Access Control in Peer-to-Peer Collaborative Systems

Yu Zhang, Xianxian Li, Jinpeng Huai
Dept. of Computer Science and Technology
Beihang University
Beijing, 100083, P.R.China
zhangyu@act.buaa.edu.cn

Yunhao Liu
Dept. of Computer Science
Hong Kong Univ. of Science and Technology
Clearwater Bay, Kowloon, Hong Kong
liu@cs.ust.hk

Abstract

As an emerging model of communication and computation, peer-to-peer networking represents a fully distributed, cooperative network design, and has recently gained significant acceptance. Peer groups share the properties of peer-to-peer overlay network, including full decentralization, symmetric abilities, and dynamism, which make security problems more complicated. In this paper, we propose a fine-grained and attribute-based access control framework for peer-to-peer systems. This design employs a novel policy model which extends role-based trust management language RT to satisfy security requirements of peer groups. Intend for a pure decentralized model without centralized server, our framework presents distributed delegation authorization mechanism which could avoid single point of failure. We also introduce our implementation experience.

1. Introduction

The emerging peer-to-peer model has recently gained significant attention due to its high potential of sharing huge amount of resources among millions of users, where each peer acts as both a resource provider and a consumer. In many cases, multiple self-organizing peers aggregate in a controlled manner, and use various communication primitives to accomplish their collective goals. Collaborative peer groups [1] is introduced to refer to such peer-to-peer networks. Peer groups are a strong and flexible structure to enable coordination among applications, server-clients, and peers in networks. Examples of such collaborations include file sharing, grid computing, cooperative system, etc. Group settings may be synchronous or asynchronous, and communication models vary from one-to-many, few-to-many, to any-to-any.

Peer group was initially introduced in JXTA [2, 3] as a way of collecting peers that have agreed upon a common set of services. Security in collaborative peer groups is an active research topic. Most previous works focus on the context of group membership authentication, group key management [4], and communication security. As a key precondition of many security services, access control, however, is not fully addressed. Collaborative peer groups share the properties of peer-to-peer overlay network [1, 5, 6], including full decentralization, symmetric abilities, and dynamism, which make security problems more complicated.

Consider the following scenario, where peer group is formed by multiple peers from three organizations, namely Genetics organization, Hospital, and Pharmaceutical company (GHP), to discover the gene sequence for a particular disease. Usually, all peers first negotiate a Collaboration Policy Instance (CPI) to satisfy multiple peers’ security requirements. Four roles are defined in GHP group: group authority, group member, director, and developer. If an employee of the pharmaceutical company wants to join the group, to grant the membership, the CPI requires at least 3 votes from existing group members, and more than half of the votes are yes. Additionally, large sensitive experiment data generated during the collaboration will be stored in a stand alone peer. The data can only be modified by two different developers in GHP group.

This paper proposes a fine-grained and attribute-based access control framework for peer-to-peer collaborative systems. The policy model extends the role-based trust management language RT [7-9] to satisfy security requirements of peer groups. The major contributions of this design are as follows:

1. To avoid single point of failure and enhance scalability of the system, instead of using a centralized model with a central server [10], we present a distributed delegation authorization mechanism. In this design, multiple authorities could exist in P2P systems to grant peer group membership, which makes both the
overhead and the response time of authority being reduced significantly.

2. Existing approaches fail to deal with the dynamicity of the peering nodes. Worse, peers are often unknown with each other. Therefore, identity-based access control, in which decisions are made based on the identity of requesters, becomes ineffective. Our framework addresses these two issues by employing an attribute based approach.

3. In P2P systems, peers wish to manage group security by themselves. We thus provide a voting scheme to let existing peer groups to accept new members and grant permissions by voting. The proposed voting scheme may be fixed or adaptive.

4. Sensitive experimental data generated during the collaboration should not be unilaterally modified by any single user. Our framework provides a secure cooperative process for multiple peers.

The rest of this paper is organized as follows. Section 2 introduces related work. Section 3 discusses the access control policy model. We present a formal joint authorization protocol by applying JXTA technology in Section 4. Section 5 describes a secure cooperation process. Section 6 introduces our implementation experience. We conclude our work and suggest future directions in Section 7.

2. Related work

Many efforts have been made on security issues in collaborative environments [10-15]. Gothic [10] provides security service for IP-Multicast, which only considers receiver access control. An external access control server provides authentication and authorization based on PKI certificates. Antigone [11] includes a flexible policy framework for secure group communication and defines group policies. Antigone employs a centralized access control approach in which member access is mediated by a session leader, and it is not designed for P2P and Grid systems.

