Information Sciences 180 (2010) 3086–3107

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A secure collaboration service for dynamic virtual organizations

Jianxin Li [a,*], Jinpeng Huai [a], Chunming Hu [a], Yanmin Zhu [b]

[a] School of Computer Science & Engineering, Beihang University, Beijing, China
[b] Department of Computer Science & Technology, Shanghai JiaoTong University, Shanghai, China

ARTICLE INFO

Article history:
Received 7 July 2009
Received in revised form 4 May 2010
Accepted 8 May 2010

Keywords:
Virtual organization
Distributed computing
Security policy
Secure collaboration
Privacy
Policy evaluation

ABSTRACT

Nowadays, various promising paradigms of distributed computing over the Internet, such as Grids, P2P and Clouds, have emerged for resource sharing and collaboration. To enable resources sharing and collaboration across different domains in an open computing environment, virtual organizations (VOs) often need to be established dynamically. However, the dynamic and autonomous characteristics of participating domains pose great challenges to the security of virtual organizations. In this paper, we propose a secure collaboration service, called PEACE-VO, for dynamic virtual organizations management. The federation approach based on role mapping has extensively been used to build virtual organizations over multiple domains. However, there is a serious issue of potential policy conflicts with this approach, which brings a security threat to the participating domains. To address this issue, we first depict concepts of implicit conflicts and explicit conflicts that may exist in virtual organization collaboration policies. Then, we propose a fully distributed algorithm to detect potential policy conflicts. With this algorithm participating domains do not have to disclose their full local privacy policies, and is able to withhold malicious internal attacks. Finally, we present the system architecture of PEACE-VO and design two protocols for VO management and authorization. PEACE-VO services and protocols have successfully been implemented in the CROWN test bed. Comprehensive experimental study demonstrates that our approach is scalable and efficient.

1. Introduction

There is an increasing demand for resource sharing and cooperation to support complex business processes and agile applications. Many promising paradigms of distributed computing, such as grid computing, peer-to-peer computing, pervasive computing [29,30] and cloud computing [2], have recently emerged for resource sharing and aggregation across multiple administrative domains. A virtual organization (VO) [4,7] is a dynamic coalition of geographically dispersed resources and users from different domains, which are unified by a common goal. Even in a centralized cloud computing environment, more and more users tend to build virtual organizations to aggregate the capabilities of both private cloud resources (e.g., the resources provided by local enterprises) and public cloud resources (e.g., the resources provided by Amazon EC2 or S3) in order to achieve collaborating goals.

There is a great security challenge for such virtual organizations. First, collaborating domains may join and leave dynamically during a business collaboration process. Second, to build a virtual organization, a participating domain may be required

* Corresponding author.
E-mail addresses: lijx@act.buaa.edu.cn, lijx@buaa.edu.cn (J. Li), huaijp@buaa.edu.cn (J. Huai), hucm@act.buaa.edu.cn (C. Hu), ymzhu@stju.edu.cn (Y. Zhu).

0020-0255/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2010.05.014

to disclose sensitive policies. It has become a fundamental problem how to create a secure collaboration environment for virtual organizations.

We look at a motivating example as follows.

Example 1. A national disease research centre encounters an epidemic disease and is not able to treat it. Thus, it needs cooperation from several other hospitals. In this case, a virtual organization comprising the national center and the hospitals should be established for the temporal cooperation. However, it is a key issue to create security policies for the new VO based on the security policies of the local domains. At the same time, the security policies of a domain is concerning privacy and therefore the domain's autonomy must be retained while the users of every domain in this VO can access a wide range of special services.

A number of approaches to security management of virtual organizations have been proposed. These approaches can be classified into two categories: *general approach* and *federation-based approach*.

A *general approach* completely creates a new set of policies for the virtual organization, and assigns new identities or attributes to all users or services in the virtual organization. It is an easy-to-implement approach and has been adopted by most grid systems. This approach, however, cannot fully accommodate the dynamism of virtual organizations in which collaborators may join or leave frequently. This implies that the security policy for local domain users and services cannot be fully utilized, and it introduces a heavy management burden. It is overwhelming for the system to assign identities to all the potential users or services. Moreover, the policies of the virtual organization have to be updated whenever the access control policy of a domain resource changes.

A *federation-based approach* reuses the original security policies of participating domains by defining their trust relationships through identity mapping, role mapping or delegation policies. This is an efficient approach, but we have found that collaboration policies defined by this approach may have possible conflicts. Policy conflicts lead to a potential security threat to local domains. For example, a user with a lower privilege may gain a higher privilege through an identity mapping loop. In this paper, we illustrate some examples through the role-based access control (RBAC) model [22], where a role is associated with permissions.

Notation statement: a role is denoted by r, with or without subscript. If role r_{A2} is senior to role r_{A1}, this inheritance relationship is denoted by $r_{A2} \prec r_{A1}$. If a user u is a member of role r_{A1}, then u acquires the permissions of role r_{A2}. A role mapping policy is denoted by m. We also denote this hierarchy relation with a role mapping policy m: (r_{A1}, r_{A2}).

As shown in Fig. 1, there are two role mapping policies: m_2: (r_{C1}, r_{B1}) and m_4: (r_{B2}, r_{C2}) between domain C and domain B with a role hierarchy relation $r_{B2} \prec r_{B1}$. Based on these polices, we derive a new relation (r_{C1}, r_{C2}) which brings on a conflict with the original role hierarchy relation $r_{C1} \prec r_{C2}$ in domain C. Thus, the security of local policies in domain C will be violated.

Traditional federation systems have different assumptions on virtual organizations. For example, secure interoperation is merely used to coordinate existing polices among security domains. In a virtual organization, new roles and policies should be defined for common tasks. Similarly, the policy conflict problem also exists in a virtual organization using federation policies. Unfortunately, such a serious problem is neither recognized nor addressed by existing work. Next, let us consider the following example.

Example 2. A federation-based virtual organization scenario shown in Fig. 2. Domain A and domain B form a virtual organization VO. The administrator of this virtual organization defines a role hierarchy relation $r_{VO3} \prec r_{VO1}$ and a task policy m_1: (r_{B1}, r_{VO1}) which means r_{B1} in domain B has the permissions of r_{VO1}. In domain B, it has $r_{B1} \prec r_{B2}$, and also defines a mapping policy m_2: (r_{VO3}, r_{B2}).

As illustrated in Fig. 2, we can also derive a new relation (r_{B1}, r_{B2}), similar to the scenario in Fig. 1, through three policies m_1, m_2 and $r_{VO3} \prec r_{VO1}$. However, this relation also violates the role hierarchy relation of domain B. Thus, such conflicts should be detected during the creation of VO collaboration policies.

Several methods have been proposed for detecting policy conflicts in a federation-based VO management system. Some novel approaches [11,23] to deal with policy conflicts. The main idea is that all participating domains first submit their local

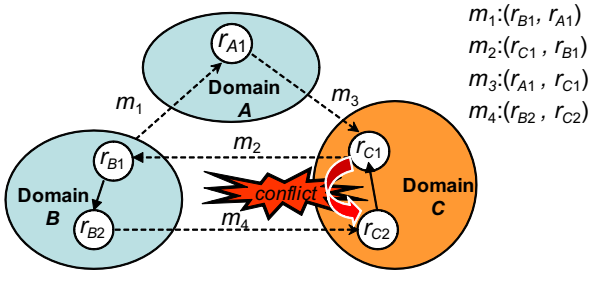


Fig. 1. Example for federation-based collaboration.

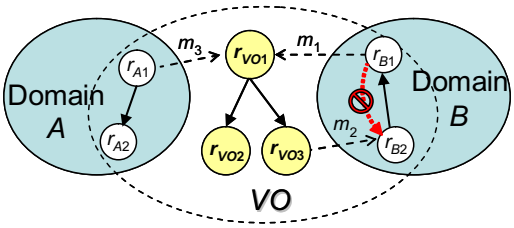


Fig. 2. Example of a policy conflict in a VO.

domain policies to a centralized server, and then the sever verifies security interoperation of policies. However, such approaches take few considerations on privacy preservation for domain security policies.

Preserving privacy for local domains in a collaboration is crucial in a distributed environment. The resources of a virtual organization are provided by all the local domains, so it is an essential requirement to preserve sensitive information of collaborated domains. It is similar to the classical millionaire problem in secure multi-party computation [1], in which some millionaires who want to know who is richer without revealing the precise amount of their wealth. In the virtual organization situation, several domains want to determine whether their collaboration in a virtual organization is secure while not revealing their local security policies. Moreover, traditional centralized evaluation approaches is vulnerable to an internal attack, and the malicious domain may submit subversive policies to the centralized server, and the security of a virtual organization collaborating policies would be compromised.

The scale of distributed computing systems is becoming larger, and the complexity of business applications is getting higher. There is an increasing demand for flexible and secure management for virtual organizations. It has been an urgent yet challenging issue to enable secure collaboration with privacy of local domains is preserved.

To this end, we propose PEACE-VO (Secure Policy-EnAbled Collaboration ServicE for Virtual Organizations). In PEACE-VO, we consider both security and privacy issues during collaboration, which clearly distinguishes our solution from existing approaches. We have made the following contributions.

1. After identifying the security issues for virtual organization management, we propose a novel secure collaboration service. Role mapping is employed to define trust relationship among domains. Two concepts, *implicit policy conflict* and *explicit policy conflict*, are introduced to describe violations against virtual organization collaboration policies.
2. In order to check the security of collaboration policies without disclosing the privacy information of participating domains, we design a fully distributed algorithm to detect any potential conflicts and hence ensure policy coherence. In addition, the complexity of this algorithm is analyzed. This algorithm has three advantages. Firstly, it is able to preserve critical domain privacy since it does not require domains to disclose their full security policies. Secondly, it is able to withhold malicious internal attacks since every domain is only responsible for its own security, and has no chance to submit fake policies to another domain. Thirdly, the overhead for security evaluation time and communication is modest.
3. We have implemented the PEACE-VO services and tools with two fundamental protocols, i.e., VO management protocol and service authorization protocol in the CROWN (China Research and development environment Over Wide-area Network)[1] [12] which is a service grid middleware system based on Web service standards. Comprehensive experimental study shows our approach is scalable and efficient.

The rest of this paper is organized as follows. We discuss related work in Section 2. Section 3 presents the design of PEACE-VO and the distributed policy evaluation algorithm. We detail the virtual organization management and authorization protocols in Section 4. The implementation of PEACE-VO is given in Section 5. Section 6 presents performance evaluation results and analysis. Finally, we conclude the paper in Section 7.

2. Related work

Security management approaches for virtual organizations have been widely studied. Existing approaches fall into two categories: *general approach* and *federation-based approach*. Representative systems using *general approach* include CAS (Community Authorization Service) [20], VOMS (Virtual Organization Management Service) [31] and TrustCoM [5] which are mainly designed for grid computing scenarios. Examples of the *federation-based approach* includes GridShib [33], CROWN-CredFed [15,18], Liberty [28], WS-Federation [19] and some delegation approaches [14]. In addition, secure interoperation [11,23,26,27] is an important direction introduced in the research area of secure collaboration of a multi-domain environment. In recent years, how to disclose access control policies to strangers is also an important research topic, and privacy preservation aims to reduce the risk of revealing policies to malicious requesters.

