协同环境中共有资源的细粒度协作访问控制策略*

雷 浩[1+], 黄 建[2], 冯登国[1]

[1](中国科学院 软件研究所 信息安全国家重点实验室,北京 100080)

[2](Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford, London OX1 3QD, England)

A Fine-Grained Coalition Access Control Policy for Jointly-Owned Resources in Collaborative Environments

LEI Hao[1+], HUANG Jian[2], FENG Deng-Guo[1]

[1](State Key Laboratory of Information Security, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

[2](Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford, London OX1 3QD, England)

+ Corresponding author: Phn: +86-10-62528254-803, E-mail: leihao00@iscas.cn, http://www.is.iscas.ac.cn

Received 2004-02-27; Accepted 2004-07-27

Lei H, Huang J, Feng DG. A fine-grained coalition access control policy for jointly-owned resources in collaborative environments. *Journal of Software*, 2005,16(5):1000−1011. DOI: 10.1360/jos161000

Abstract: Joint access to shared resources (e.g., objects, applications, and services) among autonomous domains that form a coalition has recently become important in both military and commercial areas. The brass tacks in coalition are that these domains have different self-interests although they focus on achieving a common goal. In this paper, to enable effective protection of jointly-owned resources, the notion of trust into coalition access control is built, and a fine-grained access control policy based on quantifying permission idea is proposed. The usage format of permission in this policy is meta-permission that is a share of access permission to coalition resources and is owned by multiple domain users. When accessing jointly owned resources, the sum of participants' meta-permission value must attain a predefined permission quantity called "permission-threshold" and an assigned participant member number. In addition, permissible time span of the meta-permission is also taken into account to achieve the above goals and access requesting time must fall into their common permissible time span. Doing this enables the coalition to retain control over the access to coalition resources in distributed environments. Lastly, the preserving security property of the fine-grained access control policy with respect to state transition is proven.

Key words: coalition; trust; meta-permission; permissible time span of meta-permission

* Supported by the National Natural Science Foundation of China under Grant No.60273027 (国家自然科学基金); the National Grand Fundamental Research Program of China under Grant No.G1999035802 (国家重点基础研究发展规划(973)); the National Science Fund for Distinguished Young Scholars under Grant No.60025205 (国家杰出青年科学基金); the Hi-Tech Research and Development Program of China under Grant No.2004AA147070 (国家高技术研究发展计划(863))

LEI Hao was born in 1975. He is a Ph.D. candidate at the Institute of Software, the Chinese Academy of Sciences. His current research areas are system security, theory and technique of information security. **HUANG Jian** was born in 1976. He is a Ph.D. candidate at Computing Laboratory, Oxford University, England. His current research areas are information flow, theory and technique of information security. **FENG Deng-Guo** was born in 1965. He is a professor and doctoral supervisor at the Institute of Software, the Chinese Academy of Sciences. His current research areas are cryptography and protocol analysis.

摘 要: 在军事和商业领域中,由多个自治域形成的协作群体对共有资源(如客体、应用程序以及服务等)的访问问题越来越受到重视.协作中的基本事实是:尽管这些自治域有共同的目标,但同时有不同的自身利益.为了有效地保护共有资源,把"信任"的概念引入了协作访问控制中,并在基于量化权限的思想上,提出了细粒度的协作访问控制策略.在该策略里,权限的使用形式是元权限,也就是单位权限,它是访问共有客体权限的一个划分,可为多个域中不同用户所拥有.当访问共有资源时,参与者们所拥有的元权限的值之和以及人数必须达到规定的权限门限值和人数值,并且访问时间是所有参与者的共同许可访问时间段,这使得可以对协作资源进行有效地分布控制.另外,还引入了元权限的使用时间段约束.最后,证明了该细粒度协作访问控制策略关于协作系统的状态转换是保持安全的.

关键词: 协作;信任;元权限;元权限使用时间段

中图法分类号: TP393 **文献标识码**: A

1 Introduction

The popularity of coalition access control has grown mainly due to the fact that coalition can take place across organization boundaries. Joint access to shared resources (e.g., objects, applications, and services) among autonomous domains that form a coalition has recently become important in both military and commercial areas. Autonomous domains (i.e., entities of autonomous security policy administration) form coalition to achieve common goals by sharing resources such as objects, applications and services[1,2].

A case in point provided in Ref.[1] is stated as follows: "consider a private genetics research company that has discovered the gene sequence for a particular disease and wishes to form an alliance with a private hospital and a pharmaceutical company for research into finding a cure for the disease using the gene sequence. Given the financial commitments and expected eventual impact of finding such a cure, all three domains would like to jointly own and jointly administer access to all research data generated."

Sociologists and psychologists have studied the concept of trust extensively. Chopra and Wallace have extensively surveyed the literature on social trust and trust in on-line systems, and note that trust is an essential ingredient of effective collaborations[3].

Firstly, in general, there commonly exists significant trust discrepancy such as the degree of reliability and strength for varied coalition member domains in collaboration. For example, in a company, one may put more trustworthiness on board chairman while less on a common employee. The same could be said among coalition member domains. Therefore, for accessing jointly-owned resources, participant's trustworthiness should be taken into account.