Some frameworks are focused on peer-to-peer applications. Sconce [12, 13] presents an admission control framework on Gnutella like P2Ps [16]. It provides three types of admission policy, including access control list APT_ACL, a centralized authority APT_GAUTH, and group members APT_GROUP. A group membership certificate can be issued to a new member under multi-voting schemes. However, Sconce, which lacks the attribute of peers, cannot simplify authorization in collaborative environments, and is not scalable.

JXTA [2, 3], an open-source project initiated by SUN, is designed to solve a number of problems in modern distributed computing, in which a security mechanism based on PKI certificate is proposed [17]. Intergroup [18] provides access control using an authorization service called Akenti [19], which relies on X.509 identity certificate. All group members register with the authorization service off-line to obtain a membership certificate signed by the Akenti Server. Intergroup provides a coarse granularity for access control. Spread [14] introduces roles into group. It is a hierarchical client-server architecture where an expensive distributed protocol runs among a set of servers, providing services to the clients. Spread does not discuss distributed authorization in detail.

Our work focuses on a pure decentralized model in peer-to-peer collaborative systems. It is a distributed delegation authorization mechanism. By considering joint authorization and secure cooperation under voting schemes, security for communication and sharing of sensitive data among grouped peers are provided.

3. Access control

Many sensitive operations and services need access control [11, 14]. Here we define two kinds of roles: group role and application role. Group role peers are predefined by peer groups, and application role peers are defined according to different collaborations.

Before access control to be implemented, peers need to be authenticated. Since peers are often dynamic and unknown to each other, our framework adopts credential in trust management [8, 20] as authentication method. Permissions a peer being allowed to carry out depends on the roles and environment factors [21].

Our access control policy model for peer-to-peer collaborative systems defines the relations of roles and permissions, introduces six credentials from RT, and describes admission and removal policy of roles. Elements of access policy model are defined as follows.

1. C: Context, C defines group contexts, including variables and their values.
2. OBJ: Object Set, OBJ={obj1, obj2, ..., objn}.
3. OP: Operation Set, OP={op1, op2, ..., opn}.
4. P: Permission Set, P=OP×OBJ×C, that is,
P={<op, obj, c> | op ∈ OP, obj ∈ OBJ, c ∈ C}.
5. RoleTerm: It is defined as A.r(h1, ..., hn), in which A is entity name(optional), r is role name. A RoleTerm may include zero or more restriction parameters hi.
6. R: RoleTerm Set, R=SR ∪ AR, SR and AR are all RoleTerm Set, and SR is group roles set, while AR is application roles set.
8. Credential: Our system introduces six kinds of
Credential from RT [8], each Credential has a head part and body part as (R, R, are RoleTerm, D is entity).

- $R\rightarrow D$: The body part consists of a simple entity D, which means D is the member of R.
- $R\rightarrow R_1$: The body part consists of a RoleItem $R_1$, which means the principal set of $R_1$ contains the principal set of $R_f$.
- $R\rightarrow R_1 \cap \cdots \cap R_k$: The body part consists of an Intersection element, which means the principal set of $R$ contains the principal set of $R_1 \cap \cdots \cap R_k$.
- $R\rightarrow R_1 \cap \cdots \cap R_k$: The body part consists of a LinkRole element, which means the principal set of $R$ contains the principal set of $K_R R_2$, in which $K_R$ is the member of $R_f$. If $R_f$ is a manifold role, that is, $\{K_{R_1}, \cdots, K_{R_k}\}$ is the member of $R_f$, then the principal set of $R$ contains the principal set of $K_{R_1} R_2 \cap \cdots \cap K_{R_k} R_2$.
- $R\rightarrow R_1 \cap \cdots \cap R_k$: The body part consists of Product element, which means the principal $p$ is the member of $R$ and $p=p_1 \cup \cdots \cup p_k$. $p_i$ is the member of $R_i$.
- $R\rightarrow R_1 \cap \cdots \cap R_k$: The body part consists of ExclusiveProduct element, which means the principal $p$ is the member of $R$ and $p=p_1 \cup \cdots \cup p_k$. $p_i$ is the member of $R_i$.

According to the above policy model, the kernel parts of GHP policy can be depicted by Table 1. The GHP group defines two application roles, director and developer. Section 4 and 5 will discuss in detail.

