Towards an Access Control System for Mobile Peer-to-Peer Collaborative Environments*

Pascal Fenkam, Schahram Dustdar, Engin Kirda, Gerald Reif, and Harald Gall

Technical University of Vienna, Distributed Systems Group
Argentinierstrasse 8/184-1, A-1040 Vienna, Austria
{P.Fenkam, S.Dustdar, E.Kirda, G.Reif, H.Gall }@infosys.tuwien.ac.at

Abstract

Access control is one of the key requirements in enterprise security. A number of approaches in distributed systems have been designed that support various (new) paradigms such as peer-to-peer, nomadic working, and teamworking. Few of them, however, explicitly take into account the possible superposition of these concepts. Such a superposition often results in conflicting and additional requirements. We present ongoing work in developing an access control system for Peer-to-Peer mobile teamwork environments. This system is developed as part of the MOTION project. The goal of this project is to develop a service architecture for mobile teamwork, providing support for various devices and taking into account diverse connectivity modes. We present the requirements for an access control system that simultaneously supports mobility, collaboration, and peer-to-peer, illustrate our solution, and discuss how it meets the requirements.

Keywords: Access Control, Security, Mobile Teamworking, XML meta-data and XQL, Mobile Computing, MOTION, Peer-to-Peer.

1 Introduction

In this paper we propose an access control mechanism for mobile Peer-to-Peer (P2P) teamwork environments. A number of research papers as well as product implementations exist that propose solutions for authorization when P2P and mobile teamwork paradigms are considered as separate requirements.

*This project is supported by the European Commission in the Framework of the IST Program, Key Action II on New Methods of Work and eCommerce. Project number: IST-1999-11400 MOTION (MObile Teamwork Infrastructure for Organizations Networking)

In the CoolTown [14] project for instance, an authorization infrastructure is designed for nomadic computing. Shen and Dewan [11] propose an access control system for collaborative environments (also called teamworking environments). Kim et al. [7] give an example of authorization infrastructure for P2P environments. Though these three concepts draw growing interest in the software engineering community we are not aware, to the best of our knowledge, of an authorization mechanism that takes all these paradigm specific requirements into consideration. Considered separately, none of the models designed for mobile computing, peer-to-peer systems or collaborative environments can meet the requirements for access control in mobile peer-to-peer collaborative environments.

In this paper, we first provide an overview of requirements for access control in mobile P2P teamworking systems. We proceed by superposing the requirements resulting from each paradigm. Conflicts are solved by giving priority to the functional requirements. The paper further presents our initial results and briefly discusses how it meets the requirements.

The remainder of the paper is organized as follows. Section 2 investigates the requirements for authorization in mobile, P2P, and collaborative systems. Section 3 gives an overview of the MOTION [8, 10] platform, a platform for mobile P2P teamworking, and presents its access control mechanism. Section 4 discusses the evaluation of the MOTION access control system regarding advocated requirements. Section 5 concludes the paper.

2 Requirements for Access Control in Mobile P2P Collaborative Systems

The requirements for an access control system in mobile P2P teamworking systems are ineluctably derived from requirements for access control in mobile systems, access

Proceedings of the Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'02)
1080-1383/02 $17.00 © 2002 IEEE

control in P2P systems and access control in collaborative environments.

Authorization for nomadic users is guided by place specific services. Users continuously encounter services while on the move. Exploiting them, however doesn't work without problems; poor connections and insufficient resources are often obstacles to mobility. Therefore, the access control mechanism must take into account *working offline from any central point of administration or control.* Though weak connection is the most advocated reason for taking offline working into consideration, there are many other examples showing that many peers indeed work offline, be it intentionally or not (tactical battlefield equipments, mobile computing in area without infrastructure, ad hoc networking, law enforcement, disaster recovery, administration and control of large events, etc. [1]). The second most important requirement for access control systems in mobile computing environments is the *support for various mobile devices.* They often lack resources for running conventional security mechanisms themselves. These problems are exacerbated when the number of services is large and services are mobile themselves.