Secondly, because autonomous coalition member domains may have competitive or even adversarial relationships, they do not completely trust each other. In other words, their trust to each other is limited although they are motivated by a common goal to share some of the resources[4]. The jointly-owned resources such as finical assets and research data are deemed essential for coalition operations so that any single user is not trustworthy enough to access coalition resources and it requires a set of users collectively to execute this access. Therefore, it is desirable that the participants involved in an access have the right to know what will happen to jointly-owned resources. For convenience, we call this requirement as "the right-to-know and superintendence right about participants".

Unfortunately, access control policy available today to support collaborative activities does not provide any notion of building trust. Based on quantifying permission ideas, our paper adopts the architectures proposed in Refs.[1,2] and will be concentrated on exploring a fine-grained joint access control policy. The usage format of permission in this policy is meta-permission whose value represents the reliability and strength of coalition members.

The paper is organized into 5 other Sections. Section 2 discusses the closely related work and problems about coalition access control. Section 3 proposes the quantifying permission idea and introduces the concept of meta-permission. Section 4 presents the basic elements used in policy by introducing a set of concepts, then we extend the access control matrix to a quantification access control matrix and an access need matrix. Section 5 explains how to preserve security property of the fine-grained constraints with respect to state transition. In the final Section 6, we conclude our work and provide a detailed example about how to establish meta-permission with a specific permission on a coalition object.

2 Related Work and Problems

The requirements for jointly owned resources have been identified by many researchers[1,3−5] and some work has been done on this issue. In this section we review the closely related work and problems in this area.

Khurana *et al*.[1,2] investigated an access control policy that would require the number of participants must meet that of the predefined participant member when accessing jointly-owned resources; Shands *et al*.[4] developed a Secure Virtual Enclave environment where domains with Role Based Access Control instantiations can share resources; Thompson *et al*.[6] implemented and deployed an access control mechanism that uses digitally-signed certificates to define and enforce an access policy for a set of distributed resources that have multiple, independent and geographically dispersed stakeholders.

Although the above research works address varied important concerns in coalitions, they do not take into consideration the discrepancy in trustworthiness among coalition domains and all the participants' trustworthiness during access implementation. The fine-grained policy proposed in this paper is based on quantifying permission ideas, and the usage format of permission in this policy is meta-permission whose value represents the reliability and strength of coalition members. Moreover, access to jointly owned objects requires that the sum of participants' meta-permission value should meet a predefined permission threshold and the access requesting time should fall into their common permissible time span.

3 Overview of Quantifying Permission and Meta-Permission Concept

In this section we will explore the quantitative permission control measures to deal with the discrepancy in trustworthiness among coalition members when accessing coalition resources.

Coalitions are implemented through resource sharing which is achieved by the distribution of permissions for coalition resources to coalition domain users based on resource-sharing agreements (i.e., shared accesses to resources of member domains)[1]. The question is how we can distribute permission to reflect the discrepancy such as the degree of reliability and strength among collaborating members mentioned above, and how we can establish an access control policy to support the intended goals.

In view of philosophy, everything has two attributes: quality and quantity, so does permission. Therefore, permission can be comprehensively described as (permission's quality, permission's quantity) and we call the pair (permission's quality, the unit of permission's quantity) as meta-permission. The unit of permission's quantity reflects the trustworthiness of participants, and different unit of permission's value describes the discrepancy such as the degree of reliability and strength among varied coalition members. In implementation, meta-permission is placed in the form of attribute certificate for each coalition member's assurance. Users in a specified domain share the same meta-permission, but different domain users may be authorized different meta-permission whose value reflects their discrepancy in trustworthiness.

Furthermore, access to jointly owned resources requires the sum of participants' meta-permission value must meet the predefined permission threshold and the assigned participant member number at the same time. The

(permission threshold, participant member number)-pair is named “access requirement”. For varied jointly-owned resources, there will be different access requirement according to the sensitivity of resources. Permission-threshold and participant member number depend on the information it contains. The more sensitive the content, the higher permission threshold and participant member number should be required. Moreover, setting and updating access requirement for varied jointly-owned resources and participant’s meta-permission must be agreed upon by all autonomous member domain administrators. Doing this enables coalition to retain control over the access to coalition resources: on the one hand, meta-permission is a share of access permission to coalition objects and is owned by multiple domain users; on the other hand, the relationship between meta-permission and permission threshold constrains the behavior of single user so that he alone cannot access the jointly-owned resources.

In addition to permission quantitative requirement, we also add permissible access time span constraint on meta-permission. Without time span constraint, the misuse of meta-permission may happen. For example, we may assign normal workday hours as users’ permissible access time.

Considering the scenario at the beginning of this paper, we assume that meta-permission and quantitative permission requirements to “Research Data” which are agreed upon by all the three organization’s administrators as Table 1 shows.

Table 1 Meta-Permission and quantitative access requirement of “Research Data”

(Permission-Threshold, number of articipants)	Genetics research company	Private hospital	Pharmaceutical company
(6,2)	{Write, 5, (8:00-11:00)}	{Write, 3, (9:30-11:30)}	{Write, 3,9:30-11:30}

When “Private Hospital” accessing “Research Data”, it must receive the consent from “Genetics Research Company” or “Pharmaceutical Company”. Otherwise, he alone cannot achieve the predefined access requirement, namely (6,2). That is to say, this method can guarantee the right-to-know and superintendence right of coalition members in access implementation.

