uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Kaining Wang

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Master of Computer Science

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Context-based Coalition Access Control for Ubiquitous Computing

TITRE DE LA THÈSE / TITLE OF THESIS

Ramiro Liscano

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Ivan Stojmenovic

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Michel Barbeau

Luigi Logrippo

Gary W. Slater

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES /
DEAN OF THE FACULTY OF GRADUATE AND POSTDOCORAL STUDIES

Context-based Coalition Access Control for Ubiquitous Computing

By

Kaining Wang

Thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

in partial fulfillment of

the requirements for the degree of

Master of Computer Science

Under the auspices of the

Ottawa-Carleton Institute for Computer Science

University of Ottawa

Ottawa, Ontario, Canada

Your file Votre référence
ISBN: 0-494-14967-1
Our file Notre référence
ISBN: 0-494-14967-1

I hereby declare that I am the sole author of the thesis.

I authorize the University of Ottawa to lend the thesis to other institutions or individuals for the purpose of scholarly research.

Kaining Wang

I authorize the University of Ottawa to reproduce the thesis by photocopying or other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Kaining Wang

Abstract

The need for coalition access control among individuals and organizations has increased significantly in the past years as the need for spontaneous access to information increases. However, a significant deterrent to the ability to connect in a spontaneous manner in coalition collaborative applications is the difficulty in users from different domains being able to access resources or services located and owned by other entities. Coalition access control encompasses control mechanisms dealing with access between users of two or more different organizations or enterprises. These users could be co-located or remotely located.

The thesis first presents a delegation based D-TMAC model that extends traditional TMAC across organizations for formal coalition environments, and a context-based coalition access control model, which apply context information as conditions on delegation. Then the thesis proposes a Session-based Coalition Access Control Architecture (SCACA) and provides practical implementation that enables dynamic coalition access control over a communication session in a spontaneous manner. The presented system architecture and methodology leverages the IETF SIP protocol as an underlying communication mechanism in order to greatly minimize the administration overhead and rapidly adapt the dynamic nature of access control in spontaneous coalition environments.

The result is that, during a spontaneous coalition communication across organizations, every endpoint can access other endpoints' resources and share its own resources to all the other endpoints as well. Moreover, these privileges will dynamically change as the status of the coalition communication changes.

Keywords

Coalition, Spontaneous Coalition, Access Control, Delegation, SIP

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my supervisor Dr. Ramiro Liscano for his invaluable guidance, continuous supports, and numerous encouragements. Without him, I wouldn't be able to accomplish this work. I greatly thank my supervisor Dr. Ivan Stojmenovic for his trust, support, encouragement and guidance.

I would like to thank everyone from the wireless security research group of the Carleton University for their patient listening and useful suggestions. I would like to thank my dear friends at University of Ottawa and Carleton University: Jianqiang Shi, Li Wu, Shaoying Zou, Pin Zhao, Huaizhou Ding for the fun, serious support, and wonderful talks.

The love and support provided by my wife and my parents have been a source of strength and comfort throughout my life. Words cannot express the love I have for them.

Finally, I would like to thank the financial supports from Alcatel, MITACS, and the University of Ottawa.

Table of Contents

List of Abbreviations

Abbreviation or Acronym	Meaning
AC	Access Control
ACL	Access Control List
ACM	Access Control Matrix
CA	Collaborative Application
CBAC	Coalition-Based Access Control
DA	Directory Agent
IETF	Internet Engineering Task Force
IM	Instant Messaging
PEP	Policy Enforcement Point
PDP	Policy Decision Point
RBAC	Role-Based Access Control
RTP	Real-Time Transport Protocol
SA	Service Agent
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SLP	Service Location Protocol
SM	Security Manager
SPKI	Simplified Public Key Infrastructure
S Types	Situational Types
TMAC	TeaM based Access Control
UA	User Agent
UC	Ubiquitous Computing
URI	Uniform Resource Identifier
URL	Univeral Resource Locator
XACML	eXtensible Access Cotnrol Markable Language
XML	Extensible Markable Language

List of Tables

List of Figures

Chapter 1 Introduction

Information Security deals with all "trust" aspects of information, and it covers not just information itself, but also all infrastructures that facilitate its use - processes, systems, services, equipments, and networks. Recalling its evolution in the past barely half a century, a major driving factor of information security is information technology itself, e.g. new applications, networks. The next major evolutionary step, in part already underway, is brought by ubiquitous computing. The thesis explores the issues on the intersection [Stajano02] [Adams05] of the two topics: Ubiquity and Security, particularly the access control mechanisms in spontaneous coalition environments.

This chapter introduces the notion of spontaneous coalition, defines some basic terminologies, and also points out the principal access control concerns that people shall be facing in spontaneous coalition environment.

1.1 Problem Statement

The focus of this research shall be the investigation of access control mechanisms in spontaneous coalition environment. The Merriam-Webster Online Dictionary[1] defines "coalition" as

A temporary alliance of distinct parties, persons, or states for joint action.

[1] Source: http://www.m-w.com/cgi-bin/dictionary?book=Dictionary&va=coalition, accessed on Oct. 2005

Although parties may have their own self-interest and play different roles in a coalition, all parties share resources within a coalition in order to achieve a common purpose. Basically, security is always a critical issue in a coalition. The oldest inter-organizational coalition appeared thousand years ago long before the modern computer technology emerged. A typical example of an ancient coalition was the composition of Alexander the Great's Army. Alexander could not have succeeded without a coalition army, for even the greatest military strategy in the world requires troops.

In modern society, there are two different forms of inter-organizational coalitions. The author calls the first form of coalition "*Formal Coalition*", the business process/task oriented inter-organizational coalition [BKFHDW04], which is increasingly used to improve the partners' competitive position in a global world economy. This coalition is a contractually or implicitly agreed collaboration between independent companies or organizations. Particularly, the objective of most formal coalitions is to achieve a specific common purpose.

Relatively stable, formal coalitions are made for a medium- or long-term cooperation, and usually require organizational investments from the participating companies or organizations. Examples of these investments may be time and human resource, to be spent on negotiating coalition agreements including common ontology, coalition policies, etc. or investments in information technology.

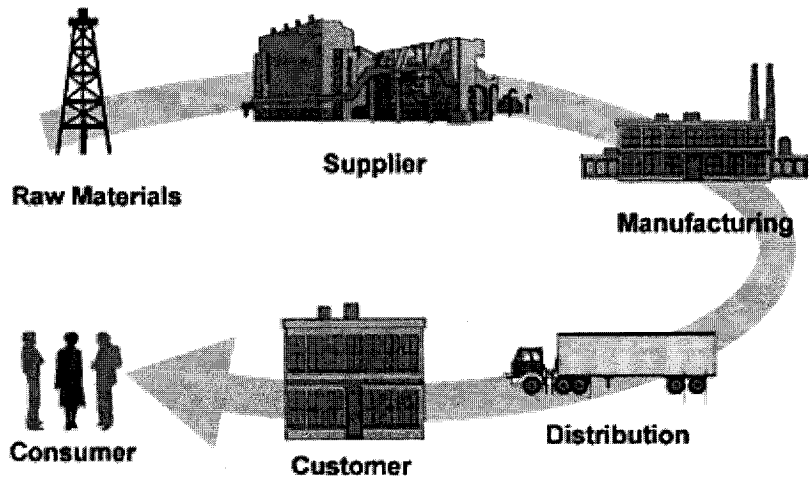

Figure 1. A Formal Coalition Example – Supply Chain Management

The economic effects, such as cost reduction, time saving, quality advantages and reduction of risks, are ultimately achieved by effectively combining resources in each enterprise or organization. Real examples of formal coalition are supply chain management (shown in Figure 1), international joint projects, logistics services, etc. Some commercial applications, like SAP, have total solutions [BKFHDW04] already in place. In general, many existing access control mechanisms (See chapter 3) for resources in participating partners require common inter-organizational agreement and non-negligible administration overhead, and shared resources are fixed most of the time, and tightly related to a business process or a task.

Nevertheless, besides business process or task-oriented formal coalition, most inter-organizational coalitions are actually spontaneous and short-term in nature. Thomas J. Allen reported the results of his pioneering series of studies in the book Managing the Flow of Technology [Allen84] on the factors that predict the success of research projects and the promotion of innovation. Allen found that informal communication was the

prime means by which useful information flowed both into and within an organization. The author is more interested with this informal communication between parties, which is called "*Spontaneous Coalition*" in the thesis, the second form of inter-organizational coalition. A spontaneous coalition refers to an ad hoc collaboration between multi-parties.

The spontaneous coalition is an increasing trend in recent years. It is continuously driven by many factors. Since the inception of the last decade, much has changed in the networking world. Advances in high bandwidth network, wireless and mobile communication, and mobile devices, have realized "anywhere, anytime" connectivity. These have increased the communication between people no matter where they are and when they want to communicate. Moreover, some recent social issues greatly drive these technologies moving forward. The European Telecommunications Resilience and Recovery Association analyzed and reported that the need for teleconference increased dramatically in the Post-September 11 world [ETR2A02].

Spontaneous coalitions are Instant Messaging, Internet telephone calls, multimedia conferences. However, it's not limited to the above. A popular example of spontaneous coalition is a teleconference between callers from several organizations. During the conference, each caller can share his/her services to others. Normally there is no particular formal task need to be achieved in this kind of coalition, organizational level agreement is not necessary, the relationships between participating parties are relatively simple, and available resources for every single coalition may be different. As such the dynamic nature of spontaneous coalition brings new challenges for coalition access

control mechanisms. The necessary requirements for access control mechanism for spontaneous coalition are summarized as follows:

1. The access control mechanism must be well defined to suit the dynamic and ad hoc nature of the spontaneous coalition;
2. The administration overhead should be minimal;
3. The administration and user interface should be simple and easy to use;

In general, it is difficult to satisfy the above requirements using existing access control models (further discussions in chapter 3). They generally start their approaches from coalition agreement by formalizing the relationships between organizations within the coalition, and building a high level policy and access control model on top of those of every participating organization. However, there are two obstacles if they are applied in spontaneous coalition environments.

- Firstly, inter-organizational spontaneous coalitions are diverse and the parties within the coalition may not only limit to several, could be many or huge (more discussion on this scalability issue in section 2.1.3). It is a tedious task to catch the complicated relationships within a coalition.

- Secondly, defining high-level access control model and policy brings down the problem domain to inter-departmental relation within an organization (further discussion in chapter 3), and hence the coalition administration overhead becomes an issue.

It is to be expected that, in the near future, the ubiquitous networking (more on this in the next chapter) and spontaneous coalition will become much more general. However, security is becoming a major obstacle for this trend. The author envisions that solving this problem requires new mechanisms. The objective of this research is to investigate a new access control mechanism for spontaneous coalition.

1.2 Scenario

Nowadays, telephone, video and instant messaging conversations are becoming a standard norm for spontaneous coalition. The following is a typical scenario that will be dealt with in the thesis.

A voice or video teleconference (shown in Figure 2) will be held between two meeting rooms, roomA and roomB, located respectively in two different companies, CompanyA and CompanyB. Alice is an employee of companyA, and she is the initiator of this conference. Bob is a member of CompanyB as well as one of the participants attending the conference and is located in his company's meeting roomB. The purpose is to allow all the participants attending this teleconference from different domains to get controlled accesses to resources related to this conference call. For example, the accesses to the resources in meeting roomA of companyA are only permitted during the conference and to those members participating in the conference call that is the communication session. As such once the meeting ends, all the access privileges are gone.

The following is an illustration that the across organization teleconference forms a spontaneous coalition, and some related services/resources (e.g. printers, whiteboards,

projectors) are attached to this spontaneous coalition, so that every participating members can access all those resources attached to this teleconference. Please note that, in CompanyB, the whiteboard is not attached to the teleconference, so users from foreign domains cannot access it, even though they are participants of the ongoing teleconference.

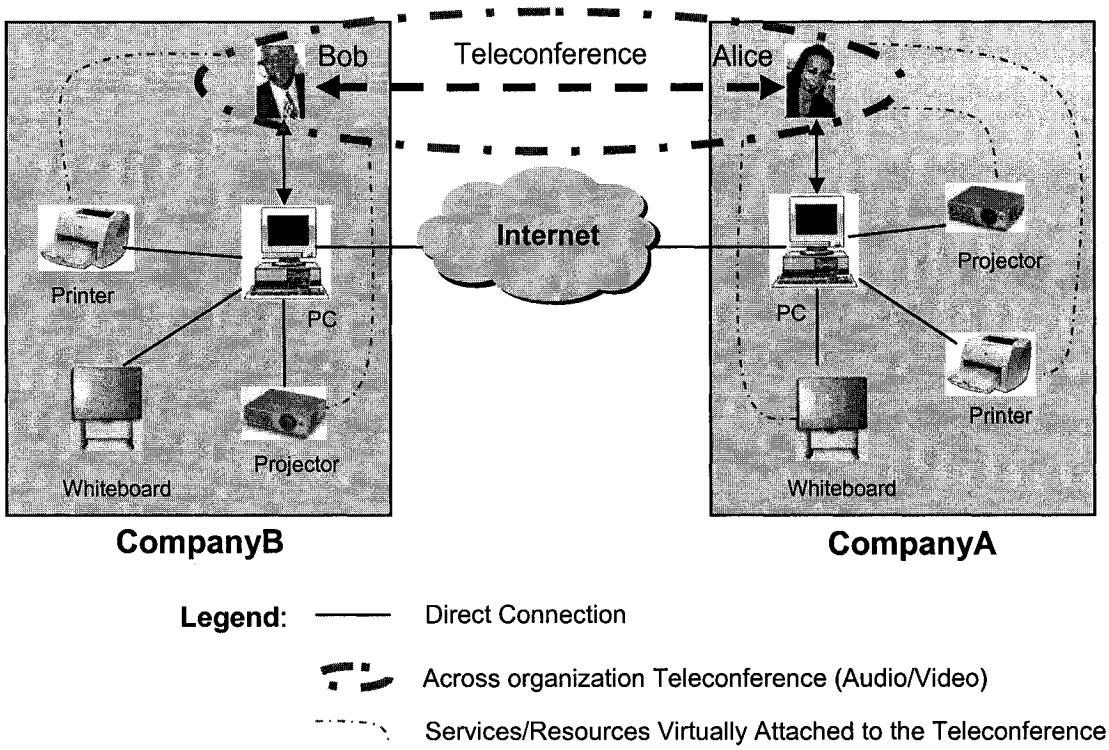

Figure 2: A Typical Teleconference Scenario

In this scenario, the number of participating parties/companies within the teleconference may not only limited to two, could be five, ten or hundreds, and the number of users on each participating party/company may not only limited to one either. A more practical situation is that a group of people is sitting in a meeting room of CompanyA, and another group of people is sitting in a meeting room of CompanyB. An ideal approach should be

able to handle these scalability issues with low cost and minimal administration overhead.

1.3 Overview of the Research

This thesis aims at developing mechanisms to facilitate access control in spontaneous coalitions. According to the discussed requirements for the access control in spontaneous coalitions (section 1.1), the proposed approaches must meet the following objectives:

1. The administration overhead can be reduced;
2. It must suit the dynamic and ad hoc nature of spontaneous coalitions;
3. The proposed approach is scalable in the number of parties in the spontaneous coalitions;

The thesis has five primary research contributions:

1. Analyze two types of inter-organizational coalitions, and propose a notion of "Spontaneous Coalition" (See section 1.1);
2. Define three types of Collaborative Application Types (CA Types) to categorize existing different access control approaches, and three particular situational type scenarios representing the situational context in which most collaborative sessions might occur (See section 2.3);
3. D-TMAC, an extension to the TeaM based Access Control (TMAC) model [Thomas97]. In order to form a team with members from different organizations,

across organization delegation is used in the D-TMAC model. "Team" is a notion first proposed in the TMAC model;

4. An extension to the distributed Role Based Access Control (dRBAC) model [FPPK02] by applying context information conditions into the definition of the delegation to provide more flexible and fine-grained access control. This context sensitive capability allows an access control application to adapt dynamic environments;

5. A Session-based Coalition Access Control Architecture (SCACA) for building and rapid prototyping of dynamic coalition access control management and services for spontaneous inter-organizational coalition. The IETF standard SIP session is leveraged in this architecture and a session role is proposed for managing across organization coalitions. The proposed architecture can automate the access control in a spontaneous coalition by securely maximize the available services on each end of participating parties.

1.4 Publications Resulting from This Research

Two papers have been published. One presentation and one poster were presented at a workshop and a conference respectively:

1. Ramiro Liscano, Kaining Wang, "*A SIP-based Architecture model for Contextual Coalition Access Control for Ubiquitous Computing*", In Proceedings of the Second Annual International Conference on Mobile and

Ubiquitous Systems: Networking and Services (MobiQuitous2005), San Diego, CA, USA, July 17-21, 2005 (Section 4.2, Chapter 3, 5)

2. Ramiro Liscano, Kaining Wang, "*Coalition Access Control for Spontaneous Networking*", In Proceedings of the Workshop on the Practice and Theory of Access Control Technologies (WPTACT), Montreal, Canada, January 19, 2005 (Section 2.3, 4.2, Chapter 5)

3. Ramiro Liscano, Kaining Wang, "*Context-based Coalition Access Control for Spontaneous Networks*", Presentation, the Workshop on New Challenges for Access Control (NCAC), Ottawa, Canada, April 27, 2005 (Section 2.3, 4.2, Chapter 5)

4. Kaining Wang, Ramiro Liscano, "*A Survey of Coalition Access Control Mechanisms*", poster presentation, the MITACS 5th Annual Conference, Halifax, NS, Canada, June, 2004 (Section 2.3, 3.1, 3.2)

1.5 Thesis Organization

The rest of the thesis is outlined as follows: Chapter 2 provides a brief introduction about related technologies and analyses of relations between the thesis topic and them. Chapter 3 gives an overview of previous work related to this research. Chapter 4 introduces D-TMAC, an extension to TeaM based Access Control (TMAC) model, and an extension to distributed Role Based Access Control (dRBAC) model. Chapter 5 presents the Session-based Coalition Access Control Architecture (SCACA). Chapter 6 validates the designed

system framework through proof of concept implementation, and Chapter 7 concludes the thesis and discusses the future work.

Chapter 2 Technical Background

2.1 Ubiquitous Computing

Many years ago, commercial companies started to embed microprocessors in everyday items – mobile phones, televisions, cars, wristwatches, washing machines, microwave ovens. From then on, PC technology is showing up in more and more consumer electronics devices. A recent example is the Dish Network 921[2] HD-DVR (Figure 3) Released in 2004, a Via Centaur based PC, with some extra circuit boards for handling the HDTV and satellite tuning chores. It runs a version of embedded Linux.

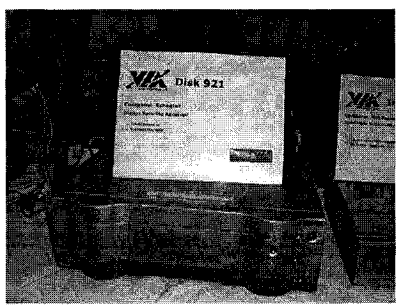

Figure 3. The Dish Network 921

At that time, most of these devices worked stand-alone. However, the continuous and fast development of wireless communication and chip technologies has enabled a rapid and huge change in this situation, which is an established trend to network these devices at very low cost. TVs, PCs, PDAs, stereos, kitchen appliances will be able to talk to each other, and electronic devices are becoming networked in some specific environments as well. For example, medical instruments such as heart monitors and blood oxygen meters

[2] Source: *http://www.dishnetwork.com/content/products/receivers/HD/index.shtml,* accessed on Sept.2005

are able to transfer real time patient's data to central monitor centers. Moreover, besides connectivity, each device may be able to interoperate with each other by taking advantage of services offered by other nearby devices. This is the world of Ubiquitous Computing, and it is to be expected that this networking will become much more general in the near future.

2.1.1 Terminology

The Merriam-Webster Online Dictionary[3] defines "ubiquitous" as

> *Existing or being everywhere at the same time; constantly encountered*

The term "*Ubiquitous Computing*" was coined by Mark Weiser in 1988 [Weiser91] to describe a future in which computers cease to be the focus of attention but instead ultimately vanish into the background, "weaving themselves into the fabric of everyday life until they are indistinguishable from it". Weiser clearly predicted and positioned ubiquitous computing as follows:

> *"Ubiquitous computing names the third wave in computing, just now beginning. First were mainframes, each shared by lots of people. Now we are in the personal computing era, person and machine staring uneasily at each other across the desktop. Next comes ubiquitous computing, or the age of calm technology, when technology recedes into the background of our lives." -- Mark Weiser*

[3] Source: *http://www.m-w.com/cgi-bin/dictionary?book=Dictionary&va=ubiquitous, accessed on Oct., 2005*

2.1.2 Device vs Human and User Mobility

A lot of research has been done since Weiser proposed this great idea, but different people have different views of ubiquitous computing. Basically there are always two important aspects in a ubiquitous computing environment: devices and human.

As discussed above, the first step of the evolution of ubiquitous computing is focused on the devices, which is embedding chips in devices to make them not only capable of computing but also of communication. This enables the "silent" or "calm" communication, interoperation, organization between devices without user's intervention.

Weiser's ideas about the relationships between people and smart things break the myth in the technology industry about how technology transforms people. Instead, the truth is an ongoing conversation and mutual adaptation between human being and technology. The ultimate purpose of ubiquitous computing is to make the systems or applications human-centric. Users can focus on what they want to do, minimizing distractions, and allowing devices to learn, adapt, and facilitate users' interactions with their surroundings. Weiser hoped to create a world in which people interacted with and used computers without thinking about them.

In order to achieve the human-centric idea, there is an important concept "Mobile Computing" needs to be addressed and clarified here, which is a key topic in ubiquitous computing research and closely related to this thesis. Many buzzwords are used to describe "mobility", "the ability to change locations while connected to the network" [FZ94]. They are mobile, ubiquitous, nomadic, pervasive, roaming, and anytime

anywhere. Terminal mobility [4] is made possible through wireless communication technologies and advances in the IP technology. There is a lot of research and commercial products have been done and developed in this area. However, mobility is not only limited to the device that's mobile and the user is attached to the device, but also user mobility at a high level. Li defined "User Mobility" in his thesis [Li01] as

> *"The ability of users to access defined services from any terminal in the network (including user's mobile devices) while maintaining their personal computing environment."*

User mobility does not restrict the underlying network infrastructure, which could be a relatively fixed network infrastructure. For example, a user updates his/her contact information at a SIP registrar server to make himself/herself continuous reachable when he/she leaves his/her office for a meeting room. Then the new coming calls will be automatically forwarded to the meeting room instead of his/her office. The thesis scenario described in section 1.2 can be extended to support this user mobility, so that a meeting participant can keep staying in the teleconference and accessing remote services while he/she moves around. The thesis addresses this important issue and tries to provide a solution, which is based on Session Initiation Protocol (SIP) as SIP technology can strongly enable user mobility. (See section 2.2)

[4] Terminal Mobility: In commercial wireless networks, the ability of a terminal, while in motion, to access telecommunication services from different locations, and the capability of the network to identify and locate that terminal. (Source: *http://www.its.bldrdoc.gov/fs-1037/dir-037/_5401.htm, accessed on Sept. 2005*)

2.1.3 Personal UC vs Large Scale UC

The traditional notion of ubiquitous computing [Weiser91] focuses on that a user might interact with a number of "smart" devices at a time, each invisibly embedded in the environment and wirelessly communicating with each other. Chetan et al [CACD04] define this kind of ubiquitous computing environment as "*Personal Ubiquitous Computing*", which are "formed ad hoc using personal devices carried by a person as well as devices that are in close proximity".