Peer-to-Peer computing on the other hand, principally *rejects the concept of central authorization control* which is not excluded in mobile computing. The CoolTown [14] authorization system for nomadic workers is, for instance, based on a central authorization control. The reason for rejecting this concept is that many peers are often responsible for the resources they provide. Besides this, *scalability is often an issue in P2P networks.* The access control mechanism must therefore suitably face the problem of large number of peers. The access control mechanism should not be a handicap for P2P specific scenarios such as ad hoc communication and must not be handicapped by these specific scenarios (e.g. independent lifetime of peers).

Most existing P2P platforms, whether intended or not, are designed exclusively for sharing only some few types of resources [7]. Security for personal P2P systems is not as critical as for enterprises, which require high security levels for P2P collaboration within their organization as well as on the Internet. As a consequence, personal P2P systems do

not address security to the extent that is required by enterprises: *a great variety of resources need to be protected, the notion of role is increasingly required* [6]. Other requirements for access control in collaborative requirements are listed in [11]: multiple and dynamic user roles (inheritance of access rights (AR) from many roles), collaboration rights (additional access rights that match collaborative requirements), flexibility (support for fine-grained subjects, objects, and access rights). To be beneficial for enterprises, collaborative applications must provide integration facilities for third party applications (e.g. databases). This implies that their authorization mechanisms must be *sufficiently generic and be able of matching many access control mechanisms and a wide range of access rights [11].*

These requirements present an overview of the challenges faced when designing access control mechanisms for mobile, P2P, and collaborative applications separately. The superposition of these requirements leads to conflicting requirements. We enumerate two examples of such conflicts:

- Central vs. non central authorization control. Zhang and Kindberg [14] have designed an authorization mechanism for nomadic computing. The system is based on a central authorization control. A reason for this design decision is the difficulty in maintaining distributed security policies. However, for Peer-to-Peer, a centralized authorization mechanism seems to be in contradiction with the goals of this architectural style such as redundancy-induced dependability, availability through replication, etc. Peer-to-Peer security designers believe that it is natural to assume each peer as being responsible for resources it provides.

- Access rights revocation. E-Speak [7] is a P2P "e-service infrastructure that allows services to advertise, discover, and interoperate with each other in a dynamic secure way." The access control mechanism of E-Speak is built on top of the Public Key Infrastructure (PKI). It allows services to maintain certificate revocation lists (CRL). Capabilities may be revoked through CRLs, which might be a bottleneck for a set of scenarios in mobile computing: ad hoc networking and weak network connectivity, to name a few. In fact, a service working offline will not be able to work appropriately since it cannot verify that the certificate was not revoked.

3 The MOTION Access Control System

In this section, we give an overview of the MOTION architecture, and describe our authorization model.

3.1 Overview of the MOTION Architecture

The MOTION [8, 10] (MObile Teamwork Infrastructure for Organizations Networking) service architecture that we have developed supports mobile teamwork and relies on a P2P middleware. The MOTION system has a layered architecture consisting of three layers (as depicted in Figure 1). The bottom layer is the communication middleware offering basic services such as publish/subscribe mechanisms (i.e. event-based system support), peer-to-peer file sharing functionality and distributed search propagation. We refer to every computing device that is connected to the MOTION system as a *peer*. Some peers in the system host services and some only act as clients. Any computing device that is

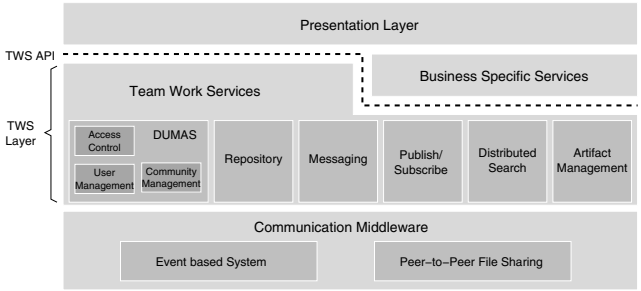

Figure 1. Overview of the MOTION Architecture

able to run the MOTION libraries can act as both, service host and client. A typical MOTION configuration, thus, consists of desktop computers, notebooks, and PDAs that can host services and also act as clients. Clients such as Web browsers or WAP enabled phones do not host services, but access them remotely. Each peer that has the technical capability hosts a repository. Artifacts (i.e. files) that the user wishes to share with others in communities are stored in this repository along with the corresponding XML metadata (i.e. *profile* in the MOTION terminology).

