Integrated Framework for Authentication and Access Control in Peer to Peer Groups
Second Annual Progress Seminar Report

Submitted by

Madhumita Chatterjee
Roll No: 04429802

under the supervision of

Prof. M. Bernard and Prof. G. Siva Kumar

Kanwal Rekhi Institute of Information Technology
Indian Institute of Technology

January 24, 2007
Acknowledgments

I take this opportunity to thank Prof M. Bernard and Prof. G. Siva Kumar for their invaluable guidance, encouragement and support.

Madhumita Chatterjee
Abstract

Collaborative applications like video/audio conferencing, IP telephony, file sharing, collaborative work spaces, and multi-user games, having varied security requirements, require a secure and reliable group communication system to provide co-ordination among the processes. Peer to peer computing allows users to interact with each other and find and share resources without requiring a centralized server. This gives rise to the need of a secure group layer which integrates authentication, admission control, authorization, fine-grained access control and key management. Currently there are some integrated solutions for secure group communication over wide area networks, but very few allow for the development of trust in a dynamic fashion. We propose a hybrid access control model for peers in groups requiring multi-level access control. Our approach is adaptive, flexible and scalable and integrates aspects of incremental trust, along with role-based and policy based access control. We also propose a generic integrated framework for secure group communication in decentralized peer to peer groups.
Contents

1 Introduction 1

2 Scenario 3
   2.1 Scenario 1: Multi Player Gaming 3
   2.2 Scenario 2: Audio/Video conference collaboration 4
   2.3 Scenario 3: Secure Messaging 4
   2.4 Scenario 4: Collaborative news-groups 4

3 Related Work 5
   3.1 Access Control 5
      3.1.1 Identity Based access control (IBAC) 6
      3.1.2 Role based access control (RBAC) 6
      3.1.3 Authenticated Role based access control (ARBAC) 7
      3.1.4 Attribute based access control (ABAC) 7
      3.1.5 Trust Based Access Control (TBAC) 8
      3.1.6 Hierarchial Access Control in Secure Group Communication 9
      3.1.7 Dynamic Access control for multi-privileged group communications 10
   3.2 Secure Key establishment 10
   3.3 Summary 10

4 Proposed Integrated Framework 11
   4.1 Proposed Architecture 11
   4.2 Model Overview 12
   4.3 Protocol for New Member Join 14
      4.3.1 Group Initialization and Advertisement 14
      4.3.2 Admission Request 15
      4.3.3 Authentication 15
      4.3.4 Voting and Authorization 15
      4.3.5 Key Management 16
   4.4 Protocol for Existing Member Level Updation 16
      4.4.1 Request for Updation 17
      4.4.2 Access Control 17
      4.4.3 Key management 18
   4.5 Per-Session Authentication of Previous Member 18
   4.6 Dynamic Leaving of Members 19
   4.7 Summary 19


<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Implementation of Secure Multi-Chat</td>
<td>20</td>
</tr>
<tr>
<td>5.1</td>
<td>JXTA-A Peer to Peer Framework</td>
<td>20</td>
</tr>
<tr>
<td>5.1.1</td>
<td>Policy language</td>
<td>22</td>
</tr>
<tr>
<td>6</td>
<td>Conclusion</td>
<td>23</td>
</tr>
<tr>
<td>6.1</td>
<td>Future Scope</td>
<td>23</td>
</tr>
</tbody>
</table>
List of Figures

3.1 Role Based Access Control .............................................. 7
4.1 Integrated Framework ................................................. 12
4.2 New Member Join ...................................................... 14
4.3 Member Level Updation ............................................... 17
5.1 JXTA Framework ...................................................... 21
6.1 Str Protocol ........................................................... 24
6.2 Tree Group Diffie Hellman ........................................... 25
6.3 Group creation ........................................................ 27
6.4 Voting Request ......................................................... 27
Chapter 1

Introduction

Web based collaborative groups are becoming increasingly popular as peers with common interests form a network among themselves and automatically tend to create interest groups among each other, called communities. Some examples of intrinsic communities being formed, are Yahoo Groups [16] and Google Groups [17]. Applications like file-sharing, online gaming, video/audio conferencing, collaborative work-space, virtual meetings, distance learning environments, discussion forums and board rooms are examples of applications that are organized as peer groups [19] [20] [21]. The group is governed by a set of rules that describe the conditions required to be part of the group.

Security in such dynamic collaborative groups is governed by membership control, authentication, access control and key management. Membership control and authentication is required to allow only authorized members to join the group. In the dynamic scenario where peers are constantly leaving and joining this becomes a difficult task. Another major challenge is access control since decisions need to be made based on collaborations from all peers and feedback mechanisms based on trust. A number of authorization systems have been developed to provide access control to shared resources in distributed environments e.g PRIMA [22], CAS [15] and Akenti [23]. These systems do not however address purely decentralized collaborations. Secure Spread [24], Antigone project [24], and Secure group layer SGL [26] are examples of secure group communication systems. However they are not fully distributed and do not have a decentralized and dynamic access control mechanism. Thus they are not typically suited for peer to peer networks.

In the decentralized environment, peers in self-organising groups should have the right to collaboratively modify the access control policies governing their levels and the levels below them. Current solutions do not allow dynamic change of access levels or policies, with incremental building of trust during the communication. Nor do they allow dynamic authentication of participants who were previously part of the group and later wish to rejoin.

We propose an adaptive access control framework which integrates role based access control, policy based access control, trust based access control, and multi-level access control.

Keeping in mind the diverse peer to peer applications with varied security requirements there is a need for an integrated framework for secure communication in dynamic groups with no dependance on a centralized server. We propose such an integrated framework, which has a flexible, dynamic and totally decentralized access control mechanism, integrated with authentication and dynamic rekeying.

