A SIP-based Architecture model for Contextual Coalition Access Control for Ubiquitous Computing

Ramiro Liscano, Kaining Wang
School of Information Technology and Engineering, University of Ottawa
{rliscano, kwang}@site.uottawa.ca

Abstract

A significant deterrent to the ability to connect in a spontaneous manner in cross-enterprise 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 security domains. In this paper we present an approach to add contextual information to the distributed Role Based Access Control (dRBAC) model to support spontaneous coalition. The dRBAC model is a relatively new approach for coalition access control based on a delegation model but has been targeted towards pre-arranged delegations among distributed enterprises.A delegation architecture is presented that leverages SIP communication sessions to discover delegation security managers that can automatically exchange roles and delegations based on location and communication session contexts.

1. Introduction

In modern society, there are two different forms of inter-organizational coalitions. The first form of coalition is known as a *Formal Coalition* and the second is *Spontaneous Coalition*. Formal coalitions are process/task oriented and are targeted to medium or long-term periods of inter-organizational coalition [16]. Formal coalitions are a contractually or implicitly agreed collaboration between organizations. Some real examples are supply chain management, international joint projects, logistics service, etc. In general, the access control mechanisms for resources of the participating partners require common inter-organizational agreements and the overhead to set this up can be significant. Shared resources are fixed most of the time, and tightly related to a business process or a task.

Spontaneous coalition is driven primarily through communication sessions like Instant Messaging, Internet telephone calls, and multimedia conferences. Within a spontaneous coalition, organizational level agreements are not necessary. The relationships between participating parties are relatively simple and the resources for every single coalition are fairly dynamic.

The need for access control among individuals and organizations of different entities has increased significantly; however, enterprise collaboration is hampered greatly by the inability of persons associated with different enterprises to share resources securely within or across organizations. Each organization manages their own access rights to resources making it difficult to automatically coordinate effective collaboration session that require use of resources among the participants. Moreover, most successful coalitions are dynamic and ad hoc in nature with changes in conditions and trust relationships that are a result of new objectives and coalition memberships.

We also claim that the participating organizations in a spontaneous coalition are unwilling to rely on a 3rd party to administer trust relationships among each other. This is primarily due to the fact that it takes time to come to an agreement on which 3rd party entity to trust as well as to the time required to acquire any of the trust certificates. Spontaneous coalitions generally rely on the trust of their employees guided by some corporate safeguard security policies. For example, a corporation might allow the delegation of access rights to particular devices under particular contextual situations knowing that this access is temporary and other resources are protected.

The challenge is to develop a mechanism where the administration overhead is minimal and every enterprise will more likely have its own unique security policy language and semantics of enterprise roles.

We believe, along with others [15] that a delegation model for access rights is a significant step towards a

Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05)
0-7695-2375-7/05 $20.00 © 2005 IEEE

mechanism that requires minimal administration burden and strongly improve access control for coalition scenarios. Moreover, this delegation-based model can be improved to support ad hoc interactions by integrating context information as well as communication protocols like the IETF Session Initiation Protocol (SIP) [9].

This article is organized in the following manner. Section 2 presents work related to this article. Section 3 overviews the distributed Role Based Access Control (dRBAC) model that was by developed by E. Freudenthal, et al. [8] that supports coalition access control. Section 4 presents the contextual additions to the dRBAC model. Section 5 presents an architecture model for the distribution of roles and delegations based on the IETF Session Initiation Protocol (SIP) [9]. Section 6 presents an example. Section 7 gives discussion about SIP security and challenges. Finally Section 8 concludes by discussing advantages and limitations to our approach.

2. Related Work

The predominant model for advanced access control is Role-Based Access Control (RBAC) introduced in 1992 by Ferraiolo and Kuhn [1] and now an American National Standard – ANSI INCITS 359-2004 [2]. RBAC has some limitations in its use in ad hoc collaborative 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. However, it appears that in general many existing approaches for coalition access control [3][4][5][6][7] 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 is reduced to a manageable number of roles. 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 than to change all the access policies when changes occur in an organization.

Coalition access control is a relatively new area of research. Most approaches presented require access control agreements [6] to be put in place between partners or the adoption of common access policy ontology [7] between organizations. As such the administration overhead is relatively high in these approaches. Nevertheless E. Freudenthal, et al. [8] introduced delegation for coalition access control by developing the dRBAC system to support decentralized delegation chains maintained across multiple domains. Delegation is a mechanism where a user in a role delegates his role membership to another user or another role. As such the recipient of the delegation 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 recipient of the delegation takes on the persona of the delegator.