In the past decade, there have been numerous research efforts in personal ubiquitous computing. Most researchers have focused their research on wireless network infrastructure. Although the communication and computing are ubiquitous, the underlying network infrastructure might not be ubiquitous, and it can be a relatively fixed or distributed network. Nowadays some researchers notice that and believe it is time to move to the large scale ubiquitous computing [BSSW03] [RL04]. "*Large Scale Ubiquitous Computing*" refers to "a global-scale ubiquitous computing system that is secure, administered by multiple independent administrators and integrates off-the-shelf hardware and software" [BSSW03]. This is a quite new and very interesting research area, and the problem domain of the thesis is right in this area because a spontaneous inter-organizational coalition is a kind of a large scale ubiquitous computing environment.

In a large scale ubiquitous computing environment, scalability of the whole system is a key requirement. In addition, because all the devices and users may be in different places or under different administrations, heterogeneous systems must be able to interoperate

automatically; decentralized management and easy deployment will be most likely suitable; security and privacy become big issues and they must be modeled into system architecture.

The thesis focuses mainly on designing access control models and an architecture for a specific large scale ubiquitous computing environment, that is, coalition environment. Some other key issues, such as scalability, authentication, will be addressed in the thesis as well.

In general, ubiquitous computing encompasses a wide range of research topics, including distributed computing, mobile computing, sensor networks, human-computer interaction, and artificial intelligence.

2.2 Session Initiation Protocol (SIP)

As discussed in chapter 1, most formal coalitions are business process or task oriented, medium- or long-term coalition, and their implementations are mainly work flow system. Nevertheless, spontaneous coalitions are formed by Instant Messaging, Internet telephone calls, multimedia conferences, and it's not only limited to the above. During these sessions there are times when the participants want to share data and/or access resources that are not hosted on their own organizations but are available on other participants' domains. As such the author concludes that most spontaneous coalitions are over a communication session. In another aspect, telephone, video and Instant Messaging conversations are now becoming a standard norm for multi-party communications.

The Session Initiation Protocol (SIP) is an application layer signaling protocol that defines initiation, modification and termination of interactive, multimedia communication sessions between multiple users. Open, flexible, and extensible, SIP, the session management protocol, is becoming a worldwide dominating standard in both IP and Telecommunication communities. One reason for this rapid acceptance is that SIP is an incredibly powerful call control protocol. However, Alan Johnston pointed out in his recent book [Johnston03] that

> *"The biggest driver for SIP on the Internet has less to do with SIP's signaling and call control capabilities. Instead, it is due to the extensions of SIP that turn it into a powerful "rendezvous" protocol that leverages mobility and presence to allow users to communicate using different devices, modes, and services anywhere they are connected to the Internet."*

From Johnston's point of view, SIP enables and improves the ubiquitous computing, such as Instant Messaging, Internet telephone calls, multimedia conferences, etc, to a big extent. Taking this advantage of SIP, the major effort of this research is to incorporate communication session (SIP) into coalition access control, called Session-based Coalition Access Control Architecture (SCACA) (further descriptions in chapter 5), to dynamically control access in spontaneous coalitions, which can be considered as large scale ubiquitous environments. The detailed reasons of this incorporation are explained in Chapter 5. This section gives a brief overview of SIP describing all important aspects of the Session Initiation Protocol.

2.2.1 The History and Future of SIP

The history of SIP is in part the history of voice over IP technology. Before the emergence of SIP, in 1995, vendors began developing software that allowed users to talk to each other over the Internet. The first project "Bamba Phone" [WKS98] in IBM incorporated video in an IP to IP software package. Its core was the H.323 protocol and similar products were MediaRing Talk, Free Phone, Message ASAP, etc. Their quality was not very good although they all worked, and the growth of voice over IP market had been very slow for quite a few years.

In 1996, Dr. Henning Schulzrinne, later an associate professor of the Department of Computer Science at Columbia University, and his research team submitted a draft of the Simple Conference Invitation Protocol (SCIP) [Schulzrinne96A], the initial version of SIP, to IETF. Until 1998, Schulzrinne and his team were aim to define a standard for Multi-party Multimedia Session Protocol [Schulzrinne96B]. In 1999, Schulzrinne made a change by removing inessential components regarding media content, and then the IETF issued the first SIP specification, RFC2543 [RFC2543], which took shape as the SIP with which people are now familiar.

From 1995 to 2001, Schulzrinne's work did not attract enough interest from the academics and industry sectors. One of the reasons was that vendors were concerned that protocols such as H.323 and MGCP (Multimedia Gateway Control Protocol) could jeopardize their investments in SIP services. However, the IETF continued its work and issued the SIP specification RFC3261 [RFC3261] in 2001, which contains the core protocol specification.

The advent of RFC3261 and its later revisions was a milestone in the development of SIP. After that, vendors began to launch SIP-based services, and see the SIP and VOIP markets as the wave of the future. Today, software SIP endpoints are more common, Microsoft Windows Messenger uses SIP, and in 2003, Apple Computer released iChatAV, a new version of their AOL Instant Messenger compatible client that supports audio and video chat through SIP. There are many softswitch implementations, such as Nortel, Sonus, which can act as SIP proxy and registrar server. Other companies, led by Ubiquity Software and Dynamicsoft have implemented products based on the proposed SIP standards, building on the Java JAIN specification[5].

Most importantly, more and more players are entering the SIP marketplace with promising new services, and SIP is on path to become one of the most significant and unified protocols in both IP and telecom world.

2.2.2 SIP Entities

Although in the simplest configuration it is possible to use just two user agents that send SIP messages directly to each other, a typical SIP network will contain more than one type of SIP entities. There are four types of logical SIP entities in a SIP network, they are user agents, proxies, registrars, and redirect servers. Each entity can act as a client to initiate a request, as a server to respond a request, or as both. Note that several entities could physically reside on one device. They are briefly described as follows:

[5] JAIN technology is changing the telecommunications market from large proprietary closed systems to an open architecture of rapidly deployable services. JAIN is a community of over 80 member companies, specifying over 25 new APIs that target converged IP and PSTN networks. JAIN includes APIs for high-level application development (eg Service Provider APIs, SLEE - Service Logic Execution Environment),

User Agent

A User Agent (UA) is an endpoint entity. User Agents initiate and terminate sessions by exchanging requests and responses. User agents usually, but not necessarily, reside on a user's computer in form of an application [RFC2543], which contains both a User Agent Client and User Agent Server, this is currently the most widely used approach, but user agents can be also cellular phones, PSTN gateways, PDAs, automated IVR systems and so on.

User Agent Client (UAC) – a client application that initiates SIP requests.

User Agent Server (UAS) – a server application that receives SIP requests and returns responses on behalf of user.

Proxy Server

A Proxy Server is an optional intermediary entity that acts both a server and a client for the purpose of making requests on behalf of other clients. Proxy servers are very important entities in the SIP infrastructure. The most important task of a proxy server is to route session invitations "closer" to callee. The session invitation will usually traverse a set of proxies until it finds one which knows the actual location of the callee.

There are two basic types of SIP proxy servers: stateless and stateful.

Registrar Server

A Registrar Server is a special SIP entity that receives registrations from users, extracts information about their current location (IP address, port number and username) and stores the information into location database. The location database is then used by SIP proxy server to retrieve user's contact information. Because of their tight coupling with proxy servers, registrar

---

Call Control, as well as protocol level APIs for signaling (SIP, MGCP, SS7, etc.). There are many JAIN products being developed in the industry. Indeed the JAIN APIs have been adopted by most softswitch and 3GPP OSA (Parlay) vendors, and are being mandated by carriers in their softswitch and OSA RFPs.

servers are usually co-located with proxy servers.

Redirect Server A Redirect Server accepts a SIP request and sends back a reply containing a list of the current location of the called party. This process includes a look up in the location database for the intended recipient of the request. Upon receiving a reply from Redirect Server, then the requestor extracts the list of destinations and sends another SIP request directly to them.

2.2.3 Entity Interaction

An interaction is a sequence of SIP messages exchanged between SIP network entities. It consists of one request and all responses to that request. Basically, the interaction between SIP entities can happen in different scenarios. In some situation, two User Agents can communicate with each other directly. One acts as User Agent Client (UAC), and the other acts as User Agent Server (UAS). More common scenario is that a session is set up between two User Agents with the assistance of an intermediate SIP Proxy Server, and usually Redirect Server and Location Database are parts of the interaction as well, especially when the session is across domains.

Because the call proxying scenario is more related to the thesis, one proxy call example is illustrated below.

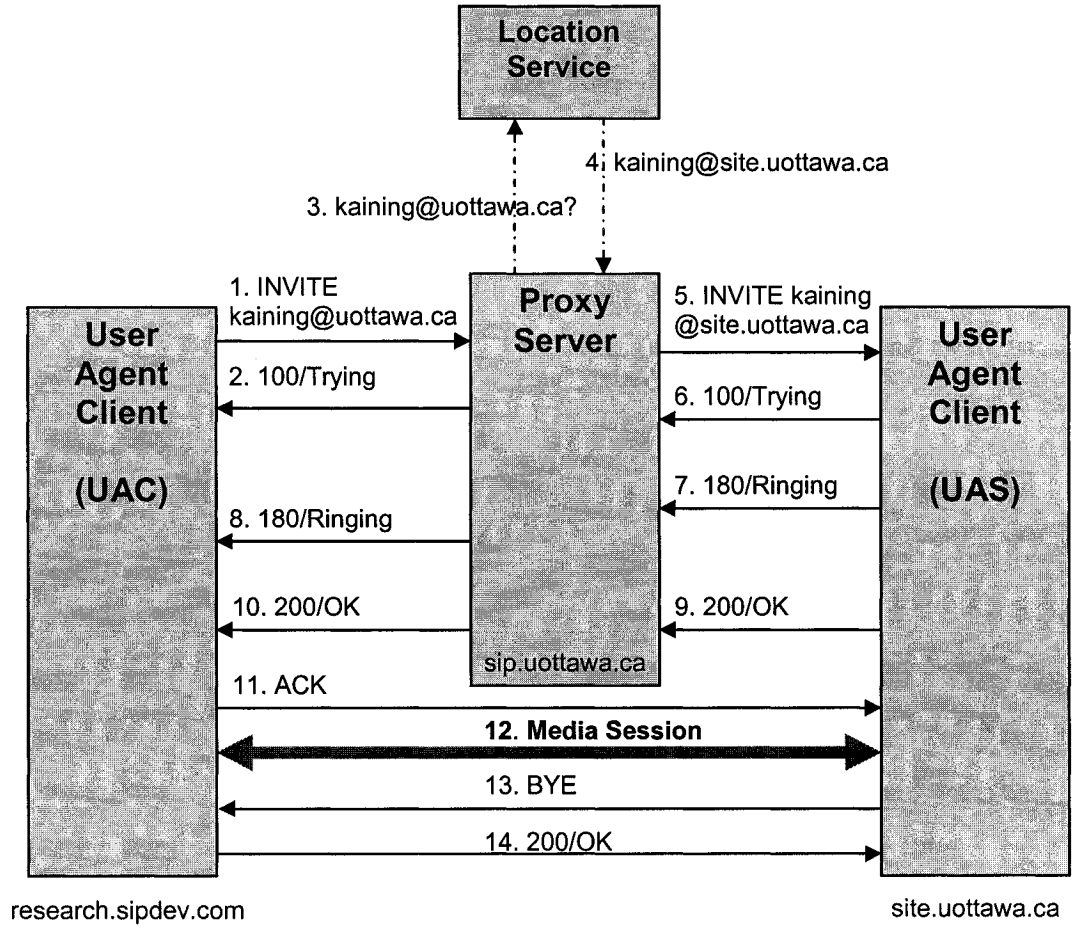

Figure 4 SIP Call Example with Proxy Server

In figure 4, a user at research.sipdev.com initiates a session by sending out an INVITE message to the callee kaining at university of Ottawa. The SIP address is described as an email like address, kaining@uottawa.ca. In order to make this example easily understood, only one proxy server is involved here. This INVITE message goes to the SIP proxy server in the callee's domain, i.e. sip.uottawa.ca. The Proxy Server firstly sends a 100/trying message back to the sender indicating that the INVITE message is in processing, and it may also query the location database to get the callee's current location and then sends the revised SIP message to the callee. Callee accepts the invitation by replying a 200/OK message. Again, this response is relayed by the SIP proxy server.

When the caller receives the response, it sends an ACK to the callee directly because it knows the callee's current address at the time. The session set-up process is finished. These two endpoints can start exchanging their media data, which is defined and negotiated in the previous SIP messages. When one endpoint wants to end the session, it just need to initiate an BYE message to terminate the session, and the receiver will acknowledge the cancellation by replying an ACK message.

2.2.4 Related Protocols and Session Description Protocol (SDP)

SIP is not the only protocol that the communicating devices will need. It is not meant to be a general purpose protocol. Purpose of SIP is just to make the communication possible. The communication itself must be achieved by another means (and possibly another protocol). This is actually one advantage of SIP: it is open to any existing or future underlying media transfer protocols.

Two protocols that are most often used along with SIP are RTP and SDP. RTP (Real-time Transport Protocol) is used to carry the real-time multimedia data (including audio, video, and text), the protocol makes it possible to encode and split the data into packets and transport such packets over the Internet.

Another important protocol is Session Description Protocol (SDP) [RFC2327], which is used to describe and encode capabilities of session participants. Such a description is then used to negotiate the characteristics of the session so that all the devices can participate (that includes, for example, negotiation of codecs used to encode media so all the participants will be able to decode it, negotiation of transport protocol used and so on).

SDP contains the following information about the media session:

- IP Address (IP address or host name);
- Port number (for UDP or TCP);
- Media Type (audio, video, interactive whiteboard, etc);
- Media encoding scheme (MPEG video, H.261 video);

In addition, SDP packets include the following session information:

- Session name and purpose;
- Start and stop times;
- Contact information about the session;

2.3 Collaborative Applications

In order to assess the effectiveness of existing coalition access control approaches, it is necessary to define three types of collaboration applications and three situational scenarios that capture most typical ad hoc collaboration situations. These scenarios are presented in section 2.3.2. Then some existing coalition access control approaches are reviewed in Chapter 3 and categorized into the following three collaboration applications.

2.3.1 Collaborative Application Types (CA Types)

Recent research efforts have been motivated by the problem of controlling access to resources in collaborative environments. In the literature the author surveyed there is no standard definition for collaborative application (CA) types, but in order to determine their usefulness to spontaneous interaction environments it is useful to define three types of CAs to categorize existing different access control approaches. The CA types are based primarily on the relationship among the participants and whether they share common access policies. These CA types are:

CA Type 1: This is the basic collaboration application type where all the individuals are considered part of the same organization. Therefore they fall under the same access administrative domain. These collaborative situations are really not coalition type scenarios and are listed here primarily for completeness. There is one situational context where coalition access control will be required. This is when the collaboration occurs at a 3rd party off-site location (See S Type 3 for details in the next section).

CA Type 2: In this collaboration type, individuals are associated with different organizations. Each organization has its own security administration but there is a common across-organization level agreement. This across-organization scenario must then share common policies and roles. Another similar situation is that collaborators are in different communities within an organization. Each community has its own

security administration model, but supports a common organization-level agreement.

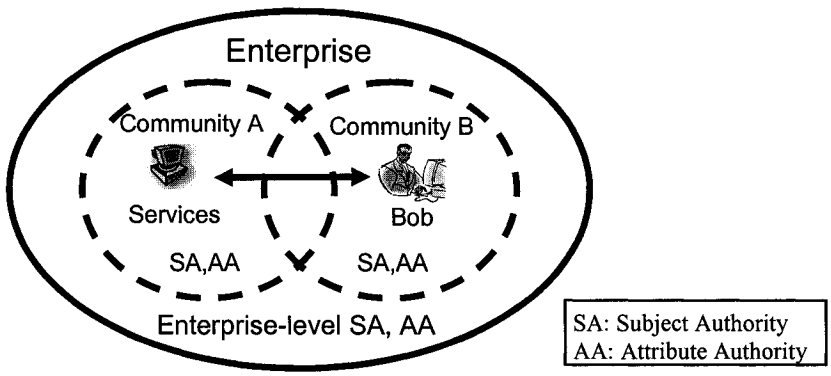

Figure 5 A CA Type 2 Example

CA Type 3: In this collaboration type, individuals are associated with different organizations as in the case of CA Type 2 but there are minimal or no organization-level agreements for access control. In other words they do not share common policy language or role ontology. These organizations are also unwilling to rely on a third party to administer trust relationships and access control between them. This collaboration type can ultimately support ad hoc interactions since it is not necessary to define an across-organization level agreement before hand.

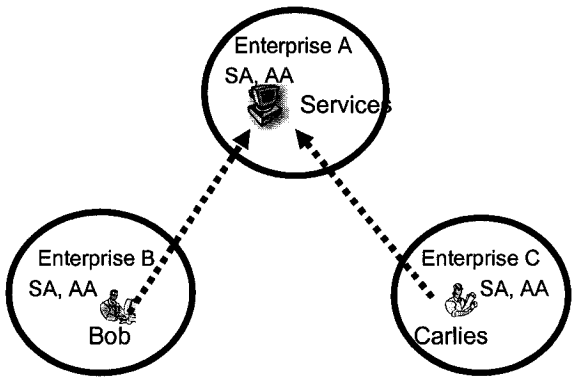

Figure 6 A CA Type 3 Example

2.3.2 Situational Types (S Types)

These particular situational type scenarios represent the situational context in which most collaborative sessions might occur. Three types of situational scenarios are presented as follows:

S Type 1: This is the in-situ case. The collaboration session takes place at a location associated with one of the participants. The implication here is that at least one person will be recognized as a member of the organization that owns the resources.

S Type 2: Multi location sites. The collaboration takes place across organizations where there are persons associated with the owners of the resources at each location.

S Type 3: Foreign remote site. The collaboration takes place at a location not associated with any of the participants. This is a special case of S Type 1 where the ownership of the resources is by a 3rd party not associated with any of the participants.

These three S types actually represent different business models to some extent. Let us take S Type 3 as an example. A telecom company can act as a third party to provide various services for facilitating transactions and interactions between partners or between end users and service providers. It is possible to combine the S Type with the CA Types creating a 3x3 table of possible scenarios that are shown below. Certain scenarios are not coalition type of scenarios and are marked with an NA for Not Applicable.

Table 1 CAS Possible Scenarios

	S Type 1 (same domain)	**S Type 2** (multi-domain)	**S Type 3** (3rd party)
CA Type 1	NA	NA	CAS 13
CA Type 2	CAS 21	CAS 22	CAS 23
CA Type 3	CAS 31	CAS 32	CAS 33

All of the S Type 3 situational scenarios involve a 3rd party entity. These types of situations will not be considered in detail. The author is more interested in the situational scenarios defined by CAS 21, CAS 22, CAS 31, and CAS 32. Below is a short description of these collaborative situational scenarios.

CAS Type 21: The participants belong to different organizations and are in one location and share a common set of policies in order to access local resources.

CAS Type 22: This is the same as CAS Type 21 except the participants are located in multiple locations. They are required to access resources from each others' organizations and share a common set of policies.

CAS Type 31: The participants belong to different organizations and are in one location but do not share a common set of policies. Delegation of authority with minimal administrative effort is required in order to share resources among the participants.

CAS Type 32: This is the same as CAS Type 31 except the participants are located in multiple locations. They are required to access resources from each other's organizations and need not share a common set of policies. Again delegation of authority with minimal administrative effort is required in order to share resources among the participants.

This research is targeting technology to address scenarios CAS Type 31 to CAS Type 32 since these scenarios are the ones with that allow users to delegate authority, and therefore reduce the need of administrative configuration. Moreover, extending this work to CAS Type 33 could be one of the future work. This classification of interaction types help better understand the access control models and mechanisms that are discussed in the following chapters.

Chapter 3 Related Work

3.1 Fundamental Access Control Models

In security, the term "access control" (AC) refers to the practice of restricting access to a physical location or facility to authorized persons, and preventing access by unauthorized persons (i.e. human guards, doors, looks). Digital schemes are used as access control mechanisms in computer security. By definition, in computer security, access control is the security service that guards protected resources against unauthorized access [Adams04]. The access control protection is policy-driven and the protected resources are those defined in access control policies.

Comparing with other security mechanisms, like cryptography, access control mechanism is the best technology applicable in network security area [Adams04]. The simplest and least effective access control mechanism is Access Control Matrix (ACM) [GD72] [HRU76], which provides a system-centric, subject-object view of access control. An ACM can get impractically large and sparse when there are many users in the system because practically most subjects do not have access rights to most objects. Access Control List (ACL) provides a straightforward means of determining the appropriate access rights to a given object. It is a data structure, usually a table, containing individual users or group rights to specific system objects. Instead of storing columns with objects in ACL, Capability List stores rows with subjects (capabilities). Both mechanisms have limitations on the expressiveness and efficiency of access control policy [Adams04].

The predominant model for advanced access control is Role-Based Access Control (RBAC) introduced in 1992 by Ferraiolo and Kuhn [FK92], which is increasingly incorporated into the security features found in almost every operating system, database management system and various applications, and now an American National Standard – ANSI INCITS 359-2004 [ANSI04]. RBAC departed from the subject-object models presented above by introducing the organizational concept of "role" and abstracting a concept of a user's job function within an organization. Users can then be assigned to roles, allowing an administrator to manage permissions on a few organizational roles rather than directly on a multitude of users.

Nevertheless, RBAC has some limitations in its use in spontaneous coalition environments. Traditional RBAC systems depend upon administration by a single authority, which maintains the entire organization's security policy. This approach does not scale to the typical anonymous users that come together for a spontaneous meeting. Moreover, traditional RBAC cannot satisfy the dynamic nature of spontaneous coalitions.

3.2 Coalition Access Control Models

Very little research has been done with respect to secure collaborations between different domains. It appears that in general many existing approaches for coalition access control are all layered on top of RBAC due to its flexibility in allowing administrators to specify access based on roles rather than entities. The advantage of this is that the number of access policies are reduced to a manageable number of roles, and even though it is necessary to keep a user entity to role relationship, it is easier and more reliable to change

the roles of a user that to change all the access policies when users come and go from an organization.

