International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol.1, Issue1, pp-094-098 ISSN: 2249-6645

Implementation Of Object Oriented Approach To Authentication Of Group Key Transfer Using Secret Sharing

D. JEEVARATNAM
2/2 M.TECH CSE, DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING,
ST.THERESSA INSTITUTE OF ENGINEERING TECHNOLOGY,
GARIVIDI, ANDHRA PRADESH, INDIA

K.VISALA
ASSISTANT PROFESSOR,
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING,
ST.THERESSA INSTITUTE OF ENGINEERING TECHNOLOGY,
GARIVIDI, ANDHRA PRADESH, INDIA

UPPE NANJI
ASSOC. PROFESSOR & HOD
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING,
ST.THERESSA INSTITUTE OF ENGINEERING TECHNOLOGY,
GARIVIDI, ANDHRA PRADESH, INDIA

***Abstract*—In modern electronic distribution networks, message authentication is an important objective of information security. This objective is met by providing the receiver of a message an assurance of the sender's identity. As physical protection such as sealed envelopes is not possible for messages expressed as binary sequences, digital tools have been developed using cryptography. A major limitation of all cryptographic methods for message authentication lies in their use of algorithms with fixed symmetric or public keys. In this paper, we propose an authenticated secret key transfer scheme that KDC (Key Distribution Center) can broadcast group key information to all group members at once and only authorized group members can recover the group key. The confidentiality of this transformation is information theoretically secure.**

***Index Terms*— ciphers, data integrity, digital signature, encryption, key transport, message authentication, public-key cryptography, prepositioned secret sharing.**

I. INTRODUCTION

With the rapid development of Internet technology and the popularization of multicast, group-oriented applications, such as video conference, network games, and video on demand, etc., are playing important roles. How to protect the communication security of these applications are also becoming more and more significant. Generally speaking, a secure group communication system should not only provide data confidentiality, user authentication, and information integrity, but also own perfect scalability. It is shown that a secure, efficient, and robust group key management approach is essential to a secure group communication system. So, the secure storage of the private keys of a cryptosystem is an important problem. The possession of a highly sensitive key by an individual may not be desirable as the key can easily be lost or as the individual may not be fully trusted. Giving copies of the key to more than one individual increases the risk of compromise. A solution to this problem is to give shares of the key to several individuals, forcing them to cooperate to find the secret key. This not only reduces the risk of losing the key but also makes compromising the key more difficult.

In threshold cryptography, secret sharing deals with this problem, namely, sharing a highly sensitive secret among a group of n users so that only when a sufficient number t of them come together can the secret be reconstructed. Well-known secret sharing schemes (SSS) in the literature include Shamir [1] based on polynomial interpolation, Blakley [2] based on hyper plane geometry, and Asmuth-Bloom [3] based on the Chinese Remainder Theorem.

A shortcoming of secret sharing schemes is the need to reveal the secret shares during the reconstruction phase. The system would be more secure if the subject function can be computed without revealing the secret shares or reconstructing the secret. This is known as the function sharing problem. A function sharing scheme requires distributing the function's computation according to the underlying SSS such that each part of the computation can be carried out by a different user and then the partial results can be combined to yield the function's value without disclosing the individual secrets. Several protocols for function sharing have been proposed in the

literature [4,5,6,7]. Nearly all the existing solutions for function sharing uses Shamir secret sharing as the underlying SSS.

II. RELATED WORK

Several good explorations have been done for dealing with the group key distribution in a large group with frequent membership changes. There are two types of key establishment protocols: key transfer protocols and key agreement protocols. Key transfer protocols rely on a mutually trusted key generation center (KGC) to select session keys and then transport session keys to all communication entities secretly. Most often, KGC encrypts session keys under another secret key shared with each entity during registration.