Abadi et al [12, 13] defined delegation as a mechanism for authentication, access control and trust, however no constraints can be applied to the delegation. E. Barka and R. Sandhu modeled role to role delegation in [14], and introduced a role based delegation model called RBDM0. They discussed the revocation of delegations using a timeout but did not support event-driven revocation. Finin [15] presented an approach where conditions are attached to the delegation, but the conditions are limited and it does not support contextual information.

3. The dRBAC Model

The dRBAC model implements access control for a multi-party coalition application through delegation. The format of 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. Local Wallets are used to store a collection of delegations. 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" 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 algebraically.

[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.

4. Adding a Contextual Model to dRBAC

In order to minimize the need for a common contextual ontology among coalition enterprises we restrict the contextual constraints to be applied only to the organization which owns the resource. This is a similar approach to that taken by dRBAC in its use of constraint attributes. Even though it is possible for each enterprise to use its own contextual model we present a contextual model as a base mark.

There are many works in representing semantics for context-aware computing. Based on the works in [19] [20], 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 we simply use the OO methodology of classes, subclasses, and instance notions to clearly represent hierarchical context information that will let us capture location and activity in such a manner that we can specify a general location or activity and any sub-ordinates that will apply for the delegation proof. A context hierarchy indicates relations between atomic context elements as shown in Table 1.

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 1. 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, we may generally define the context condition, which will be applied to the delegator, 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.SITE4004

will match the class types as TRUE.

Then we add context information as attached conditions for a delegation for dRBAC. We propose the format as follows:

[Subject → Object] (ContextConstraints*) Issuer; where ContextConditions = (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 ContextConstraints. These contextual constraints can also be applied on a right-of-assignment delegation Subject → Object'. Since the contextual constraints only apply to the issuer's domain there is no need to share a common ontology among users.

Context conditions follow 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's take an example of a member of CompanyB being delegated a role at CompanyA.

[CompanyB.member → CompanyA.roomAdmin] (CompanyA.research Activity == Communication_Session) Bob;

In this particular example Bob is the issuer of this delegation. For the delegation to be true Bob must be in the CompanyA.research role and 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.

5. An architecture model for Ad hoc Coalition Access Control

The original use case scenarios for dRBAC require that the roles of the coalition members to be a-priori assigned and the delegations to have been previously distributed. For ad hoc situations, where the parties are not known a-priori, it is necessary to consider how delegations and roles might be shared among participants that have come together. We are interested in how the dRBAC model can be leveraged to take into account particular contextual situations.

The contextual event that we have chosen as an example is a communication session and the strongest argument for doing this is the fact that most spontaneous coalitions occur are over communication sessions. With the assistance of an underlying communication session, spontaneous coalition applications are able to commence and end within a well defined event or period of time. In that way the access period to the enterprises network is well defined. Moreover the communication session can be leveraged as a bridge to bring the trust that exists in real world situations into the access control mechanism. Persons generally have some degree of trust in the other parties that are part of the communication session.

SIP, the Session Initiation Protocol, is becoming a dominating unified standard in both IP and Telecommunication communities. Alan Johnston [17] pointed out that the rapid acceptance of SIP has less to do with its powerful call control functionality but on the ability of SIP, and its extensions (e.g. SIP SIMPLE [21]), to enable and support ubiquitous computing through Instant Messaging, Internet telephone calls, multimedia conferences, etc. Internet communication session protocols like SIP are being integrated with firewall protocols to support access control to private networks. Session protocols allow for the negotiation of addressing knowledge of devices at the end points. This feature is leveraged in this article to determine the addresses of the delegation security managers at the end points. Later in section 6 we present a cross enterprise scenario where the context leveraged is the commencement of a communication session between 2 meeting rooms. The advantage of using this context is that one can leverage SIP in order to transform the dRBAC model to support spontaneous communications. In order to do this three processes in the dRBAC model must be automated. 1) Knowledge of the addresses of the wallet keepers at each end point must be exchanged at the time that the participants enter into a communication session, 2) the end points need to exchange the roles that apply to the participants of the session, and 3) each end point needs to be able to exchange the delegations.