[1] CROWN Project, http://www.crown.org.cn.

2.1. VO security mechanisms

CAS and VOMS are two popular systems in service grids. CAS, which is built on the Globus Toolkit middleware based on WSRF (Web Service Resource Framework), allows service providers to delegate their authority to the *VO Server* whilst maintaining ultimate control over their services. In CAS, the polices of a service are generally composed by virtual organization policies and original domain policies, and the virtual organization policies must be agreed by all the participating domains. VOMS provides a similar approach for a virtual organization. These two systems are architecturally similar, and both of them issue attribute assertions to a user, then the user uses them to access a target service of a virtual organization. In addition, TrustCom[2] and GOLD[3] (Grid-based Information Models to Support the Rapid Innovation of New High Value-Added Chemicals) [21] also have developed similar security management mechanisms for virtual organizations.

GridShib[4] [33] is a project that integrates the Shibboleth infrastructure with the Globus Toolkit to provide attribute-based authorization for distributed scientific communities through identity federation. This project has been successfully used in campuses. WS-Federation is a specification to federate different security domains, such that authorized access to services managed in one domain can be provided to users whose identities and attributes are managed in other domains [19]. In CROWN 2.0, a CredFed service, which based on identity mapping and credential conversion approaches, is developed for building trust relationship among heterogeneous security domains such as PKI and Kerberos domains. After we found the problems in this federation-based approach, we have extended the functions of CROWN, and release CROWN v3 with the supporting of PEACE-VO approach.

In short, security management for a virtual organization has been widely studied. These approaches gain a performance advantage since a centralized authorization server is employed. However, they also suffer several limitations, primarily in two aspects: usability and security. For usability, the general approaches like CAS and VOMS are not flexible because they create a set of new policies and assign a new identity to every user. For security, current security systems using the federation-based approach are only concerned with how to build a collaboration relation, but rarely check for possible policy conflicts.

2.2. Secure interoperation

In a virtual organization, the collaboration need to define some new roles and policies for common tasks, and a centralized *VO Server* is used for membership assignment (i.e., VO permission assignment). Secure interoperation is a related work for secure collaboration, and it aims to coordinate existing polices among local domains.

Secure interoperation has the capability of guaranteeing collaboration security through identity mapping in a multi-domain environment [11,27]. Dawson et al. [23] present a mediator-based approach to provide secure interoperability for heterogeneous databases. This approach assumes a Mandatory Access Control (MAC) policy, such as the Bell LaPadula policy, but MAC is inflexible and inapplicable in many commercial applications. Gong and Qian [11] characterize the properties that must be satisfied to compose a global secure policy. In all these approaches, a third trusted party that has a global view is required to perform secure policy composition. Moreover, these approaches easily suffer the privacy leaking problem and internal attacks. To deal with these problems, Sheha et al. [27] propose a novel distributed secure interoperability framework for mediator-free collaboration, which relies on a secure access path to make authorization as well as to check conflicts without a global view of collaboration policies. This framework provides an effective coordination approach among fully decentralized domains. In our approach, a virtual organization will require new common policies and we employ a central server to guarantee the performance of role membership assignment during collaboration.

We further illustrate their differences with Fig. 3. There are two phases during collaboration. *Phase* 1 builds a collaboration relationship through policies definition, and *Phase* 2 establishes trust before task execution every time, i.e., authorization decision for resource requests. The two phases include three key steps, *Collaboration Policy Definition*, *Policy Verification* and *Trust Establishment*. In the mediator-based approach, the *Policy Verification* step (i.e., finding possible conflicts) is involved into *Phase* 1, and the security of collaboration polices will be verified in advance. On the contrary, the *Policy Verification* step is involved into *Phase* 2, in the mediator-free approach, and the access path construction process will verify the security of collaboration policies dynamically. In Fig. 3(c), we showed an AREM example powered by CROWN Grid, it is a virtual organization formed by six domains. During the creation of this virtual organization, the first step is to define collaboration policy (AREM Task), and then the policy verification ensures the collaboration is secure. When the secure collaboration policies are agreed, then job can be scheduled to specific computers (i.e., computing resources) after a step of trust establishment.

Some detailed differences (shown in Fig. 4) between the PEACE-VO and mediator-free method are showed as follows.

First, the application scenario is different. The mediator-free method has given an approach to build an access path dynamically without policy conflicts in a decentralized collaboration environment (e.g., a P2P system). Comparably, a virtual organization adopted in grids and hybrid clouds is agreed by all participating domains, and it aims to complete some common tasks (e.g., a scientific computing application) based on virtual organization task policies. In particular, a central

[2] TrustCom Project, http://www.eu-trustcom.com/.
[3] GoldProject, http://www.goldproject.ac.uk/.
[4] GridShib Project, http://gridshib.globus.org.

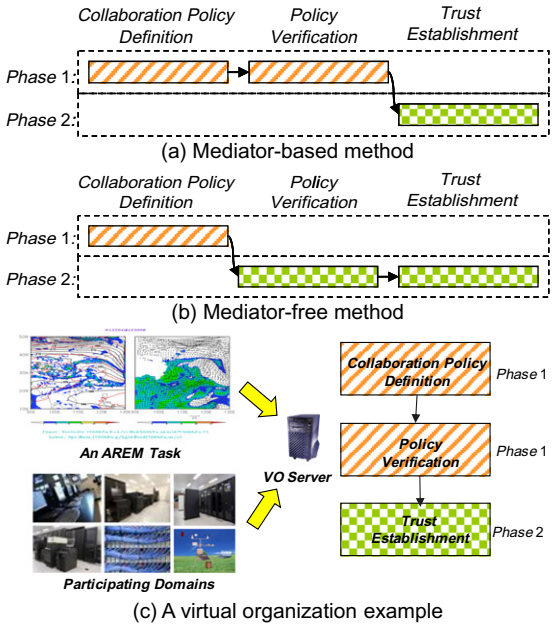


Fig. 3. Comparison for mediator-based and mediator-free method, and a virtual organization example. A collaboration generally includes two basic phases: *Phase* 1 builds a collaboration relationship through policies, and *Phase* 2 enforces this collaboration and access control. From another point of view, the collaboration policies include three key steps, *Collaboration Policy Definition*, *Policy Verification* and *Trust Establishment*. In a virtual organization example, the three steps are executed as the mediator-based method.

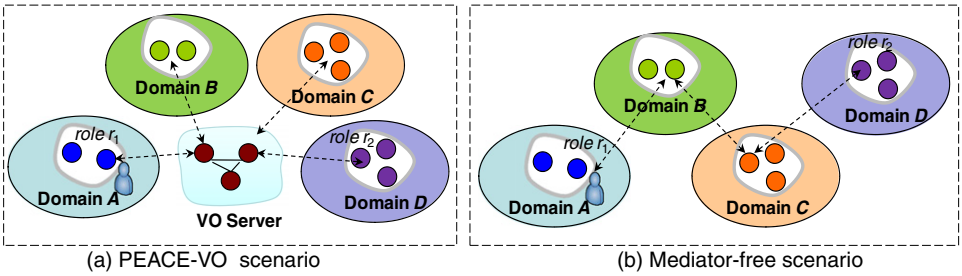


Fig. 4. Comparison for virtual organization and mediator-free scenarios.

membership server is employed in a virtual organization, and it is easy-to-deploy and has high performance. For example, EGEE gLite has used VOMS to set up a virtual organization, new roles and policies can be created by the VOMS administrator.

Second, policy definition is different. In a mediator-free method, every domain must know the roles of other domains to define their role mappings, but in PEACE-VO, every domain only need to know the VO task roles for defining role mappings. If the number of domains is large, the mediator-free method is not easy to be applied. This is because every domain cannot know all roles of other domains, and all related domains should update their policies when a domain joins or leaves (only the VO server updates policies in PEACE-VO).

Third, access procedure is different. In a virtual organization (an example shown in Fig. 4(a)), if a user of domain *A* wants to access the resources of domain *D*, the user will access the *VO Server* to get a credential before accessing the resources. Comparably, in a mediator-free method (an example shown in Fig. 4(b)), the user will access domain *B* and domain *C* before accessing domain *D*, it is de facto a dynamic access path building approach. There are also some similar research works such as SPKI/SDSI [3], RT [17], dRBAC [8]. In particular, RT also gives a distributed credential chain building algorithm which is similar to the access path construction algorithm in mediator-free method. However, the dynamic access path building method is mostly used to infrequent resource access among unfamiliar domains and it suffers from time-consuming problem, so it is inappropriate to the virtual organization management. In conclusion, these methods have different assumptions and features on virtual organization management. Compared with the fully decentralized collaboration scenario, a virtual organization is still a centralized collaboration scenario and seems easier for adoption in a real distributing computing environment. PEACE-VO makes use of some techniques from the research work on secure interoperation, and provides a new approach to deal with the policy conflict problem without disclosing domains' privacy policies.

2.3. Privacy protection for policy disclosing

Privacy protection during disclosing security policies is another hot research topic. ATN (automated trust negotiation) considers privacy with both credentials and access control policies. If a provider discloses the contents of an access control policy to a stranger, valuable business information may leak or one's privacy may be compromised [34]. For example, a collaboration organization has a policy granting a special service quality to employees of its business partners. If this policy is known to every requester for the service, then an outsider may know who is partnering with this business.

Seamons et al. [24,25] proposed a policy graph to organize the structure of a policy, only when the policy in the front node is satisfied, then the next privacy security policy can be disclosed. Yu and Winslett [35] proposed a UniPro framework which treats access control policies as first-class resources, and provides fine-grained control over policy disclosures. Such research work mainly presents how to design a policy disclosure approach to reduce the leakage of privacy policy to strangers, and they are useful during service authorization for strangers. In comparison, PEACE-VO is used to define secure collaboration policies for a virtual organization among participating domains, and it aims to not disclose the privacy policies of participating domains.

Besides, some cryptographic protocols are designed for privacy protection to sensitive credential or policy possession. For example, the fact that a user has or has not a certain credential is sensitive. Li et al. [16] proposed an OSBE (Oblivious Signature-Based Envelope) scheme to address this issue. Frikken et al. [9,10] presented some protocols that protect both sensitive credentials and sensitive policies. With these protocols, a user can access the resource only if it satisfies the policy, the provider does not learn anything about user's credentials (not even whether user got access), and the user learns neither provider's policy structure nor which credentials caused her to gain access. The typical protocols are generally used to two-party interaction scenarios (e.g., in the trust negotiation scenarios), and they are not feasible for multi-party collaborative policy verification in which each party is not willing to disclose its own sensitive information to other parties or the central *VO Server*.