According to the three types of collaborative applications (CAs) defined in section 2.3, TeaM-based Access Control (TMAC) [Thomas97] and its variation TMAC using Context (C-TMAC) [GMPT01] are the models that support CA Type 1 (same domain). They support the notion of a "team" as an abstraction that encapsulates a collection of individual users in various roles and a set of object instances required by the team to provide fine-grained and run-time access control for collaborative activity. TMAC also applies a formal model for meeting scenarios where it is necessary to define team roles. TMAC is a relatively restricted model for ad hoc interactions since it is known that knowledge transfer is better achieved through informal interactions [Allen84].

Ioannidis et al. [IBIKS03] contributed two notions related to access control for CA Type 2 (multi-domain) applications. They are the idea of using a unified policy language and an architectural model for the distribution of these policies. In 2002, Cohen et al. [CTWS02] examined the intrinsic characteristics of a dynamic coalition, which is very similar to a type 2 collaborative application (CA Type 2), and argued that specific access control models, policies and enforcement mechanisms are required in this kind of environment. They defined Coalition-Based Access Control (CBAC) model built on top of the domain models by adding conceptual elements designed to support a semantic representation for coalition access policies. However, CBAC needs an administrative model and has a limited contribution to the support of an ad hoc interaction model. Part of this work [CTWS02] may be useful for other approaches. For example, it is possible to incorporate some of CBAC notions into the unified policy language XACML

[XACML05]. Kapadia et al. [KACM00] proposed a model called I-RBAC2000, which can establish a flexible policy for dynamic role translation between two domains. In 2003, Park et al. [PH03] presented the concept of role ontology[6] to support RBAC among different communities within an organization by extending the concept of web services to the peer-to-peer level by developing a middleware platform. The above approaches presented either require high-level access control agreements [CTWS02] to be put in place or a common access policy ontology [PH03] between organizations. As such their administration overhead is relatively high. Moreover, scalability also brings complexity to the above approaches and becomes a significant issue for them.

Regarding research work for CA Type 3 application, in fact only one type of approach meets its requirements. These approaches are known as delegation access control mechanisms. The significant difference in these approaches over those considered as CA Type 2 collaborative applications is that they require a minimal common ontology or policy language.

Delegation is a mechanism where a user in a role delegates his role membership to another user or another role. As such the delegatee has the permissions of the delegator's role. The strength of this approach is that the organization controlling access to the resources in a domain does not have to change any of their access policies since the delegate takes on the persona of the delegator. Abadi et al [LABW92] [ABLP93] defined delegation as a mechanism for authentication, access control and trust. However no

[6] Ontology is a specification of a conceptualization. The purpose of designing an ontology is to enable knowledge sharing and reuse. Pragmatically, a common ontology defines the vocabulary with which queries and assertions are exchanged among agents. So most of the time, ontologism is discussed in the field of AI. The OWL Web Ontology Language is a recent development in the ontology research field,

constraints can be applied to the delegation. E. Barka and R. Sandhu model role to role delegation in [BS00], and a role based delegation model called RBDM0 with some extensions is introduced. They discuss the revocation using timeout, but the model does not support event-driven revocation. Finin [Finin03] discusses the when and why delegations should be used. In that approach conditions are attached to the delegation, but the conditions are limited and it does not support contextual information.

The author is particularly interested in approaches that use a decentralized delegation model to deal with the collaboration environment, in which multiple organizations are unwilling to rely on a 3rd party to administrate trust relationships. E. Freudenthal, et al. [FPPK02] firstly introduced delegation into the coalition access control by developing the distributed RBAC (dRBAC) system to support decentralized delegation chains maintained across multiple domains. The dRBAC model appears to offer a low overhead in coalition environments since it requires minimal involvement of IT administrative personnel and leverages the common RBAC access control model. In dRBAC every partner may use its own policy language and the only common knowledge among the participants is the exchange of roles. The dRBAC model can be improved by integrating contextual information for dynamic interaction scenarios. The contextual information introduces a finer grain of control over the access rights as well as a limiting period of time during which that access is in place. In order to better understand the contextual extensions, the fundamental components of dRBAC need to be presented.

---

which facilitates greater machine interpretability of web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with formal semantics.

3.2.1 The dRBAC Model

The dRBAC model [FPPK02] implements access control for a multi-party coalition application through delegation. The high level format of dRBAC's delegation is as follows:

Delegations: [Subject → Object] Issuer

Object == OEntity.OName

The Subject (role/entity) has the permissions of the Object (role). An Object (role) is defined as an Entity.OName which is the Object's name in the namespace associated with OEntity. This delegation is cryptographically signed by the Issuer. The dRBAC concept "Local Wallet" is used to store a collection of delegations as a repository. By doing so, dRBAC builds proofs requiring delegation discovery across multiple repositories. Proofs of the delegations are performed by traversing a graph of delegations that demonstrate that "principal P has the permissions of role R" [FPPK02] and are represented as P⇒ R. If the object role being delegated is not defined in the namespace of the Issuer, then the delegation is referred to as third-party. In this case, the Subject must be given the right-of-assignment to delegate the role Object and is shown below.

[Subject -> Object'] Issuer

The dRBAC infrastructure provides mechanisms for (1) publishing of delegations, (2) delegation discovery and validation to build proofs, and (3) continuous monitoring of credential validity. Proofs for delegations are performed in the local wallets. In order to handle the complex delegation problem in a distributed environment, dRBAC uses discovery tags to discover and authorize a delegation chain (a trust relationship which is

spread over multiple wallets). dRBAC also uses delegation subscriptions to monitor and propagate run-time delegation changes in all parties.

Let us take an example that is depicted in Figure 7. There is an ongoing tele-conference between two meeting rooms, roomA and roomB, located respectively in two different companies, CompanyA and CompanyB. Bob is a member of CompanyB as well as one of the participants attending the conference and is located in the meeting roomB. He wants to control and use roomA's resources during the meeting. In the following we summarize the steps illustrated by Figure 7,

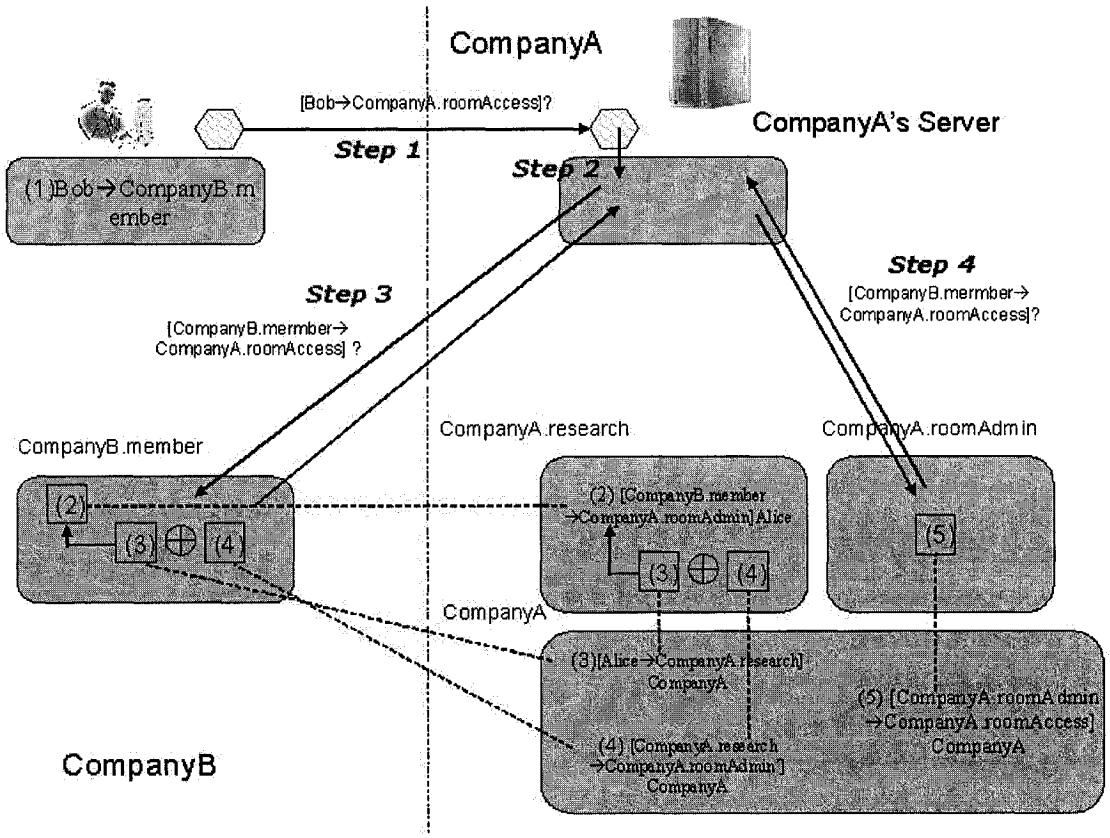

Figure 7 Example of dRBAC in action

(1) [Bob → CompanyB.member] CompanyB:

The starting point is to validate Bob's access to CompanyA.roomAccess. To authorize the access, the CompanyA server software must discover a proof for [Bob → CompanyA.roomAccess]?. In the beginning, CompanyA validates Bob as a CompanyB.member by the first delegation (1), and then CompanyA tries to validate [CompanyB.member → CompanyA.roomAccess], but it cannot discover a proof locally. Therefore, it contacts the home wallet corresponding to the role CompanyB.member and issues a subject query to obtain all proofs of the form [CompanyB.member → *]. In response to this query, it discovers delegation (2) defining a relationship between the roles CompanyB.member and CompanyA.roomAdmin.

(2) [CompanyB.member → CompanyA.roomAdmin] Alice:

It discovers delegation (2) is a third-party delegation since Alice does not directly have the right to delegate CompanyA.roomAdmin. This signifies that the delegation must have been accompanied by its support proof. In this case, the latter comprises of delegations (3) and (4).

(3) [Alice → CompanyA.research] CompanyA:

Alice is delegated role CompanyA.research.

(4) [CompanyA.research → CompanyA.roomAdmin']CompanyA:

CompanyA.research has the authority to delegate CompanyA.roomAdmin. At this point, the server wallet has a chain from Bob to CompanyA.member, but is still missing a proof that would authorize [CompanyA.roomAdmin → CompanyA.roomAccess]?. To obtain this, the wallet continues with its forward search by contacting the home wallet corresponding to the role CompanyA.roomAdmin, and issuing to it a direct query for [CompanyA.roomAdmin → CompanyA.roomAccess]?. The response to this query is a self-certified delegation (5).

(5) [CompanyA.roomAdmin → CompanyA.roomAccess] CompanyA:

Now the proof authorizing [Bob → CompanyA.roomAccess] is complete.

3.3 SIP-based Large Scale Ubiquitous Computing

Ubiquitous computing is not yet a reality for the general public, but it has provided much very interesting work over the past years. The majority of those work are in personal ubiquitous computing area, such as the Intelligent Room at MIT [Brooks97], the Composable ad hoc location-based services [HKSR97], the Aura Project at CMU [GSSS02], the Interactive Workspaces Project at Stanford University [BRTF02], and [FJHW00][WKJLJ01]. In recent years, some researchers [BSSW03] [RL04] explore the scenarios in large scale ubiquitous computing (See section 2.1.3) that is not limited to a single location or organization. They argue that ubiquitous computing systems, especially large scale ubiquitous computing systems, should be based on open standards, while most

of the existing approaches use proprietary systems and non-standard communication protocols, which prevents them from interoperating with others.

Both [BSSW03] and [RL04] utilize the IETF standard SIP as the underlying communication protocol to build architectures supporting dynamical communication in a distributed environment. The system presented in [BSSW03] only provides an authentication service to prohibit strangers' accesses, since current SIP standards support either password or certificate based authentication. In order to protect user's information, e.g. addressing information of user's agent, Rao and Li [RL04] introduce an Access Gateway (AG) and a random generated reference by user's agent to handle incoming access requests on behalf of the protected user agent. During this research there is no clue in literature that incorporates SIP into access control mechanisms for large scale ubiquitous computing systems.

Chapter 4 Extensions for Two Access Control Models

4.1 D-TMAC Model

In the previous chapter, the author reviews the CA Type 1 approaches TMAC [Thomas97] and its variation C-TMAC [GMPT01], which provide active and run-time permission activation mechanisms for collaborators in a single organization. From the author's point of view, this fine-grained and run-time permission activation and "Team" role management can be extended and applied in a coalition environment across organizations, and it is feasible to group users from different organizations into one team or association. This section presents the Delegation based TMAC (D-TMAC) model, which is an extension to the TMAC/C-TMAC Models.

This is best described again by utilizing a scenario. Let us take for example a situation where many professionals and researchers from different organizations and universities are attending a technical conference at location ABC. At this conference there are several standard meetings and events. Professor Thomas plays two active roles at this conference. He is a participant for the conference and also a chair for a standard meeting. Both conference and standard meeting are different events occurring at different times, and may require different resources. From the point view of the TMAC model, for the administrator of location ABC, it is convenient to define two teams (conference team and standard meeting team) and their associated roles, including user roles and object roles,

for the two events. However, the TMAC/C-TMAC models only deal with dynamic collaborations with participants all from one single domain. The challenge in making the TMAC/C-TMAC models suitable for across organization environments is that the user context of a "Team" is defined as the specific users that comprise a team at any given moment, that is, team members are specific users from one domain. Moreover, supposing there is a research group at a university, the conference organizers send an invitation to the whole research group instead of Professor Thomas, as such any researcher associated with the group can attend the conference. An efficient solution for a scenario like this would require that access rights be delegated to an external particular role (research group role) rather than to an entity (Professor Thomas).

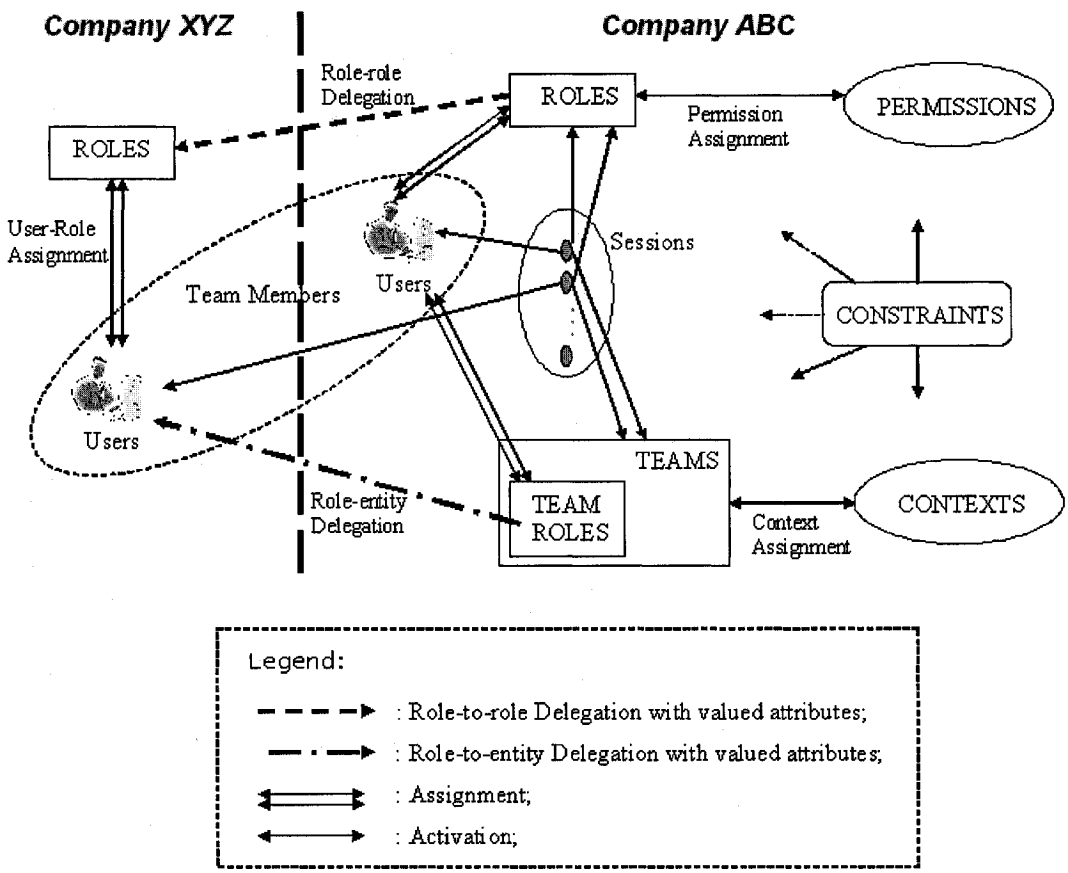

Figure 8. Delegation-based TMAC Association

The author proposes the Delegation based TMAC model (D-TMAC) as an extension of the TMAC/C-TMAC model to operate across organizations by using delegation. Since the members in a team can hold organization and local team roles, delegation can be used to map users and their roles in other organizations to the roles and team roles in local organization. This process includes two steps that are represented in the above figure as dashed lines.

1. Use role-to-role delegations to map roles in one organization to another;

2. Use a team role-to-entity delegation, so that external specific users are able to hold certain team roles.

When an external user of a particular organization wants to access to the resources of the organization, the user needs to select a local role automatically or manually from the set of roles assigned to him/her. According to this selection, a particular set of delegations and their related set of role-based permissions for this organization are granted based on the context. These are called session-role permissions in the C-TMAC model. After this, the user has to select a subset of teams to participate in, which are available to him. After the team selection procedure is completed, the permission set of the user is combined with the permission set available to the team. At this time, all the resources within the team are available to this user.

Regarding the above scenario, with role-to-entity delegation, external user Professor Thomas can be assigned a "member" role in the conference team, and a "chair" role in the standard meeting team at location ABC, and for the researchers in research team role

at that university, role-to-role delegation shall be used to facilitate their accesses to the resources at location ABC. This allows the administrator of location ABC to properly and easily manage dynamic and run-time resource accesses from external users. The same concept can be further applied to the scenarios previously presented by considering the multi-party across organization teleconference meeting between Alice and Bob. With the use of teams, Alice and Bob can have different level of privileges accessing resources.

This extended active security model provides very tight, just-in-time permissions so that only the appropriate team members, no matter which organizations they belong to, get accesses to specific resources and only at the time they provide their services without adding any significant administrative overhead.

From the author's point of view, the D-TMAC model provides dynamic access control for formal coalition environments because team management need administration involved to some extent. In section 7.1, the author has some discussions of combining this extension model into the later proposed SIP based SCACA framework to build a more complex, fine-grained coalition access control system for relatively formal coalition environments.

4.2 Adding a contextual model to dRBAC

In this section, the author presents an approach to add contextual information to a delegation model named the distributed Role Based Access Control (dRBAC) model that was developed by E. Freudenthal, et al. [FPPK02] to support dynamic and finer grain coalition access control.

As discussed in section 3.2, the author is particularly interested in approaches that use a decentralized delegation model to deal with the access control in spontaneous coalition environment, in which multiple organizations are unwilling to rely on a 3rd party to administrate trust relationships. The dRBAC model appears to offer a low overhead in coalition environments since it requires minimal involvement of IT administrative personnel and leverages the common RBAC access control model. In dRBAC every partner may use its own policy language and the only common knowledge among the participants is the exchange of roles. The dRBAC model can be improved by integrating contextual information and introducing a delegation security manager for ad hoc interaction scenarios. The contextual information introduces a finer grain of control over the access rights as well as a limiting period of time during which that access is in place.

In order to minimize the need for a common contextual ontology among coalition organizations, the contextual constraints is restricted to be applied only to the organization that owns the resource. This is a similar approach to that taken by dRBAC [FPPK02] in its use of constraint attributes. Even though it is possible for each organization to use its own contextual model, a contextual model is presented in the thesis as a base mark.

There is much work in representing semantics for context-aware computing. Leveraging the work in [LHL03] [CFJ03], in our ontology, context information is classified into entity and activity. Entities are person, location, time, and objects (resources), and activities are normally the behaviors of a group of entities. Currently the thesis simply uses the Object-Oriented methodology of classes, subclasses and instance notions to clearly represent hierarchical context information that will help us capture location and

activity in such a manner that people can specify a general location or activity and any sub-ordinates that will apply for the delegation proof. Context hierarchy indicates relations between atomic context elements.

Class Hierarchy		Attributes
Location	Location Organization Building Floor Office MeetingRoom Cafeteria Restroom Stairway	Name Latitude Longitude ContainedBy
Activity	Activity Presentation Listening WorkOut Entertainment Eating CommunicationSession PhoneSession	Name StartTime EndTime Status Participant

Table 2. An example from context ontology

An advantage of using context ontology is the ability of reasoning, for example, automatically deriving a certain context value in a given context hierarchy.

To support context sensitive access control, the context condition, which will be applied to the delegator, may be generally defined as follows:

(Activity == PhoneSession && Location == MeetingRoom)

where the value of activity and location are general classes. The ontology for PhoneSession and MeetingRoom is such that the run time instantiation of

Activity == PhoneSession.SessionID1234 and Location == MeetingRoom.SITE4009

will match the class types as TRUE.

Then context information is added as attached conditions for a delegation as an extension for dRBAC. The format is proposed as follows:

[Subject -> Object] (ContextCondition) Issuer*

ContextCondition = (Context OP value).

The contextual information is interpreted to mean that the delegation [Subject -> Object] is valid under the Issuer's domain that matches the ContextCondition* which could be a combination of multiple contextual conditions. The contextual conditions can also be applied on a right-of-assignment delegation [Subject -> Object'] Issuer.

[Subject -> Object'] (ContextCondition) Issuer*

ContextCondition = (Context OP value).

Since the contextual conditions only apply to the issuer's domain there is no need to share a common ontology among users.

Context conditions are following the delegation, and several context conditions can be combined together or applied separately. The issuer's signature ends this delegation. Context conditions are applied to the Issuer. Let us take an example of a member of CompanyB being delegated a role at CompanyA.

[CompanyB.member -> CompanyA.roomAdmin]

Alice (Activity == Communication_Session);

In this particular example, Alice, an employee at CompanyA, is the issuer of this delegation. For the delegation to be true Alice must be in the activity of a Communication_Session. When the delegation is retrieved as a proof for an access

request, the system will check the context conditions in that delegation before approving it.

Chapter 5 Session-based Coalition Access Control Architecture

In the previous chapter, the author has discussed the delegation-based TMAC (D-TMAC) model, the Context-based Coalition Access Control Model (an extension to dRBAC model). Here the author proposes a Session-based Coalition Access Control Architecture (SCACA) for building and rapid prototyping of dynamic coalition access control management and services for inter-organizational spontaneous coalitions, hence improving the functionalities and content of real-time and spontaneous cooperation between end users by securely maximizing available services on each end of participating parties.

5.1 Incorporating SIP Session into Spontaneous Coalition Access Control

As discussed in section 1.1, due to the dynamic and spontaneous nature of spontaneous coalitions, it is difficult to satisfy its access control requirements using existing access control mechanisms. It is a real challenge for an organization or user to trust and give access rights out to external users within an inter-organizational spontaneous coalition, and revoke them later in a very easy manner.

In order to solve the above problems and provide a dynamic access control mechanism for spontaneous coalitions, the author proposes a Session-based Coalition Access Control

Architecture (SCACA), which innovatively introduces the SIP session into coalition access control systems. The reasons of this incorporation are as follows.