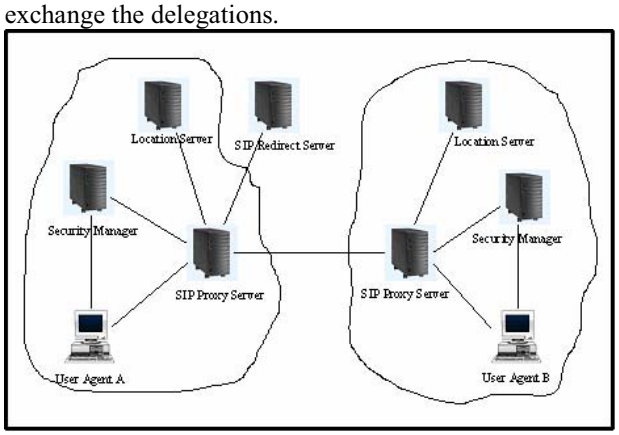

Figure 1. SIP Session-based System Architecture

In our model we use a session as a commencement point and force the initial communication to occur between a set of SIP Proxies as shown in figure 1. The session is used to get the address of a delegation security manager server that contains the delegation wallet capability as described in dRBAC. The delegation security manager is also used to negotiate for any roles as well as transfer the delegations. The communication session is also used as a context to refine and facilitate collation access control.

5.1. Acquiring the delegation security manager's addresses

To acquire the addresses of the delegation security managers we propose defining a new SIP Session Description Protocol (SDP) [10] 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. It is generally used to negotiate real time streaming codec parameters. Without introducing too many details of SDP 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 stopt-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 we propose the use of "Delegation Manager". 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 1660 TCP 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 authors are not aware of any protocols that are used for the negotiation of delegation or security policies. This approach has been proposed and implemented by the authors for the negotiation of service directories [11]. It is a relatively powerful mechanism by which some minimal exchange of capabilities can be associated with a session. Figure 2 is a message sequence chart of the SIP dRBAC session negotiation.

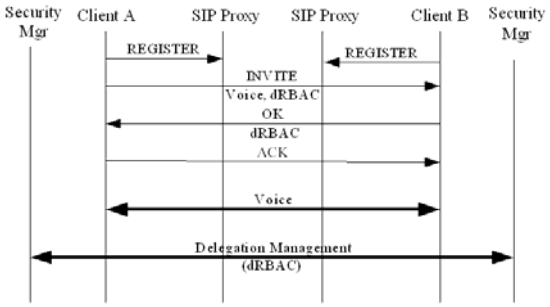

Figure 2. Sequence of a Communication Session Setup

5.2. Negotiating the user roles

At this point the delegation security managers can establish a connection in order to negotiate roles and delegations. The actual exchange of roles is not that difficult as long as a protocol between the end points is defined. The challenge is primarily in determining and managing properly the roles that can be exchanged. Even though the RBAC model 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 their role is performed automatically and by an authentication server. It is more effective to keep the same interaction model since user's have a difficult time choosing appropriate roles.

We propose the use of a session role, which is determined at run time and created when the communication session is established. This can easily be done by considering the role as an argument to the DRBAC SDP instruction. 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 their roles to this run time session role, so other session participants are able to access resources across the enterprises. There are three advantages of using a session role instead of delegating between private roles. First of all, it is more secure for the receiver of the delegation to use other roles than those used internally for his/her domain. 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. Secondly using a session role can dramatically reduce the amount of delegations to which one must give access to. Each delegator just delegates its roles to one session role instead of delegating its roles to users or roles from every organization respectively. This will make the coalition access control system more scalable. 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. If the delegation was to a real role the user that had left the session would not necessarily lose their role and therefore could still get access to the resource from the other organization. This is an added security measure since the delegations should have been revoked when the user left the session. What is not feasible is to use this session role as the delegator's roles since these roles need to match the pre-defined role-based access control rules in the delegator's domain.

When a user wishes to delegate a role, it is important to check the delegation rights by the user before delegation actually takes place. It is known that users become easily frustrated with services that do not perform as expected. If one of the users was to delegate to another, a role that could not be proven to be true the users of the service would quickly become frustrated with the inability to connect to those shared services. In order to determine which roles a user can delegate it is necessary to query the local wallet for right-of-assignment delegations based on a Subject equal to the delegator's entity name. This query would appear

in the following manner ?[Subject → *'] (Context OP value) where Subject = Context = Location | Activity, and Context = currentUserContext. In our situation we are leveraging a communication session event to be the contextual condition.