3. Design of PEACE-VO

3.1. Basic concepts

We choose the RBAC model [22] to describe policies in PEACE-VO, i.e., all domains adopt a RBAC to model their security policies. RBAC is suitable for specifying security requirements on a wide range of commercial, medical, and governmental applications. In addition, MAC (Mandatory Access Control) and DAC (Discretionary Access Control) are special examples of policies to configure in a RBAC model.

The basic notations used in this paper are as follows:

- *Domains*: We use A, B, and C, sometimes with subscripts, to denote domains, and use VO to denote a virtual organization.
- *Roles*: A *role* is denoted by r with subscripts or not, e.g., a role in domain A_1 can be denoted by r_{A1}. R denotes a set of roles, and R_i denotes a role set of the ith domain. In a virtual organization, we need to define new roles for common tasks. We call the new roles as *task roles* of a virtual organization, which are denoted by symbol R_{VO}. The whole roles of a virtual organization include the roles of participating domains and task roles of this virtual organization, which is denoted by R'_{VO}.

As a VO example shown in Fig. 2, the role sets of domain A, B are $R_A = \{r_{A1}, r_{A2}\}$ and $R_B = \{r_{B1}, r_{B2}\}$, respectively. The task role of this VO is $R_{VO} = \{r_{VO1}, r_{VO2}, r_{VO3}\}$, and all roles of this VO is $R'_{VO} = R_A \cup R_B \cup R_{VO}$.

As shown in Fig. 5, PEACE-VO essentially forms an overlay of domain security policies, where virtual organization G_{VO} consists of several participating domains, each of which consists of a set of users and services. The policy manager *VO Server*

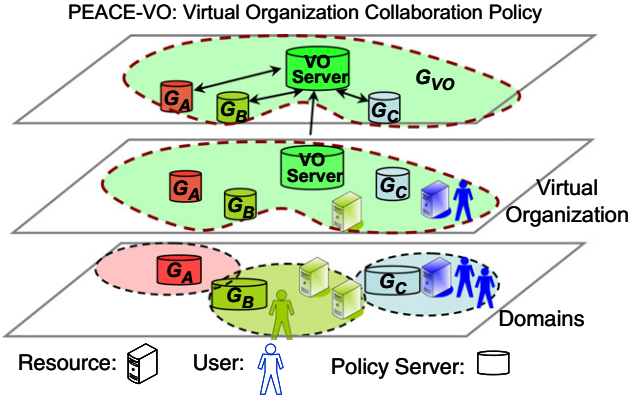


Fig. 5. The PEACE-VO organization vision.

for the new virtual organization can be established upon agreement by all domains through negotiation, or simply chosen by the virtual organization sponsor. All users, services and their trust relationships in the virtual organization are ultimately defined by collaboration policies. In this design, the central *VO Server* makes efficient service authorization for virtual organization users like CAS and VOMS.

Next, we formally define several important concepts in PEACE-VO.

We use the definition of domain security policies proposed in [27]. The domain security policies are defined as a directed graph $G = \langle R, H \rangle$, where R is a role set of a domain, and H is a set of role hierarchies, and it satisfies $H \subseteq R \times R$. The collaboration among domains is defined by role mapping policies. Let R_i and R_j be different role sets in two domains, and the role mapping policy is a binary relation M which is a subset of the Cartesian product $R_i \times R_j$, and it satisfies $\forall (r_p, r_q) \in M$, $r_p \in R_i$, $r_q \in R_j$ where $i \neq j$. Generally, the symbol m denotes an element of M. If $M_1 \subseteq R_i \times R_j$ and $M_2 \subseteq R_j \times R_k$ are two binary relations, then their composition $M_1 \circ M_2 = \{(r_p, r_q) \in R_i \times R_k | , \exists r_t \in R_j : (r_p, r_t) \in M_1 \wedge (r_t, r_q) \in M_2\} \subseteq R_i \times R_k$.

Cross-domain role mapping is a key approach to empower collaboration among domains [6]. Through role mapping policies, users belonging to a role in one domain can acquire permissions assigned to roles in another domain. It is observed that the expressions of binary relation elements in H and M are the identical, but we make two distinct definitions since the former concerns intra-domain collaboration relations, and the latter concerns inter-domain collaboration relations. This is not only much clearer for policy management, but also helpful to design the policy conflict detection. In PEACE-VO, there are two kinds of role mapping policy and one kind of forbidden role mapping policy.

- *Virtual Organization Role Mapping*: It includes the role mapping from a role of participating domains to a *task role* of the virtual organization, e.g., $(r_p, r_{vo}) \in \cup_{i=1}^{n} R_i \times R_{VO} \in M_{VO}$ (M_{VO} denotes a set of the VO role mappings).
- D*omain Role Mapping*: It includes the role mapping from a *task role* of the virtual organization to a role of participating domains, e.g., $(r_{vo}, r_q) \in R_{VO} \times R_i \in M_i$ (M_i denotes a set of domain role mappings).
- *Forbidden Role Mapping*: It is necessary to restrict that some roles in other domains could not be mapped to some appointed roles in a domain. We call such mapping policy as *forbidden role mapping* (denoted by F, sometime with a subscript). For example, a domain j defines a forbidden role mapping set F_j, and $\forall (r_p, r_q) \in F_j, \exists R_i$: $r_p \in R_i$, $r_q \in R_j$ where $i \neq j$. In some literatures [11,27], this category of policy is also called *autonomy policy*.

In PEACE-VO, if there is a violation of the collaboration policies against F, such a condition is referred to as *explicit policy conflict*. Correspondingly, if there is a violation of the collaboration policies against the role hierarchies H of a domain, such a condition is referred to as *implicit policy conflict*.

3.2. Virtual organization collaboration policies

Definition 1. [Virtual Organization Collaboration Policies]. Let VO be a virtual organization composed of n domains, the virtual organization collaboration policies are defined as $G_{VO} = \langle R'_{VO}, H'_{VO} \rangle$, where $H'_{VO} = (\cup_{i=1}^{n} (H_i \cup M_{Gi}) \cup H_{VO} \cup M_{VO}) - \cup_{i=1}^{n} F_i$, R_{VO} is a set of *task roles* of a virtual organization, H_{VO} is a set of role hierarchies on R_{VO}, and M_{VO} is a set of role mapping relations defined by the virtual organization, i.e., $(r_i, r_{VO}) \subseteq \cup_{i=1}^{n} R_i \times R_{VO}$, F_i is a set of *forbidden role mappings* defined by every participating domain G_i. The sets of R_{VO}, H_{VO} and M_{VO} together are called the *task policies* of a virtual organization.

In the following, we give an example to illustrate how the virtual organization collaboration policies are created. As shown in Fig. 6, two domains, A and B, form a virtual organization VO, and domain B wants to share its service s_{B1} with some users from domain A. In this virtual organization, the related policies are as follows:

1. In domain A, the user u_{A1} is a member of role r_{A1}.
2. In VO, role mapping policy set M_{VO} contains a policy m_1: (r_{A1}, r_{VO1}) which represents users belonging to role r_{A1} also acquire the permissions associated with role r_{VO1}, and there is also a role hierarchy $r_{VO2} \prec r_{VO1}$.
3. In domain B, role mapping policy set M_{GB} contains a policy m_2: (r_{VO2}, r_{B1}) which represents users belonging to role r_{VO2} also acquire the permissions associated with role r_{B1} in domain B, says that role r_{B1} has access permission to service s_{B1}.

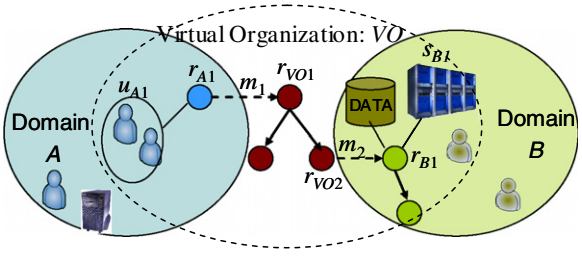


Fig. 6. An example of the virtual organization collaboration policies.

Table 1
The properties of virtual organization collaboration policies.

No	Policies	Property
1	Domain security policy	For a participating domain, its domain security policies G_i need not be disclosed to others when the security of virtual organization collaboration policies is evaluated.
2	VO task policy	In order to evaluate the security of virtual organization collaboration policies, the *task roles* and their *hierarchies, and VO role mappings* of a VO should be disclosed to participating domains.
3	Role mapping chain	To support the Property 1 and keep the privacy of a domain security policy, a role mapping chain has some validity conditions (defined in the following Definition 2).

As a result, based on these policies in the virtual organization, the user u_{A1} from domain A can access service s_{B1} provided by domain B.

When the virtual organization collaboration policies are made, how to evaluate their security motivates the design of PEACE-VO. From the previous analysis, we present the following properties that PEACE-VO should possess (shown in Table 1).

The Property 1 means that most of the local policies G_i of a domain, and its *domain role mapping policies* M_i and *forbidden role mapping policies* F_i need not be disclosed. Moreover, since the authorization policies for services are still associated with original domain roles, it is unnecessary to assign new policies to the shared services compared with general approaches for virtual organization security management.

The Property 2 requires that the *task roles* R_{VO}, their *hierarchies* H_{VO}, and the M_{VO} should be known by participating domains to support security evaluation of virtual organization collaboration policies compared with the privacy property of the domain security policies. The reasons are twofold. On the one hand, these policies should be available for domains to create their local domain role mapping policies M_{Gi}. On the other hand, these policies are generally created and agreed by all participating domains together, and hence it is reasonable to share these policies among domains.

The Property 3 requires that a role mapping chain should have some validity conditions so as to ensure the requirements of Property 1. Because the domain security policies may be invisible to other domains, the role mappings between two domains should not involve roles from a third domain. This means that role mapping policies M_i from *task roles* to domain roles are only valid for service authorization within this local domain G_i, and they cannot be a part of the role mapping chain for other domains because they may be completely undisclosed to other domains. Fig. 7 gives an example of a virtual organization formed by three domains A, B and C. It is easy to observe that the user u_{A1} belonging to role r_{A1} in domain A can be mapped to role r_{B1} in domain B. Following the relation $r_{B2} \prec r_{B1}$ and two mapping policies, m_3: (r_{B2}, r_{VO3}) and m_4: (r_{VO3}, r_{C1}), the role r_{B1} from domain B can be mapped to role r_{C1} in domain C. If the security policies of all domains are disclosed (e.g., in the mediate-based evaluation approaches), we can infer that the user u_{A1} belongs to role r_{C1}. However, the relation $r_{B2} \prec r_{B1}$ may be of privacy to domain B, so the user u_{A1} cannot be mapped to role r_{C1} without $r_{B2} \prec r_{B1}$. Therefore, a valid role mapping chain between A and C cannot involve policies from domain B. Moreover, a valid role mapping chain requires at least one *task role* of the virtual organization as a bridge, so trust relationship between two domains must be built through the VO task policies.