4 Formulation of the Fine-Grained Coalition Access Control Policy

4.1 Basic elements of the fine-grained coalition access control policy

We begin by identifying elements used in the fine-grained coalition access control policy that correspond to parts of the real system to be listed as follows:

$Users = \{u_1, u_2, ..., u_m\}$: *Users* represents all the users belonging to autonomous domains that form a coalition.

$X = \{r, w, e, a\}$: X is the set of access attributes, and its elements correspond to read, write, execute, and append respectively.

$TV = \{t_i \mid i \in N\}$: TV is the set of virtual time in computer system, and $t_i \in TV$ is the time of system running. For convenience, we define a function $real\_time(t_i)$. This function translates virtual time t_i into real time.

Based on the relationship between virtual time and real time, we can define relationship ‘<’ on virtual time as follows:

$$\forall i, j \in N, \forall t_i, t_j \in TV, t_i < t_j \Leftrightarrow real\_time(t_i) < real\_time(t_j)$$

$T=seq(TV)$ is a virtual time sequence composed of elements of TV, and the order of elements of T is strictly increasing. Namely: $\forall i, j \in N, t_i, t_j \in T, i < j \Leftrightarrow t_i < t_j$ [7].

$TSP = \{(t_i, t_j) \mid t_i, t_j \in T, i < j\}$: TSP is a time span defined by beginning time t_i and ending time t_j [4].

Coalition Resources $=\{o_1,o_2,...,o_l\}$: Coalition resources (directories, files, programs, and I/O devices, etc.) are owned by multiple domains and are consequently jointly administered by all owner-domains[2]. That is, access to coalition resources must satisfy the quantitative permission requirement.

Domain: An entity of autonomous security policy administration that administers user access to domain resources (object, application, and services) e.g., an enterprise firm, a hospital, a university, a country etc[2]. It is an element of coalition. Each autonomous domain will typically have its own identity certificate authority (CA) for distributing and revoking identity certificates to users registered in that domain.

Joint Coalition Authority: A joint coalition authority (AA) is jointly set up by members of a coalition to administer jointly owned resources that are deemed essential for coalition operations[2]. No single domain should be able to unilaterally define and modify access policies of a jointly owned resource, namely the setting and updating of access control policies and the distribution and revocation of privileges for the resource, without consent of all other resource owner-domains. For joint administration of access policies, the public key K_{AA} of the coalition Attribute Authority is generated using the shared key generation algorithm resulting in private key shares that are distributed among all member domains (i.e., a n-of-n threshold sharing of the private key K_{AA}^{-1}). The shared RSA public-key generation algorithm[8,9] enables n domains to generate a modulus $N=pq$ and exponents e and d. At the end of the computation all domains are convinced that N is the product of two primes, however none of them know the factorization of N. The public exponent e is made public while d is shared among the domains in a way that enables m-out-of-n threshold signature generation. That is, m domains are able to issue a certificate without reconstructing the key d. Once the public-key K_{AA} has been generated, all domains must apply a joint signature algorithm with their private key shares in order to sign any object with the private key K_{AA}^{-1}. The joint signature algorithm involves the requestor (one of the domains) sending a message to all the co-signers (the remaining member domains) with the message M to be signed and a key ID comprising the hash of N and the public exponent e. Each of the co-signers then applies their corresponding private key shares d_i to compute $S_i=M^{di}\ mod\ N$ and send the computations back to the requestor. The requestor then computes the message signature $S=\prod_{l=1}^{l} S_i\ mod\ N$. This joint signature protocol is illustrated in Ref.[10]. Using this joint signature algorithm, the domains sign attribute certificates distributed by the coalition AA.

Meta-Permission= $\{(x,n,tsp)\mid x\in X,n\in N,tsp\in TSP\}$: Mata-permission is a share of access permission to coalition objects and is the unit that can be authorized to users of different member domains. $x\in X$ is the permission's quality, $n\in N$ is its quantity, which reflects the domains' reliability and strength in coalition, and $tsp\in TSP$ is the permissible access time span about using this meta-permission.

