ISSN: 2277-9655
Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

A Flexible Communication of Group Key Agreement

Mrs. Waseema Masood[*1], Mrs.Tayyaba Rasheed[2]

[*1] Assistant Professor in Computer Science Engineering at Nawab Shah College Of Engineering & Technology (Affliated to JNTUH), Malekpet, Hyderabad-500024, A.P, India

[2] Assistant Professor in Computer Science Engineering at Nawab Shah College Of Engineering & Technology (Affliated to JNTUH), Malekpet, Hyderabad-500024, A.P, India

Abstract

Key transfer protocols rely on a mutually trusted Group Key Agreement (GKA) to select session keys and transport session keys to all communication entities secretly. Most often, GKA encrypts session keys under another secret key shared with each entity during registration .In this paper, we propose an authenticated key transfer protocol based on secret sharing scheme that GKA can broadcast group key information to all group members at once and only authorized group members can recover the group key; but unauthorized users cannot recover the group key. The confidentiality of this transformation is information theoretically secure. We also provide authentication for transporting this group key. Goals and security threats of our proposed group key transfer protocol will be analyzed in detail.

A Group Key Agreement (GKA) protocol is a mechanism to establish a cryptographic key for a group of participants, based on each one's contribution, over a public network and allows a set of players to establish a shared secret key which can be used to secure a subsequent communication. The key, thus derived, can be used to establish a secure channel between the participants. When the group composition changes (or otherwise), one can employ supplementary GKA protocols to derive a new key. Group key agreement is a fundamental building block for secure peer group communication systems.

Keywords: Group Key Agreement(GKA), Cryptographic key, Session Key.

Introduction

The Internet is increasingly being used to support collaborative applications such as voice- and video-conferencing, white-boards, distributed simulations, as well as games, replicated servers, and databases of all types. To be effective, these applications need supporting services, such as reliable and ordered message delivery as well as synchronization and fault-tolerance techniques. A reliable group communication system can provide an integrated platform containing such services, thus greatly simplifying the application development process and application complexity. Since most communication over the Internet involves the traversal of insecure open networks, basic security services, such as data privacy, integrity, and authentication—are necessary for collaborative applications. These services, in turn, are impossible without secure, robust, and efficient group key management. Clearly, key management is the most basic security service, since, without it, secure communication is basically impossible. In the context of collaborative groups, besides the design of the individual security building blocks, group security policy is a critical component.

Authentication and key establishment is the cornerstone of any secure communication. Without some form of authentication, all the other common security properties such as integrity or confidentiality do not make much sense. Authentication is generally based on long-term keys, which can be associated with identities. “Long-term key” is usually very broad and covers all forms of information, which can be linked to identities. As a result of the increased popularity of group-oriented applications and protocols, group communication occurs in many different settings from network layer multicasting to application layer telephonic and video-conferencing. Regardless of the underlying environment, security services are necessary to provide authenticity, integrity and communication privacy. Group communication is more complicated when the groups start, it might mutate (members leave and join) and there might not be a well-defined end. This complicates attendant

security services, in particular for key management. A group key is a cryptographic key that is shared between groups of users. Typically, group keys are distributed by sending them to individual users, either physically, or encrypted individually for each user using either that user's pre-distributed private key. The growth of group applications triggers the need for group-oriented security mechanisms over insecure network channels. The applications include IP telephony, collaborative workspaces, secure conferences, as well as dynamic coalitions common in law enforcement and disaster rescue scenarios. Standard security services required in such group settings, e.g. confidentiality of group-wide broadcasts, can be very efficiently achieved if all group members share a group-wide secret key.

A group key agreement protocol (GKA) allows *n* players to create such shared secret key. There are several widely known efficient constant-round group key agreement protocols, but their performance degrades if some of the participating players fail during the protocol execution. This is a serious concern in practice, for example for mobile nodes that communicate over a wireless media, but which can loose connectivity during protocol execution. Current constant-round GKA protocols are either efficient and non-robust or robust but not efficient: Assuming a reliable broadcast communication medium, the standard encryption-based group key agreement protocol can be robust against arbitrary number of node faults, but the size of the messages broadcast by every player is proportional to the number of players. In contrast, non-robust group key agreement can be achieved with each player broadcasting just constant-sized messages.

