x.m-us(y, r, us_1) $\leftarrow$ x.ura-us(y, r, us_2), $x.r.us_1 \hookrightarrow x.r.us_2$.

Figure 4. Compliance Checking for TS-DRH

Now we explain how TSCCA constructs the delegation chains. Given four credentials: 1. $A.r \leftarrow p$; 2. $A.r.s_e \leftarrow B.r_1.s_a$; 3. $B.r_1.s_{\{A\}} \leftarrow C$; 4. $B.r_1.s_a \leftarrow D$, we have the credential base containing the following rules:

a: A.pra(p, r)

b: A.ura-ps(x, r, s_e)$\leftarrow$$B$.ura-ps($x$, r_1, ps), $B.r_1.ps \hookrightarrow B.r_1.s_{\{A\}}$

c: A.ura-us(x, r, s_a)$\leftarrow$$B$.ura-us($x$, r_1, us), $B.r_1.us \hookrightarrow B.r_1.s_{\{A\}}$

d: B.ura-ps(C, r_1, $s_{\{A\}}$)

e: B.ura-us(C, r_1, s_r)

f: B.ura-ps(D, r_1, s_a)

g: B.ura-us(D, r_1, s_r)

where we translate credential 1 into rule a, credential 2 into b and c, credential 3 into d and e, and credential 4 into f and g.

To decide whether C and D is permitted by A to use p, the predicates A.permit(C, p) and A.permit(D, p) need to be proved. Consider the goal oriented deduction process. For the first predicate, according to rule α1, A.pra(p, r) and A.m(C, r, s_r, s_r) should be proved. A.pra(p, r) is true by rule a and by rule α2 we should prove A.m-ps(C, r, s_r) and A.m-us(C, r, s_r). By the rules b, d, and the definition of relation $\hookrightarrow$, A.m-ps(C, r, s_r) can be proved. By the rules c, e , α4 and the definition of $\hookrightarrow$, A.m-us(C, r, s_r) can be proved. Therefore, TSCCA can construct the delegation chains from A to C of both permission p and membership of $A.r$, and thus A.permit(C, p) can be proved. However, by the rule f, B does not allow D to use the permissions that $B.r_1$ inherits from its junior roles, thus A.m-ps(C, r, s_r) can not be proved by the rule b. Therefore A.permit(D, p) is not true, and A will not allow D to use p.

4. Related Work

Most role-based TM systems provide little control over the flow of privileges in DRH. Some influencing role-based TM systems mainly support boolean control on the privilege propagation in DRH policies [2, 8]. Cassandra [2] provides depth control on delegation of role capabilities, not on DRH policies. The role intersections in RT [8] may be used to control the flow of role memberships by adding roles in credential body. But role intersections need to define new roles to model domains, and they can not express most of the role member scopes and permission scopes except entire scopes.

Similar to Cassandra, SecPAL [3] is a generalized declarative authorization language that can express DRH policies by using the verb phrase *can say*$_d$ where d is an integer or ∞ and can be used to control the depth of DRH. But SecPAL can hardly express the domain scopes of role members and all the permission scopes in TS-DRH.

Role-based cascaded delegation [12] distinguishes the difference between affiliated and delegated roles, and the semantics of the model assigns the affiliated roles (directly assigned to users by the defining entities) with larger permission scopes than delegated roles. However, the model allows the members of roles to re-delegate the role capabilities delegated from higher authorities, which essentially belongs to capability based delegation and does not support DRH.

TS-DRH provides the control on delegation with the depth of 1, 2 and ∞, where the depth of 2 is enabled by the role scope s_r. We believe that the delegation depth of 2 covers a wide

range of practical delegation scenarios. Furthermore, the scoped roles in TS-DRH can be regarded as a general extension of traditional concept of roles [10], with the interpretations that roles are the abstractions of both users and permissions.

5. Conclusion

Role-based TM system use DRH to provide more flexible and scalable authorization in multi-domain environments. TS-DRH is a generalized extension of DRH and enables more refined control mechanisms on role-based TM systems. TS-DRH uses structural trust scopes to control the member scopes of senior roles and the permission scopes of junior roles, and thus helps to control the flow of role memberships and permissions in DRH policies. TS-DRH introduces the notion of scoped roles, which may enrich the role-based access control with more practical semantics.

Proof-test research on combining TS-DRH with more sophisticated DRH policies maybe interesting, such as the role intersections and linked roles [8]. It is attractive to design trust scopes for other security paradigms, such as constrained delegation [1], security administration [9] and separation of duties [11].

6. Acknowledgements

This research is supported by the National Basic Research 973 Program of China under Grant No.2005CB321804 and the National Natural Science Foundation of China under Grant No.90412011. The authors would like to thank the anonymous reviewers for valuable comments.

7. References

[1] O. Bandmann, M. Damy, B. S. Firozabadi, *Constrained Delegation*, Proceedings of the 2002 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, 2002.

[2] M. Y. Becker, P. Sewell, *Cassandra: Flexible Trust Management, Applied to Electronic Health* Records, Proceedings of the 17th IEEE Computer Security Foundations Workshop, CSFW 2004.

[3] M. Y. Becker, C. Fournet, A. D. Gordon, *Design and Semantics of a Decentralized Authorization Language*, 20th IEEE Computer Security Foundations Symposium, pages 3-15, CSF 2007.

[4] M. Blaze, J. Feigenbaum, and J. Lacy. *Decentralized trust management*. In Proceedings of 17th Symposium on Security and Privacy, pages 164-173, Oakland, 1996.

[5] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. D. Keromytis. *The KeyNote trust-management system*, version 2. IETF RFC 2704, September 1999.

[6] T. Jim. *SD3: A trust management system with certified evaluation*. In Proceedings of the 2001 IEEE Symposium on Security and Privacy, pages 106-115. IEEE Computer Society Press, May 2001.

[7] N. Li, B. N. Grosof, and J. Feigenbaum. *Delegation logic: A logic-based approach to distributed authorization*. TISSEC, February 2003.

[8] N. Li, J. C. Mitchell, and W. H. Winsborough. *Design of a role-based trust management framework*. In Proceedings of the 2002 IEEE Symposium on Security and Privacy, pages 114-130. IEEE Computer Society Press, May 2002.

[9] S. Oh, R. Sandhu, *A Model for Role Administration Using Organization*, SACMAT'02, June 3-4, 2002, Monterey, California, USA, pp. 155-162.

[10] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. *Role-based access control models*. IEEE Computer, 29(2):38–47, February 1996.
[11] R. T. Simon and M. E. Zurko. *Separation of duty in role-based environments*. In Proceedings of the Tenth Computer Security Foundations Workshop, pages 183–194. IEEE Computer Society Press, 1997.
[12] R. Tamassia, D. Yao, W. H. Winsborough, *Role-Based Cascaded Delegation*, SACMAT'04, June 2–4, 2004, Yorktown Heights, New York, USA.

Authors

YIN Gang

Born in 1975, Ph. D. Lecturer. His main research interests include distributed computing, access control, trust management and reputation management in P2P systems.

ZHOU Ning

Born in 1974, M.S. Engineer, Her main research interests include information security, software engineering, design and test of large scale distributed information systems.

WANG Huai-min

Born in 1962, Ph. D. Professor and Doctor Supervisor. His research interests include artificial intelligence, distributed computing, network and information security.