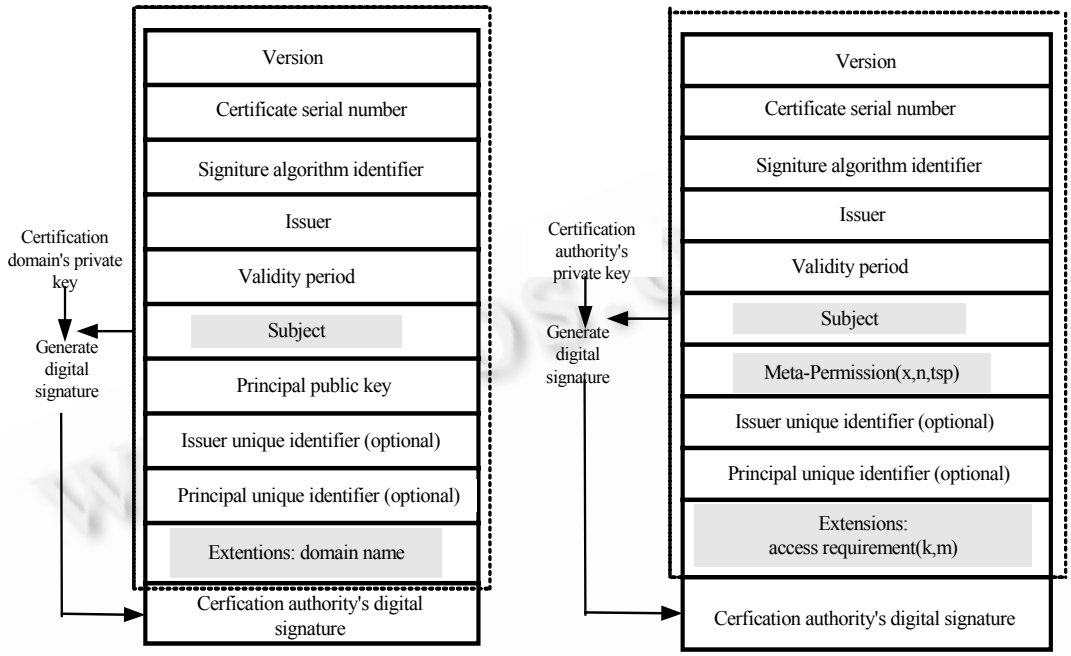
Access Requirements= $\{(k.m)\mid k,m\in N\}$: Access requirement respectively stipulates the threshold for the permission's quantity, namely k, and the number of participants, namely m.

In general, the administration of resources uses identity certificate authorities for authentication purposes and attribute authorities for authorization purposes.

Identity Certificate: Since each domain will have its own policies for registering users and issuing identity certificates, it is impractical for the coalition to establish a coalition identity CA for registering all coalition users and issuing them identity certificates. Each autonomous domain typically has its own identity certificate authority (CA) for distributing and revoking identity certificates to users registered in that domain. All coalition application servers (AA, P) must trust each domain's pre-established identity CA for distributing identity certificates to users of that domain[1,11]. Figure 1(a) illustrates the contents of identity certificate.

Attribute Certificate: An attribute certificate is a structured data containing attributes and values that certify properties of the certified owner made by the certificate issuer. In our policy, the attribute certificates are generated

by the coalition AA to coalition users for accessing jointly-owned resources[1,11]. Mata-permission, access requirement and user name are included in the format of certificate. In order to control over issuance of attribute certificate, all attribute certificates must be distributed by the coalition AA and must be signed by the coalition AA's private key, which is split among the member domain administrators to generate a joint digital signature. Figure 1(b) illustrates the details of attribute certificate.



(a) X.509 identity certificate (b) X.509 attribute certificate

Fig.1

Joint Access Request: A joint access request includes the access object *o*, the access manner *x*, all the participants' attribute certificates, and the requesting time.

Coalition Server: An Coalition Server P has the responsibility of controlling access to jointly-owned resources, receives joint access requests and checks whether the user's attribute certificate is valid. Coalition Server trusts the coalition authority *AA* for distributing attribute certificates.

$Sessions = \{se_1, se_2, ..., se_v\} : \rho(\rho(Users) \times Coalition \,\mathrm{Re}\, sources \times X \times TV)$: Session is a dynamic concept during coalition system running. For an element $se_i = \{\rho(Users), o, x, tv\} \in Sessions$, se_i describes a team of participants, namely $\rho(Users)$ involved in making joint access request *x* at the time of *tv* on coalition object *o*.

4.2 Extending classical access control matrix

The basic element of access control is the Access Control Matrix[12]. After introducing the concept meta-permission and time span, we extend the classical access control matrix to quantification access matrix (QM) and access requirement matrix (ARM). QM is recorded in the coalition AA and ARM is recorded in the coalition server P.

1. Definition of Quantification Access Control Matrix (QM)

The item $qm_{i,j} \in QM$ is meta-permission with its format being (x,n,t_i,t_j).

2. Definition of Access Requirement Matrix (ARM)

Access Requirement Matrix is a record-keeping matrix which remembers, for each possible (*x*,*o*)-pairs, the permission threshold and participant member number. The formal definition is $ARM = \{arm_1, arm_2, ..., arm_t\}$: $\rho(N \times N)$ and every item $arm \in ARM$ has the form of (*k*,*m*).

4.3 Valid session

Informally speaking, a session is a particular instance of a connection between a team of users in the system during their permissible time span. For coalition access, we can define the process of negotiating an access to jointly-owned resources as a session because coalition access requires multiple users to participate and form a joint access request to Coalition Server P, and executing request can be deemed as a session for Coalition Server P.