The rest of the report is organized as follows. In chapter 2 we identify some peer to
peer group scenario which demand multi-level dynamic access control. In chapter 3 we look at the previous work done in related areas of secure group communication and access control. Chapter 4 gives the details of our proposed model. A small part of this model has been implemented using JXTA as the base framework, and we discuss the details in chapter 5. The scope for future work is discussed in chapter 6.
Chapter 2

Scenario

Peer to peer applications where the peers organise themselves as self-organised groups demand a dynamic and fine-grained access control framework. We look at some scenario where dynamic multi-level access control would be required.

2.1 Scenario 1: Multi Player Gaming:

Massively Multiplayer Online Games MMOGs are computer games capable of supporting thousands of players simultaneously and are typically played in a giant persistent world. Main aspects of game-play involve grouping with fellow players to perform certain tasks e.g killing monsters or attacking the enemy etc. Virtually most multi-player gaming today relies on a server based paradigm. Players connect to one of at least 25 geographically dispersed server clusters and communicate with the ”persistent world” through the server. There are a number of issues surrounding this server based approach, the most obvious being the dependancy on a server [12].

To be able to handle so many players simultaneously huge amounts of computing power and bandwidth are required at the server side making MMOGs cost intensive. One of the major problems with server based approach for multi-player gaming is that games become obsolete the moment a server fails. The use of peer-to-peer approach could improve the multi-player gaming situation. Since a P2P network does not have a central point of failure, so a game cannot be made useless by the failure of a single computer.

Group implementation Issues and Challenges

Peer to peer approach in the multi-player gaming scenario would warrant the need for secure admission and access control. A new player wishing to join an existing group should not be allowed to join if he is at a rating level lower than that of the players in the group. If the group consists of players playing at multiple levels then the rating of an existing player should increase based on his performance and number of wins. Thus a player who is admitted into the group at a lower hierarchial level could later be elevated because of incremental trust thrust upon him by group fellow members because of his performance. If on the other hand a peer cheats in a game he may be deprived of his ability to join another game, by revocation of certificate and he may be forcibly ejected from the group.
2.2 Scenario 2: Audio/Video conference collaboration

Collaborative applications, need to allow for the building of trust between individuals through interactions within the collaboration environment. In an application like a Virtual meeting or an Online Board Discussion, you could be at a conference listening to the presentation of a speaker, which is based on a topic which is a common interest problem that you and your collaborators are dealing with. Ideally you should be able to bring this speaker into your other collaboration on the spot without the others voting for his admission or verifying his credentials. This calls for a partial access to the group, where the new member is treated as a guest and can only listen onto the conversation in a controlled fashion. This scenario is currently not supported by prevalent group communication systems.

In a Virtual classroom scenario (e.g. online university), geographically dispersed students can attend lectures in real time, interactively. The different roles could be Professor, student, T.A, Administrator, course designer and so on. Here authentication of users would be necessary so that peers can determine who they are interacting with. At the same time authorization mechanisms are needed to limit the groups of users who can participate in online meetings, as well as restrict the access rights of students etc.

2.3 Scenario 3: Secure Messaging

In collaborative environments some resources such as a chat room, may require a group access decision. In the case of a chat room owned by a number of users, every one of these users would want to contribute to the authorization decisions that allow other users to access that room. Whereas in multi-chat application, users may want to chat one-to-one in a secure manner as well as one-to-many. The access rights of users in such an application may vary, i.e some user may be able to receive messages from all other users while some may receive messages only from restricted people in the group. There may even be a case, like in a military setting where some user can only receive messages but not send any.

2.4 Scenario 4: Collaborative news-groups

Consider an online self-organising newsgroup, where authorised users can post articles. Different roles assigned maybe editors, reviewers, moderators, admin etc. Articles may be rated by reviewers and an editor may get updated to the level of a reviewer or a moderator with sufficient number of good postings.
Chapter 3

Related Work

Security in collaborative groups is addressed by group membership, authentication, access control, and group key management. Several efforts have been made towards securing group communication systems, for e.g Secure Spread [25], Antigone project [24] and Secure group layer SGL [26]. Secure Spread [25] is a secure group communication system that uses a fully distributed group key generation protocol, but it does not provide any authentication or group access control mechanisms and focusses primarily on LAN and interconnected LAN environments.

Antigone project [24] includes a flexible framework for secure group communication and utilizes a centralized member admission approach which is controlled by a session leader SL who interacts with an on-line trusted third party TTP in order to admit a new member. The TTP shares a symmetric key both with the SL and every potential new member. However Antigone is not designed for peer-to-peer networks.

Secure group layer SGL [26] is a secure group communication system aimed at WAN environments. It bundles a reliable group communication system, a group authorization and access control mechanism and a group key agreement protocol to provide a comprehensive and practical secure group communication platform. However the access control mechanism is not dynamic or scalable.

Some frameworks are focussed on peer to peer applications. Kim et al [1] [2] proposed an admission control framework which revolves around two basic elements, viz: a group charter which has well defined admission policies and a group authority which is an entity that can certify group admission. They proposed three types of admission policies, including Admission via Access Control Lists, Admission by a centralized authority GAUTH, and admission by group members. A collaborative decision is taken by members and admission to a new member is granted after issuance of a group membership certificate under a multi-voting scheme. However their scheme lacks the attributes of peers and cannot simplify authorization in collaborative environments. Further access control has not been considered, all members have equal access rights, and neither does it integrate admission control with group key agreement.

In the following section we give brief details about the different models of access control that have been applied for group communication.