Firstly, as concluded in section 2.2, most inter-organizational spontaneous coalitions are over a communication session, and the communication session is the basis of its higher-level spontaneous coalition application.

Secondly, besides authentication, the communication session can be leveraged as a bridge to include the trust required among users in their day-to-day activities. People have the experience that making a phone call or starting a communication means that you know the party on the other end to some extent, which is the trust in the real world. For example, people can recognize each other with the help of real time audio and video. Specifically in a communication session, the session initiator has to know the end points a-prior in order to initiate a communication session and hence start a coalition. The knowledge of the other end points helps build a mutual trust. As such, the underlying communication session can be regarded as a bridge to transfer trust to access control systems from the real world. The reason for making this point is to eliminate some unnecessary security concerns and ultimately simplify the whole process.

Thirdly, an underlying communication session fully contains and reflects the dynamic and spontaneous nature of a spontaneous coalition. A spontaneous coalition starts when its underlying communication session is set up, and ends when the communication session ends. For example, conference calls. As such, a communication session can be regarded as a kind of activity context, like other context information, e.g. locations, to help build the context-based coalition access control model. The benefit is that the

proposed SCACA architecture is able to satisfy the dynamic nature of the access control requirements for spontaneous coalitions.

Fourthly, to recap, a spontaneous coalition environment, as a kind of large scale ubiquitous computing system, requires the underlying communication protocol to be open and secure (See section 3.5). As a communication session protocol, SIP is being standardized and governed by the IETF. Moreover, its standards are supported and developed by telecommunication communities, such as 3GPP[7] and 3GPP2[8]. SIP has gained wide acceptance and is becoming a unified communication standard in the next generation network (See section 2.2). SIP is the best candidate as a session management protocol to provide scalability for large scale ubiquitous computing systems and the proposed SCACA architecture as well.

Moreover, the SIP architecture helps parties automatically discover security managers (See section 5.3) that exchange roles and delegations, and directory agents (See section 5.3) that store service description information. Knowledge of the addresses of wallet keepers (a delegation repository in dRBAC) at each end point can also be exchanged at the time that participants enter into a communication session.

The last but not the least reason is that, extensible and flexible, SIP operates independent of its underlying network transport protocols and is indifferent to media, no matter

[7] The 3rd Generation Partnership Project (3GPP) is a collaboration agreement that was established in December 1998. The collaboration agreement brings together a number of telecommunications standards bodies which are known as "Organizational Partners". The current Organizational Partners are ARIB, CCSA, ETSI, ATIS, TTA, and TTC.

[8] The Third Generation Partnership Project 2 (3GPP2) is: a collaborative third generation (3G) telecommunications specifications-setting project; comprising North American and Asian interests developing global specifications for ANSI/TIA/EIA-41 Cellular Radio telecommunication Intersystem

whether the content is voice, video, or any other type of data. This allows large scale ubiquitous computing systems to easily adapt to different environments and create new services for users at the lowest cost.

5.2 Overview of SCACA

The proposed Session-based Coalition Access Control Architecture (SCACA) is designed to provide security and access control in distributed systems. It is centered on the distributed deployment of middleware services over the network across organizations to support access control in spontaneous coalitions. The main feature of SCACA is that it supports session-oriented access control, dynamic delegation and context-aware computing.

5.3 SCACA Components and Functionalities

There are seven functional components within the SCACA framework: Directory Agent, Security Manager, Context Agent, Policy Server, SIP Proxy Server, User Agent and Resources. Figure 9 shows the components in the SCACA architecture, and they are described in detail as follows:

---

perations network evolution to 3G; and global specifications for the radio transmission technologies (RTTs) supported by ANSI/TIA/EIA-41.

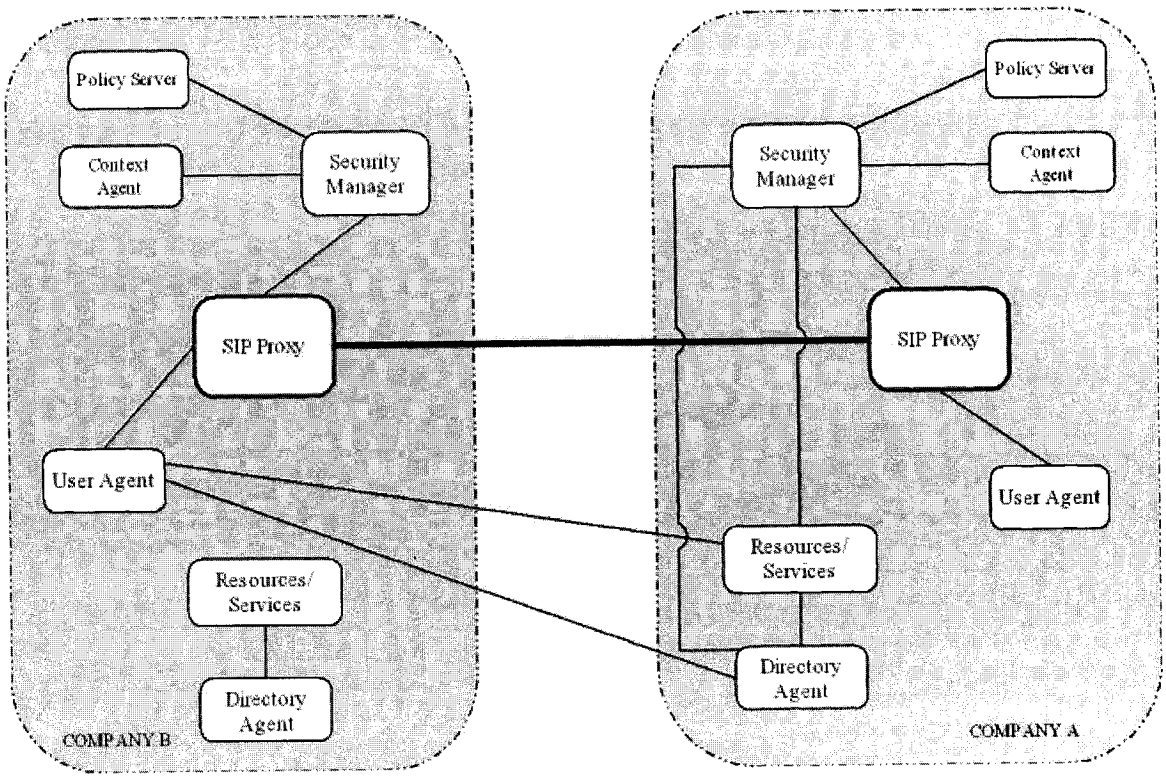

Figure 9 SCACA Framework

5.3.1 Directory Agent

A Directory Agent in SCACA collects service advertisements including service location and attribute information, and acts as a service directory. In detail, the directory agent in SCACA is responsible for the following tasks:

1. **Service registration**. Every local service or its service agent has to register, update and deregister its service description information on a directory agent, which resides in the same domain. The directory agent stores service description information and their future updates.
2. **Remote user subscription**. Upon joining a communication session, a remote user exchanges its directory agent's addressing information with other endpoints, and

subscribes him/her on directory agents in remote domains for available services. A remote user's subscription expires when that user leaves the session or the whole session is over [Hu03].

3. **Secure Service discovery**. A Directory agent provides a filtered service view for service subscribers, the participating members of a communication session, so a remote user knows what services are available for him/her to access in a remote foreign domain. The difference with traditional directory agent is that the provided available service lists for a remote user are only those services that the user can get access to. The directory agent needs negotiate with its security manager to get secure service names and attributes, retrieve the service contact information stored in the directory agent itself, and then send back to the remote subscriber user. In general, there are always two ways for this service information provision: active strategy and on demand strategy. Active strategy forces the directory agent to push available service list to remote users (UA) whenever service information gets changed. On the contrary, in on demand strategy, a remote user retrieves the service list only when he/she needs to use services in this domain. The former always provides latest service information, but the latter saves resource consumption, especially when the number of parties is large.

Digital certificates can be used to authenticate directory agents, and outgoing messages like service lists can be signed by the private key of the directory agent, so user agents in other domains will be able to authenticate the directory agent and hence trust the received available remote service information.

The directory agent in the Centaurus project [KKAJFY02] acts as an active proxy between services and users by executing services on behalf of any client that requests them. Centaurus project was designed for personal ubiquitous environment, in which clients are mostly mobile devices with limited capability. Since target clients in our system that is a large scale ubiquitous computing environment are versatile, there is no strong need to consider and minimize the resource consumption on the client side. The directory agent in SCACA is not required to be a proxy for service execution on behalf of clients, and the directory agent is only in charge of service discovery for clients. This design will simplify the system architecture and make the system more open, expandable and interoperable.

5.3.2 Security Manager

Not only acting as a PDP (Policy Decision Point) in access control system, the Security Manager is responsible for maintaining security and trust in SCACA as follows:

1. **Session role and its delegation management**. After a communication session gets set up, a session role will be automatically created in the security manager and necessary delegations for this session get generated or activated as well. Whenever the delegator leaves the session or the constraints of the delegation become false, the session role delegation will be revoked or deactivated. Furthermore, a user can manually revoke its delegations by sending proper messages to the Security Manager anytime, and then the Security Manager removes related delegations in its knowledge base.

2. **Session event processing**. When related SIP session events arrive at the SIP Proxy server, the security agent embedded on the Proxy server will communicate with and notify security manager what session events happen and related session information. These event notifications will automatically trigger the above session role and delegation management.
3. **PDP (Policy Decision Point)**. Upon receiving an access request forwarded by resources, the Security Manager starts delegation proofing process and retrieves related access control policies from the Policy Server when necessary. Sometimes delegator's context information is pulled from the Context Agent;
4. **Authentication.** In new key-oriented, distributed access control systems (SPKI/SDSI), entities are represented by their cryptographic signature keys. Whenever necessary, the Security Manager can always authenticate entities based on the digital certificate transferred within a SIP request message;

5.3.3 Context Agent

A Context Agent is an important component of SCACA architecture. Dey [D00][DA00] proposed, location, identity, environment time and activity as basic context types for characterizing the situation of a particular context entity. Context information are mostly activity, location information in the author's consideration, and they are always key factors for across organization conference scenario described in Chapter one. Other context information, like time, can also be used in SCACA. Different organizations have different context ontology, so the Context Agents are domain-specific and are always pre-defined for each domain in SCACA system. An ideal context agent has the following four functionalities:

1. **Context Acquisition**. To be able to use context in an application, there must be a mechanism to sense the context and deliver it to the application. There are two strategies for acquiring low-level context information: active strategy and on demand strategy;
2. **Context Modeling and Reasoning**; Context information are various and there is a lot of research on the modeling method. Based on a context knowledge repository, an ideal context agent has a ontology context reasoning engine that deduce facts that can be concluded from the knowledge stored in the context knowledge repository.
3. **Context Condition Computation** (condition broker). The context agent can store and update context conditions appearing in context based delegations or AC policies, and it accepts security manager's request for computing context conditions, and replies back the boolean values of those context conditions.
4. **Privacy Protection.** In a context-aware system, a context agent collects and shares user information. This raises great concern for user privacy that personal information is collected without the explicit consent of the users. An ideal system must consider the privacy of context sources as well as that of the context subjects. The system should not release the identities of context sources without permissions. Furthermore, to some extent, the system should allow subjects have control over which context sources can supply data about them.

5.3.4 Policy Server

A Policy Server acts as an access control policy repository in SCACA. According to an access request, a security manager can retrieve related access control policies by querying

the policy server. Policy server also provides a management interface for administrator and policy owners to create and manage policies.

5.3.5 Proxy Server

A Proxy Server is a standard SIP component as introduced in section 2.2.2 plus session-based coalition access control functionality. The Proxy Server plays a key role in the SCACA system. A security agent is embedded into proxy to enable the standard SIP proxy server to support SIP session event notification for the Security Manager in the SCACA framework. The SIP events are SIP INVITE, INVITE RESPONSE, BYE and BYE response.

5.3.6 User Agent

In order to setup a SIP communication session, every end user has to have a User Agent (including UAC and UAS), which is a standard SIP component as introduced in section 2.2.2 plus session-based coalition access control functionality. The User Agent can create a SIP message with user's credential and security manager's service information embedded in it.

5.3.7 Resource

Resource is data, service or system component, to which access is requested. In most cases, security manager can act as PEP (policy enforcement point) to enforce access control decisions received from PDP. Sometimes resources themselves can be PEP as

well, for example, a web server in an organization can enforce access control decisions for all web access requests.

5.4 General Support Mechanisms

5.4.1 Acquiring the delegation security manager's addresses

To acquire the addresses of the delegation security managers the author proposes defining a new SIP Session Description Protocol (SDP) [RFC3264] for the Delegation Security Manager. A SIP SDP component is used to carry information of the end point capabilities for the establishment of a communication session (See section 2.2). It is generally used to negotiate real time streaming codec parameters. Recalling the discussions of SDP in section 2.2.4, the key arguments in the SDP message are the following:

v= Version

o= username session-id version network-type address-type address

s= Session name

c= network-type address-type connection-address

t= start-time stop-time

m= media port transport format-list

The critical arguments are s, c, and m. The session name (s) can be any commonly agreed to name, and the use of "Delegation Manager" is suggested. The network address (c) is negotiated between the 2 entities. Since the connection between the delegation security managers is intended to be bi-lateral and permanent (for the duration of the communication session) the address that is passed between the caller and the callee is

originally the IP address of the caller's delegation security manager. If the callee accepts the media it will return with the IP address of its delegation security manager.

Also of great importance is the media type argument (m). This will specify the port number and transport type. For the delegation security manager this would appear in the following manner: "m=application 8900 dRBAC". In most SIP examples the format-list is a known connection protocol. For example RTP is used for many of the real time streaming applications including voice and video. The author is not aware of any protocols that are used for the negotiation of delegation or security policies. This approach has been proposed and implemented by Liscano et al. [LJDH04][Hu03] for the negotiation of service directories. It is a relatively powerful mechanism by which some minimal exchange of capabilities can be associated with a session. Figure 10 is a message sequence chart of the SIP dRBAC session negotiation.

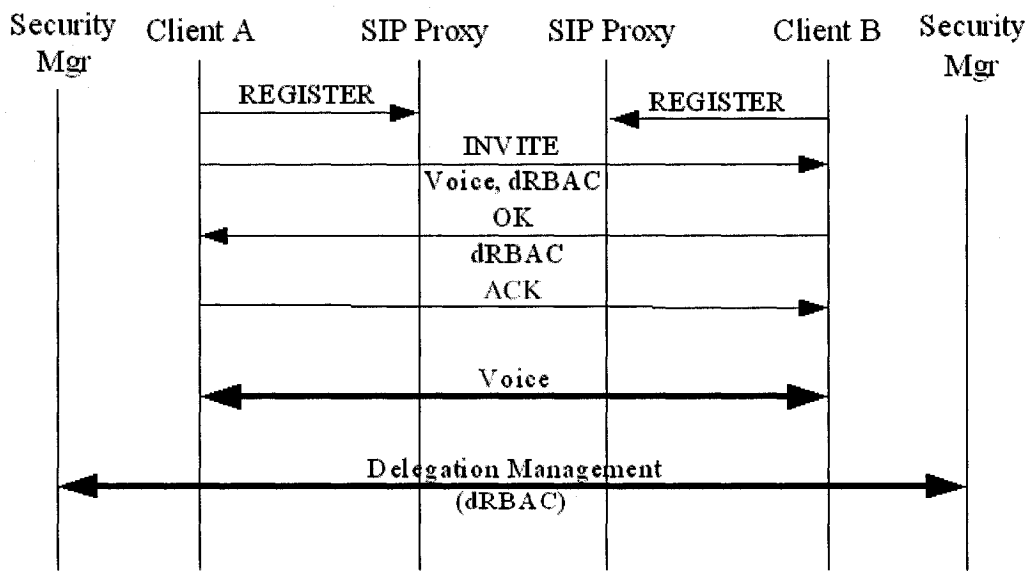

Figure 10 Sequence of a SIP dRBAC Session Negotiation

5.4.2 Session Role

At this point delegation security managers can establish a connection in order to negotiate roles and delegations whenever necessary. Even though the RBAC model [FK92] is widely used, it is primarily a tool for network administrators. Users are not aware of what particular roles they might be associated with since for most situations the correspondence between a user's identification and his/her role is performed automatically. It is more effective to keep the same interaction model since users have a difficult time choosing appropriate roles.

The author proposes the use of a session role (See figure 11), which is determined at run time and created when the communication session is established. This can easily be done by considering the CALL-ID argument in the SIP message as a role name, which can uniquely identify a SIP communication session. This session role can be shared by all session participants that belong to the same communication session no matter which organizations they belong to. Every user located at the calling end of the communication session can simply delegate his/her roles to this run time session role, so other session participants are able to access resources across the organizations.

There are three advantages of using a session role instead of delegating internal private roles directly to external individuals or roles as the dRBAC model normally does.

First of all, the session role eliminates the need to have the knowledge of role ontology in other organizations, which requires high level administration overhead. Moreover, it is more secure for the delegatee's domain to use other roles than those used internally since

this prevents the exposure of internal roles to external domains because the session role is more like a virtual role which does not belong to any organization.

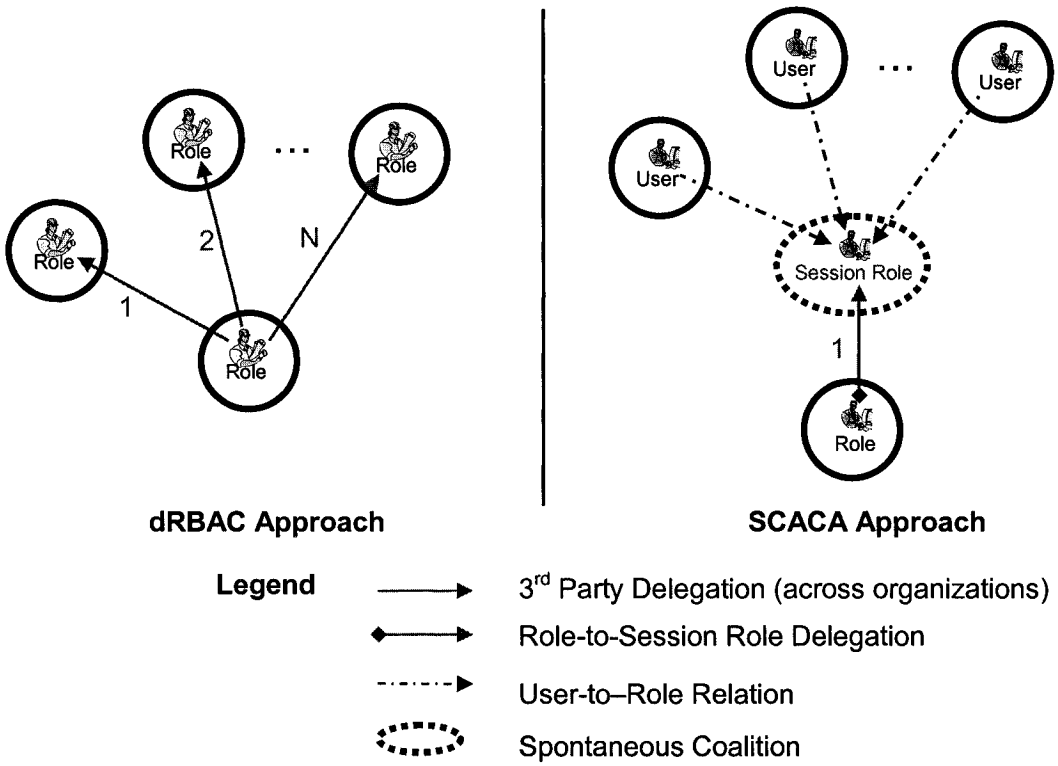

Figure 11 A Comparison of Direct Delegation and Session Role

Secondly using a session role can dramatically reduce the amount of delegations to which one must give access to from external organizations. Each delegator just delegates its roles to one session role instead of delegating its roles to users or roles from every external organization respectively. As shown in figure 11, in order to share its resources within a N-party spontaneous coalition, an endpoint (in a participating organization) only need to manage one role to a session role delegation, while the amount of delegations need to be managed in dRBAC approach is N-1. That means an endpoint always handle one simple and non-3rd party delegation, no matter how many organizations evolve in a spontaneous coalition. The complexity of the session role approach is O(n) while the

complexity of the traditional dRBAC model is $O(n^2)$. The session role approach greatly eases the administration overhead of delegation and makes a coalition access control system very scalable, hence this allows the large scale ubiquitous computing application be really large.

Thirdly it is easier to handle dynamic user behavior. Users are members of a communication's session role only when they are currently in that session. Once a user quits an ongoing session, he/she will no longer hold that session's role, so his or her access requests to the other party's resources will not be permitted any more. This is an added security measure since the delegations should have been revoked when the user left the session.

The use of a session role allows us to manage both single and multi-party collaborative scenarios. In a single party collaborative scenario a delegation can be directly targeted to an individual. In multi-party collaborative scenarios delegation needs to be targeted to a group of persons. In either situation the individuals will be delegated the session role. For example if Alice is the delegator and Bob is the delegate, Bob will be delegated the session role and Alice can delegate access to a room in CompanyA by delegating a roomAdmin role in the following manner.

[Alice@CompanyA -> CompanyA.sessionRole'] CompanyA

[CompanyA.sessionRole -> CompanyA.roomAdmin] Alice@CompanyA

[Bob@CompanyB -> CompanyA.sessionRole] Alice@CompanyA

5.5 Access Control Data Flow Model

The major access control actors in SCACA are shown in the data flow diagram of Figure 12. For detailed SIP session related support mechanisms, please refer to section 5.4 and 5.6.