The repositories hosted on peers also store community and user profile information that are replicated across some peers and are synchronized using events. From the user perspective, profiles are used to support queries based on descriptions of *resources* (e.g. communities, users, and artifacts). From the system point of view, the profiles are also used to store system-relevant information such as user names and last modification timestamps.

The middle layer in the architecture, the Teamwork Services (TWS) layer, is built on top of the communication middleware. It is responsible for the integration of the main components of the system such as the repository, access control, community management, and user management component. Furthermore, it provides an Application Programming Interface (API) to the following generic services: 1) Storing and retrieving artifacts in the local repository and from remote repositories on other peers. 2) Managing *resources* (e.g. artifacts, users, and communities) and their profile information. 3) Creating and managing communities. 4) Sharing of artifacts with other users within communities 5) Subscribing to specific events in the MOTION system (e.g. artifact insertion, artifact deletion, user joining a community, etc.). 6) Sending and receiving messages to and from other users. 7) Receiving system messages and notifications. 8) Distributed searching for artifacts, users, and communities based on their profiles.

3.2 Access Control in MOTION

3.2.1 Conceptual Overview

The access control facility is provided in the MOTION system by DUMAS (Dynamic User Management and Access Control System. See Figure 1). This component provides three main functionalities: access control, user management, and community management.

A community represents a set of users sharing some resources. Operations on communities include creating them, deleting them, making them subcommunities of other communities and removing them from other communities. We identify the notion of community as a teamworking view of the concept of roles discussed in the access control arena. Thus, the community management facility of DUMAS is in fact a role management facility.

User management consists of registering new users in the MOTION environment, assigning them to and removing them from roles, and deleting them. Obviously, users inherit access rights from the different roles they are members of.

The access control facility is composed of primitives for assigning access rights to users and roles, removing access rights from them, defining new access rights and assigning semantics (such as methods) to them. Through the capability of defining new access rights, MOTION supports different business specific requirements, and integration of third party products. This paradigm is further discussed in [2].

The architecture of DUMAS is inspired by the underlying P2P file sharing middleware and the requirement for mobility support. Two categories of peers are distinguished. L1 peers (Peers of Level 1), and L2 peers.

A summary of this categorization is shown in Table 1. A primitive such as adding a user to a community consists of three steps: validating the set of authorization certificates presented by the user (labeled V in the table), executing the real function (E), and publishing the event (P). Peers perform all or a subset of these tasks depending on their levels.

L1 peers are peers that have the capacity of maintaining a regular security infrastructure. This includes the complete intelligence for assigning permissions, removing permissions, validating and storing access control lists (ACLs). An important characteristic of an L1 peer is that it is responsible for protecting some resources (whether provided by itself or other peers). The responsibility of protecting a service includes storing ACLs related to this service and delivering authorization certificates. To use a service, a user must present his capabilities to the service provider (See Figure 2). These capabilities are authorization certificates (ACs), delivered by L1 peers based on the hosted ACLs. Each peer disposes a public/private key pair.