There are three main aspects to be considered for Property 3. Firstly, it provides a method to avoid a role mapping loop. If there are no restrictions for a valid role mapping loop, a policy conflict is impossible to be detected by a single domain because other domains may not disclose full policies for the purpose of privacy preservation. If this property does not hold, as the example showed in Fig. 7, the role r_{A1} can be finally mapped to r_{A2}, i.e., (r_{A1}, r_{A2}), which violates the original role hierarchy $r_{A1} \prec r_{A2}$. Secondly, the role mapping definition between domains is simplified, which will be helpful to design an efficient algorithm for policy conflict detection. Finally, mutual collaboration relationships are defined through the *task roles* of the virtual organization, which is an additional advantage for central security audit.

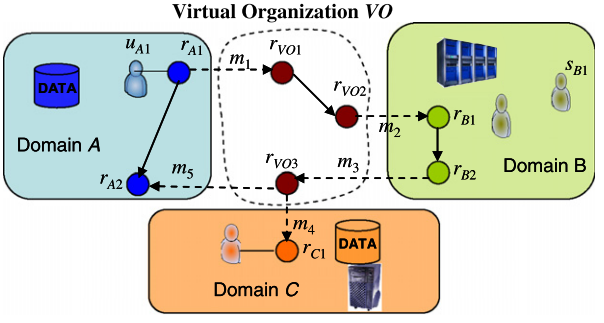


Fig. 7. A virtual organization VO formed by three domains A, B and C, where mapping policies m_1, m_3 belong to M_{VO}, mapping policies m_2, m_4, and m_4 belong to domain B, domain C and domain A, respectively.

The three properties of PEACE-VO are not only requirements of the security management for virtual organizations but also guidelines for evaluating the security of virtual organization collaboration policies. Based on Property 3 and the analysis mentioned above we give the definition of a valid role mapping chain.

Definition 2. [Valid Role Mapping Chain] In PEACE-VO, let $L = (r_0, r_1, \ldots, r_k)$ be a role mapping chain from role r_0 in domain G_i to role r_k in domain G_j. L is a valid role mapping chain if it satisfies the following conditions:

(C.1) For each $p \in \{0, \ldots, k\}$, s.t. $r_p \in R_i \cup R_j \cup R_{VO}$, and $\exists q$ s.t. $r_q \in R_{VO}$ where $q \in \{1, \ldots, k-1\}$.
(C.2) For every $p, q \in \{1, \ldots, k-1\}$ and $p \leqslant q$, if $r_p, r_q \in R_{VO}$, and $\exists s$: $p \leqslant s \leqslant q$, there is $r_s \in R_{VO}$.

This definition indicates that no additional roles from other domains can be involved in a valid role mapping chain, i.e., $(r_0, r_k) \in H_i^+ \circ M_{VO} \circ H_{VO}^+ \circ M_j \circ H_j^+$ (H^+ is a transitive closure set of H). As illustrated in Fig. 8, we further give the meaning of these two conditions. In Fig. 8(a), there are two invalid role mapping chains $L_1 = (r_{A1}, r_{C1}, r_{B1})$ and $L_2 = (r_{A1}, r_{B1})$ between domain A and B. As for L_1, the condition (C.1) cannot be satisfied due to $r_{C1} \notin R_A \cup R_B \cup R_{VO}$. As for L_2, the condition (C.1) cannot also be satisfied due to $\neg\exists r_q \in R_{VO}$. In Fig. 8(b), there is an invalid role mapping chain $L_3 = (r_{A1}, r_{VO1}, r_{C1}, r_{VO2}, r_{B1})$, and the condition (C.2) cannot be satisfied due to $r_{VO1}, r_{VO2} \in R_{VO}$ but $r_{C1} \notin R_{VO}$.

To ensure that a user can access the service through a valid role mapping chain, the original domain from which a user originate should be tracked in the authorization protocol. With the support of Property 3, if each $r_0 \in R_i$, $r_k \in R_j$ such that $(r_0, r_k) \notin F_k$ there is no *explicit policy conflict* in G_{VO}, and if each $(r_0, r_k) \in H'_{VO}, r_0, r_k \in R_i$ such that $(r_0, r_k) \in H_i$ there is no *implicit policy conflict* in G_{VO}. Therefore, if there is neither *explicit policy conflict* nor *implicit policy conflict*, the virtual organization collaboration policies are secure.

Definition 3. [Security of Virtual Organization Collaboration Policies] Let (r_0, r_k) be a role mapping policy through a valid role mapping chain in G_{VO}. The G_{VO} is secure if it satisfies the following conditions:

(C.1) For all $r_0 \in R_i$ and $r_k \in R_j$, there is $(r_0, r_k) \notin F_k$.
(C.2) For every r_0, $r_k \in R_i$, there is $(r_0, r_k) \in H'_{VO}$ if and only if $(r_0, r_k) \in H_i$.

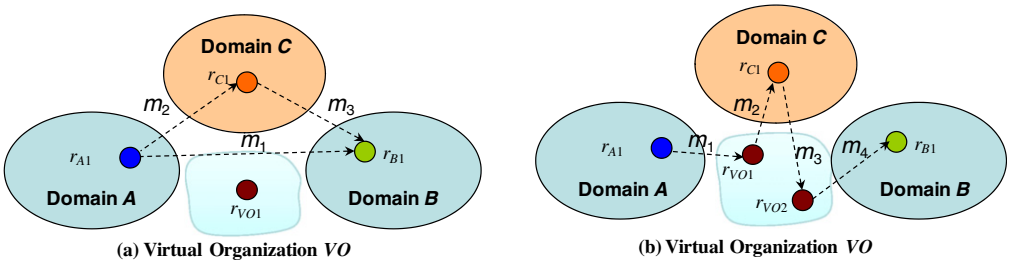
Now, the problem we encounter is how to decide whether the given virtual organization policies are secure. According to Definition 2, we can infer that $(\cup_{i=1}^n H_i)^+ = \cup_{i=1}^n H_i^+$ since there is no direct role mapping between any two domains. Each role mapping policy (r_0, r_k) where $r_0 \in R_i$, $r_k \in R_j$ in G_{VO} satisfies $(r_0, r_k) \in \left((\cup_{i=1}^n H_i)^+ \circ M_{VO} \circ H_{VO}^+ \circ M_j \circ (\cup_{i=1}^n H_i)^+\right)$. Therefore, if $(\cup_{i=1}^n H_i^+ \circ M_{VO} \circ H_{VO}^+ \circ \cup_{i=1}^n M_{Gi} \circ \cup_{i=1}^n H_i^+) \cap (\cup_{i=1}^n F_i) = \emptyset$, there is no *explicit policy conflict* in G_{VO}.

Theorem 1. *Let $G_{VO} = \langle R'_{VO}, H'_{VO} \rangle$ be the virtual organization collaboration policies, where every role mapping chain L in G_{VO} be a valid chain. The virtual organization collaboration policies are secure if and only if the evaluation result of each domain security policies G_i together with task policies $\langle R_{VO}, H_{VO} \rangle$ is secure.*

Proof. Firstly, it is obvious that security policies of each domain are secure if the virtual organization collaboration policies are secure, which implies that there is neither explicit policy conflicts nor implicit policy conflicts in each G_i.

Next, we proceed to complete the proof by contradiction, and assume that there is a case that the virtual organization collaboration policies are violated although each domain's evaluation result is secure. Without loss of generality, we assume there is a role mapping (r_1, r_3) where $r_1, r_3 \notin R_{VO}$ that violates the virtual organization collaboration policies. Then we only have two cases:

(i) $r_1 \in R_A$, $r_3 \in R_B$ that means the two roles belong to different domains. As shown in Fig. 9, there is at least a valid role mapping chain starting from a role of domain A, and ending at a role of domain B through *task roles* of *VO Server*, such that (r_1, r_3) violates the collaboration policies. By Definition 2, it is impossible that a role mapping forms a loop. Hence,



(a) Virtual Organization *VO* (b) Virtual Organization *VO*

Fig. 8. Examples of invalid role mapping policies in virtual organizations.

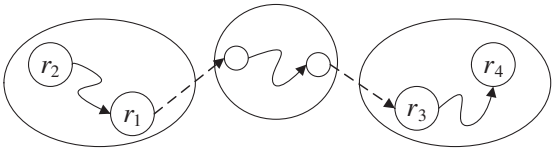


Fig. 9. The case of two roles in different domains.

there is only either $(r_1,r_3) \in F_B$ or $(r_2,r_4) \in F_B$, which implies that $(r_1,r_3) \in M_{VO} \circ H^+_{VO} \circ M_B$ or $(r_2,r_4) \in (H^+_A \circ M_{VO} \circ H^+_{VO} \circ M_B \circ H^+_B)$. As a result, domain B conclude that its domain security policies are not secure because of $(r_1,r_3) \in F_B$ or $(r_2,r_4) \in F_B$, which contradicts the assumption.

(ii) $r_1, r_3 \in R_A$ that means the two roles belong to the same domain. As shown in Fig. 10, there is at least a valid role mapping chain starting from a role of domain A, and ending at a role of domain A again through *task roles* of *VO Server*, such that (r_1,r_3) violates the collaboration policies. Because the starting and ending roles are both in the same domain, we can infer that there is an implicit policy conflict, i.e., $(r_1,r_3) \notin H^+_A$, and there is at least a valid role mapping chain as $L = (r_1, r_{VO2}, \ldots, r_{VO1}, r_3)$. Due to $(r_1,r_3) \in M_{VO} \circ H^+_{VO} \circ M_A$ and $(r_1,r_3) \notin H^+_A$, the domain A should also conclude that its domain security policies are not secure, which contradicts the assumption.

From (i) and (ii), we can conclude that the virtual organization collaboration policies are secure if all the evaluation results of the domains security policies together with task policies are secure. Therefore, this theorem is proved. □

This theorem plays a key role in securing collaboration in virtual organizations. Most importantly, not only does this theorem support the privacy preservation of domain policies, but also enables the policy evaluation algorithm to be implemented in a completely distributed fashion with greatly reduced execution time. As shown in Fig. 11, there is a virtual organization formed by three domains, and the security of its collaboration policies can be separated into three independent evaluation procedures which can be executed in parallel. The evaluated policies (the dashed box in Fig. 11) in every procedure only include domain security policies and VO task policies.

Next, we discuss how a domain evaluates its security policies to detect possible policy conflicts in the virtual organization. First, each domain creates a 2-dimensional matrix W_R, and assigns values according to its hierarchy policies H and role mapping policies M. The transitive closure set of W_R can be computed with the *Warshall* algorithm.