Existing System

Security is crucial for distributed and collaborative applications that operate in a dynamic network environment and communicate over insecure networks such as the Internet. Basic security services needed in such a group setting are largely the same as in point-to-point communication, data secrecy and integrity, and entity authentication. These services cannot be attained without secure, efficient, and robust group key management. Many critical applications (e.g., military and financial) require that all intra group communication remain confidential. Consequently, not only sufficiently strong encryption must be used to protect intra group messages, but the underlying group key management must also provide strong security guarantees

Contributory group key agreement protocols that compute a group key as a (usually, one-way) function of individual contributions from all members can provide both key independence and PFS properties. At the same time, contributory group key agreement presents a tough practical challenge: Its multi round nature must be reconciled with the possibility of crashes, partitions, and other events affecting group membership that can occur during the execution of the group key agreement.

Proposed System

We propose a robust novel 2-round group key agreement protocol which tolerates up to T node failures using O(*T*)-sized messages, for any T. To exemplify the usefulness of this flexible trade-off between message size and fault tolerance, we show that the new protocol implies a fully robust group key agreement with O(log *n*)-sized messages and expected round complexity close to 2, assuming random node faults. The proposed protocol is secure under the (standard) Decisional Square Diffie-Hellman assumption.

Group key agreement protocol is important for collaborative and group-oriented application. Recently, for a network that consists of devices with limited resource, group key management has become an issue for secure routing or multicast. The early design of contributory group key agreement (GKA) protocols focuses on the efficiency of initial GKA. Efficiency metrics include computation, computation and round complexities. Although each metric is important in practice, the round complexity can be more crucial, particularly in the distributed computing environment. Robust GKA is a serious concern in practice. Mobile nodes that communicate over a wireless medium can loose connectivity. Router failures, network partitioning, caused due to a misconfiguration or congestion and also due to malicious attacks, which also increases the failure probability.

ISSN: 2277-9655
Impact Factor: 1.852

Architecture

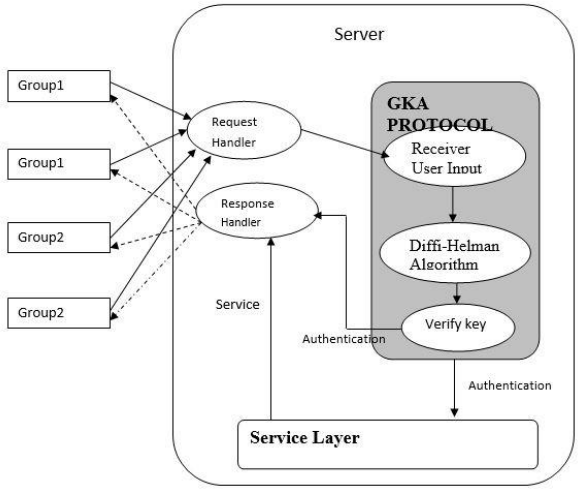

Figure 1 GKA Architecture

The Figure 1 describes the system architecture for a robust group key agreement component objects. The system communicates between clients and a server by exchanging randomly generated authentication key. The message exchanges between the clients done through a secure group key encryption and decryption mechanism. The architecture has the following modules

- **Client Node**
- **Server Node**
- **Request Handler**
- **Response Handler**
- **GKA Protocol**

1. Client Node

Client Nodes are group users who participate in communication. It initiates the communication request by sending the random key to the server with group id, client id and random key value. On completion of authentication process by the server it start communication with service layer or other client nodes.

2. Server Node

Server Nodes acts as a middleware for group key authentication for clients who want to do secure communication. It handles the client request and runs the GKA Authentication process.

The server node also provides service layer, which provides the requested information by the client nodes. In general, every distributed service environment server run some services for which the client request, to avail those service the user must need to pass the authentication mechanism being in place by the system.

3. Request Handler

Request Handler is an interface on server for handling client request. It handle the incoming request form the clients and provide input the GKA Protocol.

4. Response Handler

Response Handler is an interface for GKA Protocol and service layer. It handles the output messages from GKA Protocol to clients and also handle service layer message to clients. It broadcast the message received from GKA Protocol and Service Layer.

5. GKA Protocol

GKA Protocol is the core block of the project, which provides the security mechanism for the system. This protocol implements Diffie-Hellman Algorithm for providing group security authentication by exchanging random generated key. The protocols works on two-rounds keys exchange Group Key Agreement between client and server.

System Design

Data Flow Diagrams:

The figure 2 shows a data flow diagram for GKA protocol. The sequence pattern implies that the logic is executed in a single sequence. It begins with the server generating random key group value and sends Diffie-Hellman key value to user. The user receives the Diffie Hellman key value. The server then generates the user key value and compares both the keys, user key and group key. The program thus end by comparing both the keys.