The use of a session role allows us to manage both single and multi-party collaborative scenarios. In a single party collaborative scenario 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 Bob is the delegator and Alice is the person receiving the delegation, Alice will be delegated the session role and Bob can delegate access to a room in Company A by delegating a roomAdmin role in the following manner.

alice@companyB → sessionRole : CompanyB.securityMgr

sessionRole → CompanyA.roomAdmin : Bob

For a multi-party collaborative scenario each person at the calling end will be delegated the session role. Therefore for Alice, John, and Carl located at CompanyB three delegations would be created at CompanyB's wallet.

Alice → sessionRole; John → sessionRole; Carl → sessionRole;

sessionRole → CompanyA.roomAdmin : Bob

6. Contextual dRBAC Scenario

A tele-conference will be held between two meeting rooms, roomA and roomB, located respectively in two different companies, CompanyA and CompanyB. This tele-conference could happen within several companies, not only limited to two. Bob is an employee of companyA, and he is the initiator of this conference. Alice is a member of CompanyB as well as one of the participants attending the conference and is located in her company's meeting roomB. Our 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 resources in meeting roomA of companyA are only permitted during the conference and to those members participating in the conference call, i.e. the communication session. As such once the meeting ends, all the access privileges are gone. In the following we illustrate how our context based approaches achieve this goal and facilitate the process.

6.1. Setup of a Communication Session

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

Bob, the communication session initiator, sends an SIP INVITE message to one or more users in his own organization: companyA, and outside organizations like companyB. The INVITE message uses SIP URLs for addressing, and carries a dRBAC SDP message as described in section **Error! Reference source not found.** to the callees. These SDP messages also include a run time generated session role for this particular communication session in the SDP message (for example: sessionRole050211).

Upon receiving the INVITE message, a callee can either accept or reject the invitation. After accepting the invitation message and sending back its own security manager's address information the delegation security managers will connect in order to exchange the session role and delegations. All the participants, no matter where they are, belong to this role only when the communication session is active.

6.2. Access Control in Communication Session

Bob, the session initiator, creates or activates delegations related to this conference call. For example, Bob 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 and location == MeetingRoom.SITE4004

The roles that Bob can delegate can be manually selected by Bob from a prompted interface or be automatically done based on predefined policies. Moreover, other participants, like Alice, 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 we summarize the steps illustrated by Figure 3, in which Alice, a member of companyB, request an access to the resources located in the meeting room in companyA.

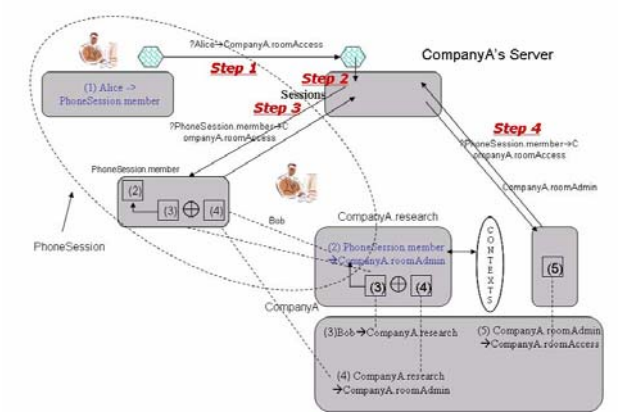

Figure 3. Contextual dRBAC Scenario

(1) [Alice → PhoneSession.SessionID1234.member]:

The starting point is to validate Alice's access to CompanyA.roomAccess. To authorize access, the CompanyA security manager must discover a proof for Alice → CompanyA.roomAccess. The first delegation is that of Alice to a PhoneSession.SessionID1234.member. Company A's security manager can validate Alice → PhoneSession.SessionID1234.member by itself. It issues a subject query to obtain all proofs of the form [PhoneSession.SessionID1234.member → *]. In response to this query, it discovers delegation (2) defining a relationship between the roles PhoneSession.SessionID1234.member and CompanyA.roomAdmin.

(2) [PhoneSession.SessionID1234.member → CompanyA.roomAdmin] Bob (activity == PhoneSession.SessionID1234 and location == MeetingRoom.SITE4004):

The security manager discovers the delegation has contextual constraints applied to the delegator Bob. It first retrieves the real time contextual values of Bob from a context agent or a SIP location server and checks the Boolean 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 rejected. If the condition is True, then the security manager further discovers that delegation (2) is a third-party delegation (right of assignment delegation) since Bob 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) [Bob → CompanyA.research] CompanyA: Bob 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 Alice 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 [Alice → CompanyA.roomAccess] is complete.