3.1 Access Control

Access control models are used to decide on the ways in which the availability of resources in a system are managed. There can be two types of scenarios, Static Access Control...
and Dynamic Access Control. In former case information about all group members is known in advance. Thus management of access policies is pretty easy and can be done using ACLs (Access control lists, ACL is a table that specifies which access rights each user has to a particular system resource and service). However in a dynamic group the peers can join and leave at any time. It should be possible therefore to specify and change policies at run time depending on the environment or collaboration dynamics.

3.1.1 Identity Based access control (IBAC)

This employs traditional access control schemes such as access control lists or ACLs. Authorization decisions are based on the identity of the requestor. Identity based credentials can be further classified into encryption based and non-encryption based. An encryption based identity credential is a piece of information produced and cryptographically signed using the key possessed by the supplicant in order to verify its possession of the key, and hence prove its identity.

One form of non-encryption based identity credential is information that is hashed using a one-way key-based hash function and the key possessed by the supplicant. In order to verify the supplicant’s identity, the authenticator must possess the same key (symmetric key) and the hashed information as the supplicant in order to re-generate the hash value and verify the claimed identity of the supplicant. However in de-centralized environments, the resource owner and requestor are often unknown to each other, making access control based on identity very ineffective or expensive to maintain.

3.1.2 Role based access control (RBAC)

A RBAC system has two phases in assigning a privilege to a user, first the user is assigned roles and then the roles are checked against the requested operation i.e users are mapped to roles and roles are mapped to permissions. When a new member wishes to join the group he/she will get a list of available roles that the application (group) supports. The new member will request for a role (may be highest available) passing his identity to other peers. If new member’s admission request is rejected, he can attempt to join in a lower role. Access rights are grouped by role name, and the use of resources is restricted to individuals authorized to have associated role. Thus permissions are associated with roles not with individual peers, refer figure 3.1.

The authors in [3] propose a fine-grained access control framework for group communication systems where they allow an application to define specific policies while the enforcement is performed in an efficient manner by the GCS. Their focus is on group communication systems built around a client-server architecture. The basic approach they follow is that for a group there is a set of basic operations that can be performed, by principals (entities) based on their role in a given context. Their framework combines role based access control mechanisms with environment parameters such as time, IP address etc, to provide policy support for a wide range of applications with different requirements. They identify the set of all possible group operations that can be controlled and define the group policy as a mapping between roles and operations using context as constraints. Thus applications defining specific policies are translated to this set of basic operations that the GCS is aware of and can enforce access control on.
3.1.3 Authenticated Role based access control (ARBAC)

Systems that require a stronger authentication can replace the login process of username, password with a public key certificate that is used to identify the accessed user. Certificates contain a public key, and properties of the owner of the corresponding secret key; in this case, the relevant property is the identity of the owner.

A trusted Certification Authority (CA) digitally signs each certificate, binding the attributes with the owner- specifically, providing the identity of the owner of the private key. The login process is replaced by an authentication protocol such as SSL, which verifies that the user has the secret key. This is the mechanism used, e.g., by certificate-based authentication by current browsers and servers and is based on the assumption that the users are known to the system in advance.

3.1.4 Attribute based access control (ABAC)

In this case the access permission are determined based on authenticated attributes of user. Attributes are normally a function of the user’s identity and its characteristic. So a policy rule which decides whether a peer p can access some resource r is a function of peer and resource attributes. A rule will therefore look like:

\[
\text{Rule}_X : \text{can}_\text{access}(s, r) \leftarrow f(\text{ATTR}(s), \text{ATTR}(r))
\]  

(3.1)

In case of RBAC, fine grained access control polices often involve multiple subject and attributes. As the number of attributes involved increases, number of roles and permission needed to encode these attribute increases exponentially. RBAC might need some centralized server to manage user to role and role to permission assignment. Peer groups share the properties of peer-to-peer overlay networks, including full de-centralization, symmetric abilities and dynamism, making security problems more complicated.

Yu Zhang et al in their work [4] propose a fine-grained attribute based access control framework for peer-to-peer systems. They extend role-based trust management language (RT) [28] to satisfy security requirements of peer groups. They define two kinds of roles:
group role and application role. Group role peers are predefined by peer groups and application role peers are defined according to different collaborations. They propose a distributed delegation authorization mechanism where multiple authorities could exist to grant peer group membership. Their access control policy defines the relations of roles and permissions including six credentials from RT and describes admission and removal policies of roles. Trust management is used as the authentication method. Joint authorization is done by voting schemes as in framework proposed by Kim et al [1]. The actual model consists of the following steps:

- **Group Initialization:** The group authority peer, creates a secure peer group and advertises it on the network. The advertisement contains the access policy of peer groups along with other parameters like group name, voting type etc.

- **Join Request:** A new peer wishing to join a group, searches for the advertisement, provides his related credential to an authority peer along with a request for issuance of a peer group certificate with its desired privileges.

- **Authorization of Request:** The authority peer first verifies the signature and then propagates the request to call a vote. Multiple peers authenticate the requestor’s attributes and reply with a signed message to approve or reject the authorization request.

- **Certificate Issuance:** Once enough votes are collected the Group Authority verifies the votes and decides whether to accept the new node as a member. If the requestor is qualified, the GA peer issues a peer group certificate to it and updates related peer group information.

3.1.5 Trust Based Access Control (TBAC)

Distributed establishments typically involve peers that do not know each other and have never met before. This brings in the concept of risk when peers perform transactions without knowing the reputation of those whom they are interacting with. This has precipitated work on trust and reputation mechanisms in peer-to-peer networks [5] [7] [8] [9] [6]. Trust is a measure of how much a peer is willing to transact with another peer. Reputation [10] is one specific way of establishing trust.