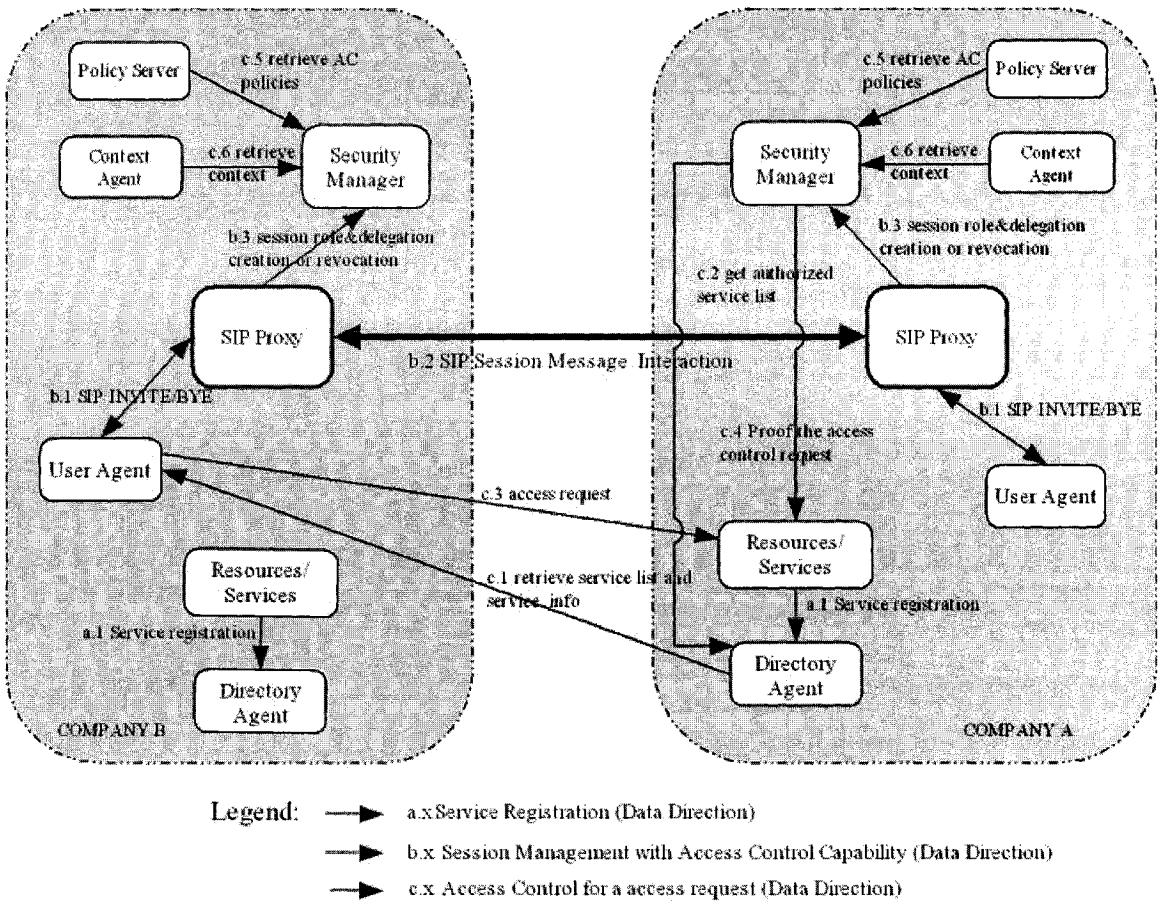

Figure 12 the SCACA Architecture Activity Diagram

Let us take the scenario described in section 1.2 to explain how SCACA works. To recap, a tele-conference will be held between employees from two different companies, CompanyA and CompanyB. Alice is an employee of CompanyA, and she is the initiator of this conference. Bob is a member of CompanyB as well as one of the participants

attending the conference. The purpose is to allow all the participants attending this tele-conference from other domains to get controlled access to resources related to this conference call. For example, the accesses to the resources of Alice in CompanyA are only permitted during the conference to those members participating in the conference call that is the communication session. As such once the conference ends, all the access privileges are gone.

The model operates by the following phases:

5.5.1 Service Registration

The inter-organizational spontaneous coalition access control brings a service discovery challenge, since knowing services or resources beforehand seems impossible and impractical in across organization environments. Suppose Alice in CompanyA gives access rights to remote user Bob in CompanyB in a spontaneous coalition access control system, then a problem arises, that is Bob has no idea about what he can access in Alice's domain, so a "Where" question will arise first: Where can I find a list of available services and their detailed service description information to me? As a result, an access control mechanism is not enough, and service discovery is an important and necessary component in inter-organizational spontaneous coalition access control system.

In the design, the author leverages the Service Location Protocol (SLP) [RFC2165], an IETF protocol for service advertisement and discovery, which provides the capabilities of discovering services automatically without human intervention. SLP allows clients to put a specific service query in a SLP message to discover services.

SLP consists of three entities: Directory Agent (DA) that caches the registered service information, Service Agent (SA) that advertises the service address and attributes information for the service provider, and User Agent (UA) that performs service discovery on behalf of the client.

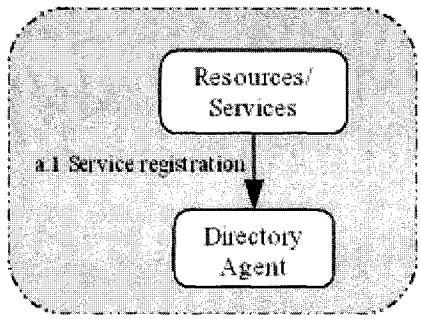

Figure 13 Service Registration Phase

In this service registration phase, every resource or service provider registers all its service description information on the directory agent, which acts as a DA (a.1 in Figure 12, 13). Then in the access control phase, the remote client will be able to retrieve service information from the directory agent.

5.5.2 Session Management with Access Control Capability

This phase includes the natural SIP session management phases.

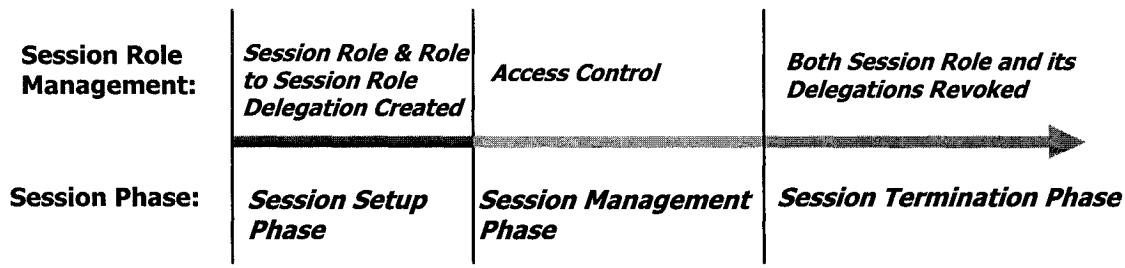

Figure 14 Lifecycle of Session Role and Delegations in SCACA

5.5.2.1 Setup of a Communication Session

Before permitting an access request from an external organization, a communication session needs to be setup up, and the address information of the security managers located in each participating organization needs to be exchanged. These security managers will then be able to communicate with each other to share delegations and proofs.

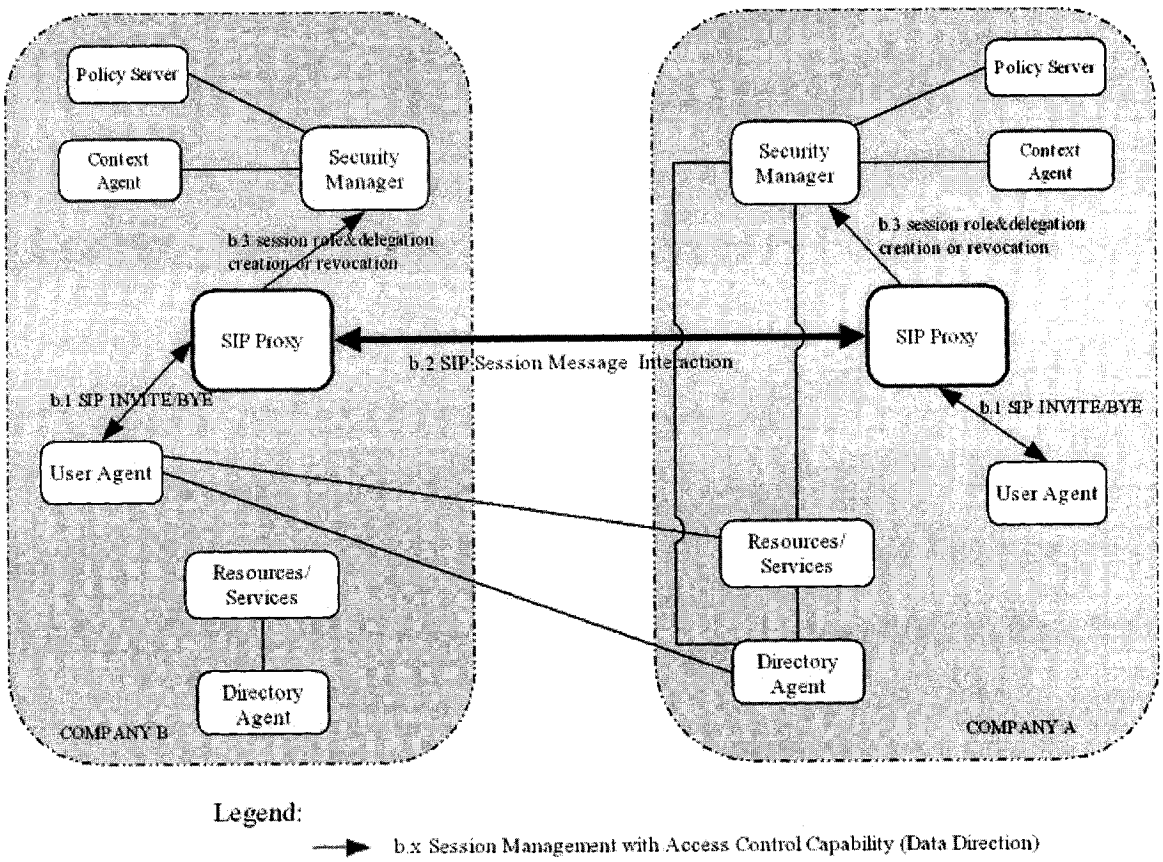

Figure 15 Session Management Phase

The author embeds a security manager agent into the SIP proxy server. This agent acts as a middle box [RFC3303] implementing a security service for the SCACA system. The

timeline pictured in Figure 16 illustrates the operations between a SIP proxy and a security manager through a security manager agent.

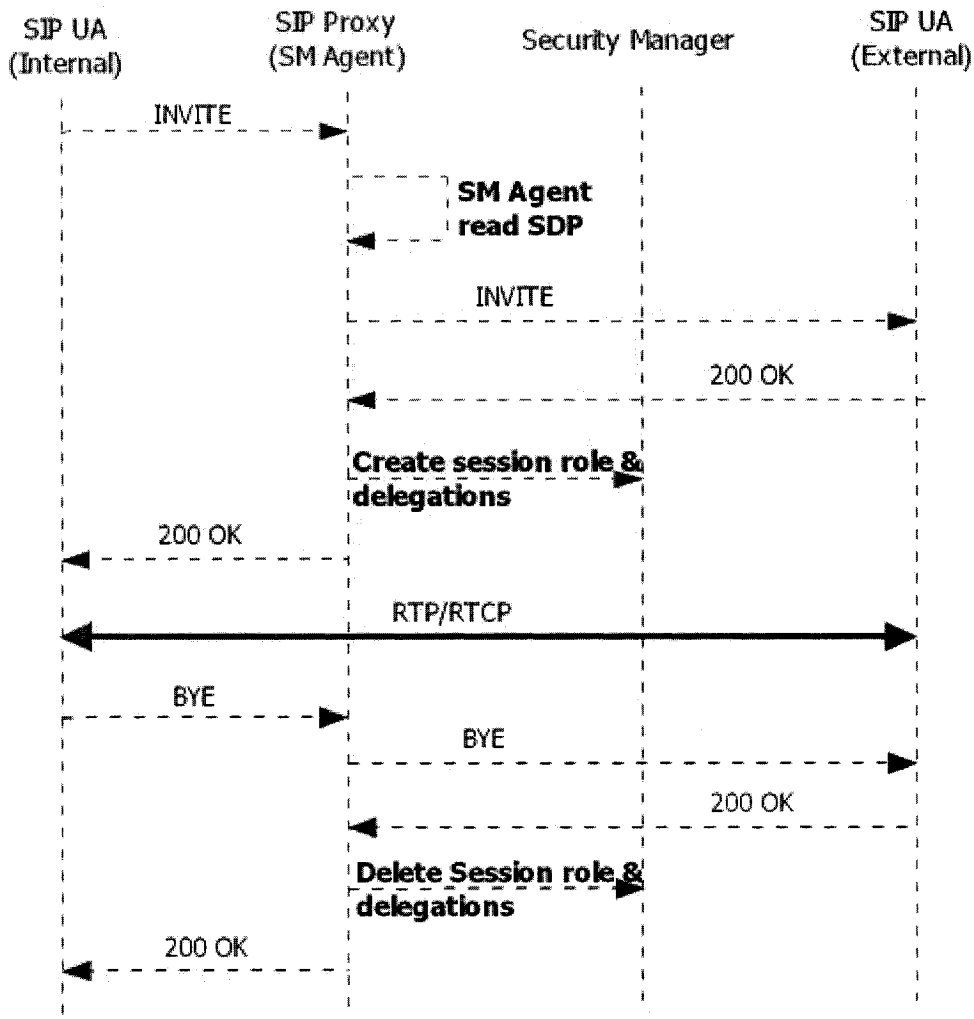

Figure 16 Time Line Flow – Security Manager Service

At the beginning of the session creation, Alice, the communication session initiator, sends a SIP INVITE message to the remote SIP user agent owned by Bob in companyB (b.1 in Figure 15). The INVITE message uses SIP URLs for addressing, and is assumed to carry a dRBAC SDP payload as described in section 5.1. The author also utilizes the "k=" field in SIP message to carry the URI information of user's credentials.

INVITE sip:bob@137.122.90.138 SIP/2.0
Via: SIP/2.0/UDP 137.122.88.36
From: <sip:alice@137.122.88.36>;tag=lcwlvkr
To: <sip:bob@137.122.90.138>;tag=zqkvqtw
Call-ID: 911071378@137.122.88.36
...

v=0
o=alice 1125184048268 1125184048268 IN IP4 AIAS356
s=Voice Conversation
k=uri:http://137.122.88.36/users/alice@137.122.88.36-CompanyA.member/deleg_000000_alice@137.122.88.36-CompanyA.member-CompanyA.xml
c=IN IP4 137.122.88.36
t=0 0
c=IN IP4 137.122.88.36
m=audio 40000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
c=IN IP4 137.122.88.36
m=application 8900 dRBAC

Upon receiving the above INVITE message, Bob accepts the invitation by sending back a SIP 200 OK response message to the inviter Alice. The response message may also carry a SDP payload including the security manager information in the invitee's domain.

SIP/2.0 200 OK
Via: SIP/2.0/UDP 137.122.88.36:1020;branch=z9hg4bk69e7f889f396821b0975f195bf2a913,SIP/2.0/UDP 137.122.88.36
From: <sip:alice@137.122.88.36>;tag=lcwlvkr
To: <sip:bob@137.122.90.138>;tag=zqkvqtw
Call-ID: 911071378@137.122.88.36
...

v=0
o=bob 1125184084312 1125184084312 IN IP4 AIAS317
s=Voice Conversation
k=uri:http://137.122.90.138/Users/bob@137.122.90.138-CompanyB.member/deleg_000000_bob@137.122.90.138-CompanyB.member-CompanyB.xml
c=IN IP4 137.122.90.138
t=0 0
c=IN IP4 137.122.90.138
m=audio 20000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
c=IN IP4 137.122.90.138
m=application 8900 dRBAC

Since all SIP messages share one Call-ID (for example: 911071378@137.122.88.36) in one SIP session, the author naturally uses this attribute value as run-time session role name in security manager. As shown in figure 16, upon receiving a 200OK response

message related to a previous INVITE message, the security manager agent, residing in the SIP proxy server, sends both inviter and invitee's certificates and the SIP Call-ID to its security manager, and ask the security manager to create a run-time session role named after the SIP Call-ID and related delegations as well (b.3 in Figure 15).

In order to make the session role related delegations automatically, the author allows users (inviter or invitee) to predefine policies for session role related delegations. An example of Alice's policy may say that Alice agrees to delegate her regular role to run-time session roles within SIP sessions. This delegation helps Alice further delegates her access rights on resources/services in her domain to external session members who belong to session roles. Session role membership management is introduced in section 5.5.2.2.

Moreover, in order to maintain the organizational wide security, the author allows the security manager to control whether a user can request to create a run-time session role within a SIP session. The author uses the access control standard language – XACML [XACML05] to define a policy for this control, an example can be found in Appendix C.

5.5.2.2 Session Role Member Management

Upon receiving an INVITE message, a callee can either accept or reject the invitation. After accepting the invitation message, all the session participants, no matter where they are, belong to the session role only when the communication session is active. The SCACA leverage SPKI (Simplified Public Key Infrastructure). The SPKI name certificate is used to identify a particular user, so the system is able to add a user to a

session role. For auditing purposes, all the session role member management work should be done by the delegator, which is the requesting user (either caller or callee), instead of the administrator or the security owner of the domain. Users are represented in SIP user name format in SCACA as shown in the following example.

[Bob@CompanyB -> CompanyA.sessionRole] ***Alice@CompanyA***

For a conference call, it is common that some participants quit a conference in the middle of a communication session. For example, Bob quits the session before the whole communication session ends. Two issues are considered here. First, Bob needs to revoke all the delegations that he gave out to the session role, generated when he joined the session, and delete the session role immediately in his own domain CompanyB, so all external session participants of CompanyB will not be able to access internal resources and services any more. Secondly, other domains' security managers need to remove Bob from their session roles in order to take back the access rights previously delegated to him. In the thesis's scenario, CompanyA's security manager will delete Bob from its session role, and then he will not be permitted accesses to the resources in CompanyA which Alice delegates to him through run-time session role in the beginning of the session.

Normally, a SIP BYE message will not traverse through the proxy, and instead reach to the remote endpoint directly, because the SIP UAs knows the endpoint address information after exchanging INVITE messages. In the design, the author forces every SIP message to go through SIP proxies, and then the SIP proxy will be able to capture the status of the ongoing session, so the middlebox security manager agent can notify

security manager to behave correctly. This is an example where a SIP proxy is being leveraged as a manager of a domain.

When a session participant leaves a communication session, the SIP proxy server captures the SIP BYE message, and then the security manager agent informs the security manager that a user is leaving the session. For the first issue, since the proposed context based coalition access control model has context constraints on the delegators, e.g. "activity == PhoneSession.SessionIDxxxx and location == MeetingRoom.xxxx". Context change triggers the change of delegation status, so those related delegations will become invalid immediately when Bob's activity is changed and is no longer in a communication session. The session role and session role delegations may be completely revoked according to the implementation. For the second issue, the security manager just simply removes the session member Bob from the session role.

In addition, the design allows a user manually revoke or even re-delegate his delegations anytime in an ongoing SIP session.

5.5.2.3 Termination of a Communication Session

When all the participants leave the session, the conference ends and the communication session is over. All the delegations activated within the communication session are deactivated or revoked, and then the situation will go back to the normal that there is no relationship or delegation between organizations, and users cannot access resources and services located in other organizations although they can do so when the session is alive.

Similar to the above section, SIP BYE messages trigger this security control through middlebox security manager agent.

5.5.3 Access Control in a Communication Session

Suppose that the delegators have given out their access rights to all session members by delegate their roles to the session role, the next step is on the client side, that is, how a client finds services, sends an access request, and then how the security manager controls the access.

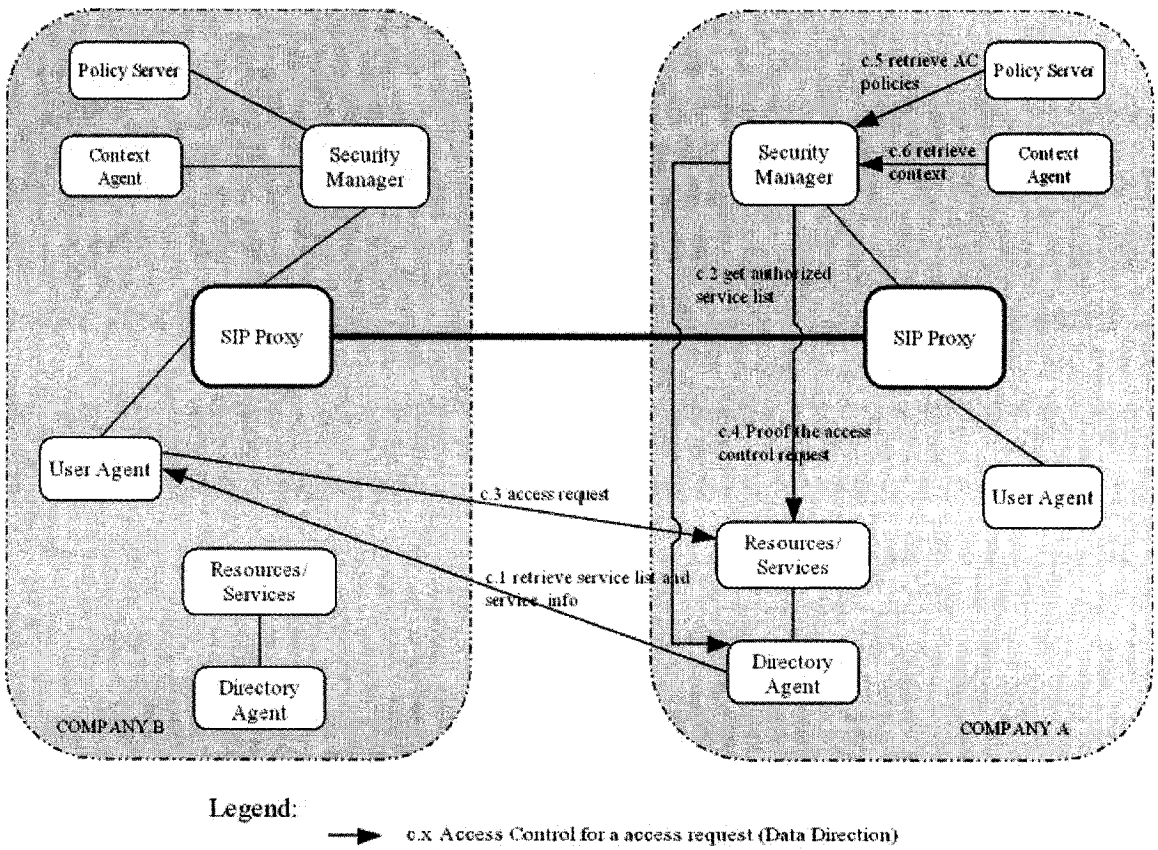

Figure 17 Access Control Phase

5.5.3.1 Service Discovery

In the service registration phase discussed in section 5.5.1, all internal services register on a SLP DA, their domain's directory agent, so SLP UA clients will be able to query and retrieve service information from the directory agent. However, in inter-organizational environments, the external SLP UA clients have difficulty to do so, since they have no a-prior knowledge about the internal SLP DA, the directory agent's contact information.

Similar to the dRBAC SDP proposed in section 5.4.1, the author adds the directory agent's description information in SDP message, and attaches it to the SIP INVITE or INVITE RESPONSE message, so both SIP endpoints will know each other's directory agent. An example SDP is shown below.

v=0
o=alice 1125184048268 1125184048268 IN IP4 AIAS356
s=Voice Conversation
k=uri:http://137.122.88.36/users/alice@137.122.88.36-CompanyA.member/deleg_000000_alice@137.122.88.36-CompanyA.member-CompanyA.xml
c=IN IP4 137.122.88.36
t=0 0
c=IN IP4 137.122.88.36
m=audio 40000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
c=IN IP4 137.122.88.36
m=application 8900 dRBAC
c=IN IP4 137.122.88.36
m=application 9000 SLP

In the design, a remote external user first actively contacts the internal directory agent for a service list with the contact information in the SDP message (step c.1 in Figure 17). Upon receiving the query, the directory agent asks its security manager what services are available or authorized for this user (step c.2 in figure 17), since the dRBAC model supports object query: [user -> ?]. The directory agent then sends back the available

services and their service description information stored in the directory agent to the remote client.