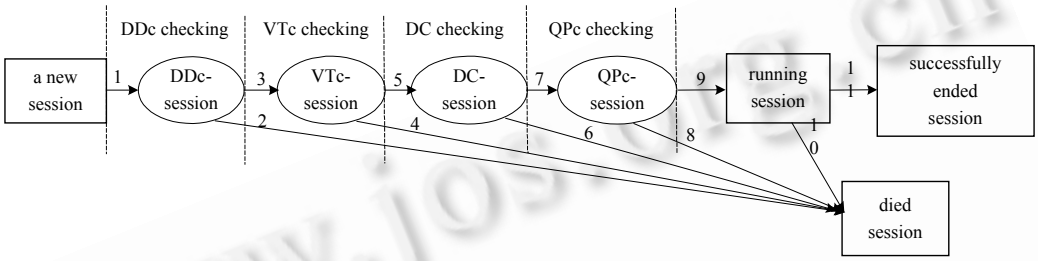


Fig.2 The transition of a session

A valid session $se = \{\rho(Users, o, x, tv)\}$ should meet the following fine-grained constraints at the same time. Details about validating a session are illustrated in Fig.2.

For every $u \in \rho(Users) \in se, o \in se$ and $x \in se$:

(1) Different Domain constraint (DDc): For all the users $u \in se$, they must come from different coalition domains.

(2) Valid Time constraint (VTc): The joint access requesting time, namely *tv*, falls into every participant permissible time span which is kept in their attribute certificates.

(3) Discretionary Constraint (DC): The access manner *x* included in joint access request must be the same as in every participant attribute certificate.

(4) Quantitative requirement of Permission constraint (QPc): The sum of participants' meta-permission value (namely *n*) must be more than that in access requirements (namely *k*). Furthermore, the number of participants involved in this joint access request must also be more than that in access requirements, namely *m*.

Coalition Server P to guarantee that it is a valid session before putting into execution must strictly check a new session created by a set of users. A session may experience several states from its successful start to successful finish, or session termination.

4.4 An example of joint access to coalition resources

In this subsection we present two different methods of negotiating a joint access for writing access to Object *O* based on the scenario illustrated in the beginning of this paper. In order to avoid replay attack, nonce is applied into requests and responses.

Before presenting this example, we introduce the architecture proposed in Ref.[1] and expand it for our policy implementation. In this architecture, each autonomous domain typically has its own identity Certification Authority (CA) for distributing and revoking identity certificates to users registered in that domain. For the access to jointly owned resources (managed by coalition server P), the domains jointly establish a coalition authority called the coalition Attribute Authority (AA). All attribute certificates distributed by the coalition AA must be signed by the

coalition AA's private key which is split among the member domain administrators to generate a joint digital signature and its public key is released to all domain users. The coalition AA operates effectively by distributing attribute certificates to coalition users granting them meta-permissions for accessing resources. Furthermore, all coalition application servers (AA, P) must trust each domain's pre-established identity CA for distributing identity certificates to users of that domain. Figure 3 illustrates the details about the access architecture based on quantifying permission idea.

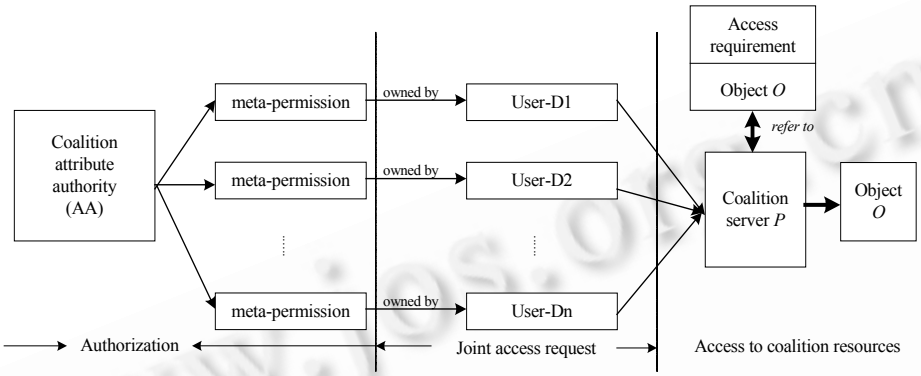


Fig.3 Access architecture based on quantifying permission idea

In our case, the process of negotiating a joint access proceeds as follows: the coalition AA distributes attribute certificates to coalition users User_D1, User_D2 and User_D3 granting them meta-permissions for accessing to Object *O*; When User_D1 needs to write Object *O*, consent must be received from User_D2 or User_D3, namely obtaining the response signed by its private key. After getting the response, a joint access request is formed and a new session is created. User_D1 can consequently get access to Object *O* by sending joint access requests with the attribute certificates to Server P (Server P trusts the coalition AA for distributing attribute certificates). Coalition Server P has the jurisdiction whether the session meets the DDc, VTc, DC and QPc at the same time. Figures 4(a)-(c) show the two different negotiating processes.

$$AA \to U_1: \ [(write, 5, (8:00-11:00)),\ (6,2)\ \ of\ \ (U_1,U_2,U_3)\ \ can\ write\ Object\ O]_{K_{AA}^{-1}}$$
$$AA \to U_2: \ [(write, 3, (9:00-11:30)),\ (6,2)\ \ of\ \ (U_1,U_2,U_3)\ \ can\ write\ Object\ O]_{K_{AA}^{-1}}$$
$$AA \to U_3: \ [(write, 3, (8:30-11:30)),\ (6,2)\ \ of\ \ (U_1,U_2,U_3)\ \ can\ write\ Object\ O]_{K_{AA}^{-1}}$$