**Reputation based systems**

Such systems establish trust among members by using feedback from peers. One well-known system is the rating scheme used by eBay an on-line auction site [18]. Ebay has a reputation system where buyers and sellers rate each other after each transaction and the overall reputation is the sum of the ratings over last six months. However the system relies on a centralized system to store and manage the ratings. The Ebay system can be easily compromised because the seller can gain reputation very fast by fulfilling small orders and defaulting on a large order.

Kazaa [19] is a peer to peer file sharing system that provides a reputation system based on Integrity rating and participation level. The drawback is that good behavior by a peer is rewarded but bad behavior is not punished.
Global Reputation

Eigen Trust [11] introduces the concept of global reputation where each peer collects the reputation value of a peer from all other peers in the network and then makes a decision as to whether this peer is to be trusted or not. So each time a peer $i$ has a transaction with peer $j$ say a file download, it may rate the transaction as positive or negative depending on the authenticity of the received file as $(tr(i, j)=1)$ or $(tr(i, j)=-1)$. The local trust value for a peer is then calculated as $s_{ij} = \sum tr_{ij} = (sat(i, j) - unsat(i, j))$ where $sat(i, j)$ is the number of satisfactory transactions and $unsat(i, j)$ is the number of unsatisfactory transactions. The global reputation of each peer $i$ is then given by local trust value assigned to peer $i$ by other peers, weighted by global reputations of assigning peers. However the drawback of Eigen trust is that the trust value is normalised to lie between 0 and 1 and thus it does not differentiate between peers with whom peer $i$ did not interact, or peers with whom peer $i$ had a poor experience. Further it does not take into account user dynamics nor does it consider the effect of credibility.

Framework for Trust Based Access Control

The authors in [9] propose a decentralized access control system for ad hoc collaborative environments where user permissions are determined and assigned based on a reputation management process which does behavior grading of a user and provides a reputation index that nodes use to determine trustworthiness of their peers. Each resource owner uses the combined characteristics of identity and reputation to make access decisions. The trust management system (TMS) track’s a user’s behavior based on past behavior. Each node gathers and processes feedback to calculate a usable reputation index $RI$ for its peers. The proposed node-based TMS provides the ability to enforce multiple access levels dynamically, based on behavior information. A node provides its peers customized views of its resources based on the individual peer’s trust profile and the peer’s trustworthiness.

Hu Tran et al in [5] present a Trust Based Access control Framework for P2P file sharing systems. The access values are evaluated via combinations of four types of scores: direct trust, indirect trust and direct contribution and indirect contribution. **Direct trust** represents the hosts belief on the clients capacities, honesty and reliability based on the hosts direct experiences. **Indirect trust** represents the hosts belief on the clients capacities, honesty and reliability based on recommendations from other peers. **Direct contribution** measures the contribution of the client to the host in term of information volume downloaded and uploaded between them. **Indirect contribution** measures the contribution of the client to the network in terms of information volume the client exchanged with other peers.

3.1.6 Hierarchical Access Control in Secure Group Communication

Many group communications require a security infrastructure that ensures multiple levels of access privilege for group members. Group applications may contain multiple related data streams and have members with various access privileges. For e.g in a video broadcast, users with normal T.V receiver can receive normal format, while others with HDTV receivers can receive both normal format and HDTV resolution. Similarly communications in hierarchically organizations like an educational institute or military group communications may also require different participants to have different access privileges. These types of applications necessitate the need for hierarchial (multi-level) access control.
Sun et al in [13] propose a multi-group key management scheme that manages keys for all members with different access privileges. They propose an integrated key graph that maintains the keying material for all members and permit user relocation on the key graph. They introduce the notations of Data Group (DG) and Service Group (SG). DG is defined as the set if users who receive the same data while SG is the set of users with the same access privilege. The actual key management is controlled by a centralized KDC.

Three steps are used to construct a key management graph. First a subtree is constructed for each service group with leaves as group members. Then a subtree is constructed for each data group with leaves as service groups which have access to this data group. Finally the subtrees of service groups and data groups are combined together to form an integrated key graph. The scheme supports dynamic privilege change when a participant switches from one service group to another. However their scheme is more suitable for group communications where the number of service groups is fixed and the data stream is scalable in one direction. The scheme is not flexible for dynamic service group formation and decomposition and is unable to efficiently handle a lot of service groups.

3.1.7 Dynamic Access control for multi-privileged group communications

Di Ma et al in [14] propose a dynamic access control scheme for group communications supporting multi-privileged service groups. Their scheme allows dynamic formulations of service groups and maintains forward/backward secrecy when users switch service groups. Each service group forms a subtree whose leaf nodes are the participants of the group and whose root is associated with an access key (AK) set. An AK set is a subset of the content encryption key (CEK) set, which consists of all unit encryption keys of a scalable stream. An AK set is possessed by a service group and represents an access privilege. Unlike the single privileged access control where the root of the group tree is a single CEK, here the root of each subtree is associated with an AK set. The KEK below the root node of the subtree is denoted as the service root key (SRK).

The scheme uses two fields revision and version to eliminate or reduce the rekey messages upon a user join, leave or switch operation. However the rekeying algorithm is controlled by a centralized KDC.

3.2 Secure Key establishment

Security of collaborative groups relies on an efficient and robust group key management. We had done an extensive survey of Group Key Management Protocols [29] and had presented it in the previous annual progress seminar report. Two of the relevant protocols for contributory group key management which could be used in the integrated framework are detailed in the Appendix A.

3.3 Summary

With the increase in the number of web collaborative applications, the need is for a hybrid access control model which combines trust based access control with dynamic access control for multi-privileged groups without dependency on a KDC to do the rekeying.
Chapter 4

Proposed Integrated Framework

In the decentralized and dynamic peer group scenario it is necessary to have an adaptive access control mechanism where peers have the right to collaboratively make access control decisions, modify these decisions and frame rules based on attributes as well as trust evolved in the group. We propose a hybrid access control framework which integrates policy based access control and trust based access control along with multi-level access control.