In key agreement protocols, all communication entities are involved to determine session keys. The common key agreement protocol used in most distributed group key management protocols is Diffie-Hellman (DH) key agreement protocol. Some of the examples are: Bresson et al. [8] constructed a generic authenticated group DH Key exchange and the algorithm is provably secure. Katz and Yung [9] proposed the first constant-round and fully scalable group DH protocol which is provably secure in the standard model. The main feature of the group DH key exchange is to establish a secret group key among all group members without relying on a mutually trusted KGC.

A. Secret Sharing Schemes

The problem of secret sharing and the first solutions were introduced in 1979 independently by Shamir [1] and Blakley [2]. A (t; n)-secret sharing scheme is used to distribute a secret d among n people such that any coalition of size t or more can construct d but smaller coalitions cannot. Shamir secret sharing is based on polynomial interpolation over a finite field. It uses the fact that we can find a polynomial of degree t-1 given t data points. Blakley secret sharing scheme has a different approach based on hyperplane geometry: To implement a (t; n) threshold scheme, each of the n users is given a hyperplane equation in a t dimensional space over a finite field such that each hyperplane passes through a certain point. The intersection point of the hyperplanes is the secret. When t users come together, they can solve the system of equations to find the secret.

B. Function Sharing Schemes

Function sharing is the concept of distribution of the computation of a function such that when a sufficient number of users come together they can compute the value of the function without revealing their secret shares but less than the threshold number of users cannot. This problem is related to secret sharing as the secret values needed for partial computations are distributed using secret sharing. Several solutions for sharing the RSA, ElGamal, and Paillier private key operations have been proposed in the literature [33,4,5,6,7]. Almost all of these schemes have been based on the Shamir SSS. The additive nature of the Lagrange's interpolation formula used in the combining phase of Shamir's scheme makes it an attractive choice for function sharing, but it also provides several challenges. One of the most significant challenges is the computation of inverses for the division operations in Lagrange's formula

III. SYSTEM ARCHITECTURE

In this section, we first describe the model of our proposed secret transfer protocol. Then we present the security goals of our group transfer protocol.

Each individuals of the group gets the key from the distribution center which will be exchanged between other members of the group.

- Key generation Each member of a group sends a request to KDC with a nonce.
- KDC collects a composite number factorizable in to prime numbers of the form (4n+3). These numbers are obtained from the expression of Vandermonde's determinant.
- Random prime number is selected and being sent to each member of the group
- This number is represented as cyclic code with parity check and checksum.
- A cyclic shift is made on each representation.

In Fig.1, KDC is the key distribution center and has three members A1, A2 and A3. Group member A1 sends request IDA/N2 to KDC, KDC grants key to A1 by a reply IDA/N1. Similarly all other group members getting keys from KDC and communicate with KDC and with other group members. Life time of the key depends upon the number of transactions or sessions of the transaction.

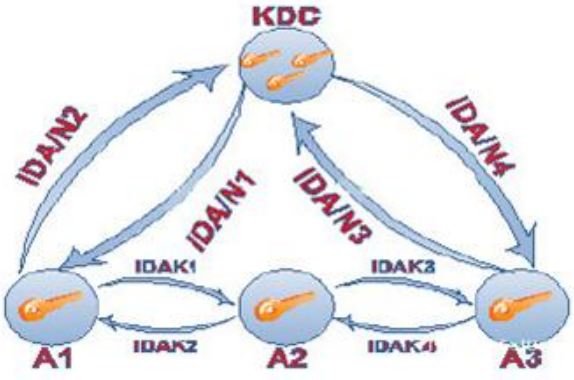

Fig 1 Key distribution

A. Goals

The main security goals of our group key transfer protocols are key authentication, key freshness and key authentication.

Each group key has never been used earlier in any step of KDC it ensures key freshness and cannot cause any further damage of group communication. Authorized group member only can recover the group key, it ensures that key confidentiality. KDC is giving assurance that the authorized group key is distributed to the group members, not by an intruder.

From our security analysis, we prove that none of the inside and outside attacks can successfully attack the authorized group members since attackers can neither obtain the group key nor share a group key with authorized group members.