(a) *AA* distributes *AC* for writing operation about Object O

$$U_1 \to U_2: request: [write\ \ object\ \ o, 5, (8:00-11:00), N_1, NonceValidity]_{K_{u1}^{-1}}$$
$$U_2 \to U_1: response: [write\ \ object\ \ o, 3, (9:00-11:30), N_1', NonceValidity, N_2]_{K_{u2}^{-1}}$$
$$U_1 \to P: request: [write\ \ object\ \ o, 5, (8:00-11:00), N_1, NonceValidity]_{K_{u1}^{-1}},$$
$$[write\ \ object\ \ o, 3, (9:00-11:30), N_1', NonceValidity, N_2]_{K_{u2}^{-1}}$$

(b) Case 1 *Nonces* generated by U_1 and U_2

$$U1 \to P: request: [write\ \ object\ \ o, 5, (8:00-11:00), U_2]_{K_{u1}^{-1}}$$
$$P \to U_2: request: [U_1\ \ request\ \ write\ \ object\ \ o, 5, (8:00-11:00), nonce]_{K_P^{-1}}$$
$$U_2 \to U_1: response: [write\ \ object\ \ o, 3, (9:00-11:30), nonce]_{K_{u2}^{-1}}$$
$$U_1 \to P: request: [write\ object\ o, 5, (8:00-11:00)]_{K_{u1}^{-1}}, [write\ object\ o, 3, (9:00-11:30), nonce]_{K_{u2}^{-1}}$$

(c) Case 2: *Nonce* generated by coalition server *P*

Fig.4

Case 1. Nonces generated by U_1 and U_2: N_1 and N_2 are two nonces generated respectively by U_1 and U_2, and NonceValidity stipulates their valid period. Server P maintains *NonceList* and it records all the received nonces within their valid period and their validity period. When receiving a joint access request, Coalition Server P adds N_1, N_2, and *NonceValidity* into *NonceList* if they have not appeared in *NonceList*. Otherwise, P will reject this request because it is a replay attack. When the nonce validity period end coming, Server P will delete them from the *NonceList* in time. In doing so, this can avoid replay attack. Note that using only nonces can also prevent replay attack occurring, but using nonces without valid period has drawbacks such as keeping nonces forever. This would lead to lower searching performance in *NonceList* because too many nonces kept in it.

The *NonceValidity* field is represented as the time at which the nonce validity period begins (notBeforeTime), and the time at which the nonce validity period ends (notAfterTime). Figure 5 shows its details. For example, we may prescribe "*notAfterTime=notBeforeTime+a day*".

NonceValidity ::= { notBefore Time(access requesting time),
notAfter Time }

Fig.5 *NonceValidity* field

Case 2. Nonce Generated by Coalition Server P: Unlike Case 1, Coalition Server P will generate nonces and they are kept in *GeneratedNonceList*. When receiving a joint access request, Coalition Server *P* will delete the nonce in *GeneratedNonceList* if the nonce is found in it. Otherwise, P will reject this request because it is a replay attack. If U_2 do not approve U_1's request, he will send the following response to P, and P will inform U_1 about it.

$$U_2 \to P : response : \left[U_1 \ \ request \ \ write \ \ object \ \ o, 5, (8:00-11:00), nonce, 'no'\right]_{K_{u2}^{-1}}$$

$$P \to U_1 : response : \left[''your\ request\ has\ been\ requested!''\right]_{K_P^{-1}}$$

If *P* does not receive any joint access request including the generated nonce within a specified period such as a day, P will also delete it from GeneratedNonceList. This may have the message lost.

Similar to Case 1 above, this case can also prevents replay attack occurring. Furthermore, this case requires only one nonce, but Coalition Server P bears more burden during negotiation.

Then Coalition Server P proceeds as follows:

(1) On receiving U_1's request message, Coalition Server P will generate a nonce and write it into *GeneratedNonceList*, and transmit U_1's request to U_2 with this fresh nonce.

(2) (Case 2 Method) When receiving a joint access request, Coalition Server P will delete the nonce from *GeneratedNonceList* if the nonce involved in this request is found in it. Otherwise, P will reject this request because it is a replay attack.

(3) U_1 and U_2 come from different domains: $U_1 \in Domain_1$(genetics company), $U_2 \in Domain_2$(private hospital).

(4) This access requesting time, assuming 10:00, falls into their common permissible time span, (8:00-11:00)∩(9:00-11:30), namely 9:00-11:00.

(5) With reference to the meta-permission included in their attribute certificates respectively, Server P affirm that both U_1 and U_2 have the "write" privilege to "object *o*" and their meta-permission value is 5 and 3 respectively.

(6) Coalition Server P finds the write access requirement of *o* being (6,2) with reference to ARM. Therefore, Server P approves the write request because 5+3>6 and the number of participants is equal 2.

Then this session is put into execution until it successfully finishes or terminates due to some exception errors.

5 Security Property Preserving about the Fine-Grained Access Policy

In this section, we will prove that the fine-grained joint access control policy preserves security with respect to state transition if, and only if, every running session in system is a valid session.