We present here a flexible architecture framework for secure group communication in decentralized peer groups, based on our proposed access control framework. The architecture is suitable for applications that involve sharing of high value resources, e-commerce applications, as well as general collaborations where security is not a major issue.

4.1 Proposed Architecture

We consider a peer based model where every peer $P_i$ has a unique user identity $UUID_i$ associated with it. A group is a named set of peers or a subset of a community that is governed under a set of rules that describe the conditions required to be part of the group, and is formed based on a particular interest criterion. The dynamic membership of the group implies that no peer will be highly available. The challenges are rapid admission of new members with appropriate access rights, dynamic elevation of access levels of existing members and dynamic revocation of access rights of leaving or defaulting members.

Peers should be individually capable of performing the tasks of authentication, admission control, access control and key management. However the extent to which each peer should be capable of performing these tasks need not be uniform. A peer in a maximal role would be expected to have the following functional components:

- **Attribute Manager**: Responsible for storing the attributes and certificates of the peers. The Attribute Manager is also responsible for updating the certificates of the peers with the new rating.

- **Authentication Manager**: Verifies the validity of the identity and invokes the voting algorithm.

- **Trust Engine**: Is responsible for calculating the trust values of each peer, based on responses from other peers. It thus has a calculation engine and a feedback engine associated with it.
• **Access Policy Manager**: Frames rules or policies based on attributes, behavior of peers and inputs from trust engine. It also checks the rules to allow members to join. It is called upon either by the Authentication Manager in case of a new member join, or the Updation Manager in case of an existing member level updation.

• **Updation Manager**: Is responsible for checking the rating levels of peers after they have collected feedback from other peers and gives the input to the Access policy manager. It is also responsible for granting or revoking access rights to peers based on the dynamic policy.

• **Key Management System**: Is invoked whenever a new peer joins or an existing peer’s access level is changed depending on the dynamic policies, and is responsible for computing the group key.

These components interact as shown in the figure 4.1.

### 4.2 Model Overview

Our initial membership admission protocol is similar to the protocol proposed by Kim et al. [1] and allows all peers to collaboratively decide whether a new user should be allowed entry into the group and if so decide the hierarchy level or role in which he can join the group. First time authentication is based on user identity(e.g an X.509
credential), or a self generated pseudo certificate and other attributes presented during initial admission request phase. This is handled by the Authentication Manager and Attribute Manager. If permitted to join he is given a signed membership certificate which serves as an authentication token. For all practical purposes a new peer will be allowed to join a group at the lowest level initially. The membership certificate contains a field which has the trust level or rating of the member.

Subsequent per session authentication could be based on an authentication token received in prior session and recognized by all peers. This would require inputs from the Access Policy Manager and the Trust engine.

Once a member of the group, the access control policy could be dynamically changed and the user could be switched from a low hierarchy level to a higher hierarchy level depending on his performance in the group as well as the rating given to him by other peers in the group. A rating of a peer can either be periodically updated by all existing peers in the group or a peer can request for rating certificates from other peers in the group with whom he has had transactions.

Rekeying algorithm would be invoked by the peer whose rating increases, so that he could switch to a higher hierarchy. If a member who was previously part of the group and has left, wishes to rejoin the group, then based on the rating token and credentials that he possesses, he could easily rejoin without having to be subjected to the voting process, or if he has a high rating i.e a high trust level, then he may be permitted to join as a trusted member even though the votes of remaining members are less than the threshold i.e 50 percent i.e the admission policy changes if the member is an old member.

The credentials provided by the user could have various levels of trust, that could be incremental.

**Phases in Proposed Model**

Our proposed model has the following phases:

- Group Initialization
- Admission Request
- Authentication
- Voting and Authorization
- Access control and Trust Building
- Key Management
- Per-Session Authentication

The different cases that we wish to support are as follows:

- New peer wishing to join an existing group.
  - A peer having a X.509 certificate signed by a trusted CA.
  - A peer without certification from CA but who provides a self signed pseudo certificate with proof of possession of private key.

- An existing member of the group who wants his access level updated.
• A peer who was previously part of the group and wishes to rejoin in a new session.
• Dynamic leaving of peers from the group.

Notations used
• $P_{\text{new}}$: New Peer
• $P_i$: Existing Peer
• $SK_i$: Private key of $i$
• $PK_i$: Public key of $i$
• $Cert_i$: Certificate of $i$ which contains the ID, Public key, Level/rating, Validity period.

In the next few sections we elaborate on the different phases of our proposed framework.

4.3 Protocol for New Member Join

The joining of a new member would have the phases as detailed below:

![Figure 4.2: New Member Join](image)

4.3.1 Group Initialization and Advertisement

In case of peer groups, it is essential that the initiator of the group should publish and advertise his group along with documents that specify the parameters of the group, such as group type, admission policies, group name, group members etc. The group advertisement should be periodically broadcast by the peers of the group.
4.3.2 Admission Request

A new peer wishing to join a group would first search for an advertisement of a group that matches his requirements. He would then submit a signed request to any peer which would include his credentials that he obtains from a CA say X.509 certificates or self-generated certificates, along with a request for the role that he wishes to join in.

For a peer to authenticate itself to a peer group, we state that just submitting his identity is not sufficient. Given that each peer has his own certificate which could be self signed or signed by a CA, a peer credential could be created by hashing the concatenation of unique user ID UUID and public key fields and then signing this hash with the private key of the user and using this digital signature as the identity of the peer. This identity could be used as the peer’s credential in the messages.

\[ P_{new} \rightarrow P_i : \{JoinREQ\}_{SK_{new}}, Cert_{new} \]

where

\[ Cert_{new} = UUID_{new}, PK_{new}, Rating_{new}, H[UUID||PK_{new}]_{SK_{new}} \]

For a new user his Rating_{new} field will contain a single value signed by himself.