B. Security Analysis

In this section, we prove that our proposed protocol achieves the security goals and is against inside as well as outside attacks. The two types of adversaries are outsiders and insiders. The outsider can try to recover the group key by impersonating as a group member by giving request to the KDC. In security analysis we will show that the outside attacker gains nothing, because the attacker cannot recover the group key, because they could not gain the individual factors of the composite number used by the KDC and the prime number difference are alone known from the Vandermonde's determinant evaluation which are the public information available to the outsiders. The individual keys are generated under cyclic permutation and cyclic code representation, getting information may not help decoding permutation and cyclic code radix. Hence the inside attacker

IV. OUR CONTRIBUTION

Although a contributory group key agreement is a promising solution to achieve access control in collaborative and dynamic group applications, the existing schemes have not achieved the performance lower bound in terms of time, communication, and computation costs. In tree-based contributory group key agreement schemes, keys are organized in a logical tree structure, referred to as the key tree. In a key tree, the root node represents the group key, leaf nodes represent the members' private keys, and each intermediate node corresponds to a subgroup key shared by all the members (leaf nodes) under this node. The key of each non-leaf node is generated by performing the two-party DH between the two subgroups represented by its two children where each child represents the subgroup including all the members (leaf nodes) under this node.

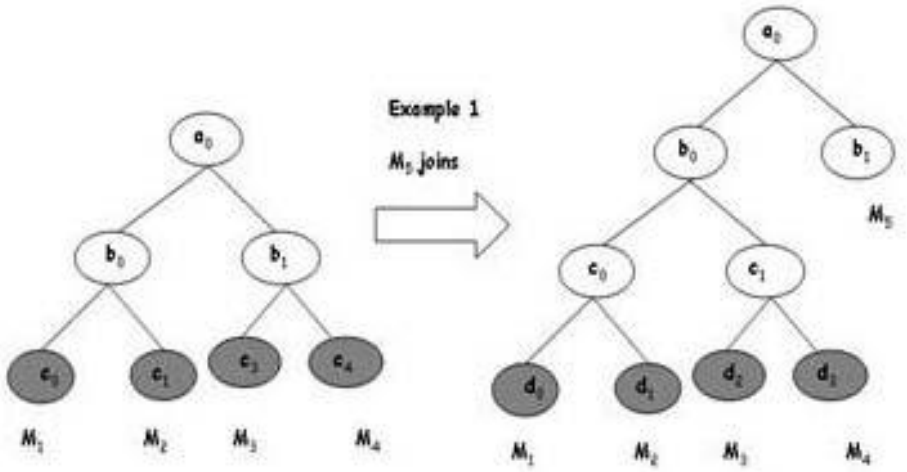

Fig.2 An example1 of key tree update upon single-user join event

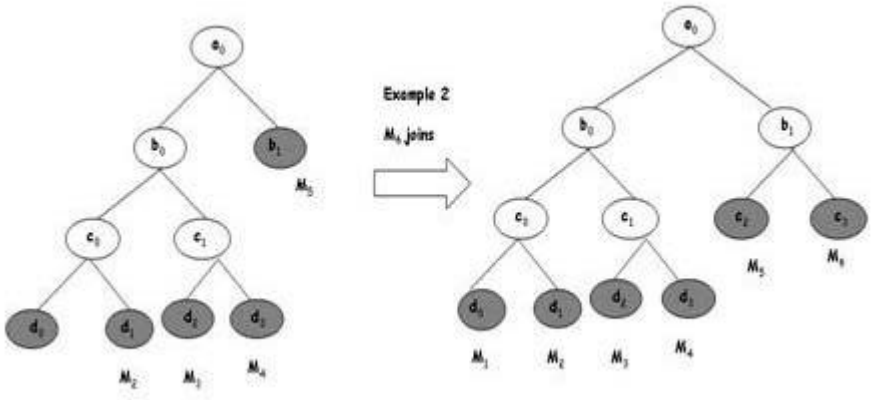

Fig.3 An example2 of key tree update upon single-user join event

A. Rekeying on Single-User Join

When a new user M wants to join the group G, the PACK initiates the single-user join protocol by broadcasting a request message that contains its member ID, a join request, its own blinded key, some necessary authentication information, and its signature for this request message. After receiving this user join request message the current group members will check and a new group key will be generated in order to incorporate a secret share from M. The rekeying upon single-user joins needs to perform two rounds of MDS code. Fig.2,3 shows two examples of a key tree update upon single-user join events.