For example, as shown in Fig. 12, there is a role mapping (r_i,r_j). If there are also two role mappings (r_j,r_p) and (r_j,r_q) at the same time, then we can obtain two new role mappings (r_i,r_p) and (r_i,r_q). Finally, we can check whether there are policy conflicts based on the matrix W_R (shown in Fig. 13). The security policies that A has are $R_A = \{r_{A1}, r_{A2}, r_{A3}\}, H_A = \{(r_{A1}, r_{A2}), (r_{A2}, r_{A3})\}$, $M_A = \{(r_{VO1}, r_{A2})\}$, $F_A = \{(r_{B1}, r_{A2})\}$, $R_{VO} = \{r_{VO1}\}, M_{VO} = \{(r_{A3}, r_{VO1}), (r_{B1}, r_{VO1})\}$. The original matrix W_R that domain A generated is shown in Fig. 13(a), and the matrix W_R processed by the *Warshall* algorithm is shown in Fig. 13(b). Then it checks the W_R to determine whether there are policy conflicts (shown in Fig. 13(c)). If there are *non-zero* diagonal elements in the matrix W_R,

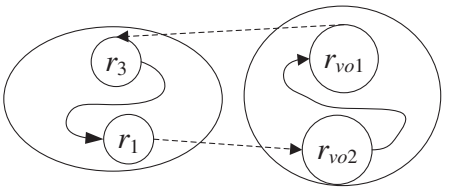


Fig. 10. The case of two roles in the same domain.

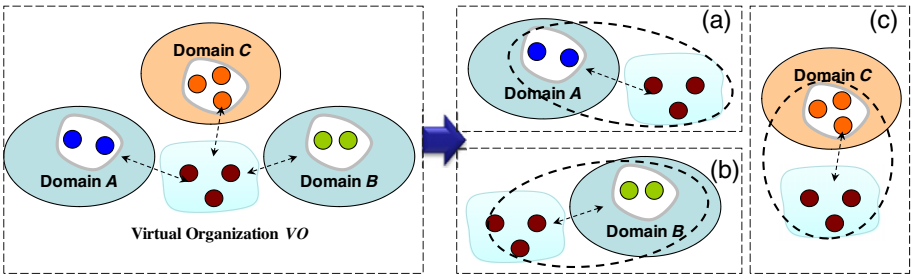


Fig. 11. The principal of distributed evaluation approach.

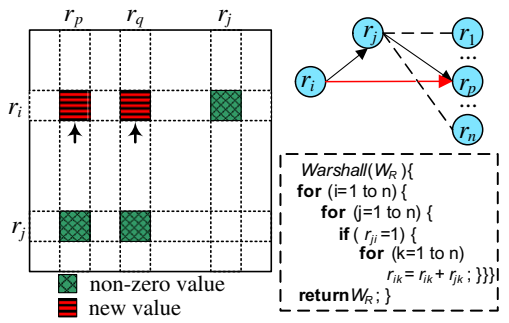
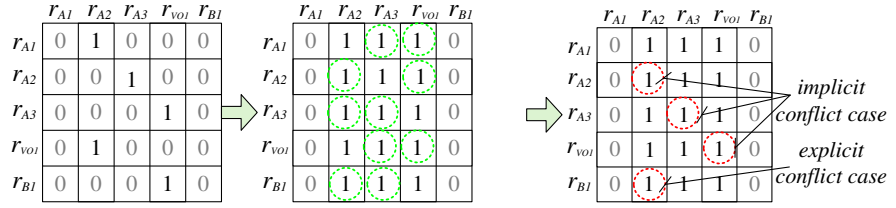


Fig. 12. Generating the transitive closure set with *Warshall* algorithm.



(a) The original matrix W_R (b) The matrix W_R processed by *Warshall* algorithm (c) The matrix W_R for policy conflict checking

Fig. 13. Detecting policy conflicts based on role mapping matrix.

then there are implicit policy conflicts; if there are *non-zero* elements corresponding to role mappings in F_A, then there are explicit policy conflicts.

The distributed algorithm is shown in Fig. 14.

Theorem 2. *Let each role mapping chain L in G_{VO} be a valid chain. The time-complexity of the distributed evaluation algorithm for virtual organization collaboration policies is* $O(\max\{|R_i|^3, |R_{VO}|^3\})$.

Proof. The distributed evaluation algorithm for virtual organization collaboration policies can effectively reduce the computation cost. As shown in Fig. 14, the time-complexity of this algorithm is mainly determined by the computation of transitive closure of a role set, where a *Warshall* algorithm is used to calculate H_i^+ and H_{VO}^+, and its complexity is $O(|R_i|^3)$.

Algorithm: *PEACE_VO_Evaluation* (G_i, R_{VO}, H_{VO}, M_{VO}) {
Input: The domain security policies G_i, and *task policies* including R_{VO}, H_{VO} and M_{VO} of a *VO*
Output: The policy evaluation result got by the i^{th} domain (boolean type)

```
1.  H+VO = Warshall(HVO);  //the transitive closure of set HVO
2.  H+i = Warshall(Hi);
3.    for (int k=1; k<=n; k++) {
4.      if (k==i){ continue;}
5.         H+k = Warshall( Hk );
6.      }
7.       S = ( ⋃_{k=1}^{n} H+k  MVO  H+VO  Mi  H+i );
8.       if (S ∩ Fi !=∅) {
9.         return false; } // explicit policy conflict
10.      if (∃(r,r)∈S && r∈Ri)){
11.        return false; } // implicit policy conflict
12.    }
13.    return true;
14.  }
```

Fig. 14. The distributed evaluation algorithm for virtual organization collaboration policies.

Hence, the maximum time-complexity of this phase is $O(\max\{|R_i|^3, |R_{VO}|^3\})$. In addition, due to the size of the set S is $|R_i|^2 \times |R_{VO}|$, the time-complexity of the phase comparing with all elements in S is $O(\{|R_i|^2 \times |R_{VO}|\})$. The total time-complexity remains $O(\max\{|R_i|^3, |R_{VO}|^3\})$.

Gong and Qian, Bertino et al. [11,26] have proved some valuable results for time-complexity on security interoperation. The mediator-based algorithm needs to evaluate the security of collaboration policies. Gong has proved that the time-complexity of this centralized algorithm is $O\left(|\cup_{i=1}^{n} R_i|^3\right)$ since the transitive closure of all role sets needed to be computed.

4. PEACE-VO management and authorization protocol

PEACE-VO consists of two protocols: a management protocol and an authorization protocol. The management protocol maintains a virtual organization and guarantees the security of collaboration policies when domain security policies or VO task policies are updated. The authorization protocol makes authorization decisions when services are requested.

We illustrate the two protocols with an example of a VO structure showed in Fig. 15, where we use symbol *VOS* to denote the *VO Server*, and symbol *DS* to denote the *Domain Server*.

4.1. Management protocol

This paper attempts to address the issues how to determine whether the virtual organization collaboration policies are secure or not. The policy for deciding whether a domain is allowed to join the virtual organization is beyond the scope of this paper, which can actually rely on existing mechanisms, such as fixed assignment, a voting, or manual configuration. Currently, PEACE-VO adopts the general approach employed for CROWN to handling new comers or leavers, with which the administrator manually approves requests after logining into the *VO Server* or directly modifies a root CA list file specified by the participating domains. How to resolve policy conflicts has also been a hot topic [4,26,32]. Currently, PEACE-VO employs two simple conflict resolution strategies. One is relying on *domain priority*, by which the domain security policies will not be changed. The other is relying on *task priority* by which the VO task policies will not be changed.

In describing the following protocols, we use the template of $A \rightarrow B$: {*msg*} to denote A sending a message *msg* to B. Table 2 lists the meta messages within the management protocol.

Next, we illustrate the procedure of this protocol with an example that the *VO Server* wants to update its task policies.

Step 1: *VOS* broadcasts the policy update message *VOServerUpdate* and evaluation notification message *VOEvaluation* together with updated M_{VO} and G_{VO} to all domains DS_i.

$$VOS \rightarrow DS_i : \{VOServerUpdate, VOEvaluation, M_{VO}, G_{VO}\}, \quad i \in \{1, \ldots, n\}.$$

Step 2: Every domain executes the algorithm shown in Fig. 14 and obtains the policy evaluation result, which is true if there is no policy conflicts and false otherwise. If the conflict resolution strategy is "*task policy priority*", the domain will revise its local policies until the result is true. If the strategy is "*domain priority*", the domain simply reports the conflict issue to *VOS*.

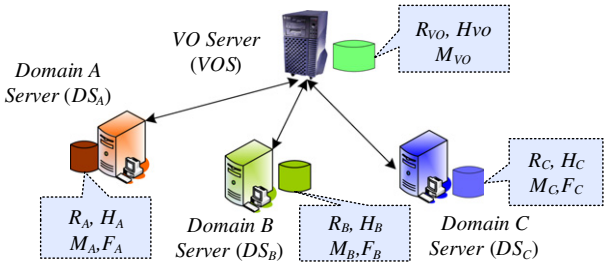


Fig. 15. An example of the VO structure.

Table 2
Basic messages in the management protocol.

Message	Description
JoinReq	An event that a domain wants to join the virtual organization
LeaveReq	An event that a domain wants to leave the virtual organization
VOServerUpdate	An event updating the policies of *VO Server*
DomainServerUpdate	An event updating the security policies of domain G_i
VOEvaluation	An event notifying all domains to evaluate the collaboration policies
ResponseMsg	Message and message identifier code

$DS_i \rightarrow VOS : \{ResponseCode\}$.

Step 3: If an evaluation result is false, and the resolution strategy is "*domain priority*", *VOS* will revise the collaboration policies according to the pre-configuration policy, and then Step 1 is repeated. Once all the returned results from all domains are true, *VOS* will inform the domains with a message of success.

$VOS \rightarrow DS_i : \{ResponseCode\}$.

4.2. Authorization protocol

During service authorization process, as depicted in Fig. 16, there are three key sub-processes: *domain roles assignment*, *virtual organization roles assignment* and *target domain roles assignment*. The complete authorization protocol is as follows:

Step 1: When a user, u, sends a request to its local domain DS_A, DS_A returns a signed credential containing the assigned roles (only roles also appeared in M_{VO}) to u:

$DS_A \rightarrow u : \{\mathbb{R}_A, PK_u, DS_A, SIG\}$,

where $\mathbb{R}_A = \{r_{A1}, \ldots, r_{Ak}\}$ is a role sequence that the user u acquired in its local domain, PK_u is the public key of user u, and DS_A is used to identify the domain this user originally belongs to, and $SIG = SIGNATURE_{PrivK\text{-}A}(Hash(\mathbb{R}_A, PK_u, DS_A))$ is the signature message signed by the private key of domain DS_A.

Step 2: After user u sends a request to *VOS*, *VOS* returns a signed credential containing the assigned VO *task roles* to u:

$VOS \rightarrow u : \{\mathbb{R}_{VO}, PK_u, DS_A, SIG\}$,

where $\mathbb{R}_{VO} = \{r_{VO1}, \ldots, r_{VOk}\}$ is a role sequence that user u acquired in this VO, $SIG = SIGNATURE_{PrivK\text{-}VO}(Hash(\mathbb{R}_{VO}, PK_u, DS_A))$ is the signature message signed by the private key of *VOS*.