4.3.3 Authentication

The peer receiving this signed request will obtain the identity of the requesting peer from the Certificate. If it is a signed certificate then the verification is easy. If however the Public Key pair is pseudo generated then the receiving peer will compute the hash of UUID and PK_i and tally this with received message. If the two hashes match then the user is authenticated and will be granted access permissions based on the access control policy. If the user’s credentials contain a rating field with a single entry he will be treated as a fresh user and the voting process will be invoked by broadcasting this join request to the other peers. The certificate of the new peer is co-signed by the broadcasting peer, so that further verification of ID is not required by other peers.

\[ P_i \rightarrow AllP_n : \{JoinREQ\}_{SK_{new}}, \{Cert_{new}\}_{SK_i}, VoteReq \]

4.3.4 Voting and Authorization

The peers all verify the authenticity of the new peer from his credentials. The peers return the results of voting to Access Policy Manager of the peer who had initiated it.

\[ P_n \rightarrow P_i : \{Vote, Level\}_{SK_n} \]

The Access Manager verifies the votes and if the votes are > 50% then the later then gives a signed membership token to the new peer and calls the rekeying algorithm. Since this is signed with the current group key which is unknown to the peer the later cannot modify it and forwards this directly to the Key Management System (KMS) for rekeying needed to join the group.

\[ P_i \rightarrow P_{new} : \{GC_{new}\}_{G_{key}} \]

The entry of the new peer is broadcast to all peers of the group. The membership token GC_{new} would contain details of the action granted to a joining peer, for e.g: join, invite, escort, attend or reject.
• **Join action** allows a peer to join the group as a full-fledged member with full access control rights.

• **Attend** allows a peer to join a group as a guest with restricted access e.g. listen to conversations.

• **Invite action** grants a peer the authority to dynamically allow other peers to join the group as guests. The invited peer gets attend action.

• **Escort** permits a trusted peer to escort or vouch for another member. However, this member who has been escorted into the group has limited access rights and is permitted to stay only as long as the member who has escorted him is part of the group.

• **Reject**: A peer who has been granted a reject action can remove another peer from the group by forcing a re-keying operation.

### 4.3.5 Key Management

The Key Management component of the existing peer who receives the signed Group Membership certificate performs the group rekeying at the necessary level. TGDH protocol can be used for computing the key.

### 4.4 Protocol for Existing Member Level Updation

Once a member of the group, a peer can periodically request the members of the group to rate him based on his performance and the trust level he has already acquired. Appropriate access privileges are granted to a peer based on a scoring system. A rating certificate can be used as a means of recommendation. This certificate would contain the original trust value of the peer along with the recommended rating value given by each peer. The recommendation would be signed by the recommending peer thus preventing a peer from being able to modify the recommendations/ratings given to him by other peers.

Each peer has a local Trust calculator which is able to compute the rating of every other peer. The rating certificate would also contain an expiry date to prevent the recommended peer from recycling good rating certificates beyond the specified period. When a certificate is about to expire, the peer can contact the recommending peer and get the validity of the certificate extended. Thus for each peer it has interacted with, the peer needs to keep a copy of the certificates it issued to that peer for validation purpose and for trust and contribution score updating. In addition a peer could maintain a revocation list of peers who are not to be trusted and periodically this information could be exchanged amongst peers. Following would be the fields in the rating certificate:

- Recommending peer’s identity
- Recommended peer’s identity
- Original trust value
- Contribution score
- Issuing date and time
The actual protocol is as follows:

4.4.1 Request for Updation

\[ P_i \rightarrow P_j : \{\text{UpdateREQ}\}_{SK_i}, \text{Cert}_i \]

where \( \text{Cert}_i \) contains rating certificates from other peers. \( P_j \) computes the global rating value after verifying the authenticity of individual ratings. The Updation Manager interacts with the trust Engine to calculate the rating of the peer. It then forwards this request with rating score to the Policy Manager.

4.4.2 Access Control

Access rights will be assigned based on a combination of the credentials, self rating and behavioral rating as well as rating given by other peers. Trust threshold, denoted as \( A_{ij} \) is a summation quantity. Any peer who has its overall trust value and overall contribution score equal to or greater than the corresponding thresholds, can get the corresponding hierarchial access right. The policy would be flexible so that it could be modified contributively above a certain level. Thus maybe peers at access level 3 could modify the policy rules for entry into that level. The following section gives the details of the algorithm used to calculate the rating.
Rating Algorithm

For calculating the trust/reputation rating we use the concept proposed by Eigen Trust \[11\] with some modifications. We state that the factors on which reputation depends are

- Peer Feedback
- Weighted Cost of transaction
- Credibility of peer who is giving the rating

Thus each peer has a local trust value, based on his initial credentials and his performance in the system, as well a global trust value which is based on the ratings provided to him by other peers.

Weighted Cost

We introduce the concept of weighted cost to overcome the drawback of the eBay \[18\] reputation system where buyers and sellers can easily default on large orders after gaining a good reputation value by successfully completing small orders. We put an upper bound on transaction as $\Theta$ and state that for all transactions whose cost or weight is $< \Theta$ the reputation is calculated normally but for every transaction with weight $> \Theta$, the reputation value is multiplied by a constant which is a multiple of this threshold value. This would ensure that smaller transactions generate lower trust value than larger transactions.