In the first example tree consists of four members. After the new member M5 joins the group, a new node is created to act as the new root, and the node (b1) becomes the new join tree that represents M5. In the second example, when M6 joins the group, at the first round, the MDS is first performed between M5 and M6 to generate a new join tree, at the second round, the MDS is performed between the new join tree and the main tree to generate a new group key.

B. Rekeying on Single-User Leave

When a current group member Y wants to leave the group, it broadcasts a leave request message to initiate the single user leave protocol, which contains its ID, a leave request, and a signature for this message. In order to reduce the rekeying cost upon a single-user leave event, PACK creates a phantom node that allows an existing member to simultaneously occupy more than one leaf node in the key tree. Fig 4 depicts the model of user leave and in this example, user M6 leaves the group where node (b0) is the root of the main tree and node (b1) is the root of the join tree. Since the size of the join tree is 2, the node representing M6 will be directly removed from the key tree, M5 changes its secret share, and a new group key will be generated by applying the MDS between M5 and the subgroup in the main tree.

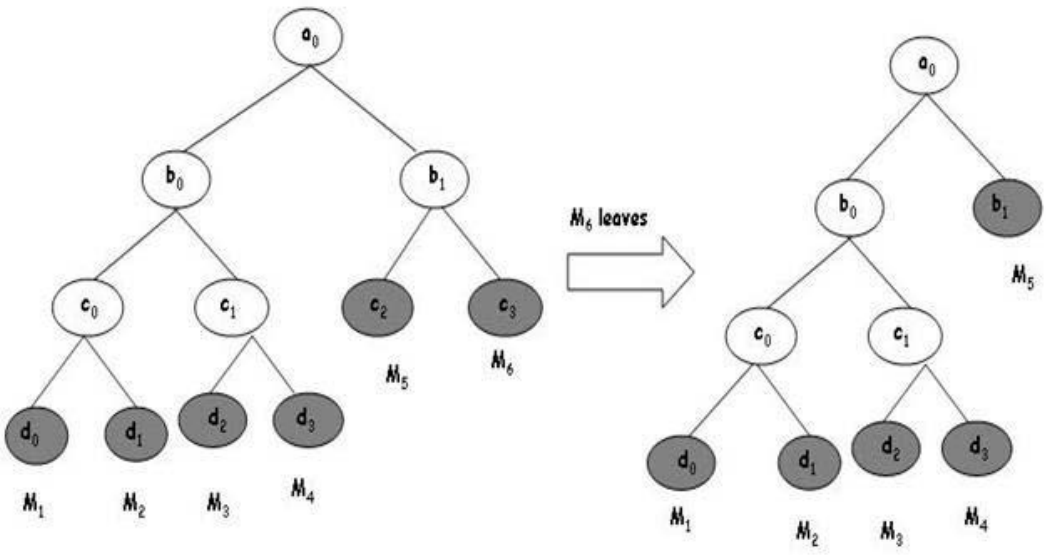

Fig.4 An example of key tree update upon single-user leave event

A. Rekeying on Multi-user Join and Leave protocol

PACK also has group merge and group partition protocols to handle simultaneously the join and leave of multiple users. Although multiple user events can be implemented by applying a sequence of single-user join or leave protocols, such sequential implementations are usually not cost-efficient. The group merges protocol, combines two or more groups into a single group, and returns a PF key tree. Group partition protocol, removes multiple group members simultaneously from the current group and construct a new PF key tree for the rest of the group members. In the group merge protocols, after removing all phantom nodes from those key trees corresponding to different subgroups, each key tree is split into several full key trees. The final result is obtained by uniting these full key trees into a PF tree using unite procedure. Similar to the group partition protocol, after removing all phantom nodes and leaving nodes, the original key tree is split into several full key trees, and the unite procedure is then applied on these full key trees to create a PF key tree. Since the height of the returned tree is log n, where n is the group size after merging/partitioning, the time cost of group merge/partition is bounded by O(log n).