<table>
<thead>
<tr>
<th>Table 1: Collaborative policy instance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Group Name</strong>: GHP</td>
</tr>
<tr>
<td><strong>C</strong>: day $\in {\text{MON}, \ldots, \text{SUN}}$</td>
</tr>
<tr>
<td><strong>R</strong>: {group authority, group member, director, developer}</td>
</tr>
<tr>
<td><strong>PA</strong>:</td>
</tr>
<tr>
<td>group authority $\leftarrow K_{\text{GeneOrg.projectleader}}, \text{true}$, true</td>
</tr>
<tr>
<td>group member $\leftarrow K_{\text{PharmCom.employee}}, \text{true}$, vote (group member, 3, 0.5)</td>
</tr>
<tr>
<td>Develop $\leftarrow K_{\text{Hospital.physician}}, \text{true}$, true</td>
</tr>
<tr>
<td>Developer $\leftarrow K_{\text{GeneOrg.researcher}}, \text{true}$, true</td>
</tr>
<tr>
<td>Director $\leftarrow K_{\text{developer}} \odot \text{developer}, \text{true}$, true</td>
</tr>
<tr>
<td><strong>RP</strong>: developer, true, vote (group authority, 2, 1)</td>
</tr>
</tbody>
</table>

4. Joint Authorization

In pure decentralized P2P systems, peers wish to manage group security by themselves without central servers such as CAs. Joint authorization by multiple peers under voting schemes could satisfy this requirement. Table 2 summarizes the notion used in the rest of the paper.

<table>
<thead>
<tr>
<th>Table 2: Notion summary</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GA</strong></td>
</tr>
<tr>
<td><strong>M_i</strong></td>
</tr>
<tr>
<td><strong>OC_i</strong></td>
</tr>
<tr>
<td><strong>PGC_i</strong></td>
</tr>
<tr>
<td><strong>SK_i, PK_i</strong></td>
</tr>
<tr>
<td><strong>S(x)</strong></td>
</tr>
</tbody>
</table>

In the previous example, group member $\leftarrow K_{\text{PharmCom.employee}}, \text{true}$, vote (group member, 3, 0.5) denotes that when the employee of pharmaceutical company requests to join the group, a voting is called among peers. If $k \geq 3$ votes are received and half votes among them are yes, then the requester can join the group. Specifically, the joint authorization protocol based on JXTA technology has five phases, which are group initialization, searching group advertisement, authorization request, voting, and PGC issuance.

**Group Initialization.** The group authority peer initializes the local secure environment by creating a secure peer group, and then publishes the secure peer group advertisement into the network. The group adv.
contains access control policy of peer groups and various parameters such as group name, voting type, etc.

Searching Group Advertisement. When a new peer wants to join the group, it must firstly obtain the advertisement of its attributable peer group. In this design peers have two ways to get this information. (1) Peers get the advertisement via some rendezvous points. A rendezvous point could be a special peer that keeps information about the groups, or a public website. (2) If the new node fails to obtain desired information from rendezvous peers, it will flood a query into the P2P system, and get response from other peers.

Authorization Request (Step 1 in Fig. 1). Having the advertisement message, new coming peer may connect with the corresponding authority peer. Firstly, the new node should provide the related credential OC\textsubscript{new}, which could be obtained offline from its organization. For example, the credential of Alice who is a physician of the Hospital is as $K_{Hospital\text{-}}\text{physician} \rightarrow Alice$. Then, the $M_{new}$ will generate a PGC\textsubscript{new} issuance request containing OC\textsubscript{new} information about the new node and its desired privileges.

$M_{new} ightarrow GA: \text{Role\_REQ}, S_{new}(\text{Role\_REQ}), OC_{new}$

Voting (Step 2, 3 in Fig. 1). Upon receipt of authorization request, the authority peer first verifies the signature. In a fully distributed peer group, the request is either accepted or rejected by the collective set of current members. The authority peer then propagates the request to call a vote of peers. According to the CPI, multiple peers authenticate the attribute of a requester, vote, and reply with a signed message to approve or reject the authorization request.

$GA ightarrow M: \text{Role\_REQ}, S_{new}(\text{Role\_REQ}), OC_{new}$

$GA ightarrow M: \text{vote}_{i}, PGC_{i} (\text{vote}_{i}= (RES)^{SK_{i}} \text{mod}\ n)$

5) PGC Issuance (Step 4, 5 in Fig. 1): Once enough votes are collected, GA verifies all the votes, and decide whether to accept the new node as a member. If the requester is qualified, the authority peer will issue the PGC\textsubscript{new} to it and update the related peer group information. Having the PGC\textsubscript{new}, the new node can join the secure peer group.