Thus local rating or reputation index $RI$ for a peer $i$ is calculated by peer $j$ as follows: $RI_i = RI_j \times \sum T_{ij} \times Trans\_cost$ for transactions whose cost or weight is $> \Theta$, where $T_{ij}$ is the opinion peer $j$ makes about peer $i$ per based on transactions and behavior.

The global rating of Peer $i$ is then a summation of the local ratings computed by all the peers it interacts with. A peer cannot modify its trust and contribution rating even though it stores the rating locally in the form of certificates.

4.4.3 Key management

The necessary rekeying is done by the Key Management system. In order to ensure multiple levels of access privilege for group members, there can be separate key trees for users with same separate access privileges i.e each hierarchy of users would have a separate group key for encrypting their data. This can be done by calling the key agreement protocol separately at each hierarchical level. Thus each user would have to maintain different sets of keys and when a user switches from one level to another, he could call the rekeying algorithm to ensure that backward secrecy is maintained. We propose a key management graph similar to the one in \[14\]. However the rekeying algorithm is not a one way function controlled by a KDC, but is a variant of the TGDH algorithm.

4.5 Per-Session Authentication of Previous Member

Authentication of a peer is based on his credentials, authentication token received in earlier sessions, and/or rating granted to him in earlier session. The role assigned to him is based on his attributes and the rating. It would be the task of the access policy manager to check whether the old rating certificates have expired or not or whether, the members who had given the rating earlier are revoked or not.
4.6 Dynamic Leaving of Members

When a peer leaves a group, it may not be identified immediately. Peers may send a query message to find out when a particular peer last communicated. If the peer is still a current member of the group, response would be immediate. However if a certain threshold of members reply stating that the last communication was not very recent then the group may assume that this particular peer is no more a current member of the group. Periodic rekeying could be another solution to tackle security threat from peers who leave without informing.

4.7 Summary

We have proposed a framework for decentralized secure peer groups. We need to test the performance of the key management algorithm with these additional authentication and access control overheads. As an initial measure we identified that JXTA [32] could be used as a base to deploy this framework. We therefore set up an initial test bed on JXTA to test our generic framework. In the next chapter we discuss the implementation issues for our test application which was a secure chat application.
Chapter 5

Implementation of Secure Multi-Chat

We have proposed a model for access control, authentication and authorization in collaborative environments, which incorporates incremental trust in a multi-level hierarchial system. Further the proposed model integrates admission and access control along with dynamic key management. We implemented a secure chat application based on JXTA as an initial test bed. Users initially register with a central server and are authenticated on the base of a static access control policy and voting. After authentication and voting, group rekeying is done and the entry of the new user into the group is broadcast to all the peers. Once a user is admitted to a group he is assigned a role based on his attribute and incremental trust developed, thus implementing a hierarchy based access control according to different roles which are mapped to access rules through our policies. Some of our sample rules are as follows: For joining as a student:

- Only students from particular department or from a particular IP address may join a group.
- Students belonging to a specific hostel may join. It may be desirable to allow the admittance of additional students in special cases.

Once rekeying is done members of one hierarchy are able to chat with each other in a secure manner by encrypting the message with their corresponding hierarchy key. Currently we have used the STR [31] protocol to achieve rekeying. However for a successful hierarchial implementation the TGDH [30] protocol would be a better choice. It was possible to use JXTA as an integrated framework for admission and policy based access control along with secure communication. Sample screenshots are given in Appendix B.

5.1 JXTA-A Peer to Peer Framework

We have used JXTA [32] as the base platform for application development which allows us to incorporate admission and access control along with secure key management for group communication. JXTA technology is a set of open protocols that allow any connected device on the network. The JXTA protocols standardize the manner in which peers:

- Discover each other
- Self-organize into peer groups
- Advertise and discover network services
- Communicate with each other
- Monitor each other

Thus one can build up applications on top of these protocols. The main components in a JXTA network are: Peers and peer groups, Messages, Pipes, Services and applications, Advertisements. A peer group forms a boundary that prevents non-authorized users from accessing to group. JXTA divides software into three layers:

- JXTA core - handles peer discovery and monitoring
- JXTA services - Generic services like files sharing and indexing
- JXTA application layer is reserved by application developed by JXTA community.

Peers are organized into peer groups and communicate by sending messages over JXTA pipes. Since pipes make use of underlying transport protocol so nothing can be assured about reliability of messages sent over JXTA pipes. However we can design our framework so as to ensure reliability by implementing our own scheme.

- JXTA uses a small number of protocols. Each is easy to implement and integrate into P2P services and applications. Thus service offerings from one vendor can be used transparently by the user community of another vendor’s system.

- JXTA is independent of programming languages, so that it can be implemented in C/C++, the Java programming language, Perl, or other languages. Heterogeneous devices with completely different software stacks can interoperate with the JXTA protocols.

- JXTA is independent of transport protocols. It can be implemented on top of TCP/IP, HTTP. This means that a system built on top of JXTA functions in the same fashion when the system is expanded to a new networking environment or to a new class of devices, as long as there is a correct transport protocol handler for the new networking protocol.
5.1.1 Policy language