The remote user picks up a service from the received service list, and asks the directory agent for the service's detailed contact information. Then the user can send an access request to the service or resource directly. Considering the various services may change from time to time, a pull model should be used for the client to proactively retrieve new service list.

5.5.3.2 Access Control Flow

There are five functional components playing in the access control phase, they are client, resource, security manager, policy server and context agent. A client is the requestor sending requests. Resources are data, service and system components that users want to get accesses to, and they may act as PEPs in access control system. In most systems, some access gateways provide PEP services for resources. A security manager acts as a PDP to receive access requests from PEPs, proof those requests and return back access control decisions to the PEPs, and then the PEPs enforces the received decisions. When a security manager proofs an access request through delegation chains, it may also retrieve related access control policies and context information from its policy server and context agent.

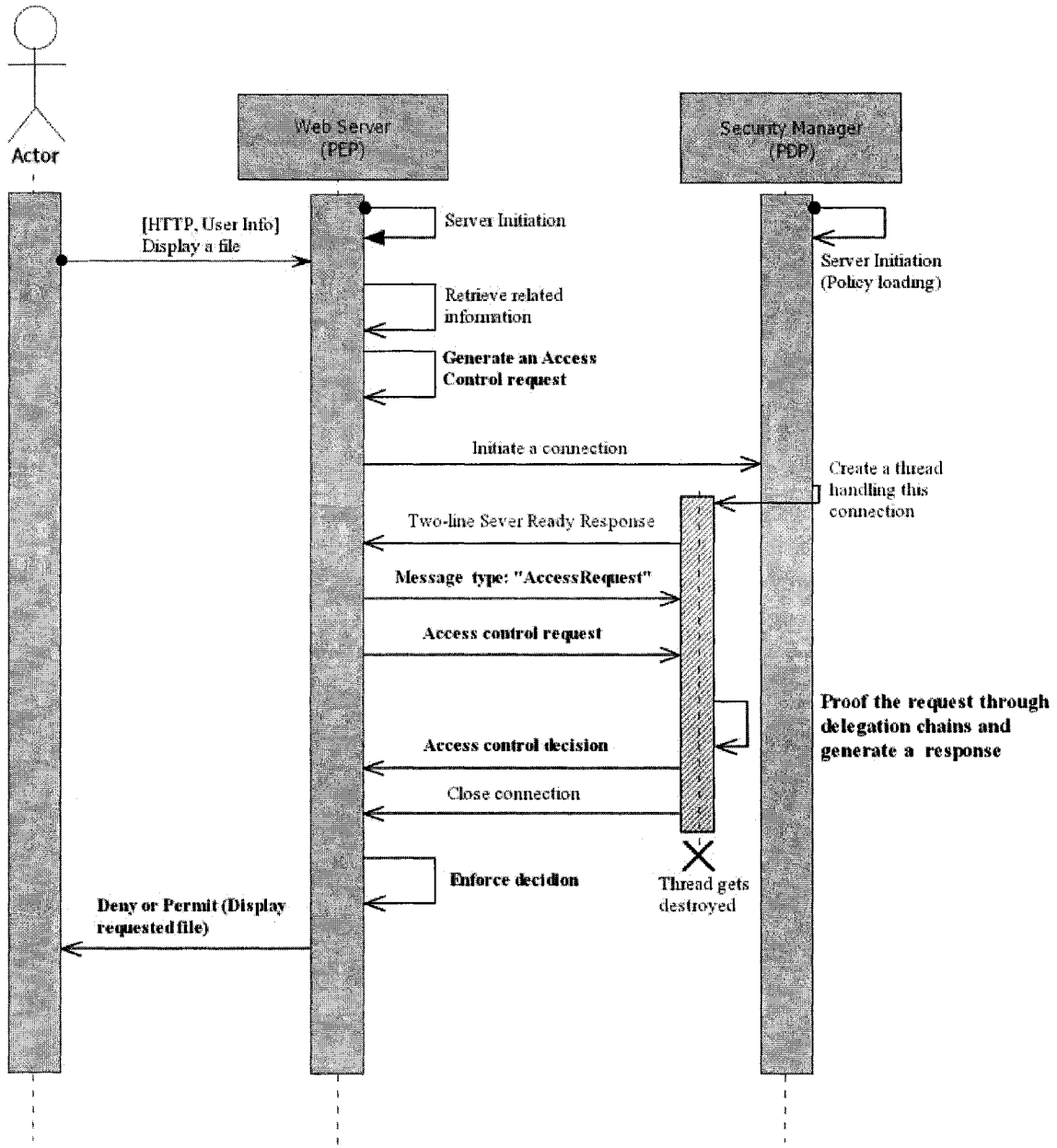

Figure 18 Access Control Sequence Diagram

The above sequence diagram shows the interactions between the main actors: requestor, resource/service, and security manager for an http access to an internal resource from a external user. The following steps illustrate the access control process in detail (steps c.3 – c.6 in Figure 17):

1. The external user picks up a service and sends an access request to the resource (step c.3 in Figure 17). In the design, a SPKI certificate or its URI is attached to every access request.

2. Upon receiving an access request, the resource/service transforms the native request to an access control request and sends it to the PDP (Policy Decision Point), its Security Manager, and waits for an access control decision (step c.4 in Figure 17);

3. The security manager (PDP) starts to proof the request through delegation chains, and it may also query related access control policies from the policy server (step c.5 in Figure 17), retrieving related context information from context agent (step c.6 in Figure 17). For example, the session role delegator's context information may be retrieved to compute a delegation constraint to help decide whether a delegation is active or not. The security manager (PDP) then makes an access control decision based on the proofing result, and sends the response back to the requesting PEP for enforcement (step c.4 in Figure 17);

4. The PEP enforces the decision. If the access is permitted, then the PEP permits the access to the resource; otherwise, it denies the access.

5.5.3.2 Delegation Proofing

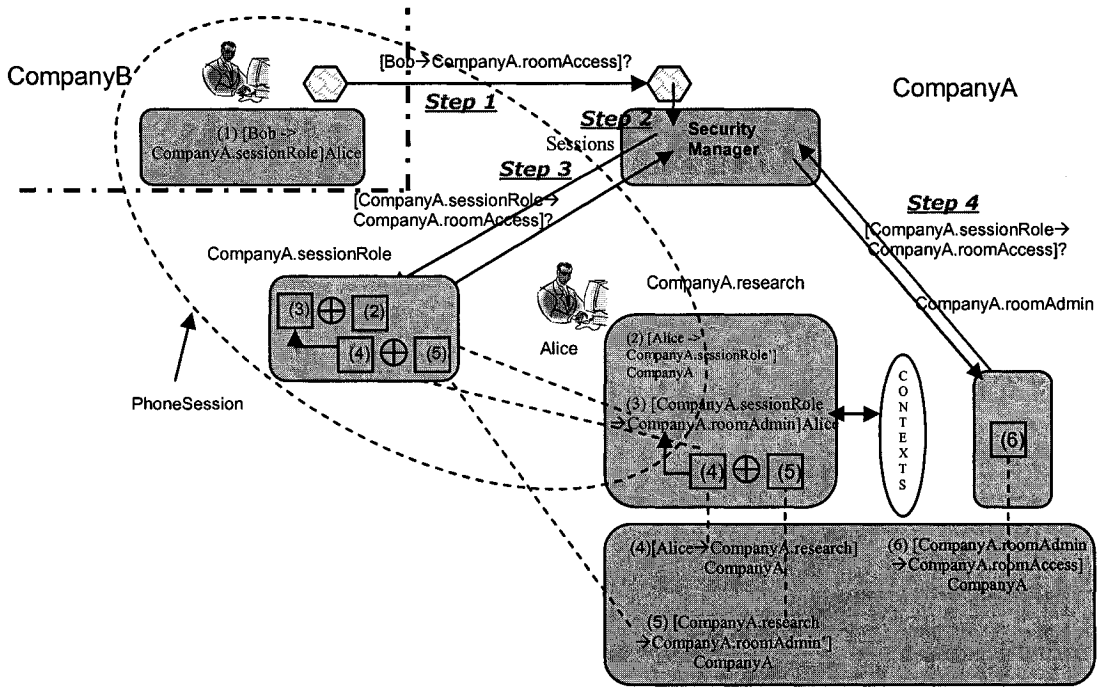

Figure 19 The SCACA Proofing Process based on dRBAC

This section illustrates the delegation proofing process in the SCACA framework. Recalling the scenario described in section 1.2, Alice, the session initiator, creates or activates delegations related to the conference call based on the communication context. Since Alice is currently in a communication session and physically in a meeting room in Company A. The contextual information is defined as follows:

$$activity == PhoneSession.SessionID1234 \text{ and } location == MeetingRoom.SITE4004$$

The roles that Alice can delegate can be manually selected by Alice from a prompted interface or be automatically done based on predefined policies. Moreover, other participants, like Bob, may also delegate their roles in their own organizations to the run-

time generated session role, so that the session participants can access each other's resources across organizations.

In the following, the thesis summarize the delegation proofing steps illustrated by Figure 19, in which Bob, a member of companyB, requests an access to the resources located in the meeting room in companyA, at the time that Bob is talking with Alice in CompanyA over a phone session. Note: since the SPKI is used throughout the design, the user-role relation is defined by delegation instead of database approaches.

(1) *[Bob@CompanyB -> CompanyA.sessionRole]Alice@CompanyA*:

The starting point is to validate Bob's access to CompanyA.roomAccess, which is an object role grouping some access rights to certain resources. To authorize the access, the CompanyA's security manager must discover a proof for [Bob@CompanyB -> CompanyA.roomAccess]. The first delegation is that of Bob to a run-time generated session role: CompanyA.sessionRole, and this delegation is signed by the caller Alice in CompanyA. The next step is to discover whether Alice has the right to delegate the run-time generated session role.

(2) *[Alice@CompanyA -> CompanyA.sessionRole'] CompanyA*:

This is a third-party delegation indicating that the root of CompanyA authorizes Alice has the right to further delegate a run-time session role. Then the security manager issues a subject query to obtain all proofs of the form [CompanyA.sessionRole -> *]. In response to this query, it discovers delegation (3)

defining a relationship between the roles CompanyA.sessionRole and CompanyA.roomAdmin.

(3) *[CompanyA.sessionRole -> CompanyA.roomAdmin] Alice@CompanyA (activity == PhoneSession.SessionID1234 and location == MeetingRoom.SITE4004)*:

The security manager discovers the delegation has contextual constraints applied on the delegator Alice. It first retrieves the real time contextual values of Alice from a context agent or a SIP location server and checks the value of the context conditions. If the condition is false, this indicates the delegation currently is not valid and hence the access control decision is made that the request is denied. If the condition is true, then the security manager further discovers that the delegation (3) must have been accompanied by its support proof. In this case, the latter comprises of delegations (4) and (5).

(4) *[Alice@CompanyA -> CompanyA.research] CompanyA*:

Alice is delegated the role CompanyA.research.

(5) *[CompanyA.research -> CompanyA.roomAdmin'] CompanyA*:

This is a third-party delegation that the all members holding the role CompanyA.research have the authority to further delegate the role CompanyA.roomAdmin. At this point, the security manager has discovered a chain from Bob to CompanyA.roomAdmin, but is still missing a proof that would authorize [CompanyA.roomAdmin -> CompanyA.roomAccess]. To obtain this, it continues with its forward search by contacting the home dRBAC wallet

corresponding to the role CompanyA.roomAdmin, and issuing to it a direct query for [CompanyA.roomAdmin -> CompanyA.roomAccess]. The response to this query is a self-certified delegation (6).

(6) *[CompanyA.roomAdmin -> CompanyA.roomAccess] CompanyA*:

Now the proof authorizing [Bob@CompanyB -> CompanyA.roomAccess] is complete.

5.6 Authentication across Domains

The access control mechanism the author discusses here is actually an implementation of authorization, which is the process of giving individuals access to system objects based on their identity, however a comprehensive security system also requires authentication that is the process of identifying an individual, although it is not addressed much in the thesis.

For authentication in one organization, a centralized certification authority is required. Public Key Infrastructure (PKI) is widely used to manage keys and certificates. PKI uses global unique naming system (X.500) to name users and securely binds the names of users to their public keys.

Regarding a realistic distributed system, in which communications happen between several organizations or independently administered subsystems, the existence of centralized certificate authority is impossible. Hence it's difficult for an organization's

security server to authenticate an access requestor from other organizations, even though that requestor is certified by the remote certificate authority.

SPKI/SDSI is designed for distributed system with local naming system and local certificate authority. Without centralized certificate authority, the trust relationships must be explicitly established across organizations in SPKI/SDSI, so that an organization will be able to trust a certificate created by external authority. Moreover, in SPKI, authentication is not really "authentication", and it is a process of authorization of answering a delegation authorization query.

5.7 Related Security Issues

Currently SIP security is still an ongoing topic. As SIP is evolving from HTTP and SMTP, all security mechanisms available for them can also be applied to SIP sessions. With the assistance of PKI, digital certificate can be embedded in SIP for authentication, and the encryption of MIME bodies, e.g. SDP payload, is supported. In the network level, the use of SIPS URI (TLS over TCP) [SKS04] and IPsec can secure the communication between coalition partners in public domain.

Incorporating SIP into access control also brings new challenges as user's various behaviors at SIP level may cause security holes. For example, one endpoint invites another person or redirects an ongoing session to an alternative device, and then withdraws from the session. The author is not clear whether this should be allowed or not because this would somewhat violates the service providers' security requirements, which may be only given access to those original session participants. This work needs to

be expanded to handle the interactions between the SIP signaling layer and the access control layer.

Chapter 6 Validation

The proposed SCACA approach is a framework system, and its validation is very similar to the process of creating an Internet Standard in IETF. The implementation and testing is one of the key goals of the Internet Standards Process in IETF. In this chapter, the author validates the designed system framework through a proof of concept implementation.

Protecting network systems from attacks is critical for a security system. The most serious attacks to computer security are to hack the authentication, and attack servers providing all kind of services. Access control is a sub domain of computer security. Attacks are not appropriate for access control. The best way of validating the access control system is to test how the system controls the accesses, that is an unauthorized access can be denied and an authorized access can be allowed. This validation will show that access is only available when the session is active.

In this chapter, a prototype system is implemented to demonstrate that a user Bob at organization CompanyB easily gets access to a confidential salary document located on a web server in a remote organization CompanyA, when Bob is talking with Alice over an audio conversation session. Bob's access rights are gained through delegations made by both Alice and CompanyA at the time the multimedia session is set up, and Bob's access automatically gets revoked after the session is over. Vice versa, at the same time, Alice at CompanyA can access resources in CompanyB in the same easy way.

Regarding the context agent and directory agent, the author didn't manage to implement them in the demo validation, since there are already many context agent and directory

agent systems in place. Since the dRBAC implementation supports a subject to object query, the idea of secure service discovery can be implemented easily, and the combination of delegation and access control policies is more difficult. This prototype is focused on leveraging a communication session as a dynamic activity context to demonstrate the proposed SCACA framework supports spontaneous coalition access control.

6.1 Prototype Environment

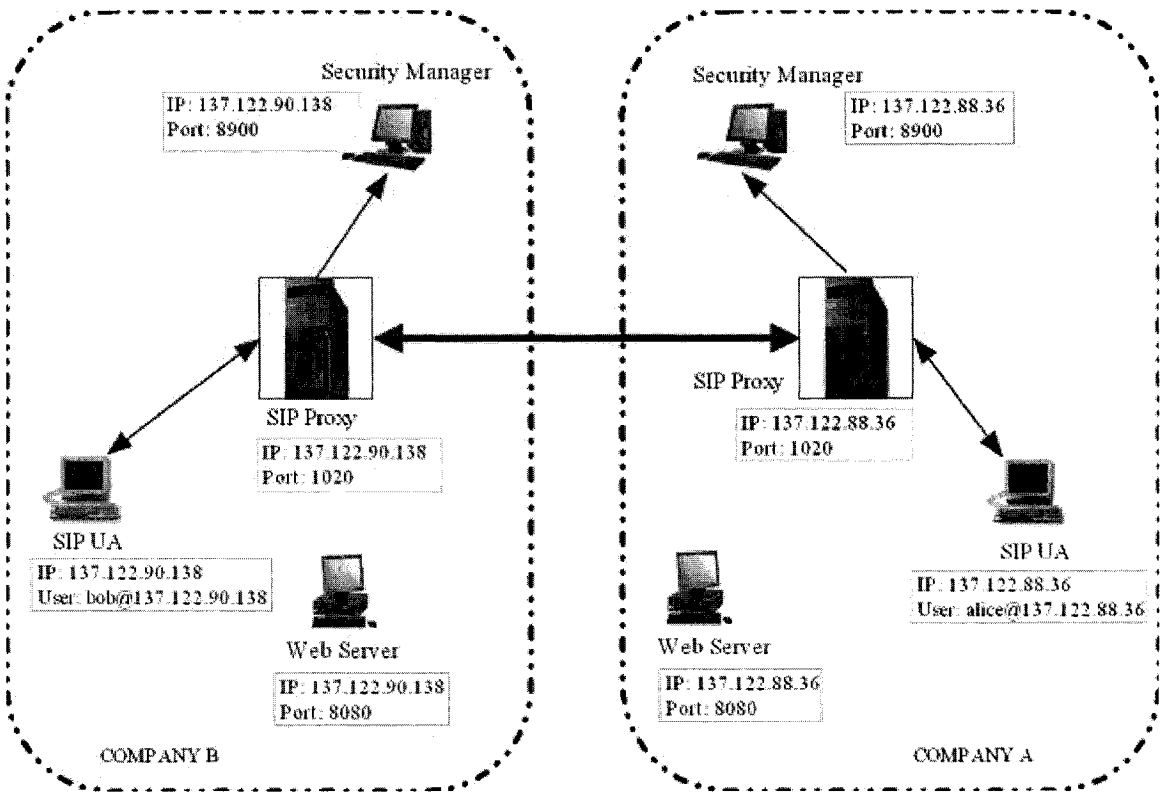

Figure 20 Simulation Environment

Figure 20 depicts the overall system environment of this simulation. As shown in the figure, this simulation includes two organizations: CompanyA and CompanyB. The

minimal devices needed for this simulation are two computers, one for each organization. The machine with IP address 137.122.88.36 represents CompanyA with a SIP proxy, SIP UA, Security Manager and web server running on it, and similarly the machine with IP address 137.122.90.138 represents CompanyB with its SIP proxy, SIP UA, Security Manager and web server running on it. Since all services including SIP proxy, security manager, and web server run on different ports on same machine, they can be physically installed on different servers when this system is placed into the real world. The prototype supports various configurations.

Four processes, a SIP UA, SIP proxy, security manager, and web server are required to run simultaneously to exercise the prototype properly.

6.2 Prototype Configuration

Before running the prototype, the thesis first describes the necessary configurations of the key components in the SCACA implementation.

For the two SIP proxies in figure 20, each proxy is in charge of one organization, and set the other proxy as its next hop, so SIP messages will be able to be routed to their destinations properly. Since a SIP proxy server plays a very important role in the proposed SCACA framework, specifically a SIP proxy server must remain on the path of the future requests in a SIP dialog, the author enables "Record-Route" functionality in SIP proxies in the prototype.

The last configuration of proxy is about access control. The author creates a configuration file for administrators to decide whether to enable the session based access control functionality of the SIP proxy server, for example:

"allow_session_access_control : ON"

When the parameter is set to "OFF", the proxy behaves exactly the same as a normal SIP proxy server. In addition, the prototype also allows administrators to specify the security manager's IP address and port number in that configuration file, so the security agent embedded in the SIP proxy server will be able to communicate with its security manager properly.

In the prototype implementation, every security manager wraps a dRBAC wallet in it. To make this prototype simple and easy to be understood, the prototype applies similar configurations for both CompanyA and CompanyB, and statically define the following user roles, object roles, resources and delegations as follows. For the representation of a delegation used in the prototype, please refer to appendix B.

	CompanyA	CompanyB	Description
Domain	137.122.88.36	137.122.90.138	
Root Entity	CompanyA	CompanyB	Represent the root of the domain.
User name	alice@137.122.88.36	bob@137.122.90.138	Use SIP URI as user name in security manager.
User role	CompanyA.member	CompanyB.member	Every user in an organization is a member of Company.member role.
Object role	CompanyA.access	CompanyB.access	Represent bunch of access rights to a group of resources, here is specifically the privilege to read a confidential salary document on web server.
Delegations	[CompanyA.member-> CompanyA.access] CompanyA; [Alice@137.122.88.36 -> CompanyA.member'] CompanyA;	[CompanyB.member-> CompanyB.access] CompanyB; [Bob@137.122.90.138-> CompanyB.member'] CompanyB;	Pre-defined delegations which are necessary for exercising the prototype.

Table 3 Pre-defined Prototype Settings

All delegations run-time delegated to a session role are made by users, either inviters or invitees, whoever belongs to the domain. The delegators are responsible for signing delegations and giving the access rights out, however in practice an organization may feel somewhat lose overall control on session role delegation management when there are many session roles get created and many delegations get generated in run time. To address this issue, besides the above settings, the prototype leverages a XACML policy for organizational administrators to control whether users could ask for creating new roles. For an example of this XACML policy, please refer to appendix C.

Regarding the SIP UA's configuration, a user configuration interface is designed below to facilitate user's configuration.

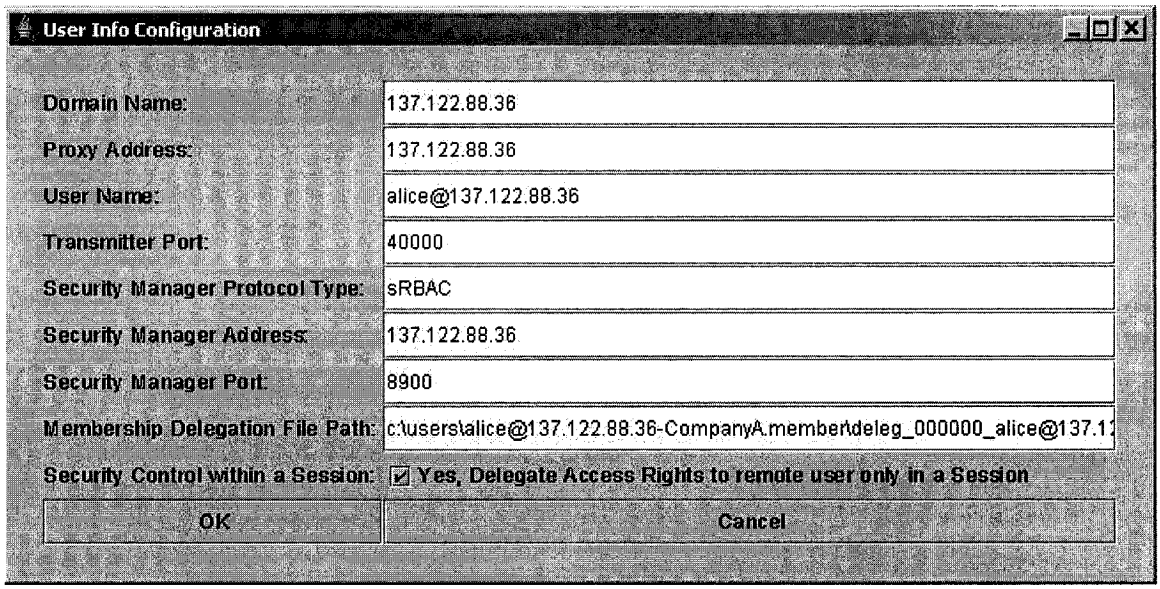

Figure 21 User Configuration Interface

As displayed in the above figure, besides the proxy and audio transmission settings, the last five settings are all for security. A user must specify its security manager's addressing information, and also the URI of his or her credential information, which will be inserted into SIP messages transferred to remote endpoints through proxies. The best URI might be a HTTP URI because it is easy to be reached anywhere. Here the author just simply uses local URIs on hard drive to represent it. The last check box allows user to control whether to turn on the access control functionality in a SIP UA. If it is unchecked, then the SIP UA behaves like a normal SIP UA.