L2 peers are devices lacking the resources for instantiating the full DUMAS engine. They rely on a special implementation of DUMAS, which functionalities consist of:

Category	Function	AR Storage	AC Delivery
L1	V, E, P	ACLs	Yes
L2	V, P	ACs	No

Table 1. Peer Categorization in DUMAS

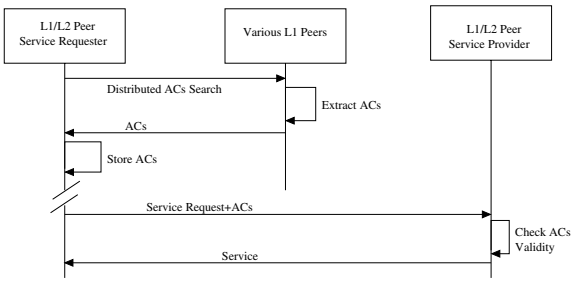

Figure 2. Authorization Process

- verification of authorization certificates. Some devices (mobile or not, online or not), though disposing weak resources, might provide services to users and therefore need to protect them. In this case ACLs are stored on some L1 peers . The L2 peer needs the intelligence to verify authorization certificates related to the service it provides.

- storing of a subset of authorization certificates. For example, an L2 peer P_{21} might belong to some nomadic user who needs to interact with another L2 peer P_{22} later on. The communication might happen in a scenario where both L2 peers are offline. For this, the nomadic user stores his authorization certificates related to services provided by P_{22} on P_{21}.

- storing of public keys of peers responsible of protecting the provided resource.

Generally, to use a service, a user must present his authorization certificates to service providers (see Figure 2). While he is online, the user performs a distributed search specifying the kind of authorization certificates he is looking for. Some L1 peers can reply by sending some authorization certificates for a short validity period. These authorization certificates are small enough (up to 300 bytes) so that they can be stored on L2 peers and presented to service providers in exchange of services. An authorization certificate has the form **(access right, user ID, object ID, expiration date, peer signature).** The signature is that of the L1 peer delivering the AC. This peer must be registered as one of the peers responsible for managing ACLs for the service for which it delivers the authorization certificates.

Any action performed in the MOTION environment is accompanied with a set of authorization certificates. Operations for assigning access rights to and removing them from users and communities are sent asynchronously to responsible L1 peers. The communication between peers is event based. This implies in fact that any peer can intercept the published event and later on distribute authorization certificates in a non authorized manner. However, these authorization certificates will not be accepted by the service provider since such authorization certificates are not delivered by registered L1 peers.

3.2.2 Implementation

The actual implementation of our access control system is done using the Java programming language. The communication facility between peers is provided by PeerWare [9], a middleware whose functionalities include peer-to-peer file sharing, mobile code deployment, distributed search propagation, and event subscription and publishing. Authorization certificates and ACLs are represented using the eXtensible Markup Language (XML [12]). Search for authorization certificates is performed by using XQL [13], a query language for XML.

Users might search for their authorization certificates on the MOTION network using various criteria: access right name, service keywords, name of the peers possibly delivering the authorization certificates, etc. An example of such a query is given below:

```
AuthorizationCertificate[
Target/ID $startswith$ '+43699'
$and$ Target/Type $contains$ 'Telephone'
$and$ @peer $startswith$ 'motion'
$and$ @Owner='fsigmund']
```

The purpose of this query is to search for any authorization certificate that the user `fsigmund` has on the numbers starting with `+43699`. The search must be performed only on L1 peers whose name start with `motion`. This query is distributed to appropriate peers. These peers locally apply the XQL query to their ACLs (using the GMD XQL engine [4]), extract the set of matching resources, include an expiration date, sign the information and return the result. These queries might seem complicated for end users, therefore we have designed an appropriate graphical user interfaces that, we believe, allows end users to formulate such queries (See Figure 3).

An example of a authorization certificate returned by an L1 peer is shown below.

```
<AuthorizationCertificate
     Signature='R24fd64he64hg7448'>
     <ExpirationDate  Date=21.03.2002
                      Time='15:34'/>
     <Owner  ID='fsigmund'/>
     <Target Type='Telephone'
                    ID='+43699111'/>
```