ISSN: 2277-9655
Impact Factor: 1.852

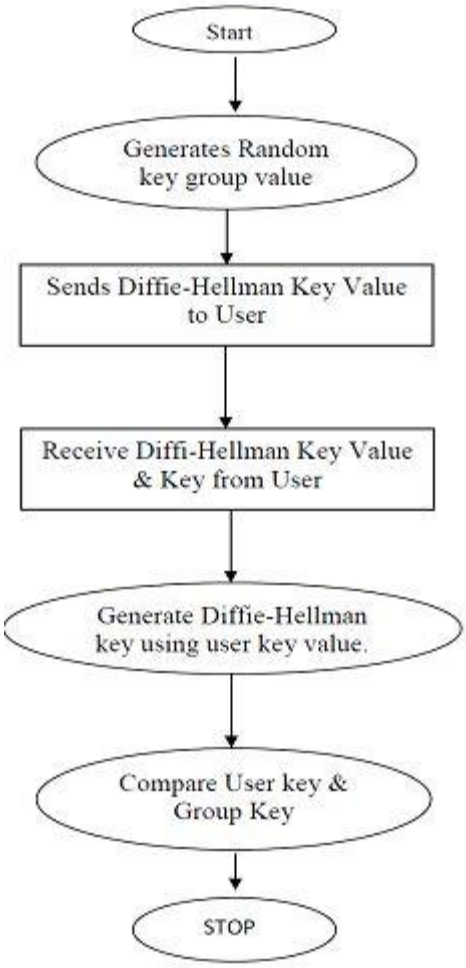

Figure 2. Data Flow Diagram for GKA Protocol

USE CASE DIAGRAM

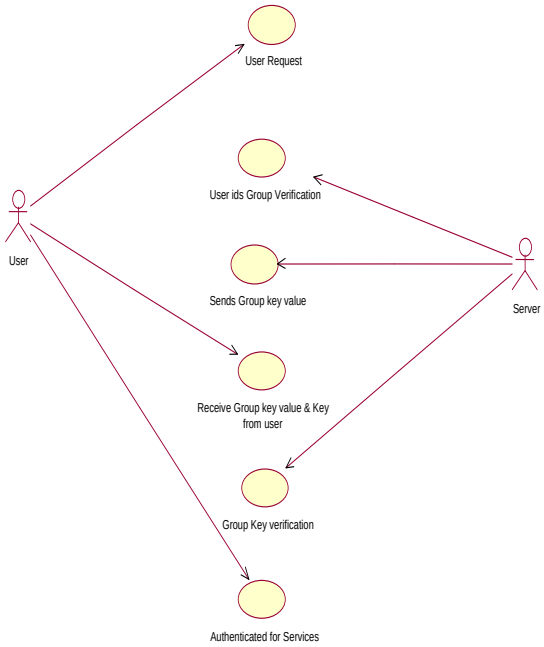

Figure 3.Use Diagram for GKA

The above Figure 3 depicts the Use case diagram for GKA Protocol System Operation.

1. The user will start by the user request .
2. The server will send the User ID Group Verification to the user .
3. The server will also sends the Group key value to the user.
4. The user will receive group key value from the server for the verification.
5. The server will check for the verification of the group key value and authenticate the service to the user.

The user is now an authenticated user and can use the service.

SEQUENCE DIAGRAM

ISSN: 2277-9655
Impact Factor: 1.852

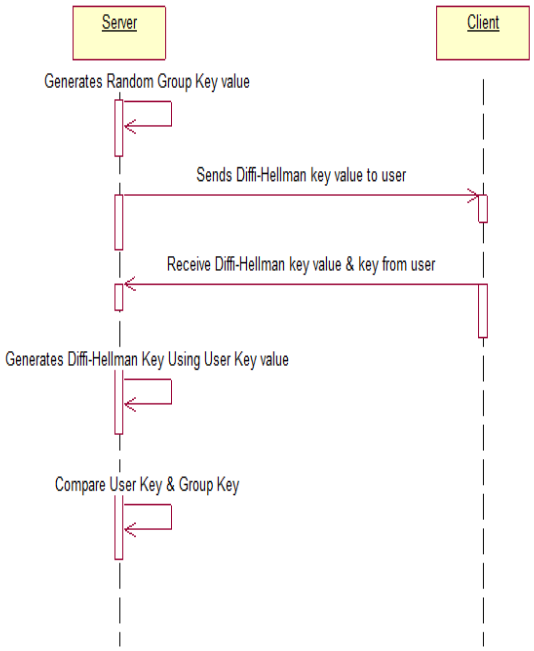

Figure 4. Sequence Diagram for GKA

The above sequence diagram depict the GKA protocol system operation

1.The server first generate Random Group key value and sends the Diffie-Hellman key value to the client.
2. The client receive the Diffie-Hellman key value from the server.
3. The server then generates Diffie-Hellman key using User key value and compare both the keys i.e.,user key and group key.