$M_{new} \leftarrow GA: \text{PGC}_{new}$

5. Secure Cooperation

Large sensitive data shared by multiple peers will be generated during the whole life cycle of collaboration and should not be modified by any single user. Such resources are usually stored in a stand alone peer. In the GHP scenario, the director constructed by two different developers of the peer group is a manifold role and may update sensitive data on Friday. The secure cooperation progress of multiple peers is illustrated in Fig. 2.

1) Cooperation Request: When a developer in the peer group wants to update the resource R, it propagates the cooperation request to all developers.

$M_{initiator} \rightarrow M_{i}: \text{update}\_\text{R\_REQ}, S_{initiator}(\text{update}\_\text{R\_REQ}), \text{PGC}_{initiator}$

2) Cooperation Response: Once the request is received, the peers verify the signature, and then the request is either accepted or rejected by the set of current developers.

$M_{initiator} \leftarrow M_{i}: \text{res}_{i}, \text{PGC}_{i} (\text{res}_{i} = (\text{update}\_\text{R\_RES})^{SK_{i}} \text{mod}\ n)$
3) **Cooperation Implementation**: Once enough signed responses are collected, the initiator sends all the signed messages to a stand alone peer Mserver where sensitive data is stored in. The stand alone peer will approve or reject the request according to the group policy.

\[
M_{initiator} \rightarrow M_{server}: \text{updateR}_\text{REQ}, \\
S_{initiator}(\text{updateR}_\text{REQ}), PGC_{initiator} \{ (res_1, \cdots , res_n), (PGC_i, \cdots , PGC_j) \}
\]

### 6. Implementation

We implemented the distributed access control in peer-to-peer collaborative systems using Java programming language. The communication facility among peers is provide by JXTA [2, 3], an overlay network middleware messaging system, whose functionalities include file sharing, auctions, distributed computing, and event subscription and publishing.

As illustrated in Fig. 3, our system has a three-layer architecture. The bottom layer, the JXTA core, encapsulates minimal and essential primitives similar with current P2P protocols. It has building blocks to enable key mechanisms for P2P applications, including transport, creation of peers and peer groups, and associated security primitives. The middle layer is built on top of the communication middleware, including security and network services. Examples of network services include searching and indexing, peer discovery, protocol translation, etc. There are three dominant security requirements in P2P systems: confidentiality, integrity, and availability. These translate into specific functionality requirements that include authentication, access control, encryption, secure communication, non-repudiation, membership, and group key management. Our scheme is mainly implemented in this layer.

Generally, to use a service, a peer must present its credential to service providers. We use XML to represent both access control policy and credentials. The credential has the form (issuer ID, owner ID, attribute, issue time, expiration date, peer signature). The credential is signed by the issuer. Delegation credentials should have short lifetime and they are revoked automatically when they expire.

The policy engine acts as the central agent which conformance to the security policy. All interpretation of policy occurs within the policy engine, so that multiple policy approaches could be integrated. Authentication and access control are performed by the policy engine as well. Services are protected by policies. Access control policy infrastructures are evolving with the complex environments they support. Context is used by policy to allow environmental factors to influence how and when a policy is enforced.

![Figure 3: Access control system for P2P](image)

In peer-to-peer systems, context handler mainly collects such information including message context, resource context, and group context, etc.

Our experiments are performed on 32 nodes with a high-speed LAN. All nodes are Intel Nocona Xeon 2.8GHz, 2G RAM Linux machines. As the setup phase of the peer group, the group authority creates and publishes the group authorization service advertisement. The new node sends the authorization request to the group authority. The group authority then propagates the vote request to all existing group members. All group access control protocol messages are encapsulated within standard JXTA messages. To satisfy the distributed authorization requirement and balance the group authority overhead, the group authority refresh the authorization service advertisement after delegating the authority attribute to another group member.

The group authorities may receive multi-requests in a short time interval. Figure 4 shows the 10 members’ average join cost for the existing centralized approach and our decentralized approach, in which the dynamic threshold is 20% and 50%, respectively. We can see the average cost is significantly reduced by our scheme.
7. Conclusion and future work

This paper presents a fine-grained and attribute-based access control framework for peer-to-peer collaborative systems. We propose a distributed delegation authorization mechanism to avoid single point of failure. In order to simplify authorization and access control in collaborations, decisions are made based on authenticated attributes of the peers, which improve flexibility of decentralized authorization. Furthermore, large sensitive data generated during the collaborations are managed by multiple peers and should not be managed by any single user. By applying JXTA technology, we implement this scheme.

In the future, we would like to investigate the process of multiple peers’ policy negotiation. We also intend to use the secure system to implement P2P applications having different group behavior, such as peer dynamics, group sizes, voting schemes, and different policies, etc.

8. References