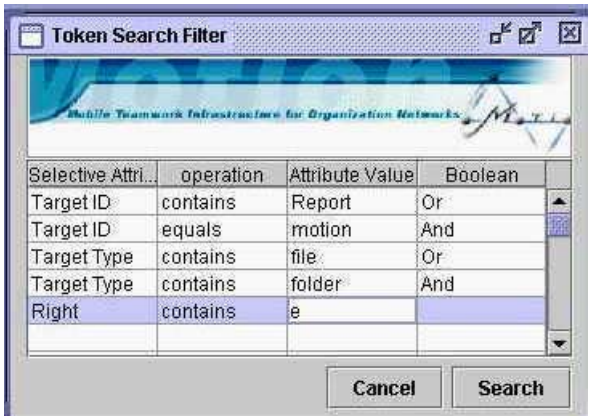

Figure 3. Formulating AC Search Queries

```
    <Right   Name='dial'/>
</AuthorizationCertificate>
```

The current efforts in the realization of our access control system are deployed 1) for the public key infrastructure, and 2) for the verification of the whole protocol. A part of the access control system was already formally specified using the formal specification language VDM-SL [3]. This formal specification was used for deriving test cases for our Java implementation. However, VDM-SL is not very well suited for specifying protocols. We are now translating the formal specification to Alloy [5], a formal specification language for micro-model of software. Its analyzer allows complete and automatic verification of various properties.

4 Discussion

We briefly discuss the fulfillment of the requirements mentioned in section 2 through our authorization infrastructure.

1. Decentralization of control: access rights information can indeed be stored on every L1 peer, provided this peer is known by the peer providing the service. Replication of information can thus be achieved among various peers.

2. Offline working: A peer can in fact be offline and still be in the MOTION network, providing its services to nomadic users. These users need to search their authorization certificates from online peers. The offline peer needs to be informed each time a new peer is added to the list of peers allowed to deliver authorization certificates.

3. Support for various access control systems: DUMAS allows syntactic and semantic definition of access rights at runtime. This implies that access rights available in other applications can easily be defined and therefore the MOTION's access control system made compatible with these applications. Further, DUMAS supports assigning access rights both to users and to roles. In this sense, application can be integrated that have either role base access control (RBAC) or user oriented access control.

4. Support for mobile devices: authorization certificates delivered by MOTION peers are small enough such that a PDA can store up to 1000 of them without problems.

5. Access rights revocation: two types of access control data are distinguished. On one side, there are ARs stored on L1 peers. These ARs can be revoked by simply removing the corresponding entry from the XML repository (DUMAS provides primitives for this). Other L1 peers are informed by means of the publish/subscribe mechanism and can update their knowledge. The other kind of data are authorization certificates. Authorization certificates have such a short lifetime that they are revoked automatically when they expire.

6. Collaboration: DUMAS maps the notion of community to roles. However, when a user searches for his authorization certificates, he will receive a authorization certificate if he is either in a community having the access right he is looking for or he was explicitly assigned this permission. All the requirements listed in [11] are satisfied by our authorization systems. Users may be member of many roles and inherit access rights from these roles. DUMAS supports any finite number of access rights. Our model is sufficiently fine grained: permissions can be assigned to users and roles.

7. RBAC: Jaeger and Prakash [6] have identified the enhancement of Role Based Access Control through discretionary access control (DAC) as an important requirement for collaborative systems. This requirement allows decentralization of the administration. A community leader can be defined for each community and given the right to further assign this right to members of the community. Access rights in DUMAS are objects on which users or communities might have permissions. These permissions include the permission to delegate this access right or not, thus the support for discretionary access control.

8. Administration: we have developed a graphical user interface for easy administration of the access control system for L1 peers. Future work includes evaluation of the usability of this user interface.

9. Scalability is an issue that we are now investigating: how to restrict the search scope of queries. This is not a problem if a user specifies the peers from which authorization certificates must be collected. We cannot assume, however, that users will always do so. We have, therefore, to find solutions for automatically narrowing down this scope. Another issue is that we strongly depend on the scalability of the underlying event based system. We assume that an event published by a peer will be delivered in time to subscribed recipients. Without this assumption, it is difficult to argue for instance that an access right removed from a user is indeed removed from that user. In small experimental settings, we have been satisfied.