Collaboration Diagram

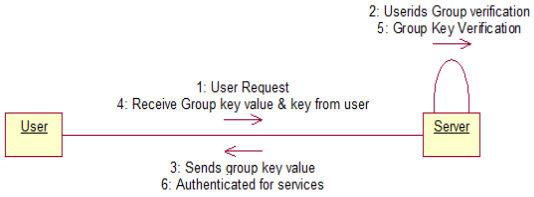

Figure 5. Collaboration Diagram for GKA.

The Figure4.8 shows the Collaboration Diagram for GKA Protocol System Operation. The user send user request to the server and receive group key value and key from user. The server generates userid group verification and sends group key value to the user and also authenticate service to the user.

Conclusion and Future Work

A group key agreement protocol (GKA) allows a set of players to establish a shared secret key which can be used to se- cure a subsequent communication. Several efficient constant- round GKAs have been proposed. However, their performance degrades if some players fail during protocol execution. This is a problem in practice, e.g. for mobile nodes communicating over wireless media, which can loose connectivity during the protocol execution. Current constant- round GKA protocols are either efficient and non-robust or robust but not efficient: Assuming a reliable broadcast communication medium, the standard encryption-based group key agreement protocol can be robust against arbitrary number of node faults, but the size of the messages broadcast by every player is proportional to the number of players. In contrast, non-robust group key agreement can be achieved with each player broadcasting just constant-sized messages. We implemented a novel 2-round group key agreement protocol which tolerates up to T node failures using O(T)-sized messages, for any T. To exemplify the usefulness of this flexible trade-off between message size and fault tolerance, we show that the new protocol implies a fully-robust group key agreement with O(log n)-sized messages and expected round complexity close to 2, assuming random node faults. The proposed protocol is secure under the (standard) Decisional Square Diffie-Hellman assumption.

The implemented group key agreement protocol, particularly well suited to any networks. It is efficient and also efficient in computational terms. It requires no special ordering of the users. Any user can be possibly chosen as the group for one session. The key derived is independent of keys in other sessions. Long-term secrets are used for authentication purposes only, thus providing weak forward secrecy. The protocol is proved secure in Standard model and with the Decisional Diffe-Hellman (DDH) assumption in any group.

References

[1] N. Asokan, V. Schoup, and M. Waidner, "Optimistic fair exchange of digital signatures," IEEE Journal on Selected Area in Communications,2000

[2] W. Diffie and M. E. Hellman, "New directions in cryptography:' IEEE Trans. Inform. Theory, vol. IT-22, pp. 644-654. Nov. 1976.

[3] Y. Amir, G. Ateniese, D. Hasse, Y. Kim, C. Nita-Rotaru, T. Schlossnagle, J. Schultz, J. Stanton, and G. Tsudik, "Secure group communication in asynchronous networks

ISSN: 2277-9655
Impact Factor: 1.852

with failures: integration and experiments," in Pruceedings vfthe 20th IEEE International Conjerence on Distributed Computing Systems, (Taipei, Taiwan), pp. 330-343, April 2000.

[4] A. Fiat and M. Naor, "Broadcast encryption," Advances in Cryptology - CRYPTO'93, August 1993

[5] M. Burmester and Y. Desmedt, "A secure and efficient conference key distribution system," Advances in Cryptology - EUROCRYPT94, May 1994.

[6] M. Just and S. Vaudenay, "Authenticated multiparty key agreement,"Advances in Cryptolugy - EUROCRYPT'96, May 1996

[7] M. Steiner, G. Tsudik, and M. Waidner, "Key agreement in dynamic peer groups," IEEE Transactions on Parallel and Distributed Systems, August 2000.

[8] G. Ateniese, M. Steiner, and G. Tsudik, "New multi-party authentication services and key agreement protocols," IEEE Journal of Selected Areas in Communication, vol. 18, March 2000.

[9] R. Poovendran, S. Corson. and J. Baras, "A shared key generation procedure using fractional keys," IEEE Milcom 98, October 1998