When users access resource (PEP), the resource forwards the access request to security manager (PDP) for an access control decision. Regarding the configuration for PEP, the author creates a configuration file for PEP with security manager's IP address and port number information in it, so that the resource will be able to communicate with its remote security manager properly.

In order to play the simulation, one user is needed at each side. As mentioned in the table 3, the author adopts SIP URI as user's name instead of user's real name in security manager that are alice@137.122.88.36 and bob@137.122.90.138.

6.3 Prototype Execution

The following illustrates and describes the three major phases of the proposed SCACA framework. Some results and outputs are shown as well.

6.3.1 Session Initiation and Delegation Creation

First of all, the author manually starts the two SIP proxy servers, security managers and web servers on both machine 137.122.88.36 and 137.122.90.138. These processes are supposed to keep running all the time.

Since the simulation has user Alice at CompanyA with username alice@137.122.88.36, and Bob at CompanyB with username bob@137.122.90.138, in the beginning, both user Alice and Bob start their SIP UAs, and register on their own SIP proxies as well.

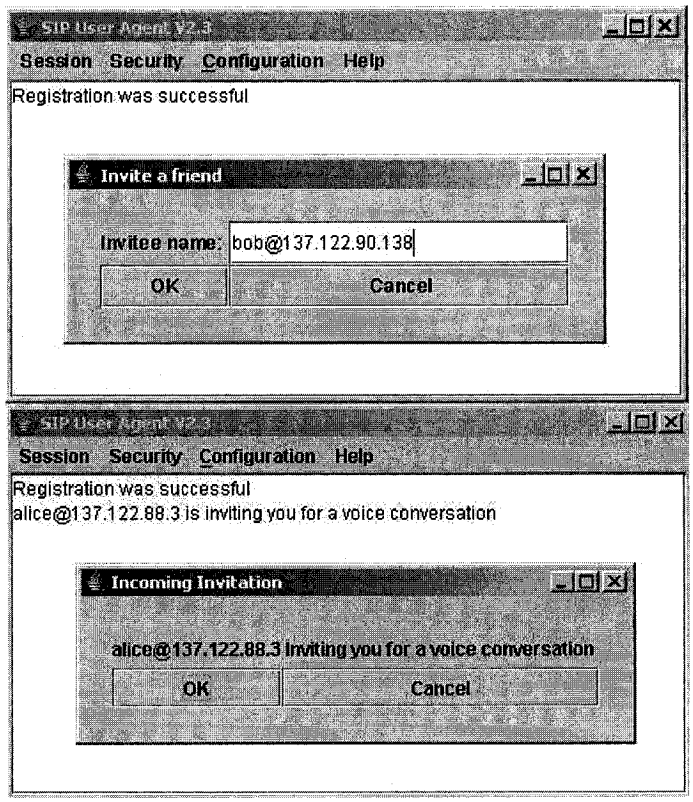

Figure 22 SIP session initiation of the simulation

Shown in Figure 22, user Alice at CompanyA invites user Bob at CompanyB to a voice conversation session by inputting the SIP username of Bob in the invite window of SIP UA program. The SIP message of Alice's INVITE request is below. The messages are captured by the SIP proxy message monitoring utility.

INVITE sip:bob@137.122.90.138 SIP/2.0
Via: SIP/2.0/UDP 137.122.88.36
From: <sip:alice@137.122.88.36>;tag=eamyqwj
To: <sip:bob@137.122.90.138>;tag=ehuapci
Call-ID: 353791834@137.122.88.36
...

v=0
o=alice 1126971164275 1126971164275 IN IP4 AIAS356
s=Voice Conversation
k=uri:c:\users\alice@137.122.88.36-CompanyA.member\deleg_000000_alice@137.122.88.36-CompanyA.member- CompanyA.xml
c=IN IP4 137.122.88.36
t=0 0

c=IN IP4 137.122.88.36
m=audio 40000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
c=IN IP4 137.122.88.36
m=application 8900 dRBAC

According to the SDP standard [RFC2327], the author extends Mitre's Java SDP API to support two service descriptions in one SDP message, and especially enable more than one services located at different IP addresses. Currently there are two service descriptions for audio and security manager in the above SIP message. In addition, the URI of Alice's credential file is set to be the value of "k=" field in the SIP message, so the remote endpoint or security manager will be able to authenticate the user. Upon receiving Alice's SIP INVITE message, Bob accepts the invitation, generates a SIP 200 OK response message below based on his service information and then sends it back to Alice's SIP UA through SIP proxies.

SIP/2.0 200 OK
Via:SIP/2.0/UDP
137.122.88.36:1020;branch=z9hg4bkf512c6cb1ae67aecd6265f184f6ae6e,SIP/2.0/UDP 137.122.88.36
From: <sip:alice@137.122.88.36>;tag=eamyqwj
To: <sip:bob@137.122.90.138>;tag=ehuapci
Call-ID: 353791834@137.122.88.36
...

v=0
o=bob 1126971120093 1126971120093 IN IP4 AIAS317
s=Voice Conversation
k=uri:C:\Users\bob@137.122.90.138-CompanyB.member\deleg_000000_bob@137.122.90.138-CompanyB.member-CompanyB.xml
c=IN IP4 137.122.90.138
t=0 0
c=IN IP4 137.122.90.138
m=audio 20000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
c=IN IP4 137.122.88.36
m=application 8700 dRBAC

Since both Alice and Bob's proxies all enable the access control functionality in configuration, they receive and read the related SIP INVITE message, its response messages, and the embedded SDP messages as well, and then the security agents on the

proxies communicate with their respective security managers. The security managers in CompanyA and CompanyB individually create run-time generated session roles and delegations below:

Security Manager at CompanyA	
Run-time Session Roles	*CompanyA.353791834@137.122.88.36* *CompanyA.353791834@137.122.88.36'*
Automatic Run-time Delegations	*[CompanyA.353791834@137.122.88.36 -> CompanyA.member w/[]] alice@137.122.88.36* *[alice@137.122.88.36 -> CompanyA.353791834@137.122.88.36' w/[]] CompanyA* *[bob@137.122.90.138 -> CompanyA.353791834@137.122.88.36 w/ []]* *alice@137.122.88.36*
Security Manager at CompanyB	
Run-time Session Roles	*CompanyB.353791834@137.122.88.36* *CompanyB.353791834@137.122.88.36'*
Automatic Run-time Delegations	*[CompanyB.353791834@137.122.88.36 -> CompanyB.member w/[]] bob@137.122.90.138* *[bob@137.122.90.138 -> CompanyB.353791834@137.122.88.36' w/[]] CompanyB* *[alice@137.122.88.36 -> CompanyB.353791834@137.122.88.36 w/ []]* *bob@137.122.90.138*

Table 4 Run-time Generated Session Roles and Delegations

To authorize the access request for a confidential salary at CopmanyA from user Bob at CompanyB, the security manager in CompanyA needs to discover a proof for [bob@137.122.90.138 -> CompanyA.access], in which the object role CompanyA.access includes the access rights for reading the confidential salary document. This process is similar to the proofing process the thesis describes in section 5.5.3.2. For the detailed pre-defined static delegations and roles, please refer to Table 3. In addition, in the prototype implementation, the delegation [CompanyA.353791834@137.122.88.36 -> CompanyA.member w/[]] signed by the user alice@137.122.88.36, as shown in the above table, is made automatically according to the user's pre-defined policy on the security manager. In the real world, this can be done manually by users themselves to eliminate their security concerns.

6.3.2 Access Control

After the above steps, all necessary session roles and related run-time delegations are in place. In order to validate the operations of SCACA and test the access control, the author uses a confidential salary document on a web server as a resource to be accessed. On the client side, a HTML form helps the user build an access request with user's credential information in it. A Java Servlet on the web server side functions as a PEP to accept access requests, forward requests to its security manager and then enforce the received access control decision.

The test below shows that Bob at CompanyB can access a confidential salary document on the web server at CompanyA when the session is ongoing.

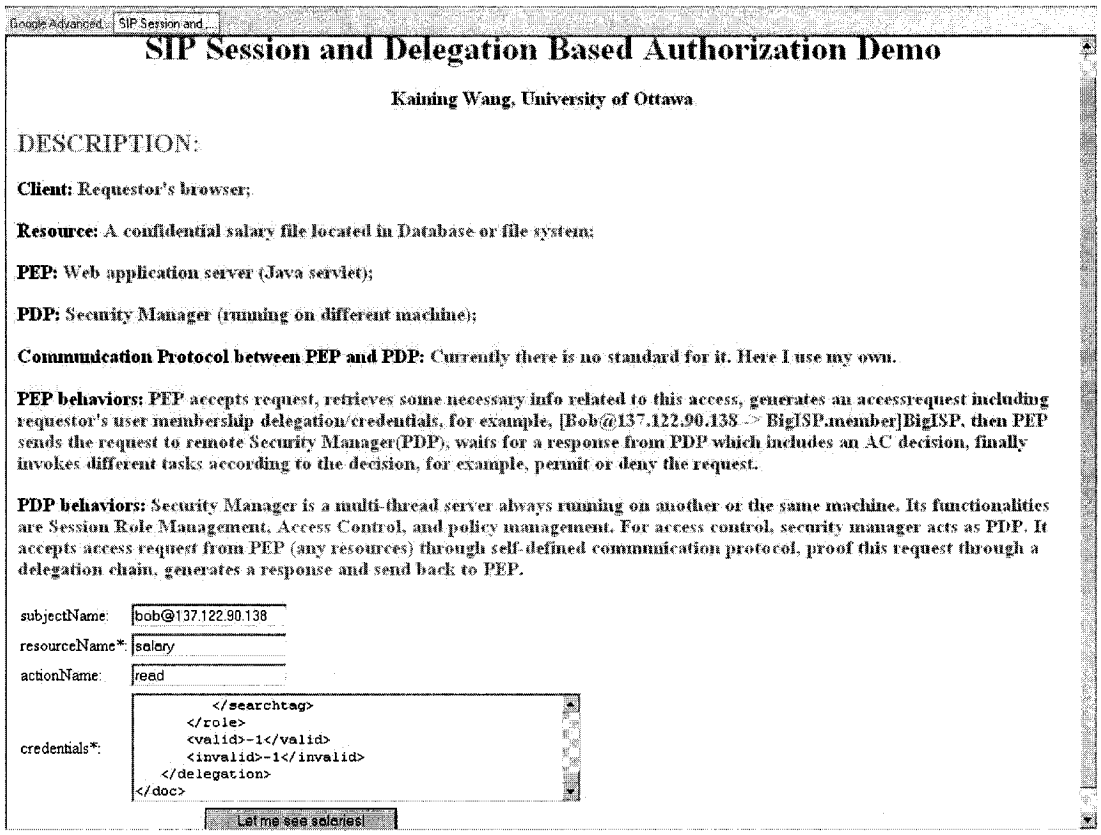

Figure 23 Build an Access Request with a HTML Form

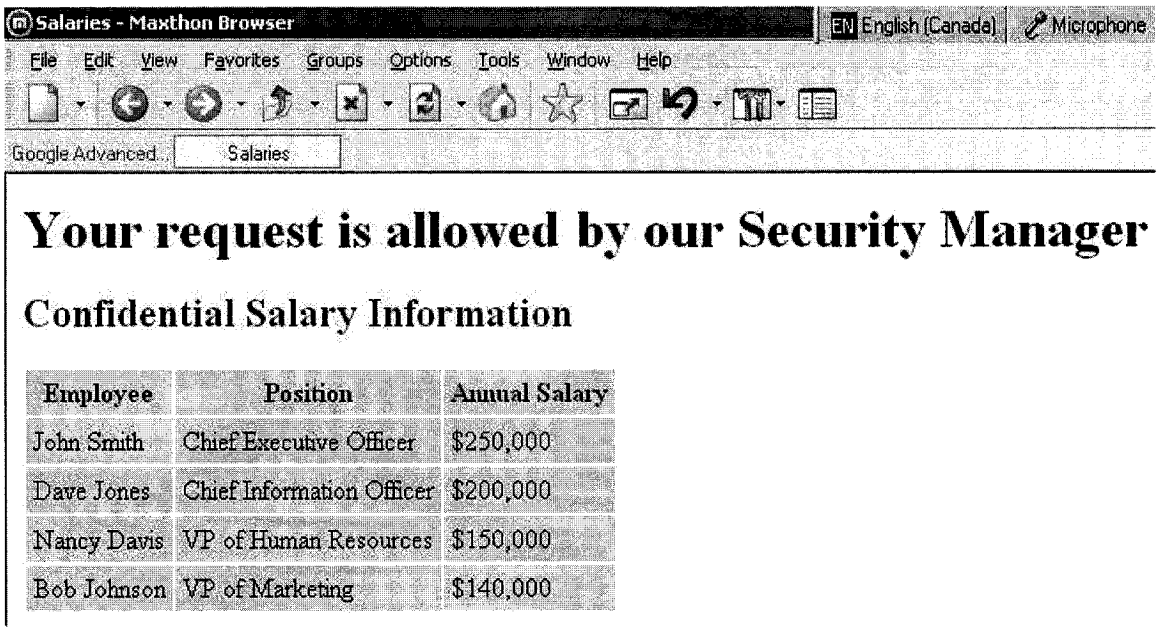

Figure 24 Result of Successful Access

6.3.3 Session member and role management

When user Bob wants to leave the audio conversation, he clicks menu item "Send Bye" in his SIP UA to send a SIP BYE request to remote user Alice at CompanyA. Since the proxy and SIP UA have been enabled to force the SIP requests traveling through proxies, both CompanyA and CompanyB's SIP proxies will receive the SIP BYE request. Hence the security manager in CompanyA will be able to delete member Bob from its session role below first by revoking the session role member delegation in the prototype, and all other related delegations to this session role as well, since this simulation is only a two-party SIP session and there is no other member left in this session role.

```
[bob@137.122.90.138 -> CompanyA.353791834@137.122.88.36 w/ []] alice@137.122.88.36
```

For the security manager in CompanyB, the session role and all run-time delegations related to this session role will be deleted and revoked too (shown in Table 4), because Bob is the original requestor for creating the session role in CompanyB and this is only a two-party session scenario.

In the prototype, the author has implemented full functionalities for session role and member management, which are suitable for multi-party multimedia communication

session. Figure 25 shows, during an active voice conversation session, user Alice click the menu item "Security -> Revoke delegation" in her User Agent GUI to manually take back the access rights which were given out to the session members in the beginning of the session. The outputs of the interactions between Alice's UA and her security manager are shown in the GUI. At this time, user Bob at CompanyB can not access the confidential salary document any more unless Alice manually ask security manager to re-create the session role and re-generate necessary delegations by click the menu item "Security -> Invoke delegation" in her UA GUI.

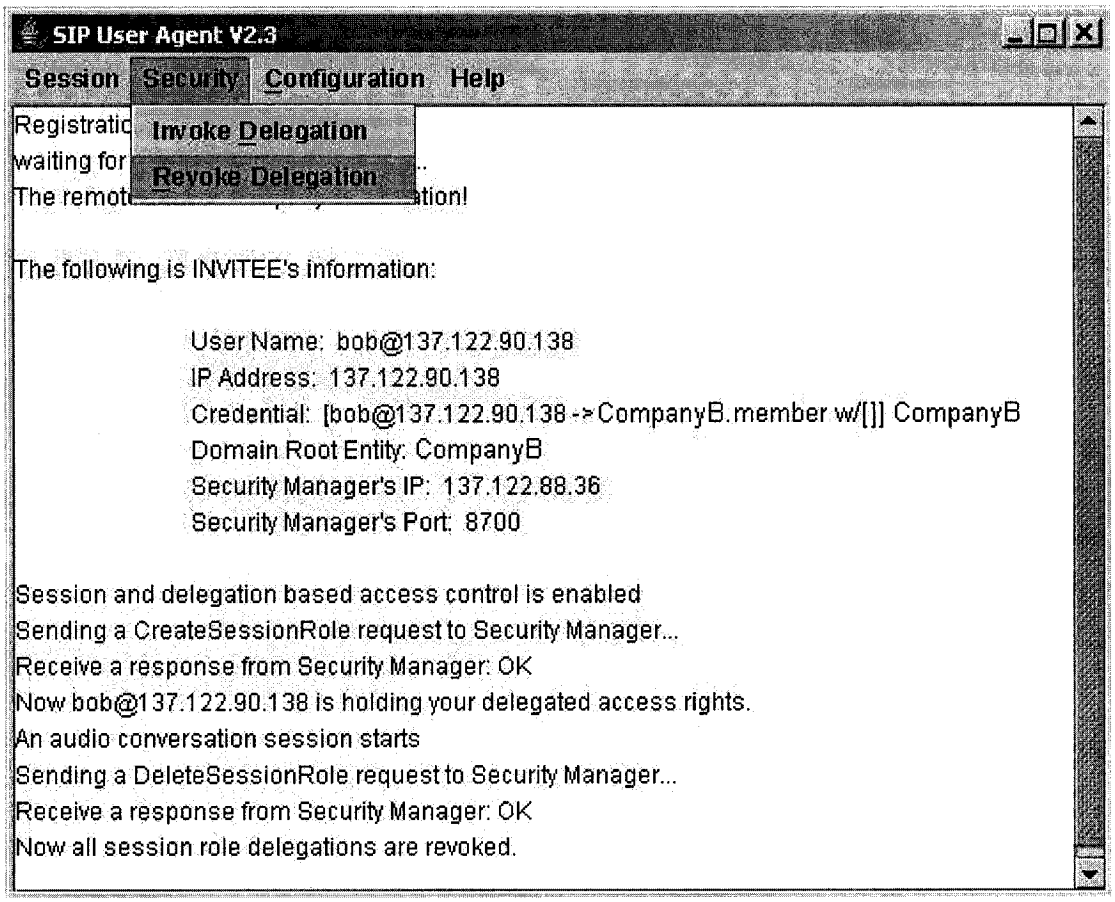

Figure 25 Session role and Delegation Management within an Active Session

Below shows that user Alice can't access the confidential salary document on CompanyB's web server after Bob leave the session.

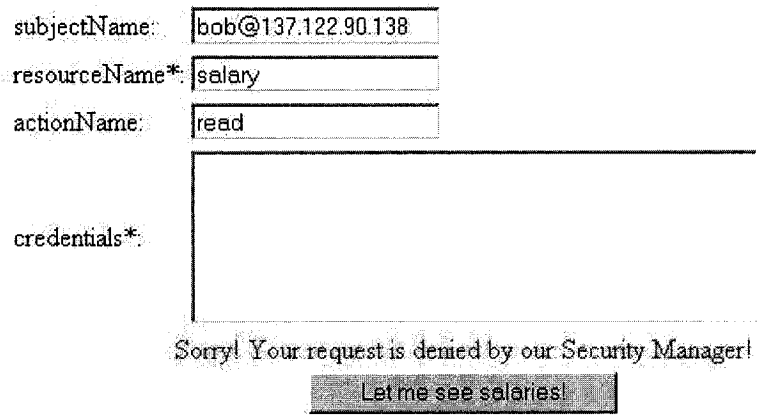

Figure 26 Result of Failed Access

Chapter 7 Conclusions and Future Work

7.1 Conclusions

In the thesis, the author has addressed the problem of developing access control models and a general architecture to facilitate secured access in a spontaneous coalition. The author discovered that the delegation mechanism places a significant trust on the delegator such that there are no restrictions on "who" the delegation is for. Delegation is leveraged to greatly minimize the administration overhead in the spontaneous coalition access control, since the delegation management can be done by users themselves rather than the administrators, and there is no need to share any common knowledge between organizations and rely on a 3rd party to administrate trust relationships.

Furthermore, context information is introduced into the spontaneous coalition access control to make the access control adapt to the dynamic nature of spontaneous coalition. The author places some form of acceptable context-based restrictions on the delegation because the dRBAC model is too restrictive for ad hoc scenarios. In the proposed SCACA framework, access rights are given out through delegations within an activity context that is a communication session, in which every party may know each other a-prior.

In comparison, the context used in the contextual delegation model is status information, and its application is a kind of passive context awareness application, while the SCACA is more likely an active context awareness application. Basically, the active context can

provide more dynamic feature then the passive context. The SIP communication session is a kind of active context that can change the application's behavior, since in our SCACA framework it will not only be used for the context constraints, but also for the generation and revocation of the delegations. The delegation generation has to be done by some sort of event handling mechanisms.

In the thesis, the author first proposes the D-TMAC model by extending the TMAC model with delegations, and the extension to the dRBAC model is proposed by applying context conditions on delegations to provide more flexible, dynamic and fine-grained access control. The SCACA, the focus of this thesis, a session and delegation based architecture is proposed that leverages the communication session as context to facilitate the spontaneous coalition access control. A session role is also proposed in SCACA for easily managing the access control in spontaneous coalitions. As discussed in section 5.4.2, the session role approach offers scalability in the number of parties in the spontaneous coalition although the prototype demo use two-party scenario in the validation.

The central idea of the thesis is to use context-based delegations to minimize the amount of administrative overhead and facilitate access control in spontaneous coalition environments. A requestor can access resources in a foreign domain by providing its identity information along with any delegations it may have to the security manager of that foreign domain, and the requestor's access to different resources in foreign domains can be seamless and flexible with the SCACA framework.

7.2 Future Work

The proposed SCACA framework is designed for users easily delegating and managing access rights to remote users in spontaneous coalition environments. A communication session role concept is proposed in the framework to group all communication session members altogether, as a result, all those session members have exact same available resources and related access rights to them. The author has discussed the advantages of using a session role in section 5.4.2, however, one disadvantage of doing so is that the access control loses granularity to some extent. In order to support giving session participants different privileges instead of assigning all of them one session role, the framework can be extended to be more general and user-centric, so that a delegator, a UA client can manually delegate different session members different access rights within an active communication session.