Coalition System Security is defined as meeting all the constraints of DDc, VTc, DC and QPc under the fine-grained joint access control policy.

5.1 Definition of coalition state and secure state

A coalition system state V is an element of the set $\rho(Sessions \times ARM \times QM)$.

D={'*ok*','*error*','?'} refers to achieving the predefined access requirement, failing to meet, and some unknown exceptions respectively such as not finding the appropriate access requirement in *ARM*.

Let N be the set of positive integers. X is defined as the set of all *request sequences* (R): the set of all functions from N to R; Y is defined as the set of all *decision sequences* (D): the set of all functions from N to D; Z is defined as the set of all *state sequences* (V): the set of all functions from N to V. Each of X, Y, and Z represents a set of sequences of successive values of requests, decisions, or states.

Let $x \in X$, $y \in Y$ and $z \in Z$, and $z_0 \in Z$ be the initial state. A *coalition system* $\sum(R,D,Z,z_0)$ is a subset of the cross-product $X \times Y \times Z$ such that $(x,y,z) \in \sum(R,D,Z,z_0)$ only if the above three sequences are consistent. That is to say, starting from state z_0, the inputs x results in the decisions y and a progression to the state z. The system $\sum(R,D,Z,z_0)$ includes all possible executions that start from state z_0.

Let $\rho = \{\rho_1,...,\rho_s\}$ be the set of transitions of sessions. The relation W is the set of state transitions and is defined for any $r_k \in R$, $d_m \in D$, set of states V and set of next-states V' as:

(a) $(r_k,d_m,V',V) \in W$ if and only if $d_m \neq$"?" and $d_m \neq$"error";

(b) $(d_m,V') = \rho_i(r_k,V)$ for a unique i, $1 \le i \le s$

A state $z \in V$ is a *secure state* only if the constraints such as DDc, VTc, DC and QPc are satisfied when accessing to coalition resources. An equivalent definition of secure state using session is that $se \in z$ is a valid session.

A state sequence $Z=\{z_1,z_2,\ldots,z_i,\ldots\}$ is a *secure sequence* only if z_i is a secure state for each i.

A system $\sum(R,D,Z,z_0)$ is a secure system only if every appearance (x,y,z) is a secure one.

5.2 Preserving Coalition System Security Based on Valid Session

Let $W = \left[R_i, D_j, (Sessions', ARM', QM'), (Sessions, ARM, QM)\right]$ and for every (o,x)-pairs in *Sessions* and (o',x')-pairs in Sessions', *arm*′ be the item that is referred to ARM by (o',x') while arm is that referred to ARM by (o,x), QM shares the same form.

Theorem. For any initial state z_0 that is a valid session, a coalition system $\sum(R,D,Z,z_0)$ preserves system security if, and only if W satisfies the following fine-grained constraints for each transition: if starting state z_0 is a valid session and the state transition meets DDc, VTc, DC and QPc, all the reachable coalition states are secure states:

(1) If $(\rho(Users),o,x,tv) \in (Sessions' - Sessions)$, then every participant comes from different coalition domains, x is included in every participants' attribute certificate, the sum of their meta-permissions value and the number of participants are more than the predefined permission requirement, namely (k,m), and the joint access request time tv falls into their common permissible time span.

(2) For each $(\rho(Users),o,x,tv) \in Sessions$ that does not satisfy the above conditions, we have $(\rho(Users),o,x,tv) \notin Sessions'$.

Proof. Pick $(x,y,z) \in \sum(R,D,Z,z_0)$ and write $z_i=(Sessions_i,ARM_i,QM_i,F_i)$ for each i.

[IF]:

Assume (x_1,y_1,z_1,z_0) is in W. The proof proceeds by showing that if z_0 is secure then z_1 must be secure, and by induction it shows that the system is secure.

Notice that $Sessions_1=(Sessions_1-Sessions_0)\cup(Sessions_1\cap Sessions_0)$, and $(Sessions_1-Sessions_0)\cap(Sessions_1\cap Sessions_0)=\varnothing$.

By (1), every $(\rho(Users),o,x,tv)\in(Sessions_1-Sessions_0)$ satisfies the DDc, VTc, DC and QPc.

Suppose $Sessions^*=\{(\rho(Users),o,x,tv)\,|\,(\rho(Users),o,x,tv)\ does\ not\ satisfy\ the\ DDc,VTc,DC\ and\ QPc\}$. By (2), we have $(Sessions^*\cap Sessions_1)=\varnothing$. But if $Sessions^*$ is in both $Sessions_0$ and $Sessions_1$ namely it is in $Sessions_1\cap Sessions_0$, $Sessions^*\cap(Sessions_1\cap Sessions_0)=(Sessions^*\cap Sessions_1)\cap Sessions_0)=\varnothing$.

Hence if $(\rho(Users),o,x,tv)\in(Sessions_1-Sessions_0)$, then $(\rho(Users),o,x,tv)\notin Sessions^*$ as hypothesized.

According to the above two discussions, $(\rho(Users),o,x,tv)$ must satisfy the DDc, VTc, DC and QPc since every $(\rho(Users),o,x,tv)$ is either in $(Sessions_1-Sessions_0)$ or in $(Sessions_1\cap Sessions_0)$. We have shown that z_1 must be a secure state with respect to the fine-grained constraints.