Step 3& 4: After user u sends service requests to the target service in another domain, the service provider transfers the requests message to its DS_B, DS_B returns a signed credential containing the roles which u can be mapped to, and the service provider makes ultimate authorization decision according to its local domain security mechanisms without consideration to the VO policies.

Generally, the lifetime of roles assignment to a user is limited, so issuing time and validity time period should been specified in a credential.

5. Implementation

We have implemented PEACE-VO with Java. Currently, we use SAML 2.0 attribute assertion [13] to describe the VO role membership in PEACE-VO. There are some key services in PEACE-VO, and they are deployed on *VOS* or *DS*.

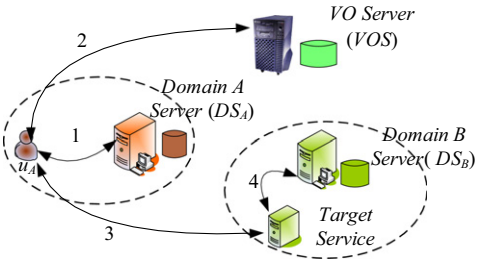


Fig. 16. Illustration of the authorization protocol.

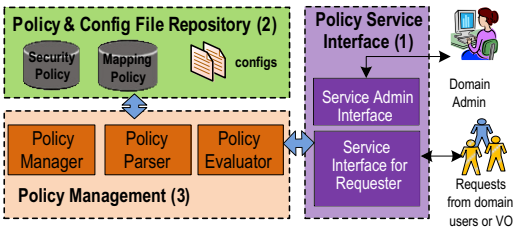


Fig. 17. The architecture of the *Policy Service*.

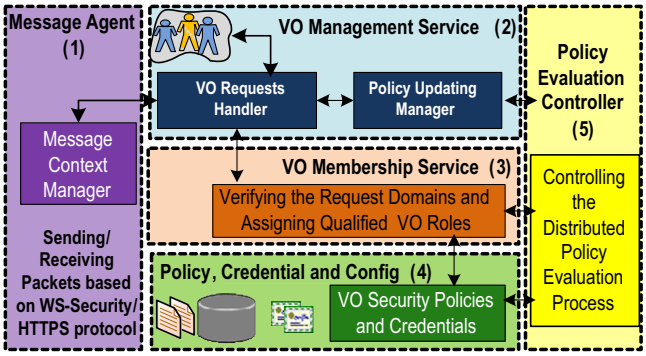


Fig. 18. The architecture of the *VO Service*.

The *Policy Service* is deployed on *DS*, and its architecture is shown in Fig. 17. The *Policy Service* consists of three key components, a *Policy Service Interface* (expressed in WSDL file), a *Policy* & *Config File Repository* and a *Policy Management System*.

1. *Policy Service Interface*: It provides two kinds of interfaces: one for a domain administrator to create, update or delete its policies or config files, another for issuing credentials to domain users or for collaboration policy evaluation initiated by *VOS*.
2. *Policy* & *Config File Repository*: It stores policies and configuration files that can be accessed by the *Policy Service*. The *Security Policy* database stores the local security policies of a domain, the *Mapping Policy* database stores the role mapping policies between this domain and VO, and configuration file specifies the configurations for policy databases.
3. *Policy Management System*: It includes three basic modules *Policy Manager*, *Policy Parser* and *Policy Evaluator*. The *Policy Manager* module provides operations to manage the policies and files, the *Policy Parser* module is used to generate and parse the credentials, policies and configuration files, and the *Policy Evaluator* module implements our distributed evaluation algorithm shown in Fig. 14.

The architecture of the *VO Service* is shown in Fig. 18, which includes two key services *VO Management Service* and *VO Membership Service* deployed on VO. The *VO Service* has five key parts as follows:

1. *Message Agent*: It sends or receives SOAP messages, and the CROWN middleware provides message-level and conversation-level security modes configured by a security handler chain for the communication security, where the WS-Security, WS-SecureConversation and HTTPS protocols are supported. The *Message Context Manager* is a module to manage the message context, conversation context, policy context and condition context of different sessions.
2. *VO Management Service*: It implements the virtual organization management protocol. The *VO Requests Handler* module invokes the next module according to the type of requests. When a VO role assignment request arrives, it invokes the *VO Membership Service*. When a domain policy request arrives, it invokes the *Policy Updating Manager*. The *Policy Updating Manager* is a module for updating the VO task policies and additionally invokes the *VO Policy Controller* for the distributed policy evaluation protocol management.
3. *VO Membership Service*: It receives requests from participating domains' users, and verifies the domain credentials and assigns qualified VO task roles to the user.
4. *Policy, Credential and Config*: It stores the VO task policies (VO mapping policies and VO task roles and hierarchy policies), the credentials of VOS, and other related configuration files, e.g., a config file configured by the VO administrator to approve new comers or leavers.

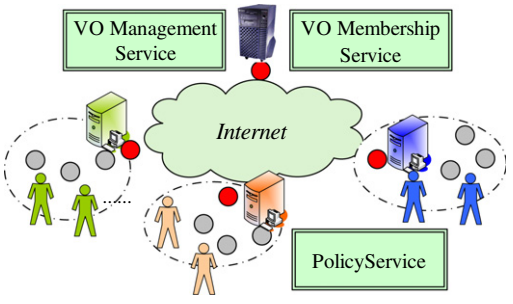


Fig. 19. Deployment of the PEACE-VO services.

5. *Policy Evaluation Controller*: It controls the process of distributed evaluation for VO collaboration policies, and collects all domain evaluation results. It also implements a simple policy conflicts resolver according to various priority strategies.

The deployment of the PEACE-VO services is demonstrated in Fig. 19. The *Policy Service* is deployed on every domain *DS node*, and the other two services are deployed on the *VOS* node.

6. Experimental results and analysis

To effectively evaluate the performance of the policy evaluation algorithm and services in the PEACE-VO system, we conduct comprehensive experiments.

6.1. Metrics and environment setup

We use the following metrics to evaluate the performance of our proposed approach.

- *Policy Evaluation Time* (PET): It is used to measure the time (excluding phases of policy parsing and evaluation) that the policy evaluator used to decide whether the collaboration policies are secure. In studying the performance of our algorithm, we compare it with the *centralized-like* algorithm as the approach of mediator-based secure interoperation.
- *Evaluation Optimization Ratio*: It is used to measure the optimization ratio of the distributed evaluation algorithm in comparison to the *centralized-like* algorithm. The optimization ratio δ of average policy evaluation time is defined as follows:

$$\delta = \frac{\mathrm{PET}_{\mathrm{Centralized}} - \mathrm{PET}_{\mathrm{Distributed}}}{\mathrm{PET}_{\mathrm{Centralized}}}.$$

- *Service Response Time*: It is the time period from the time that the domain sends joining requests to *VO Server* to the time that the requesting domain gets the response from the *VO Server*.

In order to evaluate the efficiency of the policy evaluation algorithm more accurately, we list the main parameters, shown in Table 3, that influence performance results. We set some default values for similar parameters, and generate test cases by varying three selected parameters.

Based on CROWN, we have implemented the PEACE-VO services. The fundamental security services in CROWN middleware consist of secure communication based on WS-Security, WS-Secure Conversation and WS-Policy specifications, policy-based access control mechanism based on XACML and SAML specifications, and identity federation mechanism between PKI and Kerberos security infrastructures based on WS-Federation specifications [18]. The PEACE-VO services are deployed on a cluster node with Intel Xeon 2.8 GHz CPU, 2 GB RAM, Linux operating systems and 100 Mbps Internet connection. We use a notebook with 1.6 GHz CPU, 512 MB RAM, Windows XP operating system and 100 Mbps Internet connection as a client. To make sure that measurements are accurate, no other tasks, except the necessary CROWN middleware, are running on the cluster node and the notebook. If not explicitly specified otherwise, all the experiments are executed five times and average values are used.

6.2. Experimental results

Experiment Group 1: These experiments study the performance of the distributed collaboration policy evaluation, and study the impact of the parameters n and η on the performance of the two algorithms.

In the first experiment, we generate test cases with $n = 5$, $\xi = 20$, and vary η from 50 to 500 with a step of 50, and compare performance of the two algorithms. The results are presented in Figs. 20 and 21. Fig. 20 shows the policy evaluation time increases roughly linearly with increasing η. With two algorithms, the policy evaluation time is 50 and 9 ms, respectively, when $\eta = 50$, it is 122 and 1392 ms when $\eta = 500$. Fig. 21 shows that the evaluation optimization ratio is around 80–90%, and increases with increasing η.

Table 3
Configurations of the parameters.

Parameter	Value
n	The number of participating domains
$\eta = \lvert R_i \rvert$	The value of η is equal
$\xi = \lvert H_i \rvert$	The value of ξ is equal
$\eta_{vo} = \lvert R_{vo} \rvert$	Default: 10
$\xi_{vo} = \lvert H_{vo} \rvert$	Default: 3
$\theta = \lvert M_i \rvert$	The value of θ is equal, default: 3
$\partial = \lvert F_i \rvert$	The value of ∂ is equal, default: 3
$\theta_{vo} = \lvert M_{vo} \rvert$	Default: 10

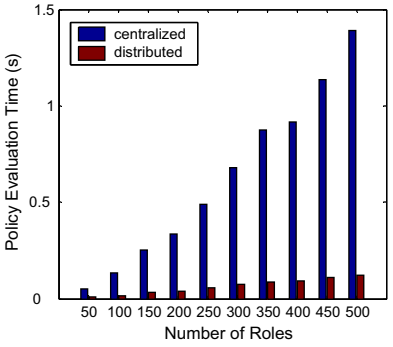


Fig. 20. Policy evaluation time vs number of roles.

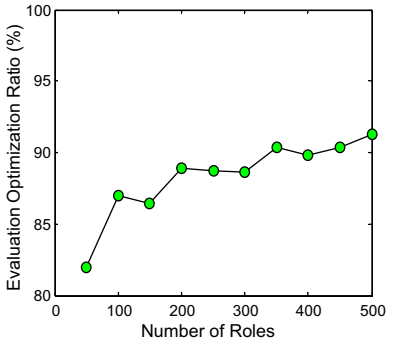


Fig. 21. Evaluation optimization ratio vs number of roles.

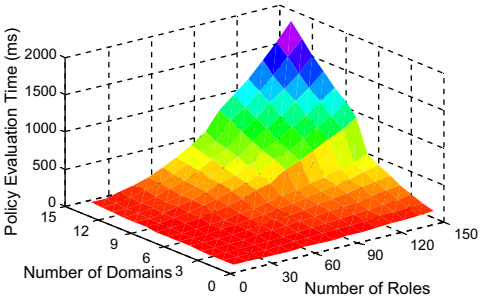


Fig. 22. Policy evaluation time in a centralized manner.

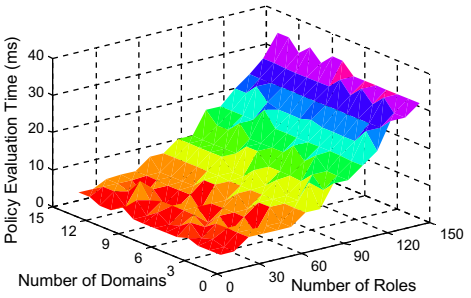


Fig. 23. Policy evaluation time in a distributed manner.