6.3. Handling User Exit in the middle of Communication Session

For a conference call, it is common that some participants may quit the conference in the middle of the communication session. For example, Alice quits the session before the whole session ends. Two issues need to be considered here. First Alice needs to take back all the delegations that she gave out to the session role when she joined the session and all delegations that involve Alice also need to be revoked so that she will not be permitted access to the resources in the other organizations.

For the first issue, since we have context constraints on the delegator, that is:

activity == PhoneSession.SessionIDxxxx and location == MeetingRoom.xxxx

Those delegations will become invalid immediately when Alice's activity is changed and is no longer in a communication session. Context change triggers the change of delegation status. For the second issue, Alice will not hold the session role right after she sends a SIP BYE message to all other parties and leaves the session, so she will not have the access rights that other participants delegate to the session role.

6.4. Termination of Communication Session

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

is alive. In addition, a user can manually revoke his/her delegations anytime in an ongoing session.

7. SIP Security and Challenges

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) [18] and IPsec can secure the communication between coalition partners in a public domain. Incorporating SIP into the access control also introduces new challenges since a user's various behaviors at the 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. We are not clear whether this should be allowed or not because this would somewhat violate the service providers' security requirements, which may be only given access to those original session participants. We need to expand our work to handle the interactions between the SIP signaling layer and the access control layer.

8. Conclusion and Future Work

We discovered that the dRBAC model places a significant trust on the delegator such that there are no restrictions on "who" the delegation is for. We are trying to place some form of acceptable context-based restrictions on the delegation because we feel that the dRBAC model is too restrictive for ad hoc scenarios. In our architecture, access rights are given out through delegations within a communication session, in which every party may know each other a-priori. For example, Bob may know Alice to some extent before Bob calls Alice.

We presented a delegation architecture that leverages communication session as context to facilitate coalition access control, and propose an extension to dRBAC model by applying context information conditions into delegations to provide more flexible and fine-grained access control. A session role is also proposed in the paper for easily managing the cross enterprise collaboration. An example is given, where a communication session can be used as the contextual information for dRBAC.

The central idea of this paper is to use context-based delegation to minimize the amount of administrative overhead and facilitate access control in coalition and pervasive computing environment. A requestor can access resource in foreign domain by providing its identity information along with any delegations it may have to the delegation security manager of that foreign domain, and the requestor's access to different resources in foreign domains can be seamless and flexible with our context-based delegation.

We are considering the use of an agent platform like FIPA to implement the delegation security managers and security negotiation protocol.

9. References

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

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

[3] 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.

[4] 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

[5] 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

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

[7] Joon S. Park and Junseok Hwang, "Role-based Access Control for Collaborative Enterprise in Peer-to-Peer Computing Environments", SACMAT'03, Como, Italy, June 2003.

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

[9] J. Rosenberg, H. Schulzrinne, G. Camarillo, SIP: Session Initiation Protocol, IETF, RFC 3261, June 2002.

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

[11] 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.

[12] Butler Lampson, Martn Abadi, Michael Burrows, and Edward Wobber. Authentication in Distributed Systems:

Theory and Practice. In ACM Transactions on Computer Systems, Nov, 1992.
[13] 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
[14] E. Barka and R. Sandhu, A Role-Based Delegation Model and Some Extensions, National Information Systems Security Conference, 2000
[15] Tim Finin, The why and how of delegations in distributed security systems, http://www.csee.umbc.edu/~finin/papers/delegation-ccs.pdf, accessed January 2005.
[16] 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
[17] Alan Johnston, "SIP: Understanding the Session Initiation Protocol, Second Edition", ISBN: 1580536557, Artech House Publishers, Nov. 2003
[18] 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, pp. 397-410.
[19] 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
[20] Harry Chen, Tim Finin, Anupam Joshi, Using OWL in a Pervasive Computing Broker, Workshop on Ontologies in Agent Systems, AAMAS-2003, July 2003.
[21] SIP for Instant Messaging and Presence Leveraging Extensions (SIMPLE), Available at http://www.ietf.org/html.charters/simple-charter.html