5 Conclusion

The goal of this paper has been to present our ongoing work in developing an enterprise-wide access control system for mobile peer-to-peer collaborative environments. Current access control systems relevant for the investigated research area are build for either personal usage of peer-to-peer systems or for mobile environments and do not consider highly distributed enterprise scenarios. However different usage scenarios lead to superposition of requirements and as a consequence to conflicts. The access control system presented in this paper takes the different requirements advocated for these paradigms into consideration and solves conflicts by giving priority to functional requirements. The contributions of this paper are manifold: The presented access control system is generic and supports any finite number of access rights. It supports discretionary access control as well as RBAC. Distributed search are performed using XQL. Furthermore it is implemented for highly decentralized architectures. We showed how Access Control Lists (ACL) may be hosted on any peer in a P2P system and not only on the peer providing the actual service or on a single central peer. This enables our solution to be highly available. We have discussed the fulfillments and implementation issues of access control requirements realized in our infrastructure. As we hope that our infrastructure will be a reference in the area, correctness of is an issue that we are investigating in our future work.

References

[1] S. Basagni, I. Chlamtac, and A. Farago. A generalized clustering algorithm for peer-to-peer networks. In *Workshop on Algorithmic Aspects of Communication, satellite workshop of ICALP'97, invited paper, Bologna, Italy*, July 1997.

[2] P. Fenkam, H. Gall, G. Reif, and E. Kirda. A Dynamic and Customizable Access control System for Distributed Applications. Technical report, Distributed System Group, Technical University of Vienna, December 2001.

[3] P. C. Fenkam. Dynamic user management system for web sites. Master's thesis, Graz University of Technology and Vienna University of Technology, September 2000. Available from http//www.ist.tu-graz.ac.at/publications.

[4] GMD. Xql ipsi, http://xml.darmstadt.gmd.de/xql/, 2002.

[5] D. Jackson, I. Shlyakhter, and M. Sridharan. A micromodularity mechanism. In *Proc. ACM SIGSOFT Conf. Foundations of Software Engineering/European Software Engineering Conference (FSE/ESEC '01), Vienna*, September 2001.

[6] T. Jaeger and A. Prakash. Requirements of role-based access control for collaborative systems. In *Proc. of the 1st ACM Workshop on Role-based Access Control; Gaithersburg, MD*, November 1995.

[7] W. Kim, S. Graupner, and A. Sahai. A secure platform for peer-to-peer computing in the internet. In *Thirty-Fifth Annual Hawaii International Conference on System Sciences*, January 2002.

[8] E. Kirda, H. Gall, G. Reif, C. Kerer, and P. Fenkam. TWS-API: A generic teamwork services application programming interface. In *IEEE Proceedings of the Internation Workshop on Mobile Teamwork 2002*, To Appear.

[9] G. P. Picco and G. Cugola. PeerWare: Core Middleware Support for Peer-To-Peer and Mobile Systems. Technical report, Dipartimento di Electronica e Informazione, Politecnico di Milano, 2001.

[10] G. Reif, E. Kirda, H. Gall, G. P. Picco, G. Cugola, and P. Fenkam. A web-based peer-to-peer architecture for collaborative nomadic working. In *10th IEEE Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE), Boston, MA, USA*. IEEE Computer Society Press, June 2001.

[11] H. Shen and P. Dewan. Access control for collaborative environments. In J. Turner and R. Kraut, editors, *Proc ACM Conf. Computer-Supported Cooperative Work, CSCW*, pages 51–58. ACM Press, 31 –4 1992.

[12] World Wide Web Consortium. eXtensible Markup Language. Technical report, W3C, May 2000. Available from http://www.w3.org/XML/.

[13] World Wide Web Consortium. XQL Query Requirements. Technical report, W3C, January 2000. Available from http://www.w3.org/TR/2000/WD-xmlquery-req200031.

[14] K. Zhang and T. Kindberg. An authorization infrastructure for nomadic computing. In *Proceedings of the 7th ACM Symposium on Access Control Models and Technologies (SACMAT)*, To Appear.