In the second experiment, we generate test cases with $\xi = 20$, and vary η from 10 to 150 with a step of 10. The results are presented in Figs. 22 and 23. Fig. 22 shows that the policy evaluation time of a centralized manner increases with increasing n and η, and the increase becomes faster when n and η are higher. For example, when $n = 10$, $\eta = 100$, the time is 512 ms. Fig. 23 shows that the policy evaluation time of a distributed manner only increases with increasing η.

The above two experiments verified the theoretical result in Section 4. The performance of policy evaluation algorithm in a distributed manner is better than a centralized one. It is also observed that the time is related to the size of matrix within the *Warshall* algorithm. This explains why the distributed algorithm reduces the time dramatically, in which the size of matrix is separated.

Experiment Group 2: These experiments study the performance of virtual organization collaboration policy evaluation, and study the impact of the parameters, n and ξ, on the performance of the two algorithms.

In the first experiment, we generate test cases with $n = 5$, $\eta = 5$, and vary n from 2 to 46 with the step of 4. We compare the performance of the two algorithms. The results are presented in Figs. 24 and 25. Fig. 24 shows the policy evaluation time has no relation with increasing ξ. With two algorithms, the policy evaluation time is 37 and 3 ms, respectively, when $\xi = 10$, and it is 34 and 3 ms when $\xi = 30$. Fig. 25 shows the evaluation optimization ratio is around 80–95%.

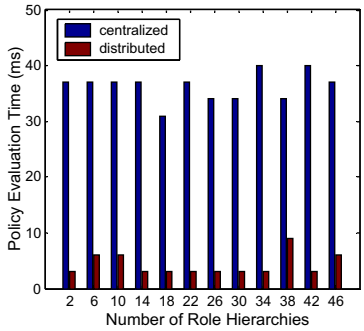


Fig. 24. Policy evaluation time vs number of role hierarchies.

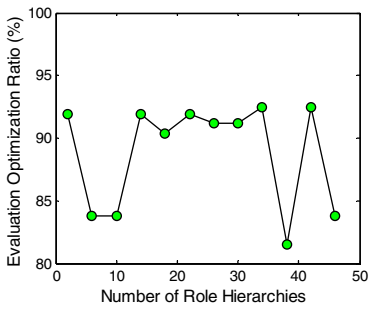


Fig. 25. Policy evaluation optimization ratio vs number of role hierarchies.

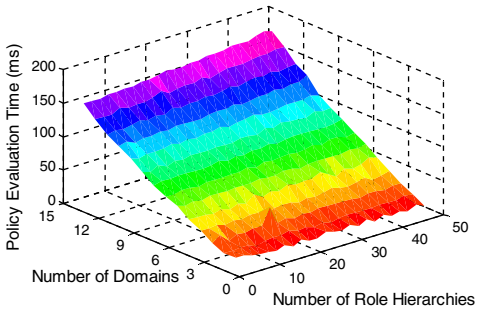


Fig. 26. Policy evaluation time in a centralized manner.

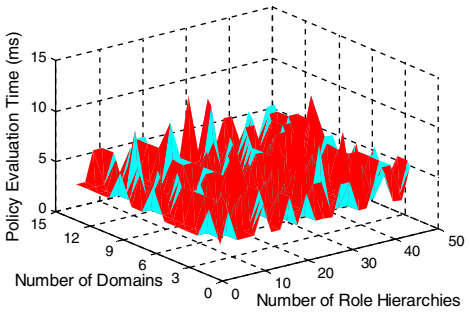


Fig. 27. Policy evaluation time in a distributed manner.

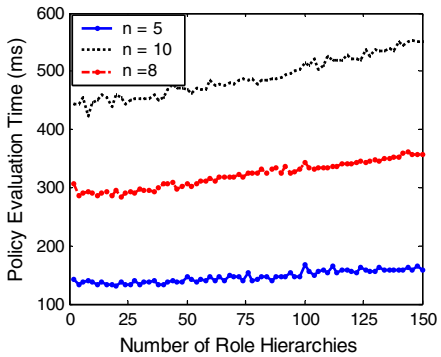


Fig. 28. Policy evaluation time vs number of role hierarchies.

In the second experiment, we generate test cases with $\eta = 50$, and vary n from 2 to 15, ξ from 2 to 48 with a step of 2. The results are presented in Figs. 26 and 27. Fig. 26 shows the policy evaluation time, in a centralized manner, only increases with increasing n. Fig. 27 shows the policy evaluation time in a distributed manner is between 3 and 10 ms. The performance almost has not been affected by the values of n and η, only increase with increasing η.

In the third experiment, we generate test cases with $\eta = 150$, $n = 5, 8, 10$, and vary ξ from 1 to 149. Fig. 28 plots the policy evaluation time against the number of role hierarchies, and it demonstrates the policy evaluation time increases slowly with increasing ξ because the evaluation time is proportional to the size of matrix, and the time used to non-zero matrix elements processing is very little.

Experiment Group 3: These experiments study the performance of virtual organization management and authorization services, and the performance that a client (domain) requests to join the virtual organization with different secure communication configurations.

In this experiment, we study the service response time with different security communication configurations (signature algorithm: RSA-SHA1, encryption algorithm: 3DES-CBC) when a new domain ($\eta = 50$) joins the virtual organization. The results, shown in Table 4, demonstrate the service response time increases with increasing n. This is because *VOS* needs to send evaluation notification to every domain, and the final result will be returned to the client after all evaluation results from the other domains are collected.

In addition, we study the communication overhead of PEACE-VO. Every SOAP message is around 3–4 KB, and it means that the total overhead is around 12–16 KB because virtual organization authorization protocol includes two rounds of interaction. In particular, the size of a domain security policies is around 8–10 KB when $n = 5$, $\eta = 50$, but around 13–15 KB when $n = 5$, $\eta = 100$. Therefore, if we adopt a centralized approach in which every domain submits its local security policies to the

Table 4
Average response time (s) when a domain joins.

	Domains (n)			
Configuration	2	3	4	5
Signature	0.964	1.059	1.108	1.163
Signature & *Encryption*	1.908	2.187	2.239	2.304

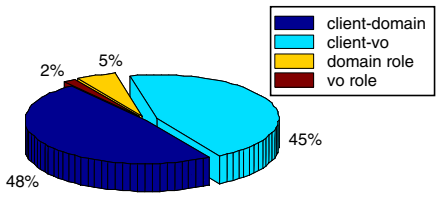


Fig. 29. Time distribution during authorization.

Table 5
Comparison of PEACE-VO and other security management systems.

Features	CAS[a]	Grid Slub[b]	M-based[c]	M-free[○3]	PEACE-VO
domain policy	No	Yes	Yes	Yes	Yes
policy valid conditions	No	No	No	Yes	Yes
policy conflict checking	No	No	Yes centralized	N/A	Yes distributed
domain privacy preservation	No	No	No	Yes	Yes

[a] CAS [20] is an authorization server for virtual organizations used in Globus Toolkit.
[b] GridShib [33] is an identity federation-based service for virtual organization.
[c] M-free [26] is a mediator-free approach for secure interoperation.

VO Server, the secure communication overhead will be increased. More importantly, the privacy of local domain policies cannot be preserved.

In the second experiment, we study performance of role assignment processing during service authorization and percentage of processing time during every phase. The processing time of communication between Client and Domain Server (client–domain), communication between Client and *VO Server* (client–vo), domain role assignment (domain role) and virtual organization role assignment (vo role) is 319, 298, 35 and 12 ms, respectively. As the pie chart shown in Fig. 29, the total time is divided into four parts, the two communication phases consume the largest portion, accounting for 93%. This experiment result shows that a majority of time is consumed for communication, thus in an actual application, the server can issue the role assignment credential with reasonable lifetime to reduce frequent communications with server, so that the efficiency of service authorization can be improved.

7. Conclusion and future work

In this paper we have proposed a novel secure collaboration service, PEACE-VO, in which a fully distributed policy evaluation algorithm is devised to improve evaluation efficiency without disclosing full domain security policies. In PEACE-VO, we have adopted a central server in a VO for the benefit of better performance for service authorization.

Compared with existing systems and approaches, PEACE-VO has several important features, as detailed in Table 5. First, the feature of reusing domain policy can help create a virtual organization based on existing domain policies, while a service access control decision is still made in its local domain. Second, policies of PEACE-VO have some properties used to support distributed policy evaluation and privacy preservation, which we believe will not restrict the administrator's freedom of creating necessary policies. Third, the feature of distributed policy conflict detection not only improves the performance of security checking, but also withholds possible internal attacks. Finally, the feature of privacy preservation is critical in an Internet-based computing environment. Local domains need not disclose all their privacy policies to form a virtual organization. In addition to differences on aforementioned features, the resource authorization performance can be improved if a central server (e.g., the *VO Server* used by CAS, GridShib and PEACE-VO) is used for membership assignment. In contrast, the M-free approach has low performance because there is an additional procedure of dynamic access path construction.

We are building a virtual computing test bed based on virtualization technologies (e.g., XEN, KVM) and CROWN. In PEACE-VO, the authorization server may suffer from the single point of failure problem. However, we can encapsulate our services within a virtual machine and make use of appealing characteristics, such as live deployment, and live migration to improve their reliability.

Acknowledgment

The authors gratefully acknowledge the anonymous reviewers for their helpful suggestions and comments. Some preliminary results of this work were presented in SRDS 2007. This work is partially supported by grants from the China National Science Foundation (Nos. 60903149 and 60731160632), China 863 High-tech Program (Nos. 2009AA01Z419, 2009AA012201 and 2009ZX03006-001), and China 973 Fundamental R&D Program (No. 2005CB321803).