V. RESULTS

The experiments are carried out on an Intel Core 2 Duo 2.80-GHz machine with a 2-Gbyte memory running windows XP. The implementation results of computations and communications are presented, from these results; we can see that upon a single-user join event, proposed method has the lowest cost among all the schemes. Compared with other methods, proposed method has more than 10 percent reduction in computation cost and a more than 65 percent reduction in communication cost and time cost. Compared with GC, the reduction is even more, about 50 percent in computation cost and about 80 percent in time and communication costs. Upon a single-user leave event, compared with other methods, proposed method has about a 25 percent reduction in computation cost, about a 15 percent reduction in time cost, and a similar communication cost.
Although other method has slightly higher computation and communication costs than other methods upon a single user leave event, when averaged over both join and leave events, the reduction is still significant, with a 20 percent reduction in computation cost, 35 percent reduction in communication cost, and 40 percent reduction in time cost. Fig 5 and 6, shows the key distribution time and key recovery time of both the scheme under various multicast group sizes. It is clear that using one-way hash functions adds none-trivial computation complexity.

VI. CONCLUSIONS AND FUTURE WORK

For any multicast group communication, group key agreement was found to be challenging because of its dynamic nature. Group key management scheme are of either distributed, centralized or hybrid architecture. Although many solutions have been proposed to handle group key changes, In this paper, We have proposed an efficient group key transfer protocol based on secret sharing. Every user needs to register at a trusted KGC initially and preshare a secret with KGC. KGC broadcasts group key information to all group members at once. The confidentiality of our group key distribution is information theoretically secure. This scheme is thus practical for many applications in various broadcast capable networks such as Internet and wired and wireless networks.

Future Work

As mentioned above related work still there is a necessity to provide the security in the network, so there is a future scope to enhancements.

REFERENCES

[1] Shamir. How to share a secret? *Comm. ACM*, 22(11):612–613, 1979.
[2] C. Asmuth and J. Bloom. A modular approach to key safeguarding. *IEEE Trans. Information Theory*, 29(2):208–210, 1983.
[3] G. Blakley. Safeguarding cryptographic keys. In *Proc. of AFIPS National Computer Conference*, 1979.
[4] Y. Desmedt. Some recent research aspects of threshold cryptography. In *Proc. of ISW '97, 1st International Information Security Workshop*, volume 1196 of *LNCS*, pages 158–173. Springer-Verlag, 1997.
[5] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *Proc. of CRYPTO'89*, volume 435 of *LNCS*, pages 307–315. Springer-Verlag, 1990.
[6] Y. Desmedt and Y. Frankel. Shared generation of authenticators and signatures. In *Proc. of CRYPTO'91*, volume 576 of *LNCS*, pages 457–469. Springer-Verlag, 1992.
[7] Y. Desmedt and Y. Frankel. Homomorphic zero-knowledge threshold schemes over any finite abelian group. *SIAM Journal on Discrete Mathematics*, 7(4):667–679, 1994.
[8] P. A. Fouque, G. Poupard, and J. Stern. Sharing decryption in the context of voting or lotteries. In *Proc. of FC 2000, 4th International Conference on Financial Cryptography*, volume 1962 of *LNCS*, pages 90–104. Springer-Verlag, 2001.
[9] E. Bresson, O. Chevassut, and D. Pointcheval, "Provably-Secure Authenticated Group Diffie-Hellman Key Exchange," ACM Trans. Information and System Security, vol. 10, no. 3, pp. 255-264, Aug. 2007.
[10] J. Katz and M. Yung, "Scalable Protocols for Authenticated Group Key Exchange," J. Cryptology, vol. 20, pp. 85-113, 2007.