By induction on N, z_i is secure so (x,y,z) is a secure appearance. Since (x,y,z) is arbitrary the coalition system $\Sigma(R,D,Z,z_0)$ is a secure coalition system.

[ONLY IF]: Proof by contradiction

A contradiction of the theorem results in the proposition that:

There is a transition (x_i,y_i,z_i,z_{i-1}) such that either:

(a) Some $(\rho(Users),o,x,tv)\in(Sessions_i-Sessions_{i-1})$ that does not satisfy the fine-grained constraints for ARM_i, or

(b) Some $(\rho(Users),o,x,tv)\in Sessions_{i-1}$ that does not satisfy the fine-grained constraints for ARM_i but is in $Sessions_i$.

Suppose (a):

then there is some $(\rho(Users),o,x,tv)\in Sessions_i$ that do not satisfy the fine-grained constraints for ARM_i, since $(Sessions_i-Sessions_{i-1})\subseteq Sessions_i$.

Suppose (b):

then there are some $(\rho(Users),o,x,tv)\in Sessions_i$ that do not satisfy the fine-grained constraints for ARM_i, by the statement of (b).

Therefore, the fine-grained constraints are not preserved by the stated transition, and this contradicts the assumption that the system $\Sigma(R,D,Z,z_0)$ is a secure system.

6 Concluding Remarks and Future Work

The fine-grained access control policy is derived from comprehensively exploring permission's attributes such as permission quality and quantity, and permissible time span of permission usage. The distributed meta-permission enables to flexibly build trust in access requirements for jointly-owned resources. Besides the requirements such as permission threshold and participant member number, we may add another constraint that requires every participants' meta-permission value should be more than a predefined number for very essential jointly-owned resources, and this can be easily achieved by expanding the access requirement from the format (k,m) to $(k,m,mini\text{-}value)$. Hence, it solidifies and enriches the access control policies proposed in Refs.[1,2] to provide sufficient protection for jointly-owned coalition resources.

Although this fine-grained access policy enables joint access control to coalition resources and minimizes trust liabilities, several problems to our work remain to be solved as discussed in Ref.[1]. Firstly, it still requires

re-keying the Attribute Authority when some member joins or leaves. Secondly, coalition server P cannot judge the joint whether access request is a collusion among users from different domains.

Acknowledgement We are very grateful to Professor Wu Chuan-Kun and Dr. Paige Best for their technical and financial support. We are also very grateful to Dr. Zhou Yong-Bin and Zhuangyong for their help. Their guidance has been crucial to the completion of this paper.

References:

[1] Khurana H, Gligor V, Linn J. Reasoning about joint administration of access policies for coalition resources. In: Proc. of the Int'l Conf. for Distributed Computer Systems. Austria (Vienna): IEEE Computer Society, 2002. 429−440.

[2] Khurana H. Negotiation and Management of Coalition Resources [Ph.D. Thesis]. University of Maryland, 2002.

[3] Chopra K, Wallace WA. Trust in electronic environments. In: Proc. of the HCISS-36. Hawaii: IEEE Computer Society, 2003.

[4] Shands D, Yee R, Jacobs J, Sebes EJ. Secure virtual enclaves: Supporting coalition use of distributed application technologies. In: Proc. of the Network and Distributed Systems Security Symposium. San Diego: Internet Society, 2000. 187−202.

[5] Gibson T. An architecture for flexible multi-security domain networks. In: Proc. of the Network and Distributed Systems Security Symposium. San Diego: Internet Society, 2001.

[6] Thompson M, Johnston W, Mudumbai S, Hoo G, Jackson K, Essiari A. Certificate-Based access control for widely distributed resources. In: Proc. of the 8th USENIX Security Symposium. Washington: USENIX Association, 1999.

[7] Huang J, Qing SH, Wen HZ. Timed role-based access control. Journal of Software, 2003,14(11):1944−1954 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/1944.htm

[8] Boneh D, Franklin M. Efficient generation of shared RSA keys. Advances in Cryptology-Crypto'97. LNCS 1233, Springer-Verlag, 1997. 425−439.

[9] Malkin M, Wu T, Boneh D. Experimenting with shared generation of RSA keys. In: Proc. of the Internet Society's Symp. on Network and Distributed System Security. San Diego: Internet Society, 1999. 43−56.

[10] Wu T, Malkin M, Boneh D. Building intrusion tolerant applications. In: Proc. of the 8th USENIX Security Symp. Washington: USENIX Association, 1999. 79−91.

[11] Linn J, Nystrom M. Attribute certification: An enabling technology for delegation and role-based controls in distributed environments. In: Proc. of the 4th ACM Workshop on RBAC. Virginia: ACM Press, 1999. 121−130.

[12] Harrison MA, Ruzzo WL, Ullman JD. Protection in Operating Systems. Communications of the ACM, 1976,19(8):461−471.

附中文参考文献:

[7] 黄建,卿斯汉,温红子.带时间特性的角色访问控制.软件学报,2003,14(11):1944−1954. http://www.jos.org.cn/1000-9825/14/1944.htm