Although the SCACA framework is proposed for spontaneous coalition environments, it is not limited to that and can be further applied to formal coalition environments (see section 1.1), for an example, the formal across organizational conference scenario described in section 4.1. Recalling the D-TMAC model proposed in section 4.1, which extends the TMAC/C-TMAC models and provides dynamic and fine-grained access control for relatively formal coalition environments, the author believes that the incorporation of SCACA and D-TMAC can provide more flexible, dynamic and fine-grained coalition access control for formal coalition environments. The difference is that it will need more organizational administration involved in the access control system, such as team role management. A possible application would be the incorporation of a SIP conference server and the implementation of the SCACA and D-TMAC.

This research is based on the dRBAC model whose representation of delegation is not that universal and standardized. The work of bringing delegation into XACML is still an ongoing work, but the author believes the future direction of the SCACA framework would be using XACML policy language to represent both access control policies and delegations.

As discussed in section 1.1, the need for teleconference increased dramatically in the Post-September 11 world. More and more government entities, organizations and companies have a critical need to securely share information and interact across non-traditional organizational boundaries over dramatically increasing spontaneous coalitions. Many commercial companies have sensed this big change in the market and developed their enterprise instant messaging products, such as AOL Instant Messenger, Jabber, Akonix, etc. In the past year or two, Instant Messaging[9] has begun to take center stage for intra-organizational communications and even inter-organizational communications. For years, the major concern of organizations on Enterprise Instant Messaging is security, and the current major industrial efforts are to find ways to secure communication and data. The author believes that the future of Enterprise Instant Messaging is to focus on sharing more resources in coalitions and providing more capabilities for coalition access control. The models and framework proposed in the thesis can be a possible solution for this direction.

[9] Instant messaging is the act of instantly communicating between two or more people over a network such as the Internet. The typical capabilities offered by major IM client packages are: Basic text mode messaging between users; Voice over IP conversations; Video over IP sessions; File transfers; Application sharing; Group chat in 'chat rooms' similar to IRC.

References

[ABLP93] Martin Abadi, Michael Burrows, Butler Lampson, and Gordon Plotkin, "*A Calculus for Access Control in Distributed Systems*", ACM Transactions on Programming Languages and Systems, 15(4):706-734, Sept, 1993

[Adams04] Carlisle Adams, "*CSI5140A Network Security and Cryptography*", course note, University of Ottawa, Canada, Jan 2004, available at http://www.site.uottawa.ca/~cadams/courses/CSI5105.html

[Adams05] Carlisle Adams, "*Access Control: Current Solutions and New Challenges*", Invited Talk at Workshop on New Challenges for Access Control (NCAC), Ottawa, Canada, April 27, 2005

[Allen84] Thomas J. Allen, "*Managing the Flow of Technology*", MIT Press, 1984

[ANSI04] *"Role-based Access Control"*, American National Standard – ANSI INCITS 359-2004, 2004, available at http://www.incits.org/standards.htm, accessed January 2005

[BKFHDW04] P. Buxmann, W. Konig, M. Fricke, F. Hollich, L. Martin Diaz, S. Weber, "*Inter-organizational Cooperation with SAP Solutions: Design and Management of Supply Networks*", 2nd edition, ISBN 3-540-20075-4, Springer-Verlag, 2004

[Brooks97] R. Brooks, "*The Intelligent Room Project*", In Proceedings of the Second International Cognitive Technology Conference (CT'97), Aizu, Japan, August 1997

[BRTF02] Jan Borchers, Meredith Ringel, Joshua Tyler, and Armando Fox, "*Stanford Interactive Workspaces: A Framework for Physical and Graphical User Interface Prototyping*", IEEE Wireless Communications, pages 64–69, December 2002

[BS00] E. Barka and R. Sandhu, "*A Role-Based Delegation Model and Some Extensions*", National Information Systems Security Conference, 2000

[BSSW03] Stefan Berger, Henning Schulzrinne, Stylianos Sidiroglou, Xiaotao Wu, *"Ubiquitous Computing Using SIP"*, The 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'03), Monterey, CA, USA, June, 2003

[CACD04] Shiva Chetan, Jalal Al-Muhtadi, Roy Campbell, M. Dennis Mickunas, *"A Middleware for Enabling Personal Ubiquitous Spaces"*, System Support for Ubiquitous Computing Workshop at the Sixth Annual Conference on Ubiquitous Computing (UbiComp2004), Nottingham, England, September, 2004

[CFJ03] Harry Chen, Tim Finin, Anupam Joshi, *"Using OWL in a Pervasive Computing Broker"*, Workshop on Ontologies in Agent Systems, AAMAS-2003, July 2003

[CTWS02] Eve Cohen, Roshan K. Thomas, William Winsborough, Deborah Shands, "*Models for coalition-based access control (CBAC)*", the 7th ACM Symposium on Access Control Models and Technologies (SACMAT'02), pages 97–106, 2002

[D00] A.K. Dey, *"Providing Architectural Support for Building Context-Aware Applications"*, PhD thesis, Georgia Institute of Technology, Nov 2000

[DA00] A. K. Dey and G.D. Abowd, *"Towards a Better Understanding of Context and Context Awareness"*, Proceedings of Workshop on the What, Who, Where and How of Context-Awareness, April 2000

[ETR2A02] The European Telecommunications Resilience and Recovery Association, *"Telecom Outlook Report: Millennial Edition, Post-September 11 Update"*, ISBN: 1-931695-08-3, 2002

[Finin03] Tim Finin, "*The Why and How of Delegations in Distributed Security Systems*", available at http://www.csee.umbc.edu/~finin/papers/delegation-ccs.pdf, accessed January 2005

[FJHW00] A. Fox, B. Johanson, P. Hanrahan, and T. Winograd, *"Integrating Information Appliances into an Interactive Workspace"*, IEEE Computer Graphics and Applications, pp. 54-65., May, 2000

[FK92] David Ferraiolo, Richard Kuhn, *"Role-Based Access Control"*, 15th NIST-NCSC National Computer Security Conference, 1992

[FPPK02] E. Freudenthal, T. Pesin, L. Port, E. Keenan. "*dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments*", the 22nd International Conference on Distributed Computing Systems (ICDCS '02), pp. 411-420, IEEE, July 2002

[FZ94] George H. Forman, John Zahorjan, *"The Challenges of Mobile Computing"*,

[GD72] G.Graham, P.Denning, "*Protection-Principles and Practice*", In Proceedings of the AFIPS Spring Joint Computer Conference, Volume 40, pages 417-429, Atlantic City, New Jersey, USA, May 1972

[GMPT01] C. Georgiadis, I. Mavridis, G.Pangalos, and R.K. Thomas, "*Flexible TeaM-based Access Control using Contexts*", In Proceedings of the Sixth ACM Symposium on Access Control Models and Technologies, Chantilly, Virginia, May 2001

[GSSS02] David Garlan, Dan Siewiorek, Asim Smailagic, and Peter Steenkiste, "*Project Aura: Toward Distraction-Free Pervasive Computing*", IEEE Pervasive Computing, special issue on "Integrated Pervasive Computing Environments", April-June 2002

[HB01] Jeffrey Hightower and Gaetano Borriello. *"A Survey and Taxonomy of Location Sensing Systems for Ubiquitous Computing"*, UW CSE 01-08-03, University of Washington, Department of Computer Science and Engineering, Seattle, WA, August 2001

[HKSR97] Todd D. Hodes, R. H. Katz, Edouard Servan-Schreiber, and Lawrence A. Rowe, "*Composable Ad Hoc Mobile Services for Universal Interaction*", In 3rd ACM/IEEE International Conference on Mobile Computing, September 1997

[HRU76] M.H.Harrison, W.L.Ruzzo, and J.D.Ullman, "*Protection in operating systems*", Communications of the ACM, 19(8):461-471, 1976

[HU03] Hao Hu, "*Extent Scope of Bluetooth WPAN Application Profile Over IP Network Based on A Communication Session*", Master thesis, Dalhousie University, Mar. 2003

[IBIKS03] S. Ioannidis, S. M. Bellovin, J. Ioannidis, A. D. Keromytis, and J. M. Smith, "*Design and Implementation of Virtual Private Services*", In the Proc. of the IEEE Int. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Workshop on Enterprise Security, Special Session on Trust Management in Collaborative Global Computing, Linz, Austria, June 2003

[Johnston03] Alan Johnston, *"SIP: Understanding the Session Initiation Protocol, Second Edition"*, ISBN: 1580536557, Artech House Publishers, Nov. 2003

[KACM00] Apu Kapadia, Jalal Al-muhtadi, Roy Campbell, and Dennis Mickunas, "*I-RBAC2000: Secure Interoperability Using Dynamic Role Translation*", In Proceedings 1st International Conference on Internet Computing (IC'2000), April 2000

[KKAJFY02] Lalana Kagal, Vladimir Korolev, Sasikanth Avancha, Anupam Joshi, Timothy Finin, and Yelena Yesha, "*Highly Adaptable Infrastructure for Service Discovery and Management in Ubiquitous Computing*", in ACM Wireless Networks(WINET) Journal, 2002

[LABW92] Butler Lampson, Martn Abadi, Michael Burrows, and Edward Wobber, "*Authentication in Distributed Systems: Theory and Practice*", In ACM Transactions on Computer Systems, Nov, 1992

[Li01] Naizhi Li, *"Supporting User Mobility with Web-based Mobile Computing"*, Master Thesis, University of British Columbia, April, 2001

[LHL03] Yang Li, Jason Hong, James Landay, "*ContextMap: Modeling Scenes of the Real World for Context-Aware Computing*", Fifth International Conference on Ubiquitous Computing (UbiComp2003), Seattle, Washington, USA, 2003

[LJDH04] R. Liscano, A. Jost, A. Dersingh, H. Hu, "*Session-based Service Discovery in Peer-to-Peer Communications*", Canadian Conference on Electrical and Computer Engineering (CCECE'04), Niagara Falls, Ontario, May 2004

[PH03] Joon S. Park and Junseok Hwang, "*Role-based Access Control for Collaborative Enterprise in Peer-to-Peer Computing Environments*", The 8th ACM Symposium on Access Control Models and Technologies (SACMAT'03), Como, Italy, June 2003

[RFC2165] J. Veizades, E. Guttman, C. Perkins and S. Kaplan, *"SLP: Service Location Protocol"*, IETF, RFC 2165, June, 1997, available at http://www.ietf.org/rfc/rfc2165.txt

[RFC2327] M. Handley, V. Jacpbson, *"SDP: Session Description Protocol"*, IETF, RFC 2327, April, 1998, available at http://www.ietf.org/rfc/rfc2327.txt

[RFC2543] M. Handley, H. Schulzrinne, E. Schooler, J. Rosenberg, *"SIP: Session Initiation Protocol"*, IETF RFC2543, March 1999, available at http://www.ietf.org/rfc/rfc3261.txt

[RFC3261] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, *"SIP: Session Initiation Protocol"*, IETF RFC 3261, June 2002, available at http://www.ietf.org/rfc/rfc3261.txt

[RFC3264] J. Rosenberg and H. Schulzrinne, "*An Offer/Answer Model with the Session Description Protocol (SDP)*", IETF RFC 3264, June 2002

[RFC3303] P. Srisuresh, J. Kuthan, J. Rosenberg, A. Molitor, A. Rayhan, "*Middlebox Communication Architecture and Framework*", IETF RFC 3303, August 2002

[RL04] Wenbi Rao, Wei Li, *"Design of an Open and Secure Ubiquitous Computing System"*, Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI'04), Beijing, China, September 2004

[Schulzrinne96A], "*Simple Conference Invitation Protocol*", IETF MMUSIC WG Internet Draft, February 22, 1996, http://www.cs.columbia.edu/sip/drafts/mmusic/draft-ietf-mmusic-scip-00.txt

[Schulzrinne96B] Henning Schulzrinne, *"Personal Mobility for Multimedia Services in the Internet"*, European Workshop on Interactive Distributed Multimedia Systems and Services (IDMS'96), Berlin, Germany, March 4-6, 1996

[SKS04] Andreas Steffen, Daniel Kaufmann, Andreas Stricker, "*SIP Security*", E-Science und Grid, Ad-hoc-Netze, Medienintegration–18, DFN-Arbeitstagung über Kommunikationsnetze, Düsseldorf, GI-Edition - Lecture Notes in Informatics P-55, Bonner Köllen Verlag 2004

[SMB03] Debashis Saha, Amitava Mukherjee, Somprahash Bandyopadhyay, "*Networking Infrastructure for Pervasive Computing: Enabling Technologies & Systems*", ISBN: 1-4020-7249-X, Kluwer Academic Publishers, 2003

[Stajano02] Frank Stajano, "*Security for Ubiquitous Computing*", ISBN: 0470844930, John Wiley & Sons, Ltd, 2002

[Thomas97] R.K. Thomas, "*TeaM-based Access Control(TMAC): A Primitive for Applying Role-Based Access Controls in Collaborative Environments*", In Proceedings of the Second ACM Workshop on Role-based Access Control, Fairfax, Virginia, November 1997

[TS97] R.K. Thomas and R.Sandhu, "*Task-Based Authorization Controls (TBAC): A Family of Models for Active and Enterprise-Oriented Authorization Management*", In Proceedings of the IFIP WG 11.3 Workshop on Database Security, Lake Tahoe, California, August 1997

[Weiser91] Mark Weiser, *"The Computer for the Twenty-First Century"*, Scientific American, 265(3):94-104, September 1991, http://www.ubiq.com/hypertext/weiser/SciAmDraft3.html

[Weiser93] Mark Weiser, *"Some Computer Science Issues in Ubiquitous Computing"*, Communications ACM, 36(7):75–84, July 1993

[WKJLJ01] P. Werle, F. Kilander, M. Jonsson, P. Lönnqvist, and C. Jansson, *"A Ubiquitous Service Environment with Active Documents for Teamwork Support"*, Ubiquitous computing: international conference proceedings (Ubicomp 2001), vol. 2201 of Lecture notes in computer science, pp.139–155, Springer-Verlag

[WKS98] M. H. Willebeek-LeMair, K. G. Kumar and E. C. Snible, "*Bamba--Audio and Video Streaming over the Internet*", IBM Journal of Research and Development, Volume 42, Number 2, 1998

[XACML05] OASIS, "*XACML 2.0 Specification Set*", February 2005, available at http://www.oasis-open.org/committees/download.php/10577/XACML-2.0-CD-ALL.zip

Appendix A:

The following is an output of related SIP messages for the prototype demo described in chapter 6.

SIP INVITE request message:

INVITE sip:bob@137.122.90.138 SIP/2.0
Via: SIP/2.0/UDP 137.122.88.36
From: <sip:alice@137.122.88.36>;tag=lcwlvkr
To: <sip:bob@137.122.90.138>;tag=zqkvqtw
Call-ID: 911071378@137.122.88.36
CSeq: 1 INVITE
Expires: 7200
Subject: Hi
Content-Type: application/sdp
Contact: <sip:alice@137.122.88.36;transport=udp>
Content-Length: 340

v=0
o=alice 1125184048268 1125184048268 IN IP4 AIAS356
s=Voice Conversation
k=uri:c:\users\alice@137.122.88.36-CompanyA.member\deleg_000000_alice@137.122.88.36-CompanyA.member- CompanyA.xml
c=IN IP4 137.122.88.36
t=0 0
c=IN IP4 137.122.88.36
m=audio 40000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
c=IN IP4 137.122.88.36
m=application 8900 dRBAC

200OK response message to a SIP INVITE message:

SIP/2.0 200 OK
Via: SIP/2.0/UDP 137.122.88.36:1020;branch=z9hg4bk69e7f889f396821b0975f195bf2a913,SIP/2.0/UDP 137.122.88.36
From: <sip:alice@137.122.88.36>;tag=lcwlvkr
To: <sip:bob@137.122.90.138>;tag=zqkvqtw
Call-ID: 911071378@137.122.88.36
CSeq: 1 INVITE
Subject: Hi
Content-Type: application/sdp
Contact: <sip:bob@137.122.90.138;transport=udp>
Content-Length: 338

v=0
o=bob 1125184084312 1125184084312 IN IP4 AIAS317
s=Voice Conversation
k=uri:C:\Users\bob@137.122.90.138-CompanyB.member\deleg_000000_bob@137.122.90.138-CompanyB.member-CompanyB.xml

c=IN IP4 137.122.90.138
t=0 0
c=IN IP4 137.122.90.138
m=audio 20000 RTP/AVP 0
a=rtpmap:0 PCMU/8000
c=IN IP4 137.122.88.36
m=application 8700 dRBAC

SIP BYE request message:

BYE sip:bob@137.122.90.138 SIP/2.0
Via: SIP/2.0/UDP 137.122.88.36
From: <sip:alice@137.122.88.36>;tag=lcwlvkr
To: <sip:bob@137.122.90.138>;tag=zqkvqtw
Call-ID: 911071378@137.122.88.36
CSeq: 2 BYE
Content-Length: 0

200OK response message to a SIP BYE message:

SIP/2.0 200 OK
Via: SIP/2.0/UDP 137.122.88.36
From: <sip:alice@137.122.88.36>;tag=lcwlvkr
To: <sip:bob@137.122.90.138>;tag=zqkvqtw
Call-ID: 911071378@137.122.88.36
CSeq: 2 BYE
Content-Length: 0

Appendix B:

The following XML document contains a real delegation used in the prototype implementation:

[CompanyA.353791834@137.122.88.36 -> CompanyA.member w/[]]
alice@137.122.88.36

The entities are actually represented by their public keys.

```
<doc>
  <delegation>
    <!-- [CompanyA.353791834@137.122.88.36 -> CompanyA.member w/ []] alice@137.122.88.36 -->
    <role type="subject">
      <roleid>
        <name>CompanyA</name>

<publickey>10832757025834006580164401880936578689043917792598169452879190090552210343371662475531373433433275790506063247319832436011807711672654060524944456553164195595805528628553087885213589126247311959172213550111707894175067823770371741380559433766447160395563699697307459566878608789067243830250636626767816639140317</publickey>
      </roleid>
      <name>353791834@137.122.88.36</name>
      <assign>false</assign>
      <searchtag>
        <homeipaddr>137.122.88.36</homeipaddr>
        <homeport>9000</homeport>
        <roleid>
          <name>CompanyA</name>

<publickey>10832757025834006580164401880936578689043917792598169452879190090552210343371662475531373433433275790506063247319832436011807711672654060524944456553164195595805528628553087885213589126247311959172213550111707894175067823770371741380559433766447160395563699697307459566878608789067243830250636626767816639140317</publickey>
        </roleid>
        <ttl>5</ttl>
        <subflagtype>1</subflagtype>
        <objflagtype>0</objflagtype>
      </searchtag>
    </role>
    <role type="object">
      <roleid>
        <name>CompanyA</name>

<publickey>10832757025834006580164401880936578689043917792598169452879190090552210343371662475531373433433275790506063247319832436011807711672654060524944456553164195595805528628553087885213589126247311959172213550111707894175067823770371741380559433766447160395563699697307459566878608789067243830250636626767816639140317</publickey>
      </roleid>
      <name>member</name>
```

```
      <assign>false</assign>
      <searchtag>
        <homeipaddr>137.122.88.36</homeipaddr>
        <homeport>9000</homeport>
        <roleid>
          <name>CompanyA</name>

<publickey>10832757025834006580164401880936578689043917792598169452879190090552210343337
16624755313734334332757905060632473198324360118077116726540605249444565531641955958055
28628553087885213589126247311959172213550111707894175067823770371741380559433766447160
39556369969730745956687860878906724383025063662676781663914 03 17</publickey>
        </roleid>
        <ttl>5</ttl>
        <subflagtype>1</subflagtype>
        <objflagtype>0</objflagtype>
      </searchtag>
    </role>
    <attributeset/>
    <role type="signer">
      <roleid>
        <name>alice@137.122.88.36</name>

<publickey>14844222050212245735858768065537771136035372171162842948510588659392363846532
29331470667486185818789655524630349362397048539297131824314311957964762541576234962601
25930037939423336628460722552473340543689496125413824566913073204150874980753072117258
73230867161984965735131363258664831897242981541106975917124 3 17</publickey>
      </roleid>
      <name/>
      <assign>false</assign>
      <searchtag>
        <homeipaddr>137.122.88.36</homeipaddr>
        <homeport>9000</homeport>
        <roleid>
          <name>CompanyA</name>

<publickey>10832757025834006580164401880936578689043917792598169452879190090552210343337
16624755313734334332757905060632473198324360118077116726540605249444565531641955958055
28628553087885213589126247311959172213550111707894175067823770371741380559433766447160
39556369969730745956687860878906724383025063662676781663914 03 17</publickey>
        </roleid>
        <ttl>5</ttl>
        <subflagtype>1</subflagtype>
        <objflagtype>0</objflagtype>
      </searchtag>
    </role>
    <valid>-1</valid>
    <invalid>-1</invalid>
  </delegation>
</doc>
```

Appendix C:

The following is an example of XACML policy to control users' privileges for creating a new role in a domain.

```
<?xml version="1.0" encoding="UTF-8"?>
<Policy xmlns="urn:oasis:names:tc:xacml:1.0:policy"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    PolicyId="GeneratedPolicy"
    RuleCombiningAlgId="urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:ordered-permit-
overrides">

 <Description>
        This policy applies to users (example: alice@137.122.88.36)at domain
        137.122.88.36, who wanna its security domain AirNet to create a role
        (session role). The one Rule applies to the specific action of
  create a role in AirNet, for example AirNet.sessionRole888, but other
        Rules could be defined that handled
  other actions. In this case, only certain groups of people are
  allowed to commit. There is a final fall-through rule that always
  returns Deny.
 </Description>

 <Target>
  <Subjects>
   <Subject>
    <SubjectMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:rfc822Name-match">
     <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">137.122.88.36</AttributeValue>
      <SubjectAttributeDesignator DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name"
                    AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"/>
    </SubjectMatch>
   </Subject>
  </Subjects>
  <Resources>
   <Resource>
    <ResourceMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
     <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#anyURI">roleDatabase</AttributeValue>
      <ResourceAttributeDesignator DataType="http://www.w3.org/2001/XMLSchema#anyURI"
                    AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"/>
    </ResourceMatch>
   </Resource>
  </Resources>
  <Actions>
   <AnyAction/>
  </Actions>
 </Target>

 <Rule RuleId="CommitRule" Effect="Permit">
```

```
  <Target>
   <Subjects>
    <AnySubject/>
   </Subjects>
   <Resources>
    <AnyResource/>
   </Resources>
   <Actions>
    <Action>
     <ActionMatch MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
      <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">createRole</AttributeValue>
       <ActionAttributeDesignator DataType="http://www.w3.org/2001/XMLSchema#string"
                    AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"/>
     </ActionMatch>
    </Action>
   </Actions>
  </Target>

  <Condition FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
   <Apply FunctionId="urn:oasis:names:tc:xacml:1.0:function:string-one-and-only">
    <SubjectAttributeDesignator DataType="http://www.w3.org/2001/XMLSchema#string"
                    AttributeId="group"
                    Issuer="AirNet"/>
   </Apply>
   <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">AirNet.member</AttributeValue>
  </Condition>

 </Rule>

 <Rule RuleId="FinalRule" Effect="Deny"/>

</Policy>
```