Appendix A. Table of the symbols in PEACE-VO

Symbol	Note
A, *B*, and *C* etc.	A domain is denoted by an uppercase letter with or without subscripts
u	A user of a domain
VO	A virtual organization formed by some domains
r	A role is denoted by r with subscripts or not
R	A set of roles is denoted by R
$\prec$	A role inheritance relation between two roles, e.g., $r_{A2} \prec r_{A1}$ implies that the role r_{A2} is senior to role r_{A1}
H	A set of role hierarchy relations
H^+	A transitive closure set of H
R_{VO}	A set of task roles in a VO, which is defined for the common task of all participating domains
R'_{VO}	All roles of a virtual organization is denoted by R'_{VO}
m: (r_{A1}, r_{A2})	A role mapping policy is denoted by m, it means that if a user is a member of role r_{A1}, then this user acquires the permissions of role r_{A2}
M	A binary relation representing the mapping policies
M_i	A set of domain role mapping policies, its element $(r_{VO}, r_q) \in R_{VO} \times R_i$
M_{VO}	A set of Virtual Organization Role Mapping policies, element $(r_p, r_{vo}) \in \cup_{i=1}^{n} R_i \times R_{VO}$
$G = \langle R, H \rangle$	The security policies of a domain
F	A forbidden role mapping policies denoted by F with subscripts or not
$G_{VO} = \langle R'_{VO}, H'_{VO} \rangle$	The collaboration policies of a virtual organization
$L = \langle r_o, r_1, \ldots, r_k \rangle$	A role mapping chain from role r_0 in domain G_i to role r_k in domain G_j
VOS	Its full name is *VO Server*
DS	Its full name is *Domain Server*
$A \rightarrow B$: {*msg*}	It means A sends a message *msg* to B
PK_u	The public key of user u
DS_A	Identifying the domain that a user u originally belongs to
$SIGNATURE_{PrivK\text{-}A}(Hash(\mathbb{R}_A, PKu, DSA))$	A signature message signed by the private key of domain DS_A
$SIGNATURE_{PrivK\text{-}VO}(Hash(\mathbb{R}_{VO}, PKu, DSA))$	A signature message signed by the private key of *VOS*

References

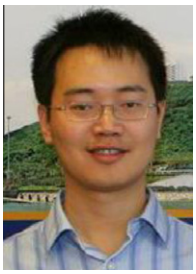
[1] C.Y. Andrew, Protocols for secure computations, in: Proceedings of the 23rd Annual Symposium on Foundations of Computer Science, 1982, pp. 160–164.
[2] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, Above the Clouds: A Berkeley View of Cloud Computing, 2009. Available from: <http://abovetheclouds.cs.berkeley.edu>.
[3] D. Clarke, J.-E. Elien, C. Ellison, M. Fredette, A. Morcos, R.L. Rivest, Certificate chain discovery in SPKI/SDSI, Journal of Computer Security 9 (4) (2001) 285–322.
[4] R. Deitos, F. Kerschbaum, P. Robinson, A comprehensive security architecture for dynamic, web service based virtual organizations for businesses, in: Proceedings of the 3rd ACM Workshop on Secure Web Services Alexandria, Virginia, USA, 2006, pp. 103–104.
[5] T. Dimitrakos, G. Laria, I. Djordjevic, N. Romano, F. D'Andria, V. Trpkovski, P. Kearney, M. Gaeta, P. Ritrovato, L. Schubert, B. Serhan, L. Titkov, S. Wesner, Towards a grid platform enabling dynamic virtual organizations for business applications, in: iTrust 2005, Oxford, UK, 2005, pp. 406–410.
[6] S. Du, J.B.D. Joshi, Supporting authorization query and inter-domain role mapping in presence of hybrid role hierarchy, in: The 11th ACM Symposium on Access Control Models and Technologies (SACMAT2006), Lake Tahoe, California, USA, 2006, pp. 228–236.
[7] I. Foster, C. Kesselman, S. Tuecke, The anatomy of the grid: enabling scalable virtual organizations, in: Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing, 2001, pp. 1–20.
[8] E. Freudenthal, T. Pesin, L. Port, E. Keenan, dRBAC: distributed role-based access control for dynamic coalition environments, in: The 22nd International Conference on Distributed Computing Systems (ICDCS'02), Vienna, Austria, 2002, pp. 411–420.
[9] K. Frikken, M. Atallah, J. Li, Attribute-based access control with hidden policies and hidden credentials, IEEE Transaction on Computers 55 (10) (2006) 1259–1270.
[10] K.B. Frikken, J. Li, M.J. Atallah, Trust negotiation with hidden credentials, hidden policies, and policy cycles, in: Network and Distributed System Security Symposium(NDSS 2006), San Diego, California, USA, 2006.
[11] L. Gong, X. Qian, Computational issues in secure interoperation, IEEE Transaction on Software and Engineering 22 (1) (1996) 43–52.
[12] J. Huai, C. Hu, J. Li, H. Sun, T. Wo, CROWN: a service grid middleware with trust management mechanism, Science in China Series F: Information Sciences 49 (6) (2007) 731–758.
[13] T. Kataoka, T. Nishimura, M. Shimaoka, K. Yamaji, M. Nakamura, N. Sonehara, Y. Okabe, Leveraging PKI in SAML 2.0 federation for enhanced discovery service, in: The Ninth Annual International Symposium on Applications and the Internet, 2009, pp. 239–242.
[14] Y. Leea, H. Kima, Yongsu Parkb, An efficient delegation protocol with delegation traceability in the X.509 proxy certificate environment for computational grids, Information Sciences 178 (14) (2008) 2968–2982.
[15] J. Li, D. Zhang, J. Huai, J. Xu, Context-aware trust negotiation in peer-to-peer service collaborations, Peer-to-Peer Networking and Applications 2 (2) (2009) 164–177.
[16] N. Li, W. Du, D. Boneh, Oblivious signature-based envelope, in: Proceedings of the 22nd Annual Symposium on Principles of Distributed Computing, Boston, Massachusetts, 2003, pp. 182–189.

[17] N. Li, W.H. Winsborough, J.C. Mitchell, Distributed credential chain discovery in trust management, Journal of Computer Security 11 (1) (2003) 35–86.
[18] Q. Li, J. Li, J. Huai, X. Liu, C. Hu, CROWN-ST: security and trustworthiness architecture for CROWN grid, in: IEEE International Conference on e-Science and Grid Computing (eScience2006), Amsterdam, Netherlands, 2006, pp. 23–31.
[19] H. Lockhart, S. Andersen, J. Bohren, Y. Sverdlov, M. Hondo, H. Maruyama, A. Nadalin, N. Nagaratnam, T. Boubez, K.S. Morrison, C. Kaler, A. Nanda, D. Schmid, D. Walters, H. Wilson et al., Web Services Federation Language (WS-Federation), 2006. Available from: <http://www.ibm.com/developerworks/webservices/library/ws-fed/>.
[20] L. Pearlman, V. Welch, I. Foster, C. Kesselman, S. Tuecke, A community authorization service for group collaboration, in: The IEEE 3rd International Workshop on Policies for Distributed Systems and Networks, Monteray, California, USA, 2001, pp. 50–58.
[21] P. Periorellis, N. Cook, H. Hiden, GOLD infrastructure for virtual organisations, Concurrency and Computation: Practice & Experience 20 (11) (2006) 1273–1288.
[22] D. Power, M. Slaymaker, A. Simpson, On formalizing and normalizing role-based access control systems, The Computer Journal 52 (3) (2009) 305–325.
[23] S. Dawson, S. Qian, P. Samarati, Providing security and interoperation of heterogeneous systems, Distributed Parallel Databases 8 (1) (2000) 119–145.
[24] K. Seamons, M. Winslett, T. Yu, L. Yu, R. Jarvis, Protecting privacy during on-line trust negotiation, in: Proceedings of 2nd Workshop on Privacy Enhancing Technologies, 2003, pp. 249–253.
[25] K.E. Seamons, M. Winslett, T. Yu, Limiting the disclosure of access control policies during automated trust negotiation, in: Network and Distributed System Security Symposium (NDSS 2001), San Diego, California, 2001.
[26] B. Shafiq, J.B.D. Joshi, E. Bertino, A. Ghafoor, Secure interoperation in a multidomain environment employing RBAC policies, IEEE Transactions on Knowledge and Data Engineering 17 (11) (2005) 1557–1577.
[27] M. Shehab, E. Bertino, A. Ghafoor, Secure collaboration in mediator-free environments, in: Proceedings of the 12th ACM Conference on Computer and Communications Security(CCS05), Alexandria, VA, USA, 2005, pp. 58–67.
[28] D. Smith, The challenge of federated identity management, Network Security 2008 (4) (2008) 7–9.
[29] D.Z. Sun, J.P. Huai, J.Z. Sun, J.X. Li, J.W. Zhang, Z.Y. Feng, Improvements of Juang et al.'s password-authenticated key agreement scheme using smart cards, IEEE Transactions on Industrial Electronics 56 (6) (2009) 2284–2291.
[30] D.Z. Sun, J.P. Huai, J.Z. Sun, J.W. Zhang, Z.Y. Feng, A new design of wearable token system for mobile device security, IEEE Transactions on Consumer Electronics 54 (4) (2008) 1784–1789.
[31] V. Venturi, M. Riedel, S. Memon, F. Stagni, B. Schuller, D. Mallmann, B. Tweddell, A. Gianoli, S.v.d. Berghe, D. Snelling, A. Streit, Using SAML-based VOMS for authorization within web services-based UNICORE grids, Lecture Notes in Computer Science 4854 (2008) 112–120.
[32] H. Wang, S. Jha, M. Livny, P.D. McDaniel, Security policy reconciliation in distributed computing environments, in: Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04), 2004, pp. 137–145.
[33] V. Welch, T. Barton, K. Keahey, F. Siebenlist, Attributes, anonymity, and access: shibboleth and globus integration to facilitate grid collaboration, in: The 4th Annual PKI R&D Workshop, NIST Gaithersburg, MD, 2005.
[34] T. Yu, Automated trust establishment in open systems, in: School of Computer, vol. PhD: University of Illinois at. Urbana-Champaign, 2003.
[35] T. Yu, M. Winslett, A unified scheme for resource protection in automated trust negotiation, in: Proceedings of the 2003 IEEE Symposium on Security and Privacy (S&P2003), 2003, pp. 110–122.

Jianxin Li is an assistant professor in the School of Computer Science and Engineering, Beihang University, Beijing china. He received the PhD. degree in Jan 2008. He has authored over 20 papers in IEEE T. on Industry Electronic, SRDS, HASE and eScience etc. His research interests include trust management, information security and distributed system.

Jinpeng Huai is a Professor and President of Beihang University. He serves on the Steering Committee for Advanced Computing Technology Subject, the National High-Tech Program (863) as Chief Scientist. He is a member of the Consulting Committee of the Central Government Information Office, and Chairman of the Expert Committee in both the National e-Government Engineering Taskforce and the National e-Government Standard office. Dr. Huai and his colleagues are leading the key projects in e-Science of the National Science Foundation of China (NSFC) and Sino-UK. He has authored over 100 papers. His research interests include middleware, peer-to-peer (P2P), grid computing, trustworthiness and security.

Chunming Hu is a research staff and associate professor in the School of Computer Science and Engineering, Beihang University, Beijing, China. He received his B.E. and M.E. in Department of Computer Science and Engineering in Beihang University. He received the Ph.D. degree in School of Computer Science and Engineering of Beihang University, Beijing, China, 2005. His research interests include peer-to-peer and grid computing; distributed systems and software architectures.



Yanmin Zhu obtained his PhD in computer science from Hong Kong University of Science and Technology in 2007, and BEng in computer science from. Xi'an Jiao Tong University in 2002. He is an Assistant Professor in the Department of Computer Science and Technology in Shanghai Jiao Tong University. His research interests include ad hoc sensor networks, mobile computing, grid computing and resource management in distributed systems. He is a member of the IEEE and the IEEE Communications Society.