We would be using XML to specify our policy language framework, because it provides us a customizable and extensible framework to represent messages. In fact JXTA’s group advertisements and other messages are in form of XML files. Data managed in XML file is easily searchable and can be validated easily. XML can also be used as specification language for describing peers, peer groups and services. A sample policy file in XML for granting or denying certain access permissions would be as follows:

```xml
<rules>
  <rule>
    <requestname>join</requestname> <permission>grant</permission> / >
    <condition>
      <parametername>Rating</parametername> <minbound>1</minbound> / >
      <parametername>vote</parametername> <minbound>40</minbound> / >
    </condition>
  </rule>
  <rule>
    <requestname>updation</requestname> <permission>grant</permission> / >
    <condition>
      <parametername>performance</parametername> <minbound>cpi</minbound> / >
      <parametername>vote</parametername> <minbound>60</minbound> / >
    </condition>
  </rule>
  <rule>
    <requestname>read</requestname> <permission>grant</permission> / >
  </rule>
  <rule>
    <requestname>write</requestname> <permission>grant</permission> / >
    <condition>
      <parametername>rating</parametername> <min>2</min> / >
    </condition>
  </rule>
  <rule>
    <requestname>modify</requestname> <permission>deny</permission> / >
    <condition>
      .................. < /condition >
    </rule>
  </rules>
```
Chapter 6

Conclusion

Currently there are some integrated solutions for secure group communication over wide area networks, but very few allow the development of trust and switching of users between hierarchial levels in an incremental fashion. We have modeled an integrated framework for decentralized groups, which will allow us to implement secure communication between different peers using admission control and trust based access control along with key management. We propose a dynamic access control policy which is integrated with trust based access control and permits peers to collaboratively frame, and modify policies at various levels. The policy also permits peers to update their access rights within a group. Idea of per-session authentication is also proposed which depends on the incremental trust factor and eliminates the need of the voting process. The access control policies can be dynamically changed by the members of the group based on this incremental trust. We have implemented a small part of our model.

6.1 Future Scope

We are currently in the process of implementing a test-bed system. Our future work includes refinement and implementation of the proposed scheme. We hope to use this test-bed to measure performance characteristics such as latency of joins and leaves, computational overhead of storage at individual peers, effect of authentication and multi-level access control on rekeying, and dynamic revocation of access rights.

Groups may be formed with peers having different functional roles, and the peer in a maximal role may not be the highest level peer for example. We would like to focus on the policies for composition of groups in the dynamic scenario. The quantum of functionality to be assigned to individual peers, what effect it would have on the working of the group, and how this could be done in a distributed manner, are some challenges. Measuring the inter-group trust level could also be a parameter.

We would also like to see how our system behaves with malicious peers.
Appendix A

STR protocol

STR \cite{31} is a contributory group key agreement protocol where each member contributes equally to the generation of the key and the group key is never broadcast over the network. STR is primarily a variation of the n-party Diffie-Hellman key agreement/exchange protocol. In this protocol every member maintains an unbalanced tree view for all the current members. For n members height of the tree is n-1, see Figure 6.1. The key tree is organized as follows: each node is associated with a key and a corresponding blinded key. The root is associated with the group and each leaf with a distinct member. The root key represents the group key shared by all members, and a leaf key represents the random contribution of each member.

Each leaf hold its secret key and it calculates $g^{k_i}$ and propagates towards its parent. Then a combined key from children’s $g^{k_i}$ and $g^{k_{i+1}}$ is calculated and at the end group key is calculated at root. In case of membership change (join/leave) the tree is re-built consequently and hence all the members update the group key which is the new key associated with root of tree.

Every member knows the structure of the tree and its initial position within the tree. Further each member knows its session random and the blinded session random numbers
of all other members. The protocol relies on the fact that every member can compute
group key if it knows all blinded keys in the key tree.

Thus for example member $M_1$ computes $k_2 = (br_2)^{r_1} \mod p = g^{r_1r_2} \mod p$,

$bk_2 = g^{k_2} \mod p$

$k_3 = (br_3)^{k_2} \mod p$

$bk_3 = g^{k_3} \mod p$ etc.

The group key is of the form: $g^{r_n}g^{r_{n-1}}...g^{r_{2r_1}}$

**TGDH protocol**

Another contributory protocol is the Tree Based Diffie Hellman Protocol TGDH [30]. Here
key trees are used in fully distributed contributory key agreement. The root is located at
level 0 and the lowest leaves are at level $h$.

The TGDH protocol uses binary trees; every node is either a leaf or a parent of two
nodes. The nodes are denoted as $< l, v >$, where $0 \leq v \leq 2l - 1$ since each level 1 hosts
at most $2l$ nodes. Each node $< l, v >$ is associated with the key $K < l, v >$ and the public
blinded key (bkey) $BK < l, v > = f(K < l, v >)$ where the function $f(\text{ })$ is modular
exponentiation in prime order groups, i.e. $f(K) = g^{k \mod p}$. Computing a key at $< l, v >$
requires the knowledge of the key of one of the two child nodes and the bkey of the other
child node. $K < 0, 0 >$ at the root node is the group secret shared by all members. For
example, in figure 6.2 M2 can compute $K < 2, 0 >, K < 1, 0 >$, and $K < 0, 0 >$ using
$BK < 3, 0 >, BK < 2, 1 >, BK < 1, 1 >$ and $K < 3, 1 >$.

The final group key $K < 0, 0 >$ is: $K < 0, 0 > = g^{(gr_3(g^{r_1r_2}))(g^{r_4(g^{r_5r_6})})}$

TGDH requires 2 rounds to form a group key, and number of exponentiations is of the
order of $\log n$. The storage cost of each member is of the order of $n$ and the number of
messages is of the order of $\log n$. Member join operation is comparatively efficient with
logarithmic number of exponentiations. Member leave operation in TGDH is best for leave
operation requiring cost of the order of $\log n$. STR requires 2 rounds, and exponentiations
of the order of $n$. The number of messages and storage requirement is of the order of $n$.
The overhead of TGDH protocol depends on the tree height i.e. balancedness of the key
tree for both join and leave operations. However the merge operation for STR is very
efficient. Over high delay Wide Area Networks STR is more advantageous.
Appendix B

Snapshots of our sample screens are shown below.

Figure 6.3: Group creation

Figure 6.4: Voting Request
Bibliography


